COMBATING FRAUD BY LEVERAGING MACHINE LEARNING

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

Credit card fraud in the United States has been on the rise, reaching over $8 million in 2018 [3]. The majority of these are card not present transactions, which have become more popular with the increasing options for online transactions. While these transactions have large dollar amounts, they are not frequent, as they only make up around 1-3% of transactions. Therefore, it is difficult to create unbiased classifiers for fraud detection systems. In this thesis, I summarize related works on the topic of anomaly detection, feature engineering for fraud, and game theory. I interview practitioners in the fraud detection space to compare their knowledge and day to day methods with research standards. I lastly implement my own fraud detection system using a synthetic mobile payment transaction dataset, and discuss other applications for anomaly detection. From my experiments, I find decision trees to be the most effective classifier given my dataset, which corresponds to the practitioners' reliance on rule based systems.

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, in addition to my friends and family who supported me through the arduous thesis writing process. 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

Every day there are new ways to pay for goods or services. As shown in Figure 1.1, cash is a way of the past; mobile, peer to peer, and online payments through credit cards are the way of the future [2]. While these convenient payment methods are very exciting, the additional methods of payment lead to increased opportunities for fraud. Credit card fraud is on the rise, with the United States reaching $8 billion in 2018 as seen in 1.2 [3]. High valued fraud is happening all over the world, and it is necessary to find more efficient and effective methods to catch fraudsters to keep everyday users safe unaffected by fraud. For example, credit card fraud through online payments in Australia hit $476 million in 2017, rising $58 million from the year before [23].

Financial fraud increases costs to a company through several means. With any type of financial fraud, there is a certain level of financial burden for the company or bank that has to pay out. Other than the cost of the fraudulent transaction itself, 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. When banks lose money to fraudsters, card holders partially or entirely pay for this loss through higher interest rates, fees, or reduced rewards and benefits for customers. Without automatic fraud detection, a significant amount of overhead exists because of the cost of investigating transactions and employing these investigators [8]. 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" [26].

Fraud detection is an instance of anomaly detection. Fraud detection and anomaly detection more broadly are difficult problems because of the heavily skewed classes and potential for intermingled classes. These two features means that there are few examples

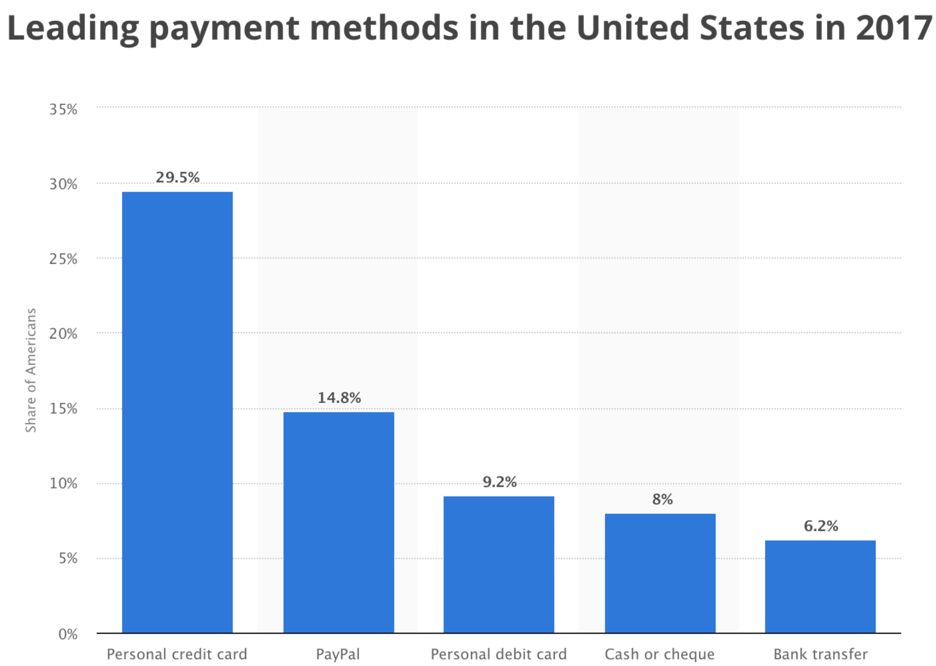

Figure 1.1: Leading payment methods in the United States in 2017 [2]

for classifiers to learn, and there may not be a clear line to divide classes by. Further, an effective fraud detector must be adaptive to new patterns and categories of fraud. The main goal of this thesis is to investigate current methods of financial fraud detection and implement one or more of those methods. I will describe the most prevalent methods for fraud detection in current use, as well as past fraud detection methods.

Chapter 2 reviews different methods used in the industry in the past and present for fraud detection. The background section will combine information from research papers with Chapter 3, which synthesizes information I learned from interviewing people at different companies in this field to provide some context for the literature review. My own implementation of a fraud detection system is described in Chapter 4 using a synthetic credit card transaction dataset. 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 5 will discuss other applications for successful fraud detectors. Chapter 6 concludes.

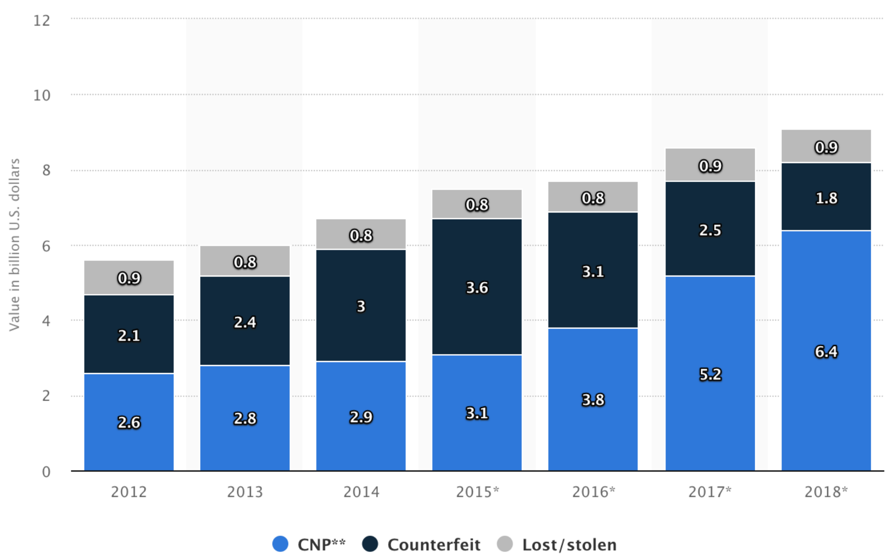

Figure 1.2: Value of payment card fraud losses in the United States from 2012 to 2018, by type (in billion U.S. dollars) [3]

CHAPTER 2

FRAUD DETECTION

Here we define fraud as "wrongful deception with the intent to gain personally or financially," or intentionally deceiving another person to obtain their property [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. It is important for businesses to balance finding all fraud and limiting their customers. A company may choose to accept some level of fraudsters in order to maximize the amount of legitimate users being classified as fraudsters. For example, in subscription based services, it can be less expensive for companies to accept all potential users to create new accounts because the cost of losing a potential client can be higher than the cost of allowing a small portion of fraudulent accounts. Because companies are normally profit maximizing, fraud detection in a business context will most likely always involve weighing the costs of false positives.

Researchers have more flexibility than a financial institution in their pursuit of fraud detection. Researchers are able to manipulate datasets by artificially changing fraud versus non-fraud proportions, and testing different combinations for the best performance. Researchers also have no time restraints as they do not need to make a split second decision of whether or not to allow or deny a transaction. Some may look for performance gains for in the moment decision making, but researchers can build systems that make decisions in any amount of time. The lack of restraints allows for a breadth of study into potential solutions to fraud detection.

2.1 Imbalanced Classes

Financial fraud detection falls under the category of anomaly detection because of the heavily skewed classes. Some datasets are perfectly split between two classes. This means that there are equal amounts of positive and negative examples for a classifier to be trained with. Imbalanced classes happen when there is more examples of one class over another, and is commonly used to refer to large differences in amount of examples. This imbalance makes classifying the minority class more difficult because the classifier has not seen as many examples of that class, and therefore the model's weights will be biased towards the majority class. Anomaly detection is an extreme example of skewed classes because the minority could be less than 1% of the examples. Therefore, it is likely a system will default to predicting only the majority class and still have high accuracy.

There are a few ways to model skewed data with classifiers. One basic and effective strategy is to get more data. With more examples of the anomaly, it is easier to detect a pattern between them. However, acquiring more data is not always an option, especially when working with data collected in the real world. Another popular method for dealing with data imbalance is to artificially balance the data. In a study by Chan and Stolfo from 1998, the authors find that artificially balancing the data leads to more accurate classifiers [8]. To balance the classes, one can under sample the majority class or over sample the minority class by creating more instances of minorities.

