Machine Learning Engineer Nanodegree

Capstone Project: Credit Card Anomaly and Fraud Detection

Abhijeet Koranne
December 31st, 2019

I. Definition

Project Overview

Financial fraud is ever growing menaces with far consequences in the financial industry. Payments are the most digitalized part of the financial industry, which makes them particularly vulnerable to digital fraudulent activities. Anomaly detection is one of the common antifraud approaches in data science. It provides simple binary answers. It is based on classifying all objects in the available data into two groups: normal distribution and outliers. Outliers, in this case, are the objects (e.g. transactions) that deviate from normal ones and are considered potentially fraudulent.

Problem Statement

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be fraud. This model is then used to identify whether a new transaction is fraudulent or not.

To solve this problem, a prediction model would be developed based on predefined evaluation metrics. Various algorithms like Logistic Regression, Decision Tree, Gaussian Naïve Bayes, Multi-Layer Perceptron, XGBoost and LightGBM would be compared on evaluation metrics for best performance on dataset.

Metrics

Usually performance of the model is evaluated based on confusion matrix with precision, Recall (sensitivity).

Precision = TP/ (TP+FP)

Precision gives the accuracy in cases classified as fraud (positive)

Sensitivity (or Recall) = TP/ (TP +FN)

Sensitivity (Recall) gives the accuracy on positive (fraud) cases classification.

But for given imbalanced class ratio in data set, I will measure accuracy using Area under Precision Recall Curve (AUPRC) score. The Area under Precision Recall Curve (AUPRC) is given by plotting Precision against Recall. A model with AUPRC score equal to one indicate perfect model.

II. Analysis

Data Exploration

The dataset is obtained from Machine Learning Group — ULB, Credit Card Fraud Detection (2018), Kaggle.

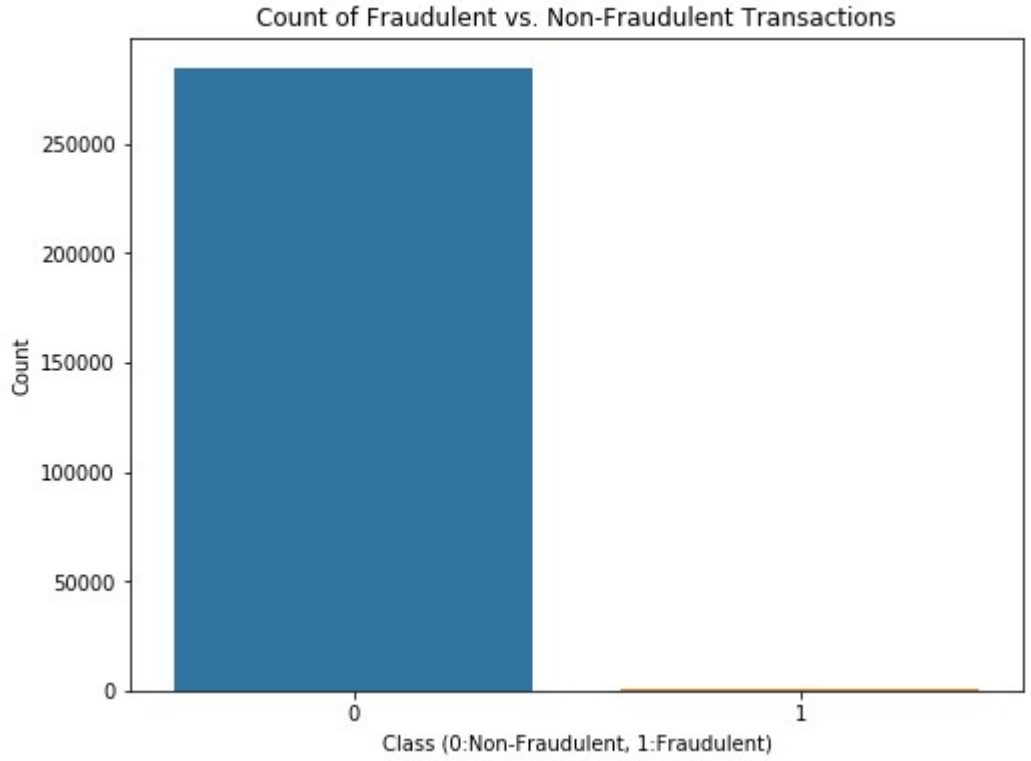
The datasets contain transactions made by credit cards in September 2013 by European cardholders. These transactions are a subset of all online transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, where the positive class (frauds) account for 0.172% of all transactions. It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2 ... V28 are the principal components obtained with PCA; the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

