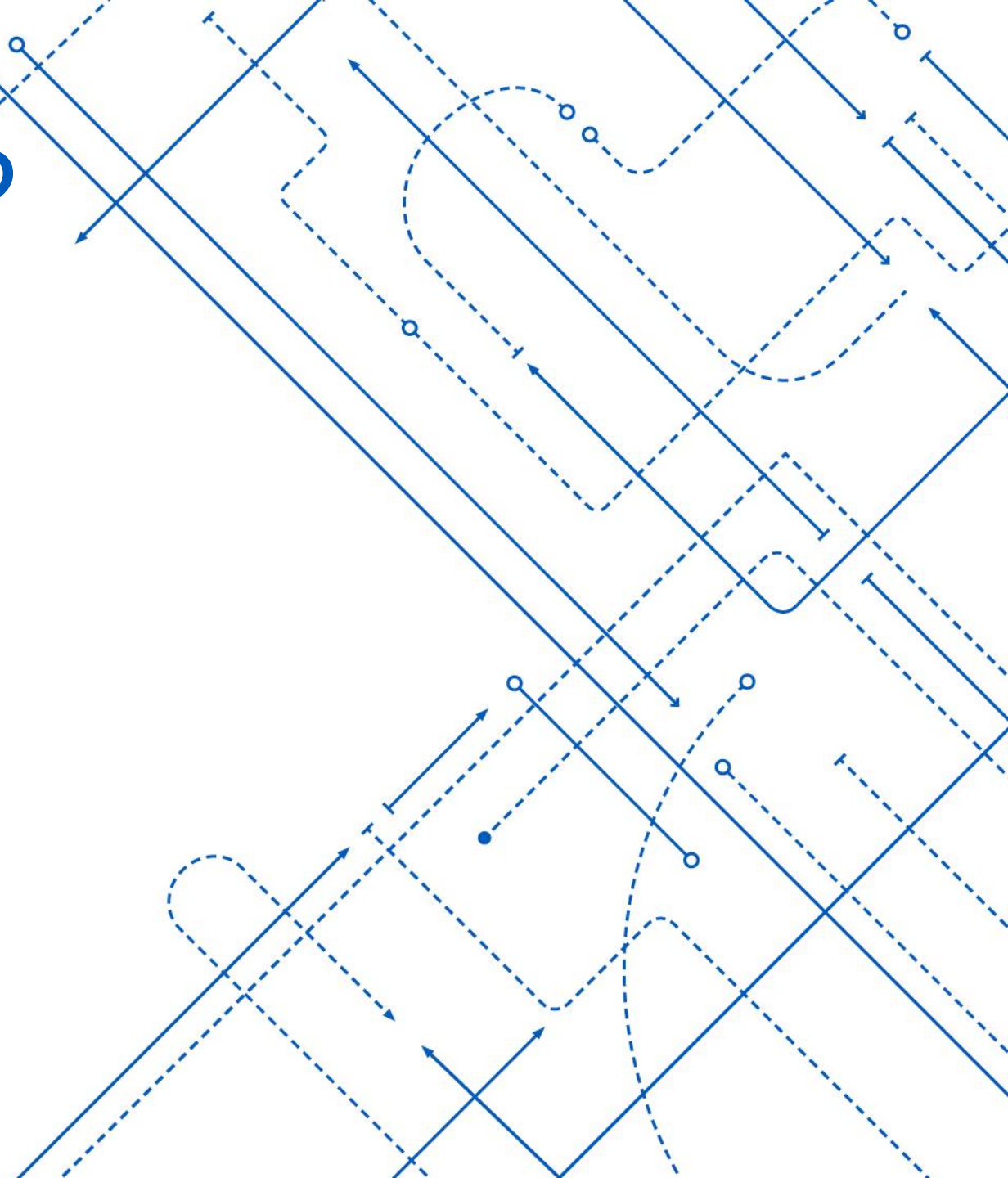
CREDIT CARD FRAUD DETECTION - SAMPLING

Ganesh Reddy

Annapa Reddy

50442295

University at Buffalo The State University of New York



Introduction

- Billions of dollars of loss are caused every year due to fraudulent credit card transactions. The design of efficient fraud detection algorithms is key to reducing these losses, and more algorithms rely on advanced machine learning techniques to assist fraud investigators. The design of fraud detection algorithms is however particularly challenging due to non-stationary distribution of the data, highly imbalanced classes distributions and continuous streams of transactions. At the same time public data are scarcely available for confidentiality issues, leaving unanswered many questions about which is the best strategy to handle this issue. Also, There is an explosion of demand for new payment methods. With new payment methods, we have an extremely complex backend methodologies which makes fraud detection all the harder. Global fraud has increased by almost three times, from $9.84 billion to $32.39 billion in less than a decade (2011 to 2020).
