Credit Card Fraud Detection

Under the guidance of Dr.Cathy Durso

Probability and

Statistics [4441-1]

Contents

Introduction

What is credit card fraud? When someone uses your credit card to buy goods & services or access your personal account without consent is called credit card fraud. In the European Union the credit card fraud in 2013 was approximately €1.44 Billion.

Types of credit card fraud: Some common types of credit card frauds are:

• Card-not-present • Counterfeit credit-card • Account or application hack

With the advent of new technology, fraudsters find new ways to scam people and so it is important to learn the signs and act quickly to report suspected frauds.

How to stop credit card fraud? There is a saying 'Set a thief to catch a thief', meaning that the best way to catch a thief is to with the help of another thief because both think alike. Hence, to tune thinking like a thief we have tried to implement machine learning models to learn to identify patterns and anomalies of fraudulent transactions from a large data set and flag such transactions in the future.

Data Source

For our project we have chosen an open-source date-set from Kaggle : https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Overview of Dataset

The data set contains a total of 31 variables, seen below, and 284,807 row entries. The data has already been PCA transformed (Dimensionality Reduction), however due to confidentiality issue a lot of the variable names have been masked. As seen below are listed the headers of the variables. Variables "Time" through "Amount" are all dependent variables and "Class" variable is the only dependent variable.

The "Class" dependent variable is labeled "0" for non-fraud transactions and "1" for fraudulent transactions. All the dependent variables are numeric, and the structure of the data can be seen below.

```
[1] "Time"   "V1"     "V2"     "V3"     "V4"     "V5"     "V6"     "V7"
 [9] "V8"     "V9"     "V10"    "V11"    "V12"    "V13"    "V14"    "V15"
[17] "V16"    "V17"    "V18"    "V19"    "V20"    "V21"    "V22"    "V23"
[25] "V24"    "V25"    "V26"    "V27"    "V28"    "Amount" "Class"

'data.frame':   284807 obs. of  31 variables:
 $ Time  : num  0 0 1 1 2 2 4 7 7 9 ...
 $ V1    : num  -1.36 1.192 -1.358 -0.966 -1.158 ...
 $ V2    : num  -0.0728 0.2662 -1.3402 -0.1852 0.8777 ...
 $ V3    : num  2.536 0.166 1.773 1.793 1.549 ...
 $ V4    : num  1.378 0.448 0.38 -0.863 0.403 ...
 $ V5    : num  -0.3383 0.06 -0.5032 -0.0103 -0.4072 ...
 $ V6    : num  0.4624 -0.0824 1.8005 1.2472 0.0959 ...
```

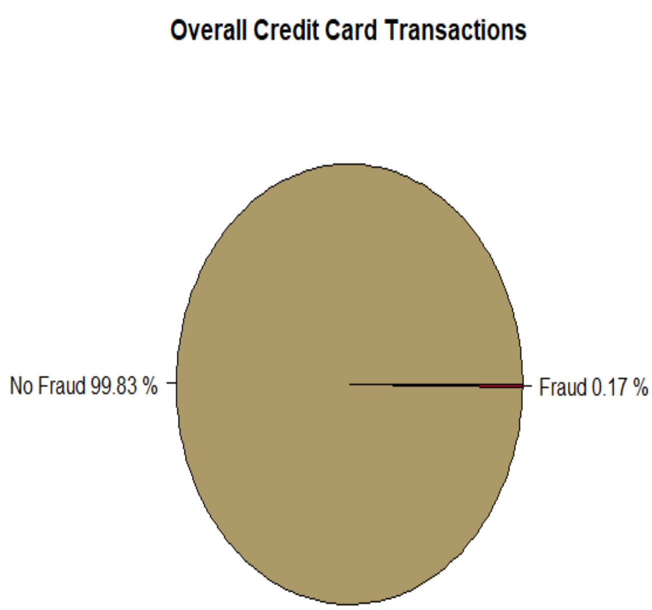
```
 $ V7    : num  0.2396 -0.0788 0.7915 0.2376 0.5929 ...
 $ V8    : num  0.0987 0.0851 0.2477 0.3774 -0.2705 ...
 $ V9    : num  0.364 -0.255 -1.515 -1.387 0.818 ...
 $ V10   : num  0.0908 -0.167 0.2076 -0.055 0.7531 ...
 $ V11   : num  -0.552 1.613 0.625 -0.226 -0.823 ...
 $ V12   : num  -0.6178 1.0652 0.0661 0.1782 0.5382 ...
 $ V13   : num  -0.991 0.489 0.717 0.508 1.346 ...
 $ V14   : num  -0.311 -0.144 -0.166 -0.288 -1.12 ...
 $ V15   : num  1.468 0.636 2.346 -0.631 0.175 ...
 $ V16   : num  -0.47 0.464 -2.89 -1.06 -0.451 ...
 $ V17   : num  0.208 -0.115 1.11 -0.684 -0.237 ...
 $ V18   : num  0.0258 -0.1834 -0.1214 1.9658 -0.0382 ...
 $ V19   : num  0.404 -0.146 -2.262 -1.233 0.803 ...
 $ V20   : num  0.2514 -0.0691 0.525 -0.208 0.4085 ...
 $ V21   : num  -0.01831 -0.22578 0.248 -0.1083 -0.00943 ...
 $ V22   : num  0.27784 -0.63867 0.77168 0.00527 0.79828 ...
 $ V23   : num  -0.11 0.101 0.909 -0.19 -0.137 ...
 $ V24   : num  0.0669 -0.3398 -0.6893 -1.1756 0.1413 ...
 $ V25   : num  0.129 0.167 -0.328 0.647 -0.206 ...
 $ V26   : num  -0.189 0.126 -0.139 -0.222 0.502 ...
 $ V27   : num  0.13356 -0.00898 -0.05535 0.06272 0.21942 ...
 $ V28   : num  -0.0211 0.0147 -0.0598 0.0615 0.2152 ...
 $ Amount: num  149.62 2.69 378.66 123.5 69.99 ...
 $ Class : int  0 0 0 0 0 0 0 0 0 0 ...
```