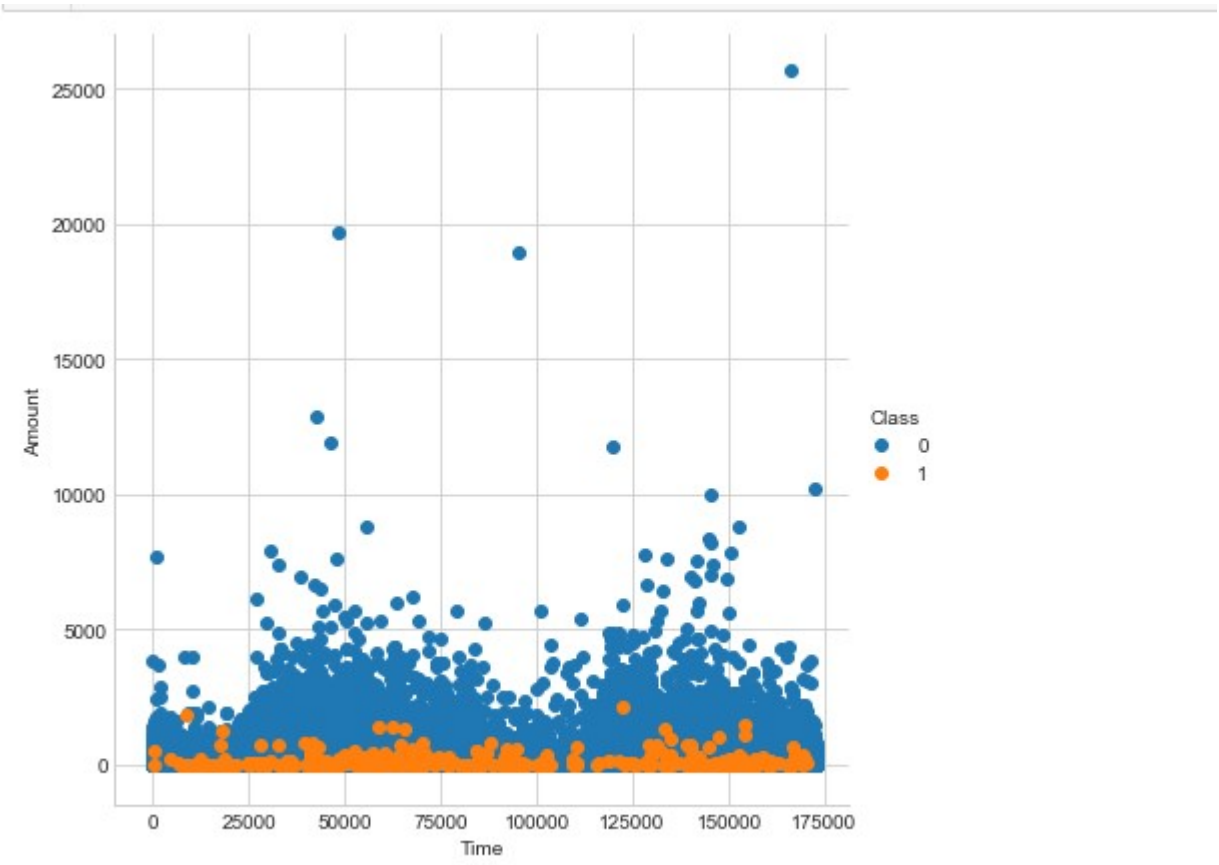
```
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
Time      284807 non-null float64
V1        284807 non-null float64
V2        284807 non-null float64
V3        284807 non-null float64
V4        284807 non-null float64
V5        284807 non-null float64
V6        284807 non-null float64
V7        284807 non-null float64
V8        284807 non-null float64
V9        284807 non-null float64
V10       284807 non-null float64
V11       284807 non-null float64
V12       284807 non-null float64
V13       284807 non-null float64
V14       284807 non-null float64
V15       284807 non-null float64
V16       284807 non-null float64
V17       284807 non-null float64
V18       284807 non-null float64
V19       284807 non-null float64
V20       284807 non-null float64
V21       284807 non-null float64
```

```
V22       284807 non-null float64
V23       284807 non-null float64
V24       284807 non-null float64
V25       284807 non-null float64
V26       284807 non-null float64
V27       284807 non-null float64
V28       284807 non-null float64
Amount    284807 non-null float64
Class     284807 non-null int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

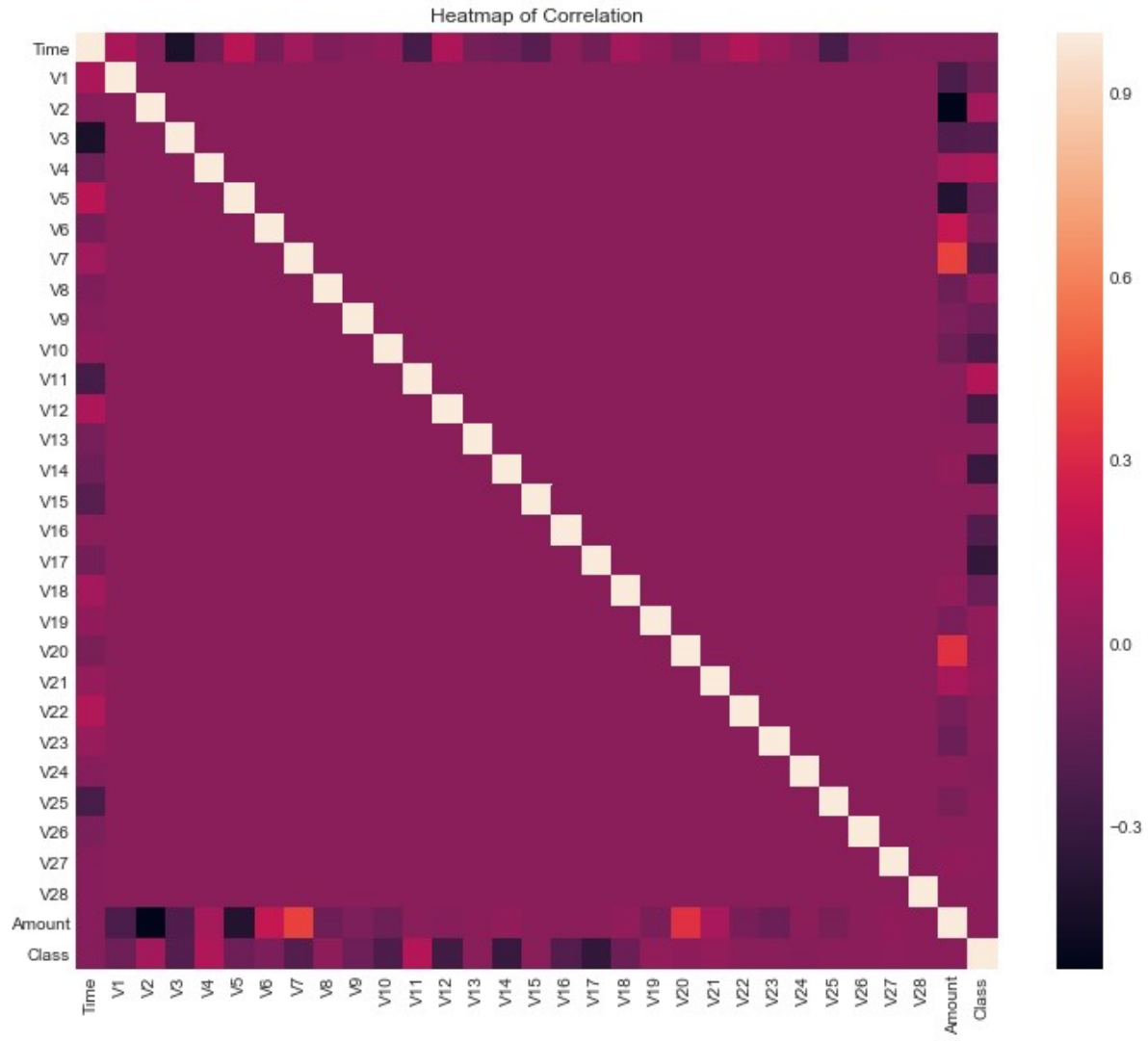
Exploratory Visualization

There were 284315 non-fraudulent transactions (99.827%) and 492 fraudulent transactions (0.173%) out off 284807. Here figure shows Imbalance in non fraudulent vs. fraudulent class.

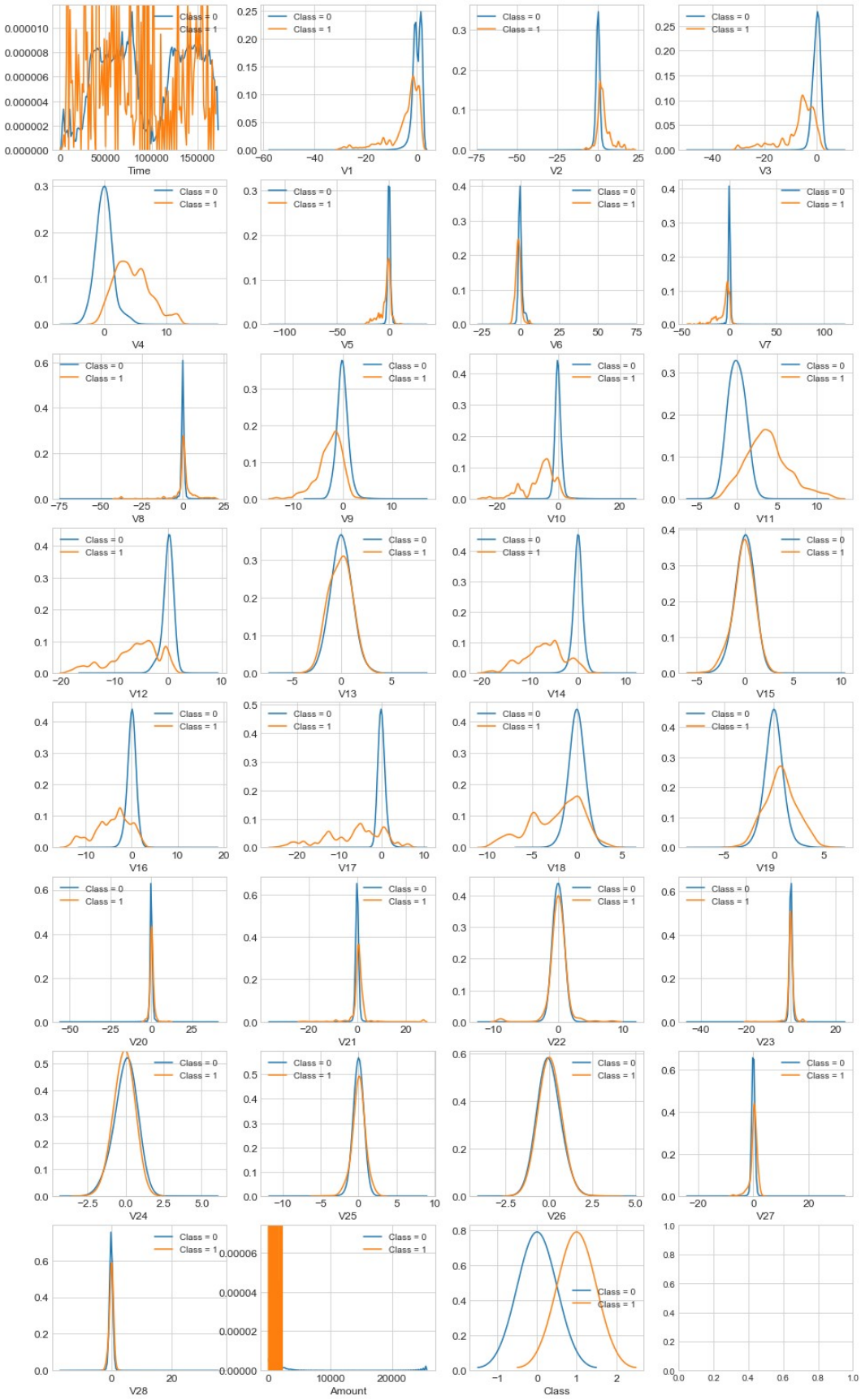




