CREDIT CARD FRAUD DETECTION USING RANDOM FOREST CLASSIFIER

A

Final Report

Submitted by

SOURADEEP BANERJEE (R110216159)
DEVYANSHI TIWARI (R110216060)
PRATYUSH SHARMA (R103216072)

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHONOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

At

UNIVERSITY WITH A PURPOSE

SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF PETROLEUM & ENERGY STUDIES
Bidholi Campus, Energy Acres, Dehradun – 248007.

April – 2020

CANDIDATE DECLARATION

We hereby certify that the project work entitled "Credit Card Fraud Detection using Random Forest Classifier" in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in CLOUD COMPUTING AND VIRTUALIZATION TECHNOLOGY and submitted to the Department of Virtualization at School of Computer Science and Engineering, University of Petroleum and Energy Studies, Dehradun, is an authentic record of our work carried out during a period from 2 Februaruy,2020 to 22nd April,2020 under the supervision of Dr. Jagdish Chandra Patni, Assistant Professor (SG), Department of Virtualization, University of Petroleum and Energy Studies.

The matter represented in this project has not been submitted by us for the award of any other degree or at any other University.

(Devyanshi Tiwari)	**(Souradeep Banerjee)**	**(Pratyush Sharma)**
(R110216060)	**(R110216159)**	**(R103216072)**

This is to certify that the statement made by the candidates is correct to the best of my knowledge.

Date:

Dr. Jagdish Chandra Patni
Project Mentor
Assistant Professor (SG)
Department of Virtualization

Dr. Deepshika Bharghava
Head of Department
School of Computer Science and Engineering
University of Petroleum and Energy Studies
Dehradun – 248007 (Uttarakhand)

CERTIFICATE

This is to certify that the project titled "Credit Card Fraud Detection using Random Forest Classifier" is the bonafide work carried out by Souradeep Banerjee, Devyanshi Tiwari and Pratyush Sharma, students of BTech (CSE) of University of Petroleum and Energy Studies, Dehradun(India) during the academic year 2019-20, in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology (Computer Science and Engineering) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

Date:

Dr. Jagdish Chandra Patni
Assistant Professor (SG)
Department of Virtualization

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide **Dr. Jagdish Chandra Patni**, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank our respected programme Head of the Department Dr. Deepshika Bharghava, for her great support in doing our project in Areas in SoCS.

We are also grateful to Dr Manish Prateek, Dean, SoCS for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all of our friends for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our parents who have shown us this world and for every support they have given us.

Devyanshi Tiwari (R110216060)	**Souradeep Banerjee (R110216159)**	**Pratyush Sharma (R103216072)**

ABSTRACT

In today's world we are on an express train to a cashless society which has led to a tremendous escalation in the use of credit card transactions. But the flip-side of this is that fraudulent activities are on the increase; therefore, implementation of a methodical fraud detection system is indispensable to card holders as well as the card issuing banks. In this project we are using Random Forest algorithm to detect card fraud with precision. Accuracy and specificity are some of the key aspects on the basis of which performance of the model is assessed and graphical model visualization is achieved. To cut down on extraneous cost of high processing servers and other hardware we are going to utilize cloud computing. Also, to decrease the processing load on the client side we are going to deploy our machine learning model on the Google Cloud platform.

Keywords: Credit Card Fraud Detection, Random Forest, Accuracy, Cloud

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Credit card fraud is an umbrella term used to refer to the use of credit cards to buy services or goods with the objective of eluding payment. This includes identity theft, identity assumption or even a fraud spree. Fraud Detection is the method of monitoring the behavior of the transactions done by the cardholder to detect any unauthorized transactions. Traditional methods of fraud detection have been used since a long period of time; however, these methods are very time consuming and often inefficient. Therefore, a combination of machine learning and artificial intelligence is required for effective detection of fraud.

This includes collection of data from millions of credit card users followed by pre-processing of data and creation of user's profile. This is followed by finding association among the data set and using different ML algorithms; inconsistency in the nature of transaction are then observed. Basic principle of fraud detection using machine learning is based on the concept of training a model which understands the transaction records that are already known to be legit and detects any variation from these transactions.

In this project Random Forest Algorithm has been used which is an example of supervised learning algorithm. This algorithm is composed of simple tree predictors which are each used for prediction and finally the best solution among them is selected. It is mainly used for classification and regression problems. The reason for the selection of Random Forest algorithm for our project are as follows:

1. This algorithm shows high accuracy as compared to other classification algorithms and works well with very large datasets.

2. Several variables can be handled and there is no problem of deletion of variables in case of missing values.

Before beginning to develop the model, data balancing needs to be taken into account. This includes dealing with the problem of huge imbalance in the dataset. Learning from unbalanced datasets is a difficult task since most learning algorithms are not designed to cope with a large difference between the number of cases belonging to different classes. As less than 0.5% of transactions are fraud among 500,000 transactions so even if a model incorrectly deems a fraud transaction to be legal the accuracy of the model still will be over 99% which is unacceptable. To solve this problem oversampling using SMOTE is being used. SMOTE or Synthetic Minority Oversampling Technique which means

duplication of examples in the minority class before fitting the model. SMOTE works by linear interpolation of the minority class. The synthetic records are generated by applying K nearest neighbor to examples in the minority class. This method is efficient as new examples are created which are relatively close to the already existing examples.

Since the data is collected from the internet PCA has already been applied to the dataset to protect the confidentiality of the users. PCA or Principle Component Analysis is statistical technique to decrease the dimension of the features space by performing feature extraction. The idea behind the approach is reduce the dimensionality of the data set while still retaining the variations present up to the maximum possible extent.

Since our project deals with a very large data set and our final aim is to enhance the efficiency of the fraud detection technique, we are going to make use of cloud computing to ensure cost efficiency, security and backup of the model.

1.2 PROBLEM STATEMENT

Balancing the unbalanced data set of a bank's credit card details about customers transaction using oversampling. The bank's dataset has already been processed using Principal Component Analysis so as to hide the details about the customers and thus maintain confidentiality. After balancing the dataset, we will be using the random forest classifier in order to create a machine learning model which can predict credit card fraud transactions with maximum accuracy.

1.3 OBJECTIVES

The main idea of this project is to demonstrate the use of random forest classifier algorithm which will be used to train the machine learning model. Again, before training the model we need to make sure that the data set we are using is balanced properly so to decrease the number of false positives in the model. We are going to do the same by an oversampling technique called the SMOTE algorithm. After we have developed the machine learning model, we are going to deploy the same on Google Cloud so as to decrease the processing overhead on client side.

