# COMBATING FRAUD BY LEVERAGING MACHINE LEARNING AND HUMAN BEHAVIOR

Casey Astiz

Adviser: Professor Michael Linderman

A Thesis

Presented to the Faculty of the Computer Science Department

of Middlebury College

in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Arts

December 2018

## ABSTRACT

I am writing about fraud and how to detect it! Wooh!

## ACKNOWLEDGEMENTS

I would like to thank Professor Linderman and Professor Scharstein first and foremost for their help in this Computer Science Thesis. Without them, this thesis would cease to exist. I would also like to thank the Computer Science department at Middlebury College for giving me the opportunity to learn and develop as a computer scientist. Last, I would like to thank all my unnamed interviewees who were willing to discuss their expertise about fraud detection with me. I truly appreciate all who helped create this thesis, and add to my knowledge!

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

CHAPTER 1

# INTRODUCTION

In today's world, every day there are new ways to pay for goods or services. Cash is a way of the past; mobile, peer to peer, and online payments are the way of the future. While these convenient payment methods are very exciting, the additional methods of payment lead to more gaps in security, in particular fraud. Fraudsters have taken advantage of every system in some form or another, and it is the financial institution's job to block or detect these fraudsters or else deal with the payments. For reference, credit card fraud through online payments in Australia hit $476 million in 2017, rising $58 million from the year before [12]. This level of fraud is happening all over the world, and it is necessary to find more efficient and effective methods to catch fraudsters to keep everyday financial transactions safe. In addition, when banks lose money to fraudsters, card holders partially or entirely pay for this loss through higher interest rates, fees, or reduced benefits. Without automatic fraud detection, a significant amount of overhead is created because of the need to investigate these transactions [3]. It is important to limit the amount of fraud because consumers pay for this fraud.

The main purpose of this thesis is to investigate current methods of fraud detection and implement one or more of those methods in a financial context. I will discuss the most prevalent and current methods for fraud detection, as well as past fraud detection methods. Fraud detection is a difficult field of study because it falls in the subset of anomaly detection and has very skewed classes as more transactions are not fraud than are fraud, and the classification of fraud is always changing. In order to build an effective fraud detector, it must be adaptive to new patterns and categories of fraud that are developing as fraudsters find new methods to deceive the system.

The main focus of this thesis will be a review of different methods used in the industry in the past and present for fraud detection, as well as my own implementation

using a synthetic credit card transaction dataset. The background section will include information from published research papers as well as information I learned from interviewing people at different companies in this field to provide some context for the literature review. Comparing business interactions with fraud to published research in the field will allow me to somewhat analyze the performance of different fraud detection methods in real life versus a research context.

Chapter 2 discusses past and current research into the field of fraud detection. Chapter 3 gives context for the problem at hand, synthesizing the information I learned from interviewing professionals in the fraud world. In Chapter 4, I describe the methodology and experiments I ran. Chapter 5 will discuss other applications for successful fraud detectors.

CHAPTER 2

# MODERN DAY FRAUD

What exactly is fraud? Fraud is defined as "wrongful deception with the intent to gain personally or financially," or intentionally deceiving another person to obtain something they have [1]. Fraud can either be a civil or criminal offense, depending on the state. There are a multitude of types of frauds, from credit card fraud and website misdirection to pyramid schemes and insurance fraud. There has been much research into the best and most efficient ways to detect different kinds of fraud. The goal of this research is to detect the largest amount of fraud with the least error. This may seem obvious, but is particularly important because this research can be applied to real scenarios. It is important for business to balance finding all fraud and limiting their customers. A company may choose to accept a small level of fraudulent transactions in order to include as many real transactions as possible. For example, it is less expensive for a company to let through a new customer that is potentially fraudulent occasionally, and maximize the number of new customers that are not fraudsters. The opportunity cost of rejecting every potential fraudster is too high for most companies to risk. For that reason, fraud detection out in the wild is never going to be at 100 percent accurate because the false positives are too big a cost.

However, in a research context, there are no such limits. The previous fraud research has covered many different types of fraud, from broad to narrow scenarios. Researchers are able to manipulate datasets to have ideal fraud to valid transaction ratios, and find the place their algorithm performs the best. Since researchers do not need to make a split second decision for whether or not to deny a transaction, they have time to find the best algorithm for the specific problem they are working on. By examining past and current research in this field, I have created a synopsis of information about fraud detection, and will implement a similar approach to one of the papers discussed in Chapter 4.

