JUNE 10, 2020

BANK MARKETING DATA ANALYSIS

INTRODUCTION

A bank supports many clients with a wide range of interests and collects a massive amount of data from its phone outreach program. This bank was aiming to understand the likelihood of a prospective client to subscribe to a term deposit at the bank.

Our team sought to help the bank find out more about these clients and their interests in subscribing to a term deposit, because of our interests in this bank and its phone marketing campaigns. Last month, we were able to help the bank by volunteering to build a classification model by performing supervised and unsupervised classification methods to calculate a client's likelihood of signing up for a term deposit.

OVERVIEW & OBJECTIVE

The dataset the bank provided to us was comprised of data related to direct phone marketing campaigns conducted by the banking institution.

With 17 variables and 45,211 observations in the data, we first wanted to be able to decipher which variables were most important for this analysis. The dataset contains 10 categorical variables, 6 numeric variables, and 1 binary variable. 16 of the variables are predictor variables, with one variable Y as the dependent variable. The dataset is also very clean, with no missing values. However, many of the survey results are "unknown", so our ***first*** objective of this analysis was to produce a clean dataset to do the analysis on.

Our ***second*** objective was to use the unsupervised method of cluster analysis to determine which features of the customer were most important for finding the likelihood of signing up for a term deposit.

Third, we wanted to run a supervised method of analysis on the clean, feature-vetted data. We hypothesized we could use any supervised method; it was just on us to pick which one produces the best results.

In the coming sections of this report, we will explain our data, analysis results, and conclusion of the analysis. We hope, like the bank did, you'll find our process interesting, and our results extremely helpful.

DATA EXPLORATORY ANALYSIS

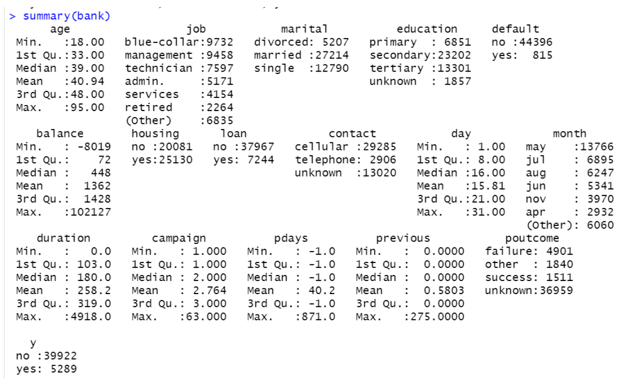
As mentioned above, the dataset that we have consists of 17 variables and 45,211 observations. The observations or records are ordered by date (from May 2008 to November 2010). Below is a tabular description of each of the variables:

Variable name	Variable Type	Description
Age	Numeric	The age of the bank client.
Job	Categorical	The type of job the bank client has. Some examples include admin, management, retired, student, unemployed, and unknown.
Marital	Categorical	The marital status of the bank client: divorced (also encompassing widowed), married, single, or unknown.
Education	Categorical	The education level of the bank client. Some examples include basic 4 years, high school, illiterate, university degree, and unknown.
Default	Categorical	This variable serves as an answer to the question "Does this bank client have credit in default?". The options are yes, no, and unknown.
Balance	Numeric	How much money the bank client has to deposit.
Housing	Categorical	This variable serves as an answer to the question "Does this bank client have a housing loan?". The options are yes, no, and unknown.
Loan	Categorical	This variable serves as an answer to the question "Does this bank client have a personal loan?". The options are yet, no, and unknown.
Contact	Categorical	The communication type used to contact this bank client: cellular or telephone.
Day	Categorical	The day of the month of last contact (ranging from 1-31).
Month	Categorical	The month of the year of last contact: January, February, March, April, May, June, July, August, September, October, November, December
Duration	Numeric	The length of the conversation had with the bank client, in seconds.
Campaign	Numeric	The number of contacts performed during this specific campaign and for this client.
pDays	Numeric	The number of days since a bank client was last contacted from a previous campaign.
Previous	Numeric	The number of times this bank client was contacted before this campaign.
pOutcome	Categorical	The outcome of the previous campaign: failure, nonexistent, success.
Y	Binary	Indicates whether the bank client subscribed a term deposit.

When we found the summary of the variables from the report, we observed a lot of unknown values. We equated these unknown values to null or missing values, so we decided to treat them using either an imputation method or remove them. Additionally, we needed to check for the outliers and remove bias from the dataset before starting the analysis.

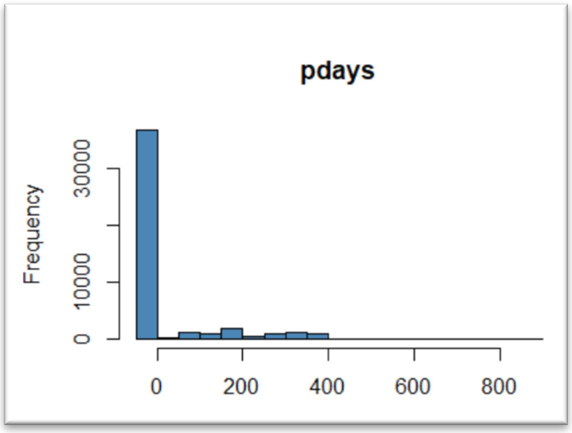
SUMMARY OF DATA

To start, we performed a summary() on the bank data. In this summary, variable ‘poutcome’ contains more than 75% of the unknown records so we decided to remove to drop ‘poutcome’.



```
> summary(bank)
      age                 job           marital          education     default
 Min.   :18.00   blue-collar:9732   divorced: 5207   primary  : 6851   no :44396
 1st Qu.:33.00   management :9458   married :27214   secondary:23202   yes:  815
 Median :39.00   technician :7597   single  :12790   tertiary :13301
 Mean   :40.94   admin.     :5171                    unknown  : 1857
 3rd Qu.:48.00   services   :4154
 Max.   :95.00   retired    :2264
                 (Other)    :6835
    balance       housing      loan            contact           day            month
 Min.   : -8019   no :20081   no :37967   cellular :29285   Min.   : 1.00   may    :13766
 1st Qu.:    72   yes:25130   yes: 7244   telephone: 2906   1st Qu.: 8.00   jul    : 6895
 Median :   448                           unknown  :13020   Median :16.00   aug    : 6247
 Mean   :  1362                                             Mean   :15.81   jun    : 5341
 3rd Qu.:  1428                                             3rd Qu.:21.00   nov    : 3970
 Max.   :102127                                             Max.   :31.00   apr    : 2932
                                                                            (Other): 6060
    duration         campaign          pdays          previous          poutcome
 Min.   :   0.0   Min.   : 1.000   Min.   : -1.0   Min.   :  0.0000   failure: 4901
 1st Qu.: 103.0   1st Qu.: 1.000   1st Qu.: -1.0   1st Qu.:  0.0000   other  : 1840
 Median : 180.0   Median : 2.000   Median : -1.0   Median :  0.0000   success: 1511
 Mean   : 258.2   Mean   : 2.764   Mean   : 40.2   Mean   :  0.5803   unknown:36959
 3rd Qu.: 319.0   3rd Qu.: 3.000   3rd Qu.: -1.0   3rd Qu.:  0.0000
 Max.   :4918.0   Max.   :63.000   Max.   :871.0   Max.   :275.0000

   y
 no :39922
 yes: 5289
```