To give a better understanding of the data we are working with, we look at the first 6 rows from the data set.

```
Time         V1          V2        V3         V4          V5          V6
1    0 -1.3598071 -0.07278117 2.5363467  1.3781552 -0.33832077  0.46238778
2    0  1.1918571  0.26615071 0.1664801  0.4481541  0.06001765 -0.08236081
3    1 -1.3583541 -1.34016307 1.7732093  0.3797796 -0.50319813  1.80049938
4    1 -0.9662717 -0.18522601 1.7929933 -0.8632913 -0.01030888  1.24720317
5    2 -1.1582331  0.87773675 1.5487178  0.4030339 -0.40719338  0.09592146
6    2 -0.4259659  0.96052304 1.1411093 -0.1682521  0.42098688 -0.02972755
          V7          V8         V9         V10        V11         V12
1  0.23959855  0.09869790  0.3637870  0.09079417 -0.5515995 -0.61780086
2 -0.07880298  0.08510165 -0.2554251 -0.16697441  1.6127267  1.06523531
3  0.79146096  0.24767579 -1.5146543  0.20764287  0.6245015  0.06608369
4  0.23760894  0.37743587 -1.3870241 -0.05495192 -0.2264873  0.17822823
5  0.59294075 -0.27053268  0.8177393  0.75307443 -0.8228429  0.53819555
6  0.47620095  0.26031433 -0.5686714 -0.37140720  1.3412620  0.35989384
         V13        V14        V15        V16         V17         V18
1 -0.9913898 -0.3111694  1.4681770 -0.4704005  0.20797124  0.02579058
2  0.4890950 -0.1437723  0.6355581  0.4639170 -0.11480466 -0.18336127
3  0.7172927 -0.1659459  2.3458649 -2.8900832  1.10996938 -0.12135931
4  0.5077569 -0.2879237 -0.6314181 -1.0596472 -0.68409279  1.96577500
5  1.3458516 -1.1196698  0.1751211 -0.4514492 -0.23703324 -0.03819479
6 -0.3580907 -0.1371337  0.5176168  0.4017259 -0.05813282  0.06865315
          V19         V20          V21          V22         V23         V24
1  0.40399296  0.25141210 -0.018306778  0.277837576 -0.11047391  0.06692807
```

```
2 -0.14578304 -0.06908314 -0.225775248 -0.638671953  0.10128802 -0.33984648
3 -2.26185710  0.52497973  0.247998153  0.771679402  0.90941226 -0.68928096
4 -1.23262197 -0.20803778 -0.108300452  0.005273597 -0.19032052 -1.17557533
5  0.80348692  0.40854236 -0.009430697  0.798278495 -0.13745808  0.14126698
6 -0.03319379  0.08496767 -0.208253515 -0.559824796 -0.02639767 -0.37142658
         V25        V26          V27         V28 Amount Class
1  0.1285394 -0.1891148  0.133558377 -0.02105305 149.62     0
2  0.1671704  0.1258945 -0.008983099  0.01472417   2.69     0
3 -0.3276418 -0.1390966 -0.055352794 -0.05975184 378.66     0
4  0.6473760 -0.2219288  0.062722849  0.06145763 123.50     0
5 -0.2060096  0.5022922  0.219422230  0.21515315  69.99     0
6 -0.2327938  0.1059148  0.253844225  0.08108026   3.67     0
```

The data set is an unbalanced data set i.e., we have only 492 fraudulent transactions of the total 284,807 transactions that is less than 0.2% of the data set. We can visualize this from the pie chart given below:



Summary Statistics

The summary statistics are shown below:

```
Time                V1                   V2                   V3
 Min.   :     0   Min.   :-56.40751   Min.   :-72.71573   Min.   :-48.3256
 1st Qu.: 54202   1st Qu.: -0.92037   1st Qu.: -0.59855   1st Qu.: -0.8904
 Median : 84692   Median :  0.01811   Median :  0.06549   Median :  0.1799
 Mean   : 94814   Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.0000
 3rd Qu.:139321   3rd Qu.:  1.31564   3rd Qu.:  0.80372   3rd Qu.:  1.0272
 Max.   :172792   Max.   :  2.45493   Max.   : 22.05773   Max.   :  9.3826
       V4                V5                   V6                 V7
 Min.   :-5.68317   Min.   :-113.74331   Min.   :-26.1605   Min.   :-43.5572
 1st Qu.:-0.84864   1st Qu.:  -0.69160   1st Qu.: -0.7683   1st Qu.: -0.5541
 Median :-0.01985   Median :  -0.05434   Median : -0.2742   Median :  0.0401
 Mean   : 0.00000   Mean   :   0.00000   Mean   :  0.0000   Mean   :  0.0000
 3rd Qu.: 0.74334   3rd Qu.:   0.61193   3rd Qu.:  0.3986   3rd Qu.:  0.5704
 Max.   :16.87534   Max.   :  34.80167   Max.   : 73.3016   Max.   :120.5895
       V8                  V9                  V10                 V11
 Min.   :-73.21672   Min.   :-13.43407   Min.   :-24.58826   Min.   :-4.79747
 1st Qu.: -0.20863   1st Qu.: -0.64310   1st Qu.: -0.53543   1st Qu.:-0.76249
 Median :  0.02236   Median : -0.05143   Median : -0.09292   Median :-0.03276
 Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.00000
 3rd Qu.:  0.32735   3rd Qu.:  0.59714   3rd Qu.:  0.45392   3rd Qu.: 0.73959
 Max.   : 20.00721   Max.   : 15.59500   Max.   : 23.74514   Max.   :12.01891
      V12                V13                 V14                V15
 Min.   :-18.6837   Min.   :-5.79188   Min.   :-19.2143   Min.   :-4.49894
 1st Qu.: -0.4056   1st Qu.:-0.64854   1st Qu.: -0.4256   1st Qu.:-0.58288
 Median :  0.1400   Median :-0.01357   Median :  0.0506   Median : 0.04807
 Mean   :  0.0000   Mean   : 0.00000   Mean   :  0.0000   Mean   : 0.00000
 3rd Qu.:  0.6182   3rd Qu.: 0.66251   3rd Qu.:  0.4931   3rd Qu.: 0.64882
 Max.   :  7.8484   Max.   : 7.12688   Max.   : 10.5268   Max.   : 8.87774
      V16                 V17                 V18
 Min.   :-14.12985   Min.   :-25.16280   Min.   :-9.498746
 1st Qu.: -0.46804   1st Qu.: -0.48375   1st Qu.:-0.498850
 Median :  0.06641   Median : -0.06568   Median :-0.003636
 Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.000000
 3rd Qu.:  0.52330   3rd Qu.:  0.39968   3rd Qu.: 0.500807
 Max.   : 17.31511   Max.   :  9.25353   Max.   : 5.041069
      V19                 V20                 V21
 Min.   :-7.213527   Min.   :-54.49772   Min.   :-34.83038
 1st Qu.:-0.456299   1st Qu.: -0.21172   1st Qu.: -0.22839
 Median : 0.003735   Median : -0.06248   Median : -0.02945
 Mean   : 0.000000   Mean   :  0.00000   Mean   :  0.00000
 3rd Qu.: 0.458949   3rd Qu.:  0.13304   3rd Qu.:  0.18638
 Max.   : 5.591971   Max.   : 39.42090   Max.   : 27.20284
      V22                  V23                 V24
 Min.   :-10.933144   Min.   :-44.80774   Min.   :-2.83663
 1st Qu.: -0.542350   1st Qu.: -0.16185   1st Qu.:-0.35459
 Median :  0.006782   Median : -0.01119   Median : 0.04098
 Mean   :  0.000000   Mean   :  0.00000   Mean   : 0.00000
 3rd Qu.:  0.528554   3rd Qu.:  0.14764   3rd Qu.: 0.43953
```

```
Max.   : 10.503090   Max.   : 22.52841   Max.   : 4.58455  
     V25                 V26                 V27            
Min.   :-10.29540   Min.   :-2.60455   Min.   :-22.565679  
1st Qu.: -0.31715   1st Qu.:-0.32698   1st Qu.: -0.070840  
Median :  0.01659   Median :-0.05214   Median :  0.001342  
Mean   :  0.00000   Mean   : 0.00000   Mean   :  0.000000  
3rd Qu.:  0.35072   3rd Qu.: 0.24095   3rd Qu.:  0.091045  
Max.   :  7.51959   Max.   : 3.51735   Max.   : 31.612198  
     V28                Amount              Class         
Min.   :-15.43008   Min.   :    0.00   Min.   :0.000000  
1st Qu.: -0.05296   1st Qu.:    5.60   1st Qu.:0.000000  
Median :  0.01124   Median :   22.00   Median :0.000000  
Mean   :  0.00000   Mean   :   88.35   Mean   :0.001728  
3rd Qu.:  0.07828   3rd Qu.:   77.17   3rd Qu.:0.000000  
Max.   : 33.84781   Max.   :25691.16   Max.   :1.000000  
```