## 2.1 Credit Card Fraud Research

An early example of credit card fraud detection comes from a study by Chan and Stolfo from 1998 [3]. This paper, titled "Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection" seeks to find a solution to classifying skewed classes. Here, the number of fraudulent transactions is relatively small compared to the legitimate ones, and the amount of financial loss for each fraudulent transaction depends on the amount of the transaction and other factors. However, millions of transactions occur every day, and only a small portion of these transactions are fraudulent. Skewed classes are also the norm for cellular phone fraud detection as well as natural language processing problems.

In order to solve the problem, Chan and Stolfo implement a cost-sensitive approach instead of a classic error rate. They use a multi-classifier meta-learning approach, where they create data subsets with appropriate class distributions and apply learning algorithms to the subsets. Then, they integrate and optimize the cost performance of the classifiers by learning from their classification behavior. The benefit of this approach is that it handles non-uniform cost per error and is cost sensitive during the learning process. Chan and Stolfo use a dataset from Chase Manhattan Bank containing half a million transactions, from which 20 % are fraudulent, and implemented a cost model based on information from a bank representative.

Given this data and information, the authors experimented with the effects of training class distributions on the credit card cost model. They used the first 10 months as training data and the 12 month for testing, and randomly sampled transactions to get varied fraud amounts. The multi-classifier meta learning approach works by randomly dividing the dataset into 4 subsets, and then learns on the subsets using one of four algorithms: CART, C4.5, RIPPER, or BAYES. The meta learning then learns from that class combiner behavior and uses that as a class-combiner strategy. The authors found

that the training class distribution can affect the performance of the learned classifier, which is an issue given that natural distributions can be different than desired training distribution. However, they found their method to be efficient and scalable for large datasets. Since thieves learn and fraud patterns evolve over time, and adaptive classifier selection method is essential. This modular multi-classifier approach facilitates adaptive over time and removal of out of date learned behavior.

A 1997 paper by Stolfo et al uses Meta-Learning for credit card fraud detection [8]. Here, the authors argue that a false positive rates and true positive rates are much better metrics than overall accuracy when evaluating these learned fraud classifiers. Their proposed meta-learning system allows financial institutions to share their models of fraudulent transactions by exchanging classifier agents in a secured agent infrastructure. This is a useful tool for financial institutions to be able to work together to solve similar fraud problems.

For this study, the authors used a database of 500,000 transaction records from the Financial Services Technology Consortium, each containing 30 fields. In their experiment, the authors tested various machine learning and meta-learning models with different proportions of fraud to non-fraud training data. Based on their experiments, they find that a 50%/50% distribution of fraud to non-fraud training data generates classifiers with the highest true positive rate and the lowest false positive rate. The best method the authors discovered is using meta-learning with BAYES as a "meta-learner to combine base classifiers" [8]. The authors' results are in line with previous research in this field.

In a paper published this year, Zanin et al take a new approach to credit card fraud detection. Here, the authors use parenclitic network analysis, which is a hybrid data mining and complex network classification algorithm [14]. The authors find that fraud detection is a similar problem to designing a recommendation machine or diagnostic medical tools, which suggests a complex network approach may be beneficial. Here, the

definition of credit card fraud is included in a wider notion of financial frauds. Like the other papers discussed, the authors emphasize the rising costs of credit card fraud, rising to billions of dollars in yearly losses, and comprising about 1.4% of online payments. However, credit card fraud is not only important for the bottom line, as "credit card frauds have important social consequences and ramifications, as they support organized crime, terrorism funding, and international narcotics trafficking" [14].

Zanin et al combine data mining techniques to detect hidden patterns in data with a networking approach, which adds the ability to synthesize metrics to describe a "global structure created by the interactions between the different features" [14]. This study uses parenclitic networks, and evaluates them on a dataset of real transactions in comparison to the results of a standard ANN approach. In general, a parenclitic network is a network reconstruction technique that allows highlighting of the differences between one instance and a set of standard instances. These parenclitic networks are summaries of the groups of features whose correlation differs from a normal or licit transaction. Therefore, the structure of a parenclitic network stores information about abnormal correlated features in a credit card transaction. The authors take the computed network and transform it into a set of features compatible with a data mining algorithm. There are two main families of fraud detection algorithms: supervised and unsupervised learning. While both have their advantages and disadvantages, supervised learning is better at detecting illegal transactions whereas unsupervised learning is better at problems like self-organizing maps.

