Project Report for

IME 672A

DATA MINING AND KNOWLEDGE DISCOVERY

TITLE:

Credit Card Fraud Detection

PREPARED BY:

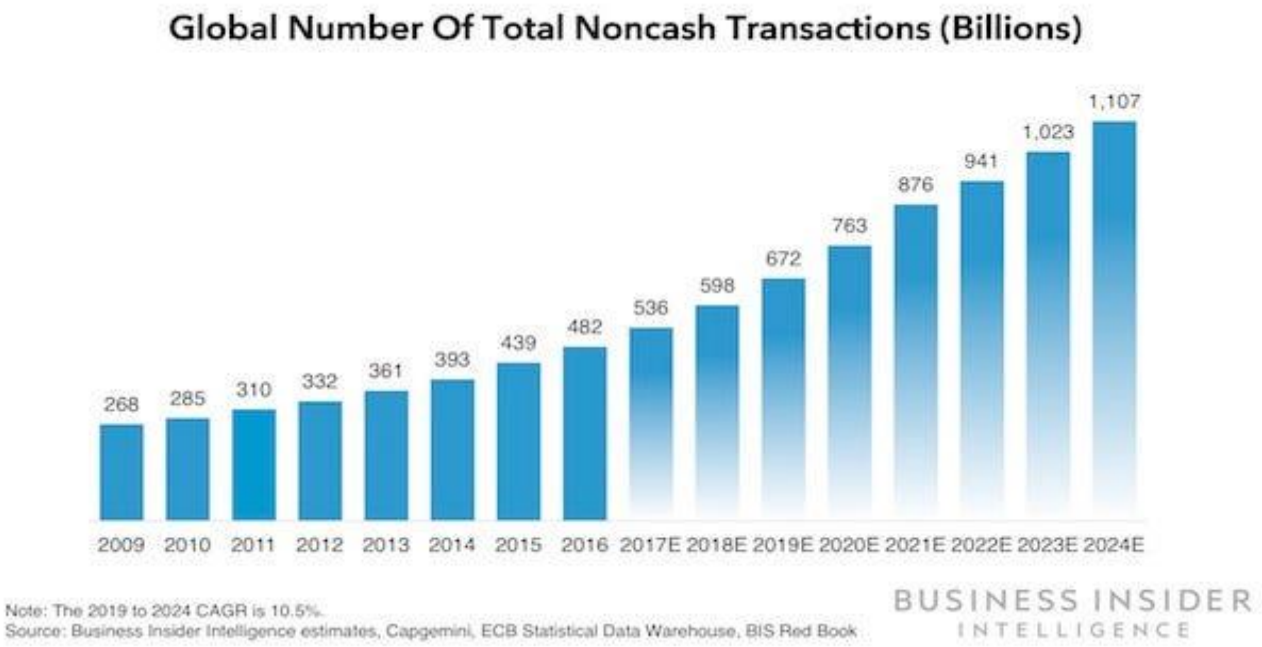
Group No. : 3	
Name	**Roll No**
Jaydeep Amrutbhai Sarvakar	20114008
Mayank Umrao	20114011
Nitesh Sharma	20114013
Prince Singh	20114014
Sumit Tripathi	20114022

Under the Supervision of:

Dr. Faiz Hamid

INTRODUCTION

We are on the express train to a cashless society in today's world. Global Non-cash transactions surged to nearly 14% from 2018-19 to reach 708.5 billion transactions, the highest growth rate recorded in the past decade. Also, it's expected in future, that there will be a steady growth of non-cash transactions as shown below:



Although it looks like a piece of promising news, the flip side of the growing number of online transactions is that fraudulent transactions are also on the rise. Even though EVM smart chips have been implemented in the credit cards, there is still a huge amount of money loss from credit card fraud.

According to data released by RBI, ATM/ Debit card fraud involved the highest number of fraud cases recorded in India in the year 2019. It involved 6117 cases of credit card fraud of worth Rs 19.7 crores, as shown below in the newspaper headline. Hence, Credit Card fraud detection methods are highly needed.