From the above plot it is clearly visible that, there are frauds only on the transactions which have transaction amount approximately less than 5000. The frauds in the transactions are evenly distributed throughout time.



According to above heat map of Correlation, there is no notable correlation between features V1-V28. There are certain correlations between some of these features and Time (inverse correlation with V3) and Amount (direct correlation with V7 and V20, inverse correlation with V1 and V5).


Class = 0
Class = 1
Time
V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16
V17
V18
V19
V20
V21
V22
V23
V24
V25
V26
V27
V28
Amount
Class

For some of the features we can observe a good selectivity in terms of distribution for the two values of Class: V4, V11 have clearly separated distributions for Class values 0 and 1, Class: V12, V14, V16, V17, V18 are partially separated, V1, V2, V3, V10 have a quite distinct profile, whilst V25, V26, V28 have similar profiles for the two values of Class.

In general, with just few exceptions (Time and Amount), the features distribution for legitimate transactions (values of Class = 0) is centered on 0, sometime with a long queue at one of the extremities. In the same time, the fraudulent transactions (values of Class = 1) have a skewed (asymmetric) distribution.

Algorithms and Techniques

As Credit card fraud detection problem is binary classification problem because of this I tried following Machine learning algorithms:-

a) Logistic Regression

b) Decision Tree

c) Gaussian Naïve Bayes

d) Multi-Layer Perceptron (MLP)

e) XGBoost (Extreme Gradient Boosting)

