Credit Card Fraud Detection

Abstract

Now a day's online transactions have become an important and necessary part of our lives. It is vital that credit card companies can identify fraudulent credit card transactions so that customers are not charged for items that they did not purchase. As frequency of transactions is increasing, number of fraudulent transactions are also increasing rapidly. Such problems can be tackled with Machine Learning with its algorithms. This project intends to illustrate the modelling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection Problem includes modelling past credit card transactions with the data of the ones that turned out to be fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications. Credit Card Fraud Detection is a typical sample of classification. In this process, we have focused on analyzing and pre-processing data sets as well as the deployment of multiple anomaly detection algorithms such as Local Outlier Factor and Isolation Forest algorithm on the PCA transformed Credit Card Transaction data.

Introduction

Credit Card Fraud can be defined as a case where a person uses someone else's credit card for personal reasons while the owner and the card issuing authorities are unaware of the fact that the card is being used.

Due to rise and acceleration of E- Commerce, there has been a tremendous use of credit cards for online shopping which led to High amount of frauds related to credit cards. In the era of digitalization, the need to identify credit card frauds is necessary. Fraud detection involves monitoring and analyzing the behavior of various users in order to estimate detect or avoid undesirable behavior. To identify credit card fraud detection effectively, we need to understand the various technologies, algorithms and types involved in detecting credit card frauds. Algorithm can differentiate transactions which are fraudulent or not. Find fraud, they need to passed dataset and knowledge of fraudulent transaction. They analyze the dataset and classify all transactions.

Fraud detection involves monitoring the activities of populations of users in order to estimate, perceive or avoid objectionable behavior, which consist of fraud, intrusion, and defaulting.

Machine learning algorithms are employed to analyses all the authorized transactions and report the suspicious ones. These reports are investigated by professionals who contact the cardholders to confirm if the transaction was genuine or fraudulent.

The investigators provide a feedback to the automated system which is used to train and update the algorithm to eventually improve the fraud-detection performance over time.

to rise and acceleration of E- Commerce, there has been a tremendous use of credit cards for online shopping which led to High amount of frauds related to credit cards. In the era of

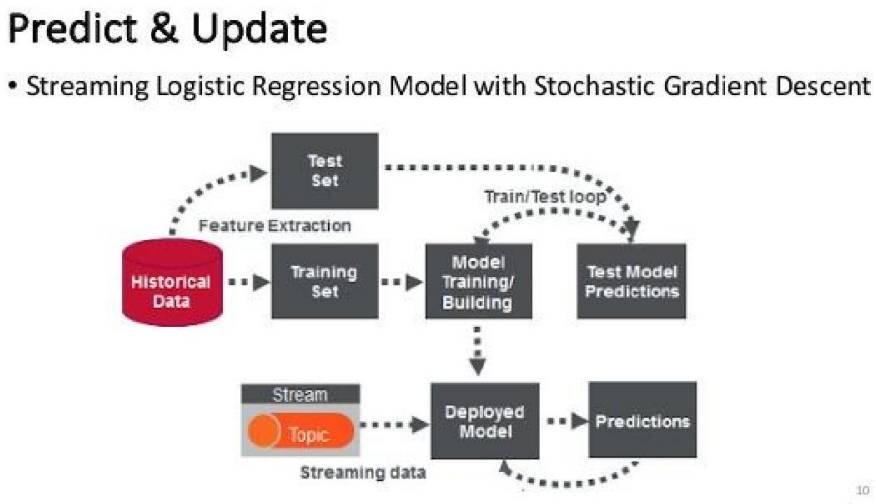
digitalization, the need to identify credit card frauds is necessary. Fraud detection involves monitoring and analyzing the behavior of various users in order to estimate detect or avoid undesirable behavior. In order to identify credit card fraud detection effectively, we need to understand the various technologies, algorithms and types involved in detecting credit card frauds. Algorithm can differentiate transactions which are fraudulent or not. Find fraud, they need to passed dataset and knowledge of fraudulent transaction. They analyze the dataset and classify all transactions.

Fraud detection involves monitoring the activities of populations of users in order to estimate, perceive or avoid objectionable behavior, which consist of fraud, intrusion, and defaulting.

