Using Anomaly Detection to Identify Credit Card Fraud

CS 5593 – Data Mining

Fall 2017

Joe Trivisonno

Trivisonno.10@gmail.com

https://youtu.be/cttp4rav_vE

Abstract

Due to the increasing use of credit cards, credit card fraud is on the rise. To limit the damage caused by credit card fraud, it is important to catch it as it happens. To do this, an application that evaluates a transaction and identifies it as legitimate or fraud is needed. Much research has been done into applications that use supervised methods such as Neural Networks, Support Vector Machines, and Decision Trees, with Neural Networks being the most popular choice of algorithm. However, these methods need a labeled set of data and need to be continually retrained as new data is gathered. Unsupervised methods do not have these drawbacks, but little research has been done on their usefulness for detecting credit card fraud. Unsupervised methods have however been researched for use in intrusion detection to great success. A distance based anomaly detection algorithm using principal component data was able to identify 99% of intrusions with only a 1% false positive rate. This report examines that algorithm and modifies it for use in credit card fraud detection. In addition to this algorithm, an application was developed to allow the user to select which columns to run the algorithm on and to view the results. The algorithm was less successful at identifying fraudulent credit card purchases, identifying just 26.42% of fraudulent purchases with a high false positive rate. Nevertheless, it is still useful in fighting against credit card fraud if implemented in real time. The high false positive rate is not concerning as the customer can simply confirm their purchase is legitimate via phone or text message and carry on.

Table of Contents

1. Introduction

Due to a massive increase in credit card transactions, credit card fraud is on the rise [Chaudhary et al, 2012]. Due to the amount of damage credit card fraud can do, it is important to catch it early, before too many purchases can be made. To do this, an application that is able to correctly detect fraudulent credit card purchases from legitimate ones is needed. Neural network, support vector machine (SVM) and decision tree classifiers are some of the most commonly used supervised methods for detecting fraudulent transactions [Zareapoor et al, 2012]. One drawback to these supervised learning methods is that they require a labeled dataset to train a model on. These datasets are often extremely unbalanced, with very few transactions being labeled as fraud [Bolton and Hand, 2001]. This makes it very hard to train a supervised model. In addition, supervised models must be continually retrained as new data is collected, making it difficult to identify fraudulent transactions in real time. In contrast, unsupervised methods such as anomaly detection, do not require a set of known legitimate and fraudulent transactions to work. They simply look for transactions that are unusual, or outliers, from other transactions in the dataset [Bolton and Hand, 2001]. For these reasons, I will create an unsupervised anomaly detection algorithm to identify credit card fraud. While unsupervised anomaly detection still requires a large amount of data initially to be able to reliably identify anomalous or fraudulent purchases, it can easily accept new data without having to retrain a model, allowing it to be used in real time to detect fraudulent transactions.

2. Related Work

There has been a significant amount of research done on supervised algorithms to detect credit card fraud, but very little done about unsupervised algorithms. Several common supervised algorithms used are Neural Networks, Support Vector Machines (SVM), and Decision Tree

classifiers [Zareapoor et al, 2012]. Of these, Neural Networks (NN) are the most popular method for fraud detection [Zareapoor et al, 2012].

One reason is that NN perform very well with large datasets with a lot of variables, which is the case when attempting to identify credit card fraud [Raghavendra 2011]. They are also very good at discovering irregularities within data, especially when the relations between variables are only vaguely understood [Burger 2010].

NN are made up of many interconnected nodes that are organized in layers. Patterns are entered into the network by the "input layer", which sends the data to additional "hidden layers" through a set of weighted connections. A visual display of these layers can be seen in Figure 1. The weight of these connections is initially random, and are modified as the NN "learns." This is done via what is known as the delta rule. With the delta rule, the NN evaluates the accuracy of the initial random weights and adjusts the connection weights based on the error. Since the error space cannot be known, a large number of individual runs are used to determine the best solution. Once drawback to NN is that it is very difficult to see what is going on in the hidden layers of the network. It is what is known as a "black box" method where all that a user can see is what is input and what the model outputs [Burger 2010]. Another issue is that performance of a NN is highly dependent on parameters that have to be set prior to any training, and there are no general rules for how to set these parameters [Raghavendra 2011]. Because the use of NN and other supervised methods of credit card fraud detection are well researched, this report will focus on unsupervised methods of detection.

