PhD Thesis

Unified Processing Framework of High-Dimensional and Overly Imbalanced Chemical Datasets for Virtual Screening

Author: Amir Ali Rafati Afshar

Supervisors: Dr. Emili Balaguer Ballester

Professor Mark Hadfield

A thesis submitted in partial fulfilment of the requirements of Bournemouth University for the degree of Doctor of Philosophy

September 2016

Copyright Statement

Abstract

Virtual screening in drug discovery involves processing large datasets containing unknown molecules in order to find the ones that are likely to have the desired effects on a biological target, typically a protein receptor or an enzyme. Molecules are thereby classified into active or non-active in relation to the target. Misclassification of molecules in cases such as drug discovery and medical diagnosis is costly, both in time and finances. In the process of discovering a drug, it is mainly the inactive molecules classified as active towards the biological target i.e. false positives that cause a delay in the progress and high late-stage attrition. However, despite the pool of techniques available, the selection of the suitable approach in each situation is still a major challenge.

This PhD thesis is designed to develop a pioneering framework which enables the analysis of the virtual screening of chemical compounds datasets in a wide range of settings in a unified fashion. The proposed method provides a better understanding of the dynamics of innovatively combining data processing and classification methods in order to screen massive, potentially high dimensional and overly imbalanced datasets more efficiently.

Table of Contents

Abstract II

List of Figures V

List of Tables XXV

Acknowledgements XXVII

List of Abbreviations XXVIII

1. Introduction 1

1.1. Background 1

1.2. Project description and goals 3

1.3. Methodology and organisation of thesis 4

1.4. Publication 5

2. Representation and Visualization of Chemical Structures 6

2.1. Visualizing of Chemical Structures 6

2.2. Searching for Compounds in Databases 11

2.3. High-Throughput Screening 23

2.4. Virtual Screening 24

2.5. Handling the Mining of Large Datasets 26

2.6. Summary of challenges in this chapter 31

3. Datasets Description and Pre-Processing Strategies 32

3.1. Background 34

3.2. Data Preparation 36

3.3. Summary of Data Pre-processing 46

4. Dataset Processing 47

4.1. Data Imbalance 47

4.2. Tackling Imbalanced Data Problem 52
4.3. Evaluating Imbalanced Learning Outcomes 59
4.4. Classification 60
4.5. Principal Component Analysis 70
4.6. Specific Methodology for Cheminformatics Data Screening 72
4.7. Summary of Data Mining Methods 76
5. Analysis of the Datasets 77
5.1. The Benchmark Dataset 78
5.2. The Slightly Imbalanced Dataset 98
5.3. The Heavily Imbalanced Dataset – AID362 139
5.4. The Heavily Imbalanced Dataset – AID456 167
6. General Discussion and Concluding Remarks 195
7. Bibliography 209
8. Appendix 235

List of Figures

Figure 1: Compound attrition and cost increase of drug discovery process by time (Bleicher et al. 2003, p.371) 3

Figure 2: A graph with nodes (a, b and c) and edges (lines that connect the nodes ab, ac and bc) 7

Figure 3: A Hydrogen-depleted molecular graph of Caffeine (Brown, 2009) 7

Figure 4: Connection table example with an example molecule 8

Figure 5: Example of a line formula for the molecule shown in Figure 4. 9

Figure 6: SMILES, IUAPC and InCHi representations for Caffeine (Source: Brown 2009) 10

Figure 7: Example of Structure-key fingerprint (Brown 2009) 17

Figure 8: Pseudocode of a typical Hash-key fingerprint (Brown 2005) 18

Figure 9: The structuring on a Hash-Key fingerprint (Brown 2009) 19

Figure 10: Example of two fingerprints and the similarity and distance coefficient calculated. 22

Figure 11: Iterative process during HTS between various research groups (Stephan & Gilbertson 2009) 23

Figure 12: Showing the key factors towards a successful HTS process (Stephan & Gilbertson 2009) 24

Figure 13: A schematic illustration of a typical virtual screening flowchart (Leach & Gillet 2007) 26

Figure 14: Typical Grid protocol computing architecture (Foster et al. 2008) 28

Figure 15: Typical Cloud computing architecture (Foster et al. 2008) 30

Figure 16: Schematic overview of chapter 3 39

Figure 17: Illustrating the generation of fingerprints 45

Figure 18: Illustrating how the introduction of noise can affect the learning classifier's ability to learn decision boundaries. (Source Weiss 2004) 50

Figure 19: Generating synthetic samples by SMOTE.....54

Figure 20: An example of how to calculate non-repeating combinations for a group of 7 fingerprints 74

Figure 21: Classification results from classifying the *Bursi* dataset by J48. 79

Figure 22: Classification results from classifying the *Bursi* dataset by Naïve Bayes. 80

Figure 23: Classification results from classifying the *Bursi* dataset by Random Forest. 80

Figure 24: Classification results from classifying the *Bursi* dataset by SMO. 81

Figure 25: Classification results from classifying the *Bursi* dataset by Majority Voting. 81

Figure 26: Results from adding numerical fingerprints to binary fingerprints for J48 82

Figure 27: Results from adding numerical fingerprints to binary fingerprints for Naïve Bayes 83

Figure 28: Results from adding numerical fingerprints to binary fingerprints for Random Forest.....83

Figure 29: Results from adding numerical fingerprints to binary fingerprints for SMO.....83

Figure 30: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 83

Figure 31: Classifier performance for EState – Original 84

Figure 32: Classifier performance for MACCS – Original.....84

Figure 33: Classifier performance for Pharmacophore – Original.....85

Figure 34: Classifier performance for PubChem – Original.....85

Figure 35: Classifier performance for Substructure – Original 85

Figure 36: Results from adding numerical fingerprints to binary fingerprints for EState 86

Figure 37: Results from adding numerical fingerprints to binary fingerprints for MACCS 86

Figure 38: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 86

Figure 39: Results from adding numerical fingerprints to binary fingerprints for PubChem 86

Figure 40: Results from adding numerical fingerprints to binary fingerprints for Substructure 86

Figure 41: Classification results from classifying the Bursi dataset by J48 – PCA 87

Figure 42: Classification results from classifying the Bursi dataset by Naïve Bayes – PCA 87

Figure 43: Classification results from classifying the Bursi dataset by Random Forest – PCA 88

Figure 44: Classification results from classifying the Bursi dataset by SMO – PCA 88

Figure 45: Classification results from classifying the Bursi dataset by Majority Voting – PCA 88

Figure 46: Results from adding numerical fingerprints to binary fingerprints for J48 89

Figure 47: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 90

Figure 48: Results from adding numerical fingerprints to binary fingerprints for Random Forest 90

Figure 49: Results from adding numerical fingerprints to binary fingerprints for SMO 90

Figure 50: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 90

Figure 51: Classifier performance for EState – PCA 91

Figure 52: Classifier performance for MACCS – PCA 91

Figure 53: Classifier performance for Pharmacophore – PCA 92

Figure 54: Classifier performance for PubChem – PCA Applied.....92

Figure 55: Classifier performance for Substructure – PCA Applied 92

Figure 56: Results from adding numerical fingerprints to binary fingerprints for EState – PCA 93

Figure 57: Results from adding numerical fingerprints to binary fingerprints for MACCS – PCA 93

Figure 58: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore – PCA 93

Figure 59: Results from adding numerical fingerprints to binary fingerprints for PubChem – PCA.....93

Figure 60: Results from adding numerical fingerprints to binary fingerprints for Substructure – PCA 94

Figure 61: Sensitivity versus False Positive rate for the methods used on the Mutagenicity dataset. 94

Figure 62: Sensitivity versus False Positive rate per classifier for the Mutagenicity dataset. 95

Figure 63: Classification results from classifying the Fontaine dataset by J48 99

Figure 64: Classification results from classifying the Fontaine dataset by NaïveBayes 100

Figure 65: Classification results from classifying the Fontaine dataset by Random Forest 100

Figure 66: Classification results from classifying the Fontaine dataset by SMO 100

Figure 67: Classification results from classifying the Fontaine dataset by Majority Voting 101

Figure 68: Results from adding numerical fingerprints to binary fingerprints for J48 101

Figure 69: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 102

Figure 70: Results from adding numerical fingerprints to binary fingerprints for Random Forest 102

Figure 71: Results from adding numerical fingerprints to binary fingerprints for SMO 102

Figure 72: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 102

Figure 73: Classifier performance for by EState - Original 103

Figure 74: Classifier performance for MAACS - Original 103

Figure 75: Classifier performance for Pharmacophore - Original 103

Figure 76: Classifier performance for PubChem - Original 104

Figure 77: Classifier performance for Substructure - Original 104

Figure 78: Results from adding numerical fingerprints to binary fingerprints for EState 104

Figure 79: Results from adding numerical fingerprints to binary fingerprints for MACCS 105

Figure 80: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 105

Figure 81: Results from adding numerical fingerprints to binary fingerprints for PubChem 105

Figure 82: Results from adding numerical fingerprints to binary fingerprints for *Substructure* 105

Figure 83: Classification results from classifying the Fontaine dataset by J48 106

Figure 84: Classification results from classifying the Fontaine dataset by NaïveBayes 106

Figure 85: Classification results from classifying the Fontaine dataset by Random Forest 106

Figure 86: Classification results from classifying the Fontaine dataset by SMO 106

Figure 87: Classification results from classifying the Fontaine dataset by Majority Voting 107

Figure 88: Results from adding numerical fingerprints to binary fingerprints for J48 107

Figure 89: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 108

Figure 90: Results from adding numerical fingerprints to binary fingerprints for Random Forest 108

Figure 91: Results from adding numerical fingerprints to binary fingerprints for SMO 108

Figure 92: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 108

Figure 93: Classifier performance for EState – Original SMOTEd All 109

Figure 94: Classifier performance for MACCS – Original SMOTEd All 109

Figure 95: Classifier performance for Pharmacophore – Original SMOTEd All 109

Figure 96: Classifier performance for *PubChem* – Original SMOTEd All 110

Figure 97: Classifier performance for *Substructure* – Original SMOTEd All 110

Figure 98: Results from adding numerical fingerprints to binary fingerprints for EState 110

Figure 99: Results from adding numerical fingerprints to binary fingerprints for MACCS 111

Figure 100: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 111

Figure 101: Results from adding numerical fingerprints to binary fingerprints for PubChem 111

Figure 102: Results from adding numerical fingerprints to binary fingerprints for Substructure 111

Figure 103: Classification results from classifying the Fontaine dataset by J48 112

Figure 104: Classification results from classifying the Fontaine dataset by NaïveBayes 112

Figure 105: Classification results from classifying the Fontaine dataset by Random Forest 112

Figure 106: Classification results from classifying the Fontaine dataset by SMO 112

Figure 107: Classification results from classifying the Fontaine dataset by Majority Voting 113

Figure 108: Results from adding numerical fingerprints to binary fingerprints for J48 113

Figure 109: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 114

Figure 110: Results from adding numerical fingerprints to binary fingerprints for Random Forest 114

Figure 111: Results from adding numerical fingerprints to binary fingerprints for SMO 114

Figure 112: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 114

Figure 113: Classifier performance for EState – Original SMOTEd Training 115

Figure 114: Classifier performance for MACCS – Original SMOTEd Training 115

Figure 115: Classifier performance for Pharmacophore – Original SMOTEd Training 115

Figure 116: Classifier performance for PubChem – Original SMOTEd Training 116

Figure 117: Classifier performance for Substructure – Original SMOTEd Training 116

Figure 118: Results from adding numerical fingerprints to binary fingerprints for EState 116

Figure 119: Results from adding numerical fingerprints to binary fingerprints for MACCS 117

Figure 120: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 117

Figure 121: Results from adding numerical fingerprints to binary fingerprints for PubChem 117

Figure 122: Results from adding numerical fingerprints to binary fingerprints for Substructure 117

Figure 123: Classification results from classifying the Fontaine dataset by J48 118

Figure 124: Classification results from classifying the Fontaine dataset by NaïveBayes 118

Figure 125: Classification results from classifying the Fontaine dataset by Random Forest 118

Figure 126: Classification results from classifying the *Fontaine* dataset by SMO . 118

Figure 127: Classification results from classifying the Fontaine dataset by Majority Voting 119

Figure 128: Results from adding numerical fingerprints to binary fingerprints for J48 119

Figure 129: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 119

Figure 130: Results from adding numerical fingerprints to binary fingerprints for Random Forest 120

Figure 131: Results from adding numerical fingerprints to binary fingerprints for SMO 120

Figure 132: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 120

Figure 133: Classifier performance for EState – PCA Dataset 121

Figure 134: Classifier performance for MACCS – PCA Dataset 121

Figure 135: Classifier performance for Pharmacophore – PCA Dataset 121

Figure 136: Classifier performance for PubChem – PCA Dataset 122

Figure 137: Classifier performance for Substructure – PCA Dataset 122

Figure 138: Results from adding numerical fingerprints to binary fingerprints for EState 122

Figure 139: Results from adding numerical fingerprints to binary fingerprints for MACCS 123

Figure 140: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 123

Figure 141: Results from adding numerical fingerprints to binary fingerprints for PubChem 123

Figure 142: Results from adding numerical fingerprints to binary fingerprints for Substructure 123

Figure 143: Classification results from classifying the Fontaine dataset by J48 124

Figure 144: Classification results from classifying the Fontaine dataset by NaïveBayes 124

Figure 145: Classification results from classifying the Fontaine dataset by Random Forest 124

Figure 146: Classification results from classifying the Fontaine dataset by SMO 124

Figure 147: Classification results from classifying the Fontaine dataset by Majority Voting 125

Figure 148: Results from adding numerical fingerprints to binary fingerprints for J48 125

Figure 149: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 125

Figure 150: Results from adding numerical fingerprints to binary fingerprints for Random Forest 126

Figure 151: Results from adding numerical fingerprints to binary fingerprints for SMO 126

Figure 152: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 126

Figure 153: Classifier performance for EState – PCA SMOTEd All 127

Figure 154: Classifier performance for MACCS – PCA SMOTEd All 127

Figure 155: Classifier performance for Pharmacophore – PCA SMOTEd All 127

Figure 156: Classifier performance for PubChem – PCA SMOTEd All.....128

Figure 157: Classifier performance for Substructure – PCA SMOTEd All 128

Figure 158: Results from adding numerical fingerprints to binary fingerprints for EState.....128

Figure 159: Results from adding numerical fingerprints to binary fingerprints for MACCS 129

Figure 160: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 129

Figure 161: Classifier performance for PubChem 129

Figure 162: Classifier performance for Substructure.....129

Figure 163: Classification results from classifying the Fontaine dataset by J48 130

Figure 164: Classification results from classifying the Fontaine dataset by NaïveBayes.....130

Figure 165: Classification results from classifying the Fontaine dataset by Random Forest 130

Figure 166: Classification results from classifying the Fontaine dataset by SMO ..130

Figure 167: Classification results from classifying the Fontaine dataset by Majority Voting 131

Figure 168: Results from adding numerical fingerprints to binary fingerprints for J48 131

Figure 169: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes.....131

Figure 170: Results from adding numerical fingerprints to binary fingerprints for Random Forest.....132

Figure 171: Results from adding numerical fingerprints to binary fingerprints for SMO.....132

Figure 172: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 132

Figure 173: Classifier performance for EState – PCA SMOTEd Training 133

Figure 174: Classifier performance for MACCS – PCA SMOTEd Training 133

Figure 175: Classifier performance for Pharmacophore – PCA SMOTEd Training 133

Figure 176: Classifier performance for PubChem – PCA SMOTEd Training 134

Figure 177: Classifier performance for Substructure – PCA SMOTEd Training 134

Figure 178: Results from adding numerical fingerprints to binary fingerprints for EState 134

Figure 179: Results from adding numerical fingerprints to binary fingerprints for MACCS 135

Figure 180: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 135

Figure 181: Results from adding numerical fingerprints to binary fingerprints for PubChem 135

Figure 182: Results from adding numerical fingerprints to binary fingerprints for Substructure 135

Figure 183: Sensitivity versus False Positive *Fontaine* methods 136

Figure 184: Sensitivity versus False Positive *Fontaine* classifiers 136

Figure 185: Classification results from classifying the AID362 dataset by NaïveBayes 140

Figure 186: Classification results from classifying the AID362 dataset by Random Forest 140

Figure 187: Classification results from classifying the AID362 dataset by Majority Voting 140

Figure 188: Results from adding numerical fingerprints to binary fingerprints for Random Forest 141

Figure 189: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 141

Figure 190: Classifier performance for MACCS – Original 141

Figure 191: Classifier performance for Pharmacophore – Original 142

Figure 192: Classifier performance for PubChem – Original 142

Figure 193: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 143

Figure 194: Results from adding numerical fingerprints to binary fingerprints for PubChem 143

Figure 195: Results from adding numerical fingerprints to binary fingerprints for Substructure 143

Figure 196: Classification results from classifying the AID362 dataset by NaïveBayes 144

Figure 197: Classification results from classifying the AID362 dataset by SMO 144

Figure 198: Results from adding numerical fingerprints to binary fingerprints for J48 144

Figure 199: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 145

Figure 200: Results from adding numerical fingerprints to binary fingerprints for Random Forest 145

Figure 201: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 145

Figure 202: Classifier performance for Pharmacophore – Original SMOTEd All 146

Figure 203: Classifier performance for PubChem – Original SMOTEd All 146

Figure 204: Classifier performance for Substructure – Original SMOTEd All 146

Figure 205: Results from adding numerical fingerprints to binary fingerprints for MACCS 147

Figure 206: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 147

Figure 207: Results from adding numerical fingerprints to binary fingerprints for Substructure 147

Figure 208: Classification results from classifying the AID362 dataset by NaïveBayes 148

Figure 209: Classification results from classifying the AID362 dataset by SMO 148

Figure 210: Classification results from classifying the AID362 dataset by Majority Voting 148

Figure 211: Results from adding numerical fingerprints to binary fingerprints for J48 149

Figure 212: Results from adding numerical fingerprints to binary fingerprints for Random Forest 149

Figure 213: Results from adding numerical fingerprints to binary fingerprints for SMO 149

Figure 214: Classifier performance for Pharmacophore – Original SMOTEd Training 150

Figure 215: Classifier performance for PubChem – Original SMOTEd Training 150

Figure 216: Classifier performance for Substructure – Original SMOTEd Training 150

Figure 217: Results from adding numerical fingerprints to binary fingerprints for EState 151

Figure 218: Results from adding numerical fingerprints to binary fingerprints for MACCS 151

Figure 219: Results from adding numerical fingerprints to binary fingerprints for Substructure 151

Figure 220: Classification results from classifying the AID362 dataset by NaïveBayes 152

Figure 221: Classification results from classifying the AID362 dataset by SMO 152

Figure 222: Results from adding numerical fingerprints to binary fingerprints for Random Forest 153

Figure 223: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 153

Figure 224: Classifier performance for EState - PCA 154

Figure 225: Classifier performance for Pharmacophore - PCA 154

Figure 226: Classifier performance for PubChem - PCA 154

Figure 227: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 155

Figure 228: Results from adding numerical fingerprints to binary fingerprints for Substructure 155

Figure 229: Classification results from classifying the AID362 dataset by NaïveBayes 156

Figure 230: Classification results from classifying the AID362 dataset by SMO 156

Figure 231: Results from adding numerical fingerprints to binary fingerprints for J48 157

Figure 232: Results from adding numerical fingerprints to binary fingerprints for Random Forest 157

Figure 233: Results from adding numerical fingerprints to binary fingerprints for SMO 157

Figure 234: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 157

Figure 235: Classifier performance for PubChem – PCA SMOTEd All 158

Figure 236: Classifier performance for Substructure – PCA SMOTEd All 158

Figure 237: Results from adding numerical fingerprints to binary fingerprints for EState 159

Figure 238: Results from adding numerical fingerprints to binary fingerprints for MACCS 159

Figure 239: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 159

Figure 240: Results from adding numerical fingerprints to binary fingerprints for Substructure 159

Figure 241: Classification results from classifying the AID362 dataset by NaïveBayes 160

Figure 242: Classification results from classifying the AID362 dataset by Random Forest 160

Figure 243: Classification results from classifying the AID362 dataset by SMO 160

Figure 244: Results from adding numerical fingerprints to binary fingerprints for J48 161

Figure 245: Results from adding numerical fingerprints to binary fingerprints for Random Forest 161

Figure 246: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 161

Figure 247: Classifier performance for MACCS – PCA SMOTEd Training 162

Figure 248: Classifier performance for Pharmacophore – PCA SMOTEd Training 162

Figure 249: Classifier performance for Substructure – PCA SMOTEd Training 162

Figure 250: Results from adding numerical fingerprints to binary fingerprints for EState 163

Figure 251: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 163

Figure 252: Results from adding numerical fingerprints to binary fingerprints for Substructure 163

Figure 253: Sensitivity versus False Positive *AID362* methods 164

Figure 254: Sensitivity versus False Positive *AID362* classifiers 164

Figure 255: Classification results from classifying the AID456 dataset by NaïveBayes 167

Figure 256: Classification results from classifying the AID456 dataset by SMO 167

Figure 257: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 168

Figure 258: Results from adding numerical fingerprints to binary fingerprints for Random Forest 168

Figure 259: Classifier performance for MACCS 169

Figure 260: Classifier performance for PubChem 169

Figure 261: Classifier performance for Substructure 169

Figure 262: Results from adding numerical fingerprints to binary fingerprints for MACCS 170

Figure 263: Results from adding numerical fingerprints to binary fingerprints for PubChem 170

Figure 264: Results from adding numerical fingerprints to binary fingerprints for Substructure 170

Figure 265: Classification results from classifying the AID456 dataset by NaïveBayes 171

Figure 266: Classification results from classifying the AID456 dataset by SMO 171

Figure 267: Results from adding numerical fingerprints to binary fingerprints for J48 172

Figure 268: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 172

Figure 269: Results from adding numerical fingerprints to binary fingerprints for Random Forest 172

Figure 270: Results from adding numerical fingerprints to binary fingerprints for SMO 172

Figure 271: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 173

Figure 272: Classifier performance for MACCS 173

Figure 273: Classifier performance for PubChem 174

Figure 274: Results from adding numerical fingerprints to binary fingerprints for EState 174

Figure 275: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 174

Figure 276: Results from adding numerical fingerprints to binary fingerprints for Substructure 175

Figure 277: Classification results from classifying the AID456 dataset by NaïveBayes 175

Figure 278: Classification results from classifying the AID456 dataset by SMO 175

Figure 279: Results from adding numerical fingerprints to binary fingerprints for J48 176

Figure 280: Results from adding numerical fingerprints to binary fingerprints for SMO 176

Figure 281: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 176

Figure 282: Classifier performance for EState 177

Figure 283: Classifier performance for Pharmacophore 177

Figure 284: Classifier performance for Substructure 178

Figure 285: Results from adding numerical fingerprints to binary fingerprints for EState 178

Figure 286: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 178

Figure 287: Classification results from classifying the AID456 dataset by NaïveBayes 179

Figure 288: Classification results from classifying the AID456 dataset by Random Forest 179

Figure 289: Classifier performance for EState 180

Figure 290: Classifier performance for Pharmacophore 181

Figure 291: Classifier performance for Substructure 181

Figure 292: Results from adding numerical fingerprints to binary fingerprints for EState 182

Figure 293: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 182

Figure 294: Results from adding numerical fingerprints to binary fingerprints for Substructure 182

Figure 295: Classification results from classifying the AID456 dataset by NaïveBayes 182

Figure 296: Classification results from classifying the AID456 dataset by SMO 183

Figure 297: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 183

Figure 298: Results from adding numerical fingerprints to binary fingerprints for SMO 184

Figure 299: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 184

Figure 300: Classifier performance for MACCS 184

Figure 301: Classifier performance for Pharmacophore 185

Figure 302: Classifier performance for PubChem 185

Figure 303: Results from adding numerical fingerprints to binary fingerprints for EState 185

Figure 304: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 186

Figure 305: Results from adding numerical fingerprints to binary fingerprints for PubChem 186

Figure 306: Results from adding numerical fingerprints to binary fingerprints for Substructure 186

Figure 307: Classification results from classifying the AID456 dataset by NaïveBayes 187

Figure 308: Classification results from classifying the AID456 dataset by SMO 187

Figure 309: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes 187

Figure 310: Results from adding numerical fingerprints to binary fingerprints for SMO 188

Figure 311: Results from adding numerical fingerprints to binary fingerprints for Majority Voting 188

Figure 312: Classifier performance for EState 189

Figure 313: Classifier performance for MACCS 189

Figure 314: Classifier performance for Pharmacophore 189

Figure 315: Classifier performance for PubChem 190

Figure 316: Classifier performance for Substructure 190

Figure 317: Results from adding numerical fingerprints to binary fingerprints for EState 191

Figure 318: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore 191

Figure 319: Results from adding numerical fingerprints to binary fingerprints for Substructure 191

Figure 320: Sensitivity versus False Positive *AID456* methods 191

Figure 321: Sensitivity versus False Positive *AID456* classifiers 192

Figure 322: Bursi dataset classifiers' performance 197

Figure 323: Bursi dataset methods' performance 197

Figure 324: Fontaine dataset methods' performance 198

Figure 325: Fontaine dataset classifiers' performance 199

Figure 326: AID362 dataset methods' performance 200

Figure 327: AID362 classifiers' performance 200

Figure 328: AID456 methods' performance 201

Figure 329: AID456 classifiers' performance 201

Figure 330: Possible class imbalance scenarios (Amended from Sáez et al. 2016, p.161) 205

Figure 331: Various types of examples identified in a multi-class situation (Sáez et al. 2016, p.167) 207

List of Tables

Table 1: AID362 specifications. Class of interest has a 1 next to the label.....34

Table 2: AID456 specification. Class of interest has a 1 next to its label.....35

Table 3: Mutagenicity dataset specification.....35

Table 4: Factor XA dataset specification.....36

Table 5: Detailing the properties of the various fingerprints used.....43

Table 6: Misclassification of raw PubChem datasets #1 51

Table 7: Misclassification of raw PubChem datasets #2 51

Table 8: A cost matrix showing the misclassification cost for positives and negatives 52

Table 9: Original number of samples in unbalanced datasets.....57

Table 10: Number of samples in balanced datasets 57

Table 11: Advantages of decision trees, Naïve Bayes, SVM classifiers. (Source: Galathiya et al. 2012).....68

Table 12: Some of the features from classifiers used in this study. (Source: Galathiya et al. 2012) 69

Table 13: Summary of the number of features generated by various fingerprinting techniques 74

Table 14: Mutagenicity dataset specification. Class of interest labelled as 1.....78

Table 15: Euclidean distance for the methods used.....95

Table 16: Euclidean distance for the classifiers used.....95

Table 17: Factor XA dataset specification. Class of interest labelled as 1 98

Table 18: Euclidean distance for the methods used.....136

Table 19: Euclidean distance for the classifiers used.....137

Table 20: AID362 dataset specification. Class of interest labelled as 1 139

Table 21: Euclidean distance for the methods used 164

Table 22: Euclidean distance for the classifiers used 165

Table 23: AID456 Dataset specification. Class of interest labelled as 1 167

Table 24: Euclidean distance for the methods used 192

Table 25: Euclidean distance for the classifiers used 192

Acknowledgements

To Emili: Thank you for being such an amazing supervisor, for your support at times when I needed it most and for believing in me and motivating me.

To Naomi: Thank you for your support in all the stages of this degree.

To Mark: Thank you for supporting us with taking this degree further.

To all my friends and colleagues at Bournemouth University: Thank you for being part of this journey and for making it a very pleasant one. Special thanks go to Ed, Manuel, Cristina, Rashid, Anna and Alex.

This work is dedicated to my Mother, who has supported me through all my decisions in life. Without you I would have never been able to get this far in my life. Thank you and God bless you.

To my Father for great advices and good thoughts throughout this journey.

I would like to thank the School of Design, Engineering and Computing, SMART Technology Research Centre and the Graduate School for their help in administration and financial support with expenses. Special thanks to Kelly Duncan Smith, Malcolm Green, Angela Tabeshfar, Pattie Davis and Fiona Knight.

Bournemouth University, thank you for providing a productive and enjoyable research and work environment.

List of Abbreviations

DM	Data Mining
ESt	EState Fingerprinter
Ext	CDK Extended Fingerprinter
Fin	CDK Fingerprinter
FNR	False Negative Rate
FPR	False Positive Rate
Gra	CDK Graph-Only
HTS	High-Throughput Screening
MAC	MACCS
NB	NaïveBayes
Pha	Pharmacophore
Pub	PubChem
RF	Random Forest
SMO	Sequential Minimal Optimisation
Sub	Substructure
TNR	True Negative Rate
TPR	True Positive Rate
VS	Virtual Screening

1. Introduction

Virtual screening in drug discovery involves screening datasets containing unknown molecules in order to find the ones that are likely to have the desired effects on a biological target. The molecules are thereby classified into active or non-active compared to the target. Misclassification of molecules in cases such as drug discovery and medical diagnosis is costly, both in time and finances. In the process of discovering a drug, it is mainly the inactive molecules classified as active towards the biological target i.e. false positives that cause a delay in the progress and high late-stage attrition.

1.1. Background

Chemoinformatics (Cheminformatics) as defined by Frank Brown (1998) is the mixing of resources in order to transform data into information and information into knowledge in order to make faster and better decisions in the field of drug identification and optimisation. In short, computational methods are used to process chemical data in particular the chemical data structure. Some of the techniques used in Chemoinformatics such as computational chemistry and QSAR (Quantitative Structure-Activity Relationship) are very well-known and –established and have been practiced for years in the industry and laboratories.

Drug discovery is the process by which new medicinal candidates are discovered. To achieve this, compounds which are likely to have wanted effects on a biological target (disease) are identified and isolated. High-Throughput screening (HTS) is used to assess the binding ability –activity of compounds against the target. This is also known as empirical or physical screening. HTS screens thousands of compounds in order to find new candidates in a fast and accurate manner. There are two stages of screenings: primary and secondary (confirmatory). The biological relevance of the compounds identified as hits from the primary stage are assessed. These compounds are then screened for a second time. Confirmed hits from this stage are called leads which will be further optimised to become candidates for clinical tests. Advances in molecular biology and the use of combinatorial chemistry have resulted in an increase in the number of biological targets and compounds in libraries. HTS is characterised by its screening capacity which is about 10000 –

100000 compounds per day. The significant increase in the number of available compounds as well as biological targets requires scientists to reduce the size of HTS assays (Mayr & Bojanic 2009). Considering the fact that HTS is a very costly process, alternative techniques such as virtual screening could be utilised in order to filter compounds which are selected for screening.

Virtual screening or biophysical screening is the in-silico screening of compounds. It uses computational methods to score, rank or filter a set of compound structures. Virtual screening can be used to determine which compounds to screen against a given target. It has been acknowledged that in order to identify desirable compounds from a library there needs to be an increase in the quality of the library rather than the quantity (Bajorath 2002). This helps carry out fewer but smarter experiments. Virtual screening assists the detection of new bioactive compounds by reducing the number of compounds that are to be screened based on scoring criteria. This reduction is achieved by eliminating the compounds which do not show activity towards a given target.

HTS has become an important source for identifying new compounds for optimisation in medium to large pharmaceuticals. It has proven to be a useful technology for providing new hits for the drug discovery process. However not all hits identified by HTS are appropriate leads for further medicinal optimisations. In fact the overall HTS success rate currently is estimated at 45-55% (S. Fox et al. 2006; Keserü & Makara 2009). HTS suffers from two types if errors: type 1 and type 2 (Martis et al. 2011). Type 1 errors are false positives. These are compounds which are regarded as actives but later turn out to be non-active. Type 2 errors are false negatives. These are active compounds which are regarded as non-actives in the screening process. One of the main challenges in HTS is to differentiate between compounds which are genuinely active towards a target and false positive compounds. In biological terms a compound that is genuinely active against a target has a high tendency to form a non-covalent bond with the target which is reversible (Thorne et al. 2010). All other compounds that form bindings with the target but do not possess the characteristics of the genuine interaction are false positives. These compounds are generally active in an assay, but their activity is target-independent. They can affect by forming aggregates, they can be protein-active or interfere with assay signalling. This all leads to them being considered as active and therefore the

secondary screenings normally include a great deal of false positive compounds. Manually filtering compounds using the knowledge of chemists is a good way of reducing false positives but as mentioned in (Sink et al. 2010) an analysis of such a method showed inconsistency in the compounds which were to be taken out.

False positive compounds escape various screenings undetected. They are one of reasons there is high late-stage attrition in the drug discovery process. These are compounds which fail to qualify as being suitable lead compounds for drug optimisation. The costs of the process increase as we get to the later stages of it, as seen in Figure 1.

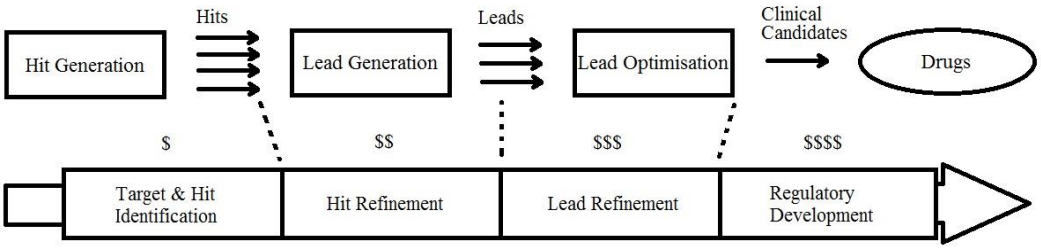


Figure 1: Compound attrition and cost increase of drug discovery process by time (Bleicher et al. 2003, p.371)

It can also be seen in Figure 1 that as we get to the later stages of the process the number of compounds decrease (the number of arrows). There are fewer compounds to work with and the processes become more expensive. For example in the pharmaceutical industry, the main pre-clinical expense is the lead optimisation process (Jorgensen 2012). It makes sense to have more suitable compounds (opportunities) at hand in order to increase the chances of discovering better leads.

1.2. Project description and goals

The main objective of this project is to explore and investigate the application and the effects of using various fingerprinting methods combined with the Synthetic Minority Oversampling technique on the classification of highly imbalanced, high-dimensional datasets.

This research tends to examine different methods of manipulating big imbalanced datasets that have not been cleared of noise, and to see how they can affect the various classification evaluation metrics. In other words we look at how the false positive numbers in specific change.

The main goal of this project is to examine the different techniques by which big and highly imbalanced datasets that have not been cleared of noise, can be manipulated in order to see the effect on classification evaluation metrics.

In order to achieve the project goal the following objectives are pursued:

- Critically investigating the various methods to classify big imbalanced datasets
- Generating fingerprints from raw datasets
- Using Synthetic Minority Oversampling TEchnique to oversample training and / or test sets
- Classifying the various resulting oversampled datasets and comparing the metrics
- Identification and recommendation of appropriate techniques

1.3. Methodology and organisation of thesis

In order to better understand this thesis a general knowledge of the drug discovery process and how Chemoinformatics has influenced it, is necessary to explore the basics behind the science of Chemoinformatics. This introductory information will be expanded in Chapter 2, accompanied by a literature review and discussion of the important contributions in the areas.

Chapter 3 will provide the reader with information about the datasets; their origin, size and class distribution. Some detail about how the datasets were collected and transformed in the format to be used for this research will also be provided.

Chapter 4 discusses the methods that were used in this research for gathering the results.

Chapter 5 will display the results from the datasets used. In this chapter a brief description of the datasets is given followed by a discussion of the results for each.

This is followed by Chapter 6 where an overall and in depth discussion of the results is given.

Finally Chapter 7 will provide the reader with the conclusion of this thesis and an overview of the future work.

1.4. Publication

List of publications:

- Rafati-Afshar, A.A. and Bouchachia, A., 2013, October. An Empirical Investigation of Virtual Screening. In *2013 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 2641-2646). IEEE.

2. Representation and Visualization of Chemical Structures

This chapter consist of a detailed critical review of the Data Visualization and Chemical Structures Analysis techniques. The first two sections discuss visualization and searching aspects in large chemical structures datasets, a necessary pre-processing step for the novel approach developed in this PhD. Since this is not a primary aspect in this dissertation, the description will be succinct. Hence, the focus will be put next on High-throughput and virtual screening methodologies, which is the main topic addressed of this project. The last two sections discuss the two major challenges involved in preforming an effective screening, the strong class-imbalance and the difficulties of handling big datasets.

2.1. Visualizing of Chemical Structures

The first step before analysing large datasets of chemical structures is the efficient database design and display. In a nutshell, there are several means by which a chemical structure can be effectively stored and displayed; drawing the structure using specialised programs such as ChemDraw (Ultra 2001) or scanning the structure as an image or in text format. In Chemoinformatics chemical compounds need to be stored in databases for search and retrieval based on chemical structure (Leach & Gillet 2007).

There are various ways of representing the chemical compound structures. Some of the more popular ones have been explained below. The popular type of representation is the two-dimensional chemical structure (Brown 2009). This representation is shown in Figure 3 in a basic form and in using Caffeine as the example compound, where the lines that connect Nitrogen and Carbon atoms are single bonds and the double lines connecting Carbon and Oxygen atoms are double bonds (Carbon atoms are not explicitly shown in Figure 3 for simplicity).

Graph

A graph is an abstract structure that has nodes connected by edges (please see Figure 2). It shows how the edges and nodes in a molecule are connected. Molecular structures are normally stored in a database using *Molecular Graphs*; a type of graph

where the nodes are the atoms and the edges are the bonds (Leach & Gillet 2007; Brown 2009).

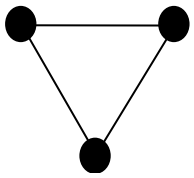


Figure 2: A graph with nodes (a, b and c) and edges (lines that connect the nodes ab, ac and bc)

One important use of the graph theory in Chemoinformatics is its application in determining structural similarity between a set of molecules (Basak et al. 1988). A requirement for two graphs to be the same or isomorphs is for both to have the same number of nodes and edges and for every one of them to have a corresponding match in the other graph (Leach & Gillet 2007).

Molecular graphs such as the example shown form the basis for molecular structure demonstration. The main reason for using this representation is simply that molecular graphs are easy to read and understand by chemists, but they are not trivial to map into databases due to the intricate nonlinearity and complexity of the graphs involved (Burden 1998; Kearnes et al. 2016); and the mapping into a database requires a nontrivial pre-processing (Polanski 2009); as will be further discussed next.

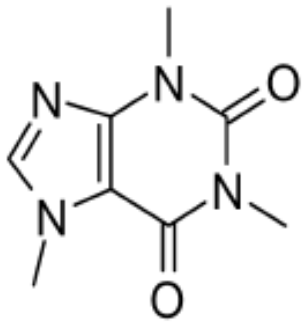


Figure 3: A Hydrogen-depleted molecular graph of Caffeine (Brown, 2009)

Connection Table

A connection table is a scheme which enables the efficient coding of molecular graphs. Connection tables record the data in a tabular form. This allows for a decrease in the amount of data with the increase in the size of the molecule (Polanski, 2009). This scheme was developed with the purpose of storing and transferring chemical structure information at the Molecular Design Limited labs (now called Symyx and merged with Accelrys), details of which can be found in Dalby et al. (1992) and in the specifications document produced by Symyx at www.symyx.com (Symyx, 2010). A very simplified example of a connection table together with an example molecule can be seen in Figure 4.

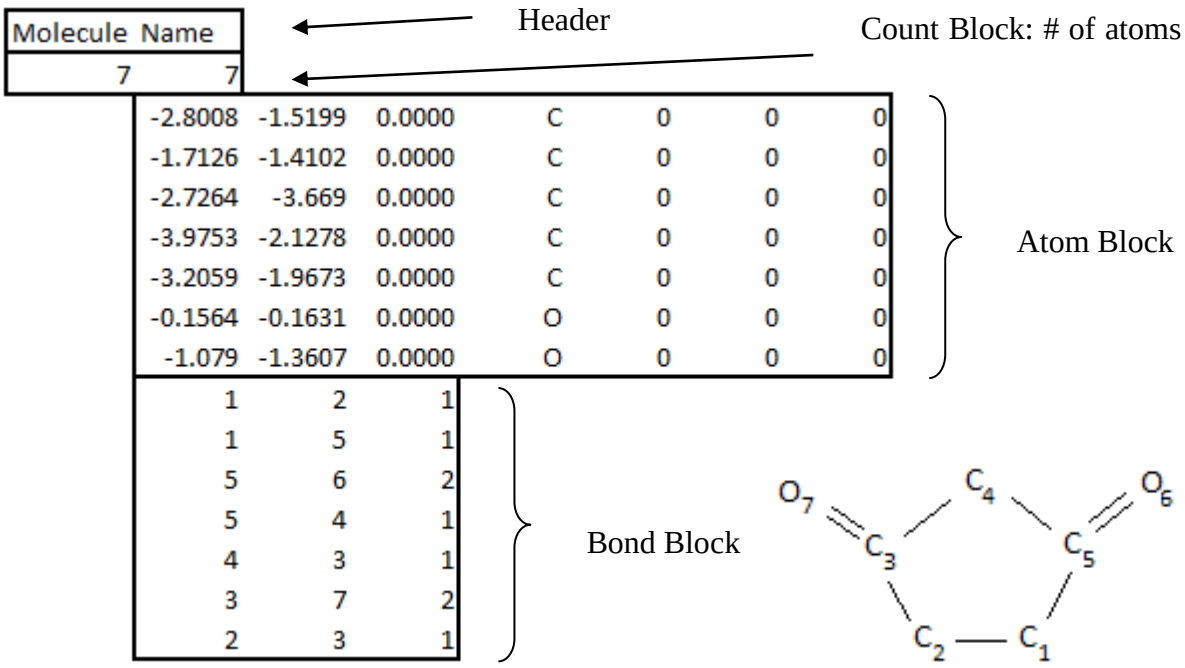


Figure 4: Connection table example with an example molecule

In the connection table shown in Figure 4, each rectangle represents a "block" as referred to in the descriptions. The header block contains information about the molecule name, user, programme used and any other comments. The count block (in Figure 4) includes information about the number of atoms and the number of bonds (any of several forces by which atoms are bound in a molecule) available in the molecule. In the atom block, there is a line of information per atom. This block contains the node information. If we consider the first line of the atom block in Figure 4, the first three real numbers indicate the x, y and z spatial coordinates of the

atom. The capital letter shows the atom type (i.e. C for Carbon and O for Oxygen). This block can also contain information about the atom-charge, stereochemistry (the three-dimensional arrangement of atoms and molecules and the effect it has on chemical reactions), associated hydrogens, etc., all related to the specified atom. The bond block as shown in Figure 4 contains information about the different bond types available between the atoms (the edges) in the molecule. The information in this block is also organised in a line by line manner i.e. if we look at the first line, the first two columns are the atom numbers connected by a bond in the molecule; and the third column is the type of the bond between the two atoms (1 = single, 2 = double). So as an example in Figure 4, the first line of the bond block has the numbers 1, 2 and 1 in it. We could refer to the picture of the molecule next to the connection table and see that atoms 1 and 2 are indeed connected by a single bond. Respectively one can see that in the same block, the third line contains the numbers 5, 6 and 2 which mean atoms 5 and 6 are connected in the molecule through a double bond.

Linear Notation

Linear notations are alternative ways of representing and communicating molecular graphs. Here alphanumeric characters are used to encode the molecular structure (Leach & Gillet 2007). This notation allows the molecule to be displayed in the form of a string similar to that of line formulae. A line formula is made up of atoms that are joined by lines representing single or multiple bonds without any indication of the spatial direction of the bonds (Polanski 2009). Please see Figure 5. Line notation became popular because it represents the molecular structure by a linear string of symbols which is quite similar to natural language (Weininger 1988).

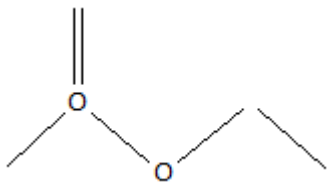


Figure 5: Example of a line formula for the molecule shown in Figure 4.

Weininger (1988) mentions in his influential review" SMILES, a Chemical Language and Information System" that in the early days processing and storing chemical information was dependent on the description of the chemical structure. Many systems were therefore developed in order to generate unique machine

descriptions amongst which were application of graph theory to chemical notation (Balaban 1985) and chemical substructure search systems (Stobaugh 1985). As mentioned a above, over the years research in molecular representation has switched towards encoding molecular structures as a simple line notation, mainly for data storage capacity which is particularly favourable in these compressed representations. Linear notations are indeed more compact than connection tables thus they take less space and are ideal for storing and sharing large molecules (Weininger 1988; Brown 2009).

The most widespread linear notation currently in use is SMILES (Simplified Molecular Input Entry System). It is simple, easy to use and understand. Only a few rules are needed in order to write most SMILES strings (Leach & Gillet, 2007; Toropov & Benfenati, 2007; Brown, 2009; Polanski, 2009; Sammadar et al. 2015). This encoding system can be found in Appendix A. An example of SMILES notation for the caffeine molecule can be seen in Figure 6a.

Connection tables and SMILES notations can be constructed in many different ways. For example with SMILES, one can start writing the alphanumeric string starting at any atom and follow a different sequence through the molecule. Same issue can arise with a connection table as one can specifically select to number the atoms in a molecule different to another one (Leach & Gillet, 2007; Brown, 2009). Therefore it is not possible to distinguish whether two SMILES notations or two connection tables are similar. To solve this problem, the Canonical (standardised) representation was introduced so that the atoms in a molecular graph would be ordered in a unique manner. Such representations manifest themselves in code systems such as IUPAC (International Union of Pure and Applied Chemistry) and InChi (International Chemical Identifier) which can uniquely encode a molecule in very compact form (Brown 2009; Fuchs et al. 2015; Heller et al. 2015).

SMILES	CN1C=NC2=C1C(=O)N(C(=O)N2C)C
IUPAC	1,3,7-trimethylpurine-2,6-dione
InChi	1S/C8H10N4O2/c1-10-4-9-6-5(10)7(13)12(3)8(14)11(6)2/h4H,1-3H3

Figure 6: SMILES, IUAPC and InCHi representations for Caffeine (Source: Brown 2009)

2.2. Searching for Compounds in Databases

A significant aspect to consider in Chemoinformatics is the design of Databases, which is necessarily high specific of this setting due to the complexity of the information stored. Databases that hold information about chemical structures tend to be specialised due to the nature of the methods which are used to store and manipulate the chemical structures. One can query a database containing chemical structures in order to find similar molecules. Brown (2009) defines this issue as "the rationalisation of a large number of compounds so that only the desirable remains".

2.2.1. Structure and Sub-Structure Searching

Molecules can be sought in a database based on their structure. For this to happen, the user query needs to be translated into a standard representation (relevant to the database). If the database is arranged in a way so that Hash-Keys correspond to the locations of structures, then information retrieval can happen almost immediately by comparing the key produced from the query to the database structure-key. Sometimes however there is a slight chance that one hash-key can match to more than one structure. This phenomenon will be explained further on in the literature when describing hash-key fingerprints.

An alternative way to search for structures which also decreases the search time is looking for specific sub-structure(s) in the molecules in a database. A chemical sub-structure is a part of a molecule; sub-structure search involves checking for the presence of a certain partial structure in the whole molecule (Willet 2009). If a query is made for a sub-structure in a set of molecules, then that specific sub-structure needs to appear completely in the matching molecule (Schomburg et al. 2013). The molecules in the database being searched either match the query or not (Hood et al. 2015). This action removes the molecules that do not contain that sub-structure. Afterwards the more time-consuming sub-structure search algorithms (i.e. graph isomorphism) can be applied to the remaining molecules to see which of them truly match the query (Leach & Gillet 2007; Brown 2009). A chemical sub-structure must not be confused with a chemical pattern. A chemical pattern can be a generic or highly specific description of a chemical function. Chemical functions are used in

many contexts, mainly to comprehensively describe a large collection of sub-structures (Schomburg et al. 2013).

Structure and sub-structure searching involve the design of a precise query and are useful for selecting compounds that have not yet been screened from a database but they have some limitations:

- The formulation of the query can be complex for the non-expert; one is required to have enough knowledge about a structure or sub-structure in order to be able to form a meaningful query (Lemfack et al. 2014). This can become a challenge when only a few active compounds are known.
- When performing this kind of search, as mentioned before, the molecules either match the query or they do not. As a result the database is effectively partitioned into two sections (matched items and non-matched items), but there exists no relative ranking of the compounds in comparison to the structure in question (Leach & Gillet 2007). In other words the output is not ranked in any way other than by the date the database was accessed (Willet 2009).
- User has no control over the volume of the output. This means that if the query is too general there can be a large number of hits, and if the query is too specific, the output could be very small and limited (Willet et al. 1998).

In order to overcome these drawbacks, an alternative method was developed called “Similarity Searching” (Downs & Willet 1996) which allows for a more flexible molecular database search; as discussed in the next sub-section. Similarity searching suffers from none of the drawbacks mentioned for sub-structure searching.

2.2.2. Similarity Searching

The concept of similarity plays an important role in Chemoinformatics (Maggiora & Shanmugasundaram 2011; Willet 2014). Similarity (fuzzy) searching is an alternative and complimentary to exact (structure and sub-structure) searching; it retrieves the exact matches to the query object and other similar ones (Monev 2004). Here a query is used to search a database for compounds that are most similar to it (Leach & Gillet 2007). A ranked list is then generated according to the similarity to the query compound. This allows the results to be ordered based on the likelihood that they would produce the same effects as the reference compound (Brown 2009).

Similarity searching is used within the family of techniques called virtual screening which we shall discuss in the next section.

Similarity searching is based on the *Similarity Property Principle* first enunciated by Johnson and Maggiora (1990), which assumes that molecules which are structurally similar to the query molecule have similar properties i.e. biological activity (Monev 2004; Brown 2009). Also according to the principle, a similar molecule which is higher in the ranking is more likely to be active than another molecule at a lower level (Willett 2006). However in some cases structurally similar molecules have shown similar biological activities and some dissimilar molecules have shown similar biological activity (Medina-Franco 2012; Rivera-Borroto 2016). But this does not invalidate its use in drug discovery. After all if it were not for some relationship between chemical similarity and biological activity of two molecules, it would be really difficult to formulate approaches for drug discovery which take into account the structures of molecules (Willet 2009).

Assessing the extent of similarity is a pure subjective matter (Leach & Gillet 2007); there are thus no "hard and fast" rules. The methods used to measure the similarity between two molecules require three components (Willet 2009; Bajorath 2011; Willet 2014):

1) The molecular representation or descriptor: For characterising the two molecules being compared.
2) The weighting scheme: Used to assign the relative importance of the different parts of the representation.
3) The similarity coefficient: This component is used to measure the similarity between two molecules based on their appropriately weighted representations.

These components control the effectiveness of the search. A more detailed explanation for the components mentioned is provided next.

2.2.3. Molecular Representation

Molecules contain many features (properties). On their own, the individual features are not particularly informative. However a combination of them will provide a better and richer characterisation of the molecule being studied. Molecular descriptors are descriptions of molecules that aim to characterise the most noticeable aspects of a molecule (Leach & Gillet 2007; Brown 2009). They are the final results

of logic and mathematical procedures which transform the chemical information encoded in the structure of a molecule, into useful numbers (Todeschini & Consonni 2009; Yap 2011).

Representation (describing) of molecules means converting molecules into a series of bits that can be easily read and interpreted by computers. Todeschini & Consonni (2009) define it as a way that a molecule is symbolically represented using specific formal procedures conventional rules. Under the concept of similarity, this involves a series of comparisons between a structure or sub-structure query (the reference molecule) and an unknown molecule from a database. Molecular descriptors are of high importance in Chemoinformatics since generating them allows chemical structure information to be statistically analysed (Brown 2009; Yap 2011). There are different techniques for representing chemical molecules. Many authors (Leach & Gillet 2007; Todeschini & Consonni 2009; Bajorath 2011; Warr 2011; Willet 2014) have classified these techniques into three main groups:

1) Whole molecule descriptors (1D)
2) Descriptors that can be calculated from 2D representations of the molecule
3) Descriptors that are calculated from 3D representations of the molecule

2.2.4. 1D Molecular Descriptors

Whole molecule descriptors are measured or computed numbers which describe bulk molecular properties such as the molecular weight or the number of rotatable bonds. 1D descriptors (on their own) do not allow for meaningful comparison between different molecules. Therefore a molecule is normally represented by many such descriptors (Bajorath 2011; Willet 2014).

2.2.5. 2D Molecular Descriptors

2D molecular descriptors are calculated from a chemical structure diagram called the connection table (explained earlier on) which details all of the atoms and bonds in a molecule. The most important 2D molecular descriptors are topological indices and fragment sub-structures. A topological index is a single number that characterises a structure according to its size and shape (Bajorath 2011). Sub-structure based descriptors characterise a molecule by the sub-structural features it has, either with the help of the molecules 2D chemical graph or by its fingerprints.

Currently different 2D molecular descriptors exist, each from distinct descriptor classes. Brown (2009) categorises molecular descriptors into two main classes, Information-based and Knowledge-based descriptors. Information-based descriptors describe what we have. These types of descriptors tend to capture as much as possible information within a molecular representation. On the other hand knowledge-based descriptors describe what we expect. The descriptors that calculate molecular properties based on existing data or models based on such data are of the knowledge-based type.

When searching for chemical molecules of interest (to the user) in a large chemical database, the use of sub-structure searching is often time-consuming and slow because it is a nondeterministic polynomial time problem. For this reason, most chemical databases use a widely used two-stage approach sub-structure search in order to save time and quickly filter out non-matching ones. The aim is to discard and eliminate most of the molecules that cannot possibly match the sought sub-structure. The molecules which remain are then subjected to the more sluggish sub-structure searching algorithms (Leach & Gillet 2007; Brown 2009). This elimination process is assisted by the use of molecule screens. Molecule screens are binary string representation of the molecules and the query sub-structure and they are called bit-strings (Leach & Gillet 2007). Bit-strings are sequences of zero(s) and one(s); a one shows the presence of a structural feature and a zero shows its absence. The great advantage about using bit-strings is that they are the natural currency of computers and therefore can be very quickly manipulated and compared. If a feature is present in the query sub-structure (bit is set to 1) and the corresponding bit in the molecule is set to zero (feature is absent) then from the bit-string comparison it is clear that the molecule does not contain the sub-structure in question and cannot be selected. The opposite does not hold as there can be features in the molecule that are not present in the query sub-structure. These bit-strings are vector-based representations which can also be referred to as fingerprints.

Most binary screening methods are performed using one of the following two approaches:

1. Using a Structural-Key fingerprint
2. Using a Hash-Key fingerprint

2.2.6. Structure-Key Fingerprints (Dictionary-Based)

The structure-key (also known as dictionary-based) fingerprint utilises a dictionary of pre-defined sub-structures, which can be identified through chemists' intuition or from empirical information mined from drug-like molecules databases (Brown et al. 2005), in order to generate bit-strings where each bit corresponds to the presence of certain features in the molecule that are present in the dictionary. This makes interpreting the structure-key fingerprints easier (Brown, 2009; Willet 2011; Willet 2014). Structural keys were the first kind of screening technique which were applied to chemical databases (DAYLIGHT Chemical Information Systems 2008).

When a molecule is added to a database, it is checked against each sub-structure in the dictionary. The bit-string for that particular molecule has all its bits initially set to zero. If a sub-structure in the dictionary is matched to a part in the molecule then that bit is set to 1. The structure-key fingerprints can contain information about the numbers and the quantity of a particular type of feature (for example particular chemical groups, rings). Therefore when designing the dictionary, the goal is to produce structure-keys which provide optimal performances when searching for chemical structures in a database. For that to happen, one needs to decide which patterns are important, the type of molecules expected to be stored in the database and the typical search queries.

Structure-key fingerprints are considered knowledge-based descriptors (Leach & Gillet 2007; Brown 2009; Bajorath 2011; Willet 2014) since the dictionaries are designed based on the knowledge of existing chemical entities and in particular, what is expected to be to be of interest for the domain the dictionary was designed for. In structure-key fingerprints each bit often corresponds with a specific sub-structure. This makes the interpretation of the analysis results easier and more straightforward, especially if it is shown that some activity is related to the presence of specific bits (Leach & Gillet 2007; Willet 2011; Willet 2014). This is the so-called reversibility of the molecular descriptor.

Figure 7 shows an example of a structure-key fingerprint. The Boolean fragment represents a generated structural key where the bits set to one (1) are each assigned to a structure and no other one.

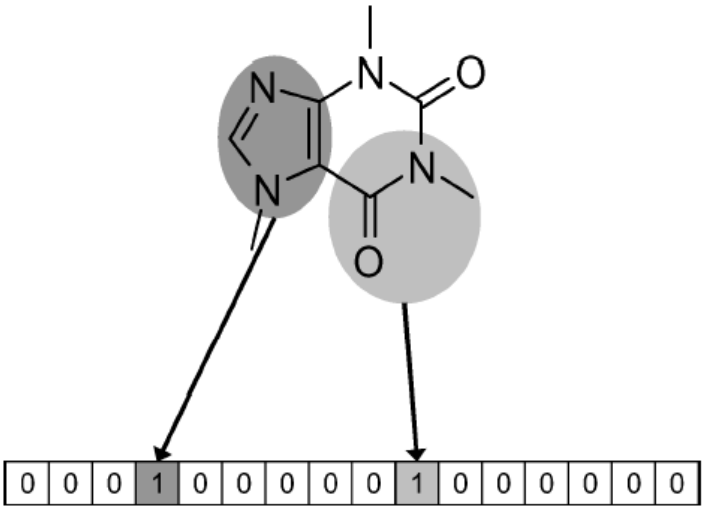


Figure 7: Example of Structure-key fingerprint (Brown 2009)

2.2.7. Hash-Key Fingerprints (non-Dictionary-Based)

Hash-Key fingerprints are an alternative to their Structure-key counterparts. They do not require a pre-defined dictionary of sub-structures of interest. In fact they can be generated directly from the molecules themselves. These fingerprints are also vector-based representations just like the structure-key fingerprints.

When generating this type of fingerprints, each atom (the smallest particle of a substance that can exist by itself or be combined with other atoms to form a molecule) in a given molecule is iterated over, with all atom-bond paths from that atom being calculated between a defined minimum and maximum (usually between 0-7). Each of these paths are then used as an input to a hash function such as Cyclic Redundancy Check (CRD) in order to generate a larger value integer(Leach & Gillet 2007; DAYLIGHT Chemical Information Systems 2008; Brown 2009; Bajorath 2011). This integer can be folded using modulo arithmetic algorithm so that it conforms to the length of the binary string used to represent the molecule. Alternatively the output from the CRD is passed as a seed to a random number generator (RNG) and a few indices, usually 4-5, are taken from the RNG result. Each of these indices are reduced to the length of the fingerprint being used by applying modulo arithmetic algorithm. The set of resulting indices are used to set or update relevant positions in the fingerprint.

Because each path in a molecule is now represented using a number of indices (4-5 as mentioned before), in order to reduce the chances of another molecular path having the same bit pattern and to avoid a molecular path collision, the RNG is used.

The pseudocode for a typical hash-key fingerprint is shown in Figure 8:

```
foreach atom in  molecule
  foreach path from  atom
        seed = crc32(path)
        srand(seed)
        for I = 1 to  N
              index  = rand( ) % bits
              setBit(index)
```

Figure 8: Pseudocode of a typical Hash-key fingerprint (Brown 2005)

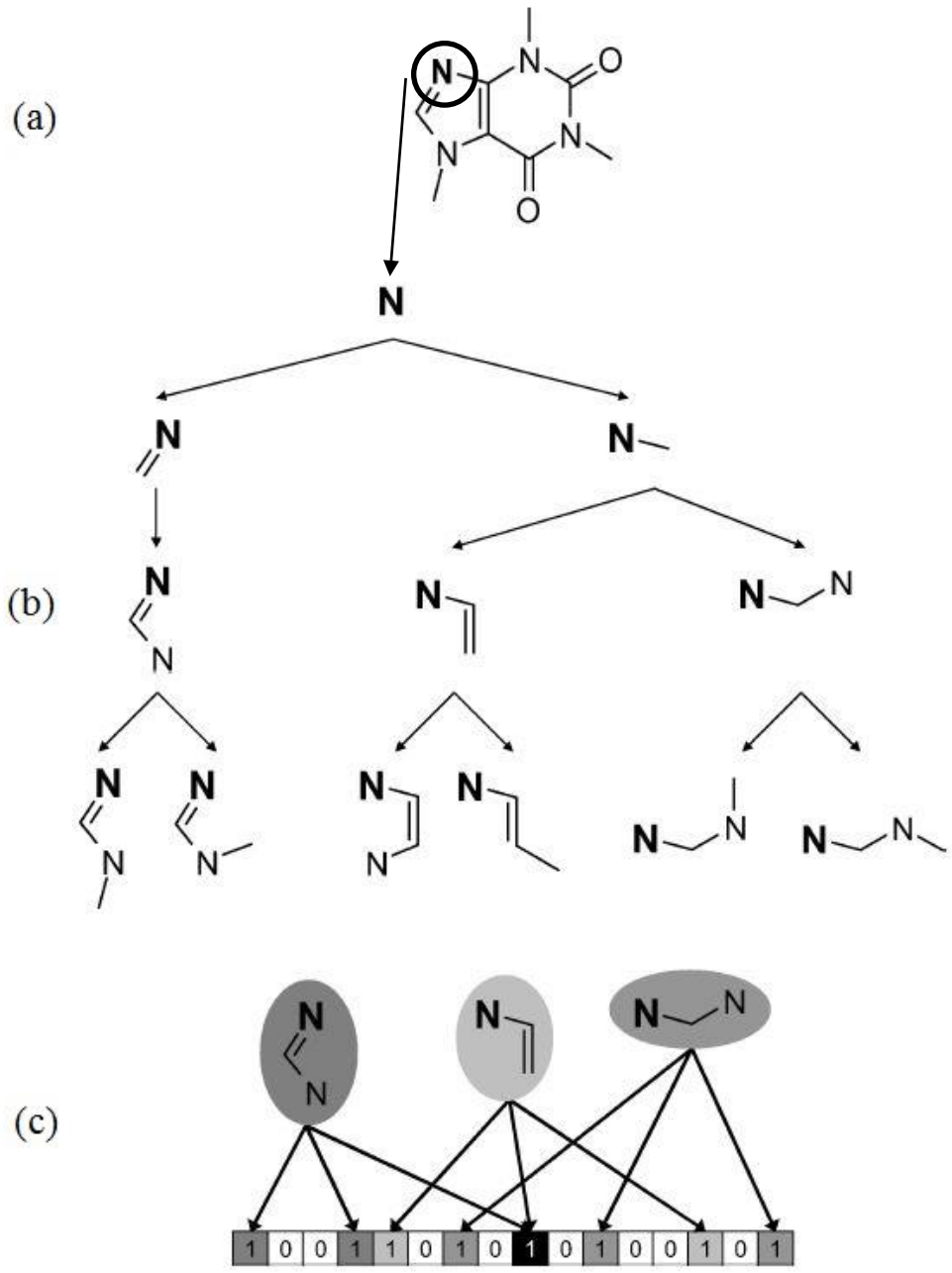


Figure 9: The structuring on a Hash-Key fingerprint (Brown 2009)

Figure 9 demonstrates an example of a hash-key fingerprint produced for the caffeine molecule. The Nitrogen atom circled (section a) is considered as the starting point for the generating the fingerprint. In this figure a path of up to three bonds has been encoded resulting in the binary fragment. Each of the chosen paths is converted into integer values using a random number generator to result in n bit positions (in this case 3). One can also see a case of a bit collision in Figure 9 (section c).

Hash-Key fingerprints fall into the information-based descriptors category as they are highly effective in encapsulation molecular information (Brown 2009; Willet 2009).

Structure-key based and hash-key based fingerprints have proven to be very effective in similarity studies. However they both suffer from some limitation courtesy of the characteristics of knowledge-based and information-based methods. When using the structure-key fingerprints one must be aware of the fact that due to the definition of the dictionary of sub-structures being fixed, the encoding process might fail to find some of the features in the molecules being encoded. Using this method some molecules may produce fingerprints that contain little or no information in them due to their sub-structures not occurring in the dictionary. This should be considered when applying the method to novel chemical classes (Brown 2009; Willet 2009). Hash-key fingerprints do not suffer from this limitation since the information already present in the molecule being encoded is used. Unfortunately there is a lot of assumption involved in the making of the structural keys due to the idea of pre-defined patterns. As mentioned above this method is partially dependent on the chemists' intuition and the results from mining drug-like databases. The patterns included in the generated structural key is crucial in the effectiveness of the search, as a bad choice can lead to many false hits and a very slow search (DAYLIGHT Chemical Information Systems 2008; Willet 2014).

Hash-key fingerprints are quick to calculate and are very effective in many applications in Chemoinformatics since they encapsulate vast amounts of information. Because they are not dependent on a dictionary, every fragment in the molecule will be encoded. This feature however prevents mapping between bits and 'unique' sub-structure fragments (Leach & Gillet 2007), therefore hash-key fingerprints are not readily interpretable and the resultant descriptors can be highly redundant (Brown 2009). On the other hand hash-key fingerprints are very difficult, almost hard to interpret since there is no direct mapping between the indices in the bit-strings and the features. Structure-key fingerprints have the advantage of having the pre-defined dictionary as reference. The fact that hash-key fingerprints describe atoms in terms of their associated properties allows them to be used in similarity searching to retrieve molecules that have similar properties to the query structure but contain different atoms. This permits the identification of new classes of molecules with the necessary bioactivities (Willet 2009).

2.2.8. 3D Molecular Descriptors

3D descriptors are more complex since they need to take into account that many molecules are “conformationally” flexible. This topic is out of scope for this research therefore we shall not describe it further.

2.2.9. Similarity and Dis-similarity Coefficients

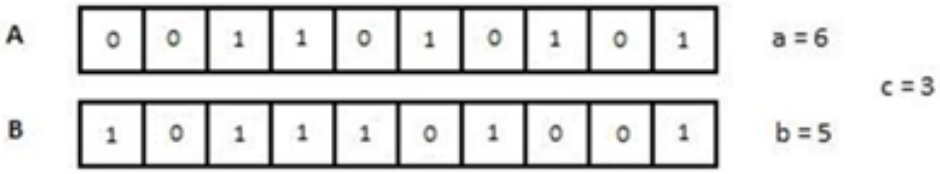
Some coefficients are measures of similarity (Dice and Tanimoto) and some other are measures of distance or dissimilarity (Hamming and Euclidian). Normalised similarity measures range between zero and one, with one indicating a full match and zero indicating no similarity. Dissimilarity measures can range between zero and a maximum value (N). With these measures zero means that there is a match (Willett et al. 1998; Leach & Gillet 2007). Similarity and dis-similarity measures can be normalised so that the output values fall in the range [0-1]. Such values allow for the inter-conversion between a similarity coefficient and its complementary dissimilarity coefficient so that: Distance = 1 – Similarity. This is called the ‘Zero-to-Unity’ or ‘Subtraction from Unity’ (Willett et al. 1998).

The most commonly used similarity methods are based on 2D fingerprints. The similarity between two molecules described by binary fingerprints is usually represented by the popular Tanimoto coefficient. This gives a measure of the number of bits that the two molecules have in common. Note that only the bits set to one (ON bits) determine similarity, not the ones set to zero (OFF bits). Tanimoto coefficient is popular for a number of reasons; it can be used to measure similarity between molecules represented by binary (dichotomous) fingerprints as well as continuous data i.e. Topological Indices (Leach & Gillet 2007; Bajorath 2011), the calculation (see formula in Figure 10) does not involve square roots therefore making it faster (Willett et al. 1998) and it still remains a yardstick against which alternative methods are judged despite the years that have passed since the study was initially done by Willet and Winterman in 1996.

An important fact to be aware of is that similarity coefficients (such as Tanimoto) depend on the number of bits two molecules have in common. Contrariwise in distance coefficients the common absence of features is regarded as similarity (Leach & Gillet 2007). Previous work done has shown that as a result smaller molecules tend to have lower similarity measures than larger ones because

they have fewer bits set to one in common with the target (Willett 2006; Leach & Gillet 2007). Tanimoto coefficient includes a degree of size normalisation via the denominator term. This helps reduce the bias towards the larger molecules which have more bits set to one compared to smaller ones. Figure 10 demonstrates an example of two binary fingerprint fragments and the similarity and dis-similarity between them is calculated.

In Figure 10, we see two fragments of fingerprint for two molecules that are being compared for similarity and dissimilarity. Note that the measures used (Tanimoto and Euclidean are two different measures and not complimentary so the Zero to Unity concept does not apply). In the figure, "a" is the number of bits set to one in fragment A and "b" is the number of bits set to 1 in fragment B. "c" is the number of bits set to one and common (set to one in the same place) between both fragments. As mentioned the Tanimoto coefficient produces values between zero and one. This can be interpreted as follows: a value of zero means the molecules have no fragments in common therefore no similarity and a value of one means unity and therefore the molecules are identical. The closer the number to one means the more similar the two compared molecules are.



Coefficient	**formula**	**Result**
Similarity (Tanimoto)	$\frac{c}{a+b-c}$	0.375
Dis-similarity (Euclidian)	$\sqrt{a+b-2c}$	2.236

Figure 10: Example of two fingerprints and the similarity and distance coefficient calculated.

2.3. High-Throughput Screening

In drug discovery compounds with unknown biological activity are screened against specific target(s) to determine if they interact with the target(s) in a productive way; that is showing binding activity. Compounds which are active against a target pass the first test on the way to becoming a drug, the ones that fail this test are sent back to the compound (screening) library to be screened later against other targets. Screening compounds against targets has been an on-going activity in the pharmaceutical industry. The process of discovering a new drug normally involves High-Throughput Screening (HTS). In HTS groups of compounds are screened against a target to assess their ability to bind to the target. Advances in molecular biology and human genetics produce increasing number of molecular targets. This is combined with increases in compound collection generated by combinatorial technologies has resulted in huge libraries of compounds ready to be screened against targets. In such cases conventional screening methods are not feasible.

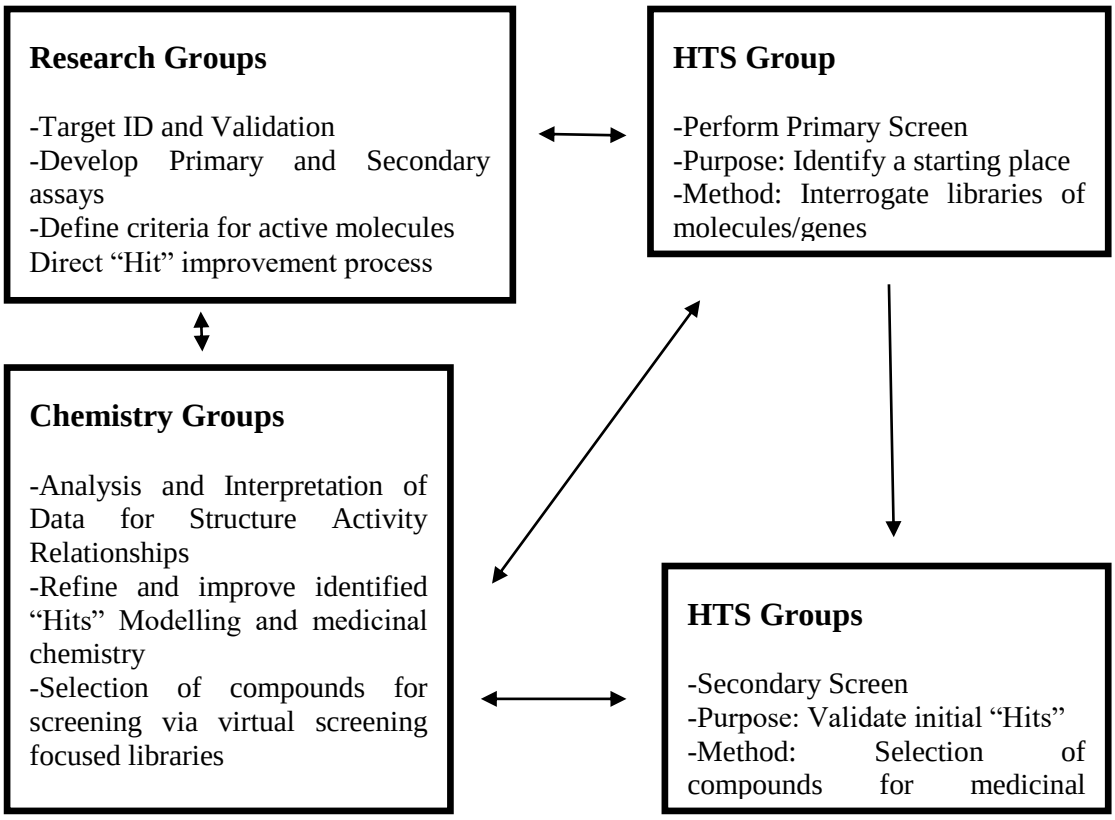


Figure 11: Iterative process during HTS between various research groups (Stephan & Gilbertson 2009)

HTS allows the researcher to screen hundreds of thousands of compounds against a target in a very short time. If the compound binds to the target then it becomes a Hit. If the hit is open to Medicinal Chemistry optimisation and is proven to be non-toxic

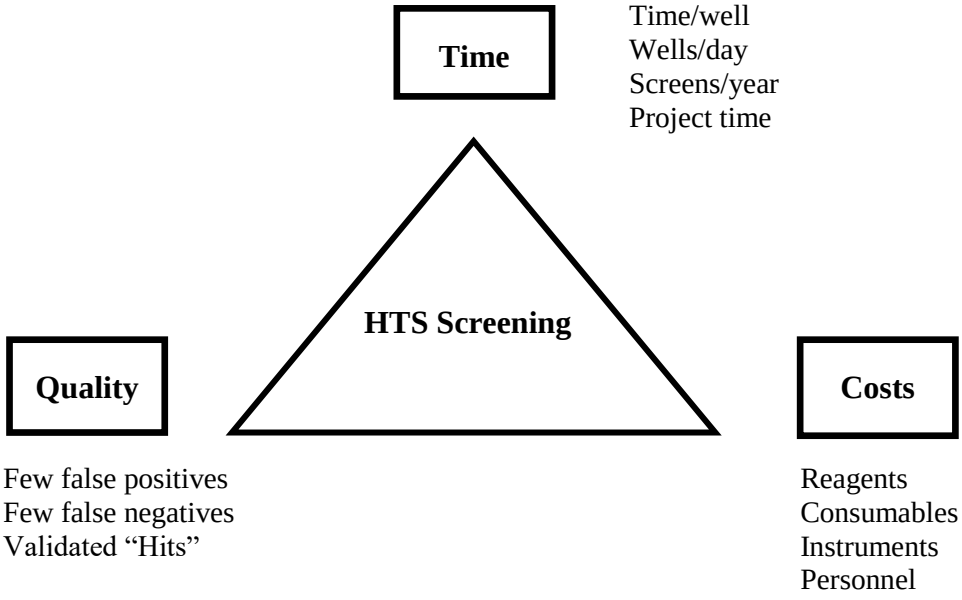


Figure 12: Showing the key factors towards a successful HTS process (Stephan & Gilbertson 2009)

in pre-clinical trials, then it becomes a Lead for a specific target (Schierz 2009). However with the increase in the size of compound libraries, the disadvantages of HTS increase too; the quality of the library, sequence miss-readings or reproducibility of assay protocol can result in incompleteness of the screening (Kato et al. 2005). As found by Schierz (2009) there is a lack of publicly available bioassay (a bioassay involves the use of tissue or cell in order to determine the biological activity of a substance) data due to HTS technology being kept at private commercial organisations and the data from freely available resources (PubChem) is not curated and potentially erroneous.

2.4. Virtual Screening

Leach and Gillet (2007) define Virtual Screening as "the in-silico screening of biological compounds". The goal is to score, rank and / or filter a set of structures using one or more computational procedures. Virtual screening complements the HTS process by helping with the selection of compounds to be screened (Willett 2006; Schierz 2009).

Virtual screening utilises an array of computational techniques for the selection and prioritisation of those molecules that may have the probability of being active for a target (Willett 2006). This is done based on the type and the amount of information available about the compounds and the target (Leach & Gillet 2007; Schierz 2009).

Wilton et al. (2003) identified four main classes of virtual screening:

- If only a single active molecule is known for a target, then similarity searching can be done where the database is ranked in decreasing order of similarity to the known structure.
- If several molecules are known to be active for a target, then Pharmacophore (Specific 3D arrangement of chemical groups common to active molecules and essential to their biological activity) mapping can be done to determine common features responsible for activity, with later a 3D sub-structure database search to find other molecules with the pharmacophore.
- If a reasonable amount of active and non-active molecules are known, then the active ones can be used as training material to build predictive models which discriminate between active and non-active compounds. Goal is to apply the models to unscreened molecules to select ones that are most likely to be active.
- If the 3D structure of the target is known, then a docking study can be carried out where candidates are docked into the binding site of the target and a scoring function is applied to estimate the likelihood of binding with high attrition.

Willet (2006) categorises these classes as two main types:

- Structure-based approaches: such as docking.
- Ligand-based approaches: such as pharmacophore methods, machine learning methods and similarity searching.

A schematic illustration of a typical virtual screening flowchart is shown in Figure 13. As seen in the figure, many virtual screening processes involve a sequence of methodologies.

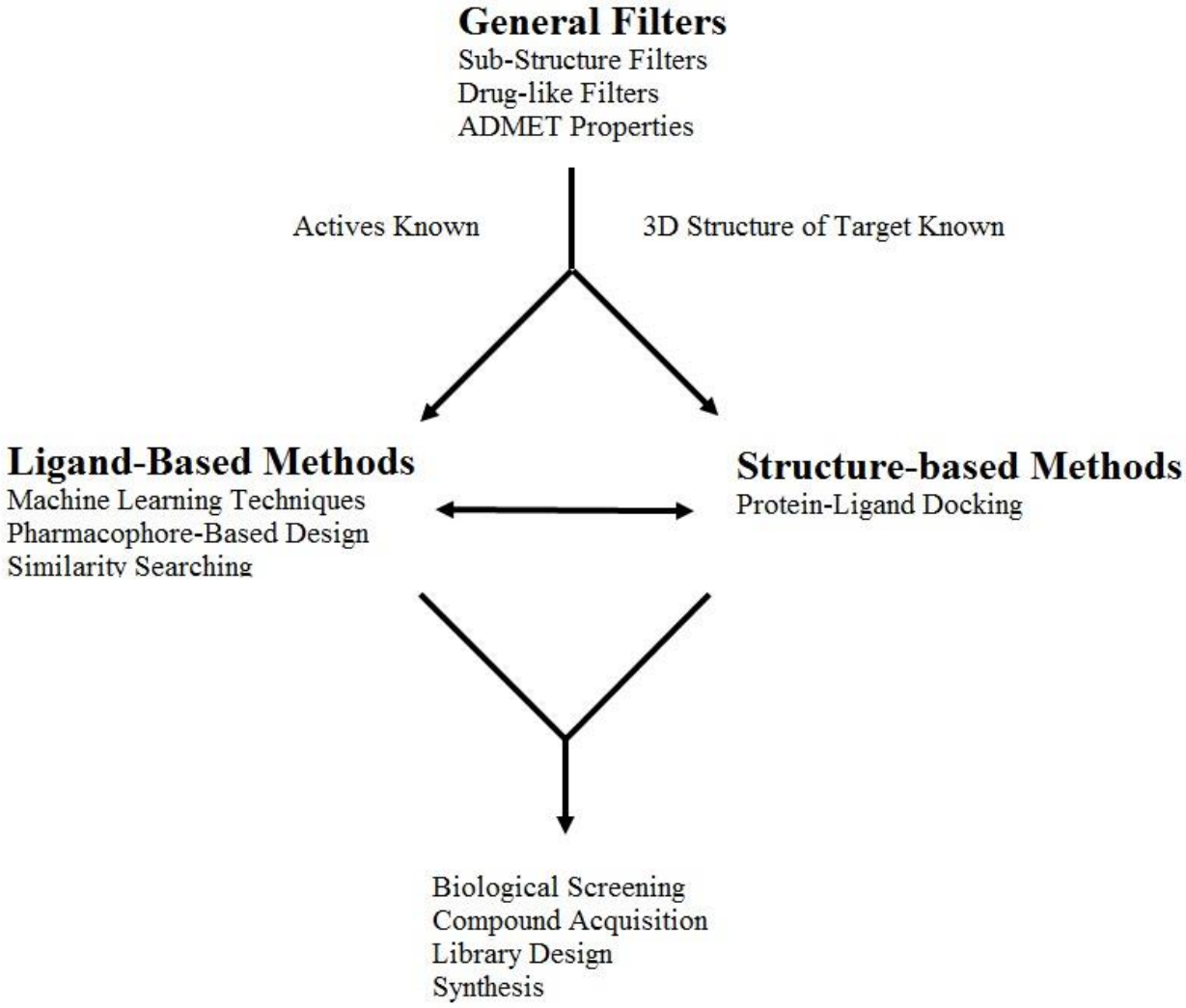


Figure 13: A schematic illustration of a typical virtual screening flowchart (Leach & Gillet 2007)

In both cases, handling large datasets is a major challenge and requires specialized methodology discussed in the next subsection.

2.5. Handling the Mining of Large Datasets

Big data also referred to as massive data has been said to be one of the major challenges of the current era (Kahng 2012; Anand 2013; Hammer et al. 2013; Cuzzocrea 2014). One might ask what can be referred to as big data. The answer can be viewed from different angles. For example the number of data points, the

dimensionality or the complexity of the data at hand. Douglas Laney (2001) pointed out the characteristics of big data as follows:

- Volume: this refers to the size of the datasets, which can be caused by the number of data points or its dimensionality or both.
- Velocity: this refers to the speed of data accumulation, the need for rapid model adaptation and lifelong learning.
- Variety: this refers to heterogeneous data formats caused by distributed data sources, different representation technologies, multiple sensors, etc.
- Veracity: this refers to the fact that data quality can vary significantly for big data sources and that manual curation is almost impossible.

