



Faculty of Science & Technology
Department of Computing and Informatics

Mohammad Heydari, PhD Thesis

Indeterminacy-Aware Prediction Model for
Authentication in IoT

Bournemouth University, Department of Computing and Informatics, PhD Thesis

Copyright Statement

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

Abstract

The Internet of Things (IoT) has opened a new chapter in data access. It has brought obvious opportunities as well as major security and privacy challenges. Access control is one of the challenges in IoT. This holds true as the existing, conventional access control paradigms do not fit into IoT, thus access control requires more investigation and remains an open issue. IoT has a number of inherent characteristics, including scalability, heterogeneity and dynamism, which hinder access control. While most of the impact of these characteristics have been well studied in the literature, we highlighted “indeterminacy” in authentication as a neglected research issue. This work stresses that an indeterminacy-resilient model for IoT authentication is missing from the literature. According to our findings, indeterminacy consists of at least two facets: “uncertainty” and “ambiguity”. As a result, various relevant theories were studied in this work. Our proposed framework is based on well-known machine learning models and Attribute-Based Access Control (ABAC). To implement and evaluate our framework, we first generate datasets, in which the location of the users is a main dataset attribute, with the aim to analyse the role of user mobility in the performance of the prediction models. Next, multiple classification algorithms were used with our datasets in order to build our best-fit prediction models. Our results suggest that our prediction models are able to determine the class of the authentication requests while considering both the uncertainty and ambiguity in the IoT system.

Dedication

I would like to dedicate my thesis to “Amoo Ghasem” and children of Iranians who lost their fathers in fighting against ISIS while this research was being carried out.

Acknowledgements

This doctoral research was funded by the company Rasa and the Department of Computing and Informatics of Bournemouth University. I am grateful to them for the sponsorship of this work.

I would like to give special thanks to my first supervisor, Dr Alexios Mylonas, for his advice and encouragement during my PhD.

I am also grateful to Prof. Vasilis Katos for all his kind support while I carried out this research.

I would also like to thank Dr Emili Balaguer-Ballester for his valuable advice on part of my research.

I am deeply grateful to my brother, Dr Vahid Heydari, for his endless support in my academic life.

Finally, I would like to thank my parents, my wife and my little son for their support, love and patience.

Contents

1. Introduction.....	13
1.1 Thesis Motivation	13
1.2 Research Questions.....	14
1.3 Thesis Overview	16
1.4 Thesis Structure	16
1.5 Publications Arising from the work of the Thesis	17
2. Literature Review.....	19
2.1 An Introduction to Access Control in IoT	19
2.1.1 Access Control Models	21
2.1.2 Access Control Protocols and Standards.....	31
2.1.3 Access Control Language	35
2.1.4 Resilient Access Control Approaches.....	36
2.2 Indeterminacy in Authentication.....	41
2.2.1 Uncertainty.....	42
2.2.2 Ambiguity	48
2.2.3 Proposed Resilient Access Control Methods in IoT	52
2.2.4 Findings on Resilient Access Control Methods	54
2.3 Chapter Summary	55
3. Methodology	57
3.1 Overview.....	57
3.2 Architecture.....	58
3.3 The Process of Building an Indeterminacy-Aware Authentication Model	59
3.3.1 Attribute Selection	60
3.3.2 Dataset Synthesis	60
3.3.3 Uncertainty-Aware Model Building	61
3.3.4 Ambiguity-Aware Model Building.....	61
3.3.5 Validation.....	61
3.3.6 Model Selection	62
3.4 Chapter Summary	62
4. Dataset Synthesis	64
4.1 E-Health Exemplar – RASA.....	64
4.2 Threat Model.....	66
4.3 Synthesizing Process.....	67
4.3.1 User ID.....	68

4.3.2 Time	70
4.3.3 Location	72
4.3.4 Credentials	78
4.3.5 Authentication Decision.....	79
4.4 Los Alamos National Lab (LANL)	80
4.5 Chapter Summary	81
5. Uncertainty-Aware Prediction Model for Authentication.....	82
5.1 Preliminary Concepts.....	82
5.2 Prediction Models	86
5.2.1 Decision Tree	87
5.2.2 Random Forest	89
5.2.3 Support Vector Machine (SVM).....	92
5.2.4 Logistic Regression.....	94
5.2.5 Naïve Bayes	96
5.2.6 K-Nearest Neighbours (K-NN).....	98
5.2.7 Boosting Algorithms	100
5.2.8 Voting Classifier	102
5.2.9 Neural Networks	104
5.3 Discussion	110
5.4 Chapter Summary	114
6. Indeterminacy-Aware Prediction Model for Authentication	115
6.1 Trust-Based Analysis	115
6.2 Indeterminacy-Aware Prediction Models	121
6.2.1 Decision Tree	121
6.2.2 Random Forest	123
6.2.3 Support Vector Machine (SVM).....	125
6.2.4 Logistic Regression.....	126
6.2.5 Naïve Bayes	128
6.2.6 K-Nearest Neighbours (K-NN).....	130
6.2.7 Boosting Algorithms	132
6.2.8 Voting Classifier	134
6.2.9 Neural Networks	137
6.3 Discussion	141
6.4 Chapter Summary	145
7. Conclusion	146

7.1 Key Findings.....	146
7.1.1 Findings on IoT Adaptability of Access Control Models	146
7.1.2 Proposing Indeterminacy Factors for Authentication in IoT	147
7.1.3 DataSet Synthesis for Authentication	148
7.1.4 Handling Uncertainty in Authentication Using Prediction Models	149
7.1.5 Handling Trust in Authentication Using Prediction Models.....	149
7.2 Evaluation	150
7.3 Future Direction	151
7.3.1 Handling Indeterminacy in Authorization	151
7.3.2 Moving Towards Adaptive Model	151
Appendix A: Dataset Synthesis in MATLAB.....	153
A.1 Generating Data samples for IDs	153
A.2 Generating Data samples for Time	154
A.3 Generating Data samples for Location.....	155
A.4 Generating Data samples for Credential	157
A.5 Aggregating UVs to Label Datasets.....	157
Appendix B: Developing and Running Model on Raspberry Machine	160
B.1 Building Indeterminacy-Aware Prediction Models.....	161
B.2 Implementing Prediction Models on Raspberry Pi	166
B.2.1 Raspberry Pi	167
B.2.2 Building Prediction Model on Raspberry Pi	168
Bibliography	171

List of Figures

Figure 1.1: Thesis roadmap.....	16
Figure 2.1: A classification of access control models, methods, protocols and language.....	21
Figure 2.2: Graphical representation of evidence theory	46
Figure 3.1: Indeterminacy handling scheme in authentication.....	58
Figure 3.2: Proposed architecture for the indeterminacy-aware authentication model.....	59
Figure 3.3: Methodological steps.....	59
Figure 4.1: Location map of the RASA	65
Figure 4.2: Uncertainty matrix consisting of UVs	68
Figure 4.3: Visualizing generated data samples using a clustered column chart	70
Figure 4.4: UAs and associated UVs defined for three PoIs.....	76
Figure 4.5: UVs assigned to UAs (indicated by colour)	76
Figure 4.6: An example of calculating a UV for a data sample (X).....	77
Figure 4.7: Labelled uncertainty matrix.....	79
Figure 4.8: Distribution of authentication requests in LANL	81
Figure 5.1: The process of building a prediction model using supervised algorithms.....	84
Figure 5.2: The structure of a confusion matrix.....	86
Figure 5.3: a, b and c: ROC analysis for the decision-tree-based models.....	89
Figure 5.4: a, b and c: ROC analysis for the random-forest-based models.....	91
Figure 5.5: a, b and c: ROC analysis for the SVM-based models.....	93
Figure 5.6: a, b and c: ROC analysis for the logistic-regression-based models.....	95
Figure 5.7: a, b and c: ROC analysis for the models created by the Naïve Bayesian algorithm.....	97
Figure 5.8: a, b and c: ROC analysis for the models created by K-NN algorithms	99
Figure 5.9: a, b and c: ROC analysis for the models created by gradient boost and AdaBoost.....	102
Figure 5.10: a, b and c: ROC analysis for the models created by voting classifiers	104
Figure 5.11: (a) Perceptron and (b) MLP architectures.....	105
Figure 5.12: ROC analysis for prediction models in the low-mobility environment.....	111
Figure 5.13: ROC analysis for prediction models in the medium-mobility environment.....	111
Figure 5.14: ROC analysis for prediction models in the high-mobility environment.....	112
Figure 6.1: a, b and c: ROC analysis for the decision-tree-based models.....	122
Figure 6.2: a, b and c: ROC analysis for the random-forest-based models.....	124
Figure 6.3: a, b and c: ROC analysis for the SVM-based models.....	126
Figure 6.4: a, b and c: ROC analysis for the logistic-regression-based models.....	128
Figure 6.5: a, b and c: ROC analysis for the models created by the Naïve Bayesian algorithm.....	130
Figure 6.6: a, b and c: ROC analysis for the models created by the K-NN algorithms	132
Figure 6.7: a, b and c: ROC analysis for the models created by gradient boost and AdaBoost.....	134
Figure 6.8: a, b and c: ROC analysis for the models created by voting classifier.....	136
Figure 6.9: ROC analysis for prediction models in the low-mobility environment.....	142
Figure 6.10: ROC analysis for prediction models in the medium-mobility environment.....	143
Figure 6.11: ROC analysis for prediction models in the high-mobility environment.....	143
Figure 7.1: The thesis's main claim and its building blocks	151

List of Tables

Table 1.1: Publications arising from the thesis	18
Table 2.1: Evaluation of traditional and emerging access control models	27
Table 2.2: Analysis of proposed access control methods for IoT	31
Table 2.3: Summary of widely deployed authentication protocols	35
Table 2.4: The pros and cons of uncertainty handling theories	53
Table 2.5: Analysis of resilient methods proposed in the literature	69
Table 4.1: Statistical analysis of generated data samples for the users in a dataset	72
Table 4.2: Defined time slots and associated probabilities and UVs	74
Table 4.3: Parameters for three Gaussian PDFs based on three degrees of mobility	78
Table 4.4: Assigned uncertainty values for three states of credentials	88
Table 5.1: <i>Performance of the prediction model trained by three datasets (decision tree)</i>	90
Table 5.2: Performance of the prediction model trained by the low-mobility dataset (random forest)	92
Table 5.3: Performance of the prediction models trained by three datasets (SVM)	94
Table 5.4: Performance of the prediction models trained by three datasets (logistic regression)	96
Table 5.5: Performance of the prediction models trained by three datasets (Naïve Bayes)	98
Table 5.6: Performance of the prediction models trained by three datasets (K-NN)	100
Table 5.7: Performance of the prediction models trained by three datasets (gradient boost)	101
Table 5.8: Performance of the prediction models trained by three datasets (AdaBoost)	103
Table 5.9: Performance of the prediction models trained by three datasets (voting classifiers)	106
Table 5.10: Performance of neural network models trained by the low-mobility dataset	108
Table 5.11: Performance of neural network models trained by the medium-mobility dataset	109
Table 5.12: Performance of neural network models trained by the high-mobility dataset	109
Table 5.13: Aggregated performance of the prediction models	110
Table 5.14: Kruskal–Wallis test results, grouping variable: no. of hidden layers	113
Table 5.15: Spearman test results: correlation	113
Table 6.1: History profile of users in the low-mobility dataset	116
Table 6.2: History profile of users in the medium-mobility dataset	118
Table 6.3: History profile of users in the high-mobility dataset	120
Table 6.4: Performance of the prediction model trained by the new datasets (decision tree)	122
Table 6.5: Performance of the prediction model trained by the low-mobility dataset	123
Table 6.6: Performance of the prediction models trained by three datasets (SVM)	125
Table 6.7: Performance of the prediction models trained by three datasets (logistic regression)	127
Table 6.8: Performance of the prediction models trained by three datasets (Naïve Bayes)	129
Table 6.9: Performance of the prediction models trained by the three datasets (K-NN)	130
Table 6.10: Performance of the prediction models trained by the three datasets (gradient boost)	132
Table 6.11: Performance of the prediction models trained by the three datasets (AdaBoost)	133
Table 6.12: Performance of the prediction models trained by the three datasets (voting classifier)	135
Table 6.13: Performance of the new neural network models trained by the low-mobility dataset	138
Table 6.14: Performance of the N.N models trained by the new medium-mobility dataset	139
Table 6.15: Performance of the neural network models trained by the new high-mobility dataset ...	140
Table 6.16: Aggregated performance of the prediction models	141
Table 6.17: Kruskal–Wallis test results, grouping variable: no. of hidden layers	144
Table 6.18: Spearman test results: correlation	144

List of Acronyms

Abbreviation	Explanation
ABAC	Attribute Based Access Control
ABE	Attribute Based Encryption
ACDF	Access Control Decision Facility
ACL	Access Control List
AF	Activation Function
AI	Artificial Intelligence
ANN	Artificial Neural Network
AUC	Area Under the Curve
BIC	Bayesian Information Criterion
BLP	Bell-La Padula
BTG	Break The Glass
CapBAC	Capability Based Access Control
CART	Classification And Regression Trees
DAC	Discretionary Access Control
DoS	Denial of Service
ECC	Elliptic Curve Cryptography
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
IBL	Instance-Based Learning
IDS	Intrusion Detection System
IEP	Indeterminacy Estimation Point
IoMT	Internet of Medical Things
IPS	Intrusion Prevention System
IoT	Internet of Things
KDC	Key Distribution Centre
K-NN	K-Nearest Neighbors
LANL	Los Alamos National Lab
LDAP	Lightweight Directory Access Protocol
MAC	Mandatory Access Control
MLP	Multilayer Perception
MLS	Multilevel Security
Oauth	Open Authorization
OrBAC	Organization Based Access Control
PAP	Policy Administration Point
PDF	Probability Density Function
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
PKI	Public Key Infrastructure
PoI	Point of Interest
RAAC	Risk Aware Access Control
RADIUS	Remote Authentication Dial in User Service
RBAC	Role Based Access Control

ReLU	Rectified Linear Unit
RFID	Radio Frequency Identification
RoC	Receiver Operating Characteristics
SAML	Security Assertion Markup Language
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TN	True Negative Rate
TNR	True Negative Rate
TP	True Positive
UA	Uncertainty Area
UCON	Usage Control
UV	Uncertainty Value
V2V	Vehicle-To-Vehicle
VO	Virtual Organization
VoD	Video on Demand
WoT	Web of Things
WSN	Wireless Sensor Network
XACML	Extensible Access Control Markup Language

1. Introduction

In this chapter we identify the research gap and introduce the research problem. We also propose the related research questions. In accordance with the research questions, we present the main claim made by this work. Finally, we present an overview and the structure of this thesis in detail.

1.1 Thesis Motivation

The Internet of Things (IoT), which is defined as the “worldwide network of interconnected objects” [1], extends connectivity from human-to-machine to machine-to-machine communication. It also extends the platform to multiple domains including e-health. The IoT-based e-health market will reach \$136.8 billion worldwide by 2021. There are 3.7 million medical devices in use all over the world that are connected to the Internet and monitor various parts of the human body to inform healthcare decisions [2]. Among several challenges for e-health in the context of IoT, security is one of the most important. This is because a security breach in this domain often directly puts the lives of patients in danger, for example if patients’ healthcare records or devices are exposed or modified.

IoT offers large-scale integration of heterogeneous networks and devices, which, apart from the obvious opportunities, introduces great security and privacy challenges. These challenges are not new as they have been well studied in the relevant literature in different IoT domains (such as e-health, smart cities, smart grids) [3], [4], [5], [6], [7]. Of the various security challenges in IoT, access control is a crucial and ongoing challenge [8].

This thesis is motivated by the growing number of access scenarios in IoT in which indeterminacy factors need to be handled in order to maximize the tradeoff between “availability” and “Confidentiality-Integrity”. This holds true as the need for sharing resources in “agile” and “collaborative” projects between “ephemeral” parties is on the rise in IoT domains. Such a *resource sharing* in different IoT domains (e.g., e-health, smart grids, smart cities) is inevitable [9], [10]. For example, resource sharing in grid environments through virtual organizations (VOs) creates added value by improving performance with less investment. Besides advantages, it also presents disadvantages such as permission misuse and insider threats against VOs [9]. The same

threats can occur in smart cities where traffic information is shared or in V2V communication where two or more vehicles share their resources (e.g., information about their locations) [10].

Moreover, technical challenges at the physical and data link layers of IoT enabling technologies (RFID, WSN etc.) may exaggerate indeterminacy factors in access scenarios. These challenges matter. If a real-time access decision depends on information that is delivered at a delay or suffers from latency [11].

In all these circumstances, “incomplete information” about the consequence of the access request or “imprecise information” about the subject (requester) may lead to a state in which making access decision based on deterministic policy does not help the above-mentioned tradeoff.

The focus of this research is to introduce “indeterminacy”, a new and neglected challenge in IoT that affects the authentication phase of access control. Indeterminacy in IoT comes into play when an access decision needs to be made based on incomplete or imprecise information. In other words, indeterminacy is introduced in authentication when the information available about an authentication request is not sufficient or not precise enough to be used to make a correct access decision.

Both traditional and emerging access control models cannot precisely manage the tradeoff between availability and confidentiality-integrity in access scenarios including indeterminate factors. Moreover, resilient access control paradigms suffer from drawbacks that make them less effective as security measures.

To address this challenge, We propose a method to predict the future of an authentication request as well as the future of the subject who requests for authentication. Our method which benefits from machine learning suggest using data-driven prediction models to handle indeterminacy factors (uncertainty, ambiguity) in every authentication request.

1.2 Research Questions

This work is motivated by the following research questions:

- What is indeterminacy in authentication and how does it affect access control decisions?
- Are existing access control methods able to handle indeterminacy in authentication in IoT e-health?

- How can prediction models handle indeterminacy in authentication for e-health in IoT?

The main claim of this research is that an indeterminacy-aware prediction model can handle indeterminacy factors in authentication for scalable, heterogeneous and dynamic environments.

Derived from the above research questions, the aims and objectives of this research are as follows:

Aims:

1. Study indeterminacy in authentication in e-health in the context of IoT.
2. Evaluate the resilience of current access control methods in terms of dealing with indeterminacy in authentication.
3. Propose an indeterminacy-aware prediction model for the e-health domain in the context of IoT.

Objectives:

1. A review of the current state of the art (Aims 1, 2).
2. Investigate the requirements for authentication in e-health in the context of IoT (Aim 2).
3. Define the components of indeterminacy in the authentication phase of access control (Aim 2).
4. Investigate whether existing access control models deal with the defined components of indeterminacy in the authentication phase (Aim 2).
5. Develop, validate and evaluate an indeterminacy-aware prediction model for authentication for IoT in e-health (Aim 3).

1.3 Thesis Overview

This thesis is made up of seven chapters. Chapters 2 and 3 situate the thesis and provide the necessary literature for it. Chapter 4 presents the process of synthesizing datasets for the thesis. Chapters 5 and 6 present the process of building and evaluating indeterminacy-aware prediction models. Finally, in Chapter 7, we review the thesis. Figure 1.1 depicts the roadmap of this work.

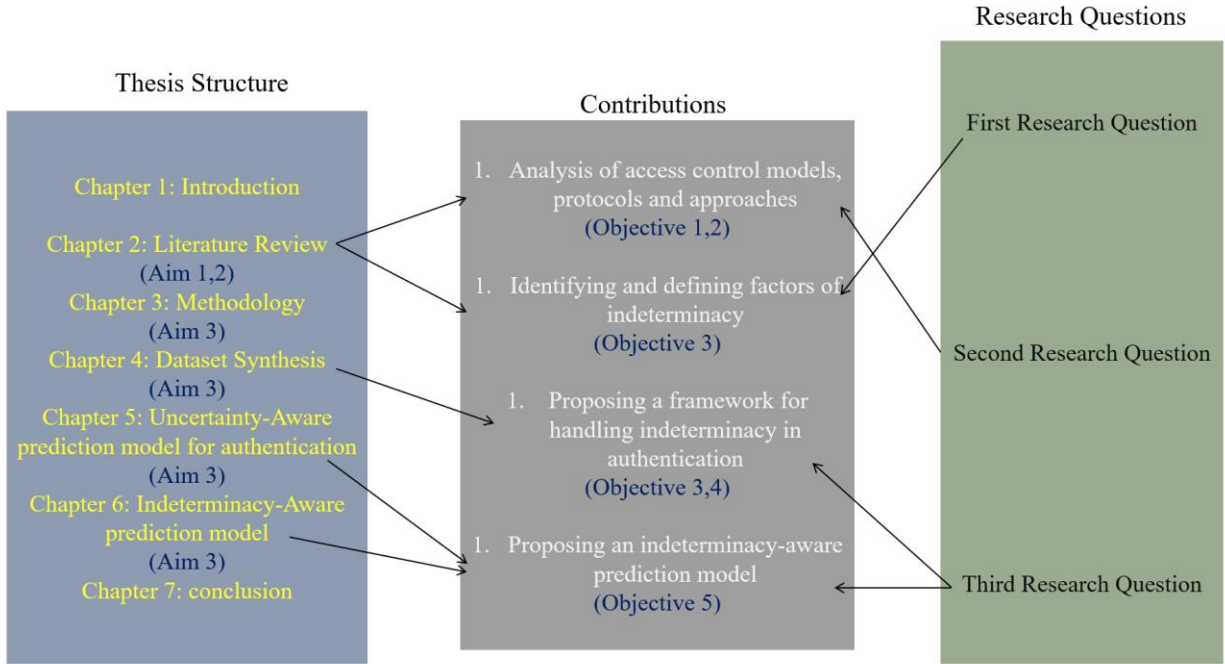


Figure 1. 1: Thesis roadmap

1.4 Thesis Structure

Chapter 2 reviews the current state of the art of access control in IoT. The chapter begins by reviewing access control models, protocols, language and approaches. In particular it reviews the authentication mechanisms included in them. It also takes a step back and reviews the preliminary concepts in access control. Because indeterminacy in authentication as introduced by this work is a new concept, in this chapter we first survey the concept and relevant studies, and then we review prevalent resilient access control models, including risk-aware and trust-based models. We also evaluate the current state of the art against the criteria defined by this work.

Chapter 3 introduces the methodology used within this thesis. According to the introduced challenges in the field of authentication, an automated, resilient and scalable authentication model is vital for IoT. To address this need, we propose a machine-learning-based prediction model, which will be discussed in this chapter.

Chapter 4 discusses the process of synthesizing datasets that were used for both training and the robust testing of our indeterminacy-aware authentication model. One of the big obstacles in conducting machine-learning-based research in the field of authentication is the lack of publicly available datasets. As a result, we generated datasets consisting with the required attributes in this work. This chapter presents the process of generating such datasets, with which we built the prediction models presented in Chapters 5 and 6. In this chapter, we also analyse the only publicly available dataset, called LANL, to evaluate our findings.

Chapter 5 shows the process of building prediction models for authentication. We explain all the classification algorithms applied in this work. Before introducing these algorithms, this chapter reviews the preliminary concepts and evaluation criteria used in our methodology.

Chapter 6 introduces our indeterminacy-aware prediction model on top of the prediction models presented in Chapter 5. In the course of this chapter, we consider the historical profile of users in terms of past successful authentication in addition to other attributes. As discussed in the previous chapter, we used classification algorithms to train and build our models. We expected that enriching our datasets with a new data attribute would enable us to build more accurate prediction models.

Chapter 7 concludes this research and proposes future work to extend the contribution made.

1.5 Publications Arising from the work of the Thesis

Table 1.1 shows all the publications arising from this research, grouped by the corresponding chapters of the thesis.

Table 1.1: Publications arising from the thesis

No.	Publication	Related Chapter
1	Mohammad Heydari, Alexios Mylonas, Vasileios Katos, Dimitris Gritzalis, Book Chapter: “Towards Indeterminacy-Tolerant Access Control in IoT”, in <i>Handbook of Big Data and IoT Security</i> , 2019, Springer	2, 3
2	Mohammad Heydari, Alexios Mylonas, Vasilios Katos, Emili Balaguer-Ballester, Vahid Heydari Fami Tafreshi, Elhadj Benkhelifa, <i>Uncertainty-Aware Authentication Model for Fog Computing in IoT</i> , 4 th International Conference on Fog and Mobile Edge Computing (FMEC), 2019	3, 4, 5
3	Mohammad Heydari, Alexios Mylonas, Vasilios Katos, Emili Balaguer-Ballester, Vahid Heydari Fami Tafreshi, Elhadj Benkhelifa, <i>Location-Aware Authentication Model to Handle Uncertainty in IoT</i> , 6 th IEEE International Conference on Internet of Things: Systems, Management and Security (IoTSMS), 2019	3, 4, 5
4	Mohammad Heydari, Alexios Mylonas, Vasilios Katos, Emili Balaguer-Ballester, Vahid Heydari Fami Tafreshi, <i>Uncertainty-Aware Authentication Model for IoT</i> , 3 rd International Workshop on SECurity and Privacy Requirements Engineering (SECPRE), 2019	3, 4, 5
5	Mohammad Heydari, Alexios Mylonas, Vahid Heydari Fami Tafreshi, Elhadj Benkhelifa, Surjit Singh, “Known Unknowns: Indeterminacy in Authentication in IoT”, <i>Elsevier Journal of Future Generation Computer Systems</i> , 2020	3, 4, 5, 6

2. Literature Review

In this chapter we review the current state of the art of access control in IoT. The chapter begins by reviewing access control models, protocols, language and approaches. In particular, we review the authentication mechanisms included in them. We take a step back and review the preliminary concepts in access control.

Because indeterminacy in authentication introduced by this work is a new concept, we first survey the concept and relevant studies, and then we review prevalent resilient access control models, including risk-aware and trust-based models. We also evaluate the current state of the art against the criteria defined by the new challenges introduced by this work.

2.1 An Introduction to Access Control in IoT

Access control is a mechanism that ensures that system resources can be used only by authorized entities based on a policy (RFC 4949 Internet Security Glossary [12]). An entity may be a user, program or process. Access control consists of the following functions [13]:

- *Authentication*, defined as a verification process to check whether the credentials of an entity are valid. In some texts, *identification* has been introduced as the process of identity verification, which should be completed before the authentication process.
- *Authorization*, defined as a process of granting an entity the rights to access and use a resource.
- *Auditing*, defined as the process of reviewing the access control system records and activities to detect any security breach. Auditing is necessary to ensure that the defined access policies are compliant with the operational security of the system.

An access control system has three basic elements [14]: 1) the subject, which is an entity that wants to access an object; 2) the object, which is a resource and a target of access requests by subjects; and 3) an access rights policy, which defines the way in which an object can be accessed by subjects. Any access control system should meet the main security objectives, known as CIA: *confidentiality*, by preventing unauthorized access to resources; *integrity*, by preventing resources

being modified without the required permission; and *availability*, by ensuring that the resource can be accessed and used on demand only by authorized subjects. Furthermore, an access control system may have some of the following characteristics, which are often used to evaluate the access control system [15]:

1. *Delegation*, which is the act of granting an access right (in full or in part) from one subject to another in the system.
2. *Revocation*, namely the act of removing from a subject the access right to a resource.
3. *Granularity*, i.e., the level of detail that can be used for making access decisions. If the required details are explicit and limited, then the type of granularity is referred to as *coarse-grained*. In contrast, *fine-grained* access control needs more detail, such as subject or object attributes, in order to make a decision about the access and govern it.
4. *Flexibility*, which is the ability of the access control system to adapt itself to different situations and to govern both planned and spontaneous interactions between subjects and objects.
5. *Scalability*, i.e., the ability of an access control system to be extensible in terms of the number of subjects, objects and access rights policies. Another dimension of scalability is the ability of an access control system to extend its structure and scope of operation.
6. *Lightweight*, which reflects the computational complexity or volume of network traffic that an access control mechanism imposes on the system.
7. *Heterogeneity*, which is defined as the ability of the access control system to be used in different domains, platforms, networks and technologies.
8. *Context-awareness*, namely the ability of the access control system to use contextual attributes of the environment, such as time and location, to make an access decision.

In the field of IoT, any access control system must be scalable, heterogeneous, lightweight and context-aware because of the characteristics of IoT itself.

We did a survey on the most widely used access control models (both traditional and emerging). We also investigated both resilient and non-resilient access control paradigms proposed in the

literature. According to the literature, resilient paradigms can be divided into three main categories: Break-The-Glass (BTG), Optimistic and the Risk-Aware. Moreover, the most widely used authentication/authorization protocols were analyzed against criteria discussed in 2.1.2. XACML as a standard for fine-grained and attribute-based access control policy language was also surveyed and discussed in this chapter. Figure 2.1 depicts a classification based on the surveyed models, paradigms protocols and the standard. The main motivation for such a survey was to investigate whether they are applicable to IoT.

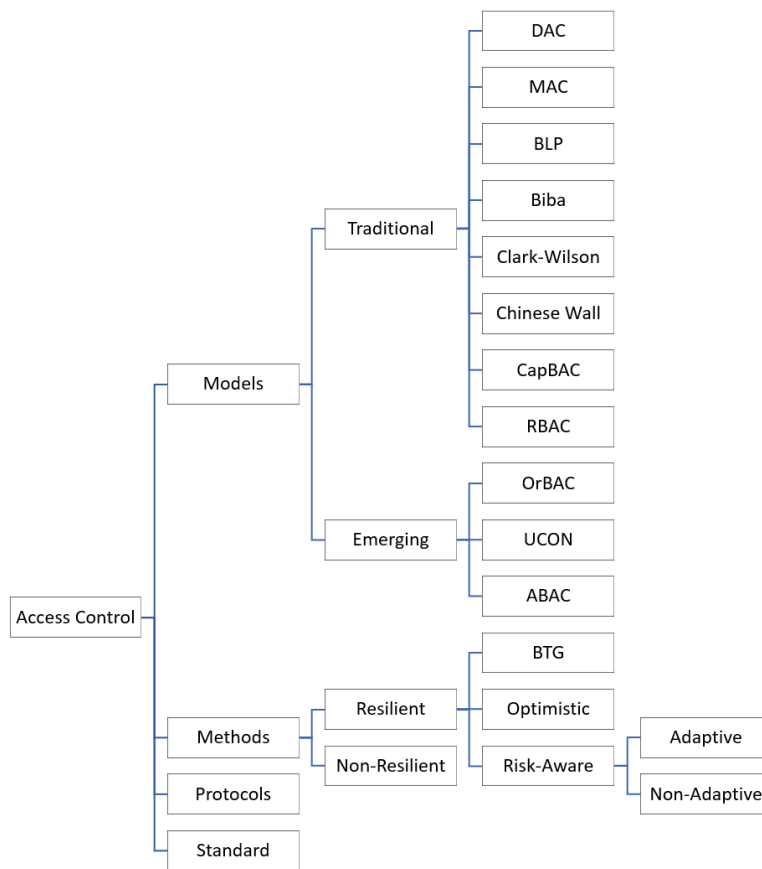


Figure 2. 1: A classification of access control models, paradigms, protocols and the standard

2.1.1 Access Control Models

Since Lampson's access matrix was introduced in the late 1960s, a number of access control models have been proposed. This Subsection briefly describes traditional access control models before moving on to emerging access control models. In a number of texts, Discretionary Access

Control (DAC), Mandatory Access Control (MAC), Biba, BLP, Clark-Wilson, Chinese Wall and Role Based Access Control (RBAC) have been classified as traditional access control models, while Attribute Based Access Control (ABAC) is considered a “relatively recent” access control model [16]. This work refers to ABAC and other models that have been proposed since ABAC, such as Access Control Based on Usage Control (UCON) and Organizational-Based Access Control (OrBAC), as *emerging* access control models. It also discusses their adaptability in a scalable, heterogeneous and dynamic environment such as IoT.

The *traditional* access control models comprise the following:

Discretionary Access Control (DAC): In DAC the access policy for any object is defined based on the discretion of the object’s owner [17]. The earliest implementation of DAC was the Access Control List (ACL), proposed in 1971. Modern operating systems such as Windows utilize this ACL-based approach. Contrary to Mandatory Access Control (MAC), in DAC a subject with certain access permissions is able to revoke its access rights and delegate them to another subject [18].

Mandatory Access Control (MAC): In MAC, which was proposed in 1973, the access policy is enforced by the system and access to a resource is granted only if the security clearance of the subject is greater than the security level of the object. A subject that has the clearance necessary to access the object cannot delegate its access to another subject or make any change in the access policy. In this model, the multilevel security (MLS) structure is defined by the system [19]. Although traditional MAC protects the confidentiality of information, it cannot protect the integrity of information since subjects with lower clearance can modify the objects that are accessible by subjects with higher clearance [19]. To address this problem, the Bell-LaPadula (BLP) model embodies two major principles: a) *no-read-up*, meaning that resources can be read only by subjects with clearance greater than or equal to the resource’s classification, and b) *no-write-down*, meaning that resources can be written only by subjects with clearance less than or equal to the resource’s classification. In contrast to DAC, another major drawback of MAC is that a subject cannot revoke its access rights or delegate them to another subject [20]. Security-Enhanced Linux (SELinux) and Mandatory Integrity Control are two examples of the use of MAC.

Bell-LaPadula (BLP): The BLP model was proposed in 1973 to focus on the confidentiality of data. For this reason, BLP enforces two main security policies known as *no-read-up* and *no-write-down*. *No-read-up* ensures that read access is granted if the security level of the subject must dominate the security classification of the object. *No-write-down* is defined for both “append” and “write” operations. In the former, *no-write-down* ensures that the security level of objects must dominate the security level of the subject. For the latter, it ensures that the security levels of subjects and objects are equal. BLP supports both MAC, by determining the access rights from the security levels associated with subjects and objects, and DAC, by governing access rights based on the access matrix.

The **Biba** model [21] was proposed in 1977 to ensure the integrity of data. For this purpose, Biba rules control the transfer of data between integrity levels. To meet this goal, subjects cannot read objects at lower integrity levels, which is known as the *no-read-down* policy. Also, subjects at lower integrity levels cannot get write access to the objects at higher integrity levels, which is known as the *no-write-up* policy.

The **Clark-Wilson** model [22] was proposed in 1989 to protect integrity, and it uses programs as a layer between users and data items. Users are authorized to execute a specific set of programs, and data items can be accessed only via those programs. The focus of this model is on the security requirements of commercial applications.

The **Chinese Wall** access control model was proposed in 1989 to avoid conflicts of interest when dealing with different subjects [23]. Conflicts arise when companies are in competition and want to access the same objects, or have accessed them in the past. The model can address the “conflict of interest” for both MAC and DAC. The Chinese Wall policy combines commercial discretion with legally enforceable mandatory controls.

Role-Based Access Control (RBAC): Ferraiolo et al. [24] proposed RBAC in 1992 to address the management complexity of DAC and MAC [25]. In RBAC, access is regulated based on the roles of the individuals within an organization. In other words, individuals performing specific roles can request access to specific resources that are necessary for this role. RBAC supports scalability and granularity and enforces the *principle of least privilege* [26]. According to the principle of least privilege, a subject should operate using the minimum set of privileges necessary to complete the task. Enforcing this principle mitigates the damage of unintended errors.

Furthermore, RBAC supports separation of duties by ensuring that at least two different individuals are responsible for carrying out the various steps of any critical task [27].

The *emerging* access control models comprise the following:

Attribute-Based Access Control (ABAC): In the ABAC model, when a subject requests access to an object, the decision will be made based on the subject's attributes, the object's attributes and the environment's attributes [28]. ABAC is widely used in the current Internet because it supports fine-grained access policies [29].

Capability-Based Access Control (CapBAC): CapBAC was introduced by Dennis [30] and governs access requests based on tokens. The subject must have a valid token to request access to a resource and the token must be tamper-proof. In this chapter, we classify those capability-based access control models as “emerging access control” that they benefit from using lightweight encryption algorithms such as elliptic-curve cryptography (ECC) to create Attribute-Based Encryption (ABE) access control models. In CapBAC, the subject needs to show the resource owner its token prior to performing corresponding resource request operations. This model has been used in several large-scale projects such as IoT@WORK.¹

Access Control based on Usage Control (UCON): In this model, an access decision is made using three factors [31]: a) *authorization rules*, which define the access policy based on the subject and object attributes, not only prior to the access but also during the access; b) *obligations*, which are responsible for verifying mandatory requirements for a subject before or during access; and c) *conditions*, which evaluate the current environment or system status. The most important aspect of this model is that if the access attributes change while the access is granted, and this change leads to a security breach, then the granted access is revoked, and the usage is cancelled. This happens because the subject and object attributes are mutable. Mutable attributes can change as a consequence of an instance of access [32].

Organizational-Based Access Control Model (OrBAC): OrBAC, proposed by Kalam et al. [33], extends the RBAC model in such a way that organization is considered as a new dimension. Contrary to DAC, MAC and RBAC, in this model policies are not restricted to static access rules,

¹ www.probe-it.eu

but also include contextual rules related to access permissions, prohibitions, obligations and recommendations.

We consider that, of the inherent characteristics of IoT, *scalability*, *heterogeneity*, *interoperability*, *dynamism* and *resource sharing* exaggerate the security challenges related to the field of access control. This holds true for the following reasons:

Scalability stems from the exponential growth in IoT that also results in an increased network connectivity requirement. According to Gartner, the number of Internet-connected devices will reach 20–50 billion by 2020 [34]. This exacerbates the security challenges in IoT by requiring more effort and resources from security controls (such as access control mechanisms) to address them [35].

Heterogeneity and *interoperability* in IoT derive from the different technologies and networks (such as radio-frequency identification (RFID), wireless sensor networks (WSN) and Global System for Mobile (GSM)) that exist in IoT. Thus, enabling seamless and secure integration of these different platforms is a challenge, as the degree of complexity increases dramatically when different technologies are merged to form a complex network. Similarly, interoperability brings new challenges to the field of data access control [36]. For example, in vehicle-to-vehicle (V2V) communication, moving from one geographical domain (e.g., the UK) to another (e.g., France) can cause data access problems, due to the interoperability issues between inter-domain public key infrastructures (PKIs) [37].

Dynamism in IoT stems from the fact that the interconnected things need to interact with each other in a real-time manner. Therefore, the need for an appropriate response to rapid changes in the physical world that are caused by these interactions poses new technological challenges, not only for access control but also for any context-aware services [38].

The above-mentioned access control models have been introduced to address a number of challenges in technological paradigms that preceded the IoT. We consider four assessment criteria to evaluate access control models. These criteria are considered according to the discussions on the access control in IoT from literature [9], [15] and chosen based on their impacts on the performance of the access control system in IoT access scenarios:

- **Dynamism:** If the access decision must change, because of changes in the subject, object or environmental attributes, after the access is granted, then the access control system is classified

as dynamic. But if the changes do not affect the access decision, then the access control system is static. Considering dynamism in IoT access control models is important, because of the rapid changes in contextual parameters that occur in this paradigm.

- **Scalability:** Scalability in access control must be evaluated according to three dimensions; that is, an access control has a) *subject/object (entities) scalability* if increasing the number of entities does not lead to an overhead in processing time or workload; b) *policy rules scalability* if increasing the number of access rules does not lead to overhead in terms of processing time or workload; and c) *extensibility* if it has the ability to extend its structure to cover more subsystems and domains. extensibility can be achieved through building a decentralized structure rather than a centralized structure in scalable environments such as IoT.
- **Heterogeneity/Interoperability:** In IoT, entities have dependencies and their workflows are tightly convergent, which increases complexity. For this reason, any access control breach in IoT can be more disruptive than in traditional networks [39]. Furthermore, as IoT is composed of different platforms, enabling technologies and domains, designing an access control model to regulate access inter-/intra-domains or technologies is a must.
- **Context-Awareness:** This refers to the ability of the access control system to use contextual attributes to make an access decision. Considering contextual parameters in an access decision brings flexibility in terms of tracking subject, object and environmental changes if these changes have impacts on the decision.

The above evaluation criteria uncover limitations in the models (both traditional and emerging), making them inapplicable to IoT. As summarized in Table 2.1, the traditional access control models do not support the above-mentioned criteria and thus cannot fit into IoT. With regard to the emerging access control models, *RBAC* does not satisfy the *interoperability* criterion [40], as it cannot support the definition of roles among heterogeneous networks with different platforms and domains. Furthermore, due to the inherent *scalability* of IoT, defining a vast number of roles and associated permission rules is impossible and leads to “role explosion”.

Nor does RBAC take *contextual* parameters into account during access decisions. Despite the advantages of ABAC, i.e., fine-grained access control, ease of use in a collaborative environment and flexibility, IoT adaptability of ABAC is hindered due to overhead. Specifically, applying ABAC in IoT is limited by computational overhead because its authorization process has high

complexity, as a result of the consideration of attributes of subject, object and environment in the access decision. Thus, applying ABAC in a highly dynamic, real-time environment such as IoT is infeasible because of the computational complexity that arises from the number of rules that rapidly increase with entities and contextual attributes, which may change frequently [41], [42], [43].

CapBAC is a coarse-grained access control model and does not consider contextual attributes, and therefore it cannot satisfy the *flexibility* criterion. Moreover, even when applying lightweight cryptography algorithms, such as ECC, using *CapBAC* brings overhead to the system in *scalable* scenarios. Another concern about *CapBAC* is the distribution of tokens in *heterogeneous* networks that are not straightforward. Also, the model fails the *interoperability* criterion as the model cannot be applied in a heterogeneous environment. The reason is that each domain has a dedicated public key infrastructure (PKI) and there is a trust issue in inter-domain interaction between these PKIs [44]. *UCON* has the same limitations as *ABAC* in terms of scalability and extensibility. Finally, *OrBAC* suffers from the same limitation regarding policy rules scalability as *RBAC*, as well as failing the *interoperability* criterion. The above evaluation, which is summarized in Table 2.1, highlights the need for a new model of access control that supports the above-mentioned characteristics for IoT domains.

Table 2.1: Evaluation of traditional and emerging access control models

Criteria Models	Scalability			Heterogeneity Interoperability	Dynamism	Context- Awareness
	Entities	Policy rules	Extensibility			
DAC	-	-	✓	-	-	-
MAC	-	-	-	-	-	-
RBAC	✓	-	-	-	-	-
CapBAC	-	✓	-	-	-	-
ABAC	-	-	✓	✓	✓	✓
UCON	-	-	-	✓	-	✓
OrBAC	✓	✓	✓	-	-	-

The literature includes a number of proposed access control models that are based on an extension of the above models. The proposed methods have tried to address the limitations of the reference models stated in Table 1. Jindou et al. [45] proposed an access control model based on RBAC for the Web of Things (WoT). This model gathers information from users' profiles on

social media platforms such as Facebook to create access policies. This, however, opens up a new type of trust and privacy challenges for all participants in the access control model. Barka et al. [46] integrated RBAC and WoT to build an access control model with a centralized architecture. Access decisions are made by the Access Control Decision Facility (ACDF) based on an RBAC policy. Because of its centralized structure, the model cannot cope with a distributed environment such as WoT. Liu et al. [47] have adapted the RBAC model to IoT using the Elliptic Curve Cryptosystem (ECC). In this method, IoT devices should be registered to a nearby trustworthy access point or gateway (termed as a Registration Authority). Furthermore, the authentication protocol suggested in this method is based on OpenID protocol.

Waleed et al. [48] proposed an access control model based on ABAC that incorporates trust and privacy into access policy to make it reliable in a collaborative environment. This model supports the privacy of subjects by authorizing certain access requests so that the purposes of access for both the subject and the object are the same. The limitations of the method include the following: a) if the contextual parameters have changed during the access time, the access decision is nonetheless consistent, and b) the proposed approach cannot be applied to distributed architecture, including P2P platforms. Kaiwen et al. [42] proposed a hybrid access control model based on RBAC and ABAC that can resolve the large-scale dynamic problem of IoT users. This model pre-assigns roles for entities (nodes/users) based on their property expressions. The model also presents a property rule policy language and a solution to the conflict with the redundancy policy. Kaiwen et al. [41] used the WeChat App as an example to illustrate the feasibility of this model. This model simplifies the complexity of traditional ABAC in rights allocation and policy management. However, it cannot deal with policy conflict and redundancy processing as the model still needs the administrator to manage roles and access policy. Harsha et al. [49] proposed an access control method based on ABAC for use in healthcare. The focus of this work is on providing both multilevel controlled access delegation and on-demand attribute revocation. Pussewalage et al. [49] suggested using assignment tokens and digital signatures to handle delegation and revocation. The complexity of using the token-based approach in conjunction with ABAC was not investigated by the authors. Furthermore, the structure of token distribution and validation schemes was not tested against forged intra-domain authorities, which may issue fake attributes and tokens.

Guoping et al. [50] proposed a method based on the extension of UCON. This method governs access by evaluating the degree of trust in the subject against the trust values of the object and the environment. If the trust value of the subject is in the range of the determined threshold for the requested object, then the access will be granted. The authors showed that their model works theoretically, but it is unclear whether it can work in a real-world scenario. Anggorojati et al. [51] proposed an access delegation method based on the context-aware CapBAC and identification. In this model, context information was added to CapBAC as a new dimension. This method used the concept of the federation in the Web for IoT by mapping identity to “thing”. Mahalle [52] proposed a novel method for authentication and access control based on the approach proposed by Gong [53]. In this method, verification of communication is done via its capability access. In other words, if any entity wants to communicate with another entity, communication is established after verifying the capability of the requesting entity. The proposed model uses a public key approach and is compatible with the lightweight, mobile, distributed and computationally limited nature of IoT. In this work, scalability, granularity and delegation were introduced as the main advantages of this method but the computational overhead of applying the model was not examined. Moreover, the interoperability of the proposed method in a heterogeneous environment such as IoT is still recognized as an ongoing challenge. Gusmeroli et al. [54] proposed another model based on CapBAC that uses a centralized approach for governing access control. The bottleneck for this method is that the majority of IoT devices have constrained resources and the overhead of the proposed method was not studied in this work. Yeh et al. [55] proposed a CapBAC-oriented access control framework for the e-healthcare domain. This method supports both fine-grained access control and revocation. The execution time for the encryption algorithms included in this method was compared with similar work to show its efficiency in terms of computational complexity. Although the proposed approach was proved theoretically, no experiment was conducted to show its efficiency in practice.

Li et al. [56] proposed a method that permits a user in a domain (e.g. smart city, smart grid) to send a message to a sensor in a domain that uses identity-based cryptography. The most important characteristic of this method is that it supports communication between heterogeneous environments. Furthermore, they showed that the computational cost of the sensor node in their method is reduced and energy consumption is consequently reduced. Patel et al. [57] proposed an energy-efficient access control method for IoT using elliptic-curve cryptography. The proposed

method was evaluated using the AVISPA² tool against attacks such as man-in-the-middle, reply attack and DoS. Even though the proposed method mitigated all these attacks successfully, one limitation of this work is that the method's efficiency was not considered. Ouaddah et al. [58] proposed a model based on an extension of OrBAC, which focuses on low power consumption. To meet this goal, part of the processing burden of the Policy Decision Point (PDP) was transferred to end-point devices to make the centralized structure more flexible. However, the overhead of the proposed method in terms of computational complexity and energy consumption was not proven experimentally. Moreover, the interoperability of the proposed scheme has not been studied. Sciancalepore et al. [59] proposed an access control framework based on OAuth 2.0, which consists of a WSN, client, gateway and authorization server. The authorization server passes the access request to the resource owner and generates the access token for the subject to which the access is granted. One of the challenges in this method is that direct communication between entities (without the presence of a gateway) is not possible due to the role of the gateway. The following conclusions arise from the study of the literature:

- In the approaches designed as an extension of RBAC, scalability in IoT was studied. Moreover, the interoperability issue was addressed through a Web-based interface (WoT).
- CapBAC-based approaches, even those using lightweight encryption algorithms (e.g., ECC), suffer from computational overhead in a scalable environment (e.g., cloud, IoT). Moreover, applying certificate-based authentication brings new challenges in terms of certificate validation and management in a heterogeneous environment such as IoT. In other words, moving from one domain into another makes interoperability a major concern for certificate validation.
- Although ABAC-based approaches bring flexibility by considering contextual parameters, managing a number of attributes in a hybrid model (with roles assigned and managed in RBAC) or (by using public key encryption, e.g., in ABE) introduces overhead and interoperability issues in IoT.

² <http://www.avispa-project.org>

Table 2.2 summarizes the evaluation results for the methods proposed based on the extension of the access control models.

