**Machine Learning for Credit Card Fraud Detection**

Eyas Abu Elhouf

Katrina Ong

Melvin Cheriyan

Pranav Gujar

Bista Bijo

Praveen Kumar

School of Business, St. Lawrence College

ADMN5006: Financial Analytics

Prof. Maverick Ramsaran

December 17, 2022

**I. Business Use Case**

The fields of finance and banking are extremely significant in the modern period since practically all people are required to interact with financial institutions, either in person or online, at some point in their lives. Because of the financial information system, both the public sector and the private sector have seen significant increases in their levels of productivity as well as their profitability. Credit cards and online net banking are the two most common payment methods used in modern-day e-commerce application system transactions. These systems have an alarmingly high risk of being compromised by newly developed methods of attack. Detecting fraudulent activity in banking is one of the most important components of the banking industry today. This is because the financial industry is such a significant part of our lives.

In recent years, a significant amount of financial harm has been made to online stores as a direct result of credit card theft, which is a major concern for many online retailers. Protecting against fraudulent use of credit cards is made possible by a number of different security measures, such as 3D Secure and SecureCode. A significant reduction in credit card fraud may be achieved by the use of these processes during the order process. However, this increased level of safety comes with a number of important drawbacks for business owners: The mandatory additional step in the order process causes a lot of clients to feel apprehensive or even causes some of them to entirely abandon their orders. The amount of lost income as a direct result of this can soon outweigh the cost savings achieved by reducing the total number of instances of fraud. This issue may be avoided entirely by using a clever algorithm. An algorithm like this one may identify "suspect" orders and will only activate the additional security measures for these questionable purchases.

**II. Methodology**

1. **Exploratory Data Analysis**

The dataset has PCA (principal component analysis) already applied. To make data less dimensional, a common unsupervised learning approach is principal component analysis. It is used to reduce the dimensions of a dataset and ensure that features are independent of each other. One of the main reasons for using PCA for this business is to keep the financial data private.

We first imported the data using pd.read\_csv and then we checked for the data types of all the columns, we used df.describe() and found out that only 2 variables (i.e., Time and amount) are useful. Next, we looked for the number of samples in the data frame and checked if there are any null values which turned out to be 0. We used df['Class'].value\_counts() and found that the data was highly imbalanced (0 is for Normal and 1 is for Fraud transactions) 284,315 : 492.

Later, we checked the distribution of the independent variables, most of them were normally distributed. We also compared the normal and fraud transactions plotted considering time and amount, but no patterns were discovered. We also found that variables are not highly correlated, as expected from the result of applying PCA. However, we found that there are a few duplicate values which we removed eventually using df.drop\_duplicates(inplace=True), reducing our dataset to 283,253 non-fraud cases and 473 fraud cases.

1. **Data Processing**

For the train-test split, we first separated the features and the target variable. Then we split the dataset into two sets. The training set consists of 75% of the dataset and the test set consists of 25% of the data samples. After this, we checked the dimensions of the split to see if there are any problems.

The Robust Scaler was trained on the training set. It was then applied to both training and testing sets to avoid data leakage. This scaler removes the median and scales the data according to the quantile range. It is also done to standardize the dataset. Further, only the independent variables were scaled since the target is encoded as binary denoting fraud (1) and non-fraud (0) transactions.

The outliers were not removed or treated, since the principal components hold no interpretable meaning after PCA has been applied to it. This choice not to treat outliers is also for the purpose of training the machine learning models to handle real life scenarios which will inevitably encounter outlier information.

1. **Model Development**

There are several ways to address the problem of having an imbalanced dataset. Some methods that we have previously attempted include a combination of oversampling the minority class and under-sampling the majority class.

In this report, we attempted three (3) new approaches to develop an appropriate machine learning model for an imbalanced dataset:

* 1. Using Class Weights to Penalize Fraud Misclassification
  2. Using Borderline Oversampling to ­­­­­­­­­­Improve Training on Separability of Classes
  3. A Combination of the First Two Approaches

In the following section, we will further elaborate on each approach.

***First Approach: Using Class Weights to Penalize Fraud Misclassification***

Misclassifying fraud cases have a greater cost to financial institutions and merchants since they will often shoulder the cost of the stolen amount. Meanwhile, misclassifying non-fraudulent transactions may incur some administrative expenses in temporarily blocking the client’s card and verifying the authenticity of the transaction. However, while the losses for the latter may be a considerable amount, these may also carry less weight. Thus, it is of greater importance to develop a model that will correctly detect fraud.

Our first approach deals with imbalanced classes by increasing the penalty for misclassifying the minority class (i.e., fraud cases). Increasing the penalty for these mistakes in classification will help the model correctly identify these few vital fraud cases.

We trained three (3) machine learning algorithms with assigned class weights. In all algorithms, a penalty of 1 was assigned for misclassifying cases tagged as non-fraud (denoted by ‘0’), while a greater penalty of 10 was assigned for misclassifying cases tagged as fraud (denoted by ‘1’). There is also an option to set class weights to “balanced mode”, so that weights are inversely proportional to class frequencies in the training data. However, “balanced mode” was not employed due to limitations in computational power.

Below are the trained algorithms and their respective parameters:

1. Logistic Regression: Default parameters were kept, except for the maximum number of iterations taken for the solvers to converge. The default maximum iteration of 100 was not enough for solvers to converge, so this was increased to 200.
2. Support Vector Machines: Default parameters were kept for this algorithm.
3. Random Forest: Default parameters were kept for this algorithm.