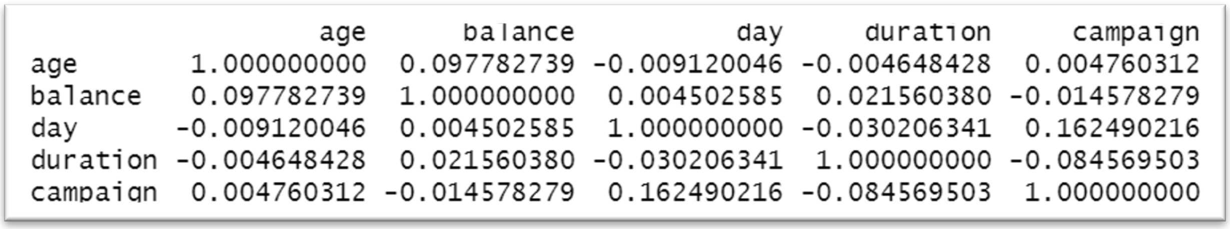
The 'previous' variable has a value ranging from 0 to 275, and contains the value '0' for more than 75% of the records. This skew in data informed us that no contacts were performed before this campaign for most of the records in the dataset. Because of the presence of so many outliers, we deemed the ‘previous’ the outliers as insignificant, and therefore removed the ‘previous ’ column.



We see in the graph above that 75% of the data have pdays = -1, which represents bank clients who were not contacted. We believed it would not add significant value to the analysis, so we dropped it from our dataset.

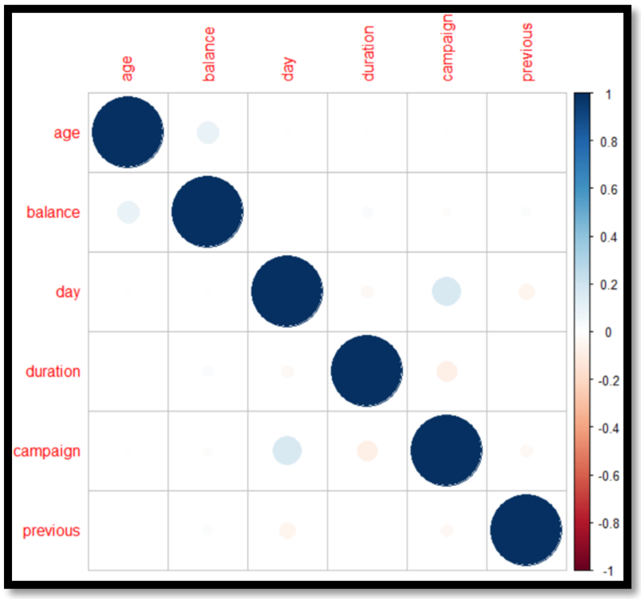
REDUNDANT VARIABLES

Next, we performed a redundancy check to remove redundant variables. For this we plotted the correlation matrix to analyze the correlation between all of the variables.



```
                  age      balance          day     duration     campaign
age       1.000000000  0.097782739 -0.009120046 -0.004648428  0.004760312
balance   0.097782739  1.000000000  0.004502585  0.021560380 -0.014578279
day      -0.009120046  0.004502585  1.000000000 -0.030206341  0.162490216
duration -0.004648428  0.021560380 -0.030206341  1.000000000 -0.084569503
campaign  0.004760312 -0.014578279  0.162490216 -0.084569503  1.000000000
```

Age and balance are positively correlated, meaning that as age increases, the expectation for clients is to include more money in each deposit to the bank. Campaign and day are also positively correlated, which showed us that as the days in the month increase and we arrive at the end of the month, the number of clients who are contacted increases.



Similarly, previous and day are negatively correlated, meaning that as the days and months come and go, the number of previous contacts decreases. Duration and campaign are also negatively correlated, meaning that less time is spent in contacting a larger number of bank clients over a span of campaigns.

CHI-SQUARE TEST

After checking for redundant and highly correlated values, we performed Chi-Square Tests to find the inter-dependency between the two categorical variables, job and y (whether the client will subscribe to the term deposit or not).

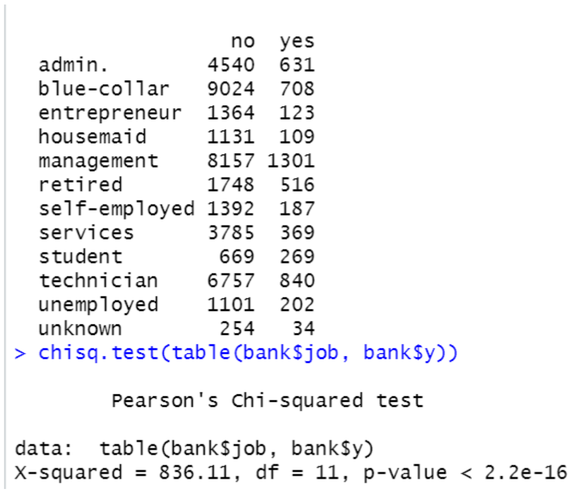
We created two hypotheses for this test, H0 (the null hypothesis) and H1:

H0: The two variables are not dependent

H1: The two variables are dependent

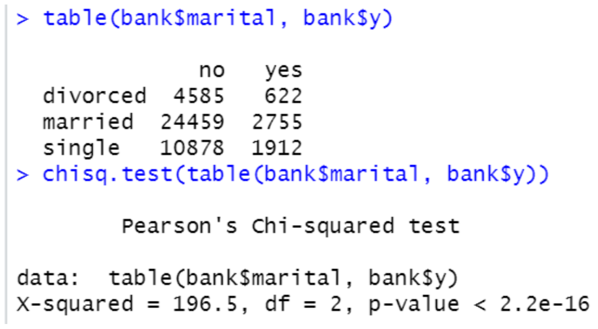
We first selected columns job, marital, education and contact to check whether they have an influence on y or not.