The literature on e-health in IoT data has so far focused on two challenges that stem from the above-mentioned characteristics in IoT: i) data interchange between medical parties, and ii) wireless medical sensor communications [60]. Medical data interchange affects access control because of heterogeneity and interoperability issues in the IoT environment [61]. WSN is a key enabling technology in the Internet of Medical Things (IoMT) [62] and the offered wireless capability can be advantageous to patients and medical staff but at the same time pose threats to medical domains [63]. The security of individual wireless devices and applications from malicious access is also vitally important, in order to prevent falsification of information and impersonation with potentially fatal results [64].

Table 2.2: Analysis of proposed access control methods for IoT

Criteria Method	Scalability			Heterogeneity Interoperability	Dynamism	Context- Awareness
	Entities	Policy rules	Extensibility			
[45]	✓	-	✓	-	-	✓
[46]	-	-	✓	-	-	✓
[47]	✓	-	✓	✓	-	-
[48]	✓	-	✓	-	✓	✓
[41], [42]	✓	✓	-	-	-	✓
[49]	-	-	-	-	✓	✓
[50]	-	-	✓	-	✓	✓
[51]	-	-	✓	-	-	✓
[52]	✓	-	✓	✓	-	-
[54]	-	-	-	-	-	✓
[55]	-	✓	-	-	✓	✓
[56]	✓	-	✓	✓	-	-
[57]	✓	-	-	-	-	-
[58]	✓	✓	✓	-	-	-
[59]	✓	-	✓	✓	-	-

2.1.2 Access Control Protocols and Standards

This Subsection first introduces the most widely used access control standards and protocols, followed by a discussion of their applicability in IoT. OAuth, OpenID, SAML, RADIUS, LDAP

and Kerberos are investigated. These protocols and standards are widely used in different domains. OAuth is used by companies like Amazon, Google, Facebook etc. in order to allow the users share their account information with third party applications. OpenID as another open protocol has more than 1 billion accounts on the Internet³. RADIUS, LDAP and Kerberos are widely used in active directory (both Windows and Linux) and database access for the sake of identification and access management.

In order to evaluate the protocols involved in access control the following criteria, which are proposed in RFC 2989 and RFC 4962, are used:

1. Overhead: IoT devices are resource-constrained and thus any proposed access control protocol for IoT must be lightweight. To evaluate overhead, two different parameters are considered: a) *communication overhead*, which can be measured by the number of messages exchanged in a data access scenario per access request, and b) *lightness of data exchange format*, which affects the amount of control traffic required per access. Increased overhead may result in power consumption. For this reason, some works have suggested using more efficient protocols for communicating over IoT, such as LoRA [65].

2. Security of data-in-transit: The confidentiality of credentials that are sent over the network should be ensured. Otherwise, the protocol is prone to breaches of confidentiality of (credential) data-in-transit.

3. Architecture: The structure of access control protocols can be centralized or decentralized. As services in the IoT environment are decentralized and distributed, centralized architecture for access control protocol does not work efficiently if the protocol is deployed in a heterogeneous environment.

The aforementioned criteria will be used to evaluate whether the following protocols fit IoT:

Open Authorization (OAuth). OAuth⁴ is an open protocol used to establish a secure authorization over the Web. This protocol does not offer an authentication service. OAuth provides a method for clients to access server resources on behalf of a resource owner. It also allows end-

³ See <https://trends.builtwith.com/docinfo/OpenID>

⁴ <https://oauth.net/>

users to authorize third-party access to their server resources without sharing their credentials, using user-agent redirections [66]. To date, over one billion OAuth-based user accounts exist (e.g., as used in Facebook, Google and Microsoft user accounts).

OpenID. OpenID⁵ is a Web-oriented single sign-on protocol that is widely used by well-known companies such as PayPal and Amazon [67]. OpenID lets applications and site developers authenticate users without storing or managing credentials. Its most recent version, i.e., OpenID Connect, is designed on top of OAuth 2.0 to provide authentication. This version of OpenID supports optional mechanisms for robust signing and encryption.

Security Assertion Markup Language (SAML). SAML is an XML-oriented and open protocol developed by OASIS. It exchanges user authentication and authorization data among security domains. SAML 2.0 is the latest version of SAML. It has four components: 1) *assertions*, which express how identities are represented; 2) *protocols*; 3) *bindings*, which describe how SAML messages are transported over HTTP or other lower-level protocols; and 4) *profiles*, which consist of the participating bindings in a use case [68]. The assertion is the main component of SAML. There are three types of assertion in SAML: i) the *authentication* assertion validates the identity of the user; ii) the *attribute* assertion holds specific characteristics about the user; and iii) the *authorization* assertion indicates what kind of actions are authorized for each user.

Remote Authentication Dial-in User Service (RADIUS).⁶ RADIUS is an authentication network protocol that works in client/server network architecture to provide centralized access to networks (RFC 6929). The RADIUS server performs authentication, authorization and accounting (AAA) for users after it receives requests from the client. Authentication and authorization in RADIUS are bonded together. When the client device requests authentication from the server, the server replies with both authentication and authorization attributes. The security of the RADIUS protocol is based on MD5. RADIUS uses multifactor authentication. RADIUS uses User Datagram Protocol (UDP) over Internet Protocol (IP) with best-effort for delivery authentication services on the network. Moreover, RADIUS encrypts only the password, meaning that sensitive data on the

⁵ <http://openid.net/>

⁶ For more information, refer to RFC 2865 and RFC 6929.

user is sent in plain text over the network. Therefore, RADIUS is not recommended for a trusted environment.

Lightweight Directory Access Protocol (LDAP).⁷ LDAP is a centralized and remote authentication network protocol that is used for authentication and authorization [69]. It uses a lightweight version of the X.500 networking standard. LDAP encrypts all data exchanged between a client and server using Transport Layer Security (TLS). It can allow for single sign-on services in the network. Moreover, LDAP uses Transmission Control Protocol (TCP) over IP to form reliable communication over the network. Finally, LDAP, by itself, does not support multifactor authentication.

Kerberos.⁸ Kerberos is a network authentication protocol developed by Massachusetts Institute of Technology (MIT) to provide access to university resources in the 1980s. Kerberos authenticates clients to services in a distributed system. In the authentication phase, instead of a password a ciphertext known as a “token” is sent over the network. The token is generated by a third party known as a Key Distribution Centre (KDC). Mutual authentication, which is done using tokens, enables clients or servers to communicate with each other.

In addition to these de facto protocols, a number of studies have suggested new protocols. Braeken et al. [70] proposed a key agreement scheme based on symmetric encryption for IoT. The approach handles the verification of authentication for communications in which entities do not have prior trust relations. These protocols suffer from vulnerabilities. Jurcut et al. [71] proposed an approach to detect exploitable vulnerabilities in authentication protocols. The proposed method used a novel logic-based technique to describe circumstances under which a weakness in authentication protocols can be exploited.

Table 2.3 summarizes the comparative study between the above-mentioned authentication protocols based on the aggregated attributes that have been discussed in the literature review.

⁷ <https://ldap.com/>

⁸ <https://web.mit.edu/kerberos/>

Table 2.3: Summary of widely deployed authentication protocols

Spec	OAuth	OpenID	SAML	RADIUS	LDAP	Kerberos
Latest Version / Year	2.0 / 2012	OpenID Connect / 2017	2.0 / 2005	2013 (RFC 6929)	LDAP V3 1995	Kerberos 5 2018
Authentication	No	Yes	Yes	Yes	Yes	Yes
Authorization	Yes	No	Yes	Yes	Yes	No
Communication Overhead	Low communication overhead due to the use of JSON format	Low communication overhead due to the use of JSON format	High overhead due to XML parsing	Low in terms of server processing overhead	Low communication overhead due to using ASN 1.0, which is lighter than JSON	It imposes overhead in terms of control traffic and KDC administration in a scalable environment
Architecture	Decentralized	Decentralized	Centralized	Centralized	Centralized	Centralized
Security of Credential Data-in-Transit	Confidential	Confidential	Confidential	Only passwords are encrypted	Confidential	Username is sent in plain text, but passwords remain confidential

2.1.3 Access Control Language

Extensible Access Control Markup Language (XACML). XACML is a de facto standard and language to express ABAC-based access control policies, which is based on XML [72]. It is developed by OASIS.⁹ It uses *policy language* to define access policies and *request/response language* to describe access request queries and responses.

XACML has been widely used in modelling authorization part of the ABAC based access control. XACML can reflect the power of ABAC like scalability in authorization phase. As mentioned in Subsection 2.1.1 (Table 2.1), ABAC offers greater efficiency, flexibility and scalability than traditional access control methods. According to the findings of this research, there are at least four reasons for using XACML in modelling authorization phase:

- XACML is a standard that has been reviewed by a wide community of experts and users.

⁹ <https://www.oasis-open.org/>

- It offers a comprehensive framework to build policies and provides an expressive language that supports a diverse collection of data types, functions and combining algorithms that can be easily extended.
- XACML is sufficiently generic to be deployed in any environment. It makes policy management easier.
- It can be utilized in distributed contexts, which means that a policy can refer to other policies. In other words, XACML can combine the results from different policies into a single decision.

2.1.4 Resilient Access Control Approaches

Traditional access control approaches operate based on a set of static policy rules that govern access. In these approaches, access is granted if the corresponding rules are fired. Each rule consists of parameters to handle a condition in the predicted access scenario. The values of these parameters should be available if the rule needs to be fired. In such a system, if some of the rule parameters are missing then the system cannot handle the access scenario. As discussed earlier, scalable and heterogeneous environments such as IoT consist of data access scenarios in which making access decisions (e.g., authentication) based on the available information is not feasible because of a lack of information. In such a non-resilient access control system, the output leads to the access request being rejected. Therefore, a new paradigm is needed to make precise access decisions based on incomplete information and bring resilience to access decision-making. This type of access control is called “resilient access control”. Three paradigms have been proposed to achieve this goal [73], [74]: i) Break-The-Glass (BTG) Access Control; ii) Optimistic Access Control; and iii) Risk-Aware Access Control (RAAC).

2.1.4.1 Break-The-Glass (BTG) Access Control

Ferreira [75] proposed BTG to allow policy overrides. The aim of this model is to allow unanticipated access to be provided in unexpected situations. The main application of this method is in emergency situations in the healthcare system [76]. One of the most important problems with BTG relates to the scalability of policy overriding. Increasing the number of instances of policy overriding in a scalable environment such as IoT means that access monitoring and detection of

misuse become impossible [77]. Another concern about BTG arises from interoperability in heterogeneous environments. Specifically, in a heterogeneous environment consisting of different networks and platforms, one entity may be assigned an overriding policy to handle emergency conditions and at the same time be denied by an overriding policy from another domain. These conflicts highlight the vital need to accurately determine the overall decision [39].

2.1.4.2 Optimistic Access Control

In cases such as emergency healthcare services, the capability of an access control system to provide openness and availability is more necessary than confidentiality [78]. In this context, Optimistic Access Control has been proposed, and it assumes that most access requests will be legitimate. An optimistic approach permits the subject to exceed its normal access rights. Therefore, an additional control layer to protect the asset from misuse is recommended for Optimistic Access Control. This approach suffers from a lack of scalability in terms of policy rules. This holds true as implementing defence layers in a scalable environment needs additional resources and causes computational complexity. Optimistic Access Control also suffers from a lack of interoperability in heterogeneous environments, as defining exceptions for access scenarios that fall between two or more domains with conflicting interests is not straightforward [73].

2.1.4.3 Risk-Aware Access Control (RAAC)

The closest concept to uncertainty is “risk”. On one hand, risk is defined as a measure of the extent to which an entity is threatened by a potential circumstance or event and is typically a function of i) the adverse impacts that would arise if the circumstance or event occurred, and ii) the likelihood of occurrence [79]. On the other hand, uncertainty is defined as a lack of information about the likelihood of an event occurring. Therefore, “likelihood of event occurring” is common between these two concepts.

RAAC was proposed to assess the risk of the authentication request (in the authentication phase) or the risk of an action made by the subject using a permission rule (in the authorization phase) to determine whether access to a resource should be granted or the action should be permitted [80].

Risk assessment is defined as the process of identifying, estimating and prioritizing risks to organizational assets and operations (NIST SP-800). It enables the resource owner to obtain a view of existing security risks and their impacts. Risk assessment is composed of risk analysis and risk evaluation (ISO/IEC 27001):

- Risk analysis, one pillar of risk assessment, is responsible for identifying valuable assets and their associated vulnerabilities. It also uncovers the threats that may take advantage of those vulnerabilities. Estimating the damage that may be caused by these risks is the last part of the risk analysis process.
- Risk evaluation, another pillar of risk assessment, is defined as the process of rating risk exposures against criteria in order to determine the significance of each risk (ISO/IEC 27001). The risk evaluation process also prioritizes the identified risks based on their probability of occurrence and their impacts.

Risk assessment approaches can be classified into three categories [81]:

- *Quantitative*: This method uses objective measurement to calculate risk based on probability theory and statistical approaches. The output of such calculations can be expressed in an analytical form composed of percentages and probability values. The major drawback with the quantitative approach is that its calculations are lengthy and time-consuming and depend on the quality and detail of information collected (e.g., information about the value of assets, or sufficient information about the history of incidents) [82].
- *Qualitative*: This method is based on a non-numerical assessment in which predefined classes (threats, vulnerabilities and the likelihood of occurrence of threats) and associated values are used to assess risk. The values in these classes are expressed by linguistic variables (e.g., low, high, medium) or using range variables (e.g., from 1 to 5). Although qualitative approaches are widely used because of their simplicity, these methods suffer from a lack of measurable detail to support cost-effective decision-making. Another

drawback of qualitative approaches is that they are prone to error in comparison with quantitative approaches [83]. The reason is that the assessment is based on the knowledge and experience of the experts involved in the process. The third problem with qualitative approaches is that values that are assigned to either range or linguistic variables are not comprehensive enough to precisely reflect the risks of the system [81].

- *Hybrid*: In order to avoid the drawbacks of using each of the above-mentioned risk assessment approaches, a combination of them can be used in a hybrid framework. In this way, assessors benefit from the simplicity and speed of qualitative methods and the precision of quantitative methods for more critical assets.

Recently, a new taxonomy has been proposed for risk assessment methods [81] that divides them into three categories based on how they analyse risk: i) *asset-driven*: all methods that start risk assessment by identifying and evaluating the assets fall into this category; as a second step, they evaluate the risks associated with the assets identified from the first step; ii) *service-driven*: in this category, services are identified first and then risks associated with these services are evaluated; and iii) *business-driven*: in this class of risk assessment methods, business goals and associated processes should be identified first, and then the risks related to those business goals are assessed.

Another taxonomy for risk assessment methods is based on risk measurement. Risk-measuring methods fall into two categories [81]: i) *non-propagated*, where risk is measured regardless of its propagation impacts on the other risk parameters, and ii) *propagated*, where dependencies among the resources and their impacts on each other are taken into consideration to measure the risk.

Measuring risk in a propagated scheme enables the prediction of potential damage costs in a more accurate way than in non-propagated schemes. However, accurate information is needed about the dependencies among various resources (e.g., assets, processes) in the system.

The main task in RAAC is to quantify the level of risk based on the risk–benefit calculation. There are four types of risk–benefit calculations for access control [84], [85]: i) risk that focuses on information leaks that occur when a subject is granted access; ii) risk that focuses on the cost of resource unavailability for a system when access to a resource is denied; iii) the benefit obtained when an access is granted. The benefit comes from reducing the risk by giving the access; and iv)

the benefit of denying access that focuses on decreasing the chance of information leak occurred by denying access. One of the main challenges in RAAC is quantifying and calculating risk. RAAC models use different risk metrics that calculate the value of risk in access control systems, such as action severity, object sensitivity and benefit of access [86], [87]. There are five classes for calculating the risk [88]:

- *Class A*: The risk is calculated based on three parameters: 1) likelihood of the threat (L_T), 2) vulnerability of the asset (V), and 3) impact of the threat (I_T): score of the risk= $L_T \times V \times I_T$.
- *Class B*: The risk is calculated based on the security requirements of the asset. In this method the vulnerability related to the asset (V) and the impact of the threat (I_T) are involved in the calculation of the risk: score of the risk= $V \times I_T$.
- *Class C*: The risk is calculated in conjunction with financial loss considerations. To meet this goal, likelihood of the threat (L_T) and the average financial loss caused by the threat (F_T) are taken into consideration: score of the risk= $L_T \times F_T$.
- *Class D*: In this class, the risk is calculated only for critical assets. The method of calculating the score of the risk for critical assets is the same as in Class B.
- *Class E*: The concepts of threat and vulnerability are combined to create a new concept called “incident”. The score of the risk for this class is calculated using the likelihood of the incident (L_I) and the impact of the incident (I_I): score of the risk: $L_I \times I_I$.

According to [89], RAAC methods can be divided into two types, *non-adaptive* and *adaptive*. In *non-adaptive* methods, the calculated risk value for each access request remains unchanged even if any of the risk factor values change during the access session. This means that the access control mechanism cannot respond to change in the risk parameters after granting access. In contrast, in the *adaptive* approach the calculated risk value for each access request may change with changes in the risk factor values or the detection of abnormal behaviour. Therefore, adaptive RAAC continuously monitors activities to detect any suspicious events after access is granted.

In this section, we surveyed both traditional and emerging access control models, protocols, and standards to investigate whether they are applicable into scalable, dynamic, heterogenous and

context-aware environments like IoT. In the next section, we will focus on two neglected facets of indeterminacy (uncertainty and ambiguity) in authentication to see whether state-of-the-art methods can handle these challenges.

2.2 Indeterminacy in Authentication

Indeterminacy has not received the attention that it deserves as a challenge in IoT, compared to other challenges that are well studied in the relevant literature, such as scalability, heterogeneity, interoperability and dynamism [3], [4], [5], [6], [7]. However, as this work stresses, indeterminacy should be considered when making access control decisions in IoT. Otherwise, if a decision is based on deterministic rules regardless of the indeterminacy concept it can lead to a binary decision (Access/Deny), which does not fit into a dynamic environment such as IoT.

We consider that a subset of the above-mentioned inherent IoT characteristics exaggerate the indeterminacy in access control. Specifically, *dynamism* may result in indeterminacy because real-time tracking of the rapid changes (joining, disjoining, displacement of entities) is not easily achieved in a scalable environment such as IoT. Therefore, the lack of information caused by the inability to track these changes results in indeterminacy. Scalability can increase dynamism in such a way that having sufficiently complete information to make access decisions is impossible. Network and service dependency in a *heterogeneous* environment such as IoT can cause delay and latency in network delivery. If a real-time access decision depends on information that is delivered at a delay or suffers from latency, the decision will suffer from indeterminacy in access control. Finally, the inherent heterogeneity of IoT introduces different sources of data communication loss. For example, data may be lost in RFID for the following reasons [90]: 1) missing readings caused by tag collision, metal/liquid effects; 2) data inconsistency caused by reading data from various readers simultaneously; and 3) ghost data caused by frequency reflection in the reading area. The incompleteness and imprecision inherent in the above-mentioned sources are considered the main causes of indeterminacy [11].

According to Novák et al. [91], there are at least two facets for indeterminacy: *uncertainty* and *ambiguity*. In the context of authentication, we consider that uncertainty is caused by a lack of information about the likelihood of an incident occurring. Also, ambiguity is caused by a lack of

precision in the information required to make a decision. In the rest of this section, uncertainty and ambiguity in access control are discussed.

2.2.1 Uncertainty

For many years the term “randomness” was used to describe probabilistic phenomena. Knight and Keynes started using the term “uncertainty” for the first time in 1921 and 1936 respectively. They made great progress to break the monopoly of probability theory [92]. Since then, uncertainty has attracted attention in various disciplines (e.g., economics, management). As mentioned earlier, *uncertainty* is caused by a lack of information about the occurrence of an event. In other words, uncertainty refers to a state in which the following question cannot be answered in a deterministic way: *What event will occur?* According to Zeng et al. [93], three types of uncertainty exist:

- *Aleatory uncertainty*: an observed phenomenon that occurs randomly and is therefore impossible to predict
- *Epistemic uncertainty*: a lack of information and knowledge about the properties and conditions of the phenomenon
- *Inconsistent uncertainty*: conflicting testimonies. The notion of inconsistency here describes the case where there is “too much” information available but this information is logically inconsistent.

In the context of a system, uncertainty leads to a situation in which an analyst cannot describe or foresee an event in the system because of a lack of information about it. Therefore, the main motivation for measuring uncertainty is decision-making [94]. The relevant literature includes five main approaches to measuring uncertainty: i) probability theory, ii) information theory, iii) evidence theory [95], [96], iv) possibility theory [97], and v) uncertainty theory [98]. We review these five theories below:

2.2.1.1. Probability Theory (1657)

Traditionally, probability theory [99] has been used to analyse uncertainty. Probability is a single-valued measure of uncertainty, in the sense that uncertainty about the occurrence of an event, A , is represented by a single number, $P(A)$. As with different notions of uncertainty, different interpretations of probabilities exist. Two interpretations of probability are widespread in the field of risk analysis: a) the relative frequency interpretation (or classic probability). and b) the subjective or Bayesian interpretation.

Classic probability. In this interpretation, the probability is defined as the number of times an event, A , occurs during experimental trials divided by the total number of trials conducted. This process generates a fraction of successes, the “true” probability: $P(A)$. This pure type of probability is used to address *aleatory uncertainty*. The probability, $P(A)$, is defined as follows [100]:

$$\begin{aligned} &\text{if } A \in \Omega, \text{ then } 0 \leq P(A) \leq 1; \\ &P(\Omega)=1; \\ &\text{if } A_1, A_2, \dots A_i, \dots \text{ is a sequence of disjoint events from } \Omega, \text{ then } P(\cup_i A_i)=\sum_i P(A_i); \\ &P(\bar{A})=1 - P(A) \end{aligned}$$

Subjective probability (Bayesian interpretation, 1774) [101]. In this interpretation, the probability of an event, A , represents the degree of belief about the occurrence of event A . The subjective interpretation of probability is useful where the probability is a purely epistemic-based expression of uncertainty, based on the assigner’s background knowledge. In other words, within the Bayesian view, randomness itself is not considered a type of uncertainty. It is seen as a basis for expressing epistemic-based uncertainty. Subjective probability relies on the background knowledge (assuming K) that forms the basis of the assignment. To show the dependency on K , the conditional probability is defined as follows:

$$P(A|K) = \frac{P(A \cap K)}{P(K)}$$

Another important method for calculating conditional probabilities is given by the Bayes formula:

$$P(A|K) = \frac{P(K|A)P(A)}{P(K|A)P(A) + P(K|\bar{A})P(\bar{A})}$$

Imprecise probability (1920) [102] refers to the likelihood of an event with two probability values: *lower probability*, $\underline{P}(A)$, and *upper probability*, $\bar{P}(A)$, giving rise to a probability interval $[\underline{P}(A), \bar{P}(A)]$, where

$$0 \leq \underline{P}(A) \leq \bar{P}(A) \leq 1.$$

$$\Delta P(A) = \bar{P}(A) - \underline{P}(A)$$

Single-valued probability is a special case of imprecise probability, where both the lower and upper probabilities coincide, and is used to handle aleatory uncertainty.

2.2.1.2. Information Theory (Shannon Entropy, 1948)

Entropy measures the uncertainty inherent in the distribution of random variables. Consider a process with n possible outcomes $(1, 2, \dots, n)$ with probabilities p_1, p_2, \dots, p_n , respectively. The value of the entropy, $H(p)$, for the whole process is defined as follows:

$$H(p) = H(p_1, \dots, p_n) = - \sum_{i=1}^n p_i \log p_i, \text{ where } H(p) = [0, 1]$$

The value of entropy indicates the degree of uncertainty in the system. As the entropy increases, so does the uncertainty of a system. *Joint entropy* and *conditional entropy* are extensions that measure the uncertainty in the joint distribution of a pair of random variables and the uncertainty in the conditional distribution of a pair of random variables, respectively. The entropy of two or more processes can be measured using *joint entropy*. Given two random variables, X, Y , with joint probability, $P(X, Y)$, the joint entropy is calculated as follows:

$$H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log_2 p(x, y)$$

The *conditional entropy* $H(Y | X)$ for the given random variables (X, Y) is defined as follows:

$$H(Y | X) = \sum_{x \in X} p(x) H(Y | X = x)$$

Entropy can measure *aleatory* and *epistemic* uncertainties in the system by demonstrating the degree of randomness in the system.

2.2.1.3. Evidence Theory (Dempster–Shafer Theory, 1975)

The motivation for evidence theory is to represent and address situations where there is more information than in the case of a probability interval but less than there is in the case of a single specific probability distribution [95], [96]. This theory consists of two important measuring functions: i) the *belief measure*, $\text{Bel}(A)$, associated with preconceived notions, and ii) the *plausibility measure*, $\text{Pl}(A)$, associated with plausible information. The belief measure represents the degree of belief, based on the available evidence. A fundamental property of the belief function is that

$$\text{Bel}(A) + \text{Bel}(\bar{A}) \leq 1$$

Therefore, the sum of belief in the occurrence of event A and belief in the non-occurrence of event A is less than or equal to one. In other words, the difference $1 - \text{Bel}(A) + \text{Bel}(\bar{A})$ is called *ignorance*. When ignorance is 0, the available evidence justifies a probabilistic description of the uncertainty. The plausibility measure can be interpreted as the total evidence that any element such as Y belongs not only to A or any of its subsets, as with $\text{Bel}(A)$, but also to any set that overlaps with A . As a result, a fundamental property of the plausibility function is as follows:

$$\text{Pl}(A) + \text{Pl}(\bar{A}) \geq 1$$

As depicted in Figure 2.3, the relationships between plausibility and the belief measure are as follows:

$$\text{Pl}(A) = 1 - \text{Bel}(\bar{A})$$

$$\text{Bel}(A) = 1 - \text{Pl}(\bar{A})$$

The representation of uncertainty based on the above two measures falls under the framework of evidence theory proposed by Shafer in 1976. Whereas in probability theory, a single probability distribution function (PDF) is introduced to define the probabilities of any event, represented as a subset of the sample space, in evidence theory there are two measures of the likelihood, *belief* and *plausibility*. Also, probability theory imposes more restrictive conditions on likelihood because of this fact that the probabilities of the occurrence and non-occurrence of an event must add up to one. Evidence theory allows epistemic uncertainty and aleatory uncertainty to be addressed separately by a single framework. Indeed, the belief and plausibility functions provide the

mathematical tools to process information that is at the same time random and imprecise in nature. These two functions also allow different beliefs from various sources to be combined even though these beliefs are inconsistent. Therefore, evidence theory can also address inconsistent uncertainty. Figure 2.2 provides a visual interpretation of evidence theory. As depicted in Figure 2.2, the sum of belief and disbelief of an event occurring is not equal to 1 and the gap between these two is known as ignorance. This is the main difference between this theory and probability theory.

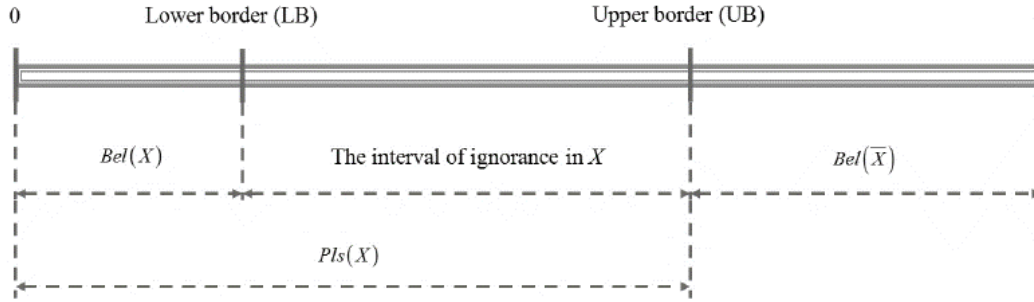


Figure 2. 2: Graphical representation of evidence theory

2.2.1.4. Possibility Theory (1978)

Possibility theory is a special branch of evidence theory [97]. Contrary to probability theory, it uses two functions, *possibility* and *necessity*, to describe uncertainty. The possibility function (Pos) has the following characteristics:

$$\text{Pos}(\emptyset)=0$$

$$\text{Pos}(\Omega)=1, \text{ where } \Omega \text{ is the universe of discourse}$$

$$\text{Pos}(U \cup V)=\text{Max}(\text{Pos}(U), \text{Pos}(V)), \text{ where } U \text{ and } V \text{ are disjoint subsets}$$

On the other hand, based on possibility theory, the necessity function, $\text{Nec}()$, is defined as follows:

$$\text{Nec}(U)=1 - \text{Pos}(\bar{U})$$

$$\text{Nec}(U \cap V)=\text{Min}(\text{Nec}(U), \text{Nec}(V)), U \text{ and } V \text{ are disjoint subsets}$$

As illustrated above, possibility theory is based on a pair of two measures, possibility and necessity, which are special forms of belief and plausibility measures from evidence theory. On the other hand, probability theory can be interpreted in such a way that the belief and plausibility measures coincide. The most significant difference between possibility theory and probability

theory is in the representation of total ignorance. Possibility theory and evidence theory represent total ignorance in the same way, using a unitary possibility distribution for the entire universe of discourse. In contrary to these two theories, probability theory uses a single distribution to represent the total ignorance.

2.2.1.5. Uncertainty Theory (2007)

Uncertainty theory was founded by Liu in 2007 and has subsequently been studied by many researchers [98]. This theory was proposed to model uncertainty based on *belief degrees*. Belief degrees cannot be treated as a subjective probability because this may lead to counterintuitive results. This problem derives from the fact that humans usually over-anticipate unlikely events, and therefore this makes the belief degree deviate far from frequency [103], [104]. Another difference between probability theory and uncertainty theory is that probability theory is a “product” mathematical system, whereas uncertainty theory is a “minimum” mathematical system. This difference implies that random variables and uncertain variables follow different operational laws. In other words, probability theory is a branch of mathematics for modelling frequencies, while uncertainty theory is a branch of mathematics for modelling belief degrees. According to Liu [105], in comparison with possibility theory, both uncertainty theory and possibility theory try to model belief degrees, where the former uses the uncertainty measure (belief degree), while the latter uses the possibility measure. Therefore, in the field of belief degrees the two theories are competitors.

There is no silver bullet to tackle uncertainty. Each of the five theories described previously has strengths and limitations.

2.2.1.6 Uncertainty in Authentication

To define uncertainty in authentication, it is essential to differentiate between its effects in the authentication and authorization phases of access control. It should be noted that, as discussed in Section 2, an instance of access control typically includes three phases: authentication (which encompasses identification), authorization and auditing. Auditing deals with the analysis of the

other two phases to detect security violations, and thus we consider that uncertainty cannot be considered in this phase. In this work we define uncertainty for authentication.

Uncertainty in authentication stems from the incompleteness of information regarding the likelihood of whether the acceptance of an authentication request leads to an incident. For instance, assume that “Alice” attempts to authenticate to a system. We also assume that authenticating her endangers the system (the access would expose an asset to a threat) with a probability of 60%. We present a formal definition for uncertainty in authentication as follows:

Definition: Given a set of authentication requests: $R=\{r_1, r_2, r_3, \dots, r_n\}$, a set of possible access decisions: $D=\{\text{Access}, \text{Deny}\}$, an access decision function: $F:R \rightarrow D$, and a set of possible outcomes for any access decision: $O=\{\text{Safe}, \text{Incident}\}$, the uncertainty of an authentication request is defined by the following conditional probability:

$$\begin{aligned} & \frac{P(\text{Incident} \mid \text{Access} \cup \text{Deny})}{P(\text{Incident} \cap (\text{Access} \cup \text{Deny}))} \\ &= \frac{P(\text{Incident} \cap (\text{Access} \cup \text{Deny}))}{P(\text{Access} \cup \text{Deny})} \\ &= P((\text{Incident} \cap \text{Access}) \cup (\text{Incident} \cap \text{Deny})) \\ &= P(\text{Incident} \cap \text{Access}) + P(\text{Incident} \cap \text{Deny}) \end{aligned}$$

2.2.2 Ambiguity

The terms “uncertainty” and “ambiguity” are used interchangeably in a number of studies [106], [107]. More importantly, the approaches proposed to address them are also used interchangeably. This stems from the fact that currently there is no clarity in the differentiation between these two concepts.

Aristotle was the first scholar to address ambiguity, which is caused by imprecise information [108]. The theories discussed in Subsection 2.2.1 fail to predict a system with imprecise or vaguely formulated information. This imprecise information may arise for several reasons, such as the complexity of the system [109]. We deal with ambiguity when the answer to the following

question is not clear: *What is the occurring event?* To model and present the ambiguity, the fuzzy set theory was proposed by Zadeh [110]. With regard to access control, we define ambiguity in authentication in the following Subsection.

2.2.2.1 Ambiguity in authentication

Ambiguity in authentication stems from a lack of precision in the information on the subject requesting to authenticate in the system. In other words, it aims to answer the question “to what extent can we trust the subject in order to authenticate it?”.

Traditionally, ambiguity has been evaluated with the use of *trust* analysis. There are several works in the literature on handling trust in access control. Nathalie Baracaldo et al. [111] proposed a method based on the extension of RBAC to handle trust in access control. To meet this goal, they assigned a trust level to each user based on his past behaviour in the context of the usage. In this method, trust is treated as a binary concept (0: totally untrusted and 1:totally trusted). Alessandro Armando et al. [112] proposed a model to handle authentication requests by balancing trust and risk. In this work, trust was defined as the level of confidence that the resource controller has on the access requester. The method suggested using credential-based analysis in order to handle the trust. For this reason, the value of trust is computed based on some attributes like user role, age of the user, and geographic location where a request created. The authentication request is accepted by this method if and only if the value for the incoming request is larger or equal to the value of the risk. Indrajit Ray et al. [113] proposed a formal trust-based access control based on the extension of RBAC. In contrary to [112], this method did not define trust as a binary concept. In order to measure trust, three types of trust were defined by this work: 1- user trust level (a value between 0 and 1 which the larger value means that it is less risky to grant access to that user), 2- role trust level that indicated the minimum trust value required for a user to be assigned to the corresponding role and 3- permission trust level which indicated the minimum trust value required by a role to be assigned a specific permission. In such a system, the role is authorized for permission if and only if its trust level is greater than the trust level of the permission. Mahdi Ghafoorian et al. [114] proposed a trust based RBAC model to govern access in the cloud. The suggested method considers three key elements to compute trust including resource owner, user (who need to access to the resource) and role. In this method, owner shares their resource based on the trust levels and reputation of the role. Therefore, users can access to the resource if their

trust level is above the trust threshold determined for the corresponding role. Authors considered two types of trust: direct trust and indirect trust. In direct trust, the value of trust for each user is calculated based on the feedbacks derived from history of the user's interactions. For measuring the indirect trust, the method benefits from a recommender system. Kamran Awan et al. [115] proposed a hierarchical architecture to handle trust in cross-domain communication network. In this method, nodes in the network can communicate with each other after evaluating their trust levels. In doing so, trustee and trustor need to know whether they belong to the same domain. If both reside on the same domain then the trustor sends the trustee information to the community server in order to evaluate its trust degree. Community server computes the trust level of the trustee based on three parameters including compatibility, honesty and competence. If trustee belongs to different community then the community server will query domain server about its trust level. Domain server was considered by this method to handle cross-domain communications. Ashish Singh et al. [116] suggested a trust-based access control based on the extension of Identity Based Access Control. In order to compute the level of user trust, the authors suggested using beta reputation approach. In the proposed method, the degree of trust for the user is computed by considering a number of parameters like time of access request, past behaviour of the user in the system and the location of the user. Parikshit Mahalle et al. [117] proposed a fuzzy based approach to handle trust in access control for IoT. In this work, trust was defined as a subjective and contextual value related to the user behaviour (trustee). Three parameters were defined by this work as the components of trust: *Experience* which is a track record of previous interactions related to the trustee, Knowledge about the trustee, and recommendation which is a summation of feedbacks from other users about the trustee. For each of the mentioned parameter, three linguistic terms (e.g. good, average and bad) were assigned. Moreover, corresponding fuzzy memberships were defined for the parameters. Final decision in the proposed method is made based on the result of the de-fuzzification process.

Farhana Jabeen et al. [118] reviewed trust and reputation in healthcare domain. According to this research, trust in healthcare can be classified into two branches namely as soft trust and hard trust. In the former, trust relationships are based on non-cryptographic mechanisms but in the latter, trust relationships among entities are based on cryptographic mechanisms. For the soft trust, degree of trust may change over time based on the trustee behavior.

Sadegh Dorri et al [9], and Ava Ahadipour et al. [119] classified trust analysis into two classes: i) *credential-based trust analysis*, and ii) *behavioural-based trust analysis*. In credential-based trust analysis, trust is established by verifying certain credentials. In credential verification, trust is established and access rights to different resources are granted, based on the access policy. This class of trust analysis is widely used in access control systems with static and predefined policies. Behavioural-based trust analysis uses the past behaviour as direct experience from the subjects to predict how the subject will behave in future.

Lessons learned from the literature; we come into the following conclusion about trust analysis:

- Soft trust analysis can be done using one of the following methods: 1- *direct* analysis can be achieved through direct experience while 2- *indirect* analysis consists of trustworthy peer experience collected during the period. 3- Soft trust analysis can also be done based on the *combination* of direct and indirect methods in a hybrid structure. Soft trust analysis has less overhead in comparison to hard trust analysis therefore, choosing soft method is recommended for IoT scenarios.
- Trust in IoT domains (e.g. E-Health) has a number of characteristics like 1- Asymmetry: Entity (A) trusts another entity (B), but not the other way around. 2- Partial Transitivity: Trust may or may not be transitive. 3- Context-sensitive: Trust establishment must be context sensitive. 4- Dynamic: Trust may change over time.
- Trust metrics can vary from one domain to another. In the context of access control, the following metrics have been used in the literature [120], [121], [122], [123]: a) number of authentication/authorization failures, b) subject anonymity, c) frequency of access requests, and d) degree of trustworthiness for subject and object.

According to the above considerations we choose soft trust method through conducting direct trust analysis (e.g. behavioral-based analysis). In doing so, we keep track of authentication history for all users and compute the value of trust based on the history. For a newcomer from which we do not have history profile, we can either assign a default value (e.g. 0.5) as trust score or assign the average value of trust in the system to it.

In authentication, behavioural-based analysis takes the history profile of successful and unsuccessful access requests for a given subject as a metric to calculate its trust value. Reputation-

based and recommendation-based trust are the subsets of this class of trust analysis that use cumulative knowledge about the past behaviour of a subject [124].

Assume that “Alice” attempts to authenticate, and, according to her access history, she has successfully authenticated seven times in the past, out of her 10 attempts. If the average number of authentication successes in the system is 60% or more, then Alice is classified as a “trusted” subject.

In the next Subsection, we will review the literature on resilient access control methods to find out whether proposed methods can address uncertainty and ambiguity in authentication.

2.2.3 Proposed Resilient Access Control Methods in IoT

A state-of-the-art review was conducted to answer the following research question: Can resilient access control methods handle indeterminacy in IoT? At the end of this Subsection, Table 2.4 summarizes the reviewed literature on resilient methods and indicates whether the proposed approaches handle uncertainty and ambiguity. Bijon et al. [125] incorporated the concept of risk awareness in RBAC. The role in the introduced RBAC model will be activated only if the total risk of its active roles does not exceed a given threshold. Furthermore, the threshold is determined dynamically in an adaptive manner. Baracaldo et al. [126] used trust and risk concepts in RBAC to deal with insiders. In this method, each user is assigned a trust level and each access permission is associated with a risk value. The risk of each role is calculated by the total risk of all direct and indirect permissions enabled by its activation. In this method, a role is activated if the user meets the minimum trust level required for that role. The value of the trust is determined based on the amount of risk exposed by activating the role. Dimmock et al. [127] proposed a method to enhance RBAC with trust and risk. To meet this goal, trust and cost evaluation measures are added to the OASIS policy language. This method introduces a risk evaluation expression language to calculate the risk based on the given values and make an access decision based on that calculation. Chen et al. [128] proposed an extension of the RBAC model to deal with the concept of risk in two dimensions: mitigation of loss and evaluation of likelihood. Evaluation of likelihood is accomplished by investigating the appropriateness of permission for a role and the trustworthiness and competency of the subjects. Mitigation of the loss is handled by assigning obligations to users to mitigate risks and deny requests with risks greater than a permission-specific threshold. Dos

Santos et al. [129] proposed an RAAC method for the cloud. In this method, if the subject of access is in the same cloud federation as the object, ABAC policies are enforced by the cloud service provider offering the object. Otherwise, risk policies are evaluated against the attributes of the subject and access is granted only if the risk is below a determined threshold. Dos Santos et al. improved their approach [130] and enriched their method by applying RAAC, not only for intra-cloud access decisions, but also for inter-cloud access decisions. Ricardo et al. [131] proposed a risk-aware framework to enforce RAAC policies in the cloud. This work is based on the extension of XACML and aggregates various risk factors to calculate the final value of the risk. Risk itself is measured based on the impact that access can cause. The calculated value is compared to a threshold to make an access decision. Atlam et al. [132] developed an adaptive RAAC model for IoT. This model accepts real-time attributes including user context, resource sensitivity, action severity and risk history as inputs and estimates the overall risk value associated with each access request. The major concern about this work is that the authors did not validate their proposed model. Dorri et al. [133] proposed an access control framework for the grid environment to address the misuse of resources in VOs. This method offers both risk and trust analysis in authorization to assess the subject's actions. The trust model uses feedback to calculate the user's trust degree in a probabilistic approach. On the other hand, the risk model is utility-based and uses the user's trust degree to calculate the probability of fulfilment of obligations. The proposed model was evaluated using simulation. The results show that it is scalable in terms of the number of entities, the number of policy rules and extensibility.

Table 2.4: Analysis of resilient methods proposed in the literature

Criteria Method	Scalability			Heterogeneity Interoperability	Dynamism	Context- Awareness	Indeterminacy-Awareness in Authentication	
	Entities	Policy rules	Extensibility				Uncertainty	Ambiguity
[125]	-	✓	-	-	✓	✓	-	-
[111], [126]	-	✓	-	-	✓	✓	-	-
[127]	-	✓	-	-	✓	✓	-	✓
[128]	✓	-	-	-	-	-	-	-
[129]	✓	✓	✓	✓	-	✓	✓	-
[130], [131]	✓	-	✓	✓	✓	✓	✓	-
[132]	✓	-	-	✓	✓	✓	✓	-
[133]	✓	✓	✓	✓	✓	✓	-	-

2.2.4 Findings on Resilient Access Control Methods

There are a number of widely used standards and methodologies for risk assessment, such as NIST-SP800,¹⁰ ISO/IEC 27005:2011¹¹ and IEC 62443-2-1.¹² Each describes a specific method for risk identification, evaluation, prioritization and mitigation. The adaptability of these risk assessment standards and methodologies in the IoT environment is controversial. Nurse et al. [134] argued that if IoT-related characteristics, such as scalability, heterogeneity and dynamism, are taken into consideration, the current risk assessment approaches are inadequate for IoT for the following reasons:

- *Limitation of periodic assessment for the IoT environment:* The current risk-based approaches are based on periodic assessment, and therefore cannot identify and evaluate significant changes in a highly dynamic system such as IoT, where there is a high degree of variability in system scale, dynamism and coupling.
- *Lack of knowledge of IoT entities:* Most of the current risk assessment approaches are based on knowledge of assets, threats, attack probabilities and potential impacts of threats. However, achieving sufficient knowledge of these parameters in IoT is extremely challenging due to the scalable and dependable environment of IoT.
- *Interoperability and dependency challenges:* Current risk assessment approaches are unable to assess all the processes associated with the assets and the inter/intra-connections that allow them to couple and operate; these introduce new areas of risk, which cannot be handled with current risk assessment methods.

Furthermore, most existing RAAC approaches rely on manual processes and thus are unable to provide a high degree of automation in risk assessment [135], [136], [137]. This lack of automation in risk assessment leads to the requirement for manual configuration from analysts, which is costly, error-prone, and vulnerable to social engineering attacks. Moreover, related work on RAAC that provides conceptual frameworks [138] or focuses only on domain-specific

¹⁰ <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf>

¹¹ <https://www.iso.org/standard/56742.html>

¹² <https://webstore.iec.ch/publication/7030>

solutions cannot be reused in other knowledge domains, and supports only restricted outcomes (i.e., permit or deny) [139]. Therefore, these approaches suffer from a lack of generalizability.

2.3 Chapter Summary

In this chapter, we have reviewed the current state-of-the art in related work contributing to the access control and the concept of indeterminacy in authentication. Summary points from this review are as follows.

- DAC, MAC, RBAC, ABAC, CapBAC, UCON and OrBAC were evaluated against the following criteria: scalability (in terms of entity, policy rules and extensibility), dynamism, heterogeneity/interoperability and context-awareness. According to the literature, none of these models can fully applicable to IoT but ABAC shows promising performance in terms of scalability (extensibility), dynamism, heterogeneity/interoperability, and context-awareness in comparison with other models.
- Aforementioned access control models benefit from deterministic set of policies that make them incapable of handling indeterminate access scenarios.
- Indeterminacy in authentication is defined as a state in which an authentication decision should be made based on “incomplete” and “imprecise” information.
- Indeterminacy has two facets: “Uncertainty” and “ambiguity (Trust)”.
- Uncertainty stems from the incompleteness of information regarding the likelihood of whether the acceptance of an authentication request leads to an incident.
- There are five theories in the literature for uncertainty handling including: probability theory, Information theory, evidence theory, possibility theory and uncertainty theory. These theories were discussed, and subjective probability theory was chosen to define and handle uncertainty in authentication due to the challenges of IoT scenarios like scalability and the need for less complexity.
- Ambiguity in authentication stems from a lack of precision in the information on the subject requesting to authenticate in the system. It is handled by “Trust” analysis methods.
- Soft trust analysis is applicable into IoT domains due to its less overhead in comparison with hard trust methods.

- Any trust-based analysis in the field of IoT may have the following characteristics: Asymmetry, Partial transitive, context-sensitive and dynamic.
- Current risk assessment standards and methodologies have limitations in evaluating “risk” in IoT environment due to the following reasons: 1- Limitation of periodic assessment for the IoT environment 2- Lack of knowledge of IoT entities and 3- Interoperability and dependency challenges.

3. Methodology

In this chapter we introduce the methodology used within this thesis. According to the introduced challenges in the field of authentication, an automated, resilient and scalable authentication model is vital for IoT. To address this need, we propose a machine-learning-based prediction model, which will be discussed in this chapter. We also consider considerations of the match funder in our methodology in order to build our prediction model working in e-health domain.

3.1 Overview

As discussed in Section 2, there are at least two facets of indeterminacy, including uncertainty and ambiguity [140]. Uncertainty in authentication stems from the incompleteness of information on the likelihood of whether the acceptance of an authentication request will lead to an incident. Authentication is affected by another element of indeterminacy called ambiguity. Ambiguity in authentication stems from a lack of precision in the information on the subject requesting to authenticate in the system. In other words, it aims to answer the question “to what extent can we *trust* the subject in order to authenticate it?”. To handle indeterminacy in authentication, we propose a machine-learning-based prediction model. This model is based on the extension of ABAC and works with three contextual parameters: time, location and credentials.

Our methodology consists of two parts. First, we built our uncertainty-aware prediction models using the mentioned attributes. Then, we applied behavioural-based analysis using the history profile of the user to improve the accuracy of our prediction models. Figure 3.1 shows the scheme of our methodology.

The rest of this section will introduce our proposed architecture and discuss the process of building prediction models.

