# CREDIT CARD FRAUD DETECTION PROJECT REPORT

**Udo Nweke**

July 8, 2019

**Introduction**
Getting a data science job after graduating from school is quite demanding. One of the reasons for this is that majority of the companies expect applicants to have an industry related experience. Having worked on a couple of projects while in school, I wanted to work on an industry related project. A friend of mine pointed me to Kaggle dataset [1] and I became interested in the project. Finally I had a dataset I could work on and learn something new.

**Problem Definition**
The goal of this project was to enable credit card companies recognize fraudulent transactions in order to protect customers from charges for items they did not purchase.

**Dataset Description**
The dataset contains transactions made by credit cards in September 2013 by European cardholders. The dataset presents transactions that occurred in two days, where there are 492 fraudulent transactions out of 284,315 transactions. It has 31 features, 28 of which have been anonymized and labeled V1 through V28. The non-anonymized three features are the Time, Monetary Amount, and the Class variable which is a binary feature that tells whether a transaction was fraudulent or not. Figure 4 shows a sample of the credit card transaction dataset.
Exploring and understanding the given dataset is one of the first steps involved in implementing a machine learning algorithm. For the credit card fraud detection project, I started with exploratory data analysis.

**Exploratory Data Analysis**
Since 28 out of the 31 features had been anonymized and labeled V1 through V28 in the form of Principal Component Analysis (PCA), I focused my exploratory analysis on the non-anonymized predictive features. A summary statistics of the predictive features in Figure 2 shows the mean, maximum and minimum and standard deviation of the predictive features. From the summary statistics (mean and maximum values) it can be seen that the distribution of the monetary value of all transactions is heavily right-skewed. This can be seen in Figure 1. The value of the mean is \$88.35% while the largest monetary transaction in the dataset \$25691.16%

Next, I explored the time feature. The dataset presents transactions that occurred in two days. The bimodal distribution in Figure 3 shows that there was a significant drop in the volume of transactions in approximately 28 hours.

I also explored the Class distributions which is the target variable. The visualization in Figure 5 shows the number of fraudulent and non-fraudulent transactions. We can see that most of the transactions are non-fraudulent which is unsurprising. 99.83% of the transactions were not fraudulent while only 0.17% of the transactions in the dataset were fraudulent. This also demonstrates how heavily imbalanced the dataset is.

As I continued to explore the dataset to have a better understanding of the features, I used a heatmap to see if there were any significant correlations between the predictors with and the target (Class) variable. Figure 6 shows that there seem to be relatively little significant correlations in the dataset. For a dataset with these many variables, this could be as a result of the huge class inbalance which might have distorted the importance of certain correlations with regards to the class variable.

**Data Cleaning and Preparation**
After exploring the dataset and understanding the features, the next step was to prepare the data for prediction. For this dataset, there were no missing values, I did not worry about missing value imputations.