For a Pearson's Chi-Square test on job and y, the p value obtained is less than 0.05. With this result, we do not reject H0.



```
                no  yes
  admin.        4540  631
  blue-collar   9024  708
  entrepreneur  1364  123
  housemaid     1131  109
  management    8157 1301
  retired       1748  516
  self-employed 1392  187
  services      3785  369
  student        669  269
  technician    6757  840
  unemployed    1101  202
  unknown        254   34
> chisq.test(table(bank$job, bank$y))

        Pearson's Chi-squared test

data:  table(bank$job, bank$y)
X-squared = 836.11, df = 11, p-value < 2.2e-16
```

For a Pearson's Chi-Square test on marital and y, we also found the result of p < 0.05, so we do not reject H0 in this case as well.



```
> table(bank$marital, bank$y)

              no   yes
  divorced  4585   622
  married  24459  2755
  single   10878  1912
> chisq.test(table(bank$marital, bank$y))

        Pearson's Chi-squared test

data:  table(bank$marital, bank$y)
X-squared = 196.5, df = 2, p-value < 2.2e-16
```

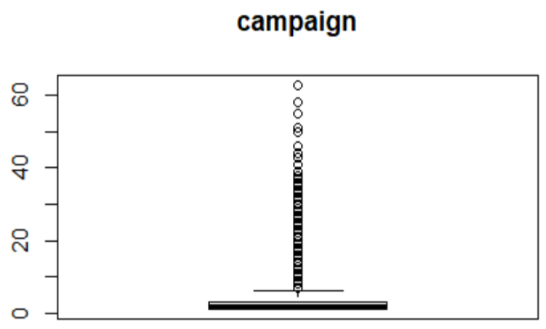
We observe the same result for education variable with y and contact with y.

For each of these tests, we do not reject the null hypothesis and conclude that there is a no dependency between the variables, and they do not influence y.

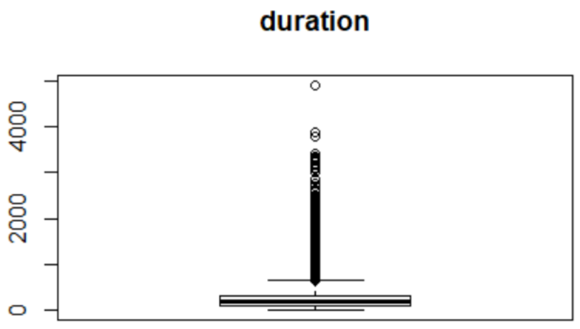
IDENTIFYING OUTLIERS

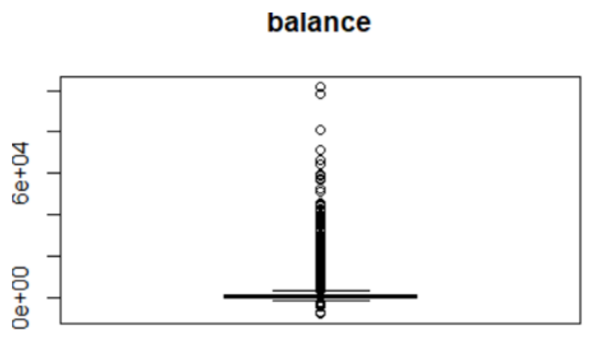
After the Chi-Square test, we proceeded with identifying the outliers. To do this, we created boxplots to help visually identify outliers. Outliers are represented in the boxplot visual as points that extend past the "whiskers".

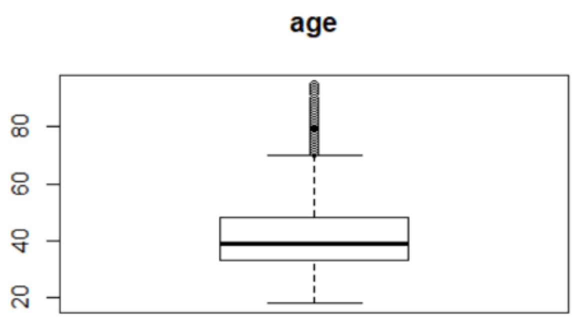
For the variable 'campaign', a large number of contacts performed (greater than 12) seemed unusual in this dataset, as it both felt like a large number of contacts to make and there were very few instances of these values. Because these values were clear outliers, we decided to address them.



'Duration', 'balance', and 'age' were important variables to us, so we did not neglect them. We instead performed capping on the data to replace outliers. To do this, we replaced outliers that were less than (Q1 - 1.5*IQR) with data from the 5th percentile, and the outliers greater than (Q3 + 1.5*IQR) with data from the 95th percentile.



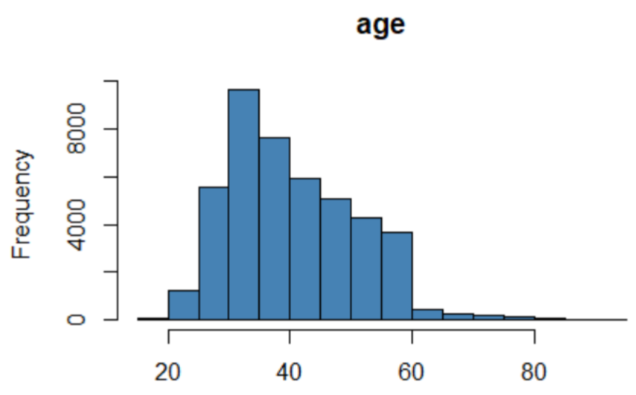


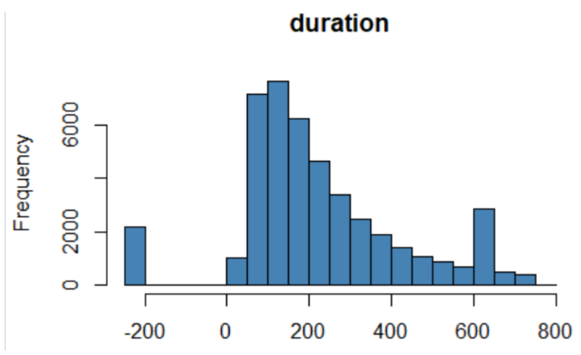


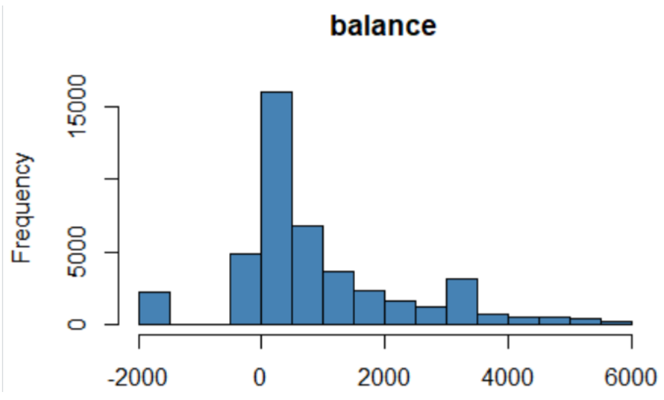
EXPLORATORY ANALYSIS

After we cleaned the data, we performed exploratory analysis.

