INDIVIDUAL ASSIGNMENT

Juan Jose Medina

INTRODUCTION

We are given a Data set to select and explain 5 different machine learning models, and setup a benchmark for those 5 models and select the best one based on different metrics.

For this project we need to :

- Select 5 machine learning predictive algorithms
- Explain the general idea behind each algorithm
- Clean the assigned database
- Preprocess the database
- Apply the predictive models on the Base Table
- Setup the benchmark experiment to select the best performing model
- The user of different tools to improve our models such as Cross Validation, Feature Selection, Evaluation Metrics

TABLE

For the database we are going to use the Bank Marketing Database, which is composed of 20000 observations and a total of 21 variables which can be separated as the following

ID

- ClientID : gives a unique value per client, in order to distinguish from other clients

PREDICTORS

- age : the age from the client
- job : tells the type of job that the user currently has
- marital : the marital situation of the current user such as single, married, divorced

- education: the education level of the user, whether he finished basic education, up to university
- default: whether if the credit is in default or not
- loan : tells if the user has a personal loan or not
- contact: the media where the user was contacted, it can be cellular or telephone
- month: last month on which the user was contacted
- day_of_week: day of the week when the last contact was made
- campaign: the total number of contacts made with the user for thi specific campaign
- pdays: time gap in days when the user was last contacted
- previous: number of contacts made with the user before the actual campaign
- poutcome: the result of the previous campaign, whether it was success, failure, or non existent
- emp.var.rate: employment variation rate
- cons.proce.idx: consumer price index
- cons.conf.idx: consumer confidence index
- euribor3m: auribor 3 month rate
- nr.employed: number of employees

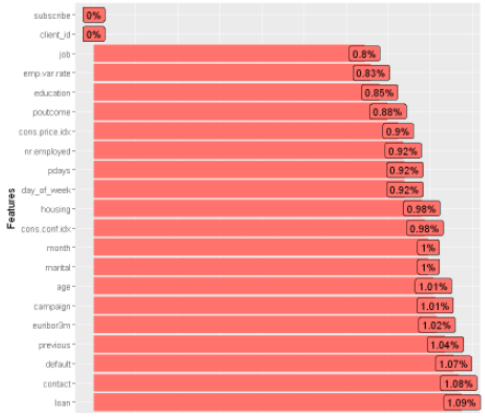
TARGET

- SUSCRIBE: indicates whether the user subscribed or not

DATA CLEANING

As a first step for the data cleaning after loading the table we want to know if there are any missing values on the database, there we discovered that there are some missing values on most of the columns

```
client_id: 0 age: 202 job: 161 marital: 199 education: 170 default: 214 housing: 195 loan: 219 contact: 217 month: 199 day_of_week: 185 campaign: 203
pdays: 185 previous: 209 poutcome: 175 emp.var.rate: 165 cons.price.idx: 181 cons.conf.idx: 197 euribor3m: 204 nr.employed: 184 subscribe: 0
```



It seems that we have a average of 1% per column of missing values, but the missing values were not on the same users, rather they were dispersed on multiple users. In this scenario the best option is to replace them rather than to drop them.

To know the best way to replace our missing values the best option is to view what's inside our variables, we can see that on most of the categorical variables we have a "unknown" value, therefore this is the best option to replace the null values on categorial, and for the numerical we got the mean of the value, after making our split into test train, to prevent data leakage.

