BANKING: PREDICTING CUSTOMER SUBSCRIPTION TO TERM DEPOSITS

AGENDA LAYOUT

1. The Goal of the Project
2. Data Overview
3. Methodology
4. Compare Results & Conclusions
5. Improvements and Future Projects

BUSINESS GOAL

Problem

Banks need to avoid spending too much on i ndividuals that will not subscribe to a term deposit.

Business Goal

To predict whether if a customer subscribes to a term deposit or not by using previous marketing campaign data.

Stakeholders

Customers, Marketing Team, Service Employees

Opportunities

Gain new customers and lower marketing cost by having more efficient marketing results

Challenge

Lose potential customers if the model predicts incorrectly

TERM DEPOSIT

What is it?

- Time deposit or fixed deposit or CD
- Usually with a financial institution
- Specific maturity date (commonly referred to as its "term")
- Higher interest rate
- Cannot be withdrawn anytime (penalty)

Why does it matter to the bank?

- Knowing that term deposits allow banks to hold onto a deposit for a specific amount of time, so banks can invest in higher gain financial products to make a profit.
- In addition, banks also hold better chance to persuade term deposit clients into buying other products such as funds or insurance to further increase their revenues. The bank would like to identify existing clients that have higher chance to subscribe for a term deposit and focus marketing efforts on such clients.

DATA OVERVIEW

Source:

Direct telemarketing campaign data from a financial institution

\# of rows: 45,212
\# of Attributes: 17 with one decision attribute

Input variables:

ID, Age, Job, Marital Status, Education, Default, Balance, Housing, Loan, Contact, Day, Month, Duration, Campaign, pdays, previous, poutcome.

Output variable: y (subscribed/Not subscribed)

Data Cleaning:

Eliminate null values, ignore data points that do not make sense(age>150 and age <0)

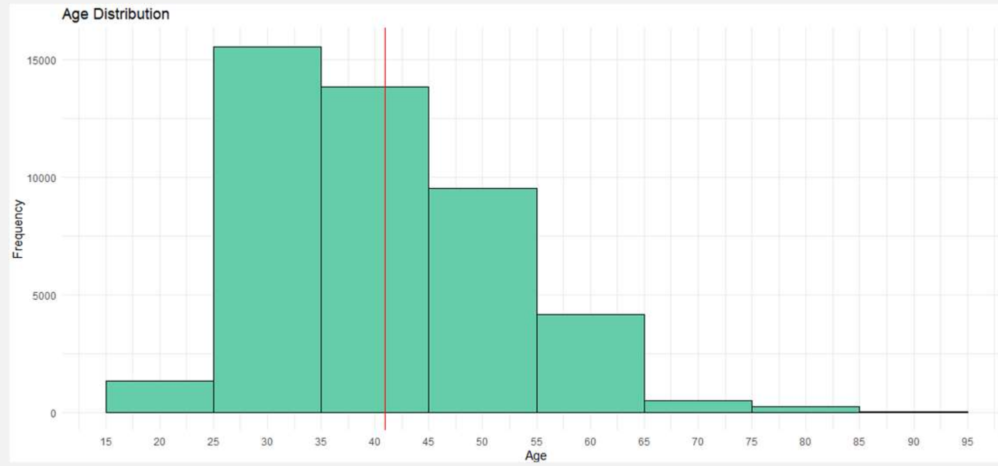
Normalize data and change data type into format usable by analysis tool

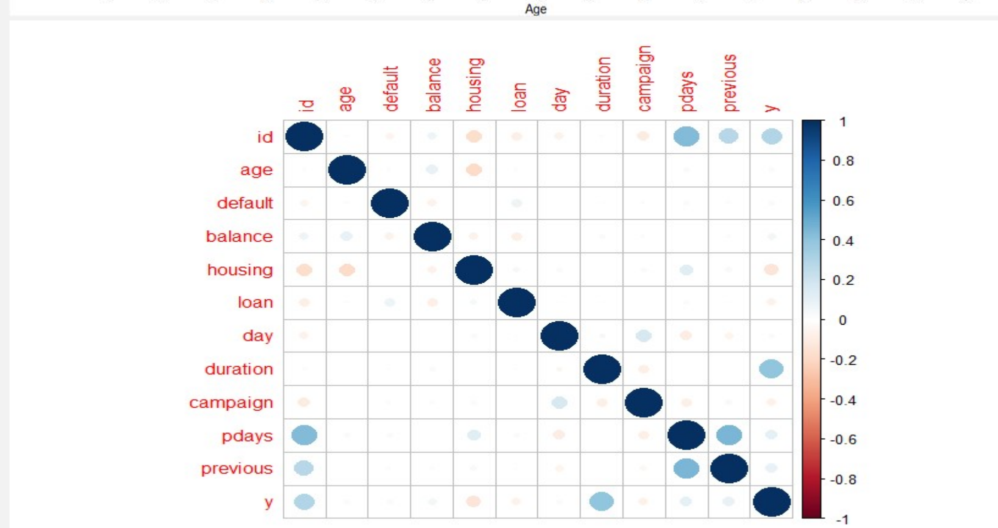
EXPLORATORY DATA ANALYSIS

Mean age is 41 (min is 18 and max is 95)

Mean balance is $1362 and standard deviation is high, So customer has a varying level of account balance.