CASES BETWEEN OCT-DEC 2019

	OCTOBER		NOVEMBER		DECEMBER	
Type	Cases	Amount Lost*	Cases	Amount Lost*	Cases	Amount Lost*
ATM/Debit	3,376	73.65	3,533	10.55	4,149	10.33
Credit Card	1,641	4.04	1,711	4.77	2,765	10.87
Netbanking	360	7.01	2,256	4.31	1,250	2.27
Total	5,377	84.70	7,500	19.63	8,164	23.47

Source: RBI | * In Rs Crore

- 2017-18 saw 34,791 cases involving ₹169 crore
- 2018-19 saw 52,304 cases involving ₹149.4 crore
- 2019-20 (April-September) saw 30,965 cases involving ₹100.6 crore
- Between April 2017 & Dec 2019 a total of 1.1 lakh cases saw ₹547 crore being stolen

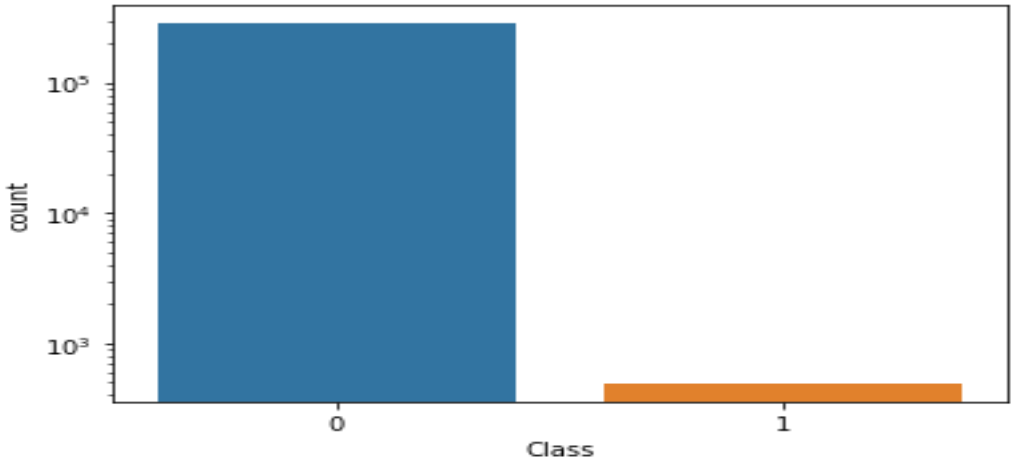
DATA DESCRIPTION

We were provided with the Kaggle Dataset which consisted of anonymized credit card transactions, which were labelled as fraudulent, or genuine.

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. The attribute Class has binary values, showing 0 for genuine transactions, and 1 for fraudulent transaction.

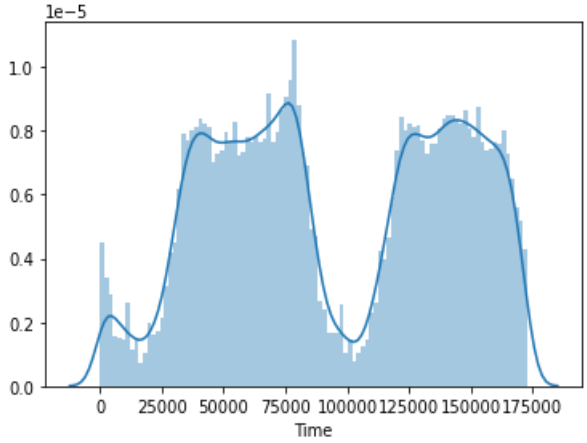
The data set contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we were not able to receive the original features and more background information about the data. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'.

Bar Graph plotted on a logarithmic Scale, showing the Class Imbalance in given Data Set.



Analysing the Time Feature

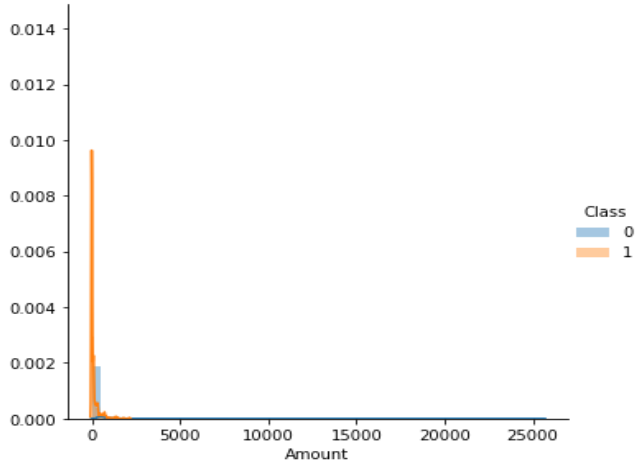
Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. By seeing the upper limit of the value that the 'Time' Feature takes, we can estimate that the data has transactions that occurred in 2 days (2-days = 1,72,800 seconds).