f) LightGBM (Light Gradient Boosting Decision Tree.)

Logistic Regression:
It is a statistical method for analyzing a data set in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes). The goal of logistic regression is to find the best fitting model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables.

Decision Trees:
Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

Multi-Layer Perceptron:
The Multi-Layer Perceptron (MLP) is a neural network, which consists of three or more layers (an input and an output layer with one or more *hidden layers*) of nonlinearly-activating nodes. Since MLPs are fully connected, each node in one layer connects with a certain weight to every node in the following layer.

Gaussian Naive Bayes:
It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

XGBoost (Extreme Gradient Boosting):

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. XGBoost dominates, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class for classification and regression predictive modeling problems.

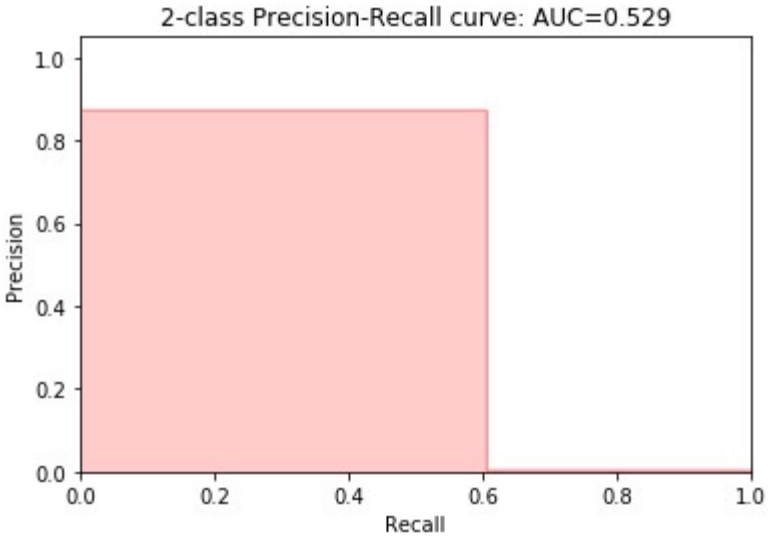
Light GBM (Light Gradient Boosting Decision Tree):

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks. Since it is based on decision tree algorithms, it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf-wise. So when growing on the same leaf in Light GBM, the leaf-wise algorithm can reduce more loss than the level-wise algorithm and hence results in much better accuracy which can rarely be achieved by any of the existing boosting algorithms. Also, it is surprisingly very fast, hence the word 'Light'.

Benchmark