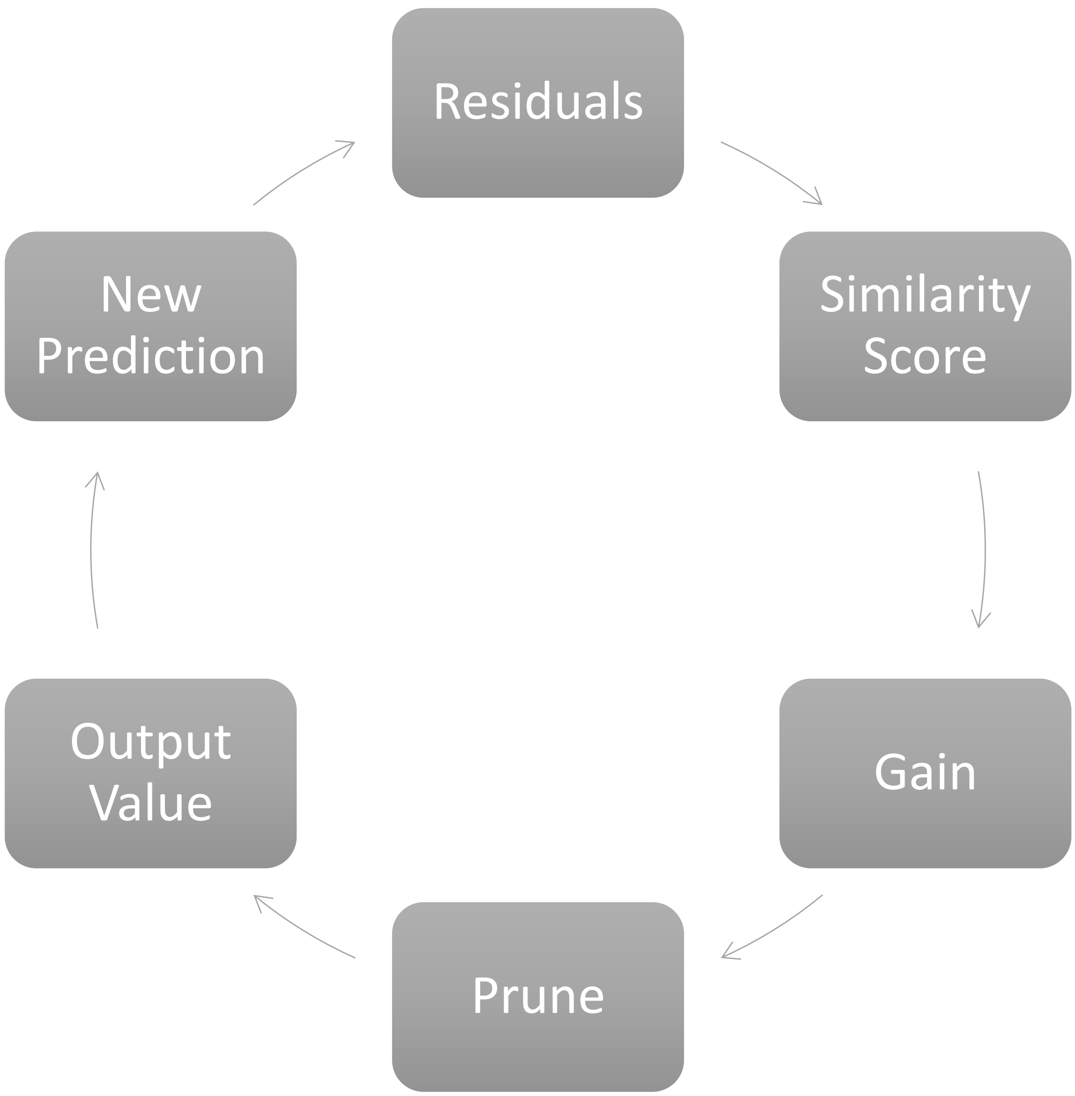
Research

Research Questions

1) Implement XGBoost and Logistic Regression (for classification) models to predict fraudulent transactions.
2) Compare the accuracy of the above models.

Methods for addressing research questions.

1) XGBoost (Classification) - eXtreme Gradient Boost a.k.a XGBoost is a regularized form of gradient boosting. The tool can be used for regression as well as classification. In this project we use the classification method of XGBoost to classify fraudulent transactions from non-fraudulent transactions.