- We have nearly 1.8 billion Euros on average of fraudulent transactions detected in Europe every year.
- The objective of this project is to apply machine learning algorithms on the dataset to successfully predict fraudulent transactions. And to determine which of the algorithms performs the best.

Data Set Description

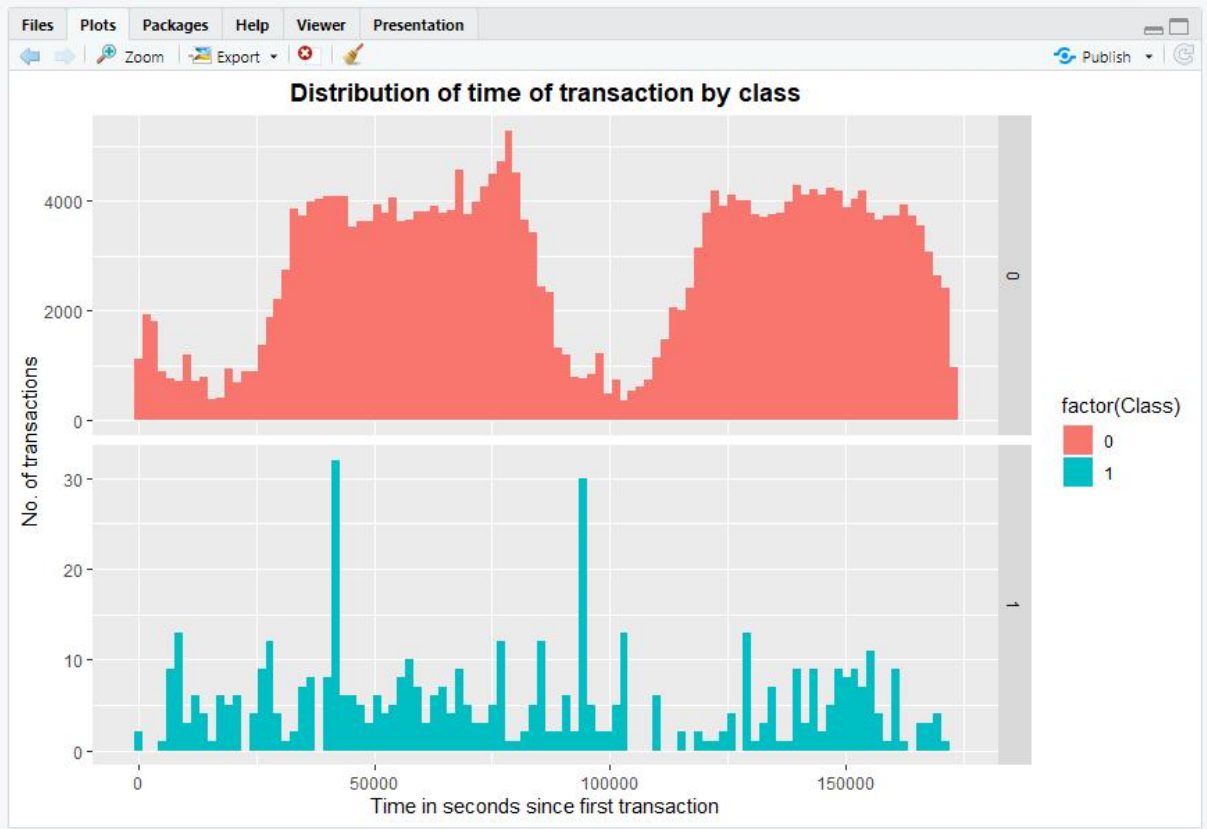
- The dataset contains transactions made by credit cards in September 2013 by European cardholders.
- This dataset presents transactions that occurred in two days, where we have **492** frauds out of **284,807** transactions. The dataset is highly unbalanced. The positive class (fraud transactions) accounts for **0.172%** of all transactions.
- It contains only numeric input variables, which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, the original features and more background information about the data are not provided. 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.
- Given the class imbalance ratio, we have measured the model accuracy using the Area Under the Precision-Recall Curve (AUPRC). Confusion matrix accuracy is not meaningful for unbalanced classification.

