Group33_Lab1_report_final

May 12, 2020

0.1 CS4035 - Cyber Data Analytics (Lab 1 -Fraud Data)

0.2 Group details - Group 33

Shipra Sharma - 5093406
Sudharshan Swaminathan - 5148340

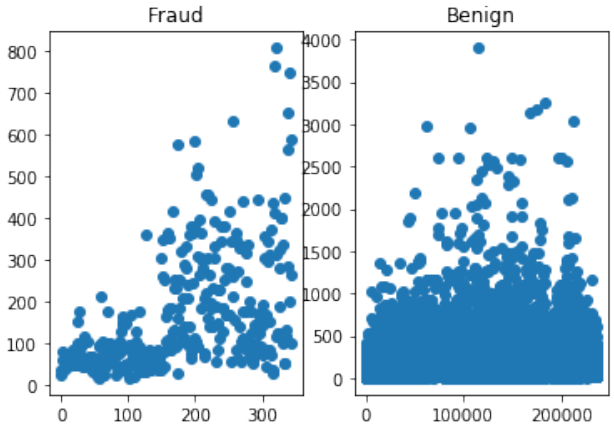
0.2.1 Readme (setup instructions)

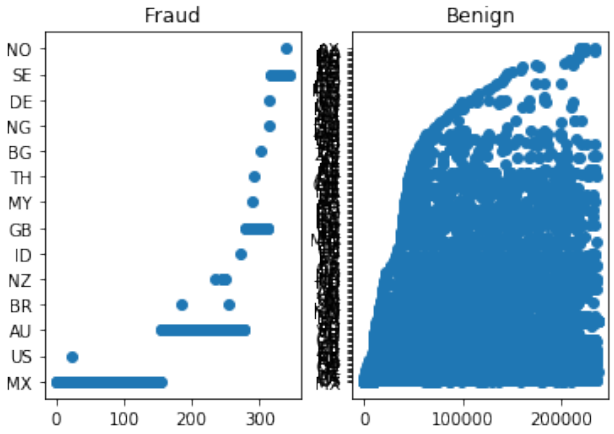
We installed two explicit modules named `scikit-learn` and `imblearn`. This could be done using `pip install` command in your python environment. Apart from these, all the functions and objects needed are included in the import statements. The other modules that are being imported are `numpy`, `matplotlib`, and `pandas`. These should have already been installed on your system while doing the assignment.

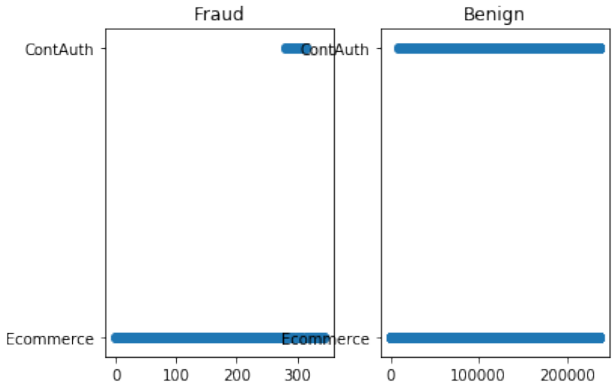
Import statements Following inbuilt functions/objects have been used to implement the requiremets.

Data Preprocessing We load the raw data in our dataframe and drop booking date and txid due to less relevance. We also drop the rows corresponding to the refused value of simple_journal as it creates ambiguity whether a transaction is fraud or non-fraud (benign). We then adjust the data type of a few of the columns like creationdate is converted to date-time format for panda to read it. Also, in the preprocessing step we create two labels for the class i.e 0 and 1 for benign and fraud respectively and make corresponding dataframes. At last we convert the amount into standardised USD amount and create a new column corresponding to it.

0.2.2 1. Visualization task - plots and respective analysis





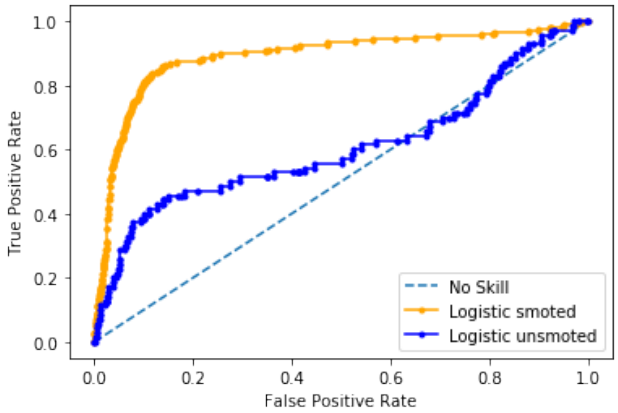


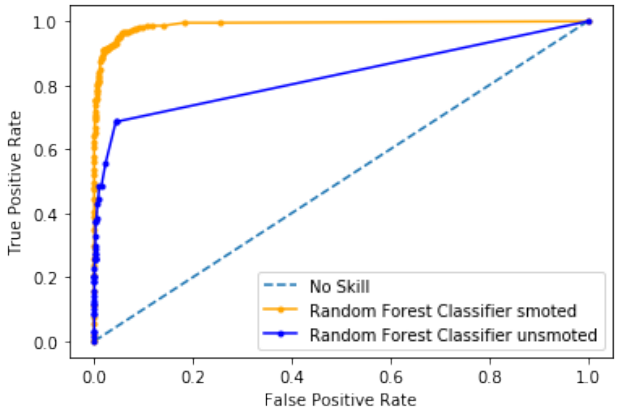
1. For the first plot it can be seen that the fraud transcations are made on smaller amount as