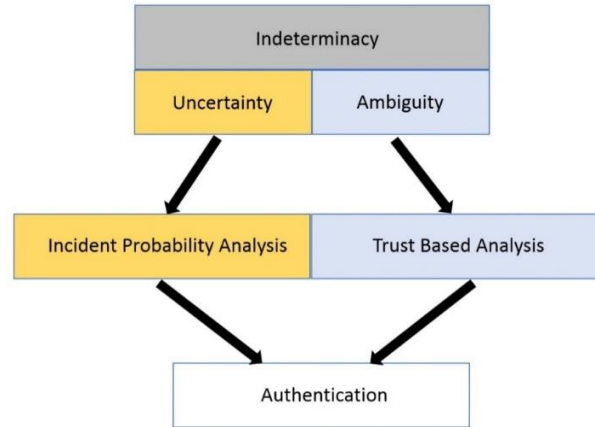


Figure 3. 1: Indeterminacy handling scheme in authentication

3.2 Architecture

We show the architecture of our methodology in Figure 3.2. As depicted, this architecture benefits from two building blocks: an “uncertainty-aware prediction engine” and an “ambiguity-aware prediction engine” to handle indeterminacy in authentication. The data flow model of the architecture is as follows:

- 1) A subject sends its authentication request to the Authentication Service Point (ASP). ASP is the interface between the system and the subject to forward the request and return the decision.
- 2) ASP sends the request to the Indeterminacy Estimation Point (IEP), which is responsible for requesting both uncertainty and ambiguity (trust) engines to calculate the uncertainty and ambiguity (trust) values associated with the authentication request.
- 3) IEP sends a request to the uncertainty-aware prediction engine to calculate the total value of the uncertainty associated with the authentication request.
- 4) The uncertainty engine returns the calculated value to IEP.
- 5) IEP sends a request to the trust engine to calculate the ambiguity (trust) value associated with the authentication request.
- 6) The ambiguity engine returns the calculated “trust “value for the ambiguity.

- 7) IEP calculates the value of indeterminacy based on the uncertainty and ambiguity values and sends it to ASP.
- 8) ASP returns the authentication decision based on the value of indeterminacy value to the user.

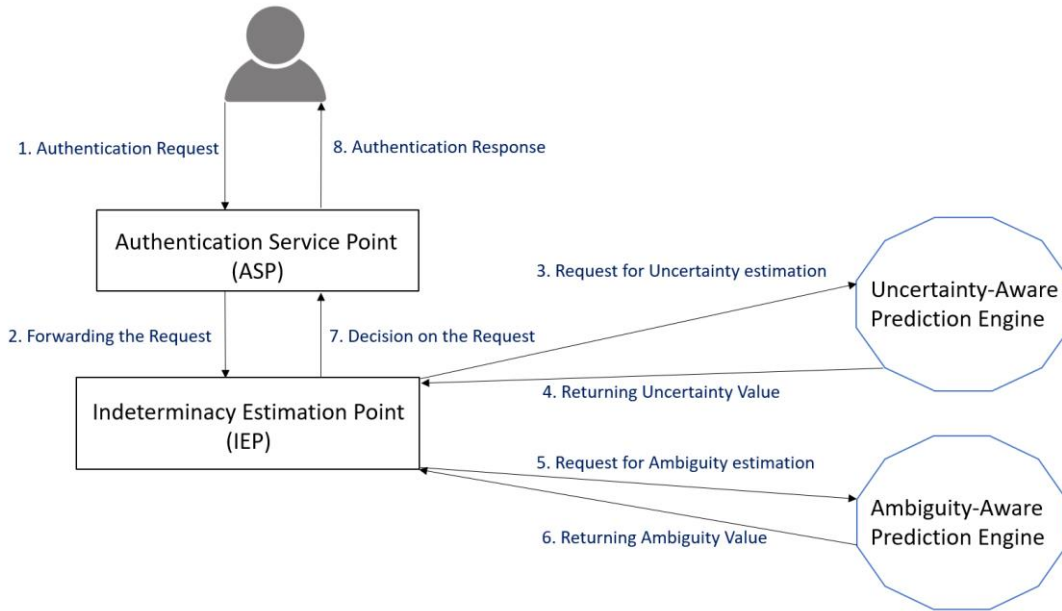


Figure 3. 2: Proposed architecture for the indeterminacy-aware authentication

3.3 The Process of Building an Indeterminacy-Aware Authentication Model

Figure 3.3 shows the steps involved in building our prediction models. We explain the steps below.



Figure 3. 3: Methodological steps

3.3.1 Attribute Selection

Since the model is an extension of the ABAC model, determining the attributes is the first task. Attributes for this model are the characteristics of the authentication request, such as time of the request, location/IP of the subject and the credentials of the subject. These attributes were chosen based on the literature. We discussed the attributes with match-funder in order to meet its requirements (for the e-health domain and for human-machine interaction). Chapter 4 discusses this step.

3.3.2 Dataset Synthesis

To the best of our knowledge, the authentication datasets consisting of the required attributes are not publicly available. As a result, in order to synthesize data samples for the attributes, each of attribute had to be represented by PDFs. To do this, each attribute had to be studied separately from two perspectives: a) determining the PDFs that reflect the likelihood of occurrence of the selected attribute, and b) determining the corresponding PDF for the probable impact of incidents caused by that attribute. Then, the generated values needed to be aggregated in order to build our dataset. Labelling the dataset was the last crucial task of this step. The dataset was labelled (Access/Deny) to be used by the supervised algorithms. Labelling was done using authentication policy and a fusion technique. The fusion technique was selected in accordance with the lessons learned from the review of uncertainty-related theories (i.e., probability, evidence, belief and uncertainty theories) in Chapter 2. The process of dataset synthesis is discussed in Chapter 4.

One of the considerations is determining the period of authentication history retention when we are dealing with real dataset. Identifying the period of retention for the authentication history depends on the type of the authentication system. In general, authentication history may be used for the sake of auditing purposes but in an authentication system in which a data-driven prediction model works at the heart of the system, determining such a period depends on the period of re-training. When the system hit by model drifting, re-training model with old data does not result in more accurate model. Therefore, the old history can be substituted with the latest dataset by which the prediction model was trained.

3.3.3 Uncertainty-Aware Model Building

After preparing a dataset consisting of three attributes (time, location and credentials), we needed to apply supervised algorithms to build our prediction models. Authentication is a matter of classification, and authentication requests were classified into the “Access” or “Deny” class. Thus, classification algorithms could be applied to the dataset in order to build the prediction models. The output of this step was uncertainty-aware prediction models. The prediction models were validated using the cross-validation method to avoid overfitting/underfitting. Chapter 5 presents the process of building uncertainty-aware prediction models.

3.3.4 Ambiguity-Aware Model Building

Besides the attributes that were chosen to build the uncertainty-aware prediction models, at this stage of the research one attribute seemed the most appropriate for use in trust analysis, i.e., the history of authentication requests for any subject. The reason for selecting it was that it has been widely used in trust-based analysis in the relevant literature. We envisaged that to perform trust-based analysis the ambiguity-aware prediction engine should retain the authentication history for each subject. The new data needed to be added to our dataset as a new attribute and the classification algorithms were applied to the new dataset. Our hypothesis was that adding new attributes to the dataset would increase the performance of the prediction models. Chapter 6 provides details on building ambiguity-aware prediction models and compares the performance of the models with the performance of the prediction models developed in Chapter 5.

3.3.5 Validation

According to the architecture depicted in Figure 3.2, proposed method is composed of two data-driven prediction models. These models are developed using classification algorithms. Prediction models are prone to two types of defects: 1- Bias and 2- Variance. Bias comes from the difference between the estimated performance of the model and its performance on unseen data and the variance which determines how much the performance of the model vary when the experiment is repeated.

In order to validate these models, cross-validation method was used to evaluate the generalizability of the prediction models. In this way, the dataset was divided into training and testing parts. The performance of the prediction models trained by the training part was tested by the testing part of the dataset.

Moreover, developed model will be evaluated using new datasets to investigate their performance in action.

These models will be also deployed on a testbed consisting of IoT entities (e.g. Raspberry Pi machines) in order to evaluate the performance of the system.

3.3.6 Model Selection

As discussed in the first step, a number of attributes were chosen to build the dataset. Involving more attributes in the dataset resulted in a more complex prediction model. One of the objectives was to reduce the number of attributes without losing the accuracy of the model. This was done using the Bayesian Information Criterion (BIC). BIC measures the relative model quality and assigns a score based on that measurement. BIC penalizes complexity in a model, where complexity refers to the number of parameters in the model. The model with the lowest BIC score is preferred. Furthermore, prediction models can be compared using a number of criteria in terms of performance (i.e., accuracy, precision, recall, F1). In short, the final prediction model must have the highest performance and the lowest complexity. The best-fit model was selected at the end of Chapter 5 and Chapter 6.

3.4 Chapter Summary

In this chapter, we have explained our research methodology for this dissertation. According to our methodology, we want to handle both uncertainty and ambiguity in authentication for scalable, heterogenous and dynamic environment. In doing so, we first need to synthesize our dataset due to lack of publicly available dataset for authentication. We chose three attributes for our dataset based on our findings and match-funder considerations. Selected attributes are time, location, and credential (username and password).

Our methodology benefits from supervised machine learning algorithms (classifiers) in order to build our data-driven prediction models. The models can determine the class of authentication requests (Access/Deny) by handling uncertainty and ambiguity.

To promote understanding of the readers, we have defined an exemplar (RASA). The exemplar reflects real characteristics of our research problem. We have also introduced validation methods which will be used to validate our results.

4. Dataset Synthesis

Datasets are used for both training and the robust testing of our indeterminacy-aware authentication model. One of the big obstacles in conducting machine-learning-based research in the field of authentication is the lack of publicly available datasets. As a result, in this work we generated datasets consisting of the required attributes. To better understand our methodology, we start this chapter by introducing an exemplar. Then we present the process of generating such datasets by which we built our prediction models as described in Chapter 5 and Chapter 6. We also analysed the only publicly available dataset, called LANL, to evaluate our findings.

4.1 E-Health Exemplar – RASA

To support the research approach, we introduce an exemplar. This exemplar helps to promote the understanding of research contributions among readers.

This exemplar is designed based on the specifications derived from match-funder (RASA company) infrastructure. We call this exemplar “RASA” in this work.

RASA decides to share data from part of its medical sensors with 60 of researchers. Users can access to the data through an aggregated node (gateway). The access is provided through WLAN (WiFi connection) provided on-site. WiFi routers are enhanced with WiFi location tracking system which can determine the location of the users and their mobilities.

Researchers are eligible to access to the data on-site. The site of RASA is located in a place represented by a map of area $2,000\text{ m} \times 2,000\text{ m}$. Figure 4.1 shows the location map of RASA. Three points of interest (PoIs) inside the area were defined. These are three buildings that the researchers are using during work hours. The first building (PoI_1) is the main building and most of the researcher are located in this building. We expect to have most of the authentication requests from this building. The building is located in $X=200\text{m}$ and $Y=200\text{m}$ on the map. The second building (PoI_2) is the seminar/meeting building. The researchers use this building for the meetings, seminars and workshops. We expect to have second-highest number of authentication requests from this building. The building is located in $X=600\text{m}$ and $Y=1000\text{m}$ on the map. The

third building (PoI_3) is library and we expect the third highest number of authentications from this building. The building is located in X=1400 and Y=1400 on the map. Researchers may send their authentication requests from each of these points of interest or they may send their request on the move between these points. Therefore, authentication system must be able to consider and handle requests for both fixed and mobile users.

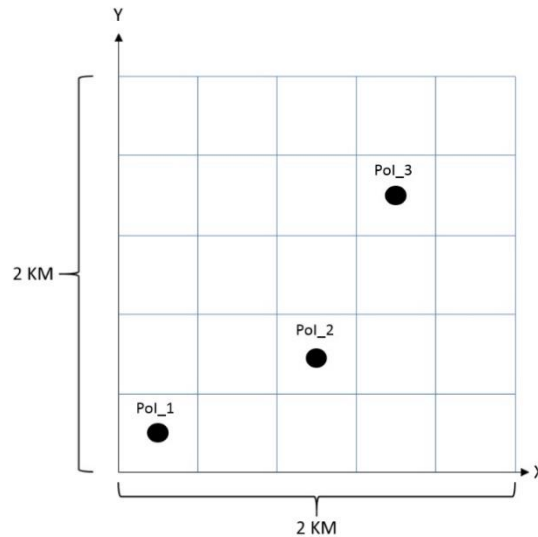


Figure 4. 1: Location map of the RASA

Researchers are using shared data during work hours (9-17). In some cases, they may request to access data before 9 and/or after 17 but we expect to have the most access requests during work hours.

Researchers are assigned and identified by ID (1 to 60). They are also assigned a pair of username and password individually.

Indeterminacy-aware authentication system needs to be designed and deployed on aggregated node (gateway). It must make an indeterminacy-aware authentication decision for each request based on the time of the request, location in which the user sends its request and the credential provided by the user.

We need to synthesize Datasets for this exemplar. The size of the dataset should reflect the scalability of the problem. Moreover, different degree of mobility needs to be considered in generating data samples in order to reflect the dynamism of the environment. This exemplar also

is composed of different network technologies like wireless sensor network (WSN) and wireless Lan and Ethernet (for fixed users). Such a heterogeneity makes this exemplar more realistic.

4.2 Threat Model

We can consider adversaries against the authentication systems discussed in our exemplar (Subsection 4.1) from two perspectives: user side, and authentication system side.

From user side:

1- ID related threat:

- Attacker may spoof user's ID to send authentication request to the system.

2- Time related threat:

- Users may send authentication request before 9AM and/or after 5PM.

3- Location related threat:

- Users may send authentication request from a location far from PoIs.

4- Credential related threats:

- Users may share credential with ineligible persons deliberately (Threat of insiders).
- Credential loss by user (e.g. users leave login credential in public place).

Physical threat:

- Physical threats against devices of users that lead to spoofing.

Communication Channel threat:

- Traffic between users and authentication system may be eavesdropped by attacker.

From Authentication system side:

- Authentication system running on aggregated node is threatened by DoS. (e.g. de-authentication attack).

Physical threat, DoS threat and channel related threats are out of the scope of this research.

4.3 Synthesizing Process

As discussed in Subsection 3.3.1, ABAC was selected as the reference model. As a result, we need to choose attributes as the authentication parameters for our data-driven model. According to the exemplar discussed in Subsection 4.1, credentials (username and password) are assigned to the users. Furthermore, as discussed in RASA exemplar, each user is assigned and identified by an ID. In addition to these parameters, match-funder asked for spatio-temporal parameters to be involved in the authentication process.

According to the literature the chosen attributes must have the following characteristics [141]:

- be atomic-valued, i.e., have a single-value attribute
- be a non-entity attribute (the attribute does not include another entity as its value)
- be a contextual attribute
- be independent (the attribute does not have any intersection with the other attributes).

As a result, based on the scope and assumptions of the proposed model discussed in Chapter 3, and the considerations discussed in Subsection 4.1, the required dataset needs to have the following attributes: i) ID of the user, ii) time of the request, iii) location of the request, and iv) credentials provided by the user.

The process of generating values for each of the above attributes started by determining their corresponding PDFs. In real-world scenarios, these attributes are derived from stochastic processes and therefore they follow a PDF or a mixture of PDFs. Finding the best PDF to describe the behaviour of these attributes helped us to synthesize a dataset similar to a real-world dataset. In doing so, we used PDFs based on similar works in the state of the art. The next step of constructing our datasets was assigning uncertainty values (UVs) to the generated data samples. We followed a logic for such an assigning to construct our datasets consisting of UVs. The final dataset can be depicted as an uncertainty matrix, as shown in Figure 4.1.

	User ID	Time	Location	Credentials
Authentication Request #1.....	12.....	0.6.....	0.32.....	0.70.....
Authentication Request #2.....	22.....	0.1.....	0.28.....	0.05.....
Authentication Request #3.....	51.....	0.2.....	0.72.....	0.70.....
Authentication Request #4.....	13.....	0.1.....	0.43.....	0.95.....
Authentication Request #5.....	7.....	0.5.....	0.78.....	0.05.....
Authentication Request #5000.....	43.....	0.1.....	0.53.....	0.05.....

Figure 4. 2: Uncertainty matrix consisting of UVs

In this work, the size of the datasets was determined to be 5,000. In other words, we assumed that the datasets consisted of 5,000 authentication requests for which the above-mentioned attributes needed to be synthesized. The size of the dataset should be determined in such a way that it avoids *overfitting* or *underfitting* (discussed in Chapter 5) and also reflects the scalability of the scenario in terms of entity. In the rest of this chapter, the process of generating data samples for these attributes is discussed.

4.3.1 User ID

As mentioned in exemplar (RASA), we assumed that 60 users were participating in our authentication scenario (IDs are identified by numbers:1,2,...,60). In order to find the corresponding PDFs, relevant studies were considered. The pattern of online activities was comprehensively studied in the literature. The most highly cited works are as follows: Lada et al. [142] analysed the activity of the online users. They found that access requests follow power law distribution (Zipf's law). In another study, Chao Wang et al. [143] analysed two datasets and concluded that users in video on demand (VoD) systems can be distinguished by their individual access requests, according to the drift power law distribution. Gutierrez et al. [144] studied user behaviour on social media (e.g., Twitter) and found that it follows a power law PDF with respect to the number of unique users participating in the conversation. Cha et al. [145] confirmed that the activity (access to the resources) of users on popular social media services such as YouTube follows power law PDFs.

According to the above studies, we used *power law* PDF in order to generate data samples for our users in all datasets. In doing so, we implemented power law PDF in MATLAB version 2018a. The formula of the corresponding PDFs is as follows:

$$P(x)=Cx^{-\alpha}(1)$$

where α is a constant parameter of the distribution known as the exponent or scaling parameter. The scaling parameter typically lies in the range $2 < \alpha < 3$. Moreover, C in the above function is the normalization constant. Table 4.1 statistically summarizes the generated data. The heading of the table indicates the ID of the users and the white column shows the frequency of their authentication requests in the synthesized dataset.

Table 4.1: Statistical analysis of generated data samples for the users in a dataset

ser ID	No. of Authentication Requests	User ID	No. of Authentication Requests
1	361	31	55
2	314	32	57
3	283	33	40
4	234	34	55
5	247	35	47
6	194	36	42
7	185	37	46
8	151	38	51
9	124	39	29
10	135	40	38
11	128	41	41
12	133	42	51
13	115	43	36
14	99	44	28
15	102	45	31
16	109	46	38
17	93	47	36
18	81	48	42
19	88	49	35
20	78	50	33
21	80	51	26
22	74	52	39
23	65	53	42
24	60	54	23
25	58	55	38
26	53	56	27
27	46	57	30
28	59	58	30
29	57	59	21
30	61	60	26

By plotting the generated data using a clustered column chart (Figure 4.2), a typical diagram of power law distribution is achieved.

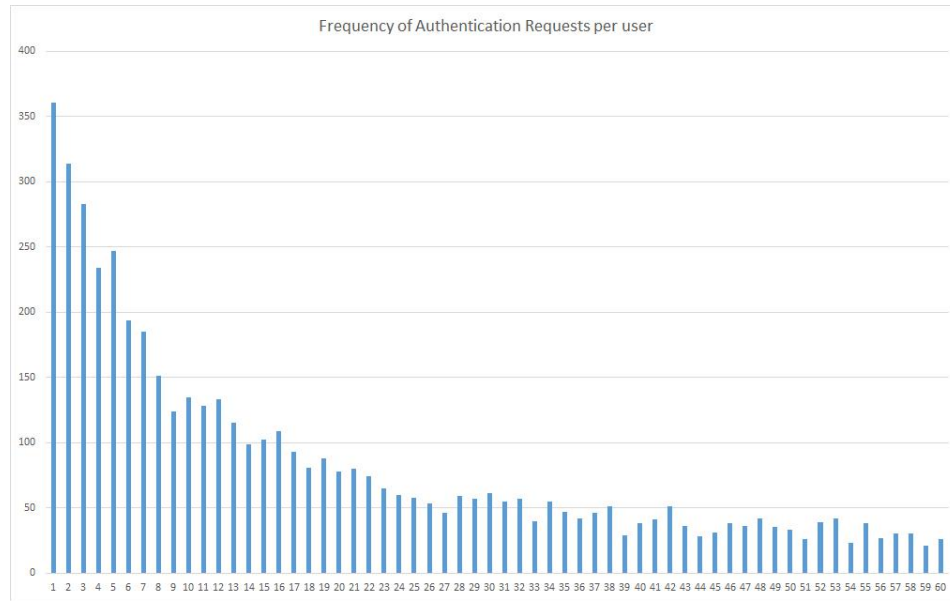


Figure 4. 3: Visualizing generated data samples using a clustered column chart

We synthesized three set of users for our three datasets (low-, medium- and high-mobility datasets).

4.3.2 Time

The pattern for the time of authentication requests depends on the business model of the service in which the authentication process is embedded. For services that are deployed to be accessible 24 hours a day, seven days a week (e.g., email services) generally no restriction is defined for the sake of access in terms of time. In such services, the timing of the authentication requests follows *uniform* distribution.

As discussed in our exemplar (Subsection 4.1), we defined an authentication scenario in the field of e-health. In order to make the scenario more challenging in terms of dataset synthesizing, we considered a service that is mostly demanded during a specific time period, such as working hours (e.g., 9 am to 5 pm), so we had to take those time preferences into consideration and find the corresponding PDFs. According to our assumption for this case study, the majority of users

send authentication requests during work hours (9 am to 5 pm) and the number of requests before 9 am and after 5 pm gradually falls. We also supposed that the number of requests between 12 pm and 1 pm decreases due to breaks/lunchtime.

Based on the above-mentioned considerations, we broke the times of the authentication requests up into 11 time slots. We also assigned weights in terms of probability values to these time slots. These weights reflect the likelihood that authentication slots will be made during each of these time slots. The logic behind these values is based on the business model of the case study. For example, the probability of receiving an authentication request from 9 am to 12 pm or 1 pm to 5 pm is higher than during the other time slots. We have also checked these values against the history of access provided by match funder.

In order to generate the values for the timing of authentication requests, we applied two PDFs. First, *multinomial* distribution was used to randomly choose the time slot from which a request comes. In the process, we used the assigned weights that were discussed earlier. Next, a *uniform* distribution was similarly applied to randomly generate the time of the request within the nominated time slot. The multinomial PDF was applied in MATLAB using the *mnrnd()* function and the uniform distribution was also applied using the *randi()* function in MATLAB.

We also defined and assigned a UV for each time slot. In doing so, we determined values in such a way that authentication requests made during work hours were supposed to be less prone to security incidents than any request made outside work hours, and therefore the value of uncertainty is lower during work hours. UVs for requests outside work hours increased gradually. We also assumed the lowest values of uncertainty for all authentication requests during work hours because of the potential threat of *insiders*. Finally, UVs for the generated request times were assigned based on the records in Table 4.2.

For different authentication scenarios, time slots, weights (probability of the time slots) and UVs may differ. Our methodology is assumption independent so the prediction models that will be discussed in Chapter 5 and Chapter 6 can learn the differences between and predict the classes of the authentication requests for different times, locations and credential patterns.

Table 4.2: Defined time slots and associated probabilities and UVs

Time Slot	Weight (Probability)	Uncertainty Value
[1–5)	0.005	0.80
[5–7)	0.006	0.75
[7–8)	0.01	0.60
[8–9)	0.04	0.50
[9–12)	0.35	0.10
[12–13)	0.10	0.20
[13–17)	0.40	0.10
[17–18)	0.06	0.40
[18–19)	0.02	0.50
[19–23)	0.007	0.70
[23–1)	0.002	0.90

4.3.3 Location

There is increasing need to consider mobility in authentication because the number of security and privacy incidents they cause is rapidly increasing [146]. For this reason, our approach considers the location the authentication request comes from to make more accurate authentication decisions. Therefore, it can handle the uncertainty of mobile users as well as fixed users in authentication.

The mobility of users has been well studied in the literature. Ekman et al. [147] analysed the mobility of users and revealed that it follows a Gaussian PDF. Chandrasekaran et al. [148] applied a mixture of Gaussian PDFs to model the mobility of users. Keränen et al. [149] discussed how the mobility of users follows a Gaussian random walk model. Shin et al. [150] proposed a location-based access control system. This work used Gaussian distribution to model user mobility. Sistla et al. [151] proposed a data model for mobile objects with uncertain location using a Gaussian PDF. In accordance with these highly cited works, we chose a Gaussian distribution to generate data samples for the location of the authentication requests.

The location defined for the scenario was in our exemplar (Figure 4.1) which is represented in a map of area $2,000 \text{ m} \times 2,000 \text{ m}$. The map was defined by match funder based on its campus located in E-Health city. We defined three points of interest (PoIs) inside the area. Figure 4.3 shows the map and corresponding PoIs. The task was to generate data samples that specify the

location of mobile users in a two-dimensional grid (X: longitude, and Y: latitude). In order to generate the samples, we applied a mixture of Gaussian PDFs.

Considering three PoIs makes our case study more challenging. The number of PoIs may vary from one case study to another. According to the assumptions, our PDF consisted of three Gaussian factors (because of our three PoIs), each of which has a weight, and each PDF belongs to one PoI respectively:

$$GT = \alpha G1 + \beta G2 + \gamma G3 \quad (2)$$

We expected most of the authentication requests to be sent from or around PoI_1. Data samples for the first PoI were generated using G1. Therefore, the magnitude of the α coefficient was chosen to reflect this fact. Next, $\beta G2$ generated the second-highest number of requests for the users from and around PoI_2 so the magnitude of β was chosen in such a way that it is lower than α while $\gamma G3$ should generate the smallest number of authentication requests associated with the location, and the magnitude of γ was determined as the lowest value to generate the least amount of location data samples which be sent from or around the PoI_3, so that

$$\alpha > \beta > \gamma \quad (3)$$

We generated our data samples for both mobile and fixed users along with the map presented in Figure 4.3 using a mixture of Gaussian PDFs.

Gaussian PDF is as follows:

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (4)$$

where μ is the mean of the distribution and σ is the standard deviation (σ^2 is the variance). In order to generate data samples, we needed to determine the values of μ and σ based on the following considerations:

- The values of μ for each factor of the mixture of Gaussian PDFs (Formula 2) are determined based on the location of the PoI in the map. We assumed that all PoIs are located in the centre of their cells, and the size of each cell is 400 m \times 400 m. For example, the values of μ for the first PoI are $\mu_x=200$ and $\mu_y=200$.

- As discussed earlier, one of the contributions of this work is to track the effect of mobility on the performance of the prediction models. In doing so, we need to synthesize different datasets in terms of mobility to investigate our hypothesis. Thus, we generated data samples for three datasets by changing the value of sigma (σ) in Gaussian PDFs in such a way that the bell-shaped curve of each Gaussian PDF becomes gradually flatter and wider. As it moves towards a flat bell-shaped curve, the degree of mobility increases.
- These values were used to generate random values in both dimensions X and Y. We call these datasets low-mobility, medium-mobility and high-mobility datasets.

Table 4.3 shows the values for three Gaussian PDFs that reflect the above-mentioned considerations.

Table 4.3: Parameters for three Gaussian PDFs based on three degrees of mobility

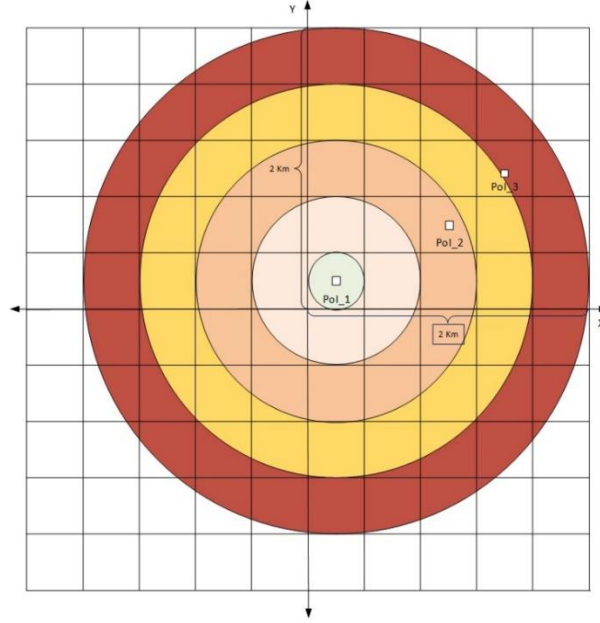
	Low Mobility		Medium Mobility		High Mobility	
PoI_1 ($\alpha=0.65$)	μ_x	200	μ_x	200	μ_x	200
	σ_x	50	σ_x	200	σ_x	600
	μ_y	200	μ_y	200	μ_y	200
	σ_y	50	σ_y	150	σ_y	500
PoI_2 ($\beta=0.20$)	μ_x	1000	μ_x	1000	μ_x	1000
	σ_x	200	σ_x	500	σ_x	900
	μ_y	600	μ_y	600	μ_y	600
	σ_y	150	σ_y	400	σ_y	700
PoI_3 ($\gamma=0.15$)	μ_x	1400	μ_x	1400	μ_x	1400
	σ_x	300	σ_x	700	σ_x	1200
	μ_y	1400	μ_y	1400	μ_y	1400
	σ_y	200	σ_y	600	σ_y	1000

After we generated data samples in terms of location, UVs related to these samples needed to be assigned. Therefore, we defined five different uncertainty areas (UAs) for each PoI. Each UA covers an area around the PoI and indicates our uncertainty about the authentication requests that come from that area. In order to define UAs for each PoI, five circles were drawn with the PoI point as the centre and with $(2n+1) \times r$ as the radius ($n=0,1,2,3\dots$ and $r=200m$). The number of circles and the length of the radius could have varied from one case study to another. Figure 4.4 (a, b and c) shows UAs for three PoIs.

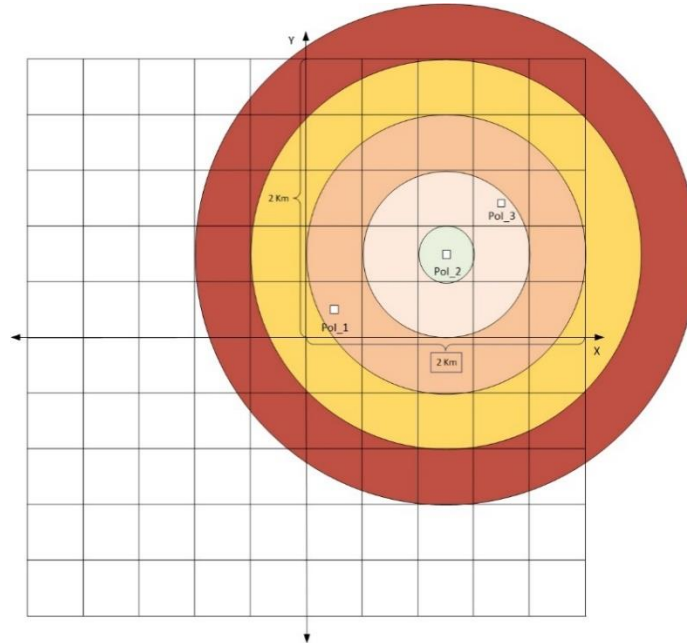
According to our mixture of Gaussian PDFs, UVs must be calculated using the following formula:

$UV(X) = \alpha \times (\text{UV assigned by PoI_1 to } X) + \beta \times (\text{UV assigned by PoI_2 to } X) + \gamma \times (\text{UV assigned by PoI_3 to } X)$ (5)

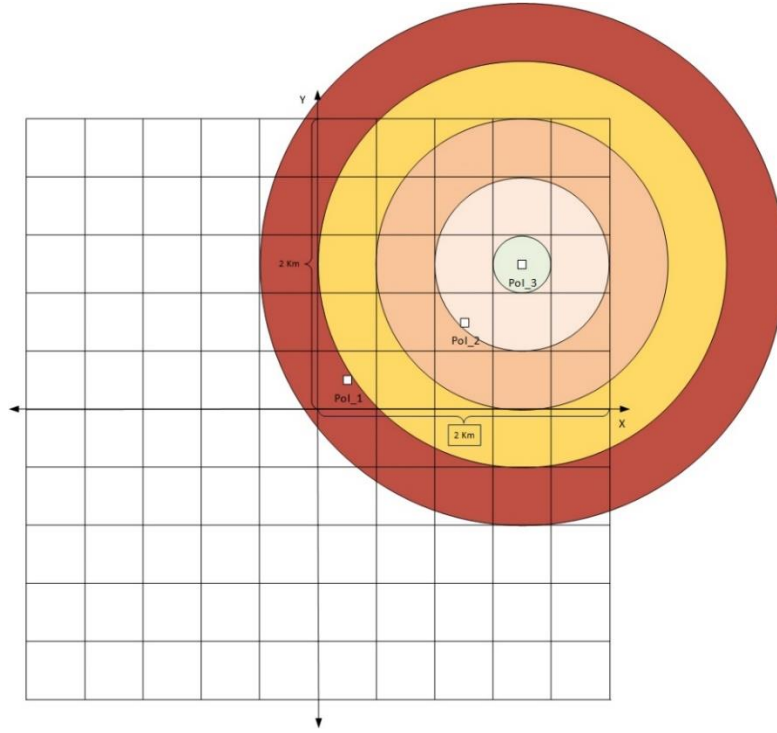
The value allocated by each given PoI in the formula is determined by the UA on which the point rests. As can be seen, the formula has three factors based on our mixture of Gaussian PDFs.



a. UAs defined for PoI_1



b. UAs defined for PoI_2



c. UAs defined for PoI_3

Figure 4. 4: UAs and associated UVs defined for three PoIs

α , β and γ in Formula 5 were discussed earlier. The UVs that are assigned to the UAs are shown in Figure 4.5. According to our threat model discussed in Subsection 4.2, distances that are closer to the PoI have lower UVs so circles that are closer to each PoI has lower UVs.

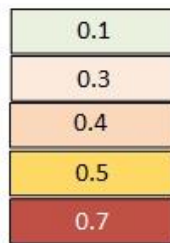


Figure 4. 5: UVs assigned to UAs (indicated by colour)

As an example, assume that one of our data samples, X, is located in a spot on the map shown in Figure 4.6. Using Formula 5 and the values listed in Figure 4.5, the UV for this sample is calculated as follows ($\alpha=0.65$, $\beta=0.20$ and $\gamma=0.15$):

- UV assigned by PoI_1 to X=0.5, according to Figure 4.4.a X falls in Yellow UA of PoI_1
- UV assigned by PoI_2 to X=0.4, according to Figure 4.4.b X falls in Dusty Beige UA of PoI_2
- UV assigned by PoI_3 to X=0.4, according to Figure 4.4.c X falls in Dusty Beige UA of PoI_3

Therefore, $UV(X)=(0.65 \times 0.5) + (0.2 \times 0.4) + (0.15 \times 0.4)=0.465$

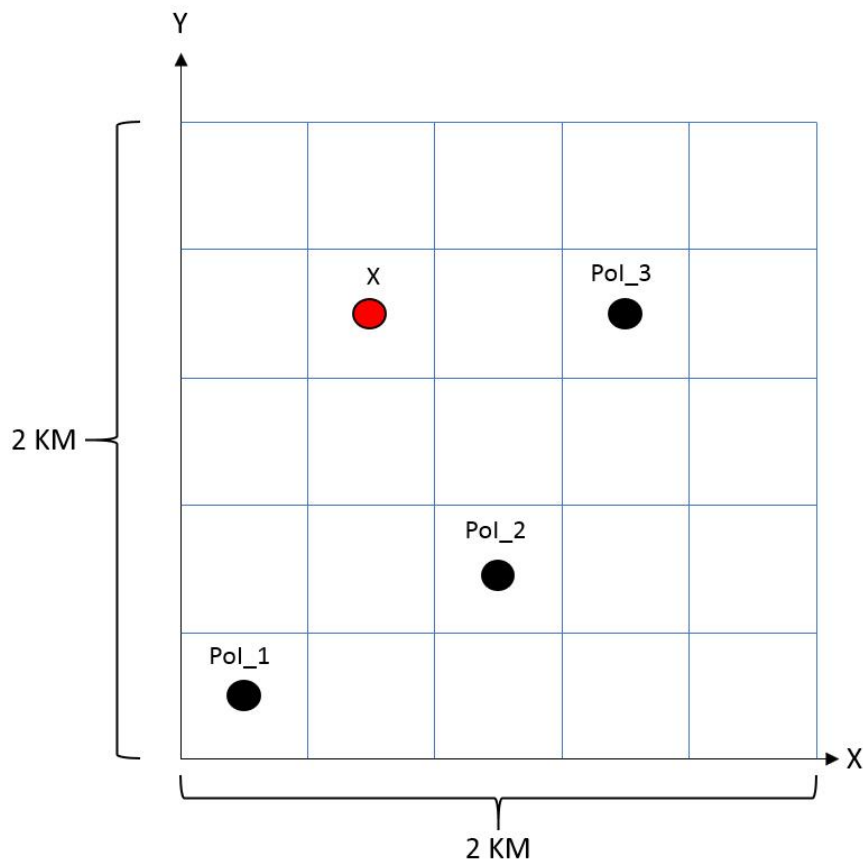


Figure 4. 6: An example of calculating a UV for a data sample (X)

4.3.4 Credentials

The most common forms of credential are username and password. We considered this information as the credentials in this research. To generate data samples and the associated UVs, the corresponding PDFs needed to be identified.

In all authentication processes using usernames and passwords, three possibilities may occur: i) both username and password provided by the user are correct, ii) only the username is correct, and iii) both username and password are incorrect. Data samples for these three possible states can be generated using a *multinomial* PDF. Therefore, the outcome of the multinomial distribution consists of three states. In order to initialize the parameters of the multinomial distribution we needed to determine the weights for the above-mentioned states. Generally, most users enter their usernames and passwords correctly. If not, they usually enter their usernames correctly but enter their passwords incorrectly. This was considered when assigning probability values (as weights) and associated UVs. We assigned the lowest UV to those users who correctly entered their usernames and passwords, because of the threat of insiders. In cases of the wrong password being entered, the assigned UV is less than in cases in which users enter both username and password incorrectly. Handling authentication using the uncertainty-aware approach helps to develop resilient authentication methods. Table 4.4 lists the weights and UVs for three states.

Table 4.4: Assigned uncertainty values for three states of credentials

Username & Password are correct	Username is correct but Password	Username & Password are incorrect
Probability: 0.85	Probability: 0.10	Probability: 0.05
Uncertainty: 0.05	Uncertainty: 0.70	Uncertainty: 0.95

4.3.5 Authentication Decision

After generating the UVs for each attribute in the matrix shown in Figure 4.1, the final UV was calculated for each request in order to make an authentication decision. The final value for each authentication request was calculated by averaging the UVs of time, location and credentials. Generally, credentials are the most important authentication attribute, in comparison with time and location. Therefore, we added weights to the generated UVs to show the priority and importance of the attributes. According to the adversary model discussed in Subsection 4.2, credential loss or sharing credential with ineligible users put security of data in danger. As a result, credential related threats have severe consequences in comparison to time and location related adversaries. For this reason, we assign the highest value as “weight” to credential. Furthermore, based on the threat model, requesting access from a location far from designated PoIs, is more dangerous than requesting access out of work hours. Therefore, we assign second highest value as “weight” to location. The assigned weights are as follows: time=2, location=3 and credentials=5. The magnitude of these weights may vary based on the research priorities.

Then, we calculated the weighted arithmetic mean by averaging weighted UVs per authentication request. Finally, to label the dataset we used the final UV for each request as the probability for *binomial* distribution to determine the class of the result: {0: Deny and 1: Access}. Figure 4.7 depicts the reconstructed version of the matrix shown in Figure 4.1. As can be seen, “authentication decision” was added to label the dataset.

	User ID	Time	Location	Credentials	Decision
Authentication Request #1.....	12.....	0.6.....	0.32.....	0.70.....	1.....
Authentication Request #2.....	22.....	0.1.....	0.28.....	0.05.....	1.....
Authentication Request #3.....	51.....	0.2.....	0.72.....	0.70.....	1.....
Authentication Request #4.....	13.....	0.1.....	0.43.....	0.95.....	0.....
Authentication Request #5.....	7.....	0.5.....	0.78.....	0.05.....	1.....
Authentication Request #5000.....	43.....	0.1.....	0.53.....	0.05.....	1.....

Figure 4. 7: Labelled uncertainty matrix

4.4 Los Alamos National Lab (LANL)

When a real data is sensitive (e.g. authentication data), synthetic data can be generated to replace real data. A synthesized dataset is generated by considering these properties:

- The number of attributes and the size of the dataset should be arbitrary.
- It should be random, and it should be generated using a wide variety of statistical distribution to base this data upon.
- To be used by classification algorithms, the degree of class separation should be adjustable to make the learning problem easy or hard
- Random noise should be interjected in a controlled manner

Synthesized dataset considering aforementioned properties may even suffer from a number of limitations like bias and generalization issues. In order to validate our synthesized dataset, to determine whether it behaves like a real dataset in action, we choose LANL [152] as our benchmark. LANL provided a public dataset consisting of user-computer authentication associations in time. The dataset contains authentication events on a separate line in the form of “time, user, computer”, delimited by commas. The number of users in the dataset is 11,362 and the number of resources is 22,284. The number of authentication events captured by the dataset is 708,304,516 for nine months.

We analysed the distribution of users in this dataset. Figure 4.8 shows the results of our analysis in the form of a diagram. As shown, the involvement of users in authentication events follows *power law* distribution.

This finding helps to confirm the correctness of our methodology to synthesize data samples for “users” in our dataset. We also analysed “time” attribute of LANL authentication dataset for two random months (June and September). The results indicated that “time” attribute follows Gaussian distribution. This finding confirms our methodology for generating data samples for time of the requests.

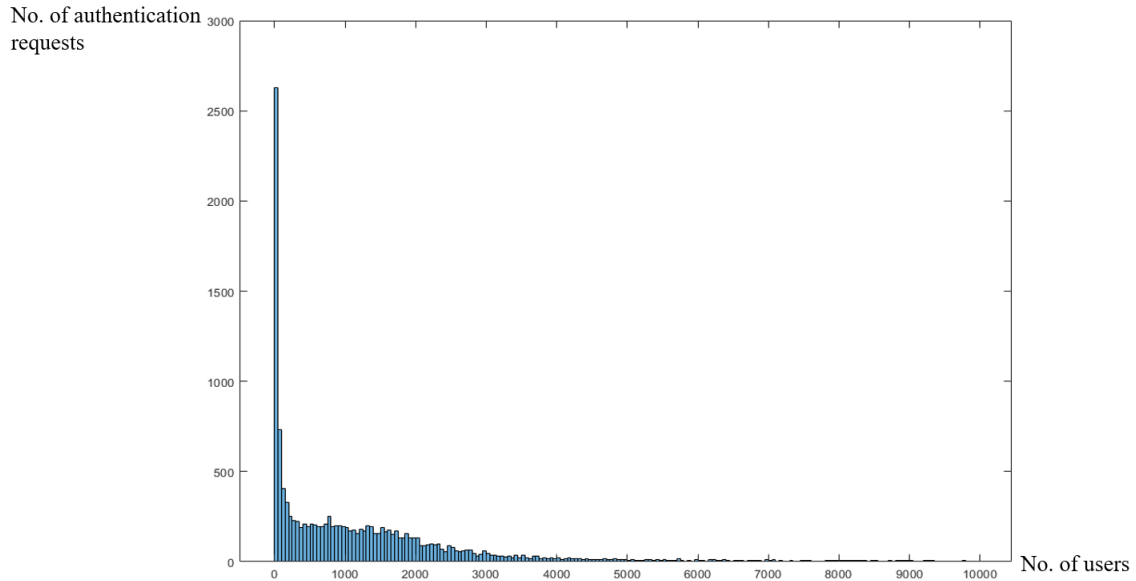


Figure 4. 8: Distribution of authentication requests in LANL

4.5 Chapter Summary

This chapter presented the exemplar (RASA) as our case study in this dissertation. We also determined adversary model for the exemplar. We described how the dataset is generated according to specifications of our exemplar. We discussed the process of synthesizing data samples for all of our attributes in the dataset. We also analysed a public authentication dataset (LANL) to make sure that our methodology in generating data samples for the attributes is correct.

5. Uncertainty-Aware Prediction Model for Authentication

This chapter presents the process of building prediction models for authentication. As described in Chapter 3, machine-learning algorithms were used to build our data-driven models. In this chapter, we explain all the classification algorithms applied in this work. Before introducing these algorithms, it is necessary to review the preliminary concepts and evaluation criteria that are used in our methodology.

5.1 Preliminary Concepts

Machine learning is a part of artificial intelligence (AI) paradigms that greatly overlaps with statistics. Samuel [153] defined machine learning as a “field of study that gives computers the ability to learn without being explicitly programmed” in 1959. A more precise definition for machine learning was provided by Mitchell. He described machine learning as a program that enables computers to program themselves by learning from past experiences [154]. He addressed three main aspects of machine learning – learning, past experiences and performance – in his definition [155]: “A computer program is said to learn from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E ”.

There are four types of machine-learning approaches [156]: supervised, unsupervised, semi-supervised and reinforcement learning.

In supervised learning methods, a machine-learning algorithm is applied to a labelled dataset to analyse data. In the process, a model is trained on that dataset, and afterwards the model is able to predict the class of the new data samples. Supervised learning methods can be divided into classification and regression algorithms. Classification algorithms are decision tree, random forest, k-nearest neighbours (K-NN), Naïve Bayes, neural networks, voting classifier and boosting classifiers. Regression algorithms are support vector machine and logistic regression. These classifiers are commonly used in intrusion detection systems (IDS), malware detection and intrusion prevention systems (IPS) [157], [158], [159].

From a learning perspective, unsupervised learning methods are not trained with labelled datasets. They partition data into a finite set of groups in which data samples have the highest inner similarity and outer dissimilarity. In these methods, partitioning is conducted without any prior knowledge. Clustering algorithms (e.g., k-means), association rules and outlier detection (e.g., behavior-based approach) methods are known as unsupervised learning paradigms. Unsupervised learning algorithms, as well as supervised algorithms, are applied in the field of cybersecurity to detect anomalies [160]. Unsupervised algorithms have advantages over supervised ones [161]:

- These methods have better performance in detecting unknown anomalies.
- These algorithms are less time-consuming and resource-intensive because they do not require labelling dataset that will be used for training.

One of the main drawbacks of using unsupervised algorithms in cybersecurity applications is their high FP rate.

Semi-supervised learning algorithms may be trained by a dataset that includes both labelled and unlabeled data samples. This method is particularly applicable when the procedure of dataset labelling is time-consuming and/or extracting relevant attributes from the dataset is difficult [162]. Graph-based algorithms and low-density separation are examples of semi-supervised techniques. Semi-supervised algorithms are applied in the field of cybersecurity to build IDS [163], [164].

Reinforcement learning has different mechanism in comparison with the mentioned approaches. The basic idea behind reinforcement learning is learning from the interaction. Reinforcement learning consists of three elements [165]:

- A policy that defines the behavior of the learning agent (learner) at a given time.
- A reward signal that defines the goal of the learning problem. At any given time in the process of learning, a single number is given to the learner as a reward for its action.
- Value functions that determine the total number of rewards a learner can expect to accumulate. Therefore, values specify the prediction of the rewards.

In this type of learning, the learner is not told which action to take but must instead find out which action may lead to the greatest reward. Therefore, unlike in supervised and unsupervised learning methods, the focus in reinforcement learning is on maximizing the reward signal.

Q-learning, state–action–reward–state–action and deep Q networks are among the reinforcement learning algorithms. Cyber-attack detection in “smart grids” is the widest application of such algorithms [166], [167], [168].

As depicted in Figure 5.1, the process of building a data-driven prediction model consists of four phases: preprocessing, learning, validation and prediction.

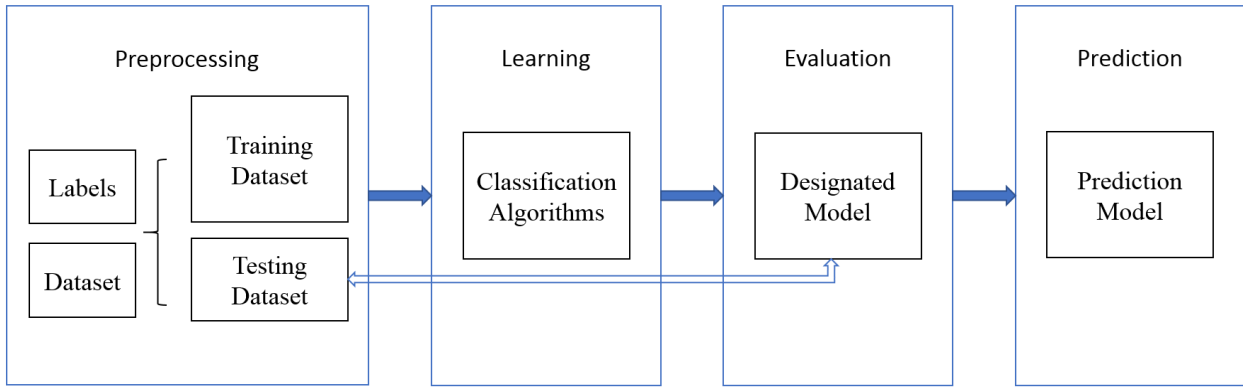


Figure 5. 1: The process of building a prediction model using supervised algorithms

In the **preprocessing phase**, data should be ready in the form and the shape necessary for the optimal performance of the learning process. Feature extraction, scaling, feature selection, dimensionality reduction and sampling are among the activities of this phase. As mentioned in Chapter 4, our synthesized dataset is generated with all these considerations in mind. The dataset also needs to be labelled if the problem is a type of classification. One of the important tasks of this phase is to randomly divide the labelled dataset into a training dataset and a test dataset. The training dataset is used to train and optimize the machine-learning model in the learning phase and the test dataset is used to evaluate the trained model in the evaluation phase.