Chan and Stolfo also implement a cost-sensitive approach to measure the success of their classifiers. In most machine learning problems, the default metric to monitor a model is accuracy of a system. However, this is not an effective measure when discussing anomalies. For example, in my own implementation, if a classifier were to predict non-fraud for each transaction, the accuracy would be over 99%, but would still not be catching any of the fraud. Metrics that are often used to measure skewed class problems are precision, recall, and F1 score. These all combine ratios of true positives to false positives, analyzing the success in predicting the minority class rather than the majority class. Chan and Stolfo take a new approach, and instead use the average ag-

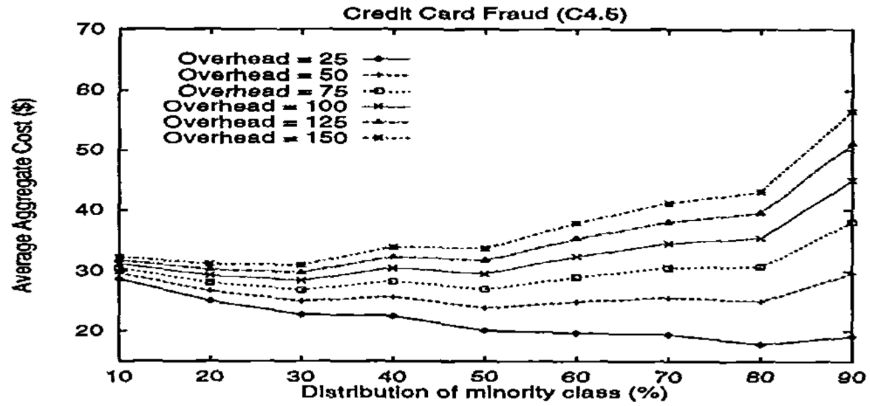

Figure 1: Training distribution vs. the credit card fraud cost model

Figure 2.1: Training Distribution vs. the credit card fraud cost model [8]

gregate cost of fraud as their metric for evaluating their model [8]. In addition to tracing the performance with different data distributions, the authors also look at costs including different levels of overhead costs.

Figure 2.1 shows the results of Chan and Stolfo's system. While they find that larger overhead leads to higher overall cost, they also notice that when overhead is smaller, cost is minimized at a larger percentage of fraudulent transactions in the training set. When overhead is smaller, a bank can afford to have a higher false positive rate, as the false positives can be investigated relatively inexpensively. When overhead is larger, a bank cannot afford as many false positives, as it is expensive to investigate all of the false positives, but would rather have a higher false negative rate. These results are helpful for banks who may be deciding where to focus their resources.

A 1997 paper by Stolfo et al uses Meta-Learning for credit card fraud detection [17]. Here, the authors similarly 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 (80%) and the lowest false positive rate (13%). The best method the authors discovered is using meta-learning with BAYES as a "meta-learner to combine base classifiers" [17]. For reference, BAYES is very similar to logistic regression, as it takes features as input and outputs a probability.

2.2 Credit Card Fraud

The papers in the previous section mainly focus on redistributing the imbalanced classes as a way to improve fraud detection rates. 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 [26]. 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. An example of parenclitic network creation is shown in Figure 2.2.

The authors take the computed network and transform it into a set of features compatible with a data mining algorithm like a multi-layer perceptron. The authors argue

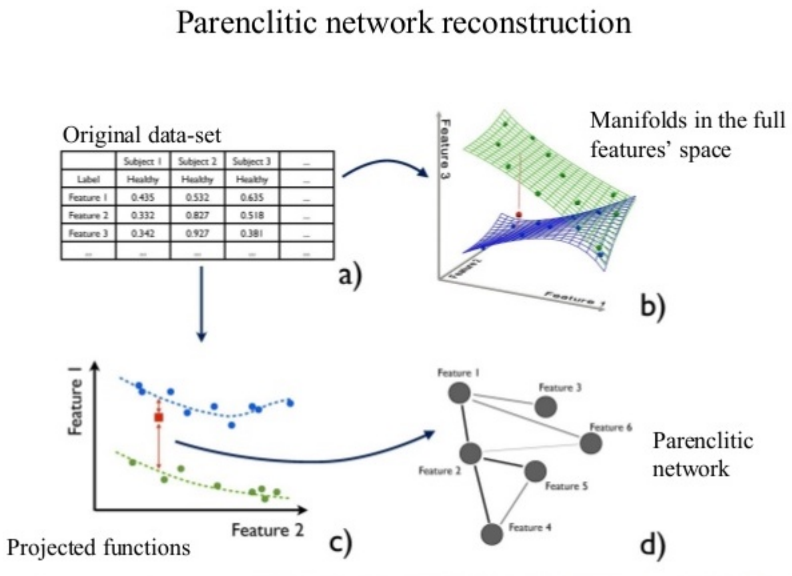

Figure 2.2: Parenclitic Network Formation [6]

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. Zanin et al combine data mining techniques to extract additional features from the 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" [26]. Succinctly, the data mining creates new features, and the network finds connections between these features. This study uses parenclitic networks, and evaluates them on a dataset of real transactions in comparison to the results of a ANN approach.

To test their parenclitic network approach, the authors used an anonymized 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. The authors have also synthesized information like average transaction size for 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

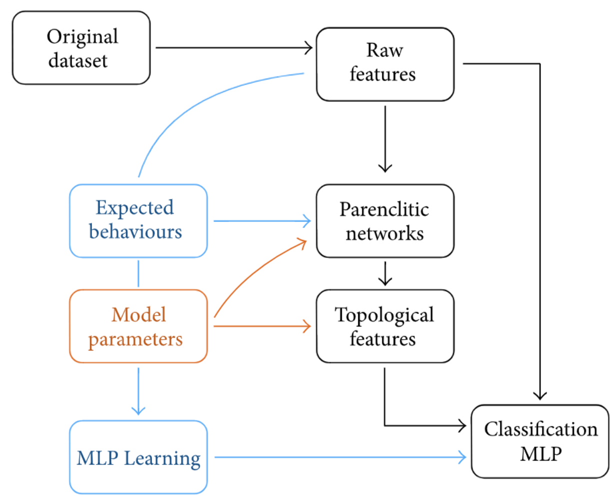

Figure 2.3: Overall system [26]

has a weight associated with it that can be adjusted. The overall system that the authors create is displayed in Figure 2.3. Zanin et al find that parenclitic networks alone are not enough to significantly lower the classification error. However, the addition of parenclitic features to the raw data set increases the overall information a model receives as input, decreasing the error rate from 19.2% to 12.23% [26].

In a 2017 paper, Wedge et al have attempted to solve the "false positive" problem in fraud prediction at an industrial scale [24]. The goal of this paper is to use a feature engineering approach to reduce false positives. It is estimated that only 1 in 5 transactions that are predicted to be fraud are actually fraud, and analysts have found that these false positives are costing more than the fraud itself [24]. 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. The dataset the authors use contains roughly 9.5 million transactions with approximately 112,000 fraudulent transactions.

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. Decision trees are effective at calculating relative feature importances. 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. This is achieved by looking for relationships between transaction entities, which are transaction groups with multiple transactions and features to describe each transaction. Through these methods, they were able to reduce false positives by 54% compared to the current BBVA solution [24].

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 in regards to the feature set because the features can be computed after a period of time instead of constantly updating the feature space. The author's system reduced the false positive rate, and reached an F score of 0.56 versus the BBVA system of 0.20. In addition, Wedge et al's system still reduced total cost to this bank by 190,000 euros. This research is helpful because it shows that features do not need to be recomputed with each new transaction, and that banks can reduce their false positives without increasing the latency for their users.

Both Zanin et al and Wedge et al have successful results, but demonstrate potential problems if fraud detection. Researchers are in a world without time constraints. Therefore, if a classifier takes long periods of time to train, it is not a major concern. However, if the lessons learned from research are to be applied to real world systems, it may be necessary to check potential latency gains for a user. For example, Zanin et al are looking to create more features that help capture what makes a transaction out of the norm. While these parenclitic networks were creating specific and meaningful features, they needed to be combined with features from the initial data. This means that the problem is getting more complicated and computationally expensive, adding time to training. Similarly, Wedge et al are working with a large feature space because of the dataset they were given. This draws attention to the need to preprocess data to ensure time is not being wasted on features that do not benefit the model. Without specifying useful and unique fields, training a model with millions of transactions and hundreds of fields becomes a computationally exhaustive calculation.

2.2.1 Financial Fraud

Credit card fraud is an instance of financial fraud. There are many other types of financial fraud including anything related to financial transaction and anything a bank may be responsible for. In a 2011 paper, Johan Perols analyzed the best statistical and machine learning algorithms for financial quarterly statement fraud detection [14].

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 during an audit. It is difficult to find fraud because 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 highest utility based on overall error rate and true positive rate.

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 the data mining tool Weka [14]. 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 reinforces the necessity to measure the success of an algorithm for a skewed class problem with metrics other than accuracy.

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. This is unlike most studies that split the dataset into a training set, evaluation set, and potentially cross validation set. The results of this study are somewhat surprising and do not agree with other research. Here the author finds that SVMs outperforms the other classification algorithms, followed by logistic regression then bagging. 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" [14]. Though these models were all given the same features, the training algorithms only picked out a small portion of the same features to judge transactions on.

2.3 Other Fraud

Other forms of fraud detection face similar challenges to credit card detection, and their results are therefore applicable to credit card fraud detection. An example of this is a paper by Fawcett and Provost from 1996. This paper describes automatic methods for fraud detection based on profiling customer behavior [10]. Instead of focusing on credit card data, this paper focuses cellphone cloning, which is an 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. 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, forming a set of features for a user which are used calculate how far an account is from typical behavior. Given this information, the authors use rule learning programs, 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 by about 1% in order to reduce the total cost of the system, but that their method was overall effective, lowering costs from $20,000 when predicting all as fraud to approximately $3,303 using their system [10]. Like other papers, the authors highlight the need to build adaptive systems to account for evolving fraud.

Crowdsourcing is another channel of fraud, introducing new types of malpractices into Internet advertising. In a paper by Tian et al, authors attempt to detect crowd fraud in internet advertising [20]. 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. 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 through manipulating pay per click advertisements. There are similar paradigms in credit card fraud, where fraudsters use multiple fake identities and credit cards to make many 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, creating a set of rules is very labor intensive.