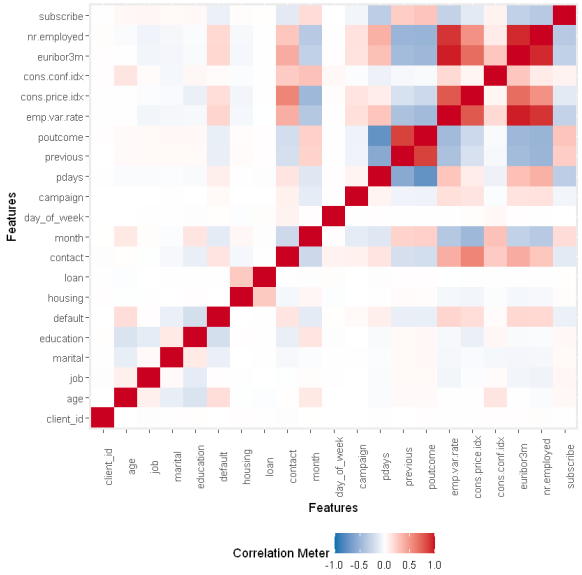
ENCODING

For the next step in or preprocessing we encoded all the Char variables, so we are able to run them on our predictive models. For this step we used the label encoder.

Only the variable of education that is an ordinal variable we encode it in a different manner, taking into account the levels of literacy as the order.

CORRELATION

We decided to plot a correlation matrix so see if there are variables that may affect our model, found out that nr.employed and euribor3m have a strong positive correlation and euribor3m and emp.var.rate also have a positive correlation, on the other hand poutcome and campaign have a negative correlation, but none has a strong correlation with our target variable



MODELS

Multiple Logistic Regression

Our first model is a GLM model, this model is uses to predict a class or category based on multiple predictors, this is best used to predict binary outcomes where our target variable will be a 0 or a 1.

The logistic regression is an extension of the linear model in which the prediction outcome is shaped in an S curve to facilitate the distinction on binary predictions. What this model does is predict an outcome based on the linear relationship between X and Y.

Now since we want to classify we need to use the logistic regression , to fit this model we use a method of Maximum Likelihood

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}.$$

The output of this model will always produce an S shaped curve, this is able to better capture the range of probabilities better than the linear regression.

```
Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -26.452934  24.625979  -1.074 0.282738
age              0.005333   0.002733   1.952 0.050974 .
marital1         0.045928   0.098215   0.468 0.640050
marital2         0.218448   0.110617   1.975 0.048289 *
marital3         0.502991   0.256087   1.964 0.049514 *
marital4         1.449564   0.522779   2.773 0.005558 **
default1        -0.208869   0.086299  -2.420 0.015507 *
default2        -8.595975 138.796790  -0.062 0.950617
contact1        -0.372478   0.097847  -3.807 0.000141 ***
month1           0.195706   0.132515   1.477 0.139713
month2          -0.349920   0.136848  -2.557 0.010558 *
month3          -0.640851   0.126542  -5.064 4.10e-07 ***
month4          -0.064902   0.156904  -0.414 0.679137
month5          -0.123229   0.172133  -0.716 0.474059
month6           1.055355   0.198035   5.329 9.87e-08 ***
month7          -0.016174   0.181065  -0.089 0.928823
month8          -0.098514   0.187403  -0.526 0.599110
month9           0.387293   0.276141   1.403 0.160761
day_of_week1    -0.024589   0.089999  -0.273 0.784691
day_of_week2     0.012218   0.093643   0.130 0.896195
day_of_week3     0.083142   0.091361   0.910 0.362804
day_of_week4    -0.237572   0.093095  -2.552 0.010713 *
campaign        -0.036441   0.014288  -2.550 0.010761 *
poutcome1       -0.422676   0.091887  -4.600 4.23e-06 ***
poutcome2        0.851601   0.234251   3.635 0.000278 ***
emp.var.rate    -0.426020   0.111015  -3.837 0.000124 ***
cons.price.idx   0.530918   0.180711   2.938 0.003304 **
cons.conf.idx    0.020793   0.007526   2.763 0.005732 **
nr.employed     -0.004663   0.001640  -2.843 0.004471 **
not_contacted   -0.498191   0.209226  -2.381 0.017261 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The output of this model gives us the betas for each variable, that are displayed on the image and the significance value of each variable given a determined significance rate, which are seen as the “***” on the table.

To train our first model the first step is to fit the GML model with all our variables. After this we applied a Stepwise The direction of this stepwise is backwards since we have more rows than columns.

The result from the stepwise told us that the best set of variables for our model where the following .