Machine learning algorithms are employed to analyses all the authorized transactions and report the suspicious ones. These reports are investigated by professionals who contact the cardholders to confirm if the transaction was genuine or fraudulent.

The investigators provide a feedback to the automated system which is used to train and update the algorithm to eventually improve the fraud-detection performance over time.

Architecture



Implementation

The datasets contain transactions made by 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 is highly unbalanced, 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-dependent cost-sensitive learning. Feature 'Class' is the response variable, and it takes value 1 in case of fraud and 0 otherwise.

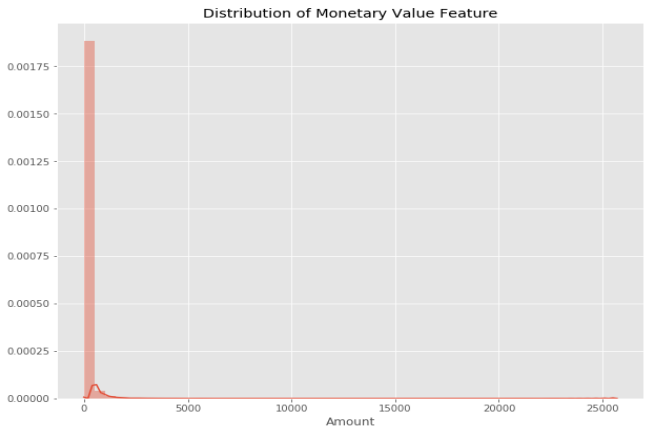
Machine Learning-Based Approaches
Below is a brief overview of popular machine learning-based techniques for anomaly detection.

Density-Based Anomaly Detection
Density-based anomaly detection is based on the k-nearest neighbors' algorithm.
Assumption: Normal data points occur around a dense neighborhood and abnormalities are far away.
The nearest set of data points are evaluated using a score, which could be Euclidian distance or a similar measure dependent on the type of the data (categorical or numerical). They could be broadly classified into two algorithms: ***K-nearest neighbor***(k-NN) is a simple, non-parametric lazy learning technique used to classify data based on similarities in distance metrics such as Euclidian, Manhattan, Minkowski, or Hamming distance.
Relative density of data: This is better known as local outlier factor (LOF). This concept is based on a distance metric called reachability distance.

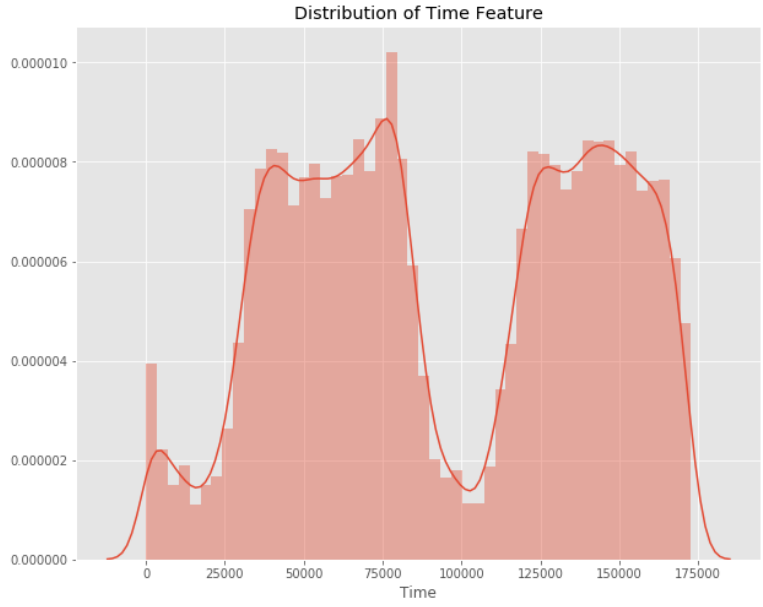
Clustering-Based Anomaly Detection
Clustering is one of the most popular concepts in the domain of unsupervised learning.
Assumption: Data points that are similar tend to belong to similar groups or clusters, as determined by their distance from local centroids.
K-means is a widely used clustering algorithm. It creates 'k' similar clusters of data points. Data instances that fall outside of these groups could potentially be marked as anomalies.

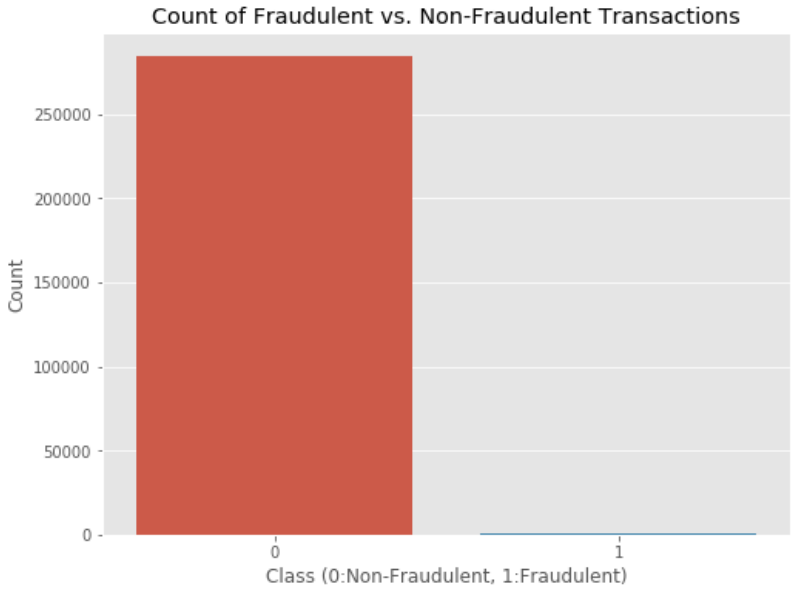
Support Vector Machine-Based Anomaly Detection
A support vector machine is another effective technique for detecting anomalies.
A SVM is typically associated with supervised learning, but there are extensions (OneClassCVM, for instance) that can be used to identify anomalies as an unsupervised problem (in which training data are not labeled).
The algorithm learns a soft boundary to cluster the normal data instances using the training set, and then, using the testing instance, it tunes itself to identify the abnormalities that fall outside the learned region.
Depending on the use case, the output of an anomaly detector could be numeric scalar values for filtering on domain-specific thresholds or textual labels (such as binary/multi labels).
In this jupyter notebook we are going to take the credit card fraud detection as the case study for understanding this concept in detail using the following Anomaly Detection Techniques namely

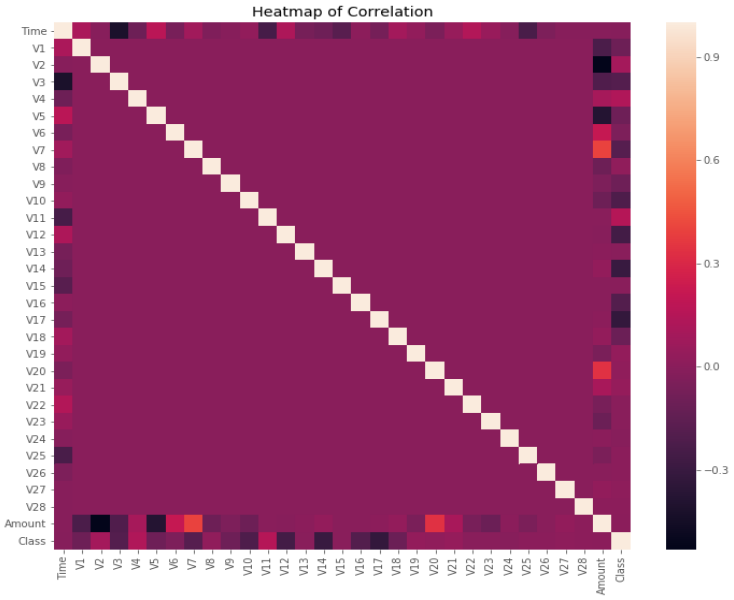
EDA



The time is recorded in the number of seconds since the first transaction in the data set. Therefore, we can conclude that this data set includes all transactions recorded over the course of two days. As opposed to the distribution of the monetary value of the transactions, it is bimodal. This indicates that approximately 28 hours after the first transaction there was a significant drop in the volume of transactions. While the time of the first transaction is not provided, it would be reasonable to assume that the drop-in volume occurred during the night.







As you can see, some of our predictors do seem to be correlated with the class variable. Nonetheless, there seem to be relatively little significant correlations for such a big number of variables. This can probably be attributed to two factors:

- The data was prepared using a PCA, therefore our predictors are principal components.
- The huge class imbalance might distort the importance of certain correlations with regards to our class variable.

Data Preparation

Before continuing with our analysis, it is important not to forget that while the anonymized features have been scaled and seem to be centered around zero, our time and amount features have not. Not scaling them as well would result in certain machine learning algorithms that give weights to features (logistic regression) or rely on a distance measure (KNN) performing much worse. To avoid this issue, I standardized both the time and amount column. Luckily, there are no missing values and we, therefore, do not need to worry about missing value imputation.

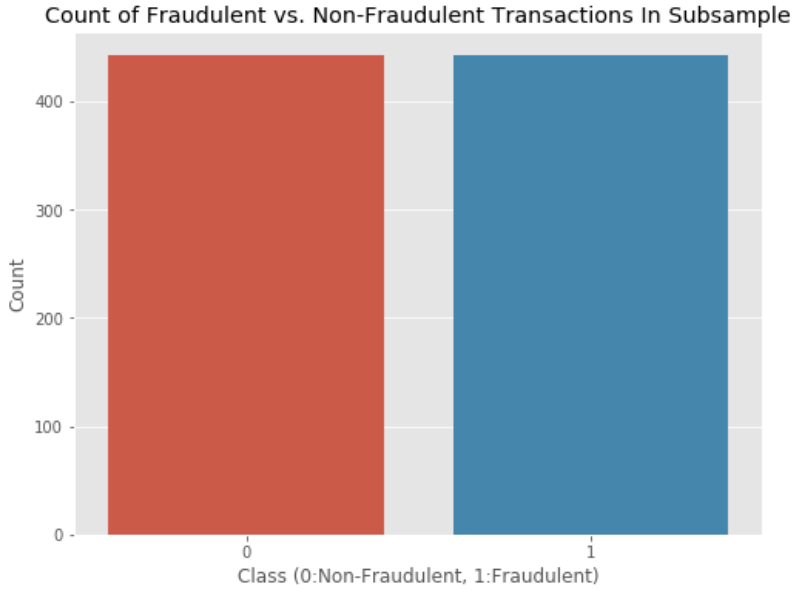
Creating a Training Set for a Heavily Imbalanced Data Set

Now comes the challenging part: Creating a training data set that will allow our algorithms to pick up the specific characteristics that make a transaction more or less likely to be fraudulent. Using the original data set would not prove to be a good idea for a very simple reason: Since over 99% of our transactions are non- fraudulent, an algorithm that always predicts that the transaction is non- fraudulent would achieve an accuracy higher than 99%. Nevertheless, that is the opposite of what we want. We do not want a 99% accuracy that is achieved by never labeling a transaction as fraudulent, we want to detect fraudulent transactions and label them as such.

There are two key points to focus on to help us solve this. First, we are going to utilize **random under-sampling** to create a training dataset with a balanced class distribution that will force the algorithms to detect fraudulent transactions as such to achieve high performance. Speaking of performance, we are not going to rely on accuracy. Instead, we are going to make use of the Receiver Operating Characteristics-Area Under the Curve or ROC-

AUC performance measure (I have linked further reading below this article). Essentially, the ROC-AUC outputs a value between zero and one, whereby one is a perfect score and zero the worst. If an algorithm has a ROC-AUC score of above 0.5, it is achieving a higher performance than random guessing.

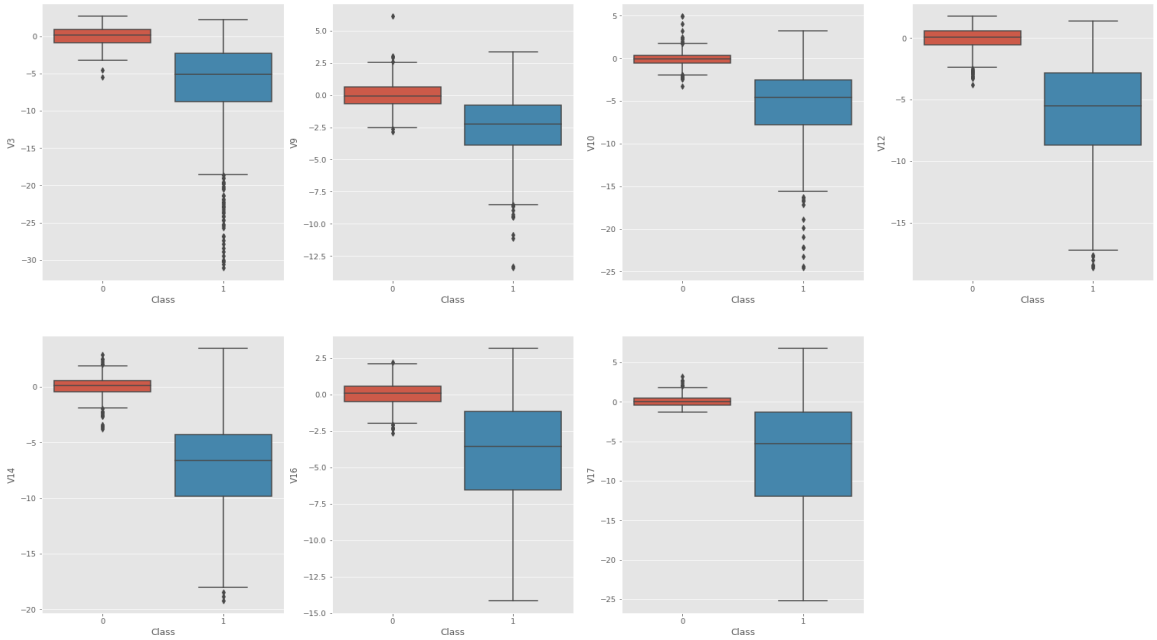
To create our balanced training data set, I took all of the fraudulent transactions in our data set and counted them. Then, I randomly selected the same number of non-fraudulent transactions and concatenated the two. After shuffling this newly created data set, I decided to output the class distributions once more to visualize the difference.



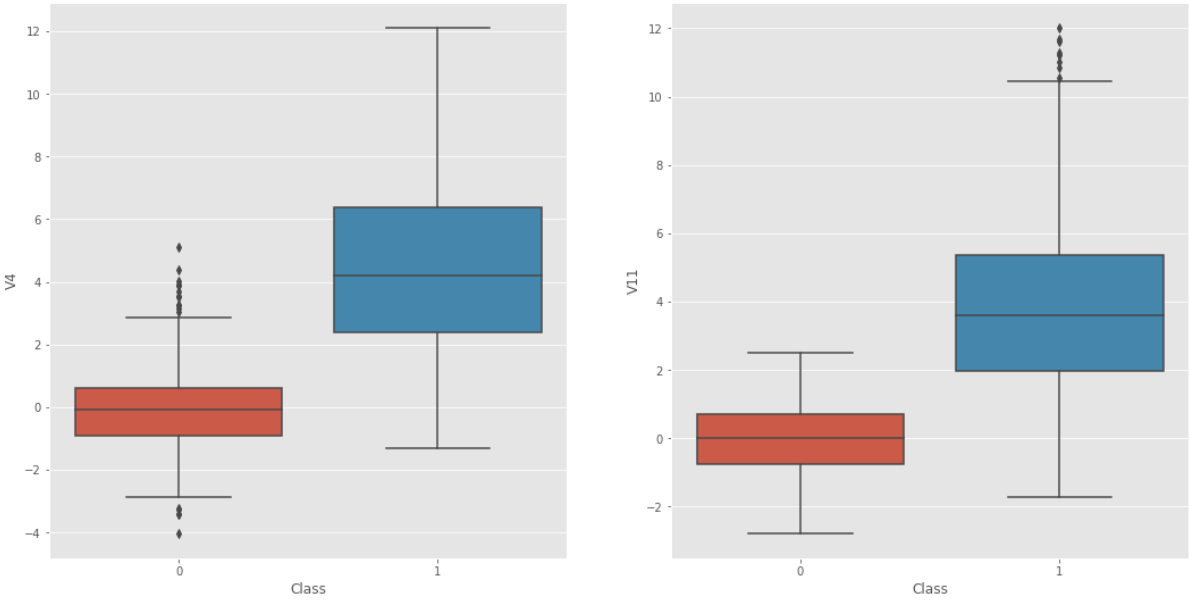
Outlier Detection and Removal

Outlier detection is a complex topic. The trade-off between reducing the number of transactions and thus volume of information available to my algorithms and having extreme outliers skew the results of your predictions is not easily solvable and highly depends on your data and goals. In my case, I decided to focus exclusively on features with a correlation of 0.5 or higher with the class variable for outlier removal. Before getting into the actual outlier removal, let's take a look at visualizations of those features:

Features With High Negative Correlation



Features With High Positive Correlation

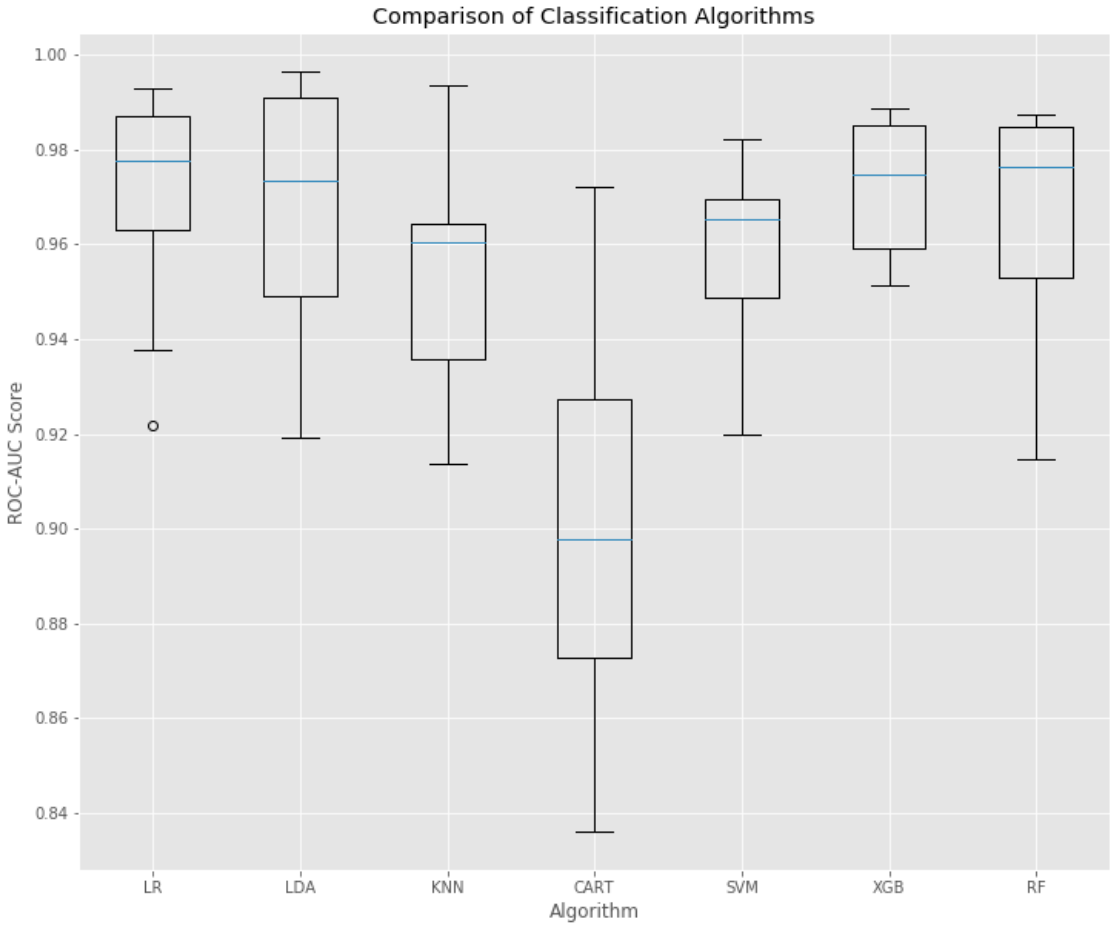


Box plots provide us with a good intuition of whether we need to worry about outliers as all transactions outside of 1.5 times the IQR (Inter-Quartile Range) are usually considered to be outliers. However, removing all transactions outside of 1.5 times the IQR would dramatically decrease our training data size, which is not very large, to begin with. Thus, I decided to only focus on extreme outliers outside of 2.5 times the IQR.

Classification Algorithms

To get a better feeling of which algorithm would perform best on our data, let's quickly spot-check some of the most popular classification algorithms:

- Logistic Regression
- Linear Discriminant Analysis
- K Nearest Neighbors (KNN)
- Classification Trees
- Support Vector Classifier
- Random Forest Classifier
- XGBoost Classifier



As we can see, there are a few algorithms that quite significantly outperformed the others. Now, what algorithm do we choose? As mentioned above, this project had not only the focus of achieving the highest accuracy but also to create business value. Therefore, choosing Random Forest over XGBoost might be a reasonable approach in order to achieve a higher degree of comprehensiveness while only slightly decreasing performance. To further illustrate what I mean by this, here is a visualization of our Random Forest model that could easily be used to explain very simply why a certain decision was made

Result

Identify fraudulent credit card transactions.
Given the class imbalance ratio, we recommend measuring the accuracy using the Area Under the Precision-Recall Curve (AUPRC). Confusion matrix accuracy is not meaningful for unbalanced classification.

The code prints out the number of false positives it detected and compares it with the actual values. This is used to calculate the accuracy score and precision of the algorithms.

The fraction of data we used for faster testing is 10% of the entire dataset. The complete dataset is also used at the end and both the results are printed.

These results along with the classification report for each algorithm is given in the output as follows, where class 0 means the transaction was determined to be valid and 1 means it was determined as a fraud transaction.

Conclusion

Fraud detection is a complex issue that requires a substantial amount of planning before throwing machine learning algorithms at it. Nonetheless, it is also an application of data science and machine learning for the good, which makes sure that the customer's money is safe and not easily tampered with.

Future work will include a comprehensive tuning of the Random Forest algorithm I talked about earlier. Having a data set with non-anonymized features would make this particularly interesting as outputting the feature importance would enable one to see what specific factors are most important for detecting fraudulent transactions

Questions and Answers

Q1. What was the size of the data?
Answer: 5000 rows and 10 columns

Q2. What was the data type?

Answer: Int and float

Q3. What was the team size and distribution?
Answer: 5

Q4. What techniques were you using for data pre-processing for various data science use cases and visualization?
Answer:

- While preparing data for a model, data should be verified using multiple tables or files to ensure data integrity.
- Identifying and removing unnecessary attributes.
- Identifying, filling or dropping the rows/columns containing missing values based on the requirement.
- Checking null values.
- Identifying, filling or dropping the rows/columns containing missing values based on the requirement.

- Based on the requirement, form clusters of data to avoid an overfitted model.
- Scaling the data so that the difference between the magnitudes of the data points in different columns are not very big.
- Converting the categorical data into numerical data.
- Replacing or combining two or more attributes to generate a new attribute which serves the same purpose.
- Trying out dimensionality reduction techniques like PCA(Principal Component Analysis), which tries to represent the same information but in a space with reduced dimensions.

Q5. What were your roles and responsibilities in the project?
Answer: My responsibilities consisted of gathering the dataset, Cleaning, and getting familiar with the data, model training. Performing various EDA and feature engineering techniques and testing by applying various machine learning models. Also performing hyper-parameter Tuning for getting optimized results.

Q6. In which area you have contributed the most?
Answer: I contributed the most to data preparation, preprocessing and model training areas. Also, we did a lot of brainstorming for finding and selecting the best algorithms for our use cases. After that, we identified and finalized the best practices for implementation.

Q7. In which technology you are most comfortable?
Answer: I have worked in almost all the fields viz. Machine Learning, Deep Learning, and Natural Language Processing, and I have nearly equivalent knowledge in these fields. But preferred one is loved working in ML.

Q8. In how many projects you have already worked?
Answer: I have worked in various projects e.g., object detection, object classification, object identification, NLP projects, machine learning regression, and classification problems.

Q9. What kind of challenges have you faced during the project?
Answer: The biggest challenge that we face is in terms of obtaining a good dataset, since it was an imbalanced dataset, cleaning it to be fit for feeding it to a model. Then comes the task of finding the correct algorithm to be used for that business case. Then that model is optimized. If we are exposing the model as an API, then we need to work on the SLA for the API as well, so that it responds in optimum time.

Q10.How did you optimize your solution?
Answer: Model optimization depends on a lot of factors.

- Train with better data (increase the quality),or do data pre-processing steps more efficiently.
- Increase the quantity of data used for training.
- Hyper- parameter tuning performs regularly