Recent advances in computing allow for the collection and storage of inconceivable amounts of data, leading to the creation of very large datasets in data repositories (Kumar et al. 2006). Scientists can now predict the properties of chemical compounds which have not yet been synthesised. Methods such as Virtual Screening take advantage of data mining techniques in order to make hypotheses based on many observations. Important decisions can be made based on this information-rich data. However the fast-growing amount of data has far exceeded the human ability to analyse and comprehend it without powerful tools (J. Han & Kamber 2001).

Data mining tools analyse the data stored in a database and unravel hidden data patterns which can contribute to business strategies and scientific researches. One problem that may arise is the ability to analyse the vast amount of information hidden in large datasets. Developing powerful computers is costly and it is easy to build datasets which are too big for even the most powerful computers. Some strategies which are commonly used to deal with large datasets are listed below. In this section we only highlight these methods but are not going into detail about them. This section mostly emphasises the physical aspect of the data (i.e. where it is stored).

Data can be stored centrally or distributed. The various distributed data mining systems differ in several ways (Grossman et al. 1999). Data Strategy**:** the decision to move the data (centralised-learning), the intermediate results, the predictive models (local-learning) or the final results of a data mining algorithm, Task Strategy: The

decision to apply data mining algorithms independently at each site or coordinate the tasks within an algorithm over several sites and Model Strategy: The decision of choosing a method to combine the models built at sites.

There are various infrastructures (methods) which assist the mining of large distributed datasets, such as Cluster-Computing, Grid-Computing and Cloud-Computing. The goal of clustering is to partition a set of patterns into disjoint and homogeneous clusters. Clusters offer two main roles which satisfy the two main steps every data mining process involves; data clusters provide storage and data management services for the datasets being mined and compute clusters provide the services needed for data cleansing, preparation and data mining tasks.

In Grid-Computing, several machines work together by linking through a network to execute a common task (Naqaash et al. 2010). The desire for sharing high-performance computing resources amongst researchers led to the development of Grid-Computing technology and some of its infrastructure (Abbas 2003). Grid-Computing has been distinguished from conventional distributed computing by its focus on large-scale resource sharing and high-performance orientation (Cannataro et al. 2004).

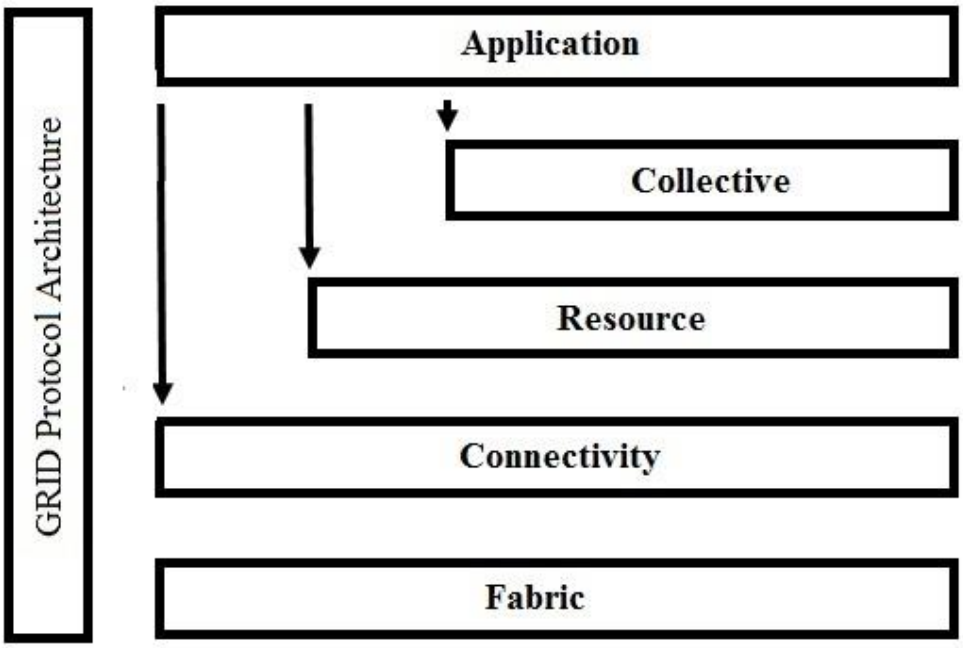


Figure 14: Typical Grid protocol computing architecture (Foster et al. 2008)

The grid architecture consists of a few layers (Please see Figure 14). The fabric layer provides access to different resources such as compute, storage and network. The connectivity layer defines the core communication and authentication protocols for easy and secure transactions. The resource layer defines protocols for publication, discovery, negotiation, monitoring, accounting and payment of sharing operations on individual resources. The collective layer captures the interactions across a collection of resources such as Monitoring and Discovery Services. The application layer comprises of the user applications built on top of the other protocols and operate in the virtual organisation environments. Each virtual organisation can consist of either physically distributed institutions or logically related projects (Foster et al. 2008).

Cloud in computing terms means an infrastructure that provides resources and / or services over the internet (Grossman & Gu 2008). Cloud computing refers to the applications delivered as services over the internet and the hardware and software in the data centres providing the services (Armbrust et al. 2010). Cloud computing has some benefits such as easy installation, centralised control and maintenance and safety. But it also suffers from disadvantages such as data lock-in (proprietary API), difficulty of a scalable storage and bugs in large-scale distribution systems such as not often being able to reproduce errors in larger configurations in smaller environments.

Cloud computing can be viewed as a collection of services which can be presented as a layered cloud computing architecture as seen in Figure 15. Clouds in general provide service at three levels. The “infrastructure as a Service” layer provides hardware, software and equipment to deliver software application environments. The “Platform as a Service” layer offers a high-level integrated environment to build, test, and deploy custom applications. The “Software as a Service” delivers special-purpose software that is remotely accessible by consumers through the internet (Foster et al. 2008). In addition to these difficulties in database storage and retrieval, a major challenge in analysing complex data of chemical compounds characterised by hundreds of variables is the so-called heavily imbalanced data scenario which will be discussed next.

Figure 15: Typical Cloud computing architecture (Foster et al. 2008)

2.6. Summary of challenges in this chapter

In this chapter we discussed how chemical molecules are shown and introduced to the computer in order to be investigated, manipulated and studied for Chemoinformatics purposes. We saw the different notations that can be used to present molecules to the computer. As a result molecules can be stored in databases. Databases can be sorted based on the needs or based on the molecules stored in them and to be able to search them efficiently different methods were devised; structure and sub-structure searching. Both methods have advantages in that they are fast and precise, however the preciseness of the methods requires their queries to be very specific and the slightest mistake could lead to no hits or too general queries could return too many results.

Similarity searching was devised as an alternative and this method would calculate the similarity between two or more molecules. We touched on molecular representation which characterises the molecules being investigated. In order to asses similarity between molecules there are metrics defined.

We discussed high-throughput screening which screens millions of molecules against specific targets and assesses their affinity to the target. Virtual screening is the more feasible, computer-based version of HTS which allows the quicker selection and prioritisation of those molecules that may have the probability of being active for a target.

With chemical datasets the libraries hold millions of unknown molecules ready to be screened. Hundreds of new molecules are added to these libraries regularly. When selecting the molecules for screening, this could result in datasets that span over hundreds or thousands of molecule samples and once some features are generated for these samples one could be faced with problem of handling large datasets. In the final part of the chapter we discussed some methods that could be utilised to handle this phenomenon.

In the next chapter we shall be introducing the datasets chosen for this study. Afterwards we will be coming back to the topic of data representation and we will be introducing the reader to the various molecular representation techniques we will be utilising to represent our datasets.

3. Datasets Description and Pre-Processing Strategies

Predictive modelling (the process that uses data mining and probability to forecast outcomes) is a data analysis task where the goal is to build a model of an unknown function $Y = f(X_1, X_2, \ldots, X_p)$ based on a training sample $\{<\mathbf{x}i, yi>\}_{i=1}^{n}$ with examples of this function. The type of the variable Y determines whether the task at hand is classification or regression. For some applications, it is of utter importance that the obtained models are accurate at some sub-range of the domain of the target variable (Branco et al. 2016). As an example one can refer to the literature and observe that this problem is faced in different application areas such as credit card fraud detection (Yang & Wu 2006; Dal Pozzolo et al. 2014), detection of oil spill from satellite images (Chi et al. 2014) and medical diagnostic imaging (Mazurowski et al. 2008). These are only a few prominent examples of a phenomenon which has put imbalanced data learning in the top 10 challenges of data mining (Bekkar & Alitouche 2013). Frequently, the sub-ranges of the target variable are poorly represented in the available training sample. In these cases we face the phenomenon called data imbalance. Data imbalance occurs when the cases that are more important for the user are rare and few of them exist in the training set. The main challenge here, which is often the case with real-world datasets, is that the class with the lower number of instances is precisely the more useful class and misclassifying this class can often be costly

Technically, every dataset that has non-balanced / unequal distribution between classes is considered imbalanced. Chawla (2005) mentions, a dataset is imbalanced if the classification categories are not approximately equally represented. The common understanding is that imbalanced datasets correspond to the ones exhibiting extreme imbalances such as 1:100, 1:1000 and 1:10000 active to non-active samples respectively (Chawla et al. 2004; He & Garcia, 2009; Ganganwar 2012; Maldonado et al. 2014). Further elaboration has been done on the topic of data imbalance in section 4.1. The reader is reminded here that in this study we work with binary datasets, therefore the classes are referred to as 0 and 1, with class 1 being the minority class (class of interest).

As indicated in the introduction, the general goals of this study is to devise an effective approach that should be ubiquitously applied regardless the dataset characteristics; since in real life screening applications the imbalance ration is often not known beforehand. As an example one can refer to the number of fraudulent transactions in comparison to honest ones (Chawla et al. 2004; Longadge & Dongre 2013). Hence, the datasets chosen for this study have been selected because they represent a wide range of scenarios; comprising the whole spectrum of typical challenges in virtual screening.

In order to overcome the effects of data imbalance in our datasets, SMOTE (Synthetic Minority Over-sampling TEchnique), the data pre-processing technique (Chawla 2002) was employed to re-establish balance of classes. In this approach, the minority class is over-sampled by creating synthetic examples rather than by over-sampling with replacement. Here the synthetic data are generated by operating in the feature space rather than the data space. In a nutshell, the generation of new synthetic samples by SMOTE is as follows: The difference between the feature vector under consideration and its nearest neighbour is taken and this number is multiplied by a random number between 0 and 1. The resulting number is then added to the feature vector under consideration. This action causes the selection of a random point along the segment line between two specific points (Pears et al. 2014). The default implementation uses five nearest neighbours (Chawla et al. 2002; Oreski & Oreski, 2014). The details of how this method generates the synthetic samples are further introduced in section 4.2 and in Figure 19.

Three of the datasets, Formylpeptide Receptor Ligand Binding Assay, VCAM-1 Imaging Assay in Pooled HUVECs (National Centre for Biotechnology Information) and the Mutagenicity Dataset (Kazius et al. 2005) were downloaded from PubChem Open Chemistry Database (Wang 2009). The other one is the Factor XA Dataset (Fontaine et al. 2005) downloaded from the chemoinformatic.org database. The two first datasets are noisy, highly imbalanced datasets. The Mutagenicity dataset is the rather balanced dataset and in the Factor XA the number of instances from the class of interest exceeds the number of the other class, making it an imbalanced dataset.

This chapter begins with a detailed description of the datasets gathered for this study in section 3.1. The chapter continues with section 3.2 wherein the various

methods used to manipulate the data from its original form into a more useable form are explained.

3.1. Background

In this section we shall introduce the reader to our selection of the datasets used for this study. The datasets vary in the number of instances and their imbalance ratio. This ratio is defined as the ratio of the number of instances of the majority class to the number of the examples in the minority class (García et al. 2008; López et al. 2013). For each dataset there exists a summary table breaking the dataset down by its instances and classes.

Formylpeptide Receptor Ligand Binding Assay (AID362)

This dataset is a whole-cell assay for another inhibitor of peptide binding associated with tissue-damaging chronic inflammation (Jabed et al. 2015). On PubChem this dataset has been described as the formylpeptide receptor (FPR) family of G-protein coupled receptors (GPCR) which contributes to the localization and activation of tissue-damaging leukocytes at sites of chronic inflammation. The number of instances, active and inactive and the imbalance ratio information can be found in Table 1. The dataset is a highly imbalanced one, with an imbalance ratio of 1.4%.

Dataset	#Total Instances	#Active Instance (class '1')	#Inactive Instance (class '0')	Active/Inactive Ratio
AID362	4279	60	4219	0.0142

Table 1: AID362 specifications. Class of interest has a 1 next to the label

VCAM-1 Imaging Assay in Pooled HUVECs (AID456)

The description on PubChem describes this dataset as follows: VCAM-1 (vascular cell adhesion molecule-1) mRNA and protein levels are potently induced by pro-inflammatory agents (TNFa, IL-1) resulting in enhanced VCAM-1 surface expression in HUVECs (human umbilical vein endothelial cells). The information relating to the number of instances for each class in this dataset is included in Table 2. This dataset is extremely imbalanced and hence is particularly challenging. It has a very large imbalance ratio and has a rather low number of instances of the class of interest compared to the majority class.

Dataset	#Total Instances	#Active Instance (class '1')	#Inactive Instance (class '0')	Active/Inactive Ratio
AID456	9982	27	9955	0.0027

Table 2: AID456 specification. Class of interest has a 1 next to its label

Mutagenicity Dataset (Bursi)

The dataset was prepared by Bursi and co-workers (Kazius et al. 2005). It contains 4337 diverse organic molecules. Of this number, 2401 were mutagens and 1936 were non-mutagens. A mutagen is a physical or chemical agent that changes the genetic material, usually the DNA of an organism therefore causing increased frequencies of mutations. They used this dataset to identify sub-structures (called toxicophors) which could help classify whether test molecules were mutagenic (Langham & Jain 2008).

At the time of performing the experiments for this study, the original Bursi dataset was not available to download therefore with the help of the Entrez system available from the National Centre for Biotechnology Information (NCBI), it was downloaded from PubChem. Entrez is the retrieval tool which allows the retrieval of set of sequences based on various descriptor fields such as source organisms, accession numbers, etc. Table 3 contains information about the number of instances for this dataset.

Dataset	#Total Instances	#Active Instance (class '1')	#Inactive Instance (class '0')	Active/Inactive Ratio
Bursi	4893	2556	2337	1.09

Table 3: Mutagenicity dataset specification.

Factor XA Dataset (Fontaine)

A drug can be classified by the chemical type of its active ingredient or by how it is used to treat a condition, resulting in a drug being classified into one or more classes. Factor XA inhibitors are anticoagulants (an agent that is used to prevent the formation of blood clots). They block the activity of the clotting Factor XA and prevent blood clots from developing or getting worse. This is especially useful in the case of people receiving organ transplants or knee and hip replacement surgeries in order to prevent blood clots from forming and leading to deep vein thrombosis and pulmonary embolism.

The data in this dataset were used to discriminate between Factor XA inhibitors of high and low activity. Since the dataset includes molecules from diverse chemical classes, the objective in the main study by Fontaine et al. (2005) was to produce a discriminant model which is potentially useful for screening molecular libraries. Table 4 contains details about the number of instances and imbalance ratio for Factor XA dataset.

Dataset	**#Total Instances**	**#Active Instance (class '1')**	**#Inactive Instance (class '0')**	**Active/Inactive Ratio**
Fontaine	435	279	156	1.79

Table 4: Factor XA dataset specification.

3.2. Data Preparation

The transformations which prepare the data for further analysis are part of data pre-processing. Some examples of the activities are normalisation and filtering. Data made available on the public domain does not always contain correct values, therefore if any incorrect inputs, out of range and missing values they need to be corrected. This is the most time-consuming activity in the pre-processing phase.

Throughout the years attempts have been made to create a unified and standard format for chemical data most notably the Chemical Markup Language (Murray-Rust et al. 2001; Spjuth et al. 2010), a dialect of XML. Such formats are yet to become widespread standard due to different application areas for chemistry, difference in the data stored by different formats, competition between software and lack of vendor-neutral formats (O'Boyle et al. 2011).

Chemical datasets available to download are normally stored in online repositories by depositors in various formats such as sdf (structure-data file), smi (SMILES format) and MOL. In order to perform similarity searching these formats need to be translated into structural properties. Open-source as well as proprietary software are available online to perform the necessary transformations. Some examples of such software are described next.

PaDel

A molecular descriptor is the product of logical and mathematical procedures which transform the chemical information encoded in the symbolic representation of a chemical molecule into a useful number or the result of some standardised

experiment (Todeschini & Consonni 2009). The molecular descriptors are calculated for the chemical molecule in order to develop a quantitative Structure Activity Relationship (QSAR) for predicting the activity of novel molecules. Currently there are a number of freely available software for calculating molecular descriptors.

Some of the characteristics a good molecular descriptor calculator should have are (Yap 2011):

- Free or cheap to purchase for easy access for researchers
- Open source so researchers can add their own libraries / algorithms to them
- Having a graphical user interface and command line interface
- Able to install and operate on multiple platforms
- Able to accept different molecular formats
- Able to calculate many molecular descriptors

A software tool which possesses most of the above mentioned characteristics is PaDel by Yap (2011). It produces molecular fingerprints from information encoded in symbolic chemical representations such as connection tables. The result can be described as matrix where the compounds are placed on the rows and the structural properties are on the columns (Huang et al 2015). The cells in between indicate the presence or absence of the structural properties by 1 or 0 respectively.

Other features include having a graphical user interface, platform independence, accepting multiple file formats and producing several molecular fingerprints. Some of these fingerprints are available in the Chemistry Development Kit (Steinbeck et al 2003) library. Some of these fingerprints have been used to produce descriptors for our datasets, therefore we shall describe the fingerprints further in the chapter. In addition to structural descriptors, PaDel has the ability to calculate 2D and 3D descriptors, which unlike their structural counterparts that have binary outcomes, have positive or negative numerical values.

PowerMV

PowerMV (Liu et al. 2005) is a software designed for statistical analysis, molecular viewing, descriptor generation and similarity search. Its environment allows for the viewing of the compound structure in 2D and 3D. This software calculates six molecular descriptors describing properties of the compound. It

produces four bit-string (binary) and two continuous descriptors. In the binary descriptors a bit is set to 1 if a certain feature is present and zero if absent. Continuous descriptors are used for searching the nearest neighbours. Bit-string descriptors use the Tanimoto (Jaccard 1901) coefficient is and continuous descriptors use the Euclidean distance.

With PowerMV one can:

- Import, view and sort files in the .sdf format.
- The software automatically generates descriptors for the input molecules and save the descriptors, attributes and chemical structures. This will become an annotated database for similarity searching that users can save and view.
- Searching is really fast as the descriptors for the candidate databases are pre-computed so for a search, only the descriptors of the target molecule need to be calculated. The databases are stored using an index-based file format which leads to faster searching.

OpenBabel

As mentioned above, in the introductory chapter, due to there being no standard format for storing chemical data, a noticeable problem in computational modelling is the conversion of molecular structures from one format to another. This process involves the extraction and the interpretation of the chemical data and the semantics of molecular structures.

The OpenBabel project, is a full-featured open chemical toolbox, designed specifically to speak to many representations of chemical data. It allows one to search for, convert, analyse and store data from molecular modelling, chemistry, biochemistry or related areas. It also provides a complete and extensible development toolkit for developers to develop Chemoinformatics software (O'Boyle et al. 2011).

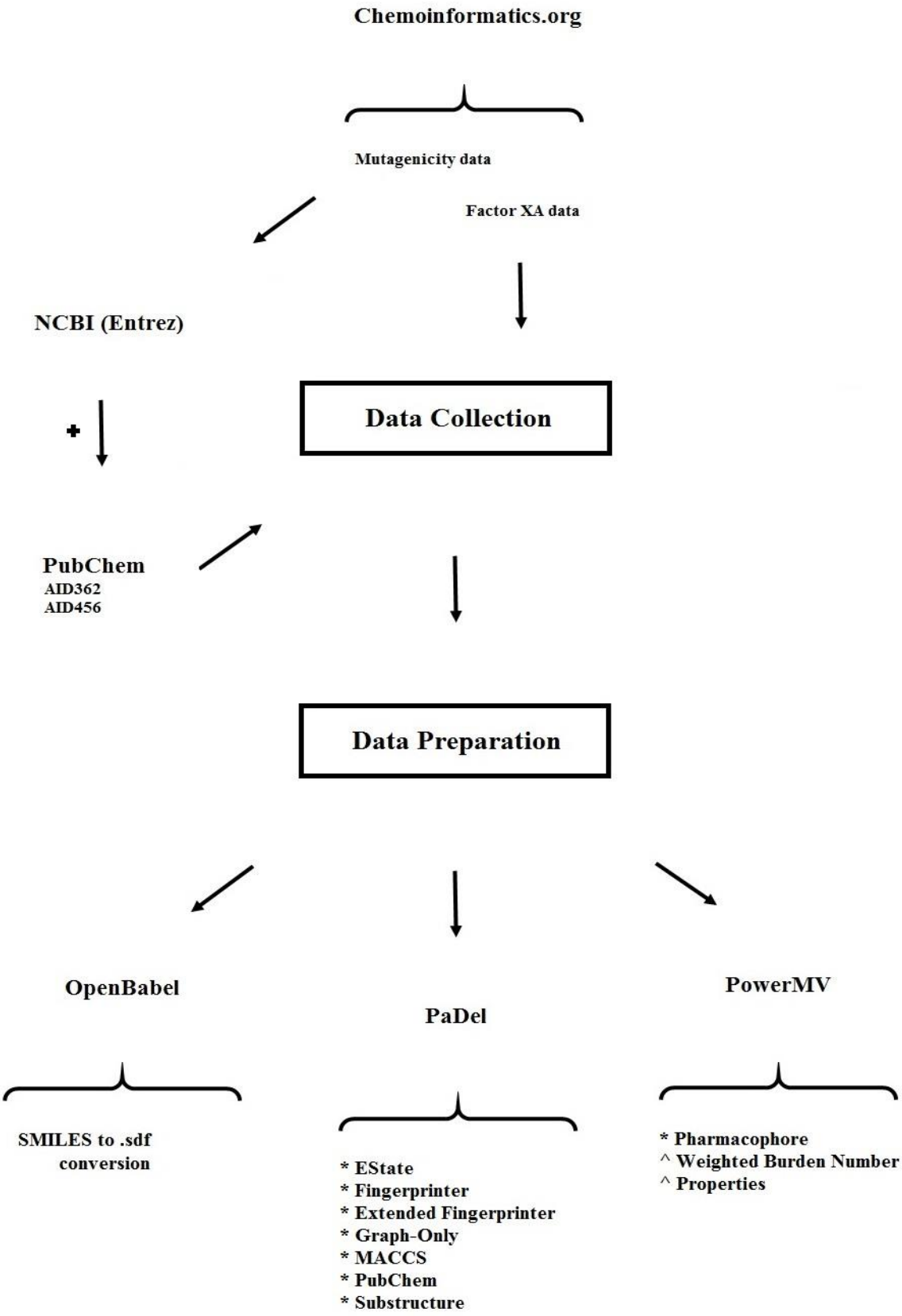


Figure 16: Schematic overview of chapter 3.

Fingerprints marked with * are bit-string fingerprints and ones marked with ^ are continuous (numeric) fingerprints**.**

Figure 16 illustrates (schematically) the process of gathering data from different sources and the preparation done in order to make the data ready for further manipulations by the various methods acquired in this study which will be discussed in detail in chapter 4.

PubChem

The National Institutes of Health (NIH) launched the Molecular Libraries Initiative (MLI) in 2004 which set out to provide academic researchers with the tools to explore potential starting points for drug discovery. At the heart of MLI is PubChem. PubChem is an online public repository for biological properties of small molecules hosted by the US National Institutes of Health (Wang et al. 2009). It comprises of three inter-linked databases; substance, compound and bioassay. The substance database contains chemical information deposited by individual contributors to the PubChem. The compound database has the unique chemical structures extracted from the substance database (Kim et al. 2015).

PubChem contains (as mentioned above) compound information from the scientific literature, but it is considered a data repository and no special effort is dedicated to the curation of the information deposited by various contributors (Fourches et al. 2011). Professor Alexander Tropsha, director of Exploratory Center for Chemoinformatics Research at North Carolina University states that PubChem does not curate the data as deposited by screening centres (Bradley 2008; Schierz 2009). The deposited data are not curated by the contributors (PubChem; Go 2010). The datasets deposited in PubChem are highly imbalanced with a ratio of active to inactive compounds on average of 1:1000 (Bradley 2008).

The bioassay database (where two of the datasets for this study, AID362 and AID456, were acquired from) is intended for archiving the biological tests of small molecules generated by High-Throughput Screening experiments, medicinal chemistry studies and drug discovery programs. PubChem aims at providing this information free to the research community (Wang et al. 2014). PubChem bioassay database is integrated with the National Centre for Biotechnology Information, making it even easier to search by Entrez queries.

Molecular Fingerprints

Molecular fingerprinting is nowadays an essential tool for determining molecular similarity. By allowing the addition of different fingerprinting methods, the user is given the choice and freedom to utilise the best method for their case. Below we shall give a description of the different fingerprinting techniques used in this study.

Fingerprinter (Fin)

The Fingerprinter class from CDK (refer to section 3.2 under the PaDel sub-section) produces Daylight-type fingerprints (James et al. 2000). This class works by searching the molecule, starts at each atom in it and creates string representations of the paths up to the length of six atoms. It works very much like the Hash-Key fingerprints (refer to section 2.2.7. and Figure 7). Based on all the paths computed from a molecule, a molecular fingerprint is obtained. The fingerprinter class assumes that the hydrogens are explicitly given. This class generates 1024 bits.

Extended Fingerprinter (Ext)

The Extended Fingerprinter class is also from the CDK and it extends the Fingerprinter class by including additional bits describing ring features. This class contains the information from the Fingerprinter class and bits which tell if the structure has 0 rings, 1 or less rings, 2 or less rings (this refers to the smallest set of smallest rings). There are also bits which indicate if there is a fused ring system with 1, 2,… 8 or more rings in it. The list of rings given by the specified bits must be the list of all rings in the molecule. The number of bits produced by this fingerprint is 1024.

Graph-Only Fingerprinter (Gra)

This class constructs a fingerprint generator which creates a specialised version of the Fingerprinter that does not take bond orders into account. This fingerprint produces 1024 bits.

EState (ESt)

The electro-topological state indices (EState) was introduced initially by Kier & Hall (1992). According to this paradigm, each atom in the molecular graph is represented by an EState variable. This variable encodes the essential electronic state

of the atom as affected by the electronic influence of all the other atoms in the molecule within the topological character of the molecule. Therefore the EState of an atom differs from molecule to molecule and depends on the detailed structure of the molecule (Hall & Kier 1995). EState indices encode important electronic and topological information and this enables them to show significant pharmacological information for database characterisation (Todeschini & Ringsted 2012).

MACCS Fingerprint (MAC)

The MACCS (Molecular Access System) fingerprint uses a set of structural features that is used to encode the molecule into a binary representation. The version of the MACCS fingerprint used in this study only has 166 bits. The fingerprint consists of a set of indicators showing whether each of these bits were present in a given molecule (Wei et al. 2007).

Pharmacophore Fingerprint (Pha)

The pharmacophore fingerprints (generated by PowerMV) are binary descriptors that are built to indicate the presence or absence of features based on bio-isosteric principles. According to this principle, two atoms or groups that have roughly the same biological effects are called bio-isosteres (Hughes-Oliver et al. 2011). There are a total of 147 bits generated by this fingerprint.

PubChem Fingerprint (Pub)

The PubChem system generates binary fingerprints for chemical structures. There are 881 bits in each fingerprint representing the Boolean determination of or test for an element count, atom pairing, a type of ring system, etc., in a molecule.

Substructure Fingerprint (Sub)

This fingerprint (Hert et al. 2009) contains the SMILES patterns for approximately 1000 chemical features such as common functional groups as classified by Christian Laggner or ring systems. This fingerprint contains 307 bits.

The 8 fingerprints mentioned above are the substructure fingerprints, all indicating the presence or absence of certain features in the encoded molecule, in the form of bit-strings.

In addition to these we have used continuous (numerical) fingerprints, Weighted Burden Number and Properties, variation on the original Burden Number by Burden (1989) and both generated by PowerMV.

Weighted Burden Number

This numerical fingerprint is achieved by placing one of the properties: electro-negativity, Gastgeiger partial charge or atomic lipophilicity on the diagonal of the Burden connectivity matrix, and weighting the off-diagonal elements by one of 2.5, 5.0, 7.5 or 10.0, twelve connectivity matrices are obtained. The largest and smallest eigenvalues are retained from each matrix resulting in 24 numerical descriptors (Liu et al. 2005; Hughes-Oliver et al. 2011).

Properties

These descriptors are useful for judging the drug-like nature of a molecule.

Dataset Structure

Each separate dataset is encoded by the 8 different substructure and the two numerical fingerprints. This will result in 32 substructure fingerprints and 8 numerical ones. The results are shown in Table 5.

Fingerprint	**# Bits**	**Abbreviation**	**Structural / Numeric**
CDK Fingerprinter	1024	Fin	Structural
CDK Extended Fingerprinter	1024	Ext	Structural
CDK Graph-Only	1024	Gra	Structural
CDK Substructure	307	Sub	Structural
CDK EState	79	ESt	Structural
MACCS Keys	166	MAC	Structural
PubChem	881	Pub	Structural
Pharmacophore	147	Pha	Structural
Weighted Burden Number	24	WBN	Numeric
Properties	8	---	Numeric

Table 5: Detailing the properties of the various fingerprints used

The prepared datasets have been introduced to the classifiers in two distinct formats:

- Structural-only: in this format the datasets are presented to the classifiers using structure-only fingerprints.
- Structure-Numerical: in this format, the numerical fingerprints have been amended to the structure-only fingerprints.

A schematic overview of the operations performed in order to prepare the datasets for the next stages has been illustrated in Figure 17. One can see in this figure how many features have been generated for the dataset by each fingerprint.

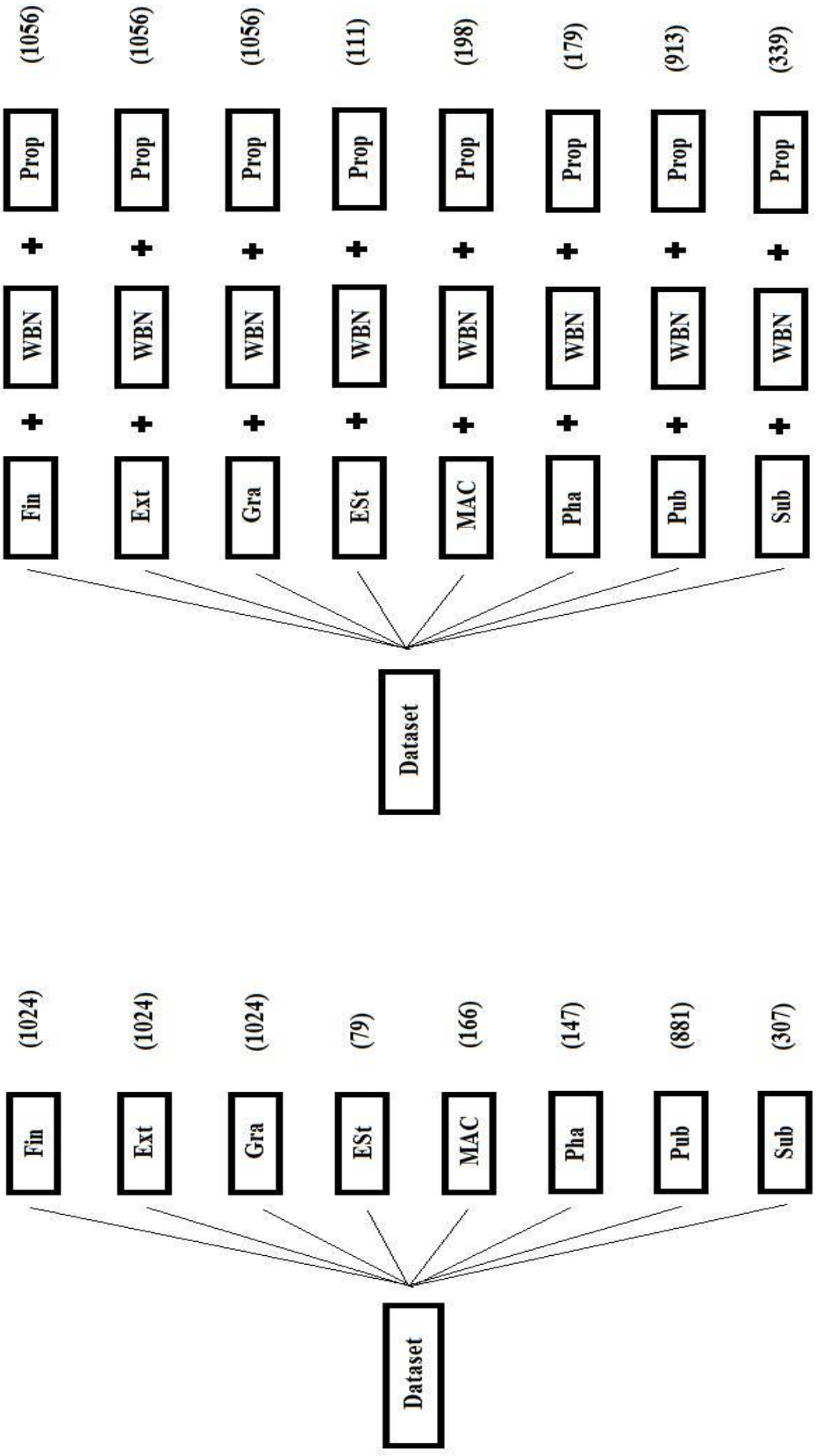


Figure 17: Illustrating the generation of fingerprints

Binary and Numerical descriptors. The numbers in the parenthesis are the number of features.

3.3. Summary of Data Pre-processing

In this chapter we explored data imbalance briefly. In a nutshell, datasets, their origins and what the levels of imbalance in them are and the number of instances in them were succinctly introduced. Next we described the data preparation and how the features for the datasets were generated. The various software used was described. Also we became familiar with the fingerprints that were used for this study.

As mentioned in the introduction to this chapter, the aim of this study was to devise a new and novel approach to classifying imbalanced high dimensional data so that it would apply to all dataset regardless of their characteristics. Importantly, the datasets that were chosen for this study are representative of a wide range of scenarios comprising the whole spectrum of typical challenges in data mining and virtual screening.

In the next chapter, we shall discuss the algorithmic tools that will enable us to analyse and perform effective virtual screening under such heterogeneous settings. We talk about data imbalance and the complications it brings with itself to classification. We also touch on how to tackle the imbalance problem and introduce the reader to SMOTE. The reader shall also become more familiar with the methodology used in this research.

4. Dataset Processing

In this chapter we embark on a journey to delve deeper into to the data imbalance problem and how it affects classification. We also explore the different methods that have been utilised to battle this phenomenon. Finally we talk about the novel methodology used in this work in order to provide a unified process (not tailored to a particular type of dataset) for classifying heavily imbalanced high-dimensional datasets regardless of the origins or the type of data used.

At this point it is worthy to remind the reader that the main novelty of the work presented is to show that the combination of over-sampling using SMOTE in specific and the utilisation of four main classifiers furnishes a generic, unified analysis for a wide range of cheminformatics data;; unlike other methods of dealing with imbalanced data in which the classifier is altered to meet the classification requirements for a specific type of data.

Therefore, in this chapter, a description based on pseudocode has been preferred over a detailed mathematical formulation since the focus is not that much on the algorithm-specifics as will be clear in the next chapters and no mathematical alterations were implemented on the classifiers used. However, where possible the mathematical equations have been shown for the readers' convenience.

In summary, this approach can be used on various datasets regardless of the imbalance ratio affecting it. This enables the cheminformatics data analysis to follow a robust protocol in cases where the unbalance changes over time and may not be representative of the scenario in future datasets at the time of the analysis.

4.1. Data Imbalance

A question which will come to the reader in this section is: what is data imbalance (imbalanced dataset) and how do we determine whether the data being studied is imbalanced? In the context of classification, an imbalanced dataset is a dataset in which the classes have an unequal number of instances. But it is only in a very ideal world where the different classes in a dataset are represented by the exact same number of instances. So the next question might be what are the requirements

for a dataset to be considered imbalanced? In truth there are no concrete or standard requirements for this definition.

But most practitioners would agree on the following (He & Ma 2013):

- A dataset where the most common class is less than twice as much as the rarest class is considered marginally imbalanced.
- A dataset in which the imbalance ratio (most common class to rare class) is 10:1 can be considered modestly imbalanced.
- A dataset in which the imbalance ratio is 1000:1 and above is considered a highly imbalanced dataset.

Most Chemoinformatics-related problems are related to datasets that are highly imbalanced and it is these rare classes that are of interest in data mining (DM). Standard chemical molecular classification techniques assume equality between classes therefore will not be very effective (Ganganwar 2012; Zięba et al. 2015). When classifying imbalanced datasets, it is more important to correctly classify minority classes. These rare classes often get misclassified because most classifiers optimise the overall classification accuracy (Ertekin et al. 2007), but one must keep in mind that using global accuracy as an evaluation metric will obviously not reflect the true performance of the classifier since minority classes have less impact on the accuracy than majority classes (He & Ma 2013). Most original classifiers tend to minimise the error rate: the percentage of incorrect prediction of class labels. They assume that all misclassification errors cost equally. But as we know in real world problems misclassifying errors is costly indeed, such as an error in diagnosing cancer in a patient.

Researchers (Visa & Ralescu 2005; Ganganwar 2012; He & Ma 2013; Cai et al. 2014) agree that the reasons for the poor performance of the existing classification algorithms on imbalanced datasets are:

- Original classifiers (classifiers in their original unaltered state) are accuracy-driven. This means that their goal is to minimise the overall error to which the minority class has very little or no contribution.
- They assume that the distribution of the data for all classes is the same.
- They assume that errors originating from the different classes have the same cost.

Some of the other reasons (Weiss 2004; He & Garcia 2009; Cai et al. 2014) for the complications caused by imbalanced datasets fir classification are:

- **Absolute lack of data:** Here the instances of rare class only cover a small area of the data in the dataset therefore it becomes very difficult to detect patterns from the data due to the misclassification and error rates introduced by the rare instances. This situation arises from the fact that the minority class instances are very rare in the whole dataset.
- **Relative lack of data:** This is when the frequency of occurrence of the instances in the dataset is much less than the whole data. Because some patterns depend on the combination of many conditions, many DM algorithms which examine conditions in isolation might not provide much information due to other more common patterns obscuring the rare patterns. This is when the minority class is not rare in its own right, but rather relative to the majority class.
- **Data fragmentation:** In most DM approaches the search space is divided into smaller spaces resulting in a fragmentation; DM algorithms employ a divide and conquer strategy whereby the original problem is decomposed into a smaller and a smaller problem. Now in the case of rare classes, detecting the presence of instances and a pattern will become very difficult since the very existence of the regularities within these decomposed spaces becomes scarcer.
- **Noise:** Classes with fewer instances are very sensitive to the existence of other instances, so for example if in a chosen data space there are many instances of the greater class, the presence of a few rare instances will not affect the learning process of the algorithm. But the presence of the greater class instances amongst the rare instances, however few in number, will have a great impact on the learning process as illustrated in Figure 18. Here the minus represents the majority class and the plus sign represents the minority class.

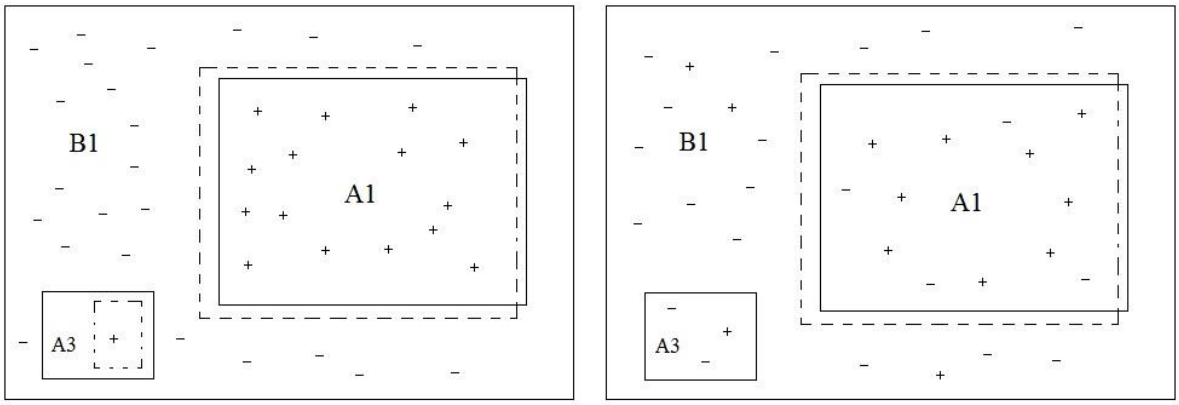


Figure 18: Illustrating how the introduction of noise can affect the learning classifier's ability to learn decision boundaries. (Source Weiss 2004)

In the right side of Figure 18, introduction of noise into the A1 space (adding negative classes) has had no effect on the classifier's ability to learn the decision boundary, because of the classifier's ability to generalise. But the two noisy instances in A3 have caused the classifier the inability to learn this rare instance at all. In this case the classifier cannot distinguish between the rare instance and noise.

As indicated in the abstract of the thesis, Virtual Screening (VS) in drug discovery involves processing large datasets containing unknown molecules in order to find the ones that are likely to have desired effects on a biological target. These molecules are different from each other and the interaction between them is not part of the screening process. The level at which this framework applies to the drug discovery process (as seen in Figure 1), is at the very early stages of it. Thus, the whole process boils down to identifying molecules that are active or non-active to a specific target. Hence the scenario is naturally described as a binary classification problem (Reddy et al. 2007; Vyas et al. 2008; Lavvecchia & Di Giovanni 2013; Lionta et al. 2014); therefore, this approach is followed here. Hypothetically, as an alternative a multi-class problem can be used but as pointed out in the answer to the first question in this document, classification of multi-class imbalanced high-dimensional datasets will be less robust considering the various types of data and computationally expensive. Plus, one may easily lose performance on one class while trying to gain it on another (Sáez et al. 2016). In addition, some heavily used robust classifiers such as support vector machines, are typically more effective in binary classification problems (Yang et al. 2013; Meyer & Wien 2015). As

mentioned above, to this date, most multi-class problems in this area are typically broken down into binary problems for an optimal solution.

Chemoinformatics data is typically imbalanced in general with a small ratio of active compounds to non-active ones. This could be seen from the observations made in the literature by various authors (Han et al. 2008; Weis et al. 2008). Data deposited in public and private repositories such as PubChem bring great opportunities for researchers in Chemoinformatics, however the imbalanced nature of the data from High-Throughput Screening in these repositories hinders the classification process (Li et al. 2009). The main problem with imbalanced datasets is that standard classifiers are often biased towards the majority class since these algorithms assume a relatively balanced distribution of classes (Chawla et al. 2004; Cieslak et al. 2006; Sun et al. 2009; López et al. 2013; Imran et al. 2014) and as a result they fail to identify the minority class. In this thesis, we have replicated these results in Figures 255 and 256. Some results can be seen in the tables below:

		Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
NB	EState	0.0151515	0.995705733	0.004294	0.984849	0.993004
	Extended	0.0878787	0.936514333	0.063486	0.912121	0.934177
	Fingerprinter	0.0636363	0.953013567	0.046986	0.936364	0.950563
	Graph-Only	0.1272726	0.933073767	0.066926	0.872727	0.930854
	MACCS	0.218181633	0.952636933	0.047363	0.781818	0.950614
	Pharmacophore	0.1060605	0.982061	0.017939	0.89394	0.979648
	PubChem	0.2060604	0.903557733	0.096442	0.79394	0.901636
	Substructure	0.090909	0.979909633	0.02009	0.909091	0.977461

Table 6: Misclassification of raw PubChem datasets #1

		Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
SMO	EState	0	0.999966533	3.35E-05	1	0.997212
	Extended	0.0757575	0.9783778	0.021622	0.924243	0.975891
	Fingerprinter	0.0575757	0.9795999	0.0204	0.942424	0.97706
	Graph-Only	0.0424242	0.981307567	0.018692	0.957576	0.978721
	MACCS	0.0242424	0.9961075	0.003893	0.975758	0.99343
	Pharmacophore	0	0.999070833	0.000929	1	0.996319
	PubChem	0.030303	0.985978533	0.014021	0.969697	0.983346
	Substructure	0.0030303	0.9989871	0.001013	0.99697	0.996243

Table 7: Misclassification of raw PubChem datasets #2

These are more prevalent in the figures placed in the appendix for AID362 and AID456.

4.2. Tackling Imbalanced Data Problem

Many strategies have been suggested to address the data imbalance problem throughout the years (Weiss 2004; He & Garcia 2009; He & Ma 2013; Cia et al. 2014; Maratea et al. 2014; Shi et al. 2015). Below are only some of the more prominent techniques that have been used to tackle the data imbalance in datasets.

Cost-Sensitive Classification

In a case where some class instances in a dataset are rare, not detecting patterns belonging to the rare class or predicting them as the common class can happen (false negatives). This can affect business decision-makings but in some cases such as medical diagnosis it can be fatal, in machine learning terms it has greater cost.

The classification results using Weka Toolkit (Hall et al. 2009) are presented as a matrix called the Confusion matrix (contingency table). This matrix has four sections as True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN) (reference in such stats, anything will do). For bioassay data and screening compound selection, it is better to minimise the number of the FNs; these are the active molecules which have been incorrectly classified as inactive. This can be done at the cost of increasing the number of FPs. Cost-sensitive classifiers offer the advantage of being able to control the number of FPs. By applying penalty on the FNs the number of FPs will increase. The number of TPs and TNs does not get affected much by applying the cost. Table 8 shows a Weka cost matrix. For example if a cost of 8 is applied to False Negatives whilst keeping the default cost for all the other misclassification schemes, this means that it is more costly misclassifying positives than misclassifying negatives. Schierz (2009) concluded that there are no guidelines for setting the misclassification costs.

	Actual Positive	**Actual Negative**
Predicted Positive	0 TP	1 FP
Predicted Negative	8 FN	0 TN
	(+)	(-)

Table 8: A cost matrix showing the misclassification cost for positives and negatives

Sampling

Sampling is a very common method when dealing with imbalanced data. Here the data is rebalanced i.e. the number of instances of each class is changed so that standard machine learning algorithm classifiers can be applied to the problem. The goal is to minimise the problems related to imbalanced data (as mentioned above) by reducing class imbalance. Sampling can be done either randomly or intelligently; according to some rule (Weiss 2004; Chawla 2009; He & Garcia 2009). Popular methods of sampling are:

Over-sampling: This method works by re-sampling the minority class instances till it has as many instances as the majority class. In random over-sampling a set of instances are randomly selected from the minority class. They are replicated and added to the whole dataset in order to balance the distribution of classes. A more informed way of over-sampling is called SMOTE which stands for Synthetic Minority Over-sampling Technique (Chawla et al. 2002; Blagus & Lusa 2012; Ramezankkhani et al. 2014; Verbiest et al. 2014; Saéz et al. 2015). SMOTE introduces non-replicated artificially created data into the dataset based on the feature space similarities between existing minority examples.

Under-sampling: In this method of sampling, instances from the majority class are removed in order to gain balance between the majority and minority classes. In random under-sampling a set from the majority class is randomly selected and removed from the whole dataset to adjust the balance between classes and make the rare class less rare. A more intelligent method is to remove majority instances which are on the borderline (close to the boundary of majority / minority), those that suffer from class-label noise and those which are redundant (Kubat & Matwin 1997).

The two methods mentioned above reduce the class imbalance but they do have their own disadvantages. Over-sampling often duplicates the same instances from the minority class which lead to over-fitting and because it does not produce any new data, it is not assisting with the lack of data problem associated with imbalance. Another issue is that over-sampling might increase the time needed to build a classifier due to increasing the number of instances.

Under-sampling removes majority class instances but by doing so it would be discarding potentially important information. This can reduce the classifier

performance because the classifier might miss important concepts about the majority class. It is unclear which of the mentioned sampling methods works better; results show that the choice of method is domain-specific (Weiss 2004).

SMOTE: As mentioned in Chapter 3, SMOTE is a sampling approach whereby the minority class samples are over-sampled by creating synthetic samples. Here we explore the creation of these synthetic samples a bit further and more technically. The schematic sample generation is demonstrated below in Figure 19. For a positive class sample X_i, its distance from other samples of the same class is calculated, then a sample X_j from the k-nearest neighbour sample of the positive class is randomly chosen and a new sample is generated (Li et al. 2014). $X_{new} = X_i + rand(0,1) \times (X_j - X_i)$ (Figure 19).

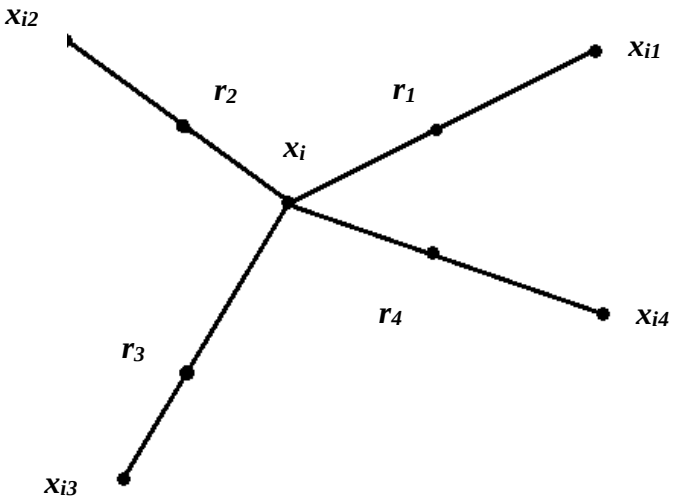


Figure 19: Generating synthetic samples by SMOTE

In Figure 19 X_i is the selected point and X_{i1} to X_{i4} are some selected nearest neighbours and r_1 to r_4 are the synthetic samples created through randomised interpolation.

In Chawla et al. (2002) and Zhang et al. (2016), the authors state that SMOTE corrects the simple over-sampling technique's side-effect, over-fitting, by creating synthetic instances rather than over-sampling with replacement. These instances are generated in the feature space rather than the data space. In SMOTE, the minority class is over-sampled by taking a minority class sample and introducing synthetic examples along the line segments joining any / all of the k minority class nearest

neighbours. The new instances stem from interpolation rather than extrapolation, so they still carry relevance to the underlying dataset (Pears et al. 2014). The neighbours are randomly chosen based on the amount of over-sampling required. This forces the decision region of the minority class to become more general (Chawla et al. 2002; Chawla 2005; Han et al. 2005; He 2010; Elrahman & Abraham, 2013; Branco et al. 2016; Ng et al. 2016). As a result, more general regions are now learned for the minority class rather than those being included by the majority class. SMOTE forces focused learning and introduces a bias towards the minority class. Thus, it is evident that the cross-validation method used must carefully consider this bias and make sure that true performance metrics in test sets (described below in section 4.3) refer to real data samples.

The pseudocode for SMOTE is as follows:

Algorithm SMOTE(T, N, k)

Input: Number of minority class samples *T*; Amount of SMOTE *N%*; Number of nearest neighbours *k*
Output: *(N/100)* T* synthetic minority class samples

1. *(∗ If N is less than 100%, randomize the minority class samples as only a random*
   percent of them will be SMOTEd. ∗)
2. **if** *N* <100
3. **then** Randomize the *T* minority class samples
4. *T = (N/100) ∗ T*
5. *N* = 100
6. **endif**
7. *N = (int)(N/100)(∗ The amount of SMOTE is assumed to be in integral multiples of*
   100. ∗)
8. *k* = Number of nearest neighbours
9. *numattrs* = Number of attributes
10. *Sample*[ ][ ]: array for original minority class samples
11. *newindex*: keeps a count of number of synthetic samples generated, initialized to 0
12. *Synthetic*[ ][ ]: array for synthetic samples
    (∗ Compute k nearest neighbours for each minority class sample only. ∗)
13. **for** i ← 1 **to** *T*
14. Compute *k* nearest neighbours for i, and save the indices in the nnarray
15. Populate(*N, i, nnarray*)
16. **endfor**
    Populate(N, i, nnarray) (∗ Function to generate the synthetic samples. ∗)
17. **while** *N* ≠ 0
18. Choose a random number between 1 and *k*, call it *nn*. This step chooses one of the *k* nearest neighbors of *i*.

19. **for** *attr* ← 1 to *numattrs*
20. Compute: *dif* = *Sample[nnarray[nn]][attr] − Sample[i][attr]*
21. Compute: *gap* = random number between 0 and 1
22. *Synthetic[newindex][attr] = Sample[i][attr] + gap ∗ dif*
23. **endfor**
24. *newindex++*
25. *N* = *N* − 1
26. **endwhile**
27. **return** *(∗ End of Populate. ∗)*

End of Pseudo-Code.

When and how SMOTE does cause over-fitting has been addressed from different angles in the literature. As a case study on potential over-fitting of SMOTE, in the research performed by Kothandan (2015), the classification of the miRNA datasets associated with cancer was performed using SMOTE as one of the techniques in overcoming class imbalance. The results obtained from using SMOTE indicated a precision of > 0.9 in all independent test runs, indicating over-fitting. This could be due to the fact that SMOTE focuses on specific regions of the feature space as the decision region for the minority class rather than increasing the overall number of the instances. As a result, SMOTE over-populated a region rather than increasing the overall instances.

One of the drawbacks of SMOTE is that it generates synthetic samples for the minority class while disregarding the majority class samples (Branco et al. 2016), which in turn increases the overlapping between classes. This may lead to over-generalisation (Zhang et al. 2010; López et al, 2013; Sáez et al. 2015). This combined with making the decision regions of the minority class more general, could lead to the creation of borderline examples (Sáez et al. 2014). SMOTE is unable to provide a scalar control of the number of the newly created instances and cannot guide the selection of them, resulting in not very good quality instances (Li et al. 2014).

Extensions to the original SMOTE have been developed in order to combat some of the side effects of SMOTE such as over-generalisation. They act as cleaning methods removing any data samples that could be on the borderline of classes, noisy samples and outliers. Examples of these methods are: SMOTE-IPF (Sáez et al. 2015) which can be used to battle the noisy and borderline examples produced by over-sampling the minority class, SMOTE-ENN (Luengo et al. 2011) which uses the Wilson's Edited Nearest Neighbour Rule (ENN) as a pre-processing method to

remove outliers, Borderline-SMOTE (Han et al. 2005) which over-sample the borderline samples of the minority class. These methods are additions to the original SMOTE algorithm. Utilising them with the datasets in this study would have potentially increased the computational costs extremely as additional processes would have needed to be run before receiving the balanced and over-samples datasets, adding to the processing time and probably extending the processing times dramatically. Plus, the original SMOTE algorithm is readily available to all researchers with different knowledge and can be used out of the box or some parameters can be changed.

After using SMOTE on our imbalanced datasets, the number of minority class samples were increased to match the number of majority class samples making our datasets balanced in order to perform our classifications. Tables 7 and 8 show the original number of samples in each dataset and how that changed after over-sampling by SMOTE.

Dataset	**# Total**	**# Class 1**	**# Class 2**	**Class Ratio**
Fontaine	435	279	156	1.7884
AID362	4279	60	4219	0.0142
AID456	9982	27	9955	0.0027

Table 9: Original number of samples in unbalanced datasets

	Dataset	**# Total**	**# Training**	**# Test**
	Fontaine	588	335	253
Method 1	**AID362**	8438	5063	3375
	AID456	19910	11946	7946
	Fontaine	508	334	174
Method 2	**AID362**	4787	3075	1712
	AID456	15944	11951	3993

Table 10: Number of samples in balanced datasets

Various elements lead to an imbalanced classification problem becoming a rather difficult one. Class imbalance on its own makes the learning task complicated by having a disproportion between class examples (Sun et al. 2009). However, that is not the only problem. The number of minority class examples might not be sufficient to train a classifier, the validation scheme used to estimate the classifier might lead to high error rates and minority class samples might form small distributed groups

(Chawla et al. 2002; Bunkhumpornpat et al. 2009; Sáez et al. 2016). In short, the difficulty in classification depends on the degree of imbalance but also on the characteristics of the data in a non-trivial fashion. In conclusion, the challenging problem in Chemoinformatics is the screening of overly imbalanced datasets and this scenario is thus the main focus of this study. Thus, there is no rule of thumb and the degree of imbalance in the test set cannot be assumed to be known in advance in a real setting. The main goal of this thesis is to devise a unified protocol to apply to all datasets regardless of the data characteristics. In conclusion, the challenging problem in Chemoinformatics is the screening of overly imbalanced datasets and this scenario is thus the main focus of this study.

4.3. Evaluating Imbalanced Learning Outcomes

In order to assess the effect of the algorithms used on imbalanced data one needs to apply standard evaluation metrics to the outcomes of the classification process. Metrics can be dependent or independent of the distribution of the data. When looking at the confusion matrix (please see Table 8) one can observe that the left column of the table represents the positive instances and the right column represents the negatives. The ratio of the two columns characterise the class distribution. An evaluation metric which uses both columns in its calculation becomes sensitive to (dependent on) any imbalance in the dataset (He & Garcia 2009). Imbalance-sensitive metrics cannot assess the performance of classifiers because variations in the distribution of data cause a change in the measures of performance even though the performance of the classifier has not changed. Examples here can be precision and accuracy. Precision determines the fraction of the instances classified by the classifier that actually belongs to that class. But as the formula reads, it depends on both column of the confusion matrix and it does not declare the false negatives. Accuracy measures how error-free the model's predictions are. Accuracy does not include cost information; it assumes equal cost for data being classified as false positive (false alarm) or false negative (misclassified).

$$Precision = \frac{TP}{TP + FP}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

One metric which is not dependent on the imbalance is recall. Recall is the ability of an algorithm to select instances of a certain class from the dataset. If we look at how recall is calculated we can see that this metric uses only one column from the table, making it non imbalance-sensitive. This makes it ideal for assessing the performance of the algorithm used. Unfortunately recall does not provide information about the false positives.

$$Recall = \frac{TP}{TP + FN}$$

F-measure is a metric which combines recall and precision (harmonic mean of both).

$$F - Measure = \frac{(1 + \beta)^2 * Recall * Precision}{\beta^2 * Recall + Pricicion}$$

It can provide more information about the classifier than accuracy and at the same time it is sensitive to the data distribution. β is a coefficient to adjust the relative importance of precision and recall, usually β = 1.

4.4. Classification

Classification is the act of assigning items in a collection to target classes. The goal here is to accurately predict the target class for each of the instances in the dataset. The classification task begins with a dataset in which the class assignments are known. Therefore, the model is built based on the observed data. In this model, the algorithm used find relationships between the values of the predictors and the values of the target. Different algorithms use different techniques to establish these relationships; but in general classification models are tested by comparing the predicted values to known targets. The data used for classification is usually divided into two datasets: the training set for building the model and the test set for testing the built model.

The datasets for this study were split into test and train sets as 60% training and 40% test. The split was done randomly and 30 runs for each experiment so that the dataset could be explored in most possible ways and the combinations could be tested in order to get a statistically sound result. The split was done in a stratified manner so that the class distribution in all the train / test cases would be the same (Bouckaert et al. 2013).

The first balancing method in this study was developed in order to analyse the effect of the SMOTE technique *before* computing a "genuine" out-of-sample prediction. In other words, we first evaluate the classification metrics when using the both real and synthetic data from SMOTE in the test set ("SMOTE" operates before the *out-of-sample prediction*), as opposed to when the test set consists exclusively of samples from the original test dataset and not artificial data generated by SMOTE (what we term here a "genuine" *out-of-sample* prediction). This is still interesting,

because SMOTE has an effect on data that is often not trivial and depends on the sparseness of the data in the space of variables (Chawla, 2005; Sruthi et al. 2015). Thus, it will enable us to discuss more specifically the potential reasons for the success or failure of the genuine validation (on non-oversampled test datasets) computed in the next balancing method where only the training set was balanced and the test set was not.

Nonetheless, is interesting to stress that the results shown, correspond to a thorough cross-validation for the over-sampled dataset. After balancing, the whole dataset was then split into training (60%) and test (40%); we performed the splitting 30 different times and in a stratified manner so that the test set does not always contain the same instances from the same classes. As a result, each random given instance has the chance to appear in both training and test sets. Using cross-validation decreases the chances of SMOTE causing over-fitting; yet of course a genuine cross validation in non-oversampled data performed next is the only fully reliable analysis.

In order to perform the classification for this study, the open source machine learning software Weka (Hall et al. 2009) was used due to its outstanding capabilities in large datasets processing unlike other commercial platforms. The 32-bit version of Weka only utilises 2GB of physical memory and the 64-bit version only 4GB. All the acquired datasets are originally in the structured data format (sdf). These files are converted to the available molecular fingerprints using PaDel and PowerMV. The datasets produced by PaDel fingerprints contain binary structural descriptors. In order to include numerical properties without the memory issue, the numerical descriptors generated by PowerMV (only 32 attributes) are added to the structural descriptors. The datasets are then imported into Weka and classification is performed using four main algorithms Random Forest, J48, Naïve Bayes and SMO. A description of the utilised classification algorithms is as follows:

- **Sequential Minimal Optimisation (Weka's implementation of Support Vector Machine)**

This algorithm implements John C. Platt's sequential minimal optimisation algorithm for training a support vector classifier. Training a Support Vector Machine requires solving a large quadratic programming optimisation problem. Sequential Minimal Optimisation (SMO) breaks down this large quadratic problem into a

succession of smaller quadratic problems. These smaller problems are then solved analytically in order to avoid becoming optimisation inner-loops in the code of the algorithm, therefore saving time. The amount of memory used for SMO is linear in the training set size allowing it to handle large training sets (Platt 1998; Flake & Lawrence 2002; Wu et al. 2013).

Support Vector Machines (SVM) offer high performance at classifying datasets with either a very small subset of features or with extreme ones (Wald et al. 2013). SVM models depend on the samples on the margins of each class, also called support vectors (Liu et al. 2013), unlike other classifiers that use all the samples in the dataset in order to determine the boundaries between classes. Support Vector Machines are believed to be less susceptible to class imbalance than the other classification algorithms. The reason behind this is that the boundaries between classes in SVMs are calculated with respect to only a few support vectors (as mentioned above) and class size should not affect the class boundaries too much. However, previous research (Wu & Chang, 2003; Akbani et al. 2004; Batuwita & Palade, 2013; Prati et al. 2015) shows that SVMs can be rendered ineffective in determining class boundaries if the class distribution is too skewed (1000:1 majority to minority rate). The reason behind this is that as the training data becomes more imbalanced, the support vector ratio between the classes also becomes more imbalanced. The small amount of cumulative error on the minority class instances count for very little in the trade-off between maximising the width of the margin and minimising the training error. As a result SVMs learn to classify everything as the majority class so that the margin becomes the largest and the error the minimum (Sun et al. 2009).

Given a set of training data ($\mathbf{x}_i$, y_i), where i = 1, …, N, $\mathbf{x}_i \in \mathbf{R}^d$, $y_i \in \{-1, 1\}$. If there are some hyperplanes that separate the data points with different classes, then hyperplane H is defined as $\mathbf{wx} + b = 0$ and the perpendicular distance between the hyperplane and the origin is $\frac{|b|}{||\mathbf{w}||}$ when $\mathbf{w}$ is normal to H (Zheng et al. 2015). For a binary classification problem such as the case of our project, two hyperplanes are defined as H_1: $\mathbf{wx} + b = -1$ and H_2: $\mathbf{wx} + b = 1$, where the data points in the majority class satisfy $\mathbf{wx} + b \leq -1$ and the data points in the minority class satisfy $\mathbf{wx} + b \geq 1$. Training data vectors unquietly defining such delta-margin hyperplane(s) are termed

support vectors; because the entire classification of the test data solely relies on these vectors. Vectors “support” the optimal solution of the classification algorithm and will determine the predicted class of the test data (Scholkopf and Smola, 2002; Bishop, 2006).

The pseudocode for SMO (Platt 1999) can be seen below:

```
target = desired output vector
point = training point matrix

procedure takeStep (i1, i2)
    if (i1 == i2) return 0
    alph1 = lagrange multiplier for i1
    y1 = target [i1]
    E1 = SVM output on point[i1] – y1 (check in error cache)
    s = y1*y2
    Compute L, H via equations (13) and (14)
    if (L == H)
        return 0
    k11 = kernel (point[i1], point[i1])
    k12 = kernel (point[i1], point[i2])
    k22 = kernel (point[i2], point[i2])
    eta = k11 + k22 – 2*k12
    if (eta > 0) {
        a2 = alph2 + y2 * (E1 – E2) / eta
        if (a2 < L) a2 = L
        else if (a2 > H) a2 = H
    }
    else
    {
        Lobj = objective function at a2 = L
        Hobj = objective function at a2 = H
        if (Lobj < Hobj – eps)
            a2 = L
        else if (Lobj > Hobj + eps)
            a2 = H
        else
            a2 = alph2
    }
    if (| a2 – aplh2| < eps * (a2 + alph2 + eps))
        return 0
    a1 = alph1 + s * (alph2 – a2)
    Update threshold to reflect change in Lagrange multipliers
    Update weight vector to reflect change in a1 & a2, if SVM is linear
    Update error cache using new Lagrange multipliers
    Store a1 in the alpha array
```

```
40.     Store a2 in the alpha array
41.     return 1
42.  endprocedure
43.
44.  procedure examineExample (i2)
45.     y2 = target [i2]
46.     Alph2 = Lagrange multiplier for i2
47.     E2 = SVM output on point [i2] – y2 (check in error cache)
48.     r2 = E2 * y2
49.     if ((r2 < -tol && alph2 < C) || (r2 > tol && alph2 > 0)) {
50.        if (number of non-zero & non-C alpha > 1) {
51.           i1 = result of second choice heuristic
52.           if takeStep (i1, i2)
53.              return 1
54.        }
55.        Loop over all non-zero and non-C alpha, starting at a random point {
56.           i1 = identity of current alpha
57.           if takeStep (i1, i2)
58.              return 1
59.        }
60.        loop over all possible i1, starting at a random point {
61.           i1 = loop variable
62.           if (takeStep (i1, i2)
63.              return 1
64.        }
65.     }
66.     return 0
67.  endProcedure
68.
69.  main routine:
70.     numChanged = 0;
71.     examineAll = 1;
72.     while (numChanged > 0 | examineAll) {
73.        numChanged = 0;
74.        if (examineAll)
75.           loop I over all training examples
76.              numChanged += examineExample (I)
77.        else
78.           loop I over examples where alpha is not 0 & not C
79.              numChanged += examineExample (I)
80.        if (examineAll ==1)
81.           examineAll = 0
82.        else if (numChanged == 0)
83.           examineAll = 1
84.     }
```

- **J48 (Weka's implementation of C4.5)**

J48 implements a state of the art Quinlan's C4.5 algorithm (Quinlan 1993; Quinlan 2014) for generating a pruned or un-pruned C4.5 decision tree. Decision trees as a predictive model, map observations about an item to the conclusions about the item's target value. In tree structures the leaves represent class labels and branches represent conjunction of features that lead to those class labels. J48 builds decision trees from a set of labelled training data using information entropy. This employs the fact that each attribute of the data can be used to make a decision by splitting the data into smaller subsets. In order to make a decision, J48 examines the information gain that comes from choosing an attribute for splitting the data. The attribute with the highest normalised information gain is used. The algorithm moves on to smaller subsets. The splitting stops when all instances in a subset belong to the same class.

The pseudocode for C4.5 is as follows (Yasin et al. 2014):

1. Input: a dataset D
2.
3. begin
4. Tree = {}
5. If (D is "pure") || (other stopping criteria met) then terminate;
6. For all attribute a $\alpha \in D$ D do
7. Compute criteria impurity function if we split on α;
8. α_{best} = Best attribute according to above computed criteria
9. Tree = Create a decision node that tests α_{best} in the root
10. D_v = Induced sub-datasets from D based on α_{best}
11. For all D_v do
12. begin
13. Tree $_v$ = J48(D_v)
14. Attach Tree $_v$ to the corresponding branch of tree
15. end
16. return tree
17. end

- **Random Forest (RF)**

Random Forests are combinations of tree predictors such that each tree depends on the value of a random vector sampled independently and with the same distribution for all trees in the forest (Breiman 2001). In short a Random Forest is an ensemble of decision trees that will output a prediction value. Each decision tree is constructed by using a random subset of the data and gives a classification and votes

for that class. The forest chooses the classification having the most votes; the most popular class.

The element that has contributed to its popularity is that it can be applied to a wide range of problems and has only a few parameters to tune. Apart from this, it is known to be able to deal well with small sample sizes, high-dimensional feature spaces and complex data structures (Scornet et al., 2015).

Random Forest has an excellent performance in classification tasks that can outperform other classifiers. Some of its features which allow for this to happen are as follows (Díaz-Uriarte & Alvarez de Andrés, 2006; Khoshgoftar et al. 2007):

- This classifier can be used where the number of features are greater than the number of observations.
- It can be used for binary and multi-class problems.
- Performs well with noise and shows robustness to large feature sets.
- As the number of trees increase, the chance of over-fitting decreases.

The mathematical equation for Random Forest can be shown as below:

Assuming a dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

Drawn randomly from a probability distribution $(x_i, y_i) \sim (X, Y)$

Given the ensemble of classifiers $h = \{h_1(x), \ldots, h_k(x)\}$

If each $h_k(x)$ is a decision tree then the ensemble is a Random Forest.

The parameters of the decision tree for the classifier $h_k(x)$ are $\Theta_k = (\Theta_{k1}, \Theta_{k2}, \ldots, \Theta_{kp})$

The decision tree k leads to a classifier $h_k(x) = h(x|\Theta_k)$

The following shows the pseudocode for Random Forest classifier (Kouzani et al. 2009):

1. select the number of tress to be generated
2.
3. for (k = 1; k <= K; k++)
4. draw a bootstrap sample Θ_k from the training data
5. grow an unpruned classification tree $h(x, \Theta_k)$
6. for (*i* = 1; *i* = *number-of-nodes*; *i*++)
7. randomly sample *m* predictor variables

```
8.          select the best split from among those variables
9.      end
10. end
11. e
12. each of the K classification tress casts 1 vote for the most popular class at input
    x
13. e
```

aggregate the classification of the *K* tress and select the class with maximum votes

- **Naïve Bayes (NB)**

This is a specialised form of the Bayesian network. The algorithm relies on two assumptions: first that the predictive attributes are conditionally independent given the class and second that no hidden attributes affect the prediction process (John & Langley 1995).

The pseudocode for Naïve Bayes can be seen as below (Yang & Webb 2003):

```
1.  “F”: frequency tables
2.  “I”: number of instances
3.  “C”: how many classes
4.  “N”: instances per class
5.
6.  Function update (class, train) {
7.  I++
8.  if (++N[class]==1
9.      then C++
10. fi
11. for <attr, value> in train
12.     do
13.         if (value != “?”)
14.             then F[class, attr, range] ++
15.         fi
16.     done
17. }
```

Each of the four algorithms used has its own advantages and disadvantages. NB can be used in HTS as a simple classifier for actives and non-actives. It is guided by the frequency of the occurrence of molecular descriptors in the training set. NB depends on the two assumptions mentioned above namely the independence of attributes from each other and that all attributes are equally important. Normally

these assumptions are violated but NB is a robust algorithm and very tolerant towards noise and handles large datasets very well (Plewczynski et al. 2006).

With decision trees (or forests) the molecular descriptors which describe the molecular features of the training set are systematically added to a decision tree model one at a time until compounds that have different biological properties are adequately separated. Decision trees take in objects and situations described by properties and output a yes or a no. In general they represent a disjunction of conjunctions of constraints on the attribute value of instances. RF can handle thousands of attributes and gives estimates of which variables are important during classification. RF does not over-fit and is a fast method (Muegge & Oloff 2006; Plewczynski et al. 2006).

There is much interest in using Support Vector Machines (SVMs) for compound classification and label prediction. One may question that whether using low-dimensional space representation is necessary for better virtual screening or molecular similarity results. SVMs project compounds as descriptor vectors into high-dimensional spaces and then construct a maximum-margin hyperplane by linear combination of training set vectors to optimally separate two classes of compounds. SVMs are one of few methods that have been developed to navigate high-dimensional descriptor spaces (Eckert & Bajorath 2007).

Table 11 contains the advantages of the classifiers used in this study. Of course, the simple, qualitative comparison in the figure refers exclusively to cheminformatics dataset (Galathya et al., 2012) although some of the differences have been observed in benchmark data.

Decision Trees	**Naïve Bayes**	**Support Vector Machine**
Easily observed and develop generated rules	Fast, highly scalable model building (parallelised) and scoring	More accurate than decision tree classification

Table 11: Advantages of decision trees, Naïve Bayes, SVM classifiers**. (**Source: Galathiya et al. 2012)

Table 12 summarises some feature comparisons between the classifiers used in this work.

Feature	Decision Trees	Naïve Bayes	Support Vector Machine
Learning Type	Eager Learner	Eager Learner	Eager Learner
Speed	Fast	Very Fast	Fast with Active Learning
Accuracy	Good	Good	Significantly High
Interpretability	Good	-	-
Transparency	Rules	Black Box	Black Box

Table 12: Some of the features from classifiers used in this study. (Source: Galathiya et al. 2012)

- **Ensemble Learning**

Ensemble learning is a general term for combining the prediction of several learning models which may be assumed weak, into a single model which is a combination of the different classifiers it is made up of (Murphree et al. 2015). The ensemble model is often found to perform better (Friedman et al. 2001). Ensemble learning can be regarded as machine learning techniques whose decisions are combined in a way to improve the performance of the overall system. The concept states that no single approach can claim to be superior to any other and the integration of several single approaches will enhance the performance of the final classifier. Therefore, an ensemble classifier can have overall better performance than the individual base classifiers. The effectiveness of the ensemble methods is highly dependent on the independence of the error committed by the base learners (Tan & Gilbert, 2003). One type of ensemble methods is Majority Voting. Majority voting counts the class prediction of all the base models and assigns a class based on the majority opinion. If there are *n* independent classifiers that have the same probability of being correct, and each of them can produce a unique decision regarding the identity of the unknown pattern, then the pattern is assigned to the class for which there is a consensus; when at least *k* of the classifiers agree. *k* is defined as:

$$k = \begin{cases} \frac{n}{2} + 1 & \text{If } n \text{ is even} \\ \frac{n+1}{2} & \text{If } n \text{ is odd} \end{cases}$$

The assumption is that each classifier makes a decision on an individual basis and is not influenced by any other classifier. The probabilities of various different

final decisions when $x + y$ classifiers are trying to reach a decision can be defined as: $(P_c + P_e)^{x+y}$ where P_c is the probability of each classifier making a correct decision and P_e is the probability of each making a wrong decision ($P_c + P_e = 1$) (Rahman & Fairhurst 2000).

Majority voting technique has the advantage that it creates a sense of decision census among the participating classifiers. Instead of the classifiers competing, the final decision is agreed by the majority, which allows for an overall moderation in the final decision (Bertolami & Bunke 2008). In short, for an ensemble of classifiers to produce a better solution than all of its members, it needs to have classifiers that are accurate and diverse. What is meant by accuracy is that a given classifier should have an error rate that is better than random guessing on new values. Diversity among classifiers can be defined as them making different errors on new data points (Dietterich 2000; Kuncheva & Whitaker 2003; Džeroski & Ženko 2004; Zhou 2012).

4.5. Principal Component Analysis

High-dimensional datasets have many instances and features which makes them very large datasets. The problem is not simply not having enough computing power to handle the data. The main issue is to make sense of the underlying structure in the data and to reach sensible conclusions about it, especially if there are hundreds of variables and thousands of individual observations involved.

PCA is therefore used to reduce the complexity and the available variables (features) to a much smaller and manageable set. The goal is to reduce the information to meaningful combination of variables without losing too much useful information (Wang 2012). In other words, PCA is a simple and non-parametric method for extracting relevant information from confusing datasets. During this process, the dimensionality of the dataset is also reduced (Shlens 2014). PCA is a data analysis technique which is used to identify some linear trends and simple patterns in datasets (Xanthopoulos et al. 2013).

The goals of PCA can be summarised as follows:

- It reduces the attribute space from a larger number of variables to a smaller number of factors and therefore it is considered a non-dependent procedure. This means that it does not assume that a dependent variable is defined.

- PCA reduces the data dimensionality and as such it is a data compression method. It aims to extract the most important data from the dataset (Abdi & Williams 2010), however when dimensionality reduction happens there is no guarantee that all resulting dimensions are interpretable.

Principle component analysis selects a subset of variables from a larger set of variables based on which variables have the highest correlations with the principal component. It identifies the most meaningful basis to re-express a dataset (Shlens 2014; Jolliffe & Cadima 2016).

In the pilot studies for this project, various attribute evaluators from the Weka software were employed in order to select attributes and reduce the dimensionality of our datasets. Some of these evaluators are CfsSubsetEval, InfoGainAttributeEval and OneRAttributeEval. However, these methods did not render any improvements and we then decided to use the Principal Component Analysis method. The goal of this project is to create a uniform protocol for all datasets and PCA behaved uniformly in all cases that it was applied to. One reason for its uniform behaviour could be that the datasets have very few outliers and in these conditions PCA is known to capture the most interesting part of the data variance (Zou et al. 2006).

Other more sophisticated approaches could include but are not limited to Recursive Feature Elimination (Guyon et al. 2002; Maldonado et al. 2014). The goal here is to find a subset of size r among n variables ($r< n$), eliminating those feature whose removal leads to the largest margin of class separation. The other method proposed by Yin et al. (2013) suggests a three phase framework where in phase K-mean clustering on class i (i=1,2,…,C) according to the user preset cluster number $K(i)$ to decompose the majority class into relatively balanced pseudo-subclasses. The labels of class i are replaced with the subclass labels provided by the K-means clustering. This way a multi-class dataset is formed with $\sum_{i=1}^{C} K(i)$ sub-classes. The pseudo-labels are acquired using the pseudo sub-classes. In phase 2, the measure of goodness of each feature is measured using the pseudo-labels and traditional measurements, and the features are ranked according to goodness based on the calculated scores. The top k good features are selected and the pseudo-labels are released to the original labels. In phase 3 classification can be done with the selected features.

4.6. Specific Methodology for Cheminformatics Data Screening

In this section, we are going to explain the different techniques used to assess the results of our novel method. As a reminder to the reader, in this study we deal with highly imbalanced datasets, some of which are extremely high dimensional. Usually the common remedy is to alter the classifiers and tailor them to the type of data being used. However, in our work we do not modify the used classifiers from their original states and settings. Instead we use the combination of utilising SMOTE together with various fingerprinting techniques and applying PCA.

To recap, the datasets used in this study were downloaded in the .sdf format. Using the PaDel and PowerMV software various fingerprints were developed for the said datasets. In addition to keeping the original imbalanced datasets as one sub-study (Sub1), the dimensionality of a copy of the same imbalanced datasets were reduced using PCA (Sub2). Then two separate options were used on both Sub1 and Sub2 in order to prepare them further for classification. In the first option (Option1), the datasets were first balanced using SMOTE and then they were split into training and test sets. In the second option (Option2), the datasets were first split (in a stratified manner) into training and test sets and afterwards only the training set was balanced using the SMOTE technique.

As a result, we obtain 6 separate states for all our datasets:

- Original imbalanced
- Balanced using Option1
- Balanced using Option2
- PCA imbalanced
- PCA balanced using Option1
- PCA balanced using Option2

The focus of data mining activity in this work is on classification. As mentioned before, four main classifiers were used for this study: J48 (Weka's implementation of C4.5), Random Forest, Naïve Bayes and SMO (Weka's implementation of Support Vector Machine).

Individual classifier approach:

In this approach, each pre-processed dataset will have 16 unique sub-datasets for classification (that is 8 different fingerprints and each fingerprint being binary only and binary plus numerical features). The number of generated features for the whole 16 sub-datasets can vary between 79 and 1056 depending on the fingerprint type used. A summary of these numbers has been shown in Table 4.

Ensemble classifier approach:

In this section, we have combined our four base classifiers in an attempt to investigate the effect this combination has on the classification accuracy of our datasets. An ensemble of classifiers is a set of classifiers whose individual decisions are combined by some method. Our method of choice for combining is majority voting, a robust approach in the case of heterogeneous solution spaces (Dietterich 2000; Murphree et al. 2015). In majority voting the predictions done by all classifiers for each instance in a dataset are counted (predictions can be correct or wrong) and the most predicted label is considered the final vote for that instance. If there is a tie between predictions then a label is randomly chosen.

Combining the predictions of multiple classifiers is more accurate than that of a single classifier. An ensemble of classifiers has stronger generalisation ability than a single classifier. A single learning algorithm searches a space of hypotheses in order to identify the best hypothesis in that space. If the amount of training data is too small compared to the size of the hypothesis, then the learning algorithm can find many hypotheses that give the same accuracy on the training data. By making an ensemble of the different accurate classifiers, their votes can be averaged and the risk of choosing the wrong classifier can be reduced. Sometimes even when there is enough training data it may still be computationally difficult for the learning algorithm to find the best hypothesis; the learning algorithm cannot guarantee finding the best hypothesis. If an ensemble is formed then a search for the hypothesis can be initiated from different starting points in the space (Dietterich 2000).