```
Call:
glm(formula = subscribe ~ age + marital + default + contact +
    month + day_of_week + campaign + poutcome + emp.var.rate +
    cons.price.idx + cons.conf.idx + nr.employed + not_contacted,
    family = binomial, data = train)
```

The next step was to train our new model with the selected variables and then apply a Cross validation with a total number of fold of 10, what this does is this, splits the data into 10 parts, take 9 of them to train the model and test it on the 10th part, then repeat the process replacing the part that is used. Then we get the mean of those metrics to get a non biased result.

Since the cross validation is done on a model that is for classification , the metrics that we get are going to be the Accuracy of the model and the Kappa Value.

Accuracy	Kappa
0.8969332	0.2798151

As we see on out first model, we got an accuracy of 89.69% and a Kappa value of 0.2798 the accuracy is good, but it can lead to incorrect assumptions, that can happen when the response rate is low, and the model predicts most of the predictions as a “0”, giving a high value on the True False values thus increasing the accuracy of the model. And the Kappa Score is a form of correlation coefficient which tell us the measure of agreement between two rates, this measure is quite controversial since depending on the field of study different values can been seen as problematic, therefore we will only take into account the closer the Kappa value gets to 1 the better.

Given this we need more measure values to know which is the best performing model and how well our model is performing.

To analyze our model even more we plot a confusion matrix that is done based on the predictions made on the test table

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4384  416
         1   73  127

               Accuracy : 0.9022
                 95% CI : (0.8936, 0.9103)
    No Information Rate : 0.8914
    P-Value [Acc > NIR] : 0.006902

                  Kappa : 0.301

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9836
            Specificity : 0.2339
         Pos Pred Value : 0.9133
         Neg Pred Value : 0.6350
              Precision : 0.9133
                 Recall : 0.9836
                     F1 : 0.9472
             Prevalence : 0.8914
         Detection Rate : 0.8768
   Detection Prevalence : 0.9600
      Balanced Accuracy : 0.6088

       'Positive' Class : 0
```

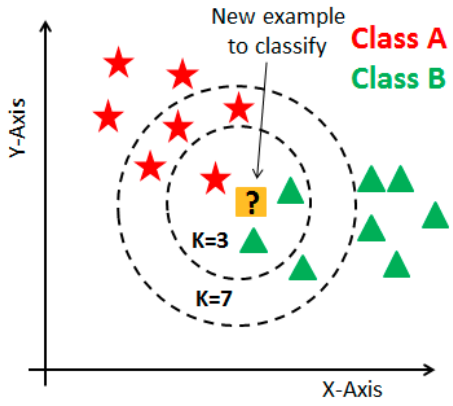
Here what we want to analyze is out F1 score, that us a score that compares the precision and recall of classifier models, in which the precision is the fraction True positives out of the true positives and false positives, and the recall is the fraction of true positives out of true positives and false negatives. In other words, the closer the F1 score to 1 the more precise and better recall the model has.

Finally, we want to analyze the metric of the AUC, which tells how much the model is capable of distinguishing between classes, the higher the better

```
"AUC = 0.774166666666667"
```

KNN

The next model that we trained is the K nearest neighbor, this model is a type of supervised algorithms, this tries to predict the class by calculating the distance between the test data and the trained centroids, the more neighbors of a specific class in a determined K value the higher the probability of that new entry being classified as the same class.