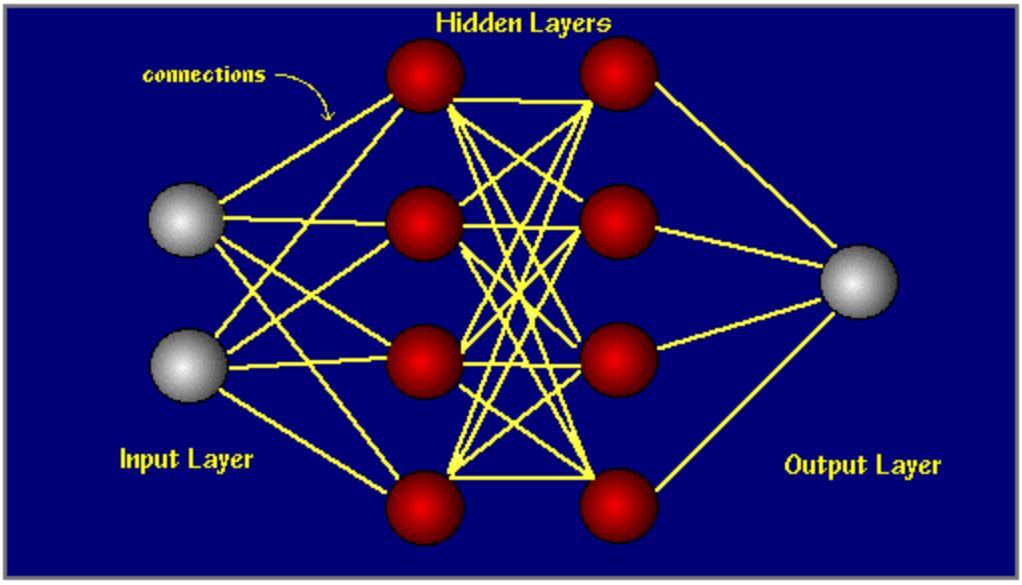

Figure 1: Structure of a Neural Network (Courtesy: http://pages.cs.wisc.edu/~bolo/shipyard/neural/local.html)

Unsupervised methods of credit card fraud detection have been significantly less researched. Anomaly detection techniques have been used in intrusion detection algorithms, however. Like in fraud detection, "intrusions" are rare compared to legitimate connections, which means any data that is collected will be highly unbalanced. Due to the similarities between intrusion detection and fraud detection, research done in this area could prove useful in creating an unsupervised fraud detection algorithm.

One example of anomaly detection being used for intrusion detection can be found in the article titled "A Novel Anomaly Detection Scheme Based on Principal Component Classifier" from the Defense Technical Information Center [Shyu et al, 2003]. In this article, Shyu et. al. created an anomaly detection algorithm to detect outliers in principle component data. This method was extremely effective at intrusion detection, identifying 99% of intrusions with a 1% false alarm rate. While the principle components of intrusion and fraud detection data will obviously look

different, the success of this sort of algorithm at intrusion detection warrants a closer look at utilizing it for fraud detection. For this reason, I will be using this intrusion detection algorithm as the baseline for my unsupervised fraud detection algorithm.

3. Proposed Work and Results

3.1 Application Description

For this project, I will be creating an application that allows the user to import a dataset and perform an unsupervised anomaly detection algorithm to identify fraudulent transactions. Section 3 describes the details of the algorithm and application as well as the format of the data.

3.2 Dataset

The data set that I used was uploaded to Kaggle.com by the user Andrea and can be found at https://www.kaggle.com/dalpozz/creditcardfraud [Pozzolo et al, 2015]. The dataset contains 284,807 credit card transactions made by European cardholders over two days in September 2013. Of these, 492 of the transactions, or 0.172%, were fraudulent. Due to confidentiality concerns, the data contains numerical input variables that are the result of Principle Component Analysis (PCA) transformation. The original features were not provided. The data set contains 31 features. Features V1 through V28 are the principle components obtained from the PCA transformation. The dataset also contains 3 features that were not transformed with PCA. The feature ‘Time’ contains the number of seconds elapsed between the transaction and the first transaction in the dataset. The feature ‘Amount’ in the amount of the transaction. The feature ‘Class’ identifies whether a transaction is fraudulent or not. It takes a value of 1 if it is fraudulent, and 0 if it is not. Table 1 displays a description of each column in the dataset. The dataset is provided as a .csv and is 69.7 MB compressed and 147.3 MB uncompressed. A sample

of the dataset can be found at the link provided. The large amount of data and very low proportion of fraudulent charges in the dataset will mimic the small percentage of all charges that are fraudulent, thus properly emulating the difficulty of identifying fraudulent charges.

Table 1: Description of dataset

Column Name	Description
Time	Time (s) since 1st transaction in dataset
V1	1st Principle Component
V2	2nd Principle Component
V3	3rd Principle Component
V4	4th Principle Component
V5	5th Principle Component
V6	6th Principle Component
V7	7th Principle Component
V8	8th Principle Component
V9	9th Principle Component
V10	10th Principle Component
V11	11th Principle Component
V12	12th Principle Component
V13	13th Principle Component
V14	14th Principle Component
V15	15th Principle Component
V16	16th Principle Component
V17	17th Principle Component
V18	18th Principle Component
V19	19th Principle Component
V20	20th Principle Component
V21	21st Principle Component
V22	22nd Principle Component
V23	23rd Principle Component
V24	24th Principle Component
V25	25th Principle Component
V26	26th Principle Component
V27	27th Principle Component
V28	28th Principle Component
Amount	Transaction Amount
Class	Fraud (1) or Legitimate (0)

3.3 Algorithm Description

For this project, I decided to use an unsupervised anomaly detection algorithm to identify fraudulent purchases. Anomaly detection based on principal component (PC) data has been shown to be very effective at intrusion detection [Shyu et. al. 2003]. Since both intrusion detection and fraud detection are similar in that the proportion of positive to negative instances of intrusion or fraud is very unbalanced, it could also be very powerful for fraud detection. Furthermore, the only data available to me contains PC transformations, so using an algorithm that is known to work well with them is ideal. This method creates two functions of principal component scores, one from the major components $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i}$ and one from the minor components $\sum_{i=p-r+1}^{p} \frac{y_i^2}{\lambda_i}$. In the previous equations, *i* represents the *i*th principal component, y_i represents the value of the *i*th principal component, and λ_i represents the eigenvalue of the *i*th principal component in the correlation matrix. The values *q* and *r* determine which principal components will be used for each function, while *p* is the number of principal components. The value for *q* was set so that the components used in the major function accounted for approximately 50% of the variance for the dataset. The value for *r* was set to contain the components whose eigenvalues was less than 0.2. In this algorithm, a transaction is identified as anomalous if either the major or minor component function exceeds a predetermined threshold. These anomalous transactions are classified as outliers. The threshold for each of these functions will be discussed in section 3.6. [Shyu et. al. 2003]

To determine which columns will be used for each function, the covariance matrix of the principle components was calculated and the eigenvalues were determined. The sum of the eigenvalues was found to be 30.73185. The sum of the eigenvalues for the first 6 principal

components is 14.547, which is a close to 50% of the total sum as possible. Thus the first six principal components were identified to be used for the major components function. Principle components 27 and 28 were the only two with eigenvalues less than 0.2, so they were identified to be used for the minor components function. Thus our two functions are $\sum_{i=1}^{6} \frac{y_i^2}{\lambda_i}$ and $\sum_{i=27}^{28} \frac{y_i^2}{\lambda_i}$.

3.4 Data Preprocessing Activities

The application will assume that the data imported has already had PCA transformations performed on the data, since due to the sensitivity of credit card transactions that is what is available for me to use. I would be unable to test the application with data sets in any other format due to non-availability. Within the application, the data preprocessing will consist of identifying which columns should be used to calculate the major and minor component scores. This is done by checking the box next to the desired columns for each function as will be explained in section 3.5.

3.5 Implementation Details

The application GUI was implemented using R Shiny. R shiny is a library in R that allows the user to create an interactive GUI using R code. Since this was my first time attempting to make a GUI of any kind, I wanted to use a language I was familiar with when creating the application and R shiny provided that. As this was my first time creating a user interface, I wanted to keep it as simple as possible, thus the number of options were kept limited. Upon startup, the user will select browse and upload their data set. All data sets must be formatted as described in section 3.2. The application will not accept any other format. The dataset that the application is using is shown in the main section of the screen, as can be seen in Figure 2.

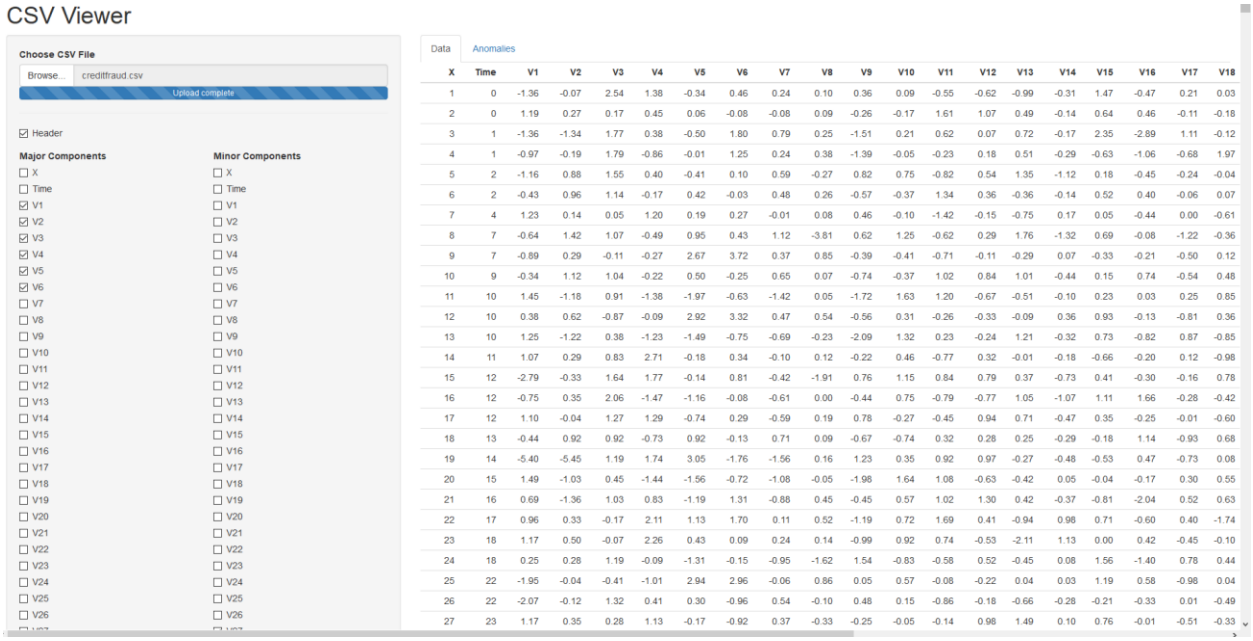

Figure 2: Screenshot of application data tab.

On the left side of the screen, the user will select which columns are considered major component, and which columns will be used for the minor components portion of the algorithm. Once these are selected, the application will run the anomaly detection by pressing the “Run” button on the bottom of the left side of the screen. Figure 3 shows a screenshot of this component selection panel.

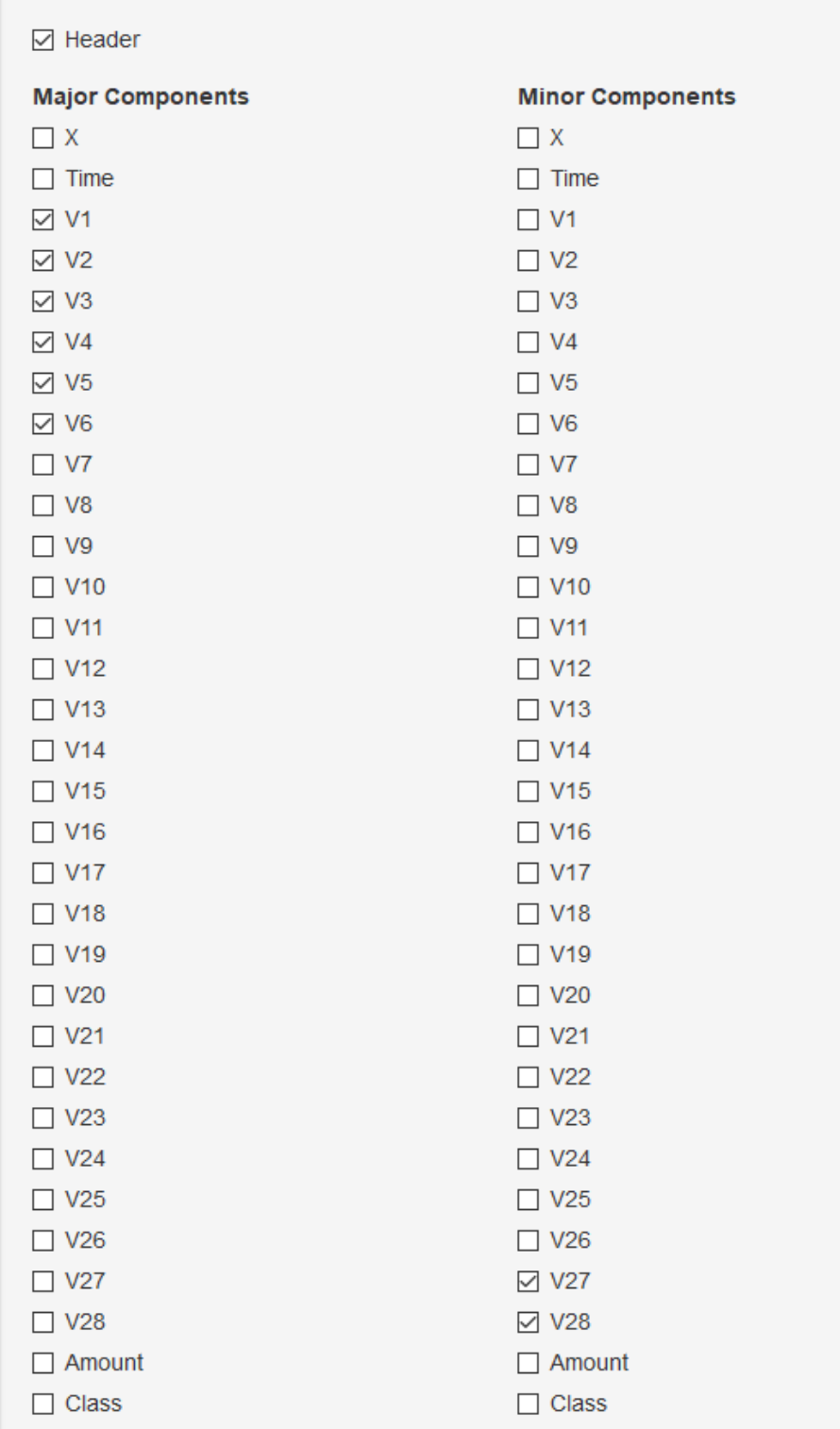

Figure 3: Components selection panel

In a separate tab, the anomalies found by with algorithm will be displayed after the user presses “Run” and the algorithm is executed. It should be noted that the algorithm will take some time to execute, so the tab will be blank until that time. The anomalies tab can be seen in Figure 4.