Just by, visual Inspection of the above graph one can notice the crests and the troughs formed, one can deduce that the high peaks of the graph would correspond towards the frequency during the day-time transactions and the low points, towards the transactions carried out in the night.

Analysing the Amount Feature

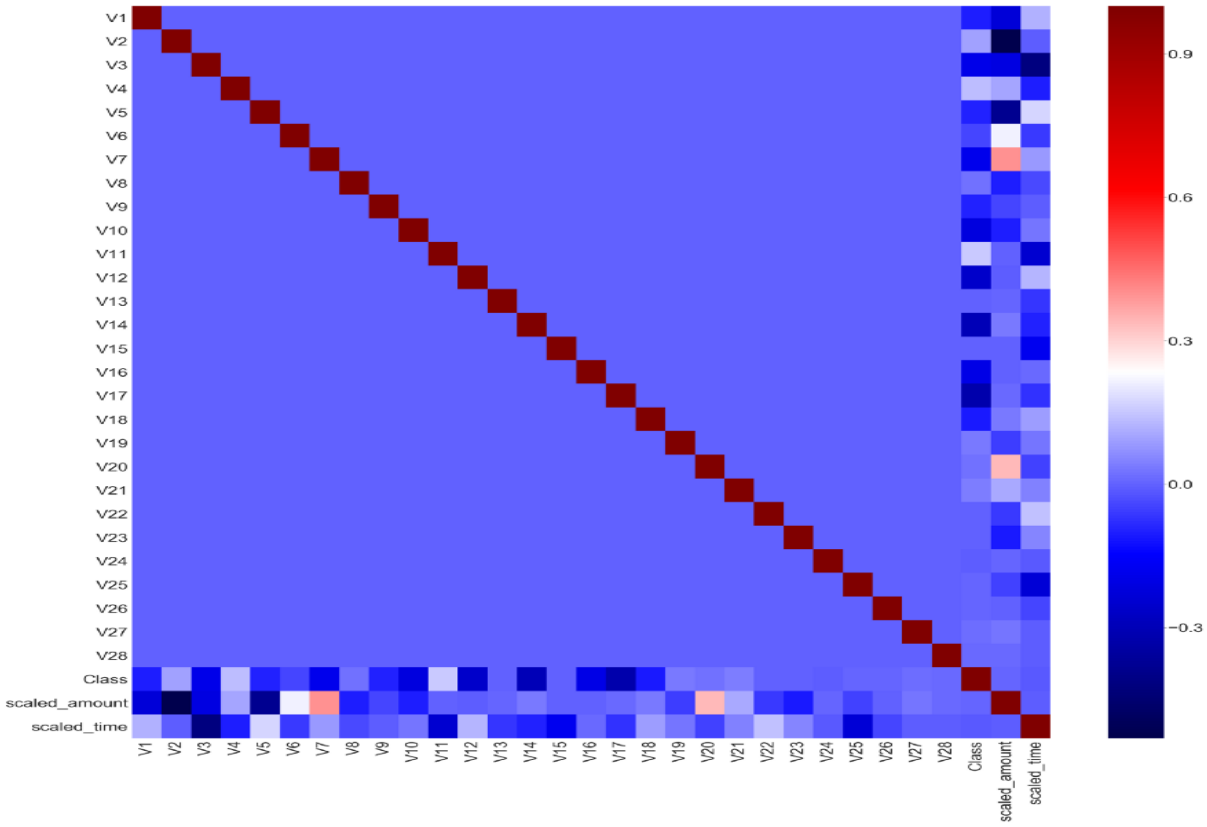
The data set contains 284,807 transactions. The mean value of all transactions is $88.35 while the largest transaction recorded in this data set amounts to $25,691.16. However, as one might be guessing right now based on the mean and maximum, the distribution of the monetary value of all transactions is heavily right skewed. Most transactions are relatively small and only a tiny fraction of transactions comes even close to the maximum. Thus, a need for standardising the amount feature arises.