Exploratory Data Analysis

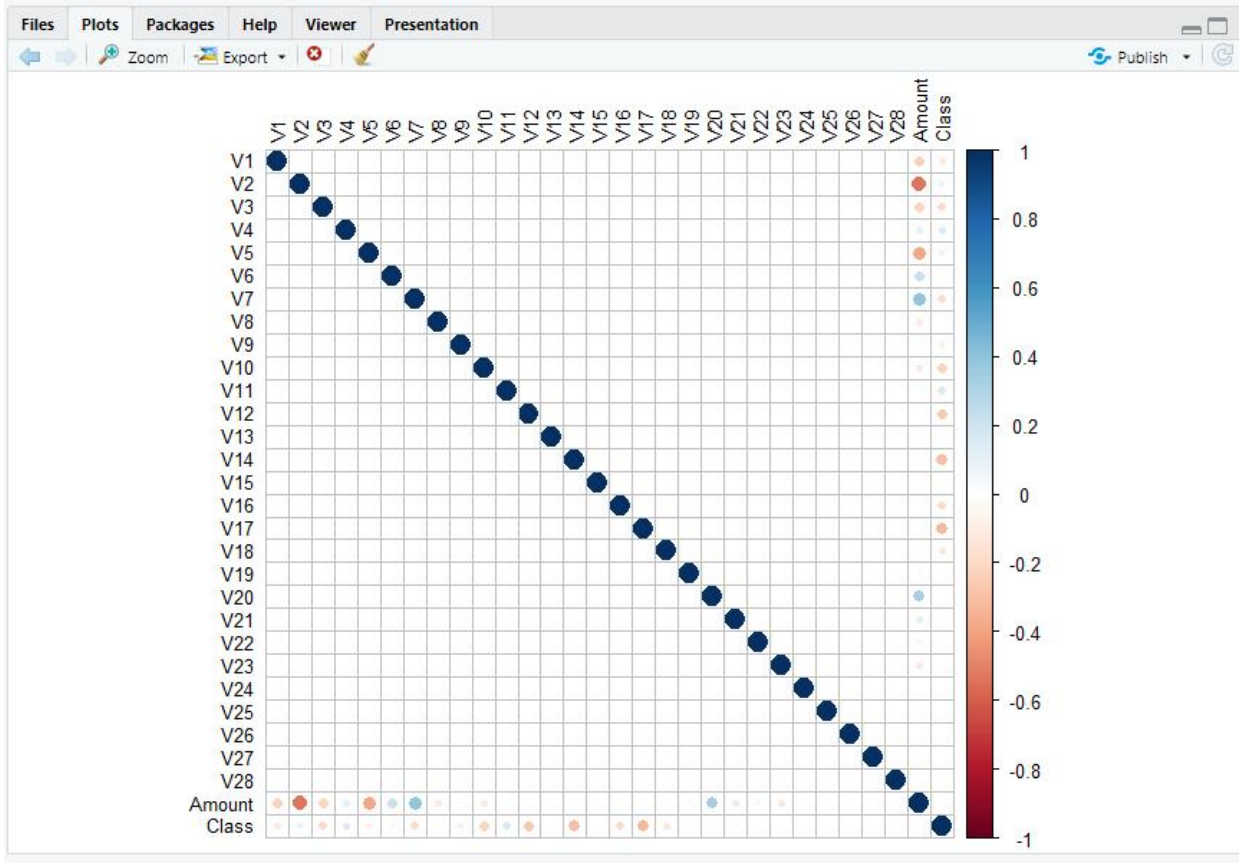
- There are no missing values in the data set.
- The Data shown below reflects the imbalance of non-fraud and fraud transactions in the dataset. We have class "0" - No fraud, "1" – fraud . We see that our dataset is highly unbalanced with respect to the class of interest "Fraud".

```
> table(df$class)

     0      1
284315    492
> prop.table(table(df$class))

          0           1
0.998272514 0.001727486
>
```

