Fraud Classification

By: Liam deVerteuil, Anthony Hu and Xuhesheng Chen

I. Introduction

Financial information is being threatened by increasingly more advanced technological methods, and thus more susceptible to cyberattacks. Banks like Wells Fargo, who are responsible for the safety of the financial information of their customers, would like to allocate financial and technological resources efficiently to help protect their customers from these financial attacks.

We believe that the elder population -- defined as adults over 59 years of age-- is more susceptible to fraudulent financial abuse, but also less likely to report financial abuse due to lack of knowledge or barriers in technology. Hence, it is logical that a significant portion of the resources dedicated to protecting financial information should be used on elders. We would like to find the most efficient way to dedicate these resources to protecting the elder consumers.

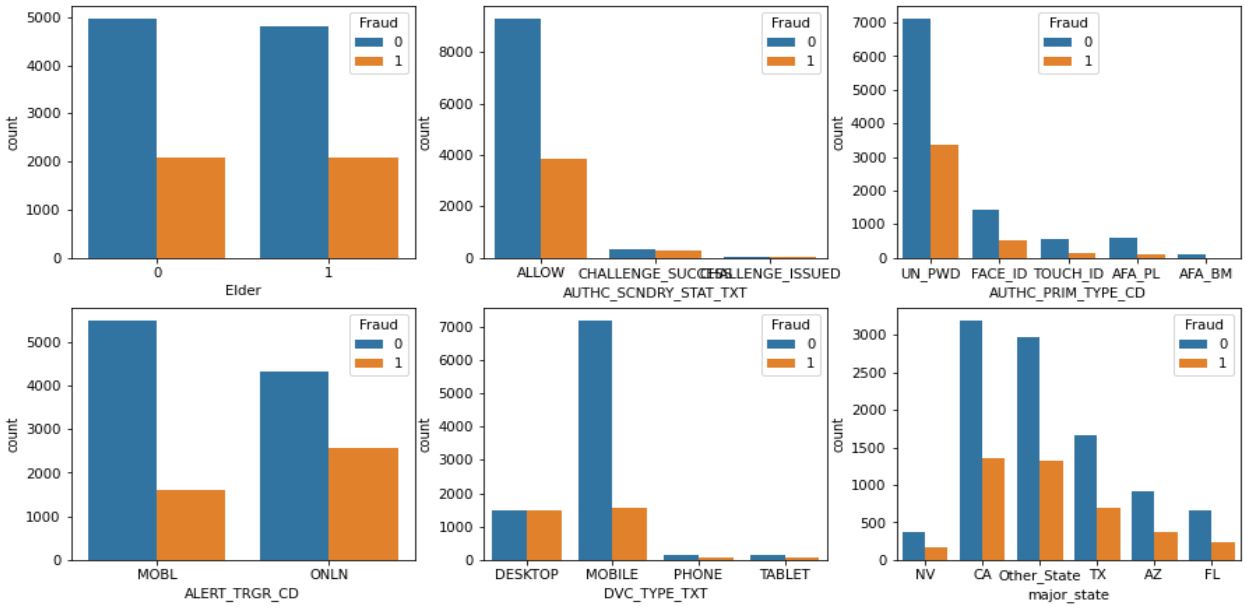
This report will be dedicated to finding the optimal classification method that best identifies fraudulent transactions amongst elder customers. We will pay special attention to minimizing the Type 2 error, which is a false negative; in this case, a false negative would be a fraudulent transaction that went unreported. Minimizing the false negative is the most important criteria in our classification model, and thus we will pay special attention to this metric.

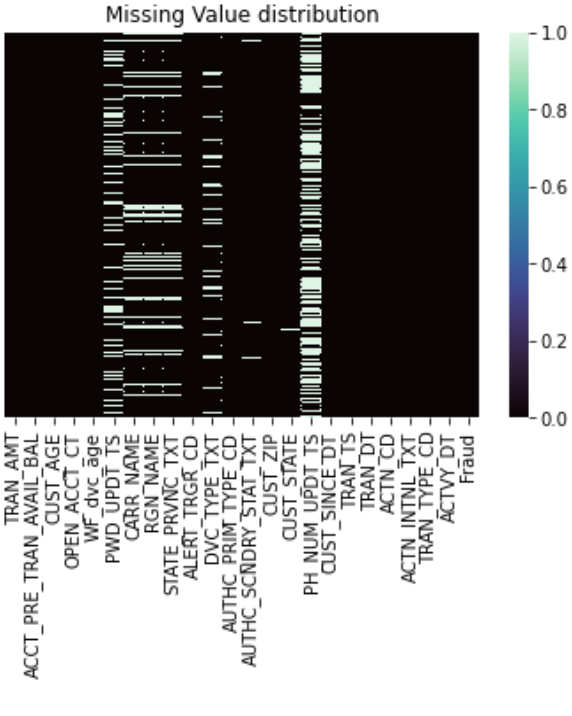
II. Dataset and Pre-Processing:

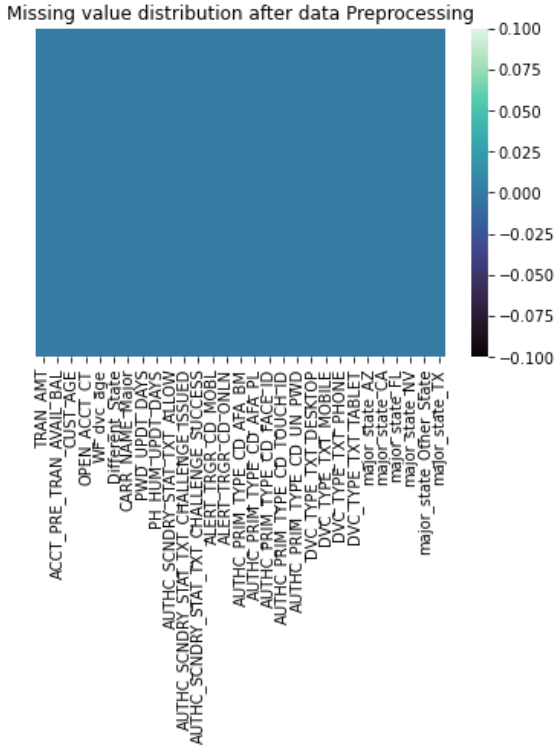
The original dataset provides 14,000 samples with 4200 Fraud labels and 9800 Non Fraud Labels, as well as 22 features including basic information (such as age, resident state of customer), bank account record of customers (including Wells Fargo account age, password updating date, phone number updating date), transaction information (like transaction amount, transaction device, the state transaction occurred, carrier name derived from IP address of customer's device, alert trigger code) and specific timestamp record(transaction timestamp, activity date).

About half (6920) of the dataset is older customers (age > 59). For this specific project, we aim to classify fraud problems for those customers. But before analysis, we need to preprocess them before analysis since some features in the original dataset contain some missing values.

Based on the feature attribute, we proposed a different solution. For example, we filled in PW_UPDT_TS, PH_NUM_UPDT_TS with the value of CUST_SINCE because we assumed that those customers with missing values never changed their password or phone number since the account was opened. For the remaining 36 samples holding missing values in all three of the aforementioned features, we removed them from the dataset completely.


Fraud
0
1
count
Elder
AUTHC_SCNDRY_STAT_TXT
ALLOW
CHALLENGE_SUCCESS
CHALLENGE_ISSUED
AUTHC_PRIM_TYPE_CD
UN_PWD
FACE_ID
TOUCH_ID
AFA_PL
AFA_BM
ALERT_TRGR_CD
MOBL
ONLN
DVC_TYPE_TXT
DESKTOP
MOBILE
PHONE
TABLET
major_state
NV
CA
Other_State
TX
AZ
FL


Missing Value distribution
TRAN_AMT
ACCT_PRE_TRAN_AVAIL_BAL
CUST_AGE
OPEN_ACCT_CT
WF_dvc_age
PWD_UPDT_TS
CARR_NAME
RGN_NAME
STATE_PRVNC_TXT
ALERT_TRGR_CD
DVC_TYPE_TXT
AUTHC_PRIM_TYPE_CD
AUTHC_SCNDRY_STAT_TXT
CUST_ZIP
CUST_STATE
PH_NUM_UPDT_TS
CUST_SINCE_DT
TRAN_TS
TRAN_DT
ACTN_CD
ACTN_INTNL_TXT
TRAN_TYPE_CD
ACTVY_DT
Fraud
1.0
0.8
0.6
0.4
0.2
0.0


Missing value distribution after data Preprocessing
TRAN_AMT
ACCT_PRE_TRAN_AVAIL_BAL
CUST_AGE
OPEN_ACCT_CT
WF_dvc_age
Different State
CARR_NAME_Major
PWD_UPDT_DAYS
PH_NUM_UPDT_DAYS
AUTHC_SCNDRY_STAT_TXT_ALLOW
AUTHC_SCNDRY_STAT_TXT_CHALLENGE_ISSUED
AUTHC_SCNDRY_STAT_TXT_CHALLENGE_SUCCESS
ALERT_TRGR_CD_MOBL
ALERT_TRGR_CD_ONLN
AUTHC_PRIM_TYPE_CD_AFA_BM
AUTHC_PRIM_TYPE_CD_AFA_PL
AUTHC_PRIM_TYPE_CD_FACE_ID
AUTHC_PRIM_TYPE_CD_TOUCH_ID
AUTHC_PRIM_TYPE_CD_UN_PWD
DVC_TYPE_TXT_DESKTOP
DVC_TYPE_TXT_MOBILE
DVC_TYPE_TXT_PHONE
DVC_TYPE_TXT_TABLET
major_state_AZ
major_state_CA
major_state_FL
major_state_NV
major_state_Other_State
major_state_TX
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
−0.075
−0.100

Secondly, we have created new features from the dataset, such as PWD_UPD_DAYS (numbers of days since password updated), PH_NUM_UPDT_DAYS (number of days since phone number updated), and Different_State (whether customer's resident state different from the state where transaction occurred).