In this analysis we first found that most of the bank clients in the dataset were of age 35+. We also found that the average conversations had with bank clients were around 180 seconds, and the balances of most clients were on the lower end, in the hundreds.







After running the summary statistics again, we found a few unknown values in variables such as job, education and contact. To fix these missing values, we imputed them using the mice package. Because each of the three variables was categorical, we used Polytomous Logistic Regression, "polyreg" to predict the values of the missing data. We ran 5 multiple imputations over 5 iterations.

This concludes our data cleaning portion of the analysis. We have successfully removed all outliers and imputed all instances of missing data.

CONVERTING CATEGORICAL VARIABLES

The number of **dummy variables** needed to represent a **categorical variable** depends on the number of values that the **categorical variable can** assume. So, in order to perform classification, we converted categorical variables to dummy variables. We also performed Label encoding on the ‘month’ variable.

Numerical Variables: Age, Balance, Day, Month, Duration, Campaign

Categorical Variables: Job, marital, education, default, contact, y (Dependent Variable)

We created dummy variables for each of the categorical variables: job, marital, education, default, and contact.

ANALYSES AND RESULTS

When we sought to choose an analysis method, we ran multiple supervised methods and K-means clustering to see which two, whether it was two supervised or one supervised and one unsupervised, provided the best results. We decided to try the following: decision tree, ANN, and KNN.

In order to perform supervised classification techniques, we performed min-max normalization, by splitting the training data and test data using an (80:20) split. Also, before running supervised classification techniques, we used SMOTE to balance our imbalanced dataset to get better model results.

K-MEANS CLUSTERING

We understand K-means clustering to be a type of unsupervised learning algorithm used for unlabeled data (data without defined categories or groups). This algorithm finds groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity.

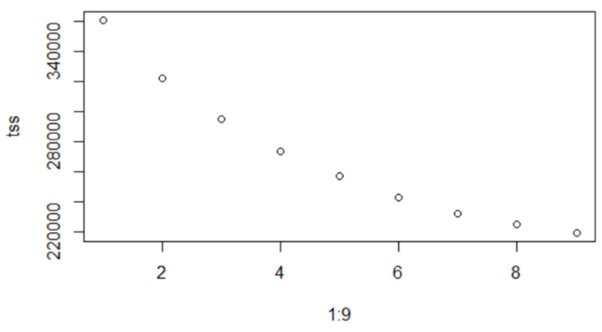
The K-means clustering algorithm will give us the following results:

- The centroids of the K clusters, which can be used to label new data
- Labels for the training data (each data point is assigned to a single cluster)

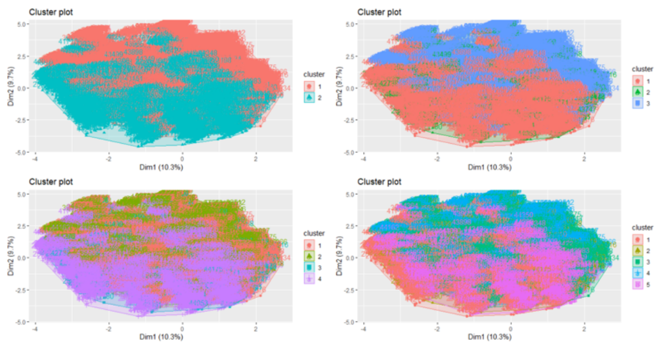
Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.

Once the algorithm has been run and the groups are defined, any new data can be easily assigned to the correct group.

Scaling is required for K-means as it uses a distance metric to calculate the distance between variables. So, we decided to scale only the first 6 variables since they are continuous or ordinal variables while others are binary variables. We selected K from the range of 1 to 9 to plot the total sum of squares.

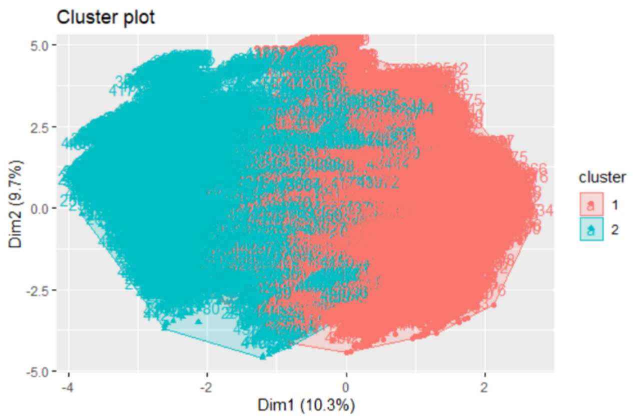


In this elbow curve plot, which we used to choose K, we can see that it does not depict a proper elbow curve, so to be safe, we chose to run K-means for K = 2, 3, 4 and 5 clusters.

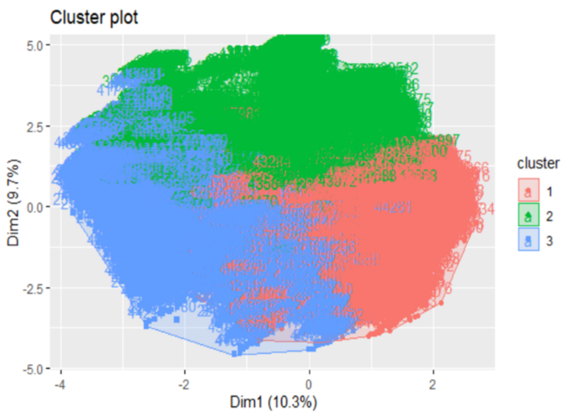


Since our data contains more binary variables, instead of using Euclidean distance, we used Manhattan distance.

