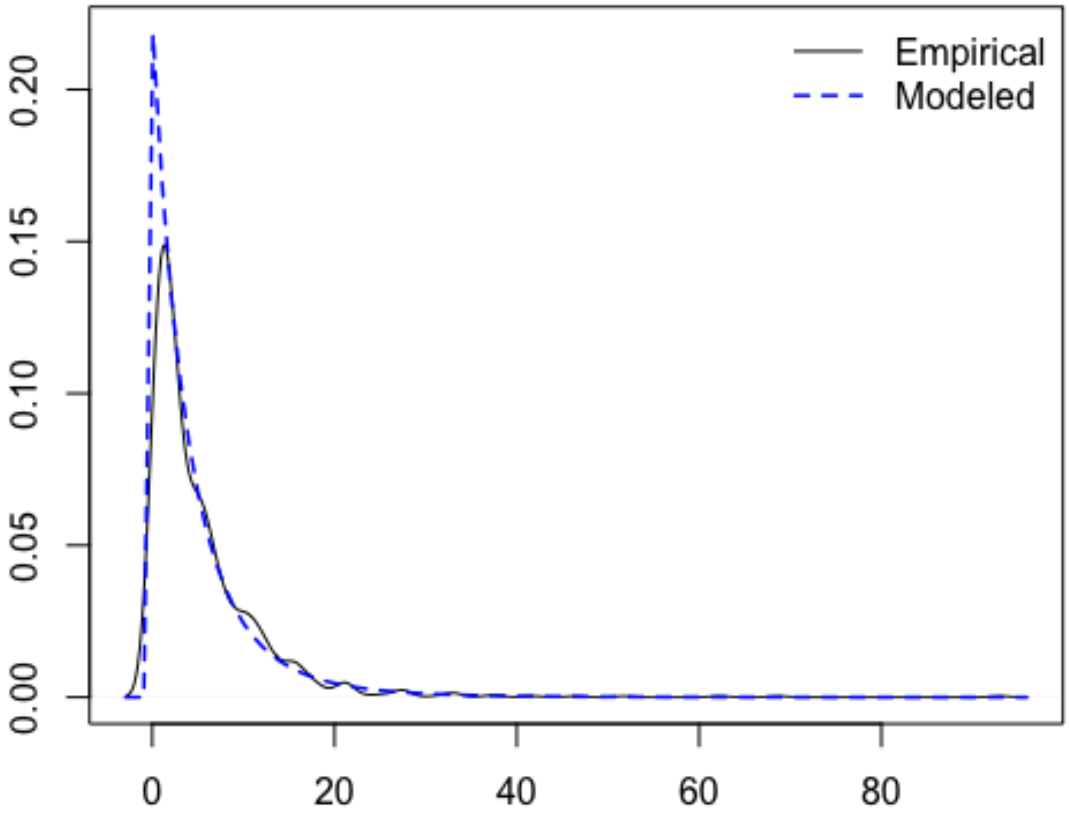


Anomaly Detection in Credit Card Transactions using Multivariate Generalized Pareto Distribution

Comparison in performance for supervised and unsupervised Machine Learning Algorithms

Master's thesis in Engineering Mathematics and Computational Science

KUBILAY MUAMELECI

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

Master's thesis 2022

Anomaly Detection in Credit Card Transactions using Multivariate Generalized Pareto Distribution

Comparison in performance for supervised and unsupervised Machine Learning Algorithms

KUBILAY MUAMELECI

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
Gothenburg, Sweden 2022

Anomaly Detection in Credit Card Transactions using Multivariate Generalized Pareto Distribution.
Comparison in performance for supervised and unsupervised Machine Learning Algorithms.
KUBILAY MUAMELECI

Supervisor: Prof. Holger Rootzén, Department of Mathematical Sciences
Examiner: Prof. Serik Sagitov, Department of Mathematical Sciences

Master's Thesis 2022
Department of Mathematical Sciences
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Kernel density *Generalized Pareto Distribution* plot of the excess anomaly scores from the $L - Supremum$ transformation in R.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2022

Abstract

There are billions of dollars that are lost to fraudulent credit card transactions every year. Many of these transactions are never noticed which causes a tremendous pressure on the economical system for the financial and credit institutions of interest. In addition to this, the usage of credit cards and thus e-business are in its arise, which together causes a threat in parallel with new developed data infringement methods. The research and progress within *Machine Learning* (ML) algorithms has been seen as an useful tool for the fraud investigators. However, there are still lacking robust frameworks which provides accurate and reliable methods within the field of ML:s. This thesis examines how the *Multivariate Generalized Pareto distribution* (MGPD) performs with regards to anomaly detection within a pre-processed data set consisting of credit card transactions in Europe for a month, compared to the *supervised* ML algorithm *Feedforward Fully Connected Neural Network* (FFCNN) and the two *unsupervised* ML algorithms *Isolation Forest* (IF) and *Support Vector Machine* (SVM), respectively. The pre-processing of the data set has been done a priori by means of *Principal Components Analysis* (PCA). The MGPD is fitted and simulated such that it has generators with independent *Gumbel* generators, whereas it is constructed in 3 dimensions consisting of standard exponentially transformed anomaly threshold excesses from the IF algorithm, *L2* and *L-Supremum* metrics. The comparison is mainly done by means of *Precision-Recall* (PR) curves and *Receiver Operating Characteristic* (ROC), *Area under ROC* (AUROC) and *Area under PR* curves (AUPRC), whereby most emphasis in the comparison has been put on the AUPRC value, due to the nature of the highly imbalanced data set. It is found that the MGPD outperforms both of the unsupervised algorithms; IF and SVM under the assumption of 0.2% anomalies in the training set. Moreover, it is slightly under performing the IF when assuming 1% anomalies in the training set. The supervised FFCNN performs best within all of the models, due to its supervised nature. Nevertheless, trained and tested with respect to the same data set, the MGPD significantly outperforms both of the unsupervised algorithms. The results from this thesis provides promising future research with respect to the MGPD within unsupervised anomaly detection.

Keywords: Multivariate Generalized Pareto, Support Vector Machine, Artificial Neural Network, Isolation Forest, Unsupervised, Supervised, Anomaly, Credit Card, Fraud, Machine Learning.

Acknowledgements

The work presented in this thesis is my final contribution towards the degree as Master of Science in Engineering Mathematics and Computational Science. The thesis has been initiated and carried out at the department of Mathematical Sciences at Chalmers University Of Technology.

Firstly, I would like to extend my most sincere and beloved appreciation to Prof. Holger Rootzén for taking me as his student, initiating this research and supervising me enthusiastically through the entire work. This would not be possible without him. In particular, I want to thank Holger for his support, encouraging ideas, the insightful conversations at his office and also for having faith in my efforts. I also want to thank Helga ólafsdottir for helping me with some of the simulations. Moreover, I feel highly privileged for having the opportunity before ending my studies, to being introduced into the world of Financial Risk and Extreme Values Statistics, under one of the most appreciated authors within this field, Prof. Holger Rootzén. Secondly, I would like to direct my gratitude towards to some of the professors at Chalmers University Of Technology who have been taking an active role towards developing my mathematical skills and interest throughout these five years. Therefore, I want to direct my gratitude to Prof. Patrik Albin for teaching me advanced topics in Stochastic Calculus and Processes, nevertheless, that his exams were absolutely the hardest ones. Moreover, I am grateful for obtaining the beautiful insights in Statistics by Prof. Serik Sagitov and also his support as my examiner in this thesis. Last but not least, Prof. Simone Calogero was the main character who introduced me into Financial Mathematics in general and its applications, which has enlightened my deep interest within this field.

Finally, I would like to express my dearest love to my wonderful parents Ethem, Serpil and my brother Mert. You have during my entire life, in your own particular way, encouraged and motivated me to perform and become my very best. Without your daily encouragement, unconditional love and support, this would have not been accomplished. I could have not be more happy and privileged for having you beside me.

Kubilay Muameleci, Gothenburg, May 2022

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AUROC	Area Under the ROC Curve
AUPRC	Area Under the Precision-Recall Curve
CDF	Cumulative Distribution Function
CI	Confidence Interval
CLT	Central Limit Theory
FDS	Fraud Detection System
FFCNN	Feedforward Fully Connected Neural Network
FPR	False Positive Rate
GEV	Generalized Extreme Value
GPD	Generalized Pareto Distribution
IF	Isolation Forest
KDE	Kernel Density Estimation
MGPD	Multivariate Generalized Pareto Distribution
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MRLP	Mean Residual Life Plot
MSE	Mean Squared Error
PCA	Principle Component Analysis
PDF	Probability Density Function
PoT	Peaks Over Threshold
PP	Probability-Probability Plot
QQ	Quantile-Quantile Plot
ROC	Receiver Operating Characteristic
SED	Standard Exponential Distribution
SVM	Support Vector Machine
TPR	True Positive Rate
TPSM	Threshold Parameter Stability Method
UGPD	Univariate Generalized Pareto Distribution

Nomenclature

Below is the nomenclature of the most frequently indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

i,j	Indices for number of instances of a sequence
t	Index for a time discrete step
μ	Input patterns
l	Left node of an Isolation Tree
r	Right node of an Isolation Tree

Sets

$\mathcal{A}$	Set of anomaly scores of L2, L-Supremum and Isolation Forest based on $\mathcal{T}_{Tr}$
$\mathcal{A}_{IF}$	Set of the anomaly scores with the origin from the Isolation Forest algorithm
$\mathcal{A}_{SVM}$	Set of the anomaly scores with the origin from the Support Vector Machine algorithm
$\mathcal{A}_{L2}$	Set of the anomaly scores with the origin from the L2-norm
$\mathcal{A}_{L-Sup}$	Set of the anomaly scores with the origin from the L-Supremum norm
$\mathcal{E}_{Trans_{Tr}}$	Set of SED transformed excesses of the training set
$\mathcal{E}_{Trans_{Test}}$	Set of SED transformed excesses of the test set
$\mathcal{T}$	Set of the total number of credit card transactions
$\mathcal{T}_{Tr}$	Set of training sequences of the credit card transactions
$\mathcal{T}_{Test}$	Set of test sequences of the credit card transactions

Parameters

γ	Shape parameter for the GPD
σ	Scale parameter for the GPD
u	Threshold parameter for the GPD
$\boldsymbol{u}$	Threshold vector for the MGPD
$\boldsymbol{Y}$	Excess vector for the MGPD
$\boldsymbol{U}$	Generator vector for the MGPD
T	Node of an Isolation Tree
$L(\cdot)$	Harmonic number
$c(\cdot)$	Average path-length
$k(\cdot, \cdot)$	Kernel function
$\sigma(\cdot)$	Softmax function
$g(\cdot)$	Activation function
$O(\cdot)$	Output function
$\boldsymbol{\beta}, \boldsymbol{\alpha}$	Lagrangian multipliers for the SVM
$\boldsymbol{w}$	Normal vector to the hyperplane
K	Amount of classes in the data set consisting of the credit card transactions
$\boldsymbol{\rho}$	Offset for the SVM algorithm
β	Location parameter for the MGPD
α	Scale parameter for the MGPD

Variables

X	Stochastic variable
w_{jk}	Weights between the inputs and the first hidden layer in FFCNN
θ_i	Thresholds for the FFCNN
Θ	Output threshold for the FFCNN
w_{mj}	Weights between the two hidden layers
w_m	Weight between the connection of the last hidden layer and the output-layer
V_j, V_m	Hidden layers in the FFCNN
ξ_i	Slack variables for the SVM algorithm

Contents

List of Figures

List of Tables

1
Introduction

This thesis examines how different *Machine Learning* (ML) algorithms, both *supervised* and *unsupervised*, performs against the *Multivariate Generalized Pareto Distribution* (MGPD) that will be fitted, in rigor of anomaly detection within credit card transactions. The comparison of the performance is done by means of statistical metrics, such as *Precision-Recall* (PR) curves, area under the *Receiver operating characteristic* (ROC), i.e., (AUROC) and area under the PR curve (AUPRC), whereas main emphasis is put on the AUPRC due to the highly imbalanced data set. Chapter 1 declares the aim of this thesis and its contribution for the financial institutions and overall its contribution for anomaly detection within other fields for the academy and different engineering principles. Chapter 2, the background, enlightens some of the fundamental properties and facts regarding *anomaly detection*, *credit card frauds* and Machine Learning. Furthermore, the theory presented in Chapter 3 possesses the required mathematical and technical aspects for the thesis - which enables the reader to have a clear understanding behind the different Machine Learning algorithms and the *Generalized Pareto Distribution* (GPD) and thus also the MPGD. Chapter 4, the method, presents the different frameworks and functions that has been deployed for the numerical computations and training of the data. This also includes model setup. Furthermore, Chapter 5 presents the results from the comparison of the various implementations and finally Chapter 6 present the conclusions of this thesis. Finally, a possible future research based on this thesis is provided.

1.1 Research aim

The aim for this thesis is to investigate how today's Machine Learning algorithms, both supervised and unsupervised, perform against the MGPD that will be fitted and simulated, by means of unsupervised anomaly detection within credit card transactions. Furthermore, it proposes a new fundamental tool for anomaly detection within other engineering principles. This, of course, enlightens the generalization of this research regarding the MGPD and its applications for different purposes within anomaly detection.

1.2 Contribution

The main contribution of this thesis is to give the financial and credit institutions a new model by means of the Multivariate Generalized Pareto Distribution for unsu-

pervised anomaly detection within credit card transactions. Moreover, this research enlightens also that the MGPD can be used within other fields when considering anomaly detection. The reason for introducing such a statistical model to the institutions is because, today's ML algorithms mostly functions as a black box. This means that, these institutions cannot employ these ML:s easily due to ethical aspects. In addition to this, ML algorithms require extremely high maintenance and expertise - which of course the latter also yields for the MGPD, however, the black box dilemma is avoided.

2
Background

For being able to have an intuitive understanding for how different Machine learning algorithms and statistical models, such as the GPD and MPGD, can be used for anomaly detection within credit card transactions and thus anomaly detection in general, one must address the issues and challenges within this area. For this purpose, one must know what a *credit card fraud* is, how it can be detected and how today's financial institutions are dealing with this unwanted phenomenon. Moreover, one must also define the meaning of a Machine learning algorithm and further what negative impacts fraud can lead to. Below sub-sections treat these topics and prepares the reader for the advanced mathematical concepts in the coming chapters. The novice reader might consider to jump directly to the results and omitting the theory - the latter is presented in Chapter 3.

2.1 Fraud

Fraud is an old phenomenon, which has been existing as long as the human being itself. When thinking about fraud, it might enlighten different scenarios for every person of interest, thus making the definition unclear. However, when considering a fraud - there are always one counterpart that looses something and the other one stealing a property that is not theirs. Fraud is a crime where the main purpose is to, by means of different techniques and methods, to overcome money or a property that does not belong to the *fraudster*. According to the Association of Certified Fraud Examiners (ACFE), fraud is defined as [1]:

> "The use of one's occupation for personal enrichment through the deliberate misuse or misapplication of the employing organization's resources or assets."

There are various types of fraud in today's society. Some of these can be referred to as *management frauds*, which is also known as a financial statement fraud, *tax fraud*, which can be carried out by a individual or an organization [2] - but also other frauds as for instance credit card frauds. The latter type of fraud is the one that will be considered throughout this thesis. However, ACFE divides typically fraud into two sub-categories as *external* and *internal* fraud [4]. Where the first one is for instance when an employee commits some sort of fraud against its employer, that is, its current working organization or business. Whereby the external fraud, is typically against an another company/organization or the government. Regardless

the origin of the fraud, one must have in mind that fraud can cause tremendous of harm for the source of trust and the economical system that today exhibit. As a consequence to prevent such fraudulent instances, extensive amount of money and resources are deployed - all this putting an enormous economical pressure for the governments and financial/insurance institutions. In addition to this, parallel to the increasing use of the *internet*, new and unique methods are being developed by the fraudsters to commit frauds through [3]. As the use of internet is strictly increasing, it opens up several new methods and techniques for the fraudster to infringement personal information and thus reach sensitive information.

2.2 Credit Card Fraud

Today, there are billions of dollars in losses due to fraudulent credit card transactions. In addition to this, the use of credit cards are strictly increasing in society whereas almost no physical cash is used. This situation is of course the utopia for the fraudsters that are committing these frauds. In parallel to this, at stated before, new techniques for committing these fraudulent credit card transactions are evolving [3].This thesis aim to focus on how these credit card frauds can be detected *a priori* and thus be prevented. A credit card fraud is basically a sort of banking fraud where the fraudster manages to obtain information regarding the cardholders physical credit card and thus commit frauds. Fraudulent credit cards transactions is defined as [5]:

> "The unauthorized use of an individual's confidential information to make purchases, or to remove funds from the user's account"

To have an understanding on how a credit card fraud might be executed - one must first know what a credit card is and its technical function. That is, not only its physical appearance and usability, but also the machinery behind the rectangular simple card that has been in the market for decades.

A credit card is a sort of payment card that is issued to the cardholder by a financial institution or a credit union, which enables the cardholder to pay a merchant for the goods of interest, but also other services, such as Forex transactions [6]. Furthermore, the card issuer usually creates an account which grants an amount of credit to the credit cardholder, by this, they can borrow money for a payment or as a cash advance. This implies that the credit card itself is connected to the issuer and hence when the cardholder is executing a payment, the transaction usually does not withdraw directly from the balance. Thus, the cardholder will complete the transaction in a later prescribed time than the actual purchase. Therefore, the cardholder and the transaction itself is exposed for a threat by means of unnoticed amount of money that can be withdrawn, before the bank notices this [7].

Credit card frauds may of course occur in various ways and forms [8]. As a few examples, one addresses *stolen card fraud*, *application fraud* and *cardholder-not-present fraud* (CNP). Whereby, the latter one is one of the most significant one appearing in

today's society. In particular, since the COVID-19 pandemic, it has been overtaking the field of credit card frauds [9], since ordering food and other supplies has risen significantly due to lock-downs. A stolen credit card fraud was one of the most common frauds committed, i.e., the cardholder physically lost its card and then the fraudster rapidly tries to spend as much money as possible. Moreover, one has an application of fraud. That is, when the fraudster tries to apply false personal information regarding the cardholder. However, those frauds are not that present anymore, since the websites usually provide some additional steps to be fulfilled before a purchase can be done, for instance, fingerprint or a password. The CNP is however very usual. This phenomenon can usually be observed in e-business, as stated above, have and is always increasing due to the popularity of online shopping goods and services. Here, the fraudster usually is dependent on the information that the credit card has - but not the physical credit card itself. Therefore, the owner might not notice anything strange going on with its balance sheet, unless they checks it. During this time, of course, the fraudster commits *abnormal* purchase activity with the credit card. The methods that fraudsters uses to obtain the card information differs. However, the most common ones is to use some sort of *software*. These softwares are usually called *malware*. To mention a few types of malwares; one has *spyware*, *ransomware* and *cryptoworms* [10]. These softwares are then implemented in some sort of *channel*. That is, for instance a website or more specifically into an advertisement in that website. Thus, regarding the spyware, where it is used by keyloggers in the background of yours device to monitor your online activity and thus extract the information you are typing on your keyboard. If you are unlucky, that might be some sensitive information in terms of important passwords. Furthermore, one also notices various types of *phishing e-mails*, that is, form of social engineering, where the fraudster imitates and acts as an important person via fake e-mail and attracts you to open the fraudsters mail. If and when that is done, then there is a high probability that you have downloaded some sort of malware to your computer that will try to extract sensitive information [10].

These mentioned types of frauds are, however, not the only ones. As stated above, the forms of committing frauds are continuously evolving [3]. Therefore, it is important to develop new techniques to overcome fraudsters. Moreover, it is important to exchange different tools for this purpose, but to have in mind that, it is not that easy. This come from the fact that, the fraudsters try to learn how the counterpart is combating and thus making new trials for data infringement. For a more general discussion on different types of fraud and properties around the topic, the interested reader can be referred to e.g. [11].

2.3 The impact of credit card frauds

The impact of credit card frauds are today tremendous and is causing billions of dollars every year in financial losses. The exact numbers of the total financial losses are not known due for confidentiality reasons. To give some numbers, the frauds involving credit cards worldwide are estimated to have a reached to a total of $27 billion in 2018 - with the prediction for this number to become $35 in five years and

$40 billion in 10 years [12]. Moreover, one must note that the numbers regarding the losses that are official is only the fraction of the losses that have been successfully detected. In addition to this, some frauds might be noticed too late or go unnoticed - which of course is not registered to the official numbers [13].

One aspect of the losses that is a consequences from the fraudulent cases is that, it is not only the merchants and the banks that are being affected. When such fraudulent instances appear in the long term, the financial institutions and insurance companies, will be forced to increase service fees in their organizations. For instance, if the cardholder uses a trading platform or has a bank-loan, due to the financial losses that is caused to the bank by fraudulent cases can imply that *interest rates* and *trading-fees* increases in price. As a consequence, the cardholder or the person that is paying interest for its loan may suffer from an overall increase in prices. In addition to this, the financial institutions may also obtain a bad reputation of their security towards frauds, causing insecurity for the service receiver i.e., the cardholder. This implies indirectly that the bank of interest can lose market share.

To give some more numbers on the impact of credit card frauds, one can refer to the Association for Payment Clearing Services (APACS). They have estimated on the amount of credit card losses in the United Kingdom between 1997-2010 to have grown from £122 million in 1997 to £440.3 million in 2010 [14]. Moreover, in 2012, the European Central Bank (ECB) reported that every €1 in €2.650 that were spent via credit and debit cards issued within SEPA (the European Union, Iceland, Liechtenstein, Monaco, Norway and Switzerland) was lost to fraud [15]. In particular, 60% of these frauds where of the nature of CNP:s - which has been defined above in section 2.2.

2.4 Today's credit card fraud detection systems

The efforts to prevent fraud within the area of credit cards are divided into two categories. These are, *fraud prevention* and *fraud detection* [16]. The difference between the two definitions is that, prevention is often referred to when the fraud is prevented *a priori*, on the other hand, fraud detection is defined as to be noticed *a posteriori*. Today, there are several methods regarding the prevention of the frauds. Two example of these is for instance *Personal Identification Number* (PIN) and *Card Verification Number* (CVN), where both of these two methods is number based i.e., pre-determined numbers that is a code to lock up access to the card. Financial institutions, such as banks and credit unions, use a combination of different methods for prevention purposes. That is, methods to both prevent frauds a priori and a posteriori. For this purpose, they use filters that are typically constructed by the *fraud investigators* - which are usually supported by data mining methods such as ML algorithms [17].

Financial institutions today are in the era of using machine learning algorithms within fraud detection. However, they are still in wide use of so called *expert driven*

detection [18]. That is, a group of investigators, usually with a mathematical background that sets up rule based fraud prevention system by means of very easy and interpretable decision making rules. This type of prevention system is of course easy to implement and to understand. However, in a dialogue with an employee at the ICA Banken which is working with the implementation of these prevention systems - it showed that it involves some problems. It was argued that the rules that are set-en up by the investigators might be biased and per definition differ what one investigator thinks of as a good rule versus the other one in the same group, leading to different opinions. In additions to this, frauds are often correlated in both time and space. Hence, the fraudster typically tries to commit frauds in the same market several times during a short time. In parallel to this, the experts, usually cannot think more than three dimensions, making it hard to realize fraudulent patterns. Moreover, with the evolving society with regards to the use of credit cards, there are thousands of credit card transactions occurring everyday - which makes it impossible for a human being to investigate every possible fraudulent instance. This is also confirmed by the worker at ICA Banken, without given any name due to confidentiality reasons.

2.5 Machine Learning

Machine learning is known as the practice of programming computers such that the programs can learn from data [19]. Today, ML makes a wide range of contribution within many scientific disciplines. Moreover, ML is today used for many applications within different fields. For instance, fraud detection, weather prediction and medical diagnosis. Fundamentally, ML aim is to learn from large data sets by statistical methods and thus be able to predict on new unseen data. Typically, one defines *training set* and *test set*. Where the training set is the data set on which the algorithm is trained on and the latter one in where the performance and accuracy in terms of statistical metrics are measured to be able to see how well it predicted the unseen data. Furthermore, ML is closely related with fields such as *pattern recognition* and statistics - but also with computer science, due to the fact that all of the fields has a huge contribution to the ML algorithms and their development [20].