After the preparation a proper dataset is prepared, different machine-learning algorithms (e.g., classifiers) are applied to the training dataset to build a model. It is necessary to apply different algorithms in the **learning phase** in order to select the best-performing model. Two problems that may occur in the learning phase: *overfitting* and *underfitting*. **Overfitting** occurs when a model performs well on the training dataset but cannot generalize well to the testing dataset. On the other hand, **underfitting** deals with the situation in which a model cannot capture the pattern in the training dataset and therefore is incapable of predicting well on the testing dataset. These problems

degrade the performance of the final prediction model. In order to limit these challenges, cross-validation can be used. The goal of cross-validation is to define a dataset to test the model in the training phase in order to limit problems such as overfitting and underfitting and get an insight into how the model will generalize to an independent dataset.

After the fitted model in the learning phase is chosen, the model should be evaluated using the testing dataset to estimate the performance of the model in dealing with unseen data. The **evaluation phase** gives an estimation of the generalization error. Afterwards, the evaluated model is called the **prediction model** and is ready to accept a new dataset to predict the label.

A number of primary measures exist that can be used for comparing the models created in the learning phase, including accuracy, precision, recall and F1. Before introducing these criteria, a set of measures should be introduced:

- **true positive (TP):** A positive sample that is correctly classified by the model
- **false negative (FN):** A positive sample that is misclassified by the model
- **false positive (FP):** A negative sample that is misclassified by the model
- **true negative (TN):** A negative sample that is correctly classified by the model

Based on the above metrics, the following measures can be calculated:

- **accuracy:** $(TP+TN)/(TP+TN+FP+FN)$
- **precision:** $TP/(TP+FP)$
- **recall or true positive rate:** $TP/(TP+FN)$
- **F1:** $(2*TP)/(2*TP+FN+FP)$ – it calculates the harmonic mean of the precision and recall
- **True Negative Rate (TNR):** $TN/(TN+FN)$
- **False Negative Rate (FNR):** $FN/(TP+FN)$
- **False Positive Rate (FPR):** $FP/(FP+TN)$

According to the metrics discussed, two other measures can be defined: confusion matrices and Receiver Operating Characteristics (ROC).

As shown in Figure 5.2, a confusion matrix demonstrates the performance of the model by showing the relationship between the actual labels and the predicted ones.

	Predicted as Positive	Predicted as Negative
Labelled as Positive	TP	FN
Labelled as Negative	FP	TN

Figure 5. 2: The structure of a confusion matrix

ROC visualizes the TPR against the FPR to depict relative trade-offs between TP (known as benefits) and FP (known as costs) [169]. In a ROC diagram, the diagonal line ($y=x$) shows the border of random guesses of the class label. Any classifier curve that appears in the lower-right triangle has lower performance than the random guesses. The curves that appear in the upper-left triangle have better performance in terms of accuracy. The size of the area under the curve (AUC) in the ROC diagram matters. The bigger it is, the better the performance.

In cybersecurity literature, the above metrics are used to give insight into the performance of the prediction models. The most commonly used metrics in assessment are accuracy, precision, recall and F1.

Access control in general and authentication in particular are classification problems. It is usually defined as a problem with (discrete) binary outputs (Access/Deny). For this reason, mentioned classification and regression algorithms are the choices for building prediction models in the field of authentication. As we demonstrate in the rest of this chapter, we applied all classification algorithms in order to build our prediction model. This gave us the opportunity to compare them and choose the most efficient and accurate algorithm with which to build our prediction model.

5.2 Prediction Models

We applied 10 classification algorithms to our training dataset and measured the performance of the prediction models using the metrics discussed. All classifiers were applied to three datasets (low-mobility, medium-mobility and high mobility) to track the changes in their performance. In order to validate the data model, a cross-validation process was used by each of the applied classifiers. As discussed earlier, cross-validation is a widely used method of evaluating the generalizability of proposed models [170]. In this way, 10% of the dataset was assigned to the test split (10-fold cross-validation). In order to increase the chance of finding the best-fit model and to improve the generalizability of the generated model, we also used the shuffling feature when dividing the dataset into the training and testing parts.

5.2.1 Decision Tree

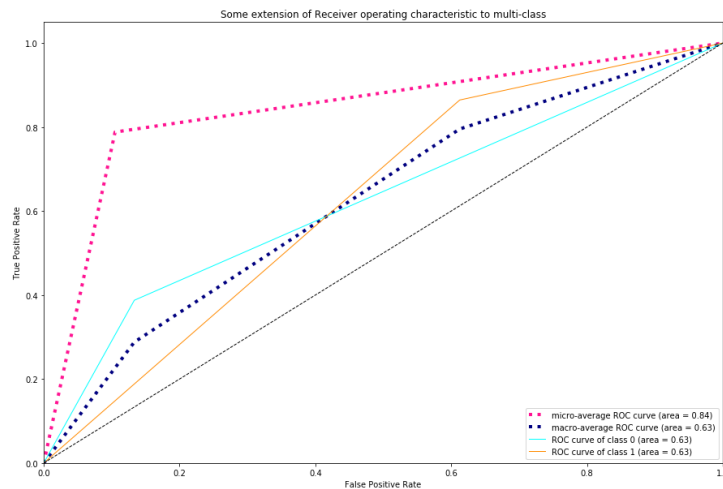
A decision tree is a classification method that makes a set of hierarchical decisions on the feature values formed in a tree-like structure. Any decision splits the tree based on a criterion in such a way that the training data are divided into two or more branches. The goal is to find the best split criterion by which the number of class variables in each branch of the tree is reduced as much as possible [171]. There are three classical algorithms for decision trees, including ID3, C4.5 and CART (classification and regression trees). These algorithms use two splitting criteria called as “Entropy” and “Gini”. Of these three classical algorithms, we applied the CART algorithm to build our data model. CART has advantages over the other algorithms in terms of reducing overfitting and the ability to handle incomplete data [172]. It also builds models for regression as well as classification. CART uses the Gini criterion for splitting. An optimized version of CART that has been implemented by the scikit-learn library is used in this work. Table 5.1 shows the performance of the models created using the decision tree algorithm (CART).

According to the results, and as expected, the accuracy goes down from 77.56% to 55.87% as the mobility changes from low to medium. The accuracy of the prediction model in the high-mobility environment is slightly lower than that in the medium-mobility environment. The highest precision is obtained by the model developed using the low-mobility dataset, with a value of 88%. Moreover, ROC curve analysis confirms this finding in a visual manner. Figure 5.3 shows the micro-average ROC curves of the “Access” class for these three models. As shown in Figure 5.3, the maximum AUC belongs to the prediction model that was built using the low-mobility dataset (AUC=0.63). The second-highest value of the AUC is for the prediction model trained by the medium-mobility training dataset.

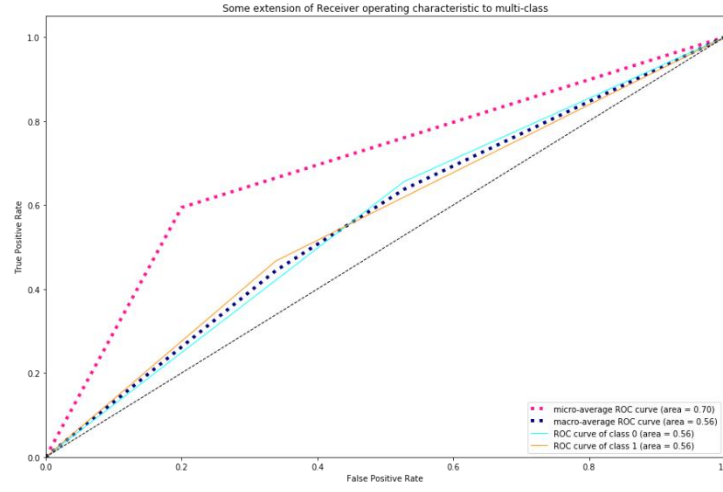
Table 5.1: Performance of the prediction model trained by three datasets (decision tree)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	77.56% (1.85%)	0	0.35	0	0.39	0	0.37
		1	0.88	1	0.86	1	0.87
Medium Mobility	55.87% (2.39%)	0	0.72	0	0.66	0	0.69
		1	0.40	1	0.47	1	0.43
High Mobility	54.29% (2.06%)	0	0.67	0	0.67	0	0.67
		1	0.44	1	0.44	1	0.44

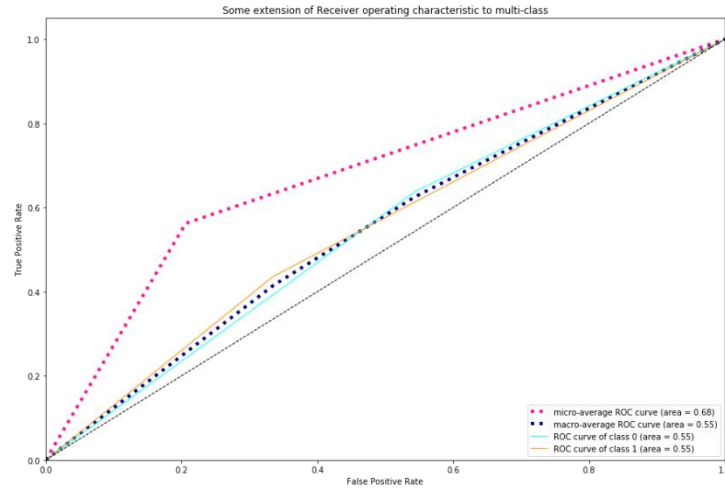
0: Deny, 1: Access



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 3: a, b and c: ROC analysis for the decision-tree-based models

5.2.2 Random Forest

A random forest is a classifier that has gained popularity because of its performance and scalability characteristics [173]. Random forests use a number of decision trees to build a more robust data model that is less susceptible to overfitting. In this work we used a random forest classifier from the scikit-learn library to train and build our data model. The depth of the classifier was changed during the experiments in order to improve the accuracy of the model and find the

best-fit model. According to the results, accuracy stopped improving for depth values of more than 2.

Table 5.2 shows that the best performance was achieved by the random forest algorithm using our three datasets.

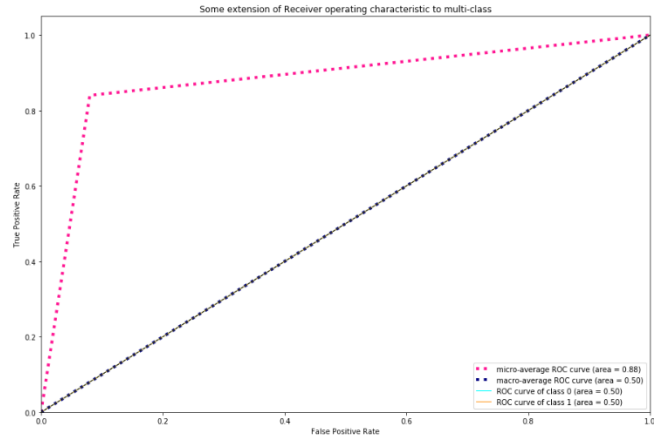
Table 5.2: Performance of the prediction model trained by the low-mobility dataset (random forest)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	82.20% (1.07%)	0	0.00	0	0.00	0	0.37
		1	0.84	1	0.10	1	0.91
Medium Mobility	64.43% (2.63%)	0	0.67	0	1.00	0	0.81
		1	0.00	1	0.00	1	0.00
High Mobility	61.69% (2.24%)	0	0.63	0	1.00	0	0.77
		1	0.00	1	0.00	1	0.00

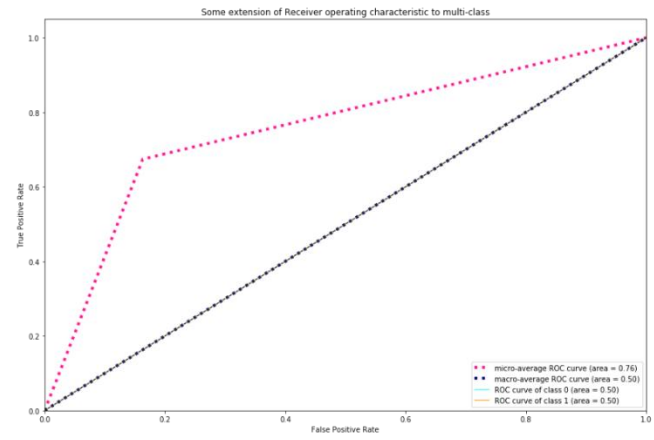
0: Deny, 1: Access

As expected, the performance of the random forest algorithm is better than that of the decision tree algorithm for all of our three datasets. The best accuracy was achieved in the low-mobility environment (82.20%). Precision, recall and F1 for the “Access” class could not be determined because of the noise effect. For this reason, we applied ROC analysis to gain insight into the performance of our models in situations in which noise affects the measurement. As with our models created by the decision tree, the accuracy of the models created by the random forest classifier dropped from 82.20% in the low-mobility environment to 64.43% in the medium-mobility environment.

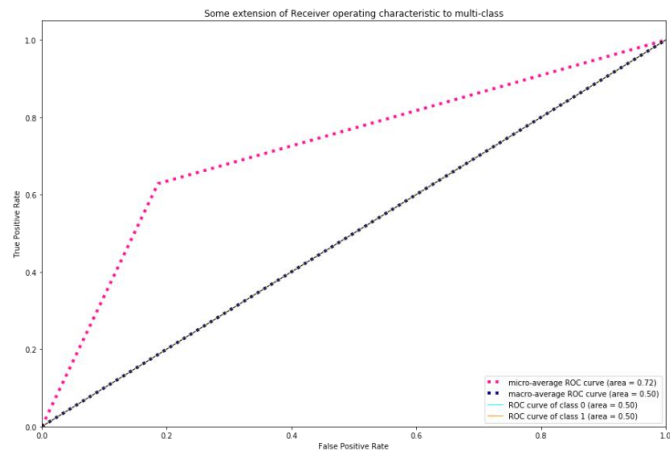
Figure 5.4 shows the micro-average ROC curves for the “Access” class related to the three models created by our datasets.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 4: a, b and c: ROC analysis for the random-forest-based models

As shown in the above figure, the AUC decreased from 0.88 in the low-mobility environment to 0.72 in the high-mobility environment. This confirms the trend of decreasing accuracy shown in Table 5.2.

5.2.3 Support Vector Machine (SVM)

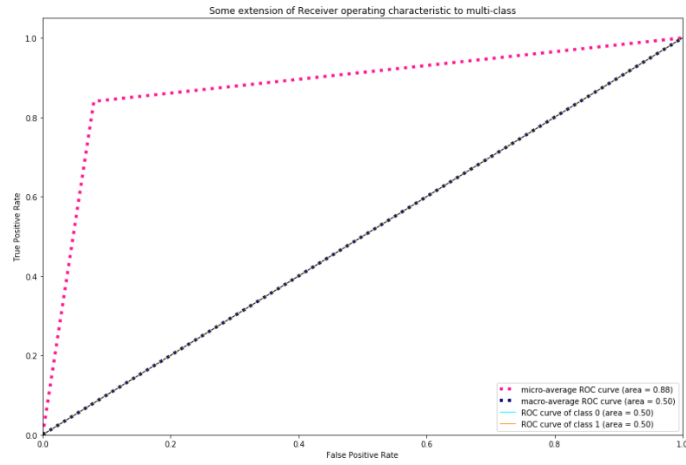
The support vector machine (SVM) is one of the most robust and widely used binary classification algorithms. The goal of the SVM optimization program is to determine the separating hyperplane that maximizes the distance between the closest training samples to it (the *support vectors*) [174]. This reduces the misclassification error while maximizing the generalization capability for test datasets. In addition, when the training set is non-linearly separable, as is the case in this study, SVM is combined with the kernel trick to expand the space implicitly and facilitate the linear separability for the two classes (i.e., Access and Deny) [174]. We applied the support vector classification algorithm from the scikit-learn library in order to build our prediction model.

Table 5.3: Performance of the prediction models trained by three datasets (SVM)

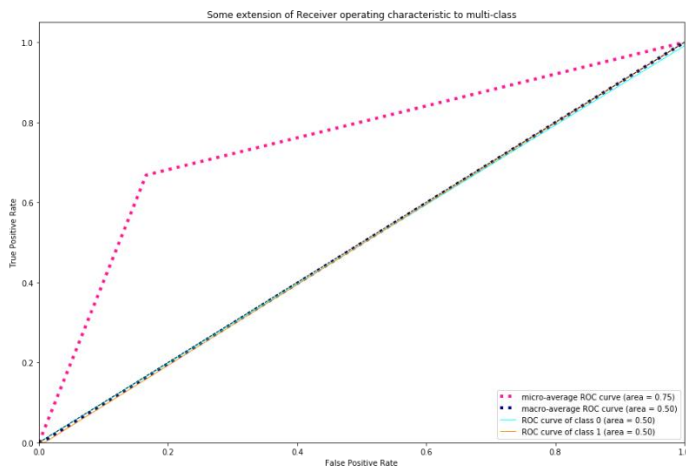
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	82.20% (1.07%)	0	0.00	0	0.00	0	0.00
		1	0.84	1	0.10	1	0.91
Medium Mobility	64.27% (2.48%)	0	0.67	0	0.99	0	0.80
		1	0.00	1	0.00	1	0.00
High Mobility	60.89% (2.00%)	0	0.62	0	0.97	0	0.76
		1	0.11	1	0.01	1	0.01

0: Deny, 1: Access

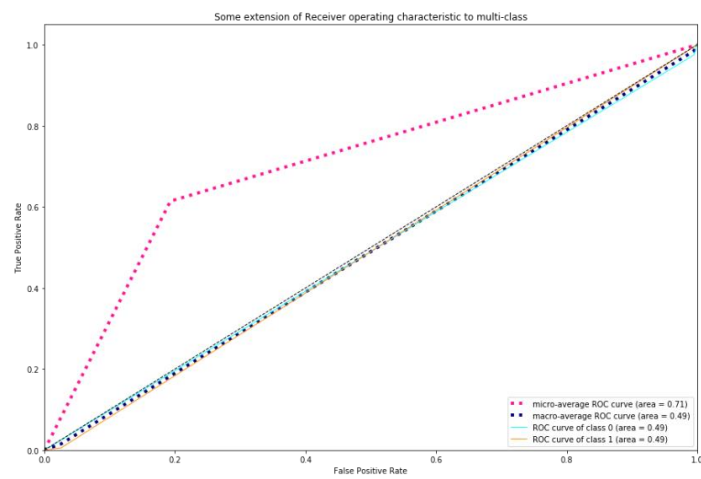
Table 5.3 shows the performance of the models created by the SVM algorithm using our three datasets. According to the results, SVM shows the same performance as the random forest classifier in the low-mobility environment in terms of accuracy (82.20%) but it shows a slightly lower performance than the random forest in the medium-mobility and high-mobility environments. Figure 5.5 shows the ROC analysis for the SVM-based models.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 5: a, b and c: ROC analysis for the SVM-based models

As presented by the micro-average ROC curves, the AUC decreased gradually from 0.88 in the low-mobility environment to 0.71 in the high-mobility environment. In other words, the ROC curves became flatter so the performance of the models decreased.

5.2.4 Logistic Regression

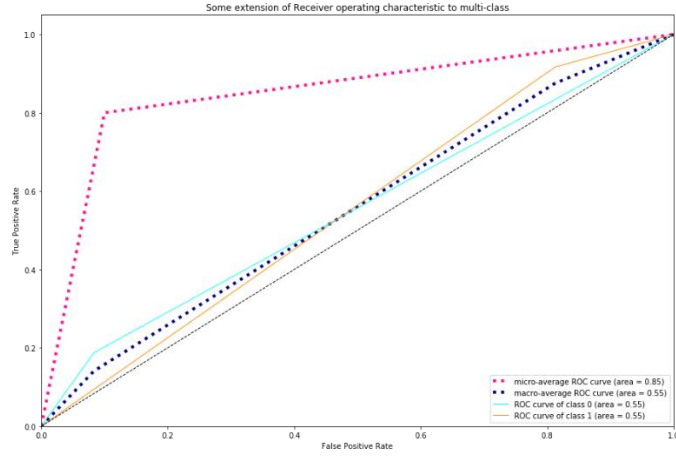
Logistic regression is an analytic method for classification problems. It is able to model scenarios with two or more possible discrete outcomes. It uses a probabilistic classifier and maps the feature variables to a class-membership probability. The most common form of logistic regression builds data-driven models with binary outcomes (i.e., Access/Deny). In this work we used a logistic regression classifier with binary outcomes.

Table 5.4 shows the performance of the prediction models created by the logistic regression algorithm for three datasets. The highest performance was achieved in the low-mobility environment, with 79.94%. In addition to this, as with the above-mentioned classifiers, the accuracy value goes down when the mobility in the environment increases. Figure 5.6 shows the ROC analysis of the prediction models developed by the logistic regression algorithm.

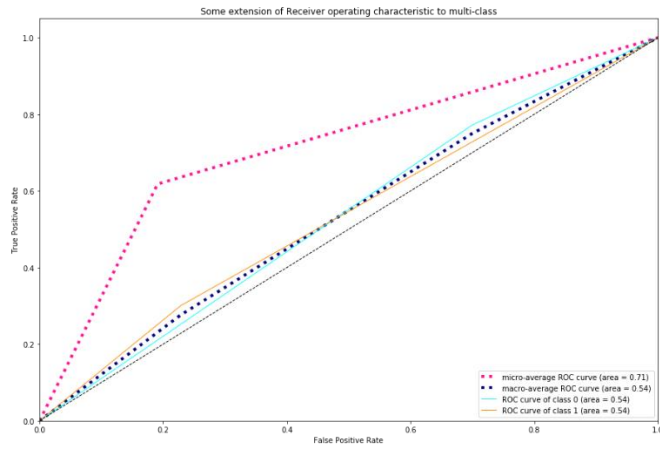
Table 5.4: Performance of the prediction models trained by three datasets (logistic regression)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	79.94% (1.69%)	0	0.30	0	0.19	0	0.23
		1	0.86	1	0.92	1	0.89
Medium Mobility	57.81% (2.13%)	0	0.70	0	0.77	0	0.73
		1	0.39	1	0.30	1	0.34
High Mobility	55.37% (1.81%)	0	0.64	0	0.72	0	0.68
		1	0.40	1	0.32	1	0.36

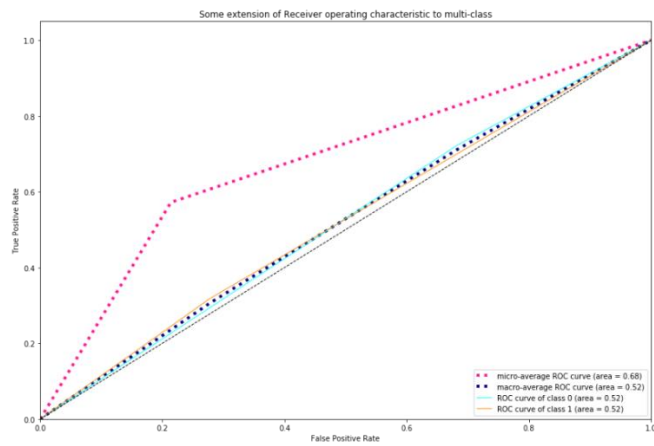
0: Deny, 1: Access



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 6: a, b and c: ROC analysis for the logistic-regression-based models

The largest AUC is for the model created using the low-mobility dataset (AUC=0.85). The AUC values for the models created using the medium- and high-mobility datasets are 0.71 and 0.68 respectively.

5.2.5 Naïve Bayes

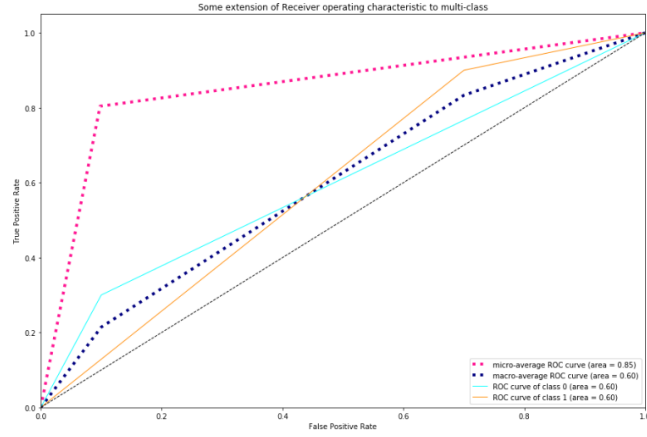
The Naïve Bayes classifier is the simplest form of a Bayesian network. It is termed “naïve” because it assumes that all attributes are conditionally independent. In spite of this controversial assumption, which is used to simplify the process of modelling, Naïve Bayes is a fast classifier and has great performance in practice for many domains [173]. We applied the Gaussian Naïve Bayes classifier implemented in the scikit-learn library. The performance of the models created by the Naïve Bayes algorithm using our datasets is shown in Table 5.5.

Table 5.5: Performance of the prediction models trained by three datasets (Naïve Bayes)

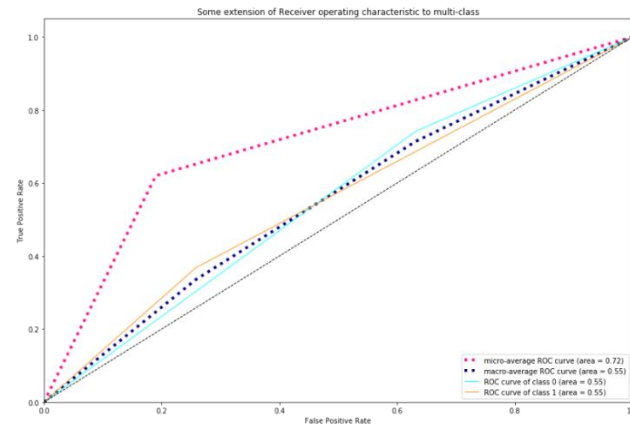
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	80.42% (2.00%)	0	0.36	0	0.30	0	0.33
		1	0.87	1	0.90	1	0.89
Medium Mobility	56.31% (2.60%)	0	0.71	0	0.74	0	0.72
		1	0.41	1	0.37	1	0.39
High Mobility	53.49% (2.64%)	0	0.64	0	0.61	0	0.63
		1	0.39	1	0.41	1	0.40

0: Deny, 1: Access

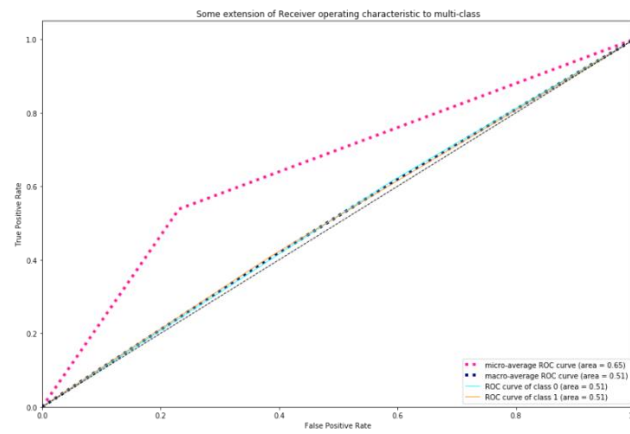
According to the results, the highest accuracy was achieved with the model created using the low-mobility dataset (80.42%). The accuracy dropped to 56.31% in the medium-mobility environment. Figure 5.7 depicts the ROC curves for the models created by the Naïve Bayes algorithm. As shown in the figures, the maximum AUC is for the model trained and created using the low-mobility dataset, with AUC=0.85. Furthermore, the micro-average ROC curves for the models created using the medium- and high-mobility datasets are flatter than the first curve. Thus, these two models have lower TP and higher FP rates than the similar criteria of the first curve.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 7: a, b and c: ROC analysis for the models created by the Naïve Bayesian algorithm

5.2.6 K-Nearest Neighbours (K-NN)

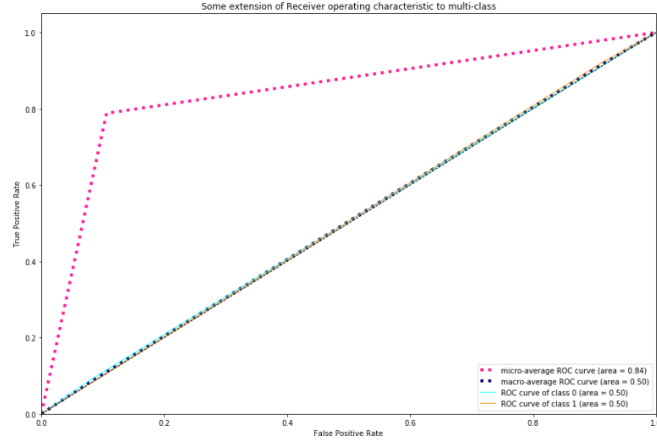
K-NN is an instance-based learning (IBL) classifier. IBL algorithms assume that similar instances have the same class labels. For this reason, these algorithms determine the closest K training samples and choose the dominant class label among them as the relevant class [175]. K-NN classifiers have several advantages [176]. The most important advantage is their simplicity. Moreover, these algorithms are noise-tolerant, and they have relatively low update cost. In this work, we applied all three IBL algorithms from the scikit-learn library – “Brute Force”, “K-D Tree” and “Ball Tree” – in order to build our prediction model with them and compare the results. Building our prediction models using these IBL algorithms enabled us to find the best-fit model. Table 5.6 shows the performance of the prediction models built using the three datasets.

Table 5.6: Performance of the prediction models trained by three datasets (K-NN)

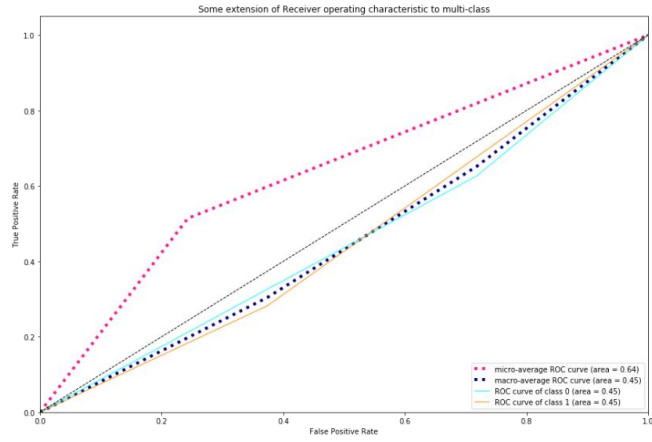
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	77.68% (1.17%)	0	0.17	0	0.09	0	0.12
		1	0.84	1	0.92	1	0.88
Medium Mobility	55.87% (2.44%)	0	0.64	0	0.63	0	0.63
		1	0.27	1	0.28	1	0.27
High Mobility	53.63% (2.04%)	0	0.62	0	0.70	0	0.66
		1	0.36	1	0.28	1	0.32

0: Deny, 1: Access

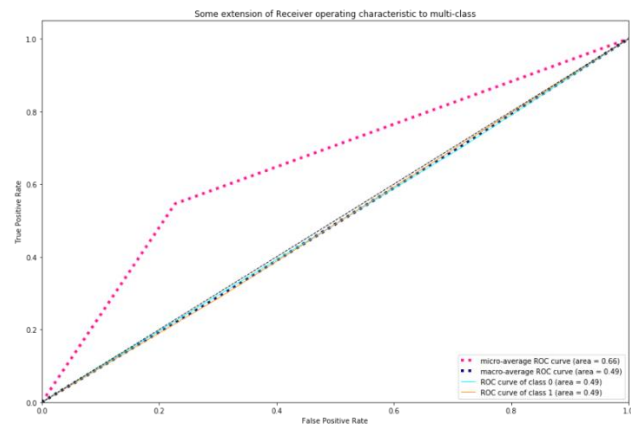
According to the results, K-NN shows lower performance in terms of accuracy, precision, recall and F1 in comparison with the above-mentioned models. The highest value of accuracy is 77.68%, in the low-mobility environment. ROC curve analysis confirms the results (Figure 5.8). As shown below, the maximum AUC is achieved in the low-mobility environment with the value of 0.84. One of the findings is that the performance of the model built for the high-mobility environment is better than the performance of the model trained by the medium-mobility dataset. This can be inferred by comparing the values of the AUC in these two environments.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 8: a, b and c: ROC analysis for the models created by K-NN algorithms

5.2.7 Boosting Algorithms

The idea behind the boosting approach is to lower the bias of the classifiers by focusing on the misclassification samples. For this reason, each training data sample is assigned a weight and different classifiers are trained with these weighted samples. In this method, future models are based on the previous ones, so it is assumed that errors from misclassified samples arise from the bias of the classifiers. Therefore, as a result of increasing the weights of misclassified samples and applying new classifiers, the bias decreases [177]. We applied two popular boosting algorithms: AdaBoost and gradient boost classifiers.

The AdaBoost algorithm is a popular machine-learning algorithm used to build strong classifiers by combining weak classifiers (tree-based classifiers). When a vast number of weak classifiers are employed, the rate of misclassification is reduced significantly [178]. Gradient boost algorithms are another type of boosting method and consist of a set of CART algorithms. Like AdaBoost, gradient boost is built incrementally by adding new trees and minimizing the misclassification error of the previous model [179].

Tables 5.7 and 5.8 show the performance of the models built using gradient boost and AdaBoost classifiers.

Table 5.7: Performance of the prediction models trained by three datasets (gradient boost)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	82.20% (1.07%)	0	0.00	0	0.00	0	0.00
		1	0.84	1	1.00	1	0.91
Medium Mobility	64.43% (2.63%)	0	0.67	0	1.00	0	0.81
		1	0.00	1	0.00	1	0.00
High Mobility	61.69% (2.24%)	0	0.63	0	1.00	0	0.77
		1	0.00	1	0.00	1	0.00

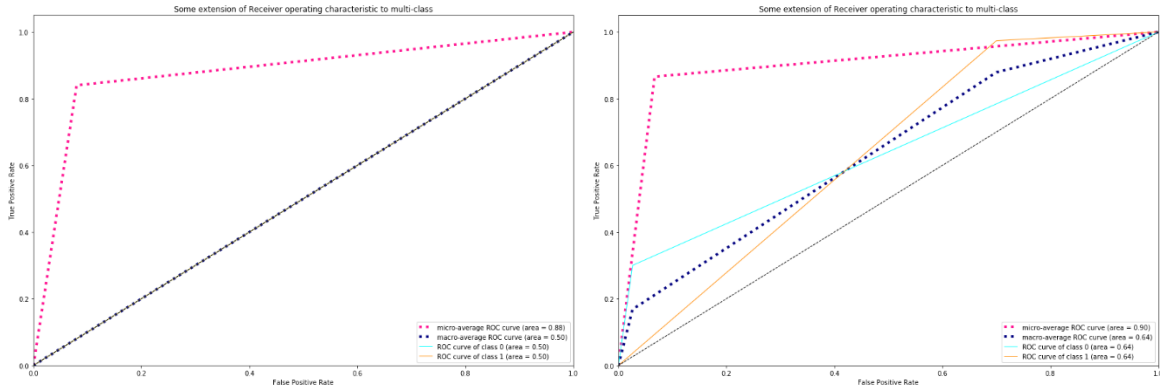
0: Deny, 1: Access

Table 5.8: Performance of the prediction models trained by three datasets (AdaBoost)

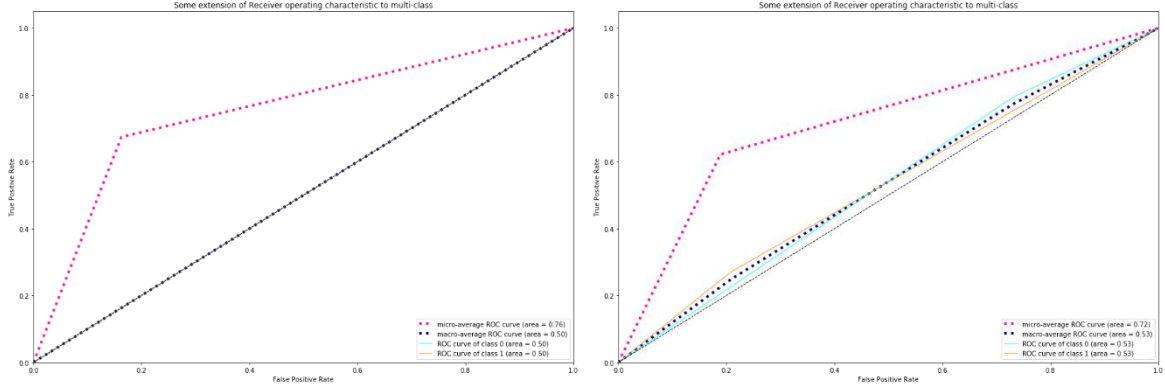
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	85.50% (1.32%)	0	0.69	0	0.30	0	0.42
		1	0.88	1	0.97	1	0.92
Medium Mobility	60.33% (1.64%)	0	0.69	0	0.79	0	0.74
		1	0.39	1	0.27	1	0.32
High Mobility	57.15% (2.10%)	0	0.63	0	0.75	0	0.69
		1	0.39	1	0.26	1	0.31

0: Deny, 1: Access

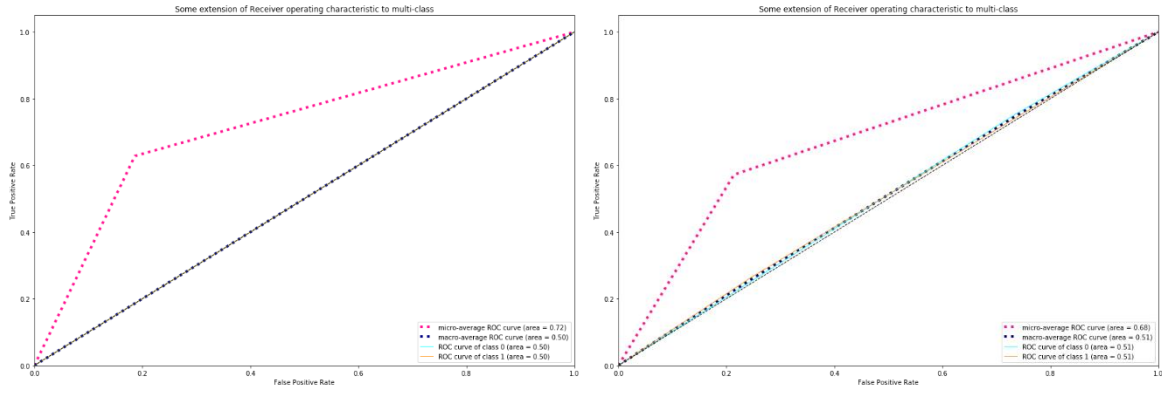
Table 5.7 shows the performance of the model created by the gradient boost algorithm. The accuracy of the model trained by the low-mobility dataset is 82.20%. Moreover, the models built for medium- and high-mobility environments have shown the best performance in comparison with the other algorithms so far. Figure 5.9 demonstrates the ROC curves for models built using the gradient boost and AdaBoost algorithms.



a. Gradient boost (left) and AdaBoost (right) – low-mobility dataset



b. Gradient boost (left) and AdaBoost (right) – medium-mobility dataset



c. Gradient boost (left) and AdaBoost (right) – high-mobility dataset

Figure 5. 9: a, b and c: ROC analysis for the models created by gradient boost and AdaBoost

As shown in Figure 5.9.a, the maximum AUC is for the AdaBoost model, with a value of 0.90 in the low-mobility environment, whereas the gradient boost model shows better performance in the medium- and high-mobility environments in terms of AUC values. Moreover, in all models, AdaBoost was superior in terms of predicting the “Deny” class.

5.2.8 Voting Classifier

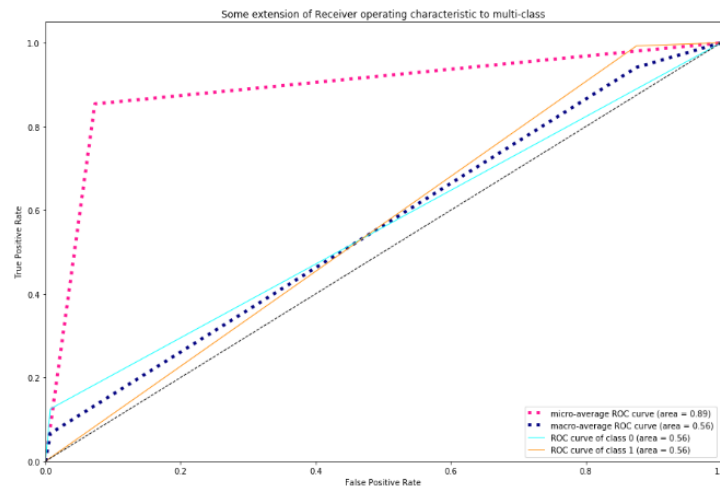
Voting classifiers can be classified into “soft vote” and “hard vote”. In a soft vote algorithm, different classifiers are aggregated to predict the class label (Access/Deny) based on the average probabilities predicted by each classifier, whereas the voting classifier in hard vote mode predicts the class label based on the majority of the labels predicted by each individual classifier [180]. We applied both soft and hard vote modes in this work and reported the best-fit model for each dataset. Table 5.9 shows the performance of the models built using voting classifiers.

Table 5.9: Performance of the prediction models trained by three datasets (voting classifiers)

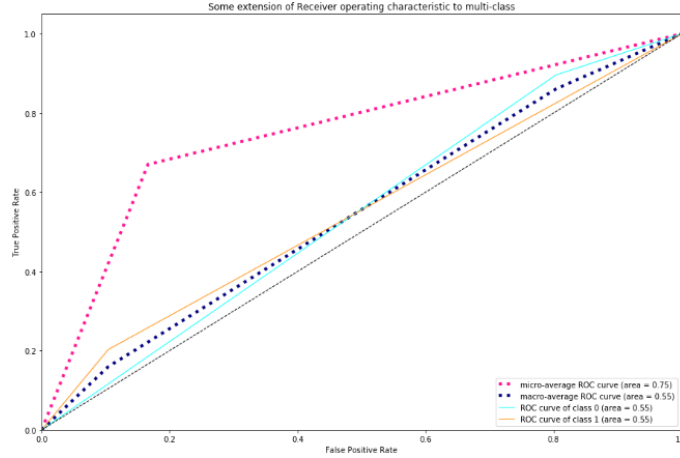
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	84.58% (1.22%)	0	0.77	0	0.12	0	0.22
		1	0.86	1	0.99	1	0.92
Medium Mobility	61.07% (2.36%)	0	0.70	0	0.90	0	0.79
		1	0.48	1	0.20	1	0.28
High Mobility	57.83% (2.21%)	0	0.64	0	0.83	0	0.72
		1	0.42	1	0.21	1	0.28

0: Deny, 1: Access

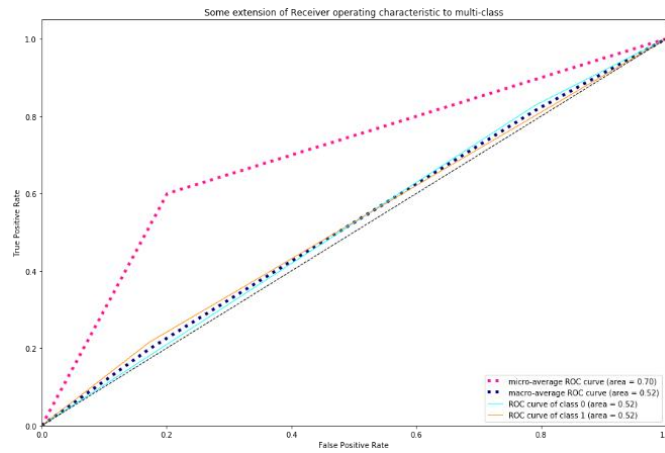
The best accuracy is achieved with the model developed by the low-mobility dataset, with a value of 84.58%. In addition, the soft vote algorithm showed better performance than the hard vote algorithm in the experiments. The ROC analysis in Figure 5.10 shows that the maximum AUC was achieved in the low-mobility environment, with a value of 0.89.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 5. 10: a, b and c: ROC analysis for the models created by voting classifiers

5.2.9 Neural Networks

5.2.9.1 Perceptron

The simplest form of neural network architecture is called a perceptron. The architecture consists of two layers, input and output, and each layer consists of one or more nodes called neurons. The number of neurons in the input layer is the same as the number of features. The number of neurons in the output layer depends on the number of class labels. Each input node is connected to the output node using a weighted connection followed by a non-linear activation function (AF), which can be represented for binary output by a sign function [181]. The computation of the perceptron follows the following formula:

$$Z_i = \text{Sign}(\sum(\bar{W} \cdot \bar{X}_i + b_i)) - (4.1)$$

where $\bar{W}=(w_1, w_2 \dots w_n)$ is the set of n input weights and $\bar{X}_i=(x_{i1}, \dots, x_{in})$ is the feature (input) dataset. The training of a neural network is conducted by maximizing the classification accuracy by computing the weights. Finally, the weights are held fixed and the accuracy is evaluated by testing datasets, usually via cross-validation [182]. Weights optimization during training is performed iteratively through multiple epochs using one of the available Afs.

5.2.9.2 Multilayer Perceptron (MLP)

In contrast to the perceptron, the MLP has one or more layers between input and output layers called “hidden layers”. Figure 5.11 shows the architecture of the perceptron and the MLP. Determining the number of hidden layers and the number of neurons per layer is performed by considering a trade-off between complexity and cross-validated performance, because adding more hidden layers may increase the computational cost for building models.

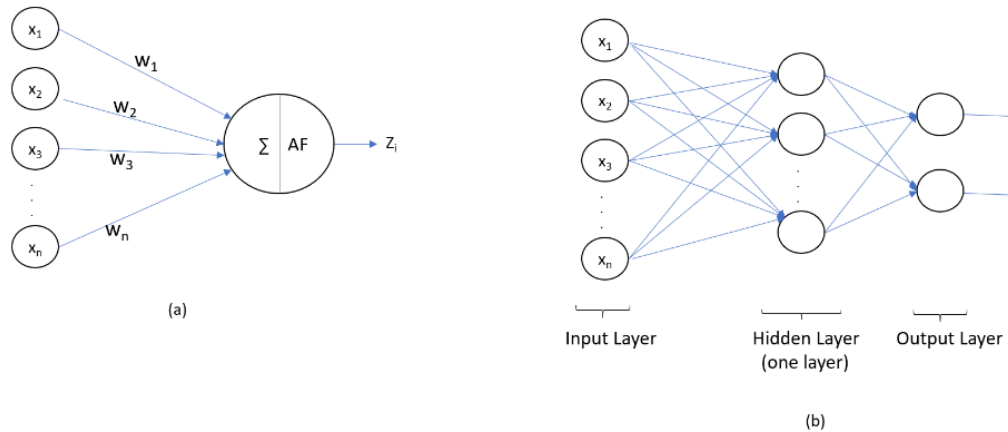


Figure 5. 11: (a) Perceptron and (b) MLP architectures

In this work we built our models using four Afs, including 1) sigmoid function, 2) hyperbolic tangent function (tanh), 3) rectified linear unit (ReLU), and 4) identity function, and compared the cross-validated performance.