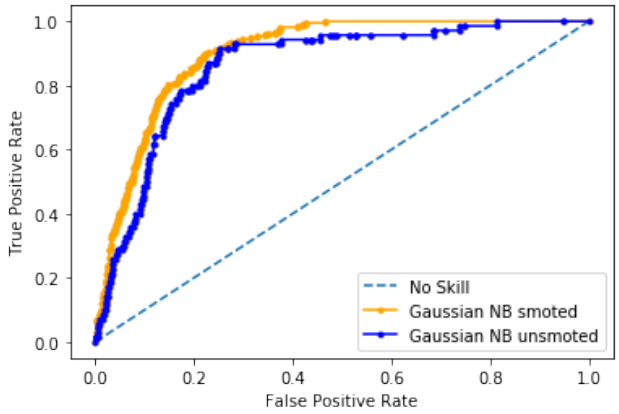
compared to the benign transactions showing that criminals generally target smaller amount so that the transcation attracts minor attention.
2. The second visual shows that the fraud transactions are mostly happening in three countries: MX, AU and GB. Thus, more attention can be invested here to prevent such cases.
3. The third visual gives a relation between Point of Sale and fraud activities. The online mode (ECommerce) attracts more frauds as compared to ContAuth (recurring events like monthly subscriptions).

0.2.3 2. Imbalance task - Sudharshan Swaminathan

1. Define smoting function as per the algorithm given in the paper - SMOTE: Synthetic Minority Over-sampling Technique
2. Apply Smoting for three classifiers - Logistic Regression, Random Forest and Gaussian NB
3. Creating ROC curves for smoted and unsmoted data
4. Analyse the performace



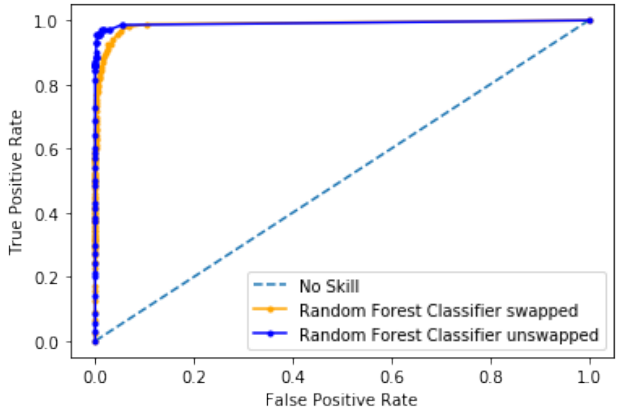


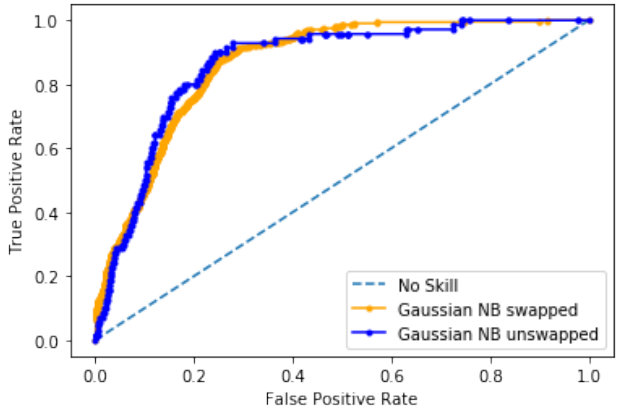


Classifiers Performance We compare Logistic Regression, Random Forests and Naive Bayes classifiers on both smoted and unsmoted data. We observe that Random Forests overall perform the best (with Smoted data). When working with Smoted data, it has the least False Positive rate and the maximum True Positive rate, and also having the maximum AUC for ROC. This goes to show that this classifier performs with relatively better accuracy than other classifiers working with Smoted data. Random Forests and Logisitic Regression give better results with Smoted data, while Gaussian NB gives almost the same results for Smoted and unsmoted data. With unsmoted data however, Gaussian Naive Bayes seems to perform better than the other 2 in terms of AUC, but has higher false positive rate than Random Forests. Logistic Regression performs the least with unsmoted data with the highest false positive rate. #### Is SMOTE a good Idea Considering that false positives are reduced when using Smote-ing data for all the 3 classifiers used above and also increases the true positive rates for 2 of them, we can say that Smote-ing the data gives a better dataset to work with while performing classification. Therefore, smote-ing (oversampling) along with undersampling is efficient while preparing the training data set and enables us to build a better classifier.