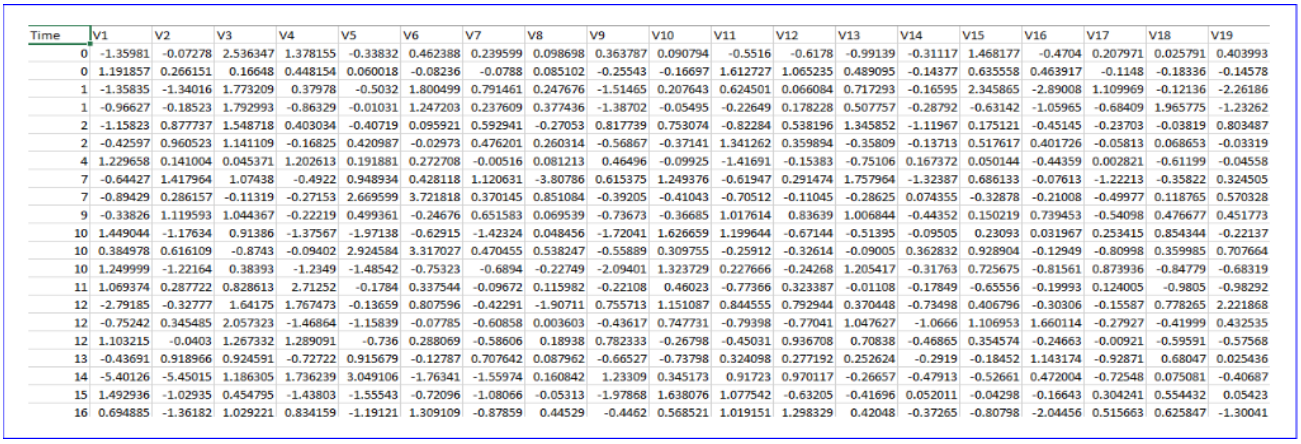
2. LITERATURE REVIEW

There are many techniques available to detect fraud transactions. Although none is able to detect all frauds completely when they are actually happening, they usually detect it after fraud has been committed. This happens because the fraudulent transactions are small in number as compared to total transactions. The authors in paper [1], compared 7 techniques to detect such transactions. ANN got the best results for all parameters such as accuracy at 99.71%, detection rate at 99.68% and false alarm rate at 0.12%. Although ANN takes most time and compute power to train. SVM has most false alarm rate at 5.2% and detection rate of 85.45% not being comparable to other better techniques. Fuzzy logic has worst detection rate at 77.8%. Decision trees are balanced towards complexity to train and results acquired with accuracy at 97.93%, detection rate at 98.52% and false alarm rate at 2.19%. Random Forest is a decision tree regression and classification technique that works well with both categorical and numerical data [2]. The authors tested random forest and SVM classifier to detect fraudulent transactions from dataset. The pre-processing was done to avoid missing values and scale feature values. The authors concluded that imbalanced data did not worked well with SVM as compared to random forest classifier. Another advantage of using random forest technique was introduction of new data points did not have major impact on the model since it used subset of data with different decision trees. Each tree has a very low chance to impact other and hence also avoid bias and overfitting to an extent. In paper [3] the authors studied random forest classification to detect fraud credit card transactions. The dataset used has values masked through PCA algorithm. Scaling of feature values were done to reduce variance among features. SMOTE algorithm has been used to balance data. SMOTE stands for Synthetic Minority Oversampling Technique. This is a statistical technique for increasing the number of cases in your dataset in a balanced way. The module works by generating new instances from existing minority cases that user supply as input. The balanced data contains 175000 classes. Random Forest classifier is used for binary classification of data points. From the results published in the paper, the precision-recall curve has equal value around 0.85. Random forest classifier has become one of the most common technique used in e-commerce to detect credit card frost due to its flexibility and scalability it provides for large datasets. Computational power required in training random forest model is low as compared to better state of art techniques like ANN. Although ANN are not being deployed in real-time e-commerce solutions in large extent due to computational and time constraints. In present day's class imbalance is a major problem in the research field. Imbalanced datasets can mislead a research work [4]. In the paper authors discussed for balancing data for efficient analysis, regression and classification problems. The major techniques they studied were Random oversampling and undersampling, statistical oversampling and

undersampling, SMOTE, Feature Selection, Hybrid Sampling, Cost-effective Learning, and Ensemble Learning. In this research work 52 research papers are reviewed and classified based on the author's country, year of publication and techniques used in those papers which helps to identify the frequency of the technique used for balancing the data. Of all the papers reviewed for this research SMOTE technique is most commonly used one and feature selection is the second most used technique. These two techniques provide best results for balancing problem in data analysis.

3. METHODOLOGY

In order to develop a machine learning model which can predict credit card frauds by seeing the transaction, we need a good amount of data of previous transactions of the customers of a bank. Now the dataset that is available from the bank has Principal Component Analysis or PCA already applied to it so as to hide the confidential data of the customers.



Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19
0	-1.35981	-0.07278	2.536347	1.378155	-0.33832	0.462388	0.239599	0.098698	0.363787	0.090794	-0.5516	-0.6178	-0.99139	-0.31117	1.468177	-0.4704	0.207971	0.025791	0.403993
0	1.191857	0.266151	0.16648	0.448154	0.060018	-0.08236	-0.0788	0.085102	-0.25543	-0.16697	1.612727	1.065235	0.489095	-0.14377	0.635558	0.463917	-0.1148	-0.18336	-0.14578
1	-1.35835	-1.34016	1.773209	0.37978	-0.5032	1.800499	0.791461	0.247676	-1.51465	0.207643	0.624501	0.066084	0.717293	-0.16595	2.345865	-2.89008	1.109969	-0.12136	-2.26186
1	-0.96627	-0.18523	1.792993	-0.86329	-0.01031	1.247203	0.237609	0.377436	-1.38702	-0.05495	-0.22649	0.178228	0.507757	-0.28792	-0.63142	-1.05965	-0.68409	1.965775	-1.23262
2	-1.15823	0.877737	1.548718	0.403034	-0.40719	0.095921	0.592941	-0.27053	0.817739	0.753074	-0.82284	0.538196	1.345852	-1.11967	0.175121	-0.45145	-0.23703	-0.03819	0.803487
2	-0.42597	0.960523	1.141109	-0.16825	0.420987	-0.02973	0.476201	0.260314	-0.56867	-0.37141	1.341262	0.359894	-0.35809	-0.13713	0.517617	0.401726	-0.05813	0.068653	-0.03319
4	1.229658	0.141004	0.045371	1.202613	0.191881	0.272708	-0.00516	0.081213	0.46496	-0.09925	-1.41691	-0.15383	-0.75106	0.167372	0.050144	-0.44359	0.002821	-0.61199	-0.04558
7	-0.64427	1.417964	1.07438	-0.4922	0.948934	0.428118	1.120631	-3.80786	0.615375	1.249376	-0.61947	0.291474	1.757964	-1.32387	0.686133	-0.07613	-1.22213	-0.35822	0.324505
7	-0.89429	0.286157	-0.11319	-0.27153	2.669599	3.721818	0.370145	0.851084	-0.39205	-0.41043	-0.70512	-0.11045	-0.28625	0.074355	-0.32878	-0.21008	-0.49977	0.118765	0.570328
9	-0.33826	1.119593	1.044367	-0.22219	0.499361	-0.24676	0.651583	0.069539	-0.73673	-0.36685	1.017614	0.83639	1.006844	-0.44352	0.150219	0.739453	-0.54098	0.476677	0.451773
10	1.449044	-1.17634	0.91386	-1.37567	-1.97138	-0.62915	-1.42324	0.048456	-1.72041	1.626659	1.199644	-0.67144	-0.51395	-0.09505	0.23093	0.031967	0.253415	0.854344	-0.22137
10	0.384978	0.616109	-0.8743	-0.09402	2.924584	3.317027	0.470455	0.538247	-0.55889	0.309755	-0.25912	-0.32614	-0.09005	0.362832	0.928904	-0.12949	-0.80998	0.359985	0.707664
10	1.249999	-1.22164	0.38393	-1.2349	-1.48542	-0.75323	-0.6894	-0.22749	-2.09401	1.323729	0.227666	-0.24268	1.205417	-0.31763	0.725675	-0.81561	0.873936	-0.84779	-0.68319
11	1.069374	0.287722	0.828613	2.71252	-0.1784	0.337544	-0.09672	0.115982	-0.22108	0.46023	-0.77366	0.323387	-0.01108	-0.17849	-0.65556	-0.19993	0.124005	-0.9805	-0.98292
12	-2.79185	-0.32777	1.64175	1.767473	-0.13659	0.807596	-0.42291	-1.90711	0.755713	1.151087	0.844555	0.792944	0.370448	-0.73498	0.406796	-0.30306	-0.15587	0.778265	2.221868
12	-0.75242	0.345485	2.057323	-1.46864	-1.15839	-0.07785	-0.60858	0.003603	-0.43617	0.747731	-0.79398	-0.77041	1.047627	-1.0666	1.106953	1.660114	-0.27927	-0.41999	0.432535
12	1.103215	-0.0403	1.267332	1.289091	-0.736	0.288069	-0.58606	0.18938	0.782333	-0.26798	-0.45031	0.936708	0.70838	-0.46865	0.354574	-0.24663	-0.00921	-0.59591	-0.57568
13	-0.43691	0.918966	0.924591	-0.72722	0.915679	-0.12787	0.707642	0.087962	-0.66527	-0.73798	0.324098	0.277192	0.252624	-0.2919	-0.18452	1.143174	-0.92871	0.68047	0.025436
14	-5.40126	-5.45015	1.186305	1.736239	3.049106	-1.76341	-1.55974	0.160842	1.23309	0.345173	0.91723	0.970117	-0.26657	-0.47913	-0.52661	0.472004	-0.72548	0.075081	-0.40687
15	1.492936	-1.02935	0.454795	-1.43803	-1.55543	-0.72096	-1.08066	-0.05313	-1.97868	1.638076	1.077542	-0.63205	-0.41696	0.052011	-0.04298	-0.16643	0.304241	0.554432	0.05423
16	0.694885	-1.36182	1.029221	0.834159	-1.19121	1.309109	-0.87859	0.44529	-0.4462	0.568521	1.019151	1.298329	0.42048	-0.37265	-0.80798	-2.04456	0.515663	0.625847	-1.30041

Figure 3.1: Converted Data Set

Now what Principal Component Analysis does is, it converts the original data into a data of smaller dimension using the linear algebra and statistical techniques while preserving the variety that is present in the data for example, correlation and variance, etc.

Now that we have got our data set, we see that it is quite overbalanced since the number of fraud transactions present in the dataset is much lower than the actual number of transactions. So, if we use the data set directly in order to train the model then we are going to get a high accuracy but actually the thing is the system is going to label a transaction safe even if it is a fraud.

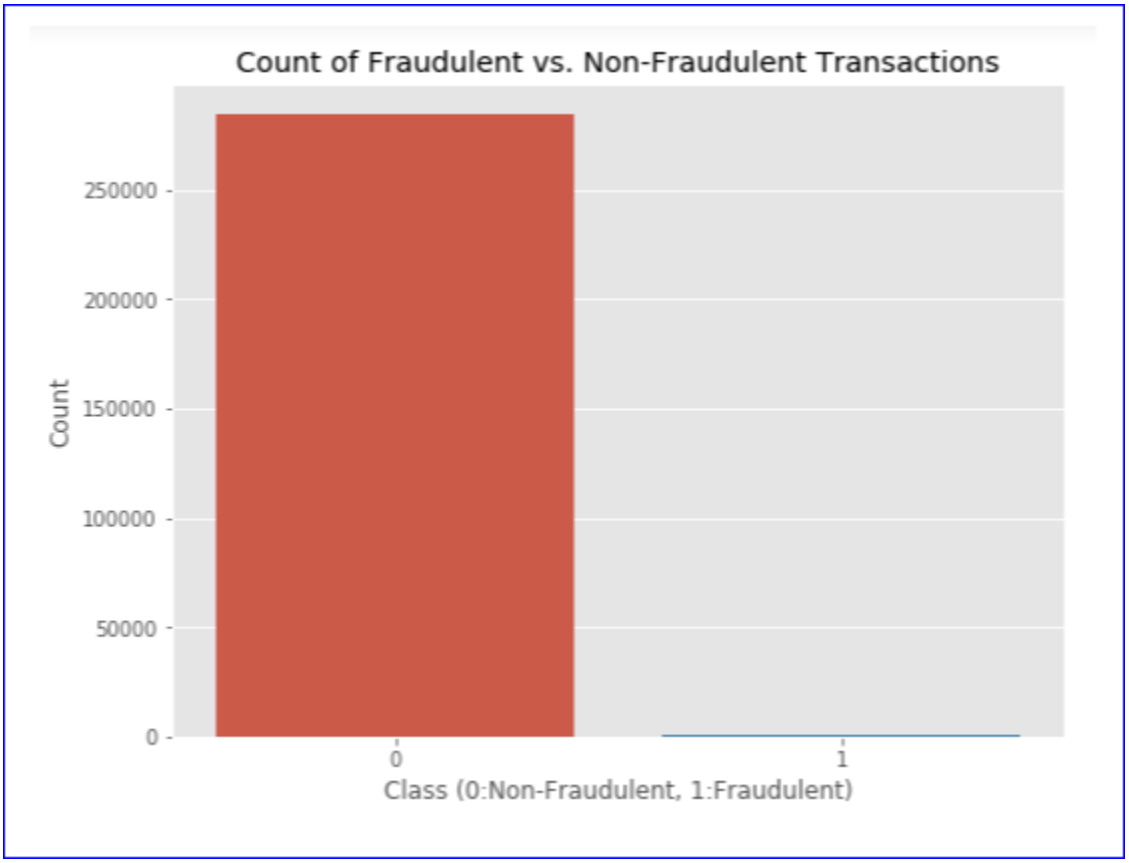


Figure 3.1: Histogram for fraudulent and Non fraudulent Transactions

So, in order to avoid such false negatives, we need to balance the data set. For balancing we are going to use the oversampling mechanism. In oversampling what we do is we try to increase the underrepresented minority class by using some kind of specific technique. In this case we are going to use the SMOTE or Synthetic Minority Oversampling Technique. SMOTE tries to increase the number of minority target class by using the features of the neighbors and thus making the new samples more generalized.