Machine learning is typically divided into two categories; *supervised* and *unsupervised* [20]. In supervised learning the aim is to learn a mapping from the input data to an output, where the correct outputs is referred to be under supervision, that is, the output variable or so called response variable is labeled. This means that the true outcome of a series in observations is known a priori. On the other hand, in unsupervised learning, there is no supervisor, just unlabeled input data. The aim becomes to find regularities in the input data, learn from it during the training of the algorithm, and then test it on new unseen data [20]. However, and especially for credit card transactions, the availability of data is scarce - and unsupervised learning is needed and is an advanced topic. In addition to this, if one has a large data set with labels, one can still train the data without the labels and check the unsupervised algorithms performance towards the true labels.

2.6 Fraud Detection System within credit card transactions

As mentioned above in sub-section 2.4, today's fraud detection system is mainly expert driven. When considering a so called more *data driven* fraud detection system, one typically builds the system using one or an another kind of ML algorithms. The fraud detection system (FDS), consisting of ML:s, mainly relies on the analysis of huge sample of credit card transactions. By the use of ML algorithms, the load on the fraud investigators can be significantly eased, since ML algorithms that are build into the FDS:s have the ability to detect fraudulent patterns in higher dimensions, process large amount of data, predict new types of frauds and also adapt to distribution changes. There are also disadvantages of ML algorithms. To mention a few, the algorithms are often very mathematically advanced which requires deep knowledge. Moreover, for the training of the data to be reliable and thus predict correctly, the data sets must be large - which is relatively hard to find, due to confidentiality reasons. However, the main purpose of FDS is to provide an useful tool for the investigators by means of presenting them a small fraction of possible frauds that needs to be examined - and thus making the overall detection of frauds more precise and effective. This is since, the ML algorithms can extract a fraction of suspicious instances and thus only inspect a fraction of the data and not all of it. As an example, Jamie Damon, CEO of J.P Morgan, stated in 2019 that the bank could have saved up to $150 million if they would have used Machine Learning algorithms for detecting credit card frauds [55].

2.7 Anomalies and anomaly detection

Anomalies or *outliers* are substantial variations from the norm - where the norm is set to be the *normal* state [21]. Within various scientific research and engineering principles, the processes that are considered, usually follow some sort of behaviour and rules, which is implied by the nature and resulting in the state of a system [21]. These systems usually formalizes observable data - which upon one must formulate hypothesis regarding the underlying *distribution* of the data. When this is verified, then the observed state and its corresponding distribution that the system implies can be assumed to be the normal state. That is, any instance deviating from the normal distribution will be seen an *abnormal* instance. This phenomenon is of course expected, since the assumed distribution will never be identical to the real raw data and thus anomalies will must likely appear. The fundamental property and task of anomaly detection is to discover such instances in the observed data, that is, anomalies. However, there is no trivial definition and path to follow for the approach to find anomalies. In addition to this, there is no simple and unique definition to evaluate how similar two points or instances are to each other [22].

> "...there is an inherent fuzziness in the concept of outlier and any outlier score is an informative indicator than a precise measure."

Anomaly detection is a wide area. Its use is applicable in many engineering principles - even though this thesis aims to find anomalies within credit card transactions. To mention a few applications of anomaly detection, one can for instance consider malware detection. Then, one for example consider today's pattern and behaviour of the malwares and assume that the observed state is normal. Then, one can by some anomaly detection algorithm discover variations if new instances appears in the normal state i.e., on new data. That is, for instance if the malwares tries out new infringement methods to fool the detection filters in later prescribed days. Moreover, one also has the application of anomaly detection in bankruptcy prediction. That is, analyzing for instance earnings over time or other metrics such as increasing loans to loan-holders, to evaluate what the different risks are for bankruptcy. This yield of course also for other fields, such as within insurance companies. For more cases and examples, the interested reader is referred to [21].

When implementing and applying anomaly detection algorithms, there are often three possible cases that are examined and considered [21].

- *False Positives*: This occurs when the process continues to be normal but values that are not expected are observed. Usually as a consequence of *noise*.

- *False Negatives*: The process itself becomes abnormal, but the outcomes does not appear to be registered in the abnormal data. This is often due to the signal of abnormality being weaker compared to the noise in the observed system.

- *Correct Detection*: The amount of abnormalities in the observed data is exactly as the amount of real abnormal instances.

Without going into more details regarding the statistical metrics, which will be reviewed in Chapter 3, the key takeaway from this section is to know that; anomaly detection is a wide scientific and engineering field with many applications and there is no unique way of finding anomalies.

2.8 Issues and Challenges with FDS

There are many sources of errors and challanges when constructing a FDS. Some of the challenges are for instance: (I) *concept of drift*; (II) *skewed class distribution*, (III) *large amount of data* [18]. Where firstly, the concept of drift refers to the problem that the model when it has been trained on a fixed pattern and it drastically differs if the cardholder changes its buying behaviour. Secondly, the problem regarding skewed class distribution comes from the fact that typically only a very small fraction of the collected data instances are in fact abnormal and the wasp majority are normal instances. This phenomenon is also known as *class imbalance*. Of course, there exist today many algorithms that enables pre-processing of the raw data which can give a better performance for the ML algorithms and statistical models. However, one must have in mind that for every intervention the raw data is exploited to, the more lead error will there be and the models will most likely be

over-fitted for the particular set of data. Thirdly, even though large data sets is a good property to have within this research, without loss of generality, it implies complexity for the models that are developed. Therefore, as in this research, the data set is often reduced in dimension by means of *Principal Component Analysis* (PCA). Hence, when developing models for future prediction, one must always have in mind the different possible sources of errors that will be in the background. Therefore, as stated above in section 2.6, the FDS:s should be considered as a powerful tool for the fraud investigators, and not as a substitute.

3

Theory

This chapter will describe the theory behind the mathematical models that are considered in this thesis. It prepares the reader for the methods that are used in the next chapter. First, the *Univariate Generalized Pareto Distribution* (UGPD) and the *Multivariate Generalized Pareto Distribution* (MGPD) are introduced. Second, the unsupervised machine learning algorithms: *Support Vector Machine* (SVM) and *Isolation Forest* (IF) and the supervised machine learning algorithm, *Feedforward Fully Connected Neural Network* (FFCNN) are presented. Third, the statistical metrics that will be employed for performance comparison between models are shown.

3.1 Univariate Generalized Pareto Distribution

The Pareto distribution (PD) is named after the Italian economist, civil engineer and sociologist Vilfredo Pareto [23]. The PD was first introduced in 1906 by Pareto itself. The PD is a power-law probability distribution. The use of the PD appears in various fields of engineering and scientific principles, such as financial risk, quality control and geophysics [24]. However, the Univariate Generalized Pareto Distribution UGPD or the Generalized Pareto Distribution GPD in general was introduced in 1975 by Pickands. The GPD is a family of continuous probability distributions and often used to model tails of an another distribution i.e., extreme events over some threshold of the observed data.

Theorem 3.1.1 (Univariate Generalized Pareto Distribution)**.** *Let $X_1, X_2, ..., X_n$ be a sequence of independent and identically distributed random variables with common distribution function, let*

$$M_n = max\{X_1, ..., X_n\}, \tag{3.1}$$

be the maximum of the variables $X_1, \ldots, X_n$. Further, denote an arbitrary variable in the above sequence by X. Moreover, suppose it holds that for some sequence of constants $\{a_n > 0\}$ and $\{b_n\}$ such that

$$\mathbb{P}\Big\{\frac{M_n - b_n}{a_n} \leq z\Big\} \to G(z) \text{ as } n \to \infty, \tag{3.2}$$

for a non-degenerate distribution function G. Then G has a Generalized Extreme Value (GEV) distribution function

$$G(z) = exp\Big\{-\Big[1 + \gamma\Big(\frac{z-\mu}{\sigma}\Big)\Big]^{-\frac{1}{\gamma}}\Big\}, \tag{3.3}$$

for some $\mu \in (-\infty, \infty)$, $\sigma>0$ *and* $\gamma \in (-\infty, \infty)$ *and for* $\{z : 1+\gamma(z-\mu)/\sigma\}$. *Then the distribution function of* $(X-u)$, *conditional on* $X > u$, *converges to a GPD function*

$$H(x) = 1 - \left(1 + \frac{\gamma x}{\tilde{\sigma}}\right)^{-\frac{1}{\gamma}}, \tag{3.4}$$

for some $\tilde{\sigma} > 0$, $\gamma \in (-\infty, \infty)$ *and for* $\{x : x > 0, (1 + \frac{\gamma x}{\tilde{\sigma}} > 0)\}$ γ *for (3.3) and (3.4) are same.*

Note that, (3.4) is the cumulative distribution function (CDF) of the GPD. A full proof is given e.g in [25] or see Appendix A.1. The probability density function (PDF) of the GPD, is obtained by taking the derivative $H(x)$ with respect to x, and is

$$h(x) = \frac{\mathrm{d}x}{\mathrm{d}x} H(x) = \frac{1}{\sigma}\left(1 + \frac{\gamma x}{\sigma}\right)^{-\frac{1}{\gamma}-1}. \tag{3.5}$$

Remark 1. *For* $\gamma < 0$ *the GPD has left-endpoint* 0 *and right-endpoint* $\frac{\sigma}{|\gamma|}$. *For* $\gamma \geq 0$ *the GPD has left-endpoint* 0 *and* ∞ *for the right. For the special case* $\gamma = 0$ *one recovers to the memoryless exponential distribution with scale parameter* $\tilde{\sigma}$.

3.2 Univariate Peaks Over Threshold (PoT) method

What is the purpose of introducing such a model as the GPD within anomaly detection, one might wonder? The answer for this question relies on *Extreme Value Statistics* (EVS). For this purpose, the *Peaks over Threshold* (PoT) method was introduced by Smith in 1984 [26].

Suppose that one has a sequence of measurements $\{y_1, ..., y_n\}$ and a high threshold u. Then the threshold excesses are defined as $x_j = y_j - u$ for $y > u$, and one only consider the positive excesses and models it with a GPD.

Remark 2. *One has to make the correct choice of the threshold,* u. *That is, one must account for the trade-off between the bias and variance between a high respectively low threshold choice. Too low threshold choice will must likely violate the asymptotic basis of the GPD model which will as a consequence imply a bias. On the other-hand, a too high threshold will lead to few instances of exceedances for which is needed to fit the GPD model, hence, it will imply a higher variance in the final model that might not be reliable.*

Remark 3. *There are various methods for choice of threshold modeling. However, the majority are automatic. The interested reader is referred to [25].*

3.3 Multivariate Generalized Pareto Distribution

In this sub-section, the theory behind the Multivariate Generalized Pareto Distribution MGPD is introduced. As proposed in section 3.1 and 3.2 , the UGPD and the PoT of the univariate case has been widely used within different engineering principles and applications, such as for instance estimating the *Value at Risk* (VaR) and the *Expected Shortfall* (ES) in financial engineering [27]. However, often, as proposed in real world applications, the outcome of different events in nature, for instance flooding or financial engineering, are not only dependent on one dike regarding flooding or one financial instrument. They depend on several variables. For example, assume that one holds S shares of a derivative on the underlying asset A, then the price fluctuations will most like depend on the asset but also the stock index, and depending in which segment, also on other aspects such as the price of the oil, electricity or interest rates. In this context, the MGPD can be used. Today, the theory regarding the applications of the MGPD is scarce [28], there are some contributions out there, such as [29] - however, the majority does not use these as statistical models [28]. This thesis hence is one of the first contributions for unsupervised anomaly detection with the MGPD, in particular, within highly imbalanced data.

3.4 Multivariate Peaks Over Threshold (PoT) method

The Multivariate Generalized Pareto Distribution PoT method was first introduced in [29][32][33]. The intuitive reasoning for the MGPD PoT method is the same as for the univariate case. That is, one want to model the excesses of some high level thresholds u for some random vector $\mathbf{Y}$. However, in the multivariate case, one defines a vector of thresholds $\boldsymbol{u}$,

$$\mathbf{u} = (u_1, ..., u_d), \tag{3.6}$$

from a random vector

$$\mathbf{Y} = (Y_1, ..., Y_d), \tag{3.7}$$

and obtains excesses defined as

$$\mathbf{X} = \mathbf{Y} - \mathbf{u} = (Y_1 - u_1, ..., Y_d - u_d), \tag{3.8}$$

where d is the dimension of the considered MPGD. For instance, $d = 1$ one recovers the univariate PoT approach. If at least one random observation Y_i in (3.8) is $Y_i > u_i$, i.e., larger than its threshold, then $\mathbf{X}$ is considered as positive [30].

Under general conditions, the joint distribution of the positive excess vectors, defined above, is asymptotically a GPD as the thresholds $u_i \to \infty$. More on this in [29].

Next, some basic properties of the MPGD. Without loss of any generality, we assume that $d = 3$ and recovers the Three Dimensional Generalized Pareto distribution, abbreviated as TGPD. However, note that it holds for any dimension d. In [31] there

are four representations that are developed for the density of MGPD. However, here, the so called U-representation will be used see. Further, the marginals of the distributions here are assumed to be standard exponential distributions [see Section 3 in Kiriliouk et al., 2019] [28].

For the thresholds u_i for $i = 1, \ldots, d$ where d is referred to the dimension of the three dimensional generalized multivariate distribution, u will operate component-wise on the $\boldsymbol{Y}$. Denote a 3-dimensional random vector $\mathbf{u}$ which has finite first moment of its maximum exponential, $\mathbb{E}[e^{max(\mathbf{U})}] < \infty$, which will be called the *generator*, and where we define $max(\mathbf{U})$ by

$$max(\mathbf{U}) = max\{(U_1, U_2, U_3)\}. \tag{3.9}$$

This should not to be confused with the threshold vector, $\mathbf{u}$. Further, assume that this random vector admits a pdf by $f_{\mathbf{U}}$. Moreover, denote the distribution and density functions of U_i for $i = 1, 2, 3$ as F_i and f_i.The density function of the three dimensional MGPD with *Standard Exponential* margins generated by $\mathbf{U}$, are given by

$$h_{\mathbf{U}}(\mathbf{x}) = \frac{1_{\{\mathbf{x} \not\leq 0\}}}{\mathbb{E}[e^{max(\mathbf{U})}]} \int_0^\infty f_{\mathbf{U}}(\mathbf{x} + log(t))dt, \tag{3.10}$$

where the indicator function

$$1_{\{\mathbf{x} \not\leq 0\}} = \begin{cases} 1, & \text{if at least one component of } \mathbf{x} \text{ are positive.} \\ 0, & \text{otherwise.} \end{cases} \tag{3.11}$$

and where we for $\mathbf{x} = (x_1, x_2, x_3)$ define $\mathbf{x} + log(t) = (x_1 + log(t), x_2 + log(t), x_3 + log(t))$. It is stressed that, different distributions of the random vector $\mathbf{U}$ will imply different MGPD models [30].

Moreover, if $U = (U_1, U_2, U_3)$ with U_i, for $i = 1, 2, 3$ independent and distributed according to a *Gumbel* distribution with parameters α_i and β_i. The marginal density f_i is given by

$$f_i(x_i) = \alpha_i e^{-\alpha_i(x_i - \beta_i)} e^{-e^{-\alpha_i}(x_i - \beta_i)}, \tag{3.12}$$

where $\alpha_i > \mathbf{1}$ since $\mathbb{E}[e^{max\boldsymbol{U}}] < \infty$ and $\beta_i \in \mathbb{R}$ for $i = 1, 2, 3$. Moreover, to ensure identifiability, the first component of the location parameter β_1 is set to zero, $\beta_1 = 0$ [30].

3.5 Isolation Forest (IF)

The *Isolation Forest* (IF) or *iForest* is an unsupervised ML algorithm which was created in 2007 by Fei Tony Lui [34]. The method builds an ensemble of *iTrees* for a given data set where anomalous instances are those which have short total *path lengths* h in the iTrees. The general basis for anomaly detection using IF takes advantage of two fundamental characteristics of anomalies: (I) Anomalies are minority in the data set; (II) They have values deviating from normal instances. The proposed method requires the choice of two variables, number of iTrees t to construct

for each sub sample and the sub-samplings ψ [34].

The training stage, i.e., building the iTrees, is done by a random and recursive partitioning of the training data without replacement and by using sub-samples ψ of the training set until all the instances are isolated to an *External Node* in the iTree. The partitioning is done by randomly selecting a feature q (column) from the sampled training set and then randomly selecting a split value p between the maximum and minimum selected feature [34] as

$$\min(q) < p < \max(q). \tag{3.13}$$

Depending on the split value p, one divides the values in the feature to the left or right in an *Internal Node* in the iTree, which in turn, is a *Proper Binary Tree*, as

$$T_l := \{T' : q \in Q, q < p\} \text{ and } T_r := \{T' : q \in Q, q \geq p\}, \tag{3.14}$$

where Q is the set of features from the sampled training set T'. The outcome of the partitioning will imply that anomalous instances will have shorter total path length $h(x)$ in the iTree structure. Thus, when the *iForest* consisting of random iTrees collectively generates shorter path lengths for some instances, those will be regarded as anomalous. The reason is that, normal instances will in general require more partitions to be isolated, resulting in a longer path length in the iTree structure. The number of partitions required to isolate an instance is the same as the path length from the root node to a terminating node (External Node), see Figure 3.1. Moreover, the *anomaly scores* for the instances are derived from the mean path length $\bar{h}(x)$ and average path length $c(\psi)$.

When the iTree is fully grown, each instance is isolated to an external node, see Figure 3.1, in which case the number of external nodes is ψ and the number of internal nodes is $\psi - 1$; the total number of nodes of an iTree is $2\psi - 1$.

The testing stage passes each test instance through all the iTrees in the iForest from the training stage to obtain an anomaly score for each instance. The anomaly score A_{score} is obtained by the mean path length $\bar{h}(x)$ and computing the average path lengths over the number of iTrees built. A single path length $h(x)$ is derived by counting the number of edges e from the root node to an external node as instance x traverses through an iTree, see Figure 3.1 [34].

Since external node termination in an iTree equals unsuccessful search in a *Binary Search Tree*, the average path length is

$$c(\psi) = 2L(\psi - 1) - \frac{2(\psi - 1)}{n}, \tag{3.15}$$

where $L(i) \approx ln(i) + 0.5772156649$ is the harmonic number, 0.5772156649 is known as the Euler's constant, ψ is the sample size and n is the total number of test instances, see [34].

The average path length $c(\psi)$ can be considered as a normalization entity. It is the average path length to reach an arbitrary node in the iTree whereas the mean path length $\bar{h}(x)$ is the mean path length to reach an external node in the iTree. The anomaly scores A_{score} for each instance are computed by

$$A_{score}(x, \psi) = 2^{-\frac{\bar{h}(x)}{c(\psi)}}. \tag{3.16}$$

Remark 4. *One notes that:*

- *As* $\bar{h}(x) \to \psi - 1$, $A_{score}(x, \psi) \to 0$.
- *As* $\bar{h}(x) \to c(\psi)$, $A_{score}(x, \psi) \to \frac{1}{2}$.
- *As* $\bar{h}(x) \to 0$, $A_{score}(x, \psi) \to 1$.

By (3.16), when $A_{score}(x, \psi) \approx 1$, *the instance* x *is most likely to be an anomaly. The other two results follows analogously. For* $A_{score}(x, \psi) \approx 0$, *the instance* x *is most likely not an anomaly and for* $A_{score}(x, \psi) \approx \frac{1}{2}$ *the instances are most likely to be regarded as a normal [34].*

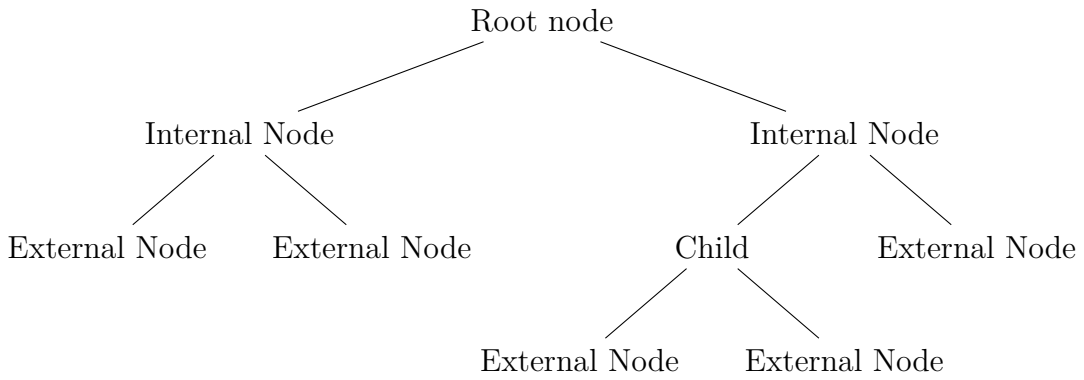


Figure 3.1: Proper Binary Tree (PBT) with its corresponding nodes.

3.6 Support Vector Machine (SVM)

Originally, the Support Vector Machine SVM was developed as a *supervised* ML algorithm by the Russian mathematician Vladimir Vapnik at the company AT&T Bell Technologies in 1995 [36].

The *unsupervised* SVM was introduced by Shölkopf et.al in 2001 [37]. This algorithm does not attempt to estimate any probability density of the underlying data. The unsupervised SVM estimates a binary function f that is supposed to capture regions in the input space $\mathcal{I}$ where the probability density lives i.e., its support. This function will produce a region $\mathcal{R}$ in the feature space $\mathcal{F}$ where most of the data are.

Consider a training set $x_i \in \mathcal{T}_{Tr}$ with unknown underlying probability distribution $\mathcal{P}$ for $i = 1, ..., n$, $x_i \in \mathbb{R}^d$ where $n \in \mathbb{N}$ is the number of observations. During the test stage, one checks if a test observation x_i is distributed according to $\mathcal{P}$ or not. This is done by determining a region $\mathcal{R}$ of the input space $\mathcal{I}$ where the probability of a test observation drawn from $\mathcal{P}$ lies outside of $\mathcal{R}$ is bounded by some a priori specified value $\nu \in (0, 1)$. To determine the region, a decision function f needs to be estimated. This will imply that test observations falling outside the region will have negative values as

$$f(\mathbf{x}) > 0 \;\text{ if }\; \mathbf{x} \in \mathcal{R} \;\text{ and }\; f(\mathbf{x}) < 0 \;\text{ if }\; \mathbf{x} \notin \mathcal{R}. \tag{3.17}$$

Using a non-linear function Φ: $\mathcal{I} \to \mathcal{F}$ the training vectors $\mathbf{x}$ are mapped via a *kernel* function

$$\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = k(\mathbf{x}_i, \mathbf{x}_j), \tag{3.18}$$

from the input space $\mathcal{I}$ to a higher dimensional feature space $\mathcal{F}$, where $k(\mathbf{x}_i, \mathbf{x}_j)$ is chosen to be the *Gaussian Radial Basis* function given by

$$k_\sigma(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}. \tag{3.19}$$

In this new space, the training vectors now follow an underlying distribution $\mathcal{P}'$ and one wants to determine a region $\mathcal{R}'$ in the feature space $\mathcal{F}$ of the probability distribution for the training data. Thus, $\mathcal{R}'$ will be the region where most of the training points will be within. The origin in the feature space is assumed to be where the anomalous observations will be [37]. To separate the anomalous instances, a hyperplane is introduced. The hyperplane will create the maximum margin from the origin. The maximum margin is found by solving the primal *Quadratic Optimization Problem*

$$\begin{cases} \min\limits_{\boldsymbol{w} \in F, \boldsymbol{\xi} \in \mathbb{R}^n, \rho \in \mathbb{R}} & \frac{1}{2}||\boldsymbol{w}||^2 + \frac{1}{\nu n}\sum\limits_{i=1}^{n} \xi_i - \rho \\ \text{subject to} & \langle \boldsymbol{w}, \Phi(\mathbf{x}_i) \rangle \geq \rho - \xi_i, \qquad \xi \geq 0, \end{cases} \tag{3.20}$$

where ξ_i are slack variables which penalize the objective function to allow some of the points to be on the wrong side of the hyperplane, ρ is the offset, $\boldsymbol{w}$ is the weight

vector and the parameter $\nu \in (0, 1]$ controls the trade of between maximizing the distance from the origin and containing most of the data in the region which was created by the hyperplane [37]. Hence, if ρ and $\mathbf{w}$ solves the primal problem (3.20), then the decision function f is given by

$$f(\mathbf{x}) = sgn(\langle w, \Phi(\mathbf{x})\rangle - \rho). \tag{3.21}$$

Thus, test observations mapped into the feature space that does not lie within $\mathcal{R}'$ will have negative values by (3.21) and vice versa. However, due to the high dimensional of $\boldsymbol{w}$ in the primal problem, one considers the easier *Dual problem*. Introducing Lagrangian multipliers α_i, $\beta_i \geq 0$ one obtains the Lagrangian, denoted by L as

$$\begin{aligned} L(w, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}||\boldsymbol{w}||^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \\ -\sum_{i=1}^{n}\alpha_i(\langle \boldsymbol{w}, \Phi(\mathbf{x}_i)\rangle - \rho + \xi_i) - \sum_{i=1}^{n}\boldsymbol{\beta}_i\xi_i. \end{aligned} \tag{3.22}$$

Setting the primal derivatives wrt. $\bar{D} = \{w, \boldsymbol{\xi}, \rho\}$ to be equal to zero, one obtains

$$w = \sum_{i=1}^{n}\alpha_i\Phi(\mathbf{x}_i), \tag{3.23}$$

and

$$\alpha_i = \frac{1}{\nu n} - \boldsymbol{\beta}_i \leq \frac{1}{\nu n},\ \sum_{i}^{n}\alpha_i = 1. \tag{3.24}$$

The dual problem is obtained by injecting (3.23) and (3.24) into the Lagrangian problem given in (3.22). This gives

$$\begin{cases} \min\limits_{\boldsymbol{\alpha}} & \frac{1}{2}\sum\limits_{i,j=1}^{n}\alpha_i\alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} & 0 \leq \alpha_i \leq \frac{1}{\nu n}, \sum\limits_{i}^{n}\alpha_i = 1. \end{cases} \tag{3.25}$$

Solving the dual problem with the kernel given in (3.19) the decision function becomes

$$f(\mathbf{x}) = sgn(\sum_{i=1}^{n}\alpha_i k(\mathbf{x}_i, \mathbf{x}) - \rho). \tag{3.26}$$

Remark 5. *The hyperplane in feature space $\mathcal{F}$ becomes non-linear in the input space $\mathcal{I}$.*

3.7 Feedforward Fully Connected Neural Network (FFCNN)

The *Feedforward Fully Connected Neural Network* (FFCNN) is a *supervised* ML algorithm. The FFCNN is a subset of *Artificial Neural Networks* and which its applications are strictly increasing within different engineering principles. The main characteristic of the FFCNN is that the information in the learning process are only forwarded in the network [38].