Figure 1: Monetary Value Distribution Showing Heavy Right Skew

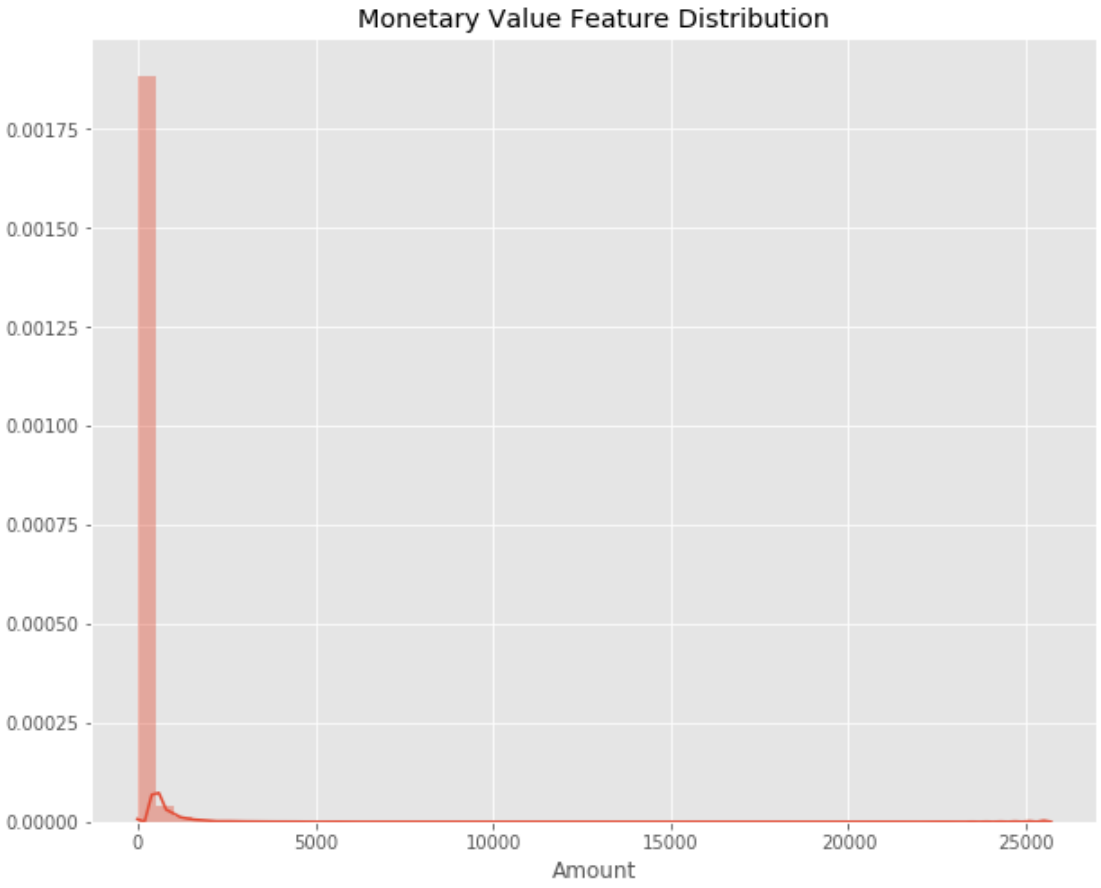


Figure 2: Summary Statistics of Non-Anonymized Predictors

| | Time | Amount |
|---|---|---|
| count | 284807.000000 | 284807.000000 |
| mean | 94813.859575 | 88.349619 |
| std | 47488.145955 | 250.120109 |
| min | 0.000000 | 0.000000 |
| 25% | 54201.500000 | 5.600000 |
| 50% | 84692.000000 | 22.000000 |
| 75% | 139320.500000 | 77.165000 |
| max | 172792.000000 | 25691.160000 |