Then we dropped all variables with only one unique value, because those features are meaningless for classification and therefore, we cannot use it for further analysis.

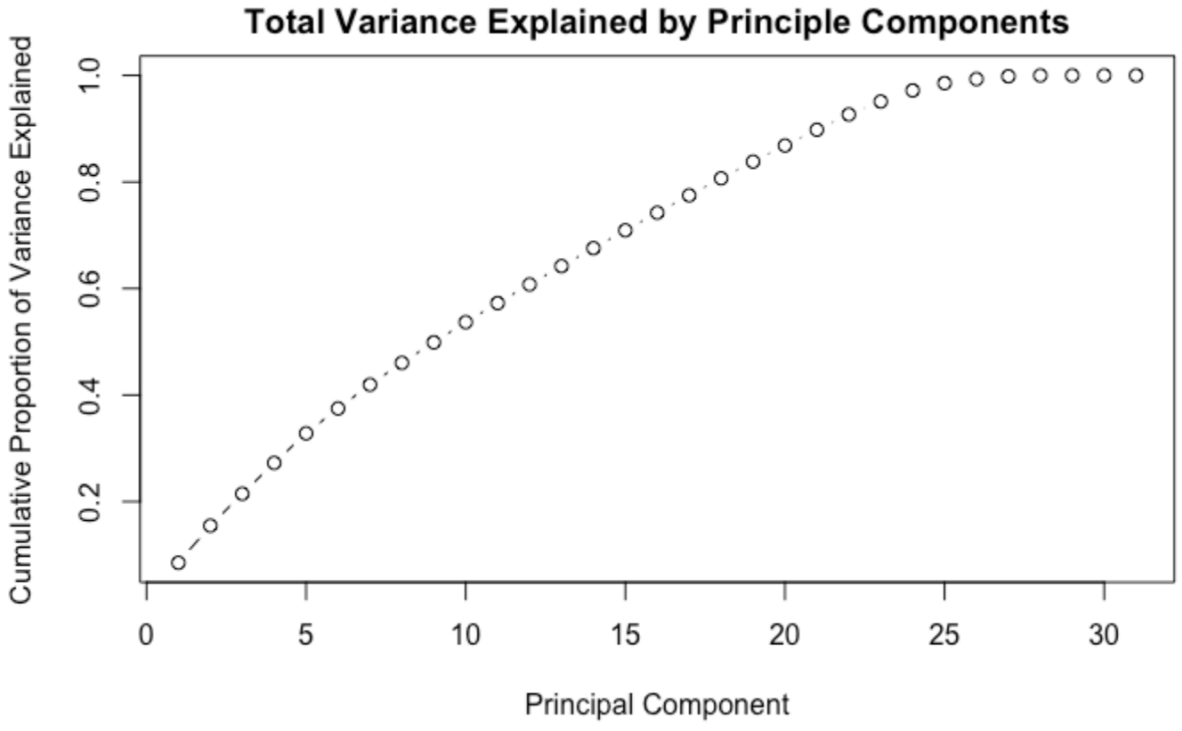
Finally, we transformed some of the original features for easier analysis. For features that only have two categorical values, we transformed them into binary values for quantitative analysis, into dummy features. For those features with multiple categorical values, we have turned them into a dummy matrix, of multiple dummy features.

After preprocessing, we have 29 features for further exploration and analysis.