To detect crowd fraud automatically, Tian et al use an enhanced graph model based

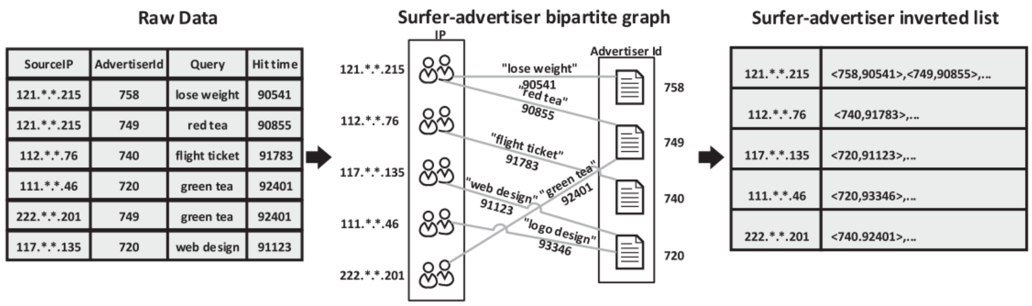

Figure 2.4: Bipartie graph visualization [20]

on anomaly detection methods for detection of coalitions, or groups attacking the same target. 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 as shown in 2.4. 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 sections 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 converges and scales at a linear rate, with a 98.7% accuracy in finding malicious coalitions [20]. Through converting this coalition detection into a clustering problem, the authors are able to build a scalable and accurate crowd fraud detection system.

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 [27]. Here, global anomalies refer to anomalies over the lifespan of an app, versus local anomalies which refer to shorter time periods that span the app's lifetime.

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 somewhat hidden 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 active periods or '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 hypothesis 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 features built into them. By testing their model with 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 features rather than the individual features [27]. Through their experiments, the authors showed that mining for features, and modeling them with hypothesis tests, creates an effective fraud detector that can be extended both in this context and other contexts.

2.4 The Human Element

Credit card fraud detection, while automated by machines, benefits from the ingenuity of humans. Fraud and fraud detection is a cat and mouse game, where each player is looking to outwit the other. Fraud systems adapt to fraudsters' patterns, while fraudsters adapt once they learn the rules of the system. This back and forth creates a complex and ever changing game between the two parties. Incorporating human intuition into a fraud detection system increases the accuracy of a system, and can also help in fraud prevention.

2.4.1 Credit Card Fraud Detection with Game Theory

In a paper by Vasta et al, a rule based system is combined with Game-theory [21]. With credit card fraud, the fraudster and fraud detecter have conflicting motives, each trying to maximize their own payoff through a multistage game between the two players. Here, fraudsters are considered to be rational adversaries, meaning that fraudsters are attempting to maximize their gain without being detected. Companies are trying to minimize their losses to fraud, and therefore the game between the two parties becomes a maximization-minimization problem. By using game theory, companies can start to predict future attacks while also providing possible courses of action to defend against them.

Credit card fraud detection is not like a traditional game. With credit card fraud, an attacker can make multiple moves insofar as the fraudster can continue to commit fraud until it is caught or sees some action from the fraud detection system. In addition, the attacker can act without time constraints; a fraudster can calculate their move for as long as possible. Lastly, the possible set of moves in this game is not well defined because of the adaptive nature of fraud and fraud detection. This is very similar to database

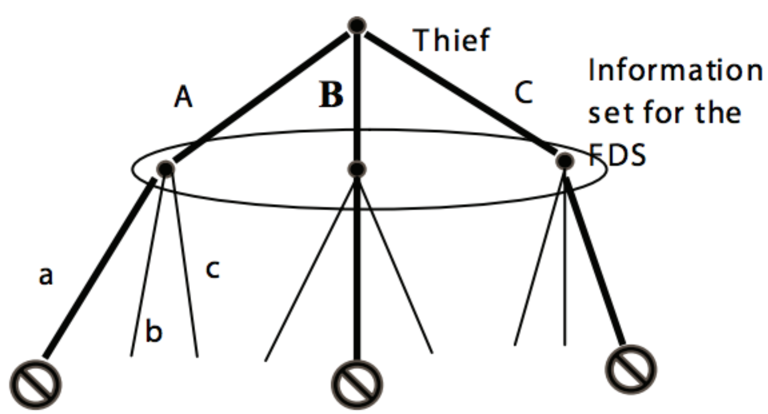

Figure 2.5: Rule Based Game Theoretic Approach: Search Structure

intrusion detection as well.

Information is the main currency in this game. To begin, there is a lack of information for both parties: a detection system does not know whether or not a party is a fraudster, and a fraudster is not aware of the exact rules of the system. A fraudster will launch attacks, aiming to learn about the rules of the system, while the detection system learns from the attacks being launched at it. In a repeated game environment, a fraudster can gain information to arrive at a Nash Equilibrium, where they cannot play anything better given the current strategy of the fraud detection system. Because of the lack of information present, this game is called information warfare.

A transaction will go through two tiers before a final decision is made: a rule based tier and a game theory tier. The rule based tier calculates a suspicion score for each transaction; if the transaction fails the rule system, it is then sent through the second tier. In the second tier, a heuristic is calculated by going through the feature space. The authors model the problem based on parameters about the type of fraudster it is, the strategy space for both parties and the action space for both parties. The structure of how the heuristic is calculated is similar to a decision tree, as shown in Figure 2.5. The benefit of using this system is that if the system thinks that the fraudster is learning its strategy, it can change strategies, or can strengthen its beliefs if it thinks its learning the type of fraudster.

Panigrahi et al similarly combine game theory with a rule based filter to create a fraud detection system. This system varies from Vasta et al's because it has 4 distinct phases. The system begins with a rule based filter, similar to those perviously described. Rules include an address check and outlier detection, where the system checks how much a transaction varies from a user profile as well as general rules. From this rule based system, transactions are put into three categories: fraud, non-fraud, and suspicious [12].

To combine these rules, Panigrahi et al use Dempster-Shafer theory, followed by a transaction history database, and a Bayesian learner [12]. The Dempster-Shafer theory assigns a fraud probability to an event based on the classification the transaction has from the rule based system. Dempster-Shafer theory can be found below in Equation 2.1. For example, if a transaction is classified as non-fraud in the address portion, it will have 0 added to its overall fraud probability prediction. Those results are then fed to a transaction history database, which compares the current transaction with the user's past transactions as well as a generic database of fraud.

$$m(h) = m_1(h) \oplus m_2(h) = \frac{\sum_{x \cap y = h} m_1(x) * m_2(y)}{1 - \sum_{x \cap y = \emptyset} m_1(x) * m_2(y)} \tag{2.1}$$

Information about how much the transaction is deviating from normal behavior in 4 mutually exclusive cases is then sent to the Bayesian learner. The Bayesian learner then calculates a suspicion score, which is combined with the earlier suspicion score using Dempster-Shafer again. Based on a threshold, a transaction is then classified as fraud or non-fraud. A visualization of this system can be seen in Figure 2.6. Through

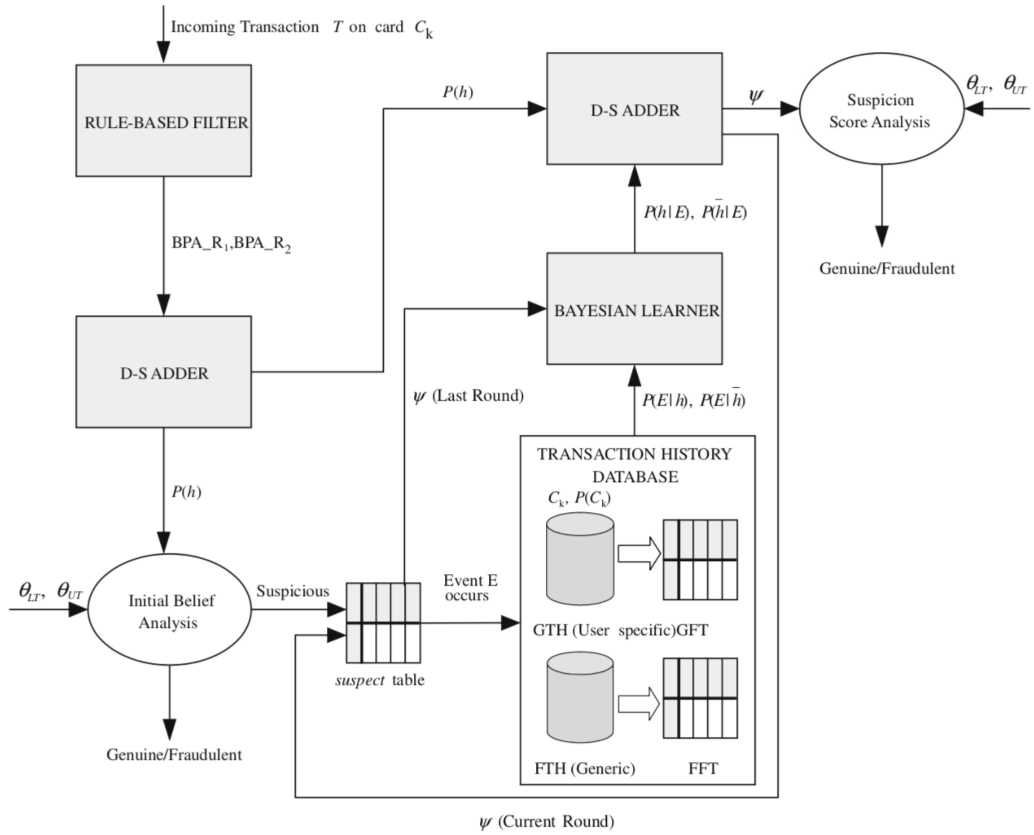

Figure 2.6: Architecture of Rule Based Filter, Dempster-Shafer Adder, Transaction History Database, and Bayesian Learner

their experiements, Panigrahi et al find that 0.3 and 0.7 are the most accurate lower and upper bounds, respectively, and that their system increased the true positive rate by about 15-20% [12].