Tables 5.10, 5.11 and 5.12 show the details of the results. As can be seen, the rows in these tables show the architecture (perceptron or MLP) and the type of AF for which the best-fit models were achieved in the experiments. Moreover, the first column of each of these tables shows the configuration of the neural networks in terms of the number of neurons in each layer. Our models were trained using a perceptron (no hidden layer), one hidden layer (with 10, 20, 30, 40 neurons), two hidden layers (with 10, 20, 30, 40 neurons) and three hidden layers (with 10, 20, 30, 40 neurons). We have conducted a number of experiments using more neurons and layers but the performance of the models did not change. We also built and tested our prediction models using 10-fold cross-validation for all prediction models. According to these considerations, the experiments were conducted 480 times per dataset. The calculation is as follows:

$$12 \text{ (configuration)} * 4 \text{ (AF)} * 10 \text{ (10-fold cross-validation)} = 480$$

Table 5.10: Performance of neural network models trained by the low-mobility dataset

	Configuration	Accuracy Rate	Precision		Recall		F1	
Single Layer No Hidden Layer (AF='identity')	3-2	80.08% (1.80%)	0	0.30	0	0.16	0	0.21
			1	0.85	1	0.93	1	0.89
One Hidden Layer (AF='ReLU')	3-10-2	75.03% (1.35%)	0	0.21	0	0.25	0	0.23
			1	0.85	1	0.82	1	0.83
	3-20-2	76.30% (1.79%)	0	0.23	0	0.15	0	0.18
			1	0.85	1	0.90	1	0.87
	3-30-2	77.34% (1.69%)	0	0.15	0	0.10	0	0.12
			1	0.84	1	0.89	1	0.86
	3-40-2	79.02% (1.40%)	0	0.18	0	0.09	0	0.12
			1	0.84	1	0.92	1	0.88
Two Hidden Layers (AF='tanh')	3-6-4-2	71.79% (2.04%)	0	0.18	0	0.24	0	0.20
			1	0.84	1	0.79	1	0.79
	3-12-8-2	73.93% (2.17%)	0	0.18	0	0.16	0	0.17
			1	0.84	1	0.86	1	0.85
	3-22-8-2	75.88% (2.21%)	0	0.13	0	0.12	0	0.13
			1	0.84	1	0.85	1	0.84
	3-30-10-2	76.82% (2.37%)	0	0.25	0	0.16	0	0.20
			1	0.85	1	0.91	1	0.88
Three Hidden Layers (AF='tanh')	3-5-3-2-2	69.75% (3.98%)	0	0.18	0	0.23	0	0.20
			1	0.84	1	0.80	1	0.82
	3-10-6-4-2	72.33% (1.95%)	0	0.21	0	0.28	0	0.24
			1	0.85	1	0.81	1	0.83
	3-15-10-5-2	73.63% (1.67%)	0	0.13	0	0.11	0	0.12
			1	0.84	1	0.86	1	0.85

0: Deny, 1: Access

Furthermore, the number of epochs is set to 500. This means that the model in each configuration was trained by 500 cycles using the whole training dataset to find the optimal weights and achieve better performance.

According to Table 5.10, the best performance was achieved by the perceptron, with 80.08%. Furthermore, as more hidden layers are added, the accuracy of the models gradually decreases. Moreover, in each MLP architecture (one hidden layer, two hidden layers and three hidden layers), adding more neurons gradually increases the accuracy of the models.

Table 5.11 shows the performance results for the neural network models trained and developed using the medium-mobility dataset. The most accurate model was achieved in the absence of any hidden layers, with 58.07% accuracy. For the neural network models trained by the high-mobility dataset, the best performance was achieved by the perceptron, with 55.77% accuracy.

Table 5.11: Performance of neural network models trained by the medium-mobility dataset

	Configuration	Accuracy Rate	Precision		Recall		F1	
Single Layer No Hidden Layer (AF='identity')	3-2	58.07% (2.29%)	0	0.70	0	0.78	0	0.74
			1	0.40	1	0.31	1	0.35
One Hidden Layer (AF='ReLU')	3-10-2	54.97% (2.53%)	0	0.71	0	0.70	0	0.71
			1	0.40	1	0.42	1	0.41
	3-20-2	55.13% (1.81%)	0	0.69	0	0.64	0	0.66
			1	0.35	1	0.41	1	0.38
	3-30-2	56.35% (2.49%)	0	0.67	0	0.68	0	0.68
			1	0.32	1	0.31	1	0.31
	3-40-2	56.41% (2.16%)	0	0.71	0	0.74	0	0.72
			1	0.41	1	0.38	1	0.39
Two Hidden Layers (AF='tanh')	3-6-4-2	54.31% (1.69%)	0	0.69	0	0.61	0	0.65
			1	0.35	1	0.44	1	0.39
	3-12-8-2	56.23% (2.24%)	0	0.69	0	0.67	0	0.68
			1	0.35	1	0.37	1	0.36
	3-22-8-2	55.69% (2.25%)	0	0.71	0	0.74	0	0.73
			1	0.42	1	0.39	1	0.41
	3-30-10-2	56.91% (2.24%)	0	0.70	0	0.76	0	0.73
			1	0.41	1	0.34	1	0.37
Three Hidden Layers (AF='tanh')	3-5-3-2-2	54.29% (1.72%)	0	0.68	0	0.64	0	0.66
			1	0.34	1	0.39	1	0.36
	3-10-6-4-2	53.73% (1.52%)	0	0.69	0	0.71	0	0.70
			1	0.36	1	0.34	1	0.35
	3-15-10-5-2	56.95% (2.32%)	0	0.70	0	0.68	0	0.69
			1	0.38	1	0.40	1	0.39

0: Deny, 1: Access

Table 5.12: Performance of neural network models trained by the high-mobility dataset

	Configuration	Accuracy Rate	Precision		Recall		F1	
Single Layer No Hidden Layer (AF='identity')	3-2	55.77% (1.61%)	0	0.64	0	0.73	0	0.68
			1	0.40	1	0.31	1	0.35
One Hidden Layer (AF='ReLU')	3-10-2	54.99% (1.87%)	0	0.65	0	0.64	0	0.65
			1	0.41	1	0.43	1	0.42
	3-20-2	53.37% (2.19%)	0	0.65	0	0.61	0	0.63
			1	0.40	1	0.44	1	0.42
	3-30-2	54.15% (1.71%)	0	0.65	0	0.70	0	0.67
			1	0.42	1	0.37	1	0.39
	3-40-2	55.53% (1.81%)	0	0.65	0	0.64	0	0.65
			1	0.39	1	0.38	1	0.39
Two Hidden Layers (AF='tanh')	3-6-4-2	52.95% (1.12%)	0	0.61	0	0.58	0	0.60
			1	0.35	1	0.38	1	0.37
	3-12-8-2	53.59% (1.37%)	0	0.66	0	0.67	0	0.67
			1	0.43	1	0.42	1	0.43
	3-22-8-2	55.53% (2.05%)	0	0.66	0	0.68	0	0.67
			1	0.42	1	0.39	1	0.41
	3-30-10-2	54.35% (2.10%)	0	0.64	0	0.62	0	0.63
			1	0.39	1	0.41	1	0.40
Three Hidden Layers (AF='tanh')	3-5-3-2-2	52.37% (2.83%)	0	0.63	0	0.59	0	0.61
			1	0.37	1	0.41	1	0.39
	3-10-6-4-2	52.71% (1.83%)	0	0.63	0	0.59	0	0.61
			1	0.37	1	0.41	1	0.39
	3-15-10-5-2	53.21% (1.14%)	0	0.63	0	0.61	0	0.62
			1	0.38	1	0.40	1	0.39

0: Deny, 1: Access

5.3 Discussion

As discussed in Subsection 5.2, 10 classification algorithms were applied to three datasets to build prediction models for environments with different degrees of user mobility (i.e., low, medium and high). Table 5.13 shows the aggregated performance results. It summarizes the performance of prediction models for three different datasets labelled as high mobility (H), medium mobility (M) and low mobility (L). The results are given for both classes – 0: Deny and 1: Access.

Table 5.13: Aggregated performance of the prediction models

Models	Accuracy Rate (Cross-Validated)			Precision				Recall				F1			
	L	M	H		L	M	H		L	M	H		L	M	H
Decision Tree	77.56%	55.87%	54.29%	0	0.35	0.72	0.67	0	0.39	0.66	0.67	0	0.37	0.69	0.67
				1	0.88	0.40	0.44	1	0.86	0.47	0.44	1	0.87	0.43	0.44
SVM	82.20%	64.27%	60.89%	0	0.00	0.67	0.62	0	0.00	0.99	0.97	0	0.00	0.80	0.76
				1	0.84	0.00	0.11	1	1.00	0.00	0.01	1	0.91	0.00	0.01
Logistic Regression	79.94%	57.81%	55.37%	0	0.30	0.70	0.64	0	0.19	0.77	0.72	0	0.23	0.73	0.68
				1	0.86	0.39	0.40	1	0.92	0.30	0.32	1	0.89	0.34	0.36
Naïve Bayes	80.42%	56.31%	53.49%	0	0.36	0.71	0.64	0	0.30	0.74	0.61	0	0.33	0.72	0.63
				1	0.87	0.41	0.39	1	0.90	0.37	0.41	1	0.89	0.39	0.40
AdaBoost	85.50%	60.33%	57.15%	0	0.69	0.69	0.63	0	0.30	0.79	0.75	0	0.42	0.74	0.69
				1	0.88	0.39	0.39	1	0.97	0.27	0.26	1	0.92	0.32	0.31
Random Forest	82.20%	64.43%	61.69%	0	0.00	0.67	0.63	0	0.00	1.00	1.00	0	0.00	0.81	0.77
				1	0.84	0.00	0.00	1	0.10	0.00	0.00	1	0.91	0.00	0.00
K-NN	77.68%	55.87%	53.63%	0	0.17	0.64	0.62	0	0.09	0.63	0.70	0	0.12	0.63	0.66
				1	0.84	0.27	0.36	1	0.92	0.28	0.28	1	0.88	0.27	0.32
ANN	80.08%	58.07%	55.51%	0	0.30	0.70	0.64	0	0.16	0.78	0.73	0	0.21	0.74	0.68
				1	0.85	0.40	0.40	1	0.93	0.31	0.31	1	0.89	0.35	0.35
Gradient Boost	82.20%	64.43%	61.69%	0	0.00	0.67	0.63	0	0.00	1.00	1.00	0	0.00	0.81	0.77
				1	0.84	0.00	0.00	1	1.00	0.00	0.00	1	0.91	0.00	0.00
Voting Classifier	84.58%	61.07%	57.83%	0	0.77	0.70	0.64	0	0.12	0.90	0.83	0	0.22	0.79	0.72
				1	0.86	0.48	0.42	1	0.99	0.20	0.21	1	0.92	0.28	0.28

0: Deny, 1: Access

As shown in the above table, for all datasets the boosting classifiers (gradient boost and AdaBoost) showed the best performance in predicting the label of the testing data samples. Moreover, random forest showed the second-highest value for accuracy in the low-mobility environment, whereas it showed the best performance in the medium- and high-mobility environments.

Figures 5.12, 5.13 and 5.14 show the aggregated ROC curve analysis for the “Access” label. As shown, AdaBoost has the maximum AUC in the low-mobility environment, with a value of 0.64. Also in the low-mobility environment, its curve (green) dominates the other curves.

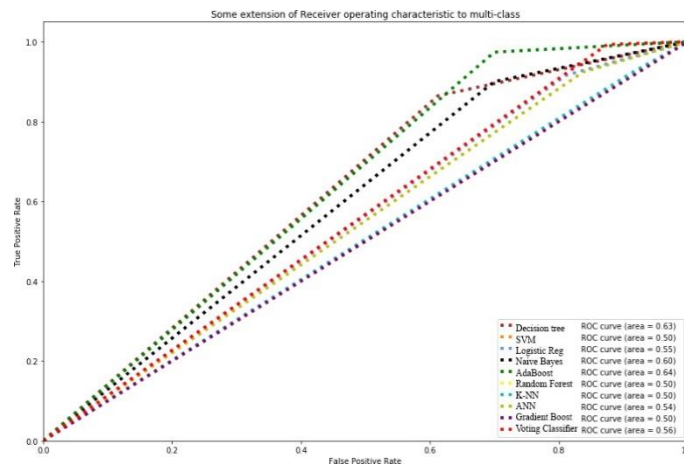


Figure 5.12: ROC analysis for prediction models in the low-mobility environment

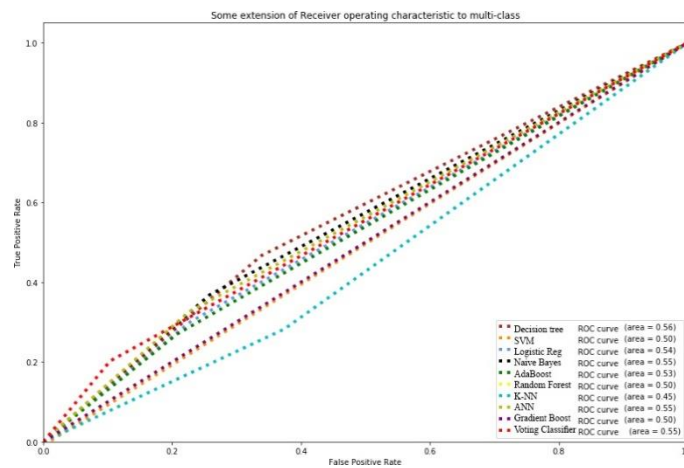


Figure 5.13: ROC analysis for prediction models in the medium-mobility environment

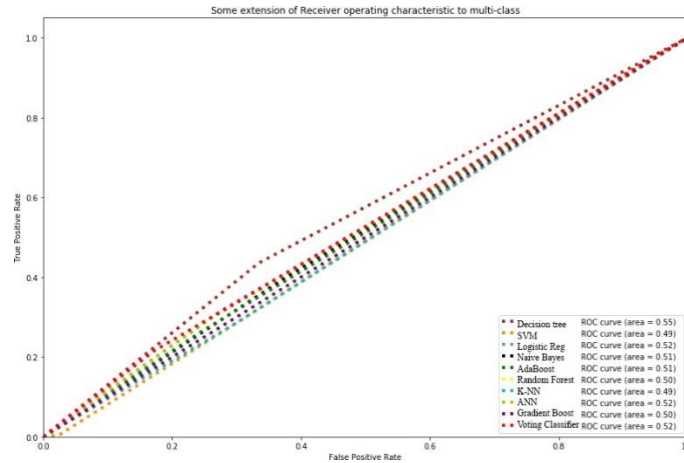


Figure 5. 124: ROC analysis for prediction models in the high-mobility environment

Figure 5.13 shows the aggregated ROC curves for the prediction models in the medium-mobility environment. The gradient boost curve dominates the other curves and confirms its superiority with a higher TP rate and lower FP rate than the rest of the algorithms. For the high-mobility environment, as depicted in Figure 5.14, the decision tree dominates the other curves and has the maximum AUC (0.55). However, the model created using the decision tree cannot be chosen as the best-fit model because a curve dominates in ROC space if and only if its precision dominates in precision space [183]. In this case we do not have precise precision values because of the associated noise.

We also applied the Kruskal–Wallis test (Table 5.14) to investigate the effects of the number of hidden layers on the accuracy of the model for all datasets. According to the results, the Asymp. Sig. (p-value) is less than 0.05, and therefore the null assumption is rejected, and it shows that the means for different accuracy groups is not the same. Furthermore, we applied the Spearman test (Table 5.15) to analyse the correlation between the number of hidden layers and the accuracy of the model. The results indicate that the number of hidden layers has an impact on the accuracy of the model (Sig.=0.000 and sig. < 0.05) in each environment.

Table 5.14: Kruskal–Wallis test results, grouping variable: no. of hidden layers

Chi Square	258.440
df	3
Asymp. Sig.	0.000

Table 5.15: Spearman test results: correlation

*** Correlation is significant at the 0.01 level (2-tailed)*

			Hidden Layer	Accuracy
Spearman's rho	Hidden Layer	Correlation Coefficient	1.000	0.152**
		Sig. (2-tailed)	.	0.000
		N	2600	2600
	Accuracy	Correlation Coefficient	0.152**	1.000
		Sig. (2-tailed)	0.000	.
		N	2600	2600

One of the challenges in applying prediction models in the projects is determining the frequency of re-training models using new datasets. The main reason for re-training is that the performance of the prediction models is degrading over time which is called “model drift”. This happens due to the changes in the environment that violates the model’s assumptions. To detect model drift, the accuracy of the model needs to be monitored. The frequency of re-training may vary from one case study to another. Determining the frequency needs to measure a threshold of divergence of the accuracy between the model working online and the model working with training datasets.

5.4 Chapter Summary

In this chapter, we presented details of our prediction models developed for handling uncertainty. As discussed, ten classification algorithms were applied to create our prediction models. Furthermore, three datasets were used for the sake of training/test processes. These datasets were synthesized using methodology discussed in Chapter 4 for three degrees of mobility (high, medium and low).

According to the cross-validated results, AdaBoost classifier showed the highest performance in terms of accuracy for low mobility environment with accuracy of 85.50%. The highest performance in medium mobility environment was reported for Boosting classifiers (both Adaboost and Gradient) with accuracy of 64.43%. In high mobility environment, Gradient Boost and random forest algorithms showed the highest accuracy (61.69%).

We also studied the behavior of our models created by neural networks (both perceptron and MLP). We applied 12 configurations of neural networks including different number of hidden layers (up to 3 hidden layers) different number of neurons (up to 40 neurons) and different types of activation functions (4 AFs) to investigate the effects of these variants on the performance of the prediction models. The results showed that the number of hidden layers affects on the performance of the model.

6. Indeterminacy-Aware Prediction Model for Authentication

In this chapter we describe how we built our indeterminacy-aware prediction model on top of the prediction models created in Chapter 5. In doing so, we consider the history profile of users in terms of past successful authentication, in addition to time, location and credential. As discussed in the previous chapter, we used classification algorithms to train and build our models. By enriching our datasets with a new data attribute, we expected to build more accurate prediction models.

6.1 Trust-Based Analysis

As discussed in Chapter 2, there is a need to propose new approaches to assess “trust” as the scalability, complexity, dynamism, heterogeneity, pervasiveness and automation of computer and communication systems increases in IoT. In traditional and emerging computer and communication systems, a number of approaches have addressed the advantages of considering “trust” in the field of access control [184], [185], [186], [113], [187], [188]. With reference to these studies, a taxonomy of trust-based analysis was given in Subsection 2.2.2.1. Based on this taxonomy, we used soft trust method by conducting behavioural-based analysis to assess the degree of trust for authentication requests and try to build our prediction model on top of that. In doing so, we kept a record of the access/deny history of each user in all of our datasets. Then, we used these data to measure the trust values for the users. Next, the trust values calculated were added to the datasets as a new attribute.

We generated 60 users for each of the datasets. The process of synthesizing these users was comprehensively discussed in Chapter 4. Tables 6.1, 6.2 and 6.3 show the details of the behaviour of these users. The first column of each of these tables indicates the ID of the users. The second column shows the total number of authentication requests in the dataset. The third column indicates the total number of successful authentication attempts for each user. Finally, the last column shows the ratio of successful authentication attempts per users. The total trust score for each dataset can be calculated by the following formula:

$$Total\ Trust\ Score = \frac{Total\ number\ of\ successful\ authentications}{Total\ number\ of\ authentication\ requests}$$

Table 6.1: History profile of users in the low-mobility dataset

ID	Total No. of Requests	Total No. of Successful Authentications	Authentication Rate
1	352	279	0.792614
2	314	258	0.821656
3	298	248	0.832215
4	228	197	0.864035
5	229	190	0.829694
6	173	147	0.849711
7	180	153	0.85
8	171	144	0.842105
9	153	128	0.836601
10	106	91	0.858491
11	136	118	0.867647
12	117	96	0.820513
13	129	109	0.844961
14	111	93	0.837838
15	116	90	0.775862
16	75	61	0.813333
17	97	76	0.783505
18	102	79	0.77451
19	85	73	0.858824
20	62	55	0.887097
21	79	65	0.822785
22	83	68	0.819277
23	74	66	0.891892
24	73	62	0.849315
25	58	47	0.810345
26	71	58	0.816901
27	55	48	0.872727
28	43	37	0.860465
29	62	56	0.903226
30	48	40	0.833333
31	61	57	0.934426
32	46	42	0.913043
33	46	36	0.782609
34	35	31	0.885714
35	42	38	0.904762
36	44	37	0.840909
37	24	20	0.833333
38	39	27	0.692308
39	36	29	0.805556
40	38	31	0.815789
41	36	30	0.833333
42	45	37	0.822222
43	47	41	0.87234
44	37	30	0.810811
45	37	31	0.837838

46	38	27	0.710526
47	32	29	0.90625
48	45	36	0.8
49	22	19	0.863636
50	51	40	0.784314
51	31	24	0.774194
52	34	28	0.823529
53	44	31	0.704545
54	26	23	0.884615
55	28	23	0.821429
56	30	26	0.866667
57	27	20	0.740741
58	39	32	0.820513
59	27	21	0.777778
60	33	26	0.787879

In the low-mobility dataset, 4,154 out of 5,000 authentication requests were authenticated. Thus, the total trust score for this dataset is as follows:

$$\begin{aligned}
 \text{Total Trust Score} &= \frac{4154}{5000} \\
 &= 0.8308
 \end{aligned}$$

The above ratio can be used as the trust threshold for the low-mobility dataset based on the last 5,000 records. This value may change from one dataset to another based on changes in the user distribution, user mobility pattern and time of the authentication request.

Table 6.2 shows details of the history profile of the users in the medium-mobility dataset, in which, 1,778 out of 5,000 requests were successfully authenticated. Therefore, the total trust score for this dataset is calculated as follows:

$$\begin{aligned}
 \text{Total Trust Score} &= \frac{1778}{5000} \\
 &= 0.3556
 \end{aligned}$$

Table 6.2: History profile of users in the medium-mobility dataset

ID	Total No. of Requests	Total No. of Successful Authentications	Authentication Rate
1	408	147	0.360294
2	324	111	0.342593
3	271	94	0.346863
4	250	94	0.376
5	190	63	0.331579
6	210	72	0.342857
7	185	59	0.318919
8	148	37	0.25
9	156	60	0.384615
10	151	50	0.331126
11	139	48	0.345324
12	106	39	0.367925
13	114	41	0.359649
14	97	39	0.402062
15	90	33	0.366667
16	98	32	0.326531
17	82	31	0.378049
18	98	38	0.387755
19	101	40	0.39604
20	74	26	0.351351
21	64	22	0.34375
22	73	25	0.342466
23	60	18	0.3
24	63	25	0.396825
25	62	23	0.370968
26	69	23	0.333333
27	60	21	0.35
28	57	18	0.315789
29	54	22	0.407407
30	65	26	0.4
31	57	32	0.561404
32	37	15	0.405405
33	48	16	0.333333
34	39	14	0.358974
35	32	9	0.28125
36	39	13	0.333333
37	52	13	0.25
38	53	19	0.358491
39	41	14	0.341463
40	41	18	0.439024
41	44	15	0.340909
42	36	10	0.277778
43	34	13	0.382353
44	33	12	0.363636
45	29	11	0.37931

46	28	12	0.428571
47	27	9	0.333333
48	47	15	0.319149
49	32	10	0.3125
50	32	10	0.3125
51	23	9	0.391304
52	37	13	0.351351
53	37	16	0.432432
54	34	14	0.411765
55	33	15	0.454545
56	25	10	0.4
57	29	10	0.344828
58	30	10	0.333333
59	24	7	0.291667
60	28	17	0.607143

Table 6.3 summarizes the history profile of users in the high-mobility dataset. Based on the statistics, 1,915 out of 5,000 were authenticated successfully. Thus, the total trust score for this dataset is as follows:

$$\begin{aligned}
 \text{Total Trust Score} &= \frac{1915}{5000} \\
 &= 0.383
 \end{aligned}$$

The trust value for each user is determined by comparing its authentication rate with the total trust score (threshold). In the other words, if the authentication rate of the user is greater than the threshold of the dataset then the trust value for that user is set to 1, otherwise 0. For example, the authentication rates for user “25” in the three datasets are 0.810345, 0.370968 and 0.5 and the threshold values of these datasets are 0.8308, 0.3556 and 0.383 respectively. Therefore, the trust values for user “25” would be 0 (in the low-mobility dataset), 1 (in the medium-mobility dataset) and 1 (in the high-mobility dataset).

According to the above-mentioned, a new column headed “trust” was added to all of our datasets. Therefore, the classification algorithms discussed in Chapter 5 will need to be trained in order to build new prediction models.

Table 6.3: History profile of users in the high-mobility dataset

ID	Total No. of Requests	Total No. of Successful Authentications	Authentication Rate
1	386	139	0.360103627
2	307	108	0.351791531
3	292	103	0.352739726
4	253	107	0.422924901
5	210	83	0.395238095
6	180	81	0.45
7	165	58	0.351515152
8	158	57	0.360759494
9	148	56	0.378378378
10	148	54	0.364864865
11	125	47	0.376
12	126	45	0.357142857
13	120	52	0.433333333
14	109	39	0.357798165
15	97	41	0.422680412
16	85	41	0.482352941
17	80	28	0.35
18	84	35	0.416666667
19	81	27	0.333333333
20	97	38	0.391752577
21	73	30	0.410958904
22	73	30	0.410958904
23	67	27	0.402985075
24	70	31	0.442857143
25	58	24	0.413793103
26	56	28	0.5
27	62	22	0.35483871
28	57	23	0.403508772
29	57	22	0.385964912
30	70	32	0.457142857
31	56	24	0.428571429
32	46	12	0.260869565
33	44	18	0.409090909
34	46	22	0.47826087
35	53	19	0.358490566
36	43	18	0.418604651
37	49	22	0.448979592
38	58	25	0.431034483
39	30	8	0.266666667
40	17	5	0.294117647
41	57	20	0.350877193
42	48	17	0.354166667
43	38	19	0.5
44	32	11	0.34375
45	35	13	0.371428571

46	50	18	0.36
47	23	8	0.347826087
48	46	15	0.326086957
49	24	12	0.5
50	41	21	0.512195122
51	30	7	0.233333333
52	23	8	0.347826087
53	21	5	0.238095238
54	19	6	0.315789474
55	23	8	0.347826087
56	32	9	0.28125
57	26	8	0.307692308
58	29	12	0.413793103
59	33	7	0.212121212
60	34	10	0.294117647

6.2 Indeterminacy-Aware Prediction Models

In this subsection, we describe how we applied the 10 classification algorithms discussed in Chapter 5 to our new datasets and measured the performance of the new prediction models. As mentioned earlier, the new datasets contained “trust” values in addition to the “time”, “location” and “credentials” values. Moreover, a 10-fold cross-validation method was applied to evaluate the generalizability of the model.

6.2.1 Decision Tree

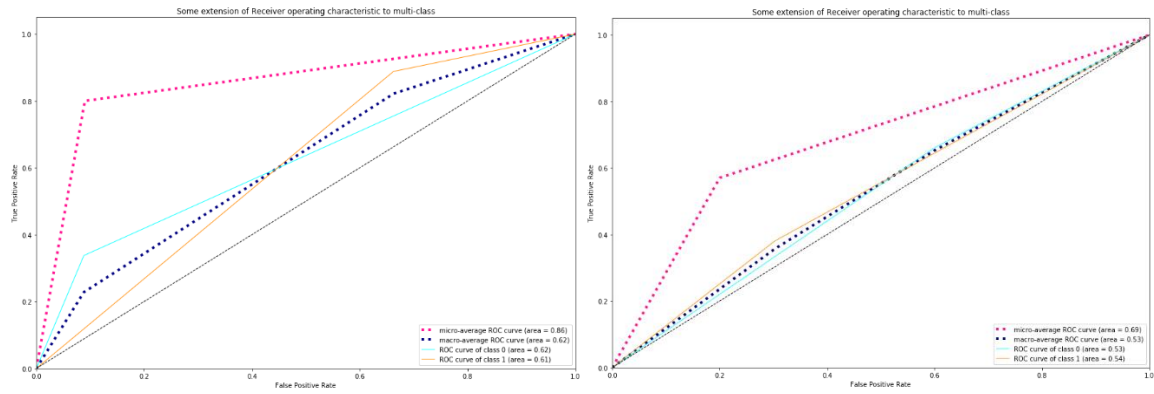
Table 6.4 shows the performance of the prediction models trained and built using the decision tree (CART algorithm). According to the results, the prediction models trained and built using the “trust” attribute perform better in terms of accuracy, precision and recall in comparison with those developed by datasets without “trust” values. Moreover, the performance of the prediction models is decreased by increasing mobility in the environment. The same trend was seen in uncertainty-aware prediction models built using the decision tree algorithm.

Figure 6.1 shows ROC curves for the “Access” class. As shown, the AUC values for the micro-average curves in all prediction models are greater than the same values for the prediction models built by the previous datasets.

Table 6.4: Performance of the prediction model trained by the new datasets (decision tree)

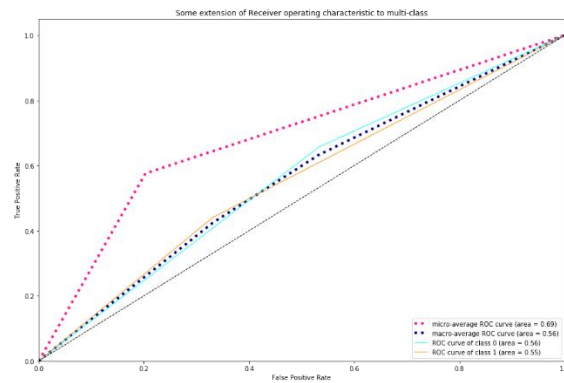
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	78.92% (1.69%)	0	0.36	0	0.34	0	0.35
		1	0.88	1	0.89	1	0.88
Medium Mobility	56.29% (2.62%)	0	0.70	0	0.70	0	0.70
		1	0.38	1	0.38	1	0.38
High Mobility	55.27% (2.52%)	0	0.67	0	0.67	0	0.67
		1	0.44	1	0.44	1	0.44

0: Deny, 1: Access



a. Low mobility

b. Medium mobility



c. High mobility

Figure 6. 1: a, b and c: ROC analysis for the decision-tree-based models

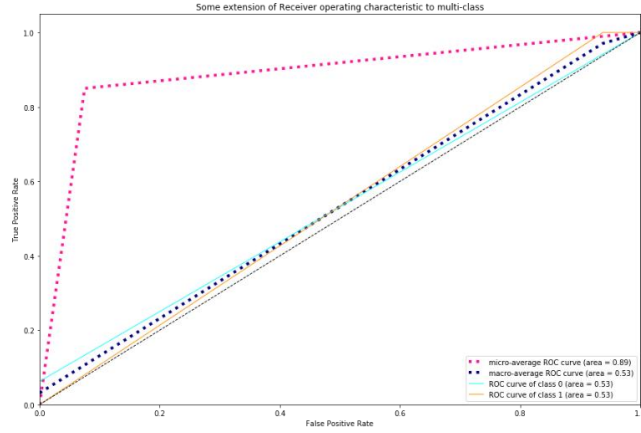
6.2.2 Random Forest

Table 6.5 shows that the best performance was achieved by the random forest algorithm using the new datasets. According to the results, the accuracy of the prediction models trained by the new dataset is better than the accuracy of the models built by the uncertainty-aware dataset for the low-mobility environment. For the medium- and high-mobility datasets, the results are approximately the same as the previous models built in Chapter 5. Figure 6.2 depicts the ROC curve analysis of prediction models for the “Access” class. The AUC value of the prediction model trained by the low-mobility dataset is greater than the AUC of the same model built by the uncertainty-aware dataset. For medium- and high-mobility environments, the AUC values of the prediction models are the same as the AUC values of the uncertainty-aware prediction models.

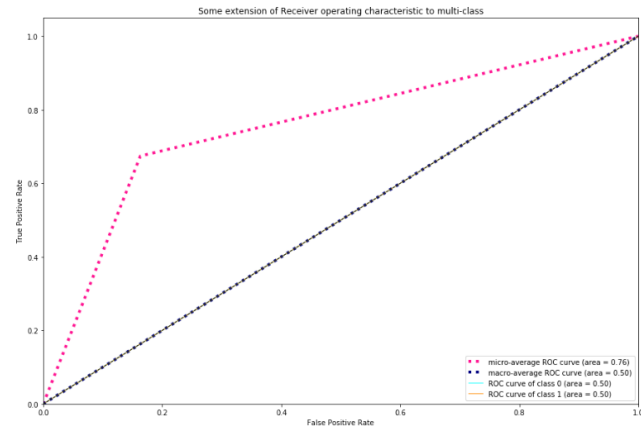
Table 6.5: Performance of the prediction model trained by the low-mobility dataset (random forest)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	82.58% (1.17%)	0	1.00	0	0.06	0	0.12
		1	0.85	1	1.00	1	0.92
Medium Mobility	64.43% (2.63%)	0	0.67	0	1.00	0	0.81
		1	0.00	1	0.00	1	0.00
High Mobility	61.67% (2.23%)	0	0.63	0	1.00	0	0.77
		1	0.00	1	0.00	1	0.00

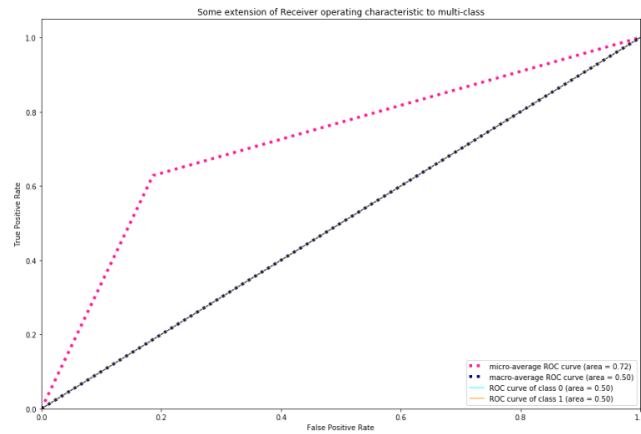
0: Deny, 1: Access



a. Low mobility



b. Medium mobility



c. High mobility

Figure 6. 2: a , b and c: ROC analysis for the random-forest-based models

6.2.3 Support Vector Machine (SVM)

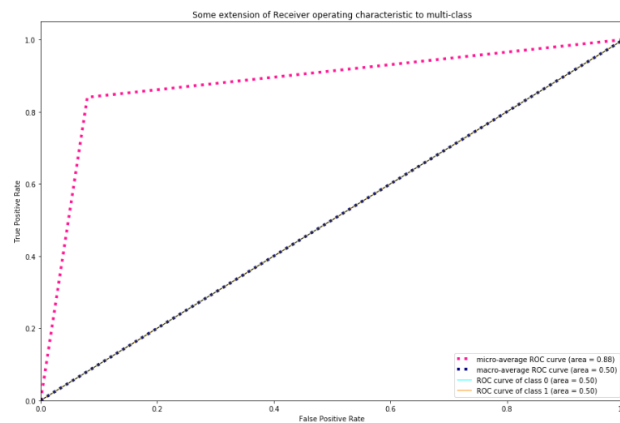
Table 6.6 shows the performance of the models trained by the SVM algorithm using our new datasets. According to the results, in the low-mobility environment, SVM-based models show the same performance as the prediction models developed in Chapter 5. For medium- and high-mobility environments, the models perform better in terms of accuracy, precision and recall. Figure 6.3 shows the ROC analysis for the SVM-based models.

Table 6.6: Performance of the prediction models trained by three datasets (SVM)

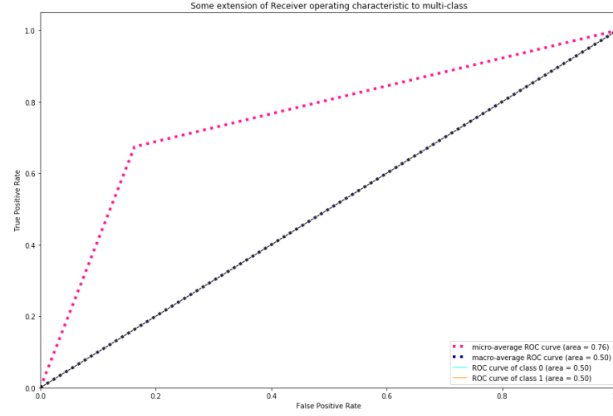
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	82.20% (1.07%)	0	0.00	0	0.00	0	0.00
		1	0.84	1	1.00	1	0.91
Medium Mobility	64.43% (2.63%)	0	0.67	0	1.00	0	0.81
		1	0.00	1	0.00	1	0.00
High Mobility	61.71% (2.20%)	0	0.63	0	1.00	0	0.77
		1	0.00	1	0.00	1	0.00

0: Deny, 1: Access

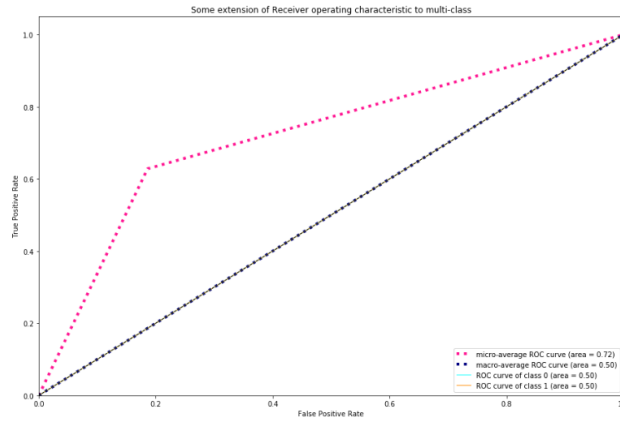
As can be seen in Figure 6.3, the AUC values for micro-average curves in the low-, medium- and high-mobility environments are slightly higher than the values measured in Chapter 5.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 6. 3: a, b and c: ROC analysis for the SVM-based models

6.2.4 Logistic Regression

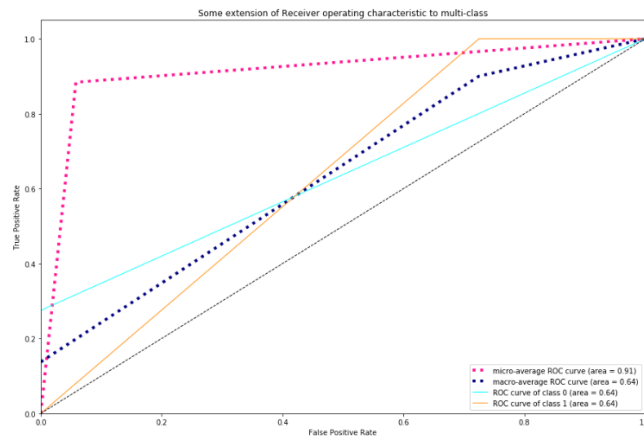
According to the data, logistic regression showed dramatically better results with the new datasets. Table 6.7 shows the results achieved. For the low-mobility dataset, the accuracy of the new model is 86.90%, whereas the accuracy of the past logistic regression model was 79.94%. For medium- and high-mobility environments, the accuracy of the models increased by 6.62% and 6.34% respectively.

Table 6.7: Performance of the prediction models trained by three datasets (logistic regression)

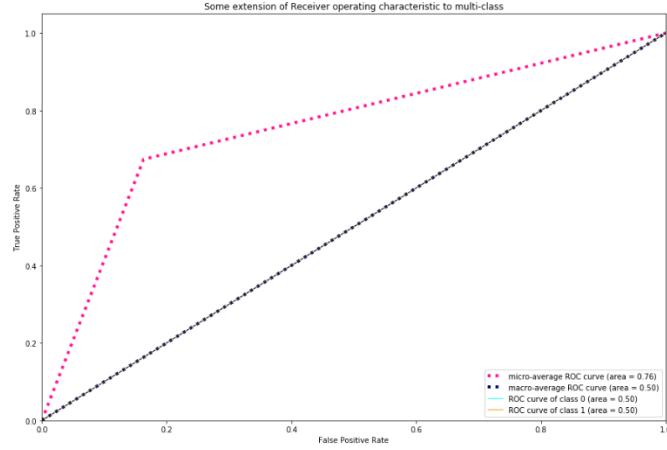
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	86.90% (0.99%)	0	1.00	0	0.28	0	0.43
		1	0.88	1	1.00	1	0.94
Medium Mobility	64.43% (2.63%)	0	0.67	0	1.00	0	0.81
		1	0.00	1	0.00	1	0.00
High Mobility	61.71% (2.20%)	0	0.63	0	1.00	0	0.77
		1	0.00	1	0.00	1	0.00

0: Deny, 1: Access

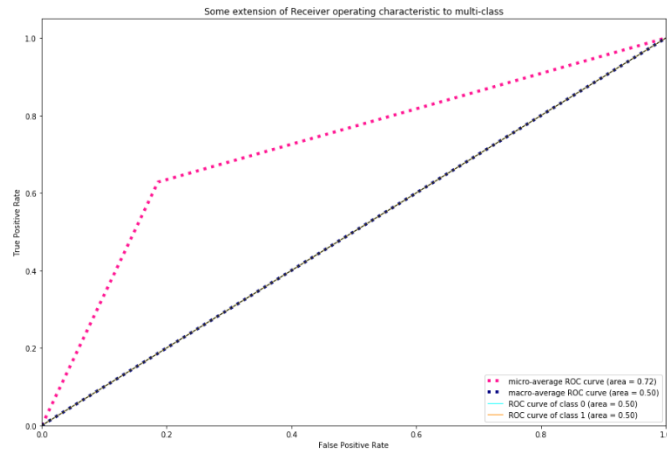
Figure 6.4 shows the ROC analysis of the prediction models developed by the logistic regression algorithm. As expected, the AUC values for the following models are greater than the values achieved by the datasets that were used in Chapter 5.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 6. 4: a, b and c: ROC analysis for the logistic-regression-based models

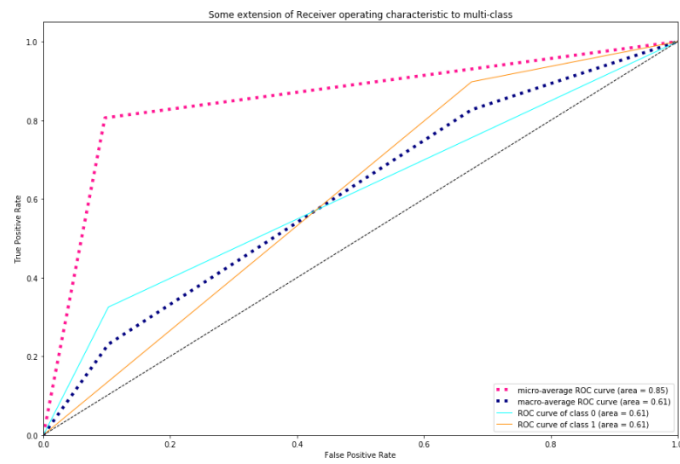
6.2.5 Naïve Bayes

As shown in Table 6.8, models developed by Naïve Bayes showed better accuracy in the medium- and high-mobility environments than the models trained by uncertainty-aware datasets. For the low-mobility environment, the accuracy of the new model is slightly lower than the accuracy of the past model, by 0.54%. The results are confirmed through ROC curve analysis for low- and high-mobility environments. Figure 6.5 demonstrates the ROC curves for these models.

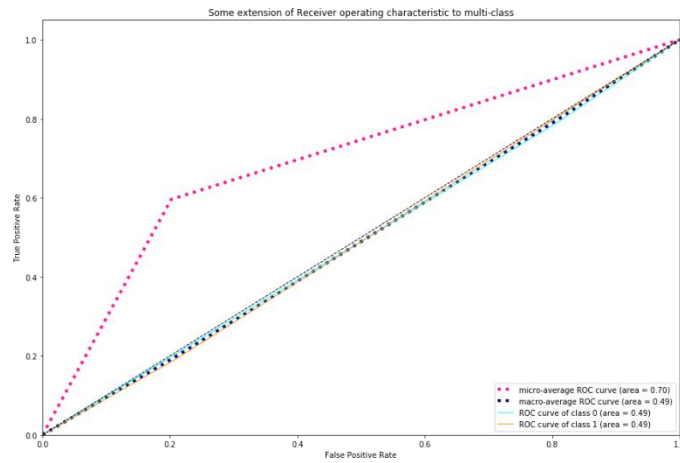
Table 6.8: Performance of the prediction models trained by three datasets (Naïve Bayes)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	79.40% (2.03%)	0	0.38	0	0.33	0	0.35
		1	0.87	1	0.90	1	0.89
Medium Mobility	59.13% (3.36%)	0	0.67	0	0.79	0	0.73
		1	0.31	1	0.19	1	0.23
High Mobility	58.09% (1.54%)	0	0.66	0	0.70	0	0.68
		1	0.43	1	0.38	1	0.40

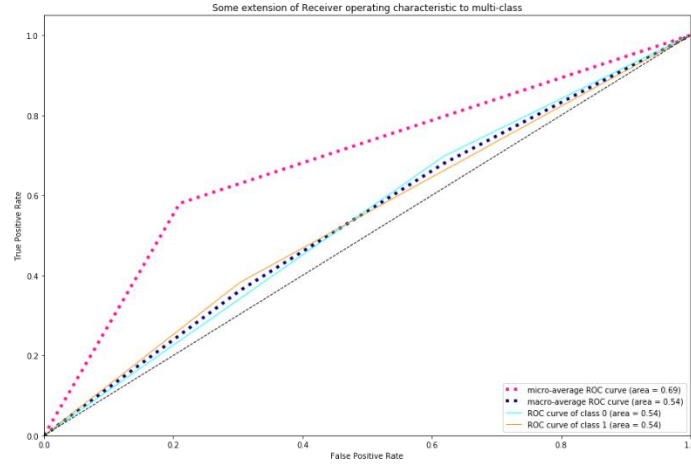
0: Deny, 1: Access



a. Low mobility



b. Medium mobility



c. High mobility

Figure 6. 5: a, b and c: ROC analysis for the models created by the Naïve Bayesian algorithm

6.2.6 K-Nearest Neighbours (K-NN)

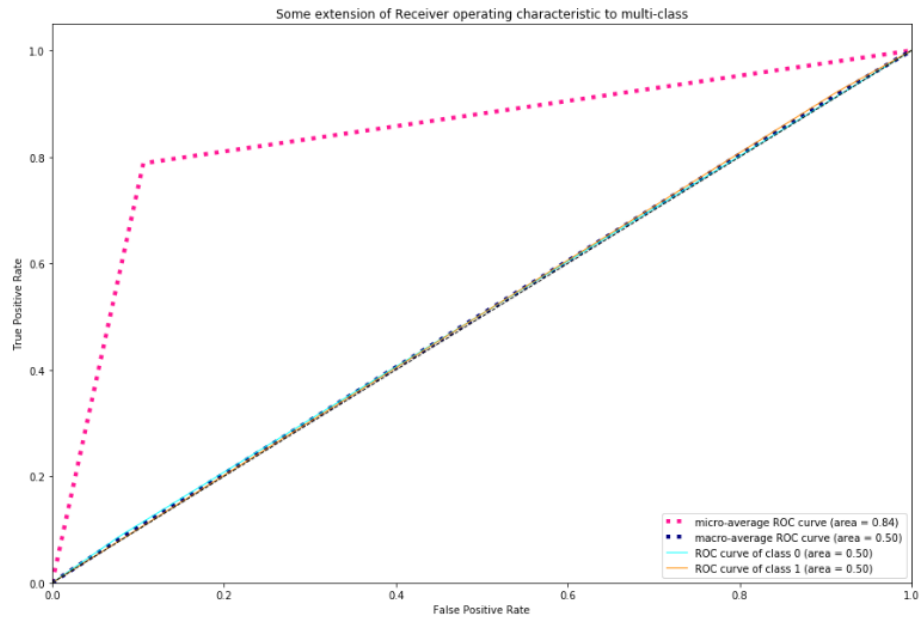
As summarized in Table 6.9, prediction models developed by the K-NN algorithm perform better for all datasets than the K-NN models trained by uncertainty-aware datasets. Accuracy of the new models increases by at least 2%. Moreover, the precision of these models is better than the precision of the past K-NN models, reported in Chapter 5.