The highest correlation coefficient is 0.39, which is between y (subscribed/not subscribed) and duration.
There is no multicollinearity as none of the dependent variables are highly correlated with each other

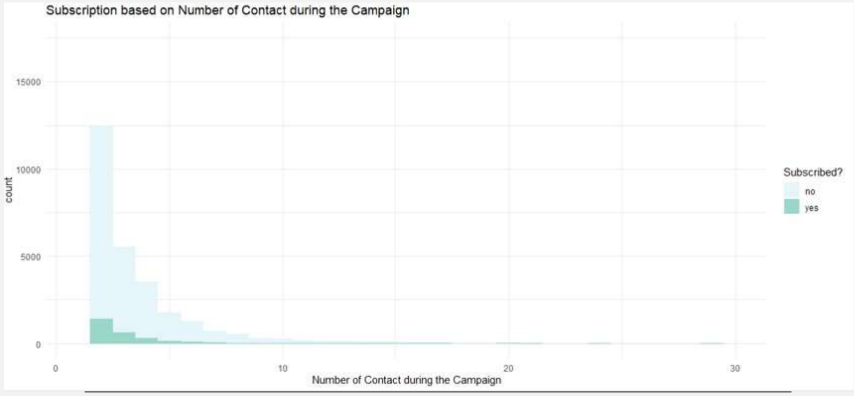




DATA VISUALIZATION

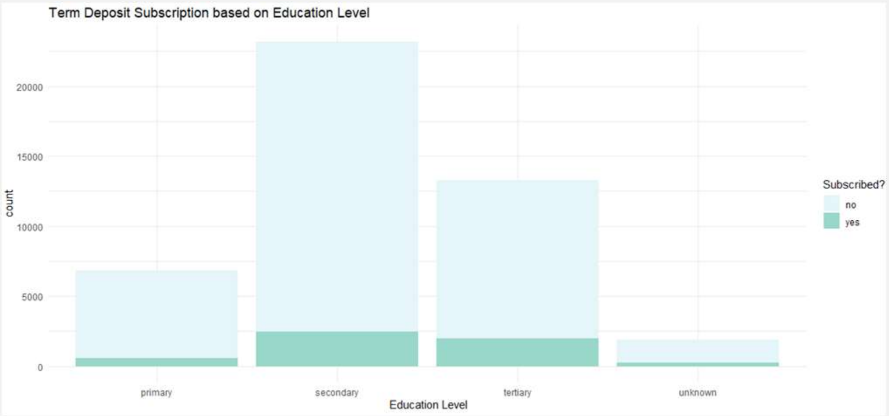
Subscription based on Number of Contact during Campaign