2.4.2 Auditing with Game Theory

Game theory concepts can also be leveraged in the auditing context. Wilks and Zimbelman study the current financial auditing methods and offer suggestions for improving systems adding game theory. The authors describe that there are several levels of reasoning: zero order, first order, and higher order [25]. Zero order logic occurs when a user is only considering conditions affecting themselves. First order is when a user considers some conditions, and higher order is when a user considers multiple levels of reasoning and layers of complexity. Since fraud and fraud detection are games highly reliant on opponent's moves, strategy can change drastically based on response from the other player. This is inherently difficult because either parties' next move is highly dependent on the other's move.

Currently, the best practices of the auditing field require auditors to go through a checklist of "red flags". While this is very effective for first order reasoning, it does not catch any fraudsters that are reacting to changes in the "red flag" checklist. The authors point out that these checklists have been shown to be less sensitive than intuitive fraud risk models. This is in part because of the dilution effect, where an auditor is focusing on too many irrelevant cues instead of the relevant cues [25].

Instead, the authors suggest decomposing the decision into multiple components, making a judgement on each of these components, and recombining to make a global decision. The three elements the authors contribute to fraud are known as the "fraud triangle": incentive, opportunity, and attitude [25]. The authors also suggest that having auditors meet with their clients and look for in person cues helps auditors learn cues and potentially fraudulent behaviors. Through this process, Wilks and Zimbelman find that accuracy improved in low risk fraud situations, but did not improve high risk situations. They also found that attitude cues are easily concealed, and that auditors often were anchored to their initial judgements, and were unlikely to change their minds given new information.

Through their experiments, the authors offer advice that can be applied to audits as well as credit card fraud detection. The two main components the authors suggest adding are unpredictability and strategic reasoning. Unpredictability can be achieved by changing the time line of the fraud detection or audit, as well as changing the way the system works from time to time. This unpredictability can help catch fraudsters that were not anticipating an audit, and therefore might not have hidden their work as well [25]. The authors also suggest adding opportunities for auditors to learn strategic reasoning, and increase their intuition.

While this is true of humans, it is also true of fraud detection systems, as models also need the opportunity to engage with new fraudulent behavior to learn new patterns. Through incorporating human intuition into computerized systems, fraud detection systems can continue to move towards taking all analysts out of the process of fraud detection. Without it, companies will continue to argue that humans have intuition machines do not, requiring overhead from manual transaction processing.

CHAPTER 3

FRAUD IN CONTEXT

Most computer science research is dedicated to finding the most accurate and most efficient methods of solving a problem. This is a great standard for research, but in many contexts, results from research may not be as applicable in the real applications. In an attempt to understand the differences between related works in fraud detection and companies detecting fraud in the present. Through these interviews, I gained more knowledge into real world fraud detection.

3.1 Interviews from Professionals

To gather a sense of business level fraud detection methods, I interviewed four professionals in the industry. Two of the people I interviewed have directly dealt with credit card fraud in their line of work. The third person I interviewed has worked in the British government, then transitioned into insurance fraud detection, and has spent large amounts of time tracing insurance fraudsters. The last interviewee has a doctorate in engineering, but has worked in fraud detection for many different industries. As a caveat, I will not be referencing these participants by name or by organization. Through these interviews, I found that fraud is handled slightly differently depending on the principles and restrictions of a business.

To start, my first credit card fraud expert began by explaining the different forms of fraud that a company looks out for. There are two main types of transactions that credit card companies are concerned with: physical and non-physical. There are a plethora of ways to spend money online that do not result in physical purchases, and each type of purchase comes with its own set of information. The straight forward, rules-based method of detecting fraud comes from the billing address zipcode, CVV, and computer timezone. When purchasing online, the first step is to verify that a CVV does indeed match with the credit card number. After this, one can check that the timezone of the zipcode is close to the timezone the computer is in. While a business will get a negative response from the credit card company if the CVV or the zipcode does not line up with the saved information, the timezone on a computer is often faked leading to further deception. With physical goods being sent to an address, one way of detecting whether it is fraud is to check if the delivery address is a normal building, and not an abandoned warehouse. Location can also be verified in other contexts such as airline credit cards and banking apps. Airline cards for example can keep information about when you are going to fly places, and not cancel your credit card when you appear in a different country after only a few hours. Banking apps can pull out location information from an app request independent of a transaction to verify that the user is in the same place.

Fraudsters have found sly paths around many fraud detection systems. While it would be optimal to cut off all paths of fraud, credit card information is continually circulated and sold to fraudsters. However, these fraudsters don't know which cards are currently in use. One way fraudsters have found to solve this problem is through leveraging free trials. A fraudster can sign up on Spotify or Netflix for example for a free trial. If the card information is good enough to pass these slightly less stringent free trail gates, that means the card information is good to use in other situations. Once the card is used in real life, the owner of that card gets the charge. To make these fraudulent dealings look more realistic, fraudsters have started using computers from the area associated with the card and using that computer for a reverse geoIP, which can then be used to proxy the transaction. Through this process, these transactions look more legitimate because the locations seem plausible.

While fraudsters have technical solutions to help hide themselves, there are methods of detecting this behavior. When a fraudster gets a list of credit cards, they usually use a bot to go through the list of cards and do the process mentioned above. These series of transactions normally take too little time for a human to have completed, and come from the same IP address. The interviewee's company analyzed these sort of bot transactions, looking for the same bot trying to do similar fraudulent transactions on multiple sites.

Surprisingly, many companies choose to buy fraud detection software rather than working in house. Banks for example use Falcon software, which looks at the flow of transactions in order to pinpoint suspicious transactions. Banks are an interesting case because they have such a high volume of transactions that they can afford to do these type of velocity checks. On the merchant side of the world, most physical goods merchants don't use much fraud detection at all and rely on the chip and basic credit card security. This may be because in the past a merchant would not be responsible for a fraudulent transaction if they had a swipe of a credit card and a signatures. It is also a question of risk versus reward and what you are selling. If you are selling diamonds, you may be more concerned with fraudulent transactions than if you are selling Beanie Babies. There are many filters for online transaction fraud such as Retail Decisions as well as 41st Parameter. These filters check for many red flags, including if the same browser is buying again and again. Companies with longer term transactions have the flexibility to involve customer service members in the fraud detection process, as they have time to verify a transaction for a tangible good. For intangible goods such as Netflix, there is less flexibility as it is an instant decision.

One last example the interviewee mentioned explains how fake physical credit cards are constructed. Once you know you have a valid card, you can magnetically create a card using a room key. While this key does not look real, you can go buy gas with it and if people aren?t paying attention go to the store. If you are more sophisticated, you can clone the card, which is much harder to do with the chip so strip will eventually go away so when you are using the card you?re much less likely to be fraud. An online fraud example involves selling iPhones. A fraudster gets a spam person to ship a real phone, take packages and mule for the fraudster and then Amazon or whatever vendor is out that cost. These plans work effectively only if not caught by local agencies or the Counterfeiting and Banking sector of the Secret Service.

The second fraud export I spoke to is part of a broader fraud detection team at a company with multiple product lines. The specific team works to detect and handle fraud and abuse across all of the company. However, he in particular helps on the credit card fraud detection sector. Since this company is larger, it has the resources to create its own fraud detection systems, which are heavily biased towards applied machine learning techniques. While this is a modern company, according to the interviewee, they use older machine learning techniques and apply them to new problems. They are not necessarily looking to upgrade their systems when new research comes out. Instead, they look for ways to enhance their system based on the problems that come up. They focus on supervised and reinforced learning for their fraud detection.

Like some of the related work, the interviewee emphasizes how fraud and security are very different problems than others because the problem is always changing. A contrary example to fraud detection is training a model to detect cancer using MRIs and scans. Here, once the classifier is trained, it will be mostly accurate for its niche. With fraud detection, by setting a model, fraudsters will change their behavior accordingly. Therefore, one classifier will never suffice for long periods of time. Because of the heavy interactions with humans with poor intentions, it is best to combine the machine learning with game theory in order to both catch and block fraudsters. For example, there is a push and pull relationship when attempting to find fraudsters. While attempting to pursue a fraudster and cut off their fraud channel, there is a risk of losing that fraudster again. According to the interviewee, there is a lot less push and pull when using a specific model, rather than changing models frequently.

Though model choice can affect results, the interviewee argues that it is not the most important element to accuracy gains. There are large differences in the macro level techniques like convergence models, linear models and others, but there is not a particular model that is perfect. Since the problem is changing so quickly, it is best to find the model that is closest without trying to find the perfect solution. Instead, the dataset is much more important than the actual model. With changing fraudulent behavior, it is better to implement new parameters rather than change the model.