***Second Approach: Using Borderline Oversampling to Improve Training on Separability of Classes***

Fraudsters try to conceal their transaction as a legitimate transaction to be successful. This makes it difficult for financial institutions to detect these irregularities since the distinction between fraud and non-fraud cases are muddled.

Relatedly, most classification algorithms attempt to achieve better prediction by learning the boundary/ borderline of each class. In this context, misclassification of fraud or non-fraud cases occurs since there may be some instances in each class that are close to the boundaries. This could be random, or it could be an effect of deliberate concealment of fraud. Nonetheless, examples that are far from the borderline are easier to detect and may not provide as much value to the training process compared to datapoints that are on the borderline which are more difficult to detect.

Thus, our second approach deals with imbalanced classes through a data processing method called Borderline Synthetic Minority Oversampling Technique (Borderline-SMOTE). This technique oversamples borderline samples from the minority class. In other words, it generates synthetic samples along the boundary between the minority sample and its nearest neighbors. We have set the k nearest neighbors to 5, which is the default setting.

We used “borderline-1” Borderline-SMOTE, and this kind of technique makes a distinction between safe, dangerous, or noisy borderline samples. The dangerous borderline samples, or the samples that make classification difficult for the algorithm, are used for oversampling. An m nearest neighbor value of 10 is used for detecting “danger” samples.

Similar to the first approach, we trained three (3) machine learning algorithms, as follows:

1. Logistic Regression: The default maximum iteration of 100 was not enough for solvers to converge, so this was increased to 200.
2. Support Vector Machines: The default parameters were kept for this model.
3. Random Forest: The default parameters were kept for this model.

***Third Approach: A Combination of the First Two Approaches***

Finally, we tried to combine the first 2 approaches, since the context for each approach can be observed in our business case. We trained the best performing model in terms of F1 Score and Recall from the first two approaches, which was Random Forest.

The same class weights in the first approach were used on oversampled data through Borderline-SMOTE. The parameters for Borderline-SMOTE from the second approach were kept the same.

The results are further discussed in the next section of this report.

**III. Model Comparison**

After training the models, they were applied to the testing set to predict the class of each sample.

The following is a summary of the key performance metrics of the machine learning models used:

**A picture containing text

Description automatically generated**

Since we are dealing with an imbalanced dataset, accuracy is not always the best metric to use, since this could be misleading due to the high number of correct detections in the majority class. The best metric would be to see how well each model detects the minority class - credit card fraud transactions. As a result, precision, recall, and the F1 score must all be considered.

Precision is the proportion of true positives (true fraud cases) to all predicted positives. Meanwhile, recall indicates how many actual positives were correctly identified. The F1 score displays the harmonic mean of these two metrics.

It is important to have a model that will minimize both administrative expenses for misclassifying non-fraud cases and minimize loss from missing actual fraud cases. Thus, the F1 score may provide more value in assessing model performance.

Logistic Regression for the first two approaches has the highest Area Under the Curve (AUC), which shows promising separability between fraud and non-fraud cases. However, its F1 scores are low compared to other algorithms used in both approaches.

Random Forest appears to perform the best in terms of F1 score within Approach 1 (0.916969) and within Approach 2 (0.931940). However, combining these approaches produced an F1 Score of 0.917794, which only offered an improvement from Approach 1. Approach 2 for Random Forest still performed the best in terms of F1 score among all models. It appears to perform well on most measures, including precision, recall, and F1 Score. It has precision, recall, and F1 scores of 0.973242, 0.897286, and 0.931940, respectively.

For these reasons, we chose Approach 2: Random Forest for deployment.

**IV. Future Steps**

While our methods produced classifiers with relatively high performance, we can improve performance using ensemble learning methods and hyperparameter tuning with grid search. Due to limitations in computational power, these methods were not employed.

By combining predictions from multiple models, ensemble learning would improve performance. Bagging, stacking, and boosting are examples of ensemble learning techniques. Bagging is the process of averaging the predictions of various decision trees. Meanwhile, another model is involved in stacking to learn how to best combine the predictions of different models. Lastly, boosting involves adding models together and using a majority vote to combine these models together and boost their performance.

The appropriate method to use would be determined by the bias and variance of each model. These can be detected by using K-fold cross-validation to look for overfitting.

Additionally, fine tuning the hyperparameters of each model may result in minor performance gains. A systematic approach would be to run a grid search on various parameters and find the combination that produces the best result.

**References**

Alamri, M., &amp; Ykhlef, M. (2022). Survey of credit card anomaly and fraud detection using sampling techniques. Electronics, 11(23), 4003. https://doi.org/10.3390/electronics11234003

Borderlinesmote. BorderlineSMOTE - Version 0.10.0. (n.d.). Retrieved December 16, 2022, from https://imbalanced-learn.org/stable/references/generated/imblearn.over\_sampling.BorderlineSMOTE.html

Brems, M. (2022, January 26). A one-stop shop for principal component analysis. Medium. Retrieved December 16, 2022, from https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c

Han, H., Wang, W.-Y., &amp; Mao, B.-H. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. Lecture Notes in Computer Science, 878–887. https://doi.org/10.1007/11538059\_91

Imbalanced classification: Handling imbalanced data using Python. Analytics Vidhya. (2022, November 30). Retrieved December 16, 2022, from https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/

Imprint. (n.d.). Retrieved December 16, 2022, from https://www.inwt-statistics.com/business-cases/credit-card-fraud.html

Patil, S., Nemade, V., & Soni, P. K. (2018). Predictive modelling for credit card fraud detection using data analytics. Procedia Computer Science, 132, 385-395. doi: 10.1016/j.procs.2018.05.199