Clearly the dataset is very imbalanced with 99.8% of cases being non-fraudulent transactions. A simple measure like accuracy is not appropriate here as even a classifier which labels all transactions as non-fraudulent will have over 99% accuracy. An appropriate measure of model performance here would be AUC (Area Under the Precision-Recall Curve).



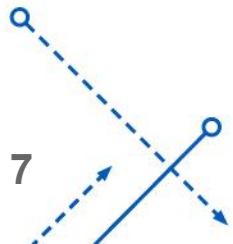
- The 'Time' feature looks similar across both types of transactions. we could argue that fraudulent transactions are more uniformly distributed, while normal transactions have a cyclical distribution
- There is clearly a lot more variability in the transaction values for non-fraudulent transactions.



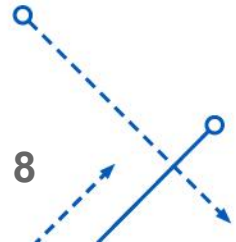
We observe that most of the data features are not correlated. This is because before publishing, most of the features were presented to a Principal Component Analysis (PCA) algorithm. The features V1 to V28 are most probably the Principal Components resulted after propagating the real features through PCA. We do not know if the numbering of the features reflects the importance of the Principal Components.

Modeling Approach

- Algorithms struggle with accuracy because of the unequal distribution in dependent variable. This causes the performance of existing classifiers to get biased towards majority class. The algorithms are accuracy driven i.e.; they aim to minimize the overall error to which the minority class contributes very little. ML algorithms assume that the data set has balanced class distributions. They also assume that errors obtained from different classes have same cost.
- The methods to deal with this problem are widely known as 'Sampling Methods'. Generally, these methods aim to modify an imbalanced data into balanced distribution using some mechanism. The modification occurs by altering the size of original data set and provide the same proportion of balance.
- We use two methods to treat the imbalanced dataset undersampling and oversampling.



- Undersampling is reducing the number of observations from majority class to make the data set balanced
- Oversampling is replicating the observations from minority class to balance the data.
- After trying all the sampling techniques upsampling yielded better AUC scores than the simple imbalanced dataset. We will test different models now using the **up-sampling technique** as that has given the highest AUC score.
- it is important to consider the tradeoff between model accuracy and model complexity (which is inherently tied to computational cost). It might be the case that having a simple model with short inference times which achieves an accuracy of 85% is sufficient for a given task, as opposed to having a, say, 10-layer neural network which trains for 2 days on a GPU cluster and is 90% accurate.
- Therefore, we will start out with logistic regression.