Figure 3: Bimodal Time Distribution of Imbalanced Dataset

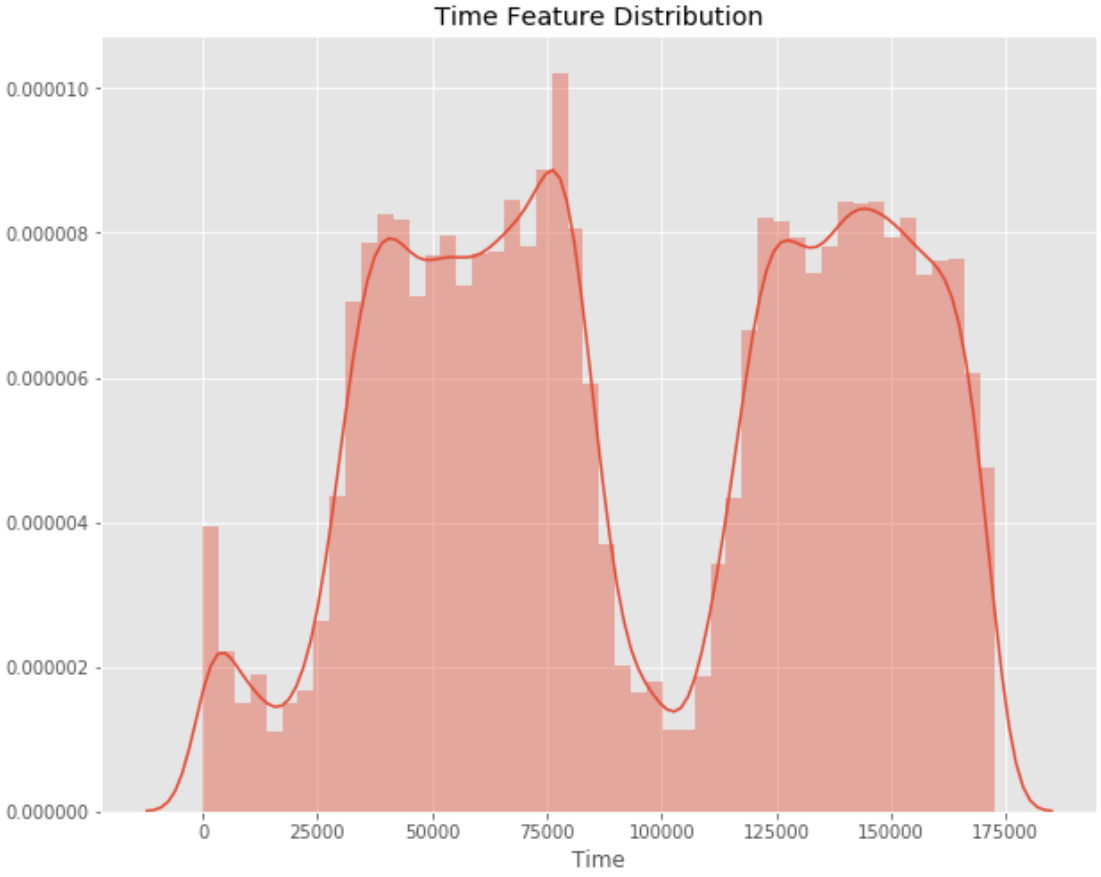



Figure 4: Sample Data State with Time and Anonymized Predictive Features Labeled V1 through V28

| Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 | V22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.0 | -1.359807 | -0.072781 | 2.536347 | 1.378155 | -0.338321 | 0.462388 | 0.239599 | 0.098698 | 0.363787 | ... | -0.018307 | 0.277838 |
| 0.0 | 1.191857 | 0.266151 | 0.166480 | 0.448154 | 0.060018 | -0.082361 | -0.078803 | 0.085102 | -0.255425 | ... | -0.225775 | -0.638672 |
| 1.0 | -1.358354 | -1.340163 | 1.773209 | 0.379780 | -0.503198 | 1.800499 | 0.791461 | 0.247676 | -1.514654 | ... | 0.247998 | 0.771679 |
| 1.0 | -0.966272 | -0.185226 | 1.792993 | -0.863291 | -0.010309 | 1.247203 | 0.237609 | 0.377436 | -1.387024 | ... | -0.108300 | 0.005274 |
| 2.0 | -1.158233 | 0.877737 | 1.548718 | 0.403034 | -0.407193 | 0.095921 | 0.592941 | -0.270533 | 0.817739 | ... | -0.009431 | 0.798278 |
| 2.0 | -0.425966 | 0.960523 | 1.141109 | -0.168252 | 0.420987 | -0.029728 | 0.476201 | 0.260314 | -0.568671 | ... | -0.208254 | -0.559825 |
| 4.0 | 1.229658 | 0.141004 | 0.045371 | 1.202613 | 0.191881 | 0.272708 | -0.005159 | 0.081213 | 0.464960 | ... | -0.167716 | -0.270710 |
| 7.0 | -0.644269 | 1.417964 | 1.074380 | -0.492199 | 0.948934 | 0.428118 | 1.120631 | -3.807864 | 0.615375 | ... | 1.943465 | -1.015455 |
| 7.0 | -0.894286 | 0.286157 | -0.113192 | -0.271526 | 2.669599 | 3.721818 | 0.370145 | 0.851084 | -0.392048 | ... | -0.073425 | -0.268092 |
| 9.0 | -0.338262 | 1.119593 | 1.044367 | -0.222187 | 0.499361 | -0.246761 | 0.651583 | 0.069539 | -0.736727 | ... | -0.246914 | -0.633753 |

