Data Science Engineering Project

Northeastern University

Summer 2019

Xuhang Liu - 001434973

Project Title: Credit Card Fraud Detection

**Problem Description:**

Banks, merchants and credit card processors companies lose billions of dollars every year to credit card fraud. Credit card data can be stolen by criminals but sometimes the criminal is simply the clerk that processes your card when you buy things.

The latest Nilson report estimates that in 2016, worldwide credit card losses topped $24.71 billion. Barclays reports that **47%** of all credit card fraud occurs in the United States. It is important that credit card companies should able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase. This can be achieved with help of **machine learning models**. In December 2013, the Abington Police arrested two Post Office employees for stealing credit cards and using it to buy more than $50,000 worth of merchandise. It took police forces a few months before identifying the criminals.

Analyzing fraudulent transactions manually is unfeasible due to huge amounts of data and its complexity. However, given sufficiently informative features, one could expect it is possible to do using Machine Learning. This hypothesis will be explored in the project.

**Dataset :**

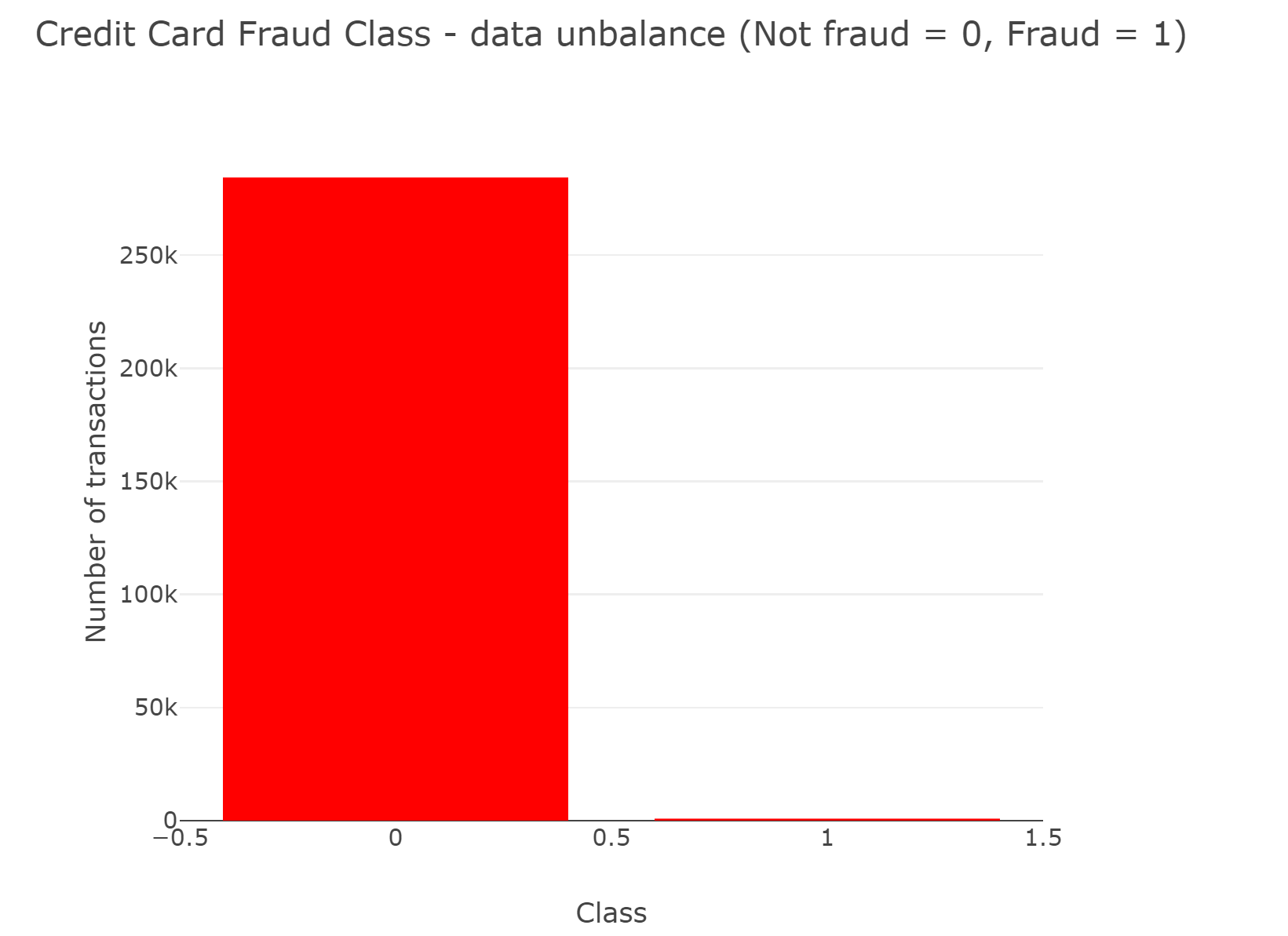
The datasets contain transactions made by credit cards in **September 2013** by European cardholders. This dataset presents transactions that occurred in two days.