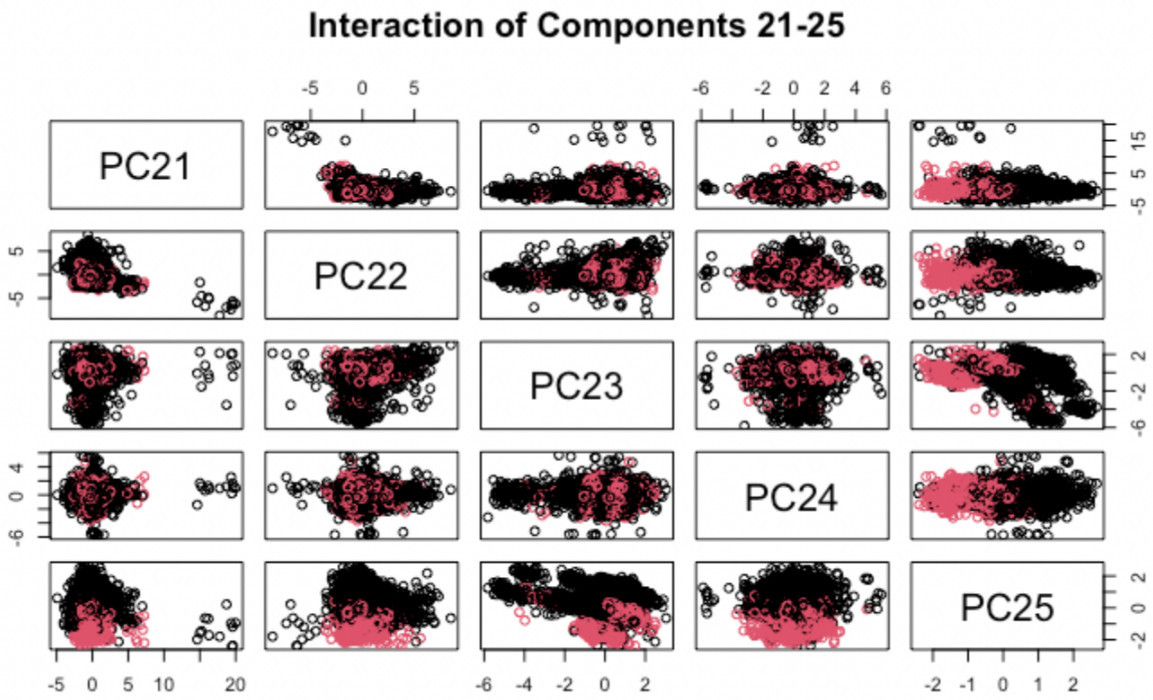
III. Learning Methods

Principal Component Analysis:

The first machine learning technique we used was the unsupervised method; Principal Component Analysis (PCA). We wanted to perform this analysis to firstly see if we could lower the dimensionality of our data while still being able to explain most of the variance within the dataset. After performing the PCA we first plotted a graph to show the proportion of variance explained by each component and also the cumulative proportion of variance explained by the components in order to see if there was anything peculiar or noteworthy with the components produced. We found that if the data was reduced to 25 components more than 95% of the variance would be explained as seen in the Total Variance explained graph below.



We then created a pairs plot that showed the interaction of each pair of the components. We color coded these scatter plots with Red depicting the fraudulent transactions and black depicting the non-fraudulent in order to see if any trends in the data could be seen. What we found is that there was little to no relationship or distinction between the fraudulent and non fraudulent transactions that could be identified through the interactions of any of the components besides the interactions with PC 25. However, when looking at the interaction of PC 25 with a few of the other principal components, as seen in the plot below, we could see a clear distinction between both the red and black data points meaning that there was some separation, however this separation was not very linear. This analysis of all the components led us to believe that the most suitable models should be more flexible such as decision trees or K nearest neighbors as opposed to more stringent techniques like logistic regression.

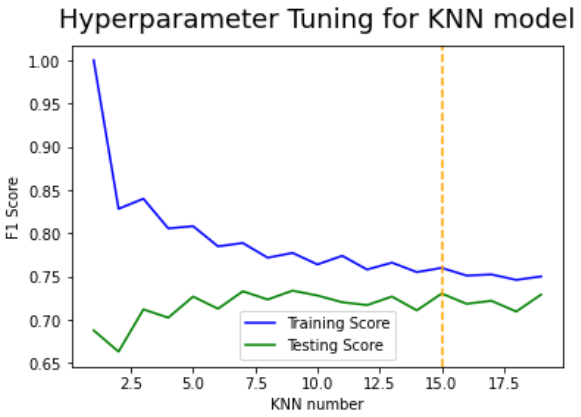


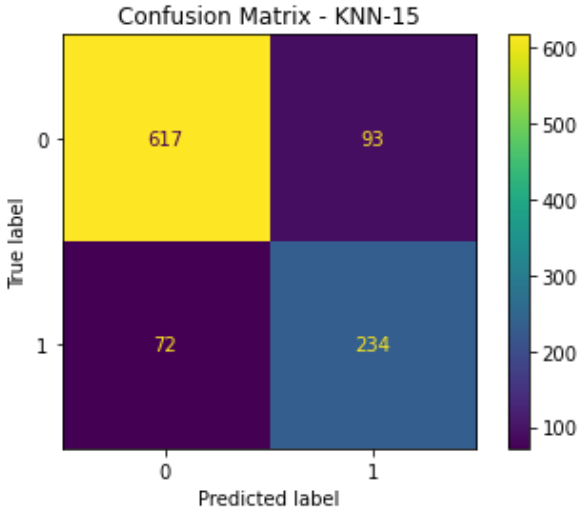
Testing and Training Sets

Before we begin testing the various algorithms and their accuracy, we split the original elder dataset into the testing and training dataset. To maintain the same proportion of fraudulent to non-fraudulent transactions, we partitioned the training and testing sets accordingly. The training set will contain 85 percent of the observations --5880 observations --in the elder dataset, and the testing set the remaining 15 percent -- 1016 observations.