Logistic Regression Results

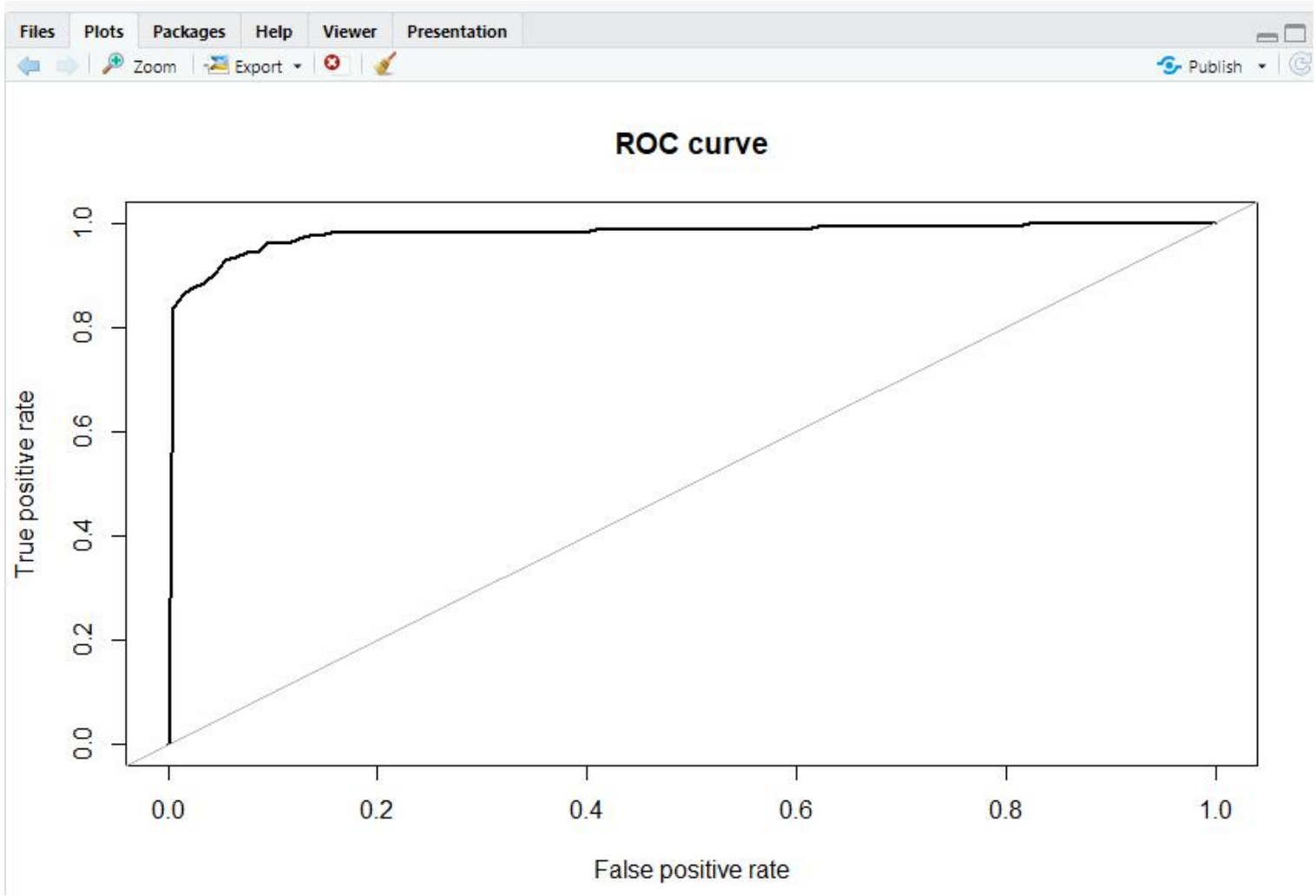
```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56861     8
         1    34    58

               Accuracy : 0.9993
                 95% CI : (0.999, 0.9995)
    No Information Rate : 0.9988
    P-Value [Acc > NIR] : 0.0010494

                  Kappa : 0.7338
 Mcnemar's Test P-Value : 0.0001145

            Sensitivity : 0.9994
            Specificity : 0.8788
         Pos Pred Value : 0.9999
         Neg Pred Value : 0.6304
             Prevalence : 0.9988
         Detection Rate : 0.9982
   Detection Prevalence : 0.9984
      Balanced Accuracy : 0.9391

       'Positive' Class : 0
```

A simple logistic regression model achieved nearly 100% accuracy, with ~99% precision (positive predictive value) and ~100% recall (sensitivity). We can see there are only 8 false negatives (transactions which were fraudulent but identified as such by the model). This means that the baseline model will be very hard to beat.

- We can further minimize the number of false negatives by increasing the classification threshold. However, this comes at the expense of identifying some legitimate transactions as fraudulent. This is typically of much lesser concern to banks, and it is the false negative rate [illegible] imized.