The property of the FFCNN is that, for some inputs/observations and *bias* or threshold and by random uniformly initializing the *weights*, pass it forward in the net through *activation functions*, g, and then forward it to the last *output layer* which in turn also has an activation function that makes the final decision on which class a particular observation belongs to [38].

The supervised nature of the FFCNN requires that the data set must be labeled a prior the training process. For understanding the FFCNN architecture, one considers an *input layer* which receives the training observations. The dimension of the input layer will be equal to the dimension of your training set. Further, the input layer is connected via weights to the *hidden layer* (there can be several hidden layers) which consists of *McCulloch-Pitts* neurons and have pre-defined activation functions. This processes is repeated until the final layer i.e., the output layer is reached and the inference can be made. The outputs of the output layer are *posterior probabilities* if one considers the *softmax* activation function in the output layer [38].

The setup for the FFCNN in this thesis will consist of two hidden layers. The first hidden layer will have 10 neurons and the second one 2 neurons. The reason for the choice of two neurons in the second hidden layer is since we have a binary classification problem. The hidden layers in the first one is set to ten by default in various applications, to avoid the over-fitting issue [38]. Hence, the FFCNN will have the following architecture, see Figure 3.2.

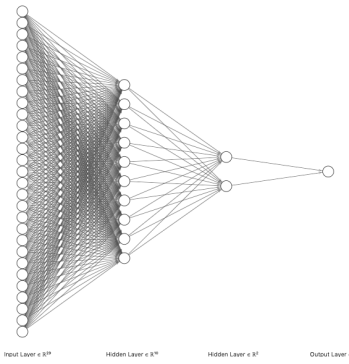


Figure 3.2: Feedforward Fully Connected Neural Network architecture. The neural network have 29 inputs due to the dimension of the input data. The first hidden layer has 10 neurons whereas the second one has 2 neurons.

The activation function for the first and second hidden layers is the popular *Rectified Linear Unit* (ReLU) function. The ReLU function is defined as follows

$$f(x) = x^+ = \max(0, x), \tag{3.27}$$

where x is an input to a neuron.

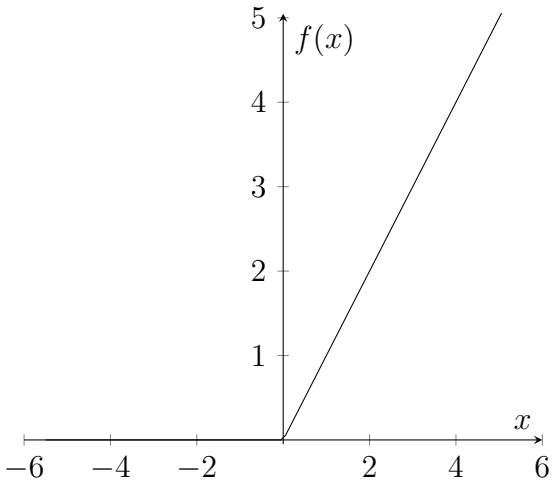


Figure 3.3: Rectified Linear Unit (ReLU) function

Hence, $f(x)$ returns the positive value of its argument from the neuron before.

For the output-layer, the *softmax* $\sigma(\cdot)$ activation function will be used. The softmax function $\sigma : \mathbb{R}^K \to (0,1)^K$, $K > 1$, is given by

$$\sigma(b(\mathbf{x}))_i = \frac{e^{b_i(x)}}{\sum_{j=1}^{K} e^{b_j(x)}}, \qquad \text{for } i = 1, .., N \tag{3.28}$$

where $K = 2$ is the number of classes for the classification problem. The term in the numerator of (3.28) is applied to each *local field* b_i output. If the input b_i is negative, the result from the term in the numerator will be small and it will be large if the input is positive and large. The term in the denominator is the normalization quantity. It assures that the outputs from the softmax function will be in the range $0 \leq \sigma(b(x))_i \leq 1$. Further, the sum of the outputs sum to unity as $\sum_{j=1}^{K} \sigma(b(x))_i = 1$, which constitutes a probability. Hence, the softmax function will give outputs in terms of probabilities where those probabilities that are $\sigma((b(x))_i \approx 1$ can be considered that the FFCNN is quiet certain that the input x_i belongs to class i in terms of targets/class labels, say class 1 i.e., a fraudulent credit card transaction [38].

The below plot shows that the softmax function give outputs in the range $[0, 1]$.

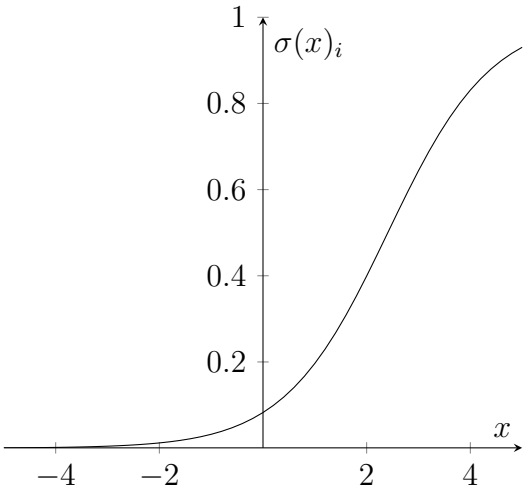


Figure 3.4: Softmax output-layer activation function

Consider a sequence of training samples that are fed to the input layer, given by the vector

$$\mathbf{x}^{(\mu)} = \begin{bmatrix} x_1^{(1)} \\ x_2^{(2)} \\ \vdots \\ x_n^{(p)} \end{bmatrix}^T , \tag{3.29}$$

where $n = 29$ denotes the dimension of the data. These input terminals are referred to the leftmost part of Figure 3.2 and where $\mu = 1, ..., p$ denotes the different patterns that an instance passes through the neural network via the different weight connections between the input layer until termination at the output layer. Denote the first and second hidden layers V_j and V_m respectively. Moreover, each of these two hidden layers will have a ReLU activation function given in (3.27). Hence, each of the two hidden layers will compute

$$V_j = g(b_j), \; b_j = \sum_k w_{jk} x_k - \theta_j, \tag{3.30}$$

$$V_m = g(b_m), \; b_m = \sum_j w_{mj} V_j - \theta_m, \tag{3.31}$$

where g is the activation function, θ is the threshold for the hidden neurons and b is the local field. The variables w_{jk} and w_{mj} are the weights. The indices for w_{mj}, can interpreted such that index m is the neuron that makes the computation and index j labels all neurons that connect to neuron m. The other one follows analogously. In particular, w_{jk} are the weights between the input and the first hidden layer and w_{mj} are the weights between the two hidden layers, see Figure 3.2. Thus, the classification problem is to approximate the class label vector $\mathbf{t}$ i.e., the list of corresponding labels or target values, with the output function $O(\mathbf{x})$. Hence, one considers the matrix

$$[\mathbf{x}^{(\mu)}, \boldsymbol{t}^{(\mu)}], \qquad \text{for } \mu = 1, ..., p \tag{3.32}$$

where the targets $t_i^{(\mu)}$

$$\boldsymbol{t}^{(\mu)} = \begin{bmatrix} t_1^{(1)} \\ t_2^{(2)} \\ \vdots \\ t_N^{(p)} \end{bmatrix}, \tag{3.33}$$

are the real classes labels for a given data set. Hence, the training samples $x_i^{(\mu)}$ are trained against their target $t_i^{(\mu)}$ and the output layer will compute

$$O(\mathbf{x}) = \sum_m w_m g\Big(\sum_j w_{mj} g\Big(\sum_k w_{jk} x_k - \theta_j \Big) - \theta_m \Big) - \Theta, \tag{3.34}$$

for each observation, where w_m and Θ is the weight respectively the threshold between the connection of the last hidden and output layer. In particular, (3.34) is the local field that (3.28) is applied on to obtain the classification probabilities [38].

3.8 Statistical Metrics

To compare the models that have been presented in this section, one needs to use statistical metrics that can be applied to all of the models. The metrics that will be used here are the *recall*, *precision*, Area under the *Receiver Operating Characteristics* Curve (AUROC) and *Area under the Precision-Recall* curve (AUPRC) metrics. However, both of the curves and the metrics are obtained through the *confusion matrix* [40], see Table 3.1. The AUROC metric is derived from the Receiver Operating Characteristic curve, or ROC curve. The ROC curve itself, is obtained by plotting the *true positive rate* (TPR) against the *false positive rate* (FPR) for some threshold setting [41]. The ROC curve is shown below

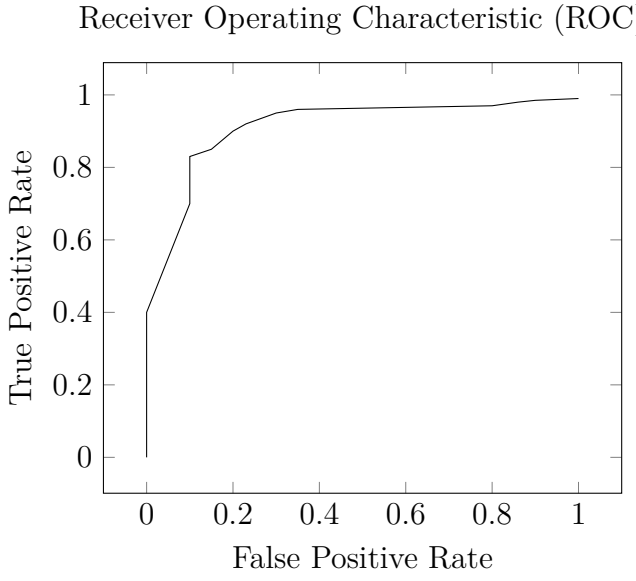


Figure 3.5: Receiver Operating Characteristic Curve

Where the TPR and FPR are defined as

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}, \tag{3.35}$$

and

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}. \tag{3.36}$$

The AUROC is thus the area under the ROC shown above. The value of AUROC ranges between [0,1] where 1 is referred to be the optimal prediction performance of the model [41].

Standard methods to compare performance of binary classification problems include positive and true negative rates and receiver operating characteristics curves [30]. However, when dealing with highly imbalanced data sets, such as anomalous credit card transactions, these mentioned methods can be less informative. Saito and Rehmsmeier [39], argued that Precision-Recall (PR) curves are more informative when working with imbalanced data sets. Hence, one considers the PR curve, shown below

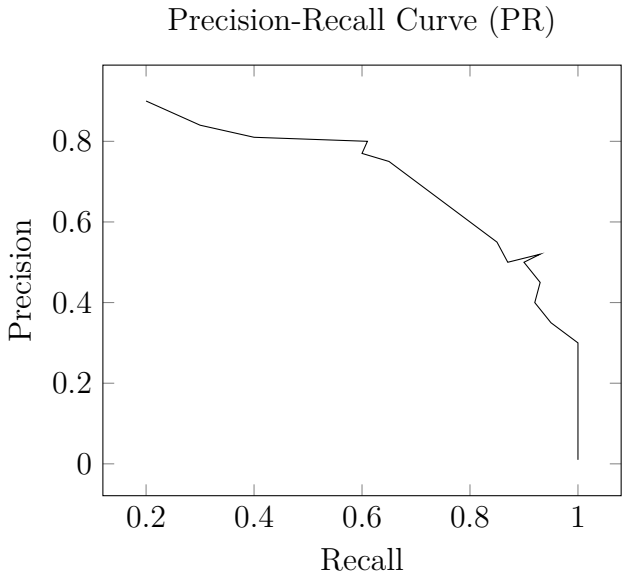


Figure 3.6: Precision-Recall Curve

and the AUPRC is the area under the PR curve. This area is usually approximated by the *Trapezoid* rule. The value of AUPRC ranges in [0,1] where 1 is the optimal prediction performance of the model.
Precision and recall are defined as

$$Precision = \frac{True\ Positive\ (TP)}{TruePositive\ (TP) + False\ Positive\ (FP)} \tag{3.37}$$

$$Recall = \frac{True\ Positive\ (TP)}{TruePositive\ (TP) + False\ Negative\ (FN)} \tag{3.38}$$

Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made. Precision quantifies the number of correct positive predictions made. The quantities in (3.37) and (3.38) are obtained through the confusion matrix, Table see 3.1.

Table 3.1: Confusion matrix of a classification problem with predicted class on the rows and actual class on the columns. n' and n are predicted and actual negatives. p' and p are predicted respectively actual positives.

		Actual Value		
		n	**p**	
Prediction Outcome	**n′**	True Negatives (TN)	False Negatives (FN)	N′
	p′	False Positives (FP)	True Positives (TP)	P′
		N	P	

where

- *False Positives*: Also known as Type I errors, false alarms, overestimates. This is when the prediction classifies for instance a credit card transaction as fraudulent - when in fact it is *not*.

- *False Negatives*: Also known as Type II errors, miss, underestimates. This metric is the opposite to the false positives. The prediction that has been made has classified a credit card transaction as non-fraudulent - when it actually *is*.

- *True Positives*: These are the cases when the prediction and the actual value both are true, i.e., if the prediction has classified a transaction as fraudulent and it actually *is* fraudulent.

- *True Negatives*: This metric is opposite to the true positives. If the prediction and the actual value both are "false", i.e., if the prediction has classified a transaction as non-fraudulent - when it actually *is* non-fraudulent.

4
Methods

After processing the necessary theory behind the models used in this thesis the aim is to implement these models into the needed programming languages and thus simulate to obtain the results that will be compared and discussed in Chapter 6. This section describes the methods used to obtain the thresholds for the UGPD, model fit of the UGPD and MGPD, training and testing procedures of the supervised FFCNN and unsupervised IF and SVM machine learning algorithms.

4.1 Data set

The data set that is used in this thesis contains credit card transactions made by credit card holders in Europe in September 2013. It includes transactions that occurred during two consecutive days. The original data has been transformed by means of *Principle Component Analysis* (PCA) due to confidential reasons. This means that, some data features has been removed for the public use. PCA is a method that is used to reduce the dimensionality of large data sets. This is done by transforming a large set of variables into a lower dimensional that still contains the relevant information, see e.g [43]. The PCA processed data set contains 284.807 observations and 30 input variables, where 492 of the observations are fraudulent transactions i.e., 0.172% of the data set is the amounted to be fraudulent. The new variables are named "V1" "V2" etc. The only two features that still has attribute names are the variables "Time" and "Amount". The variable/feature "Time" contains the seconds elapsed between each transaction and the first transaction in the data set. However, since the time in seconds for the first transaction is not given - this variable is uninformative. The feature "Amount" is the amount of the transaction that has been occurred. This data set is labeled, which means that we exactly know which transaction is fraudulent and not. The labels are assigned to the variable name "Class" in the data set where "1" is fraudulent observation and "0" is non-fraudulent. The data can be downloaded from [42].

4.2 Training and test set

The credit card transactions data was equally partitioned into a training and test set. By removing one observation, the training and the test set has 142.403 credit card transactions each. Denote the observations in the training set by $\mathcal{T}_{Tr}$ and those for the test set by $\mathcal{T}_{Test}$. See Table 4.1 below.

Table 4.1: *Number of credit card transactions for the test set $\mathcal{T}_{Test}$ and the training set $\mathcal{T}_{Tr}$. The rightmost column are number of anomalies.*

Data set	Transactions	Anomalies
$\mathcal{T}_{Tr}$	142.403	269
$\mathcal{T}_{Test}$	142.403	223

4.2.1 Simulation Equipment

The simulations regarding the training and testing of the ML algorithms were mainly executed in the *Proprietary Multi-Paradigm Programming* language `MATLAB`. The model fits of the UGPD and MGPD together with the goodness of fit plots were done in `R`. The tables below shows the PC specifications and the programming versions.

Table 4.2: *Hardware specifications for the PC. Operating system: macOS Catalina 10.15.7.*

Component	Specifications
RAM	4GB @1600MHz DDR3
CPU	@2.5GHz Dual-Core. Intel Core i5

Table 4.3: *The programs used and their corresponding versions.*

Program	Version
`MATLAB`	R2019b
`R`	R4.0.3

The model fit of the MGPD is done by the program written in [30][54].

4.3 Standardizing

In addition to the anomaly scores obtained from the ML algorithms, the anomaly scores from the *L-Supremum* and *L2*-norm were analyzed. The latter one is also known as the *Euclidean*-norm. To construct these two anomaly scores, the data set is first standardized. Consider the observations $x_{i,j} \in \mathcal{T}$ for $i = 1, \ldots, N$ where N is the number of the credit card transactions i.e., $N = 284.806$ and $j = 1, \ldots, n$, and n is the number of columns, so that $n = 29$. The standardization is done for each $x_{i,j} \in \mathcal{T}$. For this, the column-wise mean and the standard deviation for each column is

$$\bar{x}_{\cdot j} = \frac{1}{N} \sum_{i=1}^{N} x_{i,j}, \tag{4.1}$$

respectively the standard deviation

$$\bar{\sigma_{\cdot j}} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\Big(x_{i,j} - \bar{x}_{\cdot j}\Big)}. \tag{4.2}$$

The standardization can be done by re-location and re-scaling with the mean and standard deviation as follows

$$x^{\star}_{i,j} = \frac{x_{i,j} - \bar{x}_{\cdot j}}{\bar{\sigma}_{\cdot j}}, \tag{4.3}$$

where $x^{\star}_{i,j}$ is the standardized credit card transactions.

The $L2$ and $L-Supremum$ norm are as follows.

Definition 4.3.1 (L2-norm)**.** *Let* $\boldsymbol{x}^{\star} = \{x^{\star}_{i,1}, \ldots, x^{\star}_{i,n}\} \in \mathbb{R}^{N\times n}$*, where* $\mathbb{R}^{N\times n} = \{(x^{\star}_{i,1}, \ldots, x^{\star}_{i,n}) : x^{\star}_{i,1}, \ldots, x^{\star}_{i,n} \in \mathbb{R}\}$ *for* $i = 1, \ldots, N$ *denotes the n-dimensional Euclidean Space. Then the L2-norm of the standardized credit card transactions* $\boldsymbol{x}^{\star}$ *are defined as its Euclidean length*

$$||\boldsymbol{x}^{\star}{}_{i,j}||_{L_2} = \sqrt{\{x^{\star 2}_{i,1} + \ldots + x^{\star 2}_{i,n}\}}, \tag{4.4}$$

where n is the dimension of the row-vector in $\mathbb{R}$*.*

Similarly, the $L-Supremum$ norm is defined as

Definition 4.3.2 (L-Supremum norm for finite dimensional vector)**.** *Let* $\boldsymbol{x}^{\star} = \{x^{\star}_{i,1}, \ldots, x^{\star}_{i,n}\} \in \mathbb{R}^{N\times n}$*, where* $\mathbb{R}^{N\times n} = \{(x^{\star}_{i,1}, \ldots, x^{\star}_{i,n}) : x^{\star}_{i,1}, \ldots, x^{\star}_{i,n} \in \mathbb{R}\}$ *for* $i = 1, \ldots, N$ *denotes the n-dimensional row-vector space. Then the* $L-Supremum$ *of the standardized credit card transactions* $\boldsymbol{x}^{\star}$ *is given by*

$$||\boldsymbol{x}^{\star}{}_{i,j}||_{L_{sup}} = \max\{|x^{\star}_{i,1}|, \ldots, |x^{\star}_{i,n}|\}, \tag{4.5}$$

where n is the dimension of the vector in $\mathbb{R}$

This yields for $i = 1, \ldots, N$ and $j = 1, \ldots, n$ row-wise. One then obtains two column vectors, one for $L2$ and $L-Supremum$ respectively, where the size of the vectors are $N \times 1$.

4.4 Isolation Forest - Model setup

The anomaly detection using the IF is done in two stages, the *training stage* and the *testing stage* [52]. The training stage builds Isolation trees (iTrees) by the algorithm *iTree* using sub-samples of the training set and then building an ensemble of iTrees in the algorithm *iForest*, thus the iForest will contain t number of iTrees. Further, the testing stage passes through the test instances i.e., credit card transactions through the trained iTrees contained in the iForest. The anomaly scores for each instance are obtained with help of the total average and mean path lengths in the iTrees. The IF algorithm was trained with the assumption of 0.2% respectively 1% anomalies in the training set. The reason for this is to see differences in the prediction performance.

4.4.1 Isolation Forest - Training

By passing the training set $\mathcal{T}_{Tr}$, Table see 4.1, into the IF algorithm, the training is done. The training is done by recursively partitioning the given training set by sampling until the credit card transactions are isolated to an external node or a specific iTree height l is reached. In that case, the training process terminates. The height limit l, of the iTree is automatically set by the sub-sampling size ψ as

$$l = ceiling(log_2\psi). \tag{4.6}$$

The sub-sampling size ψ is chosen empirically in [34] to 256 and `MATLAB` uses this as default.

The IF algorithm works well even without increasing ψ. Thus, memory size and processing time can be seen to kept low [34]. This is of course a good property when working with large data sets.

The *iForest* algorithm below builds a forest by the t number of iTrees that have been constructed during the training stage, as discussed in section 3.5.

Algorithm 1 $iForest(\mathcal{T}_{Tr}, t, \psi)$

Inputs: $\mathcal{T}_{Tr} - input\ data, t - number\ of\ iTrees, \psi - sub\ sampling\ size$
Output: A set of t iTrees
1: **Initialize** $Forest$
2: set height limit $l = ceiling(log_2\psi)$
3: **for** i=1 to t **do**
4: $\quad \mathcal{T}'_{Tr} \leftarrow sample(\mathcal{T}_{Tr}, \psi)$
5: $\quad Forest \leftarrow Forest \cup iTree(\mathcal{T}'_{Tr}, 0, l)$
6: **end for**
7: **return** $Forest$

where $\mathcal{T}_{Tr}$ and $\mathcal{T}'_{Tr}$ are the training and sampled training sets respectively, t is the number of iTrees in the forest (iForest) and l is the height limit of the iTree. The *iTree* algorithm below, is used to construct the Isolation Trees (iTrees) which is contained in the forest (iForest).