However, there are draconian restraints on data a company can have as well as privacy concerns. For example, this company does not track users locations for use in fraud detection, and restrict other elements of the data before ever reaching the fraud detection team. The mantra here is to put the fraud detectors in a position where it would be impossible to do bad things to people in the future based on the data they have. The loophole here is that some of the restricted data can be used by running the models on device rather than at the company, meaning the fraud team or company as a whole never sees restricted elements of data. These restraints can make fraud detection difficult because the models have to learn user's behavior without seeing the data and gauge their "health" in the system. Through these constraints, fraud detection becomes an even more difficult problem.

The next person I interviewed has had a large range of jobs, investigating, analyzing, managing, and developing intelligence in all incarnations of fraud and corruption. While he is focused in insurance fraud at the current moment, he has worked in welfare fraud, investment banking "Rogue Trader" profiling, and terrorist funding to name a few. His team and his work in investigating terrorist funding has actually prevented a few bombing attacks on London, emphasizing again the importance of fraud detection.

In the industries this person has worked in, he has become a proponent of fraud analytic systems in rule based systems with human input. In his experience rule based systems are less expensive and some what more effective than feature based systems.

Feature based systems are expensive because of the work to continually increase and change the feature space. Rule based systems are easy to change and with a field investigator looking over the results is a highly effective an inexpensive solution. FICO for example has a rule based engine. For any of these systems, in Britain, there is a large repository of information about people which is incorporated with census data. This information can be leveraged by British companies for fraud detection. While insurance fraud is different than financial fraud because it is on the back end of an issue and financial fraud is trying to be stopped in the front end, they still have a lot of overlap of methods.

This insurance fraud expert referred to some of the work done by the last person I interviewed in the behavioral analytics space. The form of fraud detecting used behavioral analytics to try to spot fraud before the fraud even occurs. The system measures the likelihood of someone committing fraud at a high degree, and the system was able to reduce deductions in payouts for insurance claims by about 12%. However, this sort of system is the "Rolls Royce of fraud detection" as it is very expensive. In the insurance fraud expert's experience, organized crime will employ their best people to combat the fraud detection systems and threaten people on the inside to get their purpose. Therefore, it is quicker to get a rule based system up to speed in the short term, though feature based systems would most likely pay off in the long term.

In addition to the behavioral analytics, the last person I interviewed has also worked in banking, insurance, and online gambling fraud. Each of these fields are somewhat different because of the levels of security and data. Banking is probably the most traditional and secure sector. They are very slow and need to anonymize all data before taking it out of their premises. While all industries have certain levels of regulations, banking is particular strict with their security and users' privacy. However, because of the amount of transactions banks handle, they can provide researchers with much more data than insurance. Banks also have to have a reason for suspecting fraud for regulatory reasons, which can be difficult when using machine learning models.

Insurance and gambling are slightly different than banking. One of the major issues with fraud detection in insurance is that sometimes there are no fraudulent transactions in a data set, and the actual amount of data is small. To get more data, insurance companies are attempting to get users to use their apps and extract more information from there. Gambling is different than either of these fields because it is the most open and have more relaxed laws than banks or insurance companies. Because of this, they are able to work much faster. Most techniques are transferable between these industries, but it is still important to have experienced people working on the problem, as the interviewee before also referenced.

3.2 Research Versus Reality

Through talking with these professionals, one point that was reemphasized was the cost of false positives over false negatives. This was the topic of discussion in many of the papers I read, and the professionals I talked to continued to stress the point. An interesting point to this is that companies generally accept some amount of fraud in their business, with about 1% of transactions being an acceptable amount of fraud for a company. While 1% is quite small, this is a large amount of money when you consider some companies may be having millions to billions of transactions go through their system every day. Because of this, it seems that using a cost equation based on cost of the size of a transaction versus the number of correct and incorrect transactions would be beneficial. This way, the system would learn to pick up on high valued fraud instead of many small transactions.

The experts I interviewed all incorporated more human insight into their detection system than much of the research would suggest. Based on the very accurate results most of the related work has achieved, it would seem that companies would be able to fully automize their fraud systems. However, human intuition has continually trumped automatic systems in accordance to the four interviewees. Human involvement seems to have three factors associated with it: intuition, cost, and regulations. The intuition a human can bring to a fraud detection system relates to game theory. Fraudsters and analysts are essentially playing a game where both parties are profit maximizing, and are in a cat and mouse situation. Using human logic about how a fraudster will act is something only expensive machine algorithms may be able to do. A human can set up a rule based system using knowledge from past fraudulent behavior and expectations of where weaknesses in the system might lie. A behavioral machine learning algorithm is more computationally expensive, meaning that companies may not want to invest the money into a complex system.

Lastly, humans will most likely always have to be part of the process because of regulations. When a customer is refused for a loan or creating a new credit card, there must be a reason associated with that decision to ensure there is no bias in the institution. Unlike somewhat unpredictable machine learning algorithms, a fraud analyst can easily point to how a rule based system created a denial decision. Through these interviews, I have found the reverse of what I expected: companies are actually not keeping up with newly researched methods but are actually opting for less expensive steadfast systems that have been in place for years.

CHAPTER 4

IMPLEMENTING A FRAUD DETECTION SYSTEM

To understand further the complexity of creating a highly accurate fraud detection system, I implemented my own system based on the approaches discussed in Chapter 2. I constructed a series of machine learning models, and reported accuracy and F1 scores based on different proportions of fraud to non-fraud data. For all experimentation, I used open source tools built for Python. 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.

Linear regression is a way to estimate the best fit line through a set of points. The best fit 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 [19]. As more freatures are added to

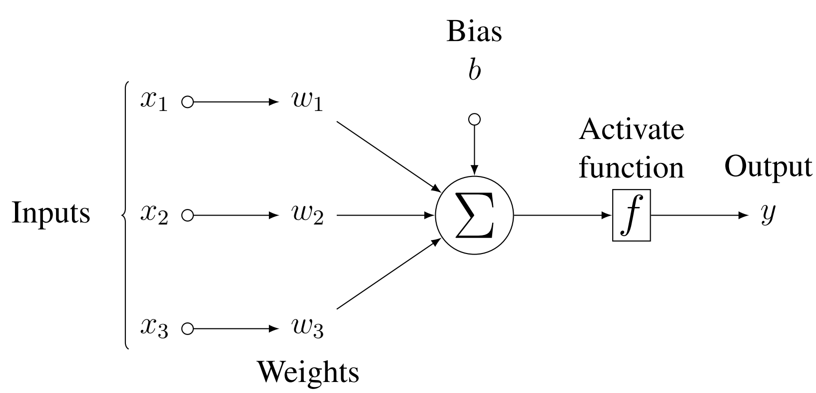

Figure 4.1: Logistic Regression

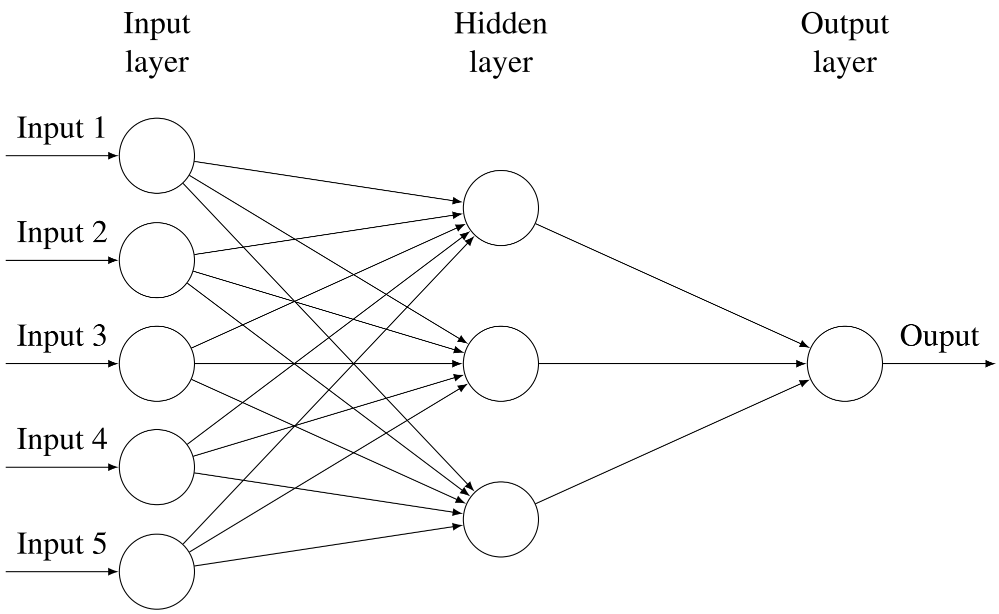

Figure 4.2: Neural Network

the estimation, the regression increases in complexity. Logistic regression is different from linear regression only in the equation used to estimate the relationship between the points. Logistic regression is a wave shape that has limits at 0 and 1. Along the wave represents probabilities of an event given another feature. For example, logistic regression can model studying for a test: based on the number of hours you study, you can calculate the likelihood of passing that test. 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.

Artificial neural networks leverage logistic regression to build systems that can solve more complex problems. Neural networks are created by combining multiple nodes in different layers to create a network [18]. Here, nodes represent logistic regression, as they take in inputs, estimate based on those inputs, and output an answer. The general structure of a neural network is to have an input layer, one or more hidden layers which can be of any size, and an output layer. 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 difference between the predictions and the real values is minimized. After the training period, the network has now been trained, and can calculate an output for any new input. 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. Cost functions are used to find the most homogeneous branches, or branches with similar inputs. Decision trees can grow arbitrarily large, and sometimes require pruning to avoid a long prediction process. In addition, decision trees can be combined to create decision forests, further enlarging the architecture and complexity of the learning model [11]. When using a decision forest, a tree would act as a branch in a single tree, focusing on learning on feature. Like neural networks, there are also many parameters to tune, and architecture elements to adjust such as maximum depth and minimum number of training inputs.