K Nearest Neighbour

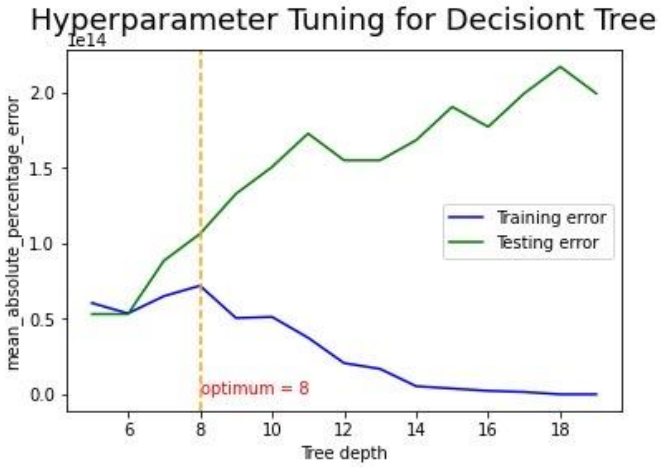
We first tried the k nearest neighbour algorithm to classify samples since it is simple to implement and flexible to non-linear distribution problems. We picked F1 score for tuning parameters because we want to achieve a balance between recall and precision of performance. When k equals 15, it looks stable on both training set and test set, so we chose it as optimal value and got accuracy of 83.76%. Then we plot a confusion matrix to see the error distribution. Both false positive errors and false negative errors look a lot. In this project, we majorly focus on false negative problems because we tried to reduce as much risk that senior customers suffer from fraud as possible. If the number of false negative samples is too large, even if the accuracy of model prediction is higher enough, the risk that the model predicts non-fraud but actually it is a fraud situation would exert a severe loss for related customers. Therefore, KNN may not be a good choice for this problem. We have to try another method.





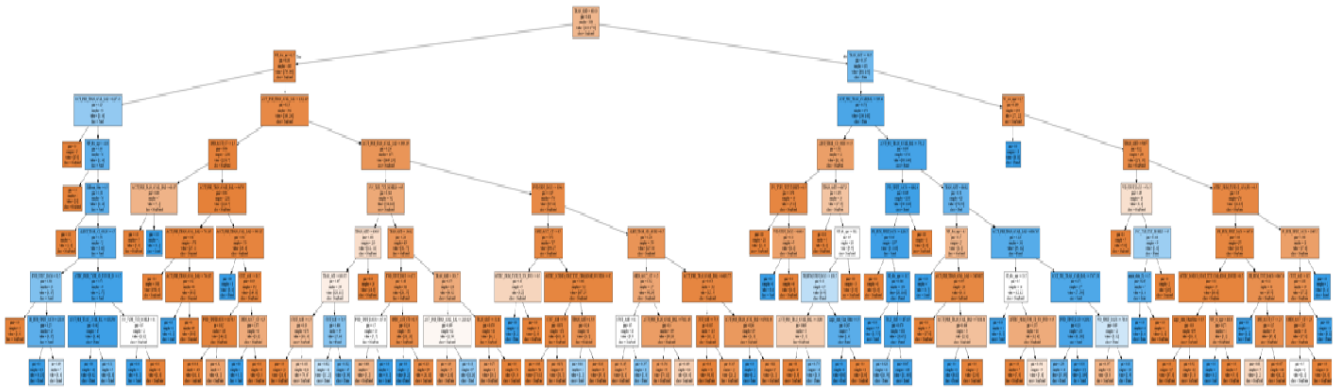
Decision Tree

The next model we decided to fit to the data was a decision tree model. However before we could produce a tree and test its performance we had to decide what would be the optimal depth for the tree. To make this decision we decided to look at the f1 scores for the different depths as this score would help in our need to reduce the type 2 errors.To do this we plotted the f1 score against trees of varying depths as seen in the tuning graph below and found that the optimal depth to be 8.