The steps involved in XGBoost are as follows:

a) Initial prediction - this is usually 0.5 be it for regression or classification.

b) Similarity Score - this step is a complex step that includes calculating the residuals and then plugging in residual values in the similarity scores formula. We do this for all the leaves combinations i.e. different thresholds. Note: this is iterated until there is only one residual in the Tree or we have achieved tree depth, which is 6 by default.

$$SimilarityScore = \frac{\sum(Residual_i)^2}{\sum[PreviousProbability_i * (1 - PreviousProbability_i)] + \lambda}$$

c) Gain - To check the clustering of the XGBoost tree, the threshold that gives a higher gain will be used as a branch in the XGBoost tree. Note: this is iterated until there is only one residual in the Tree, or we have achieved tree depth, which is 6 by default.

$$Gain = Left_{similarity} + Right_{similarity} - Root_{similarity}$$

d) Prune - This is basically cutting the leaves of the tree; the pruning is done based on the gamma value. The gamma value is 0.5 by default. If the difference in gain and gamma is negative, we prune the leaves else leave them as it is.

$$Pruning = Gain - \gamma$$

e) Output value - After the tree formed, we then calculate the output value with the same lambda as in the similarity score. Note: Lambda is a regularization parameter that reduces the sensitivity of the prediction to isolated observations.

$$Output\ Value = \frac{\sum(Residual_i)}{\sum[PreviousProbability_i * (1 - PreviousProbability_i)] + \lambda}$$

f) New Prediction - Calculated by using the old prediction value, learning rate (eta, with 0.3 as default value) and the output value. The new prediction residual will be smaller than the residual from the old prediction value.

$$log(odds)\ of\ New\ Prediction = \log(odds)_{old\ prediction} + \left(\epsilon * (Output\ Value)\right)$$

$$New\ Prediction = \frac{e^{log(odds)}}{1 + e^{log(odds)}}$$

The above steps are iterate until the residuals become very minute or we reach the maximum number of trees.

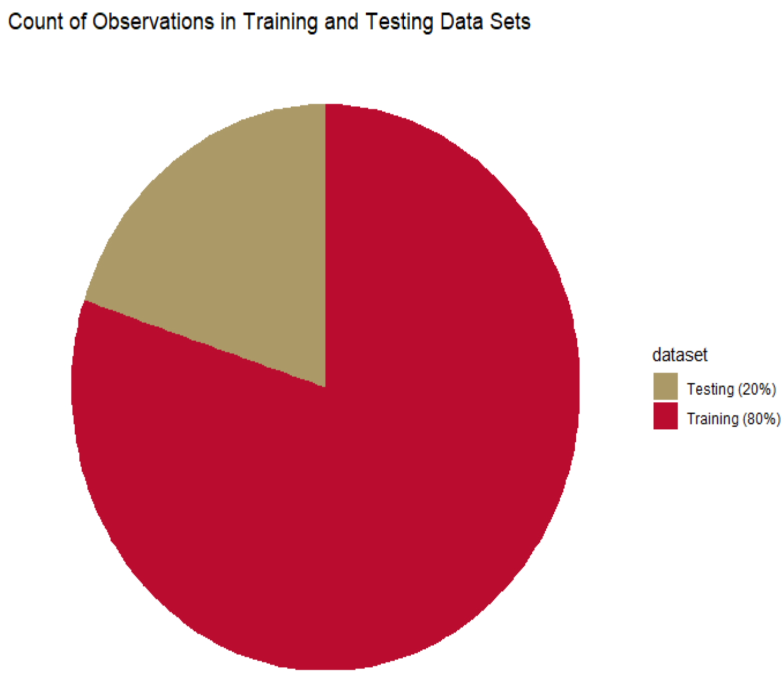
2) Logistic Regression (Classification) - This is like liner regression, but we only use this for classification based on our prediction. The default prediction value is 0.5. We fit the line using maximum likelihood i.e., the line is shifted to evaluate the likelihood and the line with the maximum likelihood is selected.

Importance of the project

1) Proof of concept for XGBoost classification.
2) Model predictor variable as a function of thirty dependent variables to automatically predict fraudulent transactions accurately.
3) Comparing accuracy of models in predicting the fraudulent transactions.
4) Analyzing significant variables contributing to the predictor variable.

Data Satisfaction

To carry out analysis there are two important pre-requisites. First, the data must all be numeric. Second, the data needs to be split into Training and Testing sets. The training set will comprise of 80% of the data and will be used to train the machine learning models whereas the testing data will be used to predict the outcome of the “Class” column i.e. 0 for non-fraud transaction and 1 for fraudulent transaction. The testing data is split into two testing sets, one stores the ‘Class’ variable (‘dat.testc’) that will be compared to the other testing data set that will not contain the class variable initially but will be used to predict the transaction in testc under ‘Predicted’ column.



Method Applied and Interpretation

XGBoost

First, we implement the XGBoost (classification) method. XGBoost only accepts matrix as input so we pass the training data set 'dat.train' with the class variable to train the model. The parameters used in the xgboost model are default values such as the eta = 0.3, gamma = 0.5, max_depth = 6. We did try to tweak the values however we found that the model worked best on these values, giving the maximum accuracy.