DATA PRE-PROCESSING

To know if there are any significant correlations between our predictors, especially with regards to our class variable. One of the most visually appealing ways to determine that is by using a heatmap. It predicts if there is any strong collinearity present in the data.

Heatmap of Correlation



From the above Heatmap, a few attributes showing relatively high correlation are:

- Time and V3 (-0.42)
- Amount and V2 (-0.53)
- Amount and V4 (0.4)

Standard-Scaler on Time and Amount

It is a good idea to scale the data so that the column(feature) with lesser significance might not end up dominating the objective function due to its larger range. In addition, features having different unit should also be scaled thus providing each feature equal initial weightage .This will result in a better prediction model.

Splitting the data

Before we start training the model, we'll need to split our dataset into a training and test portion. We'll use the training portion to train model and then evaluate it on the test portion to see how it performs on samples it hasn't seen before. It's also important to perform a stratified sampling which means that the probability of seeing a fraudulent transaction will be approximately the same in both the training data and the test data. Stratified sampling also ensures that our model metrics are as close as possible to what we'd see in a whole population.

Modelling Building

1. **Classification Models**

- Logistic Regression
- Decision Trees
- Random Forest
- Naive Bayes Classifier

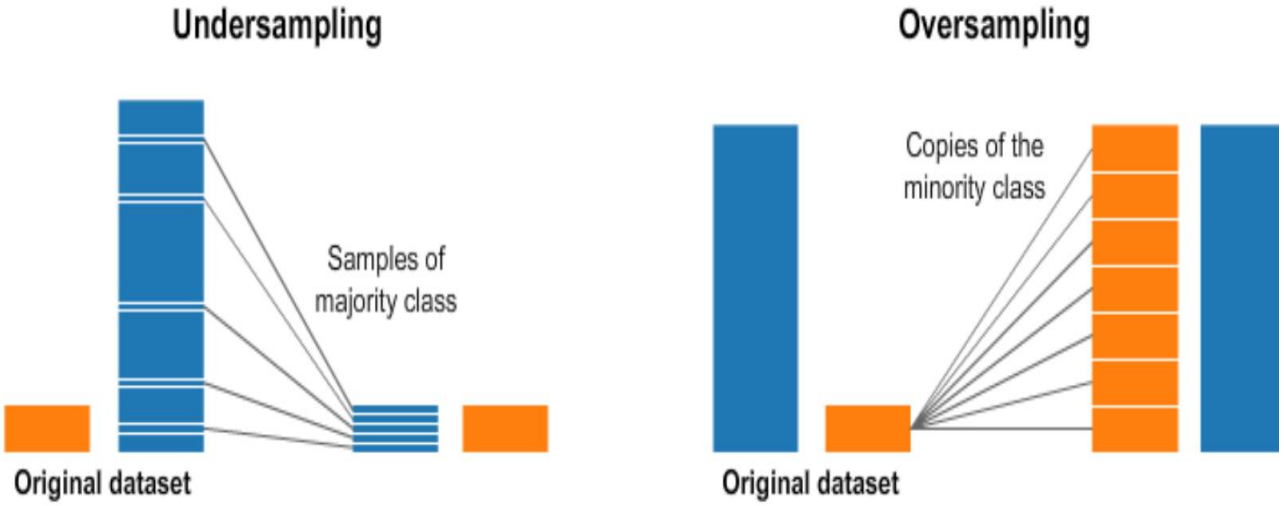
2 .**Class Imbalance Solutions**

- Under Sampling
- Over Sampling
- SMOTE
- ADAS

3. **Metrics**

- Accuracy Score
- Confusion Matrix
- Precision Score
- Recall Score
- ROC_AUC
- F1 Score

Building different models with different balanced dataset



Under sampling and Oversampling: Under sampling techniques remove examples from the training dataset that belong to the majority class to better balance the class distribution, such as reducing the skew from a 1:100 to a 1:10, 1:2, or even a 1:1 class distribution. This is different from oversampling that involves adding examples to the minority class to reduce the skew in the class distribution.