Figure 5: Class Distribution Showing the Significant Contrast in Fraudulent and non-Fraudulent Transactions

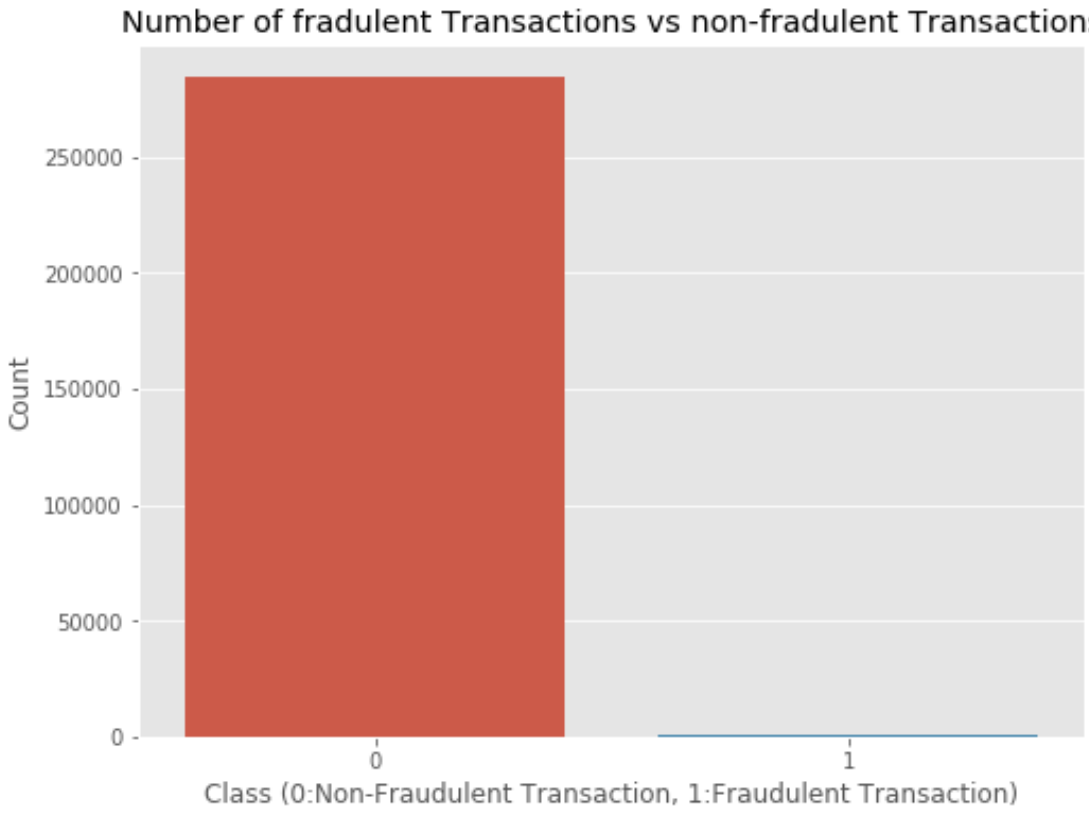
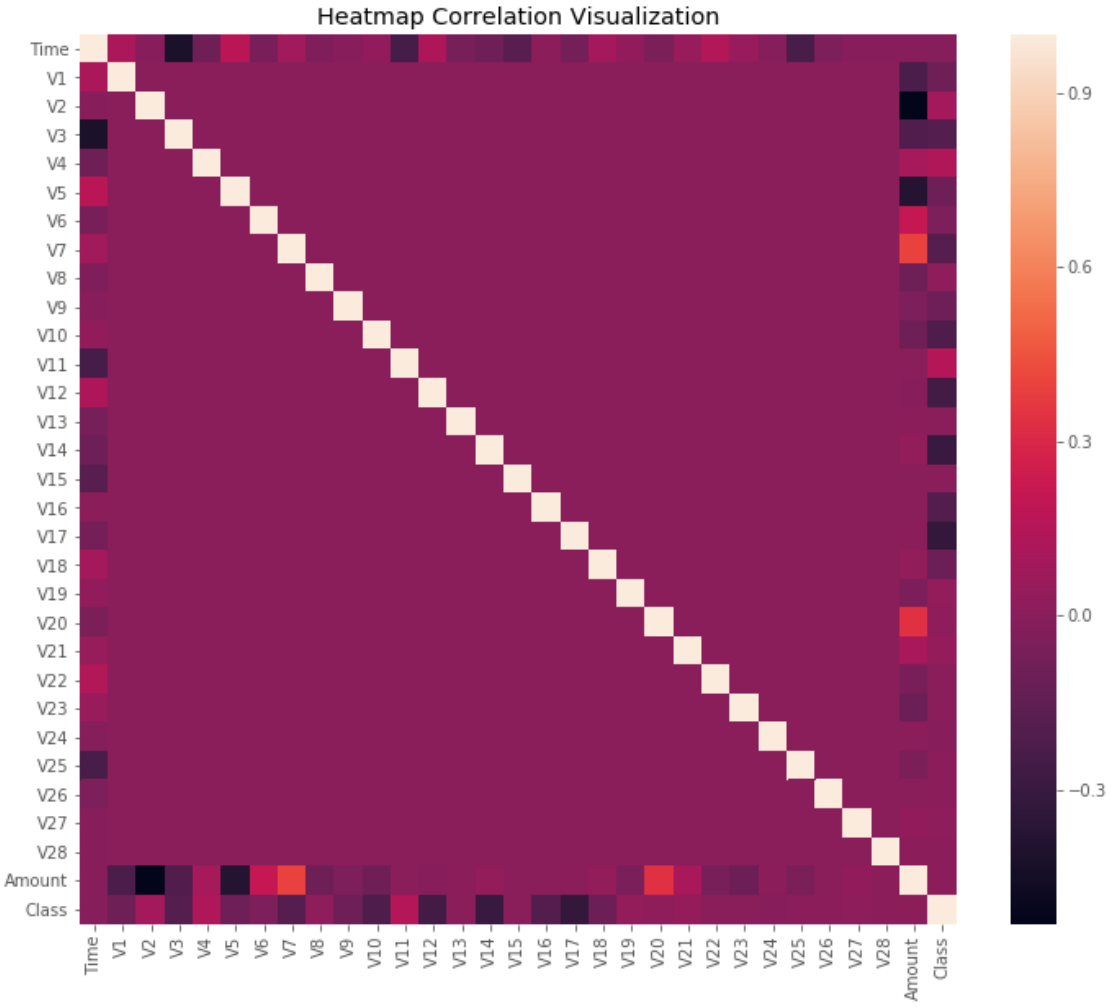