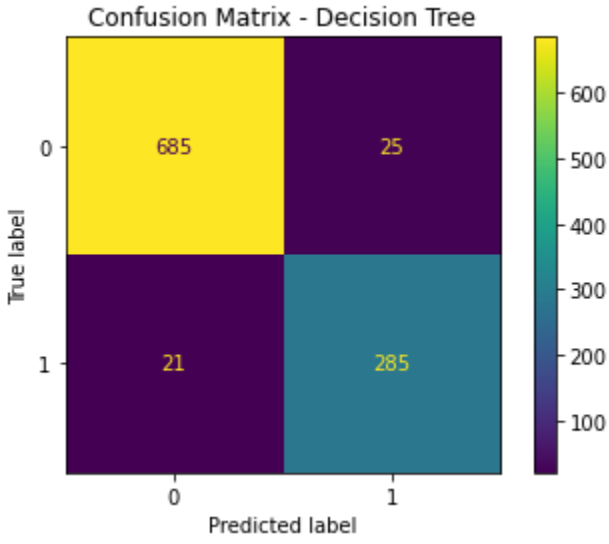


After finding this depth we modeled the decision tree a visualisation of which can be seen below where the blue nodes represent Fraudulent transactions and the Orange nodes represent Non-Fraudulent transactions. Being a very large tree many of the labels on the nodes are unreadable without access to the original file, however there were a few trends we saw in the tree that should be mentioned. Starting with the root of the tree which uses the transaction amount feature and asks whether it is greater or less than 483 where if it was less it moved to the left and was less likely to be fraud and with the opposite being seen on the branch to the right. This feature appeared many times throughout the tree and in general followed the same trend that the higher the transaction amount the more likely the transaction was fraudulent.

There were many other features such as the recency of password updates and authentication type that were seen in the lower parts of the tree, however there seemed to be 2 other important features that showed up multiple times the first being the pre transaction account balance which showed that higher balances tended to mean that the transaction was more likely to be fraudulent which could show that wealthier persons are targeted more frequently. And the 2nd was the age of the device used for the transaction which showed that in general the younger the device the more likely it was to be fraud which again makes sense seeing as the older population may be less up to date on their technology than the criminal.

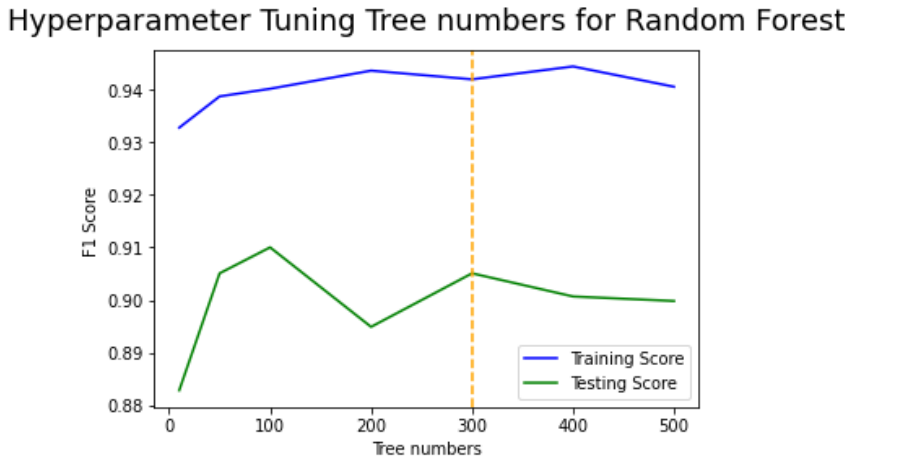


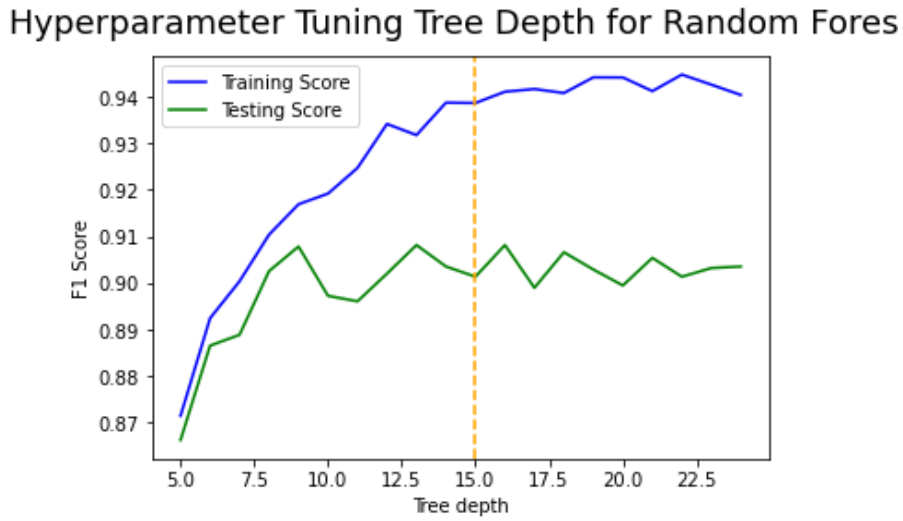
We then proceeded to test the accuracy of the model on our test set and found that there was a very sizable increase in the overall accuracy of our predictions when compared to the KNN model. The overall accuracy was seen to be 95.47% accurate on the testing set as opposed to the 83.76% found for the KNN model. This was a very good sign however we wanted to check the recall and precision of the model as well to determine how it performed in in its type 1 and type 2 error rates so we proceeded to plot a confusion matrix, as seen below, to compare the false negative predictions (type 2 errors) and the false positive predictions (type 1 errors). What we found is that both these error rates were reduced to < 2.5% with the type 2 error rate, which we are particularly hoping to minimize being 2.07%.