Instance-based approach:

This section with its experiments has been set up to observe how different fingerprinting techniques could affect the manifestation of the active (and structurally similar) compounds at the top of the dissimilarity ranking. This is

basically to observe how many of the compounds which rank the most similar to the query compound have the same effect (in our case the activity per class).

Fingerprinting Technique	# Binary Feat	# Binary + Numerical Feat
EState	79	111
Fingerprinter	1024	1056
Extended Fingerprinter	1024	1056
Graph-Only	1024	1056
MACCS	166	198
Pharmacophore	147	179
PubChem	881	913
Substructure	307	339

Table 13: Summary of the number of features generated by various fingerprinting techniques

For each of the fingerprints generated for a particular dataset, a Euclidean distance measure is calculated which will show how dissimilar each of the instances in the dataset are to a target molecule. The result can be sorted based on similarity or dissimilarity; in our case similarity was chosen. Once the results were in hand, the next step was to combine the datasets. The non-repeating combinations were done in groups of two, three, four … and eight. To find out how many non-repeating combinations this would result in, the formula in Figure 20 was used.

$$\frac{n!}{r!\,(n-r)!}$$

$$\frac{8!}{7!\,(8-7)!} = \frac{8*7*6*5*4*3*2*1}{7*6*5*4*3*2*1(1)} = \frac{40320}{5040} = 8$$

Figure 20: An example of how to calculate non-repeating combinations for a group of 7 fingerprints

Here n is the pool of the options to choose from which in our case is the number of fingerprints (8) and r is the number of unique combinations required, which in this case varies from 1 to 8. Thus, if for example we decide to select 7 unique fingerprints from the available 8 that would give us 8 unique non-repeating combinations.

ESt-Ext-Fin-Gra-MAC-Pha-Pub

ESt-Ext-Fin-Gra-MAC-Pha-Sub

ESt-Ext-Fin-Gra-Pha-Pub-Sub

ESt-Ext-Fin-Gra-MAC-Pub-Sub

ESt-Ext-Gra-Mac-Pha-Pub-Sub

ESt-Ext-Fin-MAC-Pha-Pub-Sub

ESt-Fin-Gra-MAC-Pha-Pub-Sub

Ext-Fin-Gra-MAC-Pha-Pub-Sub

Each sheet containing the generated fingerprints for each dataset has the instances on the rows (horizontally) and the features on the columns (vertically). Once the Euclidean distance measure is calculated for all fingerprints, the distance measures are added up according to the possible combinations. Extra attention should be paid when adding up to ensure that the measures from the corresponding instances are added up. Once the sheets are transposed, the instances will be on columns and the distance measures on rows. Therefore, each single instance can be sorted based on its distance measure from the target molecule.

From this we can observe how many of the instances that are similar to the target molecule are actually from the rare (positive) class and whether or not combining distance measures has increased the chance of these positive and similar instances to show up on top of the list; whether or not this would increase the accuracy of the method used. We calculated the positive and similar instances appearing in the top5, top10 and top20 of the combinations.

4.7. Summary of Data Mining Methods

In chapter we discussed data imbalance, what it means and is; reasons behind it and some of the consequences of imbalance in datasets during pre-processing and classification. The most common approaches for dealing with data imbalance were shown; and for our particular study we chose the SMOTE method from the oversampling technique. In order to evaluate imbalanced classification results we need to use class-specific metrics.

Then we delved into the different classifiers that we employed for our study; focusing on the specific implementation that yields an effective computational cost in high dimensional datasets. We then carried on by describing the various methods used in order to make our study more feasible and to reduce the computing cost of having to classify high dimensional imbalanced datasets.

The novelty of the methods used in this thesis lies in demonstrating empirically that the specific combination of oversampling SMOTE techniques together with classification provides a method valid for wide range of imbalance degrees; designed to be universally useable for the laboratory professional not expert in machine learning. This approach contrast with the alteration of inner settings of the classifiers in order to suit them to specific datasets i.e. specific level of imbalance; notwithstanding the strength of this approach in scenarios where dataset properties are known and the level of imbalance is expected to be relatively stable.

In the next chapter, we shall reveal the results from the various methods used for classifying these highly heterogeneous datasets.

5. Analysis of the Datasets

In previous chapters, we discussed the datasets used for this study. In chapter 3 we mentioned how and where the datasets were collected from and how various fingerprints were generated for them, creating 64 unique datasets for us to examine and perform experiments on. We also got familiar with the nature of the imbalance in the datasets and realised the extent of their dimensionality in the context of the number of features and instances they have. We will also briefly remind the reader about those factors in this chapter.

Chapter 4 described the data mining methods that were utilised to acquire the confusion matrix from classifying each one of the datasets. From the matrix we extracted true positives and false positives in order to assess the performance of the classifier used, towards the goal of out-of-sample testing our proposed unified approach on classifying potentially highly imbalanced high-dimensional datasets.

In this chapter, we present the analysis of the datasets used and we will show the results achieved from classifying the datasets together with visual aids in order to provide a better view of the results. As indicated earlier, the main challenge we face is the highly heterogeneous imbalance ratio between the datasets.

Datasets have been classified initially according to their original imbalanced state. Afterwards they have had their dimensionality reduced by applying the principle component analysis (PCA) and then classified again according to the literature in the area as discussed previously (section 4.4). With both the original state and the PCA-applied state, datasets have been classified using the following methods:

1. Whole datasets were balanced using the SMOTE technique and then split into training and test sets.
2. The datasets were split into training and test sets and then only the training set was balanced using the SMOTE technique.

In all datasets, the splitting into training and test sets was performed stratified, randomly and 30 times to achieve statistically sound results (May et al. 2010; Yuan et al. 2014; Zhou et al. 2016). In a stratified sampling one makes sure that the balance between the two classes in a sample of instances chosen is the same. In other

words, there are the same number of positive and negative classes available in the sample.

We initially start with our benchmarked dataset; the mutagenicity dataset. This dataset is the most balanced of all datasets used and has been used numerous times in various experiments (Ferrari & Gini 2010; Ferrari et al. 2012; Seal et al. 2012; Salama et al. 2014). The results for the Factor XA dataset will be shown afterwards. Then we will proceed to the more challenging datasets; AID362 and AID456 which have never been successfully analysed before.

Results shall be discussed from different angles:

- How well the methods performed compared to the original classification
- How well different fingerprints performed within the same method and across different methods used

There will also be a comparison between the different classifiers used for this study within the different methods utilised.

5.1. The Benchmark Dataset

As mentioned in chapter 3, this dataset was prepared by Bursi and co-workers (Kazius et al. 2005) in order to identify sub-structures that could help classify whether unseen test molecules were mutagenic. The dataset prepared for this study has a total of 4893 instances of which 2556 are active (mutagens) and 2337 are inactive (non-mutagens). Table 1 summarises the number of instances present in the training and test set of this dataset.

Dataset	**#Total Instances**	**#Active Instances (1)**	**#Inactive Instances (0)**	**Active/Inactive Ratio**
Mutagenicity	4893	2556	2337	1.09

Table 14: Mutagenicity dataset specification. Class of interest labelled as 1.

This dataset according to He and Ma (2013) is only a marginally imbalanced dataset. As seen in the above table that the ratio of active to inactive is very close to 1 (almost balanced ratio). In the next section, we classify the original dataset and show the classification metrics used.

Bursi Classification Results per Fingerprint used – Original Dataset

In this section, the *Bursi* dataset was exposed to our four chosen classifiers; Naïve Bayes, J48, Random Forest and SMO (Weka's specific implementation SVM). In the case of the SMO various kernels available with Weka were used and the results were similar regardless of the chosen kernel. Therefore, the default linear kernel was used for this study. Each of these classifiers learn the training set and create a model which then is applied to the unseen test set.

In the first part of this section we look at the classification metrics for each of the used fingerprints per classifier. In the graphs the bars represent the classification metrics; sensitivity, specificity, false positive, false negative and accuracy. The standard deviation for each bar is situated on top of the bar as a capped thinner bar. First we look at the results from J48.

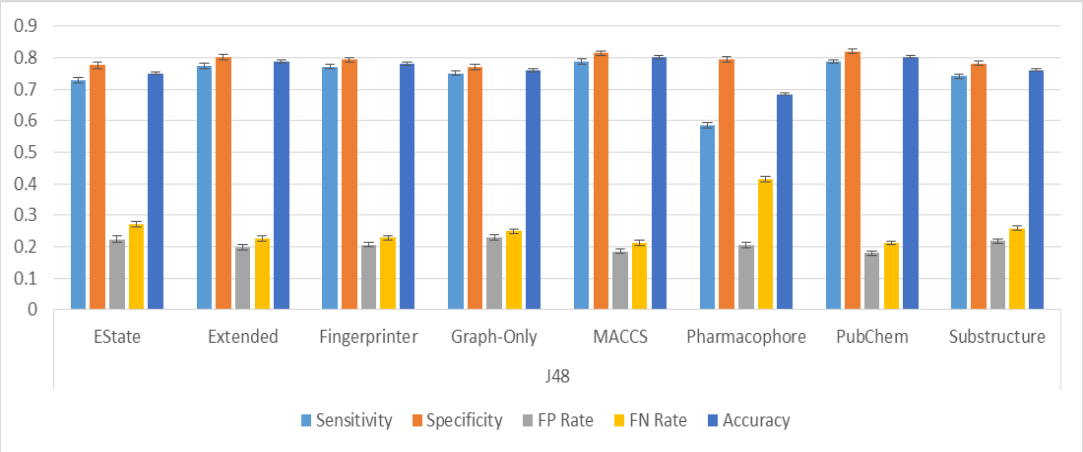


Figure 21: Classification results from classifying the Bursi dataset by J48.

In Figure 21 we see that with almost all of the fingerprints (except for Pharmacophore), there is a high percentage of true positive and true negative rate. False positive (FP) and false negative (FN) rates are particularly low, with false negatives slightly above false positives, albeit non- significantly (FP vs FN, pairwise T test, $p>0.05$). This condition is preferred since for example in a critical situation such as medical diagnosis, diagnosing healthy patients wrongfully as sick patients is better than diagnosing sick patients as healthy. The fingerprints MACCS and PubChem have performed best on this dataset with the highest sensitivity, specificity and accuracy and the lowest false positive and false negative of all eight fingerprints used.

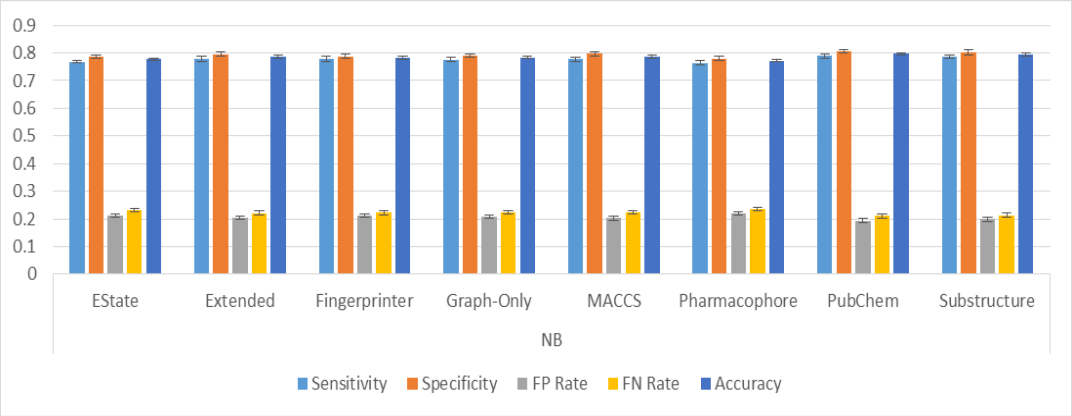


Figure 22: Classification results from classifying the Bursi dataset by Naïve Bayes.

The results from NaïveBayes (Figure 22) show better sensitivity and specificity rates and lower, more stable false positive and negative among all of the fingerprints.

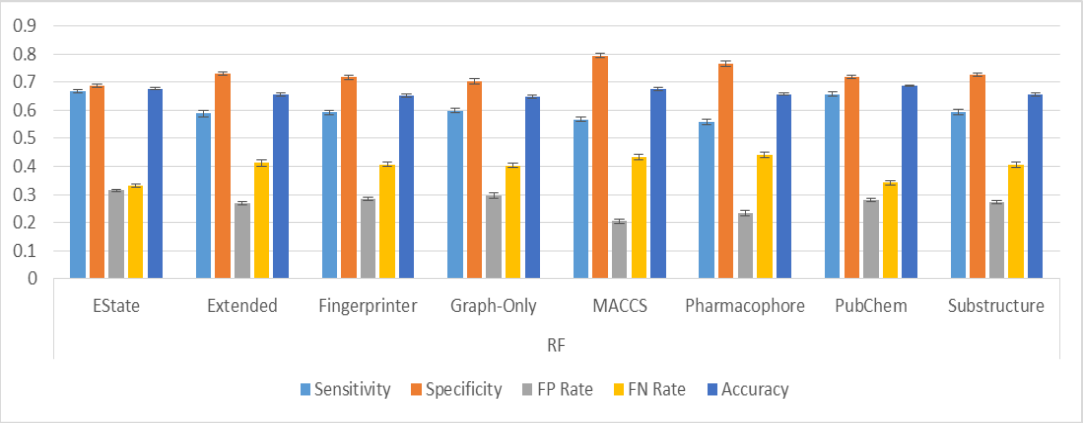


Figure 23: Classification results from classifying the Bursi dataset by Random Forest.

The results produced by Random Forest are not all at the same level. MACCS and Pharmacophore are the two fingerprints producing the better results where the false positive and false negative results are lower than the other fingerprints, despite having slightly lower sensitivity and specificity.

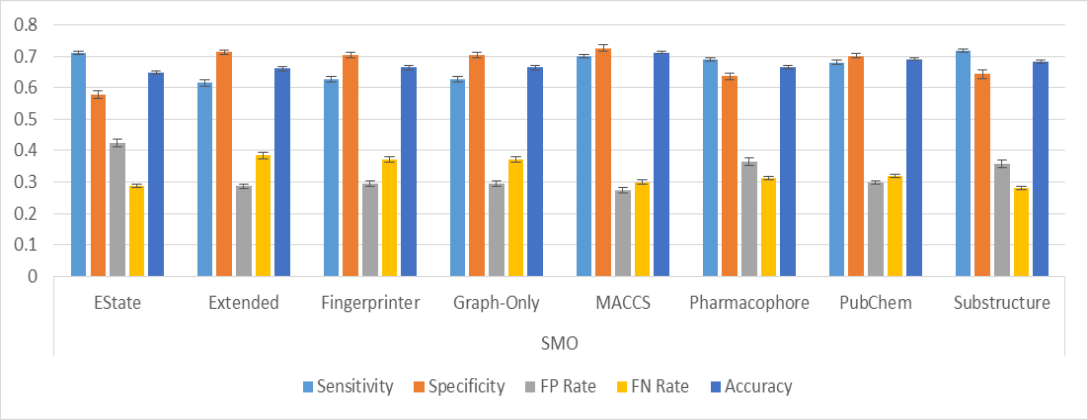


Figure 24: Classification results from classifying the Bursi dataset by SMO.

Next, the results from classifying the datasets with the classifier SVM/SMO are shown in Figure 24. From observing the graph we see that MACCS and PubChem have yet again produced better results.

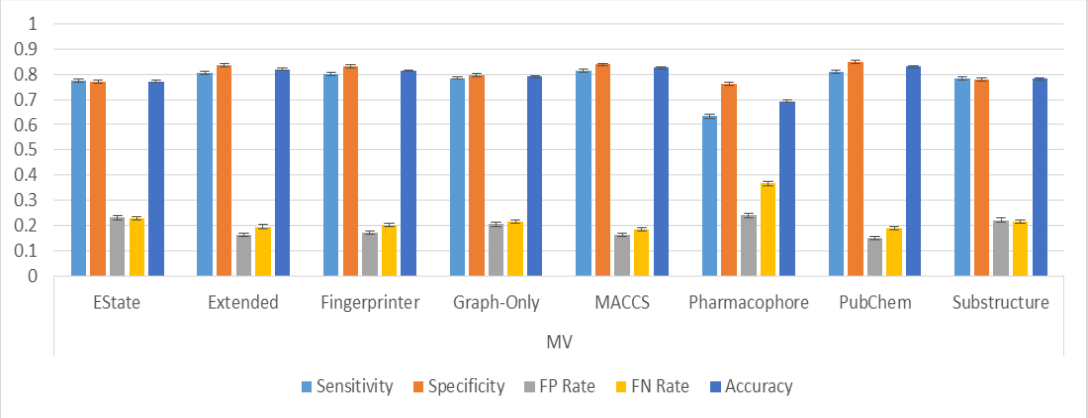


Figure 25: Classification results from classifying the Bursi dataset by Majority Voting.

To conclude the analysis, the results from Majority Voting present yet the best out of all classifiers. As mentioned before, in Majority Voting, the power of multiple models is leveraged in order to achieve better accuracy levels than the individual models could have achieved on their own. We observe this effect in Figure 25. By comparing to the other 4 figures shown before in this section (Figures 21-24), we can see the here we have the highest sensitivity and specificity and accuracy levels and the lowest false positive and false negative between the classifiers used. A summary of such results and a discussion on the criterion for the best approach is at the end of this chapter. In the next section, we will observe how adding numerical fingerprints affects our classification results with the original dataset and whether the changes are statistically significant or not.

Analysis of the Improvement with Numerical Fingerprints

The results reported above stem from applying classifiers to binary fingerprints. In our study, we have included numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results are shown below. With the metrics sensitivity, specificity and accuracy, if the difference of the two numbers is a positive number, then that is considered an improvement (a green arrow pointing upwards) and if the difference is negative then it is considered not to have improved (a red arrow pointing downwards). With false positive and false negative it is the other way round. That means that if the resulting number is a negative number, then that mean that these metrics have become smaller and we have less of them occurring, resulting in an improvement (green arrow pointing down).

The summary of results from adding the numerical fingerprints are shown in Figures 26-30. In these figures which relate to the results of adding numerical fingerprints, whilst the improvement and non-improvement is shown with the arrows. The significance of this change is calculated by utilising a standard two tailed t-test (since normality was verified in all cases, Lilliefors test $p<0.001$) and illustrated with the help of asterisks (*if the resulting change is less than 0.01 but bigger than 0.001, ** if the change is less than 0.001). The significant results have been made bold to make them clearer.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑*	↓*	↓**	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑**	↑**	↓**	↓**	↑**
MACCS	↑**	↑**	↓**	↓**	↑**
Pharmacophore	↑**	↑*	↓*	↓**	↑**
PubChem	↑	↑*	↓*	↓	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 26: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↓**
Extended	↑**	↓**	↑**	↓**	↑
Fingerprinter	↑**	↓	↑	↓**	↑**
Graph-Only	↑**	↓	↑	↓**	↓**
MACCS	↑**	↓**	↑**	↓**	↑**
Pharmacophore	↓**	↓**	↑**	↑**	↓*
PubChem	↑**	↓**	↑**	↓**	↑
Substructure	↑**	↓**	↑**	↓**	↓**

Figure 27: Results from adding numerical fingerprints to binary fingerprints for Naïve Bayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↑	↑**	↓**	↓	↑*
Fingerprinter	↑	↑**	↓**	↓	↑**
Graph-Only	↑**	↑**	↓**	↓**	↑**
MACCS	↑	↓**	↑**	↓	↑**
Pharmacophore	↑**	↓**	↑**	↓**	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 28: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑**	↑**	↓**	↓**	↑**
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↓**	↑	↓	↑**	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 29: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↑	↓	↑	↓	↑**
Fingerprinter	↑	↓*	↑*	↓	↑*
Graph-Only	↑**	↑**	↓**	↓**	↑**
MACCS	↑**	↓*	↑*	↓**	↓
Pharmacophore	↑**	↑	↓	↓**	↑**
PubChem	↑*	↓	↑	↓*	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 30: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

In summary, result shown in Figures 26-30 show a complex scenario, in the sense that the no particular fingerprint has consistently performed the best. Of

course, the settings have been different due to the classifiers used. But in general (except for when Naïve Bayes is used), metrics have improved with the addition of numerical fingerprints. In the next section, we classify the original dataset and show the classification metrics used.

Bursi Classification Results per Classifiers Used – Original Dataset

In this section, we look in more detail at the classification results per fingerprint used and for each classifier. We want to observe with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

Each chart belongs to one specific fingerprint and shows the five main classification metrics used for this study and for each of those there will be five bars corresponding to each classifier.

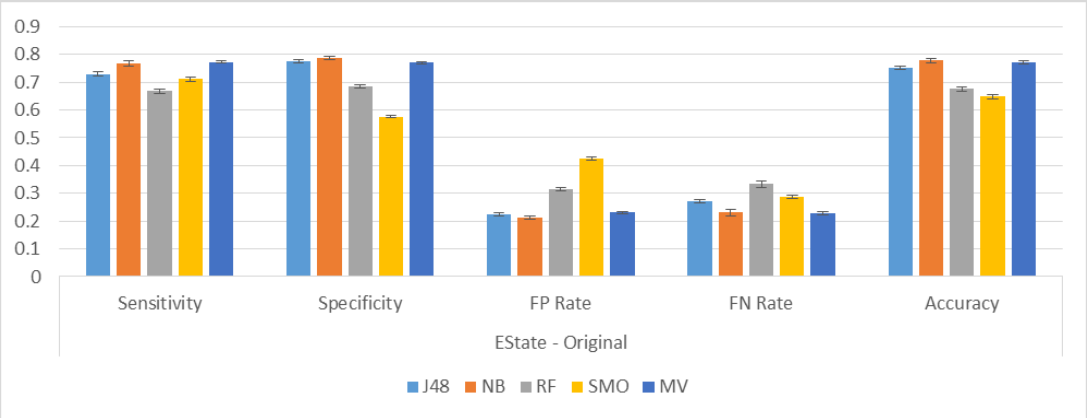


Figure 31: Classifier performance for EState – Original

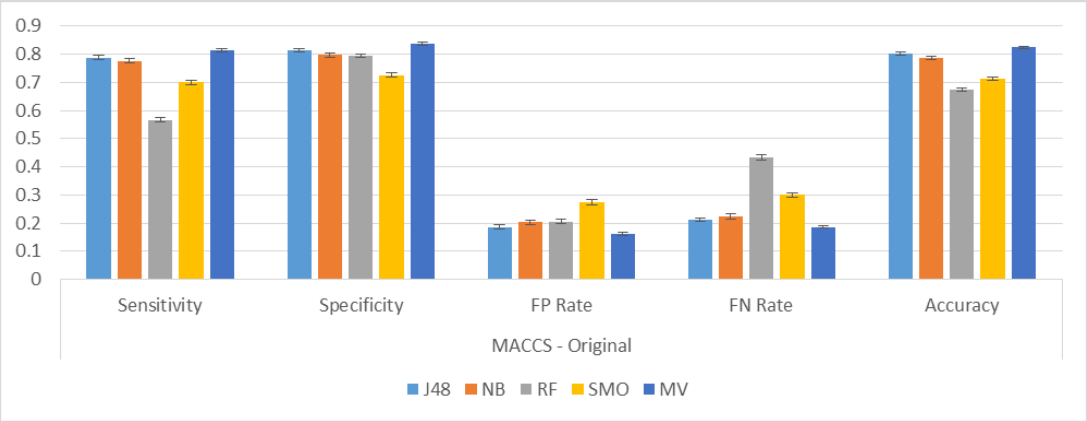


Figure 32: Classifier performance for MACCS – Original

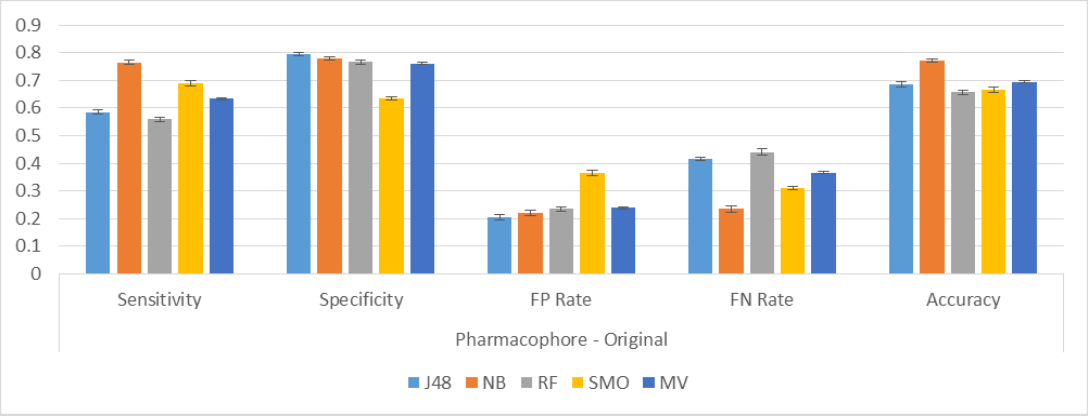


Figure 33: Classifier performance for Pharmacophore – Original

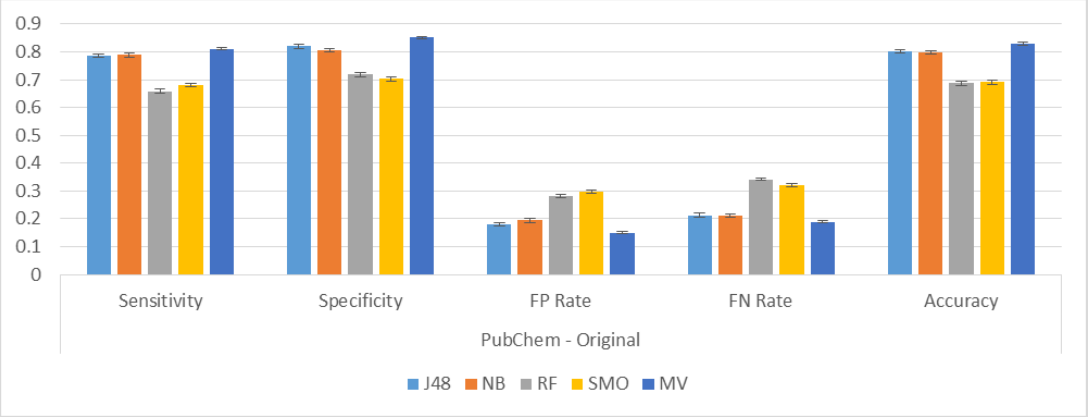


Figure 34: Classifier performance for PubChem – Original

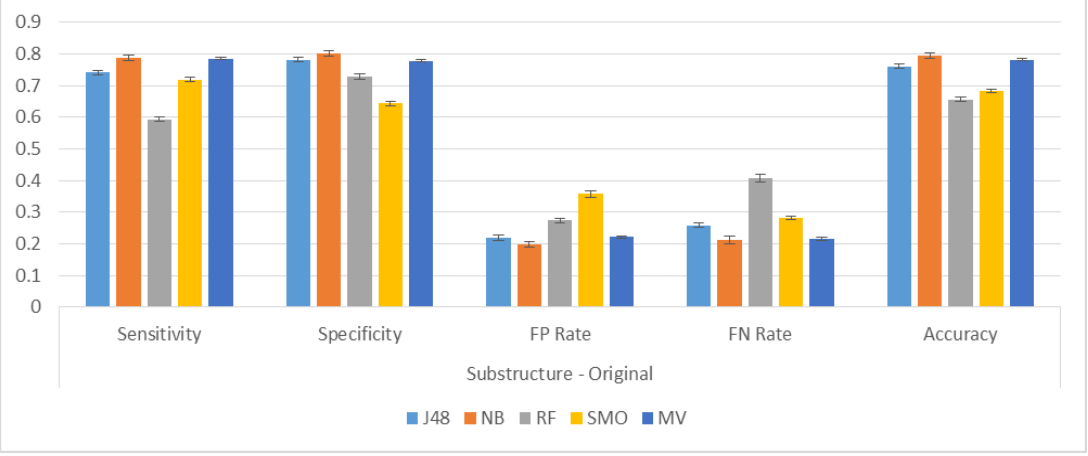


Figure 35: Classifier performance for Substructure – Original

From observing Figures 31-35 is evident that Majority Voting has consistently performed better once more than all the other classifiers for this dataset. In the next section, we will observe how adding numerical fingerprints affects our classification results and whether the changes are statistically significant or not.

Analysis of the Improvement with Numerical Fingerprints

In the next few figures we shall see the result of adding numerical fingerprints to binary fingerprints and how that has affected the performance of our classifiers.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑*	↓*	↓**	↑**
NB	↓**	↑**	↓**	↑**	↓**
RF	↓**	↑**	↓**	↑**	↑**
SMO	↑	↑	↓	↓	↑
MV	↑**	↑**	↓**	↓**	↑**

Figure 36: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↑**	↓**	↑**	↓**	↑**
RF	↑	↓**	↑**	↓	↑**
SMO	↑	↑	↓	↓	↑
MV	↑**	↓*	↑*	↓**	↓

Figure 37: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑*	↓*	↓**	↑**
NB	↓**	↓**	↑**	↑**	↓*
RF	↑**	↓**	↑**	↓**	↑**
SMO	↓**	↑	↓	↑**	↑**
MV	↑**	↑	↓	↓**	↑**

Figure 38: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑*	↓*	↓	↑
NB	↑**	↓**	↑**	↓**	↑
RF	↑	↑	↓	↓	↑
SMO	↑	↑	↓	↓	↑
MV	↑*	↓	↑	↓*	↑

Figure 39: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↑**	↓**	↑**	↓**	↓**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 40: Results from adding numerical fingerprints to binary fingerprints for Substructure

The results here show that, as a direct consequence of adding the numerical fingerprints, classifier performance has improved greatly in the cases of PubChem and Substructure, as will be discussed below. In this next section, the PCA feature selection method is applied and the original dataset is classified and we see the metrics used.

Bursi Classification Results per Fingerprint used – PCA Original

In this section, we analyse how applying the PCA method to our dataset affects the classification metrics and classifier performance when looked at from the point of view of the fingerprints used and from the classifiers' aspect. This is a common approach in the cheminformatics dataset analysis (similar results are typically obtained with other dimensionality reduction methods) (Zou et al. 2006; Maji et al. 2013; Bro & Smilde 2014). The comparison with the coloured bars show the classification metrics and standard deviation is shown as capped thinner bars on top of the coloured bars.

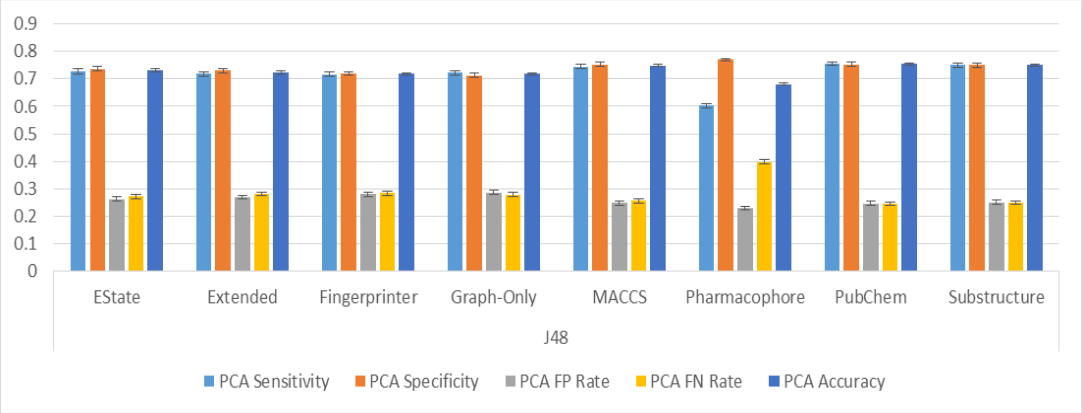


Figure 41: Classification results from classifying the Bursi dataset by J48 – PCA

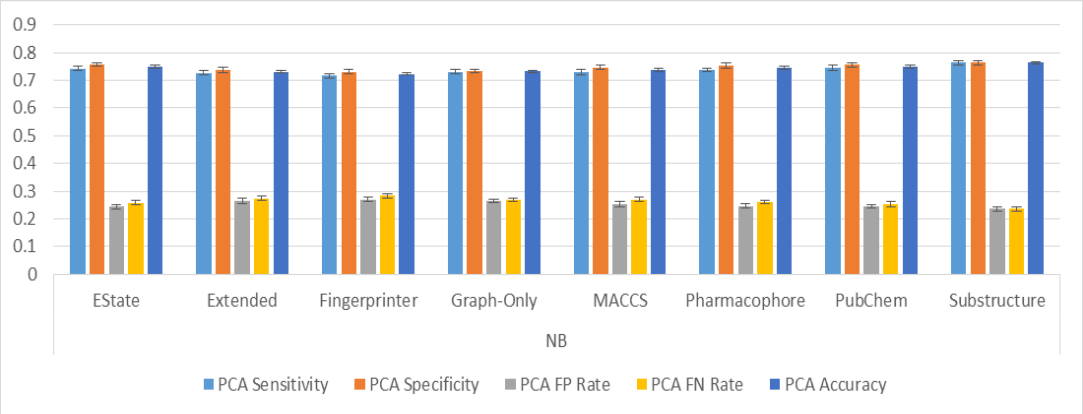


Figure 42: Classification results from classifying the Bursi dataset by Naïve Bayes – PCA

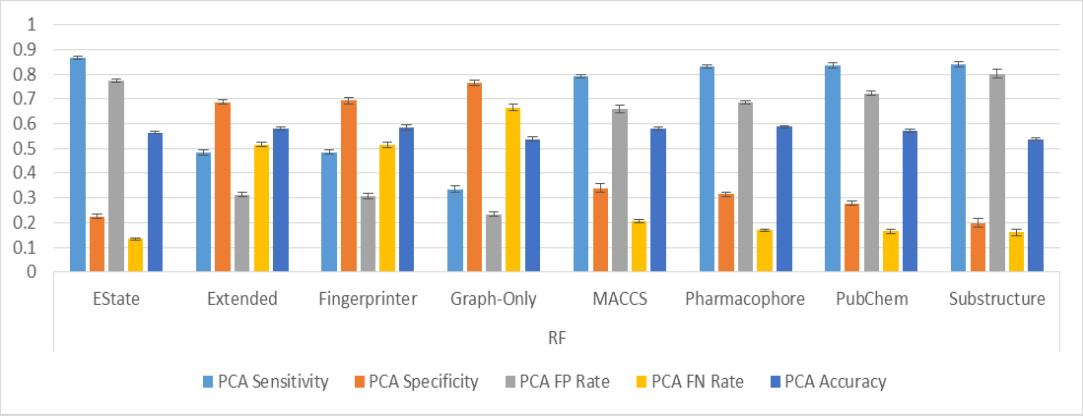


Figure 43: Classification results from classifying the Bursi dataset by Random Forest – PCA

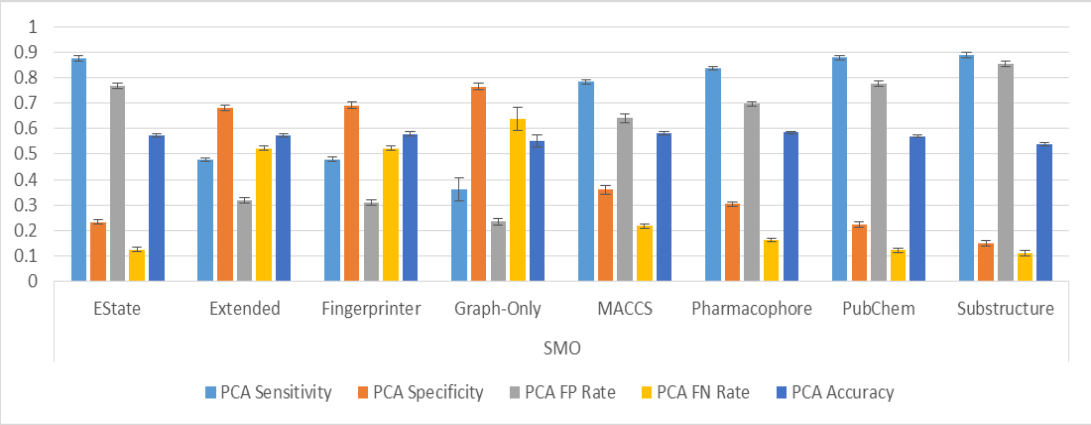


Figure 44: Classification results from classifying the Bursi dataset by SMO – PCA

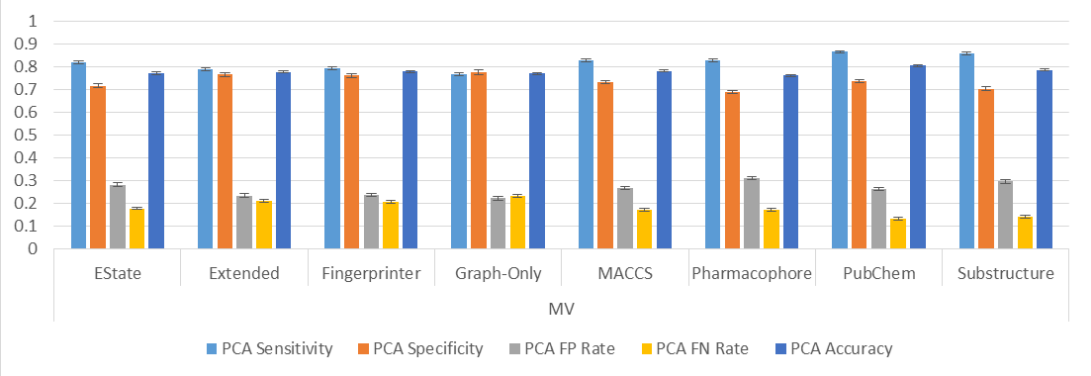


Figure 45: Classification results from classifying the Bursi dataset by Majority Voting – PCA

Figures 41- 45 show that in the cases of J48, Naïve Bayes and Majority Voting, the fingerprints have performed very well keeping sensitivity and specificity high and false positive and false negative low and very consistently. As indicated in Chapter 4, section 4.4 under Random Forest, it seems that this classifier (RF) is typically more sensitive than SVMs and ensemble approaches to high variance in certain fingerprints, which may be due to intrinsic characteristics of the algorithm which in

our datasets often provides a flexible "more optimistic" but on occasion less robust solution than the other classifiers, like in these alluded figures. The conclusions steaming for this result is that Random Forest is not always the optimal approach as seen in Figure 32. Yet in some scenarios it is still the optimal one per the metrics we have defined in the summary Figures that will be discussed in following chapters (62, 184 and 254). However, the overall differences between the three optimal classifiers are small and therefore this has no significant consequences in the robustness of the proposed protocol as will be discussed in the following chapters. In the next section, we will observe how adding numerical fingerprints affects our classification results and whether the changes are statistically significant or not.

Analysis of the Improvement with Numerical Fingerprints

Similar to the analysis performed on the original datasets; by adding numerical fingerprints, we endeavour to see the effects that this action has on the performance of our fingerprints in the case of each classifier. This has been shown in Figures 46-50. As a gentle reminder for our readers, a green arrow upwards means the same as a green arrow downwards with the difference that with sensitivity and specificity a green arrow upwards means an improvement in the metric; the number has grown and there's a positive difference when adding numerical features.

In the case of false positive and false negative a green arrow downwards is the sigh of improvement meaning that the numbers have become smaller and we have less of each metric. The significance of the change (difference) is shown using asterisks and calculated by using two-tailed t-test assessment (normality accepted at $p>0.05$, Lilliefors test).

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑*	↑**	↓**	↓*	↑**
Extended	↓	↑	↓	↑	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↑*	↑	↓	↓*	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 46: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↓	↑	↑	↓*
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓	↑	↓	↑
Pharmacophore	↓	↓	↑	↑	↓
PubChem	↑**	↑**	↓**	↓**	↑
Substructure	↓**	↓**	↑**	↑**	↓

Figure 47: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↓	↑**	↓**	↑	↑
Pharmacophore	↓**	↑**	↓**	↑**	↑**
PubChem	↓**	↑	↓	↑**	↑
Substructure	↓	↑**	↓**	↑	↑**

Figure 48: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↑	↑	↓	↓	↑*
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑**	↓	↑	↓**	↑**
MACCS	↑*	↑	↓	↓*	↑**
Pharmacophore	↓**	↑	↓	↑**	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↓**	↑*	↓*	↑**	↑**

Figure 49: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓**	↑**	↓**	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑**	↑*	↓*	↓**	↑**
MACCS	↑**	↓	↑	↓**	↓
Pharmacophore	↑**	↓**	↑**	↓**	↑**
PubChem	↑**	↓*	↑*	↓**	↑
Substructure	↑**	↓**	↑**	↓**	↑**

Figure 50: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

By observing Figures 46-50 one can see that J48, Random Forest and SMO show great improvements in specificity and lower rates of false positives. Majority

voting, once again, shows improvements in sensitivity and false negative rates. In the next section, we classify the original dataset and show the classification metrics used. Note that PCA was applied to the dataset.

Bursi Classification Results per Classifiers Used – PCA Original

In this section, we present the classification results for the Bursi dataset explored from a fingerprint-classifier relationship side; in other words, which classifier performed better at the presence of an individual fingerprint. In the previous section, we explored the opposite; we wanted to asses which fingerprint performed better in the presence of a single classifier.

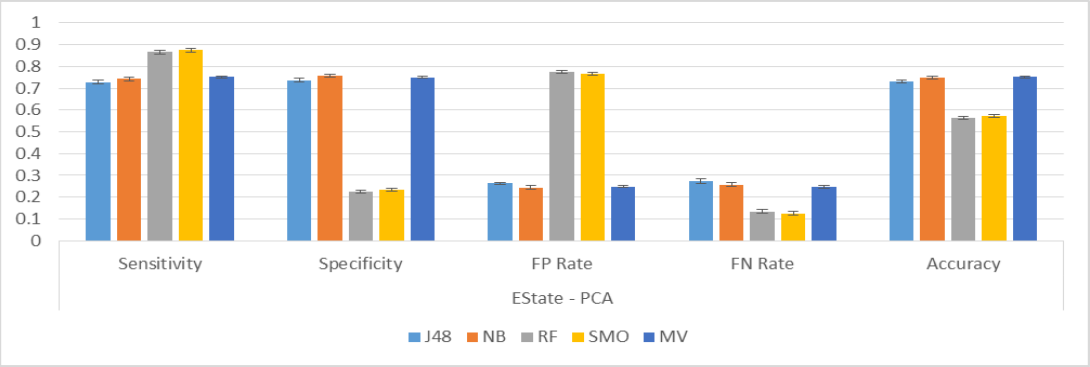


Figure 51: Classifier performance for EState – PCA

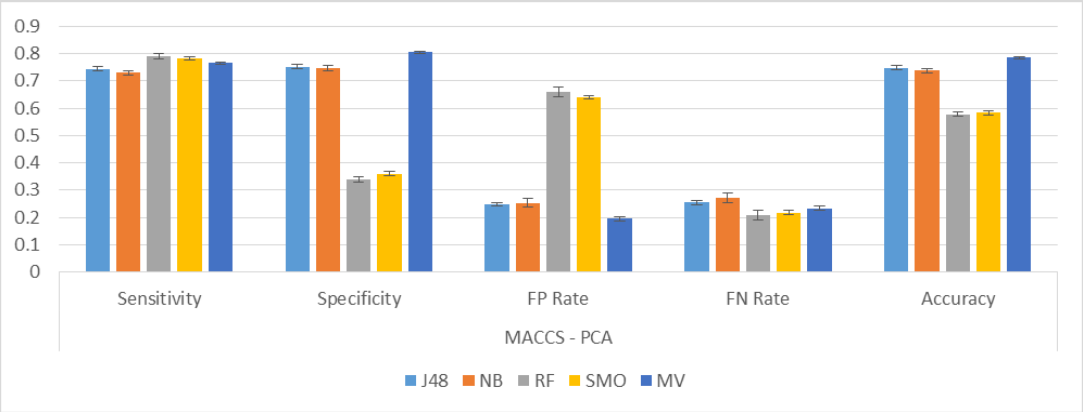


Figure 52: Classifier performance for MACCS – PCA

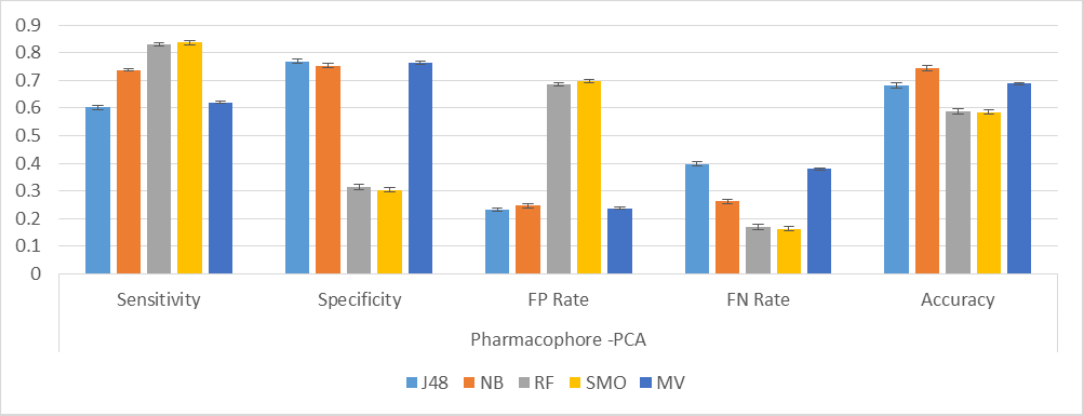


Figure 53: Classifier performance for Pharmacophore – PCA

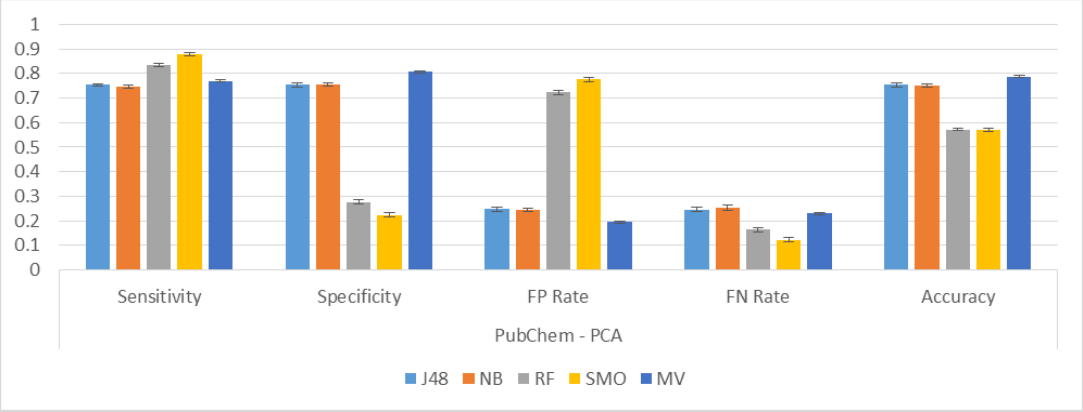


Figure 54: Classifier performance for PubChem – PCA Applied

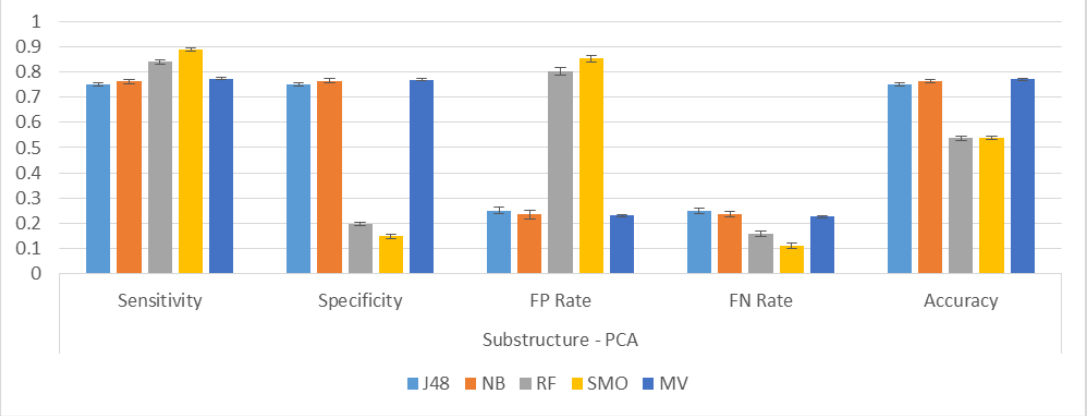


Figure 55: Classifier performance for Substructure – PCA Applied

After studying Figures 51-55 we see that PubChem, Substructure and MACCS have the better results. The classifiers that performed better than others appear to be SMO, Random Forest and Majority Voting. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

As with other sections we have added numerical features to our fingerprints in order to observe the difference in performance. J48 has constantly delivered the best results throughout the five fingerprints shown in Figures 56-60. Naïve Bayes has good results when used with PubChem.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑*	↑**	↓**	↓*	↑**
NB	↓	↓	↑	↑	↓*
RF	↓	↑**	↓**	↑	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↑**	↓**	↑**	↓**	↑**

Figure 56: Results from adding numerical fingerprints to binary fingerprints for EState – PCA

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑*	↑	↓	↓*	↑**
NB	↑	↓	↑	↓	↑
RF	↓	↑**	↓**	↑	↑
SMO	↑*	↑	↓	↓*	↑**
MV	↑**	↓	↑	↓**	↓

Figure 57: Results from adding numerical fingerprints to binary fingerprints for MACCS – PCA

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↓	↓	↑	↑	↓
RF	↓**	↑**	↓**	↑**	↑**
SMO	↓**	↑	↓	↑**	↑**
MV	↑**	↓**	↑**	↓**	↑**

Figure 58: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore – PCA

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑**	↑**	↓**	↓**	↑
RF	↓**	↑	↓	↑**	↑
SMO	↓	↑	↓	↑	↑
MV	↑**	↓*	↑*	↓**	↑

Figure 59: Results from adding numerical fingerprints to binary fingerprints for PubChem – PCA

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↓**	↓**	↑**	↑**	↓
RF	↓	↑**	↓**	↑	↑**
SMO	↓**	↑*	↓*	↑**	↑**
MV	↑**	↓**	↑**	↓**	↑**

Figure 60: Results from adding numerical fingerprints to binary fingerprints for Substructure – PCA

Summary of the results and receiver operating characteristics analysis

The next figure summarises previous observations for the mutagenicity dataset. Figure 61 shows the Sensitivity versus false positives for the different classifiers and averaged across Fingerprints with and without using numerical descriptors. A possible criterion for selection of the best classifier is simply the one that is closest to the top left corner on the graph, which is reminiscent of the ROC analysis i.e. min Euclidian distance to (0,1).

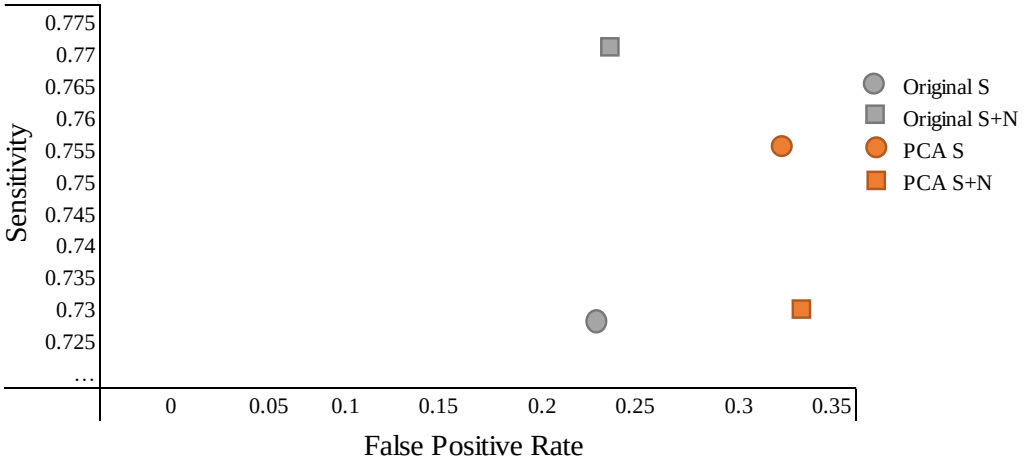


Figure 61: Sensitivity versus False Positive rate for the methods used on the Mutagenicity dataset.

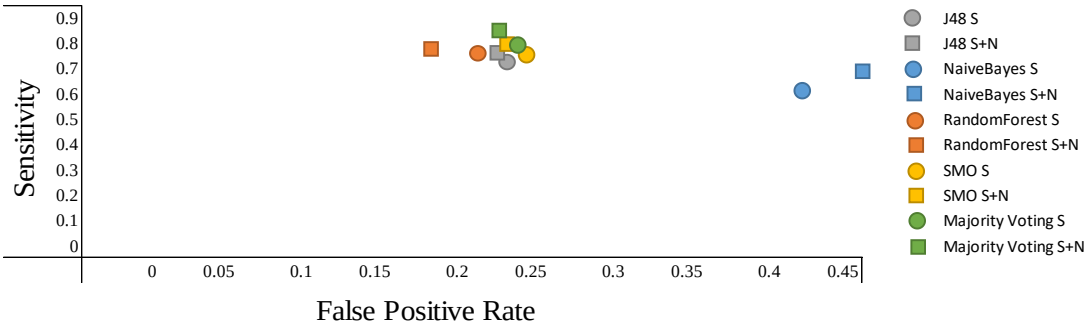


Figure 62: Sensitivity versus False Positive rate per classifier for the Mutagenicity dataset.

S and S+N in figures 60 and 61 indicate structural and structural plus numerical descriptors respectively.

Table 15 contains the Euclidean distances calculated for both figures. This calculation is based on the distance each point on the graph has from the top left corner, the point with the coordinates (0,1). The point with the least distance to the coordinates (0,1) is considered the more optimal choice (method or classifier).

Methods Used		**Euclidean Distance**
Binary Descriptors	**Original**	**0.3502**
	PCA	0.4138
Binary + Numerical Descriptors	**Original**	**0.3193**
	PCA	0.3937

Table 15: Euclidean distance for the methods used

Classifiers Used		**Euclidean Distance**
Binary Descriptors	J48	0.3571
	NaïveBayes	0.5479
	Random Forest	0.3249
	SMO	0.3509
	Majority Voting	0.3254
Binary + Numerical Descriptors	J48	0.3344
	NaïveBayes	0.5491
	Random Forest	**0.2877**
	SMO	0.3244
	Majority Voting	**0.2934**

Table 16: Euclidean distance for the classifiers used

On average, the Majority voting classifier and Random Forest are the optimal classifiers according to this criterion, as it is suggested by the previous figures and tables.

Perhaps the most interesting aspect to stress is that no significant average improvement is observed for the best classifier, the Majority Voting ($p>0.05$ in all pairwise comparisons) and hence, on average, the data evenly populating the space spanned by fingerprints contains sufficient information for a standard approach to perform a successful classification. This is not surprising giving the balanced characteristics of the dataset, which is used here merely as a benchmark.

Conclusion

In this chapter we observed the results for classifying the mutagenicity dataset. The classification results were discussed from the aspect of the fingerprints used and the classifiers used. We essentially looked at how different fingerprints performed in the presence of each classifier and then how different classifiers performed when looked at the presence of each single fingerprint. Afterwards we looked at how adding numerical fingerprints to binary fingerprints affects classifier performance and classification metrics. All this was studied with the dataset at its original state and when PCA was applied.

Initially we saw that in the presence of each single classifier, the fingerprints behaved differently. There was no consistent better-performing fingerprint that could be pointed out. But as a generalisation, the fingerprints PubChem and MACCS seemed to perform better than the rest for this dataset in its original state. When we looked at the classifier performance in the presence of each fingerprint, the one classifier which stood out was Majority Voting.

The application of PCA in this case did not affect the performance of the classifiers as much as anticipated. J48 and Naïve Bayes were the two consistent performers and Majority Voting produced the better results, yet the mean improvement across fingerprints was not significant

Adding numerical fingerprints did affect the classification metrics positively in many situations, however in some cases it did worsen our results and again statistical significance was not concluded on average. This should be discussed on a specific fingerprint or classifier level and cannot be generalised to the whole study.

Results of this benchmark, nearly fully balanced, dataset indicate that despite its complexity, a classical approach consisting of data management and pre-processing followed by any competitive classification approach directly operating in the original space of the data (i.e. the fingerprints) would suffice. Hence, the critical bottleneck for the standard approach seems to be not in the dimensionality of the space i.e. the number of fingerprints but rather specifically on how imbalanced they are.

The challenge we address in the next chapters is to discern whether similarly competitive results can be obtained in general regardless of the imbalance degree. We will also investigate which are the steps that have to be present in order to devise a systematic screening approach valid for all datasets.

5.2. The Slightly Imbalanced Dataset

In the previous section we studied our almost balanced dataset, the mutagenicity dataset. In this section we investigate the Factor XA dataset (Fontaine et al. 2005). The data in this dataset were used to discriminate between Factor XA inhibitors of high and low activity. Since the dataset includes molecules from diverse chemical classes, the objective in the main study by Fontaine et al. (2005) was to produce a discriminant model which is potentially useful for screening molecular libraries.

Dataset	#Total Instances	#Active Instances (1)	#Inactive Instances (0)	Active/Inactive Ratio
Factor XA	435	279	156	1.79

Table 17: Factor XA dataset specification. Class of interest labelled as 1

This dataset has an imbalance ratio of 1.79 indicating there is a clear imbalance between the classes (Table 17). We shall employ additional pre-processing techniques in order to balance this dataset and investigate the effect it has on or classification metrics. Thereafter the dimensionality of the dataset will be reduced using the Principle Component Analysis (PCA) method and again the pre-processing and balancing techniques will be applied so we can see the results.

As a gentle reminder to the reader, the datasets are taken in their tabular form and with the help of software such as PowerMV (Liu et al. 2005) and PaDel (Yap 2011), descriptors are generated for them. Afterwards the newly populated datasets take a journey through two options:

- Option 1: the imbalance in the dataset is altered by using the SMOTE (Chawla 2005) technique, by generating synthetic samples for the minority class. Afterwards the resulting balanced dataset is split into training and test set, 60% and 40% accordingly. As mentioned above this splitting is done in a stratified manner so all resulting sets have the same proportion of the classes. Plus this operation is performed 30 times to ensure the resulting sets are representative of the original population.
- Option 2: here, at first the imbalanced dataset is split according to the procedure mentioned in route 1, and thereafter only the training set is subjected to the balancing technique. The test set is kept in its original imbalanced state.

As a result there will be six different sets of the same dataset available to us for classification (as previously mentioned in section 4.6):

1. The dataset in its original state (referred to as original in text)
2. The dataset that has been balanced and then split into training and test set (referred to as original *SMOTEd All* in text)
3. The dataset that has been split into training and test set and then only training set has been balanced (referred to as original *SMOTEd Training* in text)
4. The original dataset with reduced dimensionality (referred to as PCA in the text)
5. The dataset that has been balanced and then split into training and test set with reduced dimensionality (referred to as *PCA SMOTEd All* in text)
6. The dataset that has been split into training and test set and then only training set has been balanced with reduced dimensionality (referred to as *PCA SMOTEd Training* in text)

Once the pre-processing procedures have completed the datasets are ready to be classified using our chosen classifiers; J48, NaïveBayes, Random Forest, SMO and Majority Voting. In the next few sections we will look at these results using graphs and charts and will make comparison between different methods and classifiers used.

Factor XA Classification Results per Fingerprint– Original

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

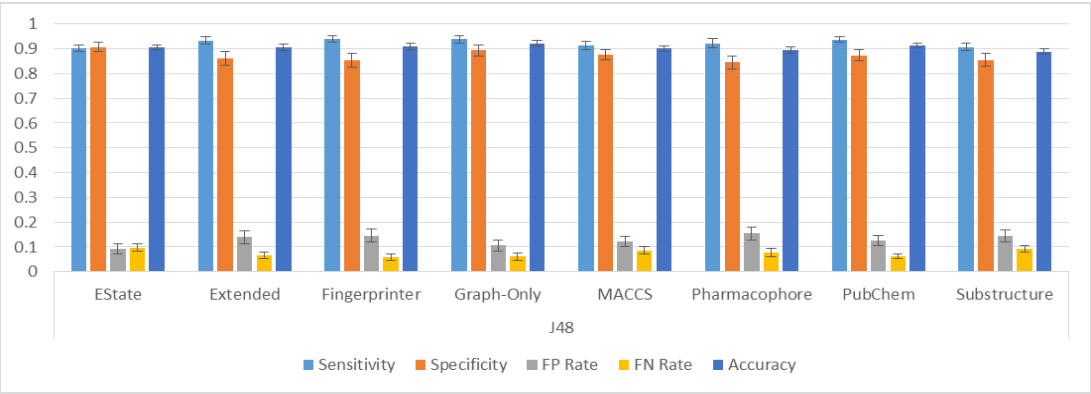


Figure 63: Classification results from classifying the Fontaine dataset by J48

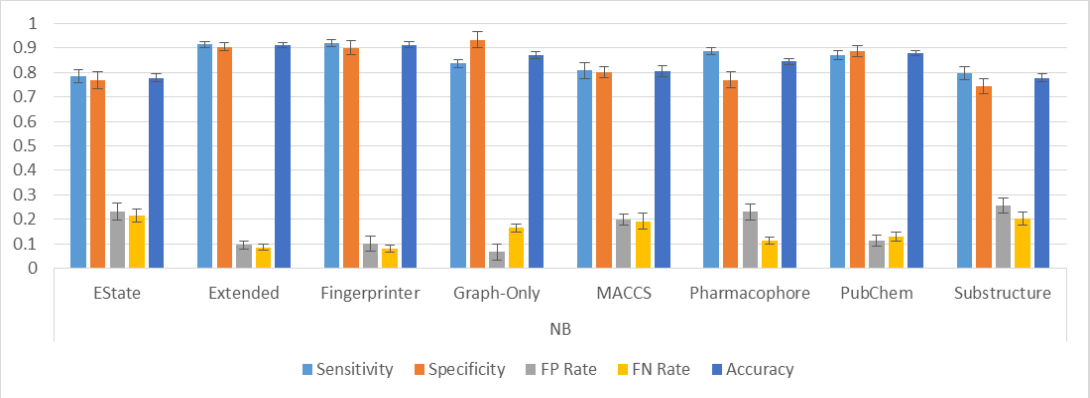


Figure 64: Classification results from classifying the Fontaine dataset by NaïveBayes

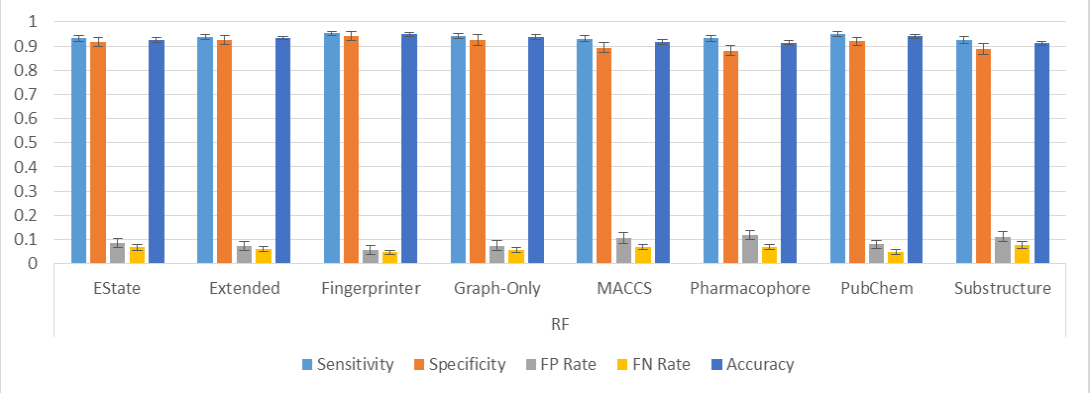


Figure 65: Classification results from classifying the Fontaine dataset by Random Forest

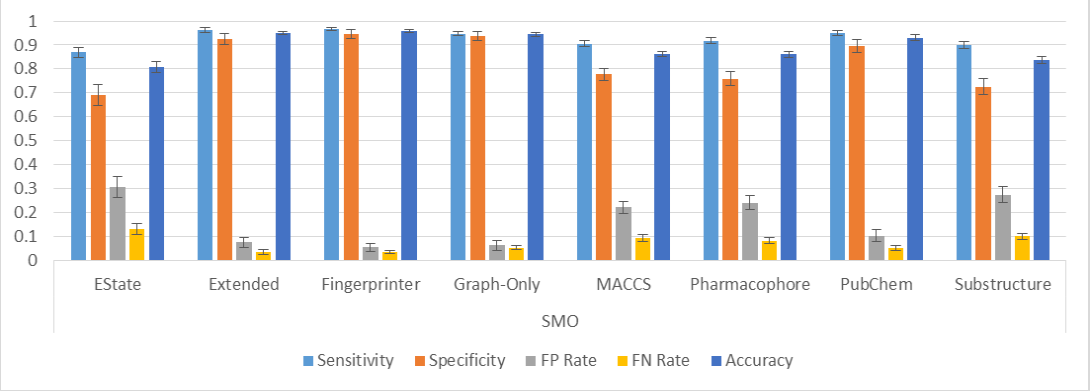


Figure 66: Classification results from classifying the Fontaine dataset by SMO

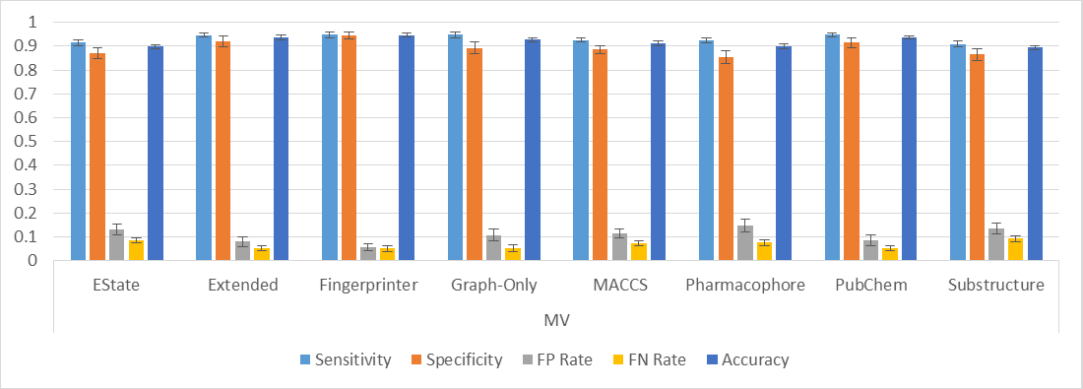


Figure 67: Classification results from classifying the Fontaine dataset by Majority Voting

By looking at previous Figures (63-67) we see that the fingerprints used have produced consistent high sensitivity and specificity and low false positive and negative rates when used with J48, Random Forest and Majority Voting. The CDK Fingerprint family (Steinbeck et al. 2003; Kristensen et al. 2010), Fingerprinter, Extended Fingerprinter and Graph-Only, have a standard fingerprint size of 1024 and produce very similar results to one another. Next, we observe how adding numerical fingerprints affects our classification results with the original dataset.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓**	↑**	↓	↓
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↓	↓	↑	↑	↓
MACCS	↑	↓	↑	↓	↑
Pharmacophore	↑	↑	↓	↓	↑
PubChem	↓	↓	↑	↑	↓
Substructure	↑	↑	↓	↓	↑

Figure 68: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑	↓	↑	↑
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↓	↑	↓	↓
MACCS	↓**	↑	↓	↑**	↓*
Pharmacophore	↓**	↑*	↓*	↑**	↓**
PubChem	↓**	↓	↑	↑**	↓**
Substructure	↑	↑**	↓**	↓	↑**

Figure 69: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↑	↑	↓	↓	↑*
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↑	↑*	↓*	↓	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 70: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑	↓	↓**	↑**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↑	↓	↑	↓	↑
Graph-Only	↑	↓	↑	↓	↓
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↑	↑**	↓**	↓	↑**
PubChem	↑	↓	↑	↓	↑
Substructure	↑*	↑	↓	↓*	↑*

Figure 71: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑*	↓	↑	↓*	↑
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑*	↑	↓	↓*	↑
Pharmacophore	↑	↑	↓	↓	↑
PubChem	↓	↓	↑	↑	↓
Substructure	↑*	↑	↓	↓*	↑

Figure 72: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

By adding numerical descriptors to the binary-only descriptors we see that the most significant improvement among our classification metrics has happened with

the Pharmacophore, Substructure and MACCS fingerprints. In the next part, we classify the original dataset and show the classification metrics used.

Factor XA Classification Results per Classifiers – Original

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

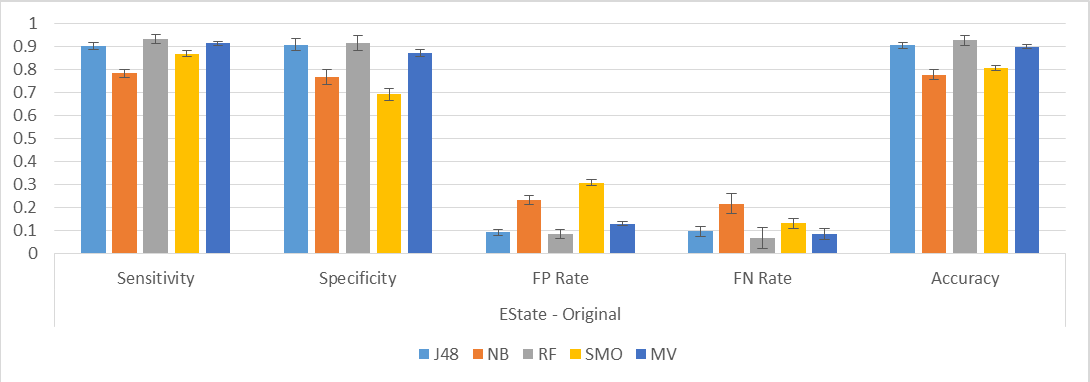


Figure 73: Classifier performance for by EState - Original

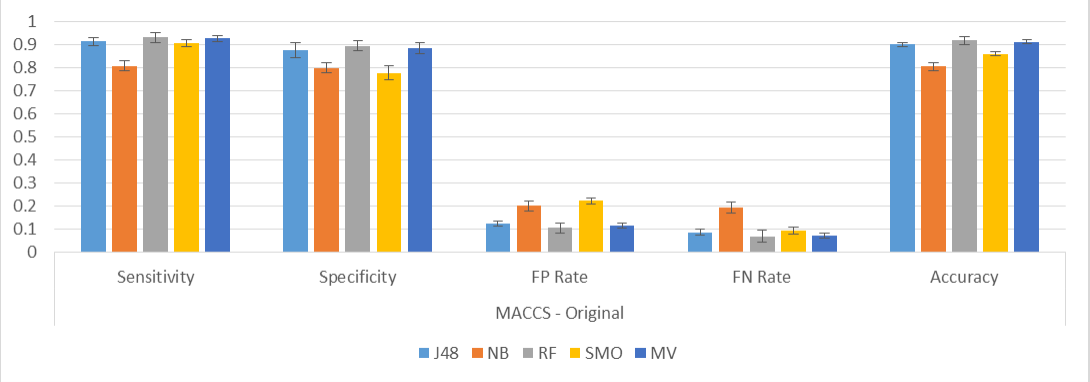


Figure 74: Classifier performance for MAACS - Original

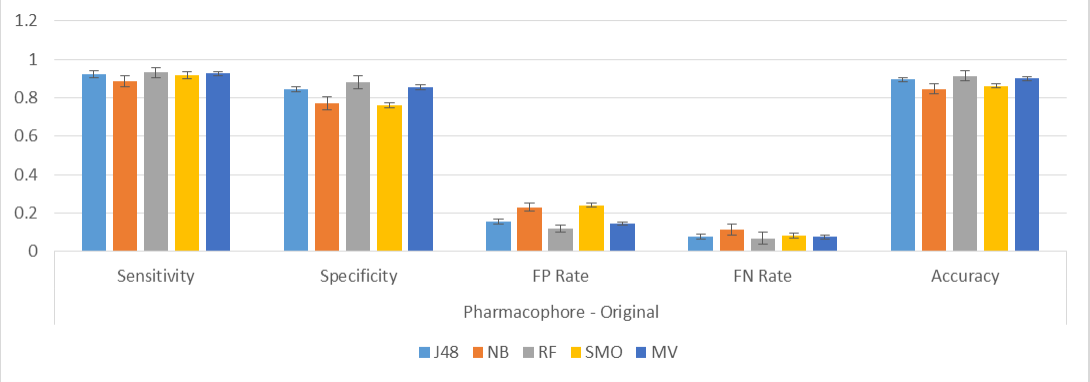


Figure 75: Classifier performance for Pharmacophore - Original

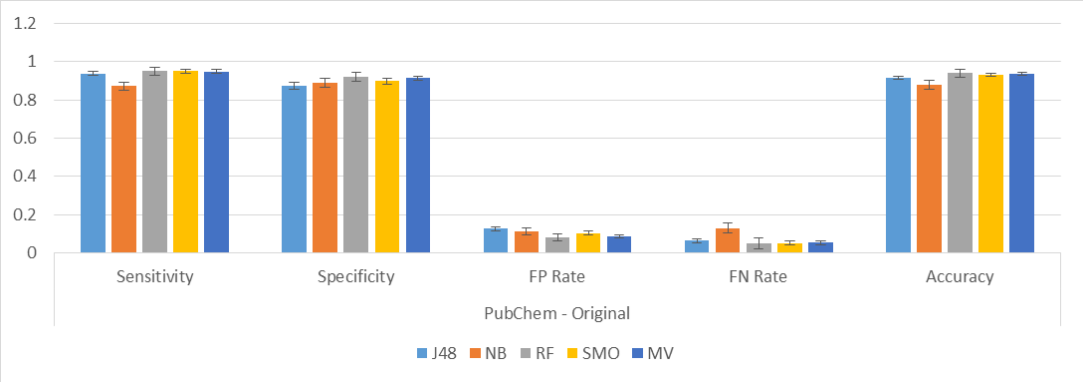


Figure 76: Classifier performance for PubChem - Original

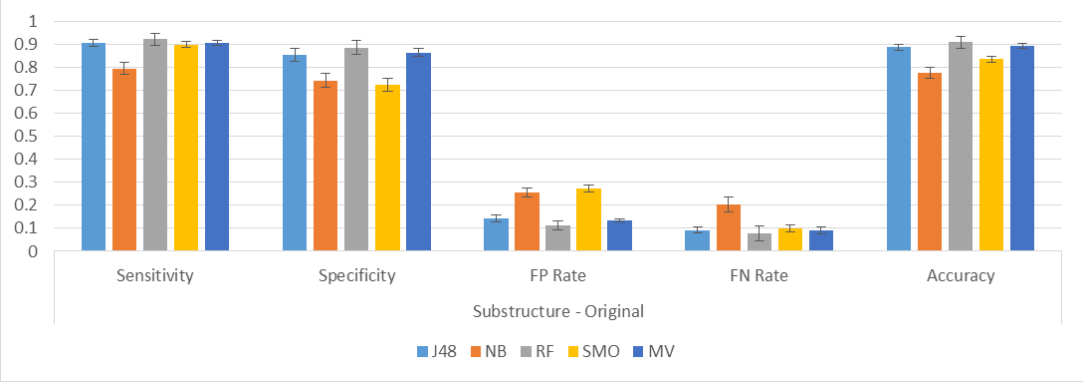


Figure 77: Classifier performance for Substructure - Original

By looking at Figures (73-77), the classifiers have their best performances when used with the PubChem fingerprint. The false positive and false negative rates are at their lowest compared to the other figures in this group. In the next section, we will observe how adding numerical fingerprints to the original affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓**	↑**	↓	↓
NB	↓	↑	↓	↑	↑
RF	↑	↑	↓	↓	↑
SMO	↑**	↑	↓	↓**	↑**
MV	↑*	↓	↑	↓*	↑

Figure 78: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↑
NB	↓**	↑	↓	↑**	↓*
RF	↑	↑**	↓**	↓	↑**
SMO	↑	↑	↓	↓	↑
MV	↑*	↑	↓	↓*	↑

Figure 79: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓**	↑*	↓*	↑**	↓**
RF	↑	↑*	↓*	↓	↑**
SMO	↑	↑**	↓**	↓	↑**
MV	↑	↑	↓	↓	↑

Figure 80: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓	↑	↑	↓
NB	↓**	↓	↑	↑**	↓**
RF	↑	↑	↓	↓	↑
SMO	↑	↓	↑	↓	↑
MV	↓	↓	↑	↑	↓

Figure 81: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑	↑**	↓**	↓	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑*	↑	↓	↓*	↑*
MV	↑*	↑	↓	↓*	↑

Figure 82: Results from adding numerical fingerprints to binary fingerprints for *Substructure*

The classifier Random Forest has consistently had the most improvement after adding the numerical descriptors to it, regardless of the fingerprint it was used with. The other two that stand out are SMO and Majority Voting, similar to what we observed in the benchmark dataset. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

Factor XA Classification Results per Fingerprint– Original SMOTEd All

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which

fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

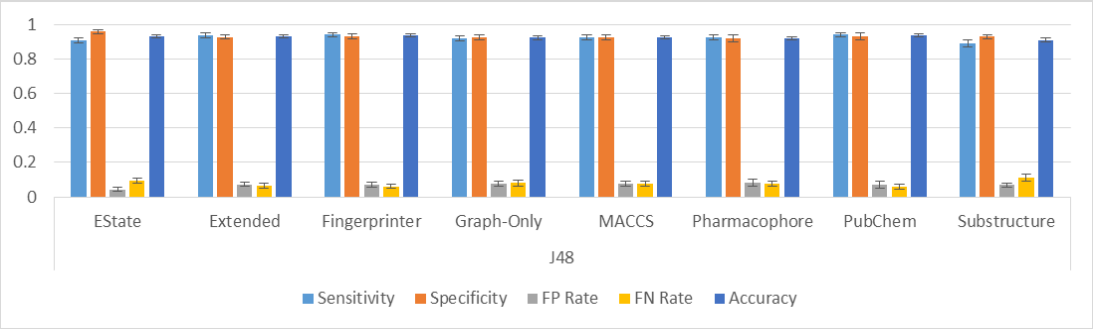


Figure 83: Classification results from classifying the Fontaine dataset by J48

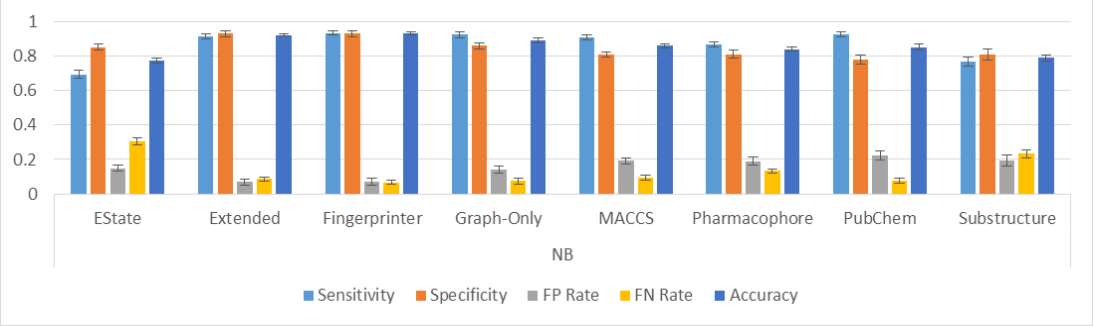


Figure 84: Classification results from classifying the Fontaine dataset by NaïveBayes

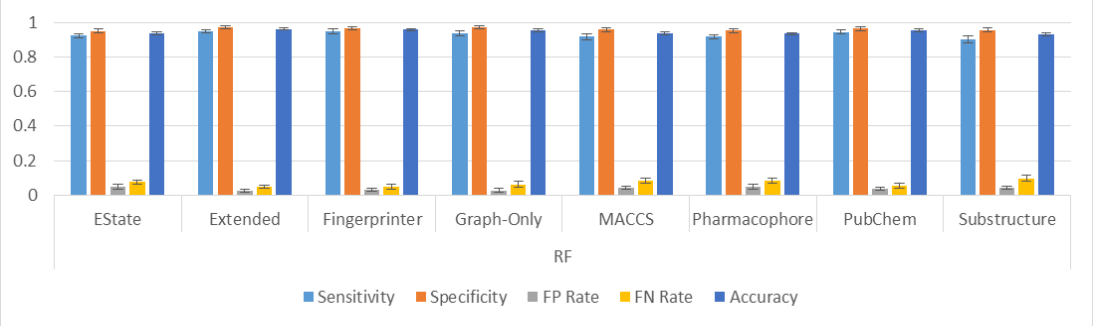


Figure 85: Classification results from classifying the Fontaine dataset by Random Forest

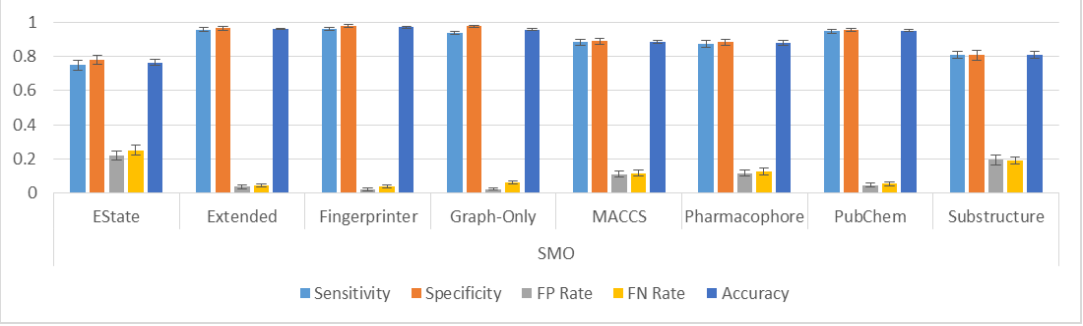


Figure 86: Classification results from classifying the Fontaine dataset by SMO

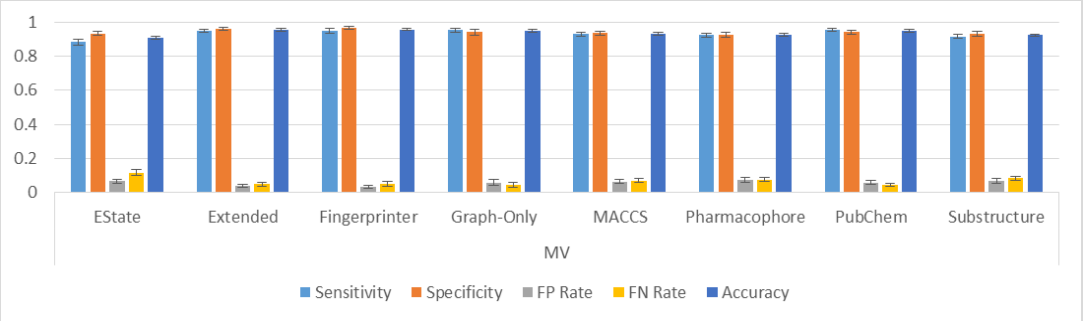


Figure 87: Classification results from classifying the Fontaine dataset by Majority Voting

When looking at the results from this section one must keep in mind that the imbalanced dataset has been balanced by adding synthetic samples to the minority class. The minority class samples could be in clusters in the dimension space or scattered among other samples of the other class. Therefore the results that we have in this section could be extremely optimal but it also may be that the added samples contributed to the classifier bias towards the majority class.