0.2.4 3. Privacy task – Shipra Sharma

1. Define rank swapping function for identifying(numerical) data i.e. for the following columns: bin, usd_amount, mail_id, ip_id, card_id. In order to do this we first sort the entire data list as per a column and then swap the row values within a restricted range (p) of the concerned column. We repeat this for the given 5 columns.
2. Apply rank swapping on smoted data for three classifiers - Logistic Regression, Random Forest and Gaussian NB
3. Creating ROC curves for smoted and unsmoted data
4. Analyse the performace





Classifiers Performance We compare Logistic Regression, Random Forests and Naive Bayes classifiers on both swapped and unswapped data. We observe that Random Forests overall per-

forms the best as it has the lowest False Positive Rate i.e less benign classes are labeled as fraud, however, the true positive rate is somewhat same for all the three classifiers. We also performed the rank swapping on unsmotted data and found that it is less efficient as compared to the smoted data and hence, we include the same in our work. Logistic Regression classifier seems to perform the worst out of the lot with highest false positive rate for both swapped and unswapped data, comparatively less true positive rate and least area under the curve. GaussianNB performs better than logistic regression. Random Forest works the best in general because of the tree structure that it uses internally. The bagging and features randomness helps to create random trees with uncorrelated forest of trees and in the given dataset there are multiple features like shoppercountrycode, amount, etc. which give predictive power to these trees, and even if some of the trees are wrong, they vote as one and the prediction of the whole group is elected which in turn makes the whole classifier perform well and make good predictions. #### Is Rank-Swapping a good idea As per our analysis **Rank swapping is a good idea** because from our results we can see that the performaces of the classifiers are not compromised while using swapped data. Only in case of the Random Forest classifier the true positive rate drops as compared to the unswapped data but in Logistic Regression and GaussianNB the TPR and FPR are marginally close in both the cases. Hence, we get similar performance even when the data is swapped that is data **privacy** is enhanced as the identifying (numerical) data is thoroughly shuffled throughout the dataset.

0.2.5 4. Classification task

As required, the classification is performed by two classifiers and these classifiers are evaluated using a 10 cross validation in creating different scores like Precision, Recall, True Positives, False Positives, True Negatives, False Negatives, Accuracy, etc.

Firstly a subset of the data is made considering only the most relevant columns from the orginal dataset (most of these are categorical data). Then we build a function to implement 10-cross validator and a fucntion to calculate the most weighted features that lead to a prediction of fraud class (this is used only in white box algorithm).

The training is done using smoted Data for both the classifiers and the test is performed on a part of the original data(unsmoted).

- Black Box Classification using **Random Forest**

We choose Random Forest as a black box classifier because in this task a good performing classifier is needed and as explained in the above section that the random forest generally peforms better due to the tree structure it creates internally because of which the predictions turn out to be good. This internal structure becomes our second reason for using random forest as a black box classifier, because the complex internal structure might be difficult to explain and the requirement gives us a leverage to focus only on performace in this task. During the implementation the maximum depth of the tree is taken to be 6 as that helped us to reduce the number of false positives in the model's predcition as more depth of the tree allows more split and capturing of the data.

Despite of tuning multiple 'k' values (ranging between 10-100) of the smote function, the least false positives that we could get are 2615. However, the algorithm identifies more that 100 true positives. We observed that the increase in 'k' lead to more true positives but also more false positives and vice versa. This might be because larger 'k' leads to more synthetic data generation and thus, both FP and TP increases.

- White Box Classification using **Logistic Regression**

The task focuses on paying more attention to explaining why a particular transaction is labeled as fraud and not on the accuracy. We use logistic regression because our label to define fraud data or benign data is a binary parameter and Logistic Regression can be used when dealing with binary classification. The algorithm uses maximum-likelihood estimation to optimize weight from the training data. In the given data, it is quite easy to identify whether the transaction is fraud or not depending on weights of several parameters. For example, in the visualisations above we saw that certain columns like shoppercountrycode, shopperinteraction easily give away the fraudulent transaction. Thus, weight metrix of such parameters can be easily obtained using LR which makes the analysing task comparatively easier.

In the result we see that the logistic classifier gave less true positives as compared to the random forest classifier, however calculates less number of false positives and more number of true negatives. From both the results we can conclude that random forest works better in fraud detection task.

[49]:
```
array([[ 0.02040877, -0.08829829,  0.00683702, -0.31892825,  0.00602958,
        -0.72354173,  0.        , -3.35394242, -0.25586805]])
```

We see that for the feature coefficients, the logistic regression classifier produces positive weights for issuercountrycode, usd_amount and shoppercountrycode, which could be closer parameters to identify a fraud and the others like cvcresponsecode, accountcode, etc. are negative, hence being closer to the identification of benign data.

0.2.6 5. Bonus task

To make the fraud detection better we tried establishing a relation between the fraud and amount deducted per card per month. It is a derived feature that can help us to etablish a relatonship that how many times a card is used in fraudulent transaction in a given month (monthly behaviour of a card involved in fraudulent transactions).

Please Note: This takes a lot of time in execution because the aggregation function goes through each row of the entire dataset multiple times. Hence, we coould not come to a proper calculation/output of our aggregator function.