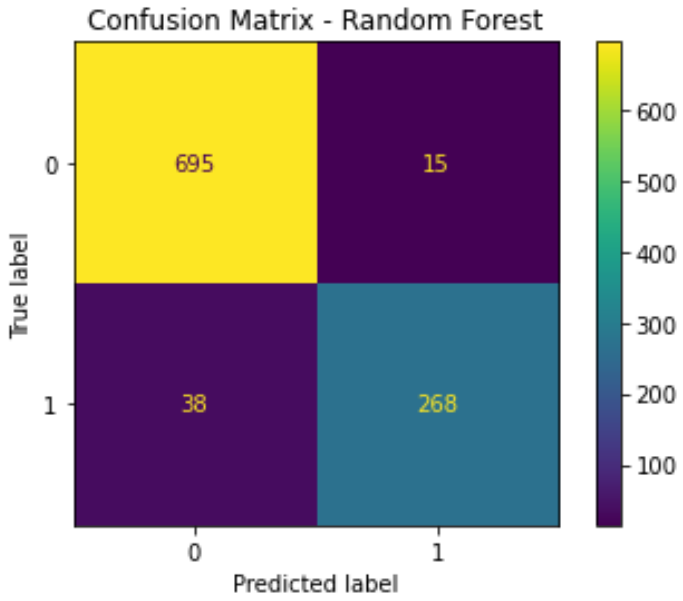
Random Forests

The random forests algorithm is built upon the decision tree method. Instead of one singular decision tree, the random forests algorithm generates a large number of relatively uncorrelated trees (hence the forests in the name). Then, each trees' class prediction is tallied up, and then the class predictor value is then decided through majority-vote. Since there are a large number of trees, the strength of the random forests is that the errors of one individual tree can be overwritten by the other trees due to the wisdom of crowds. Thus, random forests are less likely to make errors compared to a decision tree.





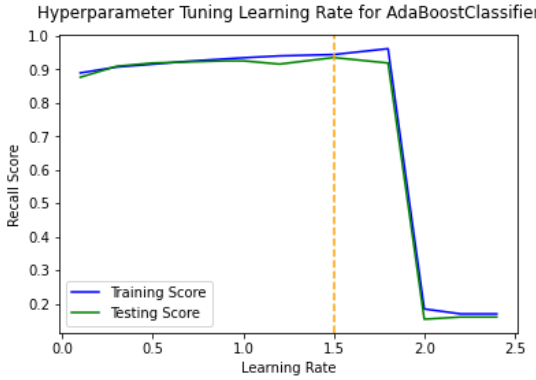
We believe that the random forests method would give us a more accurate model, due to the aforementioned wisdom of crowds. Based on the hyperparameter tuning models, we have decided that the optimum number of trees is 300, and the optimum tree depth is 15, based on the f1 scores, which reduce the Type 2 error rate.