All fingerprints have produced good results especially when used in combination with J48, Random Forest and Majority Voting. EState and Substructure appear to have produced the least optimal results with NaïveBayes and SMO. With the same two classifiers MACCS, Pharmacophore and PubChem produced higher false positive rates than false negative ones. In most cases a higher percentage of false negative is preferred to false positives. Next, we will observe how adding numerical fingerprints to the balanced dataset affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓**	↑**	↓	↓**
Extended	↓	↑	↓	↑	↓
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↓	↓	↑	↑	↓
PubChem	↓	↓	↑	↑	↓
Substructure	↑*	↓	↑	↓*	↑

Figure 88: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓**	↑**	↓**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↓*	↑	↓	↑*	↓
Pharmacophore	↑**	↓	↑	↓**	↑
PubChem	↓**	↑**	↓**	↑**	↑**
Substructure	↑**	↓	↑	↓**	↑**

Figure 89: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑*	↑	↓	↓*	↑**
Pharmacophore	↑*	↑	↓	↓*	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑	↓	↓**	↑**

Figure 90: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↓	↑	↓	↑
Graph-Only	↑	↓	↑	↓	↑
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↑	↓	↑	↓	↑
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑	↓	↓**	↑**

Figure 91: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓	↑	↓**	↑**
Extended	↓	↑	↓	↑	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↓	↑	↑	↓
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↑	↓	↑	↓	↑
PubChem	↓	↑	↓	↑	↑
Substructure	↑	↓	↑	↓	↑

Figure 92: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

The fingerprints MACCS, PubChem and Substructure have been fitted most from the addition of numerical descriptors and have had the most significant improvements especially when combined with Random Forest and in the case of

Substructure, with SMO too. In the next part, we classify the dataset that was balanced before splitting and show the classification metrics used.

Factor XA Classification Results per Classifiers – Original SMOTEd All

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

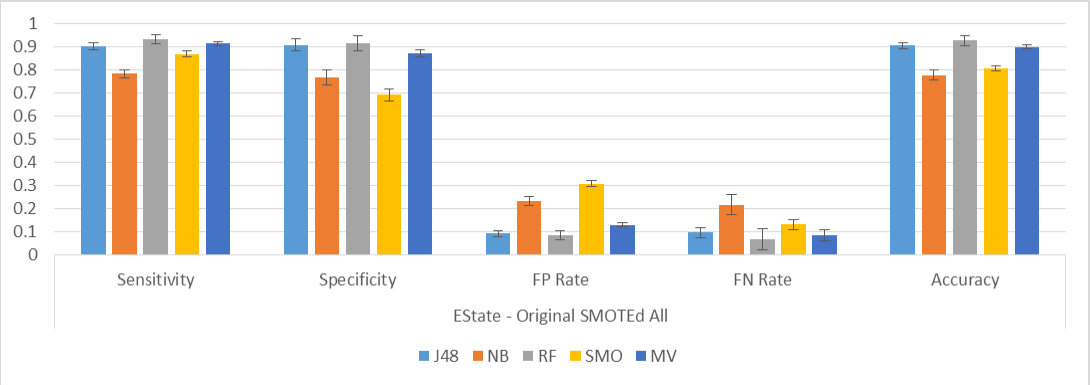


Figure 93: Classifier performance for EState – Original SMOTEd All

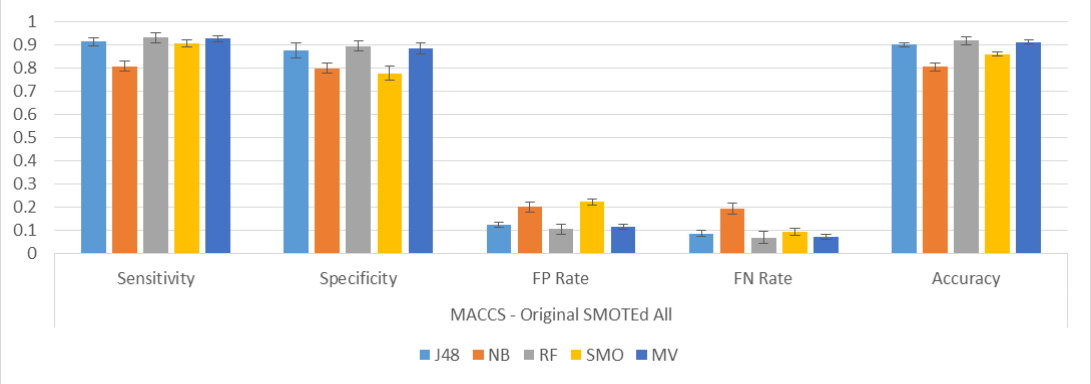


Figure 94: Classifier performance for MACCS – Original SMOTEd All

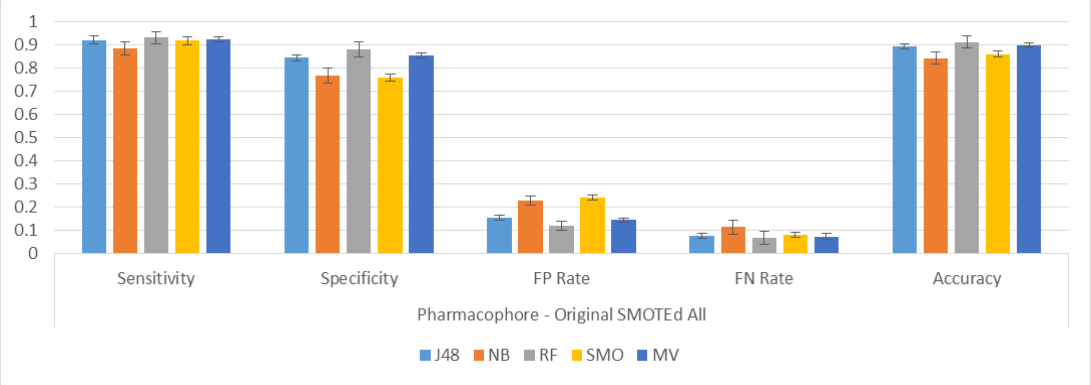


Figure 95: Classifier performance for Pharmacophore – Original SMOTEd All

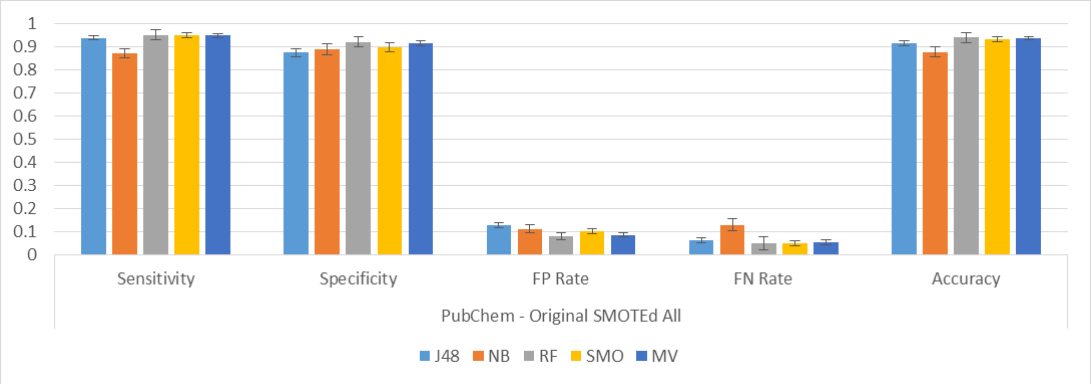


Figure 96: Classifier performance for *PubChem* – Original SMOTEd All

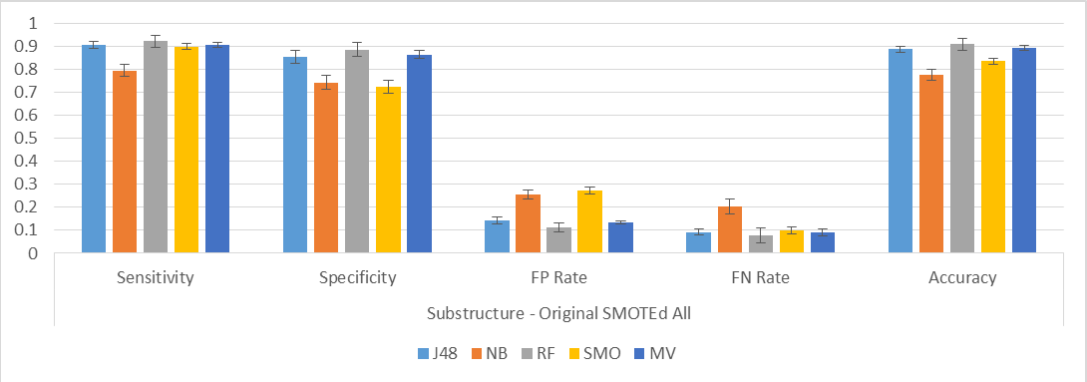


Figure 97: Classifier performance for *Substructure* – Original SMOTEd All

Of the five classifiers used (four single and one ensemble), J48, Random Forest and Majority Voting have consistently performed the best in figures (93-97). SMO performed exceptionally well when used with the PubChem fingerprint. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓**	↑**	↓	↓
NB	↓	↑	↓	↑	↑
RF	↑	↑	↓	↓	↑
SMO	↑**	↑	↓	↓**	↑**
MV	↑*	↓	↑	↓*	↑

Figure 98: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↑
NB	↓**	↑	↓	↑**	↓*
RF	↑	↑**	↓**	↓	↑**
SMO	↑	↑	↓	↓	↑
MV	↑*	↑	↓	↓*	↑

Figure 99: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓**	↑*	↓*	↑**	↓**
RF	↑	↑*	↓*	↓	↑**
SMO	↑	↑**	↓**	↓	↑**
MV	↑	↑	↓	↓	↑

Figure 100: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓	↑	↑	↓
NB	↓**	↓	↑	↑**	↓**
RF	↑	↑	↓	↓	↑
SMO	↑	↓	↑	↓	↑
MV	↓	↓	↑	↑	↓

Figure 101: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑	↑**	↓**	↓	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑*	↑	↓	↓*	↑*
MV	↑*	↑	↓	↓*	↑

Figure 102: Results from adding numerical fingerprints to binary fingerprints for Substructure

The classifiers Random Forest, SMO and Majority Voting have certainly benefited from the addition of numerical descriptors in Figures (98-102). The not so good results were achieved when in combination with PubChem and EState, with NaïveBayes producing the worst results except for when used with Substructure. In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

Factor XA Classification Results per Fingerprint – Original SMOTEd Training

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

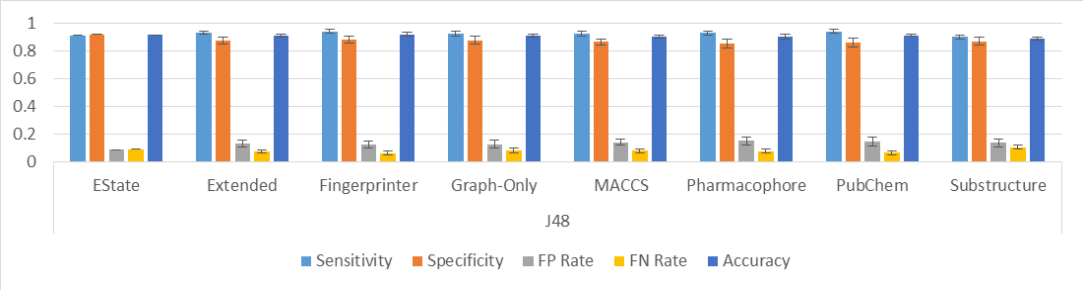


Figure 103: Classification results from classifying the Fontaine dataset by J48

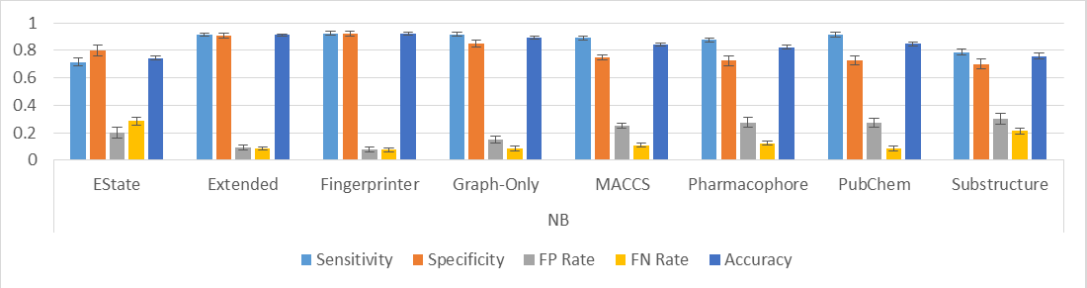


Figure 104: Classification results from classifying the Fontaine dataset by NaïveBayes

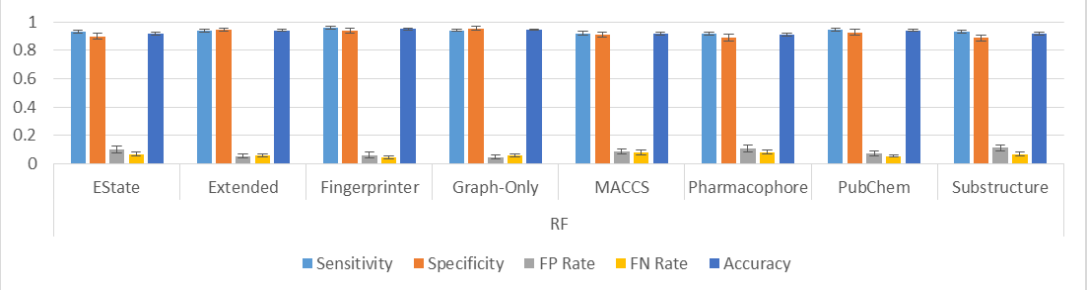


Figure 105: Classification results from classifying the Fontaine dataset by Random Forest

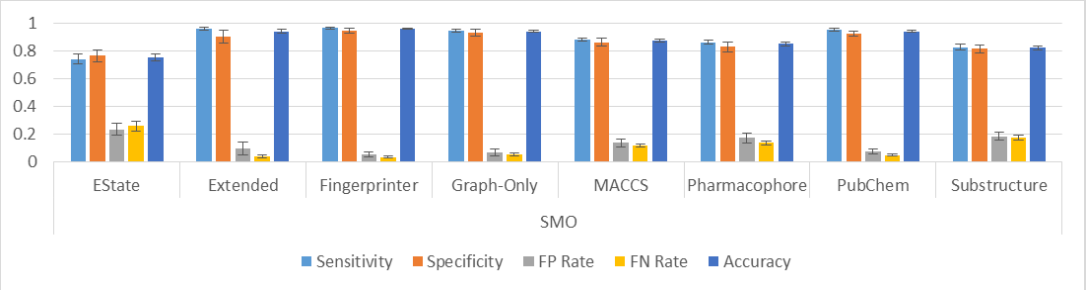


Figure 106: Classification results from classifying the Fontaine dataset by SMO

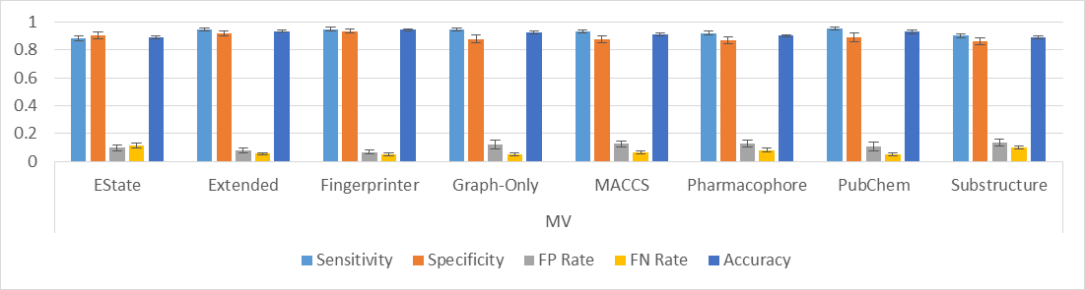


Figure 107: Classification results from classifying the Fontaine dataset by Majority Voting

With this method only the training set was balanced and the trained classifier exposed to the imbalanced unseen test set. The added synthetic minority samples may have improved the learning of the classifier or may have simply helped maintain the bias towards the majority class, depending on how the minority class samples are situated in the dimension space.

EState and Substructure can be considered the two fingerprints that have higher false positive and false negative results than the other fingerprints used in the presence of all five classifiers. Otherwise most results achieved from this set of tests seem promising. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓**	↑**	↓	↓
Extended	↑	↓	↑	↓	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↓	↑	↓	↑
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↓	↑	↓	↑	↓
PubChem	↓	↑	↓	↑	↑
Substructure	↑	↑	↓	↓	↑

Figure 108: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓	↑	↓**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↑	↓	↑	↓
MACCS	↓	↑*	↓*	↑	↑
Pharmacophore	↑**	↑	↓	↓**	↑**
PubChem	↓**	↑**	↓**	↑**	↑
Substructure	↑**	↑	↓	↓**	↑**

Figure 109: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↓	↑	↓	↑	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↓	↑	↓	↑
MACCS	↑*	↑**	↓**	↓*	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↑	↑**	↓**	↓	↑**

Figure 110: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑*	↓*	↓**	↑**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↑	↑*	↓*	↓	↑*
PubChem	↑	↑	↓	↓	↑
Substructure	↑*	↑	↓	↓*	↑

Figure 111: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓*	↑*	↓**	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓	↑	↓	↓
Pharmacophore	↑	↑	↓	↓	↑*
PubChem	↔	↑	↓	↔	↑
Substructure	↑*	↑	↓	↓*	↑**

Figure 112: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

MACCS and Pharmacophore have the better results with Random Forest; and SMO and Substructure has done exceptionally well with Random Forest and Majority Voting in terms of all criteria. In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

Factor XA Classification Results per Classifiers – Original SMOTEd Training

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

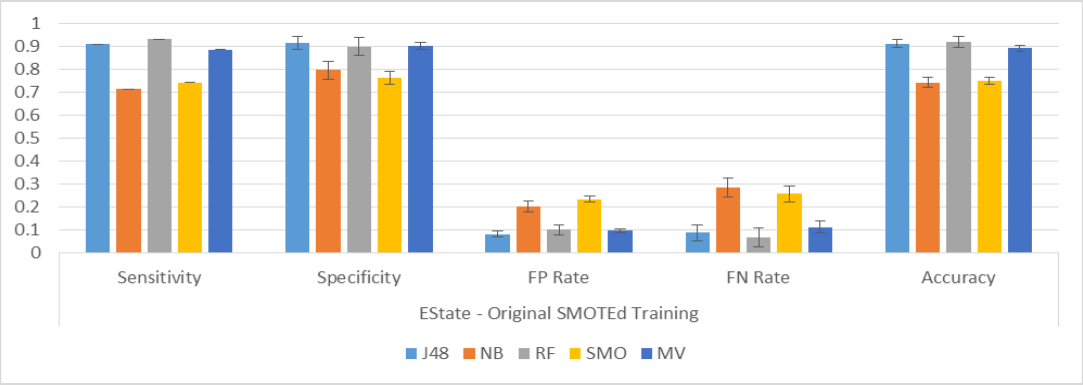


Figure 113: Classifier performance for EState – Original SMOTEd Training

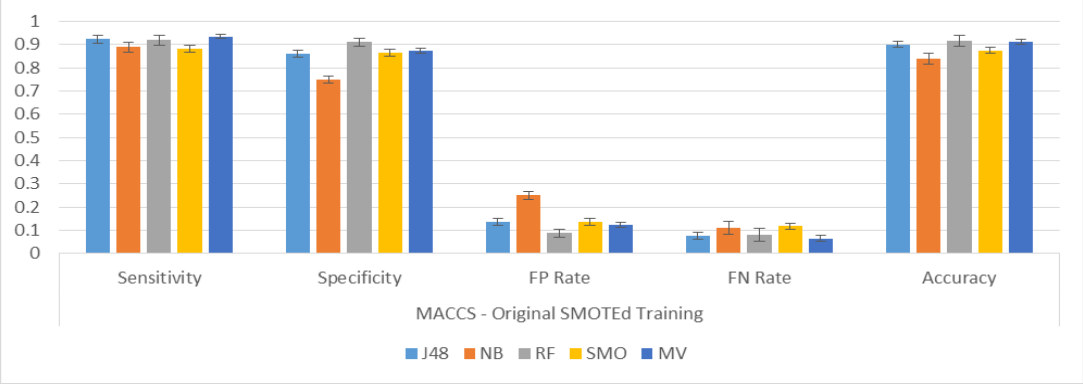


Figure 114: Classifier performance for MACCS – Original SMOTEd Training

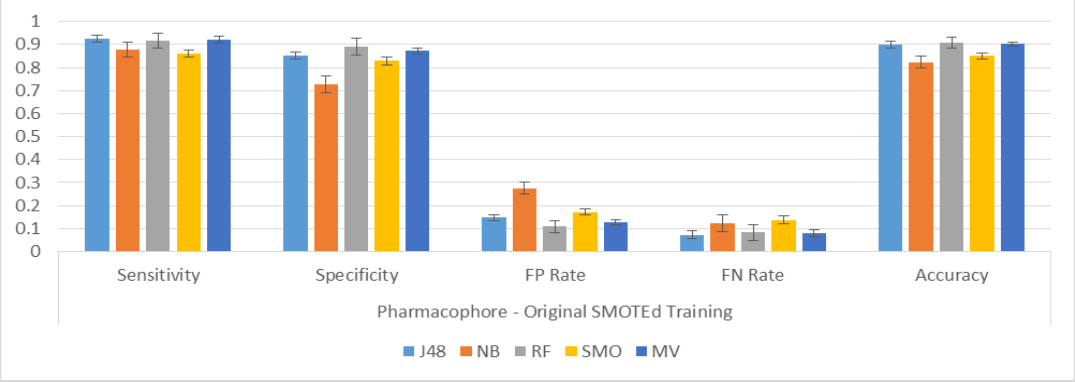


Figure 115: Classifier performance for Pharmacophore – Original SMOTEd Training

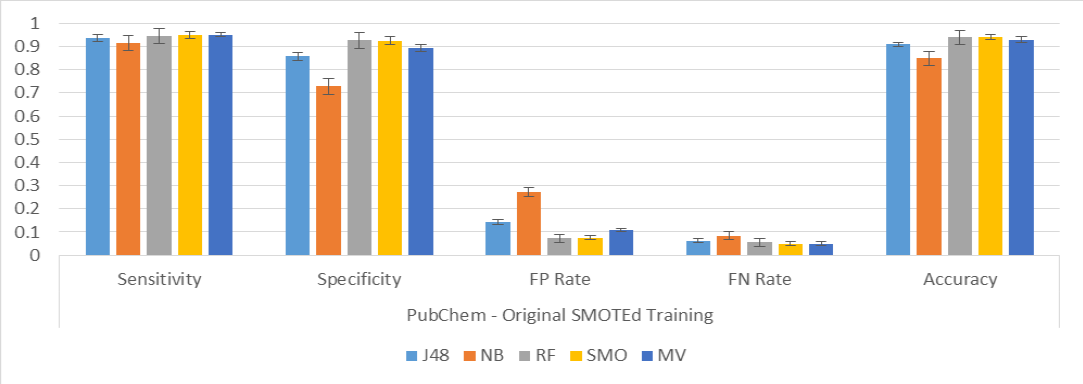


Figure 116: Classifier performance for PubChem – Original SMOTEd Training

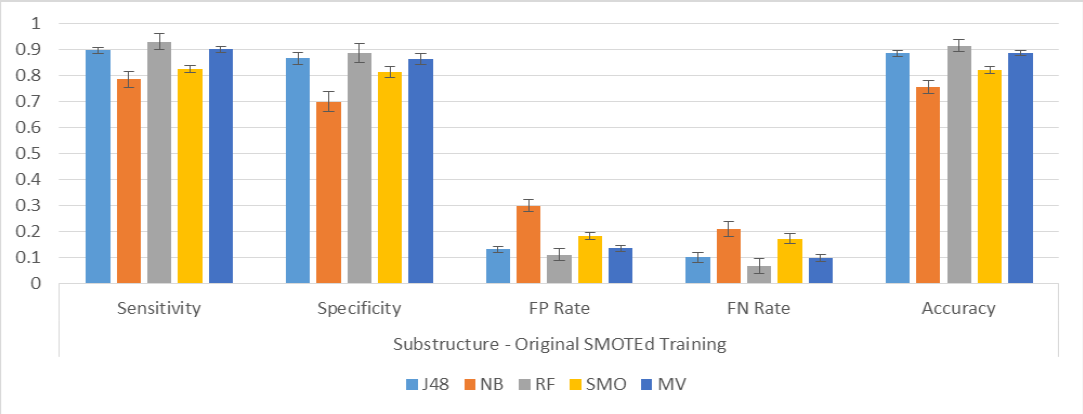


Figure 117: Classifier performance for Substructure – Original SMOTEd Training

The classifiers have performed better in the presence of the MACCS and PubChem fingerprints. Overall using this method J48, Random Forest and Majority Voting have the better results of the overall classifiers. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓**	↑**	↓	↓
NB	↑**	↓	↑	↓**	↑**
RF	↑	↑**	↓**	↓	↑**
SMO	↑**	↑*	↓*	↓**	↑**
MV	↑**	↓*	↑*	↓**	↑**

Figure 118: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓	↑	↑	↓
NB	↓	↑*	↓*	↑	↑
RF	↑*	↑**	↓**	↓*	↑**
SMO	↑	↑	↓	↓	↑
MV	↑	↓	↑	↓	↓

Figure 119: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↓
NB	↑**	↑	↓	↓**	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑	↑*	↓*	↓	↑*
MV	↑	↑	↓	↓	↑*

Figure 120: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓**	↑**	↓**	↑**	↑
RF	↓	↑	↓	↑	↑
SMO	↑	↑	↓	↓	↑
MV	↔	↑	↓	↔	↑

Figure 121: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑**	↑	↓	↓**	↑**
RF	↑	↑**	↓**	↓	↑**
SMO	↑*	↑	↓	↓*	↑
MV	↑*	↑	↓	↓*	↑**

Figure 122: Results from adding numerical fingerprints to binary fingerprints for Substructure

In short, Figures (118-122) indicate that Random Forest and SMO have the better and more significant improvements in the case of the EState, MACCS, Pharmacophore and Substructure fingerprints. Substructure has exceptionally good results with all of the classifiers. In the next section, we classify the original dataset with PCA and show the classification metrics used.

Factor XA Classification Results per Fingerprint – PCA Original

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results. Here PCA technique has been applied.

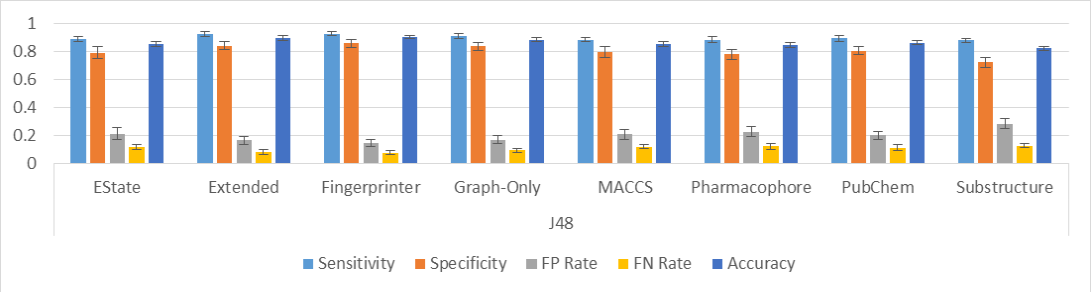


Figure 123: Classification results from classifying the Fontaine dataset by J48

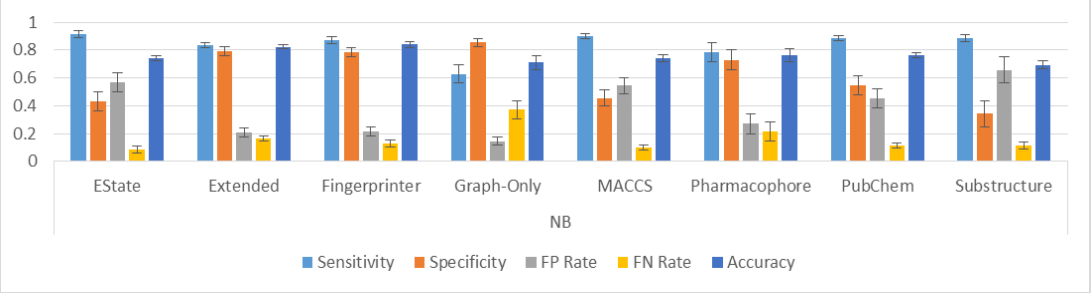


Figure 124: Classification results from classifying the Fontaine dataset by NaïveBayes

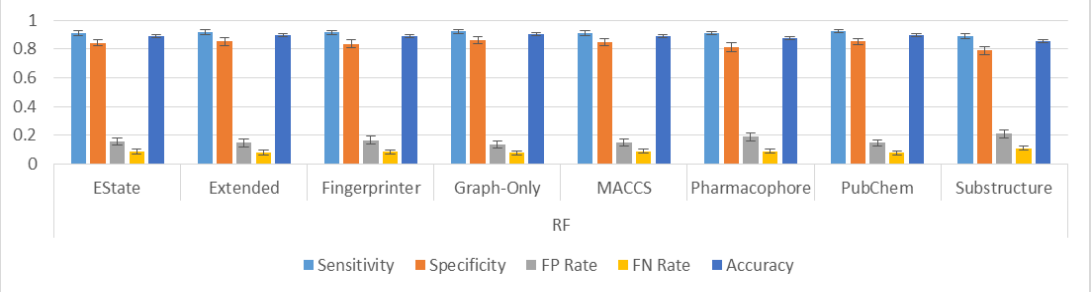


Figure 125: Classification results from classifying the Fontaine dataset by Random Forest

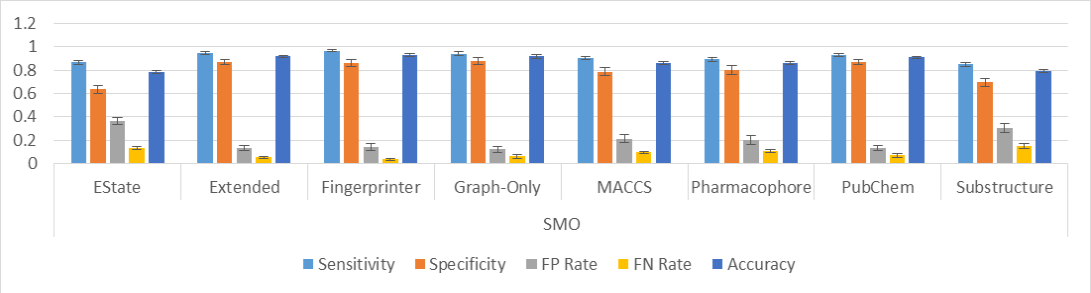


Figure 126: Classification results from classifying the *Fontaine* dataset by SMO

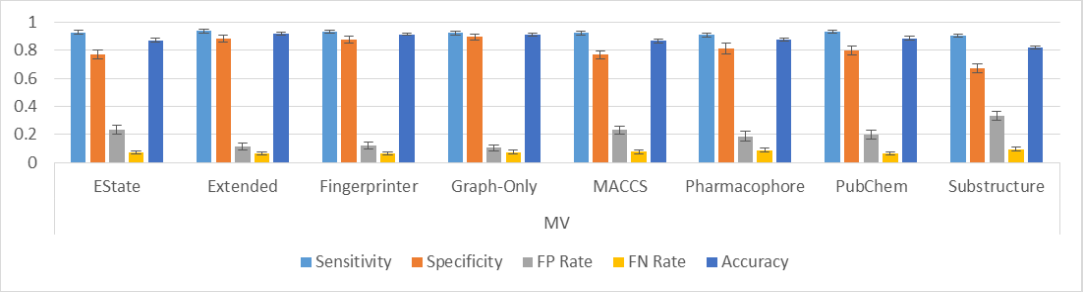


Figure 127: Classification results from classifying the Fontaine dataset by Majority Voting

All fingerprints show good results in producing the metrics. EState and Substructure appear to have less desirable results compare to the other fingerprints when it comes to false positive and false negatives. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↑
Extended	↓	↑	↓	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↓	↑	↑	↓
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↑	↑	↓	↓	↑
PubChem	↓	↑	↓	↑	↑
Substructure	↓	↑	↓	↑	↓

Figure 128: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑	↓	↑	↑
Extended	↓	↓*	↑*	↑	↓*
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↑*	↓*	↑	↑
MACCS	↓**	↑	↓	↑**	↓
Pharmacophore	↑*	↓*	↑*	↓*	↑
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓	↑	↓	↓

Figure 129: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↑	↓	↓	↑*
Pharmacophore	↑	↑	↓	↓	↑
PubChem	↑	↓	↑	↓	↑
Substructure	↑*	↑	↓	↓*	↑*

Figure 130: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↓	↑	↓	↑	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↑**	↑	↓	↓**	↑**
Pharmacophore	↑	↓	↑	↓	↑
PubChem	↑	↑	↓	↓	↑*
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 131: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↑
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↓	↑	↑	↓
MACCS	↑	↑*	↓*	↓	↑*
Pharmacophore	↑**	↓	↑	↓**	↑
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↑*	↓*	↓	↑**

Figure 132: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

The levels of improvement in classification metrics in Figures (128-132) vary in the sense that no particular fingerprint shows continuous improvement and if so it is not significant. However, Random Forest has the most improvement in its fingerprints followed by SMO and Majority Voting. In the next section, we classify the original dataset with PCA and show the classification metrics used.

Factor XA Classification Results per Classifiers – PCA Original

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results. The PCA technique was applied here to the dataset.

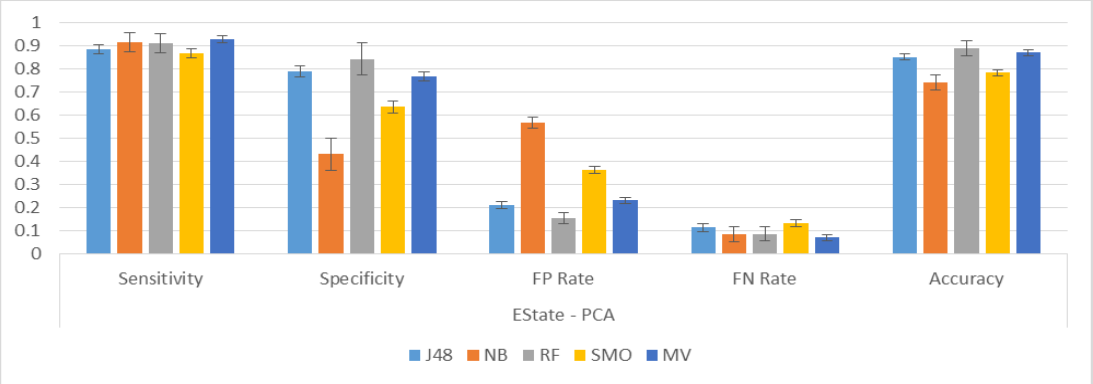


Figure 133: Classifier performance for EState – PCA Dataset

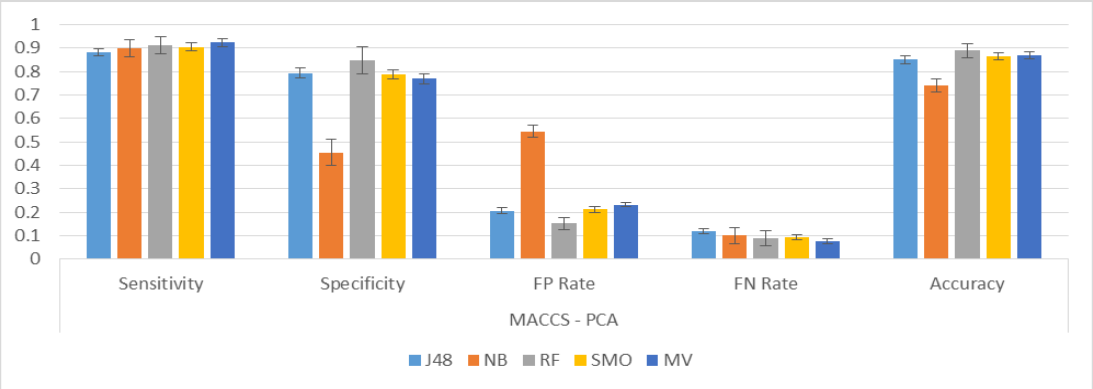


Figure 134: Classifier performance for MACCS – PCA Dataset

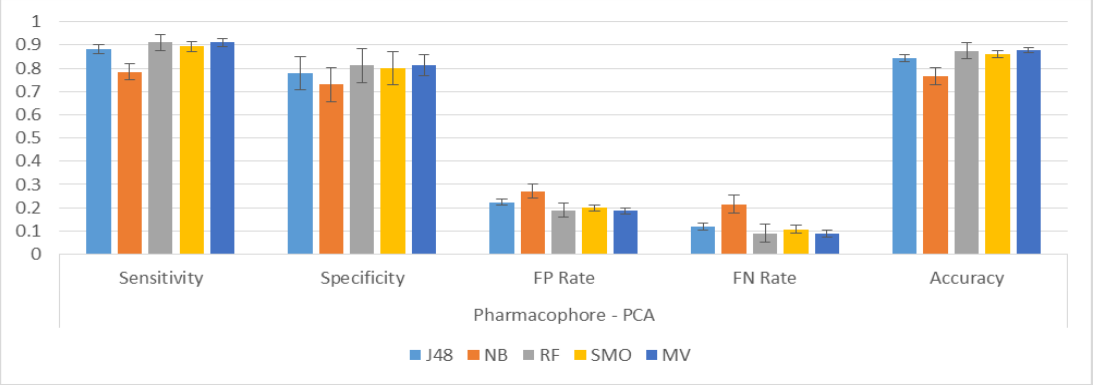


Figure 135: Classifier performance for Pharmacophore – PCA Dataset

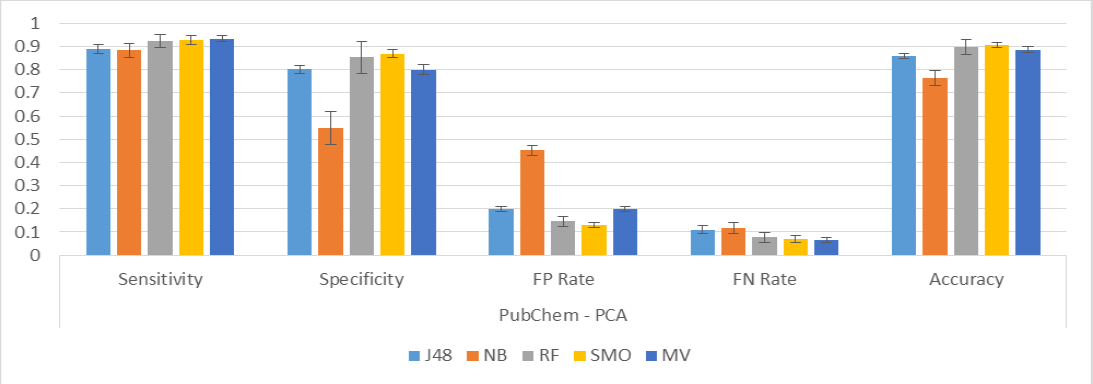


Figure 136: Classifier performance for PubChem – PCA Dataset

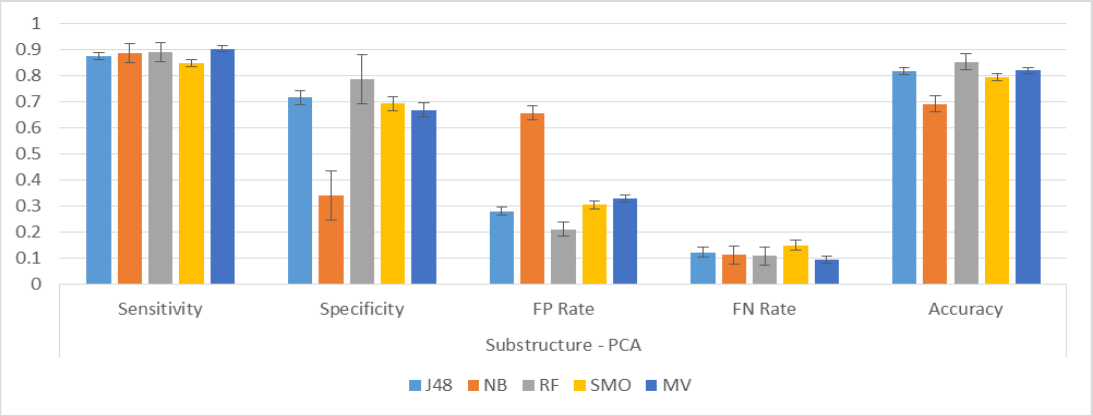


Figure 137: Classifier performance for Substructure – PCA Dataset

In this set of tests it seems that the classifiers have performed very well with the Pharmacophore fingerprint. In this case the levels of false positive and false negative are both relatively low, unlike the other four figures. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↑
NB	↓	↑	↓	↑	↑
RF	↑	↑	↓	↓	↑
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑	↓	↑	↓	↑

Figure 138: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓**	↑	↓	↑**	↓
RF	↑	↑	↓	↓	↑*
SMO	↑**	↑	↓	↓**	↑**
MV	↑	↑*	↓*	↓	↑*

Figure 139: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑*	↓*	↑*	↓*	↑
RF	↑	↑	↓	↓	↑
SMO	↑	↓	↑	↓	↑
MV	↑**	↓	↑	↓**	↑

Figure 140: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↑	↑	↓	↓	↑
RF	↑	↓	↑	↓	↑
SMO	↑	↑	↓	↓	↑*
MV	↑	↑	↓	↓	↑

Figure 141: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↓
NB	↑	↓	↑	↓	↓
RF	↑*	↑	↓	↓*	↑*
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑	↑*	↓*	↓	↑**

Figure 142: Results from adding numerical fingerprints to binary fingerprints for Substructure

The results for SMO and Majority Voting have improved in the presence of MACCS, PubChem and Substructure fingerprints. The significance of the improvement is especially visible in Figure 142. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

Factor XA Classification Results per Fingerprint– PCA SMOTEd All

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

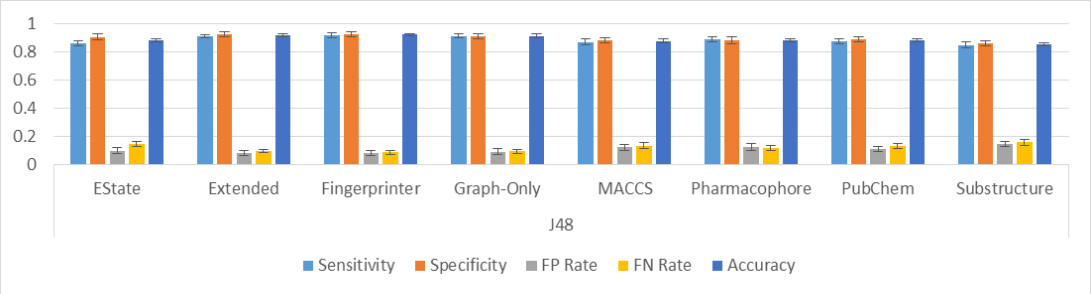


Figure 143: Classification results from classifying the Fontaine dataset by J48

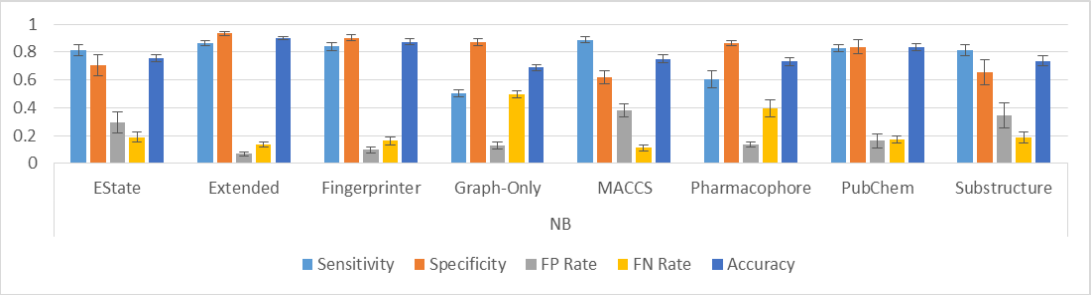


Figure 144: Classification results from classifying the Fontaine dataset by NaïveBayes

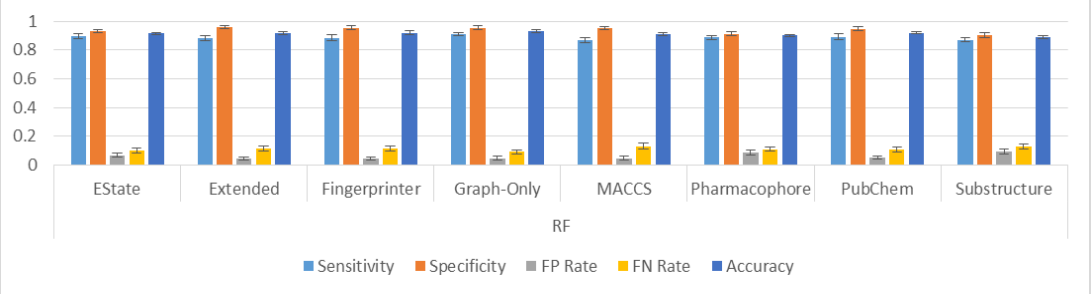


Figure 145: Classification results from classifying the Fontaine dataset by Random Forest

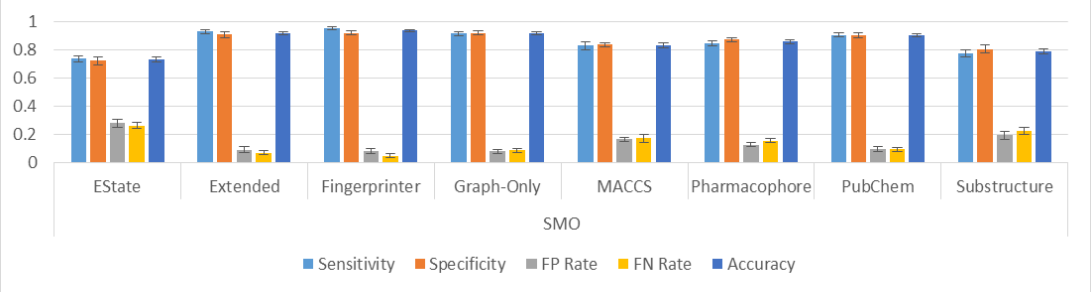


Figure 146: Classification results from classifying the Fontaine dataset by SMO

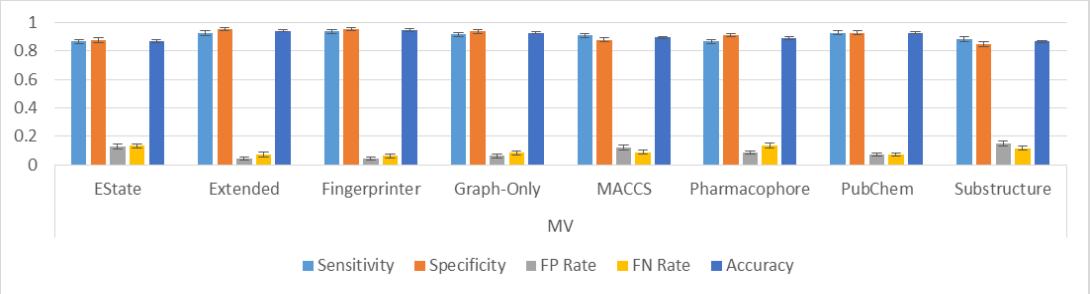


Figure 147: Classification results from classifying the Fontaine dataset by Majority Voting

The majority of the fingerprints have produced good results in the presence of J48, Random Forest and Majority Voting. With NaïveBayes, PubChem, Fingerprinter and Extended Fingerprinter have the better results. All but EState have good results with SMO. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓*	↑*	↓**	↑
Extended	↑	↓	↑	↓	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↓	↑	↓	↑	↓
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↓	↑	↓	↑	↑
PubChem	↑	↓	↑	↓	↓
Substructure	↑	↑	↓	↓	↑

Figure 148: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓	↑	↓**	↑
Extended	↓**	↓	↑	↑**	↓**
Fingerprinter	↓	↓	↑	↑	↓*
Graph-Only	↑	↑*	↓*	↓	↑
MACCS	↓**	↑	↓	↑**	↑
Pharmacophore	↑**	↓	↑	↓**	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓	↑	↓	↓

Figure 149: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑**	↑	↓	↓**	↑**
Pharmacophore	↑	↑*	↓*	↓	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↑*	↓*	↓	↑**

Figure 150: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↓	↓**	↑**	↑	↓
PubChem	↑*	↑	↓	↓*	↑*
Substructure	↑**	↑	↓	↓**	↑**

Figure 151: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓	↑	↓**	↑*
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑**	↓	↑	↓**	↑
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↑	↓	↓	↑

Figure 152: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

The most significant improvements in metrics can be seen with all fingerprints but Extended Fingerprinter with Random Forest. With all other classifiers, Substructure and Graph-Only Fingerprinter seem to be the ones benefitting from the additional of numerical descriptors. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

Factor XA Classification Results per Classifiers– PCA SMOTEd All

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

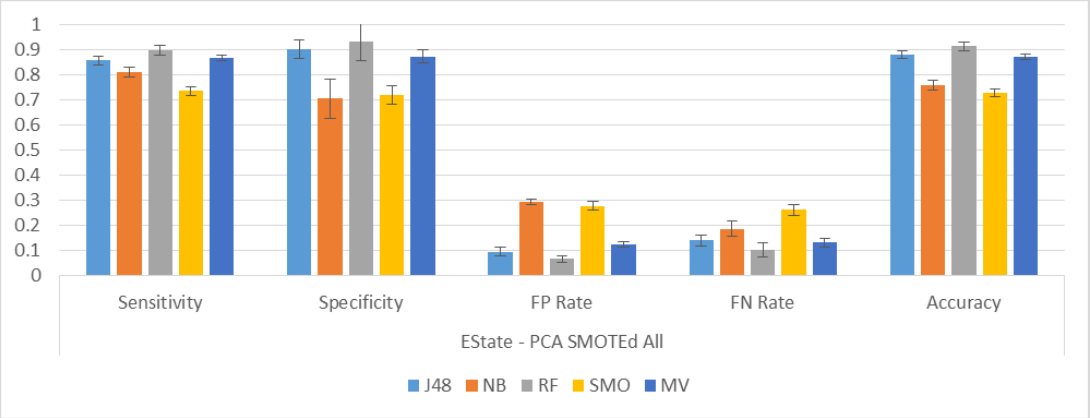


Figure 153: Classifier performance for EState – PCA SMOTEd All

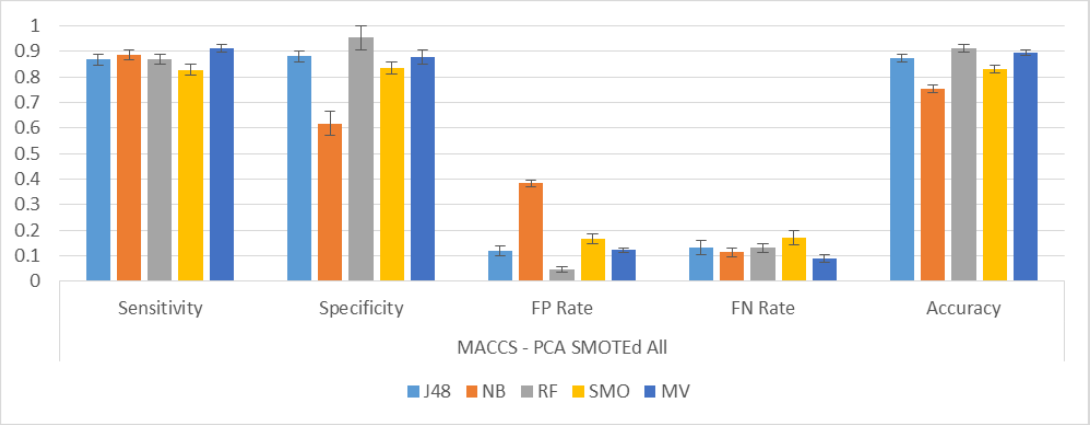


Figure 154: Classifier performance for MACCS – PCA SMOTEd All

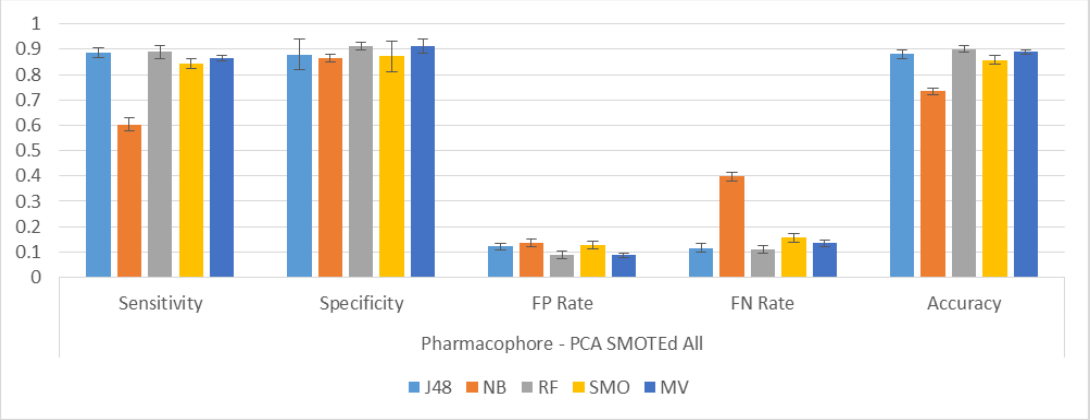


Figure 155: Classifier performance for Pharmacophore – PCA SMOTEd All

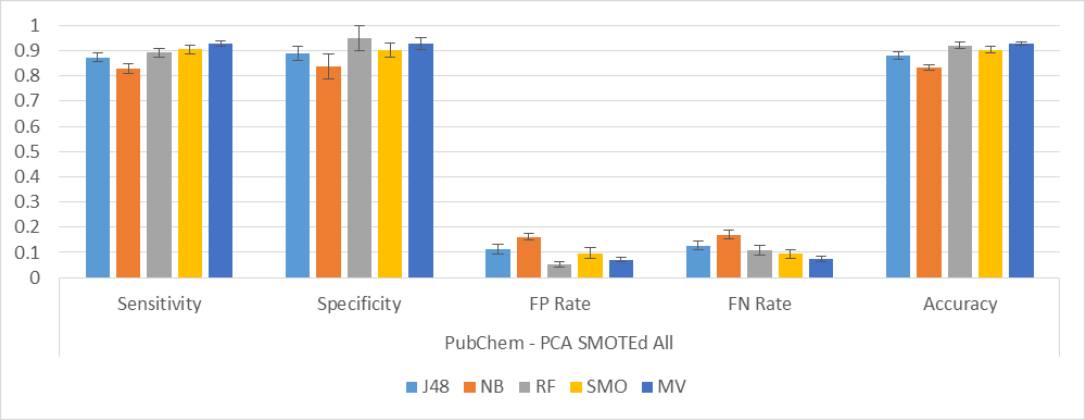


Figure 156: Classifier performance for PubChem – PCA SMOTEd All

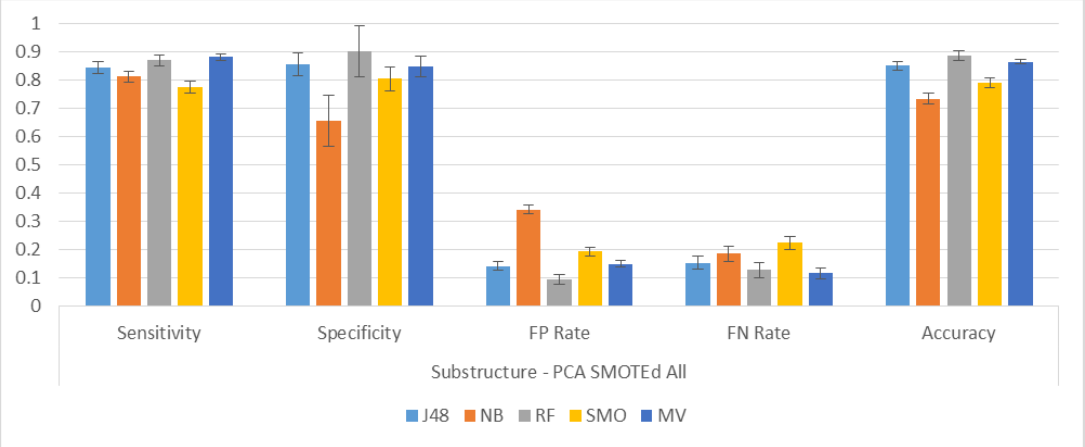


Figure 157: Classifier performance for Substructure – PCA SMOTEd All

The classifiers that have consistently performed well in these set of tests are J48, Random Forest and Majority Voting. NaïveBayes and SMO have especially performed well when used with the PubChem fingerprint. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↓*	↑*	↓**	↑
NB	↑**	↓	↑	↓**	↑
RF	↑	↑	↓	↓	↑
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↓	↑	↓**	↑*

Figure 158: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓**	↑	↓	↑**	↑
RF	↑**	↑	↓	↓**	↑**
SMO	↓	↓	↑	↑	↓
MV	↓	↑	↓	↑	↑

Figure 159: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↑**	↓	↑	↓**	↑**
RF	↑	↑*	↓*	↓	↑**
SMO	↓	↓**	↑**	↑	↓
MV	↑**	↓	↑	↓**	↑

Figure 160: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↑	↑	↓	↓	↑
RF	↑	↑	↓	↓	↑
SMO	↑*	↑	↓	↓*	↑*
MV	↑	↑	↓	↓	↑

Figure 161: Classifier performance for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑	↓	↑	↓	↓
RF	↑	↑*	↓*	↓	↑**
SMO	↑**	↑	↓	↓**	↑**
MV	↑	↑	↓	↓	↑

Figure 162: Classifier performance for Substructure

The classifier with the most significant and consistent improvement in these tests is Random Forest, followed by SMO and Majority Voting. In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

Factor XA Classification Results per Fingerprint– PCA SMOTEd Training

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

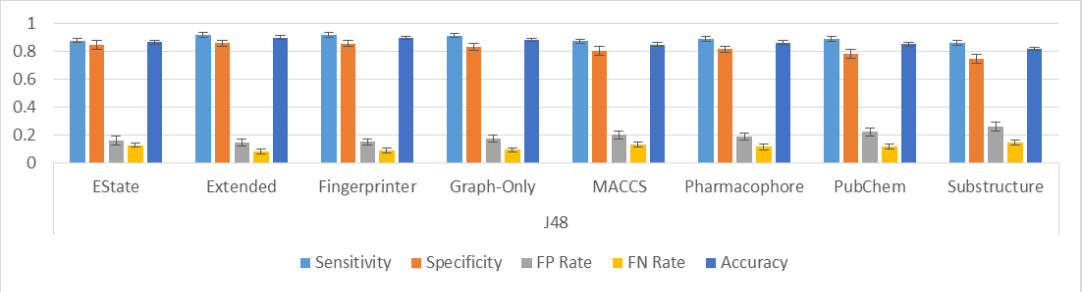


Figure 163: Classification results from classifying the Fontaine dataset by J48

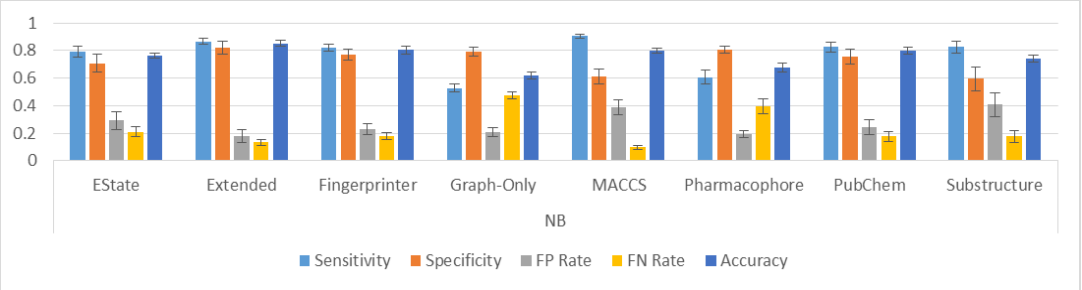


Figure 164: Classification results from classifying the Fontaine dataset by NaïveBayes

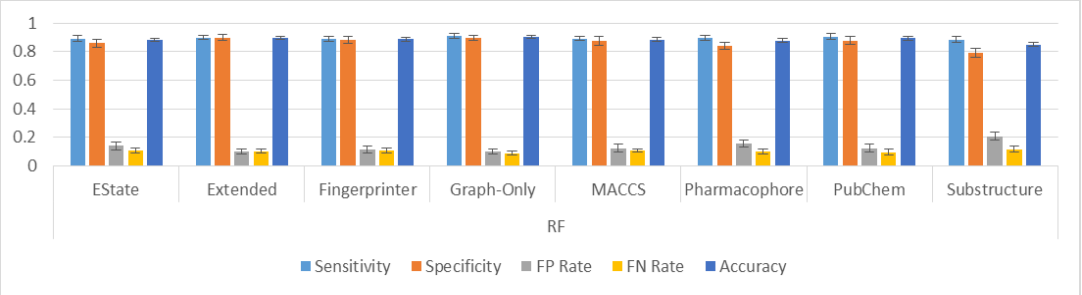


Figure 165: Classification results from classifying the Fontaine dataset by Random Forest

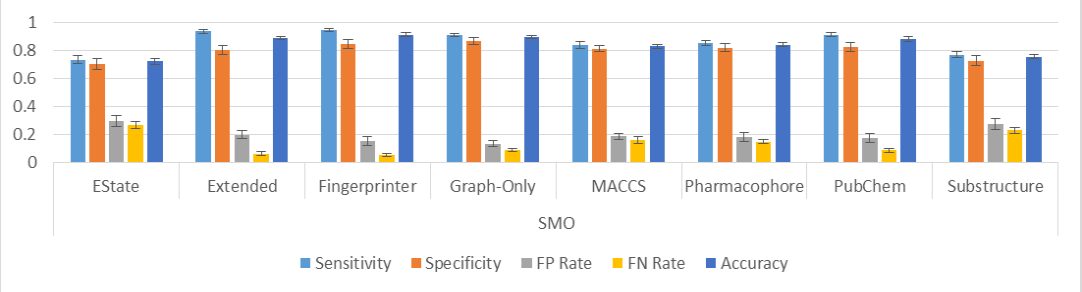


Figure 166: Classification results from classifying the Fontaine dataset by SMO

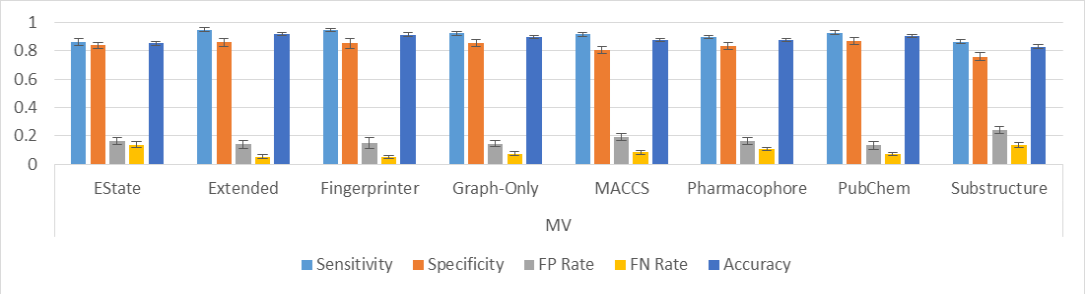


Figure 167: Classification results from classifying the Fontaine dataset by Majority Voting

In this set of tests the fingerprints have produced good optimal results with Random Forest in Figure 165. In this figure all fingerprints have consistent good outcome for all metrics. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↓
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↓	↑	↓	↑	↑
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↑*	↓*	↓	↑*

Figure 168: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓	↑	↓**	↑
Extended	↓*	↑	↓	↑*	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↓**	↑	↓	↑**	↓
Pharmacophore	↑**	↓	↑	↓**	↑*
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓	↑	↓	↓

Figure 169: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↓	↑	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑*	↑	↓	↓*	↑
Pharmacophore	↑	↑	↓	↓	↑*
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↑*	↓*	↓	↑**

Figure 170: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↑	↓	↑	↓	↑
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↓	↑	↓	↑
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↓	↓	↑	↑	↓
PubChem	↑	↑	↓	↓	↑*
Substructure	↑**	↑*	↓*	↓**	↑**

Figure 171: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓	↑	↓**	↑*
Extended	↓*	↓	↑	↑*	↓*
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑*	↓	↑	↓*	↑
PubChem	↑	↑	↓	↓	↑
Substructure	↑**	↑	↓	↓**	↑**

Figure 172: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

By adding numerical descriptors, Substructure fingerprint has shown good and significant improvement in metrics (except for when in the presence of NaïveBayes). In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

Factor XA Classification Results per Classifiers– PCA SMOTEd Training

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

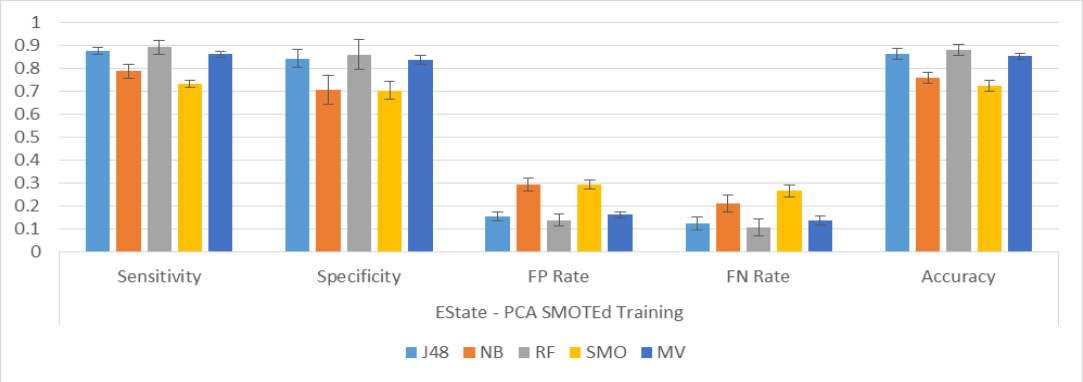


Figure 173: Classifier performance for EState – PCA SMOTEd Training

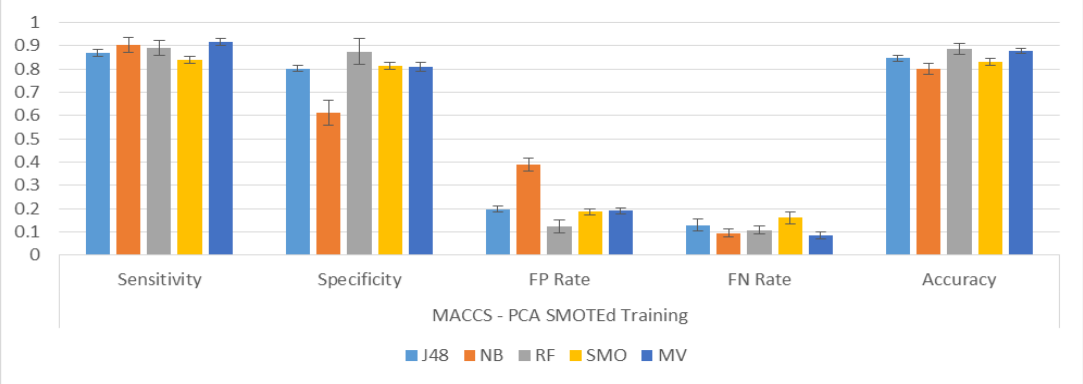


Figure 174: Classifier performance for MACCS – PCA SMOTEd Training

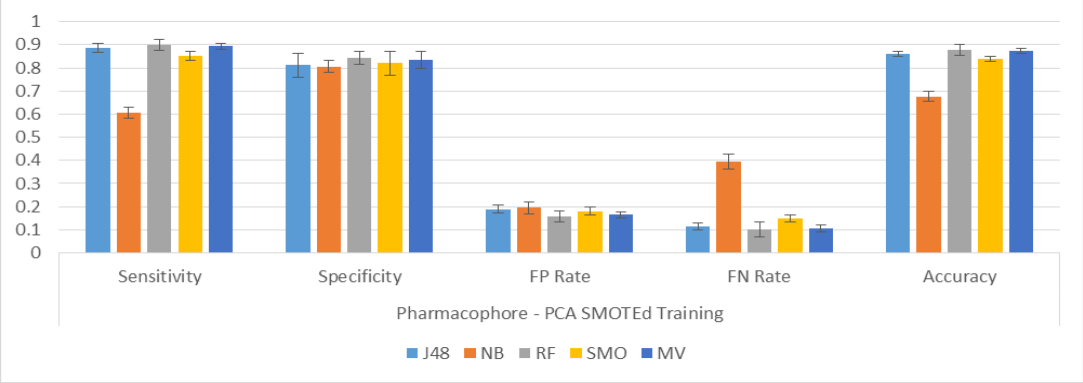


Figure 175: Classifier performance for Pharmacophore – PCA SMOTEd Training

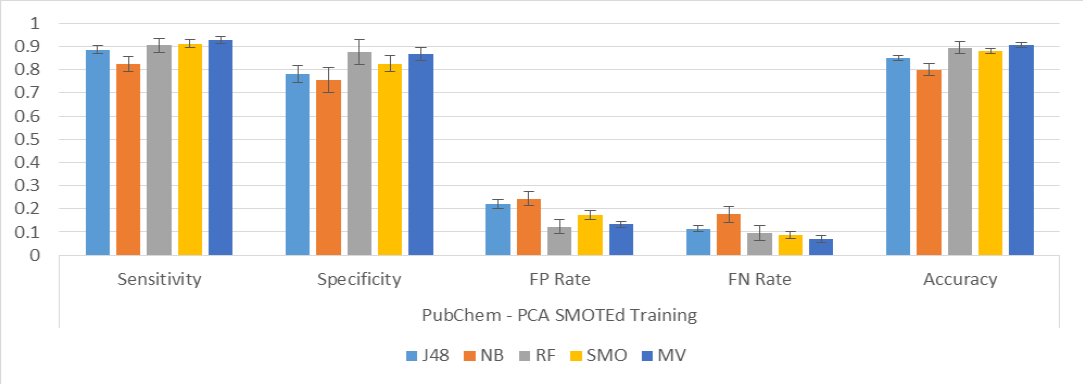


Figure 176: Classifier performance for PubChem – PCA SMOTEd Training

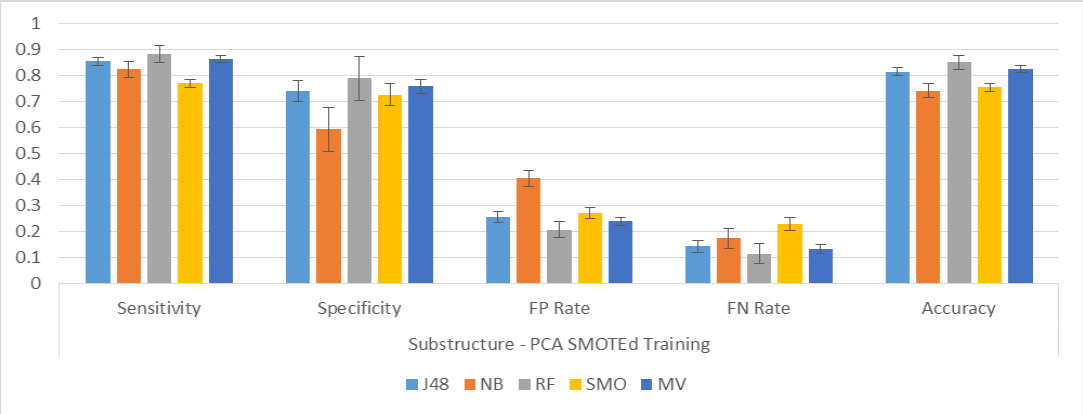


Figure 177: Classifier performance for Substructure – PCA SMOTEd Training

The classifiers Random Forest, J48 and Majority Voting have the better results among all other classifiers in this set of tests. NaïveBayes has produced the highest false positive and false negative rates together with SMO. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↑**	↓	↑	↓**	↑
RF	↑	↑	↓	↓	↑
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↓	↑	↓**	↑*

Figure 178: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓**	↑	↓	↑**	↓
RF	↑*	↑	↓	↓*	↑
SMO	↓	↓	↑	↑	↓
MV	↓	↑	↓	↑	↑

Figure 179: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↑**	↓	↑	↓**	↑*
RF	↑	↑	↓	↓	↑*
SMO	↓	↓	↑	↑	↓
MV	↑*	↓	↑	↓*	↑

Figure 180: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑	↑	↓	↓	↑
RF	↑	↑	↓	↓	↑
SMO	↑	↑	↓	↓	↑*
MV	↑	↑	↓	↓	↑

Figure 181: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑*	↓*	↓	↑*
NB	↑	↓	↑	↓	↓
RF	↑	↑*	↓*	↓	↑**
SMO	↑**	↑*	↓*	↓**	↑**
MV	↑**	↑	↓	↓**	↑**

Figure 182: Results from adding numerical fingerprints to binary fingerprints for Substructure

Random Forest has a consistent improvement between all the other classifiers, followed by SMO and NaïveBayes.

Summary of the results and receiver operating characteristics analysis

At the end of this chapter we would like to summarise the observations made throughout the chapter and different set of tests for the Factor XA dataset. We have averaged the sensitivity and false positive rates across all fingerprints and then across all classifiers, once without the numerical descriptors and once with the numerical descriptors. Then we have plotted those points using the sensitivity and false positive rate as coordinates. The criterion for selecting the better method is the one closest to the top left corner of the graph, closest to the point (0,1). Figures 183 and 184 show this summarisation.

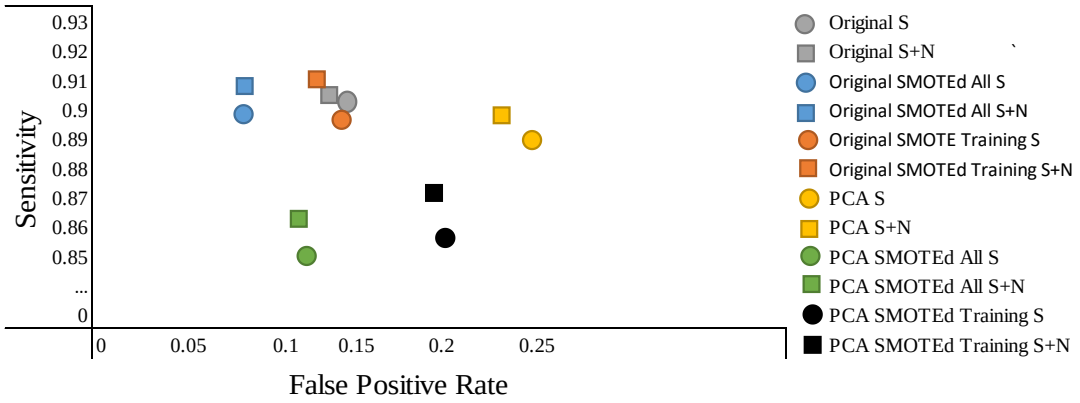


Figure 183: Sensitivity versus False Positive *Fontaine* methods

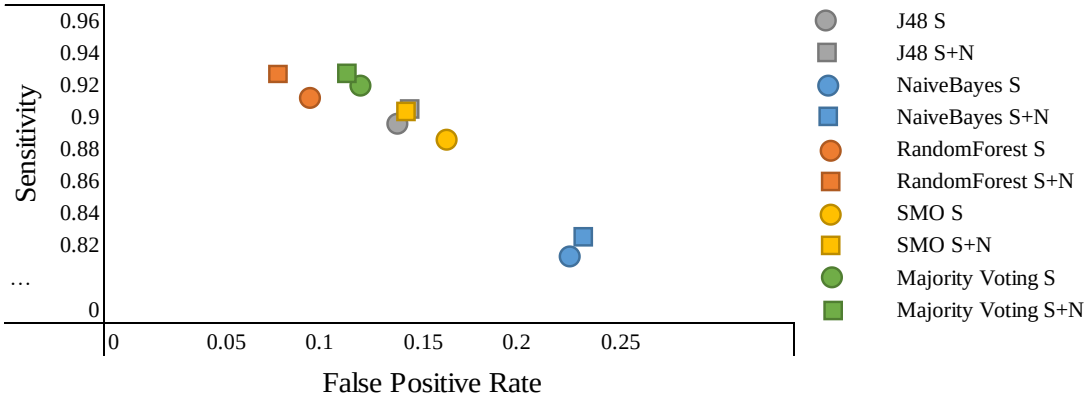


Figure 184: Sensitivity versus False Positive *Fontaine* classifiers

The distance of the resulting points to the point (0,1) is calculated using the Euclidean distance measure and the results are shown in Table 18 and Table 19. The points with the least distance have been bolded for the reader's attention.

Methods Used		**Euclidean Distance**
Binary Descriptors	Original	0.156
	Original SMOTEd All	**0.1235**
	Original SMOTEd Training	0.1579
	PCA	0.2497
	PCA SMOTEd All	0.1832
	PCA SMOTEd Training	0.2332
Binary + Numerical Descriptors	Original	0.1488
	Original SMOTEd All	**0.1139**
	Original SMOTEd Training	0.1447
	PCA	0.2414
	PCA SMOTEd All	0.1703
	PCA SMOTEd Training	0.2228

Table 18: Euclidean distance for the methods used

Classifiers Used		**Euclidean Distance**
Binary Descriptors	J48	0.1655
	NaïveBayes	0.2866
	Random Forest	**0.1257**
	SMO	0.1895
	Majority Voting	0.1423
Binary + Numerical Descriptors	J48	0.1666
	NaïveBayes	0.2826
	Random Forest	**0.1084**
	SMO	0.1673
	Majority Voting	0.1362

Table 19: Euclidean distance for the classifiers used

Conclusion

In this chapter we investigated the classification results for the Factor XA dataset. This dataset is a moderately imbalanced dataset and we performed some pre-processing in order to balance it. We observed the classification process results through 5 different methods. We classified the dataset as it was to begin with. Then we balanced the dataset and then split it into test and training set. In another method we split the dataset and then only balanced the training set. The three mentioned methods were repeated when the dataset dimensionality was reduced using the PCA method.

With this dataset, the different fingerprints behaved differently in the presence of the classifiers. Overall the fingerprint MACCS and then PubChem showed to be the ones with the better performances. Again the reader must be reminded that there were no consistently better performing fingerprints.

In the classifiers, Random Forest was definitely the better performing classifier for this set of tests and the one that benefitted most from the addition of the numerical descriptors.

In the next chapter we shall be looking at the datasets with a significantly higher imbalance ratios that were used for this study.

5.3. The Heavily Imbalanced Dataset – AID362

So far in our study we have investigated the Mutagenicity dataset (Kazius et al. 2005) and the Factor XA dataset (Fontaine et al. 2005). The first one was marginally imbalanced and the second one moderately imbalanced. The next two datasets to be studied are greatly imbalanced datasets. The first dataset we present here is the Formylpeptide Receptor Ligand Binding Assay. For the purposes of making it easier for the reader we will refer to it simply as AID362.

This dataset is a whole-cell assay for another inhibitor of peptide binding associated with tissue-damaging chronic inflammation (Jabed et al. 2015). This dataset has been described as a contributor to the localization and activation of tissue-damaging leukocytes at sites of chronic inflammation.

The number of instances, active and inactive and the imbalance ratio information can be found in the table below. The dataset is highly imbalanced, with an imbalance ratio of 1.4%.

Dataset	#Total Instances	#Active Instances (class '1')	#Inactive Instances (class '0')	Active/Inactive Ratio
AID362	4279	60	4219	0.0142

Table 20: AID362 dataset specification. Class of interest labelled as 1

In the next section, we classify the original dataset and show the classification metrics used.