Table 6.9: Performance of the prediction models trained by the three datasets (K-NN)

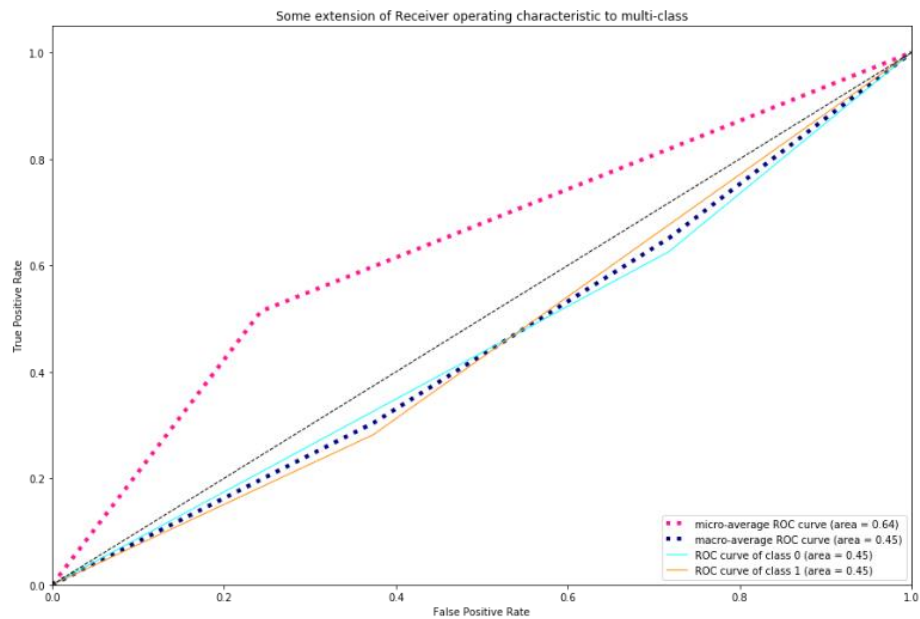
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	79.12% (2.10%)	0	0.32	0	0.15	0	0.21
		1	0.85	1	0.94	1	0.89
Medium Mobility	57.29% (2.02%)	0	0.66	0	0.67	0	0.66
		1	0.28	1	0.27	1	0.28
High Mobility	55.95% (1.90%)	0	0.63	0	0.65	0	0.64
		1	0.37	1	0.34	1	0.36

0: Deny, 1: Access

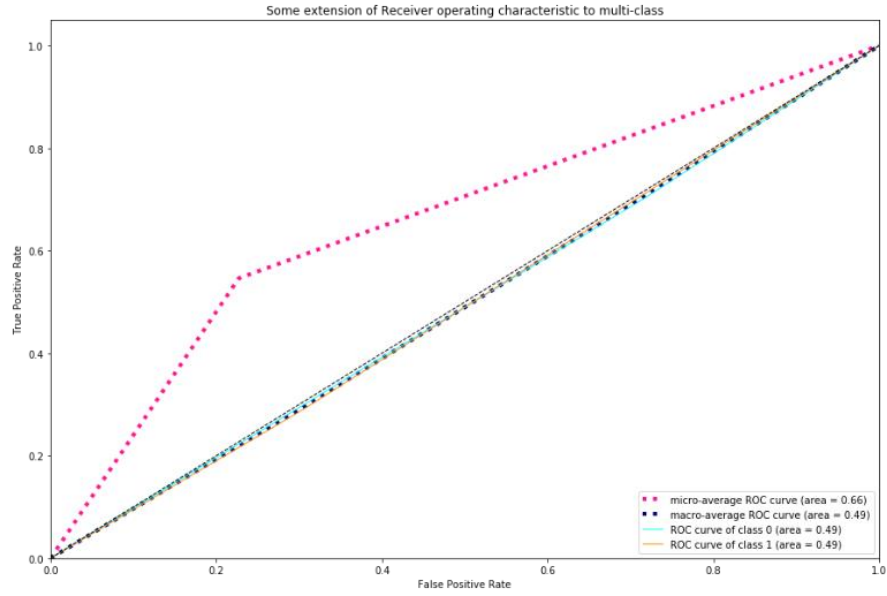
ROC curves for developed models are shown in Figure 6.6. According to the AUC values of micro-average ROC curves, the AUC for all models remained unchanged in comparison with the models presented in Chapter 5.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 6. 6: a, b and c: ROC analysis for the models created by the K-NN algorithms

6.2.7 Boosting Algorithms

We applied AdaBoost and gradient boost to our new datasets. Tables 6.10 and 6.11 show in detail the performance of the developed prediction models for these two classifiers.

Table 6.10: Performance of the prediction models trained by the three datasets (gradient boost)

	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	86.90% (1.06%)	0	1.00	0	0.28	0	0.43
		1	0.88	1	1.00	1	0.94
Medium Mobility	64.39% (2.61%)	0	0.67	0	1.00	0	0.81
		1	0.00	1	0.00	1	0.00
High Mobility	61.59% (2.28%)	0	0.63	0	1.00	0	0.77
		1	0.00	1	0.00	1	0.00

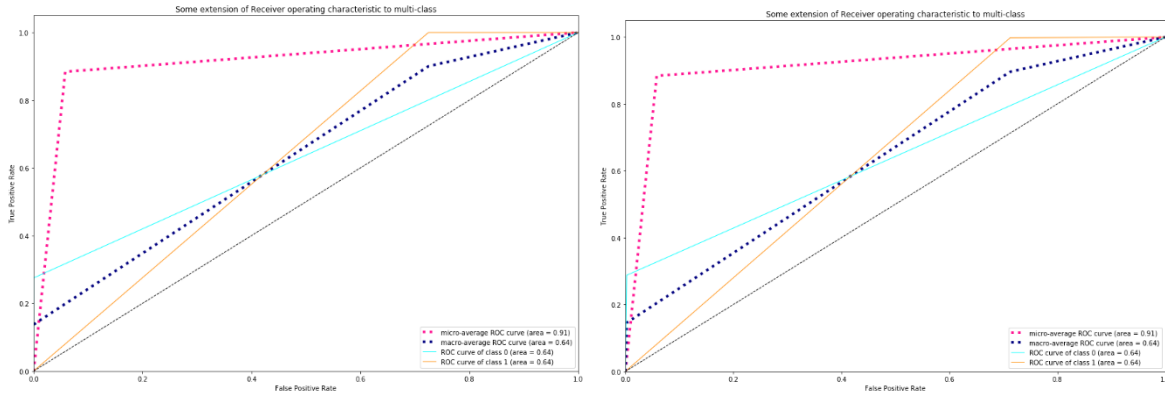
0: Deny, 1: Access

Table 6.11: Performance of the prediction models trained by the three datasets (AdaBoost)

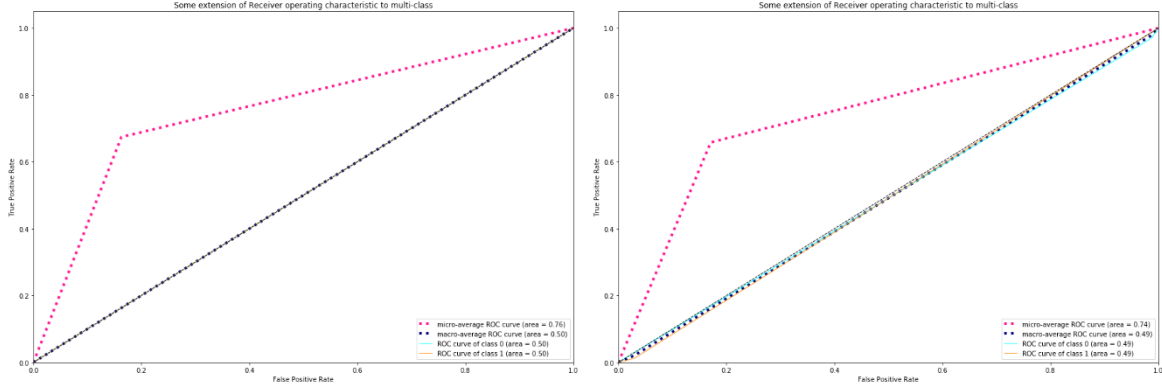
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	86.76% (1.11%)	0	0.96	0	0.29	0	0.44
		1	0.88	1	1.00	1	0.94
Medium Mobility	63.73% (2.23%)	0	0.67	0	0.97	0	0.79
		1	0.17	1	0.01	1	0.02
High Mobility	60.59% (2.24%)	0	0.64	0	0.93	0	0.76
		1	0.46	1	0.10	1	0.17

0: Deny, 1: Access

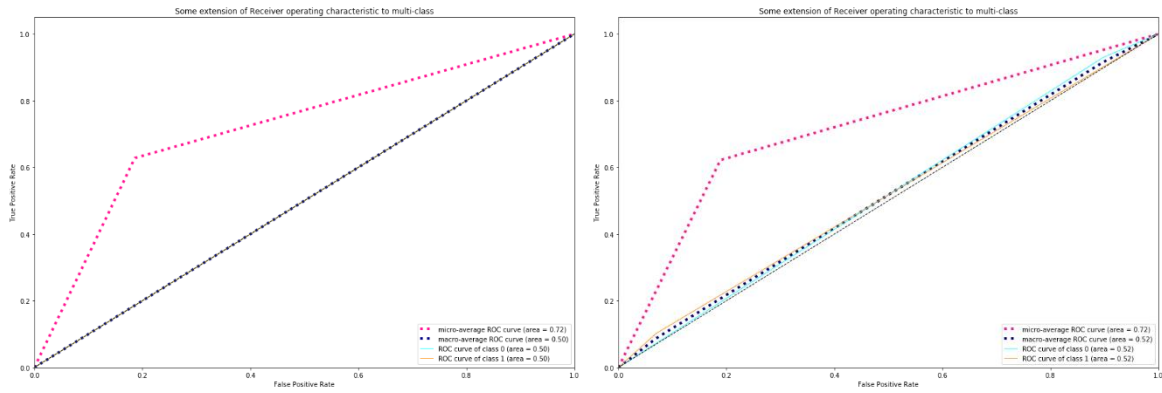
According to the prediction models developed by the gradient boost algorithm, the model trained by the new low-mobility dataset showed better performance than past models. Moreover, newly developed prediction models built by the AdaBoost algorithm showed better performance in all environments than past models. In contrast to the relevant models developed in Chapter 5, the new gradient boost models outperform the new AdaBoost models in terms of accuracy, precision, recall and F1. Figure 6.7 shows corresponding ROC curves for these two classifiers.



a. Gradient boost (left) and AdaBoost (right) – low mobility



b. Gradient boost (left) and AdaBoost (right) – medium mobility



c. Gradient boost (left) and AdaBoost (right) – high mobility

Figure 6. 7: a, b and c: ROC analysis for the models created by gradient boost and AdaBoost

Comparing the AUC values in Figure 6.7 with those in Figure 5.9 reveals that the AdaBoost classifier performs better with the new datasets than with the past datasets. For the new boosting models, gradient boost classifiers perform slightly better than the AdaBoost classifier in terms of AUC.

6.2.8 Voting Classifier

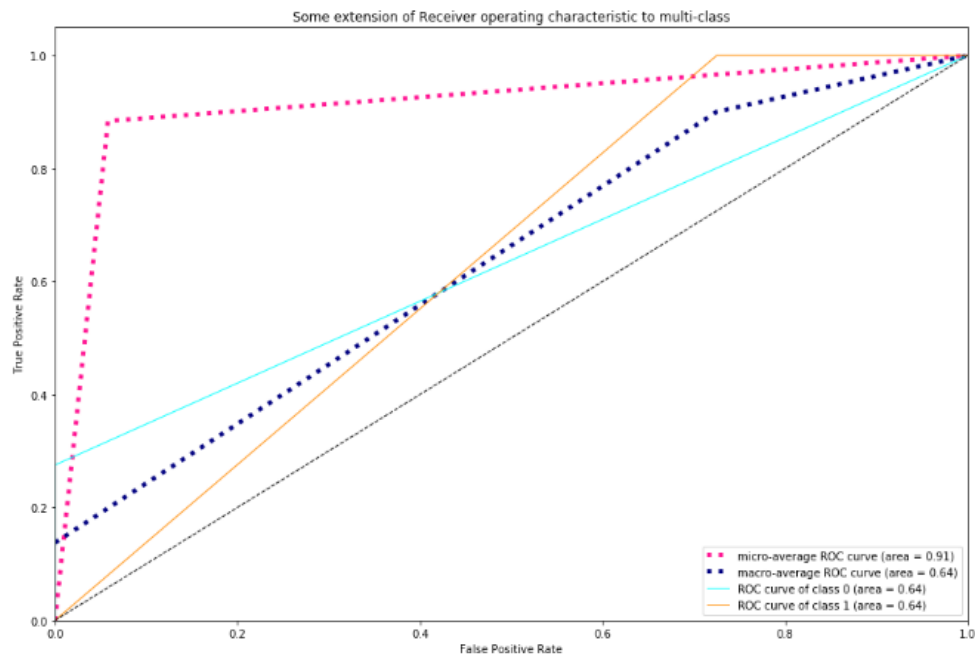
Similar to the models developed in Chapter 5, the soft mode voting algorithms showed better performance than the hard mode ones in building prediction models for new experiments. Table 6.12 shows the performance of the models.

Table 6.12: Performance of the prediction models trained by the three datasets (voting classifier)

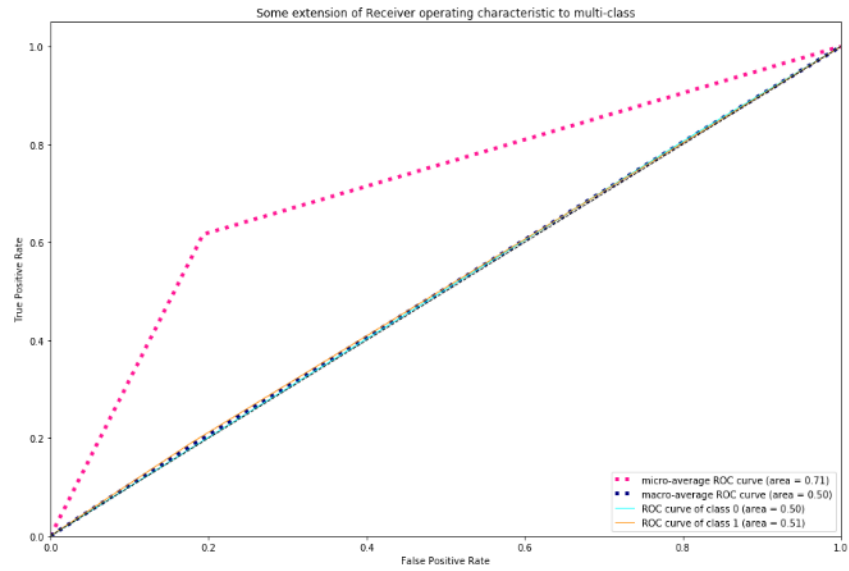
	Accuracy (Cross-Validated)	Precision		Recall		F1	
Low Mobility	86.90% (1.04%)	0	1.00	0	0.28	0	0.43
		1	0.88	1	1.00	1	0.94
Medium Mobility	59.83% (2.76%)	0	0.67	0	0.82	0	0.74
		1	0.33	1	0.18	1	0.23
High Mobility	57.57% (2.25%)	0	0.66	0	0.76	0	0.71
		1	0.46	1	0.33	1	0.39

0: Deny, 1: Access

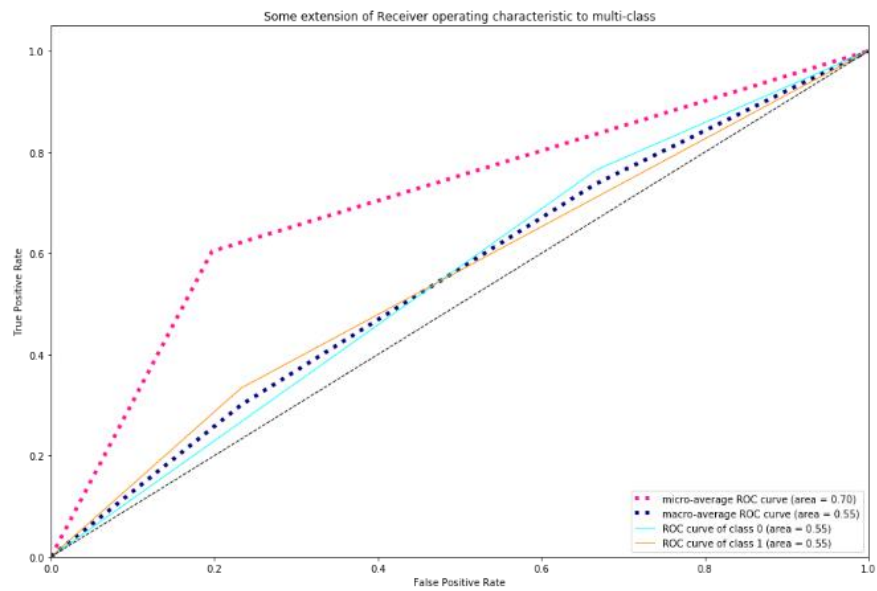
Among the new models, the one developed by the low-mobility dataset performs better than the same model in Chapter 5. The accuracy of this model increases by 2.34%. The ROC analysis shown in Figure 6.8 shows that the maximum AUC is achieved in the low-mobility environment, with a value of 0.91.



a. Low mobility



b. Medium mobility



c. High mobility

Figure 6. 8: a, b and c: ROC analysis for the models created by voting classifier

6.2.9 Neural Networks

We built our new models using all the built-in Afs implemented in the scikit-learn library, including 1) sigmoid function, 2) hyperbolic tangent function (tanh), 3) rectified linear unit (ReLU), and 4) identity function, and compared the cross-validated performance. Tables 6.13, 6.14 and 6.15 summarise the results.

As can be seen, the rows in these tables show the architecture (perceptron or MLP) and the type of AF for which the best-fit models were achieved in the experiments. Moreover, the first column of each of these tables shows the configuration of the neural networks in terms of the number of neurons in each layer. Our models were trained using a perceptron (no hidden layer), one hidden layer (with 10, 20, 30, 40 neurons), two hidden layers (with 10, 20, 30, 40 neurons) and three hidden layers (with 10, 20, 30, 40 neurons). We also built and tested our prediction models using 10-fold cross-validation for all prediction models. According to these considerations, the experiments were conducted 480 times per dataset. The calculation is as follows:

$$12 \text{ (configuration)} * 4 \text{ (AF)} * 10 \text{ (10-fold cross-validation)} = 480$$

Furthermore, the number of epochs is set to 500. This means that the model in each configuration was trained by 500 cycles using the whole training dataset to find the optimal weights and achieve better performance.

According to Table 6.13, the best performance is achieved by the MLP (consisting of one hidden layer and 20 neurons) with 86.92% accuracy. Thus, adding more hidden layers does not result in higher accuracy of the models. Overall, all new models have higher accuracy than the models developed in Chapter 5.

Table 6.13: Performance of the new neural network models trained by the low-mobility dataset

	Configuration	Accuracy Rate	Precision		Recall		F1	
Single Layer No Hidden Layer (AF='identity')	4-2	85.40% (1.25%)	0	1.00	0	0.21	0	0.35
			1	0.87	1	1.00	1	0.93
One Hidden Layer (AF='ReLU')	4-10-2	86.60% (1.02%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-20-2	86.92% (1.03%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-30-2	86.92% (1.03%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-40-2	86.90% (1.04%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
Two Hidden Layers (AF='tanh')	4-6-4-2	86.92% (1.03%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-12-8-2	86.92% (1.03%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-22-8-2	86.88% (1.04%)	0	0.96	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.93
	4-30-10-2	86.86% (1.04%)	0	0.96	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.93
Three Hidden Layers (AF='tanh')	4-5-3-2-2	86.90% (1.03%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-10-6-4-2	86.92% (1.03%)	0	1.00	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.94
	4-15-10-5-2	86.82% (1.00%)	0	0.96	0	0.28	0	0.43
			1	0.88	1	1.00	1	0.93

0: Deny, 1: Access

Table 6.14 shows the performance results for the neural network models trained and developed using the new medium-mobility dataset. The most accurate model is achieved in the MLP architecture (consisting of two hidden layers with 10 neurons), with 64.49% accuracy. The highest value of accuracy increased by 6.42% in comparison with the accuracy of the models developed

in Chapter 5. Moreover, the lowest value of accuracy in the table increased by 9.25% in comparison with the values in Table 5.11.

Table 6.14: Performance of the neural network models trained by the new medium-mobility dataset

	Configuration	Accuracy Rate	Precision		Recall		F1	
Single Layer No Hidden Layer (AF='identity')	4-2	64.43% (2.63%)	0	0.67	0	1.00	0	0.81
			1	0.32	1	0.05	1	0.09
One Hidden Layer (AF='reLU')	4-10-2	64.45% (2.62%)	0	0.67	0	1.00	0	0.81
			1	0.00	1	0.00	1	0.00
	4-20-2	64.09% (2.54%)	0	0.67	0	0.99	0	0.80
			1	0.34	1	0.09	1	0.04
	4-30-2	63.41% (2.27%)	0	0.67	0	0.96	0	0.79
			1	0.29	1	0.03	1	0.06
	4-40-2	63.25% (2.18%)	0	0.66	0	0.90	0	0.76
			1	0.23	1	0.06	1	0.10
Two Hidden Layers (AF='tanh')	4-6-4-2	64.49% (2.64%)	0	0.67	0	0.99	0	0.80
			1	0.34	1	0.07	1	0.11
	4-12-8-2	64.33% (2.67%)	0	0.67	0	0.97	0	0.80
			1	0.25	1	0.02	1	0.03
	4-22-8-2	63.65% (2.01%)	0	0.67	0	0.99	0	0.80
			1	0.23	1	0.07	1	0.10
	4-30-10-2	62.85% (2.22%)	0	0.68	0	0.88	0	0.77
			1	0.37	1	0.15	1	0.21
Three Hidden Layers (AF='tanh')	4-5-3-2-2	64.37% (2.58%)	0	0.67	0	0.99	0	0.80
			1	0.25	1	0.01	1	0.01
	4-10-6-4-2	64.15% (2.62%)	0	0.67	0	1.00	0	0.81
			1	0.24	1	0.03	1	0.02
	4-15-10-5-2	63.89% (2.45%)	0	0.66	0	0.88	0	0.75
			1	0.19	1	0.06	1	0.09

0: Deny, 1: Access

In new models developed by the high-mobility dataset, the highest accuracy was achieved through the perceptron architecture, with accuracy of 61.67%, which shows an increase of 5.9% in comparison with the values in Table 5.12. Table 6.15 summarises the performance of these models.

Table 6.15: Performance of the neural network models trained by the new high-mobility dataset

	Configuration	Accuracy Rate	Precision		Recall		F1	
Single Layer No Hidden Layer (AF='identity')	4-2	61.67% (2.27%)	0	0.63	0	1.00	0	0.77
			1	0.34	1	0.05	1	0.06
One Hidden Layer (AF='ReLU')	4-10-2	61.09% (1.58%)	0	0.63	0	0.96	0	0.76
			1	0.33	1	0.04	1	0.07
	4-20-2	60.81% (2.06%)	0	0.63	0	0.96	0	0.75
			1	0.39	1	0.07	1	0.12
	4-30-2	60.69% (1.78%)	0	0.64	0	0.90	0	0.75
			1	0.47	1	0.15	1	0.23
	4-40-2	60.75% (2.37%)	0	0.64	0	0.89	0	0.75
			1	0.48	1	0.17	1	0.25
Two Hidden Layers (AF='tanh')	4-6-4-2	61.57% (2.02%)	0	0.62	0	0.94	0	0.75
			1	0.25	1	0.03	1	0.06
	4-12-8-2	61.09% (2.24%)	0	0.63	0	0.94	0	0.75
			1	0.38	1	0.06	1	0.10
	4-22-8-2	60.21% (2.72%)	0	0.63	0	0.93	0	0.75
			1	0.35	1	0.06	1	0.11
	4-30-10-2	59.63% (2.13%)	0	0.64	0	0.85	0	0.73
			1	0.45	1	0.21	1	0.29
Three Hidden Layers (AF='tanh')	4-5-3-2-2	61.37% (2.16%)	0	0.63	0	1.00	0	0.77
			1	1.00	1	0.02	1	0.03
	4-10-6-4-2	60.75% (2.17%)	0	0.63	0	0.92	0	0.75
			1	0.38	1	0.08	1	0.13
	4-15-10-5-2	59.45% (2.38%)	0	0.64	0	0.89	0	0.74
			1	0.44	1	0.15	1	0.22

0: Deny, 1: Access

6.3 Discussion

As discussed in Subsection 6.2, 10 classification algorithms were applied to the new datasets. The datasets with which the prediction models were trained and built included a “trust” attribute in addition to “time”, “location” and “credentials”. According to the results, the performance of all the new prediction models trained by the low-mobility dataset increases. Moreover, the highest value of accuracy achieved in the low-mobility environment is 86.92%, which is higher than the highest accuracy obtained by past models by 1.42%. The highest accuracy of prediction models developed by the medium-mobility dataset is the same as that of the models developed in Chapter 5 (64.43%). For the high-mobility environment, the best performance was achieved at 61.71% accuracy, which is higher than the best results obtained from the past models.

Table 6.16 shows the aggregated performance results. It summarizes the performance of the prediction models trained and built by the new datasets labelled as high-mobility (H), medium-mobility (M) and low-mobility (L) datasets. The results are given for both classes – 0: Deny and 1: Access.

Table 6.16: Aggregated performance of the prediction models

Models	Accuracy Rate (Cross-Validated)			Precision				Recall				F1			
	L	M	H		L	M	H		L	M	H		L	M	H
Decision Tree	78.92% (1.69%)	56.29% (2.62%)	55.27% (2.52%)	0	0.36	0.70	0.67	0	0.34	0.70	0.67	0	0.35	0.70	0.67
				1	0.88	0.38	0.44	1	0.89	0.38	0.44	1	0.88	0.38	0.44
SVM	82.20% (1.07%)	64.43% (2.63%)	61.71% (2.20%)	0	0.00	0.67	0.63	0	0.00	1.00	1.00	0	0.00	0.81	0.77
				1	0.84	0.00	0.00	1	1.00	0.00	0.00	1	0.91	0.00	0.00
Logistic Regression	86.90% (0.99%)	64.43% (2.63%)	61.71% (2.20%)	0	1.00	0.67	0.63	0	0.28	1.00	1.00	0	0.43	0.81	0.77
				1	0.88	0.00	0.00	1	1.00	0.00	0.00	1	0.94	0.00	0.00
Naïve Bayes	79.40% (2.03%)	59.13% (3.36%)	58.09% (1.54%)	0	0.38	0.67	0.66	0	0.33	0.79	0.70	0	0.35	0.73	0.68
				1	0.87	0.31	0.43	1	0.90	0.19	0.38	1	0.89	0.23	0.40
AdaBoost	86.76% (1.11%)	63.73% (2.23%)	60.59% (2.24%)	0	0.96	0.67	0.64	0	0.29	0.97	0.93	0	0.44	0.79	0.76
				1	0.88	0.17	0.46	1	1.00	0.01	0.10	1	0.94	0.02	0.17
Random Forest	82.58% (1.17%)	64.43% (2.63%)	61.67% (2.23%)	0	1.00	0.67	0.63	0	0.06	1.00	1.00	0	0.12	0.81	0.77
				1	0.85	0.00	0.00	1	1.00	0.00	0.00	1	0.92	0.00	0.00
K-NN				0	0.32	0.66	0.63	0	0.15	0.67	0.65	0	0.21	0.66	0.64

	79.12% (2.10%)	57.29% (2.02%)	55.95% (1.90%)	1	0.85	0.28	0.37	1	0.94	0.27	0.34	1	0.89	0.28	0.36
ANN	86.92% (1.03%)	64.49% (2.64%)	61.67% (2.21%)	0	1.00	0.67	0.63	0	0.28	0.99	1.00	0	0.43	0.80	0.77
				1	0.88	0.34	0.00	1	1.00	0.07	0.00	1	0.94	0.11	0.00
Gradient Boost	86.90% (1.06%)	64.39% (2.61%)	61.59% (2.28%)	0	1.00	0.67	0.63	0	0.28	1.00	1.00	0	0.43	0.81	0.77
				1	0.88	0.00	0.00	1	1.00	0.00	0.00	1	0.94	0.00	0.00
Voting Classifier	86.90% (1.04%)	59.83% (2.76%)	57.57% (2.25%)	0	1.00	0.67	0.66	0	0.28	0.82	0.76	0	0.43	0.74	0.71
				1	0.88	0.33	0.46	1	1.00	0.18	0.33	1	0.94	0.23	0.39

0: Deny, 1: Access

As shown in the above table, the models developed by ANN and logistic regression showed the highest performance for all environments. In other words, adding a new attribute (trust) to the datasets resulted in an increase in the performance of these two algorithms dramatically. Furthermore, models developed by SVM showed the same performance in the medium- and high-mobility environments as the models developed by ANN and logistic regression.

Figures 6.9, 6.10 and 6.11 show the aggregated ROC curve analysis for the “Access” label. As shown in Figure 6.9, for models with less than 0.5% difference in accuracy, the value of AUC is the same (0.64). Moreover, logistic regression achieved better performance in terms of a higher TP rate and a lower FP rate than ANN in the low-mobility environment.

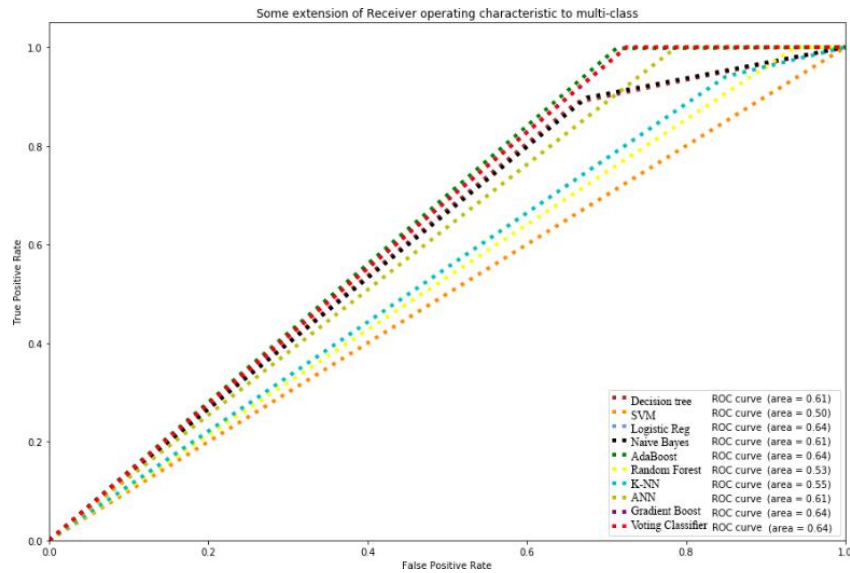


Figure 6.9: ROC analysis for prediction models in the low-mobility environment

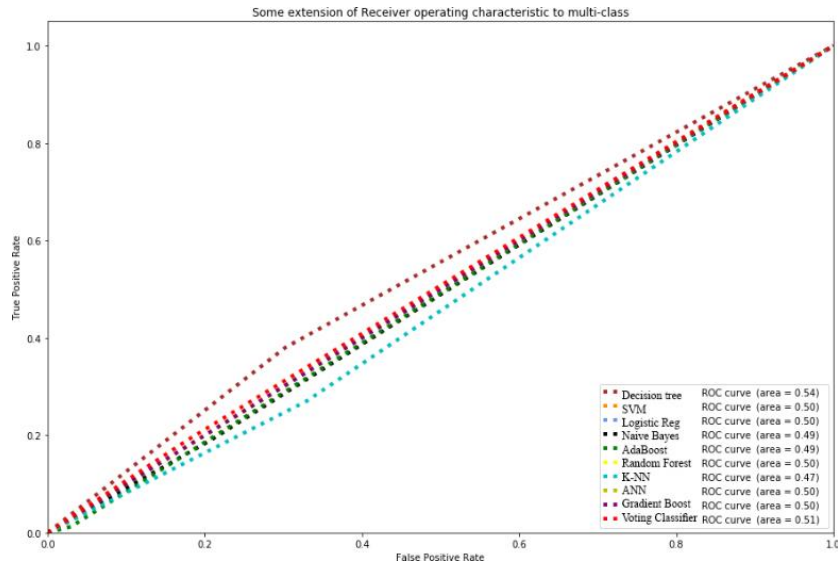


Figure 6.10: ROC analysis for prediction models in the medium-mobility environment

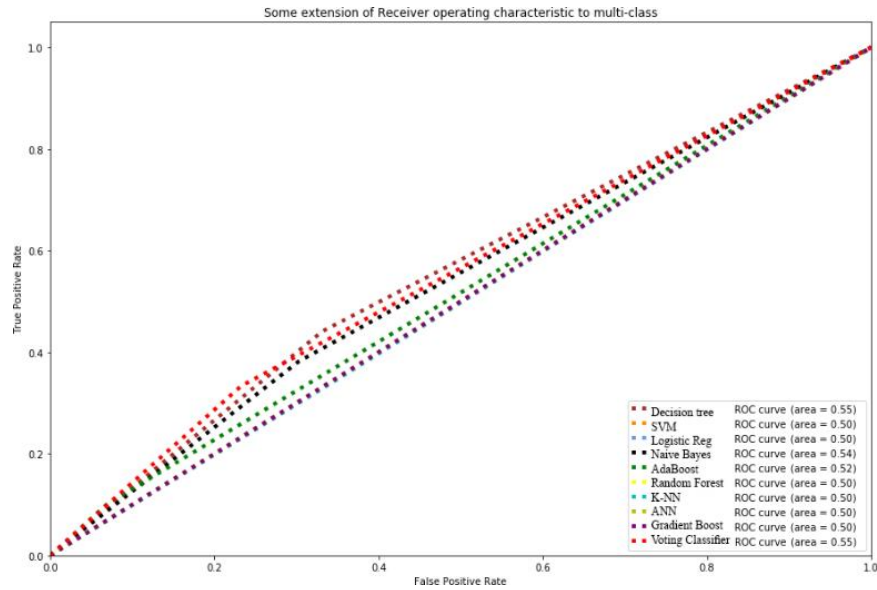


Figure 6. 11: ROC analysis for prediction models in the high-mobility environment

Figure 6.10 shows the aggregated ROC curves for the prediction models in the medium-mobility environment. Among the models in Table 6.16, ANN confirmed its superiority over the rest by its AUC value (0.50). Models with fairly similar performance, such as SVM and logistic regression, have the same AUC values as one another.

In the high-mobility environment, as shown in Figure 6.11, voting classifier (0.55), decision tree (0.55), Naïve Bayes (0.54) and AdaBoost (0.52) achieved the highest AUC values. The accuracy, precision, recall and F1 values of these three algorithms were much lower than those of the best model (i.e., logistic regression). Thus, their superiority cannot be confirmed: as stated in Chapter 5, a curve dominates in ROC space if and only if its precision dominates in precision space [183].

We also applied Kruskal–Wallis and Spearman tests to investigate the effects of the number of hidden layers on the accuracy of the models for all datasets. According to the results shown in Tables 6.17 and 6.18, the Asymp. Sig. (p-value) is less than 0.05 for the Kruskal–Wallis test, and therefore the null assumption is rejected. This shows that the mean values of different accuracy groups are not the same. Groups are defined based on the number of hidden layers. Furthermore, the results of the Spearman test indicate that the number of hidden layers has an impact on the accuracy of the model (Sig.=0.000 and Sig. < 0.05) in each environment.

Table 6.17: Kruskal–Wallis test results, grouping variable: no. of hidden layers

Chi Square	139.405
Df	3
Asymp. Sig.	0.000

Table 6.18: Spearman test results: correlation

			Hidden Layer	Accuracy
Spearman's rho	Hidden Layer	Correlation Coefficient	1.000	0.54**
		Sig. (2-tailed)	.	0.000
		N	7800	7800
	Accuracy	Correlation Coefficient	0.54**	1.000
		Sig. (2-tailed)	0.000	.
		N	7800	7800

*** Correlation is significant at the 0.01 level (2-tailed)*

6.4 Chapter Summary

In this chapter, we presented details of our prediction models developed for handling ambiguity. We synthesized a set of new datasets by adding a new attribute called “Trust”. These new datasets were synthesized using methodology discussed in Chapter 4 and 5 for three degrees of mobility (high, medium and low).

As shown, ten classification algorithms were applied to create our prediction models. According to the cross-validated results, model developed by neural networks showed the highest performance in terms of accuracy for low mobility environment with accuracy of 86.92%. Neural networks also showed the highest performance in medium mobility environment with accuracy of 86.90%. In high mobility environment, SVM and logistic regression algorithms showed the highest accuracy (61.71%).

We also studied the behavior of our models created by neural networks (both perceptron and MLP). We applied 12 configurations of neural networks including different number of hidden layers (up to 3 hidden layers) different number of neurons (up to 40 neurons) and different types of activation functions (4 AFs) to investigate the effects of these variants on the performance of the prediction models. The results showed that the number of hidden layers affects on the performance of the model.

In overall, by adding more attributes to the datasets, the accuracy of the models will increase and classifiers like ANN and logistic regression may show better performance in action.

7. Conclusion

In this chapter, we summarize the important findings of this research. We also review and analyse the thesis to determine whether the findings can answer the research questions proposed in Chapter 1. At the end of this chapter, we discuss the possible future direction of this research.

7.1 Key Findings

Starting with the literature review, we summarize the key findings in relation to our research questions. These findings came together to construct our methodology.

7.1.1 Findings on IoT Adaptability of Access Control Models

We evaluated traditional and emerging access control models against IoT adaptability criteria, as mentioned in Chapter 2. The reference models could not satisfy all the specification criteria that need to be deployed in a scalable, heterogeneous and dynamic environment such as IoT. More details were summarized in Table 2.1. According to our analysis in Subsection 2.1.2, ABAC shows promising performance in terms of scalability (extensibility), dynamism, heterogeneity/interoperability, and context-awareness in comparison with other models.

Moreover, both the traditional and the emerging models relied on deterministic access policy rules for which they were unable to make precise access decisions in non-deterministic access scenarios.

We also surveyed the state of the art for the methods proposed based on the extension of the traditional and emerging access control models in Subsection 2.1.4. As these methods inherit the disadvantages of the reference models, they suffer from a lack of one or more of the characteristics needed for the IoT environment.

Authentication protocols were analysed against the criteria defined in RFC 2989 and RFC 4962 too. According to the results mentioned in Subsection 2.1.2, four out of five authentication protocols suffer from a single point of failure in their implementation because of their centralized architecture. Maintaining the confidentiality of authentication data-at-rest and authentication data-

in-transit is another major challenge. According to our findings, these protocols were designed based on deterministic rules and cannot handle access scenarios that include unpredicted elements.

Resilient access control approaches were thoroughly studied in Subsection 2.2.4. According to our findings, the BTG and optimistic approaches are inappropriate for heterogeneous environments such as IoT. They also suffer from a lack of scalability in terms of access policy for such environments.

RAAC approaches were considered a promising paradigm for handling unpredicted access scenarios. Based on the finding of this research in Subsection 2.2.4, the IoT adaptability of these approaches remains an ongoing challenge. These approaches generally suffer from one or more of the following challenges: i) limitations on periodic assessment for the IoT environment, ii) a lack of knowledge about IoT entities, and iii) interoperability and dependency challenges.

7.1.2 Proposing Indeterminacy Factors for Authentication in IoT

We defined uncertainty and ambiguity as two pillars of indeterminacy in authentication that presents new challenges. We stated the importance of these challenges and their relationship with characteristics inherited from the IoT environment.

As defined in Subsection 2.2.1.1, uncertainty stems from the incompleteness of information regarding the likelihood of whether the acceptance of an authentication request leads to an incident. A formal definition was given based on subjective probability. Five uncertainty handling theories including probability theory, information theory, evidence theory, possibility theory and uncertainty theory existing in the literature were studied and their suitability for different types of uncertainty was analysed in Subsection 2.2.1. Of these theories, subjective (conditional) probability was chosen to measure uncertainty in authentication for a couple of reasons: lower complexity, scalability and enough data samples in authentication scenarios.

Ambiguity in authentication was also defined by this research and the relevant literature was studied in Subsection 2.2.2.1. Ambiguity can be handled through trust-based analysis. Based on the findings of Subsection 2.2.2.1, soft trust method is the choice for handling trust in IoT. As a result, behavioural-based analysis as a method of direct trust computation was chosen to calculate

the degree of trust. In this way, the historical profile of users was analysed to gain insight into previous authentication records for all users.

7.1.3 DataSet Synthesis for Authentication

In order to build prediction models capable of making indeterminacy-aware authentication decisions, machine-learning algorithms must be applied. These algorithms are classified as supervised algorithms as they require labelled datasets to be trained and tested. One of the research obstacles in building prediction models for authentication has been the lack of publicly available authentication datasets.

We presented the exemplar (RASA) as our case study in this dissertation in Subsection 4.1. We also determined adversary model for the exemplar.

In order to build our prediction models, we need datasets. Datasets need to be synthesized in accordance with the exemplar. In doing so, the relevant literature was studied in Subsection 4.3 to determine the corresponding PDFs of attributes needed in our datasets, including user, time, location and credentials. According to the state of the art discussed in Subsection 4.3.1, users' online activity follows a power law distribution function. The timing of the authentication requests followed uniform distribution based on the literature reviewed in Subsection 4.3.2, and the PDF corresponding to the location of the users as discussed in Subsection 4.3.3, was determined to be Gaussian. For the credentials attribute, three states were justified in Subsection 4.3.4 in order to define a multinomial PDF that describes the behaviour of the authentication requests in terms of credentials.

The findings of the research in the above-mentioned datasets were compared with the results of the study of publicly available datasets (LANL) in Subsection 4.4. The analysis confirmed the effectiveness of the methodology employed in synthesizing data samples for the user and time attributes.

One of the advantages of this research is that it considered the “mobility” of the users by defining different UAs and considering a number of PoIs. Subsequently, a mixture of Gaussian PDFs were used to describe the mobility of users, and corresponding parameters were determined in a way that reflected three degrees of mobility, known as low, medium and high. Consequently,

data samples were generated for three datasets with different mobility patterns. The process of generating different datasets in terms of mobility was discussed in Subsection 4.3.3.

7.1.4 Handling Uncertainty in Authentication Using Prediction Models

As discussed in Chapter 5, ten classifiers were applied to datasets in order to build prediction models: decision tree, random forest, Naïve Bayes, logistic regression, SVM, neural networks, voting classifier, gradient boost and AdaBoost classifiers, and K-NN.

The datasets used in Chapter 5 consisted of three attributes (time, location and credentials). According to the findings discussed in Subsection 5.3, the uncertainty-aware models trained and built using these datasets were able to predict the class of authentication requests in the low-, medium- and high-mobility datasets with 85.50% accuracy (using AdaBoost), 64.43% accuracy (using gradient boost) and 61.69% accuracy (using gradient boost) respectively.

We also studied the behavior of our models created by neural networks (both perceptron and MLP) in Subsection 5.2.9. We applied 12 configurations of neural networks including different number of hidden layers (up to 3 hidden layers) different number of neurons (up to 40 neurons) and different types of activation functions (4 AFs) to investigate the effects of these variants on the performance of the prediction models. The results showed that the number of hidden layers affects on the performance of the model.

7.1.5 Handling Trust in Authentication Using Prediction Models

In Chapter 6, the datasets had an extra attribute called a historical profile. Adding this new attribute improved the performance of the models for the three datasets. As discussed in Subsection 6.3, the ambiguity-aware prediction models developed by the above-mentioned classifiers were able to predict the class of the authentication requests in low-, medium- and high-mobility datasets with 86.92% accuracy (using Artificial Neural Networks (ANN), 64.49% accuracy (using ANN) and 61.71% accuracy (using logistic regression) respectively.

The models created by the datasets used in Chapter 6 were able to handle both uncertainty and ambiguity in authentication. The size of the designated models is small enough to be run by IoT-

friendly devices such as the Raspberry Pi. The prediction models were deployed on a Raspberry Pi 4 (Model B) and tested by another set of synthesized datasets to see the performance in action (Appendix B).

7.2 Evaluation

As defined in the first chapter, the main claim of this research was that an “indeterminacy-aware prediction model can handle indeterminacy factors in authentication for scalable, heterogeneous and dynamic environments”.

Figure 7.1 is a diagrammatic model of the thesis’s main claim, and the building blocks on which this is based. The main claim can be broken down, based on the indeterminacy factors, into two parts. Each part, which is shown in an orange box, was explored separately. These factors were studied using the datasets developed in Chapter 4. The research direction and the architecture of the methodology were defined in Chapter 3. Using the literature review to determine the scope of the analysis, we explored the literature to demonstrate the research gap and possible methodological approaches.

Indeterminacy-aware prediction models can handle uncertainty and ambiguity in scalable, heterogeneous and dynamic environments because of the following characteristics:

- The prediction models are fully automatic and work without human intervention.
- The prediction models can work in scalable environments because increasing the number of authentication requests does not affect the performance of the prediction models in terms of complexity.
- The type of the selected attributes in the dataset makes the approach independent of local and environmental characteristics. Therefore, it can be deployed in heterogeneous environments.
- Considering time and the location in making authentication decisions makes the proposed approach spatio-temporal. Therefore, the model is sensitive to a changing environment which is known as a dynamic environment.

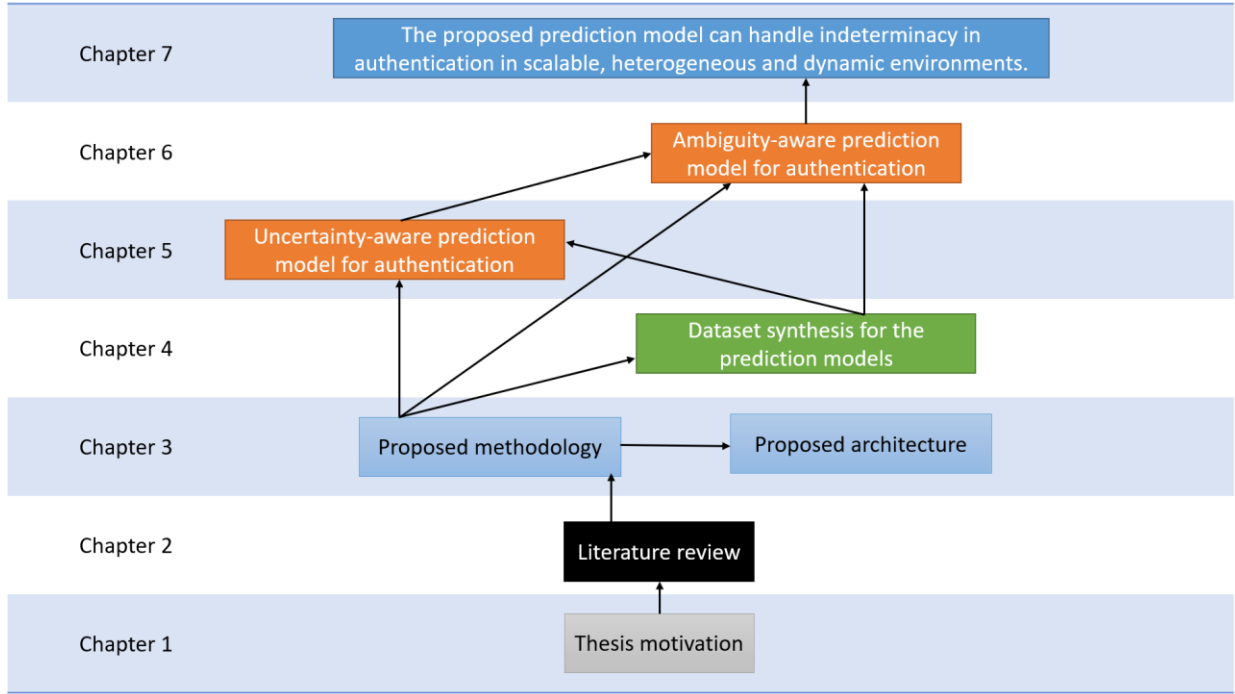


Figure 7. 1: The thesis’s main claim and its building blocks

7.3 Future Direction

In this Subsection, we propose two possible areas for future research that build upon the contributions of this dissertation.

7.3.1 Handling Indeterminacy in Authorization

As stated in the first chapter, the main focus of this research was on indeterminacy factors in the “authentication” phase of access control. Authorization as another phase of access control deals with the same challenges. We need to define uncertainty and ambiguity in authorization and find related attributes in order to build prediction models.

7.3.2 Moving Towards Adaptive Model

The second consideration of the proposed approach is that it does not support adaptive methods of calculating indeterminacy for mobile users. In other words, when the location of the user is changed, the authenticated user has access to the resource, so the proposed method supports only

persistent authentication in IoT. Any future work must consider a scheme to cover adaptive authentication for mobile users.