Visualization

Confusion Matrix

To compare the outcome

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56860    17
         1     3    81

               Accuracy : 0.9996
                 95% CI : (0.9995, 0.9998)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.8899

 Mcnemar's Test P-Value : 0.00365

            Sensitivity : 0.9999
            Specificity : 0.8265
         Pos Pred Value : 0.9997
         Neg Pred Value : 0.9643
             Prevalence : 0.9983
         Detection Rate : 0.9982
   Detection Prevalence : 0.9985
      Balanced Accuracy : 0.9132

       'Positive' Class : 0
```

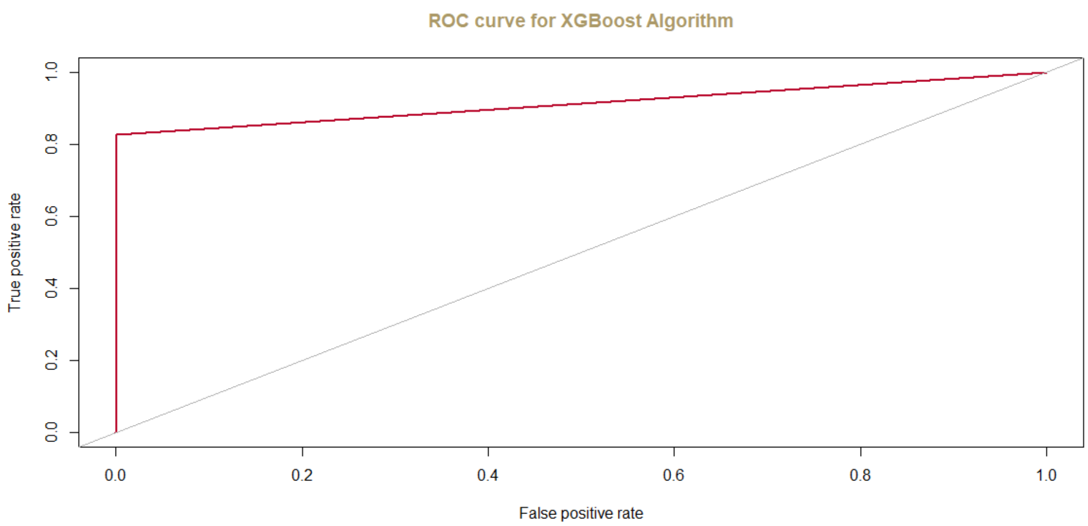
The testing data set contained 56,961 entries. As seen from the confusion matrix above, the XGBoost model for classification correct identified 99.96% of the transactions, correctly identifying 81 fraudulent transactions and incorrectly marking only 20 transactions. In the incorrect transactions the model incorrectly identified 17 fraudulent transactions as non-fraudulent and 3 non-fraudulent transactions as fraudulent.

Kappa is essentially interrater reliability testing, measure of agreement between the predicted labels and the true labels, and it considers the possibility of agreement occurring by chance. A high Kappa value of

0.8899 means that the classification of this data was not by chance and that the result has almost perfect agreement.

ROC Curve

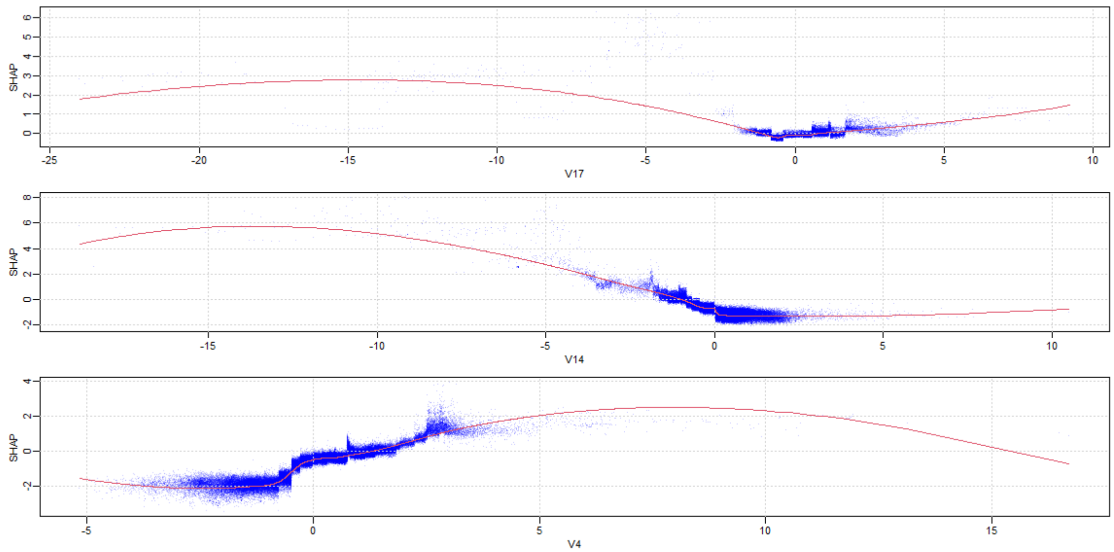
The Receiver Operating Characteristic Curve a.k.a ROC Curve, is a graph showing the classification performance of a model at different classification thresholds. The false positive is along the x-axis and the true positives are plotted against the y-axis, and essentially shows the trade-off between clinical sensitivity and specificity. The Area Under the ROC Curve a.k.a AUC provides an cumulative measure of classification performance over possible classification thresholds. The greater the AUC, the higher the ability of the model to distinguish between positive and negative classes.




```
Area under the curve (AUC): 0.913
```

The AUC from the XGBoost model is 0.913 which is considered as almost perfect. Moreover, as the goal is to find fraudulent transactions, we can accept a higher false positive rate. Hence, our best threshold will be at the peak of the curve on the top-right corner of the ROC curve.

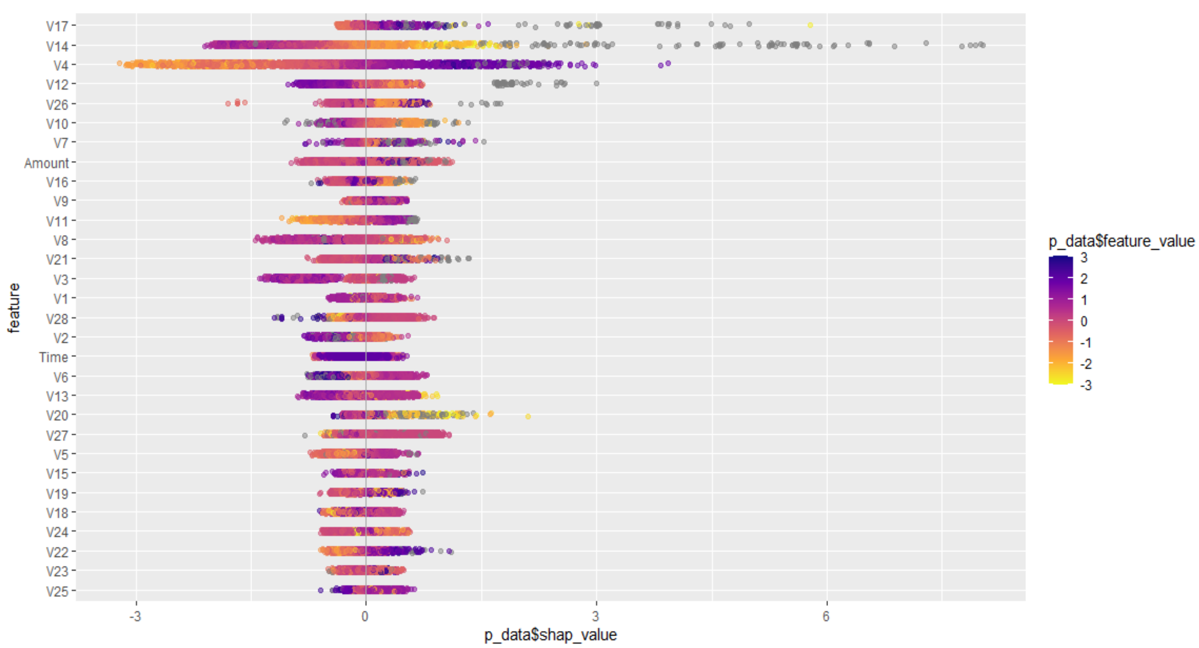
Top Contributors



SHAP is an acronym for SHapley Additive exPlanations. SHAP values indicate the contribution of each variable on the final score of the prediction. Seen above are the top 3 variables contributing to the final prediction, the variables are arranged in a descending order. The SHAP values are against the Y-axis and the variable values are against the x-axis. Each blue dot is an entry in the data set, whereas the red curve is the range of values the variable can take and corresponding SHAP values.

Positive SHAP value means positive impact on prediction, leading the model to predict 1.[citation 1] Negative SHAP value means negative impact, leading the model to predict 0.[citation 1]

From the graph of variable 'V4' we see that for the range of variable values between 1 through the SHAP values are positive and negative otherwise. This means that variable values of V4 between 1 to 15 have a positive impact, leading the model to predict 1 and predict 0 for other values.



The graph above represents a summary of all the SHAP value of all the 30 independent variables. Each dot on the graph represent an entry in the data set. The heat map on the right-hand side give the range of values that variable takes.

We can see that higher feature value of variable V14 contribute negatively to the prediction. The same can be compared with the 'xgb.plot.shap' and we can see that for V14 for values -1 and greater the SHAP values are negative.

The variables in the graph are in the descending order i.e., the variable V17 contributes the highest in terms of predicting the outcome and V25 contributes the least to the prediction of the outcome.

Logistic Regression

To compare the results of the above XGBoost Classification model we ran a logistic regression classification to predict the non-fraud and fraudulent cases.

Visualization

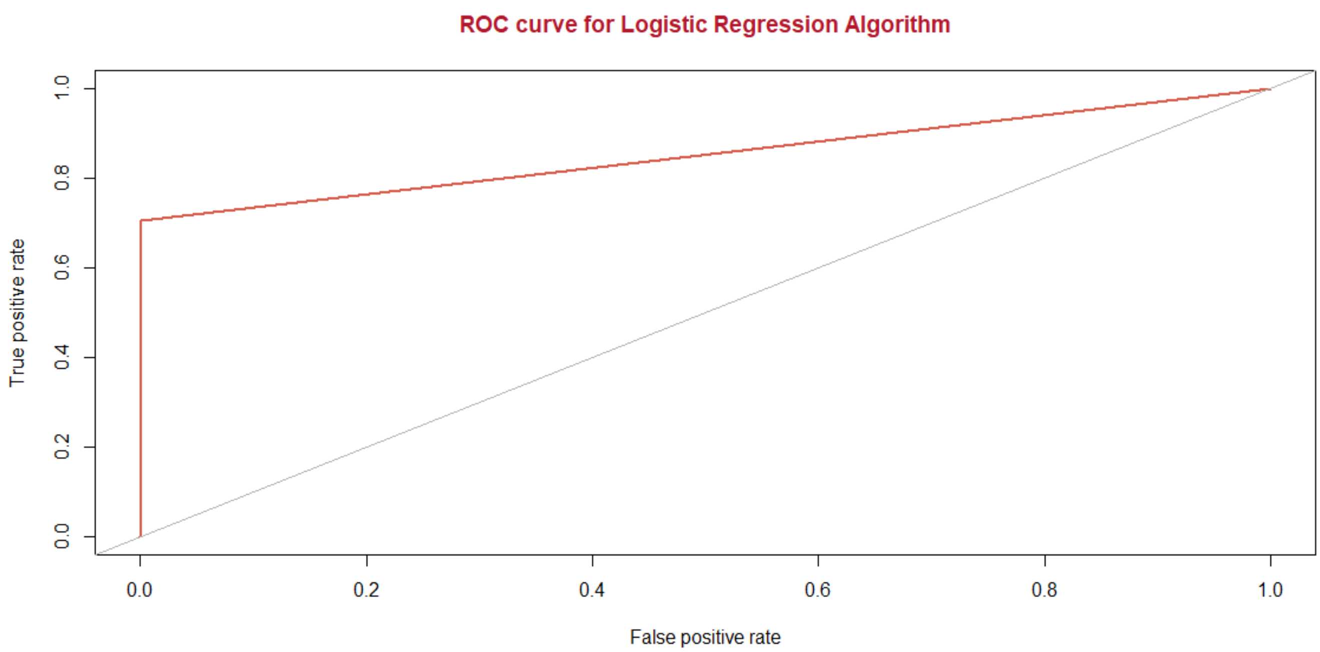
Confusion Matrix

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56857    29
         1     6    69

               Accuracy : 0.9994
                 95% CI : (0.9991, 0.9996)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : 1.953e-13

                  Kappa : 0.7974

 Mcnemar's Test P-Value : 0.0002003

            Sensitivity : 0.9999
            Specificity : 0.7041
         Pos Pred Value : 0.9995
         Neg Pred Value : 0.9200
             Prevalence : 0.9983
         Detection Rate : 0.9982
   Detection Prevalence : 0.9987
      Balanced Accuracy : 0.8520

       'Positive' Class : 0
```

The testing data set contained 56,961 entries. As seen from the confusion matrix above, the Logistic Regression model for classification correct identified 99.94% of the transactions, correctly identifying 69 fraudulent transactions and incorrectly marking 35 transactions. In the incorrect transactions the model incorrectly identified 29 fraudulent transactions as non-fraudulent and 6 non-fraudulent transactions as fraudulent.

Kappa is essentially interrater reliability testing, measure of agreement between the predicted labels and the true labels, and it takes into account the possibility of agreement occurring by chance. A high Kappa value of 0.7974 means that the classification of this data was not by chance and that the result has good agreement.

ROC Curve




```
Area under the curve (AUC): 0.852
```

The AUC from the Logistics Regression (classification) model is 0.852 which is quite high. Moreover, as the goal is to find fraudulent transactions, we can accept a higher false positive rate. Hence, our best threshold will be at the peak of the curve on the top-right corner of the ROC curve.

Conclusion

Both the XGBoost and logistic regression for classification were implemented on given unbalanced dataset. The findings are:

1) The AUC for the XGBoost is significantly better at 0.913 compared to Logistic regression at 0.852, indicating that XGB has better discriminating power.
2) The XGBoost model was 42.85% less prone to incorrect classification, which is evident from the confusion matrix of the two model where XGBoost classified 15 fewer transactions incorrectly from a data set of 59,916. Furthermore, we are more concerned about false negatives and on this front the XGBoost model classified 58.6% fewer variables as false negatives.
3) The kappa value for XGBoost model and LR model are 0.8899 and 0.7974 respectively, indicating substantial level of agreement between the predicted and true values.

To conclude, the XGBoost (Classification) model for detecting credit card frauds was more robust at correctly predicting fraudulent transactions as compared to Logistic Regression (Classification).

Citations

1. Interpretation of SHAP values
2. Interpretation of SHAP values alternate
3. XGBoost Mathematics Explained
4. XGBoost: A Scalable Tree Boosting System
5. XGBoost Documentation
6. Interrater reliability: the kappa statistic