Credit Card Fraud Detection

Rohan R. Deshmukh

Harrisburg University of Science & Technology

Abstract

This paper discusses approached to detect credit card fraud transactions using various machine learning model. The purpose of this study is to evaluate popular machine learning models for classification-based problems and formulate a model that classifies the fraudulent data with 100% accuracy and 0% False Negatives.

Keywords: credit card transactions, anomaly detection, over-sampling, random forest, Regression, high unbalanced data

Word count: 3087

Credit Card Fraud Detection

Introduction

The question: Can we detect fraudulent transactions from credit card transactions data with an almost perfect accuracy? My goal here is to detect 100% of the fraudulent transactions and also reducing the false positives i.e. incorrect fraud classifications. In this modern age of cashless economy, it is important that credit card companies are able to recognize and differentiate between fraudulent and legitimate credit card transactions. It is necessary from a customer point-of-view as well so that they are not charged for items they did not buy. The importance of this study can be estimated from the total volume generated by payment cards reached $34.119 trillion worldwide in 2017. Gross fraud losses to issuers, merchants, and merchant acquirers reached $24.26 billion, up 6.4% from 2016. In this modern age of cashless economy, it is important that credit card companies are able to recognize and differentiate between fraudulent and legitimate credit card transactions. It is necessary from a customer point-of-view as well so that they are not charged for items they did not buy.

Prior Research

Fraud detection systems existed in a typical Credit risk management department of a financial institution since 1990. The accuracy and efficiency of these systems were very low in those times. Some of the issues faced were as follows:

1. Limited ability to capture data
2. Lack of architecture components like Hadoop to store huge amount of data
3. Limitations of computational power to process data and perform complex exploratory analysis

Because of these limitations, these fraud detection systems were based on static rules rather than pattern recognition / machine learning rules. Some of these static rules covered for checking the transaction against a set threshold, location of purchase, etc. The process involved blocking transactions and confirming with customers as to whether or not they authorized the transaction. As we entered late 90s, institutions started leveraging machine learning techniques and anomaly detection methods to improve the accuracy of these systems. Brause, Langsdorf, and Hepp (1999); Ghosh and Reilly (1994).

The areas focused by these researchers covered the modeling side of things. A lot of attention was given to creating a classifier that can classify the transactions. A study published by Chan et al. (1999); Stolfo et al. (1997) presented a paper that provided a comparison and analysis of multiple classifiers along with their accuracy.

In these papers, the features for the transaction in consideration is generated by aggregating transactional information from the immediate past of the transaction under consideration. With the recent advancement in cashless economy, the amount of transactions is really high. So, it was necessary to have a automated framework to process the huge data as a data stream rather than a data dump. Carcillo et al. (2017) created a similar framework to process a stable stream of data using Apache Spark. This paper draws inspiration from Shen, Tong, and Deng (2007), who presented classification models for credit card fraud detection. Whitrow et al. (2009) emphasize the importance of aggregate features in improving accuracy. This paper was the inspiration for performing aggregated feature engineering on the Random forest model.

Data Analysis

For this project, I have used credit card transaction data. The dataset has been collected and analyzed during a research collaboration of Worldline and the Machine Learning Group (http://mlg.ulb.ac.be) of ULB (Université Libre de Bruxelles) on big data mining and fraud detection. This dataset has been listed as credit card transaction dataset in Kaggle (Credit Card Fraud Detection, Machine Learning Group – ULB. (2016). Retrieved from https://www.kaggle.com/mlg-ulb/creditcardfraud).