Algorithm 2 $iTree(\mathcal{T}_{Tr}, e, l)$

Inputs: $\mathcal{T}_{Tr} - input\ data, e - current\ iTree\ height, l - height\ limit$
Output: An iTree
if $e \geq l$ *or* $|\mathcal{T}_{Tr}| \leq 1$ **then**
    return $exNodeSize \leftarrow |\mathcal{T}_{Tr}|$
else
    let Q be a list of attributes in $\mathcal{T}_{Tr}$
    randomly select an attribute $q \in Q$
    randomly select a split point p from max and min values of attribute $q \in \mathcal{T}_{Tr}$
    $\mathcal{T}_{Tr,l} \leftarrow filter(\mathcal{T}_{Tr}, q < p)$
    $\mathcal{T}_{Tr,l} \leftarrow filter(\mathcal{T}_{Tr}, q \geq p)$
    return $inNode\{Left \leftarrow iTree(\mathcal{T}_{Tr,l}, e+1, l)$,
    $Right \leftarrow iTree(\mathcal{T}_{Tr,r}, e+1, l), SplitAtt \leftarrow q, SplittValue \leftarrow p\}$
end if

where the number of iTrees t will control the forest (iForest) size and e is the current height/length of the iTree. The default setting of t this is found to be $t = 100$. This is since, it has been shown that the path lengths of the iTrees usually converge well before t [34]. Thus, more iTrees t will not imply better performance. The default `MATLAB` setting is also $t = 100$. Next, an anomaly score A_{scores} needs to be derived for each credit card transaction.

4.4.2 Isolation Forest - Testing

In the testing stage, the aim is to obtain an anomaly score, A_{score}, for each of the credit card transaction x from the test set $\mathcal{T}_{Test}$. The anomaly score for an credit card transaction x is derived from the mean $\bar{h}(x)$ and average $c(\psi)$ path lengths, as discussed in section 3.5.

The anomaly score itself is obtained by passing the credit card transactions from the test set through the created iTrees in **Algorithm 1** which in turn is found in the $iForest$-algorithm, see **Algorithm 2**. Further, by using the function $PathLength$ in **Algorithm 3**, a single path length $h(x)$ is derived by counting the number of edges from the root node to an external node T as x traverses through an iTree in the forest (iForest), see Figure 3.1. This means that, each credit card transaction is initiated from the root node, see Figure 3.1, and then forwarded until it terminates at an external node. The depth of this distance in the iTree will be the path length. The anomaly score A_{score}, is thus computed by (3.16) and such that $A_{score} \in [0, 1]$. The output vector of the anomaly scores is

$$\boldsymbol{A}_{score} = \begin{bmatrix} A^1_{score} \\ A^2_{score} \\ \vdots \\ A^N_{score} \end{bmatrix}, \tag{4.7}$$

for $i = 1, \ldots, N$, where N is the number of credit card transactions $x_i \in \mathcal{T}_{Test}$. The algorithm for computing the path length is

Algorithm 3 *PathLength*(x, T, e)

Inputs: $x - an\ instance, T - an\ iTree\ , e - current\ path\ length$
Output: path lenght of x
1: **if** T *is an external node* **then**
2: return $e + c(T.size)$
3: **end if**
4: $a \leftarrow T.split.Att$
5: **if** $x_a < T.splitValue$ **then**
6: return $PathLength(x, T.Left, e + 1)$
7: **else**{x$_a \geq$ T.splitValue}
8: return $PathLength(x, T.Right, e + 1)$
9: **end if**

4.5 Support Vector Machine - Model setup

In the training stage of the SVM, the aim is to solve an *Quadratic programming problem*, given in (3.20). The minimization is done by the modified *Sequential Minimal Optimization* that was proposed in 1990 by Platt for classification problems. Having the optimization done, the decision function provided in (3.26) is used to obtain a decision region $\mathcal{R}_{w,\rho}$. Hence, during the testing stage, any new credit card transaction t drawn from an unknown probability $\mathcal{P}$ is expected to fall within the decision region $\mathcal{R}_{w,\rho}$. If not, with some certain priori choice of the parameter $\nu \in (0, 1)$, the credit card transaction is considered as an anomaly. The parameter ν is set to its default value $\nu = 0.50$ in `MATLAB` [53]. In this thesis, the SVM algorithm, as for the IF algorithm, is trained under the assumptions of 0.2% respectively 1% anomalies in the training set.

4.5.1 Support Vector Machine - Training

Consider a set of training samples consisting of credit card transactions $t_i \in \mathcal{T}_{Tr}$ for $i = 1, \ldots, n$ and $t_i \in \mathbb{R}^d$. Suppose these are distributed according to some unknown probability distribution $\mathcal{P}$. Then, for any new credit card transaction from the test set, we want to know if it is distributed according to $\mathcal{P}$ or not. This is done by estimating a decision region $\mathcal{R}_{w,\rho}$ of the training sequences in the input space, such that the probability that a credit card transaction drawn from $\mathcal{P}$ is bounded by ν. The decision function will output the values $f(\boldsymbol{t}) > 0$ if $\boldsymbol{t} \in \mathcal{R}_{w,\rho}$ and $f(\boldsymbol{t}) < 0$ if $\boldsymbol{t} \notin \mathcal{R}_{w,\rho}$. Thus, to define the decision region $\mathcal{R}_{w,\rho}$, f must be estimated.

Assume that the training data is not perfectly separable in the input space $\mathcal{I}$, using the kernel trick: the training data in $\mathcal{T}_{Tr}$ is mapped by a non-linear *Radial Basis* kernel function $\Phi : \mathcal{T}_{Tr} \rightarrow \mathcal{F}$ to a higher dimensional space $\mathcal{F}$. In this new space, the training sequences follow an underlying distribution $\mathcal{P}'$, hence we want to determine

a region $\mathcal{R}'_{w,\rho}$ of $\mathcal{F}$ that captures most of the training data. Since the origin in the feature space is considered belonging to the anomalous credit card transactions, we want to separate the mapped vectors using a hyperplane with maximum margin from the origin [37].

The maximum margin from the origin is found by solving the following dual problem of (3.20) as

$$\begin{cases} \min\limits_{\boldsymbol{\alpha}} & \frac{1}{2}\sum\limits_{i,j=1}^{n}\alpha_i\alpha_j k(\boldsymbol{t_i},\boldsymbol{t_j}) \\ \text{subject to} & 0 \leq \alpha_i \leq \frac{1}{\nu n}, \sum\limits_{i}^{n}\alpha_i = 1, \end{cases} \tag{4.8}$$

where the estimated decision function is given in (3.26) and the induced decision region in the feature space is

$$\mathcal{R}'_{w,\rho} := (\boldsymbol{t} : f_w(\boldsymbol{t}) \geq \rho), \tag{4.9}$$

and ρ is the offset.

Remark 6. *The priori parameter $\nu \in (0,1)$ can be adjusted when training the algorithm and thus make a trade off regarding the fraction of the anomalies that will be assimilated in contrast to minimizing the total are of $\mathcal{R}'_{w,\rho}$. Training vectors $\boldsymbol{t}$ for $f(\boldsymbol{t}) \leq 0$ are called support vectors. For $f(\boldsymbol{t}) = 0$ are called support vectors and non-margin support vectors for which $f(\boldsymbol{t}) < 0$.*

4.5.2 Support Vector Machine - Testing

The main part of the SVM algorithm is done during the training progress. In particular, the SVM is heavy during its training progress since optimization problems have to be solved. The testing stage consists of checking whether new unseen credit card transactions, taken from the test set $\mathcal{T}_{Test}$, will, or will not lie within the induced decision region $\mathcal{R}'_{w,\rho} \in \mathcal{F}$ [37].

The anomaly scores denoted by $\mathcal{A}_{SVM}$ are obtained by mapping the test sequences into the feature space $\mathcal{F}$ and see if they falls inside the induced region $\mathcal{R}'_{w,\rho}$. The anomaly scores are appointed as,

$$f(\boldsymbol{t})_w > 0 \;\text{ if }\; \boldsymbol{t} \in \mathcal{R}'_{w,\rho} \;\text{ and }\; f(\boldsymbol{t})_w < 0 \;\text{ if }\; \boldsymbol{t} \notin \mathcal{R}'_{w,\rho}. \tag{4.10}$$

Therefore, the anomaly scores $\mathcal{A}_{SVM}$ will be positive within the region $\mathcal{R}'_{w,\rho}$ and negative on its complement and be in the range $-\infty < \mathcal{A}_{SVM} < +\infty$. Strict negative scores indicates that the observation is far away from the hyperplane separating the anomalies and normal instances i.e., closer to the origin.

4.6 Univariate Generalized Pareto Distribution - Model setup

To being able to fit and simulate the MGPD, the thresholds $\boldsymbol{u} = \{u_1, u_2, u_3, u_4\}$ must be determined. For convenience, write $u_1 = u_{IF}$, $u_2 = u_{SVM}, u_3 = u_{L2}, u_4 = u_{L-Sup}$, for the thresholds of the anomaly scores from the IF, SVM, L2 and L-sup, denoted, hereafter by, $\mathcal{A}_{IF}$, $\mathcal{A}_{SVM}$, $\mathcal{A}_{L2}, \mathcal{A}_{L-Sup}$. Moreover, assuming that the threshold excesses of the anomaly scores obtained from the SVM, IF, L2 and L-Sup are independent and identically distributed. Then the parameters γ and σ of the cdf of the UGPD

$$H(x) = 1 - \left(1 + \frac{\gamma x}{\sigma}\right)^{-\frac{1}{\gamma}}, \tag{4.11}$$

can be estimated by ML method. Therefore, the thresholds u for the four models must be chosen.

4.7 Univariate Generalized Pareto Distribution - Threshold Selection

The threshold selection is done by means of the *mean residual life plot* and by mapping the parameter estimates against a sequence of thresholds numerically, *Threshold Parameter Stability Method*. However threshold selection is not trivial and that the mentioned approaches are standard methods [25], hence there is no unique approach and a compromise needs to be done. A too low threshold u will violate the asymptotic basis of the model, leading to bias, and a too high threshold u will not generate enough exceedances to give good parameter estimates. The first step for the UGPD model is to choose the threshold to each of the models. The reasoning behind the threshold parameter stability method is as follows; if **Theorem 3.1.1** applies, i.e, if a GPD is a reasonable model for excesses y_i of a high threshold u_0, then it is also true that an excess of higher threshold $u > u_0$ follows a GPD. Moreover, the shape parameter γ of the two distributions u and u_0 are identical, and letting σ_u denote the scale value of the GPD for the threshold u, then by (4.11) it holds that

$$\sigma_u = \sigma_{u_0} + \gamma(u - u_0). \tag{4.12}$$

This means that the scale parameter σ_u will change with respect to u unless the shape parameter is $\gamma = 0$. However, the parameter

$$\sigma^\star = \sigma_u - \gamma u, \tag{4.13}$$

is in fact constant for high enough u by virtue of (4.12). However, due to sampling variability these estimates will not be exactly constant. Thus one plots the parameter estimates $\hat{\sigma}^\star$ and $\hat{\gamma}$ against the threshold parameter value together with the estimated *confidence intervals* (CI) and select u_0 as the lowest value for u which the estimates $\hat{\sigma}^\star$ and $\hat{\gamma}$ remain approximately constant. Hence, one uses two plots for each of the set of anomaly scores to choose a threshold. The threshold parameter

stability method is given in Appendix A.2.

The second method for threshold selection to use, is the mean residual life plot MRLP [25]. The two graphical methods complement each other. If H has a GPD with parameters γ and σ then the mean of the distribution is

$$\mathbb{E}[H] = \frac{\sigma}{1-\gamma}, \tag{4.14}$$

provided that the shape parameter is $\gamma < 1$, while the mean is infinite for $\gamma \geq 1$. Assuming that the GPD is a valid model for the excesses y_i of a sufficiently high threshold, then the conditional expectation of the excesses are given by

$$\mathbb{E}[A - u | A > u] = \frac{\sigma_u}{1-\gamma}, \tag{4.15}$$

where A is the anomaly score and σ_u is the scale parameter corresponding to the excesses of the threshold u. Hence, if the GPD is a valid model for the higher excesses for a high level threshold u_0, then (4.15) holds also for the thresholds $u > u_0$ for appropriate change of the scale parameter σ_{u_0}. Thus for $u > u_0$

$$\mathbb{E}[A - u | A > u] = \frac{\sigma_u}{1-\gamma} = \frac{\sigma_{u_0} + \gamma u}{1-\gamma}. \tag{4.16}$$

Therefore, it is expected that the estimates γ and σ_u should change linearly with respect to the thresholds u where the GPD is a valid model for the excesses [25]. Hence, one plots

$$\left\{ \left(u, \frac{1}{n_u} \sum_{i=1}^{n_u} (A_i - u) \right) 1_{\{A_i > u\}} \right\}, \tag{4.17}$$

where A_i are the anomaly scores and n_u is the number of observations that exceed u. The interpretation of (4.17) is that, one plots a sequence of thresholds u against the mean of the threshold excesses. Further, one looks for the value above u_0 to identify linearity in the plot itself. Indeed, if the GPD assumption is correct, then the plot should be linear with its intercept $\frac{\sigma_{u_0}}{1-\gamma}$ and gradient $\frac{\gamma}{1-\gamma}$, before it becomes unstable due to low number of data points. Confidence intervals can also be included in the plots. Finally, it is also good to have in mind that the plots always converge to the point $(A_{\max}, 0)$. The mean residual life plot for the anomaly scores from the four models can be viewed in Appendix A.2.

Having the thresholds $\boldsymbol{u} = \{u_1, u_2, u_3, u_4\} = \{u_{IF}, u_{SVM}, u_{L2}, u_{L-Sup}\}$ set, it might still be good to check how well the thresholds have been chosen. For this purpose, model checking is done, see section 4.9.

Below is the threshold for each of the four models that has been chosen. Each unique threshold has been chosen in such a way that it yields $y \approx 1000$ excesses. See Table 4.4 below.

Table 4.4: *Determined threshold u for each of the four models; L-Sup, L2, IF and SVM.*

Model	u
L-Sup	9.500
L2	17.000
IF	0.615
SVM	-0.400

4.8 Univariate Generalized Pareto Distribution - Parameter Estimation

When the threshold selection is done, the next step is to find parameter estimates for the four sets of anomaly threshold excesses.

The parameter estimates can be done by several methods [47]. Depending on the statistical model of interest, there are various of numerical methods to estimate the parameters γ and σ of the GPD such as: Probability Weighted Moments method, Pickands Estimator, Moment method and Maximum Likelihood Estimation [47]. However, the ML method is the only numerical approach that combines theoretical efficiency and provides a general basis for inference [47]. The likelihood for a GPD is given by

$$\mathcal{L}(\sigma, \gamma) = \prod_{i=1}^{n_u} h(y_i; \sigma, \gamma), \tag{4.18}$$

where h denotes the pdf of H, given in (3.5) and the log-likelihood is

$$\ell(\sigma, \gamma) = \log(\mathcal{L}(\sigma, \gamma)) = \sum_{i=1}^{n_u} \log(h(y_i; \sigma, \gamma)), \tag{4.19}$$

where σ and γ are the parameters to be estimated for the anomaly scores obtained from the IF and SVM algorithms and L2 and L-Sup transformations and the chosen thresholds. In more detail, the estimates are obtained by considering a sequence of excesses y_i for $i = 1, \ldots, n_u$ of a high level threshold parameter value u, such that for $\gamma \neq 0$ the log-likelihood with respect to the parameters γ and σ is

$$\ell(\sigma, \gamma) = -n_u \log(\sigma) - (1 + \frac{1}{\gamma}) \sum_{i=1}^{n_u} \log\left(1 + \gamma \frac{y_i}{\sigma}\right), \tag{4.20}$$

provided that $(1 + \gamma \frac{y_i}{\sigma}) > 0$ for $i = 1, \ldots, n_u$; otherwise $\ell(\sigma, \gamma) = -\infty$ [25]. Note that, in the case of $\gamma = 0$ the log-likelihood is

$$\ell(\sigma) = -n_u \log(\sigma) - \frac{1}{\sigma} \sum_{i=1}^{n_u} y_i, \tag{4.21}$$

where both of the above log-likelihoods must be maximized numerically to obtain the parameter estimates $\hat{\sigma}$ and $\hat{\gamma}$. Standard errors and confidence intervals for the generalized Pareto distribution are obtained by standard likelihood theory [25]. The table below gives the parameter estimates and negative log-likelihoods for each of the four models.

Table 4.5: *Parameter estimates for each of the four models; L-Sup, L2, IF and SVM together with their negative log-likelihood. The values in brackets are the standard errors of the estimates.*

Model	$\hat{\gamma}$	$\hat{\sigma}$	Neg. log-likelihood
L-Sup	0.1706344(0.034)	4.4844074(0.198)	3154.735
L2	0.245571(0.046)	6.973362(0.385)	3302.424
IF	-0.29393901(0.028)	0.04489504(0.002)	-2555.593
SVM	-2.079071(2e-08)	8.828754(2e-08)	631.6619

Remark 7. *Note that the IF and SVM models will have right end-point limit* $\frac{\sigma}{|\gamma|}$ *since* $\gamma < 0$ *and left end-point* 0*. Analogously, the L2 and L-Sup excesses will have* ∞ *as right end-point, since* $\gamma \geq 0$*, and* 0 *as left end-point.*

4.9 Univariate Generalized Pareto Distribution - Goodness of fit

Next, one needs to check whether the fitted UGPD models are reasonable as models for the excesses y_i of u_j for $j = \{SVM, IF, L2, L-Sup\}$. For this purpose, the *Quantile-Quantile* (QQ), *Probability-Probility* (PP), *Return Level* (RL) and *Kernel Density* plots were used. Additional model checking plots are provided in Appendix A.3.

The PP plot shows the points

$$\{\Big(\frac{i}{n_u+1}\Big), \hat{H}(y_i);\ i = 1, \ldots, n_u\}, \tag{4.22}$$

where

$$\hat{H}(y) = 1 - \Big(1 + \frac{\hat{\gamma}y}{\hat{\sigma}}\Big)^{-\frac{1}{\hat{\gamma}}}, \tag{4.23}$$

provided that $\hat{\gamma} \neq 0$. Again, assuming that $\hat{\gamma} \neq 0$, the QQ plot shows the points

$$\{(\hat{H}^{-1}\Big(\frac{i}{n_u+1}\Big), y_i)\};\ i = 1, \ldots, n_u\}, \tag{4.24}$$

where

$$\hat{H}^{-1}(y) = u + \frac{\hat{\sigma}}{\hat{\gamma}}\Big[y^{-\hat{\gamma}} - 1\Big]. \tag{4.25}$$

If the generalized Pareto model is reasonable for modeling excesses of u, then both the PP and QQ plots should be approximately linear [25].

Finally, the obtained UGPD models for the respective models i.e., IF and SVM respectively L2 and L-Sup metrics is compared to the estimated probability density function that is obtained from the excesses. Here, the actual probability density is estimated by a *Kernel Density Estimate* with some kernel function, for instance as the Gaussian, Epanechnikov or Triweight. The kernel density estimate provides a smooth estimate of the pdf, see [49]. In this case, the popular Gaussian kernel were used. The estimated GDP pdf is

$$\tilde{h}(y) = \frac{1}{\hat{\sigma}}\Big(1 + \frac{\hat{\gamma}y}{\hat{\sigma}}\Big)^{-\frac{1}{\hat{\gamma}}-1}. \tag{4.26}$$

Hence, one considers a smooth estimate of the pdf rather than the usual approach of only plotting the excesses in an histogram. The kernel density estimate for an unknown univariate density f is given by

$$\hat{f}(Y; b) = \frac{1}{nb}\sum_{i=1}^{n} K\Big(\frac{y - y_i}{b}\Big), \tag{4.27}$$

for some non-negative kernel function K and where $b > 0$ is a smoothing parameter so called the *bandwidth* [50]. If the kernel function K is Gaussian, then K is defined as

$$K(y) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}y^2}. \tag{4.28}$$

For more information, see e.g. [48].

4.10 Multivariate Generalized Pareto Distribution - Simulation & Model fit

Having the parameter estimates and thresholds from the UGPD fit for the four models IF, SVM, L2 and the L-Sup metrics, the next step is to fit a MGPD for the IF, L2 and L-Sup excesses and compute the negative log-likelihood in the test data.

Recall that the MGPD is a model for the excesses of the high thresholds determined for the models IF, L2 and the L-Sup, that operates components wise on a random vector $\mathbf{X} = \{X_1, X_2, \ldots, X_d\}$ to give the excess vector $\mathbf{Y} = (X_1 - u_1, \ldots, X_d - u_d)$. Moreover, the excess vector $\mathbf{Y}$ is considered as positive and is included in the model fitting if at least one of the components exceeds its threshold, see [30].

To fit the MGPD model, the univariate threshold excesses of the IF, L2 and L-Sup are transformed to an approximate standard exponential distribution. If Y has a

univariate GPD with parameters σ and γ then

$$\frac{1}{\gamma} ln\Big(\frac{\gamma}{\sigma} \cdot Y + 1\Big), \tag{4.29}$$

has a standard standard exponential distribution as can be seen from the following computation

$$\mathbb{P}\Big[\frac{1}{\gamma} ln(\frac{\gamma}{\sigma} \cdot Y + 1)] \leq x] = \mathbb{P}\Big[ln(\frac{\gamma}{\sigma} \cdot Y + 1)] \leq \gamma x] = \mathbb{P}\Big[\frac{\gamma}{\sigma} \cdot Y + 1 \leq e^{\gamma x}\Big], \tag{4.30}$$

$$\mathbb{P}\Big[\frac{\gamma}{\sigma} \cdot Y + 1 \leq e^{\gamma x}\Big] = \mathbb{P}\Big[Y \leq \frac{\sigma}{\gamma}(e^{\gamma x} - 1)\Big] = 1 - \Big(1 + \frac{\gamma}{\sigma}\frac{\sigma}{\gamma}(e^{\gamma x} - 1)\Big)^{-\frac{1}{\gamma}} = 1 - e^{-x}. \tag{4.31}$$

Hence, if we transform the observed excesses $\{y_1, y_2, \ldots, y_n\}$ to

$$X_1 = \frac{1}{\hat{\gamma}} ln\Big(\frac{\hat{\sigma}}{\hat{\gamma}} \cdot y_1 + 1\Big), \ldots, \frac{1}{\hat{\gamma}} ln\Big(\frac{\hat{\sigma}}{\hat{\gamma}} \cdot y_n + 1\Big) = X_n, \tag{4.32}$$

then $\{X_1, X_2, \ldots, X_n\}$ will be approximately standard exponentially distributed and ln is the e-logarithm.

Next, by the goodness of fit diagnostics in Appendix A.4 it shows that positive transformed SVM excesses do not have a standard exponential distribution. Hence, the MGPD is fitted to the 3-dimensional vector of transformed IF, L2 and L-Sup excesses for which at least one component is positive.This yields, $N_{U_{Tr}} = 1559$ 3-dimensional vectors. We denote this new data set $\mathcal{E}_{Trans_{Tr}}$. Further, the previous procedure is repeated on the test set using the thresholds and parameter estimates of IF, L2 and L-Sup from the the training set, see Table 4.5. This yielded a second data set consisting of $N_{U_{Test}} = 1566$ 3-dimensional vectors. This data set is denoted by $\mathcal{E}_{Trans_{Test}}$. A 3-dimensional MGPD with independent Gumbel generators and density given by (3.10)

$$h_{\mathbf{U}}(\mathbf{x}) = \frac{\int_0^\infty \prod_{i=1}^{d=3} \alpha_i (te^{x_i - \beta_i})^{-\alpha_i} e^{-(te^{x_i} - \beta_i)^{-\alpha_i}} dt}{\int_0^\infty \Big(1 - \prod_{i=1}^{d=3} e^{-(t/e^{\beta_i})^{-\alpha_i}}\Big) dt}, \tag{4.33}$$

is then fitted to the $\mathcal{E}_{Trans_{Tr}}$ data using ML estimators. As discussed in section 3.4, β_1 was set to be 0. The fitted 3-dimensional MGPD was then used to calculate the negative log-likelihood for each observation in the data set $\mathcal{E}_{Trans_{Test}}$. The expectation is to obtain high values of negative log-likelihood correspond to the anomalous observations, whereas normal observations i.e., non fraudulent cases are expected to have lower values of negative log-likelihood.

4.11 Feedforward Fully Connected Neural Network - Model setup

The FFCNN progresses through the same stages as the two unsupervised IF and SVM ML algorithms. The algorithm is trained on the same training set $\mathcal{T}_{Tr}$ and also

tested on the same test set $\mathcal{T}_{Test}$. However, the difference is that, the class labels of the data set will be used during training. The training samples will be trained against the targets and during the test stage check if it can classify anomalous and normal instances correctly. The model setup is done in `MATLAB` [51].