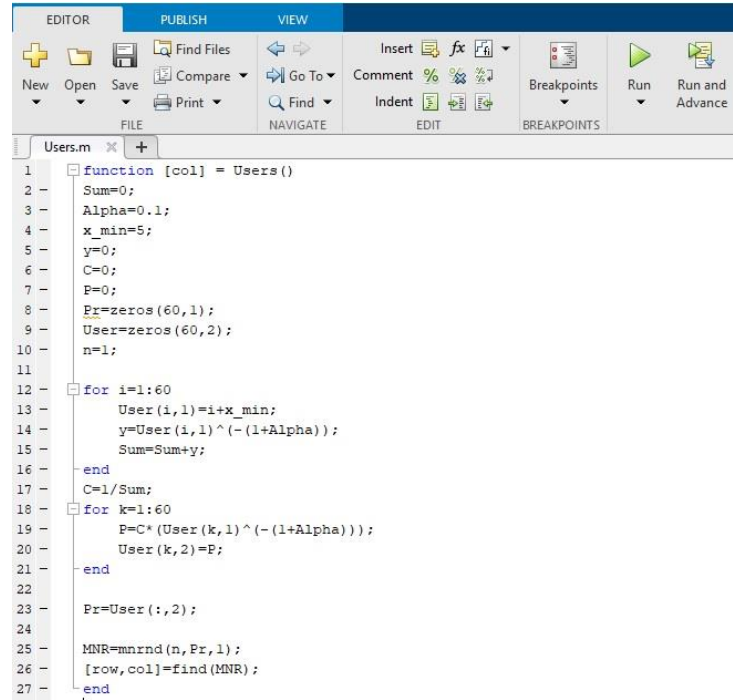
Although the generation of data samples to create the datasets used by this research is one of its contributions, working with real datasets consisting of the required attributes would give better insight into the usefulness and the performance of the proposed approach.

Appendix A: Dataset Synthesis in MATLAB

This appendix contains practical details on synthesizing datasets. We used MATLAB version 2018a to generate our data samples. The case study used in order to synthesize our datasets was explained in Subsection 4.1. We also followed the procedure discussed in Subsection 4.3 to synthesize values for our datasets. We defined a zero-matrix called UMax with the size of (5000×8) to store generated data samples using the following command: `UMax=zeros(5000,8)`. We upload the code of this section on Zenodo¹³.

A.1 Generating Data samples for IDs

As discussed in Subsection 4.3.1, we have 60 users in the system. According to the literature power law PDF needs to be used to generate data samples for IDs. Figure A.1 shows the code written in MATLAB for synthesizing these samples.

The image shows a screenshot of the MATLAB IDE. The top menu bar includes 'EDITOR', 'PUBLISH', and 'VIEW'. Below it is a toolbar with icons for 'New', 'Open', 'Save', 'Find Files', 'Compare', 'Go To', 'Find', 'Insert', 'Comment', 'Indent', 'Breakpoints', 'Run', and 'Run and Advance'. The main window displays a script named 'Users.m' with the following code:

```
1 function [col] = Users()
2     Sum=0;
3     Alpha=0.1;
4     x_min=5;
5     y=0;
6     C=0;
7     P=0;
8     Pr=zeros(60,1);
9     User=zeros(60,2);
10    n=1;
11
12    for i=1:60
13        User(i,1)=i+x_min;
14        y=User(i,1)^(-(1+Alpha));
15        Sum=Sum+y;
16    end
17    C=1/Sum;
18    for k=1:60
19        P=C*(User(k,1)^(-(1+Alpha)));
20        User(k,2)=P;
21    end
22
23    Pr=User(:,2);
24
25    MNR=mnrnd(n,Pr,1);
26    [row,col]=find(MNR);
27    end
```

Figure A. 1: MATLAB code for synthesizing data samples for IDs

¹³ Available at: <https://zenodo.org/record/3755539>

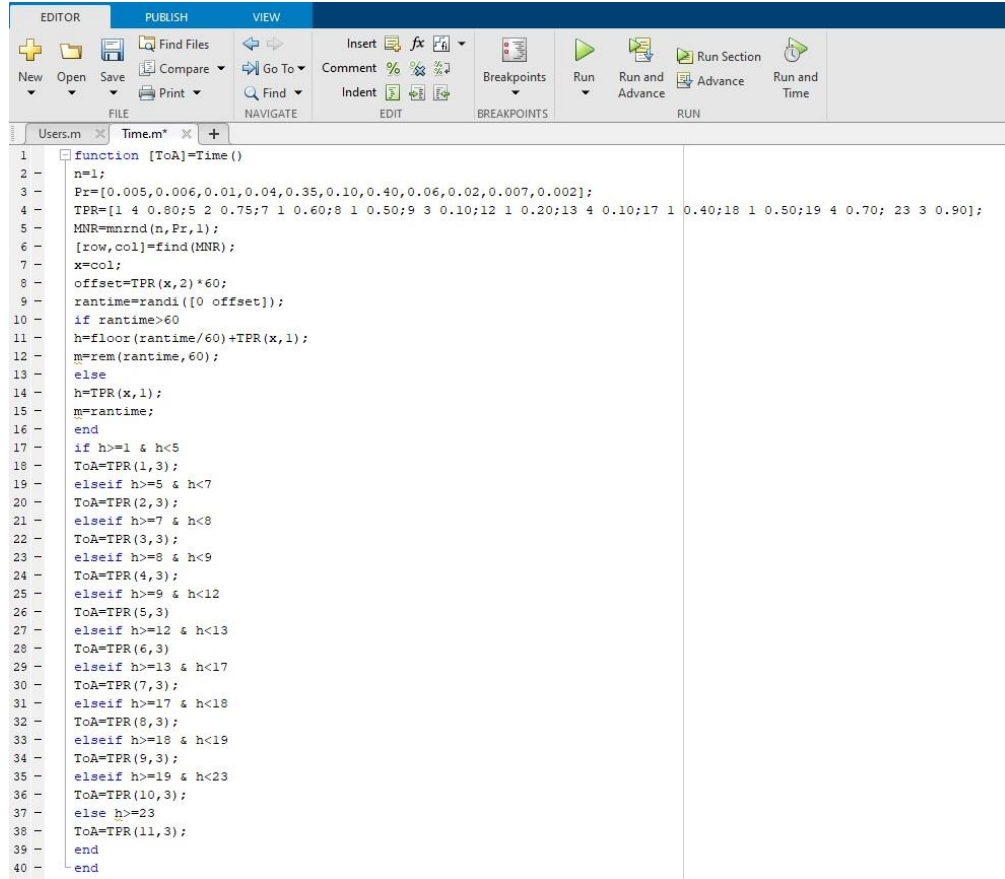
The formula used as power law distribution is as follows:

$$\Pr(x) = C(x)^{-(1+\alpha)}$$

As shown in Figure A.1, we implemented power law distribution formula in MATLAB through lines 12 to 21. The “User” function will return one ID number from 1 to 60 at each round of a function call.

A.2 Generating Data samples for Time

As discussed in Subsection 4.3.2, we divided service time into 11 time slots (Table 4.2). Moreover, we assigned weights to these time slots according to our threat model discussed in Subsection 4.2. In order to generate time related data samples, we first used multinomial PDF to choose the time slot and then generate a random value for the time in that time slot using uniform PDF.



```

1 function [ToA]=Time()
2 n=1;
3 Pr=[0.005,0.006,0.01,0.04,0.35,0.10,0.40,0.06,0.02,0.007,0.002];
4 TPR=[1 4 0.80;5 2 0.75;7 1 0.60;8 1 0.50;9 3 0.10;12 1 0.20;13 4 0.10;17 1 0.40;18 1 0.50;19 4 0.70; 23 3 0.90];
5 MNR=mnrnd(n,Pr,1);
6 [row,col]=find(MNR);
7 x=col;
8 offset=TPR(x,2)*60;
9 rertime=randi([0 offset]);
10 if rertime>60
11 h=floor(rertime/60)+TPR(x,1);
12 m=rem(rertime,60);
13 else
14 h=TPR(x,1);
15 m=rertime;
16 end
17 if h>=1 & h<5
18 ToA=TPR(1,3);
19 elseif h>=5 & h<7
20 ToA=TPR(2,3);
21 elseif h>=7 & h<8
22 ToA=TPR(3,3);
23 elseif h>=8 & h<9
24 ToA=TPR(4,3);
25 elseif h>=9 & h<12
26 ToA=TPR(5,3);
27 elseif h>=12 & h<13
28 ToA=TPR(6,3);
29 elseif h>=13 & h<17
30 ToA=TPR(7,3);
31 elseif h>=17 & h<18
32 ToA=TPR(8,3);
33 elseif h>=18 & h<19
34 ToA=TPR(9,3);
35 elseif h>=19 & h<23
36 ToA=TPR(10,3);
37 else h>=23
38 ToA=TPR(11,3);
39 end
40 end

```

Figure A. 2: MATLAB code for generating data samples for the time attribute

Line 3 shows a matrix (11×1) which consists of weights assigned to time slots based on table 4.2. Line 4 shows a matrix (11×3) which consists of time slots and the Uvs assigned to them based on table 4.2. Lines 5-7 use multinomial PDF (mnrnd) to randomly choose a time slot. Line 9 uses uniform distribution (randi) to randomly pick a time in designated time slot. Lines 10 to 16 split the chosen time to Hour / Minute. Based on the values (h and m) of generated time, the corresponding UV will be assigned through line 17 to 38. These values were determined in table 4.2.

A.3 Generating Data samples for Location

According to the findings of Subsection 4.3.3, we used a mixture of Gaussian PDF to generate data samples for the location. Figure A.3 shows the calculated UVs for the locations in the map of our exemplar discussed in Subsection 4.1. These values were calculated based on the mixture method discussed at the end of Subsection 4.3.3. Please note that at the time of writing this dissertation, the UVs assigned to different UAs were different than values indicated in Figure 4.5.

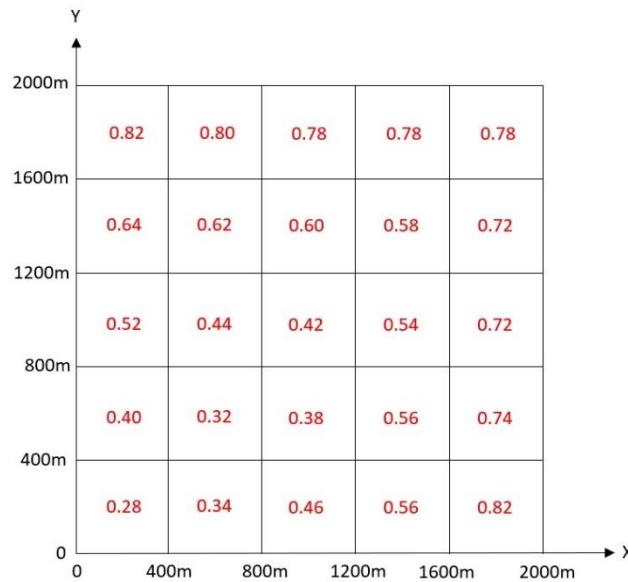


Figure A. 3: Calculated UVs for the map based on the mixture gaussian method

Figure A.4 shows the MATLAB code for synthesizing location data samples in medium mobility dataset.

```

1 function [Uncertainty] = Location()
2     Uncertainty=0;
3     Mu_x_1=200;
4     Mu_x_2=1000;
5     Mu_x_3=1400;
6     Mu_y_1=200;
7     Mu_y_2=600;
8     Mu_y_3=1400;
9     Sig_x_1=200;
10    Sig_x_2=500;
11    Sig_x_3=700;
12    Sig_y_1=150;
13    Sig_y_2=400;
14    Sig_y_3=600;
15    n=1;
16    Pr=[0.65,0.2,0.15];
17    MNR=mnrnd(n,Pr,1);
18    [row,col]=find(MNR);
19    G=col;
20    if G==1
21        x=normrnd(Mu_x_1,Sig_x_1);
22        y=normrnd(Mu_y_1,Sig_y_1);
23    elseif G==2
24        x=normrnd(Mu_x_2,Sig_x_2);
25        y=normrnd(Mu_y_2,Sig_y_2);
26    elseif G==3
27        x=normrnd(Mu_x_3,Sig_x_3);
28        y=normrnd(Mu_y_3,Sig_y_3);
29    end
30
31    if x>0 & x<400 & y>0 & y<400
32        Uncertainty=0.28;
33    elseif x>400 & x<800 & y>0 & y<400
34        Uncertainty=0.34;
35    elseif x>800 & x<1200 & y>0 & y<400
36        Uncertainty=0.46;
37    elseif x>1200 & x<1600 & y>0 & y<400
38        Uncertainty=0.56;
39    elseif x>1600 & x<2000 & y>0 & y<400
40        Uncertainty=0.82;
41    elseif x>0 & x<400 & y>400 & y<800
42        Uncertainty=0.40;
43    elseif x>400 & x<800 & y>400 & y<800
44        Uncertainty=0.32;
45    elseif x>800 & x<1200 & y>400 & y<800
46        Uncertainty=0.38;
47    elseif x>1200 & x<1600 & y>400 & y<800
48
49    elseif x>1600 & x<2000 & y>400 & y<800
50        Uncertainty=0.74;
51    elseif x>0 & x<400 & y>800 & y<1200
52        Uncertainty=0.52;
53    elseif x>400 & x<800 & y>800 & y<1200
54        Uncertainty=0.44;
55    elseif x>800 & x<1200 & y>800 & y<1200
56        Uncertainty=0.42;
57    elseif x>1200 & x<1600 & y>800 & y<1200
58        Uncertainty=0.54;
59    elseif x>1600 & x<2000 & y>800 & y<1200
60        Uncertainty=0.72;
61    elseif x>0 & x<400 & y>1200 & y<1600
62        Uncertainty=0.64;
63    elseif x>400 & x<800 & y>1200 & y<1600
64        Uncertainty=0.62;
65    elseif x>800 & x<1200 & y>1200 & y<1600
66        Uncertainty=0.60;
67    elseif x>1200 & x<1600 & y>1200 & y<1600
68        Uncertainty=0.58;
69    elseif x>1600 & x<2000 & y>1200 & y<1600
70        Uncertainty=0.72;
71    elseif x>0 & x<400 & y>1600 & y<2000
72        Uncertainty=0.82;
73    elseif x>400 & x<800 & y>1600 & y<2000
74        Uncertainty=0.80;
75    elseif x>800 & x<1200 & y>1600 & y<2000
76        Uncertainty=0.78;
77    elseif x>1200 & x<1600 & y>1600 & y<2000
78        Uncertainty=0.78;
79    elseif x>1600 & x<2000 & y>1600 & y<2000
80        Uncertainty=0.78;
81    end
82    end
83

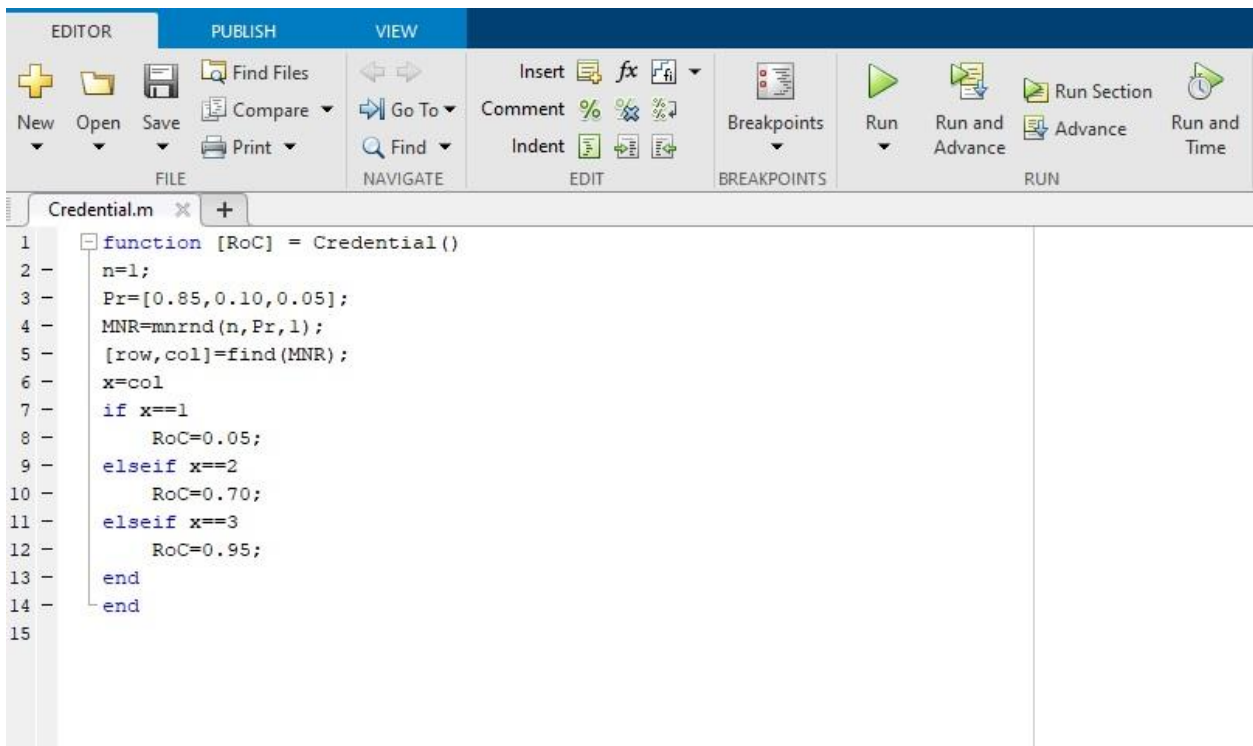
```

Figure A. 4: MATLAB code for generating data samples for the location attribute

In lines 3 to 14, parameters related to Gaussian PDF are initialized based on the values assigned to the exemplar in table 4.3 for medium mobility environment. In line 17, a multinomial PDF (mnrnd) is used to randomly choose a PoI. Then based on the chosen PoI, a Gaussian PDF (normrnd) is used to generate data samples in terms of X and Y through lines 20-29. Finally, through lines 31-81, according to the generated values for X and Y, a UV value will be assigned based on the calculate values in Figure A.3.

A.4 Generating Data samples for Credential

As discussed in Subsection 4.3.3, credentials have three states. We simply used a multinomial PDF (in line 4) to randomly choose one of those states. The weights assigned in line 3 are based on the values of probabilities listed in Table 4.4. According to generated samples for credential, corresponding UVs are assigned through lines 7 to 13. These values are listed and justified in Subsection 4.3.3.



The image shows a screenshot of the MATLAB Editor window. The title bar indicates the file is 'Credential.m'. The editor displays the following MATLAB code:

```
1 function [RoC] = Credential()  
2     n=1;  
3     Pr=[0.85,0.10,0.05];  
4     MNR=mnrnd(n,Pr,1);  
5     [row,col]=find(MNR);  
6     x=col  
7     if x==1  
8         RoC=0.05;  
9     elseif x==2  
10        RoC=0.70;  
11    elseif x==3  
12        RoC=0.95;  
13    end  
14    end  
15
```

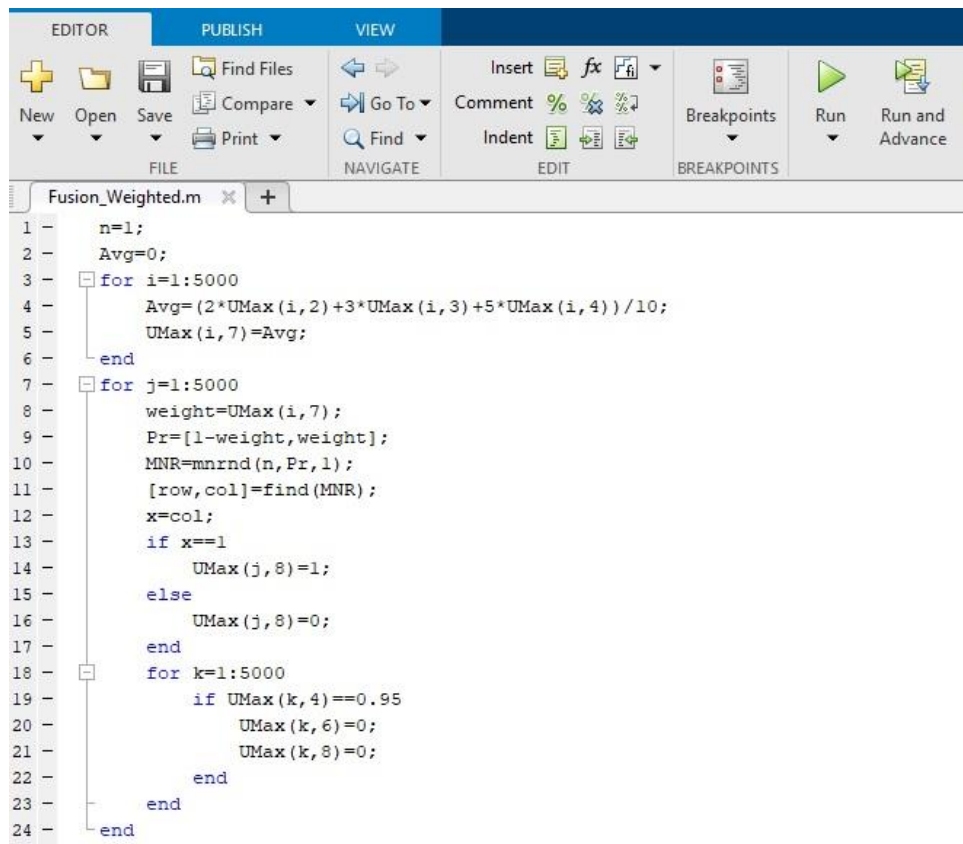
Figure: A. 5: MATLAB code for generating data samples for the credential attribute

A.5 Aggregating UVs to Label Datasets

Decision about an authentication request depends on the total value of uncertainty based on the UV values derived from Subsection A.1, A.2, A.3 and A.4. As discussed in Subsection 4.3.5, we considered weights for our attributes based on the adversary model. Therefore, we calculated the weighted arithmetic mean by averaging weighted UVs per authentication request. Then, we used

the calculated value as a weight for a binomial PDF in order to determine the class label (Access/Deny).

Figure A.6 shows the MATLAB code for labelling datasets. As shown in lines 3-6, the weighted arithmetic mean is calculated for each authentication request in a loop (for 5000 requests). Afterwards, binomial PDF is applied to determine the label of requests through lines 7-17. We also corrected our labels for those requests which provided wrong username and password, but they were labelled as “Access” in our dataset. This correction has been made through lines 18-22 in accordance with match-funder policy.



```

1 - n=1;
2 - Avg=0;
3 - for i=1:5000
4 -     Avg=(2*UMax(i,2)+3*UMax(i,3)+5*UMax(i,4))/10;
5 -     UMax(i,7)=Avg;
6 - end
7 - for j=1:5000
8 -     weight=UMax(i,7);
9 -     Pr=[1-weight,weight];
10 -    MNR=mnrnd(n,Pr,1);
11 -    [row,col]=find(MNR);
12 -    x=col;
13 -    if x==1
14 -        UMax(j,8)=1;
15 -    else
16 -        UMax(j,8)=0;
17 -    end
18 -    for k=1:5000
19 -        if UMax(k,4)==0.95
20 -            UMax(k,6)=0;
21 -            UMax(k,8)=0;
22 -        end
23 -    end
24 - end

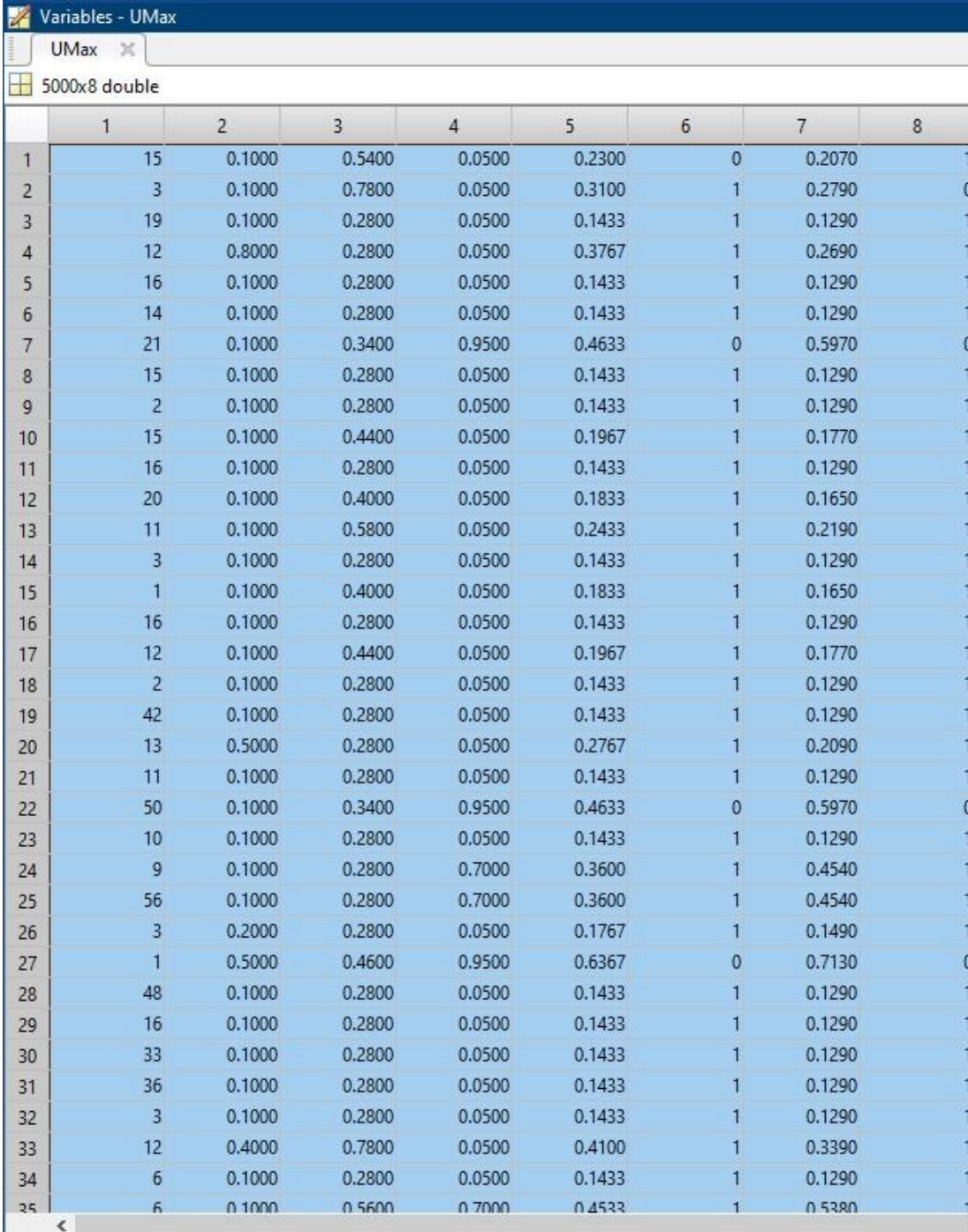
```

Figure A. 6: MATLAB code for labelling our dataset

Figure A.7 shows our final uncertainty matrix (UMax) that consists of 5000 generated authentication requests. Each request consists of 4 attributes and 4 calculated values. From left to right:

Synthesized values for attributes: Column 1: User ID, Column 2: Time UV, Column 3: Location UV, Column 4: Credential UV,

Calculated values: Column 5: Mean, Column 6: Label based on the mean, Column 7: Weighted Mean, and Column 8: Label based on the weighted mean.



	1	2	3	4	5	6	7	8
1	15	0.1000	0.5400	0.0500	0.2300	0	0.2070	1
2	3	0.1000	0.7800	0.0500	0.3100	1	0.2790	0
3	19	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
4	12	0.8000	0.2800	0.0500	0.3767	1	0.2690	1
5	16	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
6	14	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
7	21	0.1000	0.3400	0.9500	0.4633	0	0.5970	0
8	15	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
9	2	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
10	15	0.1000	0.4400	0.0500	0.1967	1	0.1770	1
11	16	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
12	20	0.1000	0.4000	0.0500	0.1833	1	0.1650	1
13	11	0.1000	0.5800	0.0500	0.2433	1	0.2190	1
14	3	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
15	1	0.1000	0.4000	0.0500	0.1833	1	0.1650	1
16	16	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
17	12	0.1000	0.4400	0.0500	0.1967	1	0.1770	1
18	2	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
19	42	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
20	13	0.5000	0.2800	0.0500	0.2767	1	0.2090	1
21	11	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
22	50	0.1000	0.3400	0.9500	0.4633	0	0.5970	0
23	10	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
24	9	0.1000	0.2800	0.7000	0.3600	1	0.4540	1
25	56	0.1000	0.2800	0.7000	0.3600	1	0.4540	1
26	3	0.2000	0.2800	0.0500	0.1767	1	0.1490	1
27	1	0.5000	0.4600	0.9500	0.6367	0	0.7130	0
28	48	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
29	16	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
30	33	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
31	36	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
32	3	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
33	12	0.4000	0.7800	0.0500	0.4100	1	0.3390	1
34	6	0.1000	0.2800	0.0500	0.1433	1	0.1290	1
35	6	0.1000	0.5600	0.7000	0.4533	1	0.5380	1

Figure A. 7: Screenshot of the results

Appendix B: Developing and Running Model on Raspberry Machine

This appendix presents an overview of technical works done on the building of prediction models. It gives a brief comment on different parts of the code and discusses the deployment of the models on Raspberry Pi machines.

We used Python version 3.6 with Jupyter¹⁴ to write our code and run it. We also used a number of libraries in Python depicted in Figure B.1. These libraries are as follows:

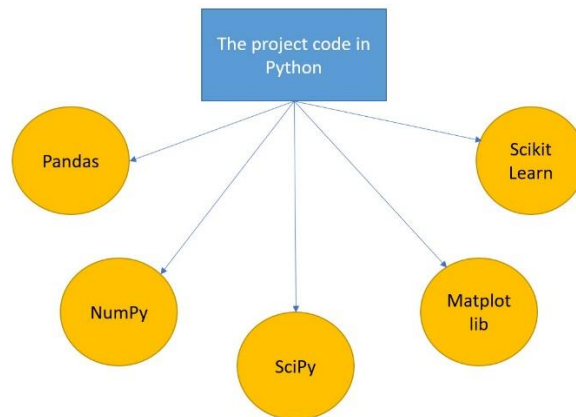


Figure B. 1: Libraries used in the Python code of the project

- *Pandas*: It is a library on top of Python used for reading and writing data between in-memory data structures and different formats. We used pandas to read data from our datasets in Excel format.
- *NumPy*: This is a package for scientific computing in Python. We used this library in our code to implement gaussian PDF in order to generate random noise.
- *SciPy*: SciPy library is one of the core packages of SciPy stack. It provides numerical routines, like routines for numerical integration, interpolation, optimization, linear algebra, and statistics. We used SciPy in order to benefit from interpolation process in ROC curves.
- *Matplotlib*: This is a library to create static, animated, and interactive visualizations in Python. We used this library to create our ROC curves for the prediction models.

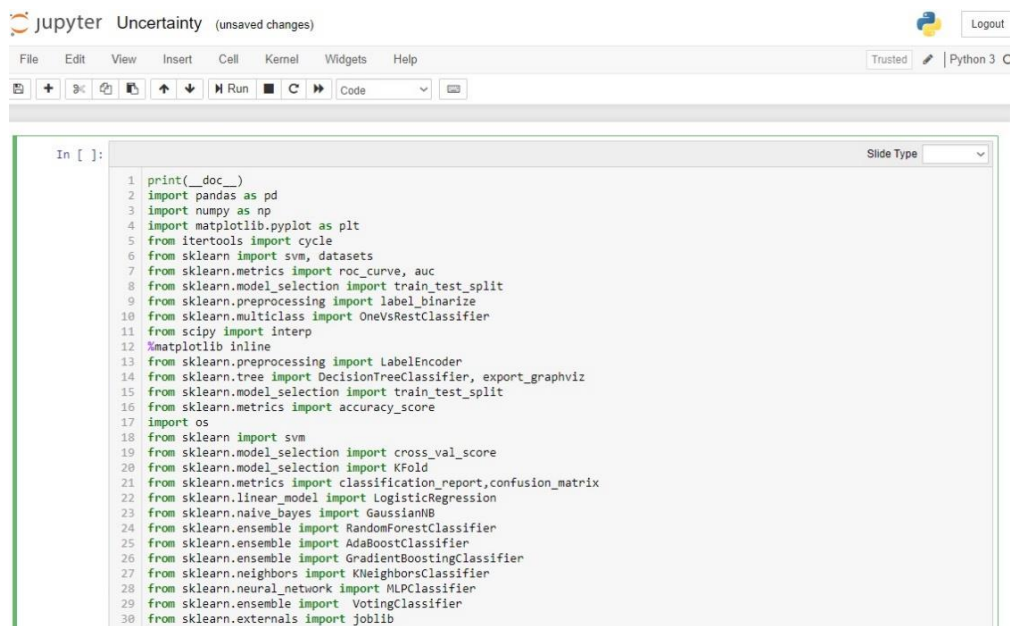
¹⁴ Project Jupyter at <https://jupyter.org>

- *Scikit Learn*: It is a library for machine learning in Python. We used this library to implement classification algorithms for the sake of training and testing. It also provides cross validation process in addition to the metrics for performance measurement.

We uploaded the python code for building our prediction models on Zenodo¹⁵¹⁶.

B.1 Building Indeterminacy-Aware Prediction Models

The code starts with importing mentioned libraries. Figure B.2 shows the screenshot of this part of the code.



```

1 print(__doc__)
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 from itertools import cycle
6 from sklearn import svm, datasets
7 from sklearn.metrics import roc_curve, auc
8 from sklearn.model_selection import train_test_split
9 from sklearn.preprocessing import label_binarize
10 from sklearn.multiclass import OneVsRestClassifier
11 from scipy import interp
12 %matplotlib inline
13 from sklearn.preprocessing import LabelEncoder
14 from sklearn.tree import DecisionTreeClassifier, export_graphviz
15 from sklearn.model_selection import train_test_split
16 from sklearn.metrics import accuracy_score
17 import os
18 from sklearn import svm
19 from sklearn.model_selection import cross_val_score
20 from sklearn.model_selection import KFold
21 from sklearn.metrics import classification_report, confusion_matrix
22 from sklearn.linear_model import LogisticRegression
23 from sklearn.naive_bayes import GaussianNB
24 from sklearn.ensemble import RandomForestClassifier
25 from sklearn.ensemble import AdaBoostClassifier
26 from sklearn.ensemble import GradientBoostingClassifier
27 from sklearn.neighbors import KNeighborsClassifier
28 from sklearn.neural_network import MLPClassifier
29 from sklearn.ensemble import VotingClassifier
30 from sklearn.externals import joblib

```

Figure B. 2: Libraries imported into the project

Dataset files are in Excel format. As mentioned earlier, we used Pandas library to read from dataset files. We have three labelled datasets for low, medium and high mobility environments. Each of them consists of three attributes. In line 141, the values of three attributes are stored in X and in line 142, the values of label are stored in Y. Figure B.3 shows the lines of code related to this part.

¹⁵ Uncertainty-Aware code is available at: <https://zenodo.org/record/3755598>

¹⁶ Ambiguity-Aware code is available at: <https://zenodo.org/record/3756014>

```

140 dataset = pd.read_excel('LD.xlsx')
141 X = dataset.iloc[:,0:3].values
142 Y = dataset.iloc[:,3].values

```

Figure B. 3: Importing dataset

For indeterminacy-aware prediction models, we used another set of datasets which have 4 attributes. As discussed in Chapter 6, these datasets have an extra attribute called trust. For reading those datasets we made necessary changes in the above code (we changed 3 to 4).

In order to make our case more realistic, we associated gaussian noise to our datasets. In doing so, we used “random” method from NumPy package. Figure B.4 shows implementation of the noise.

```

1
random_state = np.random.RandomState(0)
n_samples, n_features = X.shape
print(X.shape)
X = np.c_[X, random_state.randn(n_samples, 200 * n_features)]

```

Figure B. 4: Associating gaussian noise to our datasets

We also used 10-fold cross validation. For this reason, we split dataset into 10 parts and one of those 10 parts was used for testing and 9 parts were used for the training. We shuffled test associated part in order to build the best-fit model. Figure B.5 shows all these efforts.

```

# shuffle and split training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.1, random_state=seed)

```

Figure B. 5: Cross validation and shuffling

The project code has a function called Report. This function is responsible to report the accuracy and confusion matrix for the prediction models. As shown in Figure B.6, it uses k-fold cross validation (k=10) at line 120. The model is trained in line 123 and the function calculates prints the values of accuracy and confusion matrix after cross validation through lines 125 to 130.

```

117 def Report(name,model,X, Y, X_train,y_train,X_test,y_test):
118     print(name)
119     print('cross_val_score')
120     kfold = KFold(n_splits=10, shuffle=True, random_state=0)
121     results = cross_val_score(model, X, Y, cv=kfold)
122     print("cross_val_score: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
123     model.fit(X_train, y_train)
124     y_pred = model.predict(X_test)
125     accuracy = accuracy_score(y_test, y_pred)
126     print('accuracy score: {}'.format(accuracy))
127     #Accuracy of the predicted values
128     print('confusion_matrix')
129     print(classification_report(y_test,y_pred))
130     print(confusion_matrix(y_test,y_pred))
131
132     #save model
133     joblib.dump(model, name+'.sav')
134     return ;
135

```

Figure B. 6: Report function

Furthermore, developed prediction model is saved with “sav” extension to be used later with new datasets. You can find related code in the above figure in line 133. For the visualization purpose, we used Matplotlib to draw ROC curves for each prediction model. In order to produce ROC curves, we first need to calculate required parameters like FPR and TPR. Figure B.7 shows the screenshot of the code related to these calculations (line 102 to 115).

```

101     # Compute ROC curve and ROC area for each class
102     fpr = dict()
103     tpr = dict()
104     roc_auc = dict()
105     for i in range(n_classes):
106
107         fpr[i], tpr[i], _ = roc_curve(y_test[:, i], y_score[:, i])
108         roc_auc[i] = auc(fpr[i], tpr[i])
109
110     # Compute micro-average ROC curve and ROC area
111     fpr["micro"], tpr["micro"], _ = roc_curve(y_test.ravel(), y_score.ravel())
112     roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])
113     #return fpr,tpr,roc_auc,n_classes
114     ROCPlt(fpr,tpr,roc_auc,n_classes)
115     return fpr,tpr,roc_auc,n_classes

```

Figure B. 7: Calculating parameters for drawing ROC curves

```

34 def ROCPlt(fpr,tpr,roc_auc,n_classes):
35     plt.figure()
36     lw = 1
37     all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))
38     mean_tpr = np.zeros_like(all_fpr)
39     for i in range(n_classes):
40         mean_tpr += interp(all_fpr, fpr[i], tpr[i])
41     mean_tpr /= n_classes
42     fpr["macro"] = all_fpr
43     tpr["macro"] = mean_tpr
44     roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])
45     plt.figure()
46     plt.plot(fpr["micro"], tpr["micro"],
47             label='micro-average ROC curve (area = {0:0.2f})'
48             ''.format(roc_auc["micro"]),
49             color='deeppink', linestyle=':', linewidth=4)
50
51     plt.plot(fpr["macro"], tpr["macro"],
52             label='macro-average ROC curve (area = {0:0.2f})'
53             ''.format(roc_auc["macro"]),
54             color='navy', linestyle=':', linewidth=4)
55
56     colors = cycle(['aqua', 'darkorange', 'cornflowerblue'])
57     for i, color in zip(range(n_classes), colors):
58         plt.plot(fpr[i], tpr[i], color=color, lw=lw,
59                 label='ROC curve of class {0} (area = {1:0.2f})'
60                 ''.format(i, roc_auc[i]))
61
62     plt.plot([0, 1], [0, 1], 'k--', lw=lw)
63     plt.xlim([0.0, 1.0])
64     plt.ylim([0.0, 1.05])
65     plt.xlabel('False Positive Rate')
66     plt.ylabel('True Positive Rate')
67     plt.title('Some extension of Receiver operating characteristic to multi-class')
68     plt.legend(loc="lower right")
69     plt.show()

```

Figure B. 8: ROC curves plotting

Figure B.8 shows a code segment related to plotting ROC curves. Calculated parameters derived from the last code segment will be passed a function called “ROCPlt” in this part of the code (line 34). This function draws ROC plots for all prediction models built by classifiers in the code. It draws ROC curves for both Access and Deny classes in addition to the curves drawn as micro-average and macro-average in every plot (lines 35 to 55). It uses three different colours for these curves in each plot (lines 56 to 60).

The last comment on the code is about classifiers. As discussed in the dissertation, this research benefits from classification algorithms in order to build prediction models. Ten classifiers were implemented by this code. Figure B.9 depicts a code segment about these classifiers. These algorithms were implemented using Scikit Learn library. Each classifier has a default configuration. In order to get the best-fit model, we made changes into some of these configurations. For example, in the first algorithm (Decision Tree), we have tested different split criteria instead of “gini” to find the best-fit model for each dataset. Moreover, we made change in the value of minimum sample split and minimum sample leaf (in line 151).

For the random forest classifier (line 168), different values were used as the number of estimators and the depth of the forest.

For K-NN algorithm (line 171), different number of nearest neighbours were determined to evaluate the performance of the model.

For neural networks (lines 174 to 181), we have tested four activation functions discussed in Subsection 5.2.9 by changing the value of activation parameter. We have also changed the number of hidden layer and the number of neurons in line 176 of the code based on the configurations defined in Table 5.10. In this line, `hidden_layer_sizes()` should reflect both the number of hidden layers and the number of included neurons. As an example, shown in Figure B.9 at line 176, `hidden_layer_sizes(10)` means current MLP has one hidden layer consisting of 10 neurons. We also changed the number of epoch (at line 177) and the type of the solver (at line 179) to compare the performance of the models.

For voting algorithms, we have used both “hard” and “soft” voting modes to build and test our prediction model by making necessary changes in lines 198 and 199.


```

151 model1 = DecisionTreeClassifier(criterion='gini',min_samples_leaf=5,min_samples_split=5,max_depth=None,random_state=seed)
152 fpr[0],tpr[0],roc_auc[0],n_classes[0]=ReportWithROC('DecisionTreeClassifier',model1,X,Y)
153 print('fpr:',fpr[0])
154
155 model2 = svm.SVC(gamma='auto')
156 fpr[1],tpr[1],roc_auc[1],n_classes[1]=ReportWithROC('svm.SVC',model2,X,Y)
157
158 model3 = LogisticRegression(C=1e5, solver='lbfgs', multi_class='multinomial')
159 fpr[2],tpr[2],roc_auc[2],n_classes[2]=ReportWithROC('LogisticRegression',model3,X,Y)
160
161
162 model4 = GaussianNB()
163 fpr[3],tpr[3],roc_auc[3],n_classes[3]=ReportWithROC('GaussianNB',model4,X,Y)
164
165 model5 = AdaBoostClassifier(base_estimator=None, n_estimators=50, learning_rate=1.0,random_state=None)
166 fpr[4],tpr[4],roc_auc[4],n_classes[4]=ReportWithROC('AdaBoostClassifier',model5,X,Y)
167
168 model6=RandomForestClassifier(n_estimators=10, max_depth=2,random_state=0)
169 fpr[5],tpr[5],roc_auc[5],n_classes[5]=ReportWithROC('RandomForestClassifier',model6,X,Y)
170
171 model7=KNeighborsClassifier(n_neighbors=3)
172 fpr[6],tpr[6],roc_auc[6],n_classes[6]=ReportWithROC('KNeighborsClassifier',model7,X,Y)
173
174 model8=MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
175                      beta_2=0.999, early_stopping=False, epsilon=1e-08,
176                      hidden_layer_sizes=(10), learning_rate='constant',
177                      learning_rate_init=0.001, max_iter=500, momentum=0.9,
178                      nesterovs_momentum=True, power_t=0.5, random_state=None,
179                      shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1,
180                      verbose=False, warm_start=False)
181 fpr[7],tpr[7],roc_auc[7],n_classes[7]=ReportWithROC('MLPClassifier',model8,X,Y)
182
183 model9 = GradientBoostingClassifier(n_estimators=20, max_features=2, max_depth = 2, random_state = 0)
184 fpr[8],tpr[8],roc_auc[8],n_classes[8]=ReportWithROC('GradientBoostClassifier',model9,X,Y)
185
186
187 model10=VotingClassifier(estimators=[
188     (' model11', model11)
189     ,(' model12', model12)
190     ,(' model13', model13)
191     ,(' model14', model14)
192     ,(' model15', model15)
193     ,(' model16', model16)
194     ,(' model17', model17)
195     ,(' model18', model18)
196     ,(' model19', model19)
197     ]
198     , voting='hard'
199     #,voting='soft'
200
201 )
202
203 fpr[9],tpr[9],roc_auc[9],n_classes[9]=ReportWithROC('VotingClassifier',model10,X,Y)

```

Figure B. 9: Implementing 10 classifiers using Scikit Learn library

B.2 Implementing Prediction Models on Raspberry Pi

In this Subsection, we first introduce Raspberry Pi then we train our prediction model using Raspberry Pi to show that our approach is feasible for resource-constrained devices in IoT. Finally, we test our prediction model using a new dataset on Raspberry Pi to see how our prediction model performs in action.

B.2.1 Raspberry Pi

With the advent of ubiquitous low-cost, low-power computing devices (e.g. Raspberry Pi), exploring various solutions in the field of IoT security becomes more convenient. Raspberry Pi is a pocket size, ARM-based architecture computer. We adopted Raspberry Pi 4 Model B to our experiments in order to build and implement our prediction model. Figure B.10 demonstrated the picture of the raspberry pi board. The hardware specification of the raspberry pi used in this research is listed in Table B.1.



Figure B. 10: Board of Raspberry Pi 4, Model B

Table B. 1: Hardware specifications for Raspberry Pi 4 Model B

Processor	Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Memory	4GB LPDDR4
Connectivity	2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0, BLE, Gigabit Ethernet 2 × USB 3.0 ports, 2 × USB 2.0 ports.
SD card Support	Micro SD card slot for loading operating system and data storage
Input Power	5V DC via USB-C connector (minimum 3A) Power over Ethernet (PoE)–enabled

We used Raspbian¹⁷ as the operating system for our raspberry pi machines involved in our experiments. Raspbian is a Raspberry Pi OS based on Debian Linux. We installed Raspbian Image on a 64GB SD card to be used in the raspberry pi devices. The version of Raspbian used in the experiments is 20-06-2019. In the next Subsection, we will show screenshots from Raspbian environment which depict the process of building prediction models using raspberry pi. We also show screenshots of our experiments.

B.2.2 Building Prediction Model on Raspberry Pi

Python comes pre-installed on most Linux distributions and Raspbian is no exception. We updated the version of python on Raspbian and installed Jupyter in order to benefit from an interactive coding environment. Figure B.11 shows the process of installing Jupyter on the Raspbian. We use bash command line to install Jupyter by running the following command:

```
$ pip install jupyter
```

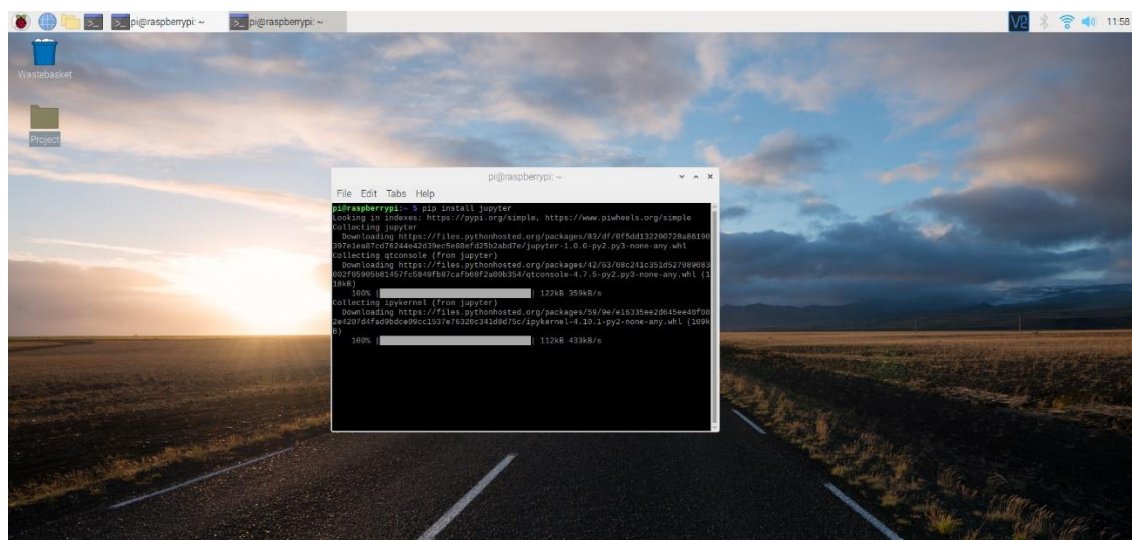


Figure B. 11: Jupyter installation process

After installing Jupyter, we imported our code and run it. Raspberry Pi machine showed acceptable performance in building our prediction models. Figure B.12 demonstrates our code running by Jupyter on Raspberry Pi machine.