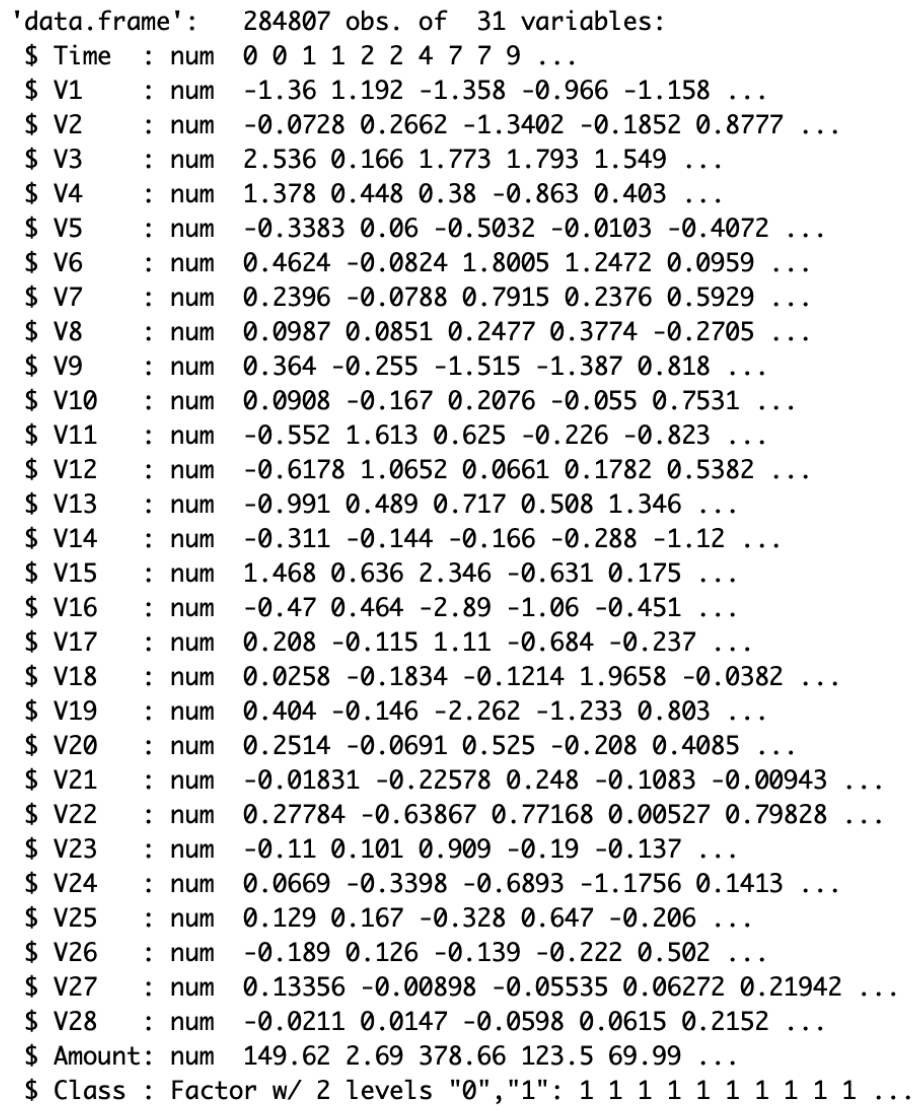
The datasets contain transactions executed using credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset description mentions that it is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. The following table summarizes the imbalance in the data.