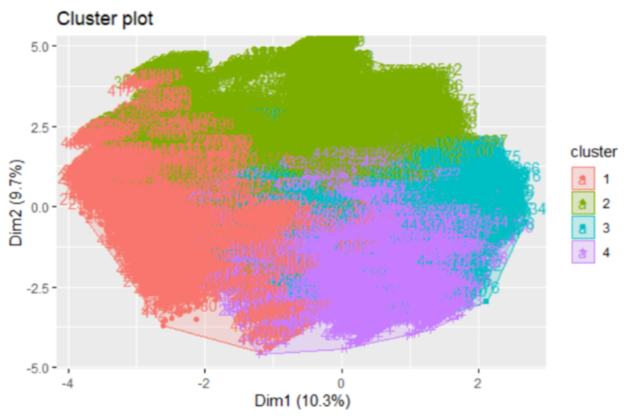
Additionally, since our data is large, we used the Clara function which considers a small sample of the data with fixed size (sample size) and applies the PAM algorithm to generate an optimal set of medoids for the sample.



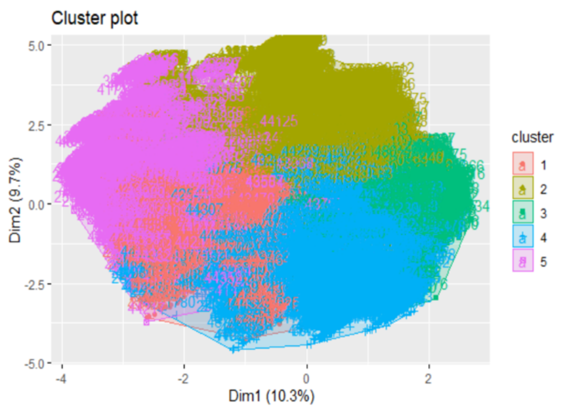
Clara 2 cluster



Clara 3 cluster

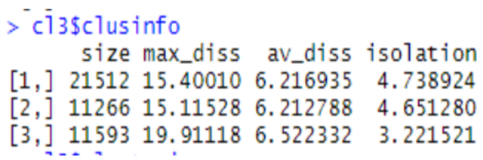


Clara 4 cluster



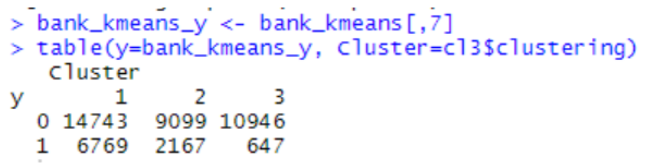
Clara 5 cluster

For Clara 3 cluster, we see data separation. We viewed the average dissimilarities of the clusters to their closest medoid, a method for measuring goodness, and determined that this model has a good goodness of fit.



```
> cl3$clusinfo
      size max_diss   av_diss isolation
[1,] 21512 15.40010 6.216935  4.738924
[2,] 11266 15.11528 6.212788  4.651280
[3,] 11593 19.91118 6.522332  3.221521
```

Additionally, we found the external validation of the cluster.



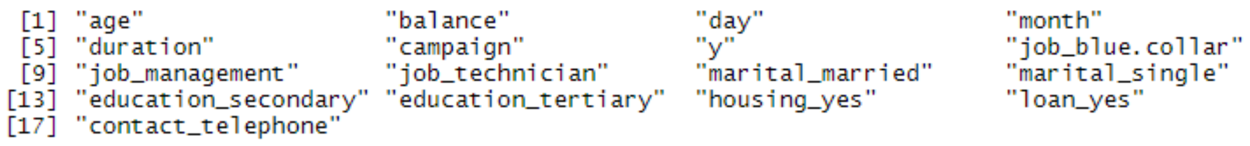
```
> bank_kmeans_y <- bank_kmeans[,7]
> table(y=bank_kmeans_y, Cluster=cl3$clustering)
   Cluster
y       1     2     3
  0 14743  9099 10946
  1  6769  2167   647
```

FEATURE SELECTION

We performed the below feature selection techniques to capture the important features:

- Decision Tree
- Random Forest
- Recursive Feature Elimination

After performing Feature Selection, we obtained 16 important features. The below screenshot contains the 16 important features along with the dependent variable y.

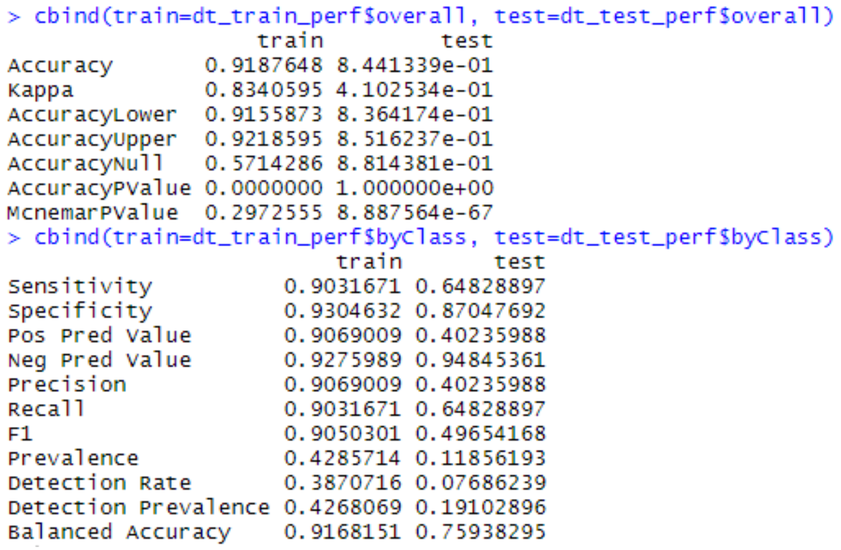


```
 [1] "age"                 "balance"             "day"                "month"
 [5] "duration"            "campaign"            "y"                  "job_blue.collar"
 [9] "job_management"      "job_technician"      "marital_married"    "marital_single"
[13] "education_secondary" "education_tertiary"  "housing_yes"        "loan_yes"
[17] "contact_telephone"
```

DECISION TREE

Decision Tree is a tree-based supervised learning algorithm. Predictive models built using Decision Tree have high accuracy, stability, and are very easy to interpret. Decision trees are used for both Regression and Classification. As the independent variable in our dataset is a categorical variable 'y' that has 2 classes, 'yes' and 'no', referring to whether the customer is going to subscribe to the term deposit or not, we thought to use decision tree for classification. Decision Tree can handle missing values, irrelevant and redundant variables without rescaling of the variables.