AID362 Classification Results per Fingerprint– Original

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

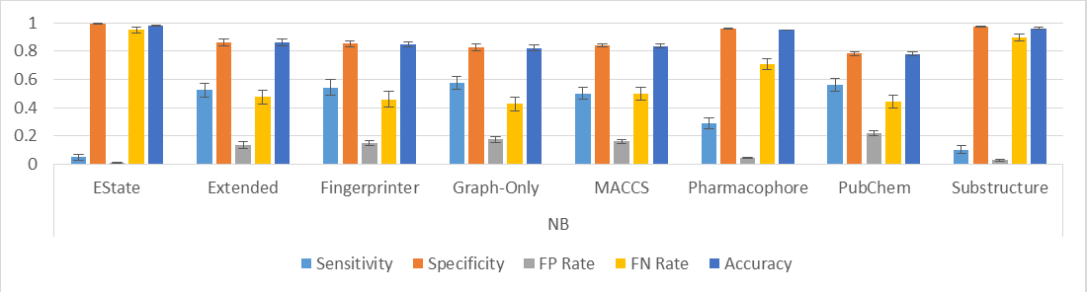


Figure 185: Classification results from classifying the AID362 dataset by NaïveBayes

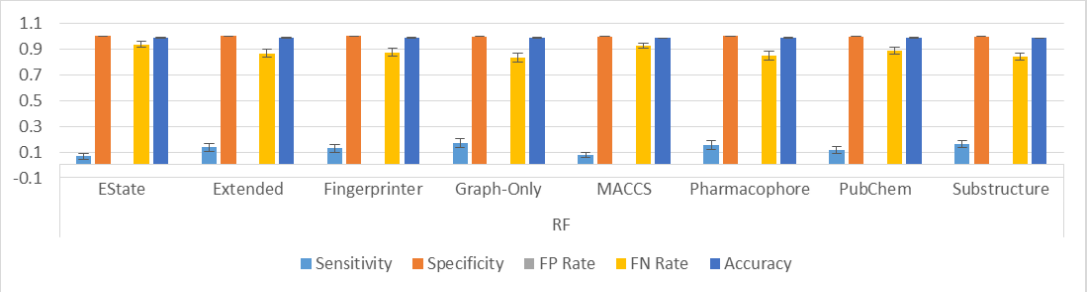


Figure 186: Classification results from classifying the AID362 dataset by Random Forest

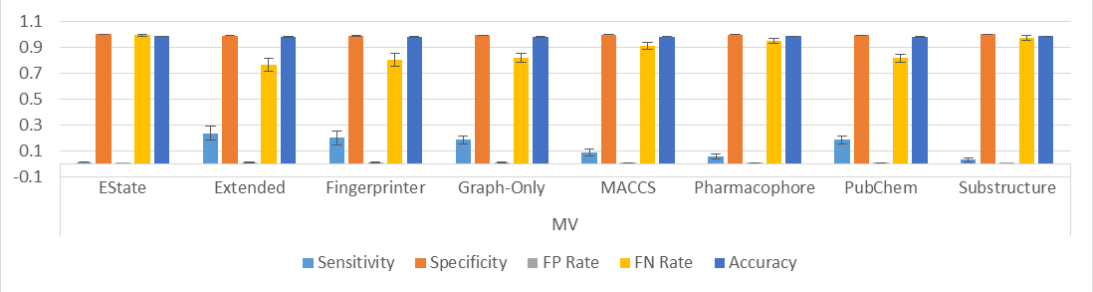


Figure 187: Classification results from classifying the AID362 dataset by Majority Voting

In this section the dataset AID362 has been classified in its original state. No pre-processing techniques were used. We look at how different fingerprints performed in the presence of each of our classifiers. From looking at the Figure 185 - Figure 187 we can see that except for EState, Pharmacophore and Substructure fingerprints, all other ones have more varied results with NaïveBayes and with the other classifiers the results seem very skewed and almost biased. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↓	↑*	↓*	↑	↑
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↓**	↑*	↓*	↑**	↑
PubChem	↑	↑*	↓*	↓	↑**
Substructure	↓**	↑**	↓**	↑**	↑**

Figure 188: Results from adding numerical fingerprints to binary fingerprints for Random Forest

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓**	↑**	↓**	↓**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↑	↓	↑	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓	↑	↓	↓
Pharmacophore	↑*	↓**	↑**	↓*	↓**
PubChem	↓	↓	↑	↑	↓
Substructure	↑**	↓**	↑**	↓**	↓**

Figure 189: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

By adding numerical descriptors we do not see and fingerprints improve in a consistent manner. But we do observe a significant improvement in specificity, accuracy and false positive rates when using Random Forest. This is followed by Majority Voting with improvements in sensitivity and false negative rate. In the next section, we classify the original dataset and show the classification metrics used.

AID362 Classification Results per Classifiers – Original

In this section we look at how various classifiers performed. Each figure represents one separate fingerprint.

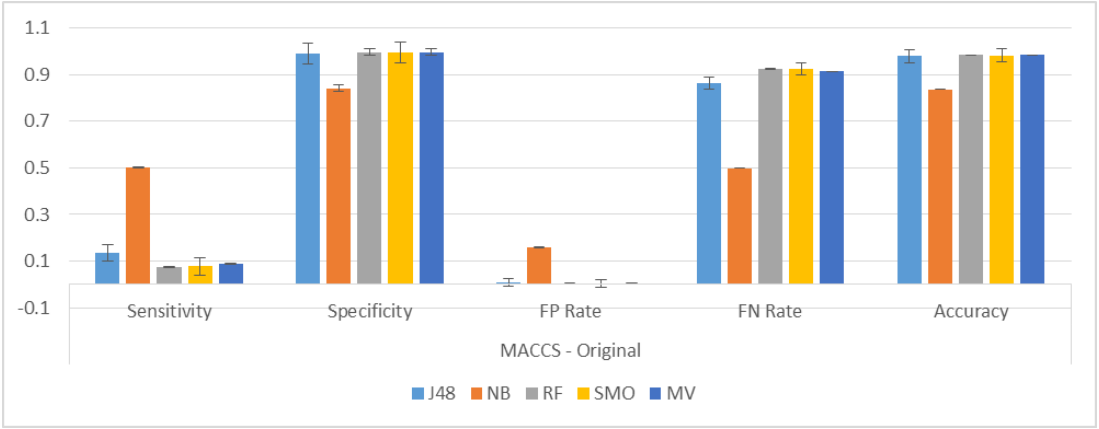


Figure 190**:** Classifier performance for MACCS – Original

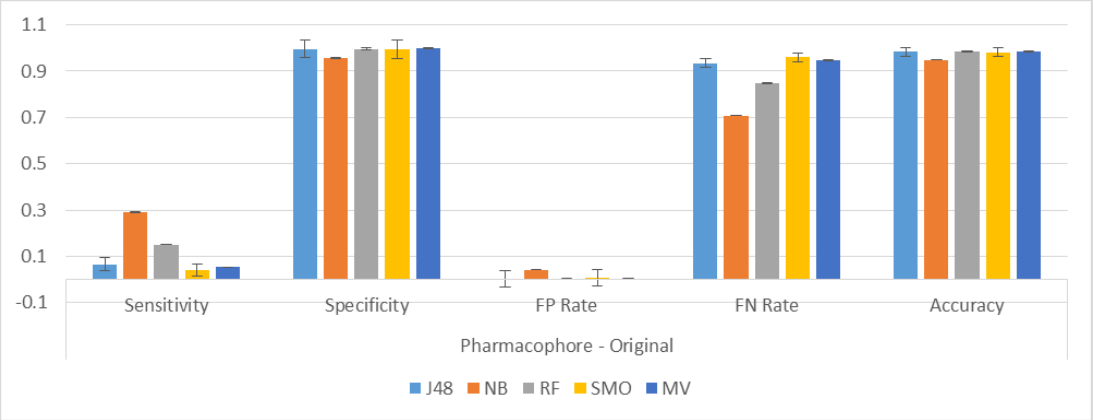


Figure 191: Classifier performance for Pharmacophore – Original

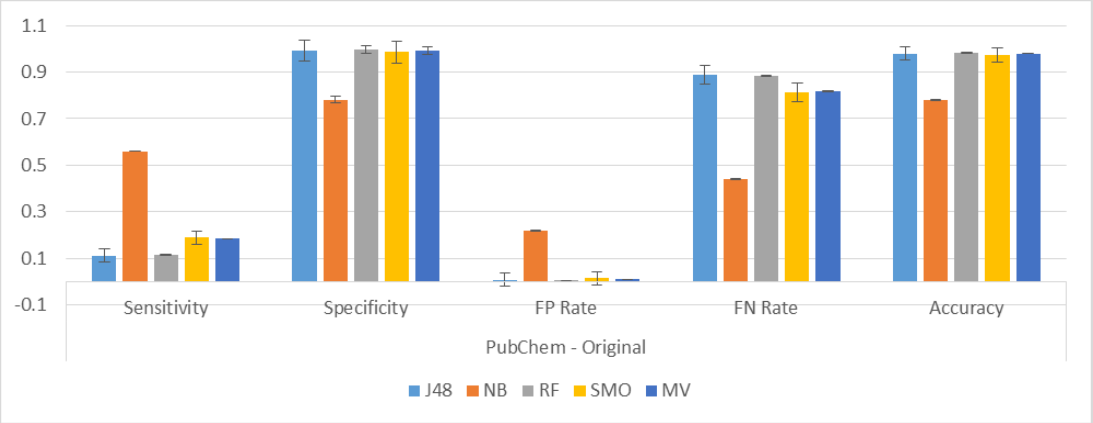


Figure 192: Classifier performance for PubChem – Original

Here we see that with MACCS, Pharmacophore and PubChem, NaïveBayes seems to be the classifier that has produced results different to all other classifiers. It has a higher sensitivity and false positive rate and a lower false negative, specificity and accuracy rates compared to the other classifiers. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↓**	↑**	↓**	↓*
NB	↑	↓**	↑**	↓	↓**
RF	↓**	↑*	↓*	↑**	↑
SMO	↑*	↓	↑	↓*	↓
MV	↑*	↓**	↑**	↓*	↓**

Figure 193: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↓	↓	↑	↑	↓
RF	↑	↑*	↓*	↓	↑**
SMO	↑	↑	↓	↓	↑
MV	↓	↓	↑	↑	↓

Figure 194: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑*	↓**	↑**	↓*	↓**
NB	↑**	↓**	↑**	↓**	↓**
RF	↓**	↑**	↓**	↑**	↑**
SMO	↑	↓**	↑**	↓	↓*
MV	↑**	↓**	↑**	↓**	↓**

Figure 195: Results from adding numerical fingerprints to binary fingerprints for Substructure

Adding numerical descriptors has certainly improved the sensitivity and false negative rates in these set of tests. If one classifier could be named as the most improved significantly, it would be Random Forest. The figures shown are related to the fingerprints with the most significant improvements. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

AID362 Classification Results per Fingerprint– Original SMOTEd All

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results. The dataset AID362 has been pre-processed here by balancing using SMOTE technique first and then splitting it into training (60%) and test (40%).

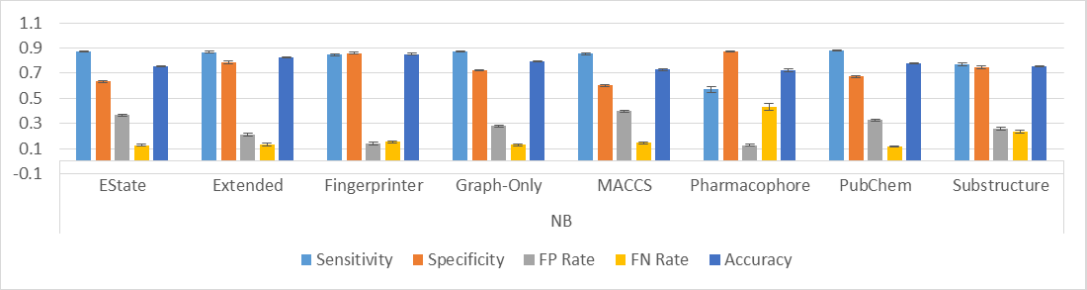


Figure 196: Classification results from classifying the AID362 dataset by NaïveBayes

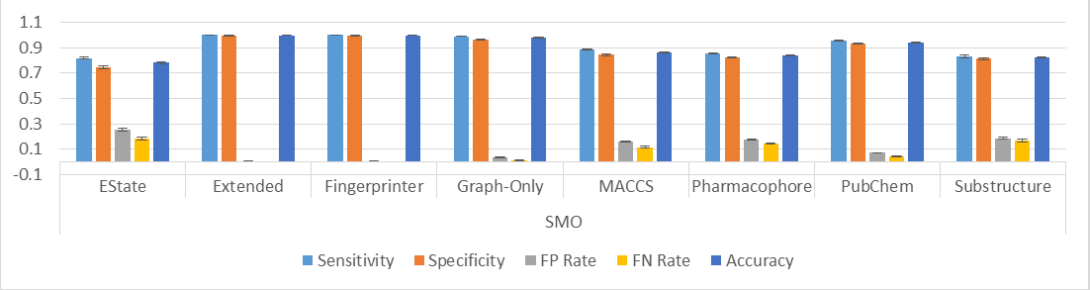


Figure 197: Classification results from classifying the AID362 dataset by SMO

The fingerprints in these set of tests have less biased results when combined with SMO and NaïveBayes, except for the three CDK fingerprints in SMO; Extended Fingerprinter, Fingerprinter and Graph-Only. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓*	↑	↓	↑*	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↓	↑**	↓**	↑	↑**
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↓*	↑*	↓*	↑*	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 198: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↓**	↑**	↓**	↑**	↑**
Fingerprinter	↑	↑*	↓*	↓	↑**
Graph-Only	↓	↑	↓	↑	↓
MACCS	↑**	↑**	↓**	↓**	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↓**	↑**	↓**	↑**	↓
Substructure	↑	↑**	↓**	↓	↑**

Figure 199: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↓	↑*	↓*	↑	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 200: Results from adding numerical fingerprints to binary fingerprints for Random Forest

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↑*	↓*	↑	↑
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↓	↑**	↓**	↑	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↑	↑*	↓*	↓	↑*
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 201: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

The one thing that stands out with these results from Figure 198 - Figure 201 is that we see great improvements in specificity, false positive and accuracy rates, except for with SMO. With Majority Voting the improvements are not as significant. Almost all fingerprints show great improvement with Random Forest. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

AID362 Classification Results per Classifiers– Original SMOTEd All

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

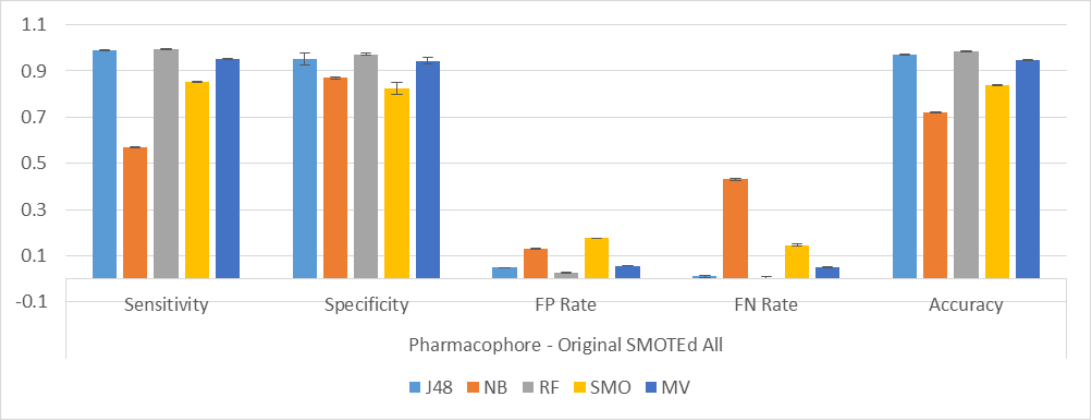


Figure 202: Classifier performance for Pharmacophore – Original SMOTEd All

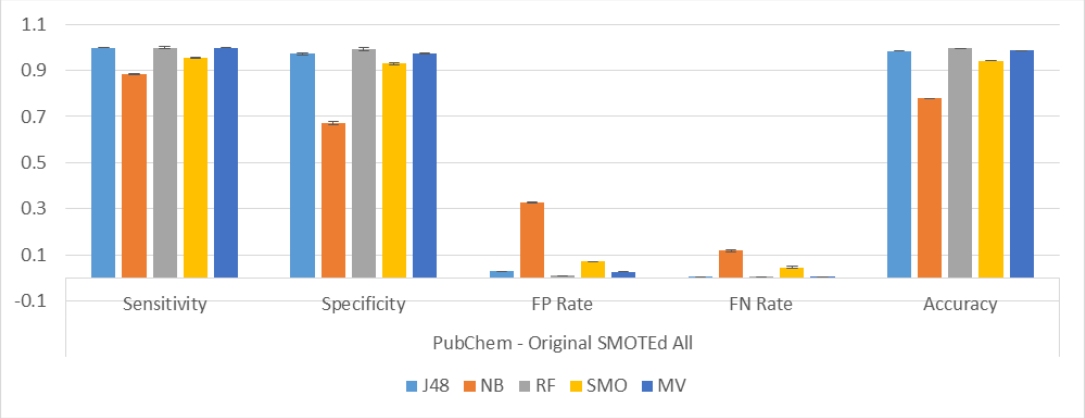


Figure 203: Classifier performance for PubChem – Original SMOTEd All

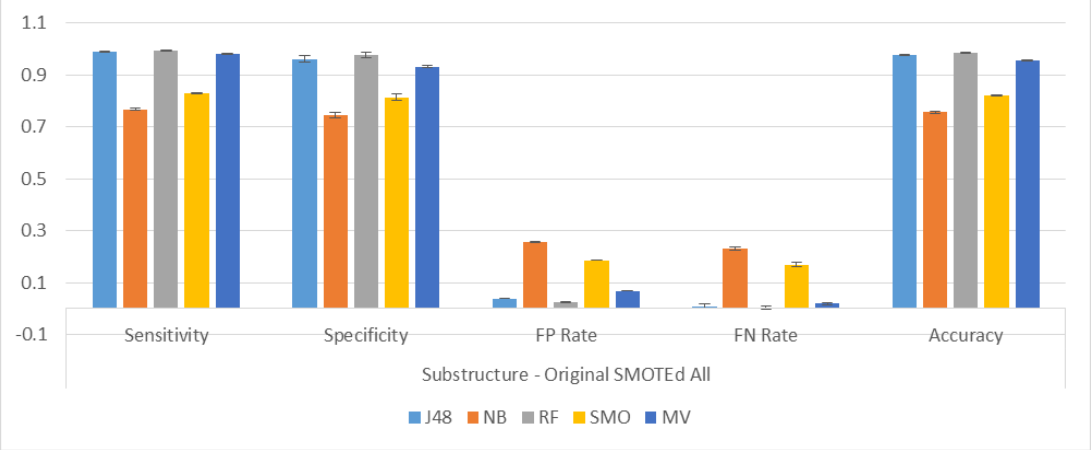


Figure 204: Classifier performance for Substructure – Original SMOTEd All

Looking at Figure 202 - Figure 204 we see that the classifiers J48, Random Forest and Majority Voting have produced better results especially with false positive and false negative. Pharmacophore fingerprint has especially good false positive results. However one should keep in mind that the dataset has been balanced so it might also be the case that the very good results are actually extremely biased

towards the majority class. It has been shown that SMOTE on occasion can cause overfitting (Kumar & Ravi 2008; Fernández-Navarro et al. 2011; Maldonado & López 2014). In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑**	↓**	↓	↑**
NB	↑**	↑**	↓**	↓**	↑**
RF	↑	↑**	↓**	↓	↑**
SMO	↑**	↑	↓	↓**	↑**
MV	↓	↑**	↓**	↑	↑**

Figure 205: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↑**	↑**	↓**	↓**	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 206: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↑	↑**	↓**	↓	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 207: Results from adding numerical fingerprints to binary fingerprints for Substructure

Results show impressive improvements from adding numerical descriptors to binary only ones. Not all classifiers show consistent improvement but the specificity, false positive rates show great and significant improvements, especially with MACCS, Pharmacophore and Substructure fingerprints. In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

AID362 Classification Results per Fingerprint– Original SMOTEd Training

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results. With these set of tests the dataset was initially split into training (60%) and test (40%) and then only the training part was balanced using SMOTE technique. The test set was left intact.

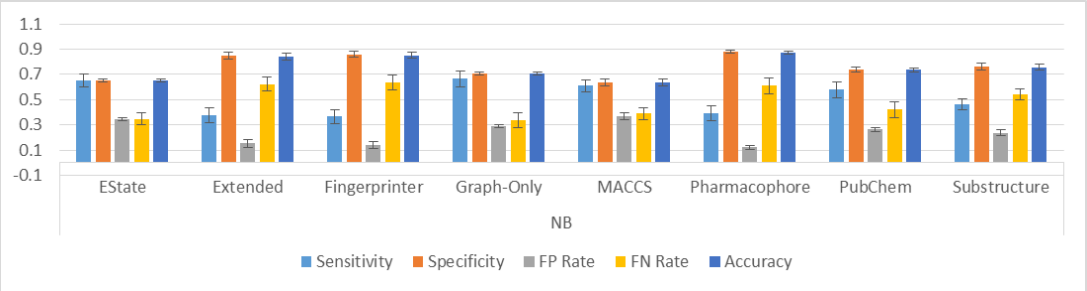


Figure 208: Classification results from classifying the AID362 dataset by NaïveBayes

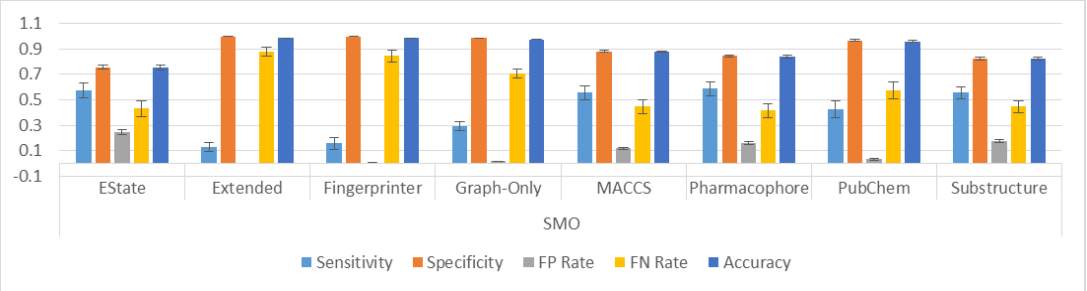


Figure 209: Classification results from classifying the AID362 dataset by SMO

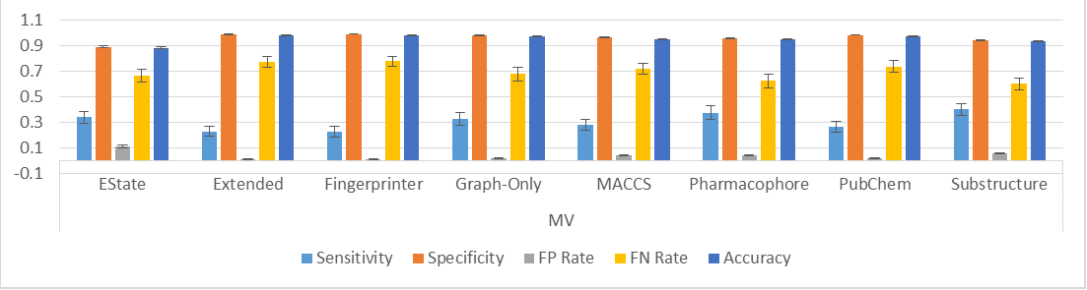


Figure 210: Classification results from classifying the AID362 dataset by Majority Voting

Results in this section contrast the results from the previous section (Original SMOTEd All) in that the sensitivity levels are much lower and the false negative rates are higher. In the next section, we will observe how adding numerical fingerprints affects our classification results and whether the changes are statistically significant or not.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↑*	↓*	↑	↑*
Graph-Only	↓	↑**	↓**	↑	↑*
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↓**	↑**	↓**	↑**	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↓**	↑**	↓**	↑**	↑**

Figure 211: Results from adding numerical fingerprints to binary fingerprints for J48

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↓	↑**	↓**	↑	↑**
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↓**	↑**	↓**	↑**	↑**
PubChem	↑	↑*	↓*	↓	↑*
Substructure	↓	↑**	↓**	↑	↑**

Figure 212: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↓	↑	↓	↓
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓	↑*	↓*	↑	↑*
PubChem	↑	↑	↓	↓	↑
Substructure	↓	↑**	↓**	↑	↑**

Figure 213: Results from adding numerical fingerprints to binary fingerprints for SMO

The specificity, false positive and accuracy levels show promising significant improvements throughout Figure 211 - Figure 213. Pharmacophore has produced less than optimal results with most of the classifiers. In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

Classification Results per Classifier– Original SMOTEd Training

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

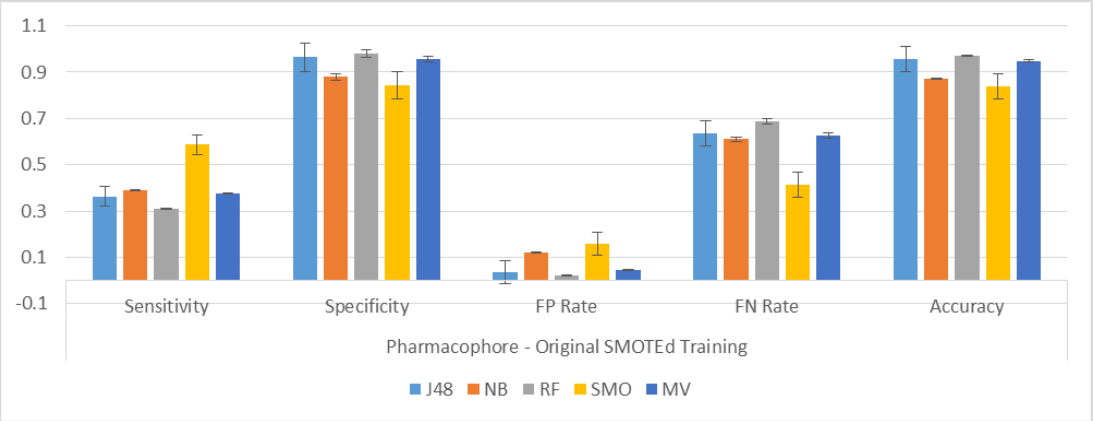


Figure 214: Classifier performance for Pharmacophore – Original SMOTEd Training

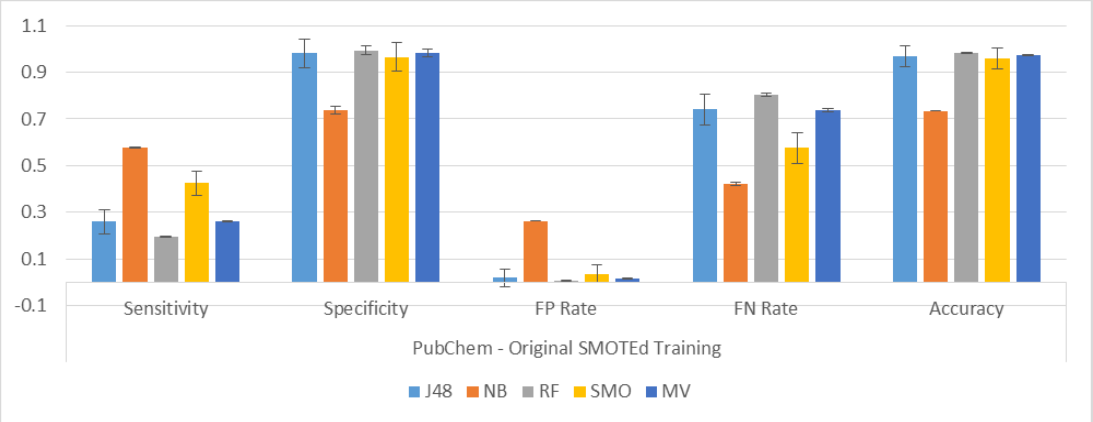


Figure 215: Classifier performance for PubChem – Original SMOTEd Training

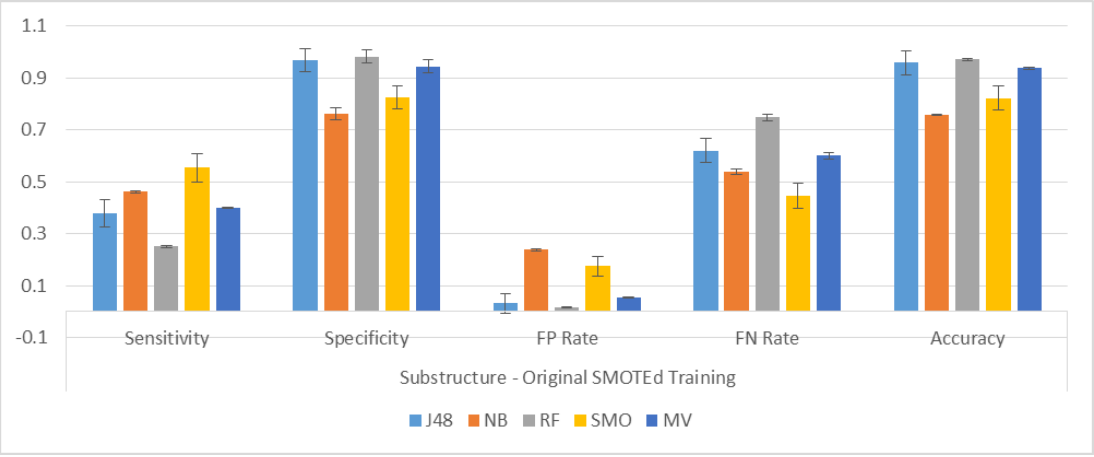


Figure 216: Classifier performance for Substructure – Original SMOTEd Training

NaïveBayes and SMO have given us the least optimal results from the group of classifiers used. They have higher false positive rates than the other ones. On the other hand J48, Random Forest and Majority Voting have good results overall. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑**	↓**	↓	↑**
NB	↓**	↑**	↓**	↑**	↑**
RF	↓	↑**	↓**	↑	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↑	↑**	↓**	↓	↑**

Figure 217: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑	↑*	↓*	↓	↑*
RF	↑	↑**	↓**	↓	↑**
SMO	↓	↑	↓	↑	↑
MV	↑	↑**	↓**	↓	↑**

Figure 218: Results from adding numerical fingerprints to binary fingerprints for MACCS

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓**	↑**	↓**	↑**	↑**
NB	↓	↑	↓	↑	↑
RF	↓	↑**	↓**	↑	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↓	↑**	↓**	↑	↑**

Figure 219: Results from adding numerical fingerprints to binary fingerprints for Substructure

Specificity, false positive and accuracy have benefited from the addition of numerical descriptors. However sensitivity and false negative rates have declined especially with Substructure and EState fingerprints. In the next section, we classify the original dataset with PCA and show the classification metrics used.

AID362 Classification Results per Fingerprint– PCA Original

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results. The PCA feature selection technique was used here.

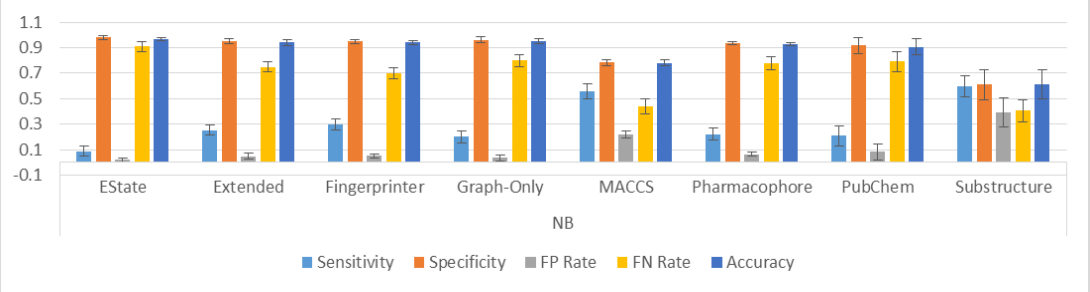


Figure 220: Classification results from classifying the AID362 dataset by NaïveBayes

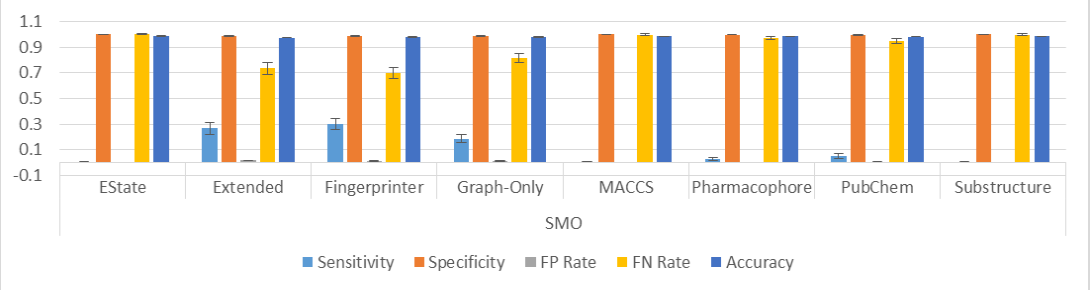


Figure 221: Classification results from classifying the AID362 dataset by SMO

In these set of tests, ones with PCA incorporated, the dimensionality of our dataset (AID362) has been reduced. The dataset was exposed to the classifiers in its original state. By looking at Figure 220 and Figure 221 we see that MACCS, Substructure and the CDK Fingerprinter family (Extended Fingerprinter, Fingerprinter and Graph-Only) have produced better sensitivity results, however the false positive results are slightly higher than desired with NaïveBayes than SMO. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑*	↓*	↑	↑
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↓	↑	↓	↑	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓*	↑**	↓**	↑*	↑
PubChem	↓	↑	↓	↑	↑
Substructure	↓*	↑**	↓**	↑*	↑**

Figure 222: Results from adding numerical fingerprints to binary fingerprints for Random Forest

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↓	↑	↑	↓
Extended	↑*	↑	↓	↓*	↑
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓	↑	↓	↓
Pharmacophore	↑*	↓*	↑*	↓*	↓
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓*	↑*	↓	↓

Figure 223: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

When adding numerical descriptors for the PCA results, we see that with Random Forest (Figure 222) there are improvements in Specificity, false positive and accuracy rates, some of which are significant. With Majority Voting (Figure 223), Sensitivity and false negative improve, yet not much significant improvement either. In the next section, we classify the original dataset and show the classification metrics used. Note that PCA has been applied to the original dataset.

AID362 Classification Results per Classifiers– PCA Original

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

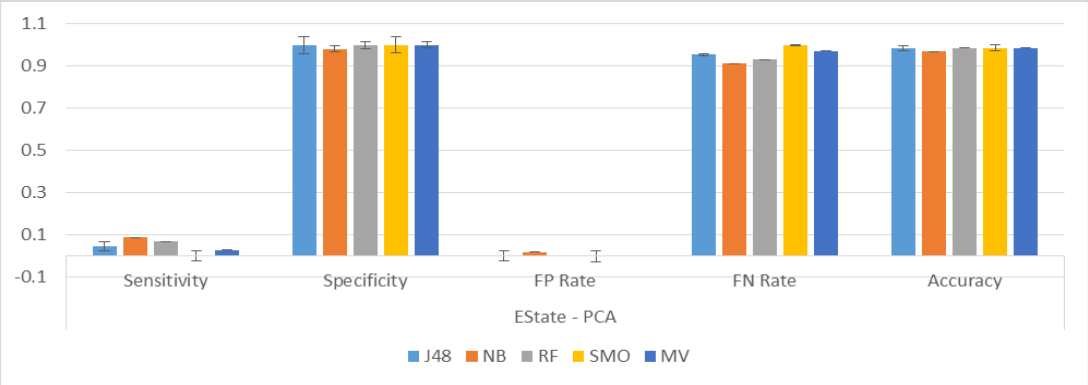


Figure 224: Classifier performance for EState - PCA

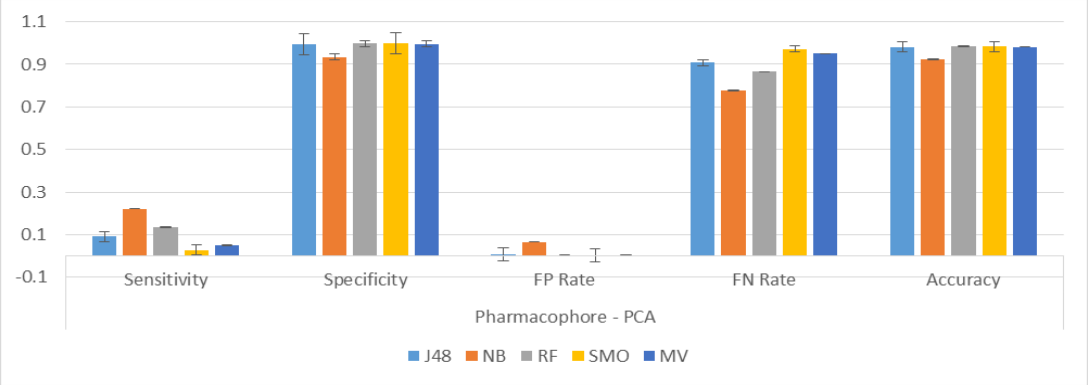


Figure 225: Classifier performance for Pharmacophore - PCA

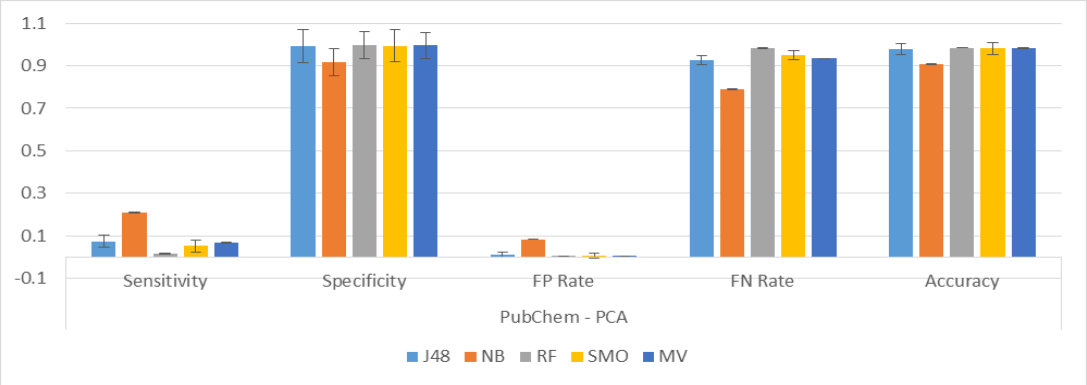


Figure 226: Classifier performance for PubChem - PCA

From reviewing Figure 224 - Figure 226 we see that almost all classifiers have produced good metrics (apart from sensitivity which is very low) except for

NaïveBayes which has higher false positive results than the others. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↑	↑	↓	↓	↑
RF	↓*	↑**	↓**	↑*	↑
SMO	↑	↓	↑	↓	↓
MV	↑*	↓*	↑*	↓*	↓

Figure 227: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓	↓	↑	↑	↓
RF	↓*	↑**	↓**	↑*	↑**
SMO	↑	↓	↑	↓	↓
MV	↑	↓*	↑*	↓	↓

Figure 228: Results from adding numerical fingerprints to binary fingerprints for Substructure

The classifiers did not benefit much from the addition of the numerical descriptors when used in combination with the PCA and the dataset in its original imbalanced state. Only Random Forest shows partial significant improvement for specificity and false positive rates. In the next section, we classify the dataset that was balanced before splitting with PCA and show the classification metrics used.

AID362 Classification Results per Fingerprint– PCA SMOTEd All

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

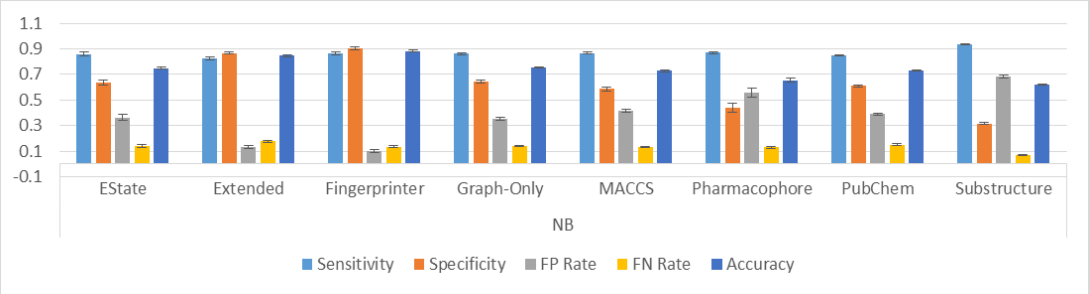


Figure 229: Classification results from classifying the AID362 dataset by NaïveBayes

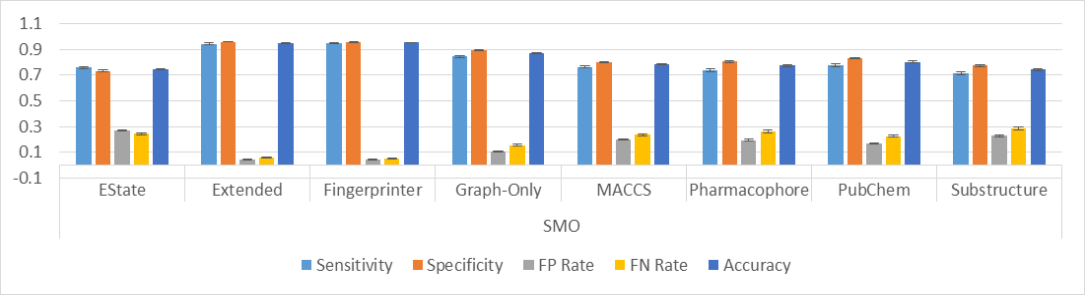


Figure 230: Classification results from classifying the AID362 dataset by SMO

After the dimensionality of the original dataset was reduced using PCA it was balanced using SMOTE and then split into training and test sets (60% and 40%). We see a rise in sensitivity (compared to PCA-only method, previous section) and a drop in the false negative rate. However the false positive rates have risen with EState, MACCS, Pharmacophore and Substructure in Figure 229. Results seem to be better using the fingerprints with SMO (Figure 230). In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↓	↑	↑	↓*
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↑	↓	↑	↓
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↑	↑**	↓**	↓	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 231: Results from adding numerical fingerprints to binary fingerprints for J48

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↑	↓	↑	↓
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 232: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↑*	↑	↓	↓*	↑*
Fingerprinter	↓	↑	↓	↑	↓
Graph-Only	↑**	↑	↓	↓**	↑**
MACCS	↑**	↑**	↓**	↓**	↑**
Pharmacophore	↑**	↓*	↑*	↓**	↑**
PubChem	↑*	↓	↑	↓*	↑
Substructure	↑**	↓	↑	↓**	↑**

Figure 233: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓*	↑**	↓**	↑*	↑*
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↑*	↑**	↓**	↓*	↑**
PubChem	↑	↑*	↓*	↓	↑
Substructure	↑**	↑	↓	↓**	↑**

Figure 234: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

Improvement in the metrics as a result of adding numerical descriptors has not been consistent throughout Figure 231 - Figure 234. We see specificity and false positive rates improve with J48, Random Forest and Majority Voting. With SMO there is more significant improvement for sensitivity, false negative and accuracy. In

the next section, we classify the dataset that was balanced before splitting with PCA and show the classification metrics used.

AID362 Classification Results per Classifiers– PCA SMOTEd All

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

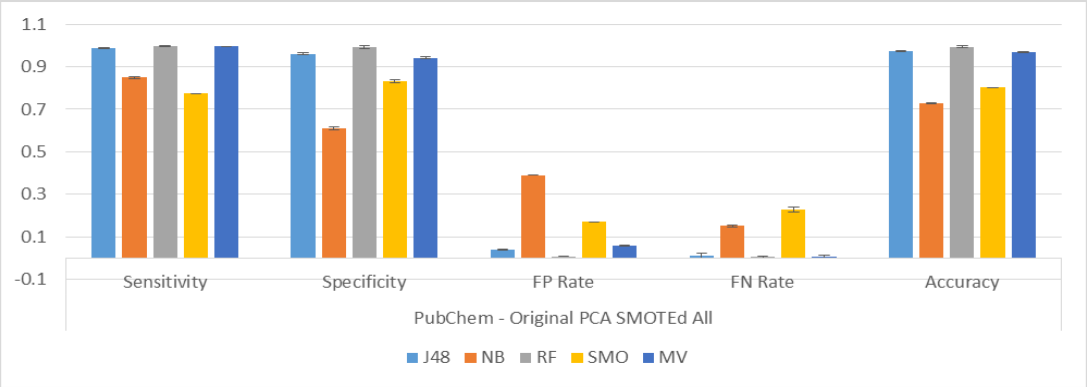


Figure 235: Classifier performance for PubChem – PCA SMOTEd All

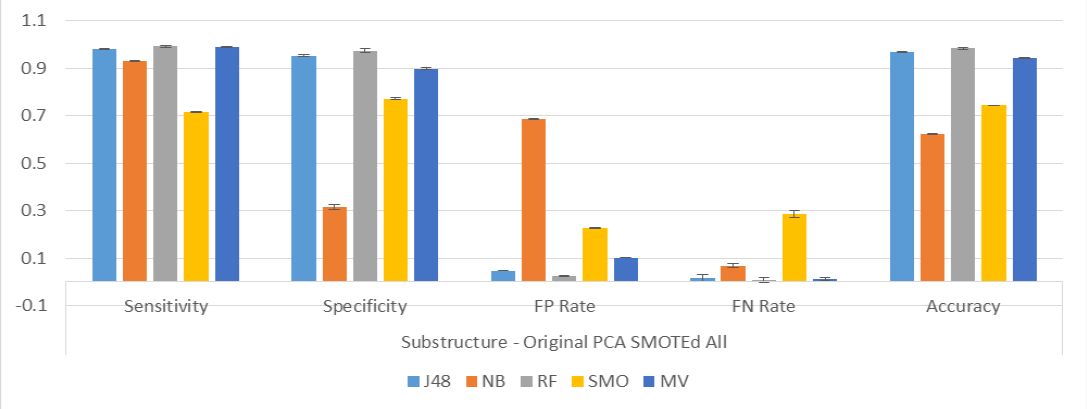


Figure 236: Classifier performance for Substructure – PCA SMOTEd All

NaïveBayes has consistently produced the highest false positive rates in these tests, especially with SMO, in contrast to J48, Random Forest and Majority Voting which have the better results especially when used in combination with PubChem. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↓**	↑**	↓**	↑**	↑
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑	↑**	↓**	↓	↑**
MV	↓**	↑**	↓**	↑**	↑**

Figure 237: Results from adding numerical fingerprints to binary fingerprints for EState

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑**	↓**	↓	↑**
NB	↑**	↑	↓	↓**	↑**
RF	↑	↑	↓	↓	↑
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑	↑**	↓**	↓	↑**

Figure 238: Results from adding numerical fingerprints to binary fingerprints for MACCS

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑**	↓**	↓	↑**
NB	↑**	↑	↓	↓**	↑
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑**	↓*	↑*	↓**	↑**
MV	↑*	↑**	↓**	↓*	↑**

Figure 239: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↑**	↓**	↑**	↓**	↓*
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑**	↓	↑	↓**	↑**
MV	↑**	↑	↓	↓**	↑**

Figure 240: Results from adding numerical fingerprints to binary fingerprints for Substructure

Adding numerical descriptors has shown great improvements in the performance of our classifiers (Figure 237 -Figure 240). Classifiers have all improved with MACCS fingerprint and the improvement with the other fingerprints is mixed with regards to the classification metrics. The most significant improvements are with sensitivity, false negative and accuracy. In the next section,

we classify the dataset where only training set has been balanced and show the classification metrics used.

AID362 Classification Results per Fingerprint– PCA SMOTEd Training

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

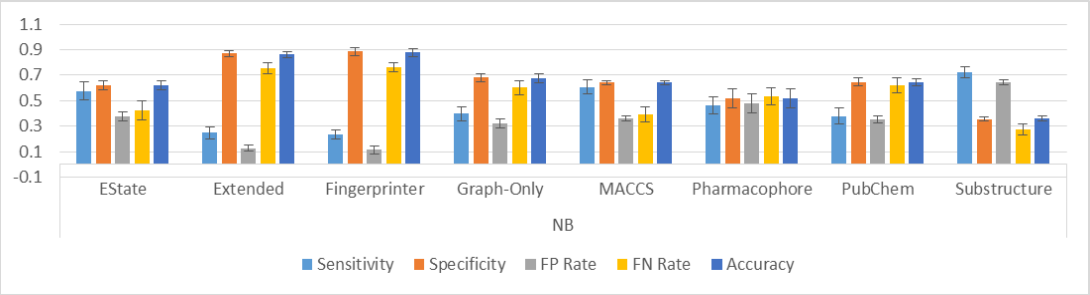


Figure 241: Classification results from classifying the AID362 dataset by NaïveBayes

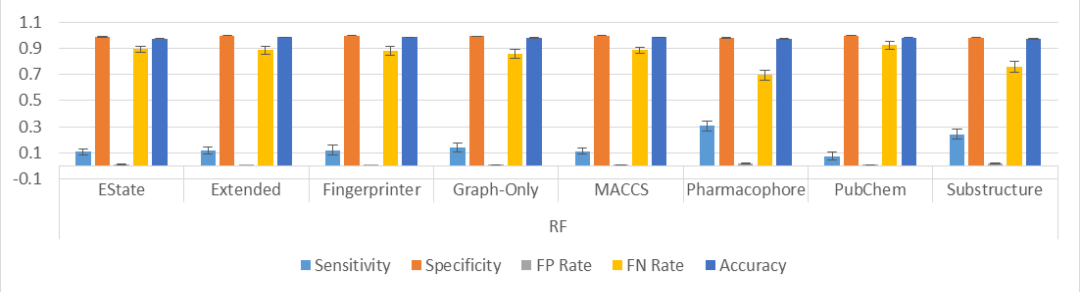


Figure 242: Classification results from classifying the AID362 dataset by Random Forest

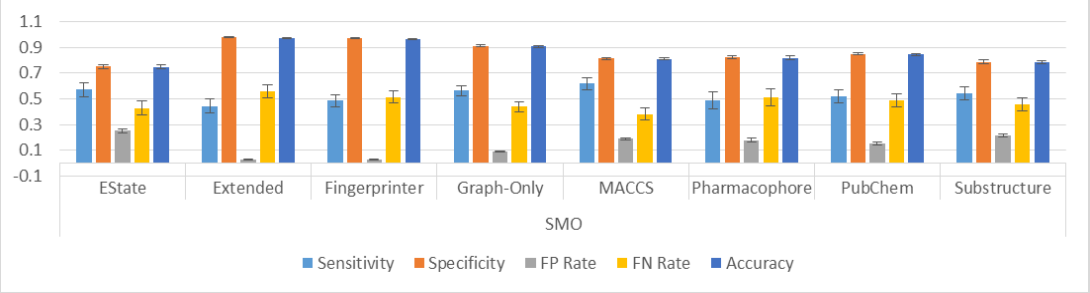


Figure 243: Classification results from classifying the AID362 dataset by SMO

The dimensionality-reduced AID362 was split into training and test set (60%-40%) first and then only the training set was balanced using SMOTE. The results show a better variance when used with SMO and NaïveBayes. With the other classifiers the results show bias towards the majority class (extremely high

specificity rates) as seen in Figure 242. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓*	↑*	↓	↓*
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↑**	↓**	↑	↑**
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓**	↑**	↓**	↑**	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↓	↑**	↓**	↑	↑**

Figure 244: Results from adding numerical fingerprints to binary fingerprints for J48

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↑**	↓**	↑	↑**
MACCS	↓*	↑	↓	↑*	↑
Pharmacophore	↓**	↑**	↓**	↑**	↑**
PubChem	↑	↓	↑	↓	↑
Substructure	↓	↑**	↓**	↑	↑**

Figure 245: Results from adding numerical fingerprints to binary fingerprints for Random Forest

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↑*	↓*	↓	↑*
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓*	↑**	↓**	↑*	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↓	↑*	↓*	↑	↑*

Figure 246: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

The most improvement which is also significant can be seen with specificity, false positive and accuracy when used with J48, Random Forest and Majority Voting when observing Figure 244 - Figure 246. In the next section, we classify the dataset where only training set has been balanced with PCA and show the classification metrics used.

AID362 Classification Results per Classifiers– PCA SMOTEd Training

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

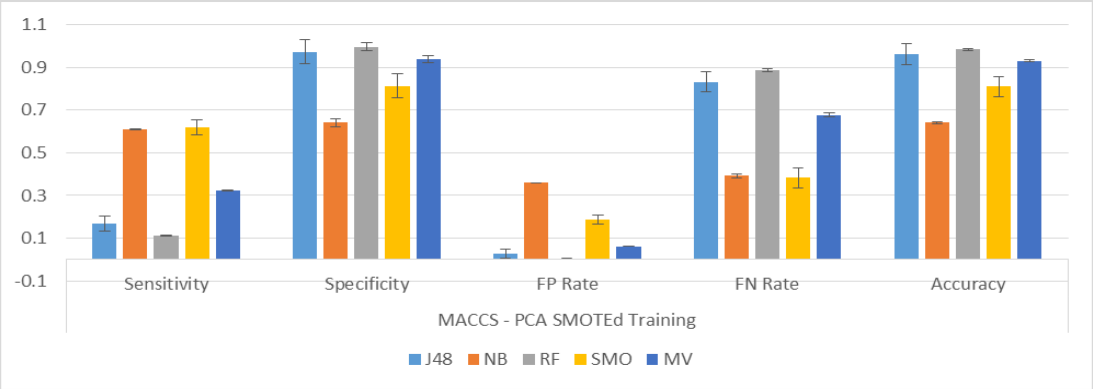


Figure 247: Classifier performance for MACCS – PCA SMOTEd Training

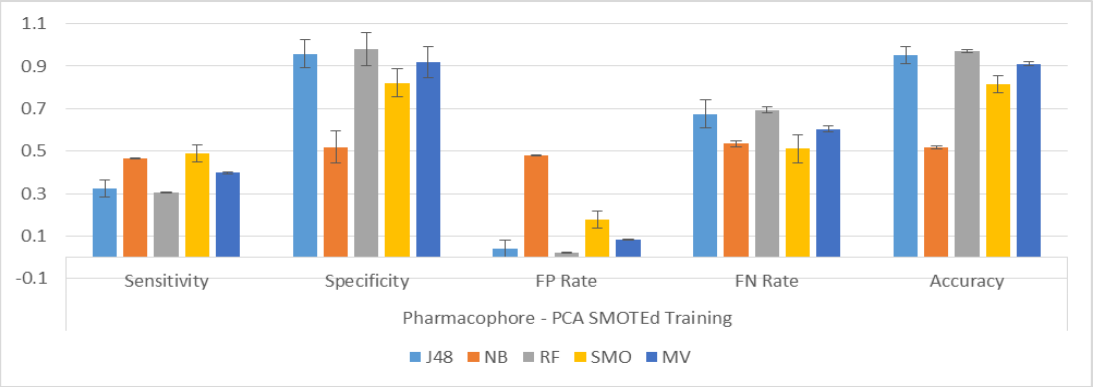


Figure 248: Classifier performance for Pharmacophore – PCA SMOTEd Training

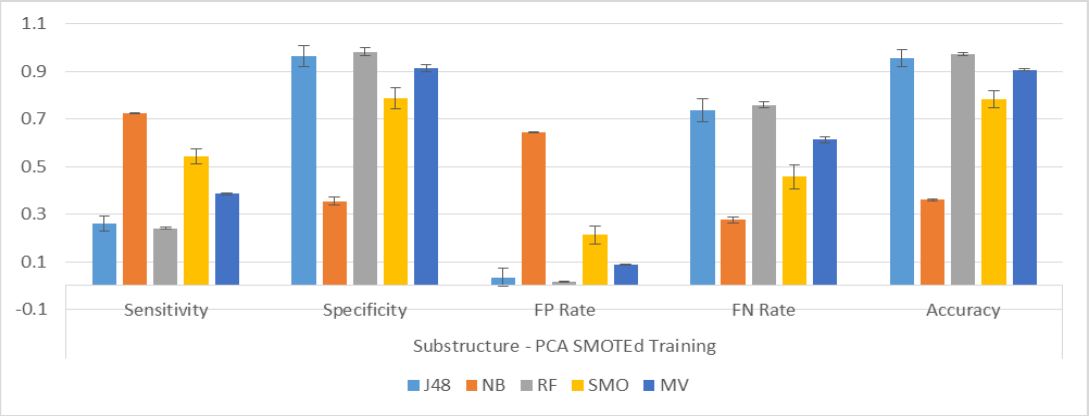


Figure 249: Classifier performance for Substructure – PCA SMOTEd Training

NaïveBayes and SMO stand out as the two classifiers with a higher sensitivity but also higher false positive rates compared to the other classifiers in these set of tests. Pharmacophore seems to have produced the better results compared to the other fingerprints when used with the classifiers (Figure 248). In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓*	↑*	↓	↓*
NB	↓*	↑**	↓**	↑*	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↑	↓	↑	↓	↓
MV	↑	↑**	↓**	↓	↑**

Figure 250: Results from adding numerical fingerprints to binary fingerprints for EState

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓**	↑**	↓**	↑**	↑**
NB	↑	↓	↑	↓	↓
RF	↓**	↑**	↓**	↑**	↑**
SMO	↓	↓	↑	↑	↓
MV	↓*	↑**	↓**	↑*	↑**

Figure 251: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑**	↓**	↑	↑**
NB	↓	↓	↑	↑	↓
RF	↓	↑**	↓**	↑	↑**
SMO	↑	↓	↑	↓	↓
MV	↓	↑*	↓*	↑	↑*

Figure 252: Results from adding numerical fingerprints to binary fingerprints for Substructure

Results from this section (Figure 250 - Figure 252) show that there has been a significant improvement on specificity, false positive and accuracy rates for J48, Random Forest and Majority Voting, also for NaïveBayes when used with EState.

Summary of the results and receiver operating characteristics analysis

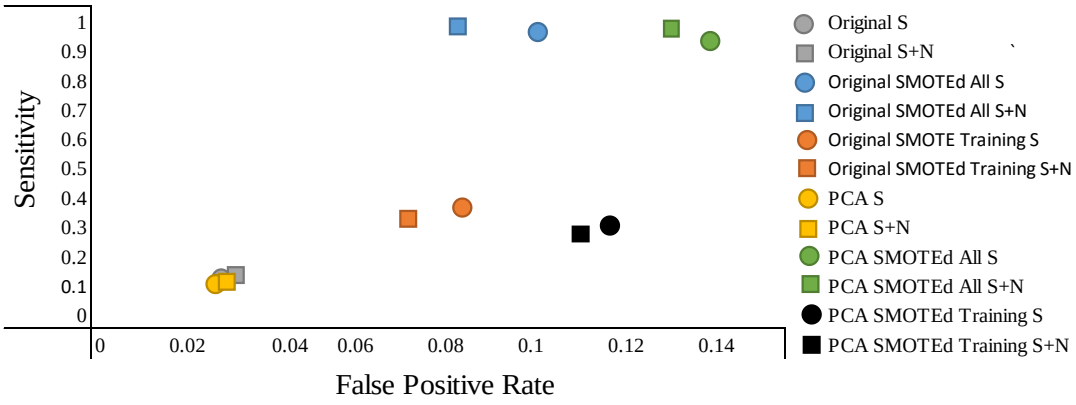


Figure 253: Sensitivity versus False Positive *AID362* methods

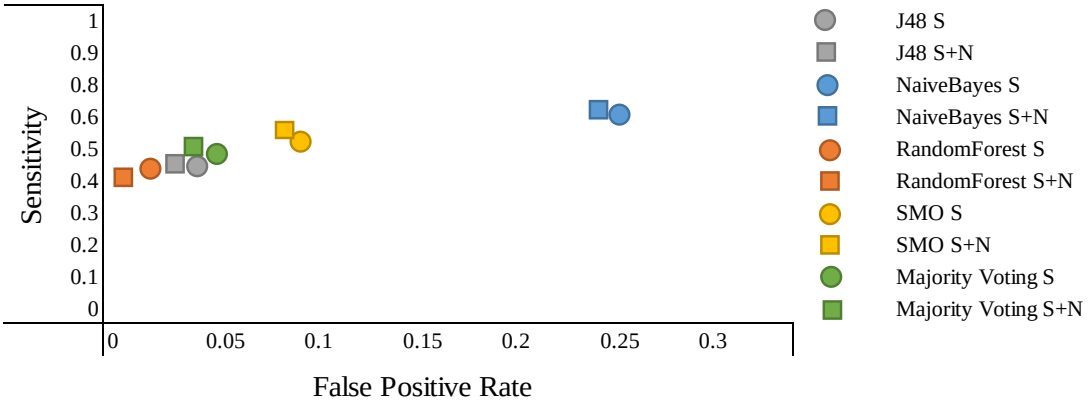


Figure 254: Sensitivity versus False Positive *AID362* classifiers

Methods Used		**Euclidean Distance**
Binary Descriptor	Original	0.8175
	Original SMOTEd All	**0.1127**
	Original SMOTEd Training	0.6564
	PCA	0.8669
	PCA SMOTEd All	0.1499
	PCA SMOTEd Training	0.6895
Binary + Numerical Descriptors	Original	0.7904
	Original SMOTEd All	0.0973
	Original SMOTEd Training	0.6706
	PCA	0.8658
	PCA SMOTEd All	**0.1401**
	PCA SMOTEd Training	0.705

Table 21: Euclidean distance for the methods used

Classifiers Used		**Euclidean Distance**
Binary Descriptors	J48	0.5511
	NaïveBayes	0.5745
	Random Forest	**0.5043**
	SMO	0.5214
	Majority Voting	0.5411
Binary + Numerical Descriptors	J48	0.5521
	NaïveBayes	0.5828
	Random Forest	**0.4976**
	SMO	0.5149
	Majority Voting	0.5368

Table 22: Euclidean distance for the classifiers used

By looking at Table 21 and Table 22 we can see that on average, the method that performed best and is closest to the point (0,1) as seen in Figure 253 is when the dataset was initially balanced and then split into training and test sets. This condition is valid both when the dataset is high-dimensional and also when the dimensionality is reduced using the PCA method and numerical descriptors are added to the dataset being classified. With regards to the classifier used Random Forest has proven overall to be the better one amongst all our classifiers (Figure 254), despite not having consistent good results in all the experiments.

Conclusion

In this section we saw the results for classifying the AID362 dataset. This dataset was a highly imbalanced dataset which made the classifiers susceptible to bias in classifying the minority class. In order to assist with the classification we applied our methods to the dataset including balancing and dimensionality reduction.

We saw that when the dataset was experimented on in its original high-dimensional state, the fingerprint Pharmacophore stood out as the better performing one in almost all of the tests with good improvements when adding numerical descriptors. Random Forest and NaïveBayes show good results among the classifiers used followed by SMO and Majority voting.

When PCA was applied to the dataset, it was soon clear that the fingerprint Pharmacophore produced better results compared to the other ones and benefited most from the addition of the numerical descriptors. PubChem and MACCS followed the performance level after Pharmacophore. The classifier Random Forest continued to be the better performing classifier.

5.4. The Heavily Imbalanced Dataset – AID456

The second dataset we investigate in the section for heavily imbalanced datasets is the VCAM-1 Imaging Assay in Pooled HUVECs. For the ease of reading we shall call this dataset AID456 from here onwards.

AID456 is related to screening compounds for VCAM-1(vascular cell adhesion molecule-1) cells induced by pro-inflammatory agents (Han et al. 2010).

Dataset	#Total Instances	#Active Instances (class '1')	#Inactive Instances (class '0')	Active/Inactive Ratio
AID456	9982	27	9955	0.0027

Table 23: AID456 Dataset specification. Class of interest labelled as 1

In this next section, we classify the original dataset and show the classification metrics used.

AID456 Classification Results per Fingerprint– Original

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

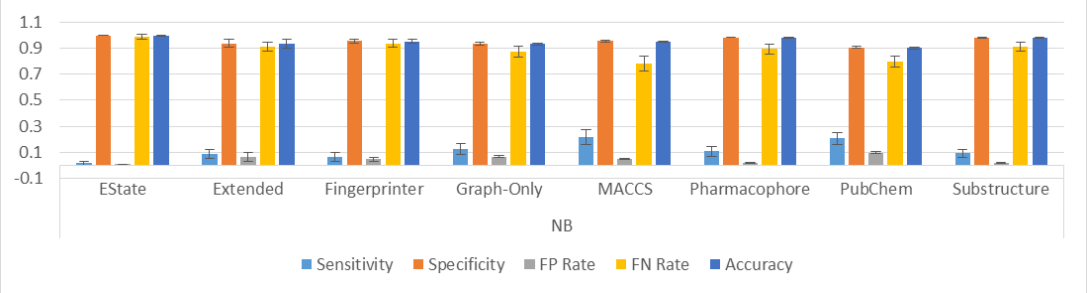


Figure 255: Classification results from classifying the AID456 dataset by NaïveBayes

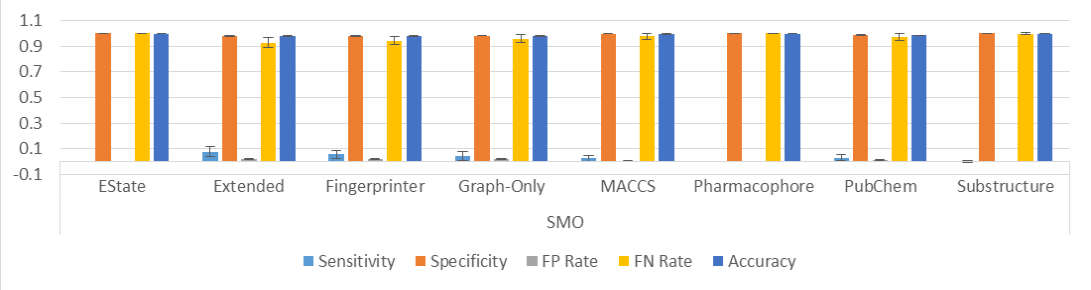


Figure 256: Classification results from classifying the AID456 dataset by SMO

By looking at Figure 255 and Figure 256 we see that the fingerprints MACCS, Pharmacophore and PubChem have produced better sensitivity than the other fingerprints used especially when accompanied by NaïveBayes. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓**	↑**	↓**	↓**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↓	↓**	↑**	↑	↓**
Pharmacophore	↑**	↓**	↑**	↓**	↓**
PubChem	↑	↓	↑	↓	↓
Substructure	↑**	↓**	↑**	↓**	↓**

Figure 257: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↔	↑*	↓*	↔	↑*
Fingerprinter	↔	↑	↓	↔	↑
Graph-Only	↔	↑	↓	↔	↑
MACCS	↓	↑**	↓**	↑	↑**
Pharmacophore	↔	↑**	↓**	↔	↑**
PubChem	↔	↑	↓	↔	↑
Substructure	↓	↑*	↓*	↑	↑*

Figure 258: Results from adding numerical fingerprints to binary fingerprints for Random Forest

Results here show that the fingerprints Pharmacophore and MACCS have the most improvement and when Random Forest is engaged, specificity and false positive rates improve most with the addition of numerical descriptors. In the next section, we classify the original dataset and show the classification metrics used.

AID456 Classification Results per Classifiers– Original

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

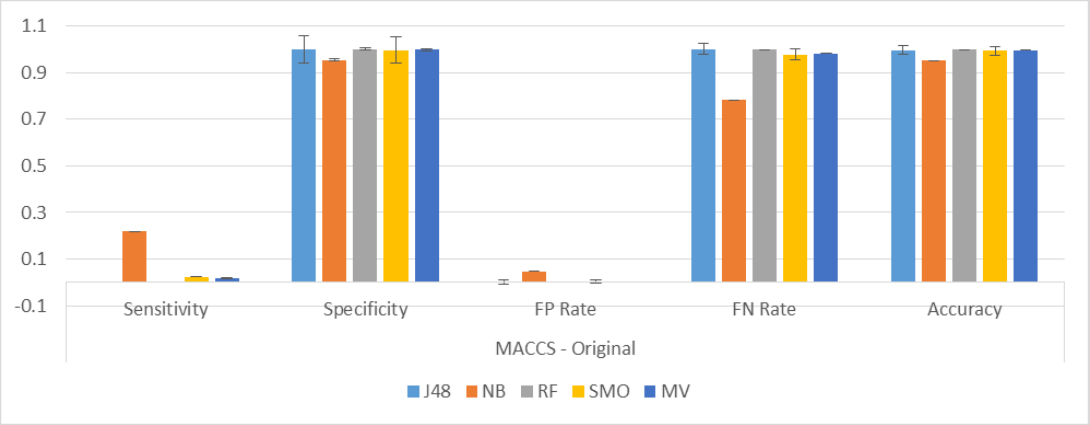


Figure 259: Classifier performance for MACCS

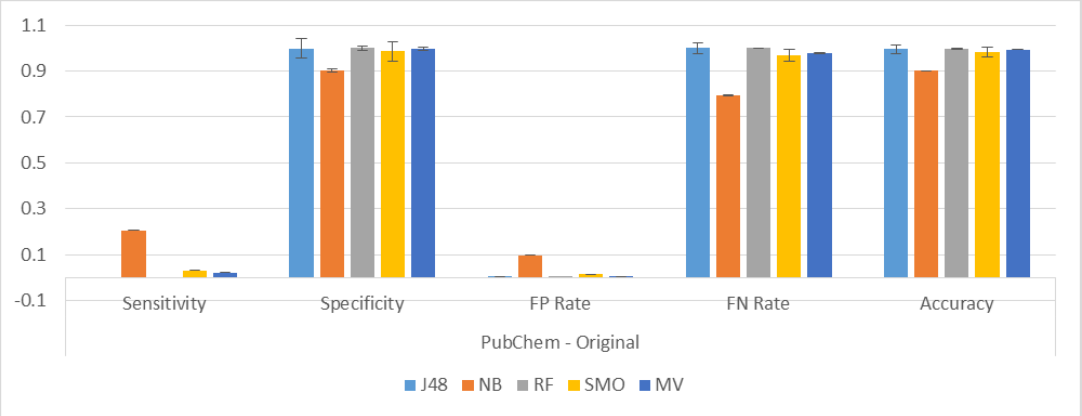


Figure 260: Classifier performance for PubChem

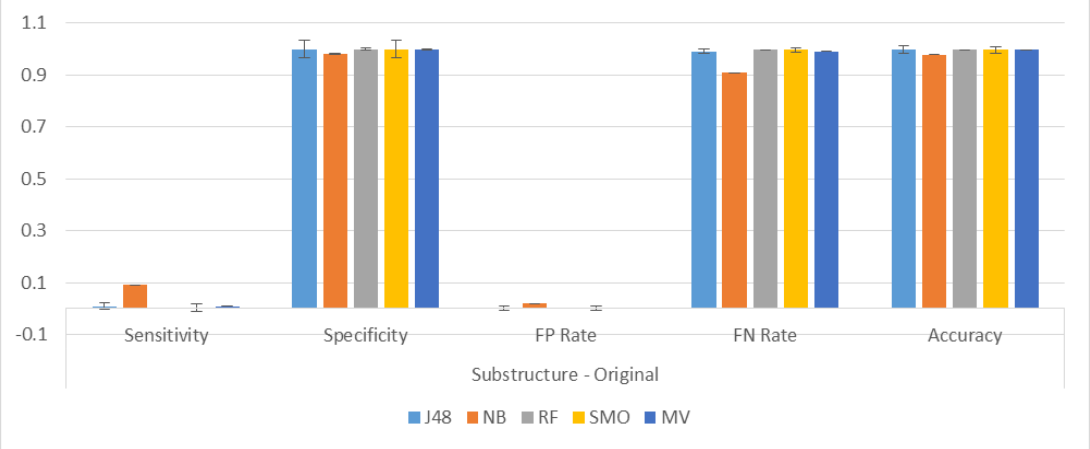


Figure 261: Classifier performance for Substructure

By looking at Figure 259 - Figure 261 we see how difficult it is to classify an extremely imbalanced high-dimensional dataset such as AID456. The sensitivity levels are at an extreme low and almost all of the data has been classified as the majority class (specificity very high). However NaïveBayes shows good sensitivity levels especially with PubChem and MACCS. SMO produces less good results for

sensitivity compared to NaïveBayes but shows better lower false positive rates using the same fingerprints. In the next section, we will observe how adding numerical fingerprints affects our classification results

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↓	↓**	↑**	↑	↓**
RF	↓	↑**	↓**	↑	↑**
SMO	↑	↓	↑	↓	↓
MV	↓	↓	↑	↑	↓

Figure 262: Results from adding numerical fingerprints to binary fingerprints for MACCS

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↑	↓	↔	↑
NB	↑	↓	↑	↓	↓
RF	↔	↑	↓	↔	↑
SMO	↑	↑	↓	↓	↑
MV	↔	↓	↑	↔	↓

Figure 263: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓**	↑**	↑	↓**
NB	↑**	↓**	↑**	↓**	↓**
RF	↓	↑*	↓*	↑	↑*
SMO	↑	↓	↑	↓	↓
MV	↑	↓	↑	↓	↓

Figure 264: Results from adding numerical fingerprints to binary fingerprints for Substructure

By looking at the figures above we can see that despite the red arrows which indicate worsening of the rates as a result of adding numerical descriptors, SMO and NaïveBayes show signs of improvement, however little. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics per fingerprint used.

AID456 Classification Results per Fingerprint– Original SMOTEd All

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

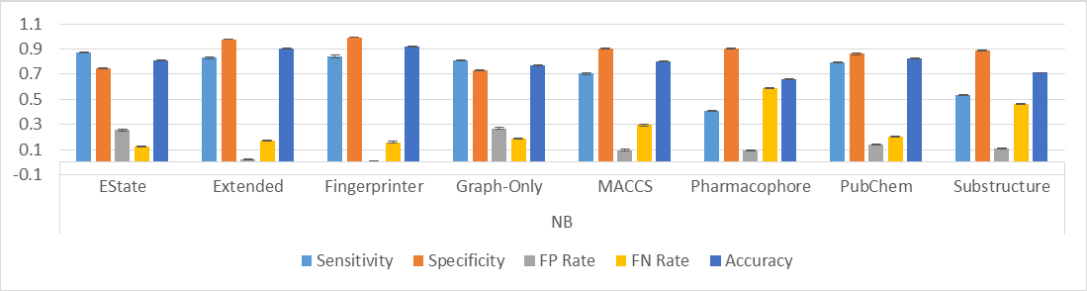


Figure 265: Classification results from classifying the AID456 dataset by NaïveBayes

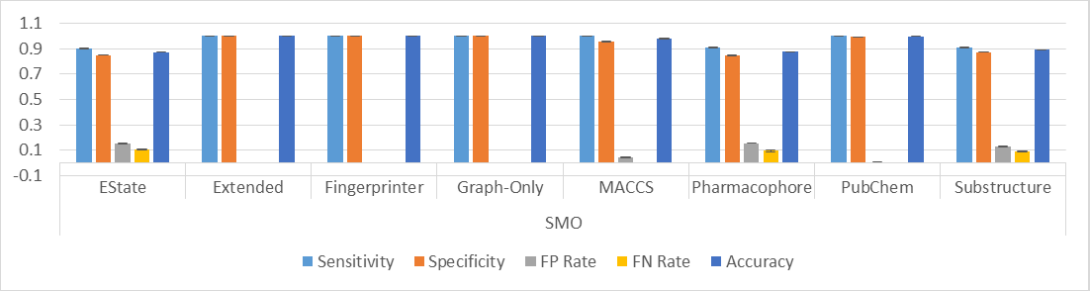


Figure 266: Classification results from classifying the AID456 dataset by SMO

In these set of tests the AID456 dataset has been balanced using SMOTE and then split into test and training sets. Since the dataset is balanced we see great results with regards to both sensitivity and specificity and also for false positive and negative rates. SMO has the better results compared to NaïveBayes and MACCS and PubChem appear to be the better performing fingerprints. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↓	↑	↓	↑	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↓	↓	↑	↑	↓
Substructure	↑	↑	↓	↓	↑

Figure 267: Results from adding numerical fingerprints to binary fingerprints for J48

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↓**
Extended	↓**	↑**	↓**	↑**	↓
Fingerprinter	↓**	↑**	↓**	↑**	↓**
Graph-Only	↑**	↑**	↓**	↓**	↑**
MACCS	↑**	↑	↓	↓**	↑**
Pharmacophore	↑**	↑	↓	↓**	↑**
PubChem	↑**	↑	↓	↓**	↑*
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 268: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↑	↑	↓	↓	↑*
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↑*	↓*	↓	↑*
Pharmacophore	↑	↑**	↓**	↓	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↑**	↓**	↓	↑**

Figure 269: Results from adding numerical fingerprints to binary fingerprints for Random Forest

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↔	↑	↓	↔	↑
Fingerprinter	↔	↓	↑	↔	↓
Graph-Only	↔	↑	↓	↔	↑
MACCS	↓	↑**	↓**	↑	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↔	↑*	↓*	↔	↑*
Substructure	↑**	↑**	↓**	↓**	↑**

Figure 270: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↑	↓	↑	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↓	↑	↓	↓
MACCS	↑	↑**	↓	↓	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↑	↓	↑	↓	↑
Substructure	↑	↑**	↓**	↓	↑**

Figure 271: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

By looking at Figure 267 - Figure 271 we clearly see that Pharmacophore is the better performing fingerprint and has improved significantly. SMO and Random Forest have performed great followed by Majority Voting and NaïveBayes. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

AID456 Classification Results per Classifiers– Original SMOTEd All

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

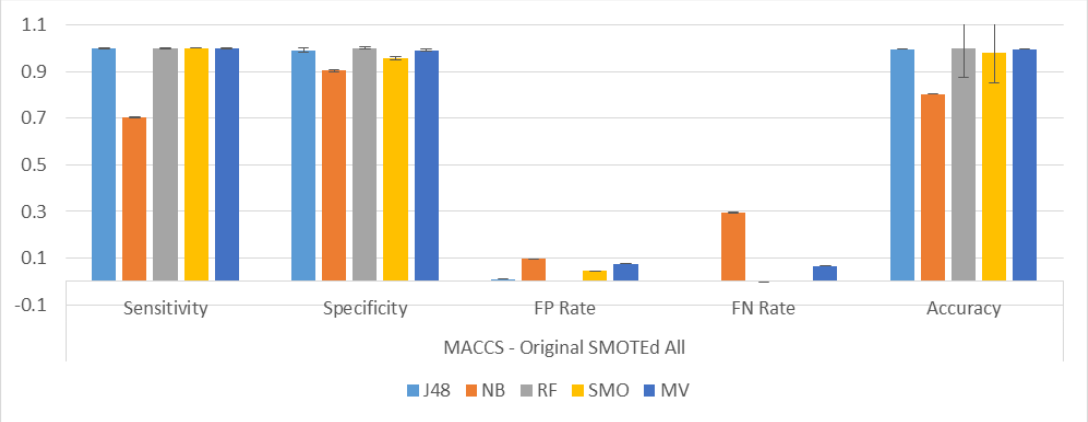


Figure 272: Classifier performance for MACCS

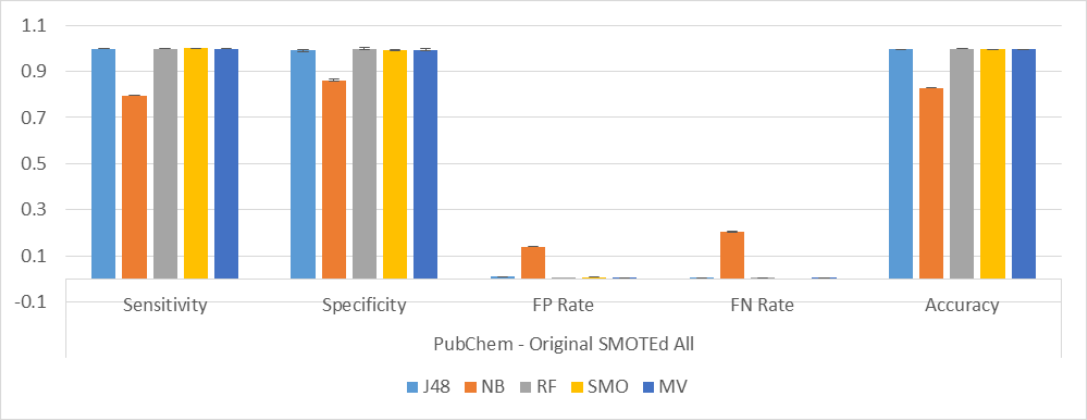


Figure 273: Classifier performance for PubChem

Here we see that SMO has performed greatly and has results as good as J48 and Random Forest and Majority Voting. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑**	↓**	↑	↑**
NB	↓**	↑**	↓**	↑**	↓**
RF	↑	↑**	↓**	↓	↑**
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 274: Results from adding numerical fingerprints to binary fingerprints for EState

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↑**	↑	↓	↓**	↑**
RF	↑	↑**	↓**	↓	↑**
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 275: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑**	↑**	↓**	↓**	↑**
RF	↑	↑**	↓**	↓	↑**
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑	↑**	↓**	↓	↑**

Figure 276: Results from adding numerical fingerprints to binary fingerprints for Substructure

From observing Figure 274 - Figure 276 we see many great improvements among the classification metrics and the classifiers used. The most significant improvements can be seen on SMO, Majority Voting and Random Forest. In the next section, we classify the dataset where only training set has been balanced and show the classification metrics used.