4.11.1 Feedforward Fully Connected Neural Network - Training

The training is done by mapping the training matrix $\boldsymbol{X}_{N,n}$ against the vector of class labels $\mathbf{t}$ that contains the labels for each credit card transaction. There will be $n = 29$ input layers to the FFCNN, which can be viewed in Figure 3.2. The credit card transactions in the training set and its features gives a 29×142.403-matrix $\mathbf{X}_{N,n}$

$$\boldsymbol{X}_{N,n} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,n} \end{bmatrix}, \tag{4.34}$$

which is then trained against the class label vector $\mathbf{t}$ as

$$\boldsymbol{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}. \tag{4.35}$$

The next step is to randomly initialize the weights w_{ij}, where i refers to the neuron that does the computation and j is all the neurons that connect to neuron i, and then forward the credit card transactions through the hidden layers which have activation functions g, as discussed in section 3.7. The weights w_{ij} are random uniformly and independently initialized at each layer as

$$w_{ij} \sim \mathcal{U}\Big[-\frac{1}{\sqrt{l}}, \frac{1}{\sqrt{l}}\Big], \tag{4.36}$$

where l is the size of the layer in number of neurons. The biases are set to zero [45]. Thus, the credit card transactions are forwarded through the net until they reach the output layer, given in (3.34). This process is then repeated while simultaneously minimizing the *Cross Entropy*, which is derived particularly with respect to the softmax function [38]. The minimization is done by *Limited-Memory Broyden-Fletcher-Goldfarb-Shanno* minimization method [44]. Thus, the FFCNN will adjust its weights for each iteration until the cross entropy is minimized. The intuitive reasoning behind minimizing the cross entropy, is because the cross entropy is minimal when the output values of the softmax function gives correct predictions. For details

of the Limited-Memory Broyden-Fletcher-Goldfarb-Shanno algorithm, see e.g. [44].

Under the assumption that the FFCNN is not over fitted during training, such as not using overwhelming many neurons and hidden layers, then with obtained optimal weights through the cross entropy minimization, the FFCNN is expected to perform well on unseen data.

4.11.2 Feedforward Fully Connected Neural Network - Testing

When the FFCNN is trained and optimal weights for the network are obtained by Limited-Memory Broyden-Fletcher-Goldfarb-Shanno Quasi-Newton minimization a prediction on the test set $\mathcal{T}_{Test}$ is done. The aim during the testing is to make the correct prediction with the softmax outputs $O_i^{(\mu)}$ for every credit card transaction i in the test set, where μ denotes the different patterns that the test instance passes through the neural network via the different weight connections between the input layer and the output layer.

For $K = 2$ classes in the data set, the softmax output unit i is assumed to represent the posterior probability that the input x_i for $i = 1, \ldots, n$ credit card transactions is in class i when having that the class labels; $t_i^{(\mu)} = 1$ while $t_k^{(\mu)} = 0 \; for \; k \neq i$. For instance, having that $O_i^{(\mu)} \approx 1$, then the FFCNN is quiet certain that the observation x_i is in class i e.g. that class i is fraudulent, then the corresponding class label for a given credit card transaction is $t_i^{(\mu)} = 1$. The classification scores are posterior probabilities and is given in a two column vector of size $N \times 2$ - one column for each class.

5
Results

This chapter describes the results that have been obtained in this thesis. This includes performance comparison metrics and the final models that have been used to fit the Multivariate Generalized Pareto Distribution with independent Gumbel generators and its parameter estimates. Moreover, the parameter estimates obtained from the UGPD by the anomaly threshold excesses originating from IF, L2, SVM and the L-Sup are presented. Goodness of fit plots for these are also provided in Appendix A.3. In particular, this chapter provides the results that are obtained when assuming 1% and when assuming 0.2% anomalies in the training set for the SVM and IF. The results are presented by PR curves and AUPRC values. ROC curves, AUROC values and confusion matrices can be viewed in Appendix A.5. For additional results obtained on different training and test sets, see also Appendix A.5. The discussion in Chapter 6, will mainly be based on the content of the PR curves and AUPRC values. The reason is since that, the ROC curves and AUROC values do not provide good comparison in highly imbalanced data sets, as discussed in section 3.8.

5.1 Parameter Estimates - UGPD

By the goodness of fit plots in Appendix A.4 for the SVM model, it did not show a good fit of the standard exponential distribution after transformation of the anomaly threshold excesses. The threshold for the models are shown in Table 5.1 below.

Table 5.1: *Determined threshold u for each of the four models; L-Sup, L2, IF and SVM.*

Model	u
L-Sup	9.500
L2	17.000
IF	0.615
SVM	-0.400

Also for other threshold choices for the SVM model, the transformed anomaly threshold excesses were not standard exponentially distributed and hence the SVM was

not used in the MGPD model fitting. Hence, the MGPD was fitted to the standard exponential transformed anomaly threshold excesses from the L2, L-Sup and IF models.

The parameter estimates for anomaly threshold excesses of the UGPD with respect to the L-Sup, L2, IF and SVM together with their standard error estimates and negative log-likelihood is shown below, see Table 5.2. Note that the anomaly scores obtained from the IF and SVM algorithms is done under the assumption of 1% anomalies in the training set.

Table 5.2: *Parameter estimates for the four models; L-Sup, L2, IF and SVM together with their negative log-likelihood. The values in brackets are the standard error of the estimates.*

Model	$\hat{\gamma}$	$\hat{\sigma}$	Neg. log-likelihood
L-Sup	0.1706344(0.034)	4.4844074(0.198)	3154.735
L2	0.245571(0.046)	6.973362(0.385)	3302.424
IF	-0.29393901(0.028)	0.04489504(0.002)	-2555.593
SVM	-2.079071(2e-08)	8.828754(2e-08)	631.6619

The goodness of fit plots for the models with the parameter estimates in Table 5.2 are given in Appendix A.3.

5.2 Parameter Estimates - MGPD

The following table shows the parameter estimates of the 3-dimensional MGPD with independent Gumbel generators that was fitted to the standard exponential transformed anomaly threshold excesses of L-Sup, L2 and IF.

Table 5.3: *Parameter estimates of the 3-dimensional MGPD with independent Gumbel generators, fitted to the standard exponential transformed anomaly threshold excesses of L-Sup, L2 and IF on the set $\mathcal{E}_{Trans_{Tr}}$ and the negative log-likelihood. The location parameter β_1 was set to $\beta_1 = 0$ in the ML estimation.*

Model	$\hat{\alpha_1}$	$\hat{\alpha}_2$	$\hat{\alpha_3}$	β_1	$\hat{\beta}_2$	$\hat{\beta}_3$	Neg. log-likelihood
MGPD	7.70	3.36	2.05	0	0.03	-1.07	4729.781

That $\alpha_i > \mathbf{1}$ and $\beta_i \in \mathbb{R}$ for $i = 1, 2, 3$ is consistent with theory.

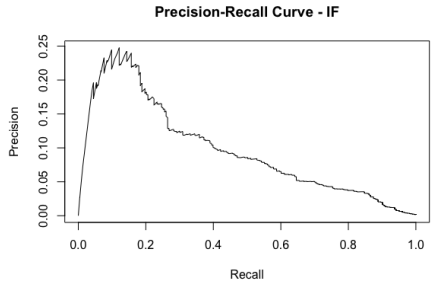
5.3 PR curves & AUPRC values for 0.2% and 1% anomalies in the training set

The PR curves and AUPRC values for FFCNN, IF, SVM, L2, L-Sup and MGPD are shown in Table 5.4 and Figure 5.1. The assumption about the percentage of anomalies in the training set $\mathcal{T}_{Tr}$ is 0.2% and 1% for the SVM and IF. The other methods do not require that one chooses a percentage of anomalies.

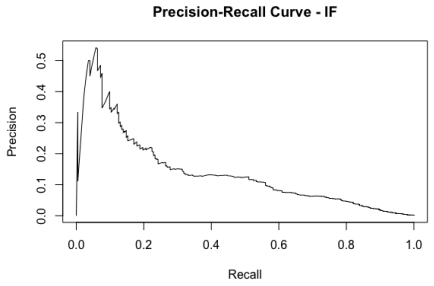
Table 5.4: *AUPRC for each of the six models; MGPD, L-Sup, L2, IF, SVM and FFCNN. Note that the FFCNN is supervised.*

Model	AUPRC
MGPD	0.0938
L-Sup	0.0797
L2	0.1070
IF 0.2%	0.0858
IF 1.0%	0.1350
SVM 0.2%	0.0010
SVM 1.0%	0.0018
FFCNN	0.6870

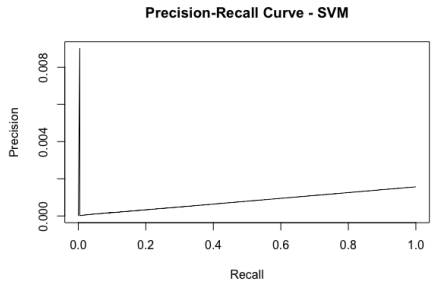
The PR curves for the AUPRC values shown above is given in Figure 5.1 below.



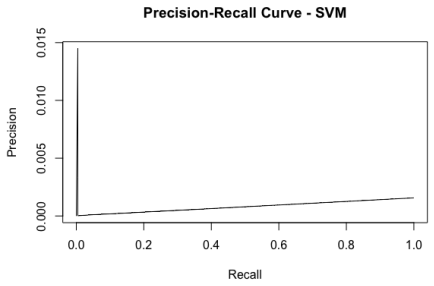
(a) PR Curve IF 0.2% on $\mathcal{T}_{Test}$.



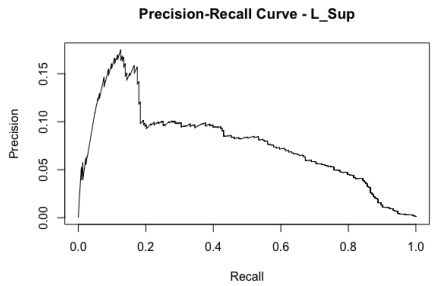
(b) PR Curve IF 1% on $\mathcal{T}_{Test}$.



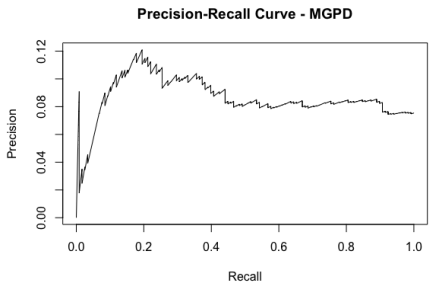
(c) PR Curve 0.2% SVM on $\mathcal{T}_{Test}$.



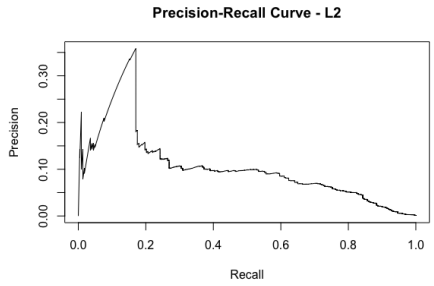
(d) PR Curve 1% SVM on $\mathcal{T}_{Test}$.



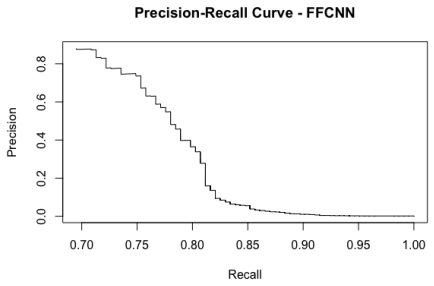
(e) PR Curve L-Sup on $\mathcal{T}_{Test}$.



(f) PR Curve MGPD on $\mathcal{E}_{Trans_{Test}}$.



(g) PR Curve L2 on $\mathcal{T}_{Test}$.



(h) PR Curve FFCNN on $\mathcal{T}_{Test}$.

Figure 5.1: Precision-Recall (PR) curve for the six models; IF, SVM, L-Sup, MGPD, L2 and FFCNN. The MGPD is fitted on the training set $\mathcal{E}_{Trans_{Tr}}$ and tested on $\mathcal{E}_{Trans_{Test}}$. The other models are trained on $\mathcal{T}_{Tr}$ and tested on $\mathcal{T}_{Test}$ respectively. The SVM and IF are trained under the assumption of 0.2% and 1% anomalies respectively in $\mathcal{T}_{Tr}$. Note the different y-scale in the plots.

Results based on the models when tested on the training set can be found in Appendix A.5.

6
Conclusion & Discussion

This chapter discusses important aspects of the results obtained in this thesis and a general outlook on today's fraud detection systems and unsupervised anomaly detection is presented. The results in Chapter 5 regarding performance comparison is discussed and benefits versus drawbacks with the models examined are discussed in detail - whereas a conclusion regarding the best model is uplifted. Finally, suggestions for possible future research within unsupervised anomaly detection and ML is provided.

6.1 General Outlook

Anomaly detection in general is hard. In particular, in field of unsupervised anomaly detection there are still lacking robust methods of detecting anomalous instances in large and highly imbalanced data sets. There are occurring fraudulent transactions each every day in today's society where the use of credit cards and e-business is increasing rapidly. In addition to this, as mentioned in section 2.2, the ways of committing data infringement into peoples computers and thus extracting sensitive information with regards to for instance credit card information are under development and is creating a threat towards the economical system of interest. As described in section 2.3, the predictions for the amount of financial losses due to credit card frauds are rising. Therefore, an effective, where effective referrers to un-expensive and robust, framework for anomaly detection needs to be contributed to the financial institutions. If this is not done, there will be ongoing frauds that will hurt the economical system in the long term and also hurt the reputation of the finance institutions that are exposed more frequently to credit card frauds.

Today's mainly expert driven fraud detection systems i.e., the fraud investigators, as put in perspective in section 2.4, is lacking. The main reason is that the rules that are put up by the fraud investigators are static, biased and can differ from investigator to investigator. Often the investigators are not able to track new fraudulent patterns where unnoticed fraudulent cases emerge. However, it is not said that fraud investigators are unnecessary - but they need robust and accurate assistance where suspicious observations from the large data sets consisting of credit card transactions can reliably be brought to their attention. This is where the new framework regarding unsupervised anomaly detection must step in and contribute and thus be an useful tool to the financial and credit institutions but it can not be a complete replacement. If new frameworks of unsupervised anomaly detection can extract a

fraction of the previously unnoticed suspicious credit card transactions in large data sets, this can without any hesitation be seen as an important contribution.

6.2 Model Discussion

By Figure 5.1 and Table 5.4 it is clear that the supervised ML algorithm FFCNN is the model with the best Precision-Recall curve and AUPRC value i.e., the model that can most accurately identify the anomalous instances and classify them correctly in the test set. Under the assumption that the financial institutions and the credit unions gives out credit cards to the same group of customers, one can expect the same transaction patterns year to year. Thus, if the supervised FFCNN model is well trained on a data set, in terms of not over-fitting it, then it is likely to be applicable on other test sets originating from the same group of card holders. This means that, labeled data sets can be manufactured by the financial and credit card institutions such that they can implement supervised algorithms on these labeled data sets. In particular, the investigators can tune their models by adding new anomalous instances into the test sets and optimize their models accordingly - and in parallel not over fitting the considered models. However, one must still beware that, new infringement methods will be developed by the fraudsters and that these can fool the trained model. So, one can never be sure that the normal or anomalous profile in the training set will be the same as in a new data set.

Even though the supervised FFCNN ML algorithm show significantly better performance compared to the IF, SVM, L2, MGPD and L-Sup, one must have in mind that, having large labeled data sets is often not realistic and it requires a lot of resources to maintain such data sets with new qualitative data. Therefore, if the financial institutions and credit unions gives out credit cards to other groups of card holders or/and if the buying behaviour changes within the group, even worse, if new data infringement methods are used, then the trained models will most likely perform less well and costly false positives and negatives will probably occur. For this reason, unsupervised models, applied on non-labeled data sets must be considered for cost and time effectiveness, and for keeping watch against not yet seen types of frauds.

The unsupervised ML algorithms IF and SVM shows interesting results. Before discussing, one should recall some important aspects of the performance comparison metrics. First, note that the ROC and thus AUROC curve can be misleading, in particular with so for to the AUROC e.g. the area under the ROC for the FFCNN in Table A.4. This in fact illustrates that the ROC curve is not a suitable performance metric with regards to highly imbalanced data sets. The reason for this is that the FPR does not decrease steeply when the total amount of real negatives is huge. The PR curve is sensitive for false positives and is neglecting the total amount of actual negatives in the data set, see (3.37). Overall, the PR curve is as stated in section 3.8 more suitable for understanding anomaly detection in highly imbalanced data sets. This is also exhibited by means of the AUPRC in Table 5.4 compared with the AUROC in Table A.4 i.e., that the AUPRC for the FFCNN is greater than the

others even though its AUROC smaller.

The SVM in an unsupervised setting was not able to predict any anomalies in the test set, regardless of the assumed amount of anomalies in the training set i.e., 0.2% or 1%. The reason is that the SVM operates by creating a decision boundary between the two classes in its mapped feature space, where it tries to capture most of the assumed normal instances within a broad region, see section 4.5.1. Having a highly imbalanced data set makes it extremely difficult to create such a boundary that is sensitive to the minority class. For this purpose, the samples at the boundary of the decision boundary/hyperplane are more likely to be classified to the majority class. Hence, the predicted classes from the SVM algorithm underestimates the observations on the border of the decision region. One option could be to decide a posteriori distance to the hyperplane/decision region, based on the anomaly scores originating from the SVM under the assumption of the percent anomalies in the data set, where one manually classifies the instances that are close to the boundary of the decision region as anomalous. This may explain why the SVM performance is poor in this thesis where the data set is highly imbalanced consisting of only 0.172% of the minority class. Examining the anomaly scores, there is an evidence that the SVM appoints smaller anomaly scores for the observations at the border, which is consistent with theory. Moreover, the anomalous instances often have very similar characteristics as the normal instances, which makes it possible for them to be in the same region as the normal instances.

Another important aspect from the result regarding the performance of the SVM, which also holds for all of the models considered, is that the data was pre-processed by PCA. This means that, the PCA processing may have removed qualitative information regarding the anomalous instances that could make the two classes more separable in terms of their characteristics, since the PCA makes dimension reduction and thus implies information loss. Applying another kernel function than the Gaussian radial basis function could be a choice for better model performance of the SVM. However, that was not further investigated.

With the assumption of 0.2% anomalies in the training set, the IF was outperformed by the L2 and MGPD by PR curve and the AUPRC metric. On the other hand, the IF outperforms the L2, L-Sup, SVM and MGPD under the assumption of 1% anomalies in the training set and hence seem sensitive to the assumption of the anomaly rate. Moreover, this also leads to significant amount of false positives, since the anomalous rate is usually far below 1%, in fact, it is often below 0.1%. The poor outcome of the IF model, is most likely due to the fact of the PCA processing of the data set and thus that the profile of the anomalous instances becomes similar to the normal ones, due to information loss. The negative impact of the PCA also affects the other models in the same fashion. As mentioned in section 3.5, the IF takes advantage of the anomalous characteristics and their minority within the data set and therefore with the information loss due to PCA, the characteristics of anomalous instances might be weaker than in the raw original data set. This will thus lead to difficulties when the IF tries to separate the observations into the external

nodes i.e., both of the classes will have similar total path length due to similar values.

The most surprising outcome of the benchmarking are the results coming from the L2 and L-Sup. When considering the L2 under the assumption of 0.2% anomalies in the training set for the IF and SVM, L2 outperforms the IF, SVM and MGPD models. The L-Sup is slightly outperformed by the IF, see Table 5.4. These results are of course captivating, since the L2 and L-Sup are very simple anomaly scores obtained by standardizing the data set and then computing the L2 and L-Sup.

As for the MGPD, it is outperforming both of two unsupervised algorithms; IF and SVM, when assuming 0.2% anomalies in the training set by the AUPRC metric, while it is slightly under performing against the L2. When assuming 1% anomalies in the training set for the IF and SVM, it is outperformed by the IF algorithm, see Table 5.4. However, the assumed amount of anomalies in that comparison is very different from the true amount of anomalies. Further, when all of the models are trained and tested with respect to the training set, the MGPD outperforms all of the models, except for the supervised FFCNN, see Table A.2. High values of the transformed standard exponential anomaly threshold excesses leads to high scores of the negative log-likelihood, indicating anomalous instances. However, there are plenty of anomalous instances which have similar anomaly scores as the normal instances. Since the MGPD builds on the L2, L-Sup and IF anomaly scores, this affects the quality of the predictions from the results obtained with the MGPD model. Further, one must note that the fitting of the 3-dimensional MGPD:s five parameters i.e., α_i and β_j for $i = 1, 2, 3$ and $j = 1, 2$ is done by means of an extremely time consuming minimization procedure, the Nelder-Mead method. It remains to be studied to what extent poorly estimated parameters can effect the MGPD method. When the minimization problem was executed with different maximum iterations, this resulted in quiet different parameter estimates and thus also quiet different negative log-likelihoods for each fitting and testing.

In conclusion, unsupervised anomaly detection is hard and the research within this area is scarce, so far. Even well developed unsupervised algorithms such as the IF and SVM have difficulties predicting anomalous instances in highly imbalanced data sets. The MGPD outperforms both of the unsupervised ML algorithms in terms of the IF and SVM with respect to 0.2% anomalous instances in the training set, which is the true amount of anomalous instances in the credit card transaction data set. Moreover, the MGPD is also outperforming the unsupervised models when trained and tested with respect to the same data set. The results from this thesis indicate that unsupervised anomaly detection with the MGPD is well worth further study. In particular, if the research can be conducted with higher dimensional data that has not been pre-processed by any means.

6.3 Future Research

There are many questions within unsupervised anomaly detection and ML prediction that needs further studies. First, concerning to supervised learning, the FFCNN; it would be of interest to see how the model originating from the training set will perform when injecting new anomalous instances from a different distribution into the test set. Second, *oversampling* and *undersampling* methods could be considered and see how the supervised FFCNN would perform during those circumstances. Third, the anomaly scores from the SVM can be further examined and see how well these in fact describe anomalous observations at the border of the decision region and also consider other kernel functions. Fourth, extend the MGPD dimension i.e., $d > 3$, together with new anomaly scores obtained from; $L1$, $Minkowski$, $Cosine - Distance$, $Manhattan - Distance$ and $Mahalanobis - Distance$ space metrics and/or their inverse values which measure how close two observations are to each other. Fifth, examine how the minimization problem for the MGPD can be improved, by for instance trying out other minimization methods and then also try to understand how the estimated parameters of the MGPD effects the predictions. Finally, proceed new research with complete and un-processed data set with all of its attributes known would be a very important contribution to research in this area.

Bibliography

[1] The Association of Certified Fraud Examiners, 2014. Report to the nations on occupational fraud and abuse. https://www.acfe.com/rttn-introduction.aspx. Last Accessed 2022-01-25.

[2] Stephen Pedneault, 2009. Techniques and Strategies for understanding Fraud

[3] R.J. Bolton and D.J. Hand, 2002. Statistical fraud detection: A review. Statistical Science.

[4] The Association of Certified Fraud Examiners. What Is Fraud? https://www.acfe.com/fraud-101.aspx. Last Accessed 2022-01-25.

[5] P. Tiwari, S. Mehta, N. Sakhuja, J. Kumar, and A. K. Singh, 2021. Statistical fraud detection:Credit card fraud detection using machine learning: A study, https://arxiv.org/abs/2108.10005. Last Accessed 2021-01-25.

[6] O'Sullivan, Arthur; Steven M. Sheffrin, 2003. Economics: Principles in action. Upper Saddle River, New Jersey: Pearson Prentice Hall. ISBN 0-13-063085-3.

[7] V. N. Dornadula and S. Geetha, 2019. Credit card fraud detection using machine learning algorithms, Procedia Compputer Science **165**, 631 (2019), 2nd International Conference on Recent Trends in Advanced Computing ICRTAC-DISRUP-TIV INNOVATION, November 11-12, 2019.

[8] Raymond Anderson, 2007.The Credit Scoring Toolkit: Theory and Practice for Retail Credit Risk Management and Decision Automation. Statistical Science. Oxford University Press.

[9] Holly D. Johnson, 2021. Biggest credit card scams to look out for in 2022.https://www.bankrate.com/finance/credit-cards/biggest-credit-card-scams/. Last Accessed 2021-01-25.

[10] Brex. What is malware? - Find out how it works, and how to protect yourself from fraud as a result of malware.https://www.brex.com/learn/fraud-security/what-is-malware/. Last Accessed 2021-01-25.

[11] Bart Baesens, Veronique Van Vlasselaer, and Wouter Verbeke, 2015. Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection. John Wiley & Sons.

[12] The Nilson Report, 2019. Payment Card Fraud Losses Reach $27.85 Billion. Annual Fraud Statistics Released by The Nilson Report. https://www.prnewswire.com/news-releases/payment-card-fraud-losses-reach-27-85-billion-300963232.html. Last Accessed 2021-01-25.

[13] Richard J Bolton and David J Hand, 2001. Unsupervised profiling methods for fraud detection. Credit Scoring and Credit Control VII.

[14] Linda Delamaire, HAH Abdou, and John Pointon, 2009. Credit card fraud and detection techniques: a review. Banks and Bank Systems, 4(2).

[15] European Central Bank. New report on card fraud shows online fraud increased in 2012. https://www.ecb.europa.eu/press/pr/date/2014/html/pr140225.en.html. Last Accessed 2021-01-25.

[16] Rodney T Stamler, Hans J Marschdorf, Mario Possamai, 2014. Fraud Prevention and Detection. Routledge Member of the Taylor and Francis Group. ISBN: 9781466554542

[17] Piotr Juszczak, Niall M Adams, David J Hand, Christopher Whitrow, and David J Weston, 2008. Off-the-peg and bespoke classifiers for fraud detection. Computational Statistics Data Analysis, 52(9): 4521–4532.

[18] Andrea Dal Pozzolo, 2015. Adaptive Machine Learning for Credit Card Fraud Detection. https://dalpozz.github.io/static/pdf/Dalpozzolo2015PhD.pdf. Last Accessed 2021-01-26.

[19] Rudolph Russel, 2018. Machine Learning: Step-by-step guide to implement machine learning algorithms with python.

[20] Ethem Alpaydin, 2010. Introduction to Machine Learning:Adaptive Computation and Machine Learning.The MIT Press Cambridge, Massachusetts London, England. ISBN 978-0-262-01243-0

[21] Kishan G. Mehrotra Chilukuri K. Mohan HuaMing Huang, 2017.Anomaly Detection Principles and Algorithms. Springer. ISBN 978-3-319-67524-4

[22] M. Müller, 2007. Dynamic time warping. Information Retrieval for Music and Motion, pp. 69–84.

[23] Amoroso, Luigi, 1938. "VILFREDO PARETO". Econometrica (Pre-1986); Jan 1938; 6, 1; ProQuest. 6.

[24] Pareto Vilfredo, 1898. "Cours d'economie politique". Journal of Political Economy. 6. doi:10.1086/250536. https://www.journals.uchicago.edu/doi/10.1086/250536. Last Accessed 2021-01-29.

[25] S. Coles, 2001. An introduction to statistical modeling of extreme values, volume 208. Springer.

[26] R. L. Smith, 2001. Threshold methods for sample extremes. In Statistical extremes and appli- cations, pages 621–638. Springer.

[27] McNeil, A. J., Frey, R., and Embrechts, P, 2015. Extreme Values, Regular Variation, and Point Processes, Princeton, NJ: Princeton University Press.

[28] Anna Kiriliouk, Holger Rootzén, Johan Segers Jennifer L. Wadsworth, 2019. Peaks Over Thresholds Modeling With Multivariate Generalized Pareto Distributions, Technometrics, 61:1, 123-135, DOI: 10.1080/00401706.2018.1462738.https://www.tandfonline.com/doi/full/10.1080/00401706.2018.1462738. Last Accessed 2021-01-31.

[29] Rootzén, H., and Tajvidi, N., 2006, "Multivariate Generalized Pareto Distributions," Bernoulli, 12, 917–930

[30] Rootzén, H., and Thomas, M., 2019. Real-time prediction of severe influenza epidemics using Extreme Value Statistics. https://arxiv.org/abs/1910.10788. Last Accessed 2021-02-02.

[31] H. Rootzén, J. Segers, and J. L. Wadsworth, 201.8 Multivariate peaks over thresholds models. Extremes, 21(1):115–145. https://research.chalmers.se/publication/246023/file/246023_Fulltext.pdf. Last Accessed 2021-02-02.

[32] R. Michel, 2009. Parametric estimation procedures in multivariate generalized Pareto models. Scandinavian journal of statistics, 36(1):60–75.

[33] E. Brodin and H. Rootzén, 2009. Univariate and bivariate GPD methods for predicting extreme wind storm losses. Insurance: Mathematics and Economics, 44(3):345–356.

[34] Liu, F. T., K. M. Ting, and Z. Zhou. "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining. Pisa, Italy, pp. 413-422.https://ieeexplore.ieee.org/document/4781136. Last Accessed 2022-02-04.

[35] P.J.RousseeuwandK.V.Driessen, 1999. A fast algorithm for the minimum covariance determinant estimator. Technometrics, 41(3):212–223.

[36] Cortes, Corinna; Vladimir Vapnik, 1995."Support-Vector Networks". Machine Learning. 20 (3): 273–297. doi:10.1007/BF00994018

[37] Scholkopf, B., J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, 2001. "Estimating the Support of a High-Dimensional Distribution." Neural Comput., Vol. 13, Number 7,pp. 1443–1471.

[38] M. Bernhard, 2021. Machine learning with neural networks An introduction for scientists and engineers.

[39] T. Saito and M. Rehmsmeier, 2015.The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PloS one, 10(3): e0118432.

[40] M. Hossin and S. M.N, 2015. A review on evaluation metrics for data classification evaluations, International Journal of Data Mining Knowledge Management Process 5, 01.

[41] T. Fawcett, 2004. "ROC Graphs: Notes and Practical Considerations for Researchers". HP Laboratories.

[42] Université Libre de Bruxelle, 2013. Credit Card Fraud Detection - Anonymized credit card transactions labeled as fraudulent or genuine. `https://www.kaggle.com/mlg-ulb/creditcardfraud`. Last accessed 2021-12-20.

[43] René Vidal Yi Ma S. Shankar Sastry, 2016. Generalized Principal Component Analysis Springer-Verlag New York.

[44] Nocedal, J. and S. J. Wright. Numerical Optimization, 2nd ed., New York: Springer, 2006.

[45] Glorot, Xavier, and Yoshua Bengio, 2010. "Understanding the difficulty of training deep feedforward neural networks." In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249–256.

[46] Christopher M. Bishop, 2011. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer.

[47] C. Anderson, D. Carter, and P. Cotton, 2001. Wave climate variability and impact on offshore design extremes. Technical report. `https://journals.ametsoc.org/view/journals/clim/19/21/jcli3918.1.xml`.

[48] M. P. Wand and M. C. Jones, 1995. Kernel smoothing. Monographs on statistics and applied probability. Chapman Hall/CRC, Boca Raton (Fla.), London, New York. ISBN 0-412-55270-1

[49] A. MacDonald, C. J. Scarrott, D. Lee, B. Darlow, M. Reale, and G. Russell, 2011. A flexible extreme value mixture model. Computational Statistics Data Analysis, 55(6):2137– 2157.

[50] Parzen, E., 1962. "On Estimation of a Probability Density Function and Mode". The Annals of Mathematical Statistics. 33 (3): 1065–1076.

[51] Train neural network classification model in MATLAB https://www.mathworks.com/help/stats/fitcnet.html.

[52] IsolationForest - Isolation forest for anomaly detection https://www.mathworks.com/help/stats/isolationforest.html.

[53] Train support vector machine (SVM) in MATLAB https://www.mathworks.com/help/stats/fitcsvm.html.

[54] Rootzén, H., and Thomas, M., 2019. Real-time prediction of severe influenza epidemics using Extreme Value Statistics MGPD code https://github.com/maudmhthomas/predict_extremeinfluenza.

[55] PYMNTS, 2019. JPMorgan Employs Machine Learning For Expense Reports https://www.pymnts.com/innovation/2019/jpmorgan-employs-machine-learning-for-expense-reports/. Last accessed 12 January 2022

A

Proofs, Threshold Selection, Goodness of Fit Diagnostics and Performance comparison Metrics

This appendix provides some proofs for the theoretical concepts that have been used throughout this thesis, threshold selection plots, goodness of fit diagnostics, comparison metrics such as the AUPRC and AUROC values, ROC and PR curves and also confusion matrices. In particular, performance metrics when the models have been tested on the training data set can be found.

A.1 Proofs

Theorem A.1.1 (Univariate Generalized Pareto Distribution)**.**

Proof. Let X have distribution function F. By the assumption of **Theorem 3.1** in [25], for large enough n,

$$F^n(z) \approx exp\Bigg\{ -\left[1+\gamma\Big(\frac{z-\mu}{\sigma}\Big)\right]^{-\frac{1}{\gamma}}\Bigg\} \tag{A.1}$$

for some parameters $\mu, \sigma > 0$ and γ. Hence,

$$n \log F(z) \approx -\left[1+\gamma\frac{z-\mu}{\sigma}\right]^{-\frac{1}{\gamma}} \tag{A.2}$$

But for large values of z, a Taylor expansion implies that

$$\log F(z) \approx -\{1-F(z)\} \tag{A.3}$$

Substitution of (A.3) into (A.2), followed by algebraic rearrangement, gives

$$1-F(u) \approx \frac{1}{n}\left[1+\gamma\Big(\frac{u-\mu}{\sigma}\Big)\right]^{-\frac{1}{\gamma}} \tag{A.4}$$

for large u. Similarly, for $y > 0$,

$$1 - F(u+y) \approx \frac{1}{n}\left[1 + \gamma\Big(\frac{u+y-\mu}{\sigma}\Big)\right]^{-\frac{1}{\gamma}} \tag{A.5}$$

Hence,

$$\mathbb{P}[X > u+y | X > u] \approx \frac{n^{-1}[1+\gamma(u+y-\mu)/\sigma]^{-\frac{1}{\gamma}}}{n^{-1}[1+\gamma(u-\mu)/\sigma)]^{-\frac{1}{\gamma}}} \tag{A.6}$$

$$= \left[1 + \frac{\gamma(u+y-\mu)/\sigma}{1+\gamma(u-\mu)/\sigma}\right]^{-\frac{1}{\gamma}} \tag{A.7}$$

$$= \left[1 + \frac{\gamma y}{\tilde{\sigma}}\right]^{-\frac{1}{\gamma}} \tag{A.8}$$

where

$$\tilde{\sigma} = \sigma + \gamma(u-\mu), \tag{A.9}$$

as required [25]. And we are done. ■

Definition A.1.1 (Separable data set)**.** *A data set*

$$\{x_1, \ldots, x_n\} \tag{A.10}$$

is called separable if there exists some $w \in F$ such that $\langle w, x_i\rangle > 0$ for $i \in n$.

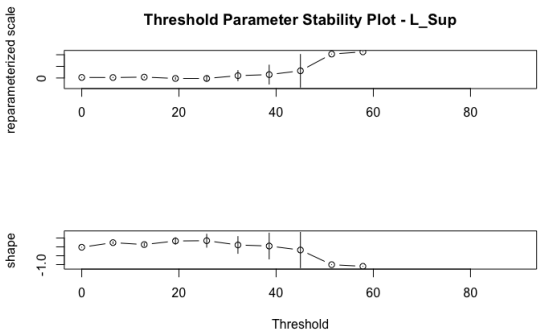
Proof. If one uses a Gaussian kernel RBF, see (3.19), then any data set $\{x_1, \ldots, x_n\}$ is separable after it has been mapped into a feature space F. To observe this, one notes that $k\langle x_i, x_j\rangle > 0$ for $\forall i, j$; thus, all inner products between the mapped patterns are positive, which implies that all patterns lie inside the same orthant. Moreover, since $k\langle x_i, x_i\rangle = 1$ for $\forall i$, they exhibit the same unit length. In fact, they are separable from the origin [37]. And we are done. ■

Definition A.1.2 (Supporting Hyperplane)**.** *If the data set of interest, see (A.10), is separable, then there exist a unique supporting hyperplane with the properties as follows; (I) It separates all data from the origin, and (II) Its distance to the origin is maximal among all such hyperplanes. For any $\rho > 0$, it is given by*

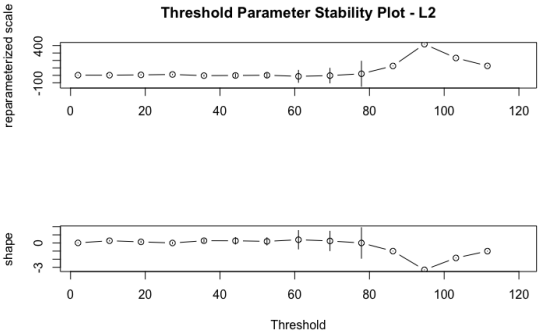
$$\min_{w \in F} \frac{1}{2}||w||^2 \quad \textit{subject to } \langle w, x_i\rangle,\ i \in n \tag{A.11}$$

Proof. Due to the separability, the convex hull of the data does not contain the origin. Moreover, the existence and uniqueness of the hyperplane then follow from the supporting hyperplane theorem, presented in (Bertsekas, 1995) [37]. Further, separability implies that there exists some threshold $\rho > 0$ and a normal $w \in F$ such that $\langle w, x_i\rangle \geq \rho$ for $i \in n$. Hence, of the hyperplane $\{z \in F : \langle w, z\rangle = \rho\}$ to the origin is $\frac{\rho}{||w||}$. Therefore, the optimal hyperplane is obtained by maximizing $||w||$ subject to these constraints, that is, by the solution of (A.11) [37]. And we are done. ■

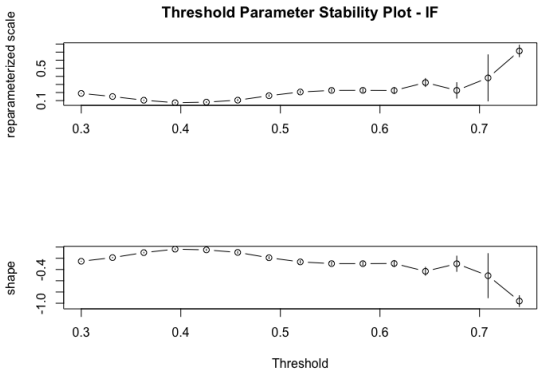
A.2 Threshold Selection - UGPD



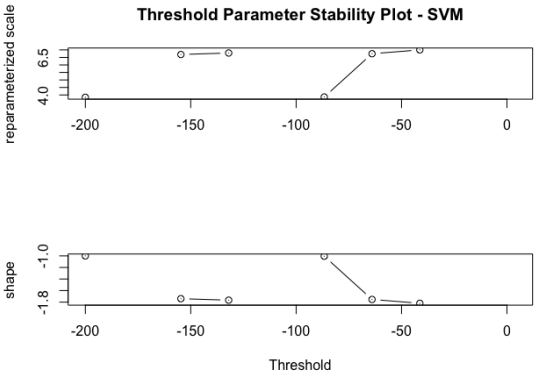
(a) TPSM for L-Sup wrt. UGPD.



(b) TPSM for L2 wrt. UGPD.

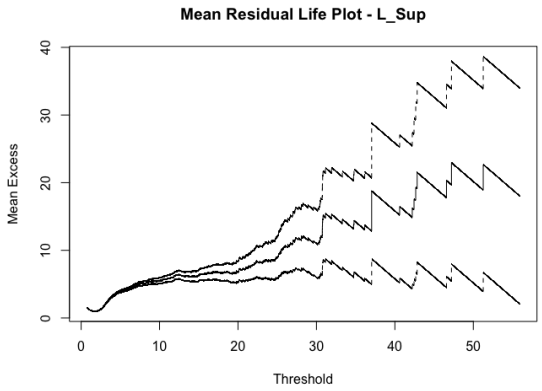


(c) TPSM for IF wrt. UGPD.

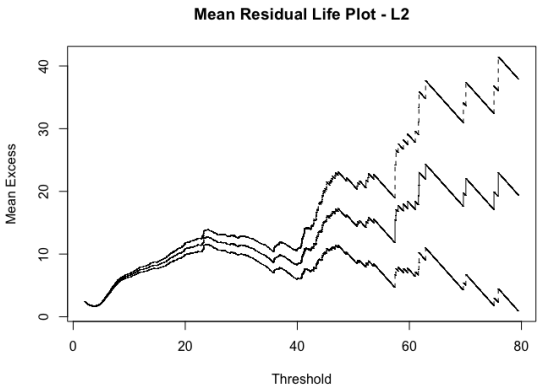


(d) TPSM for SVM wrt. UGPD.

Figure A.1: Threshold Parameter Stability Method (TPSM) applied on L-Sup, L2, IF and SVM anomaly scores with respect to the training set $\mathcal{T}_{Tr}$ and UGPD.

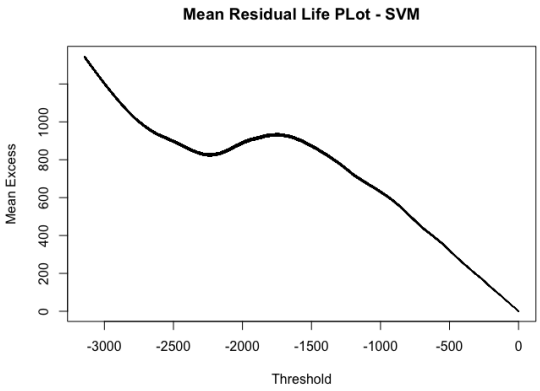


(a) MRLP for L-Sup wrt. UGPD.



(b) MRLP for L2 wrt. UGPD.

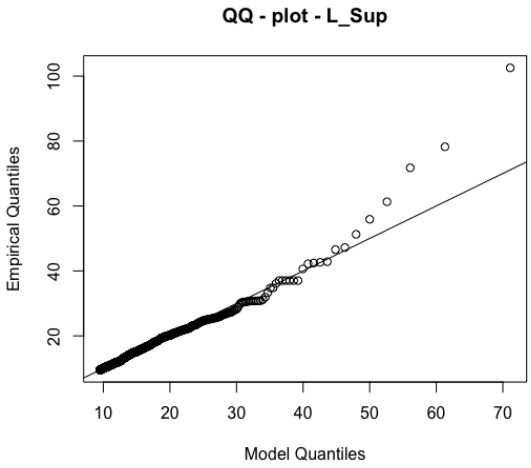
(c) MRLP for IF wrt. UGPD.



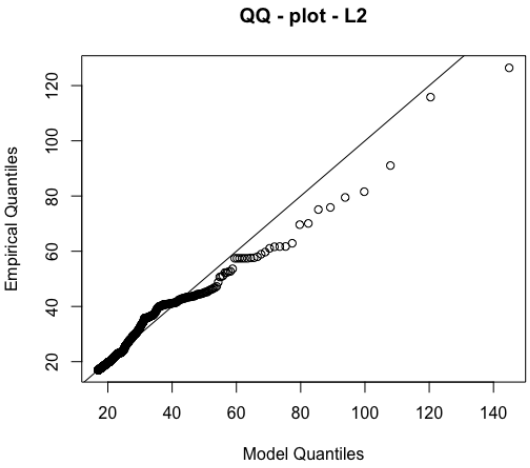
(d) MRLP for SVM wrt. UGPD.

Figure A.2: Mean Residual Life Plot (MRLP) applied on L-Sup, L2, IF and SVM anomaly scores with respect to the training set $\mathcal{T}_{Tr}$ and UGPD.

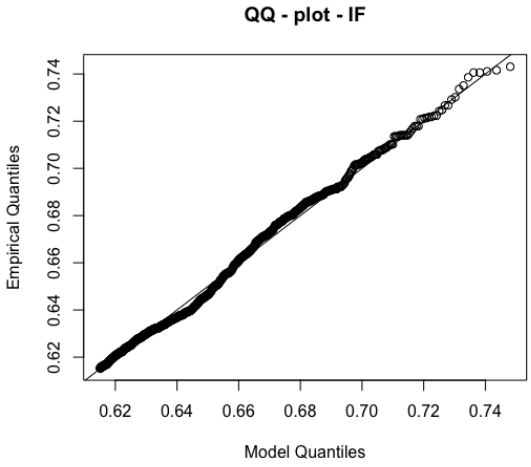
A.3 Goodness of Fit Diagnostics - Univariate Generalized Pareto Distribution



(a) QQ for L-Sup wrt. UGPD.



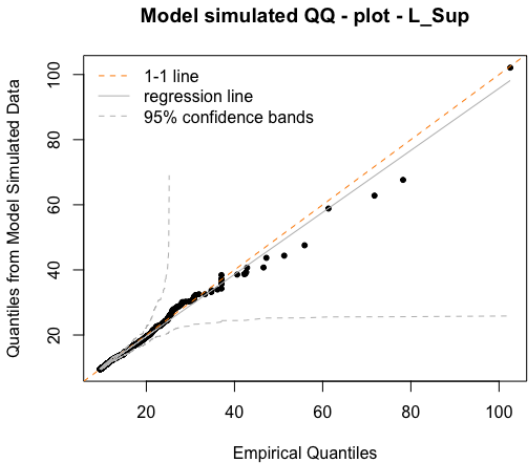
(b) QQ for L2 wrt. UGPD.



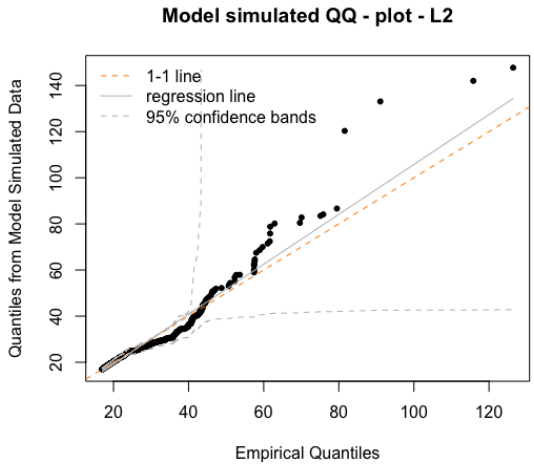
(c) QQ for IF wrt. UGPD.

(d) QQ for SVM wrt. UGPD.

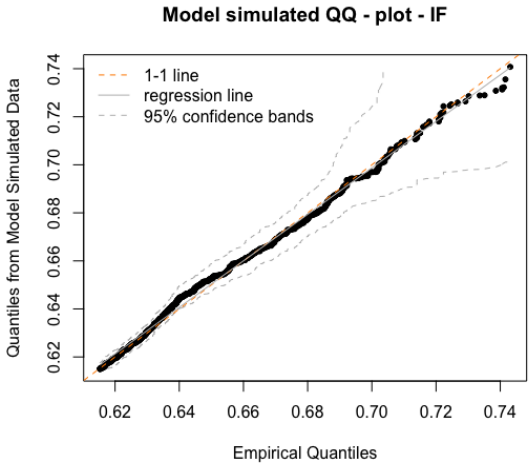
Figure A.3: Quantile-Quantile (QQ) plot for the fitted UGPD anomaly threshold excesses from L-Sup, L2, IF and SVM with respect to the training set $\mathcal{T}_{Tr}$.



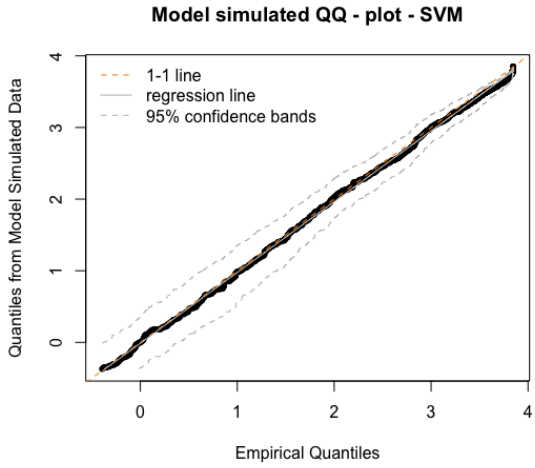
(a) QQ for L-Sup wrt. UGPD.



(b) QQ for L2 wrt. UGPD.

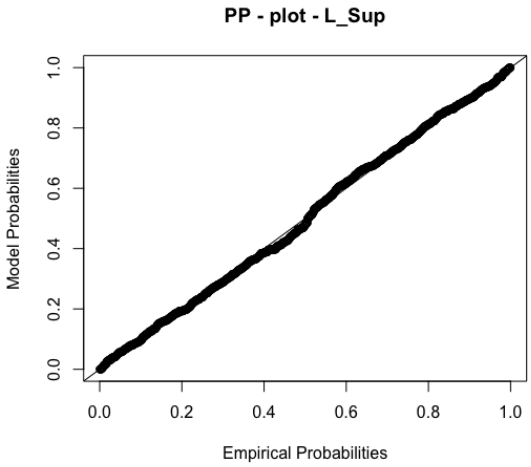


(c) QQ for IF wrt. UGPD.

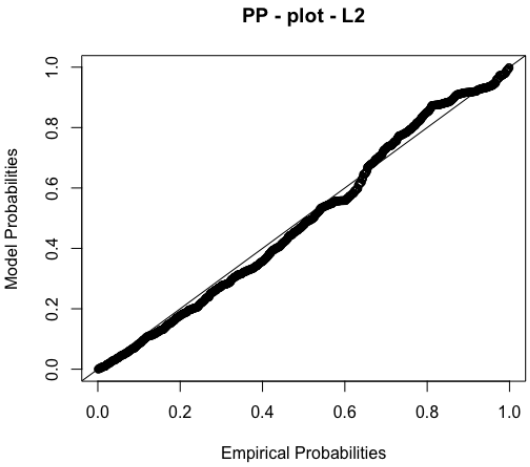


(d) QQ for SVM wrt. UGPD.

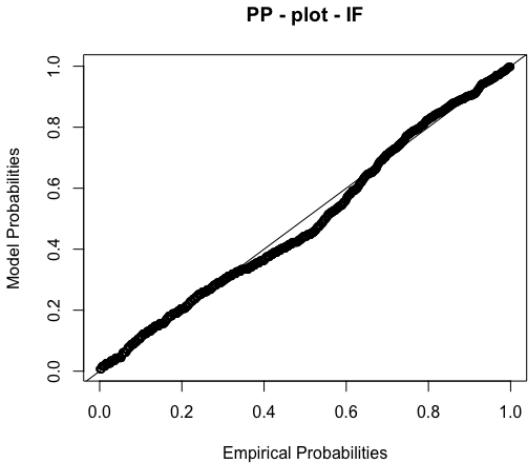
Figure A.4: Model simulated Quantile-Quantile (QQ) plot for the fitted UGPD anomaly threshold excesses from L-Sup, L2, IF and SVM with respect to the training set $\mathcal{T}_{Tr}$.



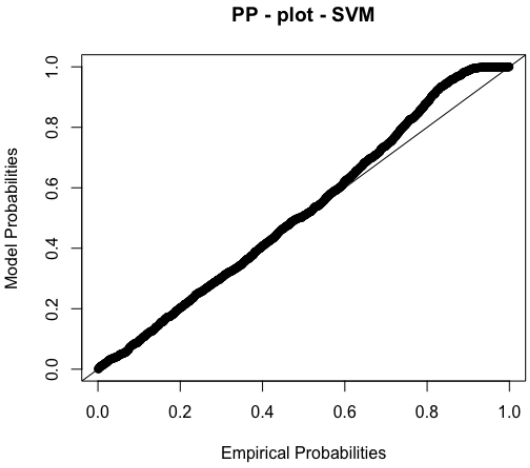
(a) PP for L-Sup wrt. UGPD.



(b) PP for L2 wrt. UGPD.

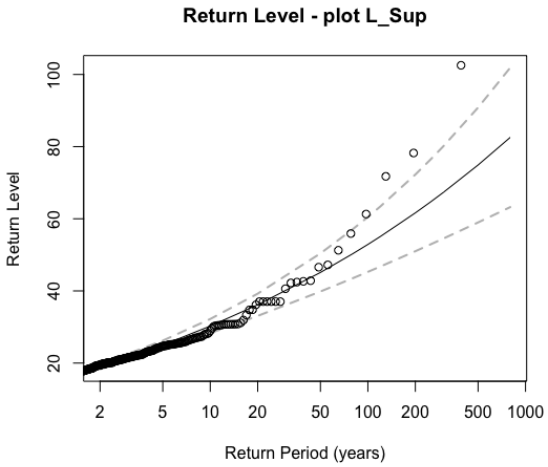


(c) PP for IF wrt. UGPD.

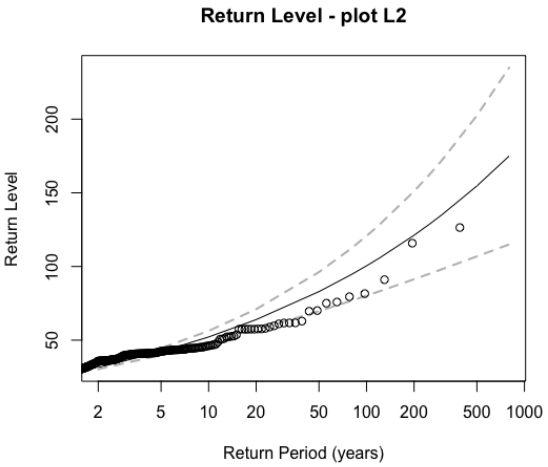


(d) PP for SVM wrt. UGPD.

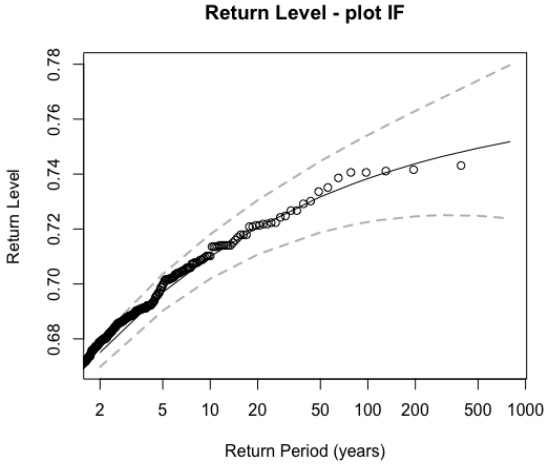
Figure A.5: Probability-Probability (PP) plot for the fitted UGPD anomaly threshold excesses from L-Sup, L2, IF and SVM with respect to the training set $\mathcal{T}_{Tr}$.



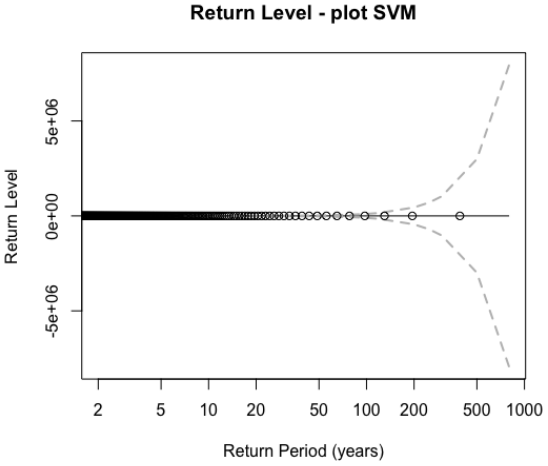
(a) RL for L-Sup wrt. UGPD.



(b) RL for L2 wrt. UGPD.

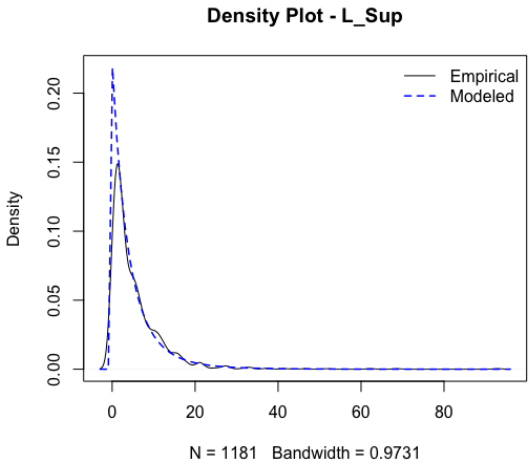


(c) RL for IF wrt. UGPD.

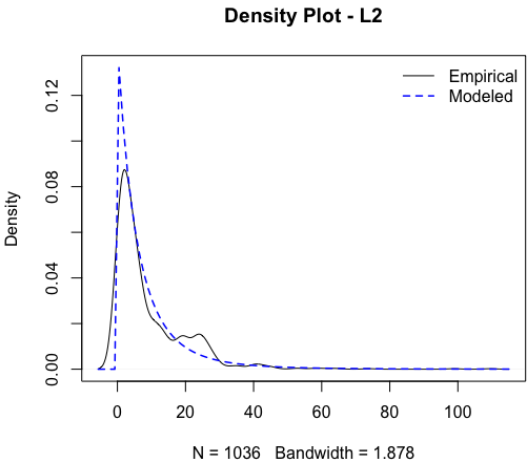


(d) RL for SVM wrt. UGPD.

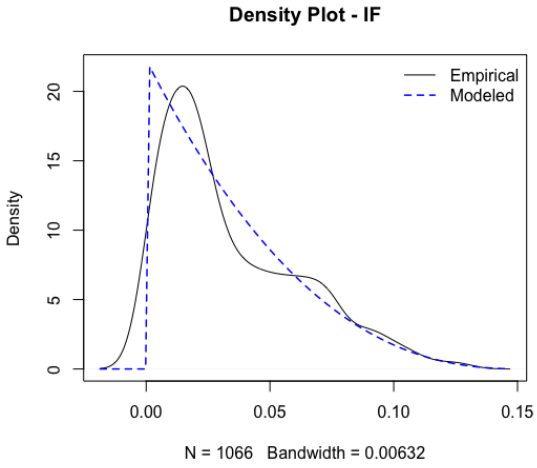
Figure A.6: Return Level (RL) Plot for the fitted UGPD anomaly threshold excesses from L-Sup, L2, IF and SVM with respect to the training set $\mathcal{T}_{Tr}$.



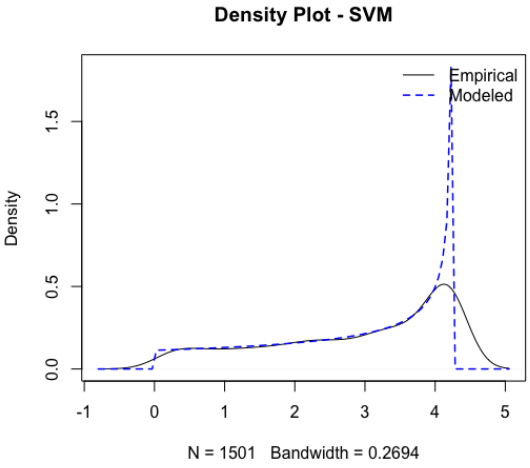
(a) KDP for L-Sup wrt. UGPD.



(b) KDP for L2 wrt. UGPD.



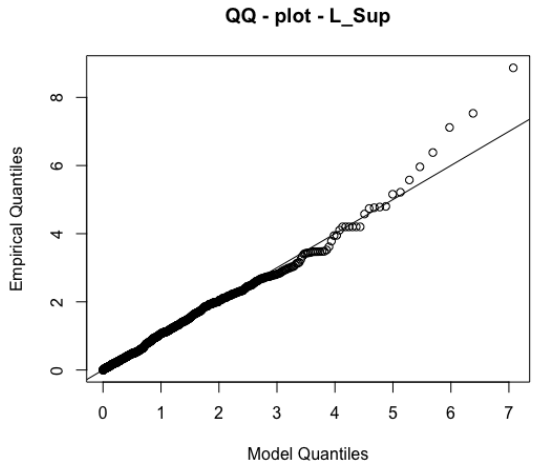
(c) KDP for IF wrt. UGPD.



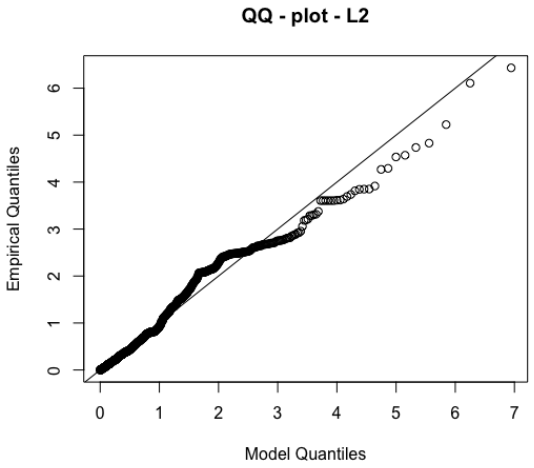
(d) KDP for SVM wrt. UGPD.

Figure A.7: Kernel Density Plot (KDP) for the fitted UGPD anomaly threshold excesses from L-Sup, L2, IF and SVM with respect to the training set $\mathcal{T}_{Tr}$.

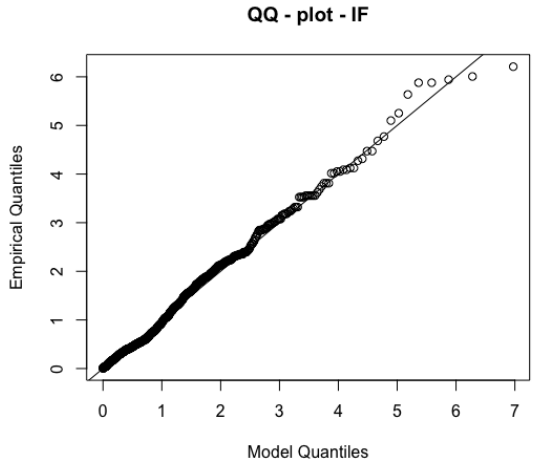
A.4 Goodness of Fit Diagnostics - Standard Exponential Distribution



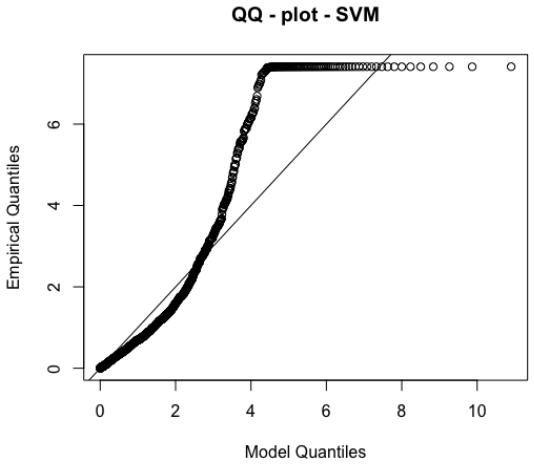
(a) QQ for L-Sup wrt. SED.



(b) QQ for L2 wrt. SED.



(c) QQ for IF wrt. SED.

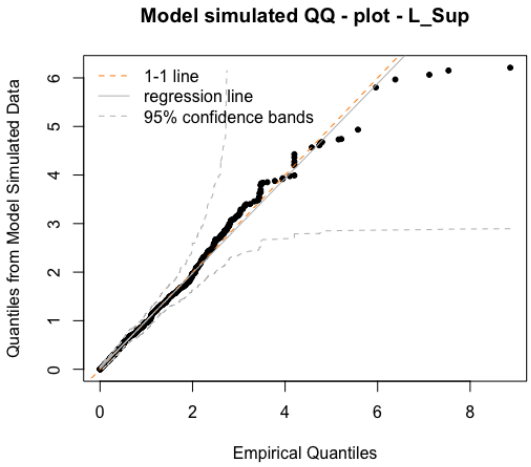


(d) QQ for SVM wrt. SED.

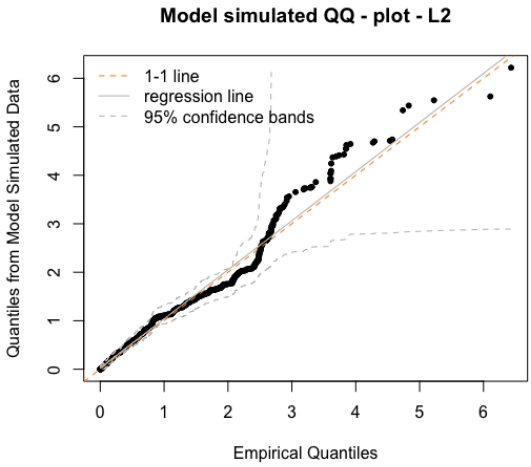
Figure A.8: Quantile-Quantile (QQ) plot for the transformed Standard Exponential Distributed (SED) anomaly threshold excesses of L-Sup, L2, IF and SVM.

Table A.1: *Estimated scale parameter $\hat{\beta}$ for the transformed standard exponential anomaly threshold excesses of; L-Sup, L2, IF and SVM. The values in brackets are the estimated standard error.*

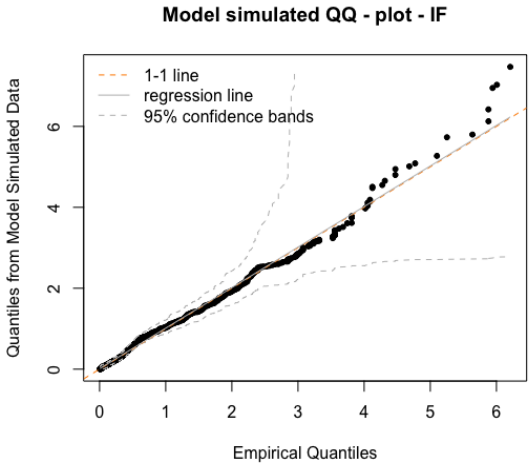
Model	$\hat{\beta}$
L-Sup	0.9999(0.029)
L2	0.9999(0.031)
IF	1(0.030)
SVM	1.490414(0.038)



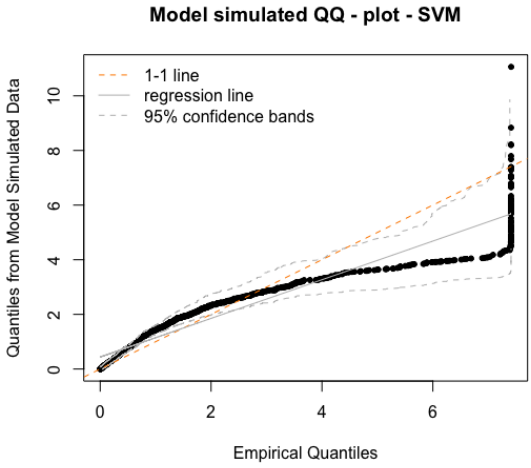
(a) QQ for L-Sup wrt. SED.



(b) QQ for L2 wrt. SED.

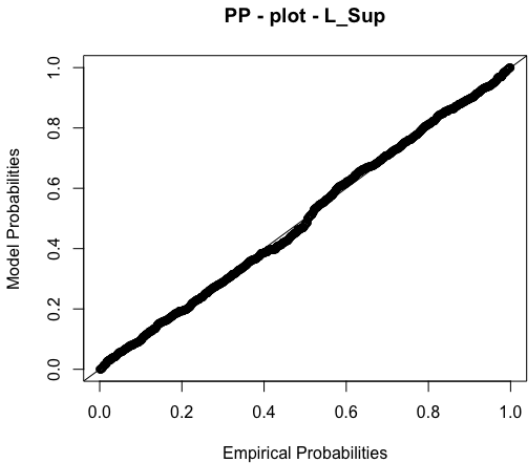


(c) QQ for IF wrt. SED.

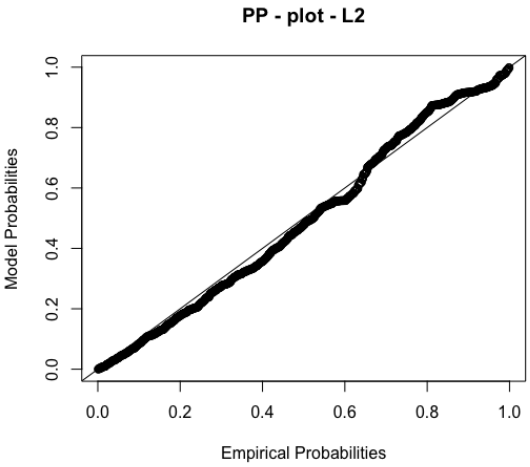


(d) QQ for SVM wrt. SED.

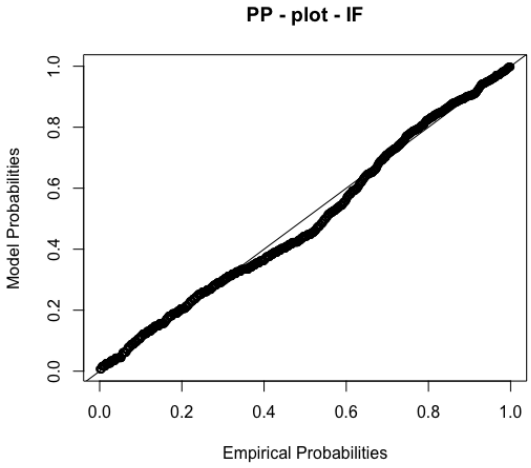
Figure A.9: Model Simulated Quantile-Quantile (QQ) plot for the transformed Standard Exponential Distributed (SED) anomaly threshold excesses of L-Sup, L2, IF and SVM.



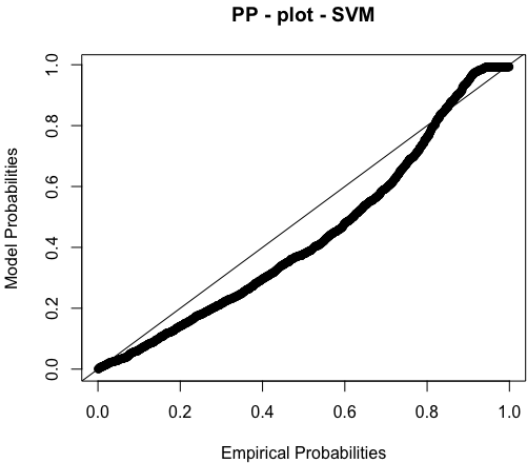
(a) PP for L-Sup wrt. SED.



(b) PP for L2 wrt. SED.

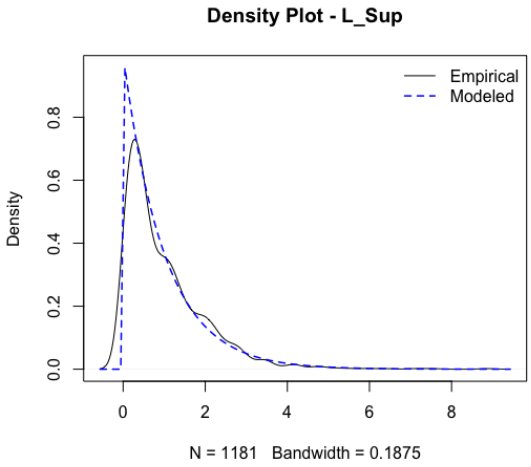


(c) PP for IF wrt. SED.

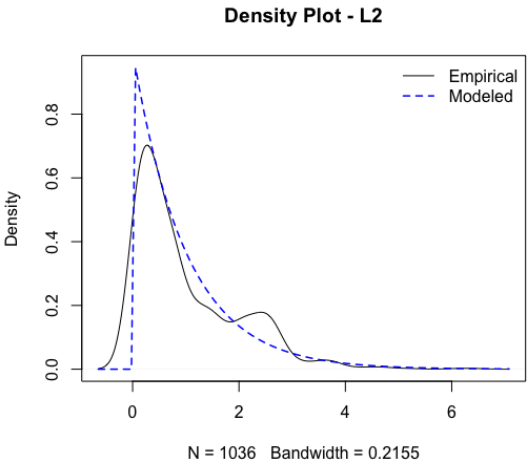


(d) PP for SVM wrt. SED.

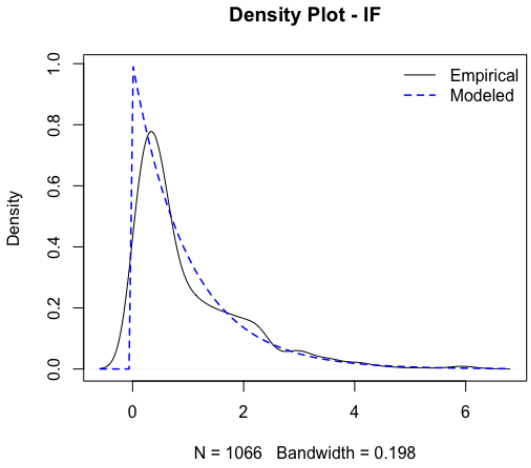
Figure A.10: Probability-Probability (PP) plot for the transformed Standard Exponential Distributed (SED) anomaly threshold excesses of L-Sup, L2, IF and SVM.



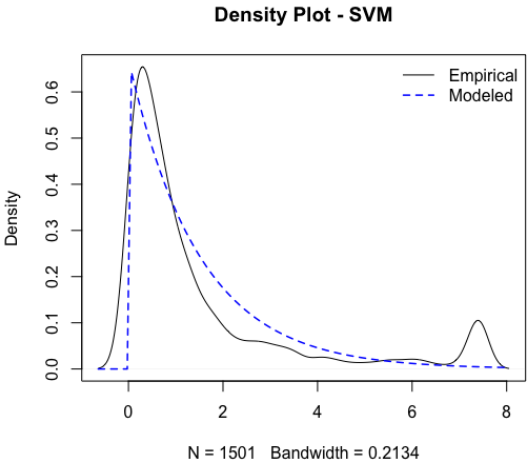
(a) KDP for L-Sup wrt. SED.



(b) KDP for L2 wrt. SED.



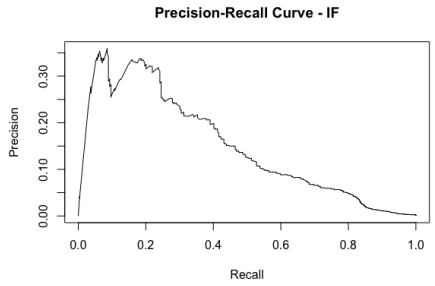
(c) KDP for IF wrt. SED.



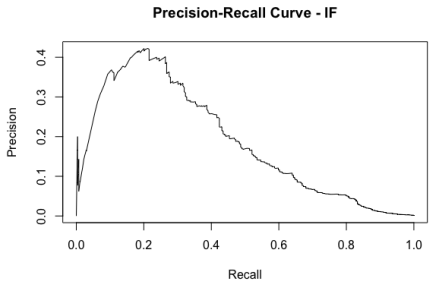
(d) KDP for SVM wrt. SED.

Figure A.11: Kernel Density Plot (KDP) for the transformed Standard Exponential Distributed (SED) anomaly threshold excesses of L-Sup, L2, IF and SVM.

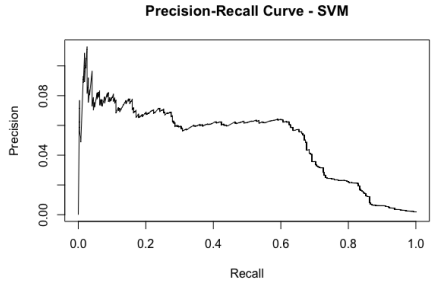
A.5 Performance Comparison Metrics



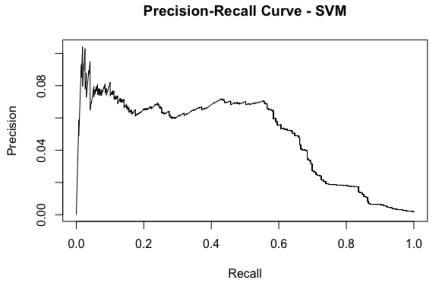
(a) PR Curve IF 0.2% $\mathcal{T}_{Tr}$.



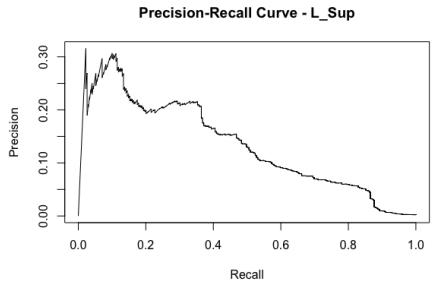
(b) PR Curve IF 1% $\mathcal{T}_{Tr}$.



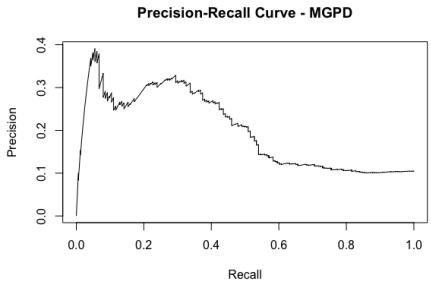
(c) PR Curve SVM 0.2% $\mathcal{T}_{Tr}$.



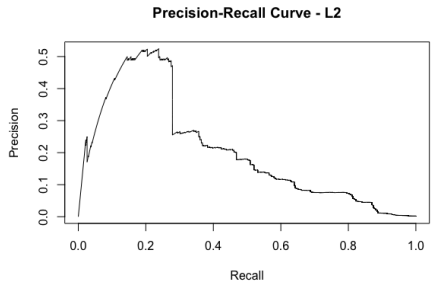
(d) PR Curve SVM 1% $\mathcal{T}_{Tr}$.



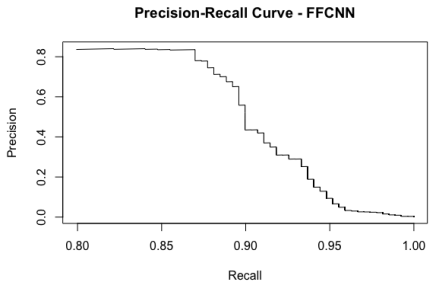
(e) PR Curve L-Sup $\mathcal{T}_{Tr}$.



(f) PR Curve MGPD $\mathcal{E}_{Trans_{Tr}}$.



(g) PR Curve L2 $\mathcal{T}_{Tr}$.

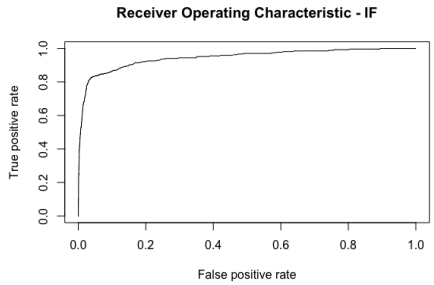


(h) PR Curve FFCNN $\mathcal{T}_{Tr}$.

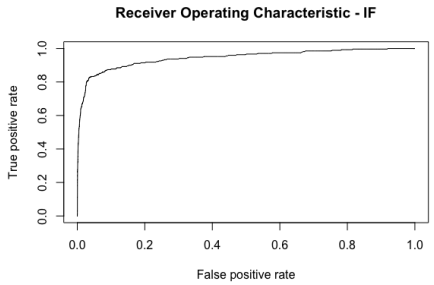
Figure A.12: Precision-Recall (PR) curve for the six models; IF, SVM, L-Sup, MGPD, L2 and FFCNN. The MGPD is fitted and tested on $\mathcal{E}_{Trans_{Tr}}$. The other models are trained and tested on $\mathcal{T}_{Tr}$. The SVM and IF are trained under the assumption of 0.2% and 1% anomalies respectively in $\mathcal{T}_{Tr}$. Note the different y-scale in the plots.

Table A.2: *AUPRC for each of the six models; MGPD, L-Sup, L2, IF, SVM and FFCNN shown in Figure A.12 above. Note that the FFCNN is supervised*

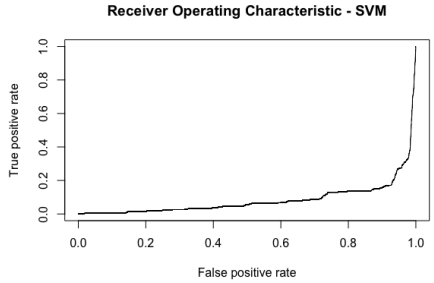
Model	AUPRC
MGPD	0.2110
L-Sup	0.1390
L2	0.2060
IF 0.2%	0.1520
IF 1.0%	0.1790
SVM 0.2%	0.0222
SVM 1.0%	0.0204
FFCNN	0.7670



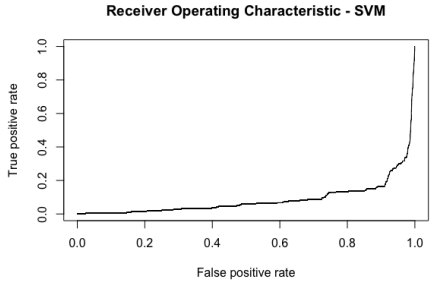
(a) ROC Curve IF 0.2% $\mathcal{T}_{Tr}$.



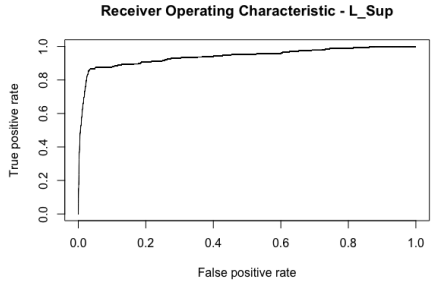
(b) ROC Curve IF 1% $\mathcal{T}_{Tr}$.



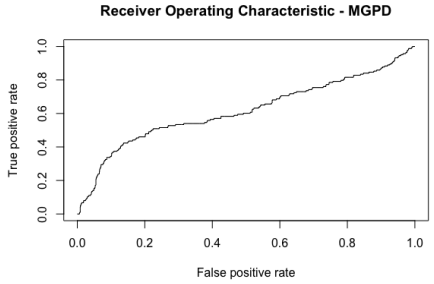
(c) ROC Curve SVM 0.2% $\mathcal{T}_{Tr}$.



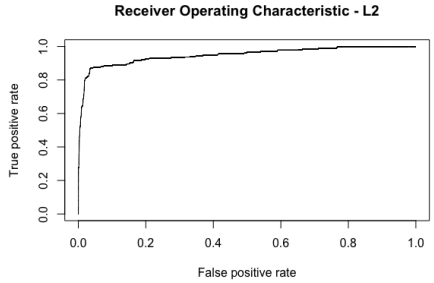
(d) ROC Curve SVM 1% $\mathcal{T}_{Tr}$.



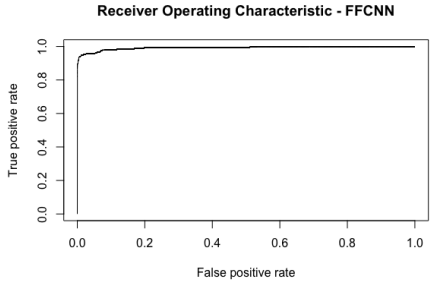
(e) ROC Curve L-Sup $\mathcal{T}_{Tr}$.



(f) ROC Curve MGPD $\mathcal{E}_{Trans_{Tr}}$.



(g) ROC Curve L2 $\mathcal{T}_{Tr}$.

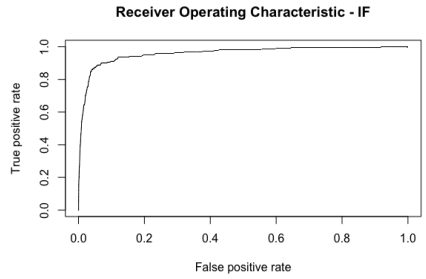


(h) ROC Curve FFCNN $\mathcal{T}_{Tr}$.

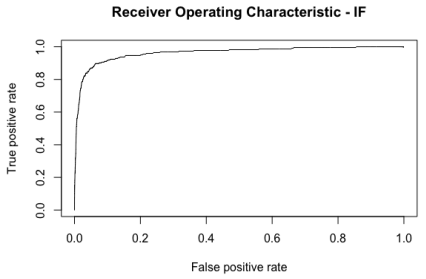
Figure A.13: Receiver Operating Characteristic (ROC) curve for each of the six models; IF, SVM, L-Sup, MGPD, L2 and FFCNN. The MGPD is fitted and tested on $\mathcal{E}_{Trans_{Tr}}$. The other models are trained and tested on $\mathcal{T}_{Tr}$. The SVM and IF are trained under the assumption of 0.2% and 1% anomalies respectively in $\mathcal{T}_{Tr}$.

Table A.3: *AUROC for each of the six models; MGPD, L-Sup, L2, IF, SVM and FFCNN shown in Figure A.13 above. Note that the FFCNN is supervised.*

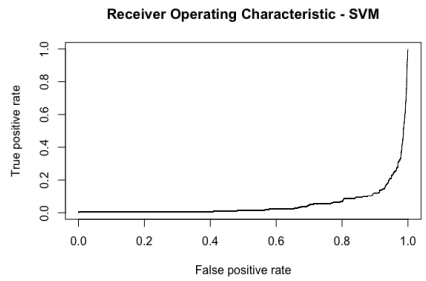
Model	AUROC
MGPD	0.7182
L-Sup	0.9402
L2	0.9485
IF 0.2%	0.9453
IF 1.0%	0.9455
SVM 0.2%	0.0858
SVM 1.0%	0.0834
FFCNN	0.9939



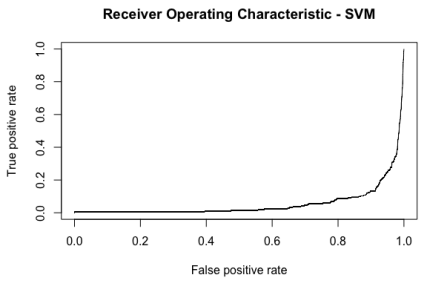
(a) ROC Curve IF 0.2% $\mathcal{T}_{Test}$.



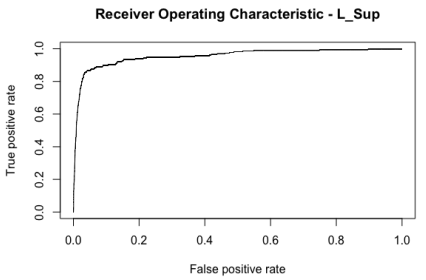
(b) ROC Curve IF 1% $\mathcal{T}_{Test}$.



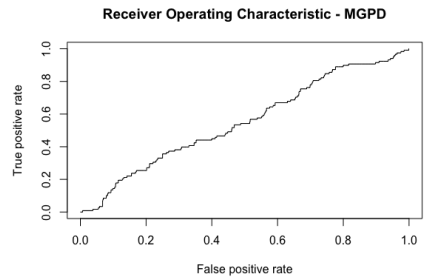
(c) ROC Curve 0.2% SVM $\mathcal{T}_{Test}$.



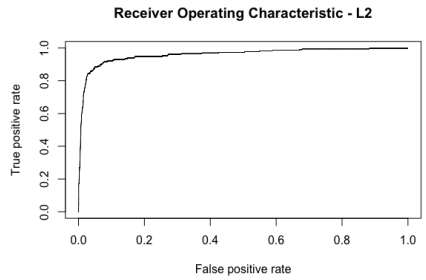
(d) ROC Curve 1% SVM $\mathcal{T}_{Test}$.



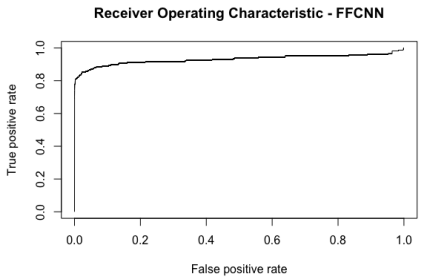
(e) ROC Curve L-Sup $\mathcal{T}_{Test}$.



(f) ROC Curve MGPD $\mathcal{E}_{Trans_{Test}}$.



(g) ROC Curve L2 $\mathcal{T}_{Test}$.

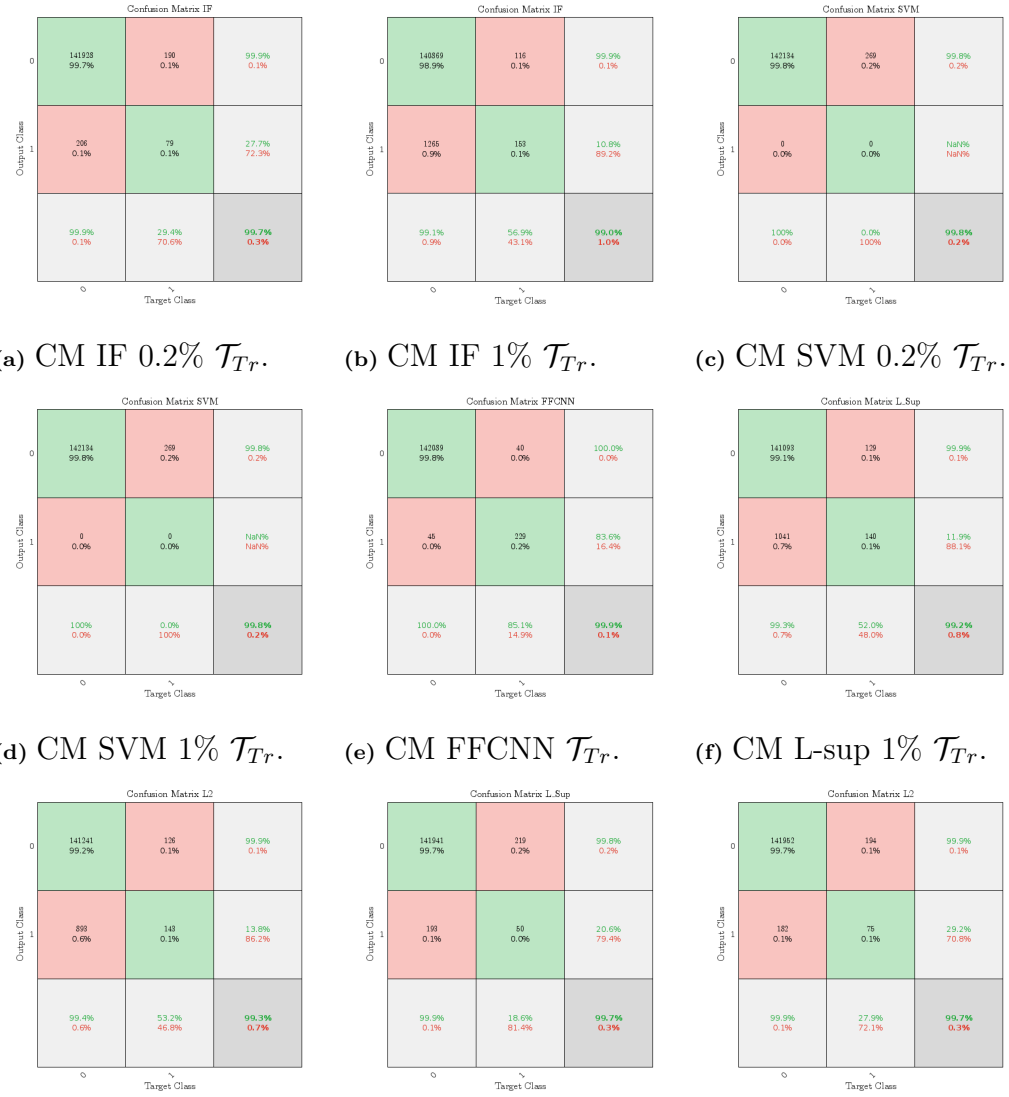


(h) ROC Curve FFCNN $\mathcal{T}_{Test}$.

Figure A.14: Receiver Operating Characteristic (ROC) curve for each of the six models; IF, SVM, L-Sup, MGPD, L2 and FFCNN. The MGPD is fitted on the training set $\mathcal{E}_{Trans_{Tr}}$ and tested on $\mathcal{E}_{Trans_{Test}}$. The other models are trained on $\mathcal{T}_{Tr}$ and tested on $\mathcal{T}_{Test}$ respectively. The SVM and IF are trained under the assumption of 0.2% and 1% anomalies respectively in $\mathcal{T}_{Tr}$.

Table A.4: *AUROC for each of the six models; L-Sup, L2, IF, FFCNN, SVM and MGPD in Figure A.14. Note that the FFCNN is supervised.*

Model	AUROC
MGPD	0.6578
L-Sup	0.9540
L2	0.9590
IF 0.2%	0.9600
IF 1.0%	0.9630
SVM 0.2%	0.0504
SVM 1.0%	0.0541
FFCNN	0.9290

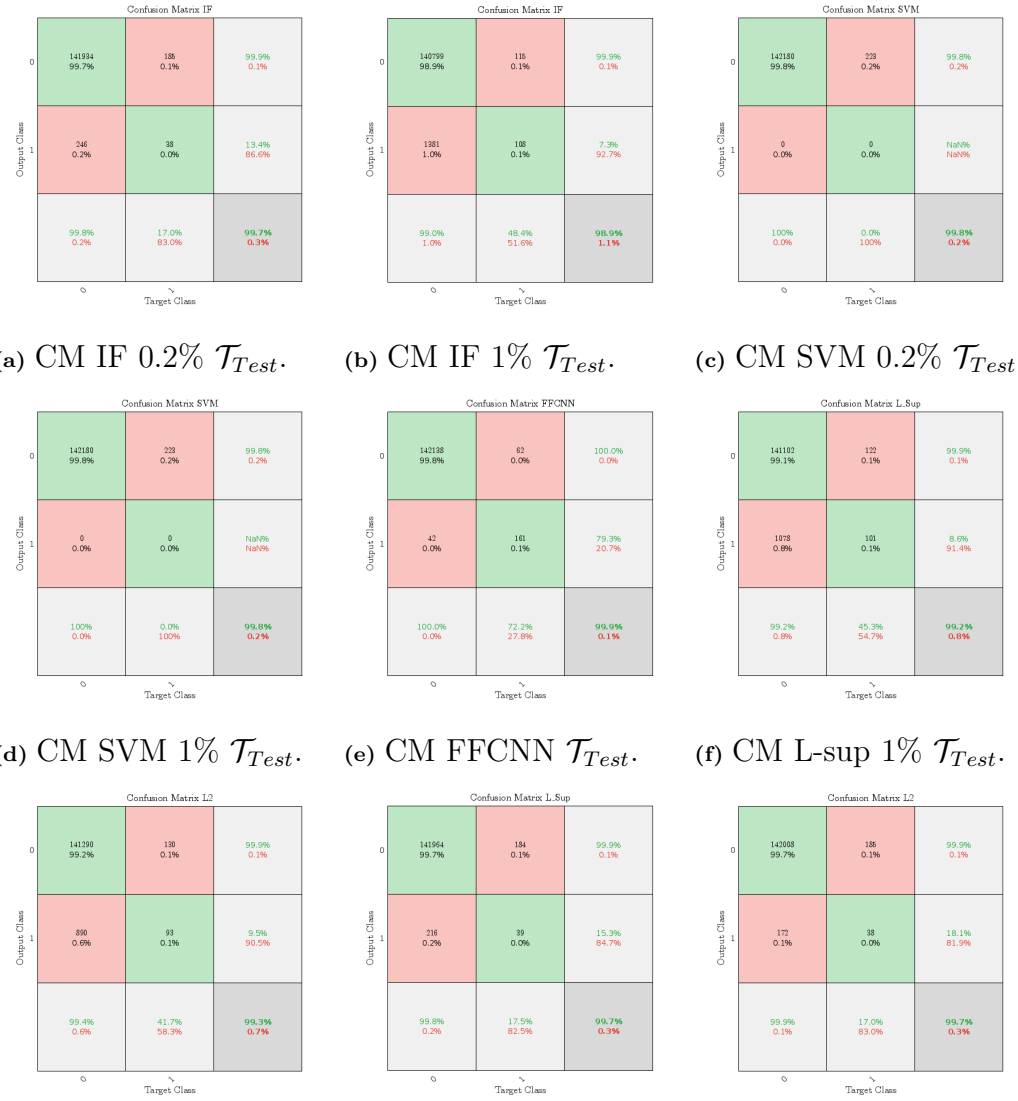


(a) CM IF 0.2% $\mathcal{T}_{Tr}$. **(b)** CM IF 1% $\mathcal{T}_{Tr}$. **(c)** CM SVM 0.2% $\mathcal{T}_{Tr}$.

(d) CM SVM 1% $\mathcal{T}_{Tr}$. **(e)** CM FFCNN $\mathcal{T}_{Tr}$. **(f)** CM L-sup 1% $\mathcal{T}_{Tr}$.

(g) CM L2 1% $\mathcal{T}_{Tr}$. **(h)** CM $u_{L-Sup} = 18$ L-sup $\mathcal{T}_{Tr}$. **(i)** CM $u_{L2} = 30$ L2 $\mathcal{T}_{Tr}$.

Figure A.15: Confusion matrices (CM) for; IF, SVM and FFCNN trained and tested on $\mathcal{T}_{Tr}$. Confusion matrices for L-Sup and L2 are obtained by directly testing on $\mathcal{T}_{Tr}$. The IF and SVM are trained under the assumption of 0.2% and 1% anomalies respectively. The confusion matrices for L-Sup and L2 are obtained by the thresholds determined in Table 5.1 and with $u_{L-Sup} = 18$ respectively $u_{L2} = 30$ to approximately obtain 210-260 anomalies on the half of the data set.



(a) CM IF 0.2% $\mathcal{T}_{Test}$. **(b)** CM IF 1% $\mathcal{T}_{Test}$. **(c)** CM SVM 0.2% $\mathcal{T}_{Test}$.

(d) CM SVM 1% $\mathcal{T}_{Test}$. **(e)** CM FFCNN $\mathcal{T}_{Test}$. **(f)** CM L-sup 1% $\mathcal{T}_{Test}$.

(g) CM L2 1% $\mathcal{T}_{Test}$. **(h)** CM $u_{L-Sup} = 18$ L-sup $\mathcal{T}_{Test}$. **(i)** CM $u_{L2} = 30$ L2 $\mathcal{T}_{Test}$.

Figure A.16: Confusion matrices (CM) for; IF, SVM and FFCNN trained on $\mathcal{T}_{Tr}$ and tested on $\mathcal{T}_{Test}$. Confusion matrices for L-Sup and L2 are obtained by directly testing on $\mathcal{T}_{Test}$. The IF and SVM are trained under the assumption of 0.2% and 1% anomalies respectively. The confusion matrices for L-Sup and L2 are obtained by the thresholds determined in Table 5.1 and with $u_{L-Sup} = 18$ respectively $u_{L2} = 30$ to approximately obtain 210-260 anomalies on the half of the data set.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY