Hannah Laper

Adam Wachsman

28 November 2020

DATA 602

Dr. Diana

## Systematic Evaluation of Supervised Learning Algorithms in Credit Fraud Detection and the Impact of PCA-1 Augmentation

**Introduction**

Despite various advances in security measures, credit card fraud remains a ubiquitous problem in our society. In 2015, there were 61.7 cases of fraud in the United States that totaled $8.34 billion dollars, and the total number of cases nearly doubled since a previous measurement taken in 2012 (The Federal Reserve, 2018). Credit card fraud can incur severe financial damage to businesses and individual cardholders alike and Markus Bergthaler, who educates businesses on fraud statistics, states “Recent figures suggest that over 80 percent of credit cards currently in people’s wallets have already been compromised” (Hadar, 2019). The threat of fraud drives businesses to spend on security systems and banks to spend on notification, card replacement, and investigation. All of these costs can pull financial resources away from other services. Larger firms may be able to bear the cost of security and the cost of fraudulent sales, but smaller businesses struggle to afford the security and costs associated with fraudulent sales (Hadar, 2019). While companies often absorb the costs of fraud when restitution cannot be retrieved, individual consumers must expend considerable effort to recover from fraud as well. The consumer must take time to contact the company and/or their bank to report fraud, and then take further action to recover funds. The most diligent fraud victims will also implement further security measures and change the businesses they patronize. The high economic impact and the heterogeneous nature of credit card fraud motivated us to examine how different machine learning algorithms perform when tasked with separating fraudulent and legitimate transactions.

**Literature Review**

Fraud detection and prevention strategies have evolved significantly over the decades according to shifts in technology and the nature of commerce. In this section, we will briefly review relevant literature in order to gain a better understanding of how to develop our models.

Detecting fraudulent and anomalous behavior has been a popular area of machine learning research. For example, in addition to the credit card fraud domain, Kou and Huang (2004) also cite the use of neural networks for detecting computer intrusions and fraudulent use of telecommunications systems. The general justification for this application is the expectation that, given enough data, these algorithms can find patterns of distinction between “baseline” patterns of legitimate behavior and patterns of fraudulent behavior. Crucially, machine learning also creates the potential for models that can dynamically adapt to changes in patterns over time (Dal Pozzolo et. al. 2014) over simple rule-based models described by Bhatla et. al. (2003). However, the proper utilization of credit card fraud datasets and model implementation both present significant challenges.

On the challenge of dataset utilization, class imbalance presents the most immediate difficulty; fraud makes up a very small proportion of total transactions and model training must be adapted accordingly (Bolton & Hand 2002; Dal Pozzolo et. al. 2014). For researchers, data scarcity is another major problem; any disclosed dataset naturally must be stripped of information that could identify cardholders or facilitate further fraud. Yet the loss of information through anonymization strictly limits the potential effectiveness of models built around those datasets. Choosing granularity is a further dilemma; should a fraud detection model implement based on global transaction data, card-specific transaction history, global account descriptions, or other levels of granularity? Global transaction and account data in particular might be suited to the detection of fraud involving new accounts, typically created by means of identity theft, a fraud subtype which nearly doubled in size as of 2019 (Federal Trade Commission 2020). Finally, researchers must decide which features, if any, to engineer out of the data.

Numerous researchers have studied the problem of model choice and implementation. These studies typically involve supervised models in some form, including Logistic Regression (Boucher 2020; Niu et. al. 2019), Support Vector Machines (Boucher 2020; Niu et. al. 2019), Random Forest (Carcillo et. al. 2019; Niu et. al. 2019; Dal Pozzolo 2017), and Neural Networks (Lebichot et. al. 2019). According to Dal Pozzolo’s group (2017) and Carcillo’s group (2019), several studies have found Random Forest models to achieve the highest performance, although Niu et. al.'s study (2019) found the Xgboost classifier to have slightly better performance. Some studies have also utilized unsupervised learning models, either standalone (Niu et. al 2019; Bolton & Hand 2001) or as part of an ensemble (Micenkova et. al. 2014; Veeramachaneni et. al. 2016; Carcillo et. al. 2019). Niu’s group (2019) found that unsupervised models used with a global transaction dataset performed worse overall compared to the supervised models in terms of Area Under Receiver Operating Characteristic (AUC-ROC), possibly because a typical transaction on one account may appear extreme to another. Micenkova et. al.’s (2014) work suggests that the outlier scores generated by unsupervised clustering models can augment the existing set of features to improve fraud detection performance over standalone models. Carcillo et. al.(2019) expanded on this approach by testing outlier scores generated by different methods and data granularities, discovering that global outlier scores were not informative, but account-level and cluster-level outlier scores successfully augmented the fraud detection performance of a Random Forest classifier. Using a Logistic Regression model with L1 regularization, they determined that the Isolation Forest model was the most effective for account-level granularity, and that the reconstruction error using the first PCA component was the most effective for cluster-level granularity.

**Dataset:**

For our project, we utilized Credit Card Fraud Detection Dataset (2018), provided for public use on Kaggle by Université Libre de Bruxelles’ Machine Learning Group (see references). According to the publishers, the dataset contains a timeseries of credit card transactions by European cardholders from September 2013. In order to comply with aforementioned security concerns, all features have been transformed into Principal Components and all contextual column labels removed with the exception of ‘Time’ and ‘Amount’. The order of the components may also have been rearranged. ‘Time’ is measured in seconds elapsed from an unknown point in time and, while units were not specified, ‘Amount’ most likely corresponds to amounts in Euros. The ‘Class’ column contains binary target labels, where values of 1 correspond to fraudulent transactions. The dataset spans 284,806 rows with 30 feature columns, spanning approximately 48 hours. Of those rows, 492 represent fraudulent transactions, or about 0.17% of the total observations. Framed in terms of the reviewed literature, this dataset is highly class-imbalanced and is only available to global, transaction-level analysis due to the inability to group transactions by individual accounts

**Design and Research Questions**

Using this dataset, we constructed eight fraud detection models trained by four supervised learning algorithms: Logistic Regression, Support Vector Machines (SVM), Random Forests, and Artificial Neural Networks. We also utilize a single-component Principle Component Analysis (PCA) model trained on the entire dataset to generate the PCA reconstruction error (PCA-1) as an outlier score for each observation in the dataset based on Carcillo et. al.’s work (2019). Four of the models were trained on the original dataset while the other four were trained on a version of the dataset which was augmented by the PCA-1 for each observation as a feature. Our experiment sought to answer three questions:

1. Which of the supervised models best maximizes accuracy while minimizing false positive rate in detecting fraud?
2. How does the addition of PCA-1 as a feature individually affect the performance of each model?
3. Does the impact of the impact of PCA-1 depend on the type of training algorithm?

**Methods**

We conducted our experiments within a Google Colaboratory notebook environment. The dataset was loaded into a dataframe using the pandas library (1.1.4) in Python, we started by checking the dataset to verify that it was clean and summarizing basic descriptive statistics. After that, we created a scatter plot matrix using the mlxtend library (0.14.0) to look at the relationships between the time, amount, and class of each transaction. These steps allowed us to have a better understanding of our dataset, and then proceed to creating different supervised classification models.

Prior to constructing models, the dataset was split 70-30 into training and testing sets. Using the scikit-learn library (0.22.2.post1), the first model we created was a cross-validated logistic regression model using defaults for all parameters except class weights (balanced) and maximum iterations. We conducted grid searches to optimize the maximum number of iterations without convergence and settled on value max\_iter = 4000. The runtime of constructing and training the model took approximately 18 minutes. Once trained, we evaluated prediction performance with the testing set to generate model comparison metrics in the form of accuracy averages, precision, recall (Table 1), F1 score, confusion matrix, Receiver Operating Characteristic (ROC) curve, and Area Under ROC curve (AUC-ROC).

The second model created with scikit was an SVM model. For this model, we started by making a basic SVM model using default parameters except for class weights and Cs value. Due to the lengthy training runtime, we conducted a manual search without cross-validation instead of a grid search to optimize the Cs value, settling on 0.1. Our model’s training time, with balanced class weights, was approximately two hours. We calculated the same performance metrics listed above for the SVM model (Table 1).

We continued with a Random Forest model using scikit. In this case, we tuned the model according to the number of estimators (100) and maximum tree depth (2) based on suggestions from a literature search (Koehrsen 2018a; Koehrsen 2018b; Srivastava 2015). We also trained with balanced class weights, as with the previous scikit models. Once trained, we calculated the same detection performance metrics (Table 1).

The last model we created was a Neural Network model, built using the Keras library (2.4.3). For this model, we first standardized the data, and then created the model. The model consisted of a dense input layer, hidden layer, and output layer (see notebook). We trained the neural network for 100 epochs. Finally, we tested the model and recorded metrics.

For the dataset augmented with the PCA-1 outlier scores, we ran the same series of models with the same hyperparameters and metrics, with the exception of the SVM model due to processing limitations. We will use all of this information in order to determine the best supervised classification model for predicting fraudulent transactions and examine the effect of the PCA-1 outlier scores. The PCA-1 outlier scores were calculated by transforming the original dataset into a single PCA component and then performing an inverse transform to get a new series of observation vectors. We calculated the difference between the reverse-transformed vectors and the original dataset, converted the differences to absolute values, and then summed the components of each difference vector. Higher values thus correspond to observations which deviated further from the reverse-transformed “predictions” of the PCA model (Carcillo et. al. 2019).

**Analysis and Results**

To analyze the project notebook, accuracy, precision, recall, and the confusion matrix will be compared for each of the models. Included below are the definition of these terms.

* Accuracy: the rate of outcomes that were predicted with the correct tag; the fraction of the predictions the model got right ( true positives and true negatives / total)
* Precision: the number of true positives over the total predicted positive.
* Recall: the number of true positives over true positives and false negatives (undetected frauds in this case) (Shung, 2020)
* AUC-ROC: the area below the ROC curve, defined as the rate of true positive predictions in terms of the rate of false positive predictions over the course of the training set. The optimal value is equal to 1, corresponding to a model which only ever yields true positive predictions. This measure is important for understanding how many false positives one can expect from a model for a given degree of precision.

When deciding which model is “most accurate” we considered what the model is trying to solve. In our project, we want a model that has a minimum number of false negatives, meaning a fraudulent transaction is classified as normal. This is optimal for credit card fraud because it is easier to fix a transaction that is not fraudulent and predicted as fraudulent (a simple text to the consumer to verify the transaction), then to fix a transaction that is fraudulent and predicted as normal (can either go undetected or the consumer will have to bring it the attention of the corporation). Because we want to limit false negatives, we look at each model’s recall, the highest rate of true positives out of the total number of positives. Simultaneously however, a model with high recall but a very high false positive rate would burden fraud investigators and runs the risk of being unduly disregarded (dal Pozzolo et. al. 2017). Below (Table 1) is a table of each model’s accuracy, precision, and recall and then the models confusion matrix.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** |
| Logistic Regression | .9723 | .0558 | .9145 |
| SVM | .9874 | .1112 | .8684 |
| Random Forest | .9980 | .4723 | **.9421** |
| Neural Network | **.9994** | **.8984** | .7566 |
| Logistic Regression with PCA-1 | .9809 | **.**0782 | .9013 |
| Random Forest with PCA-1 | .9974 | .3886 | .8487 |
| Neural Network with PCA-1 | .9992 | .75460 | .8092 |

\*The highest values in each column are bolded

Table 1: Accuracy, Precision, and Recall for models

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Figure 1: Confusion Matrices for each model

The accuracy of all the models are between .97 and .99. Random Forest has the largest recall. When the recall is large, the number of false positives is low. Looking at the confusion matrices, Logistic Regression had 13 false negatives but 2,354 false positives. Random Forests had 24 false negatives and 143 false positives. While Random Forest has more false negatives than Logistic Regression, it has significantly fewer false positives. While our priority is to minimize false negatives, we also do not want an unnecessary amount of false positives. From this analysis, Random Forest is the model to best predict credit card fraud because it minimizes the number of false negatives without drastically increasing the number of false positives.

In terms of PCA-1 augmentation, for the Logistic Regression model the augmented dataset improved accuracy by reducing the false positive rate by 0.75%, but also missed two additional fraud instances (0.01%) as can be seen in Figure 1. The Random Forest model caught one additional instance of fraud and gained 0.07% in false positive rate. The Neural Network model lost a single false positive but also detected four fewer true positives. The effects can also be seen based by comparing between the shapes of ROC curves (Figure 2), where the Random Forest and Neural Network models’ curves increased slightly in “steepness”.

|  |  |
| --- | --- |
|  |  |
|  |  |

Figure 2: Comparison of ROC curves of models using original datasets vs. PCA-1 augmented datasets, with Area Under the Curve labelled in the lower right legend. The Random Forest model and the Neural Network model’s AUC-ROC both benefited slightly with the PCA-1 outlier scores, reducing the expected rate of false positives for a maximized true positive rate, However the Logistic Regression model’s AUC-ROC worsened with PCA-1.

**Conclusion**

Due to the class imbalance of the dataset, the accuracy values were all significantly inflated, so the most important points of comparison became the true positive, false positive, and false negative rates. Computational performance also became a significant factor, leading to dropping the PCA-1 augmented SVM model entirely. Based on the combination of these factors, our experiment supports the Random Forest model as the most effective fraud detection model, because while the Neural Network drastically reduced the false positive rate with only minor gains in false negatives, the Random Forest model has the advantage of being an order of magnitude faster in terms of computation speed. Neural Network models may have advantage in fraud detection contexts where computation speed is less critical, however. As for the effect of PCA-1 augmentation with the dataset, it appears to lead to tradeoffs between precision and recall that depend on the specific model type. It may be useful for tweaking models when such tradeoffs are desirable, but further research is needed to understand the exact nature of its effects on these models.

For future research, we would like to be able to further tune the parameters. While we tuned our parameters to maximize recall, we would have liked to implement more GridSearches and RandomizedSearches. The computers in which we ran our program did not have enough processing power to compute these in a timely manner. Second, we would like to do further research and experimentation with voting ensembles to see how different voting configurations would impact precision and recall. Last, though the dataset was a timeseries, training and testing sets were sampled randomly from the entire dataset. Creating training and testing sets based on time slices would be a more realistic representation of the fraud detection problem and may better separate performance between the models.

From our project, we were able to further understand the concepts presented in DATA 602, and apply the concepts to a real life scenario. We were able to create models, tune the parameters, calculate measurements of accuracy, precision, recall, and confusion matrices in order to make informed decisions on the best model to implement for credit card fraud detection. From our analysis we weighed multiple performance metrics and model properties to reach a conclusion about the most effective training algorithm for the problem. An implementation of this algorithm could be used for banks and credit card companies, for example as a rapid consumer notification system. To extend upon this project, we could implement further parameter tuning, and create voting ensembles. By creating predictive models, that forecast if a transaction is fraudulent, companies and consumers can be more vigilant against fraud. With successful implementation of adaptive, automated fraud detection, the economic impact from fraud and delayed detection of fraud can be mitigated.

**References**

Bhatla, T. P., Prabhu, V., & Dua, A. (2003). Understanding credit card frauds. Cards business review, 1(6), 1-15.

Bolton, R. J., & Hand, D. J. (2001). Unsupervised profiling methods for fraud detection. Credit scoring and credit control VII, 235-255.

Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. Statistical science, 235-249.

Boucher, E. A. M. (2020). Outlier Detection Methods Applied to Financial Fraud. LU-CS-EX.