Once we are done with the balancing of the data, we then need to apply random forest classifier algorithm in order to train the machine learning model. Random forest classifier is the best algorithm to carry out this classification problem as it does the decision by aggregating the outcomes of thousands of decision trees. Once the model is trained, we are going to use it on the test data set and calculate the accuracy of the model.

Once the modelling is done, we are going to upload the model to Google cloud so that the cloud platform carries out all the processing needed for training and testing and decrease client processing overhead.

4. DESIGN

The data set that is being considered is being loaded into the program as a data frame using the pandas library. While displaying the first five rows of the data set, we can see that there are many random values in the dataset which also have negative values. This is due to the fact the dataset has already been processed using PCA algorithm in order to hide the confidential details of the customers, without affecting the variance and co-relationship that was present in the original data set.

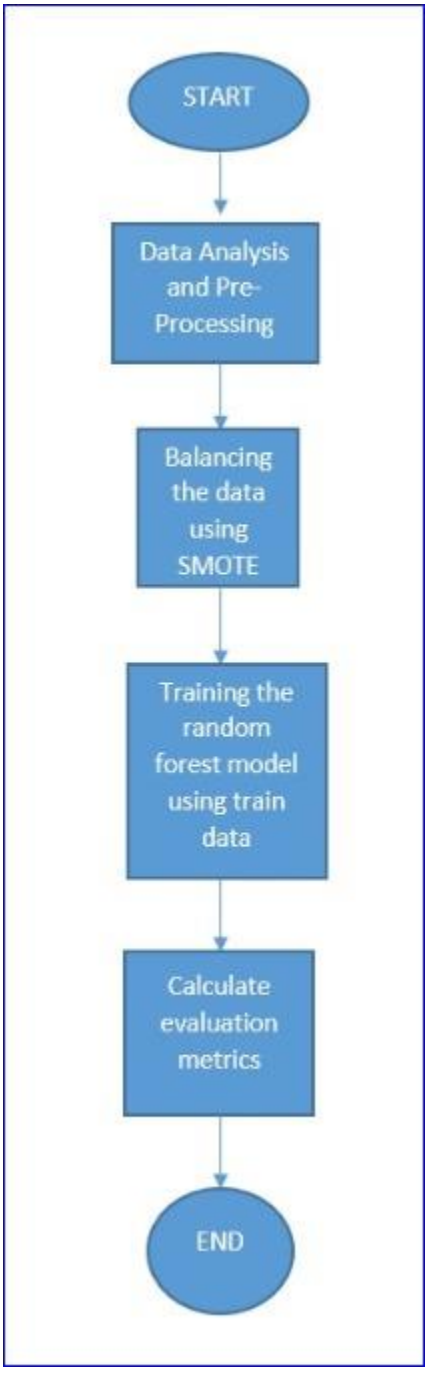


Figure 4.1: Flow of System

Once the data set is being loaded, we create a visual representation of the fact that the safe transaction class that is class 0 is the majority one and the class 1 which is the flagged transaction class is in minority. After that we apply a normalization technique in order to normalize the amount attribute within the range creating a new attribute called normAmount. After the normalization is done the original Amount attribute is being dropped along with Time which has no use in this case.

We use SMOTE or Synthetic Minority Oversampling Technique in order to increase the number of minority classes. After applying smote we apply random forest classifier in order to train the model. After the model is trained, we predict the values of testing classes and compare it with the original classes to calculate accuracy, precision, recall and f1 score.

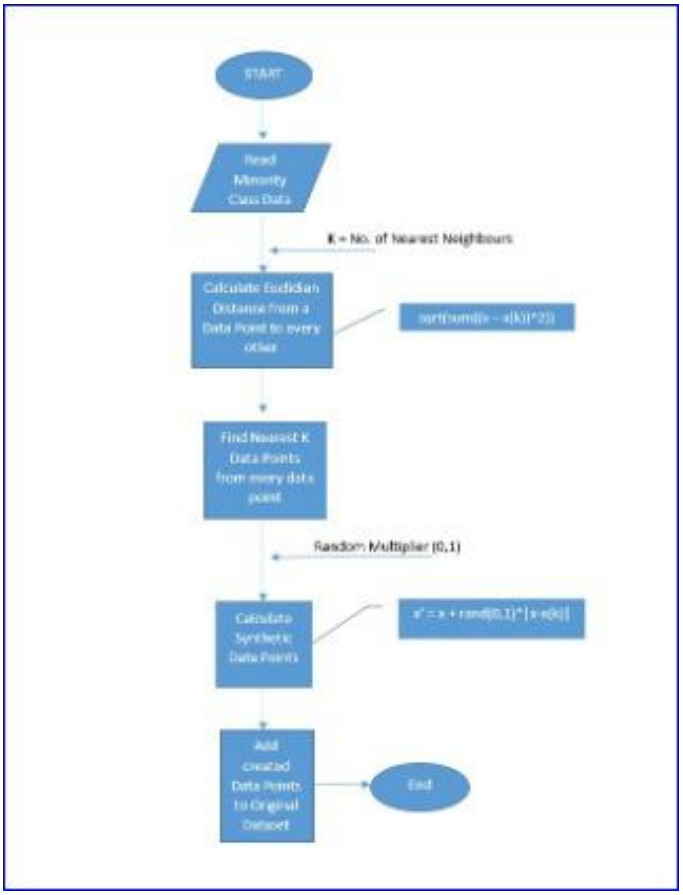


Figure 4.2: SMOTE Flow Chart

5. IMPLEMENTATION AND RESULTS

We are using the anaconda framework to create Jupyter notebooks in which we are going to import our data set and train the model. In total there are three workbooks and we are going to look at each one of them separately.

1. Data_Analysis.ipynb

In this workbook, we have imported the dataset into a Pandas DataFrame for ease of access and manipulation. The operations performed has given us basic idea of what our dataset represents and how to deal with it during designing, training and testing our model later on. Seaborn and Matplotlib libraries are used for visualization of data.

Summarizing the findings from the data analysis –

When the amount attribute is described using .describe() function in pandas library we found that most expenditures lie below $100. Minimum expenditure was 0.0 and maximum was $25691.160. Mean spending was $88.350.