Figure 6: Heatmap Correlation Showing How Predictive Features are Correlated with Class Variable. The dark color shows strong correlation and the light color shows little or no correlation



I observed that the anonymized features were all scaled. The non-anonymized features were not scaled. Because I utilized a couple of machine learning algorithms, not normalizing the non-anonymized predictors could cause some algorithms that give weight to features (Logistic Regression) or other machine learning algorithms that rely on distance measure (KNN) to perform badly [2]. Also, normalizing the features will allow Decision Tree and Random Forest Classification algorithms to converge quickly. I used *sklearn StandardScaler* to standardize the Time and Amount columns in other to normalize the features

**Splitting the Original Dataset for Testing**
For the purpose of model testing and performance evaluation, I split the original dataset into two. Another reason for this is to avoid overfitting. For this project, testing the model with the original (Imbalanced) dataset will show how well the model performed. The model should predict the class features as accurate as possible. I could not rely on accuracy as the metric of evaluation because the testing had to be done on the original dataset

**Creating Training Set for an Imbalanced Data set**
In the course of understanding the dataset, it was discovered that over 99% of the transactions were non-fraudulent. Coming up with a training dataset that would allow an algorithm to detect a specific characteristics that makes a transaction more or less likely fraudulent is very challenging. An algorithm that always predicts that a transaction is non-fraudulent will achieve a very high accuracy but this was not the goal of this project. The goal of this project is to detect fraudulent transactions and label them as such.

One way of creating a training set for an imbalanced dataset is to utilize random under-sampling [5]. Under-sampling is a machine learning technique that is used to adjust the class distribution of a dataset. Utilizing random under-sampling will force the algorithms to detect fraudulent transactions and also achieve high accuracy.

For the training dataset, I counted all the fraudulent transactions in the dataset and randomly selected the same number of non-fraudulent transactions and concatenated both and shuffled the new dataset. Figure 7 shows the distribution of the balanced dataset after random under-sampling.

**Dimensionality Reduction for Visualization**
It is impossible to produce a 30-dimensional plot considering all the predictors on a 2-dimensional surface. Visualizing the classes would show how clearly separable the balanced features are. In order to achieve this visualization, I used PCA dimensionality reduction to project these higher dimensional distributions into a lower dimensional visualizations. Figure 8 shows a lower dimensional visualization of the predictors and how they are separable.

**Classification Algorithms**
After data exploration and preprocessing, the next step was to train the machine learning algorithm. In order to test the performance of the algorithm, I performed and 80/20 train-test split with *sklearn train_test_split* on the balanced data set. 80% is for training the machine learning algorithm while 20% is for model testing. I used k_fold cross validation resampling technique in order to avoid overfitting.
Overfitting is simply a problem in machine learning where a model tends to memorize the data, rather than learn from it. This makes it difficult for the model to make a prediction on an unseen dataset. Separating the dataset into k parts and then fitting the model on k-1 folds before making predictions for the kth hold-out is used to deal with the problem of overfitting.

For this project, I tested the performance of five machine learning algorithms with Receiver Operating Characteristics - Area Under the Curve or ROC-AUC metric. I couldn't rely on accuracy because of the

Figure 7: Class Distribution Showing Equal Number of Fraudulent and non-Fraudulent Transactions After Applying Random Under-Sampling Technique on the Imbalanced Dataset