To tune the cost complexity parameter, or cp, we performed Hyperparameter Tuning to choose the cp that is associated with the smallest cross-validated error (highest accuracy).



```
> cbind(train=dt_train_perf$overall, test=dt_test_perf$overall)
                   train         test
Accuracy       0.9187648 8.441339e-01
Kappa          0.8340595 4.102534e-01
AccuracyLower  0.9155873 8.364174e-01
AccuracyUpper  0.9218595 8.516237e-01
AccuracyNull   0.5714286 8.814381e-01
AccuracyPValue 0.0000000 1.000000e+00
McnemarPValue  0.2972555 8.887564e-67
> cbind(train=dt_train_perf$byClass, test=dt_test_perf$byClass)
                         train       test
Sensitivity          0.9031671 0.64828897
Specificity          0.9304632 0.87047692
Pos Pred Value       0.9069009 0.40235988
Neg Pred Value       0.9275989 0.94845361
Precision            0.9069009 0.40235988
Recall               0.9031671 0.64828897
F1                   0.9050301 0.49654168
Prevalence           0.4285714 0.11856193
Detection Rate       0.3870716 0.07686239
Detection Prevalence 0.4268069 0.19102896
Balanced Accuracy    0.9168151 0.75938295
```

The performance information is as follows for the training and testing sets:

Training set: Accuracy = 92%, Kappa = 0.53, F1 = 0.90

Testing set: Accuracy = 84%, kappa = 0.41, F1 = 0.50

The model performs well on the data on which it was trained, but not on testing data, since the F1 measure (goodness of fit) value on the training dataset is good but on the testing dataset it is not. Additionally, Recall and Precision are high for the training set but not for the testing set. However, the performance information on the testing data is fairly consistent with the training data.

ARTIFICIAL NEURAL NETWORKS (ANN)

Artificial Neural Networks (ANN) is a supervised learning algorithm built of multi-layer fully connected neural nets called neurons. Each neuron makes simple decisions, and those decisions are fed to other neurons, organized in interconnected layers.

Together, the neural network can emulate any function, and answer practically any question, given enough training samples and computing power. A neural network has only three layers of neurons:

- An **input layer** that accepts the independent variables or inputs of the model
- Multiple **hidden layers**
- An **output layer** that generates predictions

Every node in one layer is connected to every other node in the next layer. Increasing the number of hidden layers makes the network deeper. We performed hyperparameter tuning using the 'nnet' package, and adjusted the size and decay. Size represents the number of nodes in the hidden layer.

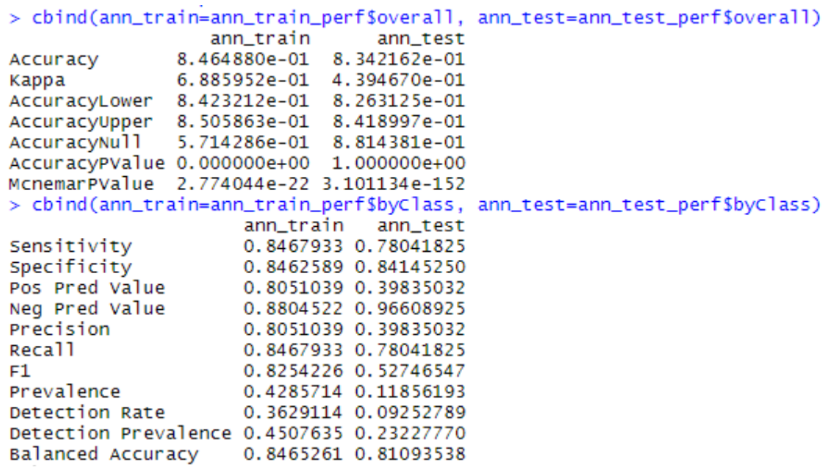
There can only be one hidden layer using nnet. Decay is weight decay which is a regularization parameter to avoid overfitting and adds a penalty for complexity.

The performance information is as follows for the training and testing sets:

Training set: Accuracy = 84%, Kappa = 0.69, F1 = 0.82

Testing set: Accuracy = 83%, kappa = 0.44, F1 = 0.53

Accuracy for the model is good on both the training and testing sets. The F1 measure (goodness of fit) value on the training set is good, but on the testing set it is fairly consistent with the training data. Precision is high on the training set but on testing it is low. Recall is high on both testing and training datasets.



```
> cbind(ann_train=ann_train_perf$overall, ann_test=ann_test_perf$overall)
                  ann_train      ann_test
Accuracy       8.464880e-01  8.342162e-01
Kappa          6.885952e-01  4.394670e-01
AccuracyLower  8.423212e-01  8.263125e-01
AccuracyUpper  8.505863e-01  8.418997e-01
AccuracyNull   5.714286e-01  8.814381e-01
AccuracyPValue 0.000000e+00  1.000000e+00
McnemarPValue  2.774044e-22 3.101134e-152
> cbind(ann_train=ann_train_perf$byClass, ann_test=ann_test_perf$byClass)
                     ann_train   ann_test
Sensitivity          0.8467933 0.78041825
Specificity          0.8462589 0.84145250
Pos Pred Value       0.8051039 0.39835032
Neg Pred Value       0.8804522 0.96608925
Precision            0.8051039 0.39835032
Recall               0.8467933 0.78041825
F1                   0.8254226 0.52746547
Prevalence           0.4285714 0.11856193
Detection Rate       0.3629114 0.09252789
Detection Prevalence 0.4507635 0.23227770
Balanced Accuracy    0.8465261 0.81093538
```

K-NEAREST NEIGHBOR (KNN)

The final supervised method we tried was KNN.

The KNN algorithm is one of the simplest classification algorithms and it is one of the most used learning algorithms. KNN is a non-parametric, lazy learning algorithm. Its purpose is to use a database in which the data points are separated into several classes to predict the classification of a new sample point.

K-nearest neighbor classification for the testing dataset takes from the training dataset. For each row of the testing set, the k-nearest (in Euclidean distance) training set vectors are found, and the classification is decided by majority vote, with ties broken at random. If there are ties for the Kth nearest vector, all candidates are included in the vote.

The following libraries were used: class, caret.

To choose K we understand that the value of **k** indicates the number of training samples that are needed to classify the test sample. The value of **k** is non-parametric and a general rule of thumb in choosing the value of **k** is **k** = sqrt(N)/2, where N stands for the number of samples in your training dataset.