AID456 Classification Results per Fingerprint– Original SMOTEd Training

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

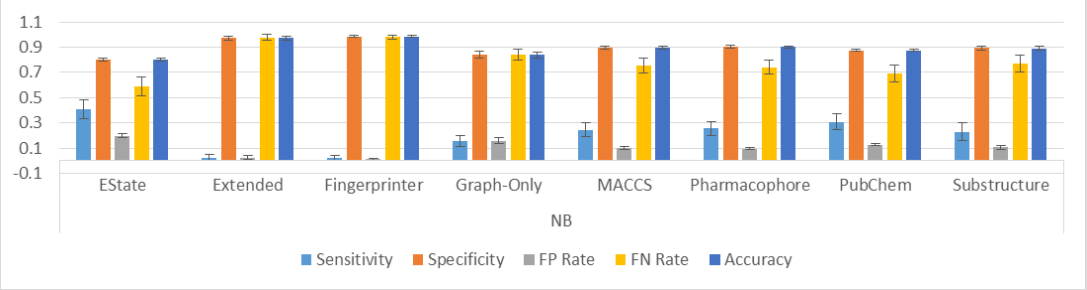


Figure 277: Classification results from classifying the AID456 dataset by NaïveBayes

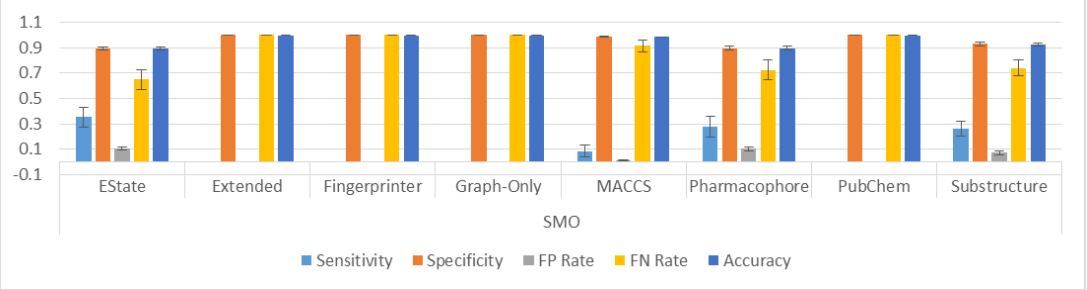


Figure 278: Classification results from classifying the AID456 dataset by SMO

Our dataset has been split into training and test and then only the training set has been balanced. Results from Figure 277 and Figure 278 Show that NaïveBayes has slightly better results compared to SMO with regards to the sensitivity levels.

Pharmacophore, MACCS and Substructure are the better performing fingerprints. In the next section, we will observe how adding numerical fingerprints affects our classification results and whether the changes are statistically significant or not.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑*
Extended	↓	↑**	↓**	↑	↑**
Fingerprinter	↑	↑**	↓**	↓	↑**
Graph-Only	↑	↑	↓	↓	↑
MACCS	↔	↑	↓	↔	↑
Pharmacophore	↓*	↑**	↓**	↑*	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↓	↑	↓	↑	↑

Figure 279: Results from adding numerical fingerprints to binary fingerprints for J48

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑*	↓*	↓	↑*
Extended	↔	↓	↑	↔	↓
Fingerprinter	↔	↓	↑	↔	↓
Graph-Only	↔	↓	↑	↔	↓
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓	↑**	↓**	↑	↑**
PubChem	↔	↑	↓	↔	↑
Substructure	↓*	↑*	↓*	↑*	↑*

Figure 280: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↔	↑**	↓**	↔	↑**
Graph-Only	↓	↓	↑	↑	↓
MACCS	↑	↑*	↓*	↓	↑*
Pharmacophore	↓	↑**	↓**	↑	↑**
PubChem	↔	↑	↓	↔	↑
Substructure	↓	↑**	↓**	↑	↑**

Figure 281: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

Specificity, false positive and accuracy levels have improved as a result off adding numerical descriptors in Figure 279 - Figure 281 (except for the CDK Fingerprints in Figure 280). J48 has performed well and from the fingerprints EState and Pharmacophore have the most improvements. In the next section, we classify the

original dataset where only training set has been balanced and show the classification metrics used.

AID456 Classification Results per Classifiers– Original SMOTEd Training

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

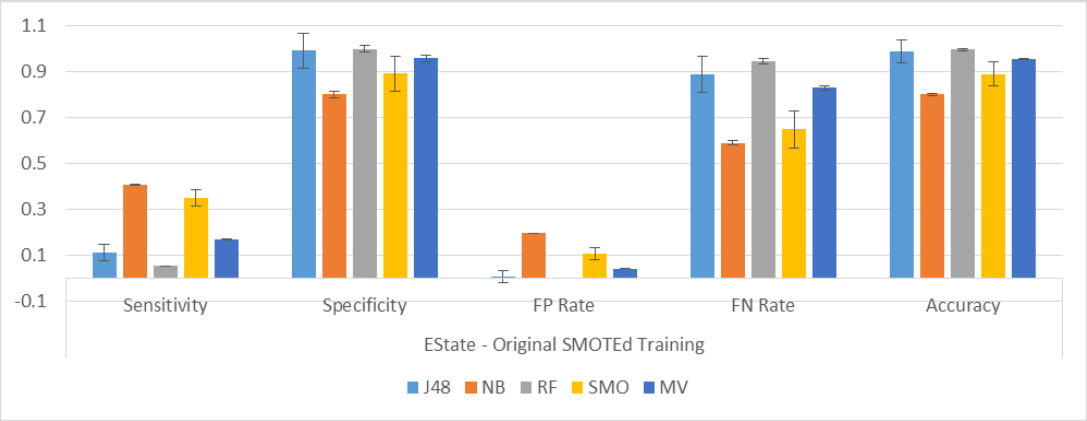


Figure 282: Classifier performance for EState

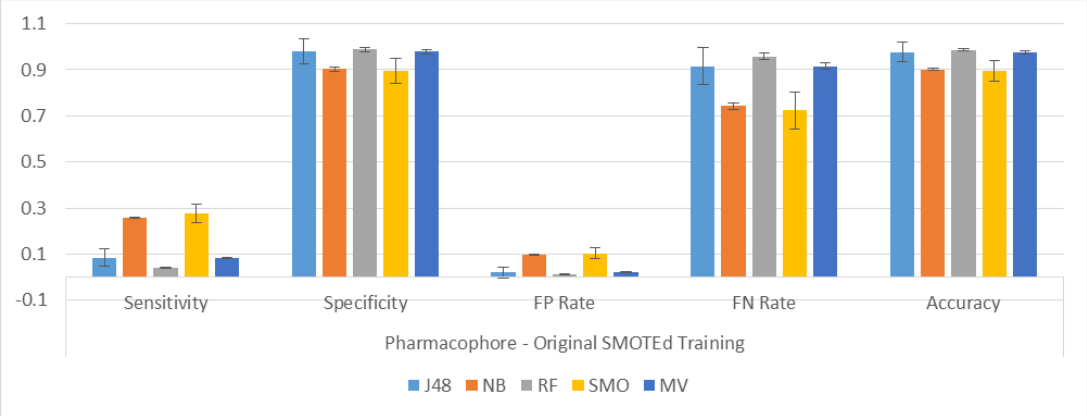


Figure 283: Classifier performance for Pharmacophore

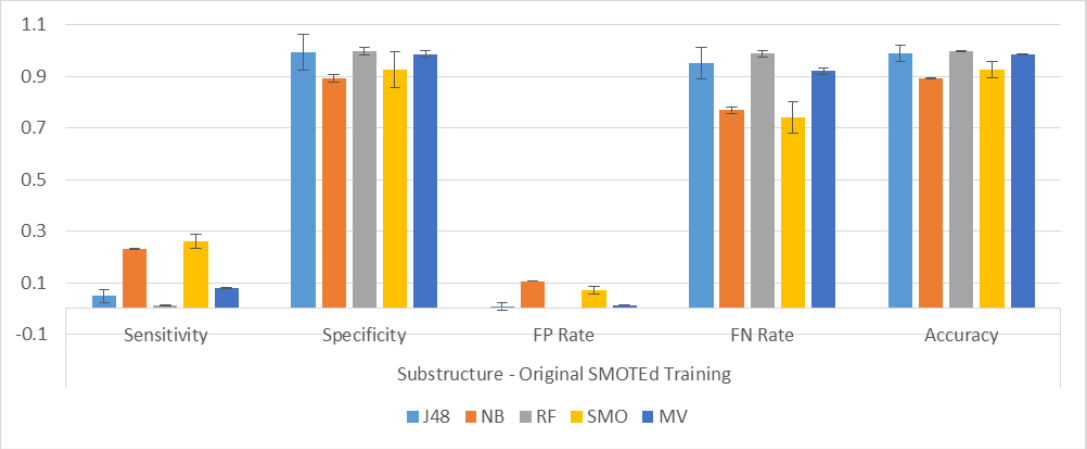


Figure 284: Classifier performance for Substructure

In these set of tests, SMO and NaïveBayes show better sensitivity results but also higher false positive results compared to the other classifiers. EState and Pharmacophore have better results for SMO and NaïveBayes too. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓**	↑**	↓**	↑**	↑*
NB	↓	↓	↑	↑	↓
RF	↓**	↑**	↓**	↑**	↑**
SMO	↑	↑*	↓*	↓	↑*
MV	↓	↑**	↓**	↑	↑**

Figure 285: Results from adding numerical fingerprints to binary fingerprints for EState

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓*	↑**	↓**	↑*	↑**
NB	↑	↓	↑	↓	↓
RF	↓**	↑**	↓**	↑**	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↓	↑**	↓**	↑	↑**

Figure 286: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

In Figure 285 and Figure 286 there is a great level of improvement for specificity and false positive rates together with accuracy. This improvement is mostly seen in J48, Random Forest, SMO and Majority Voting. In the next section, we classify the original dataset with PCA and show the classification metrics used.

AID456 Classification Results per Fingerprint– PCA Original

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results. Here we have applied PCA to the dataset.

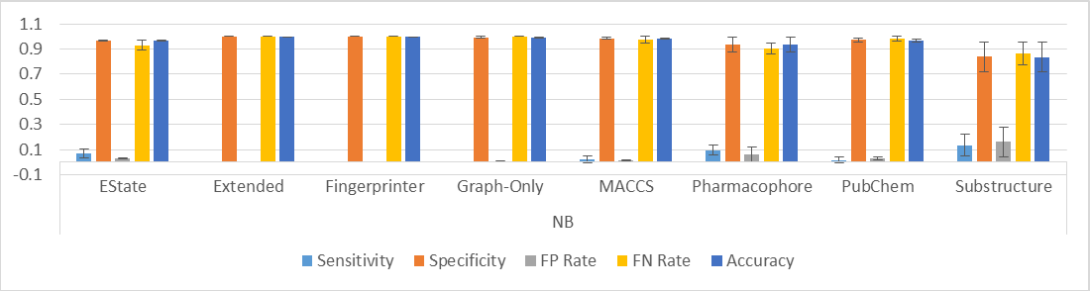


Figure 287: Classification results from classifying the AID456 dataset by NaïveBayes

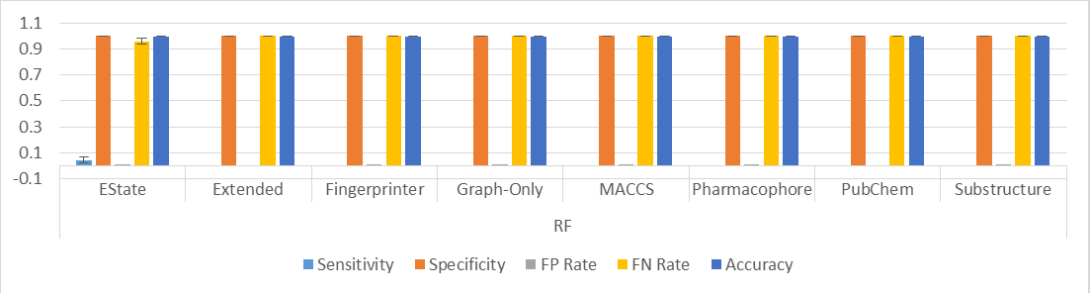


Figure 288: Classification results from classifying the AID456 dataset by Random Forest

Reducing the dimensionality of AID456 using PCA and classifying it in the imbalanced state, has not produced many good results. As expected the bias of the classifiers is towards the majority class and almost all samples (majority or minority) have been classified as the majority class. EState, Pharmacophore and Substructure have produced some sensitivity with NaïveBayes. No fingerprint has performed consistently or particularly well. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↔	↔	↔	↔	↔
Fingerprinter	↔	↔	↔	↔	↔
Graph-Only	↔	↑	↓	↔	↑
MACCS	↔	↑	↓	↔	↑
Pharmacophore	↑	↑**	↓**	↓	↑**
PubChem	↔	↓	↑	↔	↓
Substructure	↑	↑*	↓*	↓	↑*

Results from adding numerical fingerprints to binary fingerprints for Random Forest

Adding numerical descriptors has not improved much in our classification metrics. EState and Pharmacophore are the only two fingerprints to improve significantly. In the next section, we classify the original dataset with PCA and show the classification metrics used.

AID456 Classification Results per Classifiers– PCA Original

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results. Here the PCA technique was applied to the dataset.

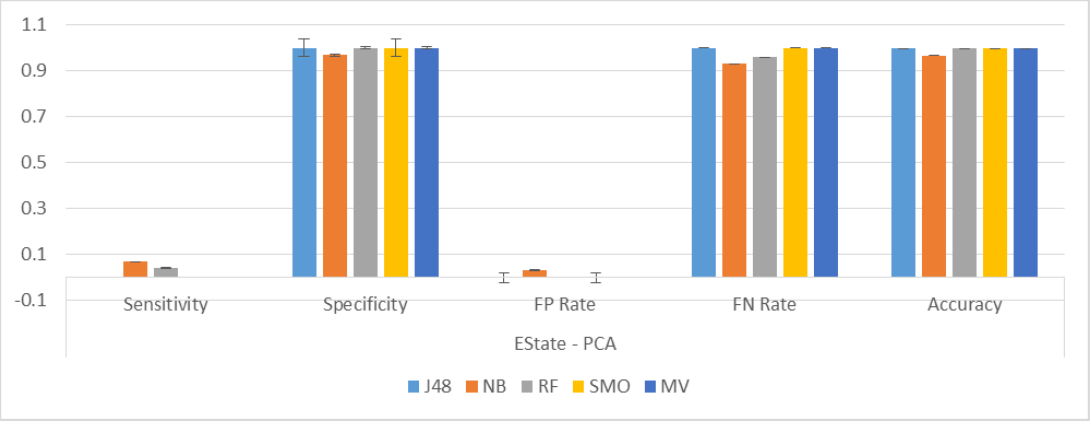


Figure 289: Classifier performance for EState

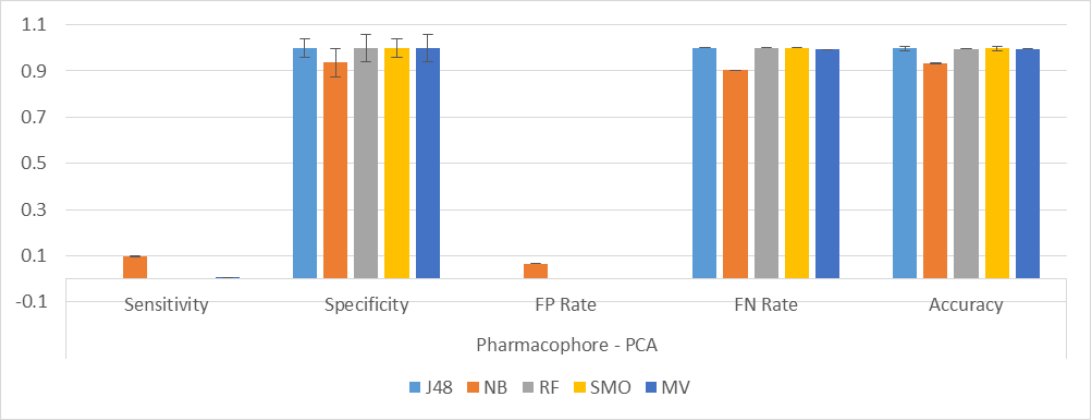


Figure 290: Classifier performance for Pharmacophore

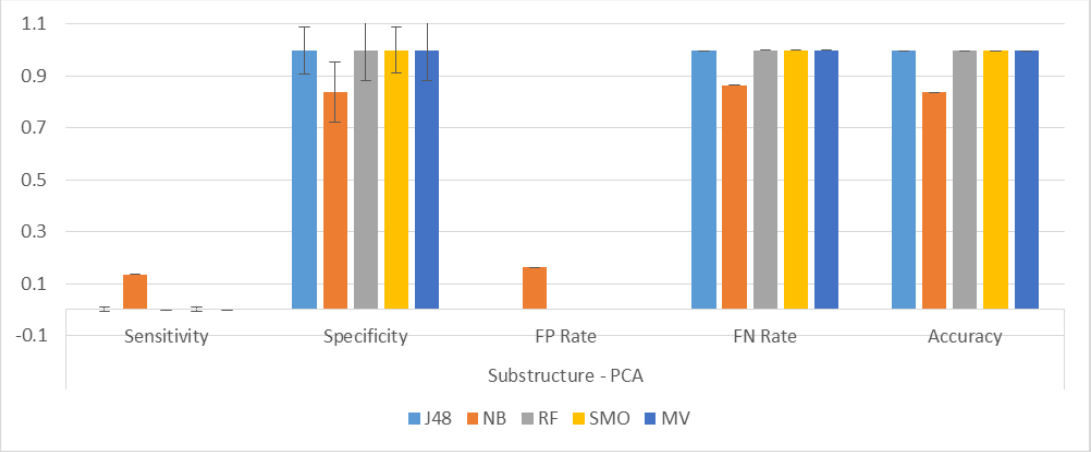


Figure 291: Classifier performance for Substructure

In Figure 289 - Figure 291 NaïveBayes seems to have classified some minority class instances correctly. It seems that the level of sensitivity and false positive for this classifier go hand in hand; as the sensitivity rises, so does the false positive rate. No classifier has performed particularly well. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↓	↑	↔	↓
NB	↑	↑*	↓*	↓	↑*
RF	↓**	↑**	↓**	↑**	↑**
SMO	↔	↔	↔	↔	↔
MV	↑	↑	↓	↓	↑

Figure 292: Results from adding numerical fingerprints to binary fingerprints for EState

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↓	↑	↔	↓
NB	↓	↑	↓	↑	↑
RF	↑	↑**	↓**	↓	↑**
SMO	↔	↑	↓	↔	↑
MV	↑	↔	↔	↓	↑

Figure 293: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓	↑	↓	↑	↑
RF	↑	↑*	↓*	↓	↑*
SMO	↑	↓*	↑*	↓	↓*
MV	↑	↓	↑	↓	↓

Figure 294: Results from adding numerical fingerprints to binary fingerprints for Substructure

From looking at the figures above we see that Random Forest has benefited mostly from the addition of the numerical descriptors and no other classifier has major improvements. In the next section, we classify the dataset that was balanced before splitting with PCA and show the classification metrics used.

AID456 Classification Results per Fingerprint– PCA SMOTEd All

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

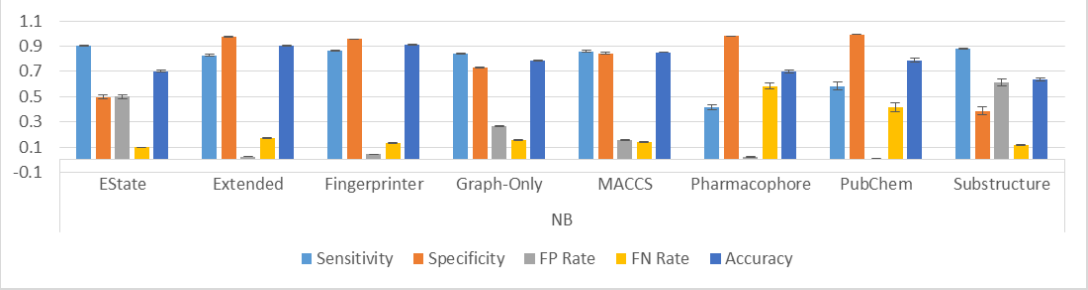


Figure 295: Classification results from classifying the AID456 dataset by NaïveBayes

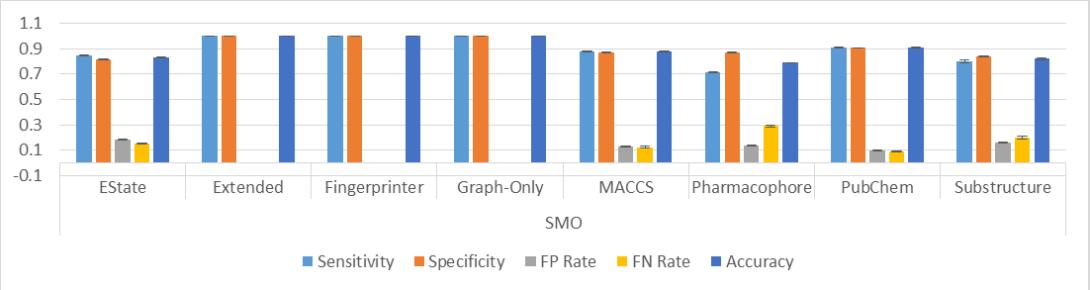


Figure 296: Classification results from classifying the AID456 dataset by SMO

As a result of balancing the dimensionality-reduced dataset, we observe improvement in the sensitivity rates. As mentioned before this might be as a result of overfitting by using SMOTE, but research shows that in general resampling techniques using oversampling perform on average better (Orriols-Puig & Bernadó-Mansilla 2009). EState, MACCS and PubChem have produced better results in the presence of SMO (Figure 296). In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↓*	↑*	↑	↓
Fingerprinter	↑	↑**	↓**	↓	↑
Graph-Only	↓	↑	↓	↑	↑
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↑	↑**	↓**	↓	↑
PubChem	↑*	↓**	↑**	↓*	↑
Substructure	↓**	↑**	↓**	↑**	↑**

Figure 297: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↔	↑	↓	↔	↑
Fingerprinter	↔	↓	↑	↔	↓
Graph-Only	↔	↑**	↓**	↔	↑**
MACCS	↓	↑	↓	↑	↓
Pharmacophore	↑**	↓**	↑**	↓**	↑**
PubChem	↑**	↑**	↓**	↓**	↑**
Substructure	↓	↑**	↓**	↑	↑

Figure 298: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↑**	↓**	↓**	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↓	↑	↑	↓
MACCS	↓	↑**	↓**	↑	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↑*	↓**	↑**	↓*	↓
Substructure	↓**	↑**	↓**	↑**	↑**

Figure 299: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

EState, MACCS and Pharmacophore are the three fingerprints that have benefited most from the addition of the numerical descriptors. SMO and Majority Voting have the most significant improvements. In the next section, we classify the dataset that was balanced before splitting and show the classification metrics used.

AID456 Classification Results per Classifiers– PCA SMOTEd All

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results. PCA was applied here to the dataset.

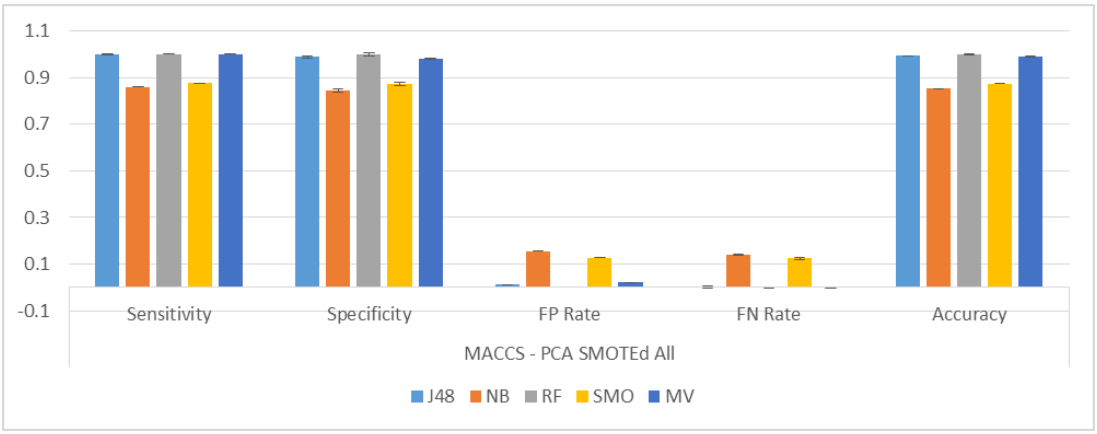


Figure 300: Classifier performance for MACCS

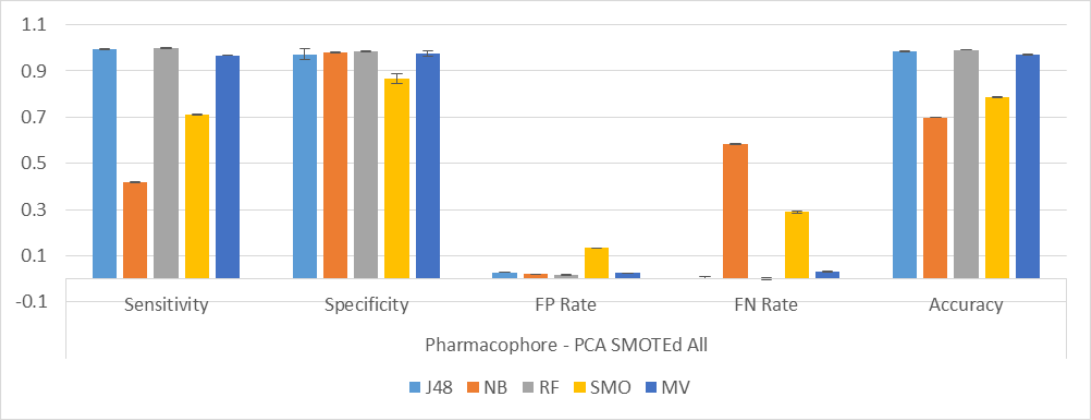


Figure 301: Classifier performance for Pharmacophore

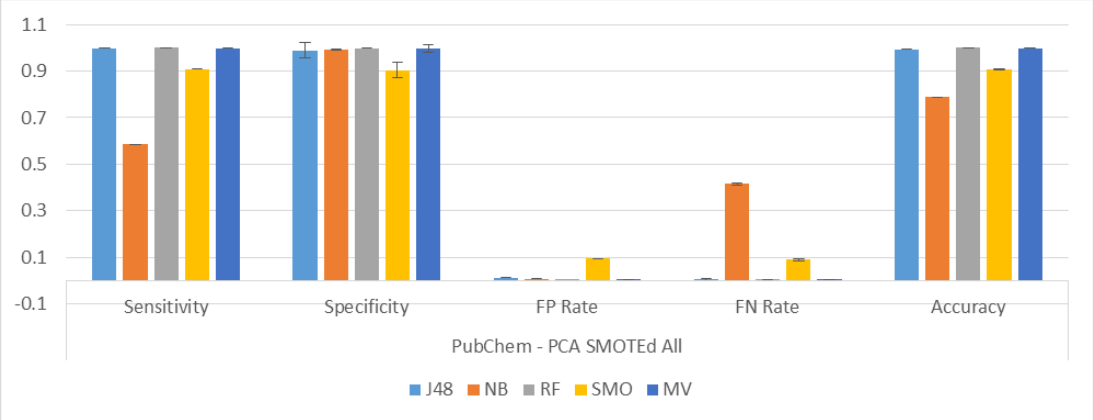


Figure 302: Classifier performance for PubChem

J48, Random Forest and Majority Voting have performed well and produced good metrics in these tests. NaïveBayes has performed the worst by producing less sensitivity and more false positive rates. In the next section, we will observe how adding numerical fingerprints affects our classification results

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↑**	↑**	↓**	↓**	↑**
RF	↓	↑**	↓**	↑	↑**
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 303: Results from adding numerical fingerprints to binary fingerprints for EState

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑**	↓**	↓	↑**
NB	↑	↑**	↓**	↓	↑
RF	↑	↑**	↓**	↓	↑**
SMO	↑**	↓**	↑**	↓**	↑**
MV	↑**	↑**	↓**	↓**	↑**

Figure 304: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑*	↓**	↑**	↓*	↑
RF	↑	↓	↑	↓	↓
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑*	↓**	↑**	↓*	↓

Figure 305: Results from adding numerical fingerprints to binary fingerprints for PubChem

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓	↑	↑	↓
NB	↓**	↑**	↓**	↑**	↑**
RF	↓	↑**	↓**	↑	↑
SMO	↓	↑**	↓**	↑	↑
MV	↓**	↑**	↓**	↑**	↑**

Figure 306: Results from adding numerical fingerprints to binary fingerprints for Substructure

Apart from when Substructure is used as the fingerprinting technique, all other fingerprints show good and significant improvements in the presence of the classifiers used. Not all metrics have improved consistently but there is overall a good rate of improvement. In the next section, we classify the dataset where only training set has been balanced with PCA and show the classification metrics used.

AID456 Classification Results per Fingerprint– PCA SMOTEd Training

In this section we look in more detail at the classification results per classifier used and then per each fingerprint. We want to see with every classifier, which fingerprint performed better regarding the classification metrics. In the next few pages we shall be showing these results.

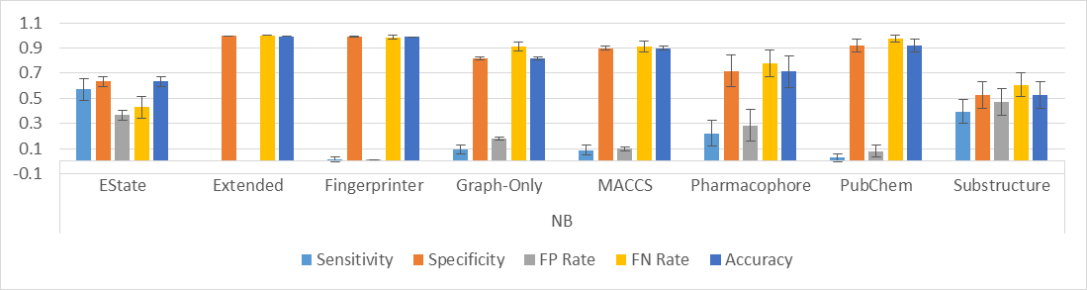


Figure 307: Classification results from classifying the AID456 dataset by NaïveBayes

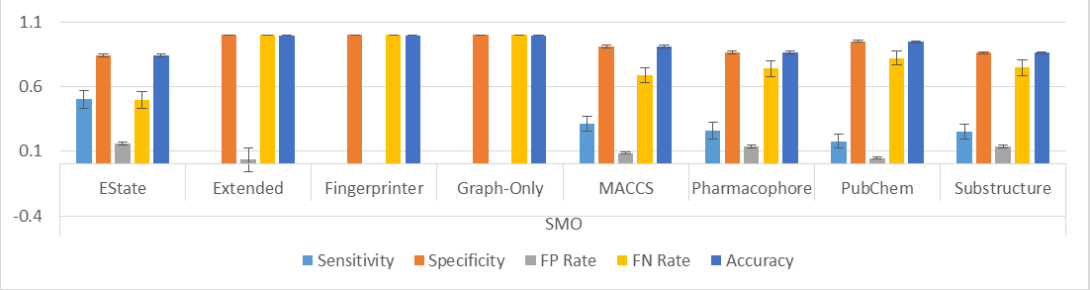


Figure 308: Classification results from classifying the AID456 dataset by SMO

The sensitivity rates have fallen as a result of only balancing the training set, since there is a very low number of minority examples in the test set to be classified and the slightest misclassification can have a great cost. EState and Substructure have produced more balanced results with NaïveBayes. MACCS, Pharmacophore and Substructure have good results with SMO. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↑	↓	↑	↓	↓
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↑	↓	↑	↓	↓
Substructure	↓**	↑**	↓**	↑**	↑**

Figure 309: Results from adding numerical fingerprints to binary fingerprints for NaïveBayes

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↔	↑	↓	↔	↑
Fingerprinter	↔	↑	↓	↔	↑
Graph-Only	↔	↑	↓	↔	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑	↑	↓	↓	↑
PubChem	↓**	↑**	↓**	↑**	↑**
Substructure	↑	↑	↓	↓	↑

Figure 310: Results from adding numerical fingerprints to binary fingerprints for SMO

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↔	↓	↑	↔	↓
Fingerprinter	↔	↑	↓	↔	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓	↑**	↓**	↑	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↓	↑**	↓**	↑	↑**

Figure 311: Results from adding numerical fingerprints to binary fingerprints for Majority Voting

Results produced per fingerprints have improved by adding numerical descriptor in these tests however the results are not too significant. Substructure and Pharmacophore are the two fingerprints showing significant improvements. In the next section, we classify the dataset where only training set has been balanced With PCA and show the classification metrics used.

AID456 Classification Results per Classifiers– PCA SMOTEd Training

In this section we look in more detail at the classification results per fingerprint used and then per each classifier. We want to see with every fingerprint, which classifier performed better regarding the classification metrics. In the next few pages we shall be showing these results.

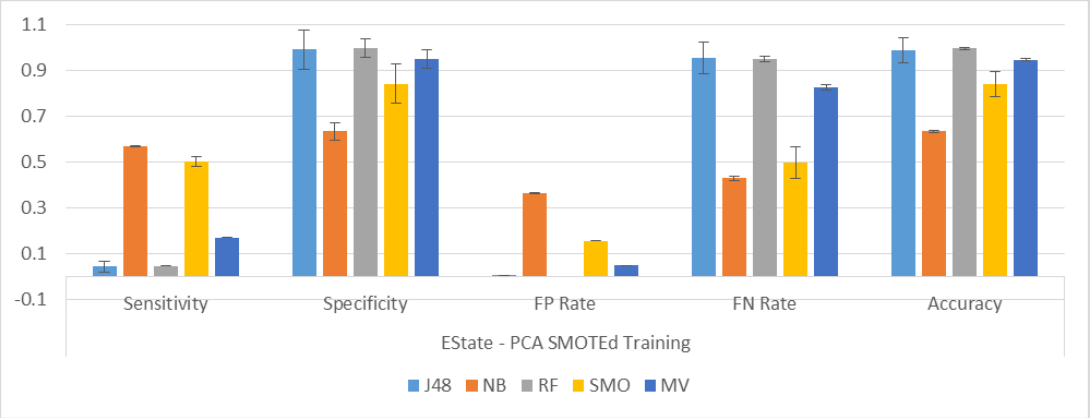


Figure 312: Classifier performance for EState

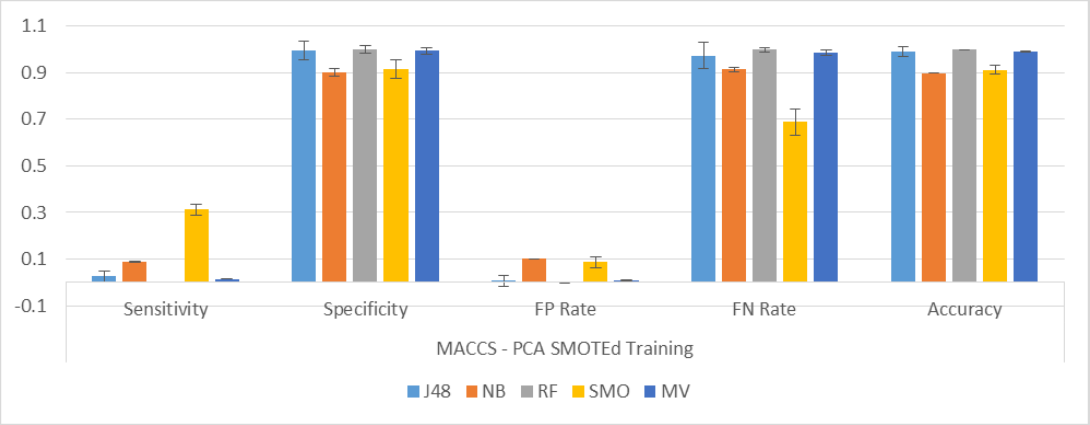


Figure 313: Classifier performance for MACCS

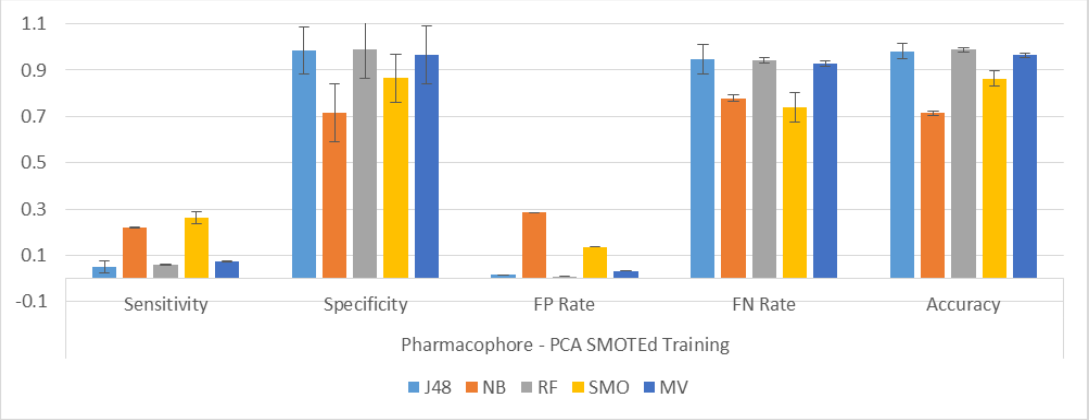


Figure 314: Classifier performance for Pharmacophore

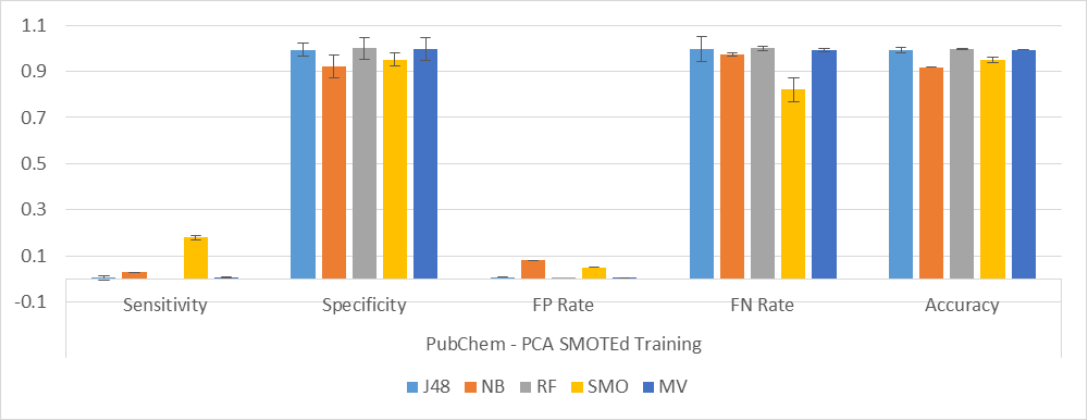


Figure 315: Classifier performance for PubChem

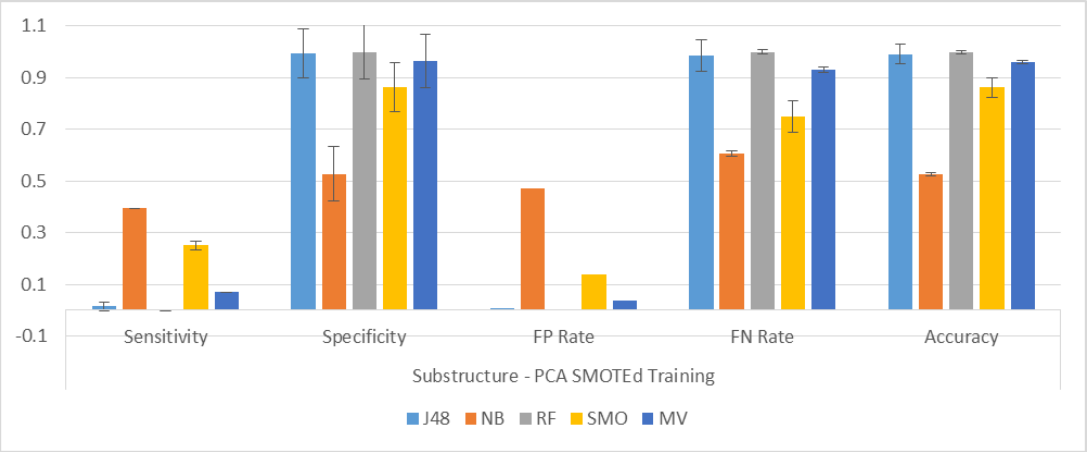


Figure 316: Classifier performance for Substructure

SMO and NaïveBayes have produced the highest level of sensitivity in Figure 312 - Figure 316. And they also have the highest false positive rates amongst the classifiers in these set of tests. J48, Random Forest and Majority Voting have better results with Pharmacophore. In the next section, we will observe how adding numerical fingerprints affects our classification results.

Analysis of the Improvement with Numerical Fingerprints

In this section we have included the numerical fingerprints to the binary ones to see the effect this might have in the classification process and our metrics. These results and whether the change is significant is shown in this section.

EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓	↑**	↓**	↑	↑**
RF	↓**	↑**	↓**	↑**	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↓	↑**	↓**	↑	↑**

Figure 317: Results from adding numerical fingerprints to binary fingerprints for EState

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑**	↓**	↑	↑**
NB	↑	↓	↑	↓	↓
RF	↓*	↑**	↓**	↑*	↑**
SMO	↑	↑	↓	↓	↑
MV	↓	↑**	↓**	↑	↑**

Figure 318: Results from adding numerical fingerprints to binary fingerprints for Pharmacophore

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↓**	↑**	↓**	↑**	↑**
RF	↑	↑	↓	↓	↑
SMO	↑	↑	↓	↓	↑
MV	↓	↑**	↓**	↑	↑**

Figure 319: Results from adding numerical fingerprints to binary fingerprints for Substructure

Results from this section show that there is great improvement in the specificity and false negative and accuracy rates. There are however no classifiers that have consistently improved and significantly too.

Summary of the results and receiver operating characteristics analysis

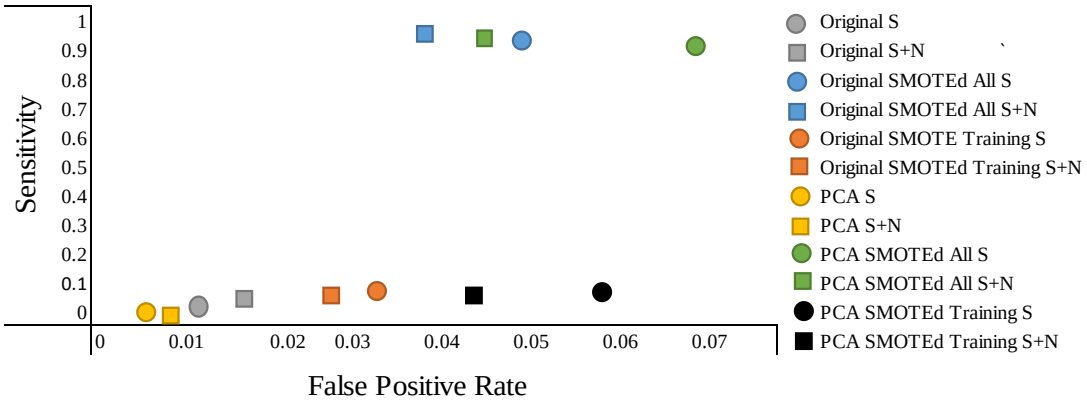


Figure 320: Sensitivity versus False Positive *AID456* methods

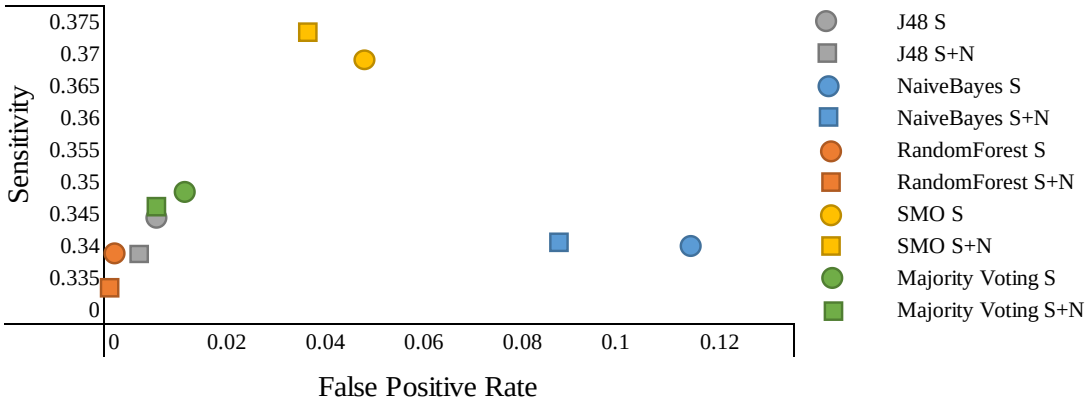


Figure 321: Sensitivity versus False Positive *AID456* classifiers

Methods Used		**Euclidean Distance**
Binary Descriptors	Original	0.968
	Original SMOTEd All	**0.0778**
	Original SMOTEd Training	0.9138
	PCA	0.9866
	PCA SMOTEd All	0.0953
	PCA SMOTEd Training	0.9132
Binary + Numerical Descriptors	Original	0.9563
	Original SMOTEd All	0.0842
	Original SMOTEd Training	0.9247
	PCA	0.9862
	PCA SMOTEd All	**0.0664**
	PCA SMOTEd Training	0.9253

Table 24: Euclidean distance for the methods used

Classifiers Used		**Euclidean Distance**
Binary Descriptors	J48	0.6555
	NaïveBayes	0.6707
	Random Forest	0.6605
	SMO	**0.6331**
	Majority Voting	0.6516
Binary + Numerical Descriptors	J48	0.6604
	NaïveBayes	0.666
	Random Forest	0.6655
	SMO	**0.6289**
	Majority Voting	0.6537

Table 25: Euclidean distance for the classifiers used

Conclusion

In this section we experimented with our most imbalanced dataset, AID456. The results were discussed from the point of view of the fingerprints and the classifiers used. We looked at how the fingerprints performed in the presence of each classifier and vice versa. We also looked at how adding numerical descriptors to binary fingerprints affect the results of classification metrics. These tests were done once with the dataset in its original state and once when PCA was applied. Throughout these tests we applied our unique methodology of combining the use of fingerprints and balancing using SMOTE.

In the presence of each classifier the fingerprints behave differently. There was no consistent behaving fingerprint, but overall we could point out that Pharmacophore and MACCS did stand out as better performing ones. When looked at the performance of classifiers in the presence of the fingerprints we observe that SMO was indeed the better performing classifier. This result can also be seen in Figure 321. This can also be confirmed by looking at Table 25.

The application of PCA did worsen our results and there was almost no sensitivity produced by the fingerprints. If the classifiers did show sensitivity rate it was NaïveBayes and on occasion SMO, but this would go hand in hand with higher false positive rates. Adding numerical attributes did affect classification metrics in positive ways in many situations. Although compared to AID362 there were fewer of these instances. The statistical significance of the improvements was not concluded on average and this should be discussed on a specific fingerprint or classifier level and cannot be generalised.

On the methods used to classify this dataset Figure 320 shows that when the dataset is balanced initially and then split into training and test sets it performs best. This is true for the dataset in its original state and when the dimensionality has been reduced and numerical attributes have been added (as seen in Table 24).

The results of classifying this dataset indicate that the high level of imbalance compounded by high numbers of instances and attributes makes it extra difficult to obtain good levels of classification metrics, especially sensitivity and false positives. When the whole dataset was balanced using SMOTE, the results achieved were optimal, but one might wonder whether this is a result of the good effect of resampling using SMOTE, or is it because of the overfitting it causes.

6. General Discussion and Concluding Remarks

Chemoinformatics is the use of computational techniques in the field of chemistry in order to assist with the process of drug discovery. Most Chemoinformatics-related problems are associated with datasets that are highly imbalanced and these rare classes that are of interest in data mining. In this thesis, we propose that a unified processing approach, applicable both to standard and to particularly challenging chemical datasets (High-Dimensional and/or strongly Imbalanced), enables us to perform an effective Virtual Screening.

Virtual screening in drug discovery involves analysing datasets containing unknown molecules in order to find the ones that are likely to have the desired effects on a biological target. The molecules are thereby classified into active or non-active compared to the target. Standard classifiers assume equality between classes and therefore will not be very effective (Ganganwar 2012; López et al. 2013; Zięba et al. 2015). When classifying imbalanced datasets, it is more important to correctly classify minority classes also known as classes of interest. These rare classes often get misclassified because most classifiers optimise the overall classification accuracy. Thus, a number of classification approaches are focused on addressing this issue by modifying the algorithm (Estabrooks & Japkowicz 2004; Orriols-Puig & Bernadó-Mansilla 2009; García-Pedrajas et al. 2012; Lin & Chen 2012; Wang et al. 2012; Batuwita & Palade 2013; Ducange et al. 2013; Zong et al. 2013; Dittman & Khoshgoftar 2014; Maldonado et al. 2014). These approaches, however, are typically specifically designed to suit the dataset for which they were developed, whilst having limited success in different scenarios.

It is worthy to remind the reader that this is the main novelty of the work presented in the current study. It shows that the combination of over-sampling using SMOTE in specific and the utilisation of four main classifiers furnishes a generic, unified analysis for a wide range of cheminformatics data unlike other methods of dealing with imbalanced data in which the classifier is altered to meet the classification requirements for a specific type of data, therefore not providing a general, unified methodology for applying to a wide range of chemical datasets with varying imbalance ratios.

The lack of an effective approach to visual screening can have a significant negative impact in industrial settings. Misclassification of molecules in cases such as drug discovery and medical diagnosis is costly, both in time and finances. In the process of discovering a drug, it is mainly the inactive molecules classified as active towards the biological target i.e. false positives that cause a delay in the progress and high late-stage attrition.

In order to overcome this drawback, the methodology followed in this project consists of analysing the effects of using various fingerprinting methods combined with the Synthetic Minority Oversampling Technique on the classification of highly imbalanced, high-dimensional datasets in a collection of the most successful classifiers in Chemoinformatics, convening a wide range of classification criteria. This research was set up to examine different methods of manipulating big imbalanced datasets that have not been cleared of noise, and to see how they can affect the entire range of classification evaluation metrics beyond the mere performance. Crucially, the combination of the two techniques, should be successful for big and highly imbalanced datasets that have not been cleared of noise in order to account for a realistic screening scenario manipulation.

Chemoinformatics' settings are a predominantly challenging problem for classifiers and the screening process can be complex to comprehend intuitively. Thus, in order to better understand this thesis, we have introduced the general concepts of the drug discovery process and how Chemoinformatics has influenced. This introductory information has been expanded in Chapter 2, accompanied by a literature review and discussion of the important contributions in the areas. Chapter 3 provided the reader with information about the datasets; their origin, size and class distribution. Some detail about how the datasets were collected and transformed in the format to be used for this research has also been provided. Chapter 4 discussed the methods that were used in this research for gathering the results. These results were presented to the reader in Chapter 5.

In total we experimented with 128 unique datasets (refer to Figure 17 in chapter 3 for a summary of the generation). Our main findings are summarised in the next figures. These figures illustrate, in short, the performance of the methods and the classifiers used for this study. Specifically, the sensitivity versus false positive figures have been re-produced here for the sake of reminding our readers of the state

of our classification results and in order to compare the datasets. We start with the Mutagenicity dataset and then Factor XA Dataset. These will be followed by our highly imbalanced datasets AID362 and AID456.

For the Mutagenicity dataset, the fingerprints behaved differently in the presence of each classifier. There was no one particular fingerprint that performed better consistently throughout the experiments performed. In general though, PubChem and MACCS produced better results than the other fingerprints used. The classifiers which did stand out in the presence of each fingerprint were Majority Voting and Random Forest (Table 16), albeit not highly significantly with respect to other classifiers. Applying PCA did not affect the performance of the classifiers used as much as anticipated i.e. the Euclidean distance to the top left corner of a sensitivity-specificity plane was not reduced (Figure 322).

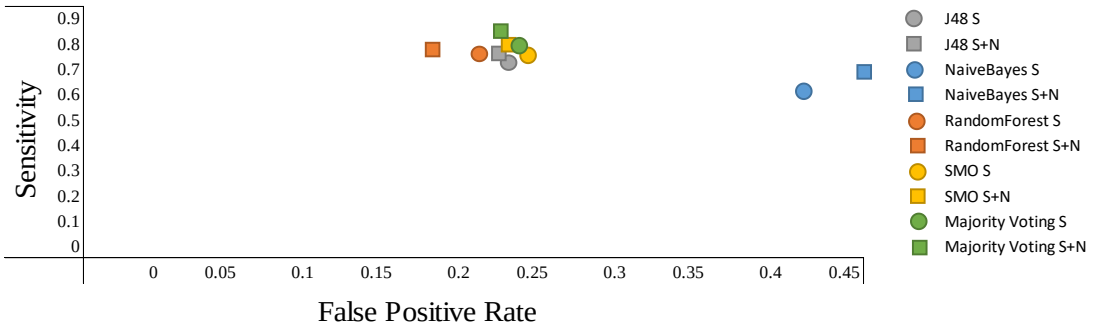


Figure 322: Bursi dataset classifiers' performance

The best performance for the methods is achieved when using the original dataset in the classification process. The reason could be that when a dataset is less imbalanced or not at all then no pre-processing is needed (Yin & Gai 2015).

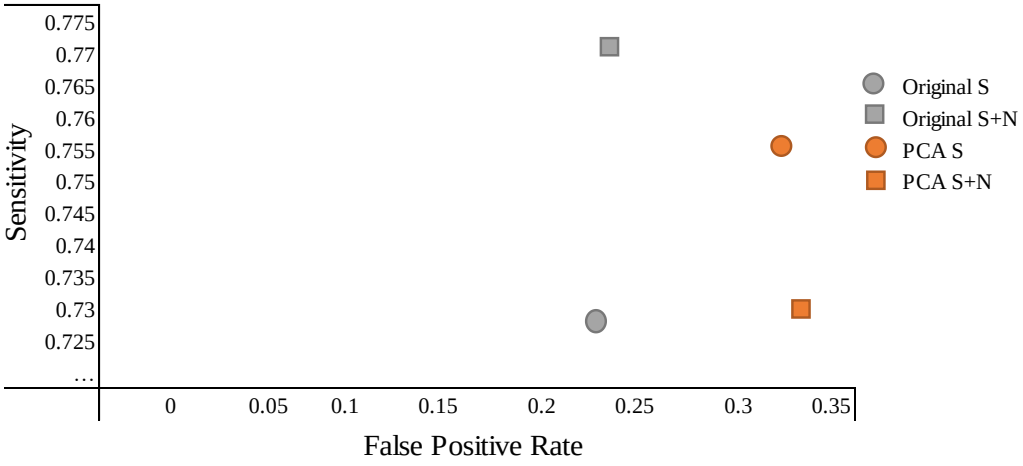


Figure 323: Bursi dataset methods' performance

The classifiers that performed best for the mutagenicity dataset were Random Forest and Majority Voting especially with the addition of numerical attributes (Please see Figure 322).

Results of this benchmark, nearly fully balanced, dataset indicate that despite its complexity, a classical approach consisting of data management and pre-processing followed by virtually a*ny* competitive classification approach directly operating in the original space of the data (i.e. the fingerprints) would suffice. Hence, the critical bottleneck for the standard approaches seems to be not in the dimensionality of the space i.e. the number of attributes produced by the fingerprints alone but also the size of the training set, the degree of overlapping between classes and rather specifically on how imbalanced they are (Prati et al. 2004).

The Factor XA dataset is the moderately imbalanced dataset. The fingerprints behaved differently with the classifiers used; there was no one fingerprint that could be pointed out as the consistent better performing. In general MACCS, Pharmacophore and PubChem performed better than the other fingerprints for this dataset. As for the classifiers used, Random Forest definitely outperformed other classifiers as indicated by the Euclidean distance to the (0,1) vector of the sensitivity-specificity plane (Figure 325) and stood out as the better performing classifier, regardless of the fingerprint or method used.

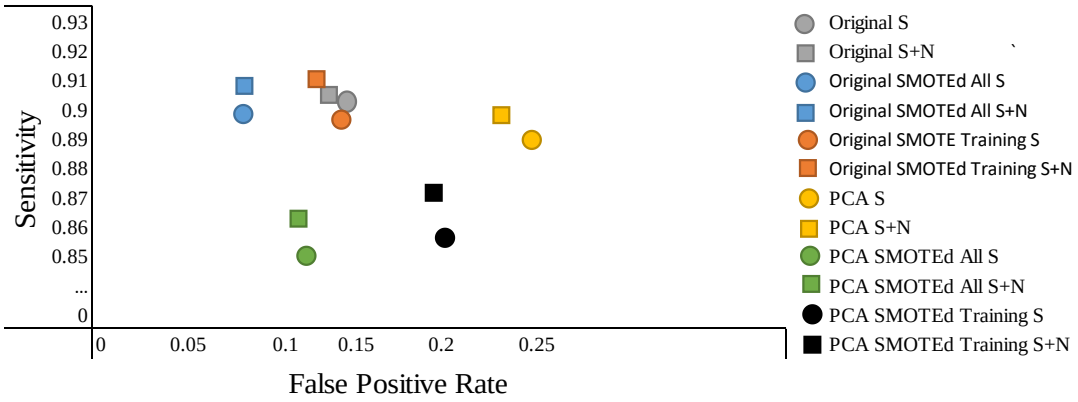


Figure 324: Fontaine dataset methods' performance

Moreover, in this slightly imbalanced dataset the oversampling played a significant role. The better method for use with the Factor XA dataset was when the dataset was balanced using SMOTE and then split into training (60%) and test (40%) and then classified (Figure 324).

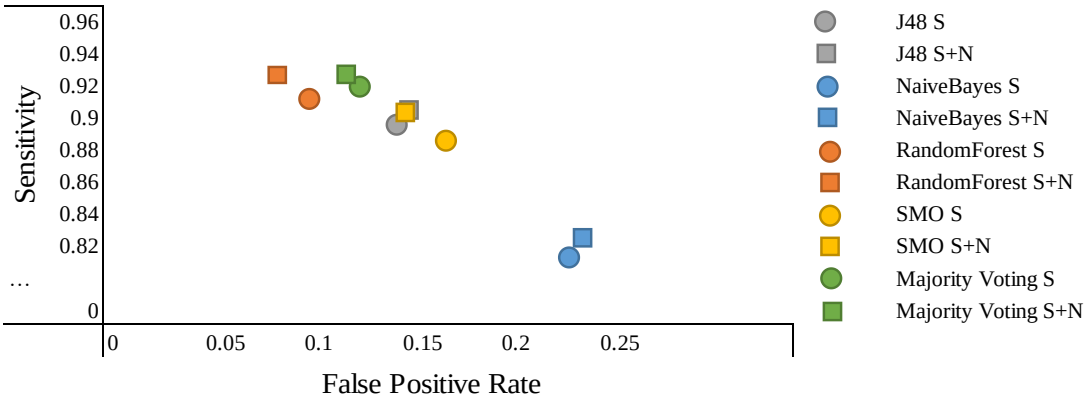


Figure 325: Fontaine dataset classifiers' performance

Random Forest was the one classifier that performed better than any other classifier, both when used with binary only descriptors and when numerical descriptors were added (Figure 325) A potential reason is the fact that Random Forest is resistant to over-training and the risk of overfitting. It is also resilient to outliers, deals with missing values, is insensitive to data skew and robust to a high number of variable inputs (Mascaro et al. 2014; Youssef et al. 2015).

AID362 presents the dataset with a high imbalance ratio and moderately high number in instances compared to the two previous datasets. With regards to the fingerprints used, MACCS and PubChem appear to have produced the better results with most classifiers used, with Pharmacophore and on occasion Substructure. However this good performance was not consistent throughout the tests. Surprisingly, in sharp contrast with the previous datasets, NaïveBayes stands out as the classifier that consistently performed better than the others. Since Naïve Bayes would perform optimally when the class-probability distributions are normal, this probably can be explained on the basis of the effect of the oversampling; in the minority class, lots of samples have had IDs generated and mixed with the original ones. The resulting process possibly renders class-probability distributions which tend to be closer to Gaussian distributions than the original ones where, at least in the minority class, due to the central limit theorem (Bishop, 2006). Here, an extremely simple, conservative approach would be the optimal choice.

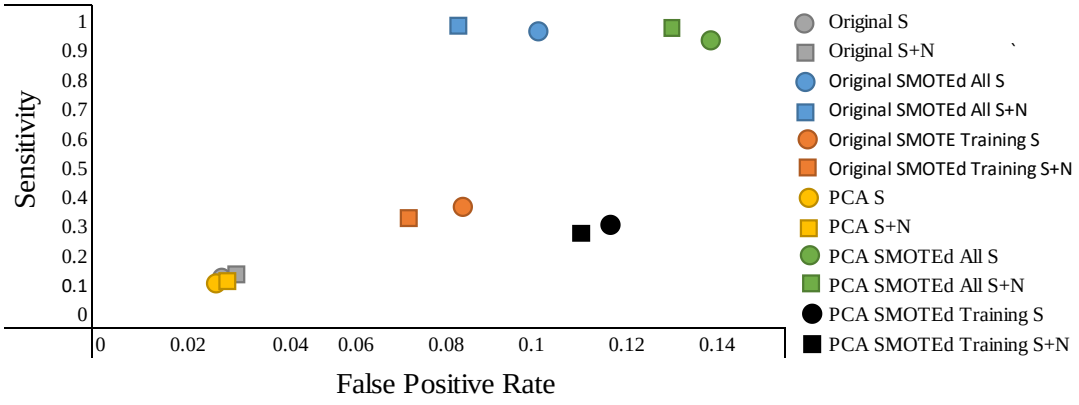


Figure 326: AID362 dataset methods' performance

With the AID362 dataset, when only binary descriptors were used, the method in which the dataset was balanced first and then split into training and test set performed best (See orange circle sign in Figure 326). When numerical descriptors were added, the same method mentioned above, stood out with the best performance i.e. the closest distance to the (0,1) corner (See orange square sign in Figure 326).

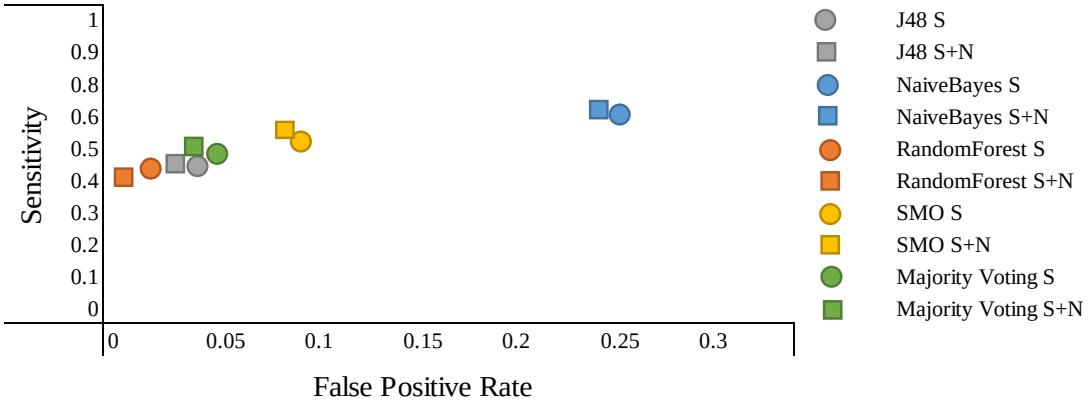


Figure 327: AID362 classifiers' performance

As with the classifiers used, in the case of this dataset, Random Forest stands out with the best results among all other classifier, both with and without the use of numerical descriptors (Figure 327).

Finally, we focused on AID456, which is by far the most imbalanced dataset with the most instances present in the dataset amongst the ones chosen for this study. In experiments performed for this dataset we have not seen as much improvement overall, however specificity and false positive and, on an occasion accuracy, have shown great improvement when numerical descriptors were added.

The fingerprints MAACS, Pharmacophore and PubChem appear to show the most diversity in their produced results. Most other fingerprints especially the CDK Fingerprinter, CDK Extended Fingerprinter and CDK Graph-only appear to have results that correspond to the classifier being biased towards the majority class. With regards to the classifiers used for this dataset, there is really no one classifier that consistently performed better or had the most improvement with the addition of numerical descriptors.

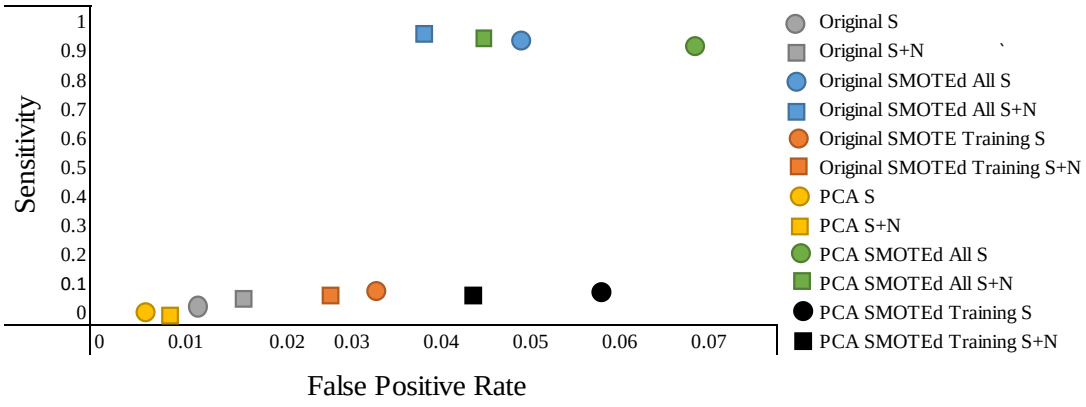


Figure 328: AID456 methods' performance

With AID456, the method with which the dataset was balanced first and then split into training and test set stands out with by far better results of all other methods used (Figure 328). In this figure we see that Original SMOTEd All and PCA SMOTEd All have the better performance of all other methods. This performance enhances as numerical attributes are added.

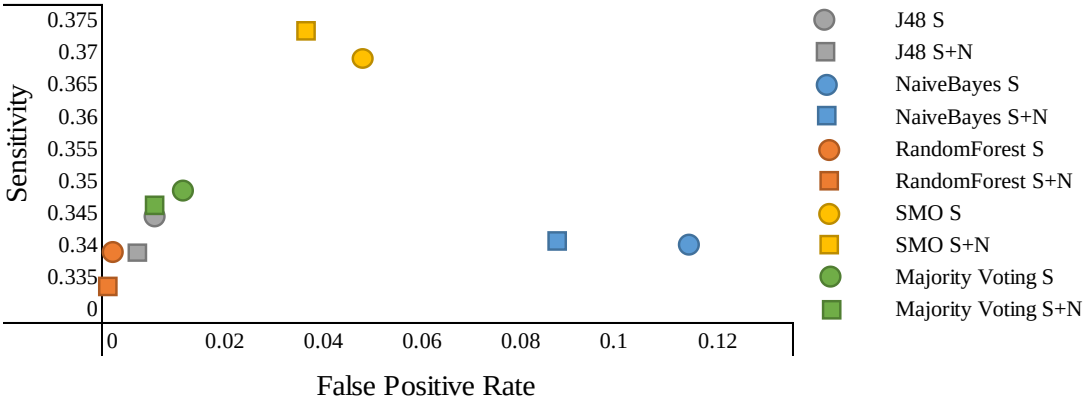


Figure 329: AID456 classifiers' performance

Interestingly, with regards to the classifiers used to classify AID456, overall SMO (linear ν-SVM) has outperformed all other classifiers both with and without the addition of numerical attributes. This result contrasts with the success of the

NaïveBayes classifier in the previous dataset (AID362), which can be perhaps explained on the basis of the different characteristics of these two heavily imbalanced dataset: AID362 has many more original data patterns available (45% more) and thus the oversampling may not have had a strong effect in the "normalisation" of the class probabilities as in the previous dataset. On the other hand, *kernel* approaches are, in general, particularly effective for classifying instances which are entangled in the space spanned by the variables of the system (Scholkopf & Smola, 2002). The SVM algorithm can be used in combination with *kernel* approaches that allow us to expand the original space until the problem because linearly separable (Bishop, 2006); and can be modified to deal with noise the training set class labels (the ν-SVM used in this thesis). However, in this thesis, we did not observe any significant advantages on using non-linear kernel functions, probably due to the intrinsic high-dimensionality of the problem.

In our experiments. Random Forest has generally been the better classifier consistently with the existing literature in highly imbalanced dataset classification. In cases that ν-SVM has outperformed Random Forest it is likely due to the fact that the class boundaries were clear enough for it to separate classes with sufficient margin. Another potential reason for this slight disadvantage of SVM is that this is a very conservative classifier, which is based on "pessimistic bounds of generalisation" (Scholkopf and Smola, 2002) designed to minimise the risk of future misclassification; but at the cost of being less flexible to adapt to a specific dataset.

More generally, the overall results achieved from classifying the AID362 and AID456 datasets using the different methods suggests that unlike the situation where the dataset is nearly balanced, when the imbalance ratio rises, the need for oversampling becomes obviously evident. However the question remains whether this improvement in results and good outcome and performance is due to the balance of the dataset being restored i.e. the distribution of the minority class samples is even across the feature space. As well as being productive, SMOTE can present several drawbacks with regards to its blind over-sampling (refer back to section 4.2 , and sub-section SMOTE for more clarification).

These drawbacks include the following (Sáez et al, 2015):

- Creating too many examples around unnecessary positive examples which do not facilitate in the learning of the minority class.
- Introducing noisy positive examples in areas belonging to the majority class
- Creating borderline positive examples and disrupting the boundaries between the different classes in the dataset.

Therefore, the question remains: did SMOTE restore the imbalance but only to add to the problem of sparseness in the feature space as mentioned above? The other question with regards to balancing the imbalanced datasets is the optimal balance ratio as discussed in Dittman & Khoshgoftar (2014). It was found that a 50:50 balance ratio between the classes is not always the optimal and appropriate final class ratio for all scenarios of classifying datasets with high levels of class imbalance.

Concluding Remarks and Future Work

At the beginning of this project a number of objectives were set in order to achieve the goal. In the course of completing the project, the various methods with which large datasets with imbalance between the classes are classified were investigated and researched. These methods fall into two big categories of manipulating the dataset (external manipulation) and manipulating the classifier (internal manipulation e.g. cost-sensitive classification). When performing the external manipulation one usually performs feature selection and / or sampling techniques. In this project we set on a journey to combine the use of fingerprinting methods with the SMOTE technique in order to analyse the virtual screening of large and highly imbalanced datasets in a unified manner. We successfully fulfilled this goal and performed the necessary experiments.

We successfully generated eight fingerprints for the datasets used in this study and as a result 16 unique datasets were born from each of the original datasets. The SMOTE technique was successfully used in order to bring balance between the majority and minority classes in our datasets in two different manners. In the first manner the datasets were balanced and then split into training and test set. In the second manner the datasets were first split into training and test sets and then only the training set was balanced. This action itself doubled the number of our already unique datasets resulting in a total of 128 datasets that were used for our study.

When results were gathered, the relevant classification metrics were compared and the classifiers, fingerprints and methods which produced better results were chosen in order to observe any patterns or possible combination. At the end we found that with datasets that have moderate to higher levels of imbalance, pre-processing is needed in order to restore balance to the dataset. The balancing method SMOTE in conjunction with Random Forest and Majority Voting produced the best results out of the classifiers used in this study for our imbalanced datasets. However on occasion NaïveBayes and SMO have been seen to outperform the former two possibly due to the differential effects of oversampling with respect to the dimensionality and number of data patterns, as discussed before . With regards to the fingerprinting methods, the fingerprints MACCS, Pharmacophore and PubChem have shown promising results in the classification process. The performance of a classifier largely depends on the underlying distribution of the data in each class (Lin

& Chen 2012). A large standard deviation (variance) between the classes results in between-classes overlap. As a result the minority class instances are likely to be classified as majority class instances as there are more majority class instances in the overlapping area.

When it comes to imbalanced datasets, the most obvious characteristic is the skewed data distribution between the different classes. Research shows that this is not the only cause of the difficulties for modelling a capable classifier. Other parameters involved are small sample size (very limited minority samples available) which could lead to overfitting (Chen & Wasikowski, 2008; TaşCı, Ş. and Güngör, 2013), class overlapping and small disjoints which are the presence of within-class sub-concepts (Sun et al, 2009; Sáez et al, 2016). Recently, many solutions have been introduced to solve the binary imbalanced classification problem (see section 3.1 tables 1 through 4), and therefore the use of multi-class classifiers is not mandatory. However, this possibility has been explored in other settings (Sáez et al, 2016).

Multi-class problems are more involved than their binary counterparts because of the more complex relationship between their classes. In a binary setting the classes have a well-defined relationship between the classes: one class is the majority and the other is the minority. In a multi-class situation, a certain class can be a majority class in relation to a given subset of classes or a minority class. It could even be of similar distribution to some of them (see Figure 330).

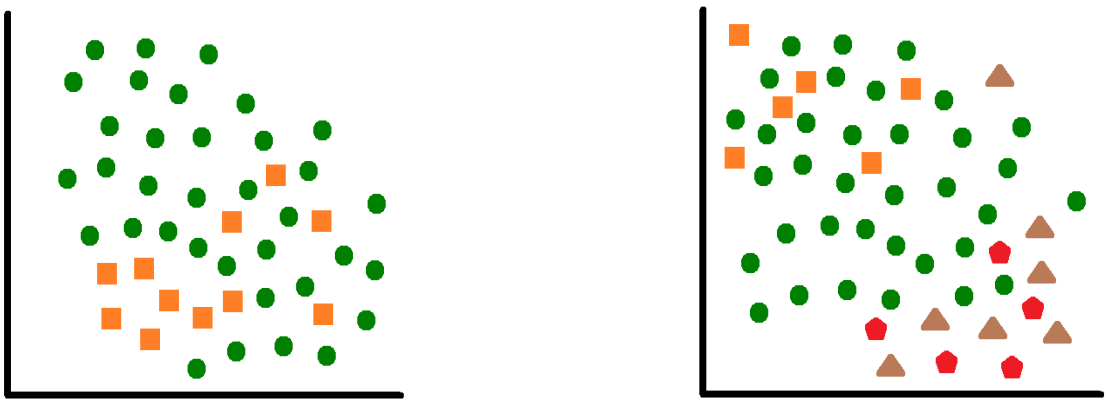


Figure 330: Possible class imbalance scenarios (Amended from Sáez et al. 2016, p.161)

We can see in Figure 330 two possible class imbalance scenarios. On the left side, a binary imbalanced problem and on the right hand a multi-class imbalanced problem. In the case of the multi-class problem the relationship between the classes is evidently more complicated.

There have been a few proposals for solutions to this problem in the literature. In Fernández-Navarro et al. (2011) the concept of static-SMOTE has been introduced. In this method, the resampling is applied in M steps, where M is the number of classes. With each iteration, the resampling technique selects the minimum-sized class and increases the number of instances of that class in the original dataset. An ensemble learning algorithm for multi-majority and multi-minority was proposed in Wang and Yao (2012). Here the authors combine AdaBoost and negative correlation learning. As with binary class imbalanced datasets, cost-sensitive neural networks based on over-sampling, under-sampling and moving thresholds have been adapted to multi-class imbalanced classification (Zhou & Liu, 2006). The most popular solution is probably the one where the multi-class structure is broken down into binary ones (Hoens et al, 2012; Nag & Pal, 2016). However, one needs to be careful as multi-class imbalanced datasets introduce new difficulties. As mentioned above, unlike binary cases, in multi-class cases classes can be a minority and or majority depending on the way they are looked at. The following situations can form:

- Many minority, one majority
- One minority, many majority
- Many minority, many majority

In order to overcome these issues, there is a need to identify the nature of the different types of examples in a multi-class imbalanced dataset in order to understand the characteristics of the distribution of each minority class and how to proceed with it (Kubat & Matwin, 1998; Napierala & Stefanowski, 2016). In the research by Napierala and Stefanowski (2012; 2016), the minority class examples were divided into four different groups:

- Safe examples: are in regions surrounded by members of same class and separated from the other class.
- Borderline examples: are in the boundary regions of classes, where examples overlap.
- Rare examples: these examples are also situated in boundaries of regions but surrounded more by the other class than their own.
- Outliers: are isolated examples surrounded by examples of other classes.

In the research conducted by Sáez et al, (2016), a framework was proposed so that the groups mentioned above are extended to accommodate for multi-class cases. Figure 331 illustrates this extended concept.

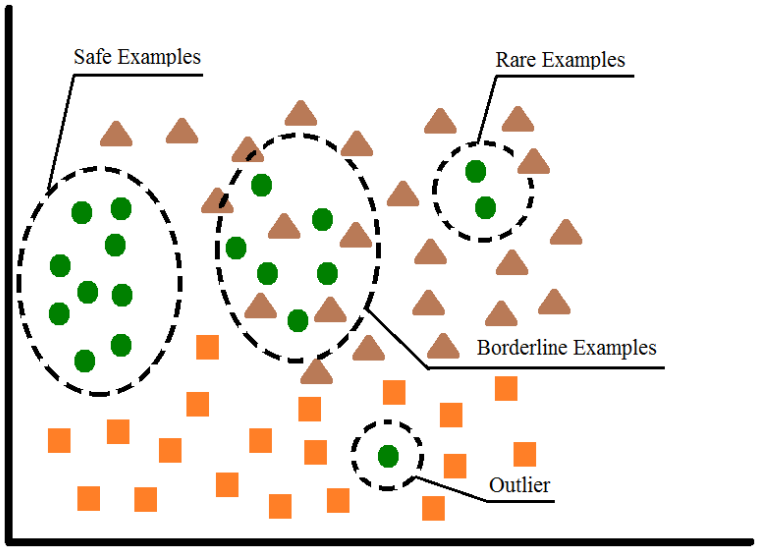


Figure 331: Various types of examples identified in a multi-class situation (Sáez et al. 2016, p.167)

In this framework, the over-sampling is computed based on the type of example from each class. The emphasis here is not so much on the over-sampling, but it is to show that multi-class tasks are complex structures that are made up of heterogeneous examples that vary in the levels of difficulty.

In a nutshell, there has been research done into the classification of multi-class imbalanced datasets; yet still the most prevalent method for unbalanced datasets such as the ones presented in this study is decomposing the problem into binary problems by taking into consideration the type of examples. Thus, there are different possibilities to tackle imbalanced datasets, and the effect on the dataset cannot be inferred intuitively in many cases. In order to understand better the effect of the oversampling in a specific dataset, it is interesting to evaluate how the over-sampling affects both training set and the test sets when cross validation is performed (Sáez et al, 2016); as is shown in this thesis.

This framework can be extended in a relatively straightforward fashion to different settings than the one studied in this work, where a multi-class problem definition is advantageous. Towards this goal, SMOTE can be adapted to bring balance to multiple classes (Fernández et al. 2010; Prachuabsupakij &

Soonthornphisaj 2012; Wang & Yao 2012; Tomar & Agrawal 2015). As mentioned previoulsy, one of the main methods in solving multi-class imbalanced classification is the use of binarisation strategy where the problem is decomposed into binary problems and a different binary model is learned for each new subset (Galar et al. 2011). Multi-class versions of the robust classifiers used in this work are well-known in the literature (Aly 2005; Bishop 2006; Venkatesan & Er 2016). However, it must be mentioned that the computational cost of the exigent cross-validation would increase and may require the use of approximate computations for real-time applications.

Likewise, to explore in more detail the effect of recent approaches to balance datasets such as Recursive Feature Elimination (Maldonado et al. 2014; discussed in this thesis) is another interesting future direction. However, it is worth to stress that, in the light of our results, we hypothesise that it is not likely that other classifiers or recent SMOTE variants render a statistically significant improvement in the sensitivity-specificity trade-off. This suggestion is based in that the optimal approaches, although different through datasets (Random Forests, Ensemble, ν-SVM), perform statistically similarly (see for instance summary figures). Indeed, as reported in the literature (Sáez et al. 2014; Murphree et al. 2015), an ensemble of such classifiers is typically advantageous in providing robust and uniform performance simultaneously for a range of heterogeneous scenarios; such as the ones addressed in this thesis.

Nevertheless, it is possible that very recent approaches which are revolutionising the areas of big data classification and encoding, such as deep learning auto encoders reformulated for binary or multi-class classification purposes (Vincent et al. 2010) used as individual learners in ensemble, would be flexible yet robust enough to improve the results shown in this thesis. These approaches exhibit unprecedented adaptation capability to heterogeneous datasets such as the ones studied in this thesis, and they would therefore constitute an interesting future extension of our study.

7. Bibliography

Abbas, A., 2003. Grid Computing Technology - An Overview. In *Grid Computing: A Practical guide to Technology and Applications*. Herndon, VA, USA: Charles River Media / Cengage Learning, pp. 43–73.