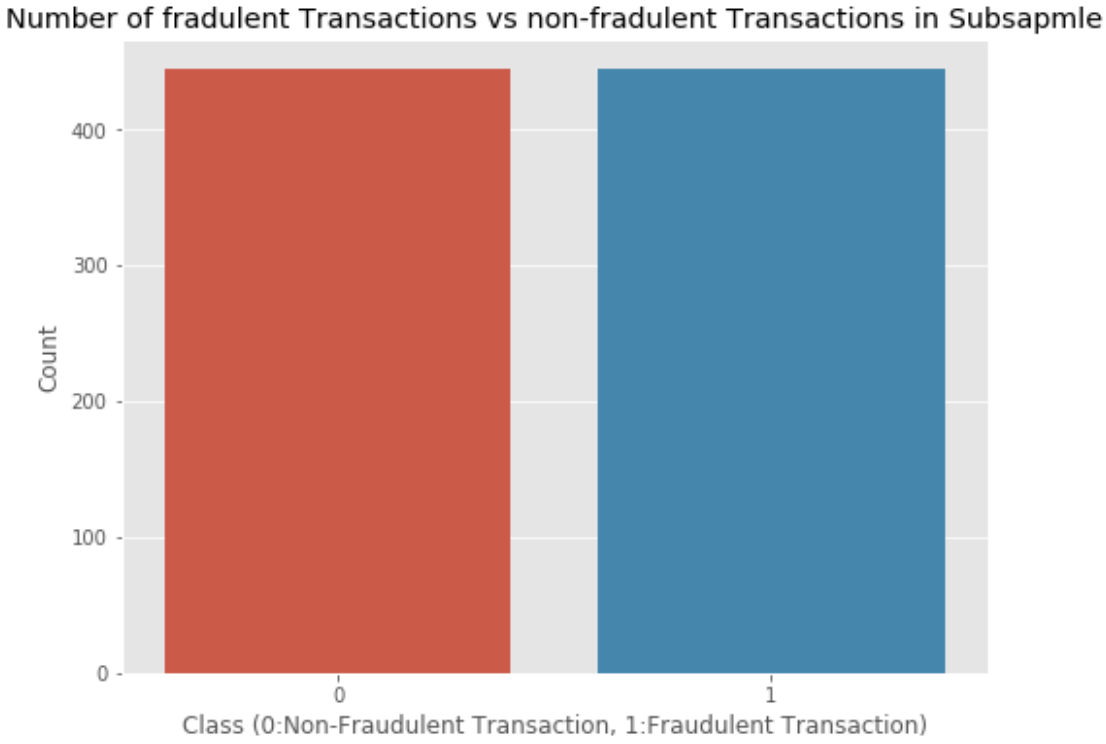



Figure 8: PCA Dimensionality Reduction Visualization Showing a 2-Dimensional Plot of a 30-Dimensional Dataset. Blue dots represent fraudulent transactions while the red dots represent non-fraudulent transactions