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56866     3
         1    67    25

               Accuracy : 0.9988
                 95% CI : (0.9984, 0.999)
    No Information Rate : 0.9995
    P-Value [Acc > NIR] : 1

                  Kappa : 0.4162
 Mcnemar's Test P-Value : 5.076e-14

            Sensitivity : 0.9988
            Specificity : 0.8929
         Pos Pred Value : 0.9999
         Neg Pred Value : 0.2717
             Prevalence : 0.9995
         Detection Rate : 0.9983
   Detection Prevalence : 0.9984
      Balanced Accuracy : 0.9458

       'Positive' Class : 0
```

Now we have just 3 false negatives, but we identified many more legitimate transactions as fraudulent compared to 0.5 threshold. When adjusting the classification threshold, we can have a look at the ROC curve to guide us

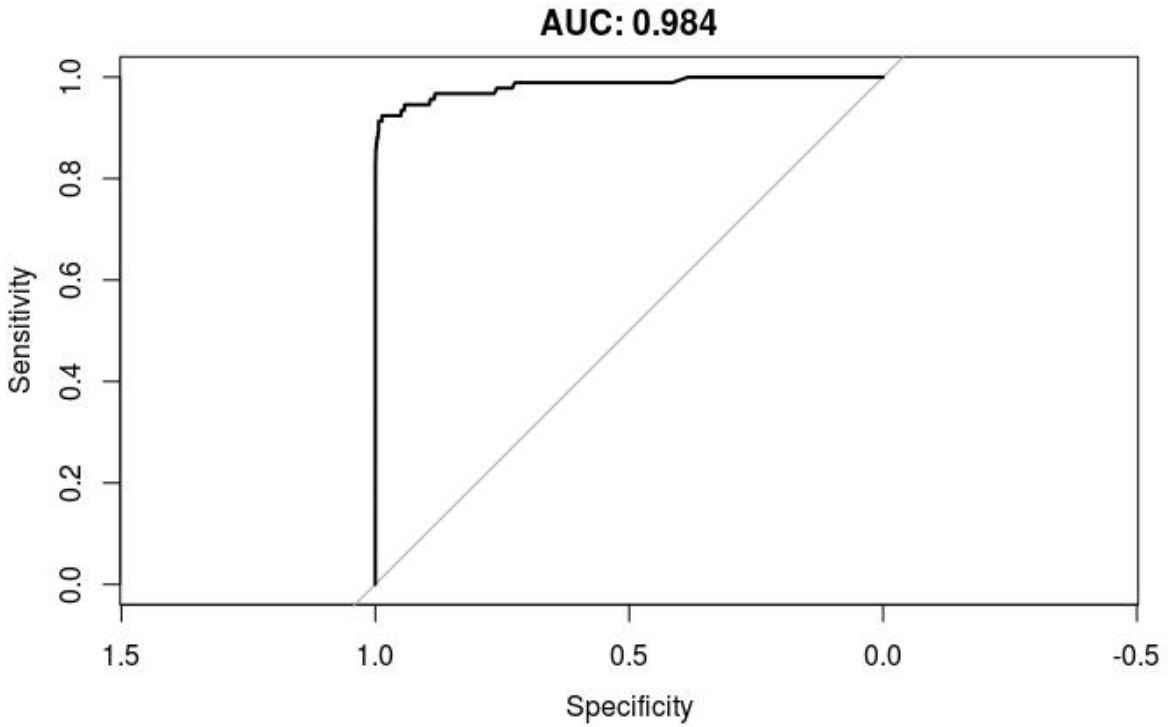


Area under the curve (AUC): 0.978

Random Forest Results

```
Random Forest

227846 samples
    30 predictors
     2 classes: 'X0', 'X1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 205062, 205062, 205061, 205062, 205062, 205061, ...
Addtional sampling using SMOTE

Resampling results across tuning parameters:

  mtry  ROC        Sens       Spec
   2    0.9796401  0.9954671  0.8775
  16    0.9825342  0.9901339  0.8850
  30    0.9814775  0.9850074  0.8850