From the training set, the best guess is made for finding the starting value for K. In our case we got k=172 but in order to avoid ties we went ahead and opted for an odd number closest to the number obtained, k=173.

With kNN, no model is built, so we cannot assess performance on the training set. For this reason, we went ahead to verify the performance of the testing set.

Though the accuracy was very good, **F1 score**, defined as the weighted harmonic mean of the test's precision and recall, should also hold a good number. But in our case, it is only 0.41 (very small).

In order to improve the results, we proceeded to use Hyperparameter tuning. In using this method, we tuned the number of nearest neighbors. Hence, we chose a K value that was associated with the smallest cross validated error and high accuracy.

Hyperparameters are important as they control the overall behavior of a machine learning model. The goal of hyperparameter tuning is to find an optimal combination of hyperparameters that minimizes a predefined loss function to give better results.

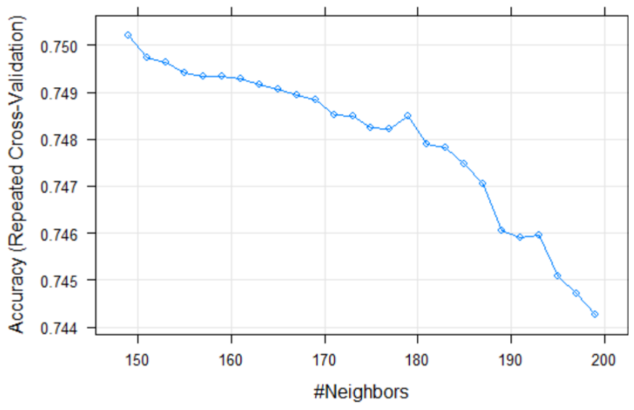
We used the caret package to perform the following:

1. Grid search using accuracy as the performance metric
2. Grid search using the AUC as the performance metric.

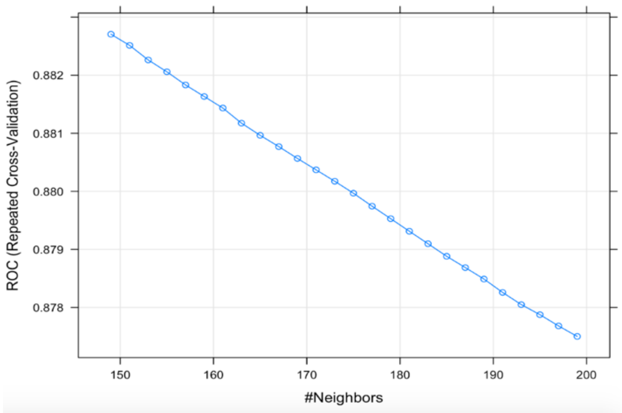
We then performed k-Fold Cross-Validation.

Cross-validation is when the dataset is randomly split up into 'k' groups. One of the groups is used as the test set and the rest are used as the training set. The model is trained on the training set and scored on the test set. Hence, 10-fold cross-validation technique is performed on the kNN model.

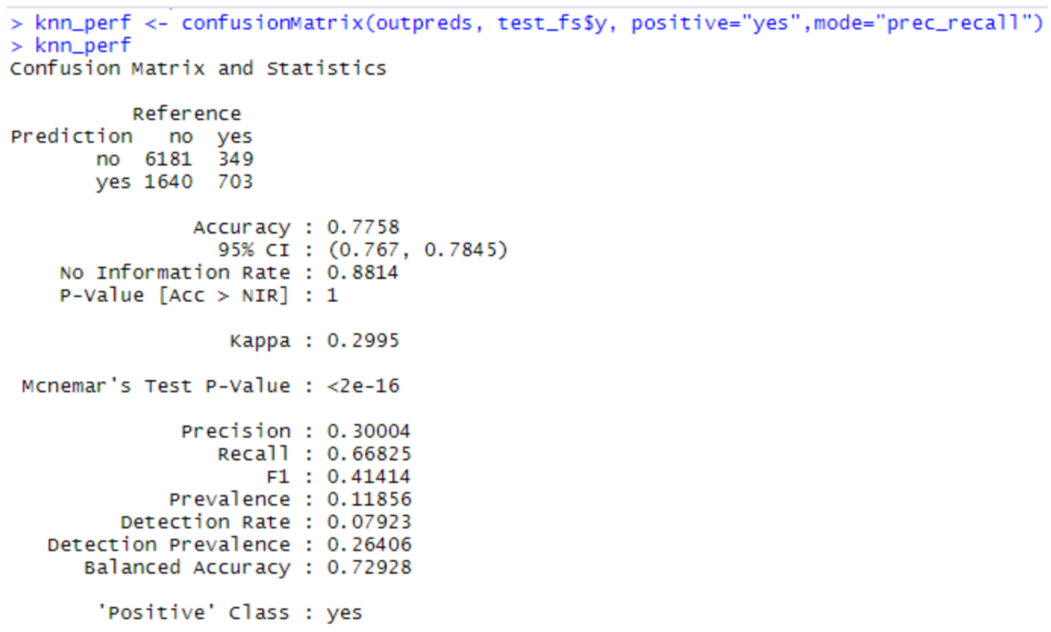
We created the kNN fit object as our cross-validated model, with which we are tuning the k hyperparameter using a grid search, using both Accuracy and ROC as performance metrics. The accuracy was used to select the optimal model using the largest value. The final value used for the model was k=149.



We understand that it is advisable to look to other metrics other than accuracy like ROC to assess the model, and as true positive and false positive rate play an important role, we will use AUC to choose the k-value.