People that were going to subscribe did not receive many campaigns



Term Deposit Subscription based on Educational Level

People with higher education were more likely to subscribe to a term deposit



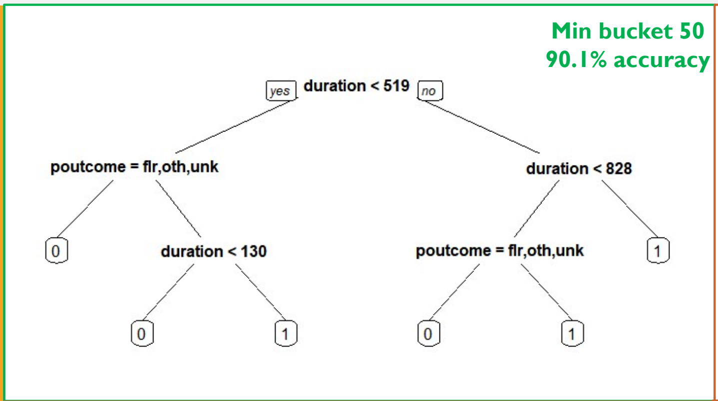
CART ANALYSIS

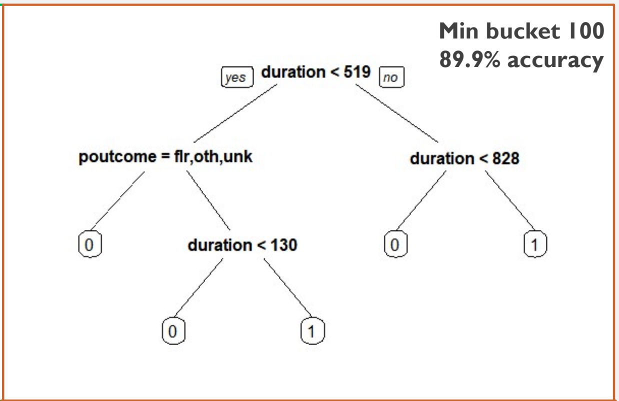
Different Trees

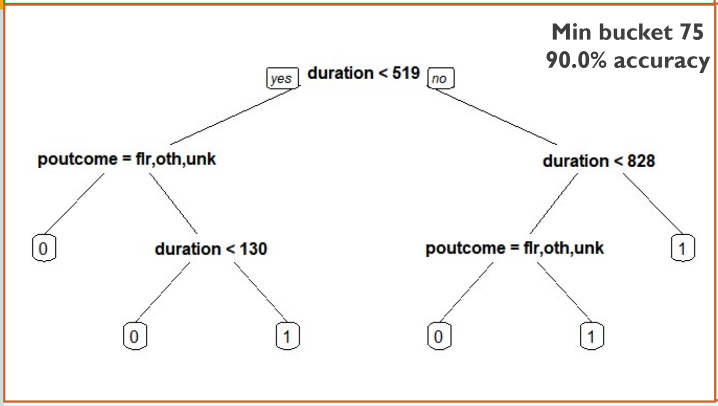
Changing the minimum size bucket alters the analysis, so we tried a few different groupings