For Under Sampled Data

- Original dataset shape Counter({0: 284315, 1: 492})
- Resampled dataset shape Counter({0: 492, 1: 492})

For Oversampled Data

- Original dataset shape Counter({0: 284315, 1: 492})
- Resampled dataset shape Counter({0: 284315, 1: 284315})

SMOTE sampling and ADASYN sampling: The key difference between ADASYN and SMOTE is that the former uses a density distribution, as a criterion to automatically decide the number of synthetic samples that must be generated for each minority sample by adaptively changing the weights of the different minority samples to compensate for the skewed distributions. The latter generates the same number of synthetic samples for each original minority sample

For SMOTE Data

- Original dataset shape Counter({0: 284315, 1: 492})
- Resampled dataset shape Counter({0: 284315, 1: 284315})

For ADASYN Data

- Original dataset shape Counter({0: 284315, 1: 492})
- Resampled dataset shape Counter({0: 284315, 1: 284310})

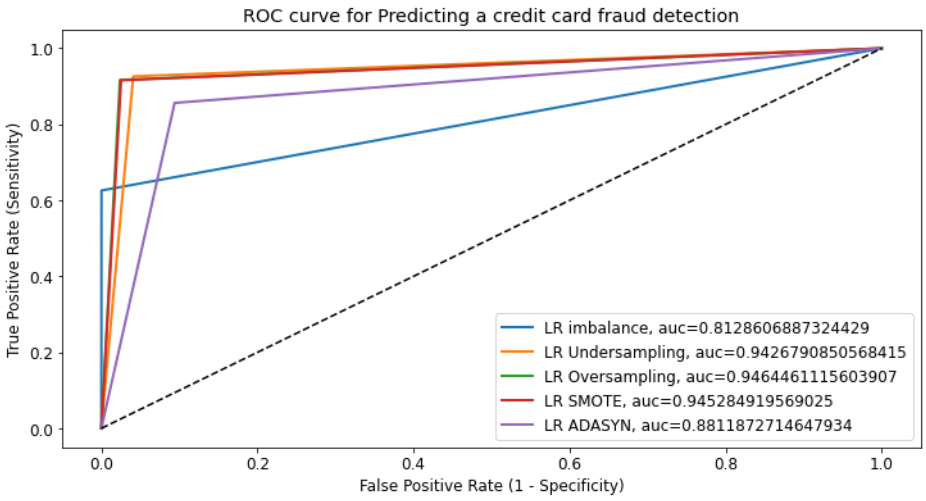
1. Logistic Regression (LR)

Logistic regression mathematically means finding, $y = f(x)$, when y is a categorical variable. The use of this method is to find the label of categorical variable when know our predictors x. In our project the dependent variable y is a categorical variable, and the predictors are continuous variable. The input given to the model is the weighted average of attributes value and some bias. The input is processed by the logistic function called sigmoid function.

$$f(x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x+\cdots\ldots)}}$$

where $\beta_0 + \beta_1 x + \cdots$ is the weighted input given to the sigmoid function. The function processed the input data and result the output. The output is then compared with a threshold value. If the output is at least equal to the threshold value, then the output is 1 otherwise 0. Confusion Matrix is shown below for the Logistic Regression model on the Oversampling data, (which) turns out to give the best result out of the 4 sampling techniques employed.

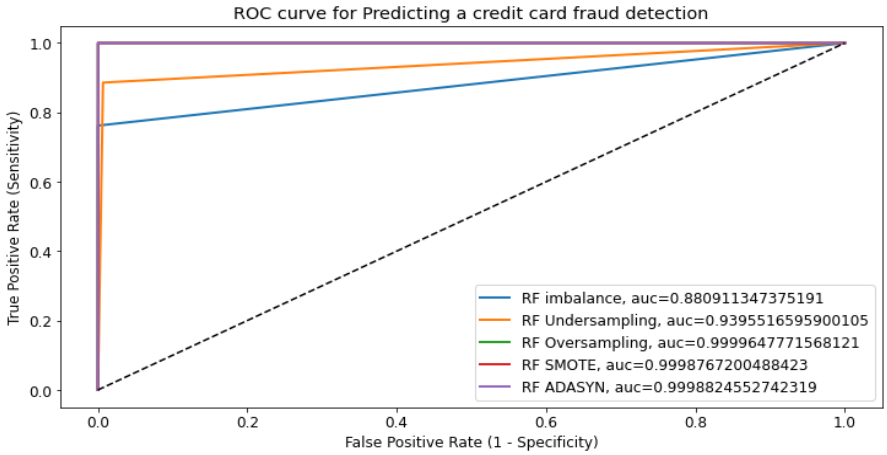
LR Oversampling		Predicted Value	
		No	Yes
Actual Value	No	8317	2025
	Yes	7118	78299



2. Random Forest (RF)

As its name suggests, Random Forest consists of many individual decision trees, that act as an ensemble. Every single tree in the Random Forest spits out a class prediction, and the class with most votes becomes the prediction of our model. Given below is the confusion matrix with the best result

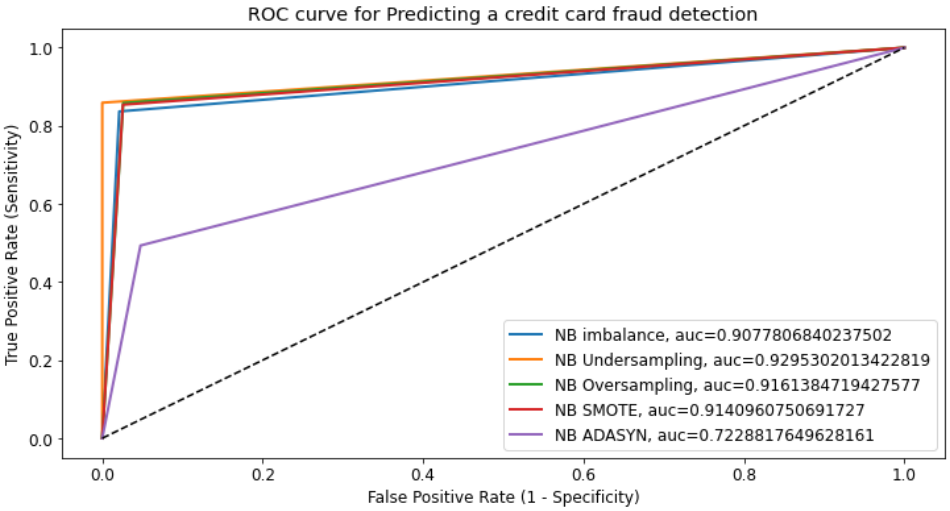
RF Oversampling		Predicted Value	
		No	Yes
Actual Value	No	85166	6
	Yes	0	85417



3 .Naïve Bayes (NB)

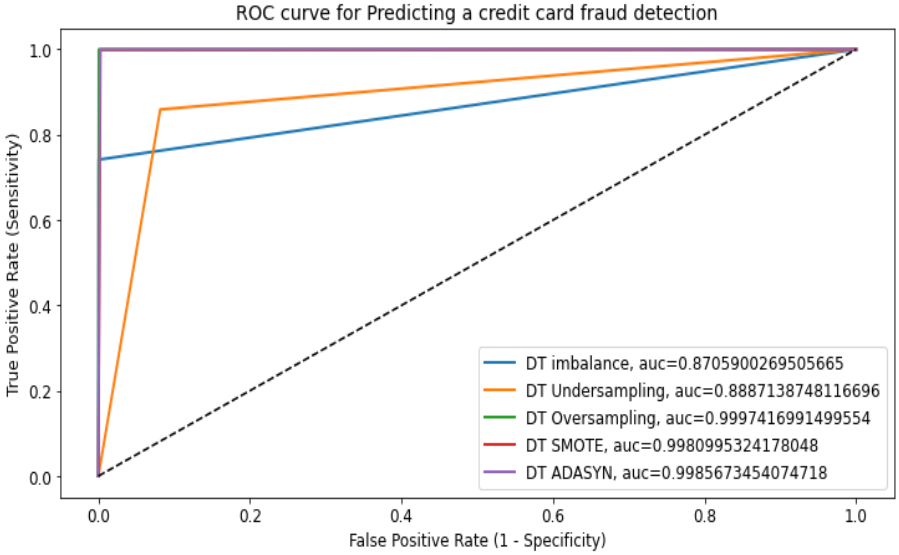
Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

While, on the other hand Decision Trees makes use of Information Theory to apply classification procedure over the data set. Choosing of an attribute to split, lies at heart of the algorithm.