A slightly different approach to this classification problem is using probability distributions such as a Gaussian distribution. The idea behind this classification is that anomaly data is almost always not the anomaly. Therefore, assume all the data you have is the majority class. If this data is approximately normally distributed, then you can calculate the probability of a new example being part of the majority class. Since anomalies are infrequent, they have low probabilities of being part of the majority class, and therefore are most likely anomalies. The last step of this approach is to choose a threshold for how likely an example must be to be classified as an anomaly.

The last classification method I will be discussing is support vector machines or SVMs. SVMs classify data into categories by finding the most accurate boundary line to divide the data [7]. 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 most accurate 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 a point of interest in regard to the line dividing the data.

4.2 Experiments

I will be experimenting with the models listed above to model 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 [14]. 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 patterns in the data unseen by an analyst, and may perform better than the other models I have selected. However, given enough data and the correct tuning, all of the tested models should theoretically approach the same accuracy levels.

4.2.2 Data

The data used from this project was obtained from Kaggle.com [13]. I used the "Synthetic Financial Datasets for Fraud Detection" which was generated by the PaySim mobile money simulator and is scaled down to a quarter of the original dataset [13]. The dataset includes attributes 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, and example of the information displayed in one transaction is displayed in Table 4.2.

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. As discussed in Chapter 3, banks and companies have strict policies they have to follow in regards to sharing user data. Therefore, there are no readily available datasets online. For reference, in the paper by Chan et al referenced above, their dataset had 30 fields per transaction [8]. 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. Lastly, there are essentially no repeated users in this dataset, eliminating the possibility of creating features based on user profiles.

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			

Table 4.2: Example Transaction

Transaction Number	Category
Hour	
Transaction 1	1
Type	
Transaction 1	Payment
Amount	
Transaction 1	$9839.64
Name Origin	
Transaction 1	Alice
Old Balance Origin	
Transaction 1	$170,136.00
New Balance Origin	
Transaction 1	$160,296.36
Name Destination	
Transaction 1	Bob
Old Balance Destination	
Transaction 1	$0.00
New Balance Destination	
Transaction 1	$ 0.0
Is Fraud	
Transaction 1	0

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.

Once my data is cleaned, I randomly split the data into training sets and test sets. I reserve about 15% of my dataset for testing. I do this to ensure my model does not learn the patterns of the tests, so I can test the accuracy of the model when exposed to new information. With my data prepared, I then run my series of tests with different classifiers and different architectures. The results are shown below.

As discussed in the experiment section, the majority of the methods employed are to test what alterations to my models creates large increases to accuracy and correct predictions. For each model, I used a subset of my data and altered the parameters and hyperparameters. Examples of this include changing the learning rate for the neural network model or choosing the max depth of the decision tree classifier. One of the main parameters I was interested in was the learner formula; which is the actual equation for which weights are calculated. This sort of guess and check method may seem somewhat misguided, but is one of the only ways to tune many machine learning algorithms. I focus on data manipulation as my computationally expensive tests.

Testing the neural network model is slightly more complicated, as you can have drastically different results based on the architecture. Neural networks are constructed of an input layer, output layer, and as many hidden layers in between as desired. Both the number of hidden layers as well as the number of nodes within each of the layers can be specified, growing or shrinking the model in complexity. In addition to the architecture, I will also be testing the epoch or max iterations parameter. An epoch is one forward propagation of data and one backward propagation. Each time data cycles through the network, the weights adjust and supposedly become more accurate. In addition to the other experiments, I will analyze the best amount of epochs for the fraud classification problem.

One of the elements of data manipulation I have tested is incorporating engineered features. The data I have is synthetic mobile payments as discussed in the data section. While this is a useful dataset for this project, it is somewhat limited in comparison to the information a credit card company would have on a normal brick and mortar transaction. The main limitation I attempt to expand upon is the lack of user profiling. There are over 6 million transactions in this dataset, with some of the same users repeating transactions. Each transaction includes information about which parties a transaction is between, the time as represented in hours and the amount.

The other two elements of data manipulation I test are fraud to non-fraud data proportions as well as normalization. While Gaussian probability distributions are effective at anomaly detection, much of the related work in this field have leveraged standard classifiers as their most successful models. In order for these models to work correctly, they must see many examples of all classes of data in order to make accurate predictions. Since fraud detection is a binary problem, it is logical to assume a classifier will do best in an environment where it is learning with 50% fraud and 50% legitimate transactions. However, this is not always attainable in the real world, so I test with multiple data proportions, with fraud making up between 50% of the transactions to less than

1% using the unaltered dataset. In addition to this, I also test whether normalizing the data increases the accuracy of the classifiers. Normalizing stretches the data so that it is equally spread along a distribution, while maintaining the original proportions of data. The normalization equation can be found in equation 4.1.

$$Normalized(transaction) = \frac{(transaction - min(transaction))}{(max(transaction) - min(transaction))} \tag{4.1}$$

Arguably the most important element of a machine learning experiment is deciding how to quantify a successful algorithm. The most common method of measuring whether a model is improving or worsening is accuracy. As discussed above, this is helpful in more balanced class situations, but is a little biased when working on anomaly detection. Because of this, I include precision, recall and an F1 score of each model in addition to the accuracy. The equations for precision, recall and F1 are shown in equations 4.2, 4.3, and 4.4, respectively.

$$Precision = \frac{TruePositives}{(TruePositives + FalsePositives)} \tag{4.2}$$

$$Recall = \frac{TruePositives}{(TruePositives + FalseNegatives)} \tag{4.3}$$

$$F1 = \frac{2(Precision * Recall)}{(Precision + Recall)} \tag{4.4}$$

As seen in the equations above, these measures are focused more so on the proportion of true positives to false positives. Precision describes a ratio of correctly identified fraud to all transactions identified as fraud. Recall is the proportion of correctly identified fraud to all identified as fraud. The F1 score is a measure incorporates both precision and recall, and is a helpful tool for combining these metrics. As discussed in 1, depending on the industry, false positives are often more expensive than false negatives.

This is because it is sometimes less expensive to pay out for small levels of fraud rather than lose a potential client because of suspected fraud. Therefore, my analysis of results comes from both accuracy and the F1 score.

4.2.4 Results

The most successful models and their metrics are shown in the following tables. For trials with non-normalized data, refer to Table 4.3. Normalized results can be found in Table 4.4.

Decision trees performed the best in almost all trials, as measured by the F1 score and accuracy. On the flip side, SVM's performed poorly. As a caveat, the neural network results are the best of the results found in experimenting with different architectures. In general, the neural networks biased towards the same results regardless of architecture or training epochs. In Table 4.4, I normalized the data, then performed the experiments. This is an approach used to eliminate bias due to magnitude of a feature and to put all the features on similar scales. In these experiments, however, normalizing the data lead to lower results for almost every model, as well as heavy biases in the system. The only model to improve was the SVM's, which became less biased towards predicting only fraud.

4.2.5 Discussion

It is possible that the data divisibility impacted the results of the paper. Classifiers perform best when the data is easily dividable between the data classes. When the data between classes are mixed, then it is more difficult to fit a classifier that is accurate while also not overfitting to the data. If the model overfits, then new examples are likely to be classified incorrectly.

Table 4.3: Results: Non-normalized Data

Method	Accuracy	Precision	Recall	F1 Score
Original data distribution				
Logistic Regression	0.999	0.3735	0.493	0.425
Neural Network	0.999	0	0	0
Decision Tree	0.9997	0.8974	0.903	0.899
SVM	0.004	1	.0013	0.003
Gaussian	0.952	0.532	0.015	0.029
50/50 data distribution				
Logistic Regression	0.910	0.002	0.902	0.004
Neural Network	0.0002	0.0002	1	0.0004
Decision Tree	0.989	0.017	0.995	0.033
SVM	0.002	0.0002	1	0.0004
Gaussian	0.491	0.0003	0.856	0.0006
66.6/33.3 data distribution				
Logistic Regression	0.910	0.854	0.885	0.869
Neural Network	0.999	0	0	0
Decision Tree	0.993	0.029	0.994	0.055
SVM	0.002	0.0002	1	0.0004
Gaussian	0.509	0.0003	0.857	0.0007
75/25 data distribution				
Logistic Regression	0.956	0.004	0.841	0.007
Neural Network	0.999	0	0	0
Decision Tree	0.995	0.036	0.997	0.070
SVM	0.002	0.0002	1	0.0004
Gaussian	0.505	0.0003	0.882	0.0007
80/20 data distribution				
Logistic Regression	0.971	0.005	0.809	0.011
Neural Network	0.999	0	0	0
Decision Tree	0.996	0.048	0.981	0.091
SVM	0.002	0.0002	1	0.0004
Gaussian	0.738	0.0005	0.617	0.001