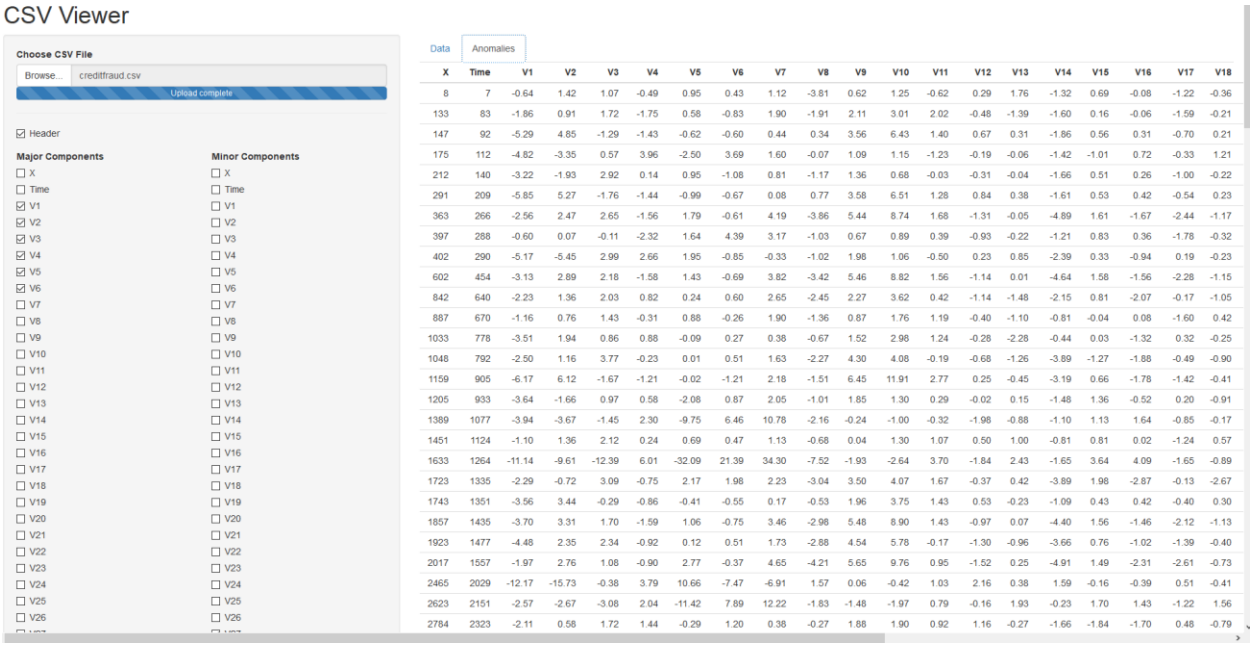

Figure 4: Screenshot of application Anomalies tab.