So, in this scenario the new values with a k of 3 has more probability of being classified as a Class B, but when we increase our K to 7 we see that for class A we have 4 neighbors and only 3 neighbors for class B, therefore this new value would be classified as a Class A

Since this model is a little different, we don't need to train the model, we directly make the predictions based on our Train set. we applied a Cross validation with a total of 10 folds to the model. in which we can determine that the K value to get the best accuracy and Kappa value is 9.

```
k-Nearest Neighbors

15000 samples
   20 predictor
    2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 13500, 13500, 13500, 13500, 13500, 13500, ...
Resampling results across tuning parameters:

  k  Accuracy   Kappa
  5  0.8875332  0.3057714
  7  0.8899332  0.3023267
  9  0.8921995  0.3058210

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 9.
```

Given this value we now make our predictions with the best K value and get a confusion matrix to analyze our metrics.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4366  383
         1   91  160

               Accuracy : 0.9052
                 95% CI : (0.8967, 0.9132)
    No Information Rate : 0.8914
    P-Value [Acc > NIR] : 0.0007666

                  Kappa : 0.359

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9796
            Specificity : 0.2947
         Pos Pred Value : 0.9194
         Neg Pred Value : 0.6375
              Precision : 0.9194
                 Recall : 0.9796
                     F1 : 0.9485
             Prevalence : 0.8914
         Detection Rate : 0.8732
   Detection Prevalence : 0.9498
      Balanced Accuracy : 0.6371

       'Positive' Class : 0
```

Here we can see that the results from the previous model are rather similar to the GML model, having a good accuracy of 90%, a Kappa value of 0.30 and a good F1 score if 0.9485, never the less we want to compare our models with multiple metrics, so we get the AUC

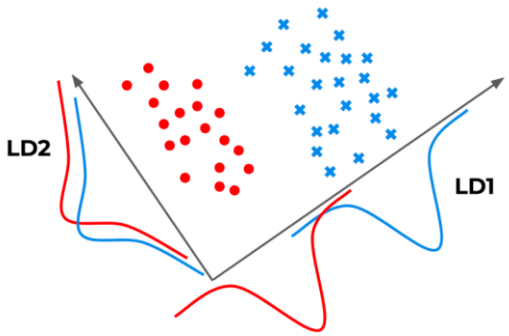
```
"AUC = 0.778400820806058"
```

On which we see that it marginally increased from the previous model.

LDA

The next model in our list is the Linear Discriminant Analysis. This model is better for classification predictions that have 2 or more classes, this model is more stable than the logistic regression.

This model works by finding the directions that maximizes the separation between class, these directions are called linear discriminant and are a combination of the predictor variables, for this reason we lose a lot of interpretabilities when we run this model.



This model have some advantages over the logistic regression, such as being more stable when the classes are not wee separated, also if the number of rows is small and the distribution is normal, this model is more stable than the logistic

For this model to work we must make several assumptions such as that our predictors have a normal distribution and that the different classes have specific means in order to separate the 2 distributions.

For the LDA we follow a similar process as the GLM, first start with a stepwise feature selection, having the direction as backward.

```
correctness rate: 0.89113;  starting variables (26): client_id, age, job, marit
al, educationbasic.6y, educationbasic.9y, educationhigh.school, educationillite
rate, educationprofessional.course, educationuniversity.degree, educationunknow
n, default, housing, loan, contact, month, day_of_week, campaign, pdays, previo
us, poutcome, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employ
ed
correctness rate: 0.893;  out: "cons.conf.idx";  variables (25): client_id, ag
e, job, marital, educationbasic.6y, educationbasic.9y, educationhigh.school, ed
ucationilliterate, educationprofessional.course, educationuniversity.degree, ed
ucationunknown, default, housing, loan, contact, month, day_of_week, campaign,
pdays, previous, poutcome, emp.var.rate, cons.price.idx, euribor3m, nr.employed

 hr.elapsed min.elapsed sec.elapsed
       0.00        0.00       39.95
```

Next step is to measure our improved model with a cross validations, with the same settings as the previous

```
Linear Discriminant Analysis

15000 samples
   19 predictor
    2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 13500, 13500, 13500, 13500, 13500, 13500, ...
Resampling results:

  Accuracy   Kappa
  0.8877341  0.3583412
```

The results from the CV are similar in accuracy and we see an improvement on the Kappa score.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4267  349
         1  190  194

               Accuracy : 0.8922
                 95% CI : (0.8833, 0.9007)
    No Information Rate : 0.8914
    P-Value [Acc > NIR] : 0.4391

                  Kappa : 0.3611

 Mcnemar's Test P-Value : 1.007e-11

            Sensitivity : 0.9574
            Specificity : 0.3573
         Pos Pred Value : 0.9244
         Neg Pred Value : 0.5052
              Precision : 0.9244
                 Recall : 0.9574
                     F1 : 0.9406
             Prevalence : 0.8914
         Detection Rate : 0.8534
   Detection Prevalence : 0.9232
      Balanced Accuracy : 0.6573

       'Positive' Class : 0
```

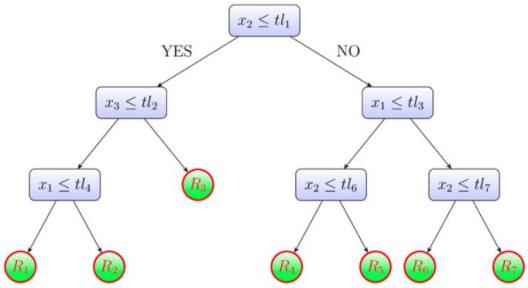
On our confusion matrix we can see that we have a better accuracy on the test data and a F1 score also like the previous models.

Finally, we see that the AUC of this model is rather low compared to the previous models, but we will compare all the, at the end

```
"AUC = 0.714800873772386"
```

DECISION TREE

Decision trees belongs to the supervised learning algorithms, this model can be used for regression and classification problems. On this model we start from the root which contains all our observations and then we make a split based on a single variable to divide it into 2 groups, after this we continue to make splits until we reduce the entropy of our data within each group, the entropy is the randomness that exists within the dataset. And we make this by making the splits based on simple decision rules, ending up with our decision nodes that contains our classes.



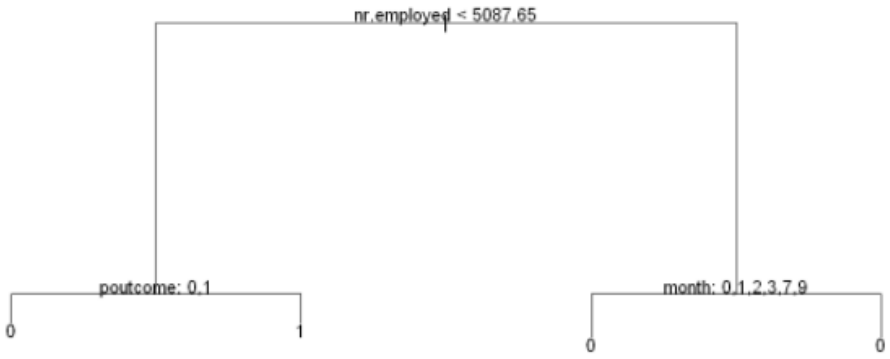
The advantages of the decision trees are its highly interpretability, since some people believe that this is closely similar to human decision making, therefore easier to interpret. And trees can handle qualitative predictors with out the need to create dummy variables.

On the other hand, tree are not robust, meaning a small change in data can greatly affect this model and the model has not the best accuracy compared with other models

To train this model there is no need to use a feature selection, Since the decision tree on its own select the most significant features and make the rules based on them, so we start my training our data with all the variables.