ROC was used to select the optimal model using the largest value.
The final value used for the model was mtry = 16.
```

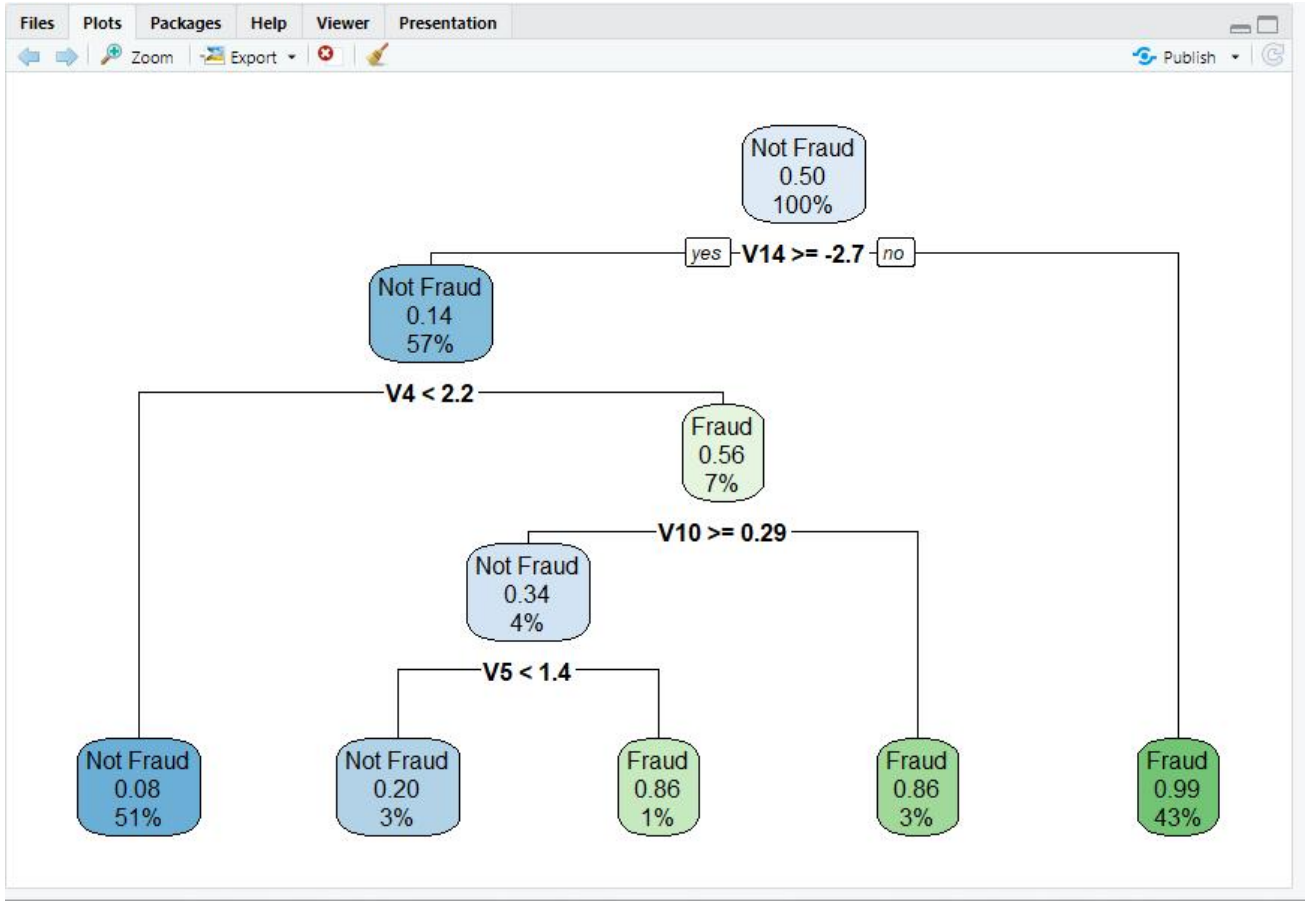
```
          Reference
Prediction     0     1
         0 56443     8
         1   426    84