To test their parenclitic network approach, the authors used a dataset of all credit and debit card transactions from clients of a Spanish Bank BBVA from January 2011 to December 2012. This dataset includes standard fields about transactions such as amount, origin and destination, which has been anonymized. The authors have also synthesized information like average transaction size from a user from the given information. They

use multi-layer perceptrons, a type of ANN, as the model for classifying their transactions. These multi-layer perceptrons are represented by a set of connected nodes in which each connection has a weight associated with it that can be adjusted. Zanin et al find that the parenclitic networks themselves are not enough to reach a low classification error. However, the addition of parenclitic features to the raw data set enforces the results, decreasing the error rate from 19.2% to 12.23% [14]. Like others have mentioned before, false positives are extremely expensive.

In a 2017 paper, Wedge et al have attempted to solve the "false positive" problem in fraud prediction at an industrial scale. The goal of this paper is to use a feature engineering approach to dramatically reduce false positives [13]. It is estimated that 1 in 5 declared fraud transactions are actually fraud, and analysts have found that these false positives are costing more than the fraud itself. However, improving false positive rates with human involvement is very costly, prompting Wedge et al to study the automatic method by the name of deep feature synthesis. With so many online payments, there has been a massive increase in available transaction data, meaning that companies and banks are better equipped to handle fraud detection. The dataset the authors use contains roughly 9.5 million transactions with approximately 112,000 fraudulent transactions. The data is incredibly rich and granular, and requires "a 59-page dictionary to describe each transaction attribute and all of its possible values" [13].

The authors create their features using only transaction information and aggregated historical information. These transactions were classified using scikit-learn's random tree classifier with 100 trees. The random forest design are helpful for this project because it allows the relative feature importances to be calculated. In addition to this random forest, the authors use a deep feature synthesis algorithm to automatically derive behavioral features based on historical data of the card. Through these methods, they were able to reduce false positives by 54% [13]. Interestingly, they found that their

solution can maintain similar benefits when the historical features of a card are computed once every 7 days. This eliminates the need for streaming computing because the features can be computed after a period of time instead of using a constant stream of data. While the authors' system reduced the false positive rate, it did not do significantly differently than BBVA's current system in determining false transaction, meaning that BBVA's system does well in detecting high valued fraud. However, Wedge et al's system still reduced total cost to this bank by 190,000 euros. As a conclusion, this system is helpful in reducing false positives in a way that does not increase latency to a noticeable point for users, while saving banks and firms significant amounts of money.

### 2.1.1 Financial Fraud Research

Though credit card fraud is under the umbrella of financial fraud, there is a large field of study devoted to financial fraud as a whole. This type of fraud can be anything related to financial transactions, and anything that a bank may be responsible for. In a 2011 paper, Johan Perols analyzed the best methods analyzes statistical and machine learning algorithms for financial statement fraud detection [7]. In the United States, the cost of financial statement fraud is estimated at around $572 billion per year. With any type of financial fraud, there is a certain level of financial burden for the company or bank that has to pay out from fraudulent actions. Other than the pure fraud costs, having high levels of fraud leads to uncertainty for both the customers and bank trustees. This uncertainty then creates higher transaction costs for all and less efficient markets.

As is the case with credit card fraud, there is a high imbalance between fraudulent financial statements and non-fraudulent statements. With financial statements, it is much more costly to a bank or a firm to classify a fraudulent firm as a legitimate firm than it is to classify a legitimate firm as a fraud firm. This is because fraudsters will continue to cost a company or bank money until they are caught. In addition, fraudsters are actively

trying to cover up their fraud, making fraud detection increasingly more difficult. For these reasons, Perols is looking for the best classification method with the most utility. The results of this paper are of particular use to institutions like the Securities and Exchange Commission as well as other auditors. In this study, Perols uses open source classification algorithms from a data mining tool called Weka [7]. From Weka, six algorithms were chosen for analysis: artificial neural networks (ANN), support vector machines (SVM), C4.5, logistic regression, stacking, and bagging. The experiment was conducted using a dataset consisting of fraud investigations reported in the Accounting and Auditing Enforcement Releases from 1998 through 2005. Similar to other studies referenced in this thesis, the author enforces the necessity to not rely upon accuracy to measure the success of an algorithm for a skewed class problem.

To test the different models, the author uses a ten-fold cross validation instead of using the same dataset as both the training set and the test or evaluation set. The results of this study are somewhat surprising and do not follow other research. Here the author finds that SVM outperforms the other classification algorithms, followed by logistic regression then bagging. This is a surprising result because previous research has found neural networks to have better results at classifying fraud. In addition, "out of 42 predictors examined, only six are consistently selected and used by different classification algorithms: auditor turnover, total discretionary accruals, Big 4 auditor, accounts receivable, meeting or beating analyst forecasts, and unexpected employee productivity" [7]. It is interesting that the different training algorithms were picking out only a small portion of the same metrics to judge these transactions on. As an extension of Perols' research, future research could examine other classification algorithms, and leverage data mining research focusing on the class imbalance problem.

## 2.2 Other Fraud Research

While this thesis mainly focuses on fraud research in a credit card and online payments format, there has also been much research into fraud in other forms. However, the broader goals of this research align with the goals of credit card fraud detection, and the results are widely applicable. An example of this is a paper by Fawcett and Provost from 1996. This paper, "Combining Data Mining and Machine Learning for Effective User Profiling," describes automatic methods for fraud detection based on profiling customer behavior [4]. Instead of focusing on credit card data, this paper uses cellphone cloning data, which is a particularly expensive type of fraud. This is when a customer's Mobile Identification Number and Electronic Serial Number are cloned and programmed into a cellphone that does not belong to that customer.

In this paper, the authors present a framework for automatically generating fraud detectors. Here, the model uses a lot of calls to determine fraudulent patterns then use these broader patterns and apply them to individual accounts. Some of the standard methods used to detect fraud in these calls are to search for collisions or overlapping calls between the original user and the fraudster, or to search for calls in terminal proximity that could not have possibly been placed by the same user. The profilers the paper describe capture the typical behavior of an account and calculate how far an account is from typical behavior. Given this information, the authors use rule learning program, which searches for rules with certainty factors above a user-defined threshold. Here, each account has its own set of rules, and each call has 313 attributes that allow for partitions of the calls. Through their data mining process, 3630 rules were created for detecting fraud, which were then narrowed down to 99 rules. The conclusions of this paper were that the authors had to sacrifice accuracy in order to reduce the total cost of the system, but that their method was overall effective [4]. Like other papers, the authors highlight the need to build adaptive systems to account for ever changing fraud.

Crowdsourcing is another way fraud is introduced into daily life, introducing new types of malpractices into Internet advertising. In a paper by Tian et al, authors attempt to detect crowd fraud in internet advertising [11]. In this paper, authors are focusing on detection when malicious crowdsourcing platforms attack other advertisers. These types of attacks are much harder to detect than automated attacks because they are human generated crowd frauds, and ever changing. Here, fraudsters manipulate pay per click method of payments to make more money. Crowd fraud is defined by a few characteristics: moderateness, synchronicity, and dispersivity. Crowd fraud often arises from a vast number of attacking sources, but each source has a low fraudulent traffic. These attacks are meant to hurt advertisers by raising their advertising expenses. There are similar paradigms in credit card fraud, where fraudsters use multiple fake identities and credit cards to make a lot of low cost transactions. As of the time of this paper in 2015, current methods for crowd fraud detection rely on previously known knowledge and rules based on suspicious queries. However, this is very labor intensive.

To detect crowd fraud automatically, Tian et al use an enhanced graph model based on anomaly detection methods for detection coalitions. There are a few main stages to this system. First, the authors construct a surfer-advertiser bipartie graph, where each edge represents a click log. Then, the authors look for clusters to find surfer coalitions that exhibit synchronicity, and then filter out the large coalitions. Once these large coalitions are determined, they can be removed from the domain of the centralized advertisers. Before constructing these graphs, the authors also pre-filter to remove large amounts of non-fraudulent data; in this case they remove more than 70% of the click logs. The authors test their nonparametric clustering algorithm on real world data and find that the system does indeed converge and scale at a linear rate, with a 98.7% accuracy in finding malicious coalitions [11]. Through converting this coalition detection into a clustering problem, the authors are able to get very high results in the crowd fraud

detection field.

Another type of fraud in everyday places is ranking fraud for mobile apps. This ranking fraud is committed in order to move apps up the ranking lists. Here, mobile app developers are using shady means to artificially raise their app's ranking by inflating their apps' sales or posting phony app ratings. Zhu et al investigate ways of accurately locating the ranking fraud by mining the active periods, mainly focusing on detecting local anomalies versus global anomalies of app rankings [15]. With over 1.6 million apps in the Apple App Store and Google Play, app leader boards are an important marketing tool for developers. However, developers can manipulate their ratings by implementing "bot farms" or "human water armies" which increase app statistics like downloads and ratings in a very short time. App ranking fraud detection is particularly difficult because the fraud can occur at any point of the app's life cycle, which is why the authors focus on local fraud detection instead of the global anomaly of mobile apps. Another challenge for this problem is that there are a vast number of apps and no easy way to determine the ones that have committed fraud, enforcing the need for automatic fraud detection methods. Lastly, the authors must find implicit fraud patterns as evidence for ranking fraud because of the dynamic nature of mobile app rankings.

In order to detect leader board fraud, the authors actually look for leading sessions of an app, and find that fraudulent apps' leading session ranking patterns have different characteristics of normal apps. Zhu et al use statistical hypotheses tests to find evidences for ranking fraud, then use an unsupervised evidence-aggregation method to combine the three types of evidences. The three types of evidences the authors focus on are ranking based, rating based, and review based, with each of these evidences having several factors built into them. By testing their model in different permutations of evidences, the authors find that their combination of all three evidences outperforms single evidence models and other baseline models. This can be because the app ranking fraud

does not necessarily cause app rankings to increase, but may lead to higher downloads or reviews. Therefore, it is more important to look at all the factors rather than the individual evidences [15]. Through their experiments, the authors showed that mining for evidences, and modeling them with statistical hypothesis tests, allows for an optimal fraud detector that can easily be extended both in this context and in other contexts.

## 2.3 The Human Element

In this section I will discuss some game theory and behavioral elements of fraud and ways of eliminating frauds and fraudsters.

CHAPTER 3

# FRAUD IN CONTEXT

## 3.1 Interviews from Professionals

This section will be about what professionals are doing in the fraud field.

## 3.2 Research Versus Reality

This section I will compare whatever I find out from professionals to the most modern research on this topic. I assume research will be slightly behind what these professionals are dealing with and how they are solving these problems.

CHAPTER 4

# IMPLEMENTING A FRAUD DETECTION SYSTEM

To understand further the complexity of creating a highly accurate fraud detection system, I implement my own system based on past research discussed in Chapter 2. I construct a series of machine learning models, and compare the results based on different proportions of fraud to non-fraud data. All experimentation is done open source tools built for python, such as Scikit-learn and Numpy. I will discuss the model and the experiment in depth in the sections below. The overall implementation is broken down into subsections under Experiment: Hypothesis, Data, Methods, Results, and Discussion.

## 4.1 Model

For this project, I will be implementing several different machine learning models. I will be following 2011 Perols and will be comparing the performance of several popular classification models. To start, I will be implementing logistic regression.

Logistic Regression is a way to estimate the best fit line through a set of points. The best fit logistic line is the line that minimizes the squared error of each point in relation to the estimation on the line. This algorithm is used to create a relationship between points, and to analyze the relationship between points [10]. As more elements are added to the estimation, the logistic regression increases in complexity. It is a particularly good method for classifying a binary variable because it calculates the probability that something is equal to 1 based on the given inputs. This is different than linear regression, where the best fit line is estimated in a straight line instead of a logistic.