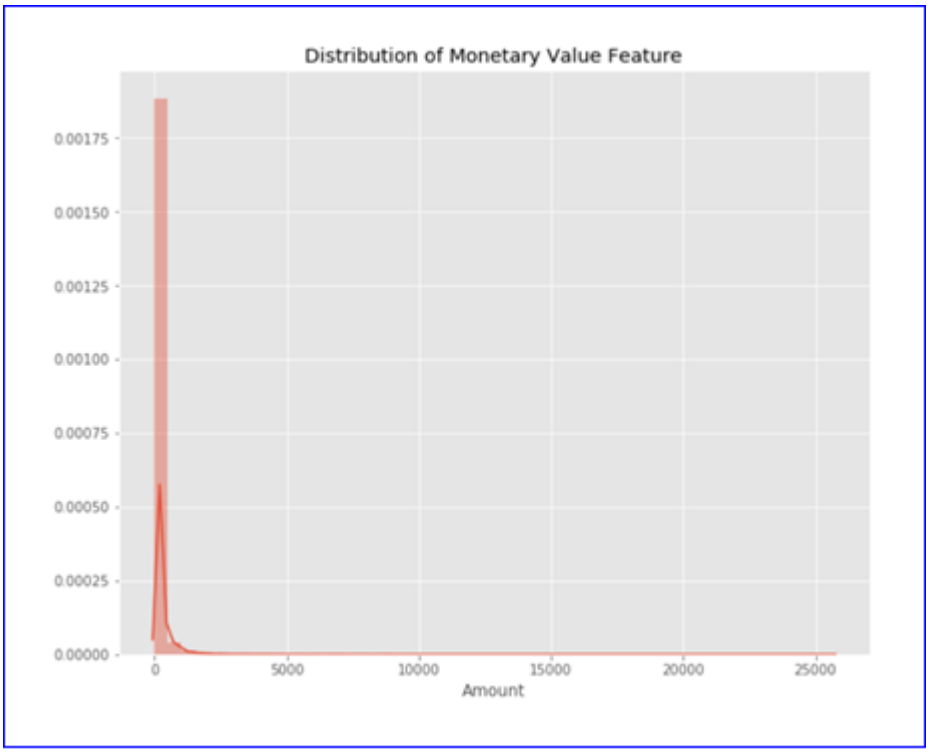


Figure 5.1.1: Distribution of Amount Feature

Plotting the graph for the time attribute and observing the frequency of transactions we came to the conclusion that customers did not spend much in the time frame of about 12 – 6 am.

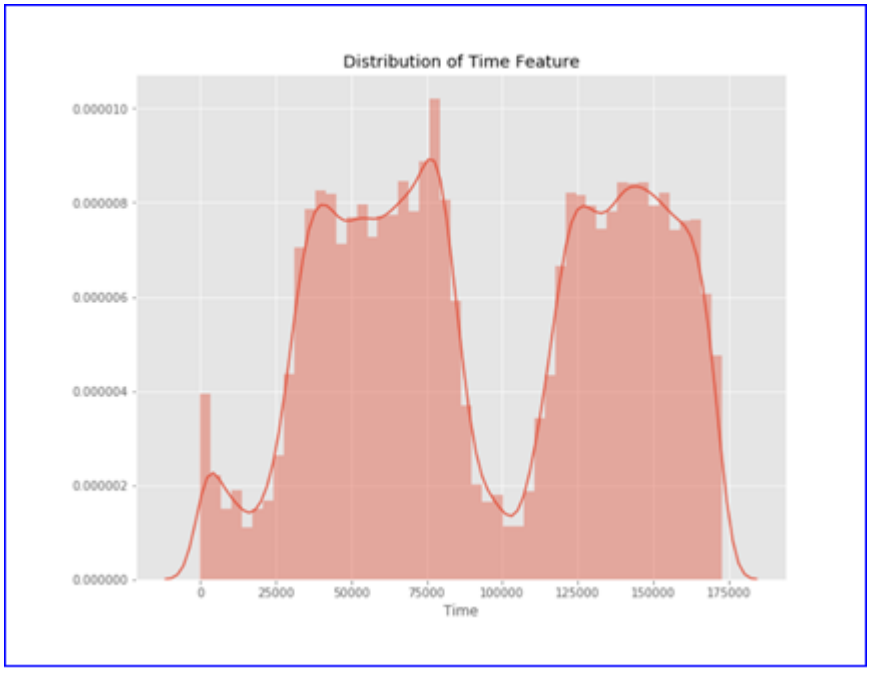


Figure 5.1.2: Distribution of Time Feature

At the end of the module correlation between each attribute towards the classes has been visualized. The findings of the plot is given in the figure.

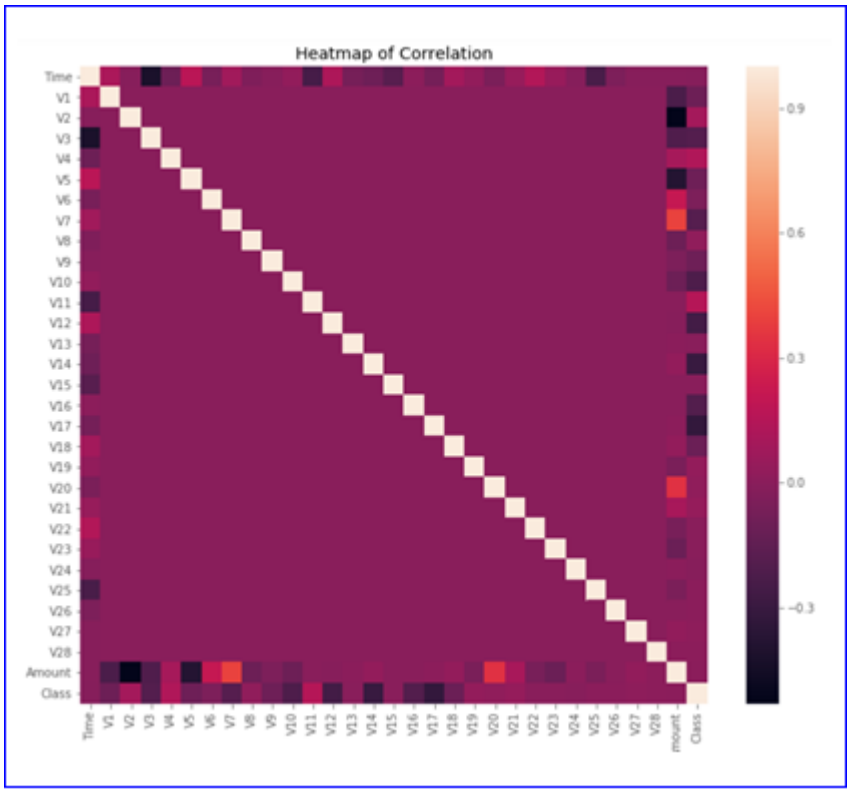


Figure 5.1.3: Heatmap of Correlation

From the figure we can conclude that attributes V3, V4, V9, V10, V11, V12, V14, V16, V17 has correlation greater than 0.5 and less than -0.5. Hence, they have greater effect on determination of our classes than other attributes.