Table 4.4: Results: Normalized Data

Method	Accuracy	Precision	Recall	F1 Score
Original data distribution				
Logistic Regression	0.999	0	0	0
Neural Network	0.999	0	0	0
Decision Tree	0.999	0	0	0
SVM	0.999	0	0	0
Gaussian	0.999	0	0	0
50/50 data distribution				
Logistic Regression	0.501	0	0	0
Neural Network	0.501	0	0	0
Decision Tree	0.499	0.499	1	0.666
SVM	0.534	0.536	0.499	0.516
Gaussian	0.499	0.499	1	0.666
66.6/33.3 data distribution				
Logistic Regression	0.670	0	0	0
Neural Network	0.670	0	0	0
Decision Tree	0.330	0.330	1	0.497
SVM	0.330	0.330	1	0.497
Gaussian	0.330	0.330	1	0.497
75/25				
Logistic Regression	0.750	0	0	0
Neural Network	0.750	0	0	0
Decision Tree	0.250	0.250	1	0.400
SVM	0.250	0.250	1	0.400
Gaussian	0.250	0.250	1	0.400
80/20				
Logistic Regression	0.801	0	0	0
Neural Network	0.801	0	0	0
Decision Tree	0.199	0.199	1	0.332
SVM	0.801	0	0	0
Gaussian	0.167	0.120	0.500	0.192

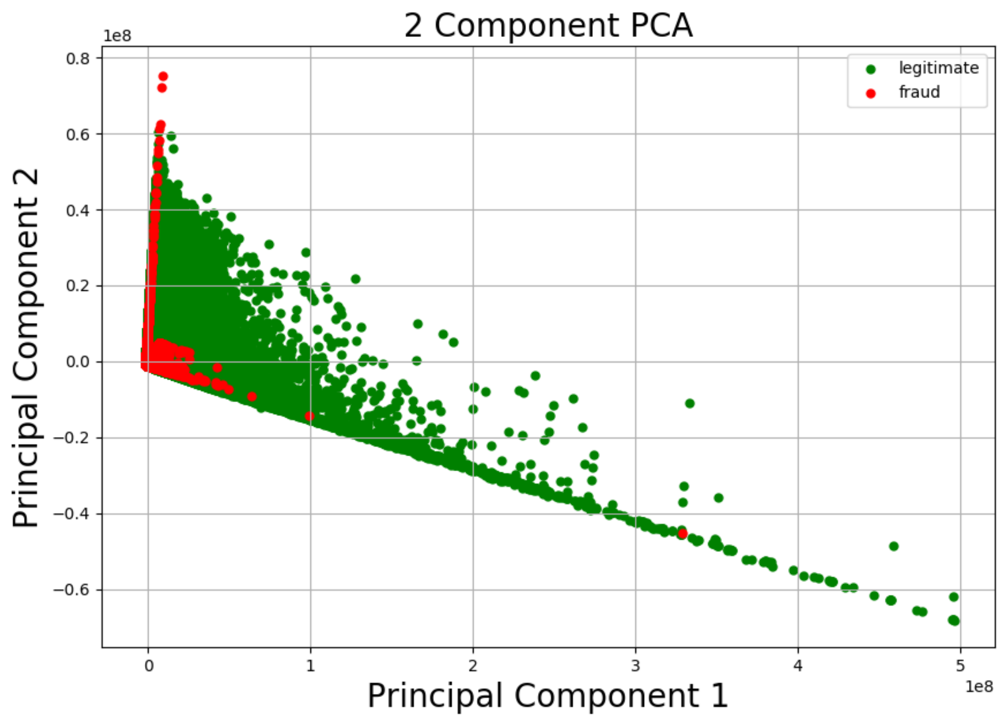

Figure 4.3: Principal Components Analysis of Data

In Figure 4.3, two principal components of the data is shown, with fraudulent transactions depicted in red. This figure shows that the classifiers may have had difficulty identifying fraud because of the lack of separability in this data. There seems to be a grouping of fraudulent transactions on the left section of this graph, but these are mixed in with legitimate transactions. This means that there is not a clear line a classifier could draw to divide fraudulent and legitimate transactions.

Based on the results in Table 4.3, decision trees perform the best in almost all the tests. Using the non-normalized data, training with a 1/3 proportion of fraudulent data has created the best results, with a 50/50 data proportion being a close second. In many of the interviews discussed in Chapter 3, practitioners discussed rule based systems as popular fraud detection systems. Rule based systems are very similar to decision trees, and can be implemented as decision trees. These results imply that these rule based systems are more effective than I originally hypothesized.

While most models performed well in these trials, the SVM's consistently performed poorly in all trials. I hypothesize this may be because of the poor separability of the data. SVM's attempt to divide the data with the largest margin of error possible. If there is no clear division, the SVM cannot make an accurate division. In addition, when changing the data proportions, the SVM may have found a division between fraud and legitimate transactions that was present in the training data, but not in the testing data, which represented the original dataset. This is further evidenced by the high recall of most of the SVM runs, showing that there were no false positives for any of the SVM runs with training data manipulation. SVM's in all of these cases were predicting almost all fraudulent, biasing heavily towards the minority class. When the data was normalized, the SVM's became less skewed, but most runs still biased towards predicting fraud.

While precision, recall and F1 scores can be used to determine the strength of a system, the true positive rate and false positive rate is also helpful in analyzing a system. Figure 4.4 shows the true positive rates as the fraud data proportion increases in the non-normalized run. Each of the models had increases in true positive rate as the data proportion increased, though by different magnitudes. The most drastic increase was in the neural network, where the model biased towards predicting fraud. As seem in Figure 4.5, the false positive rates also tend to increase as the true positive rates increase. To return to related work, if a company has low overhead, increased false positives are not as expensive as if a company has very high overhead [8]. Therefore, the increased sensitivity to fraud is worthwhile. In these results, the decision tree has both high true positive rates and low false positive rates, reinforcing the success of the model.

Through these experiments, I have found decision trees to be the most successful in classifying fraudulent transactions in this setting. My initial hypothesis was incorrect; neural networks did not perform well in this setting, and easily biased towards predicting

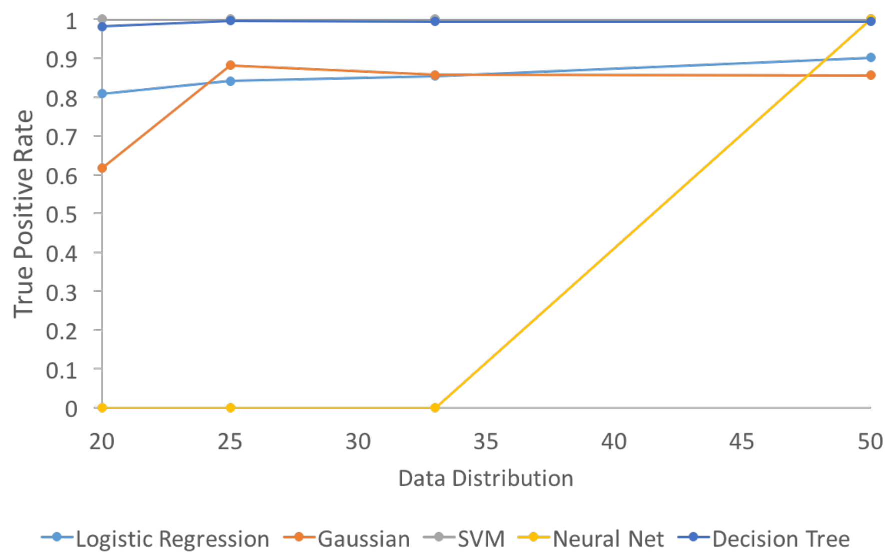

Figure 4.4: True Positive Rates by Data Proportions

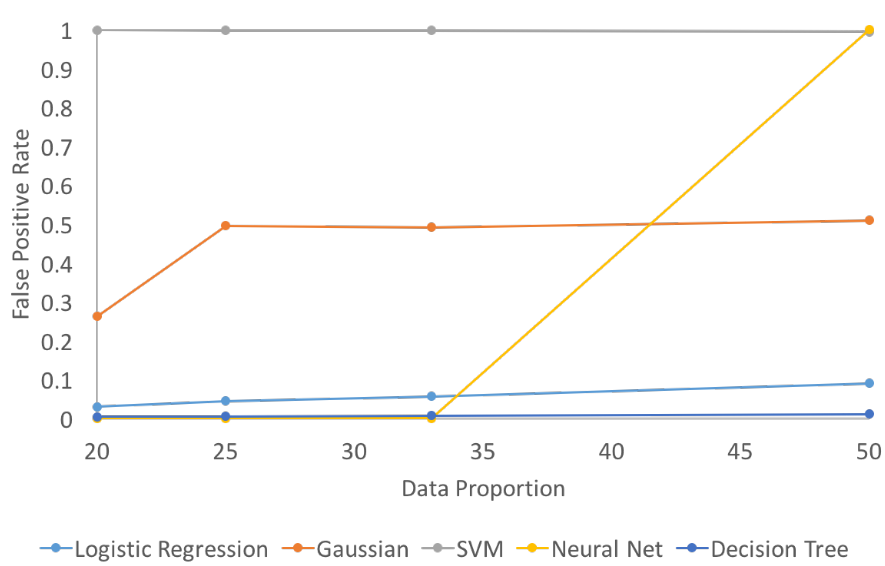

Figure 4.5: False Positive Rates by Data Proportions

one class or another, regardless of the architecture. A lesson from this is to always choose a classifier based on the problem at hand, not by the supposed complexity of a model. In addition, I found false positive and true positive rates to be helpful in analyzing the success of the model.

CHAPTER 5

FURTHER APPLICATIONS

The challenge in credit card fraud detection comes from the lack of examples of fraudulent behavior; this is because fraud is an anomaly. The work done in this thesis is framed by previous anomaly detection research as well as credit card fraud specific research. Because of this, the implementations of credit card fraud detection systems can be applied to other fields of anomaly detection with slight alterations. In this chapter, I describe other fields in which anomaly detection techniques are applied.