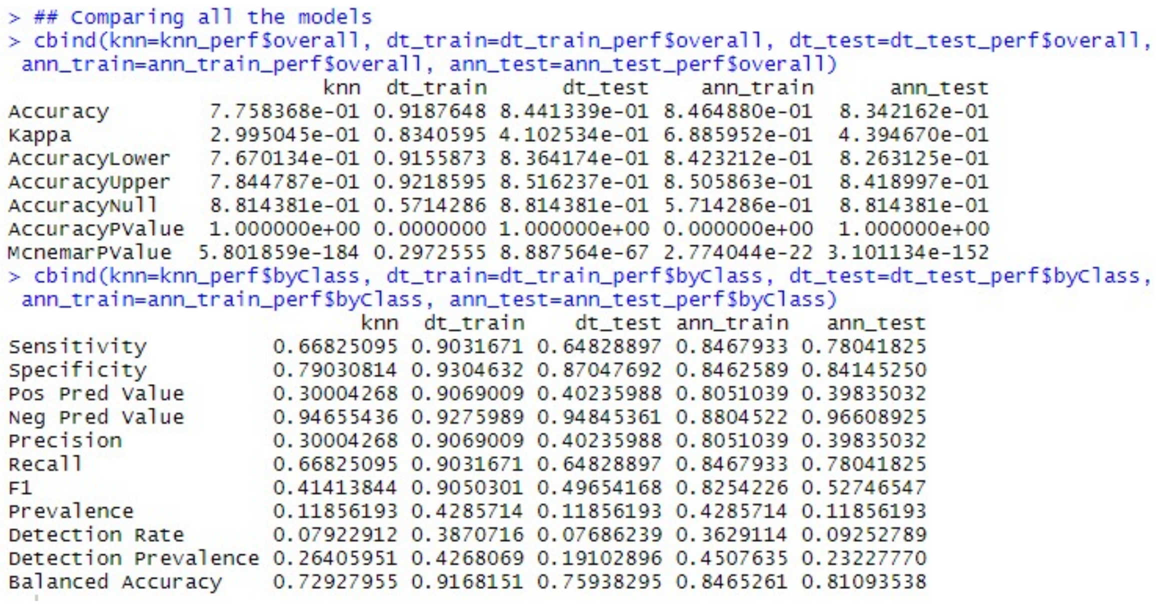
For outsample prediction using the tuned model, we found the performance information of our testing dataset. The performance information of our testing dataset for the best kNN model is obtained based on Accuracy and ROC for which the confusionMatrix function is used. This function takes the value obtained from the predict function and tests it on the testing dataset.



```
> knn_perf <- confusionMatrix(outpreds, test_fs$y, positive="yes",mode="prec_recall")
> knn_perf
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  6181  349
       yes 1640  703

               Accuracy : 0.7758
                 95% CI : (0.767, 0.7845)
    No Information Rate : 0.8814
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2995

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

              Precision : 0.30004
                 Recall : 0.66825
                     F1 : 0.41414
             Prevalence : 0.11856
         Detection Rate : 0.07923
   Detection Prevalence : 0.26406
      Balanced Accuracy : 0.72928

       'Positive' Class : yes
```

COMPARISON OF ALL MODELS & CONCLUSION

We compared all of the models below:



```
> ## Comparing all the models
> cbind(knn=knn_perf$overall, dt_train=dt_train_perf$overall, dt_test=dt_test_perf$overall,
 ann_train=ann_train_perf$overall, ann_test=ann_test_perf$overall)
                        knn  dt_train      dt_test    ann_train      ann_test
Accuracy       7.758368e-01 0.9187648 8.441339e-01 8.464880e-01  8.342162e-01
Kappa          2.995045e-01 0.8340595 4.102534e-01 6.885952e-01  4.394670e-01
AccuracyLower  7.670134e-01 0.9155873 8.364174e-01 8.423212e-01  8.263125e-01
AccuracyUpper  7.844787e-01 0.9218595 8.516237e-01 8.505863e-01  8.418997e-01
AccuracyNull   8.814381e-01 0.5714286 8.814381e-01 5.714286e-01  8.814381e-01
AccuracyPValue 1.000000e+00 0.0000000 1.000000e+00 0.000000e+00  1.000000e+00
McnemarPValue 5.801859e-184 0.2972555 8.887564e-67 2.774044e-22 3.101134e-152
> cbind(knn=knn_perf$byClass, dt_train=dt_train_perf$byClass, dt_test=dt_test_perf$byClass,
 ann_train=ann_train_perf$byClass, ann_test=ann_test_perf$byClass)
                            knn  dt_train    dt_test ann_train   ann_test
Sensitivity          0.66825095 0.9031671 0.64828897 0.8467933 0.78041825
Specificity          0.79030814 0.9304632 0.87047692 0.8462589 0.84145250
Pos Pred Value       0.30004268 0.9069009 0.40235988 0.8051039 0.39835032
Neg Pred Value       0.94655436 0.9275989 0.94845361 0.8804522 0.96608925
Precision            0.30004268 0.9069009 0.40235988 0.8051039 0.39835032
Recall               0.66825095 0.9031671 0.64828897 0.8467933 0.78041825
F1                   0.41413844 0.9050301 0.49654168 0.8254226 0.52746547
Prevalence           0.11856193 0.4285714 0.11856193 0.4285714 0.11856193
Detection Rate       0.07922912 0.3870716 0.07686239 0.3629114 0.09252789
Detection Prevalence 0.26405951 0.4268069 0.19102896 0.4507635 0.23227770
Balanced Accuracy    0.72927955 0.9168151 0.75938295 0.8465261 0.81093538
```

We found that of all the models created, the ANN model is preferred because its accuracy is high for both train and test. Its F1 measure is high for the training set but for the test set its consistent. Recall value is high for both train and test. Precision is high for the training set and low for the test. Here, our main focus is on recall than precision because we need to maximize the minority class 'yes', for the bank clients subscribing for the term deposit. We can afford to wrongly predict a few 'no' class as 'yes' because we can manage to make a few more calls but we cannot lose potential customers/clients who are more likely to subscribe to a term deposit.

ANN LIMITATIONS

While we identified ANN as the preferred model, there are some limitations to ANN. ANN is far more complicated than many other models, such as decision tree and regression. It's hard to interpret and even harder to visualize. It is also sensitive to noise, so it is easier to overfit a model as well cause more bias. Computational complexity is also very high for ANN.

Works Cited

BobboBobbo. "How to Replace Outliers with the 5th and 95th Percentile Values in R." *Stack Overflow*, 12 Nov. 2012, stackoverflow.com/questions/13339685/how-to-replace-outliers-with-the-5th-and-95th-percentile-values-in-r.

Dhana, Klodian. "Handling Missing Data with MICE Package; a Simple Approach." *DataScience+*, 30 Oct. 2017, datascienceplus.com/handling-missing-data-with-mice-package-a-simple-approach/.

Moro, S., et al. "A Data-Driven Approach to Predict the Success of Bank Telemarketing." *UCI Machine Learning Repository: Bank Marketing Data Set*, June 2014, archive.ics.uci.edu/ml/datasets/Bank+Marketing.

Sharma, Natasha. "Ways to Detect and Remove the Outliers." *Medium*, Towards Data Science, 23 May 2018, towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba.