UPPSALA
UNIVERSITET

IT 18 039
Examensarbete 15 hp
August 2018

Monitoring Financial Transactions:

Efficient Algorithms for Streaming Data

Miel Verkerken

Institutionen för informationsteknologi
Department of Information Technology

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Monitoring Financial Transactions: Efficient Algorithms for Streaming Data

Miel Verkerken

Real-time payment systems are vital to the emerging financial ecosystem. Fluent and reliable solutions demand algorithms that can flag transactions efficiently as fraudulent. Efficiency points both to computational challenges (able to process tens of thousands of transactions per second), as well as to high accuracy rates. Accuracy in this context translates as having high detection rates while raising few false alarms. This project studies such an algorithm, named FADO. Different variations of FADO are designed, implemented in python, and tested on publicly available data with anonymous features. We find that the so-called Extended FADO, a contribution from this thesis, was able to detect most of the frauds in this data, but without raising too many false alarms, nor sacrificing its computational efficiency. The best result was achieved through a combination of the suggested Extended FADO with a binary preprocessing technique. Those promising results ask for further confirmation in cooperation with a financial partner.

Handledare: Marcus Björk
Ämnesgranskare: Kristiaan Pelckmans
Examinator: Justin Pearson
IT 18 039
Tryckt av: Reprocentralen ITC

Contents

1. Introduction

The number of fraudulent transactions, for example terrorism funding or credit card fraud, is increasing fast every year. According to chargeback, a company providing a software solution for chargebacks in the e-commerce, the e-commerce in the US lost 6.7 billion dollars on fraud in 2016 [13]. Chargeback also said that in 2014 the amount of money lost due to false positives (normal transactions falsely blocked) was 118 billion dollars [17]. This shows that companies lose more money by checking the results of their fraud detection systems than it actually saves them. They do this out of fear of losing their image of being a trusted partner. On the other hand, the need is growing for real-time processing of transactions. This is due to the rise of platforms like 'Swish', that guarantee the clients that their transactions are getting processed immediately [4].

Putting the above facts together shows that it is essential to develop efficient streaming algorithms for fraud detection. Thereby it is crucial to catch as much as possible fraudulent transactions in real-time without getting a big number of false positives. Fraud detection is often referred to in the literature as anomaly detection, or in this specific case as an outlier detection. This paper describes an efficient implementation of the online fault detection algorithm (FADO) developed by Kristiaan Pelckmans [6], as well as different techniques to increase the detection performance. The advantage of FADO is that it can handle a high dimensional online stream of data efficiently in an unsupervised manner, unlike other clustering algorithms or supervised algorithms like neural networks. A motive to do research in this direction is insinuated by the future work in the paper "Identifying intended and unintended errors in financial transactions: a case study" by Zahra Rabiei [9].

Datasets with financial transactions are not publicly available as they contain sensitive information and could lead to a privacy breach, therefore the algorithm is tested on a simulated dataset and an anonymous actual dataset. This made it impossible to extract business information out of the findings but opens the path for future work.

1.1 Project Goal

This project will continue the exploration of suitable machine learning techniques for monitoring a stream of financial transactions. The aim is to identify transactions in this stream which are due to intended ('fraud') or unintended

('clumsy') errors. While the groundwork for this unsupervised learning task was already described in earlier work [15], extensions to handling efficiently a large number of financial transactions in a streaming fashion - without scaling up computational resources needed - prompts new challenges. This thesis adopts the theoretical method described at [6]. The focus lies on the implementation of an effective online clustering scheme for such financial application. Specifically, the design, running and testing an efficient implementation of FADO in Python. The resulting software is evaluated in terms of computational efficiency, the capability to handle high dimensional data as well as detection capability.

1.2 Delimitations

Rather than using sensitive actual bank data, the algorithm got tested with artificially generated data mimicking actual data and an actual dataset with anonymous features. Therefore it was impossible to translate the extracted insights in the data to useful business insights, or to use financial information from domain experts to improve the performance of the preprocessing. This requests future work to test FADO on real financial transactions.

1.3 Related Work

Citation [14] gives some related work regarding fault detection in streaming data.

Anomaly detection in general is a very broad topic. Excellent surveys on various topics are [16], [5].

A number of papers have discussed results of the data used in this project [10], [7], and [8].

2. Theory

2.1 Clustering

Well-known clustering algorithms such as K-means or Hierarchical clustering normally work in a batched manner. This means that it needs the whole dataset, or batches, available at once to decide if a transaction is fraudulent, which leads to the three following problems. First of all it is impossible to work in a real streaming way by processing transactions in real-time. This problem can be avoided by applying the clustering in a mini-batch mode. Then it is possible to wait until a predefined set of transactions is collected and run the adjusted version of the clustering algorithm on it. When the size of the mini-batch is well configured, the latency can be minimized but never eliminated. The second problem that arises is the memory requirements. The memory complexity is $\mathcal{O}(n^2)$ where n is the number of transactions [9]. This is a serious limitation to make it scalable in a realistic environment. The last problem with aforementioned clustering algorithms is that they are not efficient on high dimensional data. Therefore before financial data with ~30 features can be used, it needs feature engineering, feature selection and/or some other preprocessing technique, for example Principal Components Analysis (PCA).

2.2 Online Fault Detection

An algorithm is characterized as online when it receives and processes data sequentially in real-time. This is in contrast to batching algorithms where the whole dataset or batch is available at once. The fault detection, or in case of financial transactions, fraud detection, can be supervised or unsupervised. In a supervised setting, there is a training set available, that represents the problem, with labels that mark the fraudulent transactions. These labels can then be used to train the model that decides if transactions are fraudulent. This paper focuses on the unsupervised way of detecting fraud. The model then identifies, over time, what a normal transaction looks like and decides for every transaction if they are dissimilar enough to be flagged as fraud. The labels will only be used in the end to measure the performance.

2.2.1 FADO

FADO is an online fraud detection algorithm that is developed by Kristiaan Pelckmans with a guaranteed worst-case performance [6]. Algorithm 1

Algorithm 1: FADO(ε)

Initialize $w_0 = 0_n$.
for $t = 1, 2, \ldots$ **do**
(1) Receive transaction $y_t \in \mathbb{R}^n$.
(2) Raise an alarm if

$$\|y_t - w_{t-1}\|_2 \geq \varepsilon,$$

and set

$$v_t = \frac{y_t - w_{t-1}}{\|y_t - w_{t-1}\|_2} \in \mathbb{R}^n.$$

(3) If an alarm is raised, update

$$w_t = w_{t-1} + \gamma_t v_t.$$

Otherwise, set $w_t = w_{t-1}$.
end

presents it in pseudo code. Before starting the first iteration, the model gets initialized. The model $w_0 \in \mathbb{R}^n$ is a zero-vector with dimension equal to the number of features in the stream of transactions. For every iteration, a transaction is retrieved out of the stream and encoded as a vector (1). In (2) the Euclidean distance between the transaction and the model after the previous iteration is tested against ε, a predefined non-negative real number ($\varepsilon \in \mathbb{R}^+$). The model raises an alarm and gets updated with a factor $y_t v_t$ if the Euclidean distance was greater or equal to ε otherwise the model stays untouched. An alarm does not necessarily mean that the transaction is fraudulent. During the first iterations, the model needs to learn what a normal transaction is and will give a lot of false positives. This is called the learning phase. Just like all machine learning algorithms, FADO has to make an exploration-exploitation trade-off. This is achieved by every alarm. Each alarm will update the memory of the algorithm. An alarm can also mean that a normal transaction is flagged as fraud and that more exploration of the search environment was needed. After the learning phase, the model achieves a certain power of detection and the model will start exploiting in case of a decreasing learning-rate or stays exploring in case of a fixed learning-rate. The algorithm can be manually tuned by changing ε and γ, respectively the radius around w_t that defines the non-fraud area and the learning-rate. As proved in [6] a worst-case performance is guaranteed when the decreasing learning-rate is defined as in the next formula with C as a strictly positive constant and m_t the number of alarms until step t.

$$\gamma_t = \frac{C}{\sqrt{m_t}} \tag{2.1}$$

This vanilla version of FADO has a single genuine area and therefore this thesis makes the assumption that all normal transactions can be clustered together.

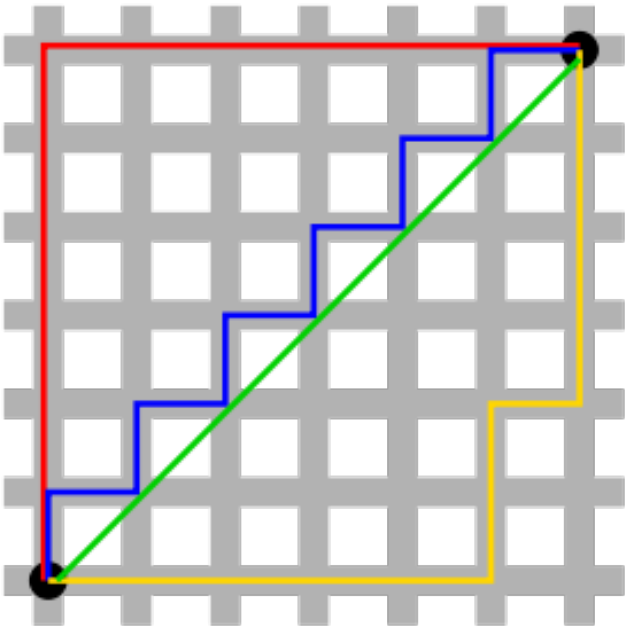

Figure 2.1. Difference between the Euclidean distance (green) and the Manhattan distance (red, blue or yellow).

If this is not the case an upgrade is necessary to work with multiple clusters. The vanilla version is the center of this thesis and later on an updated version is suggested to achieve better results with financial transactions.

2.2.2 Distance Function

FADO relies on a distance function to define if a transaction is not too deviating from the current model. The influence of this function is tested by trying out two options: the Euclidean distance and the Manhattan distance. The distance function is used by FADO to calculate the distance between the model and the next transaction. If this distance is smaller than the preset value (ε) the transaction is flagged as normal. A different distance function will result in a other calculated distance and therefore in a different genuine area.

Euclidean Distance

The Euclidean distance or norm is calculated by following formula.

$$\|v\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2} \tag{2.2}$$

This represents the length of the vector. The green line in figure 2.1 visualizes that line. All points on equal Euclidean distance in n dimensions define a sphere. Figure 2.2 shows that in a 2-dimensional space this results in a circle.

Manhattan Distance

One could also use the Manhattan distance to define the distance. The Manhattan distance, or City block-metric, defines the distance by only taking straight lines parallel to one of the axes. The following formula describes this.

$$\|v\|_1 = \sum_{i=1}^{n} |v_i| \tag{2.3}$$

The red, blue and yellow lines in figure 2.1 have all the same Manhattan distance. All points on an equal Manhattan distance define a rhombic space figure. Figure 2.2 shows that in a 2-dimensional space this results in a rhombus.

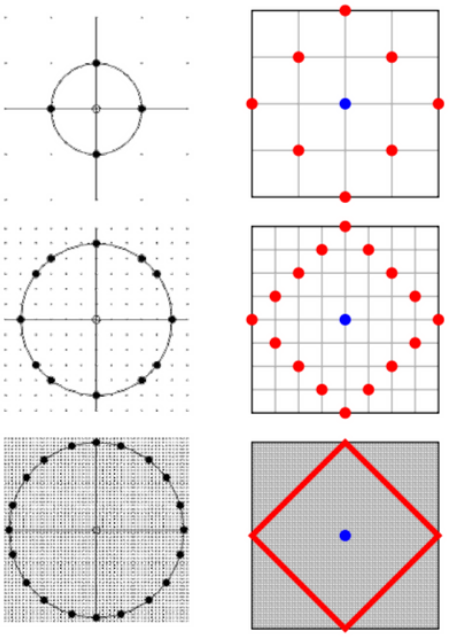

Figure 2.2. Visualization of connecting all points on same distance in 2D for respectively Euclidean and Manhattan distance.

2.3 Extended FADO

This thesis found a way to improve the results of FADO on financial transactions by extending the standard FADO. This is done by introducing a gray zone in which the model updates without raising an alarm. Algorithm 2 presents the pseudo code. An alarm is now only raised when $\|y_t - w_{t-1}\|_2 \geq \delta$ with $\delta \geq \varepsilon$. This is based on the hypothesis that a fraudulent transaction, that causes an update of the model, will be further away from the normal area than a genuine transaction. This will lead to more updates of the model but finally fewer alarms.

Algorithm 2: extended FADO(ε)

Initialize $w_0 = 0_d$.
for $t = 1, 2, \ldots$ **do**
  Receive transaction $y_t \in \mathbb{R}^n$.
  if $\|y_t - w_{t-1}\|_2 \geq \varepsilon$ **then**
    if $\|y_t - w_{t-1}\|_2 \geq \delta$ **then**
      (1) raise an alarm
    end
    (2) set

$$v_t = \frac{y_t - w_{t-1}}{\|y_t - w_{t-1}\|_2} \in \mathbb{R}^n,$$

    (3) and update

$$w_t = w_{t-1} + \gamma_t v_t$$

  end
  else
    (4) set $w_t = w_{t-1}$.
  end
end

2.4 Principal Components Analysis

Principal Components Analysis (PCA) transforms a vector with a certain dimension, number of features in a financial context, to a new vector with lower dimensions. It uses an orthogonal transformation to project correlated features into new uncorrelated features while losing as little information as possible [11]. Figure 2.3 illustrates a dimensional reduction from 3D to 2D by applying the orthogonal transformation. This means that there is a new coordinate system introduced that lays in the direction of the most variance data.

There are two common use cases for a PCA transformation. First of all, if the performance of the algorithm needs to be tweaked in sense of computer power (CPU & memory). Then the PCA transformation is used to reduce the dimensional complexity of the data while keeping most of the information in the data. This use case is not relevant for this thesis because FADO can handle high dimensional data with ease. PCA can also be used to reduce dimensions of data to make it possible to visualize it. Then it needs to be reduced to two or three dimensions for easy plotting. Visualization is an important tool to learn more about the data and decide if things run as expected. When PCA is used for visualization it is useful to calculate how much variance is lost due to the transformation to have an idea how much of the information in the data is visualized. This is calculated by taking the fraction of variance of the used components, and the total variance before transformation.

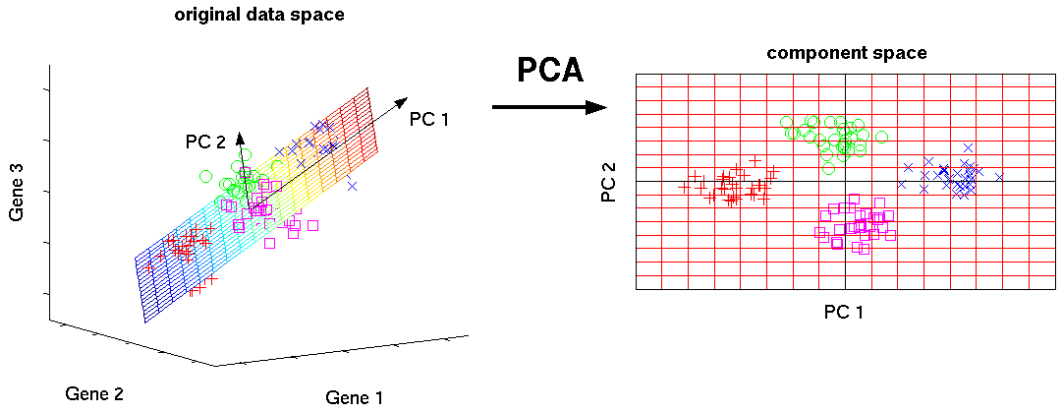

Figure 2.3. Practical illustration how a PCA transformation a dimension reduction achieves with losing as less as possible information [11].

2.5 One-hot encoding

Machine learning algorithms can only work with numerical data but in real life we often find categorical data. It is therefore important to transform these features into a different form. One way to do this is integer encoding. This feature engineering technique maps every alphabetical category to a number (integer), see two first columns in table 2.1. After transformation of a categorical feature there is a numerical feature left. The downside of the transformation is that there is a explicit order introduced in the data, that is often not representative. For example why would color 'yellow' be worth a higher numerical value than 'red'. Another technique that is used for the same purpose is one-hot encoding. One-hot encoding transforms a categorical feature into as many binary features as there are different categories. Hereby is ensured that every category is as important as another, see last sub table in table 2.1 [3].

color
red
green
yellow

color
1
2
3

red	green	yellow
1	0	0
0	1	0
0	0	1

Table 2.1. *First subtable contains a categorical feature with three categories. The second and third subtable represent the result after respectively integer encoding and one-hot encoding.*

2.6 Metrics

In the end, it is important to measure the performance in a representative way. For unbalanced problems, like fraud detection on financial transactions, accuracy is a bad metric to check the performance of the algorithm. For example if a fictive model would mark all transactions of the creditcard dataset, 284807

transactions with 492 frauds (3.1), as genuine the accuracy would be 99.8%, because only 492 false predictions are made.

$$accuracy = \frac{\#correctlabeled}{\#transactions} = \frac{284315}{284807} = 99.8\% \tag{2.4}$$

But in reality, none of the fraudulent transactions were correctly flagged and this model is worthless. Therefore it is more convenient to use metrics like true positives (TP), false positives (FP), true negatives (TN), false negatives (FN), recall, precision, f1 score and area under the receiver operating characteristic (AUROC). The rest of this paragraph explains these metrics. In the case of financial transactions TP, FP, TN, and FN can respectively be translated to blocked fraud, false alarm, normal transaction passed and missed fraud. Recall or true positive rate (tpr) defines how many of the positive class (frauds) are flagged as positive (fraud).

$$recall = \frac{TP}{TP+FN} \tag{2.5}$$

Precision defines how many of the flagged transactions were actually positive (fraud).

$$precision = \frac{TP}{TP+FP} \tag{2.6}$$

The f1 score is a metric that combines recall and precision into one value. The f1 score is defined by

$$f_1 score = \frac{2 * recall * precision}{recall + precision} \tag{2.7}$$

A high f1 score can only be achieved when none of both metrics are low. Because it is a well-known metric it is easy to compare with other algorithms. The last metric used in this thesis to measure the performance is AUROC. This metric is introduced to mark algorithms on unbalanced problems. The score is visually interpretable as the area under the ROC-curve. The ROC-curve plots the tpr (=recall) as a function of the false positive rate (fpr).

$$fpr = \frac{FP}{FP+TN} \tag{2.8}$$

The maximum score is 1 and minimum is 0. The previous example that marked all transactions as genuine would score a 0 as AUROC because tpr stays all the time 0. A completely random classifier would score an AUROC of 0.5 so everything higher than that is an improvement.

3. Methods

3.1 Datasets

3.1.1 Public Financial Datasets

Financial transactions contain sensitive data and are for privacy reasons not publicly available. Therefore this thesis relied on a synthetic dataset, which is similar to the real financial dataset used to generate it, and an anonymous dataset, which is gathered by 'Université Libre de Bruxelles (ULB)'. Fraudulent transactions are in general very rare in comparison with normal transactions. This is due to the fact that for every fraudulent transaction typically several hundred or even thousands of normal transactions happen. This makes financial datasets very unbalanced and often requires techniques like under- or oversampling before the data is useful for a machine learning algorithm. Balanced data is not required for FADO and these techniques are not further relevant for this thesis.

3.1.2 Generated Dataset

The dataset, Paysim, was simulated by extracting information from the logs of a mobile money service in an African country. A multinational company, active in 14 countries, is the owner of the mobile money service [1]. To make fraud detection possible on this dataset there were fraudulent transaction injected with predefined patterns by the simulator. The sample generated dataset that mimicked the original one was publicized on Kaggle. The data contains 11 columns, including the label 'isFraud' saying if the transaction is fraudulent and the label 'isFlaggedFraud' if a transaction is flagged as a fraud by the simulator. After a comprehensive exploration of the dataset and several tests it was not clear that the quality of this generated dataset is sufficient for fraud detection.

3.1.3 Anonymous Dataset

The creditcard dataset has been collected by the 'Université Libre de Bruxelles (ULB)'. The dataset contains all transactions made during two days in September 2013 by European cardholders and contains 492 fraudulent transactions out of 284.807 transactions in total. To solve the confidentiality issue the original features are removed by a PCA transformation resulting in 28 anonymous features. The total dataset also contains the columns time, amount and, a class that defines if the transactions are fraudulent.

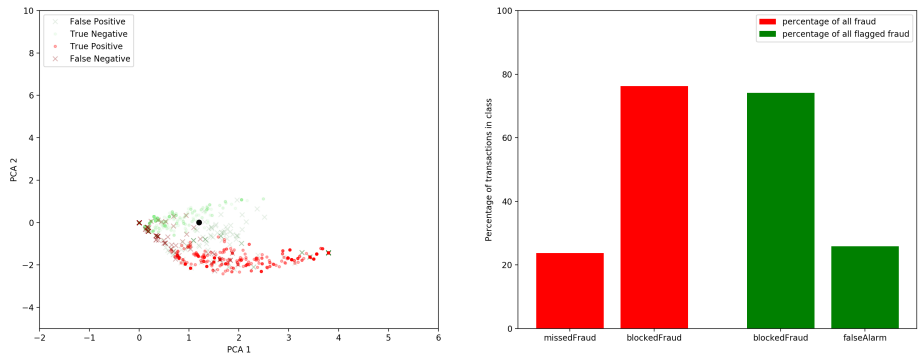

Figure 3.1. Example of the visualizations during processing a stream of transactions with FADO.

3.2 Setup

To simulate a stream of financial transactions, one by one the transactions were submitted from the dataset to FADO. FADO immediately flags every transaction as fraud or not. While processing the stream of transactions the implemented algorithm for this thesis visualizes in 2D, the evolution of the memory and the last transactions, through a PCA transformation, together with the distribution of fraudulent and flagged transactions. Figure 3.1 shows a preview of the visualizations. On the left graph, the transactions are plotted through a PCA transformation in two dimensions. The right graph contains the distributions. The first two red bars show how the fraudulent transactions (positives) are distributed between missed fraud (false negative) and blocked fraud (true positive) in percentages. The two other green bars show how the flagged transactions are distributed between blocked fraud (true positives) and false alarms (false positives). The colors on the scatter plot and statistics could only be shown because the labels, that reveals if a transaction is fraudulent, were used. This helped the development by faster detecting problems.

3.3 Preprocessing

Before the creditcard dataset was streamed to FADO it is initially preprocessed in 3 different ways: standard, binary cut-off and one-hot encoded. The different ways of preprocessing were compared to each other. Preprocessing is a useful tool to enhance detection performance of FADO. It can reduce noise in the dataset and increase the difference between normal and fraudulent transactions.

3.3.1 Standard

The creditcard dataset was normalized with the standard scaler of scikit-learn [12]. This ensures all features are in the same scale. This was necessary because the columns 'time' and 'amount' were not anonymous and so not default transformed with PCA. Figure 3.3 visualizes the dataset after a PCA transformation to two dimensions. The red dots represent the fraudulent transactions while the green dots represent the normal transactions.

3.3.2 Binary Cut-Off

This manner of preprocessing tries to map all normal transactions around the origin and the fraudulent ones as far away as possible, so the learning phase of FADO can be minimized. But because all the features, excluding 'time' and 'amount', were anonymous, it was impossible to exhaust financial knowledge to generate new features. For example, if there would be a feature present with the country of destination, one could test this against a list of high-risk countries to create a new feature 'high-risk destination country'. Then with a few of those features, it can be easier to detect fraud. To extract that kind of information out of the anonymous features, the labels were used. If the distribution of a feature between the genuine transaction and the fraud ones are skewed enough then this feature is used as an indicator of fraud. Figure 3.2 shows the distribution of feature V14, a generic feature created by PCA, and the fraud and genuine ones are clearly divided.

The new feature is extracted out of the old one by setting a cut-off at the quantiles 0,01 and 0,99. If the number of fraudulent transactions, that are smaller than the cut-off for the 0,01 quantile or greater than the cut-off for the 0,99 quantile, is significantly greater than 1% of the number of fraudulent transactions, a new feature is created. This is done by setting the value of the feature to 1 if it is respectively smaller or greater than the 0.01 or 0,99 quantile or else to 0. In the case of V14, see figure 3.2, the transaction will have a 1 if their value for feature V14 is smaller than -5. This is done for every feature. Out of the 30 features 17 new ones are obtained. Figure 3.4 visualizes the dataset after a PCA transformation to 2 dimensions. The red dots, that mark the fraudulent transactions, are explicitly more clustered than in figure 3.3 for the standard dataset.

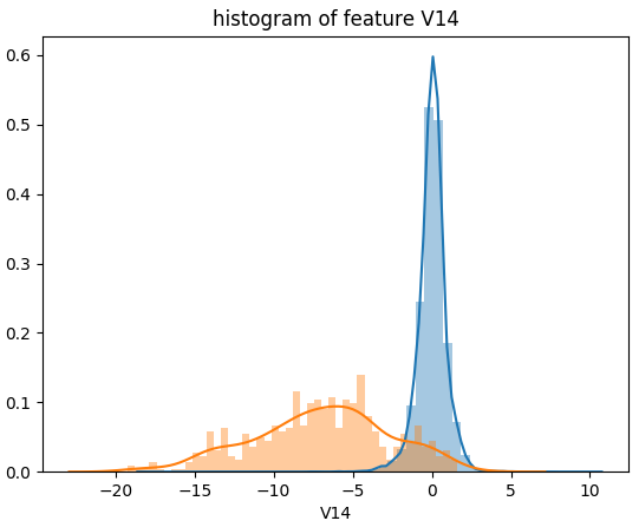

Figure 3.2. Distribution of feature V14 for the genuine (blue) and the fraudulent (orange) transactions.

3.3.3 One-Hot Encoding

This thesis will use a more advanced version of one-hot encoding, namely rank one-hot encoding. Hereby are all leading zero's from one-hot encoding set to 1. It indicates that a feature has at least that value. Further, in this thesis one-hot encoding will refer to the rank one-hot encoding. This preprocessing was introduced to mimic the binary cut-off without using any label and to be truly unsupervised.

Every feature, excluding 'time', got divided into different pieces based on quantiles, to make them of equal size. From now on we call every piece a bin. The number of bins can be chosen but increases the number of dimensions linear because every feature goes through this process. If ten bins are chosen the total amount will be 29 multiplied by 10 what gives a 290D transaction. Later the influence of the number of bins on the performance of FADO will be discussed. Figure 3.5 visualizes the dataset after a PCA transformation to 2 dimensions.

3.4 Oracle classifier

To see how the unsupervised FADO relates to a supervised fraud detection system, an oracle classifier was introduced. The oracle classifier used the most skewed feature in the dataset, V14. Figure 3.2 shows the distribution of the fraud and non-fraud transactions for feature V14. The classifier flags a transaction as fraud if the value for feature V14 was lower than a certain number. This is a very simple but still not realistic case because the labels

were used to set the cut-off. One could see this as a theoretical classifier to have an indication of detection performance of the dataset.

3.5 Validating performance

FADO starts flagging transactions as fraudulent from the first one that is submitted till the last one without any pretraining. All the learning is done while running. Therefor the metrics are calculated from the first till the last processed transaction.

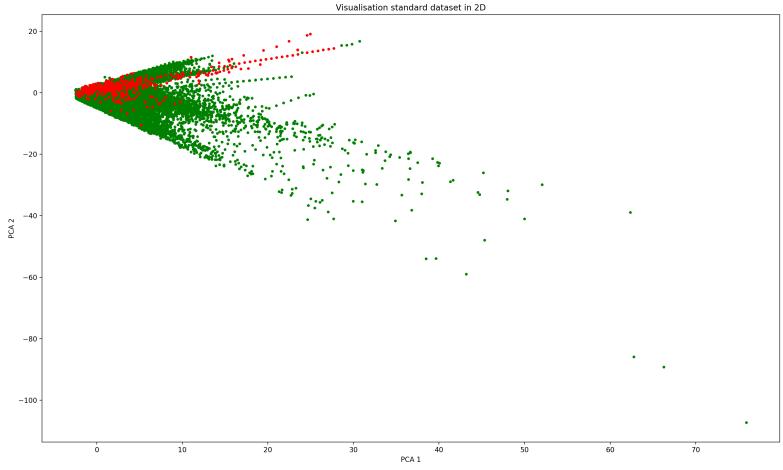

Figure 3.3. Standard dataset visualized by its two principal components.

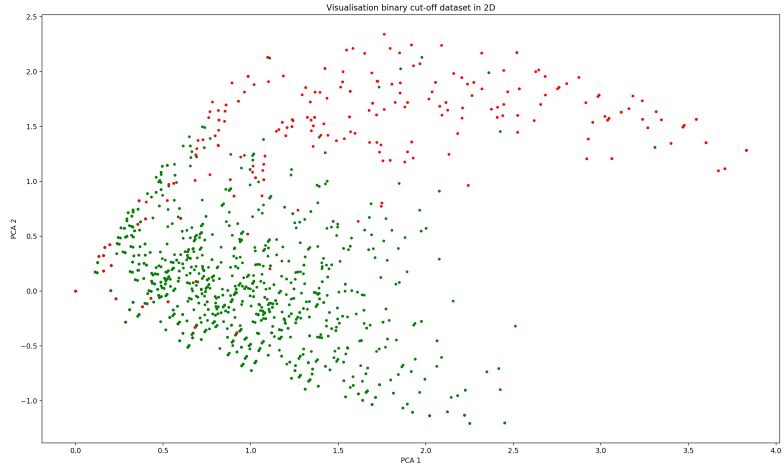

Figure 3.4. Binary cut-off dataset visualized by its two principal components.

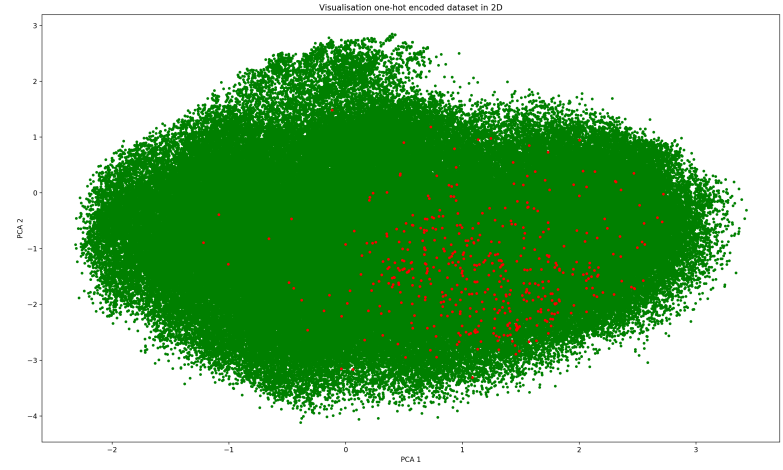

Figure 3.5. One-hot encoded dataset visualized by its two principal components.

4. Results

4.1 Oracle Classifier

In chapter 3 the oracle classifier is introduced by flagging a transaction as a fraud by comparing the value for feature V14 to a cut-off. Figure 4.1 shows how the recall, precision and f1-score relate to this cut-off. The higher the cut-off is set the higher the recall will be but on the other hand, the precision will decrease. The green line on figure 4.1 shows that the f1 score reaches its maximum in the intersection between precision and recall with a value of 0.63 for a recall-precision combination of 66% / 60%. Figure 4.2 shows how the f1 score fluctuates in proportion to the recall. Figure 4.2 and 4.3 will be compared with the other results. Figure 4.4 shows the ROC-curve with the AUROC.

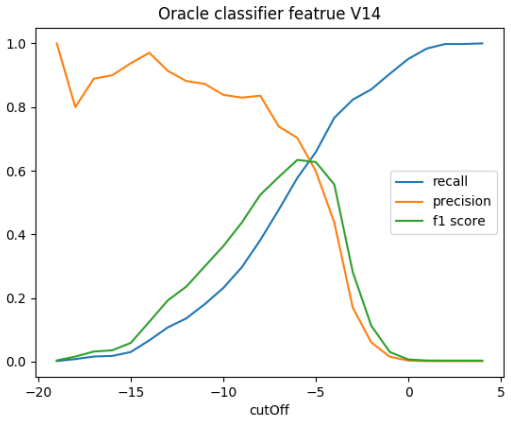

Figure 4.1. The track of the recall, precision and f1-score in function of the cut-off.

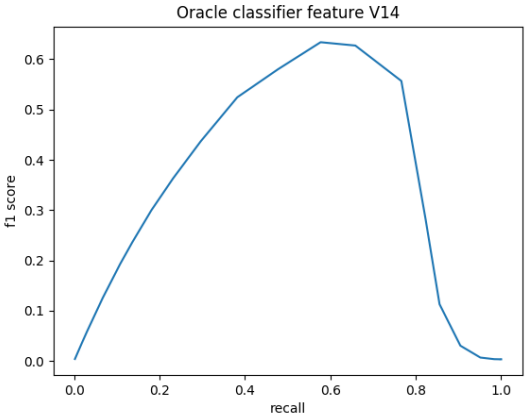

Figure 4.2. The f1 score as a function of the recall.

4.2 FADO

4.2.1 Standard Creditcard Dataset

The f1-score obtained when applying FADO to the creditcard dataset, without any preprocessing is shown in figure 4.5 (as function of the recall). The exact same experiment was done with the Euclidean (circle) and Manhattan distance (rhombus). Figure 4.6 shows the corresponding ROC-curve with an AUROC

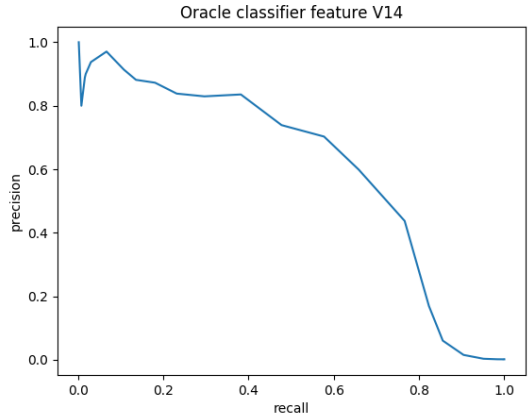

Figure 4.3. The ratio between the precision and the recall.

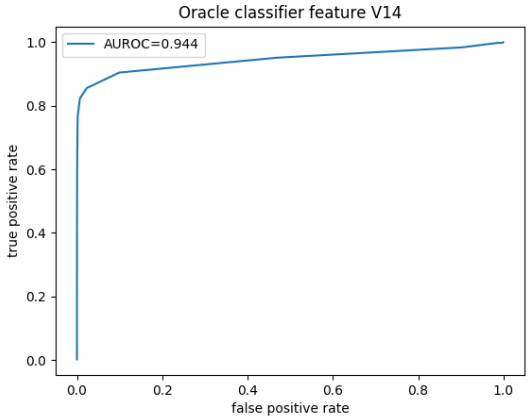

Figure 4.4. The ROC curve with the AUROC.

of approximately 0.95 for both distance functions. In figure 4.7 the relation between the recall and precision is plotted.

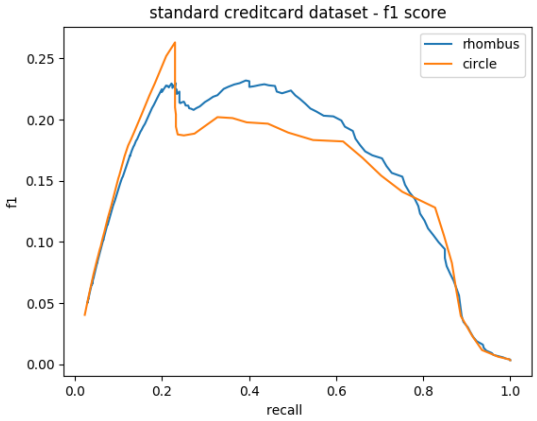

Figure 4.5. The f1 score in proportion to the recall for the Euclidean and the Manhattan distance function.

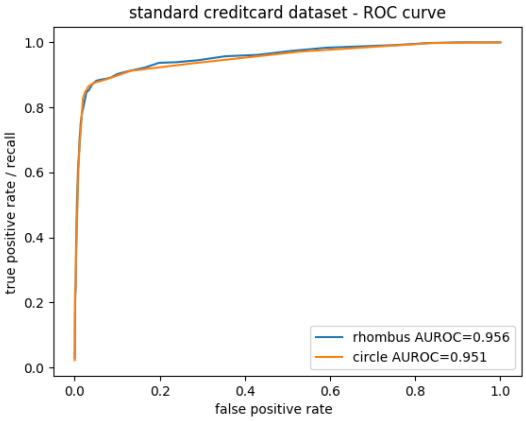

Figure 4.6. The ROC curve with the AUROC for the Euclidean and the Manhattan distance function.

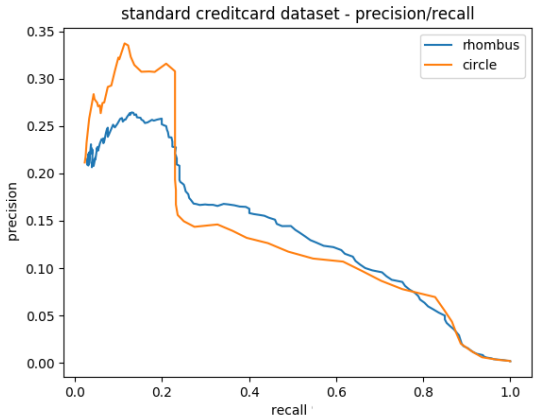

Figure 4.7. The ratio between the precision and the recall for the Euclidean and the Manhattan distance function.

4.2.2 Binary Cut-Off Dataset

The f1-score obtained when applying the creditcard dataset, with the binary cut-off preprocessing is shown in figure 4.8 (as function of the recall). The exact same experiment was done with the Euclidean and Manhattan distance function. Here circle and rhombus stand again for respectively the Euclidean and Manhattan distance. Figure 4.9 contains the ROC-curve with an AUROC of approximately 0.94 for both distance functions. In figure 4.10 the relation between the recall and the precision is plotted.

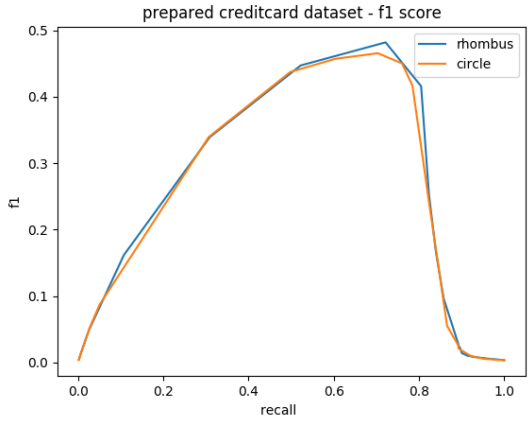

Figure 4.8. The f1 score in proportion to the recall for 2 different distance functions.

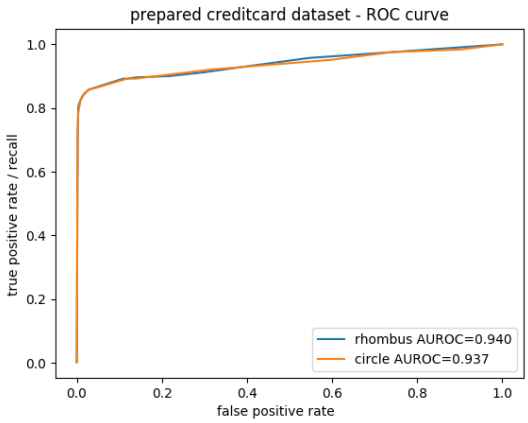

Figure 4.9. The ROC curve with the AUROC for 2 different distance functions.

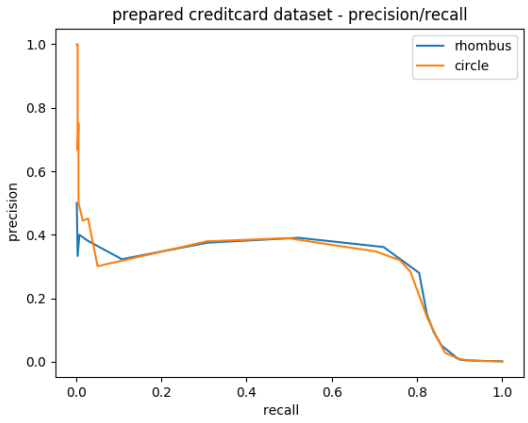

Figure 4.10. The ratio between the precision and the recall for 2 different distance functions.

4.3 Extended FADO

4.3.1 Binary Cut-Off Dataset

FADO uses a parameter, ε or radius, to determine if the model needs to be updated and an alarm should be raised. In contrast, Extended FADO uses the

first parameter, ε only to determine if the model needs to be updated and a second parameter, δ or cut-off, to determine if also an alarm should be raised. Figures 4.11 through 4.13 show the results for a carefully selected range of radii with the best corresponding cut-off. This makes it possible to compare them and find the best combination. Figure 4.11 shows how the f1 score relates to the recall. This plot indicates that a radius of 1.55 (light blue) or 1.6 (dark blue) gives the highest corresponding f1 score. Figure 4.12 shows an adjusted ROC-curve based on the amount of money saved ratio the absolute number of false positives (false alarms). This confirms that a radius of 1.55 or 1.6 gives the highest saved money with the lowest corresponding false alarms. In figure 4.13 the relation between the recall and precision is plotted.

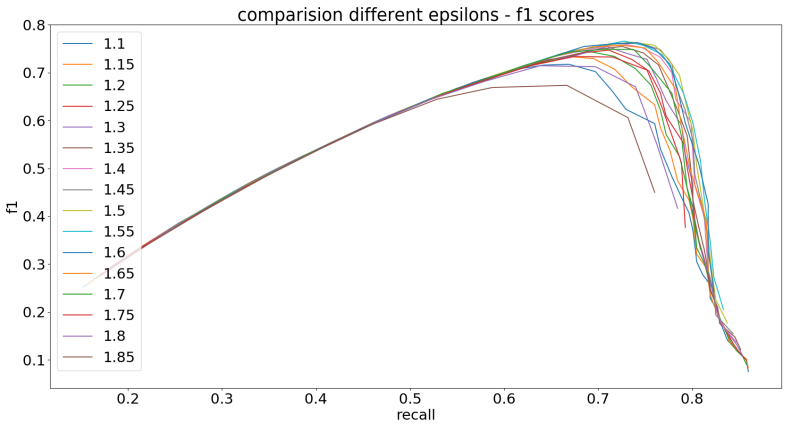

Figure 4.11. The f1 score in proportion to the recall for multiple radii.

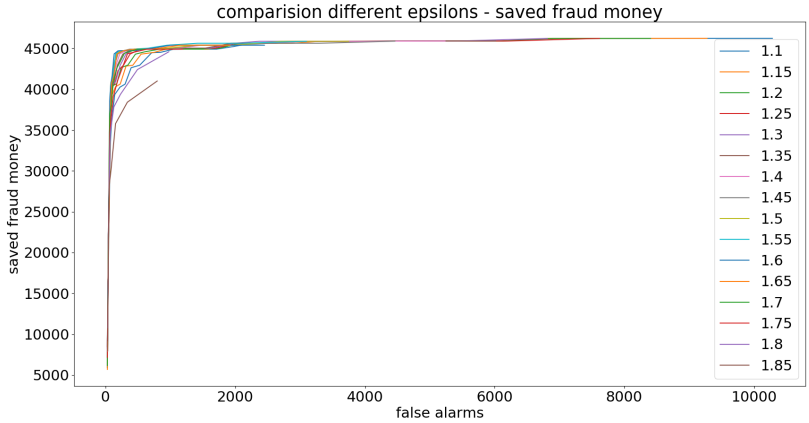

Figure 4.12. A money-based ROC curve with multiple radii.

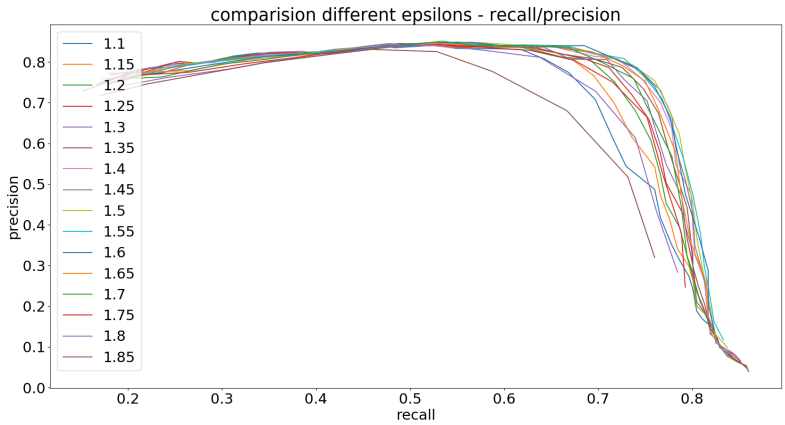

Figure 4.13. The ratio between the precision and the recall for multiple radii.

Detail Best Performance

Previous results indicated that a radius of 1.55 gives the best radius/cut-off combination. Therefore, it is important to have a closer look at it. Figure 4.14 contains the most important metrics. The plot on top shows for the different classes TP (blocked fraud, dark green), FP (false alarm, light green) and FN

(missed fraud, red) the distance away from the model determined by the distance function, here euclidean distance. This graph shows that the TP's are further away than the FP if an update of the model occurs. The second graph in figure 4.14 plots the difference in distribution for TP's and FP's and are clearly skewed. The third graph shows the relationship between the cut-off and the f1 score and shows that the highest score is achieved with a cut-off at 2.05. In the fourth graph, the relationship between the recall and precision is plotted with a maximum value of 81% precision for a corresponding 73% recall. The last graph shows the money based ROC curve but instead of absolute numbers, fractions are used. While blocking less than 1% of the money in normal transactions, more than 70% of all fraudulent money is successfully blocked.

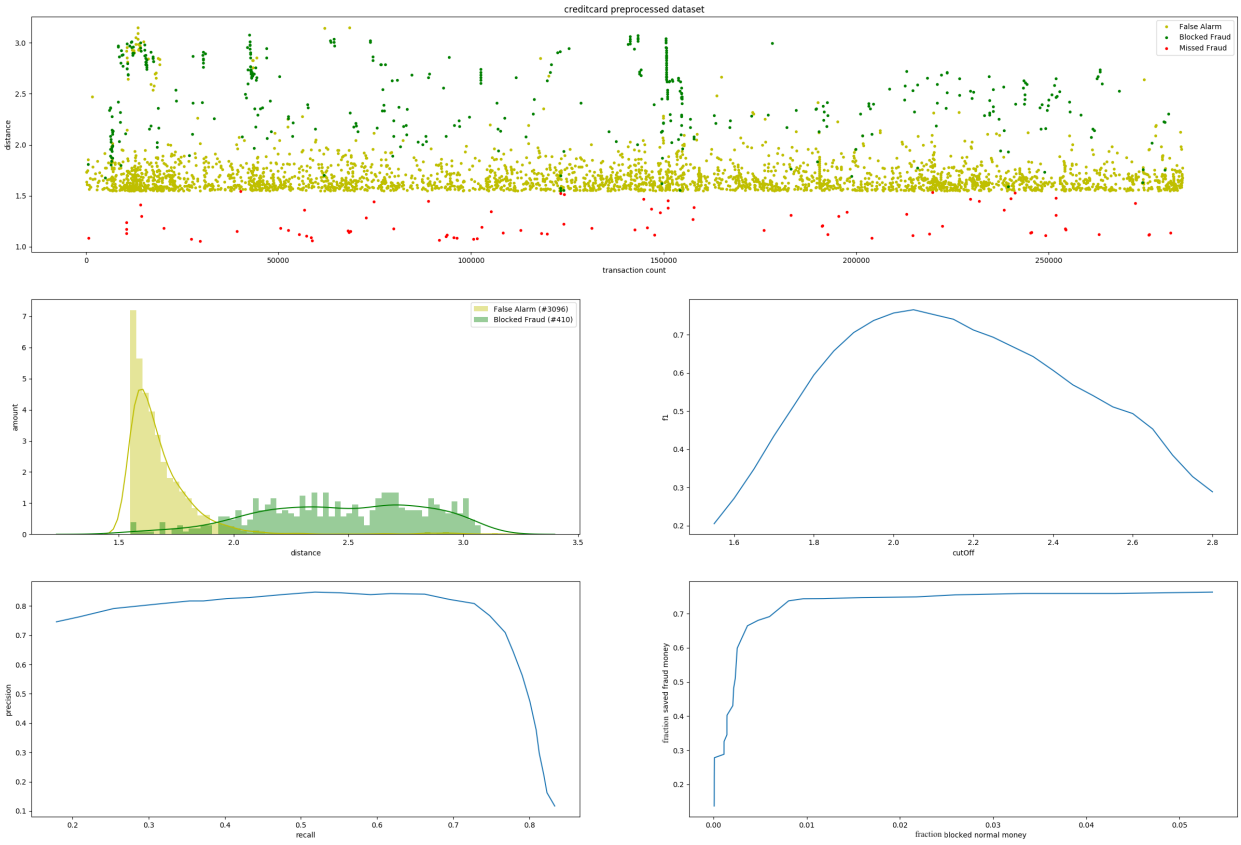

Figure 4.14. Specific metrics for extended FADO with a radius of 1.55.

4.3.2 One-Hot Encoded Dataset

Detail Best Performance

The one-hot encoding preprocessing is used to test the performance of FADO in a truly unsupervised way. Figure 4.15 contains the most important metrics for the one-hot encoded dataset. On the first graph, that shows the distance away from the model for each transaction, one can now see a learning phase when the algorithm processes the first transactions. The second graph shows the skewed distribution of the false alarms and the blocked fraud in function of the distance away from the model. In the third graph, the f1 score is plotted against the corresponding cut-off. The maximum f1 score of 0.32 is reached at a cut-off of 11,55 but if we aim for a 70% recall, an f1 score of 0.23 is reached. The fourth graph plots the relationship between the precision and recall. The last graph presents a money-based ROC-curve, to block 70% of the fraudulent money the system will now block 10% of the normal money.

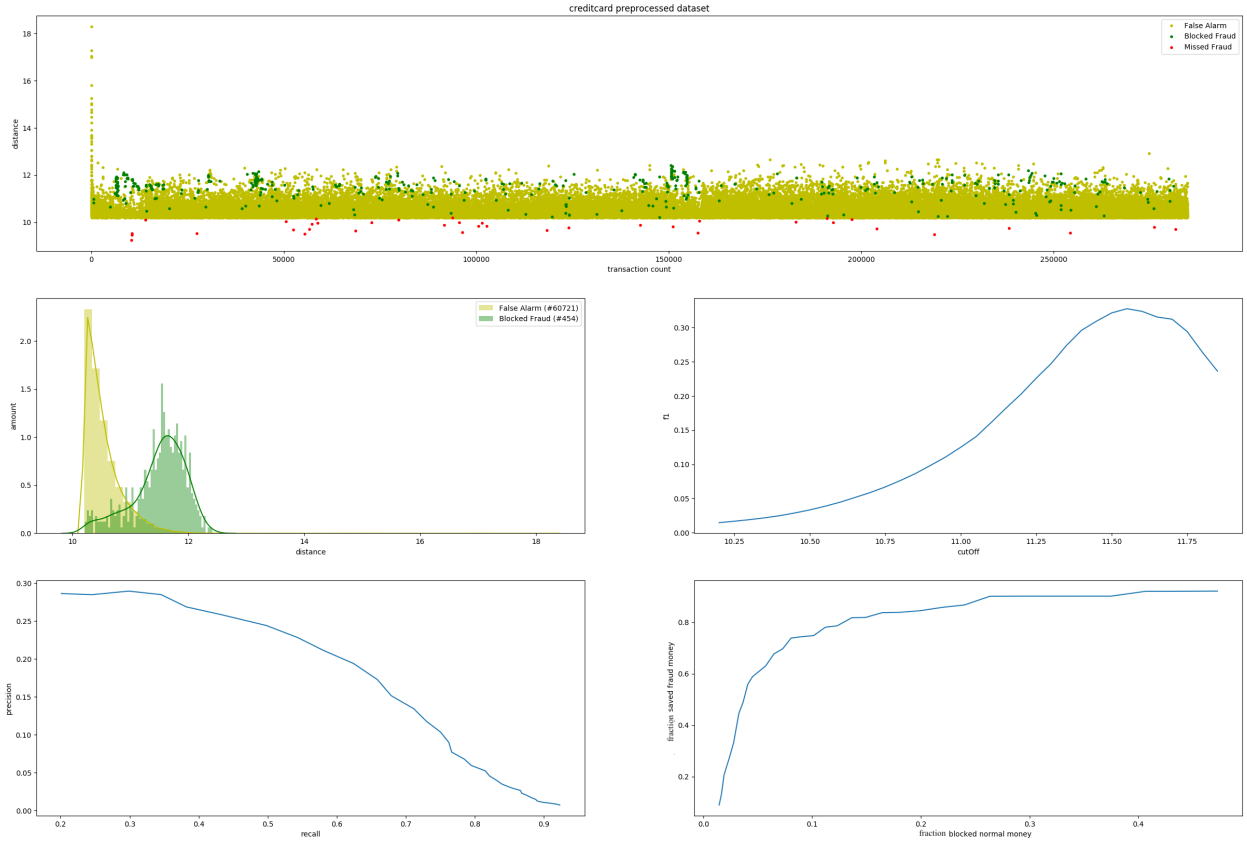

Figure 4.15. Detail metrics of the one-hot encoded dataset with 20 bins for a radius of 10.2.

Influence bins and learning-rate constant

Chapter 2 explained that a number of bins had to be selected to preprocess the dataset. The green line in figure 4.16 connects the corresponding f1 scores to a recall of 70% for the datasets with a different number of bins. This line is stagnating when it reaches higher dimensions. Chapter 2 also requested attention for the strictly positive constant in the decreasing learning-rate, from here on there will be referred to this constant as hidden constant. Figure 4.16 also includes the f1 score of five different learning constants in a logarithm scale. Despite the f1 score rises faster for a hidden constant equal to 0.3, a value of 1 as hidden constant reaches the highest f1 score. To easily compare the dashed line shows the corresponding f1 score for 70% recall without any preprocessing. The blue line with a constant of 0.1 shows that a too low learning-rate is destined to fail, this is because of the learning phase that becomes significantly larger. To indicate this behavior figure 4.17 shows the distances away from the model over time with a hidden constant equal to 0.01.

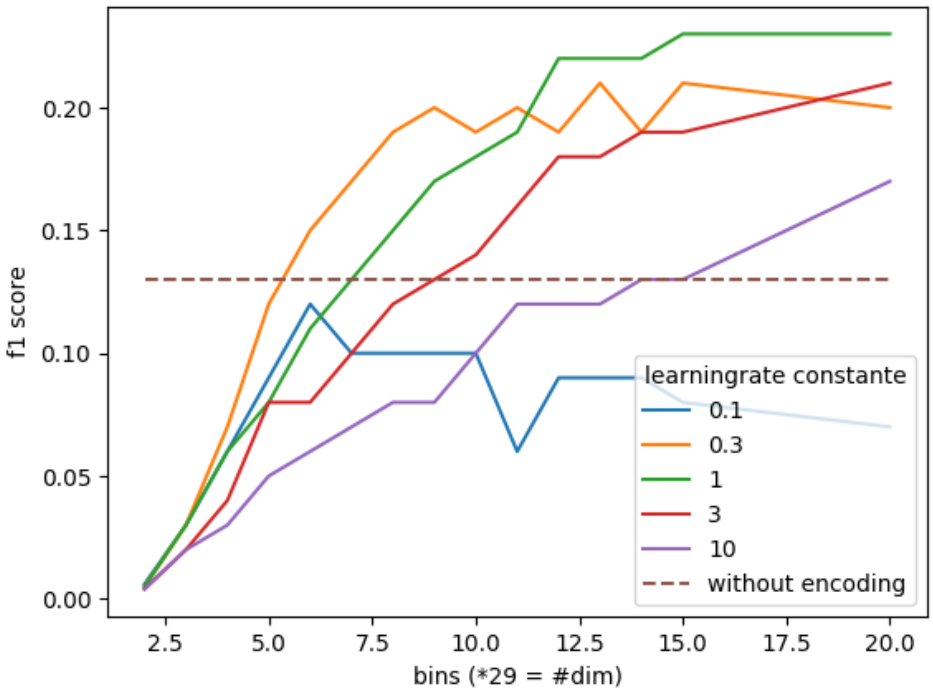

Figure 4.16. Influence of the number of bins and the hidden constant in the decreasing learning-rate on the f1 score, f1 scores corresponding to a 70% recall.

4.4 Overview Results

Table 4.1 gives a numerical recap of all the different setups with an aim of 70% recall ordered in decreasing f1 score. The extended version of FADO didn't

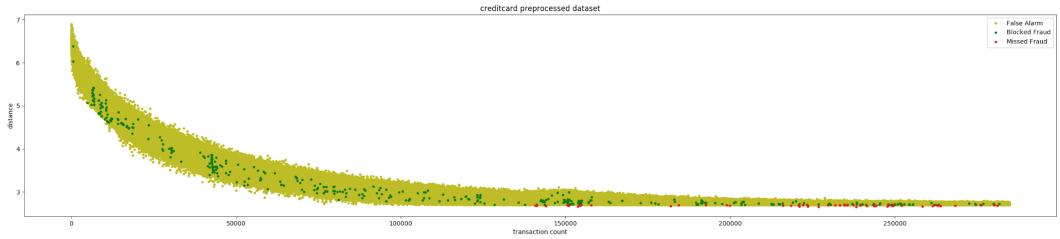

Figure 4.17. The path of distances between model and transaction over time with a learning-rate constant of 0.01.

improve the detection performance on the standard dataset. This is because the dataset is not preprocessed to create a distance between normal and fraudulent transactions.

Setup	TP	FP	FN	Recall	Precision	F1 Score
Ext. FADO Binary Circle	358	85	134	73%	81%	0.77
Oracle Classifier Best	324	217	168	66%	60%	0.63
FADO Binary Rhombus	355	629	137	72%	36%	0.48
FADO Binary Circle	346	649	146	70%	35%	0.47
Ext. FADO One-Hot Circle	350	2241	142	71%	14%	0.23
(Ext.) FADO Standard Circle	347	3284	145	71%	10%	0.17
(Ext.) FADO Standard Rhombus	346	3649	146	70%	9%	0.15
FADO One-Hot Circle	322	6377	170	65%	5%	0.09

Table 4.1. *Overview of results for used setups with aim for 70% recall. The first section contains the setups where the labels were used.*

5. Discussion

5.1 Euclidean vs Manhattan Distance

A first important variable that could be used to tweak the performance of FADO is the distance function. Figures 4.5 till 4.10 does not show a significant difference between the Euclidean distance and the Manhattan distance. One could say that for the standard dataset at a low recall level the Euclidean distance function gives better results but a discussion with a financial institute made it clear that such a low recall is not relevant and there should be an aim for a 70% recall. In the further discussion of the results there is no more attention spend to the used distance function.

5.2 Standard vs Binary Cut-Off Dataset

The maximum f1 score of the binary cut-off dataset for FADO is almost twice as high as the one for the standard dataset. The maximum f1 score for the standard dataset is 0.26 for a corresponding 23% recall while only an f1 score of 0.15 is reached for a 70% recall. A maximum f1 score of 0.47 for a recall of 70% is reached with the binary cut-off dataset. Not only the peak of the curve is shifted to the desired recall, the absolute value is also almost doubled. In contrast, the AUROC curve is slightly higher for the standard dataset but if we have a closer look at the graph we can see that the gain is made by having a higher recall for a high FP rate. In practice, a lot of FP's are not tolerable and one should focus on a high recall with as few as possible FP's. An improvement with a factor of four is almost achieved for the binary cut-off dataset against the standard dataset. The precision increased from 9% to 35% for a corresponding 70% recall.

One should keep in mind that labels were used to do the preprocessing, we can justify this because of lack of feature names. The supervised way of preprocessing could in all probability be replaced by financial insights by experts.

5.3 FADO vs Extended FADO

The improvement in performance of extending FADO was tested by submitting the binary cut-off dataset to FADO as well to the extended FADO. Therefore we compare the best results for both setups. As discussed before FADO

gave for the binary cut-off dataset a recall/precision combination of 70% / 35%. The extended FADO setup gives a 73% / 81%, this is translated to a f1 score of 0.77. This is a big achievement and great improvement of FADO. Future work could prove that the supervised way of preprocessing can be replaced with financial insights of experts then could this fraud detection system be used in the real world.

5.4 One-Hot Encoded Dataset

5.4.1 One-Hot Encoded vs Binary Cut-Off Dataset

The one-hot encoded dataset was introduced to imitate the behavior of the binary cut-off dataset without using the labels. This aim is not achieved as the maximum f1 score with this dataset is 0.32 but for the suggested 70% recall only 0.23 in contrast to the 0.77 with the dataset that used the labels. The imitation did not succeed because of the fact that the features contain too much noise. The noise could not been reduced to a bare minimum because the lack of financial insights.

5.4.2 One-Hot Encoded vs Standard Dataset

Figure 4.16 proved that the one-hot encoded dataset performs better than the dataset with standard preprocessing. The dashed line visualizes the barrier that should be broken to improve the standard dataset. With a hidden constant of 0.3 for the learning-rate is this barrier already broken with only 6 bins. The best result was gained with a default 1 as hidden constant and 15 bins, a recall-precision combination of 71% / 14% gives an f1 score of 0.23. This corresponds to a 65% increase of precision compared to the 9% achieved by the standard dataset, but a precision of only 14% is still too low to be useful in a real fraud detection system.

5.5 Hidden constant Learning-Rate

The hidden constant in the learning-rate defines how long the initial learning phase will take. If the constant is too big the learning will go very quickly but when a fraud occurs the model will be updated too much in the direction of the fraud. This results in a false alarm on the next genuine transaction or in a possible missed fraud if the next transaction is again a fraudulent one. On the other hand, if the learning rate is too small then the learning phase will take long before giving a useful fraud detection. A way to avoid this is to learn the model before using it. In figure 4.17 this would mean that at least 200.000 transactions are needed before the fraud detection system becomes useful.

5.6 FADO vs Oracle Classifier

When comparing all setups of FADO against the oracle classifier there is only one that outperforms it, namely, the extended FADO on the binary cut-off dataset. This particular setup combined all the researched improvements to achieve a better performance.

5.7 Benefits FADO

FADO has several big advantages. One of them is that the time complexity of deciding if a transaction is fraudulent is only $\mathcal{O}(1)$. Together with the fact that FADO works in a streaming way makes it perfect for modern payment systems. In contrast to more typical, static fraud detection systems, FADO is designed to learn how normal transaction look like instead of hard-coding fraudulent patterns. This means that FADO will detect new patterns of fraud without any required update as long as the fraudulent transaction differ enough from a normal one. FADO only makes one strong assumption, that all normal transactions can be clustered together, if that is not the case an update of the basic FADO is required.

6. Conclusion and Future Work

6.1 Conclusion

The need for an efficient online fraud detection algorithm is rising together with modern payment systems. Hereby should the focus lay on detecting as many fraud as possible without raising a lot of false alarms. The inspection of alarms is a time-demanding process that costs a lot of money. FADO seems very promising but needs some more research in co-operation with a financial partner to answer on this demand.

This thesis explored different techniques of preprocessing with a different rate of success. The one-hot encoded dataset was not able to achieve the same performance as the binary cut-off dataset but was significantly better than the dataset without preprocessing. The more bins are used with one-hot encoding the better the results will be but this increases the dimension of the transaction. Because of the gain in detection performance stabilizes with more bins, it is recommended to use 10-15 bins. Further, is the influence of the variables that control FADO examined. When the hidden constant in the learning-rate is too big then the model will be updated too much in the direction of a fraud but on the other hand, when the hidden constant is too small it will take a lot of transactions before the model learns what a normal transaction is. Selecting a different distance function doesn't change anything significantly except that the use of a corresponding ε is required. Eventually, an improved version for fraud detection of FADO was presented as 'extended FADO'. Extended FADO uses a second cut-off when deciding to raise an alarm or only update the model. This improvement reduced the amount of false alarms without handing in a lot of blocked frauds.

6.2 Future Work

The lack of publicly available dataset made it impossible to extract business insights in what the binary cut-off dataset exactly was doing. Therefore it is recommended to test the same technique on a real dataset with a financial partner. This could lead to the discovery of unseen fraud patterns. Further, should one test if it is possible to achieve the same results as with the binary cut-off dataset by replacing the preprocessing with labels with financial insights of experts. This thesis made a strong assumption that all normal transactions are clustered together. This was a reasonable assumption on this dataset but other

datasets may require that FADO gets upgraded to work with multiple clusters. It would also be interesting to see the one-hot encoding being implemented in a streaming way. Initially, there could be a relatively low amount of bins that increases over time which becomes more accurate as the learning process goes on.

References

[1] Lopez-Rojas Edgar Alonso. *Applying Simulation to the Problem of Detecting Financial Fraud*. 2016. http://bth.diva-portal.org/smash/record.jsf?pid=diva2

[2] Reid A. Johnson Andrea Dal Pozzolo, Olivier Caelen and Gianluca Bontempi. *Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM)*. 2015. https://www.kaggle.com/mlg-ulb/creditcardfraud.

[3] Jason Brownlee. *Why One-Hot Encode Data in Machine Learning?* 2017. https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/.

[4] Etienne Brunet. *Swish, the secret Swedish FinTech payment company created by Nordic banks and used by 50% of Swedes is challenging Swedish unicorns*. 2017. https://medium.com/@etiennebr/swish-the-secret-swedish-fintech-payment-company-created-by-nordic-banks-and-used-by-50-of-swedes-cfcf06f59d6f.

[5] Animesh Patcha and Jung-Min Park. *An overview of anomaly detection techniques: Existing solutions and latest technological trends*. 2007. https://doi.org/10.1016/j.comnet.2007.02.001.

[6] Kristiaan Pelckmans. *FADO: A Deterministic Detection/Learning Algorithm*. 2017. https://arxiv.org/abs/1711.02361.

[7] Andrea Dal Pozzolo. *Adaptive Machine Learning for Credit Card Fraud Detection*. 2015. http://www.ulb.ac.be/di/map/adalpozz/pdf/Dalpozzolo2015PhD.pdf.

[8] Andrea Dal Pozzolo. *Calibrating Probability with Undersampling for Unbalanced Classification*. 2015. https://www3.nd.edu/ dial/publications/dalpozzolo2015calibrating.pdf.

[9] Z. Rabiei. *Identifying intended and unintended errors in financial transactions: a case study*. 2017.

[10] Hanumantha Rao. *Credit Card Fraud Detection, Anomaly Detection Using Python*. 2018. http://www.datajango.com/credit-card-fraud-detection-with-python-complete-classification-anomaly-detection/.

[11] Matthias Scholz. *PCA - Principal Component Analysis*. 2016. http://www.nlpca.org/pca_principal_component_analysis.html.

[12] Scikit-learn. *StandardScaler*. 2018. http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

[13] Scott Stone. *E-commerce Can Expect Nearly $7 Billion in Chargebacks in 2016*. 2016.

https://chargeback.com/ecommerce-can-expect-nearly-7-billion-chargebacks-2016/.

[14] Scott Purdy Zuha Agha Subutai Ahmad, Alexander Lavin. *Unsupervised real-time anomaly detection for streaming data*. 2017. https://doi.org/10.1016/j.neucom.2017.04.070.

[15] Jerome Friedman Trevor Hastie, Robert Tibshirani. *The Elements of Statistical Learning Data Mining, Inference, and Prediction*, volume 2nd Edition. 2008.

[16] ARINDAM BANERJEE VARUN CHANDOLA and VIPIN KUMAR. *Anomaly Detection: A Survey*. 2009. https://dl.acm.org/citation.cfm?doid=1541880.1541882.

[17] Emily Vuitton. *E-commerce Payment Fraud Outlook 2017-2020*. 2017. https://chargeback.com/ecommerce-payment-fraud-outlook-2020/.