Artificial neural networks leverage logistic regression to create more in depth decisions. Neural networks are created by combining multiple nodes in different layers to create a network [9]. Here, nodes represent logistic regression, as they take in inputs,

estimate an answer, and put out an output. The general structure of a neural network is to have an input layer, one or more hidden layers which can be of any size. Information is propagated forward through the network, and then back propagated again to adjust the parameter estimates of each node. Through this forward and backward process, the system eventually reaches a minimum, where the estimates minimize cost. Once the network weights have been estimated, any new input can be sent through the network, and an estimate will be output. This system is extremely powerful, and can be tuned significantly based on learning rate, hidden layer numbers, activation functions and other features of the network.

Decision trees are somewhat similar to neural networks, but instead of traversing a network, decisions are made through branching at features. Each parent node represents a decision or feature and the children nodes represent the outcomes of said feature. The goal is to minimize the cost as you traverse the tree, and reach a leaf node. Decision trees can grow arbitrarily large, and sometimes require pruning to avoid a very long prediction process. In addition, decision trees can be combined to create decision forests, further enlarging the architecture and complexity of the learning model [5]. Like neural networks, there are also many parameters to tune, and architecture decisions to make such as maximum depth and minimum number of training inputs.

The last classification method I will be discussing is support vector machines or SVMs. SVMs classify data into categories by finding the best way to divide the data [2]. This is particularly simple with a binary example like fraud detection. Here, we use kernels to calculate the plane or division that best splits the data with the lowest error rate. Like neural networks, to find the lowest cost split between the data, we implement stochastic gradient descent to find the minimum cost points. Once this division is created, predictions are made based on the location of your point of interest in regard to the line dividing the data.

## 4.2 Experiments

I will be experimenting with the models listed above to create the best classifying algorithm possible. The experiments will be two fold. I will be experimenting with the distribution of the data and with the parameters and hyper-parameters of the data. In Perols' paper, he finds the best results when looking at an equal split of fraudulent and non-fraudulent transactions [7]. I would like to replicate his study by altering the proportion of fraud to non-fraud, and compare results. I will also be experimenting with the structure of the models themselves. As discussed above, each model has several many elements that can be tuned to find the best results. Neural networks in particular can take many forms, and require certain levels of trial and error to find the best results. Through these two axis, I will compare the performance of each model both to other models in this thesis and performance in Perols' paper.

### 4.2.1 Hypothesis

Based on Perols' paper, I hypothesize that the best results will come with more equally distributed data. I think that neural networks will perform best because they are able to capture many nuances of the data, and may perform better than the other models I have selected. However, given enough data and the correct tuning, these models should theoretically approach the same accuracy levels.

### 4.2.2 Data

The data used from this project comes from a public source on Kaggle.com. I used the "Synthetic Financial Datasets for Fraud Detection" which was generated by the PaySim mobile money simulator [6]. This dataset comes from the paper "PaySim: A financial mobile money simulator for fraud detection" and is scaled down to a quarter of the

original dataset. The dataset includes information about type of the transaction, amount, the customer that started the transaction, the initial balance of the sender, the balance after the transaction of the sender, whether its fraud, and other variables. The summary statistics are shown in Table 4.1.

Table 4.1: Summary Statistics

| Transaction Measure | Mean | Min | Max |
|---|---|---|---|
| *Legitimate Transaction* | | | |
| Amount | 178,197 | 0.01 | 92,445,516 |
| Count = 6,354,407 | | | |
| *Fraudulent Transaction* | | | |
| Amount | 1,467,967 | 0 | 10,000,000 |
| Count = 8,213 | | | |
| *Cash In* | | | |
| Amount | 168,920 | 0.04 | 1,915,267 |
| Count = 1,399,284 | | | |
| *Cash Out* | | | |
| Amount | 176,273 | 0 | 10,000,000 |
| Count = 2,237,500 | | | |
| *Debit* | | | |
| Amount | 5,483 | 0.55 | 569,077 |
| Count = 41,432 | 17.54 | | |
| *Payment* | | | |
| Amount | 13,057 | 0.02 | 238,637 |
| Count = 2,151,495 | | | |
| *Transfer* | | | |
| Amount | 910,647 | 2.60 | 92,445,516 |
| Count = 532,909 | | | |

### 4.2.3 Methods

In order to implement my models, I will be leveraging the python package Sci-kit learn. Sci-kit learn has readily available implementations of major classifiers that I am interested in testing, as well as the infrastructure for predicting and analyzing the results of the models. I use the python package Pandas to read in and manipulate my data in a data-frame format. After reading in my data, I clean the data by selecting the fields I am interested in, and converting them to a usable format. For experiments where I am changing the distribution of fraud to non-fraud transactions, I extract the two transaction types from the total dataset. I then randomly sample a portion of the non-fraudulent transactions based on that experiment, and recombine that dataset with the entirety of the fraudulent dataset. This ensures that there is no selection bias in my results.

### 4.2.4 Results

This section will be full of tables and figures. There will probably also be an appendix with the entire set of experiments run. The base results are currently showed in Table 4.2.

Table 4.2: Results

| Method | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| *Base Results* | | | | |
| Logistic Regression | 0.9977 | 0.781 | 0.333 | 0.467 |
| Neural Network | 0.998 | 0 | 0 | 0 |
| Decision Tree | 0.9997 | 0.8947 | 0.909 | 0.9017 |

### 4.2.5 Discussion

This section will compare my hypothesis to the results, as well as my results to the model I'm basing it off of. I also plan on adding any notes I learn along the way of this

implementation.

### Limitations

This project was limited by the dataset I had access to. In the real world, a transaction would include much more information that is not available with the fields of Paysim. Banks and credit card companies for example have the ability to see exactly where a transaction took place. Some banks pull information from their own banking app on a user's phone and compare the location of the user's phone to the location of the past transaction. With credit card information, companies can detect fraud based on if transactions are occurring in two places very far away from each other. Since I do not have any sort of location information, I cannot leverage this type of rule based fraud detection. In addition, I do not have access to the name of the business a transaction is taking place at. By building out a user profile that includes frequent locations and businesses the user goes to, a detection system can detect if a person is shopping at a store that is out of the norm and calculate the likelihood that this is fraud.

CHAPTER 5

# FURTHER APPLICATIONS

## 5.1 Anomoly Detection

### 5.1.1 Cell Phone Fraud

### 5.1.2 Biometric Fraud

### 5.1.3 Bioinformatics

CHAPTER 6

# CONCLUSIONS

## BIBLIOGRAPHY


[1] Fraud - Definition, Types, Examples and Processes.

[2] Jason Brownlee. Support Vector Machines for Machine Learning.

[3] Philip K Chan and Salvatore J Stolfo. Toward Scalable Learning with Non-uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection. Technical report.

[4] Tom Fawcett and Foster Provost. Combining Data Mining and Machine Learning for Effective User Profiling. Technical report, 1996.

[5] Prashant Gupta. Decision Trees in Machine Learning  Towards Data Science.

[6] Paysim. Synthetic Financial Datasets For Fraud Detection — Kaggle.

[7] Johan Perols. Financial Statement Fraud Detection: An Analysis of Statistical and Machine Learning Algorithms. *AUDITING: A Journal of Practice & Theory*, 30(2):19–50, may 2011.

[8] Salvatore J Stolfo, David W Fan, Wenke Lee, Andreas L Prodromidis, and Philip K Chan. Credit Card Fraud Detection Using Meta-Learning: Issues and Initial Results 1. Technical report, 1997.

[9] S. Suryansh. Neural Networks: All YOU Need to Know  Towards Data Science.

[10] Saishruthi Swaminathan. Logistic Regression  Detailed Overview  Towards Data Science.

[11] Tian Tian, Jun Zhu, Fen Xia, Xin Zhuang, and Tong Zhang. Crowd Fraud Detection in Internet Advertising.

[12] Mac Wang. How to protect your business from online payments fraud - CSO — The Resource for Data Security Executives, 2018.

[13] Roy Wedge, James Max Kanter, Kalyan Veeramachaneni, Santiago Moral Rubio, Sergio Iglesias Perez, Banco Bilbao, Vizcaya Argentaria, and Spain Madrid. Solving the ”false positives” problem in fraud prediction Automated Data Science at an Industrial Scale. Technical report.

[14] Massimiliano Zanin, Miguel Romance, Santiago Moral, and Regino Criado. Credit Card Fraud Detection through Parenclitic Network Analysis. *Complexity*, 2018:1–9, may 2018.

[15] Hengshu Zhu, Hui Xiong, Yong Ge, and Enhong Chen. Discovery of Ranking Fraud for Mobile Apps. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):74–87, jan 2015.