Abdi, H. and Williams, L.J., 2010. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4), pp.433-459. Available from: http://ead.ipleiria.pt/ucs201415/pluginfile.php/168687/mod_resource/content/14/ABDI-WIRE%20CS-PCA%202010.pdf [Accessed 18 June 2012]

Aggarwal, C.C. and Zhai, C. eds., 2012. *Mining text data*. Springer Science & Business Media. Available from: https://books.google.com/books?hl=en&lr=&id=vFHOx8wfSU0C&oi=fnd&pg=PR3&dq=Mining+text+data&ots=obaaUGlASr&sig=B8dbYKg5ISLzeC9i1fczrZu8ePU#v=onepage&q=Mining%20text%20data&f=false [Accessed 22 June 2013]

Alshomrani, S., Bawakid, A., Shim, S.O., Fernández, A. and Herrera, F., 2015. A proposal for evolutionary fuzzy systems using feature weighting: dealing with overlapping in imbalanced datasets. *Knowledge-Based Systems*, 73, pp.1-17. Available from: http://www.sciencedirect.com/science/article/pii/S0950705114003323 [Accessed 14 March 2016]

Aly, M., 2005. Survey on multiclass classification methods. *Neural Netw*, pp.1-9. Available from: https://www.vision.caltech.edu/malaa/publications/aly05multiclass.pdf [Accessed 21 February 2017]

Anand, P., 2013. Big Data Is a Big Deal. *Journal of Petroleum Technology*, 65(04), pp.18-21. Available from: http://198.63.44.44/documents/JPTPradeepAnandBigDataisaBigDeal1304.pdf [Accessed 23 January 2015]

Anil Kumar, D. and Ravi, V., 2008. Predicting credit card customer churn in banks using data mining. *International Journal of Data Analysis Techniques and Strategies*, 1(1), pp.4-28. Available from: http://www.inderscienceonline.com/doi/abs/10.1504/IJDATS.2008.02002 [Accessed 12 June 2014]

Armbrust, M. et al., 2010. A view of cloud computing. *Communications of the ACM*, 53(4), p.50. Available from: http://portal.acm.org/citation.cfm?doid=1721654.1721672 [Accessed February 28, 2013]

Bajorath, J. ed., 2011. Chemoinformatics and computational chemical biology. Humana Press

Bajorath, J., 2002. Integration of virtual and high-throughput screening. *Nature reviews. Drug discovery*, 1(11), pp.882–94. Available from: http://www.ncbi.nlm.nih.gov/pubmed/12415248 [Accessed March 5, 2013]

Balaban, A.T., 1985. Applications of graph theory in chemistry. *Journal of chemical information and computer sciences*, 25(3), pp.334-343. Available from: http://oak.ucc.nau.edu/ctd27/F14/226/Quizzes/Balaban.pdf [Accessed September 9, 2013]

Balderud, L.Z., Murray, D., Larsson, N., Vempati, U., Schürer, S.C., Bjäreland, M. and Engkvist, O., 2014. Using the BioAssay Ontology for analyzing high-throughput screening data. *Journal of biomolecular screening*, p.1087057114563493. Available from: http://jbx.sagepub.com/content/early/2014/12/12/1087057114563493.abstract [Accessed 16 August 2015]

Barnes, M.R., Harland, L., Foord, S.M., Hall, M.D., Dix, I., Thomas, S., Williams-Jones, B.I. and Brouwer, C.R., 2009. Lowering industry firewalls: pre-competitive informatics initiatives in drug discovery. *Nature Reviews Drug Discovery*, *8*(9), pp.701-708. Available from: http://www.nature.com/nrd/journal/v8/n9/abs/nrd2944.html [Accessed 22 February 2017]

Basak, S.C., Magnuson, V.R., Niemi, G.J. and Regal, R.R., 1988. Determining structural similarity of chemicals using graph-theoretic indices. *Discrete Applied Mathematics*, 19(1), pp.17-44. Available from: http://www.sciencedirect.com/science/article/pii/0166218X88900042 [Accessed September 24, 2013]

Batuwita, R. & Palade, V., 2009. microPred: effective classification of pre-miRNAs for human miRNA gene prediction. *Bioinformatics (Oxford, England)*, 25(8), pp.989–95. Available from: http://www.ncbi.nlm.nih.gov/pubmed/19233894 [Accessed March 8, 2013]

Batuwita, R. and Palade, V., 2013. Class imbalance learning methods for support vector machines. *Imbalanced learning: Foundations, algorithms, and applications*, *83*. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.899.8839&rep=rep1&type=pdf [Accessed 24 February 2017]

Batuwita, R. and Palade, V., 2013. Class imbalance learning methods for support vector machines. *Imbalanced learning: Foundations, algorithms, and applications*, pp.83-99. Available from: http://onlinelibrary.wiley.com/doi/10.1002/9781118646106.ch5/summary [Accessed 2 May 2015]

Bekkar, M. and Alitouche, T.A., 2013. Imbalanced data learning approaches review. *International Journal of Data Mining & Knowledge Management Process*, 3(4), p.15. Available from: http://aircconline.com/ijdkp/V3N4/3413ijdkp02.pdf [Accessed March 3, 2015]

Bertolami, R. and Bunke, H., 2008. Ensemble methods to improve the performance of an English handwritten text line recognizer. In *Arabic and Chinese Handwriting Recognition* (pp. 265-277). Springer Berlin Heidelberg.

Bishop, C.M., 2006. Pattern recognition. *Machine Learning*, *128*, pp.1-58. Available from: http://s3.amazonaws.com/academia.edu.documents/30428242/bg0137.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1491163465&Signature=oJE5t3N5mB1o7FQh3YK5RyjXa24%3D&response-content-disposition=inline%3B%20filename%3DPattern_recognition_and_machine_learning.pdf [Accessed 19 February 2016]

Blagus, R. and Lusa, L., 2012, December. Evaluation of smote for high-dimensional class-imbalanced microarray data. *In Machine learning and applications (icmla), 2012 11th international conference on* (Vol. 2, pp. 89-94). IEEE. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6406733 [Accessed 2 June 2015]

Blagus, R. & Lusa, L., 2013. SMOTE for high-dimensional class-imbalanced data. *BMC Bioinformatics*, 14(1), p.106. Available from: http://www.biomedcentral.com/1471-2105/14/106 [Accessed March 23, 2013]

Bleicher, K.H. et al., 2003. Hit and Lead Generation: Beyond High-Throughput Screening. *Nature reviews. Drug discovery*, 2(5), pp.369–78. Available from: http://www.ncbi.nlm.nih.gov/pubmed/12750740 [Accessed March 3, 2013]

Bosch, A., Zisserman, A. and Munoz, X., 2007, October. Image classification using random forests and ferns. In Computer Vision, 2007. ICCV 2007. *IEEE 11th International Conference on* (pp. 1-8). IEEE. Available from:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4409066 [Accessed March 10, 2016]

Bouckaert, R.R. et al., 2013. *Weka Manual for Version 3-7-8*, Hamilton, New Zealand.

Bradley, D., 2008. . *Nature Reviews: Drug Discovery*, *7*, pp. 632-633. Available from: http://resolver.ebscohost.com/openurl?sid=google&auinit=D&aulast=Bradley&atitle=Dealing+with+a+data+dilemma&id=doi%3a10.1038%2fnrd2649&title=Nature+Reviews+Drug+Discovery&volume=7&issue=8&date=2008&spage=632&linksourcecustid=518&site=ftf-live [Accessed 17 February 2017]

Branco, P., Torgo, L. and Ribeiro, R.P., 2016. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, *49*(2), p.31. Available from: http://dl.acm.org/citation.cfm?id=2907070 [Accessed 19 February 2017]

Breiman, L., 2001. Random Forests. *Machine Learning*, 45(1), pp.5–32. Available from: http://link.springer.com/article/10.1023%2FA%3A1010933404324?LI=true [Accessed 9 April 2011]

Bro, R. and Smilde, A.K., 2014. Principal component analysis. *Analytical Methods*, *6*(9), pp.2812-2831. Available from: http://pubs.rsc.org/is/content/articlehtml/2014/ay/c3ay41907j [Accessed 17 February 2015]

Brown, N., McKay, B. and Gasteiger, J., 2005. Fingal: A Novel Approach to Geometric Fingerprinting and a Comparative Study of Its Application to 3D-QSAR Modelling. *QSAR & Combinatorial Science*, 24(4), pp.480-484. Available from: http://onlinelibrary.wiley.com/doi/10.1002/qsar.200430923/abstract [Accessed September 29, 2015]

Brown, N., 2009. Chemoinformatics—an introduction for computer scientists. *ACM Computing Surveys*, 41(2), pp.1–38. Available from: http://portal.acm.org/citation.cfm?doid=1459352.1459353 [Accessed March 5, 2013]

Burden, F.R., 1989. Molecular identification number for substructure searches. *Journal of Chemical Information and Computer Sciences*, 29(3), pp.225-227. Available from: http://pubs.acs.org/doi/pdf/10.1021/ci00063a011 [Accessed 11 August 2012]

Cai, Q., He, H. and Man, H., 2014. Imbalanced evolving self-organizing learning. *Neurocomputing*, 133, pp.258-270. Available from: http://www.ele.uri.edu/faculty/he/PDFfiles/neurocomputing_imbalancedlearning.pdf [Accessed 12 July 2015]

Cannataro, M. et al., 2004. Distributed Data Mining on Grids: Services, Tools, and Applications. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 34(6), pp.2451–2465. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1356036 [Accessed March 8, 2013]

Cereto-Massagué, A., Ojeda, M.J., Valls, C., Mulero, M., Garcia-Vallvé, S. and Pujadas, G., 2015. Molecular fingerprint similarity search in virtual screening. *Methods*, 71, pp.58-63. Available from: https://www.researchgate.net/profile/Adria_Cereto-Massague/publication/264866258_Molecular_fingerprint_similarity_search_in_virtual_screening/links/553f986e0cf2736761c02d26.pdf [Accessed November 23, 2015]

Chawla, N. V et al., 2002. SMOTE: Synthetic Minority Over-Sampling Technique. *Journal of Artificial Intelligence Research*, 16(1), pp.321–357. Available from: http://www.jair.org/papers/paper953.html [Accessed 2 September 2012]

Chawla, N.V., Japkowicz, N. and Kotcz, A., 2004. Editorial: special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1), pp.1-6. Available from: https://www3.nd.edu/~dial/publications/chawla2004editorial.pdf [Accessed 14 February 2014]

Chawla, N.V., 2009. Data mining for imbalanced datasets: An overview. In Data mining and knowledge discovery handbook (pp. 875-886). *Springer US*. Available from: https://www3.nd.edu/~nchawla/papers/SPRINGER05.pdf [Accessed 29 August 2013]

Chen, X.W. and Wasikowski, M., 2008, August. Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 124-132). ACM. Available from: http://dl.acm.org/citation.cfm?id=1401910 [Accessed December 19, 2016]

Chi, M., Shen, J., Sun, Z., Chen, F. and Benediktsson, J.A., 2014, July. Oil spill detection based on web crawling images. In Québec Canada: *IEEE the International Geoscience and Remote Sensing Symposium* (IGARSS). Available from: ftp://ftp.legos.obs-mip.fr/pub/tmp3m/IGARSS2014/abstracts/2867.pdf [Accessed January 18, 2016]

Cieslak, D.A., Chawla, N.V. and Striegel, A., 2006, May. Combating imbalance in network intrusion datasets. In *GrC* (pp. 732-737). Available from: https://pdfs.semanticscholar.org/5f3b/bb2ff46445a7638fcbb3a2de727e36974923.pdf [Accessed 22 February 2017]

Cuzzocrea, A., 2014, November. Privacy and security of big data: current challenges and future research perspectives. *In Proceedings of the First International Workshop on Privacy and Secuirty of Big Data* (pp. 45-47). ACM. Available from: http://dl.acm.org/citation.cfm?id=2669614 [Accessed 16 November 2015]

DAYLIGHT Chemical Information Systems, I., 2008. Fingerprints - Screening and Similarity. Available from: http://www.daylight.com/dayhtml/doc/theory/theory.finger.html

Dal Pozzolo, A., Caelen, O., Le Borgne, Y.A., Waterschoot, S. and Bontempi, G., 2014. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert systems with applications*, 41(10), pp.4915-4928. Available from:http://www.ulb.ac.be/di/map/adalpozz/pdf/FraudDetectionPaper_8.pdf [Accessed January 11, 2015]

Davies, T. and Fearn, T., 2004. Back to basics: the principles of principal component analysis. *Spectroscopy Europe*, 16(6), p.20. Available from: ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf [Accessed 21 January 2015]

Dietterich, T.G., 2000, June. Ensemble methods in machine learning. In *International workshop on multiple classifier systems* (pp. 1-15). Springer Berlin Heidelberg. Available from: http://link.springer.com/chapter/10.1007/3-540-45014-9_1 [Accessed 25 February 2017]

Dittman, D.J., Khoshgoftaar, T.M. and Napolitano, A., 2014, November. Selecting the appropriate data sampling approach for imbalanced and high-dimensional bioinformatics datasets. *In Bioinformatics and Bioengineering (BIBE), 2014 IEEE International Conference on* (pp. 304-310). IEEE. Available from: http://ieeexplore.ieee.org/document/7033597/?arnumber=7033597&tag=1 [Accessed 5 June 2016]

Downs, G.M. and Willett, P., 1996. Similarity searching in databases of chemical structures. *Reviews in computational chemistry*, 7, pp.1-66.

Dubey, R., Zhou, J., Wang, Y., Thompson, P.M., Ye, J. and Alzheimer's Disease Neuroimaging Initiative, 2014. *Analysis of sampling techniques for imbalanced data: An n= 648 ADNI study. NeuroImage*, 87, pp.220-241. Available from: http://www.sciencedirect.com/science/article/pii/S1053811913010161 [Accessed 11 August 2015]

Ducange, P., Lazzerini, B. and Marcelloni, F., 2010. Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets. *Soft Computing*, 14(7), pp.713-728. Available from: http://link.springer.com/article/10.1007/s00500-009-0460-y [Accessed 14 November 2014]

Dutt, R. and Madan, A.K., 2013. Models for the prediction of PPARs agonistic activity of indanylacetic acids. *Medicinal Chemistry Research*, *22*(7), pp.3213-3228. Available from: http://link.springer.com/article/10.1007/s00044-012-0315-4 [Accessed 24 February 2017]

Džeroski, S. and Ženko, B., 2004. Is combining classifiers with stacking better than selecting the best one?. *Machine learning*, *54*(3), pp.255-273. Available from: http://link.springer.com/article/10.1023%2FB%3AMACH.0000015881.36452.6e?LI=true [Accessed 25 February 2017]

Eckert, H. & Bajorath, J., 2007. Molecular Similarity Analysis in Virtual Screening: Foundations, Limitations and Novel Approaches. *Drug discovery today*, 12(5-6), pp.225–33. Available from: http://www.ncbi.nlm.nih.gov/pubmed/17331887 [Accessed March 10, 2013].

Ekins, S., 2014. Progress in computational toxicology. *Journal of pharmacological and toxicological methods*, 69(2), pp.115-140. Available from: http://www.sciencedirect.com/science/article/pii/S1056871913003250 [Accessed 17 September 2015]

Elrahman, S.M.A. and Abraham, A., 2013. A review of class imbalance problem. *Journal of Network and Innovative Computing*, *1*(2013), pp.332-340. Available from: http://ias04.softcomputing.net/jnic2.pdf [Accessed 23 February 2017]

Ertekin, S. et al., 2007. Learning on the Border: Active Learning in Imbalanced Data Classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM '07*. New York, New York, USA: ACM Press, pp. 127–137. Available from: http://portal.acm.org/citation.cfm?doid=1321440.1321461 [Accessed March 5, 2013]

Estabrooks, A., Jo, T. and Japkowicz, N., 2004. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1), pp.18-36. Available from: http://onlinelibrary.wiley.com/doi/10.1111/j.0824-7935.2004.t01-1-00228.x/abstract [Accessed 18 February 2012]

Fallahi, A. & Jafari, S., 2011. An Expert System for Detection of Breast Cancer Using Data Preprocessing and Bayesian Network. *International Journal of Advanced Science and Technology*, 34, pp.65–70. Available from: https://pdfs.semanticscholar.org/5ed8/0ef6861e7e4abfc0ebb2bde80cef596f1293.pdf [Accessed 7 May 2015]

Fernández, A., Del Jesus, M.J. and Herrera, F., 2010. Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* (pp. 89-98). Springer Berlin Heidelberg. Available from: http://link.springer.com/chapter/10.1007/978-3-642-14049-5_10 [Accessed 18 February 2017]

Fernández-Navarro, F., Hervás-Martínez, C. and Gutiérrez, P.A., 2011. A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition*, 44(8), pp.1821-1833. Available from: http://www.sciencedirect.com/science/article/pii/S0031320311000823 [Accessed 23 September 2013]

Ferrari, T. and Gini, G., 2010. An open source multistep model to predict mutagenicity from statistical analysis and relevant structural alerts. *Chemistry Central Journal*, 4(Suppl 1), p.S2. Available from: http://ccj.springeropen.com/articles/10.1186/1752-153X-4-S1-S2 [Accessed 12 October 2015]

Ferrari, T., Gini, G., Bakhtyari, N.G. and Benfenati, E., 2011, April. Mining toxicity structural alerts from SMILES: a new way to derive structure activity relationships. *In Computational Intelligence and Data Mining (CIDM)*, 2011 IEEE Symposium on (pp. 120-127). IEEE. Available from: https://pdfs.semanticscholar.org/68a5/b64bc1254835b33953d414a9af9a100f2c2b.pdf [Accessed 23 May 2015]

Flake, G.W. and Lawrence, S., 2002. Efficient SVM regression training with SMO. *Machine Learning*, *46*(1-3), pp.271-290. Available from: http://link.springer.com/article/10.1023/A:1012474916001 [Accessed 2 March 2015]

Fontaine, F., Pastor, M., Zamora, I. and Sanz, F., 2005. Anchor-GRIND: filling the gap between standard 3D QSAR and the GRid-INdependent descriptors. *Journal of medicinal chemistry*, 48(7), pp.2687-2694. Available from: http://pubs.acs.org/doi/abs/10.1021/jm049113%2B [Accessed March 28, 2011]

Foster, I., Zhao, Y., Raicu, I. and Lu, S., 2008, November. Cloud computing and grid computing 360-degree compared. *In Grid Computing Environments Workshop*, 2008. GCE'08 (pp. 1-10). Ieee. Available from: https://arxiv.org/ftp/arxiv/papers/0901/0901.0131.pdf [Accessed July 2, 2015]

Fourches, D., Muratov, E. and Tropsha, A., 2010. Trust, but verify: on the importance of chemical structure curation in cheminformatics and QSAR modeling research. *Journal of chemical information and modeling*, *50*(7), p.1189. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2989419/ [Accessed 22 February 2017]

Fox, S. et al., 2006. High-throughput screening: update on practices and success. *Journal of biomolecular screening*, 11(7), pp.864–9. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16973922 [Accessed March 5, 2013]

Friedman, J., Hastie, T. and Tibshirani, R., 2001. *The elements of statistical learning* (Vol. 1). Springer, Berlin: Springer series in statistics. Available from: statweb.stanford.edu/~tibs/book/preface.ps [Accessed 23 February 2017]

Fuchs, J.E., Bender, A. and Glen, R.C., 2015. Cheminformatics Research at the Unilever Centre for Molecular Science Informatics Cambridge. Molecular informatics, 34(9), pp.626-633. Available from: http://onlinelibrary.wiley.com/doi/10.1002/minf.201400166/full [Accessed 16 January 2016]

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H. and Herrera, F., 2012. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(4), pp.463-484. Available from: http://ieeexplore.ieee.org/abstract/document/5978225/ [Accessed 25 February 2017]

Ganganwar, V., 2012. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4), pp.42-47. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.3344&rep=rep1&type=pdf [Accessed July 24, 2014]

García, V., Mollineda, R.A. and Sánchez, J.S., 2008, December. A new performance evaluation method for two-class imbalanced problems. *In Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)* (pp. 917-925). Springer Berlin Heidelberg. Available from: http://link.springer.com/chapter/10.1007/978-3-540-89689-0_95#page-1 {Accessed 29 September 2014]

García-Pedrajas, N. et al., 2012. Class imbalance methods for translation initiation site recognition in DNA sequences. *Knowledge-Based Systems*, 25(1), pp.22–34. Available from: http://linkinghub.elsevier.com/retrieve/pii/S095070511100089X [Accessed May 18, 2013]

Gawehn, E., Hiss, J.A. and Schneider, G., 2015. Deep Learning in Drug Discovery. *Molecular Informatics*. Available from: http://onlinelibrary.wiley.com/doi/10.1002/minf.201501008/pdf [Accessed February 11, 2016]

Go, E.P., 2010. Database resources in metabolomics: an overview. *Journal of Neuroimmune Pharmacology*, *5*(1), pp.18-30. Available from: http://link.springer.com/article/10.1007/s11481-009-9157-3 [Accessed 21 February 2017]

Gokgoz, E. and Subasi, A., 2015. Comparison of decision tree algorithms for EMG signal classification using DWT. *Biomedical Signal Processing and Control*, *18*, pp.138-144. Available from: http://www.sciencedirect.com/science/article/pii/S1746809414002006 [Accessed 19 February 2017]

Grossman, R. et al., 1999. The Preliminary Design of Papyrus: A System for High Performance, Distributed Data Mining over Clusters, Meta-Clusters and Super-Clusters. In *In Proceedings of Workshop on Distributed Data Mining, alongwith KDD98*. pp. 259–75. Available from: https://www.researchgate.net/profile/Stuart_Bailey2/publication/2936391_The_Preliminary_Design_of_Papyrus_A_System_for_High_Performance_Distributed_Data_Mining_over_Clusters_Meta-Clusters_and_Super-Clusters/links/551566380cf2d70ee27022d5.pdf [Accessed 2 May 2011]

Grossman, R. & Gu, Y., 2008. Data Mining Using High Performance Data Clouds. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08*. New York, New York, USA: ACM Press, pp. 920–27. Available from: http://dl.acm.org/citation.cfm?doid=1401890.1402000 [Accessed March 22, 2013].

Guha, R. et al., 2006. The Blue Obelisk-Interoperability in Chemical Informatics. *Journal of chemical information and modeling*, 46(3), pp.991–8. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16711717 [Accessed March 8, 2013]

Gunera, J. and Kolb, P., 2015. Fragment-based similarity searching with infinite color space. *Journal of computational chemistry*, 36(21), pp.1597-1608. Available from: http://onlinelibrary.wiley.com/doi/10.1002/jcc.23974/full [Accessed September 06, 2015]

Hall, M. et al., 2009. The WEKA Data Mining Software. *ACM SIGKDD Explorations Newsletter*, 11(1), pp.10–18. Available from: http://portal.acm.org/citation.cfm?doid=1656274.1656278 [Accessed March 4, 2013]

Han, H., Wang, W.Y. and Mao, B.H., 2005. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing* (pp. 878-887). Springer Berlin Heidelberg.

Han, J. & Kamber, M., 2001. *Data Mining: Concepts and Techniques*, San Francisco, Calif.: Morgan Kaufmann.

Han, L., Wang, Y. & Bryant, S.H., 2008. Developing and validating predictive decision tree models from mining chemical structural fingerprints and high-throughput screening data in PubChem. *BMC bioinformatics*, 9, p.401. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2572623&tool=pmcentrez&rendertype=abstract [Accessed March 27, 2013]

Han, L., Suzek, T.O., Wang, Y. and Bryant, S.H., 2010. The Text-mining based PubChem Bioassay neighboring analysis. *BMC bioinformatics*, 11(1), p.1. Available from: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-11-549 [Accessed 27 March 2015]

Hao, M., Bryant, S.H. and Wang, Y., 2016. Cheminformatics analysis of the AR agonist and antagonist datasets in PubChem. *Journal of cheminformatics*, *8*(1), pp.1-13. Available from: http://link.springer.com/article/10.1186/s13321-016-0150-6 [Accessed 21 February 2017]

Hao, M., Wang, Y. and Bryant, S.H., 2014. An efficient algorithm coupled with synthetic minority over-sampling technique to classify imbalanced PubChem BioAssay data. *Analytica chimica acta*, 806, pp.117-127. Available from: http://www.sciencedirect.com/science/article/pii/S0003267013013937 [Accessed 6 November 2015]

He, H. & Garcia, E.A., 2009. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), pp.1263–1284. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5128907 [Accessed February 28, 2013].

He, H. and Ma, Y. eds., 2013. Imbalanced learning: foundations, algorithms, and applications. John Wiley & Sons. Available from: https://books.google.com/books?hl=en&lr=&id=CVHx-Gp9jzUC&oi=fnd&pg=PT9&dq=2013.+Imbalanced+learning:+foundations,+algorithms,+and+applications&ots=2gTiFlCu7l&sig=GtLFELtOF6C1dIHQF79kJeSA6uk#v=onepage&q=2013.%20Imbalanced%20learning%3A%20foundations%2C%20algorithms%2C%20and%20applications&f=false [Accessed 19 July 2016]

Heller, S.R., McNaught, A., Pletnev, I., Stein, S. and Tchekhovskoi, D., 2015. InChI, the IUPAC international chemical identifier. Journal of cheminformatics, 7(1), p.1. Available from: http://jcheminf.springeropen.com/articles/10.1186/s13321-015-0068-4 [Accessed 25 September 2015]

Hert, J., Irwin, J.J., Laggner, C., Keiser, M.J. and Shoichet, B.K., 2009. Quantifying biogenic bias in screening libraries. *Nature chemical biology*, 5(7), pp.479-483. Available from: http://web.a.ebscohost.com/ehost/pdfviewer/pdfviewer?sid=c23737bc-21bf-4fcc-8c8e-50a379203aec%40sessionmgr4001&vid=1&hid=4212 [Accessed 17 June 2012]

Hoens, T.R., Qian, Q., Chawla, N.V. and Zhou, Z.H., 2012. Building decision trees for the multi-class imbalance problem. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 122-134). Springer Berlin Heidelberg. Available from: http://link.springer.com/chapter/10.1007%2F978-3-642-30217-6_11?LI=true [Accessed 18 February 2017]

Huang, S.H., Tung, C.W., Fülöp, F. and Li, J.H., 2015. Developing a QSAR model for hepatotoxicity screening of the active compounds in traditional Chinese medicines. *Food and Chemical Toxicology*, 78, pp.71-77. Available from: http://www.sciencedirect.com/science/article/pii/S0278691515000332 [Accessed January 10, 2016]

Hughes-Oliver, J.M., Brooks, A.D., Welch, W.J., Khaledi, M.G., Hawkins, D., Young, S.S., Patil, K., Howell, G.W., Ng, R.T. and Chu, M.T., 2011. ChemModLab: A web-based cheminformatics modeling laboratory. *In silico biology*, 11(1, 2), pp.61-81. Available from: http://www.cs.ubc.ca/~rng/psdepository/chemmod2009.pdf [Accessed March 12, 2012]

Imran, M., Mahmood, A.M. and Qyser, A.A.M., 2014, December. An empirical experimental evaluation on imbalanced data sets with varied imbalance ratio. In *Computer and Communications Technologies (ICCCT), 2014 International Conference on* (pp. 1-7). IEEE. Available from: http://ieeexplore.ieee.org/abstract/document/7066742/ [Accessed 22 February 2017]

Iwaniak, A., Minkiewicz, P., Darewicz, M., Protasiewicz, M. and Mogut, D., 2015. Chemometrics and cheminformatics in the analysis of biologically active peptides from food sources. *Journal of Functional Foods*, 16, pp.334-351. Available from: http://www.sciencedirect.com/science/article/pii/S175646461500211X [Accessed November 23, 2015]

James, C.A., Weininger, D. and Delany, J., 2000. Daylight theory manual-Daylight 4.71. Daylight Chemical Information Systems. Inc., Mission Viejo, CA. Available from: http://www.daylight.com/dayhtml/doc/theory/theory.finger.html [Accessed 17 January 2011]

John, G.H. & Langley, P., 1995. Estimating Continuous Distribution in Bayesian Classifiers. In *UAI'95 Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*. San Francisco, Calif.: Morgan Kaufmann, pp. 338–45.

Jolliffe, I.T. and Cadima, J., 2016. Principal component analysis: a review and recent developments. *Phil. Trans. R. Soc*. A, 374(2065), p.20150202. Available from: http://rsta.royalsocietypublishing.org/content/374/2065/20150202.full [Accessed 27 May 2016]

Jorgensen, W.L., 2012. Challenges for Academic Drug Discovery. *Angewandte Chemie (International ed. in English)*, 51(47), pp.11680–4. Available from: http://www.ncbi.nlm.nih.gov/pubmed/23097176 [Accessed March 5, 2013].

Kahng, A., 2012. Predicting the future of information technology and society [The Road Ahead]. *Design & Test of Computers, IEEE*, 29(6), pp.101-102. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6497099 [Accessed 29 May 2015]

Kato, R. et al., 2005. Novel Strategy for Protein Exploration: High-Throughput Screening Assisted with Fuzzy Neural Network. *Journal of molecular biology*, 351(3), pp.683–92. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16019025 [Accessed March 5, 2013]

Kavitha, A.P., Jaleel, U.A., Mujeeb, V.A. and Muraleedharan, K., 2016. Performance of knowledge-based biological models in higher dimensional chemical space. *Chemometrics and Intelligent Laboratory Systems*, 153, pp.58-66. Available from: http://www.sciencedirect.com/science/article/pii/S0169743916300314 [Accessed 26 June 2016]

Kazius, J., McGuire, R. and Bursi, R., 2005. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of Medicinal Chemistry*, 48(1), pp.312-320. Available from: http://pubs.acs.org/doi/abs/10.1021/jm040835a [Accessed February 6, 2011]

Kearnes, S., McCloskey, K., Berndl, M., Pande, V. and Riley, P., 2016. Molecular Graph Convolutions: Moving Beyond Fingerprints. *arXiv preprint* arXiv:1603.00856. Available from: https://arxiv.org/pdf/1603.00856.pdf [Accessed 27 November 2015]

Keserü, G.M. & Makara, G.M., 2009. The Influence of Lead Discovery Strategies on the Properties of Drug Candidates. *Nature reviews. Drug discovery*, 8(3), pp.203–12. Available from: http://www.ncbi.nlm.nih.gov/pubmed/19247303 [Accessed March 5, 2013]

Kier, L.B. and Hall, L.H., 1992. An atom-centered index for drug QSAR models. *Advances in Drug Design*, 22, pp.1-38.

Kim, S., Thiessen, P.A., Bolton, E.E., Chen, J., Fu, G., Gindulyte, A., Han, L., He, J., He, S., Shoemaker, B.A. and Wang, J., 2015. PubChem substance and compound databases. *Nucleic acids research*, p.gkv951. Available from: http://nar.oxfordjournals.org/content/early/2015/09/22/nar.gkv951.long [Accessed December 14, 2015]

Kothandan, R., 2015. Handling class imbalance problem in miRNA dataset associated with cancer. *Bioinformation*, *11*(1), p.6. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4349932/ [Accessed 200 February 2017]

Kotsiantis, S.B., Zaharakis, I. and Pintelas, P., 2007. Supervised machine learning: A review of classification techniques. Available from: http://s3.amazonaws.com/academia.edu.documents/33229897/Supervised_Machine_Learning--A_Review_of_Classification_Techniques.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1491108626&Signature=PnbCGUtHefKo27vbRa%2B0isafyYA%3D&response-content-disposition=inline%3B%20filename%3DSupervised_Machine_Learning_A_Review_of.pdf [Accessed 19 February 2017]

Kouzani, A.Z. and Nasireding, G., 2009. Multilabel classification by bch code and random forests. *International journal of recent trends in engineering*, 2(1), pp.113-116. Available from: http://dro.deakin.edu.au/eserv/DU:30028670/kouzani-multilabelclassification-2009.pdf [Accessed 12 March 2014]

Kristensen, T.G., Nielsen, J. and Pedersen, C.N., 2010. A tree-based method for the rapid screening of chemical fingerprints. *Algorithms for Molecular Biology*, 5(1), p.1. Available from: http://almob.biomedcentral.com/articles/10.1186/1748-7188-5-9 [Accessed 19 May 2016]

Kubat, M. and Matwin, S., 1997. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML* (Vol. 97, pp. 179-186). Available from: http://sci2s.ugr.es/keel/pdf/algorithm/congreso/kubat97addressing.pdf [Accessed 21 February 2017]

Kubat, M., Holte, R.C. and Matwin, S., 1998. Machine learning for the detection of oil spills in satellite radar images. *Machine learning*, 30(2-3), pp.195-215. Available from: http://link.springer.com/article/10.1023/A:1007452223027#page-1 [Accessed January 18, 2016]

Kumar, A., Kantardzic, M. & Madden, S., 2006. Guest Editors' Introduction: Distributed Data Mining--Framework and Implementations. *IEEE Internet Computing*, 10(4), pp.15–17. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1704751 [Accessed March 5, 2013]

Kuncheva, L.I. and Whitaker, C.J., 2003. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, *51*(2), pp.181-207. Available from: http://link.springer.com/article/10.1023%2FA%3A1022859003006?LI=true [Accessed 23 February 2017]

Langham, J.J. and Jain, A.N., 2008. Accurate and interpretable computational modeling of chemical mutagenicity. *Journal of chemical information and modeling*, 48(9), pp.1833-1839. Available from: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2753474/pdf/nihms142964.pdf [Accessed October 17, 2014]

Laney, D., 2001. 3D data management: Controlling data volume, velocity and variety. META Group Research Note, 6, p.70.

Leach, A.R. & Gillet, V.J., 2007. *An Introduction To Chemoinformatics*, Dordrecht: Springer Netherlands. Available from: http://www.springerlink.com/index/10.1007/978-1-4020-6291-9 [Accessed March 5, 2013]

Lemfack, M.C., Nickel, J., Dunkel, M., Preissner, R. and Piechulla, B., 2014. mVOC: a database of microbial volatiles. *Nucleic acids research*, 42(D1), pp.D744-D748. Available from: http://nar.oxfordjournals.org/content/42/D1/D744.full [Accessed February 16, 2015]

Lengauer, T., Lemmen, C., Rarey, M. and Zimmermann, M., 2004. Novel technologies for virtual screening. *Drug discovery today*, *9*(1), pp.27-34. Available from: http://www.sciencedirect.com/science/article/pii/S1359644604029393 [Accessed 25 February 2017]

Lavecchia, A. and Di Giovanni, C., 2013. Virtual screening strategies in drug discovery: a critical review. *Current medicinal chemistry*, *20*(23), pp.2839-2860. Available from: http://www.ingentaconnect.com/content/ben/cmc/2013/00000020/00000023/art00001 [Accessed 19 February 2017]

Li, G.-B. et al., 2011. Discovery of novel mGluR1 antagonists: a multistep virtual screening approach based on an SVM model and a pharmacophore hypothesis significantly increases the hit rate and enrichment factor. *Bioorganic & medicinal chemistry letters*, 21(6), pp.1736–40. Available from: http://www.ncbi.nlm.nih.gov/pubmed/21316965 [Accessed March 12, 2013]

Li, K., Zhang, W., Lu, Q. and Fang, X., 2014, October. An improved SMOTE imbalanced data classification method based on support degree. In *Identification, Information and Knowledge in the Internet of Things (IIKI), 2014 International Conference on* (pp. 34-38). Available from: http://ieeexplore.ieee.org/abstract/document/7063993/ [Accessed 26 February 2017]

Li, Q., Wang, Y. and Bryant, S.H., 2009. A novel method for mining highly imbalanced high-throughput screening data in PubChem. *Bioinformatics*, *25*(24), pp.3310-3316. Available from: bioinformatics.oxfordjournals.org/content/25/24/3310.short [Accessed 24 February 2017]

Lin, W.J. and Chen, J.J., 2012. Class-imbalanced classifiers for high-dimensional data. *Briefings in bioinformatics*, p.bbs006. Available from: http://bib.oxfordjournals.org/content/early/2012/03/08/bib.bbs006.short [Accessed 27 June 2015]

Lionta, E., Spyrou, G., K Vassilatis, D. and Cournia, Z., 2014. Structure-based virtual screening for drug discovery: principles, applications and recent advances. *Current topics in medicinal chemistry*, *14*(16), pp.1923-1938. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4443793/ [Accessed 21 February 2017]

Liu, K., Feng, J. & Young, S.S., 2005. PowerMV: A Software Environment for Molecular Viewing, Descriptor Generation, Data Analysis and Hit Evaluation. *Journal of chemical information and modeling*, 45(2), pp.515–22. Available from: http://www.ncbi.nlm.nih.gov/pubmed/15807517 [Accessed March 5, 2013]

Liu, M., Wang, M., Wang, J. and Li, D., 2013. Comparison of random forest, support vector machine and back propagation neural network for electronic tongue data classification: Application to the recognition of orange beverage and Chinese vinegar. *Sensors and Actuators B: Chemical*, *177*, pp.970-980.IEEE. Available from: http://www.sciencedirect.com/science/article/pii/S0925400512012671 [Accessed 23 February 2017]

Longadge, R. and Dongre, S., 2013. Class Imbalance Problem in Data Mining *Review*. *arXiv preprint* arXiv:1305.1707. Available from: https://arxiv.org/ftp/arxiv/papers/1305/1305.1707.pdf [Accessed February 11, 2015]

López, V., Fernández, A., García, S., Palade, V. and Herrera, F., 2013. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250, pp.113-141. Available from: http://www.sciencedirect.com/science/article/pii/S0020025513005124 [Accessed 9 June 2014]

Luengo, J. et al., 2010. Addressing data complexity for imbalanced data sets: analysis of SMOTE-based oversampling and evolutionary undersampling. *Soft Computing*, 15(10), pp.1909–1936. Available from: http://link.springer.com/10.1007/s00500-010-0625-8 [Accessed May 3, 2013]

Maggiora, G.M. and Shanmugasundaram, V., 2011. Molecular similarity measures. *Chemoinformatics and computational chemical biology*, pp.77-84.

Maji, S., Berg, A.C. and Malik, J., 2013. Efficient classification for additive kernel SVMs. *IEEE transactions on pattern analysis and machine intelligence*, *35*(1), pp.66-77. Available from: http://ieeexplore.ieee.org/document/6165310/ [Accessed 26 June 2014]

Maldonado, S. and López, J., 2014. Imbalanced data classification using second-order cone programming support vector machines. *Pattern Recognition*, 47(5), pp.2070-2079. Available from: http://www.sciencedirect.com/science/article/pii/S0031320313005074 [Accessed 2 May 2015]

Maldonado, S., Weber, R. and Famili, F., 2014. Feature selection for high-dimensional class-imbalanced data sets using Support Vector Machines. *Information Sciences*, 286, pp.228-246. Available from: http://www.sciencedirect.com/science/article/pii/S0020025514007154 [Accessed 3 February 2015]

Maratea, A., Petrosino, A. and Manzo, M., 2014. Adjusted F-measure and kernel scaling for imbalanced data learning. *Information Sciences*, 257, pp.331-341. Available from: http://www.sciencedirect.com/science/article/pii/S0020025513003137 [Accessed 14 September 2015]

Martis, E., Radhakrishnan, R. & Badve, R., 2011. High-Throughput Screening: The Hits and Leads of Drug Discovery- An Overview. *Journal of Applied Pharmaceutical Science*, 1(1), pp.02–10.

Mascaro, J., Asner, G.P., Knapp, D.E., Kennedy-Bowdoin, T., Martin, R.E., Anderson, C., Higgins, M. and Chadwick, K.D., 2014. A tale of two "forests": Random Forest machine learning aids tropical forest carbon mapping. *PloS one*, *9*(1), p.e85993. Available from: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0085993 [Accessed 29 June 2015]

Maunz, A., Gütlein, M., Rautenberg, M., Vorgrimmler, D., Gebele, D. and Helma, C., 2013. Lazar: a modular predictive toxicology framework. *Frontiers in pharmacology*, 4, p.38. Available from: http://journal.frontiersin.org/article/10.3389/fphar.2013.00038/full [Accessed 24 May 2015]

May, R.J., Maier, H.R. and Dandy, G.C., 2010. Data splitting for artificial neural networks using SOM-based stratified sampling. *Neural Networks*, *23*(2), pp.283-294. Available from: http://www.sciencedirect.com/science/article/pii/S0893608009002949 [Accessed 24 February 2017]

Mayr, L.M. & Bojanic, D., 2009. Novel Trends in High-Throughput Screening. *Current opinion in pharmacology*, 9(5), pp.580–8. Available from: http://www.ncbi.nlm.nih.gov/pubmed/19775937 [Accessed February 28, 2013]

Mazurowski, M.A., Habas, P.A., Zurada, J.M., Lo, J.Y., Baker, J.A. and Tourassi, G.D., 2008. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural networks*, *21*(2), pp.427-436. Available from: http://www.sciencedirect.com/science/article/pii/S0893608007002407 [Accessed 28 February 2017]

Medina-Franco, J.L., 2012. Scanning structure–activity relationships with structure–activity similarity and related maps: from consensus activity cliffs to selectivity switches. *Journal of chemical information and modeling*, 52(10), pp.2485-2493. Available from: http://pubs.acs.org/doi/abs/10.1021/ci300362x [Accessed January 4 2015]

Monev, V., 2004. Introduction to Similarity Searching in Chemistry. *MATCH-COMMUNICATIONS IN MATHEMATICAL AND IN COMPUTER CHEMISTRY*, 51, pp.7–38.

Muegge, I. & Oloff, S., 2006. Advances in Virtual Screening. *Drug Discovery Today: Technologies*, 3(4), pp.405–411. Available from: http://linkinghub.elsevier.com/retrieve/pii/S1740674906000758 [Accessed March 5, 2013].

Murphree, D., Ngufor, C., Upadhyaya, S., Madde, N., Clifford, L., Kor, D.J. and Pathak, J., 2015, August. Ensemble learning approaches to predicting complications of blood transfusion. In *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE* (pp. 7222-7225). IEEE. Available from: http://ieeexplore.ieee.org/abstract/document/7320058/ [Accessed 23 February 2017]

Murray-Rust, P., Rzepa, H.S. and Wright, M., 2001. Development of chemical markup language (CML) as a system for handling complex chemical content. *New journal of chemistry*, 25(4), pp.618-634. Available from: http://pubs.rsc.org/en/content/articlehtml/2001/nj/b008780g [Accessed February 11, 2013]

Nag, K. and Pal, N.R., 2016. A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification. *IEEE transactions on cybernetics*, *46*(2), pp.499-510. Available from: http://ieeexplore.ieee.org/abstract/document/7055929/ [Accessed 22 February 2017]

Napierala, K. and Stefanowski, J., 2012, March. Identification of different types of minority class examples in imbalanced data. In *International Conference on Hybrid Artificial Intelligence Systems* (pp. 139-150). Springer Berlin Heidelberg. Available from: http://link.springer.com/chapter/10.1007%2F978-3-642-28931-6_14?LI=true [Accessed 19 February 2017]

Napierala, K. and Stefanowski, J., 2016. Types of minority class examples and their influence on learning classifiers from imbalanced data. *Journal of Intelligent Information Systems*, *46*(3), pp.563-597. Available from: http://link.springer.com/article/10.1007/s10844-015-0368-1 [Accessed 23 February 2017]

Naqaash, M. et al., 2010. Grid Computing Used For Next Generation High Speed Processing Technology. *International Journal on Computer Science & Engineering*, 1(5), pp.1926–1933.

National Center for Biotechnology Information. PubChem BioAssay Database; AID=362, Available from: https://pubchem.ncbi.nlm.nih.gov/bioassay/362. [Accessed October 17, 2010]

National Center for Biotechnology Information. PubChem BioAssay Database; AID=456, Available from: https://pubchem.ncbi.nlm.nih.gov/bioassay/456. [Accessed October 17, 2010]

Ng, W.W., Zeng, G., Zhang, J., Yeung, D.S. and Pedrycz, W., 2016. Dual autoencoders features for imbalance classification problem. *Pattern Recognition*, *60*, pp.875-889. Available from: www.sciencedirect.com/science/article/pii/S0031320316301303 [Accessed 23 February 2017]

O'Boyle, N.M., Banck, M., James, C.A., Morley, C., Vandermeersch, T. and Hutchison, G.R., 2011. Open Babel: An open chemical toolbox. *J Cheminf*, 3, p.33. Available from: http://download.springer.com/static/pdf/489/ [Accessed December 14, 2011]

O'Boyle, N.M., Guha, R., Willighagen, E.L., Adams, S.E., Alvarsson, J., Bradley, J.C., Filippov, I.V., Hanson, R.M., Hanwell, M.D., Hutchison, G.R. and James, C.A., 2011. Open Data, Open Source and Open Standards in chemistry: The Blue Obelisk five years on. *J. Cheminformatics*, 3, p.37. Available from: http://download.springer.com/static/pdf/493/art%253A10.1186%252F1758-2946-3-37.pdf [Accessed December 14, 2011]

Oreski, G. and Oreski, S., 2014, January. An experimental comparison of classification algorithm performances for highly imbalanced datasets. In *Central European Conference on Information and Intelligent Systems* (p. 4). Faculty of Organization and Informatics Varazdin. Available from: http://search.proquest.com/openview/d6087985191c5bab6bcba81c708a07bd/1?pq-origsite=gscholar&cbl=1986354 [Accessed 25 February 2017]

Orriols-Puig, A. and Bernadó-Mansilla, E., 2009. Evolutionary rule-based systems for imbalanced data sets. *Soft Computing*, 13(3), pp.213-225. Available from: http://link.springer.com/article/10.1007%2Fs00500-008-0319-7 [Accessed 22 May 2013]

Panagopoulos, O.P., Pappu, V., Xanthopoulos, P. and Pardalos, P.M., 2016. Constrained subspace classifier for high dimensional datasets. *Omega*, 59, pp.40-46. Available from: http://www.sciencedirect.com/science/article/pii/S0305048315001188 [Accessed 25 July 2016]

Pears, R., Finlay, J. & Connor, A.M. 2014 "Synthetic Minority Over-sampling TEchnique (SMOTE) for Predicting Software Build Outcomes", *Proceedings of the Twenty-Sixth International Conference on Software Engineering and Knowledge Engineering (SEKE 2014).* Available from: https://arxiv.org/abs/1407.2330 [Accessed 19 February 2017]

Platt, J., 1998. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In B. Schölkopf, C. J. Burges, & A. J. Smola, eds. *Advances in Kernel Methods*. MIT Press, pp. 41–64. Available from: http://www.cs.utsa.edu/~bylander/cs6243/smo-book.pdf [Accessed 24 June 2012]

Platt, J.C., 1999. 12 fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods*, pp.185-208. Available from: http://www.msr-waypoint.com/pubs/69644/tr-98-14.pdf [Accessed 16 November 2014]

Plewczynski, D., Spieser, S.A.H. & Koch, U., 2006. Assessing Different Classification Methods for Virtual Screening. *Journal of chemical information and modeling*, 46(3), pp.1098–1106. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16711730 [Accessed March 8, 2013].

Polanski, J., 2009. Chemoinformatics. In S. D. Brown, R. Tauler, & B. Walczak, eds. *Comprehensive Chemometrics: Chemical and Biochemical Data Analysis*. Elsevier, pp. 459–503.

Prachuabsupakij, W. and Soonthornphisaj, N., 2012. A new classification for multiclass imbalanced datasets based on clustering approach. In *The 26th Annual Conference of the Japanese Society for Artificial Intelligence*. Available from: https://www.researchgate.net/profile/Wanthanee_Prachuabsupakij/publication/268057630_A_New_Classification_for_Multiclass_Imbalanced_Datasets_Based_on_Clustering_Approach/links/546e9b710cf2b5fc17607a5a.pdf [Accessed 22 February 2017]

Prati, R.C., Batista, G.E. and Monard, M.C., 2004, April. Class imbalances versus class overlapping: an analysis of a learning system behavior. In *Mexican international conference on artificial intelligence* (pp. 312-321). Springer Berlin Heidelberg. Available from: http://link.springer.com/chapter/10.1007/978-3-540-24694-7_32#page-1 [Accessed 2 March 2013]

Prati, R.C., Batista, G.E. and Silva, D.F., 2015. Class imbalance revisited: a new experimental setup to assess the performance of treatment methods. *Knowledge and Information Systems*, *45*(1), pp.247-270. Available from: http://link.springer.com/article/10.1007/s10115-014-0794-3 [Accessed 18 February 2017]

Qiao, X. and Zhang, L., 2015. Flexible high-dimensional classification machines and their asymptotic properties. *Journal of Machine Learning Research*, 16, pp.1547-1572. Available from: http://www.jmlr.org/papers/volume16/qiao15a/qiao15a.pdf [Accessed 29 June 2016]

Quinlan, J.R., 2014. *C4. 5: programs for machine learning*. Elsevier

Quinlan, J.R., 1993. *C4. 5: programs for machine learning*. Morgan Kaufmann San Mateo

Radivojac, P. et al., 2004. Classification and knowledge discovery in protein databases. *Journal of biomedical informatics*, 37(4), pp.224–39. Available from: http://www.ncbi.nlm.nih.gov/pubmed/15465476 [Accessed April 2, 2013].

Rahman, A. and Fairhurst, M., 2000. Decision combination of multiple classifiers for pattern classification: hybridisation of majority voting and divide and conquer techniques. In *Applications of Computer Vision, 2000, Fifth IEEE Workshop on.* (pp. 58-63). IEEE. Available from: http://ieeexplore.ieee.org/abstract/document/895403/ [Accessed 23 February 2017]

Ramezankhani, A., Pournik, O., Shahrabi, J., Azizi, F., Hadaegh, F. and Khalili, D., 2014. The impact of oversampling with SMOTE on the performance of 3 classifiers in prediction of type 2 diabetes. *Medical Decision Making*, p.0272989X14560647. Available from: http://mdm.sagepub.com/content/36/1/137.full.pdf+html [Accessed 19 November 2014]

Ramyachitra, D. and Manikandan, P., 2014. Imbalanced Dataset Classification and Solutions: A Review. *International Journal of Computing and Business Research*, 5(4). Available from: http://www.researchmanuscripts.com/July2014/2.pdf [Accessed January 15, 2016]

Reddy, A.S., Pati, S.P., Kumar, P.P., Pradeep, H.N. and Sastry, G.N., 2007. Virtual screening in drug discovery-a computational perspective. *Current Protein and Peptide Science*, *8*(4), pp.329-351. Available from: http://www.ingentaconnect.com/content/ben/cpps/2007/00000008/00000004/art00003 [Accessed 24 February 2017]

Ringsted, T. and Todeschini, R., 2012. Marie Curie Initial Training Network Environmental Chemoinformatics (ECO). Available from: http://www.ecoitn.eu/sites/eco-itn.eu/files/reports/Report_Tine_Ringsted.pdf [Accessed 24 July 2014]

Rivera-Borroto, O., Garcia-de la Vega, J., Marrero-Ponce, Y. and Grau, R., Relational agreement measures for similarity searching of Cheminformatic data sets. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7096989 [Accessed March 11, 2016]

Sabet, M.J., Baratian, A., Habibi, M. and Hadizadeh, F., 2015. Iran Virtual Screening (IranVScreen): An Integrated Virtual Screening Interface. *Journal of Archives in Military Medicine*, *3*(3). Available from: http://jammonline.com/?page=article&article_id=30745 [Accessed 27 February 2017]

Sáez, J.A., Luengo, J., Stefanowski, J. and Herrera, F., 2014, September. Managing borderline and noisy examples in imbalanced classification by combining SMOTE with ensemble filtering. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 61-68). Springer International Publishing. Available from: http://link.springer.com/chapter/10.1007/978-3-319-10840-7_8 [Accessed 19 February 2017]

Sáez, J.A., Luengo, J., Stefanowski, J. and Herrera, F., 2015. SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, 291, pp.184-203. Available from: http://www.sciencedirect.com/science/article/pii/S0020025514008561 [Accessed 14 January 2016]

Sáez, J.A., Krawczyk, B. and Woźniak, M., 2016. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition*, *57*, pp.164-178. Available from: http://www.sciencedirect.com/science/article/pii/S0031320316001072

Salama, M.A., Fouad, M.M.M., El-Bendary, N. and Hassanien, A.E.O., 2014. Mutagenicity Analysis Based on Rough Set Theory and Formal Concept Analysis. *In Recent Advances in Intelligent Informatics* (pp. 265-273). Springer International Publishing. Available from: http://link.springer.com/chapter/10.1007/978-3-319-01778-5_27 [Accessed 25 April 2016]

Samaddar, A., Goswami, T., Ghosh, S. and Pal, S., 2015, June. An algorithm to input and store wider classes of chemical reactions for mining chemical graphs. *In Advance Computing Conference (IACC), 2015 IEEE International* (pp. 1082-1086). IEEE. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7154871 [Accessed December 19, 2015]

Sawada, R., Kotera, M. and Yamanishi, Y., 2014. Benchmarking a Wide Range of Chemical Descriptors for Drug-Target Interaction Prediction Using a Chemogenomic Approach. *Molecular Informatics*, 33(11-12), pp.719-731. Available from: http://onlinelibrary.wiley.com/doi/10.1002/minf.201400066/pdf [Accessed August 14, 2015]

Seal, A., Passi, A., Jaleel, U.A. and Wild, D.J., 2012. In-silico predictive mutagenicity model generation using supervised learning approaches. *Journal of cheminformatics*, 4(1), p.1. Available from: https://jcheminf.springeropen.com/articles/10.1186/1758-2946-4-10 [Accessed 27 June 2014]

Seiffert, C., Khoshgoftaar, T.M., Van Hulse, J. and Folleco, A., 2014. An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences*, 259, pp.571-595. Available from: http://www.sciencedirect.com/science/article/pii/S0020025511000065 [Accessed 27 July 2015]

Schierz, A.C., 2009. Virtual screening of bioassay data. *Journal of Cheminformatics*, 1, p.21. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2820499&tool=pmcentrez&rendertype=abstract [Accessed March 5, 2013]

Scornet, E., Biau, G. and Vert, J.P., 2015. Consistency of random forests. *The Annals of Statistics*, *43*(4), pp.1716-1741. Available from: http://projecteuclid.org/euclid.aos/1434546220 [Accessed 17 February 2017]

Schölkopf, B. and Smola, A.J., 2002. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press. Available from: https://books.google.co.uk/books?hl=en&lr=&id=y8ORL3DWt4sC&oi=fnd&pg=PR13&dq=Scholkopf+%26+Smola,+2002&ots=bKBR6AUbDH&sig=wutWiHzO62psNUhNzqgjrtZInbU#v=onepage&q=Scholkopf%20%26%20Smola%2C%202002&f=false [Accessed 23 May 2016]

Shi, X., Xu, G., Shen, F. and Zhao, J., 2015, July. Solving the data imbalance problem of P300 detection via Random Under-Sampling Bagging SVMs. *In Neural Networks (IJCNN)*, 2015 International Joint Conference on (pp. 1-5). IEEE. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7280834 [Accessed 26 October 2015]

Shlens, J., 2014. A tutorial on principal component analysis. arXiv preprint arXiv:1404.1100. Available from: http://arxiv.org/pdf/1404.1100.pdf [Accessed 22 February 2015]

Shoichet, B.K., 2004. Virtual screening of chemical libraries. *Nature*, *432*(7019), pp.862-865. Available from: http://www.nature.com/nature/journal/v432/n7019/abs/nature03197.html [Accessed 24 February 2017]

Schomburg, K.T., Wetzer, L. and Rarey, M., 2013. Interactive design of generic chemical patterns. *Drug discovery today*, *18*(13), pp.651-658. Available from:http://www.sciencedirect.com/science/article/pii/S1359644613000366 [Accessed March 14, 2013]

Sink, R. et al., 2010. False Positives in the Early Stages of Drug Discovery. *Current Medicinal Chemistry*, 17(34), pp.4231–4255. Available from: http://www.eurekaselect.com/openurl/content.php?genre=article&issn=0929-8673&volume=17&issue=34&spage=4231 [Accessed March 8, 2013].

Speck-Planche, A., V Kleandrova, V., Luan, F. and Cordeiro, N.D., 2012. Chemoinformatics in multi-target drug discovery for anti-cancer therapy: in silico design of potent and versatile anti-brain tumor agents. *Anti-Cancer Agents in Medicinal Chemistry (Formerly Current Medicinal Chemistry-Anti-Cancer Agents)*, 12(6), p.678. Available from: https://www.ncbi.nlm.nih.gov/pubmed/22043995 [Accessed June 21, 2014]

Speck-Planche, A., Kleandrova, V.V. and Cordeiro, M.N.D.S., 2013. Chemoinformatics for rational discovery of safe antibacterial drugs: Simultaneous predictions of biological activity against streptococci and toxicological profiles in laboratory animals. *Bioorganic & medicinal chemistry, 21(10)*, pp.2727-2732. Available from: https://www.ncbi.nlm.nih.gov/pubmed/23582445 [Accessed February 27, 2015]

Spjuth, O., Willighagen, E.L., Guha, R., Eklund, M. and Wikberg, J.E., 2010. Towards interoperable and reproducible QSAR analyses: Exchange of datasets. *Journal of cheminformatics*, 2(1), pp.1-7. Available from: http://link.springer.com/article/10.1186/1758-2946-2-5 [Accessed 23 October 2011]

Sruthi, V., Kesh, N.T., Priyanka, R. and Jacob, S.G., 2015, October. Binary categorization of DNA data with unbalanced class distribution for prediction of hepatocellular carcinoma. In *Applied and Theoretical Computing and Communication Technology (iCATccT), 2015 International Conference on* (pp. 490-494). IEEE. Available from: http://ieeexplore.ieee.org/abstract/document/7456934/ [Accessed 17 February 2017]

Steinbeck, C., Han, Y., Kuhn, S., Horlacher, O., Luttmann, E. and Willighagen, E., 2003. The Chemistry Development Kit (CDK): An open-source Java library for chemo-and bioinformatics. *Journal of chemical information and computer sciences*, 43(2), pp.493-500. Available from: http://pubs.acs.org/doi/pdf/10.1021/ci025584y [Accessed September 25, 2011]

Steinbeck, C., Hoppe, C., Kuhn, S., Floris, M., Guha, R. and Willighagen, E.L., 2006. Recent developments of the chemistry development kit (CDK)-an open-source java library for chemo-and bioinformatics. *Current pharmaceutical design*, 12(17), pp.2111-2120. Available from: http://www.ingentaconnect.com/content/ben/cpd/2006/00000012/00000017/art00005 [Accessed December 12 2014]

Stephan, C and Gilbertson, S.R, 2009. Small Molecule Screens.[PowerPoint Presentation]. Available from: cohesion.rice.edu/centersandinst/gcc/.../CSSmMolec_20090623Final.ppt [Accessed May 13, 2015]

Stobaugh, R.E., 1985. Chemical substructure searching. *Journal of Chemical Information and Computer Sciences*, 25(3), pp.271-275. Available from: http://pubs.acs.org/doi/pdf/10.1021/ci00047a025 [Accessed September 9, 2013]

Stumpfe, D. et al., 2010. Targeting multifunctional proteins by virtual screening: structurally diverse cytohesin inhibitors with differentiated biological functions. *ACS chemical biology*, 5(9), pp.839–49. Available from: http://www.ncbi.nlm.nih.gov/pubmed/20614894 [Accessed May 16, 2013]

Sun, Y., Wong, A.K. and Kamel, M.S., 2009. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, *23*(04), pp.687-719. Available from: http://www.worldscientific.com/doi/abs/10.1142/S0218001409007326 Accessed 24 February 2017]

Symyx, 2010. *CTfile Formats*, Symyx software: San Ramon.

Tan, A.C. and Gilbert, D., 2003. Ensemble machine learning on gene expression data for cancer classification. Available from: sites.google.com/site/aikchoon/ABI-2-3-suppl-Tan.pdf [Accessed 18 February 2017]

Tang, Y. et al., 2009. SVMs modeling for highly imbalanced classification. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 39(1), pp.281–8. Available from: http://www.ncbi.nlm.nih.gov/pubmed/19068445 [Accessed May 15, 2013]

TaşCı, Ş. and Güngör, T., 2013. Comparison of text feature selection policies and using an adaptive framework. *Expert Systems with Applications*, *40*(12), pp.4871-4886. Available from: http://www.sciencedirect.com/science/article/pii/S0957417413001358 [Accessed 26 February 2017]

Thorne, N., Auld, D.S. & Inglese, J., 2010. Apparent Activity in High-Throughput Screening: Origins of Compound-Dependent Assay Interference. *Current opinion in chemical biology*, 14(3), pp.315–24. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2878863&tool=pmcentrez&rendertype=abstract [Accessed March 5, 2013]

Todeschini, R. and Consonni, V., 2009. Molecular Descriptors for Chemoinformatics, *Volume 41 (2 Volume Set) (Vol. 41). John Wiley & Sons.* Available from: http://lori.academicdirect.org/didactic/attends/06_2008_June_Strasbourg_University_Louis_Pasteur/RTodeschini.pdf [Accessed March 15, 2015]

Tomal, J.H., Welch, W.J. and Zamar, R.H., 2015. Ensembling classification models based on phalanxes of variables with applications in drug discovery. *The Annals of Applied Statistics*, 9(1), pp.69-93. Available from: http://projecteuclid.org/euclid.aoas/1430226085 [Accessed 28 July 2015]

Tomar, D. and Agarwal, S., 2015. An effective weighted multi-class least squares twin support vector machine for imbalanced data classification. *International Journal of Computational Intelligence Systems*, *8*(4), pp.761-778. Available from: http://www.tandfonline.com/doi/abs/10.1080/18756891.2015.1061395 [Accessed 25 February 2017]

Tomašev, N. and Mladenić, D., 2014. Hubness-aware shared neighbor distances for high-dimensional k-nearest neighbor classification. *Knowledge and information systems*, 39(1), pp.89-122. Available from: http://link.springer.com/article/10.1007/s10115-012-0607-5 [Accessed 14 September 2015]

Tomašev, N. and Buza, K., 2015. Hubness-aware kNN classification of high-dimensional data in presence of label noise. *Neurocomputing*, 160, pp.157-172. Available from: http://www.sciencedirect.com/science/article/pii/S0925231215001228 [Accessed 22 April 2016]

Toropov, A.A. and Benfenati, E., 2007. SMILES as an alternative to the graph in QSAR modelling of bee toxicity. *Computational biology and chemistry*, 31(1), pp.57-60. Available from: http://www.sciencedirect.com/science/article/pii/S1476927107000047 [Accessed November 22, 2014]

Ultra, C., 2001. 6.0 and Chem3D Ultra. Cambridge Soft Corporation, Cambridge, USA.

Venkatesan, R. and Er, M.J., 2016. A novel progressive learning technique for multi-class classification. *Neurocomputing*, *207*, pp.310-321. Available from: http://www.sciencedirect.com/science/article/pii/S0925231216303137 [Accessed 19 February 2017]

Verbiest, N., Ramentol, E., Cornelis, C. and Herrera, F., 2014. Preprocessing noisy imbalanced datasets using SMOTE enhanced with fuzzy rough prototype selection. *Applied Soft Computing*, 22, pp.511-517. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.642.8393&rep=rep1&type=pdf [Accessed 24 July 2015]

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y. and Manzagol, P.A., 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, *11*(Dec), pp.3371-3408. Available from: http://www.jmlr.org/papers/v11/vincent10a.html [Accessed 26 February 2017]

Visa, S. and Ralescu, A., 2005, April. Issues in mining imbalanced data sets-a review paper. *In Proceedings of the sixteen midwest artificial intelligence and cognitive science conference* (Vol. 2005, pp. 67-73). sn. Available from: https://www.researchgate.net/profile/Anca_Ralescu/publication/228386653_Issues_in_mining_imbalanced_data_sets_-_A_review_paper/links/02e7e51cdc0c87d98f000000.pdf [Accessed 14 October 2014]

Vyas, V., Jain, A., Jain, A. and Gupta, A., 2008. Virtual screening: a fast tool for drug design. *Sci Pharm*, *76*(3), pp.333-60. Available from: http://www.scipharm.at/default.asp?id=294&lid=-a [Accessed 17 February 2017]

Wald, R., Khoshgoftaar, T., Dittman, D.J. and Napolitano, A., 2013. Random forest with 200 selected features: An optimal model for bioinformatics research. In *Machine Learning and Applications (ICMLA), 2013 12th International Conference on* (Vol. 1, pp. 154-160). IEEE. Available from: http://ieeexplore.ieee.org/abstract/document/6784604/ [Accessed 20 February 2017]

Wang, B. & Japkowicz, N., 2004. Imbalanced Data Set Learning with Synthetic Samples. In *Proc. IRIS Machine Learning Workshop*.

Wang, Juanjuan et al., 2006. Classification of Imbalanced Data by Using the SMOTE Algorithm and Locally Linear Embedding. In *2006 8th international Conference on Signal Processing*. IEEE. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4129201 [Accessed April 17, 2013]

Wang, Y. et al., 2009. PubChem: a public information system for analyzing bioactivities of small molecules. *Nucleic acids research*, 37(Web Server issue), pp.W623–33. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2703903&tool=pmcentrez&rendertype=abstract [Accessed February 27, 2013]

Wang, J., 2012. Principal component analysis. In Geometric Structure of High-Dimensional Data and Dimensionality Reduction (pp. 95-114). *Springer Berlin Heidelberg*. Available from: http://www.johnverostek.com/wp-content/uploads/2015/06/Excel_CH8_PCA.pdf [Accessed 23 March 2016]

Wang, J., You, J., Li, Q. and Xu, Y., 2012. Extract minimum positive and maximum negative features for imbalanced binary classification. *Pattern Recognition*, 45(3), pp.1136-1145. Available from: http://www.sciencedirect.com/science/article/pii/S0031320311003827 [Accessed 29 October 2014]

Wang, S. and Yao, X., 2012. Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *42*(4), pp.1119-1130. Available from: http://ieeexplore.ieee.org/abstract/document/6170916/ [Accessed 19 February 2017]

Wang, Y., Suzek, T., Zhang, J., Wang, J., He, S., Cheng, T., Shoemaker, B.A., Gindulyte, A. and Bryant, S.H., 2013. PubChem bioassay: 2014 update. *Nucleic acids research*, p.gkt978. Available from: http://nar.oxfordjournals.org/content/early/2013/11/05/nar.gkt978.full [Accessed December 14, 2015]

Warr, W.A., 2011. Representation of chemical structures. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(4), pp.557-579. Available from: http://onlinelibrary.wiley.com/doi/10.1002/wcms.36/pdf [Accessed August 11, 2014]

Wei, H., Zhang, R., Wang, C., Zheng, H., Li, A., Chou, K.C. and Wei, D.Q., 2007. Molecular insights of SAH enzyme catalysis and implication for inhibitor design. *Journal of theoretical biology*, 244(4), pp.692-702. Available from: http://pubs.acs.org/doi/abs/10.1021/ci400391s [Accessed 28 November 2012]

Wei, Q. and Dunbrack Jr, R.L., 2013. The role of balanced training and testing data sets for binary classifiers in bioinformatics. *PloS one*, *8*(7), p.e67863. Available from: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0067863 [Accessed 23 February 2017]

Weininger, D., 1988. SMILES, A Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *Journal of Chemical Information and Modeling*, 28(1), pp.31–36. Available from: http://pubs.acs.org/cgi-bin/doilookup/?10.1021/ci00057a005 [Accessed March 5, 2013]

Weiss, G.M., 2004. Mining With Rarity: A Unifying Framework. *ACM SIGKDD Explorations Newsletter*, 6(1), pp.7–19. Available from: http://portal.acm.org/citation.cfm?doid=1007730.1007734 [Accessed March 5, 2013]

Weiss, G.M. and Provost, F., 2003. Learning when training data are costly: the effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, *19*, pp.315-354

Willett, P., Barnard, J.M. & Downs, G.M., 1998. Chemical Similarity Searching. *Journal of Chemical Information and Modeling*, 38(6), pp.983–996. Available from: http://pubs.acs.org/cgi-bin/doilookup/?10.1021/ci9800211 [Accessed March 8, 2013]

Willett, P., 2006. Similarity-Based Virtual Screening Using 2D Fingerprints. *Drug discovery today*, 11(23-24), pp.1046–53. Available from: http://www.ncbi.nlm.nih.gov/pubmed/17129822 [Accessed March 2, 2013].

Willett, P., 2009. Similarity methods in Chemoinformatics. *Annual review of information science and technology*, 43(1), pp.1-117. Available from: http://eprints.whiterose.ac.uk/77605/8/WRRO_77605.pdf [Accessed September 29, 2014]

Willett, P., 2011. Similarity searching using 2D structural fingerprints. *Chemoinformatics and computational chemical biology*, pp.133-158. Available from: http://eprints.whiterose.ac.uk/76258/ [Accessed October 11, 2013]

Willett, P., 2014. The calculation of molecular structural similarity: principles and practice. *Molecular Informatics*, 33(6-7), pp.403-413. Available from: http://onlinelibrary.wiley.com/doi/10.1002/minf.201400024/pdf [Accessed August 11, 2015]

Wu, G., Kechavarzi, C., Li, X., Wu, S., Pollard, S.J., Sui, H. and Coulon, F., 2013. Machine learning models for predicting PAHs bioavailability in compost amended soils. *Chemical engineering journal*, *223*, pp.747-754. Available from: http://www.sciencedirect.com/science/article/pii/S1385894713003161 [Accessed 4 September 2015]

Xanthopoulos, P., Pardalos, P.M. and Trafalis, T.B., 2013. Principal component analysis. *In Robust data mining* (pp. 21-26). Springer New York. Available from: http://link.springer.com/chapter/10.1007%2F978-1-4419-9878-1_3 [Accessed 19 January 2015]

Xiao, Jiamin et al., 2011. Identification of microRNA precursors based on random forest with network-level representation method of stem-loop structure. *BMC bioinformatics*, 12, p.165. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3118167&tool=pmcentrez&rendertype=abstract [Accessed May 18, 2013]

Yang, Y. and Webb, G.I., 2003. Weighted proportional k-interval discretization for naive-bayes classifiers. In Advances in knowledge discovery and data mining (pp. 501-512). *Springer Berlin Heidelberg*. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.138.6299&rep=rep1&type=pdf [Accessed 24 February 2014]

Yang, Q. and Wu, X., 2006. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(04), pp.597-604. Available from: http://www.worldscientific.com/doi/abs/10.1142/S0219622006002258 [Accessed February 11, 2016]

Yang, X., Yu, Q., He, L. and Guo, T., 2013. The one-against-all partition based binary tree support vector machine algorithms for multi-class classification. *Neurocomputing*, *113*, pp.1-7. Available from: http://www.sciencedirect.com/science/article/pii/S0925231213002282 [Accessed 17 February 2017]

Yap, C.W., 2011. PaDEL-Descriptor: An Open Source Software to Calculate Molecular Descriptors and Fingerprints. *Journal of Computational Chemistry*, 32(7), pp.1466–74. Available from: http://www.ncbi.nlm.nih.gov/pubmed/21425294 [Accessed February 28, 2013]

Yasin, W., Ibrahim, H., Udzir, N.I. and Hamid, N.A.W.A., 2014, December. Intelligent Cooperative Least Recently Used Web Caching Policy based on J48 Classifier. *In Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services* (pp. 262-269). ACM. Available from: http://dl.acm.org/citation.cfm?id=2684299 [Accessed 14 July 2015]

Yin, H. and Gai, K., 2015, August. An empirical study on preprocessing high-dimensional class-imbalanced data for classification. *In High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on (pp. 1314-1319). IEEE*. Available from: http://ieeexplore.ieee.org/document/7336349/?arnumber=7336349&tag=1 [Accessed 4 June 2016]

Youssef, A.M., Pourghasemi, H.R., Pourtaghi, Z.S. and Al-Katheeri, M.M., 2015. Landslide susceptibility mapping using random forest, boosted regression tree, classification and regression tree, and general linear models and comparison of their performance at Wadi Tayyah Basin, Asir Region, Saudi Arabia. *Landslides*, pp.1-18. Available from: http://link.springer.com/article/10.1007/s10346-015-0614-1 [Accessed 12 June 2016]

Yu, H. and Ni, J., 2014. An improved ensemble learning method for classifying high-dimensional and imbalanced biomedicine data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 11(4), pp.657-666. Available from: http://dl.acm.org/citation.cfm?id=2687041 [Accessed 24 March 2015]

Yu, L., Wang, S. and Lai, K.K., 2008. Credit risk assessment with a multistage neural network ensemble learning approach. *Expert systems with applications*, *34*(2), pp.1434-1444. Available from: http://www.sciencedirect.com/science/article/pii/S0957417407000206 [Accessed 24 February 2017]

Yuan, Y., Van Allen, E.M., Omberg, L., Wagle, N., Amin-Mansour, A., Sokolov, A., Byers, L.A., Xu, Y., Hess, K.R., Diao, L. and Han, L., 2014. Assessing the clinical utility of cancer genomic and proteomic data across tumor types. *Nature biotechnology*, *32*(7), pp.644-652. Available from: http://www.nature.com/nbt/journal/v32/n7/abs/nbt.2940.html [Accessed 26 February 2017]

Zięba, M., Tomczak, J.M. and Gonczarek, A., 2015. RBM-SMOTE: restricted Boltzmann machines for synthetic minority oversampling technique. In Intelligent Information and Database Systems (pp. 377-386). *Springer International Publishing*. Available from: http://mlg.ii.pwr.edu.pl/~adam.gonczarek/Papers/ACIIDS2015.pdf [Accessed 12 February 2016]

Zezula, P., 2015. Similarity Searching for the Big Data. *Mobile Networks and Applications*, 20(4), pp.487-496. Available from: http://link.springer.com/article/10.1007/s11036-014-0547-2 [Accessed December 12, 2015]

Zhai, Y., Ong, Y.S. and Tsang, I.W., 2014. The Emerging" Big Dimensionality". *IEEE Computational Intelligence Magazine*, *9*(3), pp.14-26. Available from: http://ieeexplore.ieee.org/abstract/document/6853478/ [Accessed 29 August 2016]

Zhang, Y.P., Zhang, L.N. and Wang, Y.C., 2010. Cluster-based majority under-sampling approaches for class imbalance learning. In *Information and Financial Engineering (ICIFE), 2010 2nd IEEE International Conference on* (pp. 400-404). IEEE. Available from: http://ieeexplore.ieee.org/abstract/document/5609385/ [Accessed 23 February 2017]

Zhang, F., Ren, C., Zhao, H., Yang, L., Su, F., Zhou, M.-M. and Walsh, M. J. 2016. Identification of novel prognostic indicators for triple-negative breast cancer patients through integrative analysis of cancer genomics data and protein interactome data. *Oncotarget*, *7*(44), 71620–71634. Available from: http://doi.org/10.18632/oncotarget.12287 [Accessed 26 February 2017]

Zhang, Z., Krawczyk, B., Garcìa, S., Rosales-Pérez, A. and Herrera, F., 2016. Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data. *Knowledge-Based Systems*, *106*, pp.251-263. Available from: http://www.sciencedirect.com/science/article/pii/S0950705116301459 [Accessed 19 February 2017]

Zheng, B., Zhang, J., Yoon, S.W., Lam, S.S., Khasawneh, M. and Poranki, S., 2015. Predictive modeling of hospital readmissions using metaheuristics and data mining. *Expert Systems with Applications*, *42*(20), pp.7110-7120. Available from: http://www.sciencedirect.com/science/article/pii/S0957417415003085 [Accessed 18 February 2017]

Zhou, Z.H., 2012. *Ensemble methods: foundations and algorithms*. CRC press. Available from: http://www.islab.ece.ntua.gr/attachments/article/86/Ensemble%20methods%20-%20Zhou.pdf [Accessed 20 February 2017]

Zhou, Z.H. and Liu, X.Y., 2006. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, *18*(1), pp.63-77. Available from: http://ieeexplore.ieee.org/abstract/document/1549828/?reload=true [Accessed 17 February 2017]

Zhu, B.Y., Bauer, S.M., Jia, Z.J., Probst, G.D., Zhang, Y. and Scarborough, R.M., Millennium Pharmaceuticals, Inc., 2012. Factor Xa inhibitors. U.S. Patent 8,153,670. Available from: http://www.google.com/patents/US8153670 [Accessed 26 September 2015].

Zong, W., Huang, G.B. and Chen, Y., 2013. Weighted extreme learning machine for imbalance learning. *Neurocomputing*, 101, pp.229-242. Available from: http://www.sciencedirect.com/science/article/pii/S0925231212006479 [Accessed 11 August 2014]

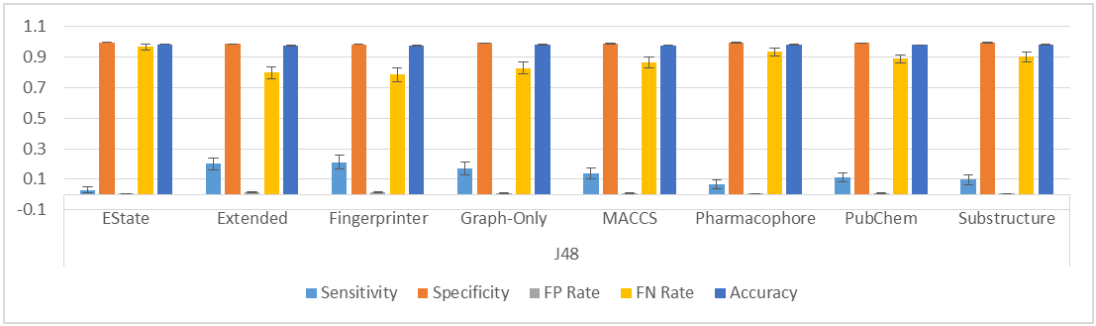
Zou, H., Hastie, T. and Tibshirani, R., 2006. Sparse principal component analysis. *Journal of computational and graphical statistics*, *15*(2), pp.265-286. Available from: http://www.tandfonline.com/doi/abs/10.1198/106186006X113430 [Accessed 19 July 2013]

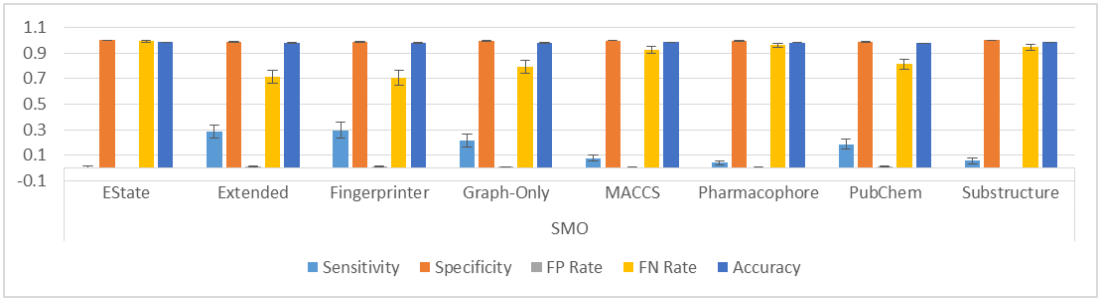
8. Appendix

This section contains figures from

Analysis of the Datasets chapter (Chapter 5) which were either redundant or did not include much information.