¹⁷ Available at: <https://www.raspberrypi.org/documentation/raspbian/>

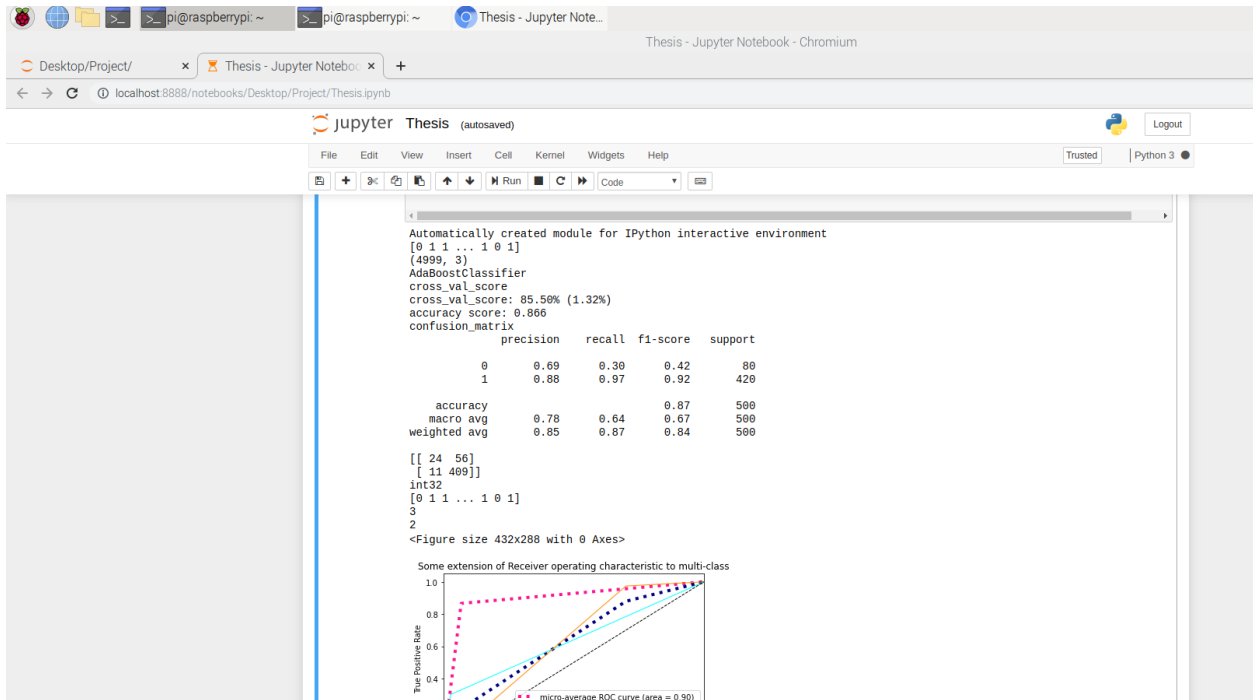


Figure B. 12: Building our prediction model on Raspberry Pi

As shown in the above picture, we built our designated model for low-mobility environment using AdaBoost classifier algorithm. The model is 29.5KB. Then we tested our model using new authentication requests on Raspberry Pi. Figure B.13 shows the results of our test.

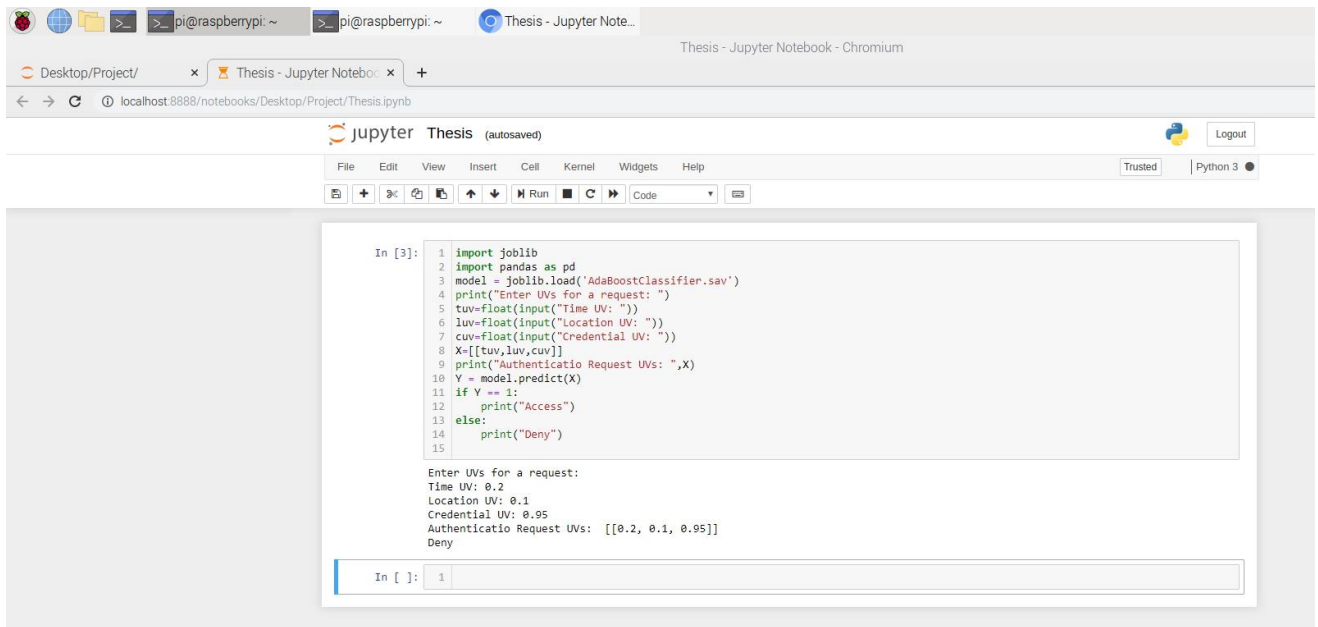


Figure B. 13: Prediction of an authentication request

In the above test, we passed three uncertainty values for time, location, and credential (0.2, 0.1 and 0.95 respectively) to the model and the model predicted the class of authentication as “Deny” correctly. We also tested the performance of our model running by raspberry pi in terms of scalability of entities. For this reason, we assumed that our model is dealing with 5000 authentication requests simultaneously. Figure B.13 shows the results.

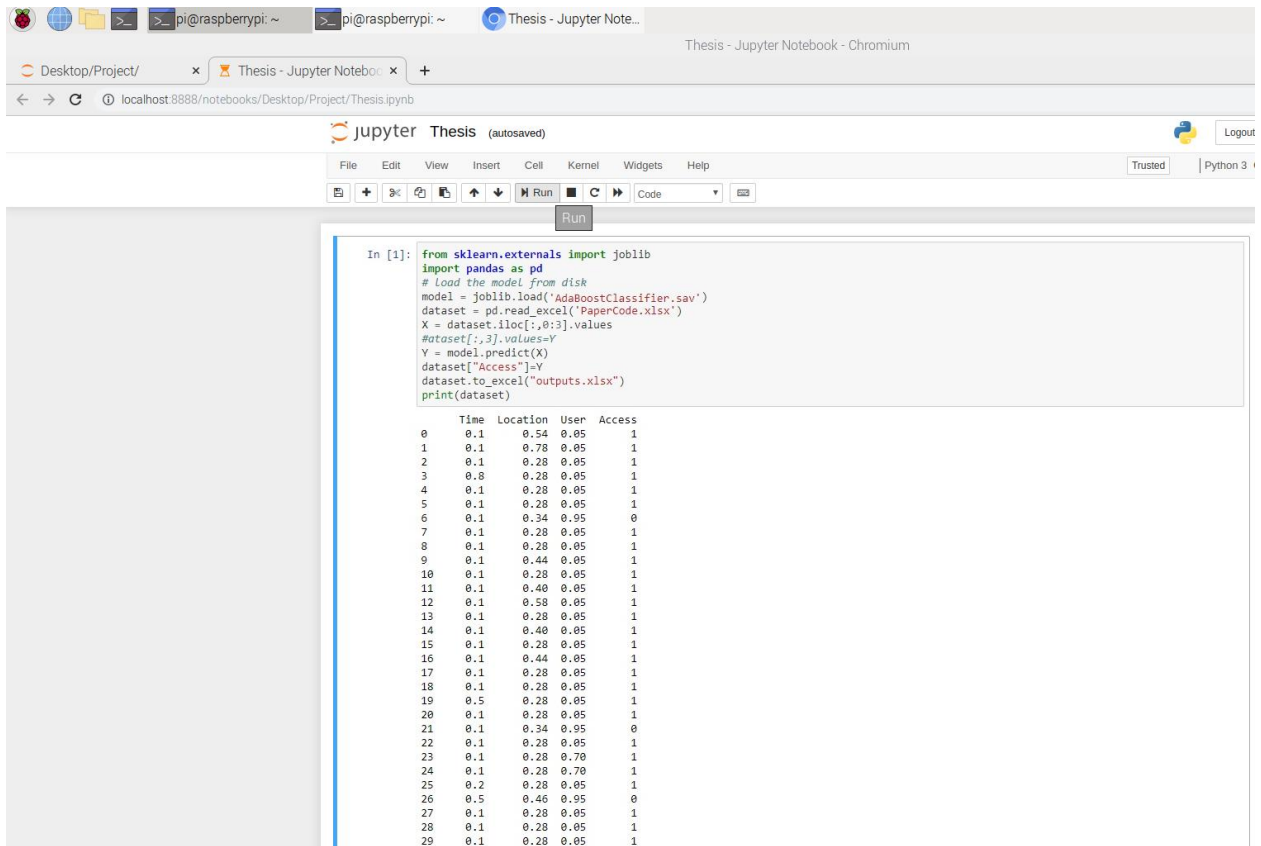


Figure B. 14: Running prediction model for 5000 authentication requests on Raspberry Pi

As shown, we ran our model for both single and concurrent authentication request. The model can be run in bash environment using “python AdaBoosClassifier.py” command as well. The time for handling 5000 authentication requests on Raspberry Pi was measured using “time ./AdaBoostClassifier” command. The real time, the time elapsed between invocation and termination of the command, was 23s.

Bibliography

- [1] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, ""Context aware computing for the Internet of Things: A survey", " *IEEE Communication surveys and tutorials*, vol. 16, no. 1, 2014.
- [2] B. Marr, "Why The Internet Of Medical Things (IoMT) Will Start To Transform Healthcare In 2018," *Forbes*, 2018.
- [3] Wei Zhou, Yan Jia, Anni Peng, Yuqing Zhang, and Peng Liu, "The Effect of IoT New Features on Security and Privacy: New Threats, Existing Solutions, and Challenges Yet to Be Solved," *IEEE Internet of Things Journal*, pp. 1-11, 2018.
- [4] ELISA BERTINO, KIM-KWANG RAYMOND CHOO, DIMITRIOS GEORGAKOPOLOUS, SURYA NEPAL, "Internet of Things (IoT): Smart and Secure Service Delivery," *ACM Transactions on Internet Technology*, vol. 16, no. 4, pp. 22-29, 2016.
- [5] Francesco Restuccia, Salvatore D'Oro and Tommaso Melodia, "Securing the Internet of Things in the Age of Machine Learning and Software-defined Networking," *IEEE Internet of Things*, vol. 1, no. 1, p. IEEE Early Access Service, 2018.
- [6] H. Reza Ghorbani ; M. Hossein Ahmadzadegan, "Security challenges in internet of things: survey," in *IEEE Conference on Wireless Sensors (ICWiSe)*, 2017.
- [7] Mario FRUSTACI ; Pasquale PACE ; Gianluca ALOI ; Giancarlo FORTINO, "Evaluating critical security issues of the IoT world: Present and Future challenges," *IEEE Internet of Things Journal* , pp. 2327-4662, 2017.
- [8] C. Zhang and R. Green, "Communication Security in Internet of Thing: Preventive measure and avoid DDoS attack over IoT network," in *IEEE Symposium on Communications & Networking*, 2015.
- [9] Sadegh Dorri, Rasool Jalili, "TIRIAC: A trust-driven risk-aware acces control framework for Grid enviroments," *Future Generation Computer Systems*, vol. 55, pp. 238-254, 2016.
- [10] Jiawen Kang, Rong Yu, Xumin Huang, Magnus Jonsson, Hanna Bogucka, Stein Gjessing, and Yan Zhang, "Location Privacy Attacks and Defenses in Cloud-Enabled Internet of Vehicles," *IEEE Wireless Communications*, pp. 52-59, 2016.
- [11] Novák, Perfilieva, Dvorak, "What is fuzzy modelling?," in *Insight into Fuzzy Modeling*, Wiley, 2016, pp. 3-9.
- [12] "Information on RFC 4949," IETF, 1 1 2018. [Online]. Available: <https://www.rfc-editor.org/info/rfc4949>. [Accessed 1 1 2018].
- [13] William Stallings, "Access Control," in *Computer Security, principles and practice*, Pearson, 2017.
- [14] D. Gollmann, "Access Control," in *Computer Security*, Wiley, 2011.

- [15] Aafaf Ouaddah , Hajar Mousannif , Anas Abou Elkalam , Abdellah Ait Ouahman , "Access control in the Internet of Things: Big challenges and new opportunities," *Elsevier Computer Networks*, vol. 112, pp. 237-262, 2017.
- [16] William Stallings, Lawrie Brown, "Access Control," in *Computer Security: Principles and Practice, 3rd Edition*, Pearson, 2015, pp. 113-154.
- [17] D. Gollmann, "Chapter 5: Access Control," in *Computer Security*, John Wiley & Sons, 2011.
- [18] Jin, X., Krishnan, R., & Sandhu, R., "A Unified Attribute-Based Access Control Model Covering DAC, MAC And RBAC," *Springer Lecture Notes in Computer Science: Data and Applications Security and Privacy*, vol. 7371, pp. 41-55, 2012.
- [19] R.S. Sandhu and P. Samarati, "Access control: Principle and practice," *IEEE Communication Magazine*, vol. 32, pp. 40-48, 1994.
- [20] Vijayakumar, H., Jakka, G., Rueda, S., Schiffman, J., & Jaeger, T., "Integrity Walls: Finding Attack Surfaces from Mandatory Access Control Policies," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, 2012.
- [21] K. J. Biba, "Integrity consideration for secure computer systems. Technical Report," The MITRE Corporation, Bedford, MA, 1977.
- [22] D. Clark, and D. Wilson, "A comparison of commercial and military computer security policy," in *IEEE Symposium on Security and Privacy* , 1987.
- [23] D. F. C. Brewer and M. J. Nash., " The Chinese Wall security policy.," in *In Proceedings of 1989 IEEE symposium on Security and Privacy*, 1989.
- [24] D. K. Ferraiolo, D. Kuhn, "Role Based Access Control," in *15Th International Computer Security Conference*, 1992.
- [25] V. Suhendra, "A Survey on Access Control Deployment," in *International Conference on Security Technology (FGIT)*, 2014.
- [26] Lagutin, D., Visala, K., Zahemszky, A., Burbridge, T., & Marias, G. F, "Roles and Security in a Publish/Subscribe Network Architecture," in *IEEE Symposium on Computers and Communications (ISCC)*, 2012.
- [27] A. Singh, "Role Based Trust Management Security Policy Analysis," in *International Journal of Engineering Research and Applications (IJERA)*, 2012.
- [28] W.W. Smari, P. Clemente, J.-F. Lalande, "An extended attribute based access control model with trust and privacy: application to a collaborative crisis management system," *Future Generation of Computer System*, vol. 31, pp. 147-168, 2014.
- [29] Li, J., Chen, X., Li ,J., Jia, C., Ma, J., & Lou, W, "Fine-Grained Access Control System Based on Outsourced Attribute-Based Encryption," *Springer Computer Security*, vol. 8134, pp. 592-602, 2014.

- [30] J.B. Dennis, E.C. Van Horn, "Programming semantics for multiprogrammed computations," *ACM Communication*, vol. 3, pp. 143-155, 1966.
- [31] A. Lazowski, F. Martinelli, P. Mori, "Usage control in computer security: a survey," *Elsevier Journal of Computer Science*, vol. 4, 2010.
- [32] X. Zhang, M. Nakae, M.J. Covington, R. Sandhu, "Toward a usage-based security framework for collaborative computing systems," *ACM Transaction on Information system security*, vol. 11, 2008.
- [33] A. Kalam, R. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Mieke, C. Saurel, G. Trouessin, "Organization based access control," in *IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, 2003.
- [34] A. Nordrum, "The Internet of Fewer Things," *IEEE Spectrum*, vol. 10, pp. 12-13, 2016.
- [35] YUANKUN XUE, JI LI, SHAHIN NAZARIAN, and PAUL BOGDAN, "Fundamental Challenges Toward Making the IoT a Reachable Reality: A Model-Centric Investigation," *ACM Transactions on Design Automation of Electronic Systems*, vol. 22, no. 3, 2017.
- [36] Raffaele Giaffreda ; Luca Capra ; Fabio Antonelli, "A pragmatic approach to solving IoT interoperability and security problems in an eHealth context," in *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on*, 2016.
- [37] Yanping Li ; Yanjiao Qi ; Laifeng Lu, "Secure and Efficient V2V Communications for Heterogeneous Vehicle Ad Hoc Networks," in *International Conference on Networking and Network Applications (NaNA)*, 2017.
- [38] Bo Cheng, Member, IEEE, Ming Wang, Shuai Zhao, Zhongyi Zhai, Da Zhu, and Junliang Chen, "Situation-Aware Dynamic Service Coordination in an IoT Environment," *IEEE/ACM TRANSACTIONS ON NETWORKING*, vol. 25, no. 4, pp. 2082-2095, 2017.
- [39] Srdjan Marinovic, Robert Craven, Jiefei Ma, "Rumpole: A Flexible Break-glass Access Control Model," in *The ACM Symposium on Access Control Models and Technologies (SACMAT)*, Austria, 2011.
- [40] Syed Zain R. Rizvi Philip W. L. Fong, "Interoperability of Relationship- and Role-Based Access Model," in *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, 2016.
- [41] Sun Kaiwen, Yin Lihua, "Attribute-Role-Based Hybrid Access Control in the Internet of Things," in *Web Technologies and Applications*, Springer, 2014.
- [42] Sun Kaiwen Yin Lihua, "Attribute-Role-Based Hybrid Access Control in the Internet of Things," in *International Conference on Web Technologies and Applications. APWeb* , 2014.
- [43] Prosunjit Biswas, Ravi Sandhu, Ram Krishnan, "Attribute Transformation for Attribute-Based Access Control," in *Proceedings of the 2nd ACM International Workshop on Attribute-Based Access Control*, 2017.

- [44] Bayu Anggorojati ; Ramjee Prasad, "Securing communication in inter domains Internet of Things using identity-based cryptography," in *International Workshop on Big Data and Information Security (IWBIS)*, 2017.
- [45] Jia Jindou ; Qiu Xiaofeng ; Cheng Cheng, "Access Control Method for Web of Things Based on Role and SNS," in *IEEE 12th International Conference on Computer and Information Technology (CIT)*, 2012.
- [46] E. Barka, S.S. Mathew, Y. Atif, "Securing the Web of Things with Role- Based Access Control," in *Springer International Conference on Codes, Cryptology, and Information Security*, 2015.
- [47] Jing Liu ; Yang Xiao ; C.L. Philip Chen, "Authentication and Access Control in the Internet of Things," in *IEEE 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2012.
- [48] Waleed W.Smaria Patrice Clemente Jean-Francois Lalande, "An extended attribute based access control model with trust and privacy: Application to a collaborative crisis management system," *Future Generation Computer Systems*, vol. 31, pp. 147-168, 2014.
- [49] Harsha S.Gardiyawasam Pussewalage Vladimir A.Oleshchuk, "Attribute based access control scheme with controlled access delegation for collaborative E-health environments," *Elsevier Journal of Information Security and Applications*, vol. 37, pp. 50-64, 2017.
- [50] G. Zhang, W. Gong, "The research of access control based on UCON in the in- ternet of things," *Journal of Software*, vol. 6, no. 4, 2011.
- [51] Anggorojati, B. , Mahalle, P.N., Prasad, N.R., "Secure Access Control and Authority Delegation Based on Capability and Context Awareness for federated IoT," in *Internet of Things and M2M Communications*, River Publisher, 2013, pp. 135-160.
- [52] P. Mahalle, "Identity authentication and capability based access con- trol (IACAC) for the internet of things," *Journal of Cyber Security and Mobility*, vol. 1, pp. 309-348, 2013.
- [53] L. Gong, "A secure identity-based capability system," in *IEEE Symposium on Security and Privacy*, 1989.
- [54] Sergio Gusmeroli, Salvatore Piccione, Domenico Rotondi, "A capability-based security approach to manage access control in the Internet of Things," *Mathematical and Computer Modelling*, vol. 58, pp. 1189-1205, 2013.
- [55] Lo -Yao Yeh, Pei-Yu Chiang, Yi-Lang Tsai and Jiunand Jiun-Long Huang, "Cloud-based Fine-grained Health Information Access Control Framework for Lightweight IoT Devices with Dynamic Auditing and Attribute Revocation," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, 2015.
- [56] Fagen Li , Yanan Han , Chunhua Jin, "Practical access control for sensor networks in the context of the Internet of Things," *Elsevier Computer Communications*, Vols. 89-90, pp. 154-164, 2016.

- [57] Sudha Patel, Dhiren R. Patel, Ankit P. Navik, "Energy Efficient Integrated Authentication and Access Control Mechanisms for Internet of Things," in *IEEE International Conference on Internet of Things and Applications (IOTA)*, 2016.
- [58] Aafaf Ouaddah ; Imane Bouij-Pasquier ; Anas Abou Elkalam ; Abdellah Ait Ouahman, "Security analysis and proposal of new access control model in the Internet of Thing," in *IEEE International Conference on Electrical and Information Technologies (ICEIT)*, 2015.
- [59] Savio Sciancalepore, Giuseppe Piro, Daniele Caldarola, Gennaro Boggia and Giuseppe Bianchi, "OAuth-IoT: an access control framework for the Internet of Things based on open standards," in *IEEE Symposium on Computers and Communications (ISCC)*, 2017.
- [60] Faisal Alsubaei, Sajjan Shiva, Abdullah Abuhussein, "Security and Privacy in the Internet of Medical Things: Taxonomy and Risk Assessment," in *IEEE 42nd Conference on Local Computer Networks Workshops*, 2017.
- [61] Raffaele Martino, Salvatore D'Antonio, Luigi Coppolino, Luigi Romano, "Security in Cross - Border Medical Data Interchange: a Technical Analysis and a Discussion of Possible Improvements," in *IEEE 41st Annual Computer Software and Applications Conference*, 2017.
- [62] STEPHANIE B. BAKER, WEI XIANG, (Senior Member, IEEE), AND IAN ATKINSON, "Internet of Things for Smart Healthcare: Technologies, Challenges, and Opportunities," *IEEE Access* , vol. 5, pp. 26521-26544, 2017.
- [63] Paul A. Wortman ; Fatemeh Tehranipoor ; Nima Karimian ; John A. Chandy, "Proposing a Modeling Framework for Minimizing Security Vulnerabilities in IoT Systems in the Healthcare Domain," in *2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, 2017.
- [64] Hamid Sharif, Michael Hempel, Thomas Mikchael Bohnert, Ali Khoyneshad, "Wireless communications for E-health applications," *IEEE Wireless Communications*, vol. 20, no. 4, pp. 10-11, 2013.
- [65] Md. Eshrat E. Alahi, Najid Pereira-Ishak, Subhas Chandra Mukhopadhyay, Lucy Burkitt, "An Internet-of-Things Enabled Smart Sensing System for Nitrate Monitoring," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4409-4417, 2018.
- [66] E. D. Hardt, "The OAuth 2.0 Authorization Framework," Internet Engineering Task Force (IETF), RFC: 6749, 2012.
- [67] Daniel Fett, Ralf Küsters, and Guido Schmitz, "The Web SSO Standard OpenID Connect: In-Depth Formal Security Analysis and Security Guidelines," in *IEEE 30th Computer Security Foundations Symposium*, 2017.
- [68] "Securing Digital Identities in the Cloud by Selecting an Apposite Federated Identity Management from SAML, OAuth and OpenID Connect," in *IEEE 11th International Conference on Research Challenges in Information Science (RCIS)*, 2017.

- [69] Xin Wang, Henning Schulzrinne, Dilip Kandlur and Dinesh Verma, "Measurement and Analysis of LDAP Performance," *IEEE/ACM TRANSACTIONS ON NETWORKING*, vol. 16, no. 1, pp. 232-243, 2018.
- [70] An Braeken, Madhusanka Liyanage, Anca Delia Jurcut, "Anonymous Lightweight Proxy Based Key Agreement for IoT," *Springer Wireless Personal Communications*, p. 345–364, 2019.
- [71] Anca Jurcut, Tom Coffey and Reiner Dojen, "A Novel Security Protocol Attack Detection Logic with Unique Fault Discovery Capability for Freshness Attacks and Interleaving Session Attacks," *IEEE Transactions on Dependable and Secure Computing*, 2017.
- [72] T. Moses, " "Extensible Access Control Markup Language (XACML)," OASIS, 2013.
- [73] S. Savinov, "A Dynamic Risk-Based Access Control Approach: Model and Implementation," *PhD Thesis, University of Waterloo*, 2017.
- [74] F. Salim, "Approaches to Access Control Under Uncertainty," *PhD Thesis, Queensland University of Technology*, 2012.
- [75] A. Ferreira, R. Cruz-Correia and L. Antunes, "How to Break Access Control in a Controlled Manner," in *19th IEEE International Symposium on Computer-Based Medical Systems*, 2006.
- [76] Htoo Aung Maw, Hannan Xiao, Bruce Christianson, and James A. Malcolm, "BTG-AC: Break-the-Glass Access Control Model for Medical Data in Wireless Sensor Networks," *IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS*, , vol. 20, no. 3, pp. 763-774, 2016.
- [77] Schefer-Wenzl, S., & Strembeck, M., "Generic Support for RBAC Break-Glass Policies in Process-Aware Information Systems," in *28Th Annual ACM Symposium on Applied Computing*, 2013.
- [78] D. Povey, "Optimistic Security: A New Access Control Paradigm," in *ACM workshop on New security paradigms*, 1999.
- [79] Patrick D. Gallagher, "NISP SP800-30 Guide for Conducting Risk Assessment," NIST, 2012.
- [80] Molloy, I., Dickens, L., Morisset, C., Cheng, P. C., Lobo, J., & Russo, A., "Risk-Based Security Decisions under Uncertainty," in *Proceedings of the Second ACM Conference on Data and Application Security and Privacy*, 2012.
- [81] Alireza Shameli-Sendi, Rouzbeh Aghababaei-Barzegar, Mohamed Cheriet, "Taxonomy of information security risk assessment (ISRA)," *computers & security* , vol. 57, pp. 14-30, 2016.
- [82] Elaine Hulitt, B. Vaughn, "Information system security compliance to FISMA standard: a quantitative measure," *Springer Telecommunication Systems*, vol. 45, p. 139–152, 2010.
- [83] E. Wheeler, building an information security risk management program from the ground up., Syngress, 2011.

- [84] Fugini, M., Teimourikia, M., & Hadjichristofi, G., "A web-based cooperative tool for risk management with adaptive security," *Elsevier Journal of Future Generation Computer Systems*, 2015.
- [85] Molloy, I., Dickens, L., Morisset, C., Cheng, P. C., Lobo, J., & Russo, A., "Risk-Based Security Decisions under Uncertainty," in *Proceedings of the Second ACM Conference on Data and Application Security and Privacy*, 2012.
- [86] Hany F. Atlam^{1, 2}, Ahmed Alenezi¹, Robert J. Walters¹, Gary B. Wills¹, Joshua Daniel, "Developing an adaptive Risk-based access control model for the Internet of Things," in *IEEE International Conference on Internet of Things (iThings)*, 2017.
- [87] Hemanth Khambhammettu, Sofiene Boulares, Kamel Adi, Luigi Logrippo, "A framework for risk assessment in access control systems," *Elsevier Computers and Security*, vol. 39, pp. 86-103, 2013.
- [88] DIMITRIS GRITZALIS, GIULIA ISEPPI, ALEXIOS MYLONAS and VASILIS STAVROU, "Exiting the Risk Assessment maze: A meta-survey," *ACM Computing Surveys*, 2018.
- [89] Khalid Zaman Bijon, Ram Krishnan, Ravi Sandhu, "A framework for risk-aware role based access control," in *IEEE Conference on Communications and Network Security (CNS)*, 2013.
- [90] Dong Xie, Yongrui Qin, Quan Z. Sheng, "Managing Uncertainties in RFID Applications: A Survey," in *11th IEEE International Conference on e-Business Engineering*, 2014.
- [91] Vilém Novák, Irina Perfilieva, Antonín Dvůrák, "What is fuzzy modeling?," in *Insight into Fuzzy Modeling*, Wiley, 2016.
- [92] Y. Sakai, "J. M. Keynes on probability versus F. H. Knight on uncertainty: reflections on the miracle year of 1921," *Springer Japan Association for Evolutionary Economics*, 2016.
- [93] ZHIGUO ZENG, RUI KANG, MEILIN WEN and AND ENRICO ZIO, "A Model-Based Reliability Metric Considering Aleatory and Epistemic Uncertainty," *IEEE Access Journal*, vol. 5, 2017.
- [94] Aven and Zio, "Some considerations on the treatment of uncertainties in risk assessment for practical decision making," *Reliability Engineering & System Safety*, vol. 96, no. 1, pp. 64-74, 2011.
- [95] A. P. Dempster, "Upper and Lower Probabilities Induced by a Multivalued Mapping," *The Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325-339, 1967.
- [96] G. Shafer, A mathematical theory of evidence, Princeton University, 1976.
- [97] Baudrit, C. and Dubois, D., "Practical representations of incomplete probabilistic knowledge," *Elsevier Journal of Computational Statistics & Data Analysis*, vol. 51, no. 1, 2006.
- [98] L. B, Uncertainty Theory, Springer, 2017.

- [99] G. Apostolakis, "The concept of probability in safety assessments of technological systems," *Science*, vol. 250, 1990.
- [100] Helton, J. C. and Oberkampf, W. L., "An exploration of alternative approaches to the representation of uncertainty in model predictions.," *Journal of Reliability Engineering & System Safety*, vol. 85, no. 1, pp. 39-71, 2004.
- [101] D. V. Lindley, "The Philosophy of Statistics," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 49, no. 3, 2000.
- [102] D. V. Lindley, *Understanding uncertainty*, Wiley, 2014.
- [103] Kahneman D, and Tversky A, "Prospect theory: An analysis of decision under," *Econometrica*, vol. 47, no. 9, pp. 263-292, 1979.
- [104] L. B, "Why is there a need for uncertainty theory?," *Springer Journal of Uncertain*, vol. 6, no. 1, pp. 3-10, 2012.
- [105] L. B, *Uncertainty Theory*, Springer, 2015.
- [106] Nick Firoozye, Fauzian Arrif, *Managing Uncertainty Mitigation Risk*, Springer, 2016.
- [107] J. Bancroft, *Tolerance of Uncertainty*, Author House, 2014.
- [108] J. Barnes, *The Complete Works of Aristotle: The Revised Oxford Translation*, Princeton, 1984.
- [109] "Towards Fuzzy Type Theory with Partial Functions," *Springer Journal of Advances in Fuzzy Logic and Technology*, 2018.
- [110] L.A.Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, 1965.
- [111] N. Baracaldo, J. Joshi, "A trust-and-risk aware RBAC framework: tackling insider threat," in *ACM Proceedings of the 17th Symposium on Access Control*, 2012.
- [112] Alessandro Armando, Michele Bezzi, Francesco Di Cerbo, and Nadia Metoui, "Balancing Trust and Risk in Access Control," in *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, 2015.
- [113] Indrajit Ray, Dieudonne Mulamba, Indrakshi Ray and Keesook J. Han, "A Model for Trust-Based Access Control and Delegation in Mobile Clouds," in *IFIP Annual Conference on Data and Applications Security and Privacy*, 2013.
- [114] Mahdi Ghafoorian, Dariush Abbasinezhad-Mood, and Hassan Shakeri, "A Thorough Trust and Reputation Based RBAC Model for Secure Data Storage in the Cloud," *IEEE Transaction on Parallel and Distributed Systems*, vol. 30, no. 4, pp. 778-788, 2019.
- [115] KAMRAN AHMAD AWAN, IKRAM UD DIN, MAHDI ZAREEI AND SULTAN ULLAH JADOON, "HoliTrust-A Holistic Cross-Domain Trust Management Mechanism for Service-Centric Internet of Things," *IEEE Access*, vol. 7, pp. 52191-52201, 2019.

- [116] Ashish Singh, Kakali Chatterjee, "ITrust: identity and trust based access control model for healthcare system security," *Multimedia Tools and Applications*, vol. 78, p. 28309–28330, 2019.
- [117] Mahalle, Parikshit N.; Thakre, Pravin A.; Prasad, Neeli Rashmi; Prasad, Ramjee, "A fuzzy approach to trust based access control in internet of things," in *2013 3rd International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace and Electronic Systems*, 2013.
- [118] Farhana Jabeen, Zara Hamid, Adnan Akhunzada, Wadood Abdul, Sanaa Ghouzali, "Trust and Reputation Management in Healthcare Systems: Taxonomy, Requirements and Open Issues," *IEEE Access*, vol. 6, pp. 17246-17263, 2018.
- [119] Ava Ahadipour, Martin Schanzenbach, "A Survey on Authorization in Distributed Systems: Information Storage, Data Retrieval and Trust Evaluation," in *The 16th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (IEEE TrustCom-17)*, 2017.
- [120] "CASTRA: Seamless and Unobtrusive Authentication of Users to Diverse Mobile Services," *IEEE Internet of Things Journal*, vol. Early Access, pp. 1-16, 2018.
- [121] Guoyuan Lin ; Danru Wang ; Yuyu Bie ; Min Lei, "MTBAC: A mutual trust based access control model in Cloud computing," *IEEE Communication*, vol. 11, no. 4, 2014.
- [122] Zheng Yan, Xueyun Li, Mingjun Wang and Athanasios V. Vasilakos, "Flexible Data Access Control Based on Trust and Reputation in Cloud Computing," *IEEE TRANSACTIONS ON CLOUD COMPUTING*, vol. 5, no. 3, pp. 485-498, 2017.
- [123] Lan Zhou, Vijay Varadharajan, and Michael Hitchens, "Trust Enhanced Cryptographic Role-Based Access Control for Secure Cloud Data Storage," *IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY*, vol. 10, no. 11, pp. 2381-2395, 2015.
- [124] Loubna Mekouar, Youssef Iraqi, Raouf Boutaba, "Reputation-Based Trust Management in Peer-to-Peer Systems: Taxonomy and Anatomy," in *Handbook of Peer-to-Peer Networking*, Springer, 2009, pp. 689-732.
- [125] K.Z. Bijon, R. Krishnan, R.S. Sandhu, "Risk-aware RBAC sessions," in *IEEE 8Th International Conference on Information Systems Security, (ICISS)*, 2012.
- [126] N. Baracaldo, J. Joshi, "An adaptive risk management and access control framework to mitigate insider threats," *Elsevier Journal of Computers & Security*, vol. 39, pp. 237-254, 2013.
- [127] N. Dimmock, A. Belokosztolszki, D. Eysers, J. Bacon, K. Moody, "Using trust and risk in role-based access control policies," in *ACM Symposium on Access Control Models and Technologies (SACMAT)*, 2014.
- [128] L. Chen, J. Crampton, "Risk-aware role-based access control," in *International Workshop on Security and Trust Management, Cited by Springer*, 2012.

- [129] D.R. dos Santos, C.M. Westphall, C.B. Westphall, "Risk-based dynamic access control for a highly scalable cloud federation," in *IEEE Proceedings of the Seventh International Conference on Emerging Security Information, Systems and Technologies*, 2013.
- [130] D.R. dos Santos, C.M. Westphall, C.B. Westphall, "A dynamic risk-based access control architecture for cloud computing," in *IEEE Network Operations and Management Symposium (NOMS)*, 2014.
- [131] Daniel Ricardo dos Santos, Roberto Marinho, Gustavo Roecker Schmitt, "A framework and risk assessment approaches for risk-based access control in the cloud," *Elsevier Journal of Network and Computer Applications*, vol. 74, 2016.
- [132] Hany F. Atlam, Ahmed Alenezi, Robert J. Walters, Gary B. Wills, Joshua Daniel, "Developing an adaptive Risk-based access control model for the Internet of Things," in *IEEE International Conference on Internet of Things*, 2017.
- [133] Sadegh Dorri Nagoorani, Rasool Jalili, "TIRIAC: A trust-driven risk-aware access control framework for Grid environments," *Future Generation Computer Systems*, vol. 55, pp. 238-254, 2016.
- [134] Jason R.C. Nurse, Sadie Creese, David De Roure, "Security Risk Assessment in Internet of Things Systems," *IT Professional*, vol. 19, no. 5, pp. 20-26, 2017.
- [135] Giuseppe Petracca, Frank Capobianco, Christian Skalka, Trent Jaeger, "On Risk in Access Control Enforcement," in *Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies*, Indianapolis, Indiana, USA, 2017.
- [136] Divya Muthukumaran, Trent Jaeger, and Vinod Ganapathy, "Leveraging "Choice" to Automate Authorization Hook Placement.," in *ACM Conference on Computer and Communications Security*, 2012.
- [137] Sooel Son, Kathryn S. McKinley, and Vitaly Shmatikov, "Fix Me Up: Repairing Access-Control Bugs in Web Applications," in *Proceedings of the 20th Annual Network and Distributed System Security Symposium.*, 2013.
- [138] Salehie, M., Pasquale, L., Omoronyia, I., Ali, R., & Nuseibeh, B., "Requirements-driven adaptive security: Protecting variable assets at runtime," in *20th IEEE International Conference on Requirements Engineering Conference (RE)*, 2012.
- [139] Zhao, Z., Hu, H., Ahn, G. J., & Wu, R., "Risk-aware mitigation for MANET routing attacks.," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 2, pp. 250-260, 2012.
- [140] Vilém Novák, Irina Perfilieva, Antonín Dvůrák, "What is fuzzy modelling?," in *Insight into fuzzy modelling*, Wiley, 2016, pp. 3-11.
- [141] Tahmina Ahmed, Ravi Sandhu, Jaehong Park, "Classifying and Comparing Attribute-Based and Relationship-Based Access Control," in *7TH ACM Conference on Data and Application Security and Privacy*, 2017.

- [142] Lada A. Adamic, Bernardo A. Huberman, "Zipf's law and the Internet," *Glottometrics*, pp. 143-150, 2002.
- [143] Chao Wang, Jun Li, Li Zhang, Min Zhu, "The Drift Power Law Distribution of the Number of Requests in Video-on-Demand," in *Proceedings of the 36th Chinese Control Conference*, 2017.
- [144] S. Martin-Gutierrez, J. C. Losada, and R. M. Benito, "Recurrent Patterns of User Behavior in Different Electoral Campaigns: A Twitter Analysis of the Spanish General Elections of 2015 and 2016," *Complexity*, 2018.
- [145] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn and Sue Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," in *IMC '07 Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007.
- [146] Uthpala Subodhani, Premarathne, Ibrahim Khalil, Mohammed Atiquzzaman, "Location-dependent disclosure risk based decision support framework for persistent authentication in pervasive computing applications," *Computer Networks*, vol. 88, pp. 161-177, 2015.
- [147] Frans Ekman, Ari Keranen, Jouni Karvo, Jörg Ott, "Working day movement model," in *ACM Proceedings of the 1st ACM SIGMOBILE workshop on Mobility models*, 2008.
- [148] GANESH CHANDRASEKARAN, NING WANG, MASOUD HASSANPOUR, MINGWEI XU AND RAHIM TAFAZOLLI, "Mobility as a Service (MaaS): A D2D-Based Information Centric Network Architecture for Edge-Controlled Content Distribution," *IEEE Access*, vol. 6, 2018.
- [149] Ari Keränen, Jörg Ott, Teemu Kärkkäinen, "The ONE simulator for DTN protocol evaluation," in *ACM Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, 2009.
- [150] Heechang Shin, Vijayalakshmi Atluri, "Spatiotemporal Access Control Enforcement under Uncertain Location Estimates," in *IFIP Annual Conference on Data and Applications Security and Privacy*, 2009.
- [151] A. Prasad Sistla, Ouri Wolfson, Sam Chamberlain and Son Dao, "Querying the uncertain position of moving objects," in *Temporal Databases: Research and Practice*, Springer, 2006, pp. 310-337.
- [152] A. Hagberg, A. Kent, N. Lemons, and J. Neil, "Credential hopping in authentication graphs," in *International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, 2014.
- [153] Gio Wiederhold ; John McCarthy, "Arthur Samuel: Pioneer in Machine Learning," *IBM Journal of Research and Development*, vol. 36, no. 3, 1992.
- [154] T. M. Mitchell, "The Discipline of Machine Learning," Machine Learning Department, Carnegie Mellon University, 2006.
- [155] T. M. Mitchell, Machine Learning, McGraw-Hill Science/Engineering/Math, 1997.
- [156] Yang Xin, Lingshuanh Kong, Zhi Liu, Yuling Chen, Yanmiao Li, Hongliang Zhu, Mingcheng Gao, "Machine Learning and Deep Learning Methods for Cybersecurity," *IEEE Access*, vol. 6, pp. 35365-35381, 2018.

- [157] Yi Li, Kaiqi Xiong, Tommy Chin and Chengbin Hu, "A Machine Learning Framework for Domain Generation Algorithm-Based Malware Detection," *IEEE Access*, vol. 7, pp. 32765-32782, 2019.
- [158] Nadia Chaabouni, Mohamed Mosbah , Akka Zemmari, Cyrille Sauvignac, and Parvez Faruki, "Network Intrusion Detection for IoT Security Based on Learning Techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 03, pp. 2671-2701, 2019.
- [159] Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K., "Network anomaly detection: methods, systems and tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303-336, 2014.
- [160] Anna L. Buczak, and Erhan Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 02, pp. 1153-1176, 2016.
- [161] Antonia Nisioti, Alexios Mylonas, Paul D. Yoo and Vasilios Katos, "From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods," *EEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3369-3388, 2018.
- [162] Olivier Chapelle, Bernhard Schölkopf and Alexander Zien, *Semi-Supervised Learning*, The MIT Press, 2006.
- [163] Rana Aamir Raza Ashfaq, Xi-Zhao Wang, Joshua Zhexue Huang, Haider Abbas, Yu-Lin He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Information Sciences*, vol. 378, pp. 484-497, 2017.
- [164] Peining Shi, Zhiyong Zhang, Kim-Kwang & Raymond Choo , "Detecting Malicious Social Bots Based on Clickstream Sequences," *IEEE Access*, vol. 7, pp. 28855-28862, 2019.
- [165] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 2015.
- [166] Zhen Ni, Shuva Paul , "A Multistage Game in Smart Grid Security: A Reinforcement Learning Solution," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2684 - 2695, 2019.
- [167] Ying Chen, Shaowei Huang, Feng Liu, Zhisheng Wang, Xinwei Sun , "Evaluation of Reinforcement Learning-Based False Data Injection Attack to Automatic Voltage Control," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2158 - 2169, 2019.
- [168] Nguyen Cong Luong, Dinh Thai Hoang, Shimin Gong, Dusit Niyato, Ping Wang, Ying-Chang Liang, Dong In Kim, "Applications of Deep Reinforcement Learning in Communications and Networking: A Survey," *IEEE Communications Surveys & Tutorials*, vol. Eerly Access, 2019.
- [169] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861-874, 2006.
- [170] Ian H., Eibe Frank ,Mark A. Hall, Christopher J. Pal , *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Series in Data Management Systems, 2016.

- [171] C. C. Aggarwal, Data Mining, The Text Book, Springer, 2015.
- [172] M Balamurugan ; S Kannan, "Performance analysis of cart and C5.0 using sampling techniques," in *IEEE International Conference on Advances in Computer Applications (ICACA)*, 2016.
- [173] Sebastian Raschka, Vahid Mirjalili, Python Machine Learning, Machine Learning and deep learning with python, scikit-learn and TensorFlow, Packt, 2017.
- [174] Bernhard Schölkopf and Alexander J. Smola, Learning with Kernels, Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2002.
- [175] "Nearest-neighbour methods," in *Pattern Recognition and Machine Learning*, Springer, 2007, pp. 124-127.
- [176] DAVID W. AHA , DENNIS KIBLER, MARC K. ALBERT, "Instance-Based Learning Algorithms," *Springer Journal of Machine Learning*, vol. 6, pp. 37-66, 1991.
- [177] Charu C. Aggarwal, "Data Classification, Advanced Concepts," in *Data Mining, The textbook*, Springer, 2015, pp. 381-383.
- [178] Weiming Hu, Wei Hu and Steve Maybank, "AdaBoost-Based Algorithm for Network Intrusion Detection," *EEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* , vol. 28, no. 2, pp. 577 - 583, 2008.
- [179] Rajiv Punmiya, Sangho Choe , "Energy Theft Detection Using Gradient Boosting Theft Detector With Feature Engineering-Based Preprocessing," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2326 - 2329, 2019.
- [180] Sebastian Raschka and Vahid Mirjalili, "Combining Different Models for Ensemble Learning," in *Python Machine Learning*, Packt Publishing, 2017, pp. 219-233.
- [181] C. C. Aggarwal, "Neural Networks," in *Data Mining, The Textbook*, Springer, 2015, p. 327.
- [182] S. Haykin, Neural Networks And Learning Machines, Pearson, 2018.
- [183] Jesse Davis, Mark Goadrich, "The relationship between Precision-Recall and ROC curves," in *ICML '06 Proceedings of the 23rd international conference on Machine learning*, 2006.
- [184] Rafae Bhatti, Elisa Bertino, Arif Ghafoor, "A Trust-based Context-Aware Access Control Model for Web-Services," in *IEEE International Conference on Web Services (ICWS'04)* , 2004.
- [185] Sudip Chakraborty, Indrajit Ray, "TrustBAC - Integrating Trust Relationships into the RBAC Model for Access Control in Open Systems," in *IEEE SACMAT'06*, 2006.
- [186] Manachai Toahchoodee, Ramadan Abdunabi, Indrakshi Ray, and Indrajit Ray, "A Trust-Based Access Control Model for Pervasive Computing Applications," in *IFIP Annual Conference on Data and Applications Security and Privacy*, 2009.

- [187] Roshan K. Thomas, Ravi Sandhu, "Models, Protocols, and Architectures for Secure Pervasive Computing: Challenges and Research Directions," in *Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'04)* , 2004.
- [188] Pho Duc Giang, Le Xuan Hung, Sungyoung Lee, Young-Koo Lee and Heejo Lee, "A Flexible Trust-Based Access Control Mechanism for Security and Privacy Enhancement in Ubiquitous Systems," in *IEEE International Conference on Multimedia and Ubiquitous Engineering(MUE'07)*, 2007.
- [189] Abdallah Zoubir Ourad, Boutheyna Belgacem, and Khaled Salah, "Using Blockchain for IOT Access Control and Authentication Management," in *Third International Conference Held as Part of the Services Conference Federation, SCF 2018*, 2018.
- [190] OASIS, "eXtensible Access Control Markup Language," OASIS, 2013.