               Accuracy : 0.9924
                 95% CI : (0.9916, 0.9931)
    No Information Rate : 0.9984
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2771
 Mcnemar's Test P-Value : <2e-16
```



If we compare variable importance's of the RF model with the variables identified earlier as correlated with the "Class" variable. The top 3 most important variables in the RF model were also the ones which were most correlated with the "Class" variable. Especially for large datasets, this means we could save disk space and computation time by only training the model on the most correlated/important variables, sacrificing a bit of model accuracy.

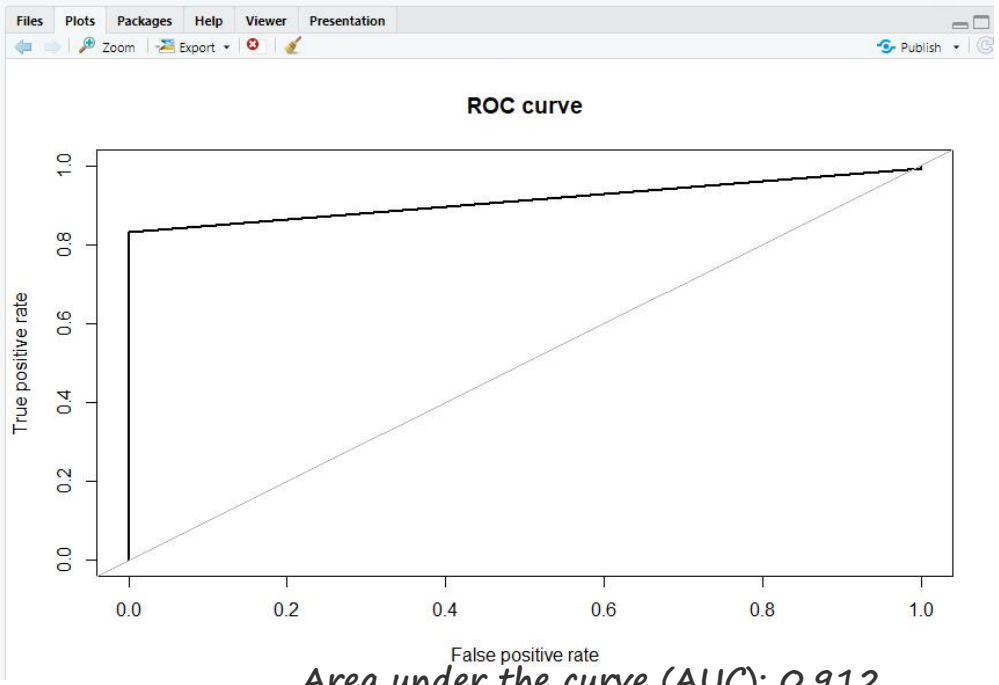
The Random Forest model achieved ~100% precision and 98% recall, which is - surprisingly - lower than for logistic regression. This might be because Random Forest has too high model capacity and hence overfits to training data. As can be seen above, it also achieves marginally higher AUC score compared to logistic regression (since that's what the objective function was).

Decision Tree Analysis



From this Decision Tree model, we can see that v14 is the most important variable that separates fraud and non-fraud transactions.

Let us first look at how Decision Trees (CART) performs with the data. We use the function roc.curve available in the ROSE package to gauge model performance on the test set



Area under the curve (AUC): 0.912

We evaluate the model performance on test data by finding the roc AUC score. We see that the AUC score on the original dataset is 0.912 .

Conclusion

- In this project we have tried to show different methods of dealing with unbalanced datasets like the fraud credit card transaction dataset where the instances of fraudulent cases is few compared to the instances of normal transactions.
- We have argued why accuracy is not an appropriate measure of model performance here and used the metric AREA UNDER ROC CURVE to evaluate how different methods of oversampling or under sampling the response variable can lead to better model training.
- We concluded that the oversampling technique works best on the dataset and achieved significant improvement in model performance over the imbalanced data.
- The best score of 0.984 was achieved using random forest and logistic regression, Decision Tree models performed well too. It is likely that by further tuning the we can achieve even better performance.
- This project has demonstrated the importance of sampling effectively, modelling and predicting data with an imbalanced dataset.

Thank You

I would like to express my deep gratitude to Professor Rachel Hageman Blair for her advice and assistance in keeping my progress on schedule. And I would like to thank our TAs for their valuable inputs and support.