Carcillo, F., Le Borgne, Y. A., Caelen, O., Kessaci, Y., Oblé, F., & Bontempi, G. (2019). Combining unsupervised and supervised learning in credit card fraud detection. Information Sciences.

Chollet, F. et. al. (2015). Keras (2.4.3) [Python library]. <https://keras.io>

Dal Pozzolo, A., Caelen, O., Le Borgne, Y. A., Waterschoot, S., & Bontempi, G. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. Expert systems with applications, 41(10), 4915-4928.

Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2017). Credit card fraud detection: a realistic modeling and a novel learning strategy. IEEE transactions on neural networks and learning systems, 29(8), 3784-3797.

Federal Trade Commission. (2020). Consumer Sentinel Network Data Book 2019 [Report]. Retrieved from <https://www.ftc.gov/system/files/documents/reports/consumer-sentinel-network-data-book-2019/consumer_sentinel_network_data_book_2019.pdf>

Hadar, M. (2019, September 13). Think your credit card is safe in your wallet? Think again. Retrieved November 27, 2020, from <https://www.washingtonpost.com/business/think-your-credit-card-is-safe-in-your-wallet-think-again/2019/09/11/05e316e4-be0e-11e9-b873-63ace636af08_story.html>

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95. IEEE Computer Society.

Lebichot, B., Le Borgne, Y. A., He-Guelton, L., Oblé, F., & Bontempi, G. (2019, April). Deep-learning domain adaptation techniques for credit cards fraud detection. In INNS Big Data and Deep Learning conference (pp. 78-88). Springer, Cham.

Koehrsen, W. (2018a). Hyperparameter Tuning the Random Forest in Python [Web Article]. Towards Data Science. Retrieved from <https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74>

Koehrsen, W. (2018b). An Implementation and Explanation of the Random Forest in Python [Web Article]. Towards Data Science. Retrieved from <https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76>

Kou, Y., Lu, C. T., Sirwongwattana, S., & Huang, Y. P. (2004, March). Survey of fraud detection techniques. In IEEE International Conference on Networking, Sensing and Control, 2004 (Vol. 2, pp. 749-754). IEEE.s business review, 1(6), 1-15.

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference (Volume 445, pp 56-61).

Micenková, B., McWilliams, B., & Assent, I. (2014, August). Learning outlier ensembles: The best of both worlds–supervised and unsupervised. In Proceedings of the ACM SIGKDD 2014 Workshop on Outlier Detection and Description under Data Diversity (ODD2). New York, NY, USA (pp. 51-54).

Niu, X., Wang, L., & Yang, X. (2019). A comparison study of credit card fraud detection: Supervised versus unsupervised. arXiv preprint arXiv:1904.10604.

Pedregosa et. al. (2011). Scikit-learn: Machine Learning in Python. JMLR 12. (pp. 2825-2830)

Raschka, S. (2018). MLxtend: Providing machine learning and data science utilities and extensions to Python’s scientific computing stack. The Journal of Open Source Software. 3(24). The Open Journal.

The Federal Reserve. (2018, October 18). Changes in U.S. Payments Fraud from 2012 to 2016. Retrieved November 27, 2020, from <https://www.federalreserve.gov/publications/2018-payment-systems-fraud.htm>

The Pandas Development Team. (2020). pandas-dev/pandas (1.1.4) [Python Library]. Zenodo. <https://doi.org/10.5281/zenodo.3509134>

Shung, K. (2020, April 10). Accuracy, Precision, Recall or F1? [Web Article]. Towards Data Science. Retrieved November 28, 2020, from <https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9>

Srivastava, T. (2015) Tuning the parameters of your Random Forest model [Web Article]. Analytics Vidhya. Retrieved from <https://www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/>

Veeramachaneni, K., Arnaldo, I., Korrapati, V., Bassias, C., & Li, K. (2016, April). AI^ 2: training a big data machine to defend. In 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS) (pp. 49-54). IEEE.

Waskom, M. et. al. (2020). mwaskom/seaborn (0.11.0) [Python library]. Zenodo.