4. Decision Trees (DT)

Decision Trees makes use of Information Theory to apply classification procedure over the data set. Growing a tree involves deciding on **which features to choose** and **what conditions to use** for splitting, along with knowing when to stop. As a tree generally grows arbitrarily, **you will need to trim it down** for it to be of use. Choosing of an attribute to split, lies at heart of the algorithm.



Comparing Models and Result

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 labelling a transaction as fraudulent, we want to detect fraudulent transactions and label them as such. For the above reason, Accuracy is not a good metric for the model comparison, especially for an imbalanced data set there are more effective metrics to use, such as:

1. Recall: Recall answers the question: **out of the fraudulent transactions, what percentage of these are correctly identified by our model?** In our best model has a recall ratio as 1 .

$$Recall = \frac{TP}{TP + FN}$$

2. Precision: Precision answers the question**: out of all the transactions predicted to be fraudulent, what percentage were fraudulent?** In our best model has a recall ratio as 0.99 .

$$Precision = \frac{TP}{TP + FP}$$

3. F-1 Score: The F1 score combines Recall and Precision into one metric as a weighted average of the two. Unlike Recall and Precision individually, **F1 takes both false positives and false negatives into consideration.** In imbalanced classes such as this, F1 is much more effective than accuracy at determining the performance of the model.

$$F1Score = 2 * \frac{Recall * Precision}{Recall + Precision}$$

The result with all the model and corresponding metrics are given in below table:

No.	Model	Accuracy	AUC	Precision	Recall	F1 Score
1	RF oversampling	0.999965	0.999965	0.999930	1	0.999965
2	RF ADASYN	0.999883	0.999882	0.999766	1	0.999883
3	RF SMOTE	0.999877	0.999877	0.999754	1	0.999877
4	DT oversampling	0.999742	0.999742	0.999485	1	0.999743
5	DT ADASYN	0.998570	0.998567	0.9976898	0.999462	0.998575
6	DT SMOTE	0.998101	0.998100	0.997300	0.998911	0.998105
7	LR oversampling	0.946403	0.946446	0.974790	0.916668	0.944836
8	LR SMOTE	0.945243	0.945285	0.973204	0.915860	0.943661
9	LR under-sampling	0.942568	0.942679	0.958333	0.926174	0.941980
10	RF under-sampling	0.939189	0.939552	0.992481	0.885906	0.936170
11	NB under-sampling	0.929054	0.929530	1	0.859060	0.924188
12	NB Oversampling	0.916056	0.916138	0.970548	0.858401	0.911036
13	NB SMOTE	0.914010	0.914096	0.970800	0.853952	0.908636
14	NB imbalance	0.978582	0.907781	0.063764	0.836735	0.118497
15	DT under-sampling	0.888514	0.888714	0.914286	0.859060	0.885813
16	LR ADASYN	0.881123	0.881187	0.901659	0.856246	0.878366
17	RF imbalance	0.999508	0.880911	0.941176	0.761905	0.842105
18	DT imbalance	0.999239	0.870590	0.801471	0.741497	0.770318
19	LR imbalance	0.999228	0.812861	0.893204	0.625850	0.736000
20	NB ADASYN	0.722290	0.722882	0.912306	0.493440	0.640469

From the above In the ROC graph above, the AUC scores for Random Forest with Oversampling technique are high, which is what we'd like to see. As we move further right along the curve, we both capture more True Positives but also incur more False Positives. This means we capture more fraudulent transactions, but also flag even more normal transactions as fraudulent.

So Random Forest with Oversampling technique is our final model, as this gives highest Recall approx. score of 100% on both train and test datasets.

Conclusion & Future Work

Fraud detection is a complex issue that requires a substantial amount of planning before applying machine learning algorithms at it. Nonetheless, an effective credit card detection system is the need of today, 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.