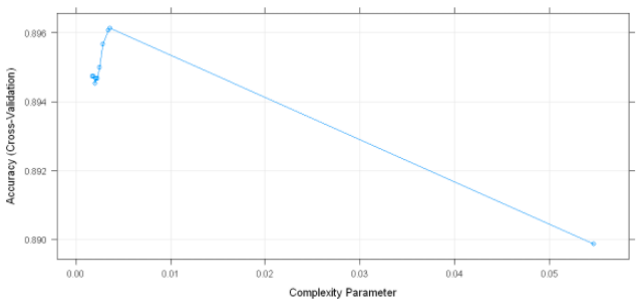
```
Classification tree:
tree(formula = subscribe ~ ., data = train)
Variables actually used in tree construction:
[1] "nr.employed" "pdays"       "month"
Number of terminal nodes:  4
Residual mean deviance:  0.5829 = 8741 / 15000
Misclassification error rate: 0.1026 = 1539 / 15000
```

Here we can see that our miss classification error is 10% and that the decision tree only uses 3 variables for its predictions, the first split is done with the “nr.employed” saying if its lower thank 5087.65 we continue with the poutcome variable and if its higher we use the month variable, When making the other split using poutcome greater than 0.5 we have our terminal node with the class of 1 or people that subscribed, if the poutcome value is less than 0.5 then it’s a 0, and on the other side even if we split based on month there is no difference on the output since both terminal nodes are classified as a 0.



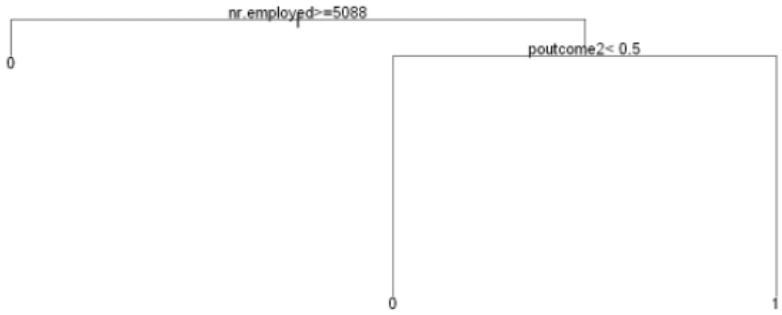
Having this tree, it seems that our model is under fitted since we only consider 2 variables to make our predictions. In order to improve our model, we do some parameter tunning, while also applying a Cross validation. This gives us that if we use a cp of 0.004050926. we get a better accuracy

```
No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 13500, 13500, 13500, 13500, 13500, 13500, ...
Resampling results across tuning parameters:

  cp           Accuracy   Kappa
  0.001350309  0.8932663  0.3016944
  0.001446759  0.8933331  0.3008829
  0.001736111  0.8941331  0.2964600
  0.002025463  0.8943331  0.2878439
  0.002507716  0.8944666  0.2877327
  0.002604167  0.8948666  0.2840381
  0.002893519  0.8948666  0.2840381
  0.003182870  0.8951334  0.2678341
  0.004050926  0.8960669  0.2521261
  0.052372685  0.8892660  0.1145450

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was cp = 0.004050926.
```



We also can see that if we try to make our model more complex the Accuracy of it is greatly affected.

Now with this parameters can fit the model again, but we got a similar result as the previous



Ignoring this we continue to plot our confusion matrix, with the predictions made on the test set we can see that our model has a good accuracy and a Kappa value similar to the previous models.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4422  436
         1   35  107

               Accuracy : 0.9058
                 95% CI : (0.8974, 0.9138)
    No Information Rate : 0.8914
    P-Value [Acc > NIR] : 0.000466

                  Kappa : 0.28

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9921
            Specificity : 0.1971
         Pos Pred Value : 0.9103
         Neg Pred Value : 0.7535
              Precision : 0.9103
                 Recall : 0.9921
                     F1 : 0.9494
             Prevalence : 0.8914
         Detection Rate : 0.8844
   Detection Prevalence : 0.9716
      Balanced Accuracy : 0.5946

       'Positive' Class : 0
```

```
[1] "AUC = 0.823469489663437"
```

Having this good accuracy and a good AUC we can make the assumption that there is a really strong relationship between nr.employed and poutcome with our target variable.

RANDOM FOREST

Our final model is the Random Forest which is also a supervised learning algorithms, this model works similar to the decision tree, and as its name implies it creates random samples of multiple trees, then takes their majority vote for classification, one difference is that this can have datasets with continuous variables and performs better on classification problems.

On this model each tree is different, not all variables are considered while making a single tree, and since this model doesn't use all the variables there is no need for feature selection.

To train our model we can start with a grid search to get the best mtry value which is the number of maximum variables considered by a tree, after running this and applying a Cross validation we get the result that using a mtry of 7 gave us the best accuracy and a relevant Kappa value.