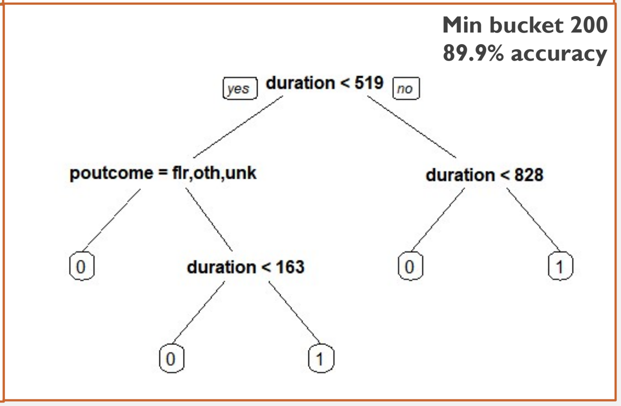
Min Bucket=50

Ultimately, we agreed that a minimum grouping of 50 people was the most accurate

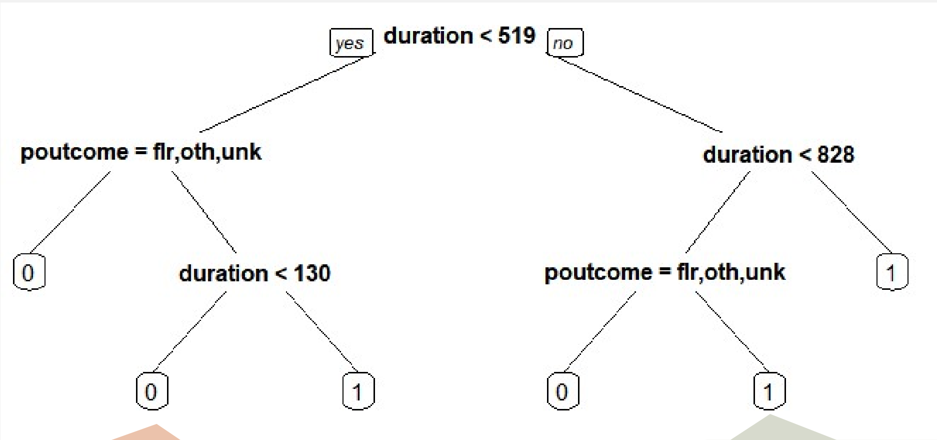








CART ANALYSIS

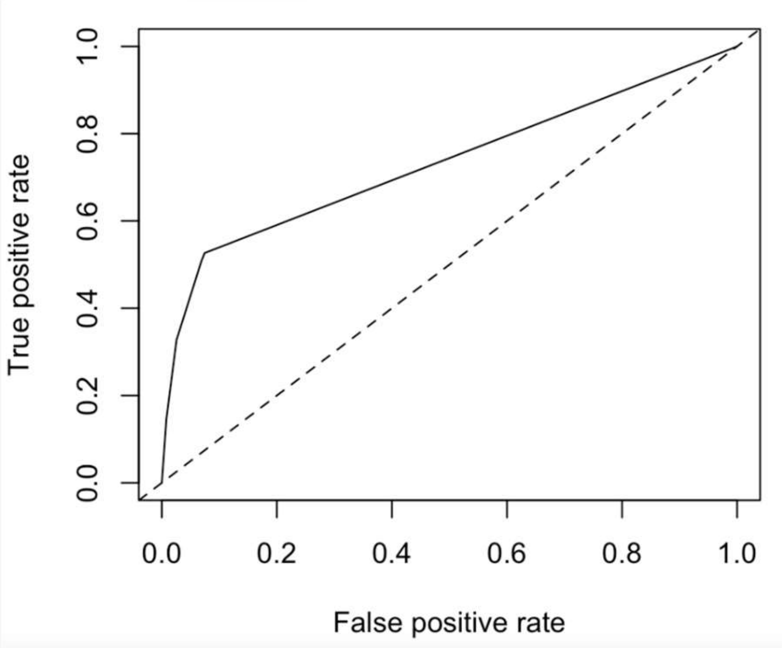


For customers, whose last contact was under 519 seconds, the outcome of the marketing campaign was not a success, and lastly, the last contact duration was under 130 seconds, a term deposit is unlikely (<50%)

For customers, whose last contact was over 519 seconds (and over 828 seconds) and the outcome of the marketing campaign was a success, a term deposit is likely (>50%)

Accuracy: 0.90

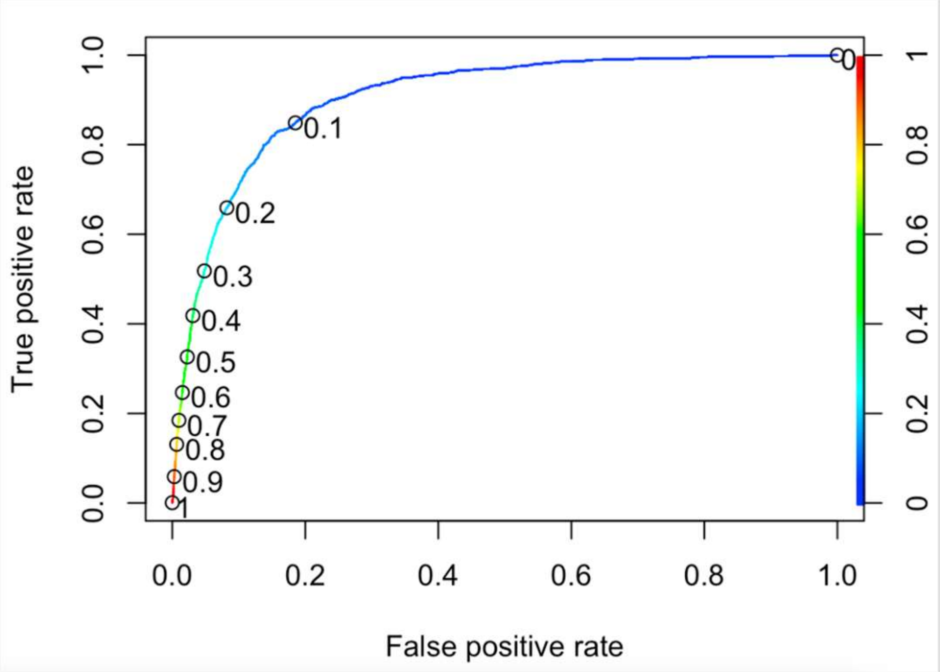
Sensitivity: 0.37



	Pred_0	Pred_1
Actual_0	11627	333
Actual_1	1012	587

LOGISTIC REGRESSION ANALYSIS

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-5.6225  -0.4574  -0.2937  -0.1613   3.6717

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -2.302e+00  9.236e-02 -24.922  < 2e-16 ***
age              -1.208e-03  1.736e-03  -0.695    0.487
default          -1.239e-01  1.781e-01  -0.695    0.487
balance           2.451e-05  5.065e-06   4.839 1.31e-06 ***
housing          -1.006e+00  4.274e-02 -23.549  < 2e-16 ***
loan             -7.454e-01  6.594e-02 -11.304  < 2e-16 ***
contacttelephone -6.111e-02  7.735e-02  -0.790    0.430
contactunknown   -1.337e+00  6.575e-02 -20.337  < 2e-16 ***
day              -3.302e-03  2.371e-03  -1.393    0.164
duration          3.917e-03  6.983e-05  56.087  < 2e-16 ***
campaign         -1.420e-01  1.170e-02 -12.132  < 2e-16 ***
pdays             2.173e-03  1.926e-04  11.279  < 2e-16 ***
previous          6.211e-02  8.741e-03   7.106 1.20e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



Accuracy: 0.89

Sensitivity: 0.36

	Pred_0	Pred_1
Actual_0	9719	259
Actual_1	959	363

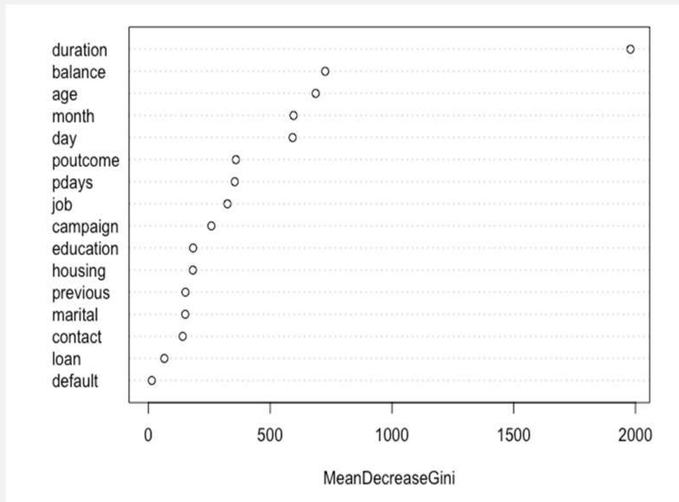
RANDOM FOREST ANALYSIS

Model Output

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 9731  798
         1  247  524

               Accuracy : 0.9075
                 95% CI : (0.902, 0.9128)
    No Information Rate : 0.883
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4536

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

            Sensitivity : 0.39637
            Specificity : 0.97525
         Pos Pred Value : 0.67964
         Neg Pred Value : 0.92421
             Prevalence : 0.11699
         Detection Rate : 0.04637
   Detection Prevalence : 0.06823
      Balanced Accuracy : 0.68581

       'Positive' Class : 1
```

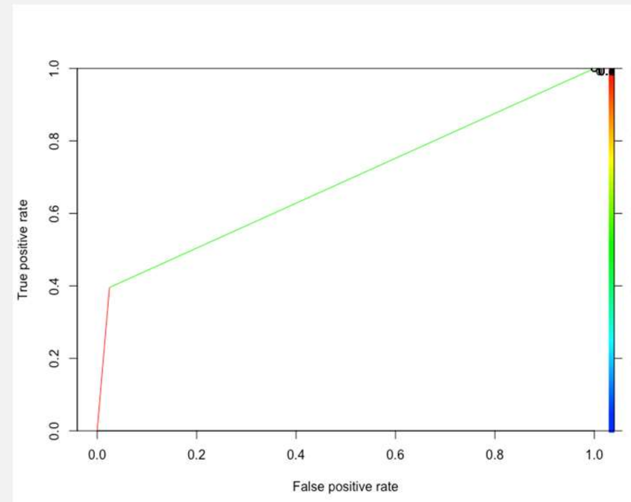
Accuracy: 0.9075

Variable Importance



Sensitivity: 0.40

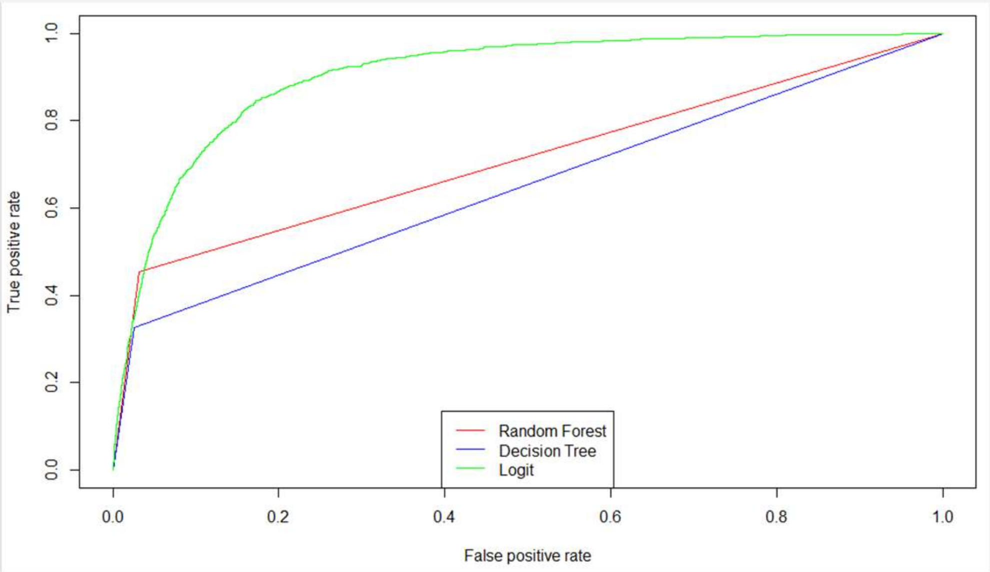
ROC Curve



	Pred_0	Pred_1
Actual_0	9731	247
Actual_1	798	524

COMPARISON RESULTS

Model	Sensitivity	AUC	Accuracy	Specificity
Random Forest	0.40	0.69	0.91	0.98
Logistic Regression	0.36	0.87	0.89	0.97
CART	0.37	0.65	0.90	0.97



Since our purpose is to generate a model for bank marketing purposes, a false-negative is way more harmful to the marketing strategy than a false positive. Since the goal of marketing is to get as many positives as possible, we are going to use sensitivity as the metric to compare our models' performance.

Based on sensitivity and accuracy, the Random Forrest model proves the most useful in identifying customers interested in getting a term deposit

POTENTIAL IMPROVEMENTS

Data Imbalance: (88.3% do not subscribe, 11.7% subscribe). An imbalanced dataset may lead to inaccurate predictions
Fix: Random Sampling Technique by sampling data from minority class and duplicate it to create more samples

Problem: What can we do to have a more efficient/flexible model?
Fix: Improve our models by using gradient boosting framework
-> XGBoost , SVM

Problem: Too many variables that could cost a lot to analyze the dataset.
Fix: Use Feature Selection & Grid Search to reduce the number of input variables (ID, day, month)

FUTURE PROJECTS

Seasonality

What is the best season (month/day) to contact the potential client?

Scoring Prospects

Use scoring/rating to measure a prospect's intent and interest. Their scores based on how they interact with the bank and change over time

Customer Segmentation

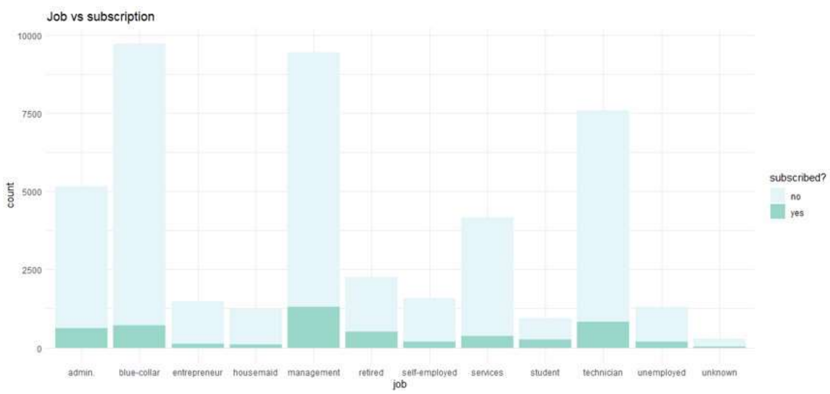
Segment customers into small groups to develop better marketing strategies for each group

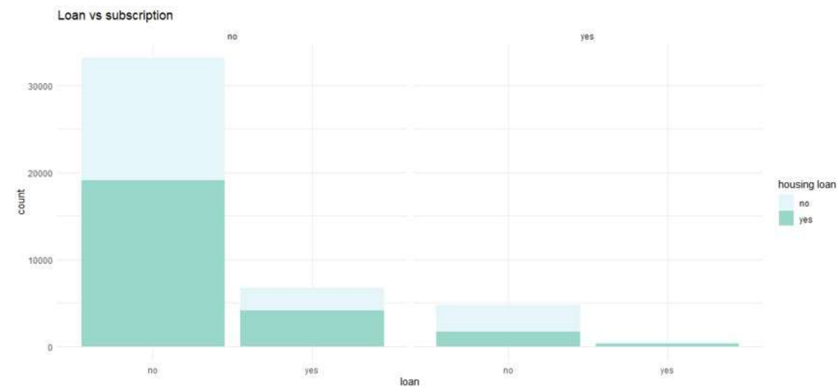
A/B Testing

Run two versions of marketing campaigns to collect data to see which one performs better

THANK YOU

APPENDIX


Job vs subscription
10000
7500
5000
2500
0
count
subscribed?
no
yes
admin.
blue-collar
entrepreneur
housemaid
management
retired
self-employed
services
student
technician
unemployed
unknown
job


Loan vs subscription
no
yes
30000
20000
10000
0
count
housing loan
no
yes
no
yes
no
yes
loan

CART CODE

```
#Loading Libraries
library(caTools)
library(rpart)
library(rpart.plot)

#Data Cleaning (removing weird ages and NAs)
FinalData <- FinalData[-c(which(FinalData$age <=
0 | FinalData$age>150)),]
FinalData <- na.omit(FinalData)
FinalData$y[which( FinalData$y == "yes")] <- 1
FinalData$y[which( FinalData$y == "no")] <- 0
FinalData$y <- as.numeric(FinalData$y)

#Training and Testing Dataset
set.seed(88)
split <- sample.split(FinalData$Id, SplitRatio = 0.7)
Train <- subset(FinalData, split==TRUE)
Test <- subset(FinalData, split==FALSE)

#Set baseline
nrow(Train)
sum(Train$y)
sum(Train$y)/nrow(Train)

#CART Analysis
Tree_min50= rpart(y~ age + job + marital + edu
cation + default + balance + housing + loan + con
tact + day + month + duration + campaign + pda
ys + previous + poutcome, method = "class", dat
a=Train, minbucket=50)
prp(Tree_min50)
Tree_min50Predict = predict(Tree_min50, newd
ata = Test, type="class")
tbl_min50 = table(Test$y, Tree_min50Predict)
tbl_min50
sum(diag(tbl_min50))/sum(tbl_min50)

Tree_min100= rpart(y~ age + job + marital + ed
ucation + default + balance + housing + loan + co
ntact + day + month + duration + campaign + pd
ays + previous + poutcome, method = "class", da
ta=Train, minbucket=100)
prp(Tree_min100)
Tree_min100Predict = predict(Tree_min100, ne
wdata = Test, type="class")
tbl_min100 = table(Test$y, Tree_min100Predict)
tbl_min100
sum(diag(tbl_min100))/sum(tbl_min100)

Tree_min200= rpart(y~ age + job + marital + ed
ucation + default + balance + housing + loan + co
ntact + day + month + duration + campaign + pd
ays + previous + poutcome, method = "class", da
ta=Train, minbucket=200)
prp(Tree_min200)
Tree_min200Predict = predict(Tree_min200, ne
wdata = Test, type="class")
tbl_min200 = table(Test$y, Tree_min200Predict)
tbl_min200
sum(diag(tbl_min200))/sum(tbl_min200)

Tree_min75= rpart(y~ age + job + marital + edu
cation + default + balance + housing + loan + con
tact + day + month + duration + campaign + pda
ys + previous + poutcome, method = "class", dat
a=Train, minbucket=75)
prp(Tree_min75)
Tree_min75Predict = predict(Tree_min75, newd
ata = Test, type="class")
tbl_min75 = table(Test$y, Tree_min75Predict)
tbl_min75
sum(diag(tbl_min75))/sum(tbl_min75)
```

LOGIT CODE

```
data = read.csv("Assignment-2_Data.csv")
summary(data)
str(data)

# remove ages that do not make sense
data <- data[-c(which(data$age <= 0 | data$age>
150)),]

# remove na values
bank <- na.omit(data)
bank$y[bank$y == "no"] = 0
bank$y[bank$y == "yes"] = 1
bank$housing[bank$housing == "no"] = 0
bank$housing[bank$housing == "yes"] = 1
bank$loan[bank$loan == "no"] = 0
bank$loan[bank$loan == "yes"] = 1
bank$default[bank$default == "no"] = 0
bank$default[bank$default == "yes"] = 1

bank$housing <- as.integer(bank$housing)
bank$default <- as.integer(bank$default)
bank$loan <- as.integer(bank$loan)
bank$y <- as.integer(bank$y)
```

```
# Creating Training and Testing Sets
library(caTools)
# Normalization
library(caret)
str(bank)
#mean and sd of each variable
preproc = preProcess(bank)
#normalize the data
BankNorm = predict(preproc, bank)
# Creating Training and Testing Sets
set.seed(88)
split = sample.split(BankNorm$y, SplitRatio = 0.7
5)
Train = subset(bank, split==TRUE)
Test = subset(bank, split==FALSE)

# Building a Logistic Regression Model
str(Train)
bankLog = glm(y ~ age+default+balance+housing
+loan+contact+day+duration+campaign+pdays+p
revious, data = Train, family=binomial)
summary(bankLog)
```

```
# Evaluating the Model

predicted_values<-ifelse(predict(bankLog,type="r
esponse", newdata = Test)>0.4,1,0)
actual_values<-Test$y
conf_matrix<-table(predicted_values,actual_valu
es)
conf_matrix
specificity(conf_matrix)
sensitivity(conf_matrix)
library(ROCR)
PredictTest = predict(bankLog, type="response",
newdata = Test)
summary(PredictTest)
tbl = table(Test$y, PredictTest > 0.4)
sum(diag(tbl))/sum(tbl)
ROCRpred = prediction(PredictTest, Test$y)
ROCCurve = performance(ROCRpred, "tpr", "fp
r")
plot(ROCCurve)
plot(ROCCurve, colorize=TRUE, print.cutoffs.at
=seq(0,1,0.1), text.adj=c(-0.2,0.7))
as.numeric(performance(ROCRpred, "auc")@y.v
alues) # AUC value
```

ROC CODE