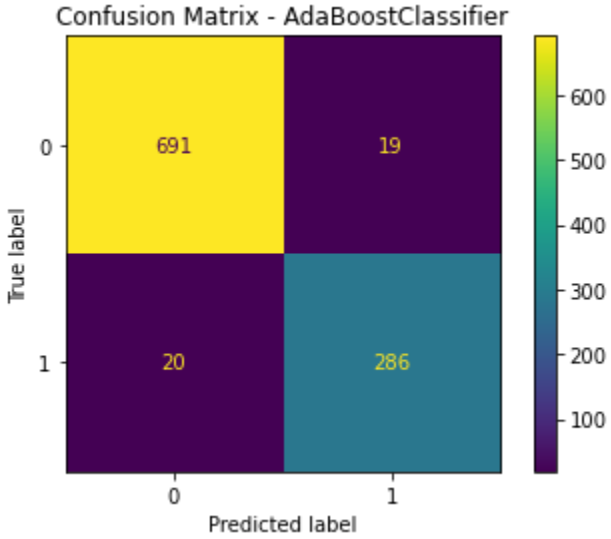


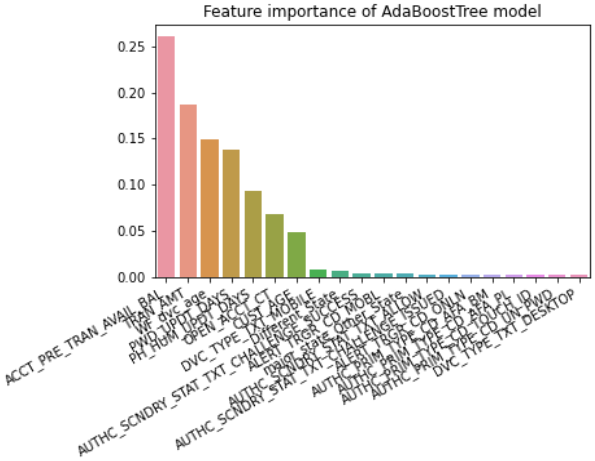
We ran the algorithm and proceeded to test the accuracy of the model on our test set. The results are shown above in the confusion matrix. Surprisingly, both overall accuracy fell and Type II error rate increased, compared to the decision tree. Overall accuracy fell, from 95.47% to 94.78%, and Type II error rate increased from 2.07% to 3.74%. Compared to the decision tree, the random forests algorithm is suboptimal, especially because of the increase in Type II error, which was something we wanted to minimize.

AdaBoost Tree

Since random forest did not provide an improvement on the decision tree model we thought that we could do some more analysis by attempting a boosted tree using the AdaBoost algorithm. Adaboost could enable us to improve performance of a single decision tree by iteratively learning information from misclassification samples. After tuning the hyperparameter, we pick 90 iterations as optimal value, and successfully achieve higher accuracy(96.23%). However, the number of false negatives is larger than that of the decision tree. Then we realized that since adaboost tree learnt both false negative and false positive samples iteratively, therefore the optimal parameter may not be the best for our problem. So we decided to tune the hyperparameter more specifically based on recall score and chose 450 iterations and 1.5 learning rate. Finally we achieved 96.16% accuracy with 20 false negative errors on the test set.







Observing the result, we found that the features "Account Pre Transaction Available Balance","Transaction Amount","Wells Fargo Device Age", "Password Update Days" and "Phone Number Update Days" are most important for this classifier to detect fraud problems.

IV. Results and Discussion

Below is a summary table showing the overall accuracy of each of our models predictions against the testing set. As we can see the boosted tree with the AdaBoost algorithm is the most accurate at 96.23% followed closely by the Decision Tree and the Random Forests models with 95.47% and 94.78% accuracy respectively with the K nearest Neighbors model performing significantly worse than the other 3. This initial analysis of the results makes us believe that our AdaBoost model will most likely be the best to use when predicting fraudulent transactions.

Model Type	Accuracy
K Nearest Neighbors	83.76%
Random Forests	94.78%
Decision Tree	95.47%
AdaBoost	96.16%

As stated before, this accuracy is a good baseline indicator for the performance of our models however when considering the real world implications of fraud classification the types of errors must be considered more closely in order to identify the best model for our purposes. Below is a table comparing the type 1 and type 2 error rates for each of the models. Again, we see that the AdaBoost model is the most accurate model, with the lowest Type 2 error rate of the four models presented.

Model Type	Type 1 error: False Positives	Type 2 error: False Negatives
K Nearest Neighbors	7.09%	9.15%
Random Forests	1.48%	3.74%
Decision Tree	2.46%	2.07%
AdaBoost	1.87%	1.97%

One shortcoming of the results is that only the AdaBoost was optimized to minimize Type 2 Errors. Had other models been given the same treatment (to minimize Type 2 Errors), the results of the model could be different. That could be a further area of exploration.

Another shortcoming of our results that we noticed came from the fact that we were not able to get the unique Customer ID codes due to privacy. These codes would identify the specific customer that made each transaction. We felt there were many features we could have created by linking multiple transactions

from a specific customer, for example the person's average transaction amount compared to the actual transaction amount in question or most common device type used by that specific customer compared to the device type used for the specific transaction in question. We feel these extra features that would identify trends in a specific customer's transactions could provide great help to our models. And we felt that if a bank like Wells Fargo was to further this dive into fraud classification these features would be very interesting to look into with respect to finding the best classification model possible.

V. Conclusion

By running these four algorithms, and testing their accuracy on the testing set, we have concluded that the AdaBoost model is the most accurate model for detecting fraudulent transactions among the elder subset of consumers, due to their low Type 2 error rate (1.97%). Decision tree is a close second, with a Type 2 error rate slightly higher than the Type 2 error rate of the AdaBoost model (2.07%).