2. FulllCode_DEV-V2.ipynb

In this workbook we are going to implement the core of this project. First, we are going to import all the libraries that are essential in order to do the implementation. The libraries that we are using are pandas, numpy, matplotlib, sklearn, seaborn and pickle. Our next job is to import the csv file into a dataframe using pandas. Once we do that, we will be able to represent the data properly and do the needed operations accordingly.

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V2
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454	0.111864	1.014480	-0.50934
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205	0.924384	0.012463	-1.01622
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045	0.578229	-0.037501	0.64013
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245	0.800049	-0.163298	0.12320
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057	0.643078	0.376777	0.00879

Figure 5.2.1: Data representation using pandas

After the dataset is being imported into a data frame, we need to follow the process of standardization. Standardization helps us to speed up the training process of a dataset by lowering the data range. If there is a drastic difference in the lowest and highest value in that feature then the training process slows down. In our dataset we are going to standardize the column 'Amount' which appears to have drastically high and low values. In order to do that we are going to use a package from the scikitlearn library called StandardScaler. We normalize the values of the Amount column and copy the normalized values to a new column in the data frame called 'normAmount'. One this is done; we drop the original 'Amount' column because now we are going to work with the normalized values.

V6	V7	V8	V9	V10	...	V21	V22	V23	V24	V25	V26	V27	V28	Class	normAmount
i2388	0.239599	0.098698	0.363787	0.090794	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	0	0.244964
i2361	-0.078803	0.085102	-0.255425	-0.166974	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	0	-0.342475
)0499	0.791461	0.247676	-1.514654	0.207643	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	0	1.160686
17203	0.237609	0.377436	-1.387024	-0.054952	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	0	0.140534
)5921	0.592941	-0.270533	0.817739	0.753074	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	0	-0.073403

Figure 5.2.2: Data frame with normAmount

Now we split the data set into two parts, one of them is going to be used for training purpose and other one is for testing. This split is done in order to show explicitly the difference between validation accuracy and the testing accuracy. We use 90% of the data frame for training the data set and calculating the validation accuracy and rest 10% is for calculating testing accuracy. The new data frame called 'data' is being used for training.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V21	V22	V23	
0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	...	-0.018307	0.277838	-0.110474	0.066
1	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	...	-0.225775	-0.638672	0.101288	-0.339
2	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	...	0.247998	0.771679	0.909412	-0.689
3	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	...	-0.108300	0.005274	-0.190321	-1.175
4	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	...	-0.009431	0.798278	-0.137458	0.141
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
256321	-1.994580	-1.635319	1.006695	-1.433183	3.259194	4.008239	-1.660020	1.099466	1.919685	-0.518544	...	-0.258971	0.080971	-0.579908	0.830
256322	1.919950	-0.529575	-0.291774	0.644989	-0.766757	-0.384374	-0.616882	-0.030380	1.250724	0.013274	...	0.239401	0.864998	0.078354	-0.109
256323	-0.356158	1.191407	-0.437619	1.173899	0.525368	-0.471524	0.199218	0.239205	-0.757000	-0.433462	...	-0.182842	-0.552938	0.033595	0.614
256324	2.025835	-0.046070	-1.249393	0.274412	-0.027561	-1.099714	0.154687	-0.263755	0.335313	0.185335	...	0.302148	1.003410	-0.000059	0.113
256325	-1.257880	0.577768	0.456237	-1.012853	-1.568841	0.939295	-2.090669	1.690625	-1.061729	0.100651	...	0.127880	0.390366	0.107342	0.030

Figure 5.2.3: Data frame for validation accuracy

We further split it into 70-30 percent values so as to train the model and get the validation accuracy calculated. But before that we need to split the data frame into input and output features. Once that is done, we see that the output feature ‘Class’ has only two values 0 and 1. The rows with the value 1 is much less than those with the value 0. So, we can say that the data frame is unbalanced and we need to apply some algorithm in order to balance the same. This is where SMOTE algorithm comes into play. According to this algorithm we can increase the number of minority classes by processing the feature values of the majority and the minority classes both.

```
sm = SMOTE(random_state=2) # using SMOTE
X_train_res, y_train_res = sm.fit_sample(X_train, y_train.ravel())

print('After OverSampling, the shape of train_X: {}'.format(X_train_res.shape))
print('After OverSampling, the shape of train_y: {} \n'.format(y_train_res.shape))
# now the major and minor classes gets balanced equally
print("After OverSampling, counts of label '1': {}".format(sum(y_train_res==1)))
print("After OverSampling, counts of label '0': {}".format(sum(y_train_res==0)))
```

```
After OverSampling, the shape of train_X: (358208, 29)
After OverSampling, the shape of train_y: (358208,)

After OverSampling, counts of label '1': 179104
After OverSampling, counts of label '0': 179104
```

Figure 5.2.4: SMOTE Analysis of the training data set

After this we are ready to train the model. We do so by using random forest classifier. Once the model is trained, we export it to a file using pickle module so that we don’t need to train the model again and again while showing the project. Now we use this trained module file to calculate the validation accuracy and confusion matrix.

```
Validation Accuracy is:0.9995448516216286
F1 Score is:0.8745519713261649
Precision is:0.9172932330827067
Recall is:0.8356164383561644
```