```
Random Forest

15000 samples
   20 predictor
    2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 12000, 12001, 12000, 11999, 12000
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   1    0.8864668  0.02864641
   7    0.8951332  0.31096911
   9    0.8948001  0.31945440
  10    0.8954001  0.32316700
  14    0.8928002  0.31692661
  18    0.8922670  0.31806263
  21    0.8916668  0.31888088
  23    0.8914668  0.31430947
  33    0.8908001  0.31707809
  34    0.8909337  0.31585588
  39    0.8900671  0.31198197
  42    0.8889337  0.31078731
  43    0.8902004  0.31719671
  46    0.8903338  0.31797571

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 10.
```

Finally, we train our model with the best parameters and plot our confusion matrix and AUC, we can get the F1 Score of 0.9467and a AUC of 0.75705.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4335  366
         1  122  177

               Accuracy : 0.9024
                 95% CI : (0.8938, 0.9105)
    No Information Rate : 0.8914
    P-Value [Acc > NIR] : 0.00605

                  Kappa : 0.372

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

            Sensitivity : 0.9726
            Specificity : 0.3260
         Pos Pred Value : 0.9221
         Neg Pred Value : 0.5920
              Precision : 0.9221
                 Recall : 0.9726
                     F1 : 0.9467
             Prevalence : 0.8914
         Detection Rate : 0.8670
   Detection Prevalence : 0.9402
      Balanced Accuracy : 0.6493

       'Positive' Class : 0
```

```
"AUC = 0.757058734390107"
```

COMPARISON

Now that we have all our models fitted and tested we can compare them based on the 4 metrics that we got from our results, the Accuracy, Kappa Score, F1 Score and AUC.

Column1	GML	KNN	LDA	TREE	RF
Acc	0.89693	0.8921995	0.8877341	0.8960669	0.8954001
Kappa	0.2798	0.35821	0.3583412	0.252126	0.323167
F1	0.9472	0.9485	0.9406	0.9494	0.9467
Auc	0.774166	0.7784	0.7148008	0.8318861	0.75705

We can see that all our models have a good accuracy of 89% having the RF model on the first place by a small margin.

Analyzing the Kappa value that measures the agreement we see that all models have a low Kappa score, usually its recommended that we have a Kappa score of 0.8 with a minimum of 0.4, in our case the closest one is the LDA model with a value of 0.35 and the worst one is Tree with a value of 0.25.

The next metric is the F1 Score on which all the models have a good score getting close to 1 with values of 0.94 and 0.95, analyzing only on this metric the best performing is
Tree that wins with a small margin.

Finally the AUC, we have all or models with a relative good AUC that ranged from 0.71 to 0.0.83, with TREE as the best model on AUC ad LDA with the lowest one at 0.83 and 0.71 respectively.

Having this information in mind we can suggest that the best model to predict on this scenario is the KNN model, having a accuracy of 0.8921 which is high enough a kappa value of 0.358 being on second place for this metric, the second best f1 score, telling us that this model is more precise and has a better recall and a second best AUC of 0.7784, telling that this model is able to differentiate between its predictive classes.0

REFERENCES

http://www.sthda.com/english/articles/36-classification-methods-essentials/

https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

https://en.wikipedia.org/wiki/F-score

https://en.wikipedia.org/wiki/Precision_and_recall

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900052/#:~:text=Cohen%20suggested%20the%20Kappa%20result,1.00%20as%20almost%20perfect%20agreement.

https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4#:~:text=K%2Dnearest%20neighbors%20(KNN),closet%20to%20the%20test%20data.