Available dataset has 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, where the positive class (frauds) account for 0.172% of all transactions. Considering this imbalance, I will measure accuracy using Area under Precision Recall Curve (AUPRC) score. The Area under Precision Recall Curve (AUPRC) is given by plotting Precision against Recall.
For this project, I built a simple logistic regression classifier as benchmark model which gave me base value of

Precision = 0.8710
Recall = 0.6067
AUPRC Score = 0.5291



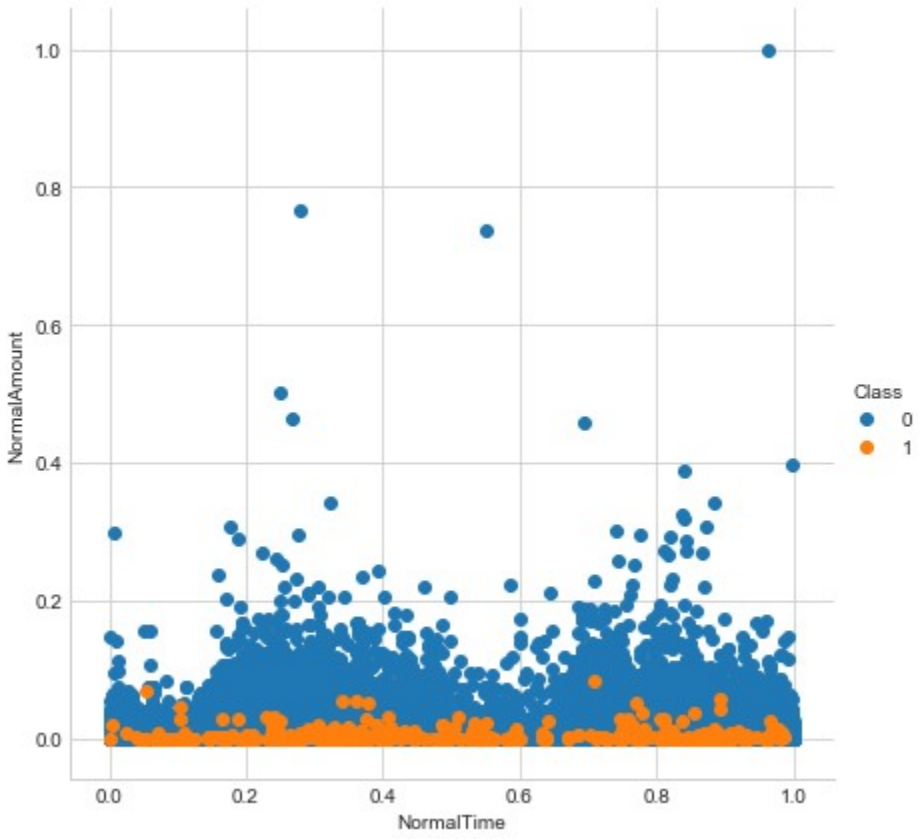
As this base value is very low, I will try to tune model to get an Area under the Precision-Recall Curve (AUPRC) score of 74% or greater.

III. Methodology

Data Preprocessing

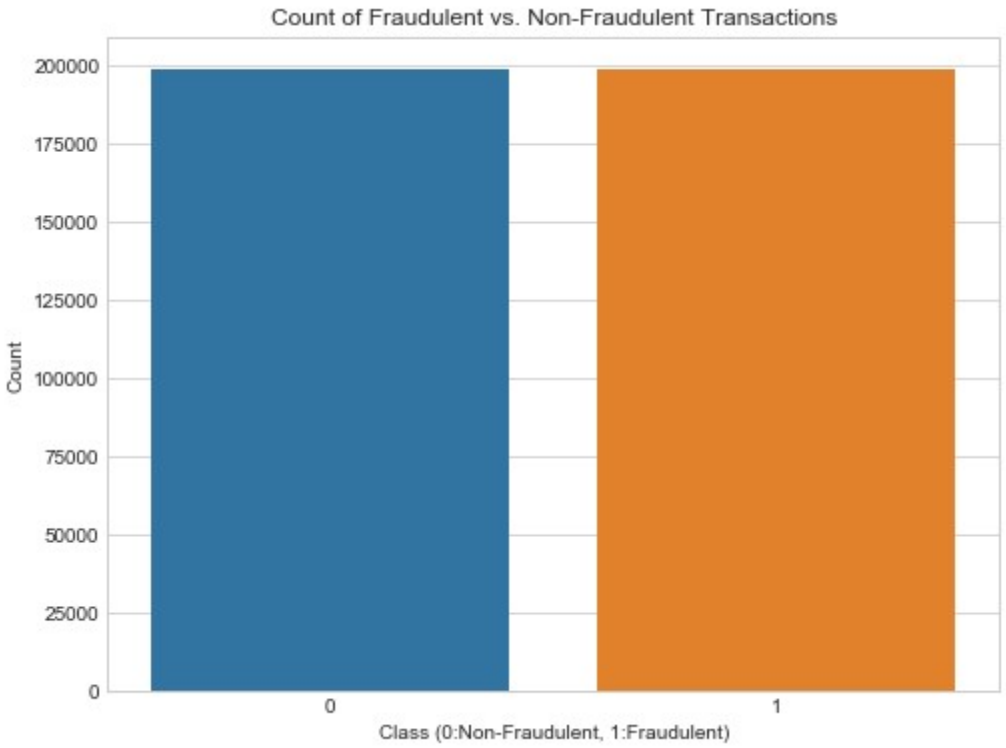
First of all in data preprocessing, I checked for null or empty values but dataset does not contain any null value .After that, I performed normalization of time and amount feature. Except these two feature, rest all features were obtained by PCA so no need to normalize them.

Here figure shows after applying normalization on time and amount feature on dataset.



In next step, as dataset is highly imbalanced and contains very less fraudulent transactions (0.173%).So I used Over sampling algorithm called SMOTE to balance the class ratio.

Here figure shows after applying SMOTE on dataset



Implementation

The implementation stage involved creating training and predicting pipeline. This stage involved testing six algorithms to see which best suits the problem. The following algorithms were used:-

a) Logistic Regression

b) Decision Tree

c) Gaussian Naïve Bayes

d) Multi-Layer Perceptron (MLP)

e) XGBoost (Extreme Gradient Boosting)

f) LightGBM (Light Gradient Boosting Decision Tree.)

All these binary classification algorithms were initialized. Afterwards the training dataset was broken into four different portions, 1%, 10%, 50% and 100%. The purpose of breaking the dataset into different portions was to track the performance of each algorithm as the training set size increased.

The following metrics were used to measure performance as the data set scaled;

a) Area under the Precision Recall Curve (AUPRC) on the train and cross validation set.
b) Recall score on the train and cross validation set.
c) Time taken to train the model.
d) Time taken to make predictions on the train and cross validation set.

Figure Showing Performance of all six algorithms and how they scaled across different portions of the dataset.

Though, all six algorithms performed well. But I choose LGBM Classifier as LGBM Classifier out performed between all other algorithms. There are many advantages of choosing LGBM as below:-

- Faster training speed and higher efficiency
- Lower memory usage
- Better accuracy than any other boosting algorithm
- Compatibility with Large Datasets

Refinement

After the LGBM algorithm was chosen, parameters tuning were performed to optimize the model. Grid Search was used to find the optimal parameters for the “num_leaves'” ,“min_data_in_leaf” ,” lambda_l1”and “lambda_l2” values.

Afterwards both the Non-optimized model returned an AUPRC of 0.9996 and optimized model returned an AUPRC of 0.9997. This implies that the Area under Precision Recall Curve of both models were very good scores

IV. Results

Model Evaluation and Validation

During the optimization/refinement stage of the project, the Logistic Regression algorithm was tested on a cross validation set and got a cross validation score of 0.9997 with AUPRC as the evaluation metric.
The parameters of the final model were;

a) num_leaves: 50,
b) min_data_in_leaf: 100,
c) lambda_l1: 1,
d) lambda_l2: 1

To verify the robustness of the model, the model was used to predict fraudulent transactions in the test dataset. The result from this prediction on the test set was an AUPRC of 0.9998 and a Recall Score of 1.

A recall score of 1 implies that the final model predicted 100% of the fraudulent transactions in the test dataset correctly while the precision measures that fraction of cases predicted to be fraudulent that are truly fraudulent.

Justification

Comparing the final model (AUPRC-0.9997) with the benchmark model (AUPRC- 0.5291), the final model did outperform the benchmark with an increase of 52.9% in the AUPRC of the final model.

Though In the global context of credit card fraud detection this model might not be a silver bullet but it is definitely a great starting point to a larger project that seeks to tackle credit card fraud detection globally.

V. Conclusion

Free-Form Visualization

Figure showing AUPRC Score of Benchmark Model

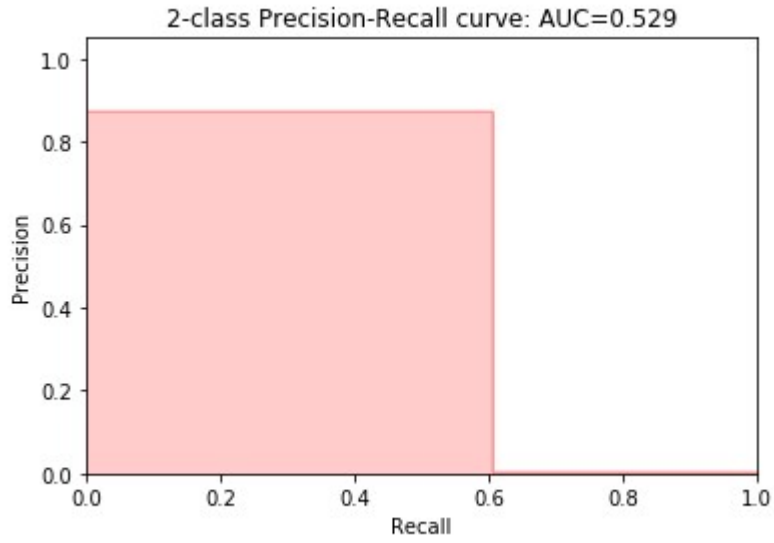


Figure showing AUPRC Score of Final Model

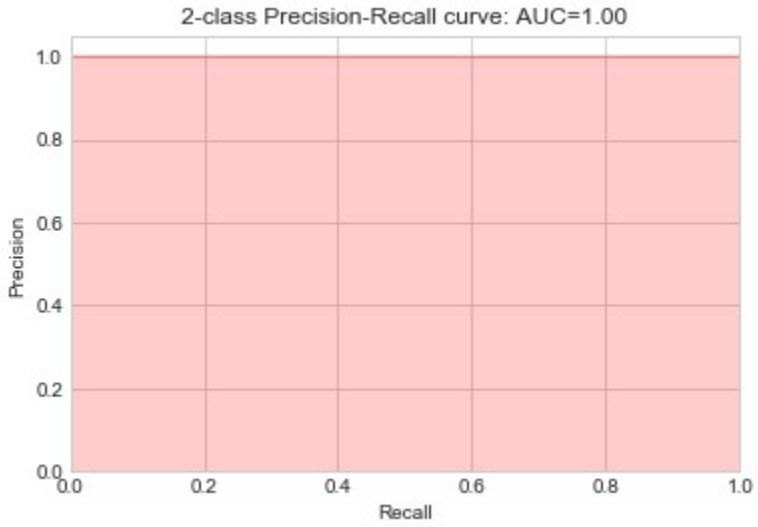
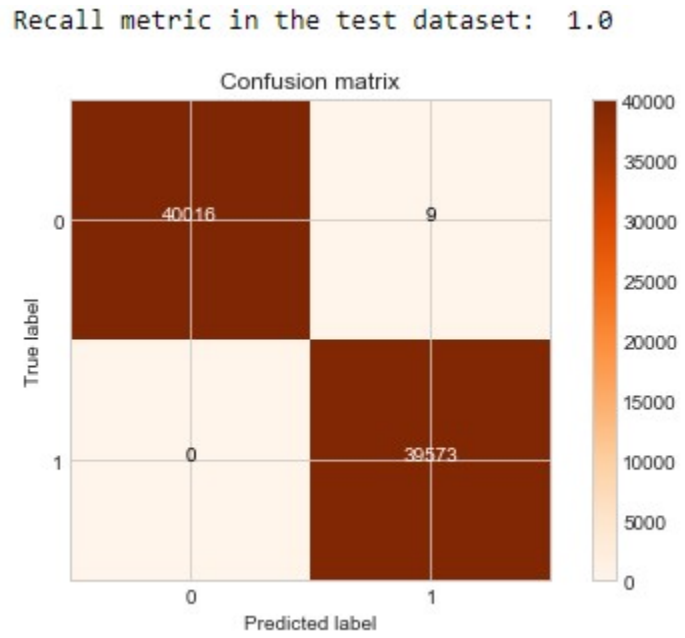


Figure showing Confusion Metrics for final model



The final model predicted 40016 fraudulent transactions correctly, 9 fraudulent transactions incorrectly, 39573 genuine transactions correctly and 0 genuine transactions incorrectly.

Reflection

This project can be divided into following parts:-

a) Getting the dataset from Kaggle was the easiest part
b) Exploratory data analysis part was also not so difficult.
c) Model Selection part was time consuming and difficult part.
d) The most difficult phase for me was picking the right algorithm to use in building and tuning parameters for the final model.

Improvement

The dataset is highly unbalanced, where the positive class (frauds) account for 0.172% of all transactions.(492 frauds out of 284,807 transactions).If more fraudulent data would be available ,Then I need not required to use oversampling algorithm SMOTE .

Dataset contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, original features and more background information about the data was not available. Due to which significance of features were not known. If it would be provided then only solution of this real world problem could be provided.

References:

i. Machine Learning Group — ULB, Credit Card Fraud Detection (2018),Kaggle

ii. Nathalie Japkowicz, Learning from Imbalanced Data Sets: A Comparison of Various Strategies (2000), AAAI Technical Report WS-00–05

iii. Types of classification algorithms in Machine Learning

iv. Which algorithm takes the crown: Light GBM vs XGBOOST?