3.6 Performance

Due to imbalanced nature of the data set, traditional performance measures are not good indicators of performance. Simply identifying every transaction as legitimate would result in a 99.83% accuracy, yet would be completely ineffective. Furthermore, due to the nature of the application, a large amount of false positives is not much of a concern. While ideally, the number of false positives would be low, identifying fraudulent purchases is much more important than not falsely identifying legitimate purchases as fraudulent. Since the idea is for this algorithm to be used mainly for real time monitoring, false positives merely need to be confirmed by the customer via phone or text message as legitimate and otherwise there are very limited

consequences for false positives. This factor is important because through extensive testing of the algorithm, it was determined that unlike for intrusion detection, it cannot be used to identify cases of credit card fraud without a significant amount of false positives. So the challenge became identifying as many fraudulent purchases as possible while keeping the number of false positives to an acceptable rate. With this in mind, I decided have the algorithm identify approximately 1% of all transactions as fraudulent. To do this, it was determined the threshold value for the major components function is 100, and the threshold value for the minor components function is 40. The performance was then determined by the sensitivity, which can be defined as $\frac{True\ Positives}{True\ Positives+Fals\ Negatives}$. This algorithm correctly identified 130 of the 492 fraudulent transactions, for a specificity of 0.2642. These 130 correctly identified fraudulent transactions were worth $10,340 out of $60,127.97 in total fraudulent purchases. Increasing the number of transactions identified as fraudulent had little impact on the number of correctly identified fraudulent transactions unless the false positive rate was increased to unacceptable levels. Modifying the columns used for each function also did not result in improved performance.

4. Conclusions and Future Work

While this unsupervised algorithm only identifies 26.42% of fraudulent transactions and identifies many transactions that are legitimate as fraudulent, it is still useful as a tool to prevent credit card fraud. If it is implemented in a real time fashion, the high amount of false positives are not a major concern as the customer can simply confirm the transaction is legitimate and carry on with their day. And while it won't stop all fraud, it can help to limit the impact caused by credit card fraud. If use of this algorithm for real time detection is combined with slower,

more accurate supervised methods to detect fraud after it has occurred, the impact of credit card fraud can be even further reduced.

There are opportunities to improve both the algorithms and the user interface for this application. With the current GUI, the application runs very slow, so there are opportunities to optimize the interface so that either it runs faster, or the application lets the user know when the algorithms are running. There are also opportunities to incorporate additional algorithms into the application. I would like to add the supervised methods discussed in this report to the application in the future. An application that can utilize multiple methods of fraud detection would be very useful, since each method will likely miss instances of fraud that other methods might catch.

5. References

References

[1] Bolton, Richard J. and David J. Hand. September 2001. “Unsupervised Profiling Methods for Fraud Detection”. *Credit Scoring and Credit Control VII*. Accessed 16 September 2017, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.5743&rep=rep1&type=pdf>.

[2] Burger, Josef. “A Basic Introduction to Neural Networks”. 2010. Accessed 10 October 2017, < http://pages.cs.wisc.edu/~bolo/shipyard/neural/local.html>.

[3] Chaudhary, Khyati, Jyoti Yadav and Bhawna Mallick. May 2012. “A Review of Fraud Detection Techniques: Credit Card.” *International Journal of Computer Applications*, Volume 45-No. 1. Pages 39-44. Accessed 16 September 2017, <http://research.ijcaonline.org/volume45/number1/pxc3878991.pdf>.

[4] Raghavendra, Patidar, and Lokesh Sharma. June 2011. “Credit Card Fraud Detection Using Neural Network”. *International Journal of Soft Computing and Engineering*, Volume 1, Issue NCAI2011, Pages 32-38. Accessed 12 October 2017, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.301.8231&rep=rep1&type=pdf>.

[5] Shyu, Mei-Ling, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. 2003. “A Novel Anomaly Detection Scheme Based on Principal Component Classifier.” *Defense Technical Information Center*. Accessed 16 October 2017, <http://www.dtic.mil/docs/citations/ADA465712>.

[6] Zareapoor, Masoumeh, Seeja. K. R. and M. Afshar Alam. August 2012. “Analysis of Credit Card Fraud Detection Techniques: Based on Certain Design Criteria”. *International Journal of Computer Applications*, Volume 52-No. 3. Pages 35-42. Accessed 16 September 2017, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.259.16&rep=rep1&type=pdf>.

Dataset courtesy of: Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson and Gianluca Bontempi. Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, 2015.