Figure 5.2.5: Validation Accuracies

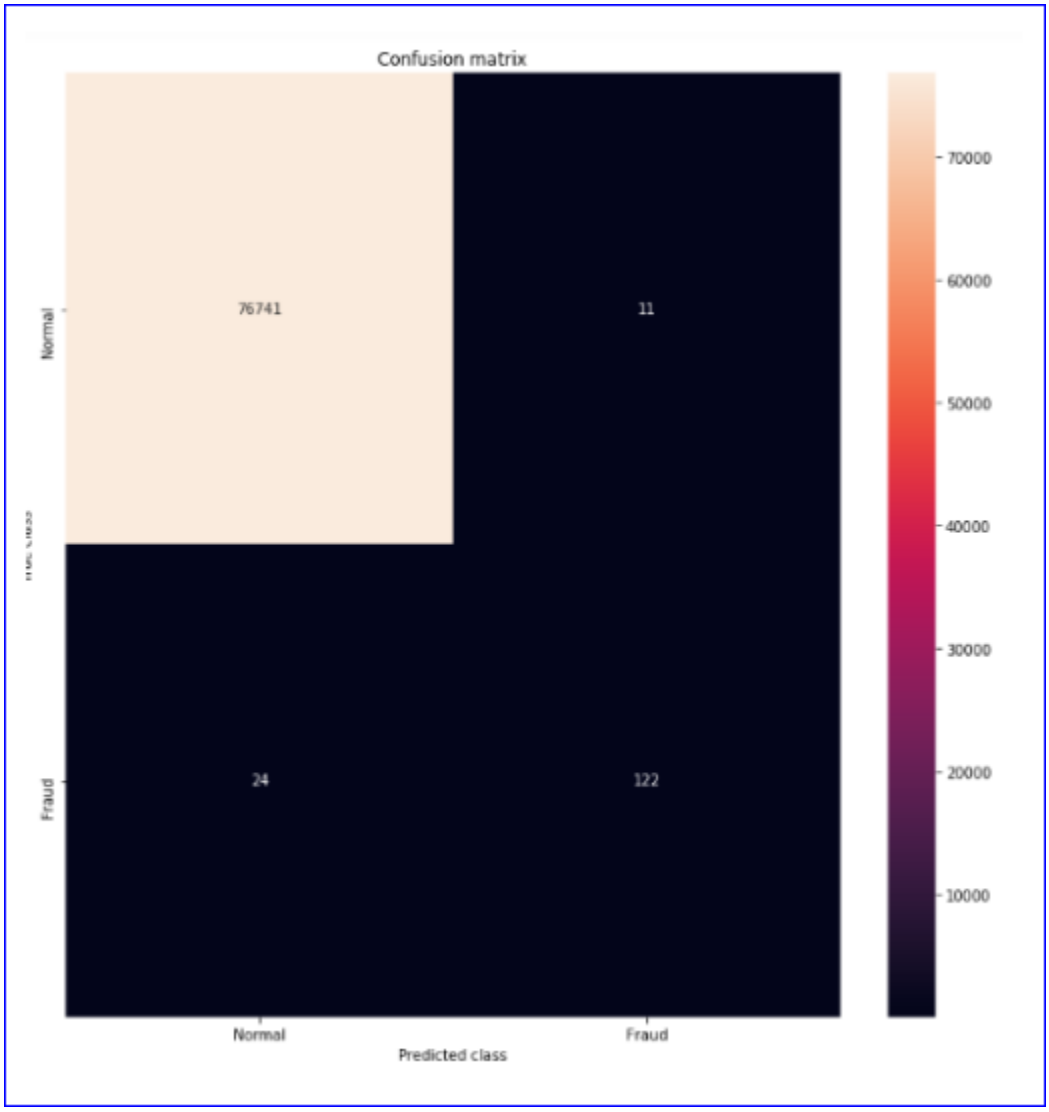


Figure 5.2.6: Confusion Matrix for Validation Accuracy

We calculate the same accuracy values again but this time we pass the data frame data_test to the model, thus calculating the testing accuracy.

```
Accuracy is:0.9996254856982351
F1 Score is:0.7777777777777778
Precision is:0.7368421052631579
Recall is:0.8235294117647058
```

Figure 5.2.7: Testing Accuracy Values

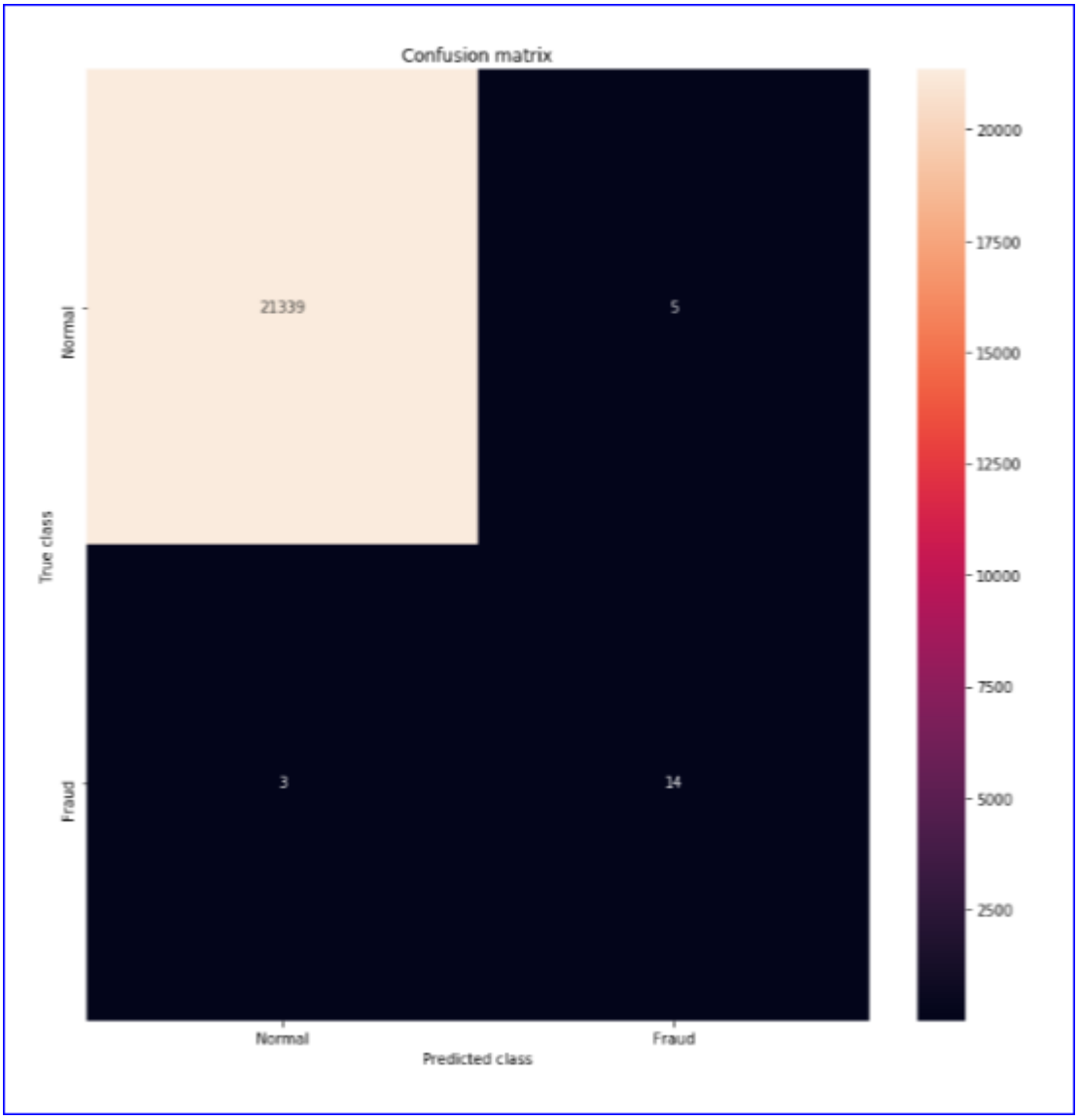


Figure 5.2.8: Confusion matrix for testing accuracy

3. SMOTE.ipynb

In this notebook we are showing the basic working of SMOTE algorithm. And for that also we need to standardize the values in the 'Amount' column. After doing that we apply SMOTE in two parts. First part is the nearest_neighbour function.