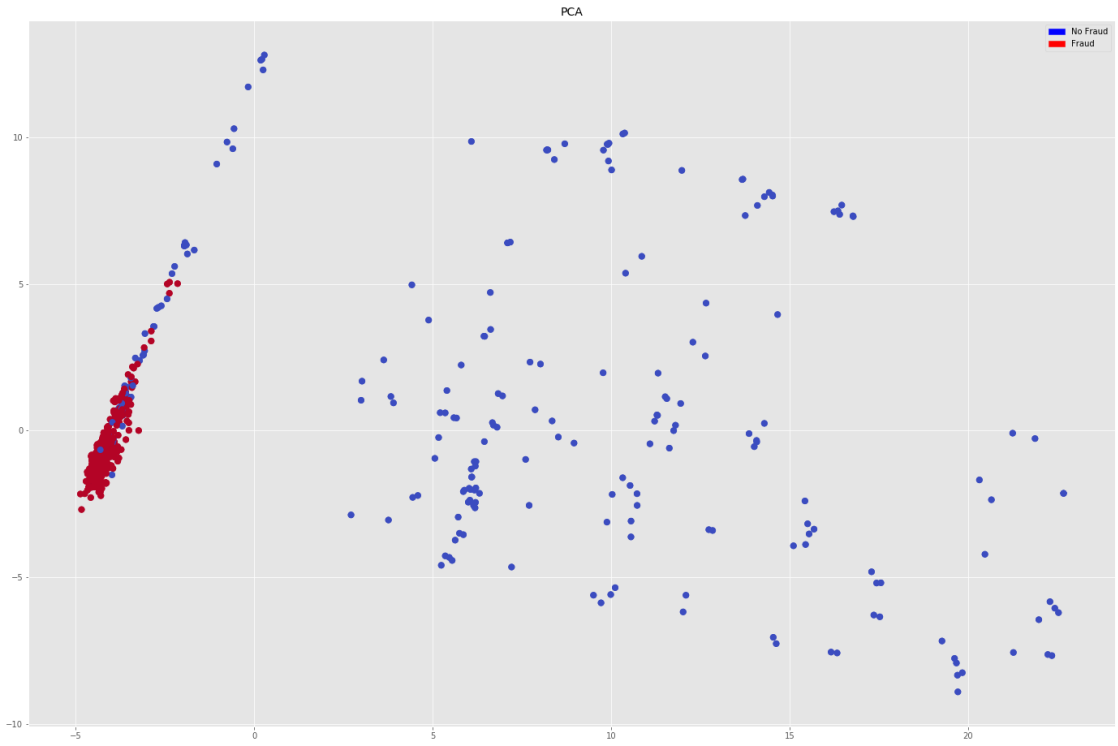


Figure 9: Classification Algorithms Evaluation Result

```
Classifiers:  LogisticRegression Has a training score of 98.0 % accuracy score
Classifiers:  RandomForestClassifier Has a training score of 95.0 % accuracy score
Classifiers:  DecisionTreeClassifier Has a training score of 92.0 % accuracy score
Classifiers:  SVC Has a training score of 97.0 % accuracy score
Classifiers:  GaussianNB Has a training score of 96.0 % accuracy score
```

imbalanced nature of the dataset. The ROC-AUC outputs a value between zero and one, one is a perfect score and zero is a worst score.

I used the following well-known classification algorithms:

- **Logistic Regression Classifier**
- **Naive Bayes Classifier**
- **Support Vector Machine**
- **Decision Tree**
- **Random Forest**

Figure 9 shows the result of the classification algorithms I considered. Decision Tree Classifier had the least score of 92.0% Logistic Regression had 98.0% followed by SVM classifier with 97.0% score, and Random Forest Classifier with 95.0% score. The goal of this project was not just to achieve a higher accuracy but also create business value. The nature of a dataset plays a major role when it comes to choosing a machine learning algorithm. Since this was a classification problem, I decided to utilized some well known machine learning algorithms. Considering the nature of this project, choosing an easy-to-explain algorithm was prioritized as well as performance. This is because decision makers are not often Data Analytics experts. In this case, I will propose Random Forest classifier because it is easy to explain the algorithm and the result.

Finally, the Random Forest Classifier was used for model prediction.

**Conclusion**

Understanding a business problem plays a huge role in any machine learning project. The credit card fraud detection project requires a substantial amount of planning before applying any machine learning algorithm. This project also provides some insight as it relates to what happens in the industry and also shows how data science and machine learning can be used for good.

**Future Work**

The first time I worked on this project, I did not consider anomaly detection and outlier removal. I started looking at features with negative and positive correlation. I plan on paying more attention to correlation and outlier detection. The reason I am paying more attention to this is to understand the most relevant predictive features.

**Other Sample Projects I worked On**

1. **Predicting Product Sales Through Ads Delivered on Social Networking Sites** [4] : In this project, I implemented a machine learning algorithm that tells whether a user of a social networking site, after clicking the ad's displayed on a website ends up buying a product or not. With this model, companies can determine the best crowd to direct their product advertisement to and know the category of people that are most likely to purchase a product based on the features which describe the type of users who had purchased the products previously by clicking on the ads.

2. **Implementing K-Nearest Neighbor Algorithm without using any Package** [3] : This was a homework assignment I completed when I took Machine Learning class in my final year Masters program. In this project, I implemented K-NN algorithm based on the intuition behind the algorithm.

# References

[1] Credit Card Fraud Detection. https://www.kaggle.com/mlg-ulb/creditcardfraud.

[2] Detecting Credit Card Fraud Using Machine Learning. https://towardsdatascience.com/detecting-credit-card-fraud-using-machine-learning-a3d83423d3b8.

[3] K-NN Implementation without Packakes. https://github.com/UdochukwuNweke/credit-card-fraud-detect/blob/master/AdditionalProjects/k-nn.ipynb.

[4] Social Network Ads Purchase Prediction. https://github.com/UdochukwuNweke/credit-card-fraud-detect/blob/master/AdditionalProjects/SocialNetworkAds.ipynb.

[5] Nathalie Japkowicz et al. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI workshop on learning from imbalanced data sets*, volume 68, pages 10–15. Menlo Park, CA, 2000.