It contains only numerical input variables which are the result of a **PCA transformation**.

Due to confidentiality issues, there are not provided the original features and more background information about the data.

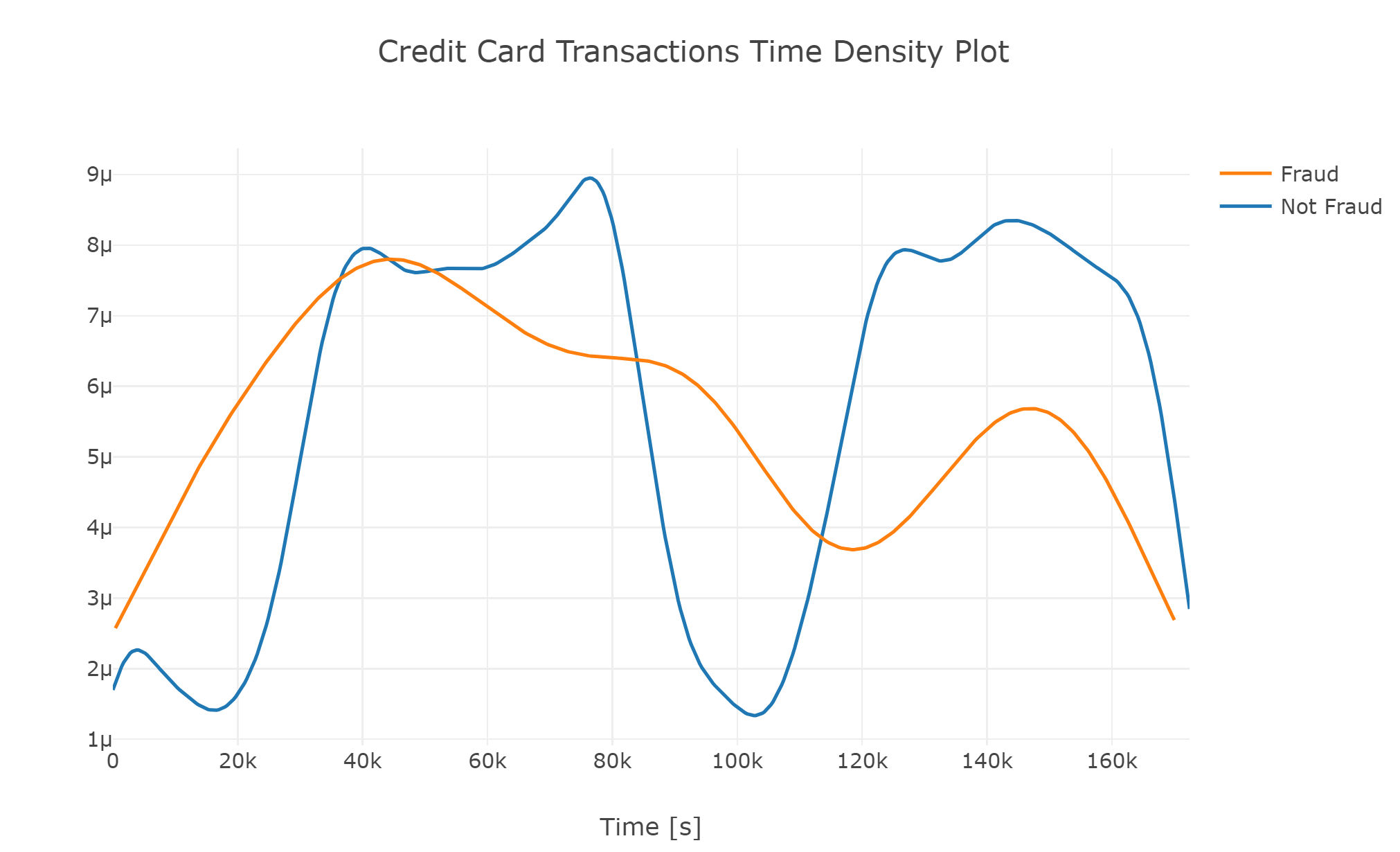
* Features V1, V2, ... V28 are the **principal components** obtained with PCA;
* The only features which have not been transformed with PCA are **Time** and **Amount**. Feature **Time** contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature **Amount** is the transaction Amount, this feature can be used for example-dependent cost-sensitive learning.
* Feature **Class** is the response variable and it takes value 1 in case of fraud and 0 otherwise.

The figure below describes the distribution of classes in the dataset. It is evident that the dataset is highly unbalanced as it contains only **0.172%** of fraudulent data and rest all are non-fraudulent data, i.e. there are 492 frauds out of 284,807 transactions in the given dataset.



Time Distribution:

The graph below shows us that **Fraudulent** transactions have a distribution more even than valid transactions. The valid transactions are equally distributed in time, including the low real transaction times, during night in Europe time zone.



**Methodology**:

Random Forest Classifier

Random forest classifier creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object. Basic parameters to Random Forest Classifier can be total number of trees to be generated and decision tree related parameters like minimum split, split criteria etc.

Gradient Boosting Classifier

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. It can benefit from regularization methods that penalize various parts of the algorithm and generally improve the performance of the algorithm by reducing overfitting.

Logistics Regression

We began by creating the model with sklearn.model\_selection.GridSearchCV. This method was chosen because it determined the parameter that give the best predictive score for the model.

This produced a cross validation score of 94.28% with the train datasets.

We then plotted the Learning Curves for both train and test and saw that training score decreased as that of test increased. The train ROC AUC curve score was 97.8.

SVM

Our SVM was built using the default SVM Classifier with kernel. – Linear, Radial Basis Function. We then trained it with our balanced dataset. We the made predictions for the train dataset which were then used to plot a confusion matrix. This predicted 106 of the total 115 Fraud classes in the dataset.

DNN (Deep Neural Network Classifier)

DNN finds the correct mathematical manipulation to turn the input into the output, whether it be a linear relationship or a non-linear relationship.

KNN (KNeighborsClassifier)

It can be implemented simply.

**Moreover, we introduce example-dependent cost-sensitive learning.**

BMR

The BMR classifier is a decision model based on quantifying tradeoffs between various decisions using probabilities and the costs that accompany such decisions. It's example dependent.

CostSensitiveRandomForestClassifier

A new cost-based impurity measure taking into account the costs when all the examples in a leaf. Introduce cost during training.

**Results:**

F1 Score, Precision, Recall, Accuracy

These metrics just show us a clearer comparison among the performances of different models. It’s important to pay attention to recall of fraud because we would rather regard not fraud transaction as fraud than let those fraud records go.

Regarding f1 Score , precision, recall, accuracy: Randomforest and DNN are better models.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | F1 Score | Precision | Recall | Accuracy |
| Randomforest | 0.936441 | 0.932127 | 0.971698 | 0.895652 |
| DNN | 0.940678 | 0.936364 | 0.980952 | 0.895652 |

ROC AUC

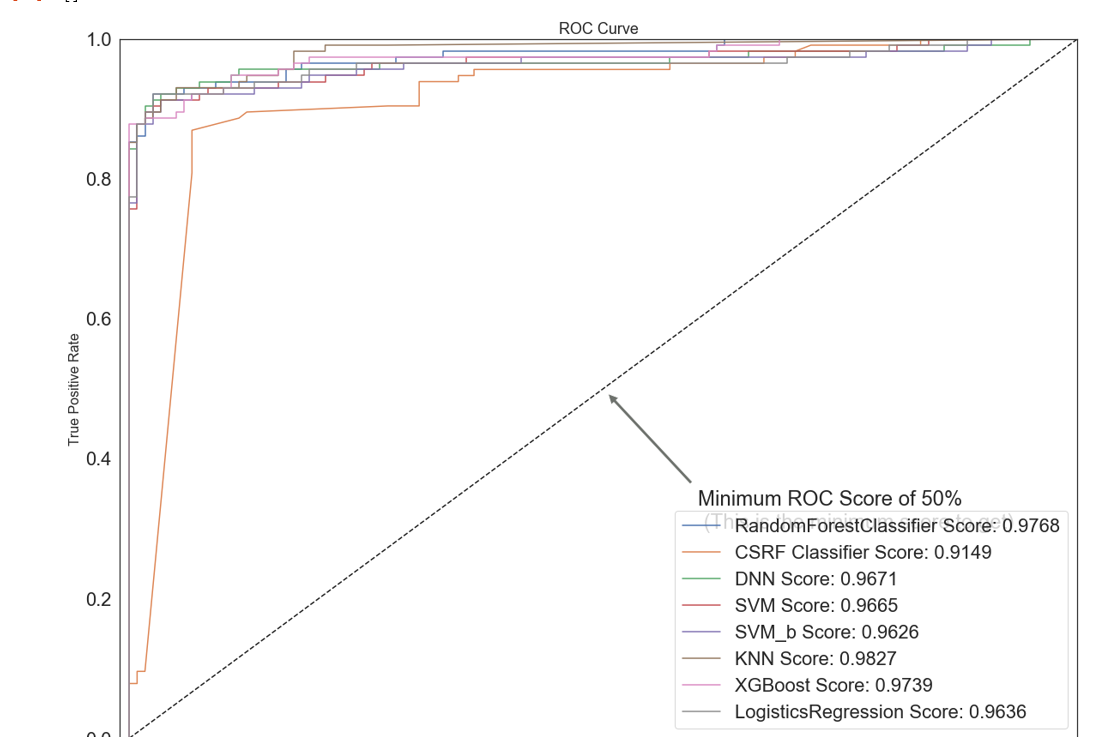
A receiver operating characteristic curve, or **ROC** curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The **ROC** curve is created by plotting the true positive rate (**TPR**) against the false positive rate (**FPR**) at various threshold settings. The true-positive rate is also known as **sensitivity** and the true-negative rate is also knows as **specificity**.

Area Under Curve **AUC** represents the probability that a random positive example is positioned to the right of a random negative example. AUC ranges in value from 0 to 1. A model whose predictions are **100% wrong** has an AUC of **0.0**; one whose predictions are **100% correct** has an AUC of **1.0.**

Why use it:

We think it’s the better metric since we got balanced dataset after using undersampling.

Regarding Area under the ROC Curve: RandomForest (0.9768), KNN (0.9827) and XGBoost (0.9739) are the better models which got roc curve scores above 0.97.



**In conclusion, with all the models in comparison, Randomforest classifier seems to be the best model regarding all of the metrics we used with score of 0.9768.**