Table 1

Table listing Non-fraud and fraud transactions

Non-Fraud	Fraud
284315	492

Using summary () function, we can see that there are no missing (NA) values in the data. There are totally 31 columns in the data. One column, Class is the dependent variable; and, can have either 0 (not fraud) or 1(fraud) value. Another two columns, Amount is the amount of the transaction; Time is the time of the transaction. The rest of the features (28), are named from V1 to V28. The names of these features are anonymized. The following figure 1 summaries the available data.



```
'data.frame':   284807 obs. of  31 variables:
 $ Time  : num  0 0 1 1 2 2 4 7 7 9 ...
 $ V1    : num  -1.36 1.192 -1.358 -0.966 -1.158 ...
 $ V2    : num  -0.0728 0.2662 -1.3402 -0.1852 0.8777 ...
 $ V3    : num  2.536 0.166 1.773 1.793 1.549 ...
 $ V4    : num  1.378 0.448 0.38 -0.863 0.403 ...
 $ V5    : num  -0.3383 0.06 -0.5032 -0.0103 -0.4072 ...
 $ V6    : num  0.4624 -0.0824 1.8005 1.2472 0.0959 ...
 $ V7    : num  0.2396 -0.0788 0.7915 0.2376 0.5929 ...
 $ V8    : num  0.0987 0.0851 0.2477 0.3774 -0.2705 ...
 $ V9    : num  0.364 -0.255 -1.515 -1.387 0.818 ...
 $ V10   : num  0.0908 -0.167 0.2076 -0.055 0.7531 ...
 $ V11   : num  -0.552 1.613 0.625 -0.226 -0.823 ...
 $ V12   : num  -0.6178 1.0652 0.0661 0.1782 0.5382 ...
 $ V13   : num  -0.991 0.489 0.717 0.508 1.346 ...
 $ V14   : num  -0.311 -0.144 -0.166 -0.288 -1.12 ...
 $ V15   : num  1.468 0.636 2.346 -0.631 0.175 ...
 $ V16   : num  -0.47 0.464 -2.89 -1.06 -0.451 ...
 $ V17   : num  0.208 -0.115 1.11 -0.684 -0.237 ...
 $ V18   : num  0.0258 -0.1834 -0.1214 1.9658 -0.0382 ...
 $ V19   : num  0.404 -0.146 -2.262 -1.233 0.803 ...
 $ V20   : num  0.2514 -0.0691 0.525 -0.208 0.4085 ...
 $ V21   : num  -0.01831 -0.22578 0.248 -0.1083 -0.00943 ...
 $ V22   : num  0.27784 -0.63867 0.77168 0.00527 0.79828 ...
 $ V23   : num  -0.11 0.101 0.909 -0.19 -0.137 ...
 $ V24   : num  0.0669 -0.3398 -0.6893 -1.1756 0.1413 ...
 $ V25   : num  0.129 0.167 -0.328 0.647 -0.206 ...
 $ V26   : num  -0.189 0.126 -0.139 -0.222 0.502 ...
 $ V27   : num  0.13356 -0.00898 -0.05535 0.06272 0.21942 ...
 $ V28   : num  -0.0211 0.0147 -0.0598 0.0615 0.2152 ...
 $ Amount: num  149.62 2.69 378.66 123.5 69.99 ...
 $ Class : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Figure 1 Summary of credit card transaction data

The description of the data says that all the features went through a PCA transformation (Dimensionality Reduction technique) except for Time and Amount. During PCA transformation process, features need to be previously scaled. So, in our case, it is safe to assume that all the "V" features are scaled by the data publishers. Correlation matrices are very useful in understanding our data. We want to know if there are features that influence heavily in whether a specific transaction is a fraud. The following figure gives us an intuition with regards to which features are influencing the most.

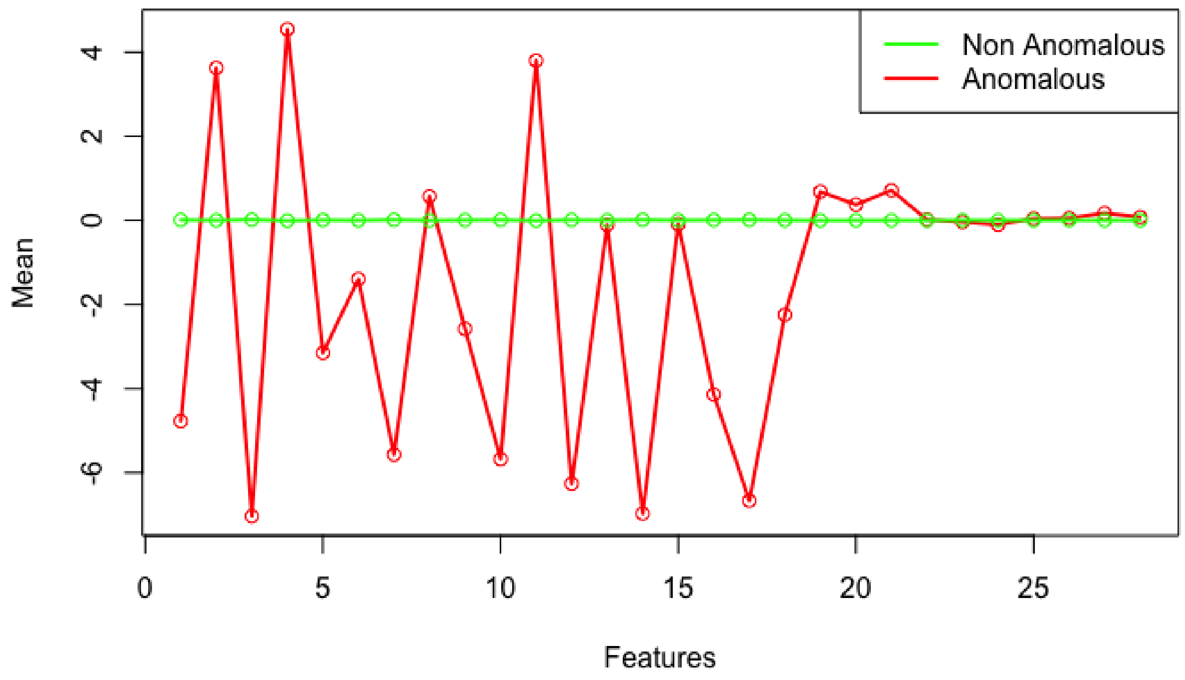


Figure 2 represents the mean of the data points (both non-anomalous and anomalous) variation against the available features.

From the above plot, we can conclude that V17, V14, V12 and V10 are negatively correlated.

While V2, V4, V11, and V19 are positively correlated.

Using barplot, we observe that there is huge imbalance. There are only 0.17% of the rows with value Class = 1. Typically, in such cases, we can either choose to preserve the data unbalancing or use oversampling (of the data with minority value of target variable) or undersampling (of the data with majority value of the target variable). We will be using over sampling technique in our approach. The following figure shows the imbalance in the target class.

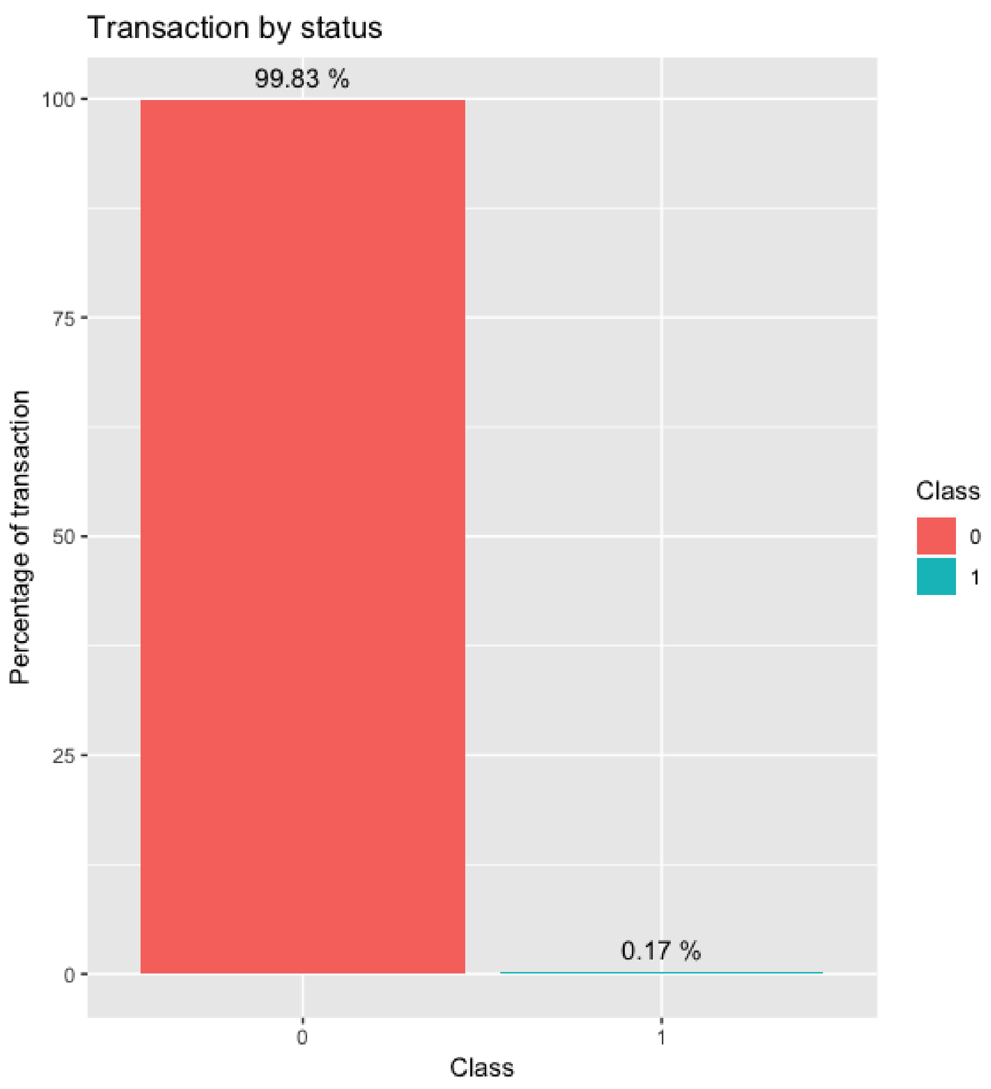


Figure 3 represents the imbalance in the data.

Also, using cor () and corrplot () functions, we can find the correlation between the variables. We can observe that most of the data features are not correlated. This is because before publishing, most of the features were presented to a Principal Component Analysis (PCA) algorithm. The features V1 to V28 are most probably the Principal Components resulted after propagating the real features through PCA. This might create some limitations in terms of tuning the variables as per some context information. The following figure gives us a correlation plot that can be used to examine any relationship between variables.

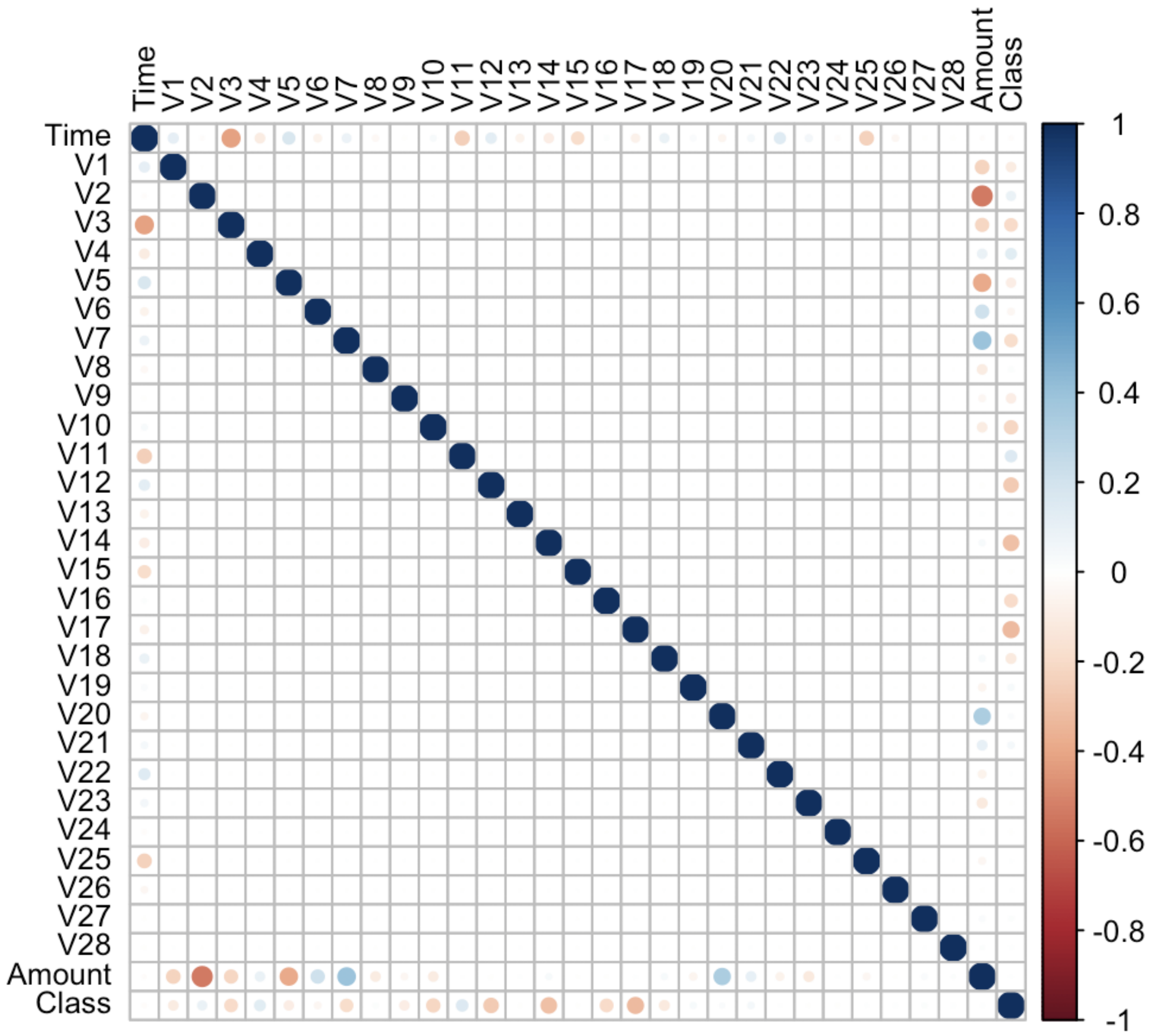


Figure 4 displays the correlation plot

Model Formulation

I have split the dataset into training and testing set (70:30). Thus, training the model on 70 percent of dataset and testing it on the rest of 30% of transactions.

Logistic Regression

These models are normally used to manage correlation between the predictors to create an almost accurate linear model with the fewest number of correlated predictors. However, the values in the dataset have been 'decorrelated' via principle component analysis (PCA) resulting in non-correlated independent predictors. Therefore, standard logistic regression considering only major effects is used. A p-value of 0.01 is used as the significance threshold for terms in the glm.

Random Forest

Random forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. Random forests are a good option for decision trees' disadvantage of overfitting to their training set. I have used ntree=50 for the Random forest. Also, for finding the optimal value of 'ntree', I plotted the model output for ntree=200 and it was observed that after ntree=50 the model doesn't drastically improve. At this stage, the decision to increase the ntree value revolves around the dilemma of nominal increase in accuracy/ efficiency vs computation time.

In the random forests' literature, this is referred to as the 'mtry' parameter. There is extensive discussion in the literature about the influence of 'mtry'. Cutler et al. (2007) reported that different values of 'mtry' did not affect the correct classification rates of their model and that other performance metrics (sensitivity, specificity, kappa, and ROC AUC) were stable under different values of 'mtry'. On the other hand, Strobl et al. (2008) reported that 'mtry' had a strong influence on predictor variable importance estimates. Due to the conflicting evidence reported in the literature, it is suggested to start with the default value. For classification models, the default value of 'mtry' variable is the square root of the number of predictor variables (rounded down). For regression models, it is the number of predictor variables divided by 3 (rounded down). I have used a loop evaluation method to find the optimal value of 'mtry' variable. This helps us in understanding the least MSE at every step. Thus, running the loop from mtry = 3 to mtry = 8. We would expect the optimal value to fall between 5 and 6 (Sqaure root of number of available features: 28). The following figure justifies the selection of mtry=5.

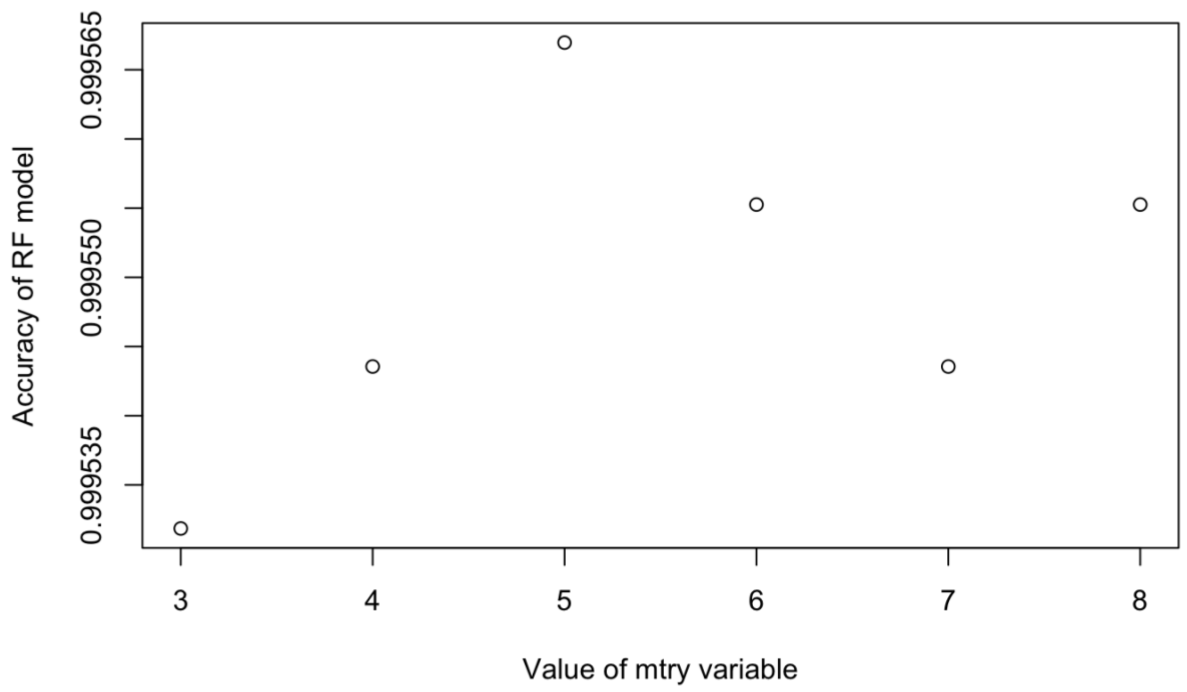


Figure 5 represents the value of the mtry variable against the accuracy score for Random forest algorithm

Support Vector Machines

A Support Vector Machine (SVM) is a classifier formally defined by a separating hyperplane. In short, given labeled training data, the algorithm creates an optimal hyperplane which categorizes new test data points. In two-dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side. In our case, as we 28 variables that positively and negatively affect the hyperplane, the svmlinear method within caret package is very time consuming. I was able to run this on a NVIDIA GPU powered AWS instance and it took us almost 3 hours to train this model. As a matter of fact, I have done cross validation using number =5 parameter in both Logistic Regression and SVM

Model testing and evaluation

I have trained the credit card transaction data on namely three main models viz. Logistic Regression, Linear Support Vector machine and Random Forest Algorithm. As part of the data pre-processing step, I split the data into training and testing set using the sample () function. As per the study conducted, one of the most efficient and used splitting strategy is 70:30 (Kevin K Dobbin and Richard M Simon). There are various other packages like caret which offer efficient functions for splitting data. I have experimented these functions and observed that the time efficiency did not improve drastically. Following are the testing results for the formulated models. In addition, I have also performed over-sampling for the fraud class using SMOTE () function from the DMwR R package.

As observed from data exploration, the dataset is very unbalanced. The total percentage of Non-Fraud transactions (Class = 0) is 99.83%, while that of fraud transactions (Class = 1) is 0.17%. The baseline accuracy to compare our models is 99.83. Because of this, even if our classifier wrongly predicts "1" many times the Specificity metric wouldn't decrease too much. Thus, a very low probability threshold would lead to a high Specificity and Recall classifier, giving a very high area under the ROC curve. However, this would not be considered to be a good classifier. Because of the nature of the problem, we prefer false positives than false negatives. i.e. we prefer high Recall than Precision. The number of TP is very small in comparison with the number of TN, the Confusion Matrix is less useful, since it is important to use a metric that include evaluation of FP and FN as well. It is important to minimize as much as possible the number of FN (Predicted: Not Fraud and True: Fraud) since their cost could be very large. Typically, AUC is used for such cases. Thus, we are aiming for a high Precision/Recall classifier with no less than 99.83% accuracy.

Table 2

Test results for predictive models

Model	Accuracy	AUC
Logistic Regression	99.93	84.1
Support Vector Machines	99.81	86.2
Random Forest	99.96	89.4

Random Forest – SMOTE over sampling

As far as the models I considered, I prefer using Random Forest. The model had a good AUC score as well as the model is efficient in terms of computation time. One of the few reasons behind using Random forests are as follows:

1. Random data subsets are created from the original dataset using bootstrapping.
2. At each node in the decision tree, a random set of features are considered to decide the best split. The model summary gives a good analysis about the value of mtry argument used in building the trees.
3. A decision tree model is fitted on each of the subsets.
4. All the predictions from each decision tree is then averaged out to calculate the final prediction.
5. Also, the importance () function helps to understand the contribution of each variable to the model. This is pretty useful in fine tuning the model.

I have coupled the random forest model with over-sampling technique. Oversampling technique focusses on increasing the proportion of the minority class by replicating the minority class samples in the data. In our case, we have accomplished this by increasing the number of fraud samples in the data. Eventually, the oversampling dataset had a 1:1 ratio in terms of Non fraud/Fraud transactions. This technique does not increase the information gain, it does raise the over-fitting issue, which causes the model to be too specific. It may well be the case that the accuracy for the training set is high, yet the performance for new datasets is actually worse. SMOTE synthesizes new minority cases between existing (real) minority cases. Hypothetically, SMOTE draws lines between existing minority cases. Because of this synthesizing process, the imbalance shrinks from very poor ration 99.83:0.17 to 1:1. This was helpful in increasing the AUC score. The following table lists the model metrics after performing over sampling of data using SMOTE. This over sampled data is used to train Random forest model. The random forest model has also been improved by performing feature engineering.

Table 3

Test results after Over sampling and feature engineering

Model	Accuracy	AUC
Random Forest	99.96	89.4
Random Forest + SMOTE over sampling	98.06	94.7

Conclusion

One of the most important conclusions of this project is that the classifying accuracy of a highly unbalanced dataset is not accurately represented the accuracy score of models. It is important to take into consideration all the aspects of the model including FP, TP, TN and FN. Also, over sampling the minority class drastically increases the AUC for Random forest. Furthermore, random forest (ensemble learning) is a better model than SVM and Linear regression. Credit card fraud has become very frequent and common these recent times. In order to reduce the losses caused by fraud transactions, there needs to be an efficient and automated mechanism of detecting frauds and strengthening the risk profile of a financial institution. One of the goals of this study is to identify a simple machine learning model that predicts fraud with a good accuracy score. In recent years, the trend of combining more than one algorithm has become common and widely used across many banks. This paper is a step in combining many existing techniques and contribute towards the effective ways of credit card fraud detection. In this paper, we survey three existing techniques for credit card fraud detection followed by comparing their results. Also, we have experimented with oversampling of data and hence we conclude that out of the top most classification techniques, random forest proves to be a good model with AUC score of 94.7.

Replication of work

My above work can be replicated on a system with at least 8GB RAM. I would recommend having a Nvidia GPU if you want to reduce the running time for a couple of algorithms, specifically SVM. The libraries that are needed have been listed in the code using require () function. So, even if the user doesn't have the necessary packages installed, the code will install and load the libraries. An evaluator needs to run the R-markdown file that has been submitted along this paper.

Future work

One of the many things that I would like to perform as part of any future work would be using Neural Networks. Though, this is a classification problem and general classification algorithms yield fair results. However, using ANN models from caret and h2O packages to classify the highly unbalanced data would be really interesting. This might be one of those machine learning problems, where a pipeline of algorithms might help us in achieving near 100% accuracy.

References

Machine Learning Group (http://mlg.ulb.ac.be) of ULB (Université Libre de Bruxelles) on big data mining and fraud detection.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer. (2002).SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research 16 (2002) 321–357. Retrieved from https://arxiv.org/pdf/1106.1813.pdf

Kevin K Dobblin, Richard M Simon. (2011). Optimally splitting cases for training and testing high dimensional classifiers. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3090739/

Ghosh, S., and Reilly, D. L. (1994). Credit card fraud detection with a neural-network. Proceedings of the Twenty-Seventh Hawaii International Conference on, volume 3, 621–630. IEEE. Retrieved from https://pdfs.semanticscholar.org/ba70/a74262adec9dcfa47b5710752d2537a07af4.pdf

Brause, R.; Langsdorf, T.; and Hepp, M. (1999). Neural data mining for credit card fraud detection. Proceedings. 11th IEEE International Conference on, 103–106. IEEE. Retrieved from http://sphinx.rbi.informatik.uni-frankfurt.de/asa/papers/ICTAI99.pdf

Chan, P. K.; Fan, W.; Prodromidis, A. L.; and Stolfo, S. J. (1999). Distributed data mining in credit card fraud detection. IEEE Intelligent Systems and Their Applications 14(6):67–74. Retrieved from https://ieeexplore.ieee.org/document/809570

Carcillo, F.; Dal Pozzolo, A.; Le Borgne, Y.-A.; Caelen, O.; Mazzer, Y.; and Bontempi, G. (2017). Scarff: a scalable framework for streaming credit card fraud detection with spark. Information fusion. Retrieved from https://arxiv.org/abs/1709.08920

Shen, A.; Tong, R.; and Deng, Y. (2007). Application of classification models on credit card fraud detection. International Conference on, 1–4. IEEE. Retrieved from https://ieeexplore.ieee.org/document/4280163

Whitrow, C.; Hand, D. J.; Juszczak, P.; Weston, D.; and Adams, N. M. (2009). Transaction aggregation as a strategy for credit card fraud detection. Data Mining and Knowledge Discovery 18(1):30–55. Retrieved from https://www.researchgate.net/publication/225586212_Transaction_aggregation_as_a_strategy_for_credit_card_fraud_detection