Nearest_neighbour function calculates or return "K" number of nearest neighbours from a particular data point. This is calculated using Euclidian distance among all data points from the given data point.

In our implementation we have created 100 "synthetic" or oversampled data points using the given formula - { x' = x + rand(0,1) * |x-x(k)| } where 'x(k)' are k nearest neighbors from 'x'. The newly

found data points or “synthetic” data points lie on the line between the two data points originally used. Like this many data points can be used to create desired number of synthetic data points for oversampling.

6. SYSTEM REQUIREMENTS

Hardware Requirements

- Processor (CPU) with 2 gigahertz (GHz) frequency or above.
- A minimum of 2 GB of RAM.
- Internet Connection Broadband (high-speed) Internet connection with a speed of 4 Mbps or higher.

Software Requirements

- OS supported: Windows/Linux/MacOS
- Web browser preinstalled to access the portal.
- The system is developed in Python.

7. SCHEDULE

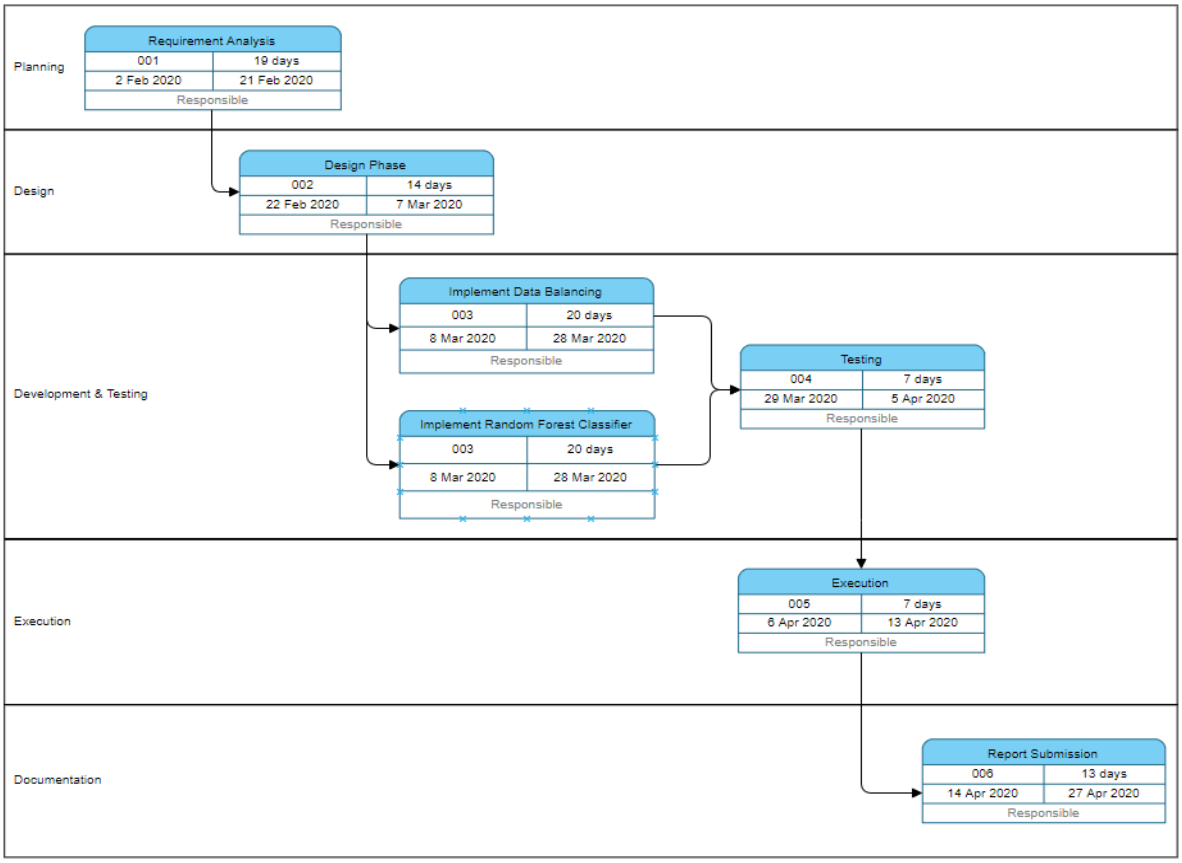


Figure 7.1: Pert Chart

8. CONCLUSION

Credit Card fraud detection is an intricate subject that requires a considerable proportion of groundwork before applying different machine learning algorithms at it. Besides that, it is also a critical application of data science ensuring that privacy and money of the customer is safe. This project demonstrated that fraud can be detected efficiently using Random Forest Classifier, which proves to be an effective algorithm for the problem in question. It works better with a large number of training data and the results obtained is 99.9% accurate.

9. FUTURE SCOPE

In the future we can improve our classifier so that can get close to the goal of 100% accuracy. In the future multiple algorithms can be amalgamated together and their results can be compounded to improve the overall accuracy of the system. Since the size of dataset directly influences the precision of algorithm, so with due support from the banks we can improve our system. Also, a data set with non-anonymized features would allow one to see what factors are the most critical in finding fraud. Also, the project can be applied to a distributed environment which can resolve issues relating to privacy.

10. REFERRENCES

[1] Yashvi Jain, NamrataTiwari, Shripriya Dubey,Sarika Jain, “A Comparative Analysis of Various Credit Card Frau8d Detection Techniques”, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7 Issue-5S2, January 2019.

[2] Devi Meenakshi. B1, Janani. B2, Gayathri. S3, Mrs. Indira. N, “Credit Card Fraud Detection using Random Forest Technique”, IRJET, Volume: 06 Issue: 03, March, 2019.

[3] B. Mohankumar, Dr. K. Karuppasamy, “Credit Card Fraud Detection Using Random Forest Technique”, IJIRSET, Vol. 8, Issue 4, April 2019.

[4] P. Pavithra, S. babu, PhD, “Data Mining Techniques for Handling Imbalanced Datasets: A Review”, IJSRED, Volume 2, Issue 3, 2018.

EMAIL APPROVAL:

SOURADEEP BANERJEE <500052778@stu.upes.ac.in> Thu, Apr 23, 2020 at 1:45 PM
To: "Dr. Jagdish Chandra Patni" <jcpatni@ddn.upes.ac.in>

Respected sir,

Good afternoon and greetings of the day!

I did the changes and below attached are the word and pdf files for the same.
I will format it in research paper format in the meantime and revert back to you in a couple of days.

Awaiting your approval.

Yours sincerely
Souradeep Banerjee

[Quoted text hidden]

2 attachments

Final Report.pdf
812K

Final Report.docx
1507K

Dr. Jagdish Chandra Patni <jcpatni@ddn.upes.ac.in> Thu, Apr 23, 2020 at 1:49 PM
To: SOURADEEP BANERJEE <500052778@stu.upes.ac.in>

Approved
Go ahead

Get Outlook for iOS