5.1 Anomaly-Based Intrusion Detection

Anomaly detection techniques can be used to find instances of intrusion attempts into a system, and have been leveraged in the network space. Intrusion detection systems monitor a network or system for malicious activity, and often use machine learning to detect these malicious traffic from good or policy following traffic. Wang and Stolfo analyze payloads in order to detect anomalies [22]. This system used unsupervised learning to determine the distribution of byte frequency of a single application payload between a single host and port. This distribution and standard deviation is learned during a training phase. They then use Mahalanobis distance to calculate simiarlity of new data against the profiles computed during the training period. If a new input exceeds the threshold they set, then an anomaly is detected. While testing on 1999 DARPA IDS data, the system achieves an almost 100% accuracy with a 0.1% false positive rate [22]. In this case, anomaly detection can be leveraged using unsupervised learning to define a user profile and compare against that.

5.1.1 Biometric Anomaly Detection

In a study by Ahmed and Traore, biometrics are used to detect system intrusion at the user level [4]. This system uses behavioral biometrics, which include patterns of a person's physical movements, to identify a user. This is different than physiological biometrics, in which a user is identified by a feature such as a fingerprint or iris. In the enrollment phase of this model, the client machine runs software that monitors a user's mouse movement and keystroke data. This information is then feed to a server, which computes a profile that is used for behavior comparison in the verification stage. Using the mouse and keystroke data, Ahmed and Traore create dynamic signatures using neural networks and statistics, which capture features that define a user uniquely. During the verification stage, the sum of the absolute difference between the expected user's profile signature and the current user, and a decision is made based on a predefined threshold.

5.2 Hardware Health Monitoring

One common use for anomaly detection is in monitoring health of a system, including the health of hardware. In the Aerospace field, anomaly detection has been use to monitor the health of liquid-fueled rocket propulsion testbeds. Schwabacher et al used four different algorithms to test on historical data from the Space Shuttle Main Engine and an experimental rocket engine at NASA Stennis Space Center [15]. The four approaches used are as follows. The first approach defines an anomaly as a point that is far from its closest neighbor, using a nearest neighbor to calculate each point's nearest neighbor. In the next algorithm called the Inductive Monitoring System, system clusters are learned from training data, then distance to the nearest cluster is used to determine to measure its anomalousness. The third system they use is similar to a decision tree classifier, as it learns rules from the training data and marks a point as anomalous if they violate these rules. The last algorithm the authors implement is an SVM. These methods all correspond to those used in the credit card fraud detection, and have similar benefits of reducing the number of human experts needed to analyze spacecraft sensor data.

5.3 Bioinformatics

Bioinformatics is a field of study concerned with analyzing complex biological data, in particular genetic codes. There is a lot of existing research into detecting anomalies in discrete and symbolic sequences, as surveyed by Chandola et al [9]. There are a couple different methods that are commonly used for this bioinformatic anomaly detection problem. The first way is unsupervised, where anomalous sequences are detected from an unlabeled database of sequences. The other version is called semi-supervised. This involves training a model on a database that is assumed to be all normal sequences, then testing to see how different the example sequence is from the training examples. This model is semi-supervised because all examples are assumed to be normal, but no labels are used. Here, a popular method is also to use k-nearest neighbor and clustering, similar to the methods discussed above. Anomaly detection in bioinformatics is yet another field where similar anomaly detection methods can be leveraged.

5.4 Cloud Computing

Cloud computing systems have become more complex and popular in the recent years. Because of this increased use, dependability assurance has become a very important issue in system design and management. Song Fu creates a framework to automatically detect anomalies in cloud computing [16]. In this framework, principal component analysis is applied to a number of performance related metrics or features. This allows for a reduced metric dimension, but keeps the variance in the health related data. This reduced space is then fed through a semi-supervised decision tree classifier, which further reduces the dimensionality and identifies the anomalies [16]. This is a very similar method to the decision tree classifier built in Chapter 4. Even though cloud computing and credit card fraud are very different topics, almost identical frameworks can be used to solve both problems.

CHAPTER 6

CONCLUSIONS

Credit card fraud is growing with the advent of new payment methods and online transactions. While not everyone falls victim to these attacks, there are negative impacts to all from fraudulent transactions. These transactions directly impact users through the redistribution of costs from company to user. Increased fraud can also decrease trust in the overall financial system, which can lead to chaos. There is also a negative societal impact, as money gained from fraudulent transactions are used for illicit purposes. Because of these reasons, it is worth building effective fraud detection systems.

Fraud is particularly hard to detect because it is an anomaly, and only makes up about 1-3% of transactions. There are not many examples of the fraudulent class in order to train classifiers with. There are several ways to deal with this, such as fitting probability distributions to the data or changing the proportion of the training dataset. As in related works and the results in Chapter 4, this method helps build classifiers that are more sensitive to fraudulent transactions, but may increase false positive rates simultaneously. With low overhead costs, this trade off is worthwhile, and will decrease overall costs from fraud. If overhead is high, costs may not decrease and could possibly increase due to higher sensitivity to fraud.

After performing a series of experiments using synthetic mobile money data, my results indicate that decision trees perform the best for fraud detection. The decision tree model performed the best in all trials in the non-normalized data experiments, and tied with the Gaussian model in the normalized experiments. This implies that rule based systems, which can be implemented as decision trees, are probably better suited for the fraud detection problem than I hypothesized. In my second set of experiments, I found normalizing the data mostly decreased the results from almost all models except the SVM. Normalizing seemed to cause heavy bias towards predicting fraud, leading to high recall but low accuracy. In other cases, normalizing the data led to predicting no fraud, with zero true positives detected in the data.

6.1 Future Work

In this thesis, I was somewhat limited by the data I had access to. I would have liked to have more realistic credit card transaction data that included repeated users as well as location information. As future work, I would have liked to try some feature engineering based on historical information about a user. For example, I would have created a feature based on whether or not a transaction was above a user's average transaction size. This would have caught when transactions were abnormally high for a user, indicating fraud.

In addition, I would have liked to create features based on time between transactions of a user. If two transactions are two close together, it is unlikely that a user made both of these transactions. This paired with location data could indicate whether a series of transactions was even possible to be made by the same user. Lastly, I would create a feature indicating whether a user had ever transacted with the destination user before. It might be an indication of fraud if someone is transacting with another user or store that is outside of their normal behavior. Through these features, the classifiers would have had information about how far the transaction is straying from past behaviors of that user in addition to how different the transaction is to other legitimate transactions.

Based on the results from this experiment, and extension of my fraud detection system could involve combining a rule based system with a more sensitive classifier. Since decision trees were the best suited for this problem, I would have liked to try building out a rule based system based on some of the rules discussed in previous literature. In this architecture, all transactions would filter through the rule based system first, potentially eliminating transactions that are certain to be legitimate and certain to be fraud. This would leave the uncertain cases, which a classifier would then train. This architecture is similar to Panigrahi et al, which combines the results from the different elements using Dempster-Shafer Theory [12].

With more realistic credit card data, I also could have used some of the network analysis approaches to feature engineering discussed in the related work. There has been much research into feature engineering using deep feature analysis or even social network analysis. These approaches mainly look at patterns in the behavior of users in relation to their own past transactions and other users. Social network analysis looks at how a user is acting in comparison to other members in their community or immediate circle of acquaintances [5]. Future work on this topic would involve implementing some of these systems, and incorporating them into the experiments ran in this thesis.

BIBLIOGRAPHY

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

[2] Most popular payment methods in the U.S. 2017 — Statistic.

[3] U.S. payment card fraud losses by type 2018 — Statistic.

[4] Ahmed Awad E. Ahmed and Issa Traore. Anomaly Intrusion Detection based on Biometrics. Technical report, IEEE, 2005.

[5] Bart Baesens, Veronique Van Vlasselaer, Wouter Verbeke, and Veronique van Vlasselaer. *Fraud analytics using descriptive, predictive, and social network techniques : a guide to data science for fraud detection*.

[6] Stefano Boccaletti. Complex Network Theory: introduction and applications.

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

[8] 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.

[9] V. Chandola, A. Banerjee, and V. Kumar. Anomaly Detection for Discrete Sequences: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):823–839, may 2012.

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

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

[12] Suvasini Panigrahi, Amlan Kundu, Shamik Sural, and A.K. Majumdar. Credit card fraud detection: A fusion approach using DempsterShafer theory and Bayesian learning. *Information Fusion*, 10(4):354–363, oct 2009.

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

[14] 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.

[15] Mark Schwabacher, Nikunj Oza, and Bryan Matthews. Unsupervised Anomaly Detection for Liquid-Fueled Rocket Propulsion Health Monitoring. *Journal of Aerospace Computing, Information, and Communication*, 6(7):464–482, jul 2009.

[16] Song Fu. Performance Metric Selection for Autonomic Anomaly Detection on Cloud Computing Systems. In *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, pages 1–5. IEEE, dec 2011.

[17] 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.

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

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

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

[21] Vishal Vatsa, Shamik Sural, and A. K. Majumdar. A Rule-Based and Game-Theoretic Approach to Online Credit Card Fraud Detection. *International Journal of Information Security and Privacy*, 1(3):26–46, jul 2007.

[22] Ke Wang and Salvatore J. Stolfo. Anomalous Payload-Based Network Intrusion Detection. pages 203–222. Springer, Berlin, Heidelberg, 2004.

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

[24] 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.

[25] T. Jeffrey Wilks and Mark F. Zimbelman. Using Game Theory and Strategic Reasoning Concepts to Prevent and Detect Fraud. *Accounting Horizons*, 18(3):173–184, sep 2004.

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

[27] 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.