AID362 Figures:

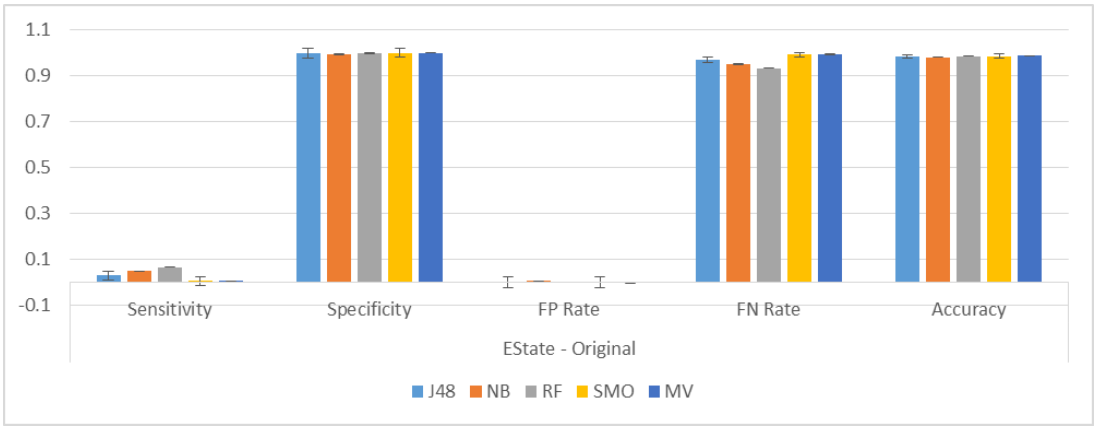


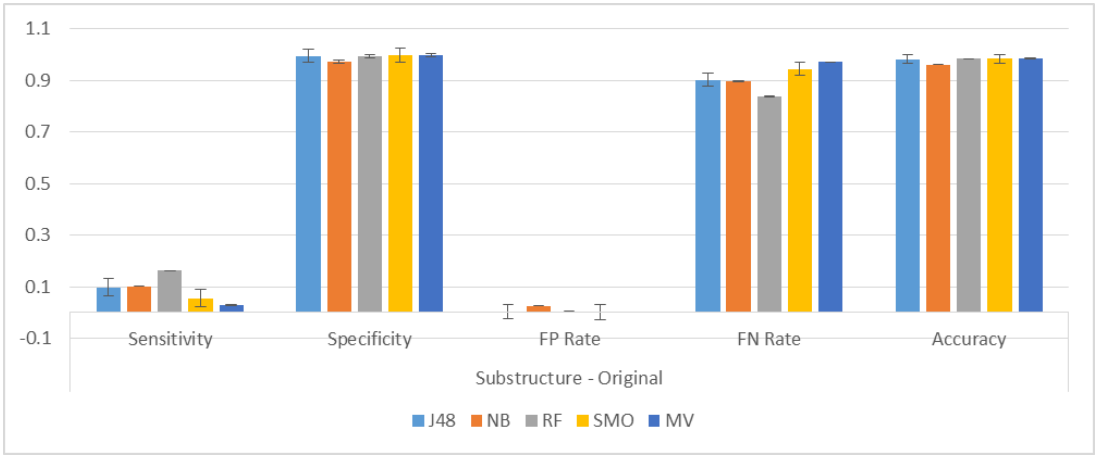


J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓**	↑**	↓**	↓**
Extended	↓	↑	↓	↑	↑
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↓	↑	↑	↓
MACCS	↑	↓	↑	↓	↑
Pharmacophore	↑**	↓**	↑**	↓**	↓*
PubChem	↑	↓	↑	↓	↓
Substructure	↑*	↓**	↑**	↓*	↓**

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑**	↓**	↑**	↓**	↓**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓**	↑**	↓	↓*
Pharmacophore	↑	↓**	↑**	↓	↓**
PubChem	↓	↓	↑	↑	↓
Substructure	↑**	↓**	↑**	↓**	↓**

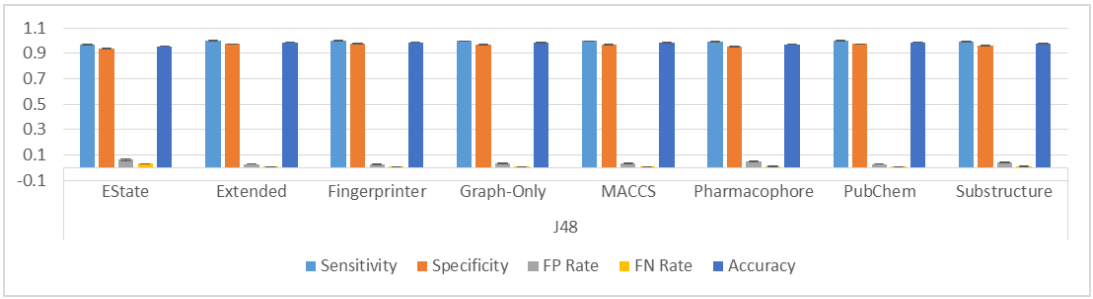
SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↓
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↓	↑	↓	↓
MACCS	↑	↓	↑	↓	↑
Pharmacophore	↑*	↓	↑	↓*	↓
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓**	↑**	↓	↓*

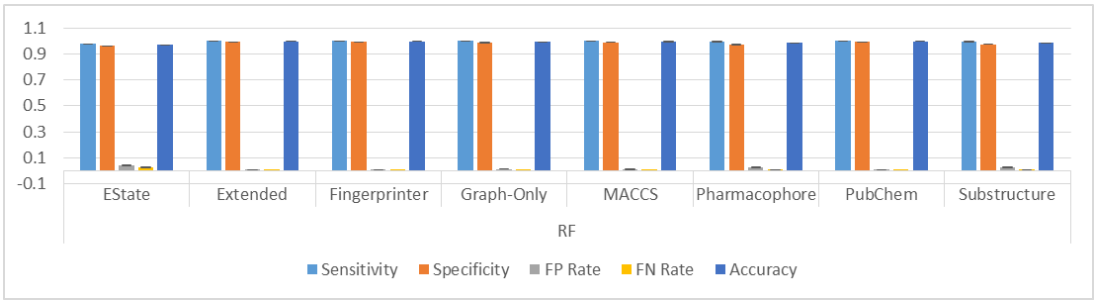


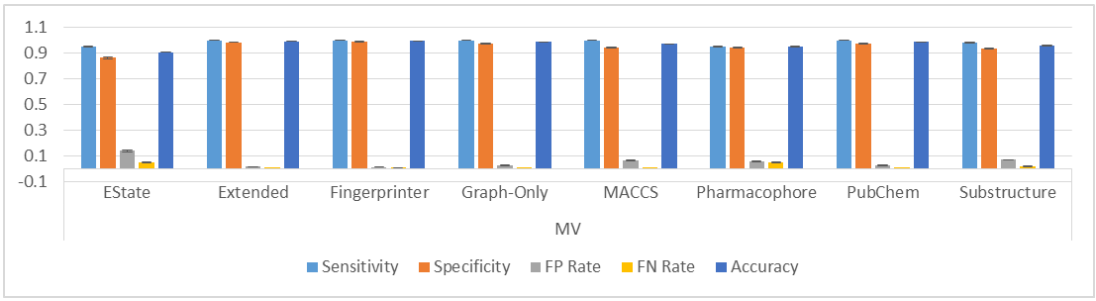


EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↓**	↑**	↓**	↓**
NB	↑**	↓**	↑**	↓**	↓**
RF	↑	↑**	↓**	↓	↑**
SMO	↑	↓	↑	↓	↓
MV	↑**	↓**	↑**	↓**	↓**

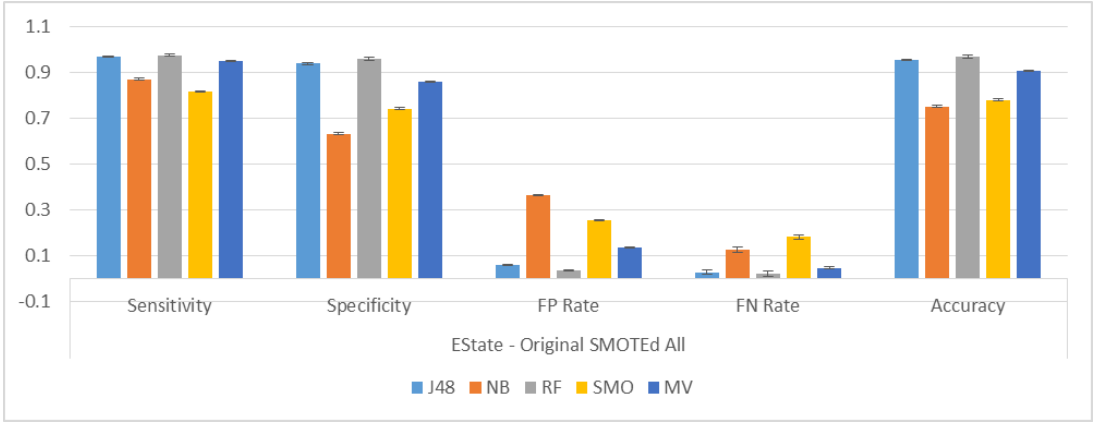
MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↑
NB	↑	↓**	↑**	↓	↓*
RF	↑	↑**	↓**	↓	↑**
SMO	↑	↓	↑	↓	↑
MV	↑	↓	↑	↓	↓

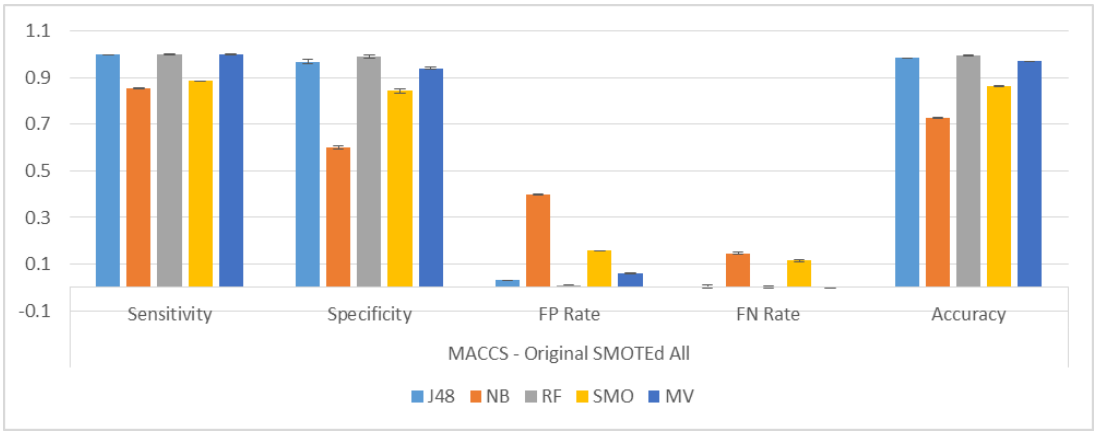


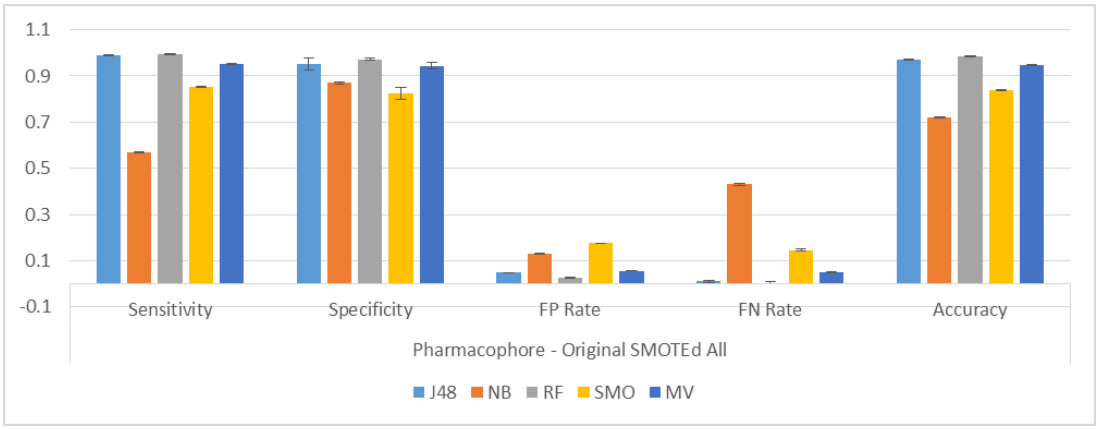




SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↔	↓	↑	↔	↓
Fingerprinter	↔	↓	↑	↔	↓
Graph-Only	↓*	↑	↓	↑*	↓
MACCS	↑**	↑	↓	↓**	↑**
Pharmacophore	↑**	↑**	↓**	↓**	↑**
PubChem	↑**	↑**	↓**	↓**	↑**
Substructure	↓	↑**	↓**	↑	↑**

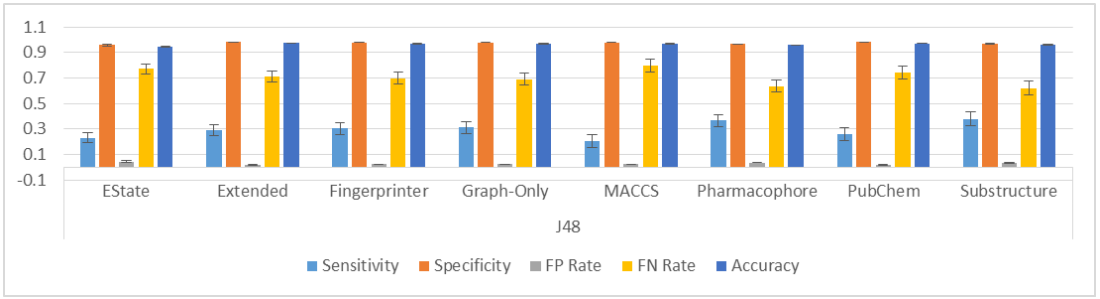


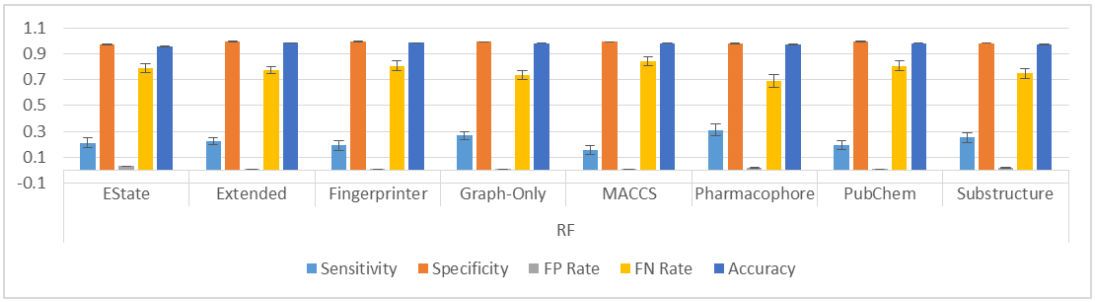




EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑**	↑**	↓**	↓**	↑**
NB	↓**	↑**	↓**	↑**	↑**
RF	↑**	↑**	↓**	↓**	↑**
SMO	↓	↑**	↓**	↑	↑**
MV	↑**	↑**	↓**	↓**	↑**

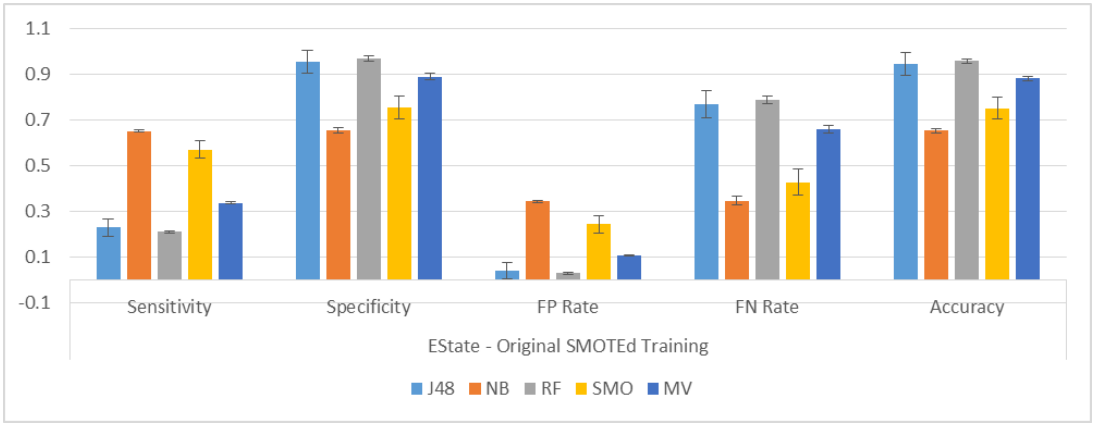
PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓*	↑*	↓*	↑*	↑
NB	↓**	↑**	↓**	↑**	↓
RF	↓	↑*	↓*	↑	↑
SMO	↑**	↑**	↓**	↓**	↑**
MV	↑	↑*	↓*	↓	↑*

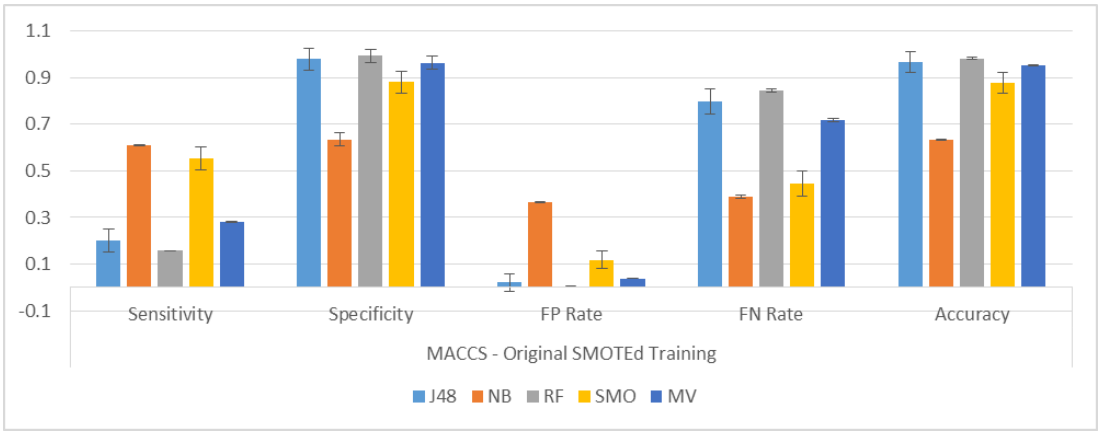




Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑*	↓*	↑*	↓*	↓*
MACCS	↑	↑*	↓*	↓	↑*
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↓	↓	↑	↑	↓
Substructure	↓	↑	↓	↑	↑

Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑**	↓**	↓	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↓	↑	↓	↑	↑
MACCS	↑	↑**	↓**	↓	↑**
Pharmacophore	↓*	↑**	↓**	↑*	↑**
PubChem	↓	↑	↓	↑	↑
Substructure	↓	↑**	↓**	↑	↑**

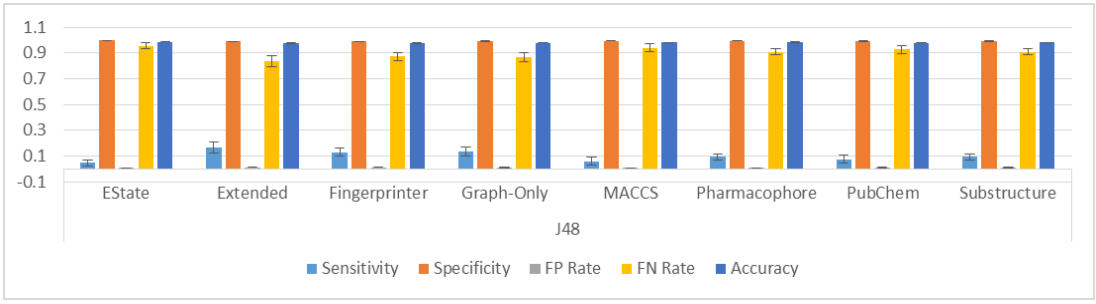


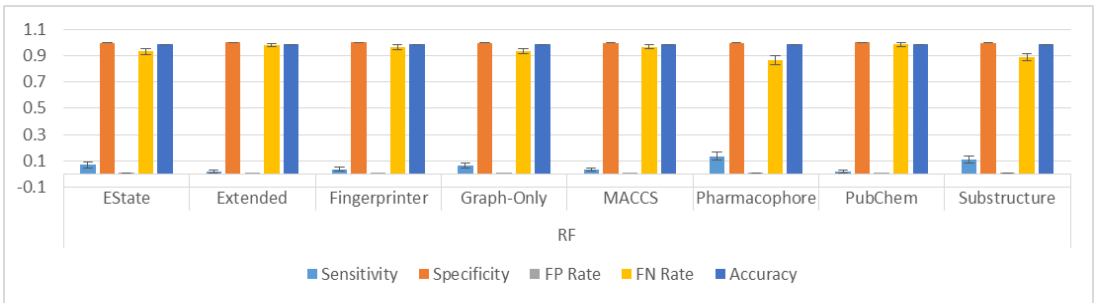


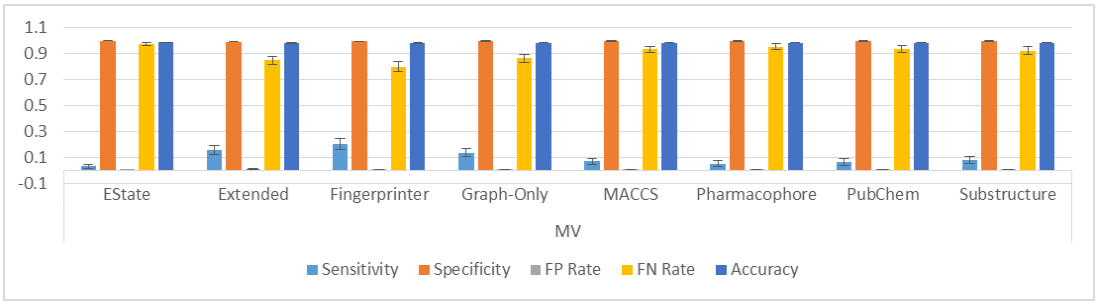
PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↓	↓	↑	↑	↓
RF	↑	↑*	↓*	↓	↑*
SMO	↑	↑	↓	↓	↑
MV	↓	↑	↓	↑	↑

Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓**	↑**	↓**	↑**	↑**
NB	↑	↓	↑	↓	↓
RF	↓**	↑**	↓**	↑**	↑**
SMO	↓	↑*	↓*	↑	↑*
MV	↓*	↑**	↓**	↑*	↑**

SSSS



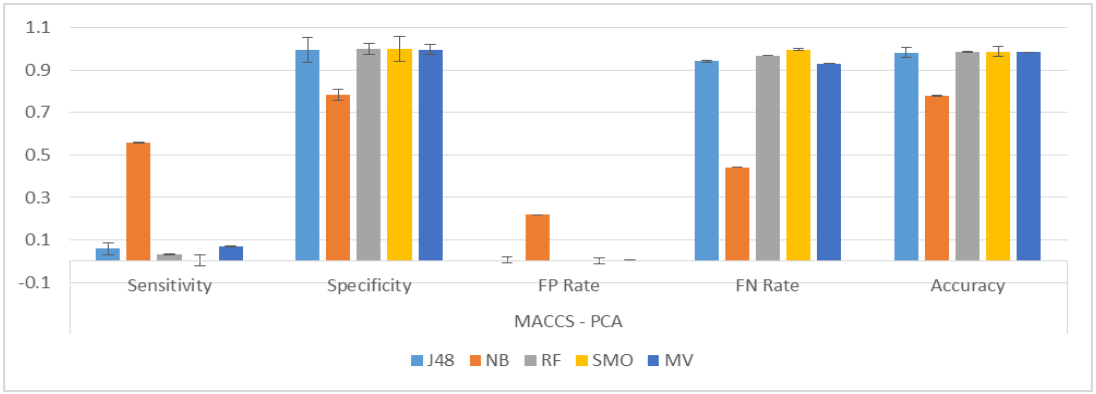


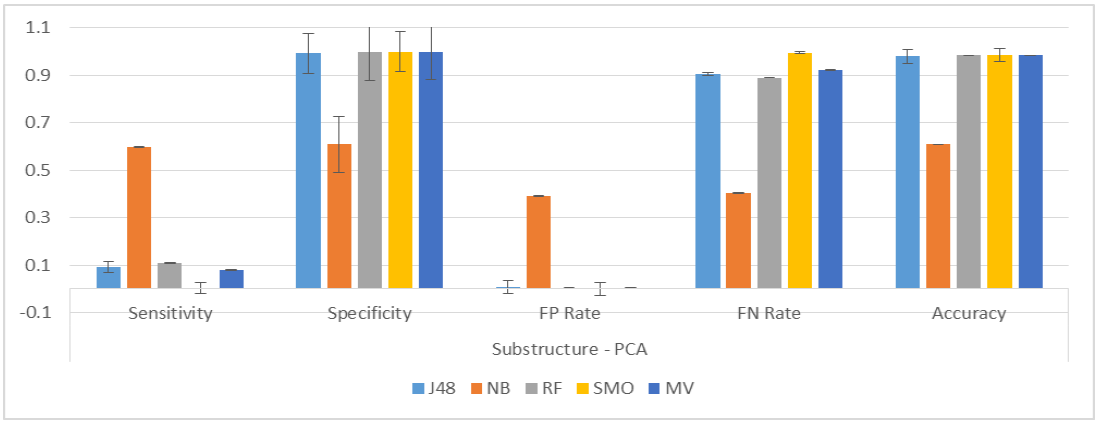


J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↓
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↓	↑	↓	↓
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↑	↓	↑	↓	↓
Substructure	↑	↑	↓	↓	↑

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑	↓	↑	↑
Extended	↑	↓	↑	↓	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↑	↑	↓	↓	↑
PubChem	↑	↓	↑	↓	↓
Substructure	↓	↓	↑	↑	↓

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↔	↓	↑	↔	↓
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↑	↓	↑	↓	↓
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↑	↓	↑	↓	↓
Substructure	↑	↓	↑	↓	↓

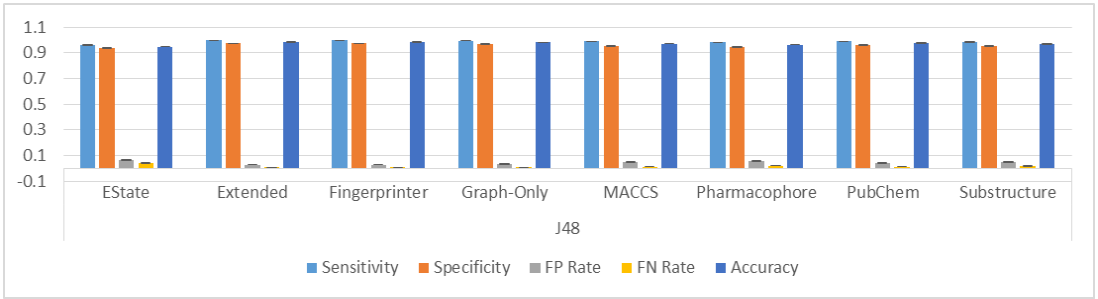


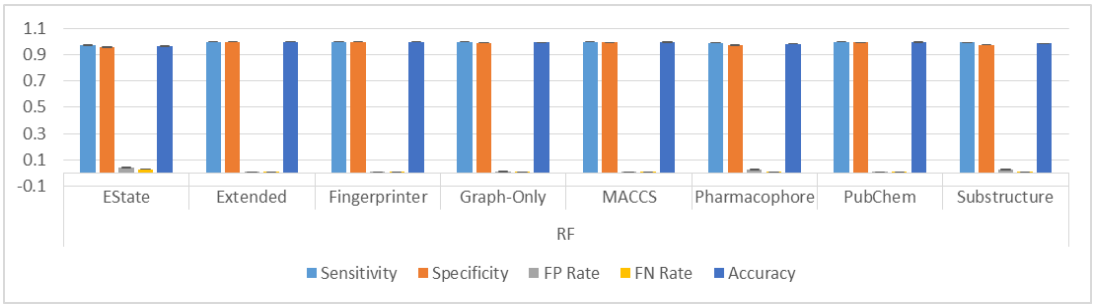


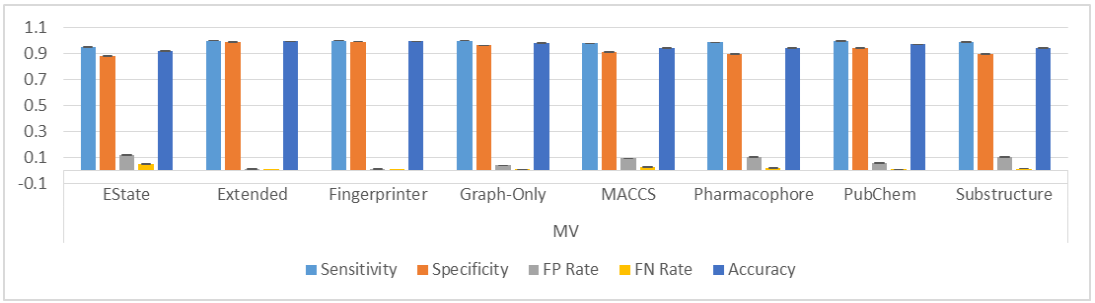
EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↓	↑	↓	↑	↑
RF	↓	↑*	↓*	↑	↑
SMO	↔	↓	↑	↔	↓
MV	↓	↓	↑	↑	↓

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓	↓	↑	↑	↓
RF	↓	↑	↓	↑	↑
SMO	↑	↓	↑	↓	↓
MV	↑	↓	↑	↓	↓

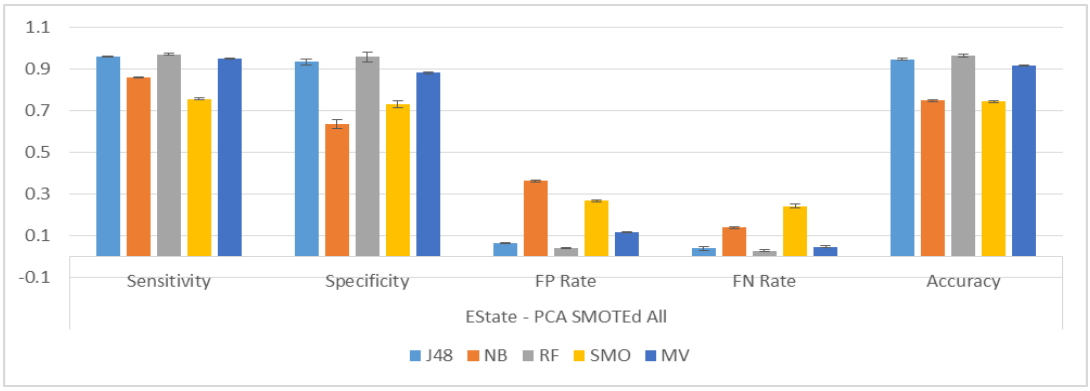
PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↓	↑	↓	↓
NB	↑	↓	↑	↓	↓
RF	↓	↑	↓	↑	↑
SMO	↑	↓	↑	↓	↓
MV	↑	↑	↓	↓	↑

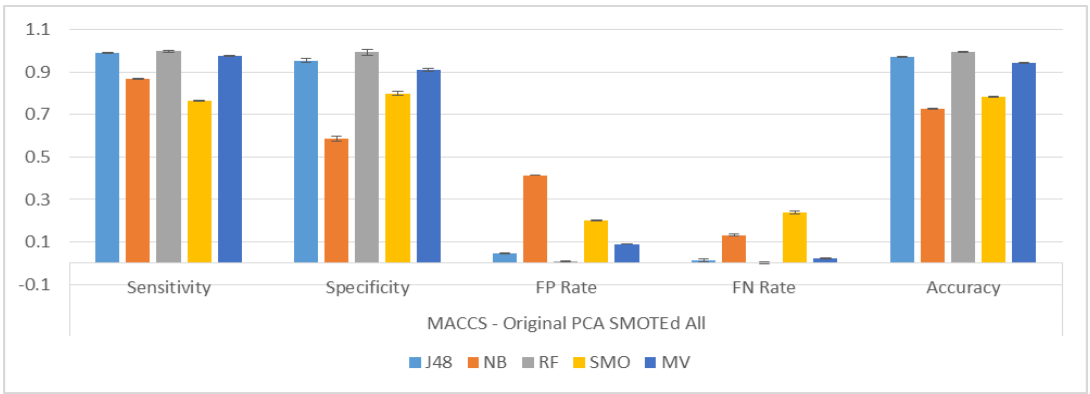


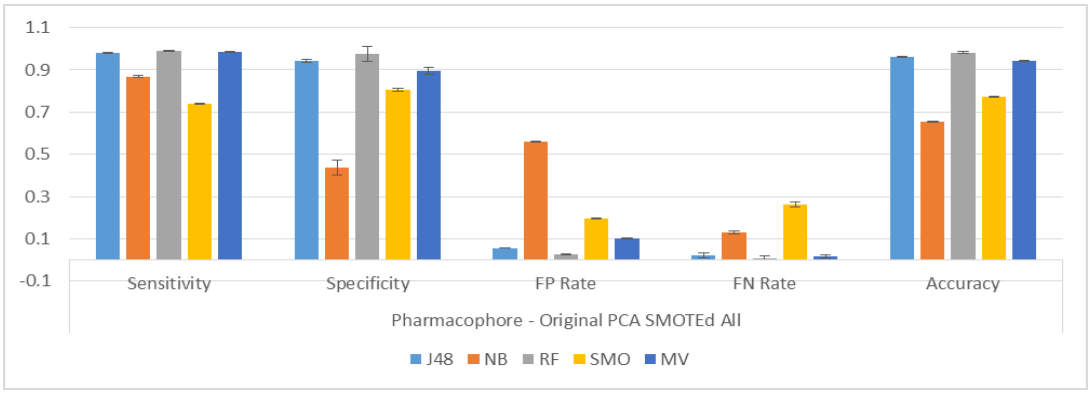




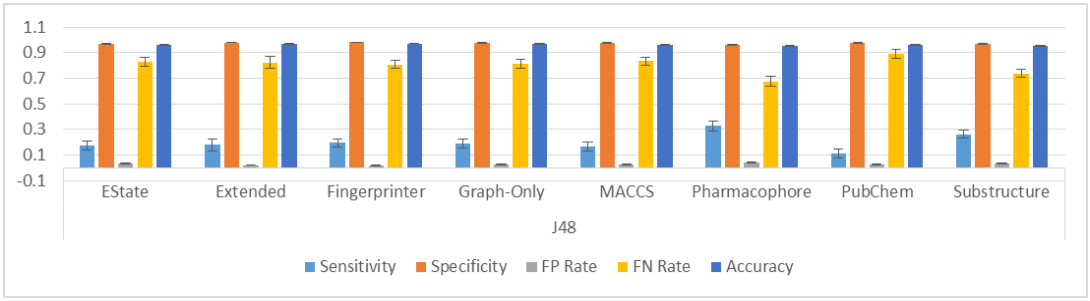
Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑
Extended	↑	↓	↑	↓	↑
Fingerprinter	↓**	↑	↓	↑**	↓
Graph-Only	↑**	↑	↓	↓**	↑**
MACCS	↑**	↑	↓	↓**	↑**
Pharmacophore	↑**	↑	↓	↓**	↑
PubChem	↓	↓	↑	↑	↓
Substructure	↑**	↓**	↑**	↓**	↓*

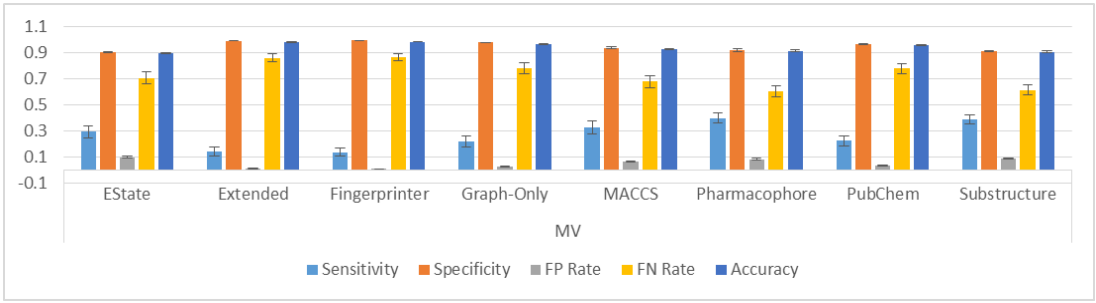






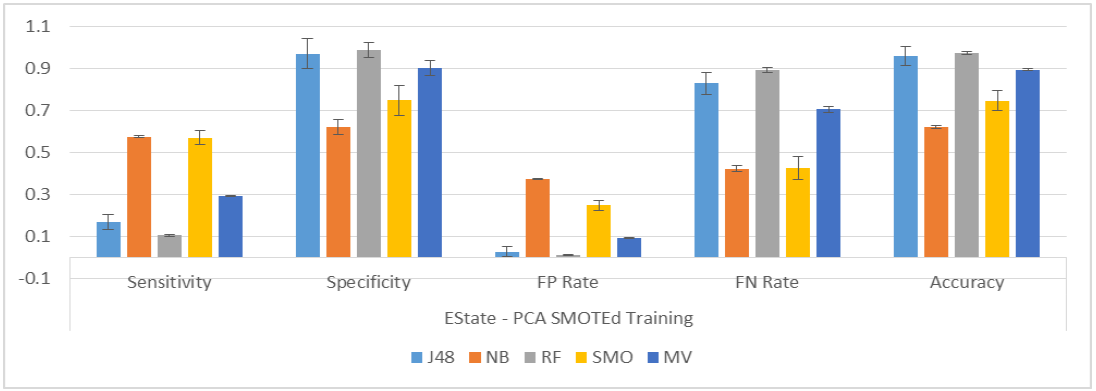
PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓	↓	↑	↑	↓
RF	↓	↑	↓	↑	↑
SMO	↑*	↓	↑	↓*	↑
MV	↑	↑*	↓*	↓	↑

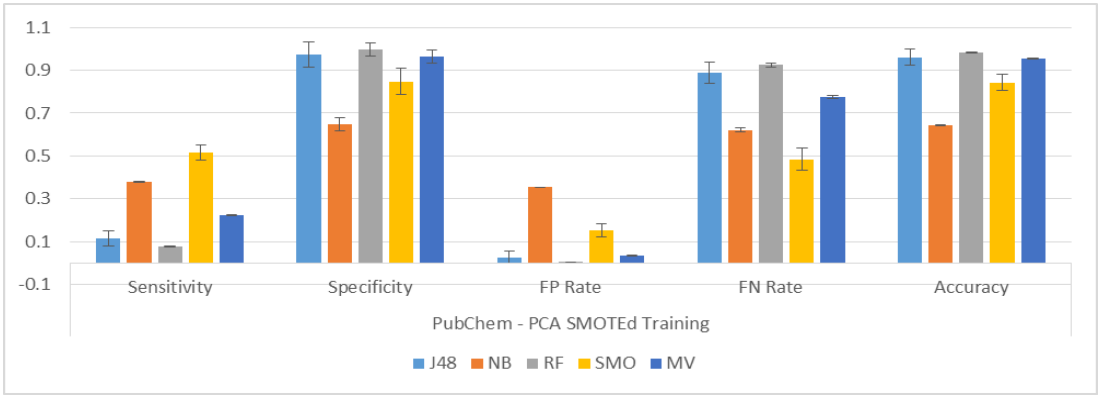




Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓*	↑**	↓**	↑*	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↑	↓	↑	↓	↓
Substructure	↓	↓	↑	↑	↓

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↓
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↓	↑	↓	↑	↑
MACCS	↓	↑*	↓*	↑	↑*
Pharmacophore	↓	↓	↑	↑	↓
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓	↑	↓	↓

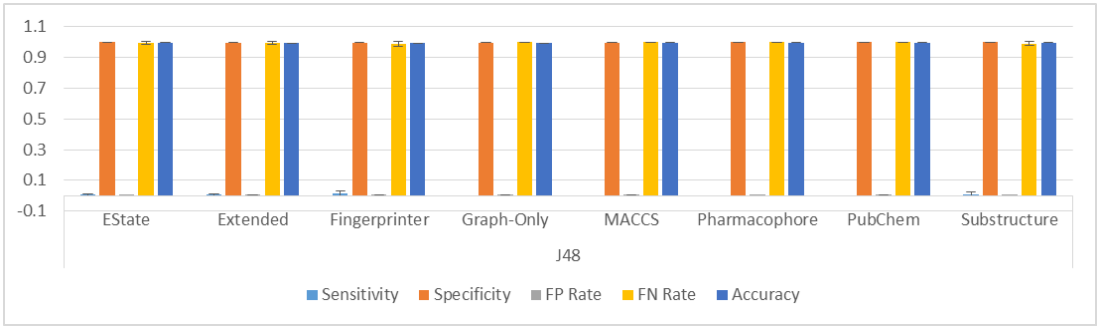


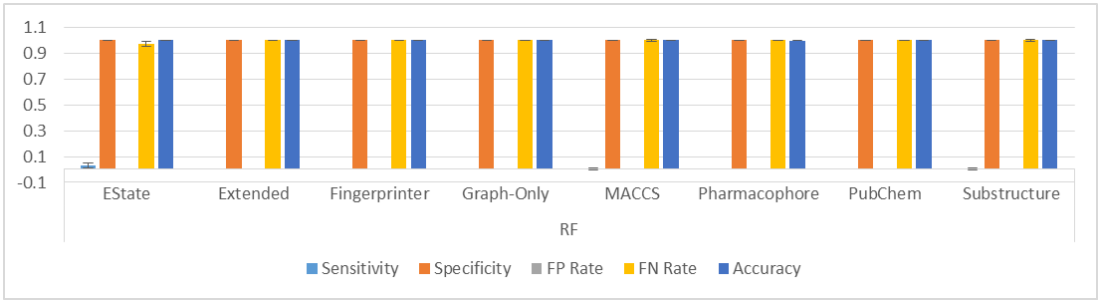


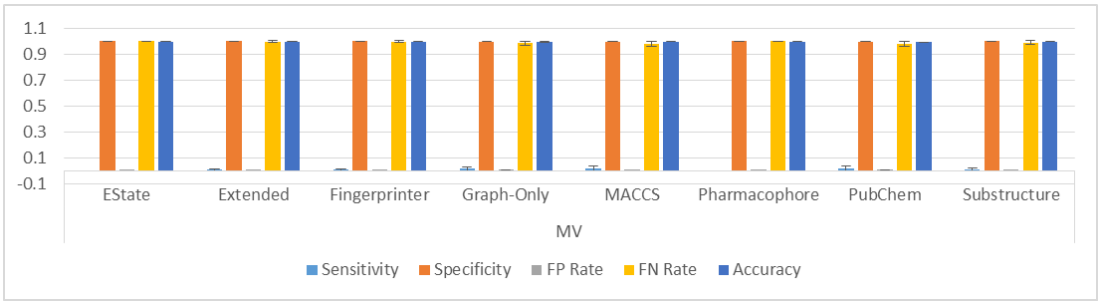
MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓	↑	↓	↑	↑
RF	↓*	↑	↓	↑*	↑
SMO	↓	↑*	↓*	↑	↑*
MV	↓	↑	↓	↑	↑

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑
NB	↑	↓	↑	↓	↓
RF	↑	↓	↑	↓	↑
SMO	↑	↑	↓	↓	↑
MV	↓	↑	↓	↑	↑

AID456 Figures:



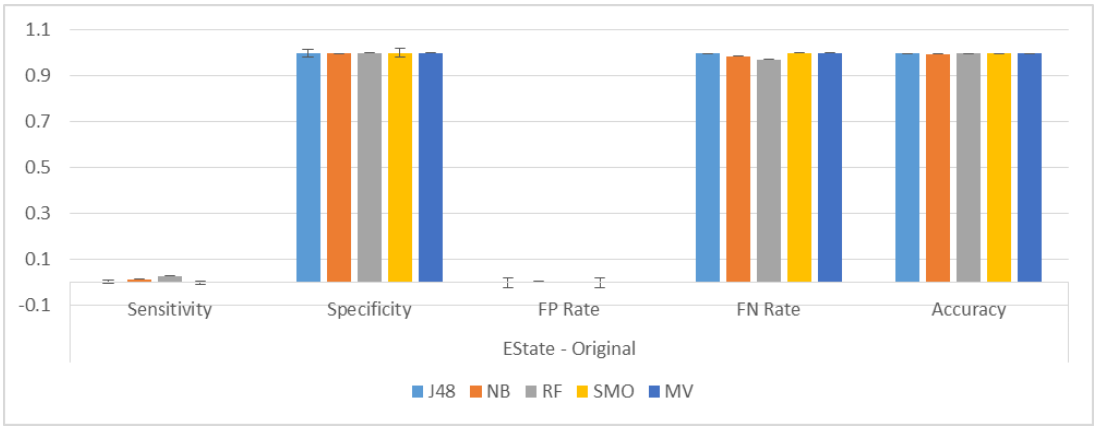


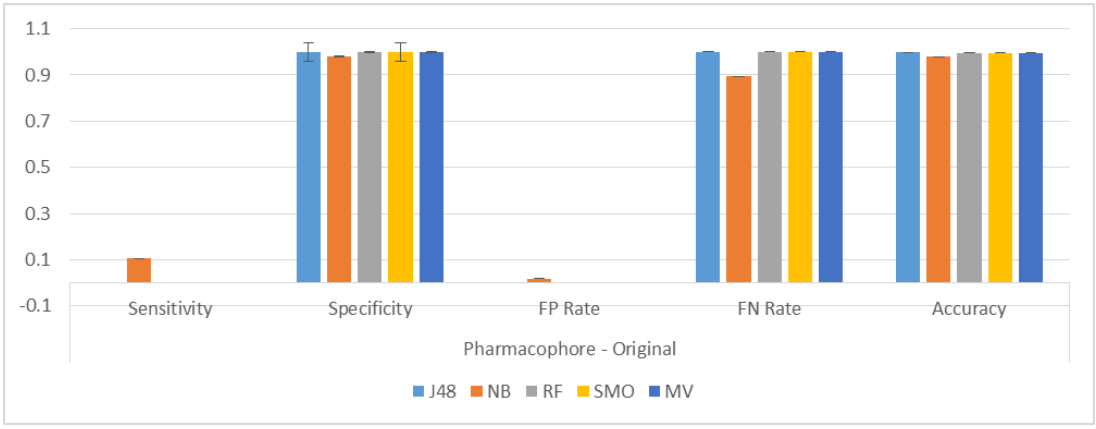


J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↓**	↑**	↑	↓**
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↓	↑	↑	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓	↑	↓	↓
Pharmacophore	↔	↓**	↑**	↔	↓**
PubChem	↔	↑	↓	↔	↑
Substructure	↓	↓**	↑**	↑	↓**

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↔	↓*	↑*	↔	↓*
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↑	↓	↓	↑
MACCS	↑	↓	↑	↓	↓
Pharmacophore	↑	↓*	↑*	↓	↓*
PubChem	↑	↑	↓	↓	↑
Substructure	↑	↓	↑	↓	↓

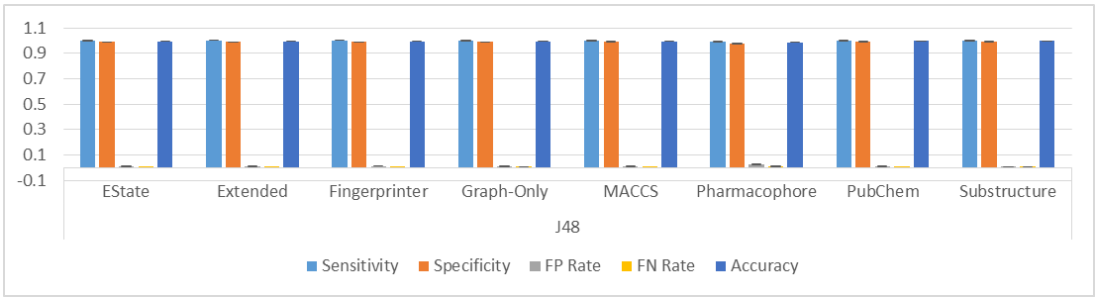
Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↔	↓**	↑**	↔	↓**
Extended	↓	↑	↓	↑	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↓	↓	↑	↑	↓
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↔	↓	↑	↔	↓
Substructure	↑	↓	↑	↓	↓

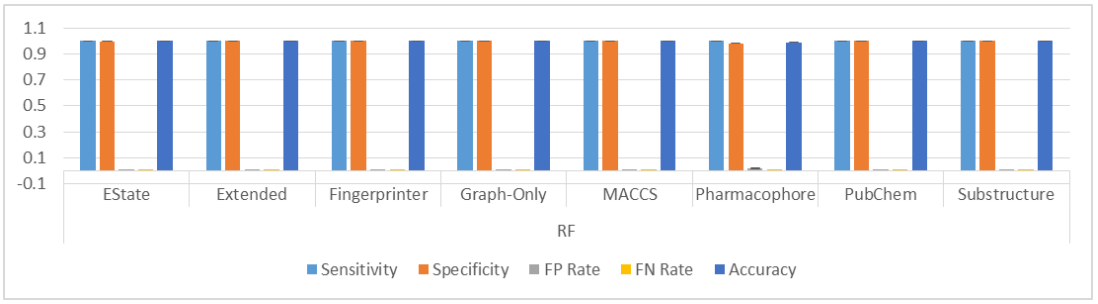


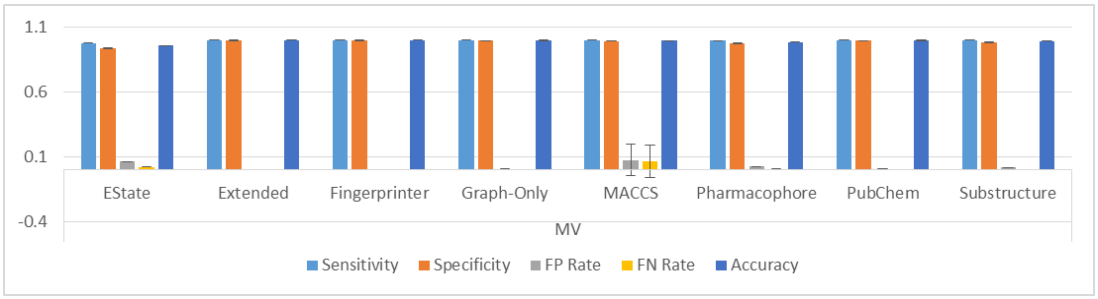


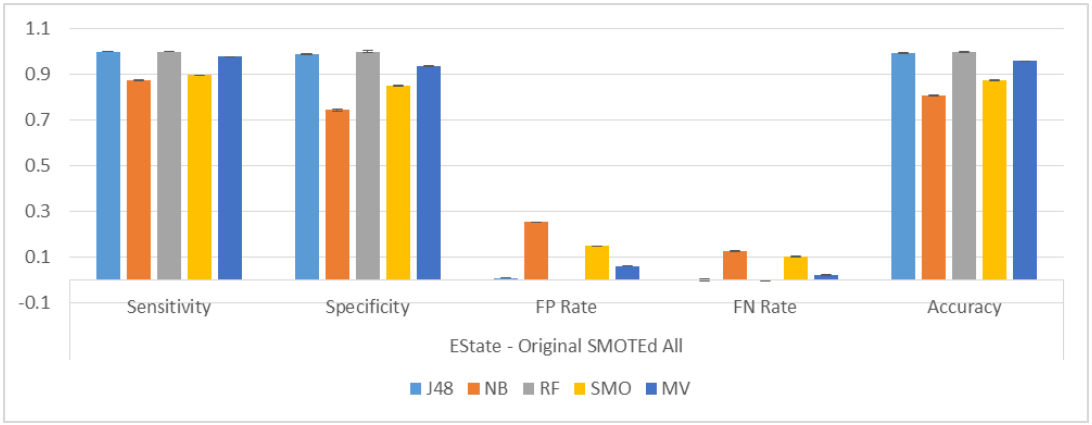
EState	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓**	↑**	↑	↓**
NB	↑**	↓**	↑**	↓**	↓**
RF	↓**	↑**	↓**	↑**	↑**
SMO	↔	↓*	↑*	↔	↓*
MV	↔	↓**	↑**	↔	↓**

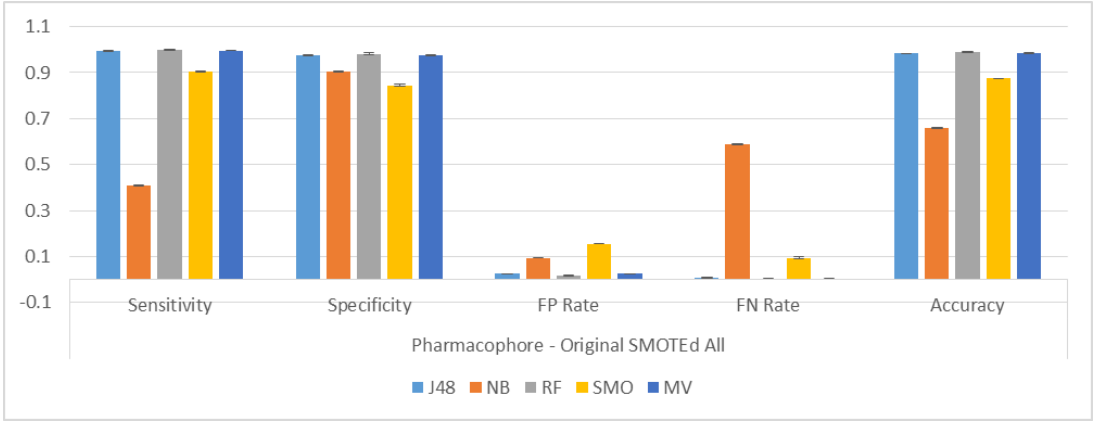
Pharmacophore	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↓**	↑**	↔	↓**
NB	↑**	↓**	↑**	↓**	↓**
RF	↔	↑**	↓**	↔	↑**
SMO	↑	↓*	↑*	↓	↓*
MV	↑	↓	↑	↓	↓

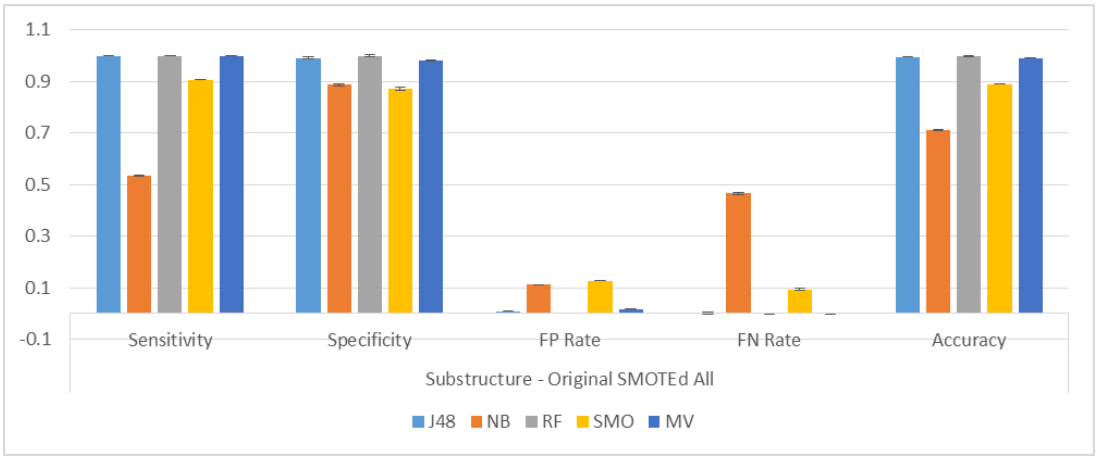

1.1
0.9
0.7
0.5
0.3
0.1
-0.1
EState
Extended
Fingerprinter
Graph-Only
MACCS
Pharmacophore
PubChem
Substructure
J48
Sensitivity
Specificity
FP Rate
FN Rate
Accuracy


1.1
0.9
0.7
0.5
0.3
0.1
-0.1
EState
Extended
Fingerprinter
Graph-Only
MACCS
Pharmacophore
PubChem
Substructure
RF
Sensitivity
Specificity
FP Rate
FN Rate
Accuracy


1.1
0.6
0.1
-0.4
EState
Extended
Fingerprinter
Graph-Only
MACCS
Pharmacophore
PubChem
Substructure
MV
Sensitivity
Specificity
FP Rate
FN Rate
Accuracy

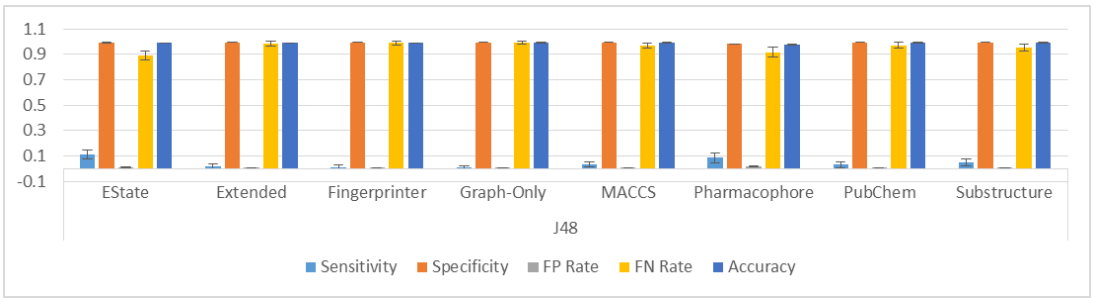

1.1
0.9
0.7
0.5
0.3
0.1
-0.1
Sensitivity
Specificity
FP Rate
FN Rate
Accuracy
EState - Original SMOTEd All
J48
NB
RF
SMO
MV

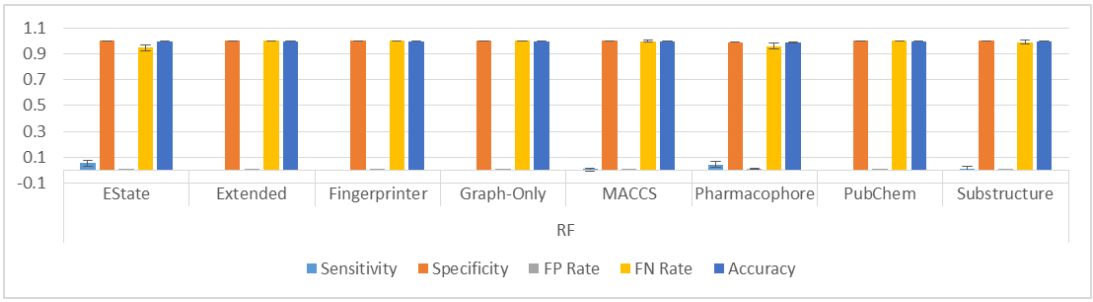

1.1
0.9
0.7
0.5
0.3
0.1
-0.1
Sensitivity
Specificity
FP Rate
FN Rate
Accuracy
Pharmacophore - Original SMOTEd All
J48
NB
RF
SMO
MV

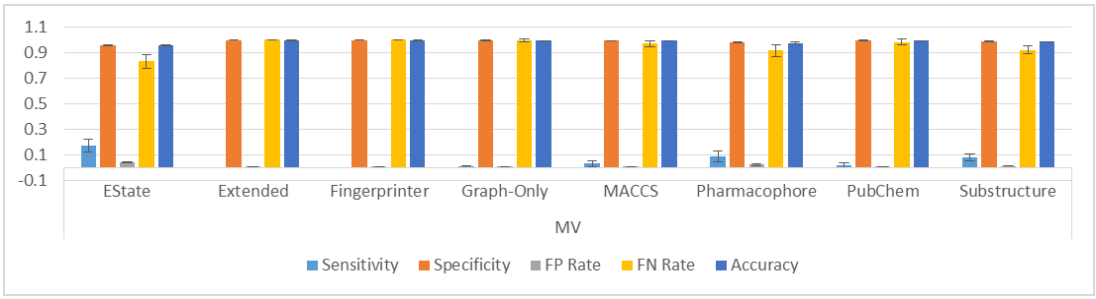


MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↑**	↑	↓	↓**	↑**
RF	↑	↑*	↓*	↓	↑*
SMO	↓	↑**	↓**	↑	↑**
MV	↑	↑**	↓	↓	↑**

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓	↑	↑	↓
NB	↑**	↑	↓	↓**	↑*
RF	↑	↑	↓	↓	↑
SMO	↔	↑*	↓*	↔	↑*
MV	↑	↓	↑	↓	↑

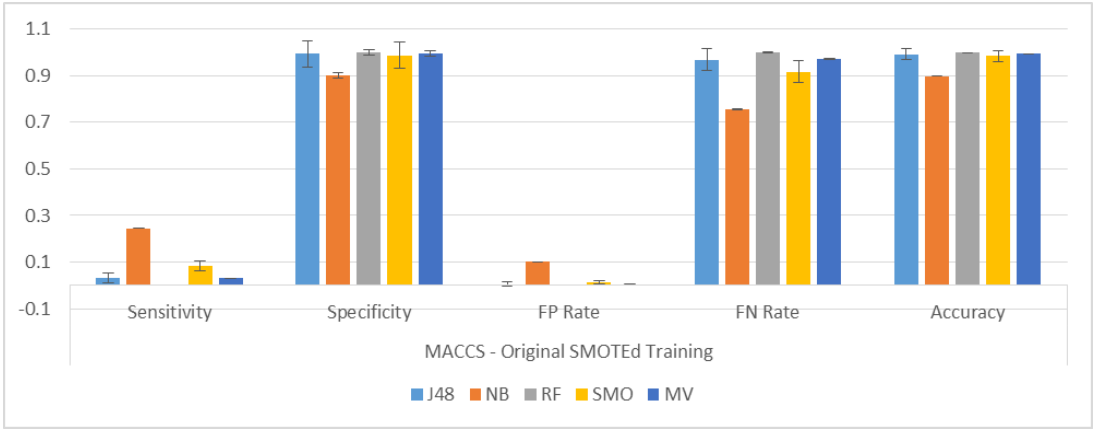


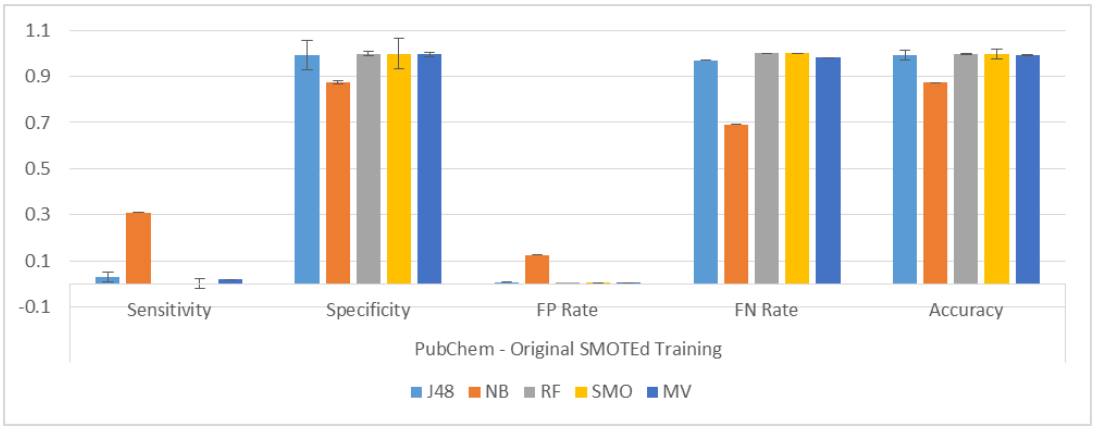




Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↓	↑	↑	↓
Extended	↑	↑	↓	↓	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↑	↑**	↓**	↓	↑**
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↑	↓	↑	↓	↓
PubChem	↓	↑	↓	↑	↑
Substructure	↑	↓	↑	↓	↓

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↔	↓	↑	↔	↓
Fingerprinter	↔	↑	↓	↔	↑
Graph-Only	↔	↑	↓	↔	↑
MACCS	↑	↑	↓	↓	↑
Pharmacophore	↓**	↑**	↓**	↑**	↑**
PubChem	↔	↑	↓	↔	↑
Substructure	↓	↑**	↓**	↑	↑**

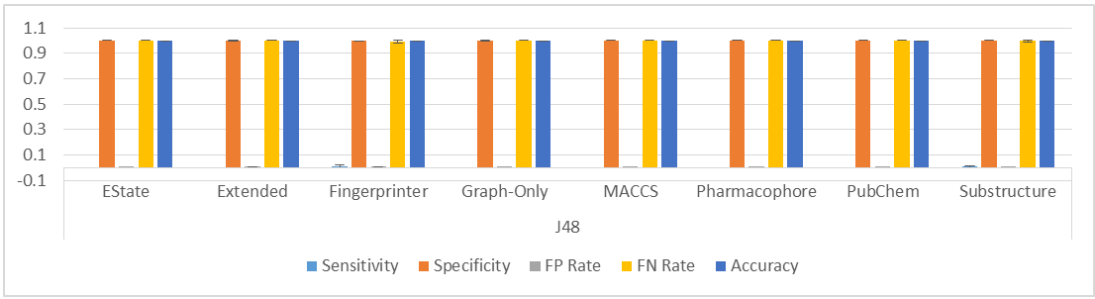


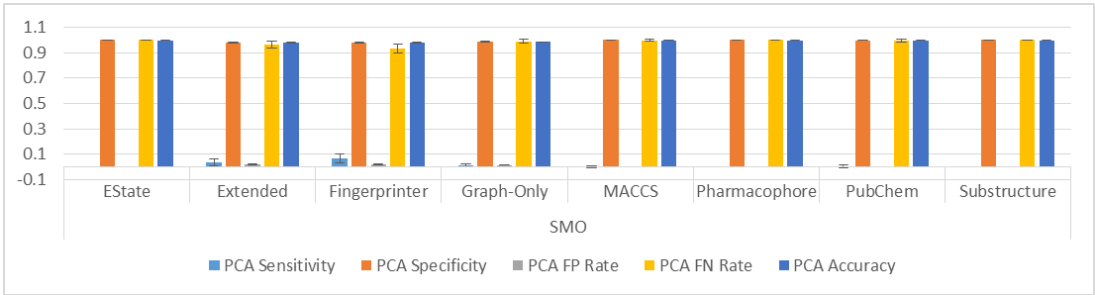


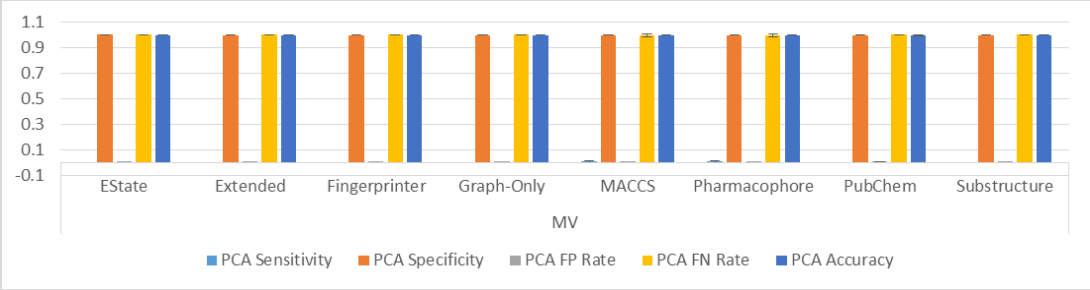
MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↑	↓	↔	↑
NB	↓	↑	↓	↑	↑
RF	↑	↑	↓	↓	↑
SMO	↓	↑	↓	↑	↑
MV	↑	↑*	↓*	↓	↑*

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↓	↑	↓	↑	↑
RF	↔	↑	↓	↔	↑
SMO	↔	↑	↓	↔	↑
MV	↔	↑	↓	↔	↑

Substructure	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↑	↓	↑	↑
NB	↑	↓	↑	↓	↓
RF	↓	↑**	↓**	↑	↑**
SMO	↓*	↑*	↓*	↑*	↑*
MV	↓	↑**	↓**	↑	↑**





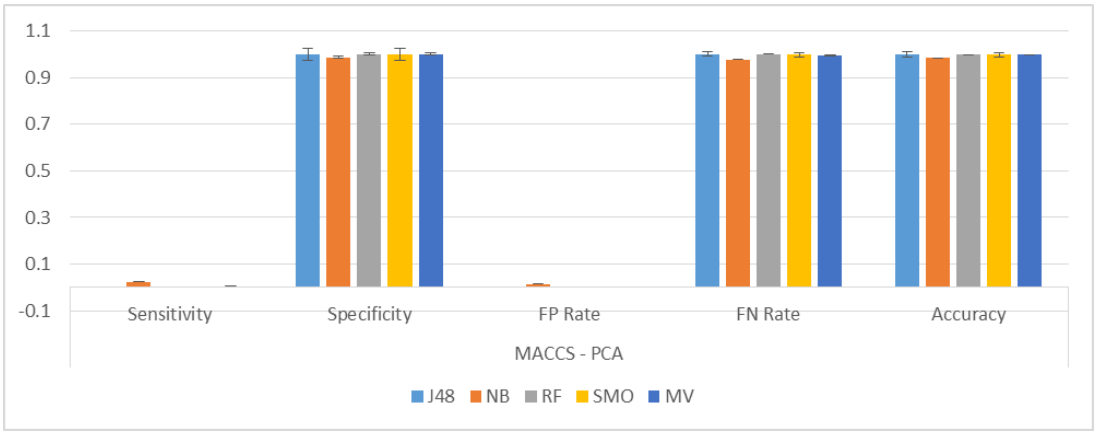


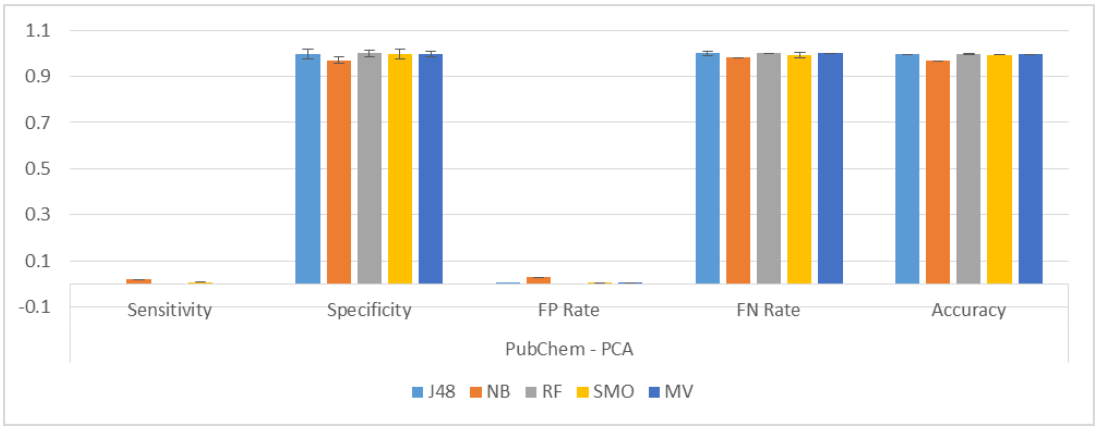
J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↔	↓	↑	↔	↓
Extended	↑	↓	↑	↓	↓
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↔	↓	↑	↔	↓
MACCS	↔	↑	↓	↔	↑
Pharmacophore	↔	↓	↑	↔	↓
PubChem	↔	↓	↑	↔	↓
Substructure	↓	↑	↓	↑	↑

Naïve Bayes	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑*	↓*	↓	↑*
Extended	↔	↓	↑	↔	↓
Fingerprinter	↔	↓	↑	↔	↓
Graph-Only	↔	↑	↓	↔	↑
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓	↑	↓	↑	↑
PubChem	↔	↓	↑	↔	↓
Substructure	↓	↑	↓	↑	↑

SMO	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↔	↔	↔	↔	↔
Extended	↓	↑	↓	↑	↑
Fingerprinter	↑	↓	↑	↓	↓
Graph-Only	↑	↑	↓	↓	↑
MACCS	↔	↓	↑	↔	↓
Pharmacophore	↔	↑	↓	↔	↑
PubChem	↑	↓	↑	↓	↓
Substructure	↑	↓*	↑*	↓	↓*

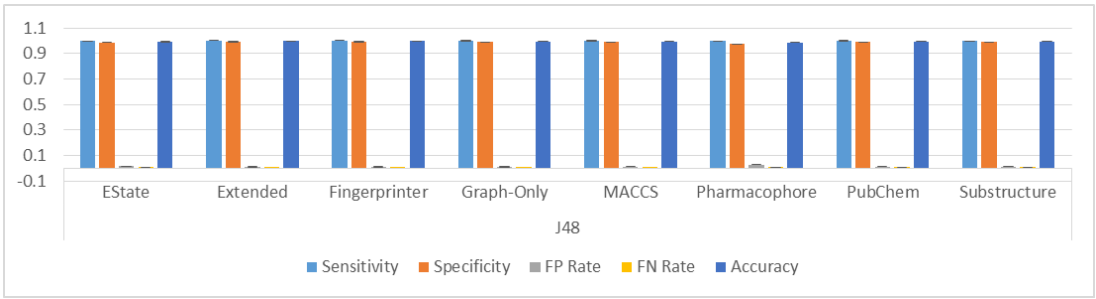
Majority Voting	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↑	↓	↓	↑
Extended	↔	↓	↑	↔	↓
Fingerprinter	↔	↓	↑	↔	↓
Graph-Only	↔	↑	↓	↔	↑
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↑	↔	↔	↓	↑
PubChem	↔	↑	↓	↔	↑
Substructure	↑	↓	↑	↓	↓

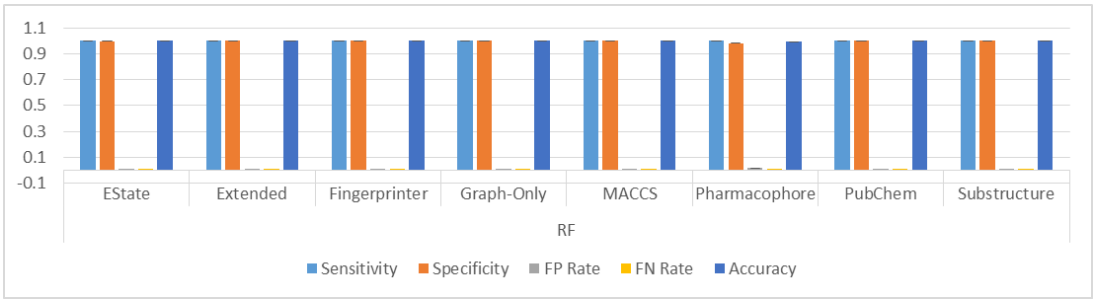


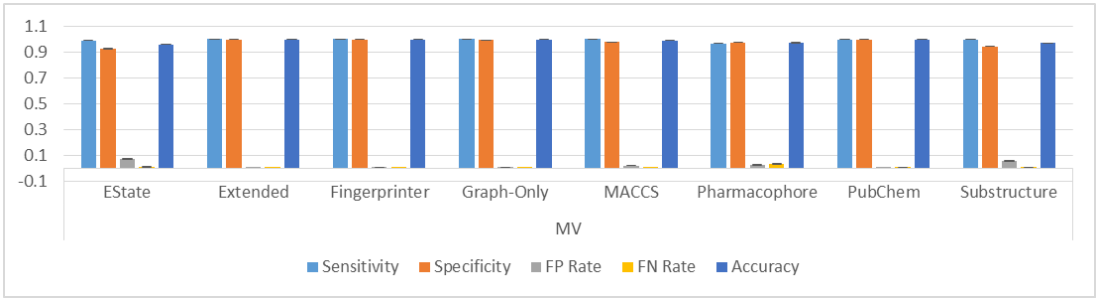


MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↑	↓	↔	↑
NB	↓	↑	↓	↑	↑
RF	↔	↑	↓	↔	↑
SMO	↔	↓	↑	↔	↓
MV	↓	↓	↑	↑	↓

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↓	↑	↔	↓
NB	↔	↓	↑	↔	↓
RF	↔	↓	↑	↔	↓
SMO	↑	↓	↑	↓	↓
MV	↔	↑	↓	↔	↑

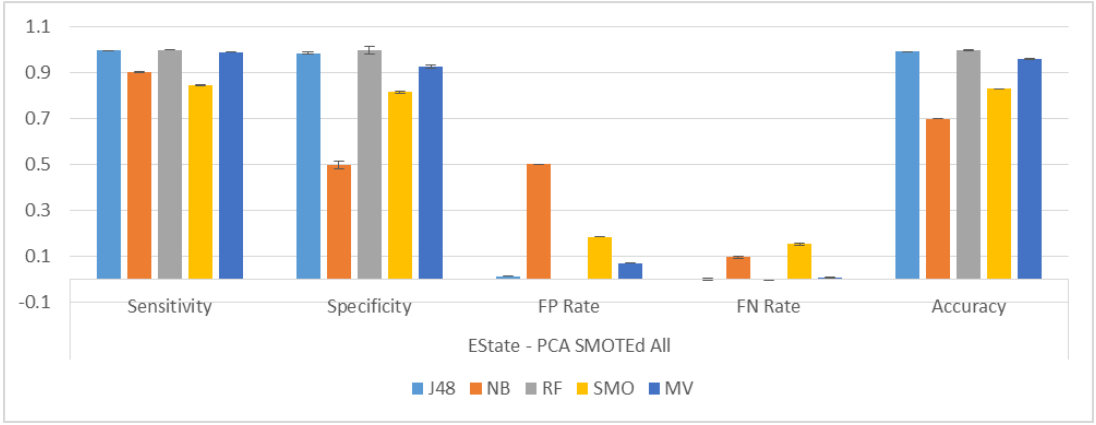


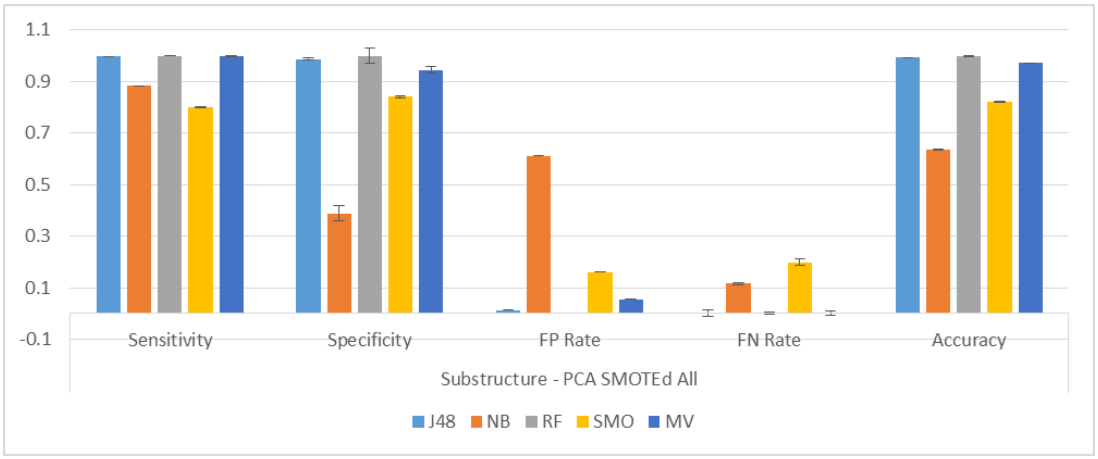




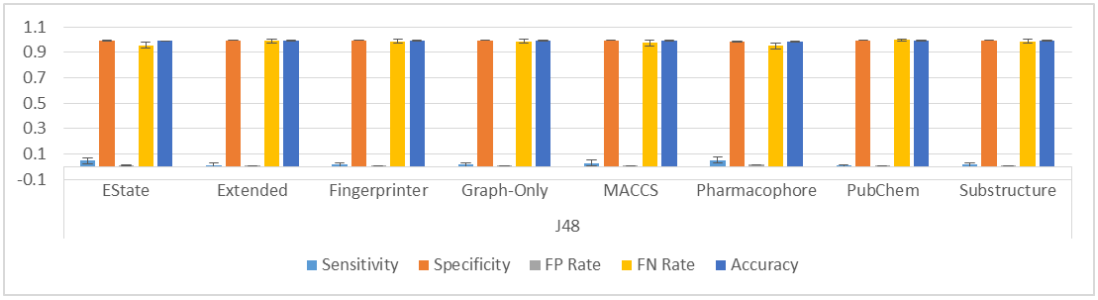
J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↑	↓	↑	↓	↓
Extended	↓	↓	↑	↑	↓
Fingerprinter	↓	↓**	↑**	↑	↓**
Graph-Only	↓	↑	↓	↑	↑
MACCS	↑	↑	↓	↓	↑*
Pharmacophore	↑	↑**	↓**	↓	↑**
PubChem	↑	↑	↓	↓	↑
Substructure	↓	↓	↑	↑	↓

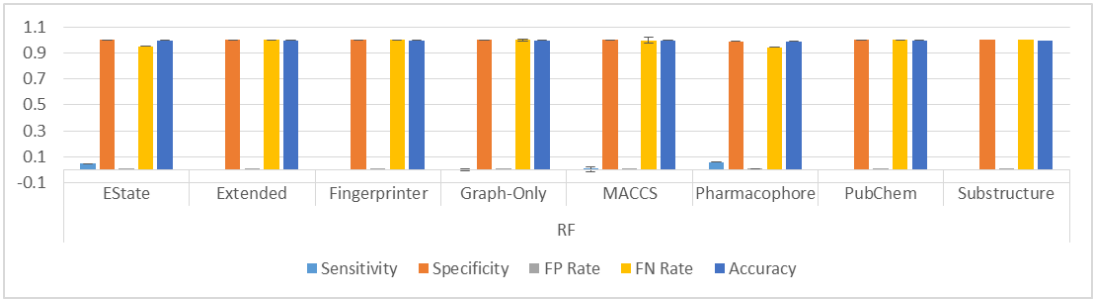
Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑**	↓**	↑	↑**
Extended	↓	↓	↑	↑	↓
Fingerprinter	↑	↑	↓	↓	↑
Graph-Only	↑	↔	↔	↓	↑
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↑	↑**	↓**	↓	↑**
PubChem	↑	↓	↑	↓	↓
Substructure	↓	↑**	↓**	↑	↑

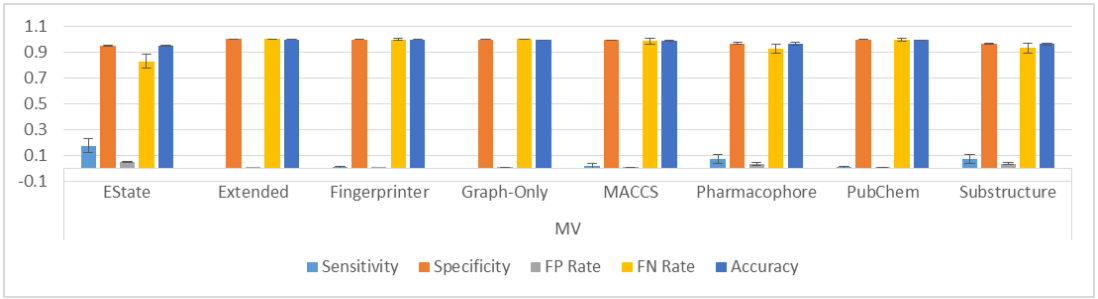




MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↑	↑	↓	↓	↑*
NB	↑	↑**	↓**	↓	↑**
RF	↓	↓	↑	↑	↓
SMO	↓	↑	↓	↑	↓
MV	↓	↑**	↓**	↑	↑**







J48	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓	↑	↓	↑	↑
Extended	↓	↑	↓	↑	↑
Fingerprinter	↓	↑	↓	↑	↑
Graph-Only	↓	↓	↑	↑	↓
MACCS	↓	↓	↑	↑	↓
Pharmacophore	↓	↑**	↓**	↑	↑**
PubChem	↔	↑	↓	↔	↑
Substructure	↑	↓	↑	↓	↓

Random Forest	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
EState	↓**	↑**	↓**	↑**	↑**
Extended	↔	↑	↓	↔	↑
Fingerprinter	↔	↑	↓	↔	↑
Graph-Only	↔	↓	↑	↔	↓
MACCS	↓	↑	↓	↑	↑
Pharmacophore	↓*	↑**	↓**	↑*	↑**
PubChem	↔	↔	↔	↔	↔
Substructure	↑	↑	↓	↓	↑

MACCS	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↓	↓	↑	↑	↓
NB	↑	↑	↓	↓	↑
RF	↓	↑	↓	↑	↑
SMO	↓	↑	↓	↑	↑
MV	↓	↑	↓	↑	↑

PubChem	Sensitivity	Specificity	FP Rate	FN Rate	Accuracy
J48	↔	↑	↓	↔	↑
NB	↑	↓	↑	↓	↓
RF	↔	↔	↔	↔	↔
SMO	↓**	↑**	↓**	↑**	↑**
MV	↓	↑	↓	↑	↑