```
library(pacman)
library(janitor)
# Reading in the data
term <- clean_names(read.csv("Assignment-2_Data.csv"))
term <- term[-c(which(term$age <= 0 | term$age>150)),]
term <- na.omit(term)
str(term)
summary(term)
term$y <- ifelse(term$y=="yes", 1,0)
term$y <- as.integer(term$y)
term$housing <- ifelse(term$housing == "yes", 1,0)
term$housing <- as.integer(term$housing)
term$default <- ifelse(term$default=="yes", 1,0)
term$default <- as.integer(term$default)
term$loan <- ifelse(term$loan=="yes", 1,0)
term$loan <- as.integer(term$loan)
term[sapply(term, is.character)] <- lapply(term[sapply(term, is.character)],
                                    as.factor)
# calling the library need for splitting
library(caTools)
set.seed(88)
# splitting into training and testing
split = sample.split(term, SplitRatio = 0.75)
train = subset(term, split == TRUE)
test = subset(term, split == FALSE)
# Random Forest
library(randomForest)
train$y <- as.factor(train$y)
test$y <- as.factor(test$y)
TermForrest <- randomForest(y ~ age + job + marital + education + defa
ult + balance + housing +
                  loan + contact + day + duration + campaign + pdays + pr
evious
                + month + poutcome, data=train, ntree=200, nodesize=15)

TermPredict <- predict(TermForrest, newdata=test)
tbl <- table(test$y, TermPredict)
sum(diag(tbl))/sum(tbl)

# Building a CART model
library(rpart)
library(rpart.plot)
TermTree = rpart(y ~ age + job + marital + education + default
+ balance + housing +
                 loan + contact + day + duration + campaign + pday
s + previous
                + month + poutcome, method="class", data=train,
minbucket=100)
prp(TermTree)

# automatically assumes a threshold of .5
TermPredict2 = predict(TermTree, newdata=test, type="class")
tbl2 <- table(test$y,TermPredict)
sum(diag(tbl))/sum(tbl)


# Running a logistic Regression
bankLog = glm(y ~ age+marital+education+contact+month+pou
tcome+default+balance+housing+loan+day+duration+campaign
+pdays+previous , data = train, family=binomial)
summary(bankLog)

# Evaluating the Model
PredictTrain = predict(bankLog, type="response")
table <- table(train$y, PredictTrain > 0.5)
sum(diag(table))/sum(table)

# Testing model on new data
PredictTest = predict(bankLog, type="response", newdata=
test)
table2 <- table(test$y, PredictTest > 0.5)
sum(diag(table2))/sum(table2)
# ROC Curve
library(ROCR)
ROCRpredrf = as.numeric(TermPredict)
ROCRpred = prediction(ROCRpredrf, test$y)
ROCCurve = performance(ROCRpred, "tpr", "fpr")
as.numeric(performance(ROCRpred, "auc")@y.values)
# ROC Curve 2
ROCRpredrf2 = as.numeric(TermPredict2)
ROCRpred2 = prediction(ROCRpredrf2, test$y)
ROCCurve2 = performance(ROCRpred2, "tpr", "fpr")
as.numeric(performance(ROCRpred2, "auc")@y.values)
# ROC Curve 3
ROCRpred3 = prediction(PredictTest, test$y)
ROCCurve3 = performance(ROCRpred3, "tpr", "fpr")
as.numeric(performance(ROCRpred3, "auc")@y.values)
plot(ROCCurve, col = 'red')
plot(ROCCurve2, add = TRUE, col = 'blue')
plot(ROCCurve3, add = TRUE, col = 'green')
legend("bottom", c("Random Forest","Decision Tree","Logit
"), lty=1,
      col = c("red", "blue", "green"))
```

RANDOM FOREST CODE

```
data = read.csv("Assignment-2_Data.csv")
summary(data)
# remove ages that do not make sense
data <- data[-c(which(data$age <= 0 | data$age>150)),
]

# remove na values
bank <- na.omit(data)
bank$y[bank$y == "no"] = 0
bank$y[bank$y == "yes"] = 1
bank$housing[bank$housing == "no"] = 0
bank$housing[bank$housing == "yes"] = 1
bank$loan[bank$loan == "no"] = 0
bank$loan[bank$loan == "yes"] = 1
bank$default[bank$default == "no"] = 0
bank$default[bank$default == "yes"] = 1

bank$housing <- as.integer(bank$housing)
bank$default <- as.integer(bank$default)
bank$loan <- as.integer(bank$loan)
bank$y <- as.integer(bank$y)
bank[sapply(term, is.character)] <- lapply(bank[sa
pply(bank, is.character)],
                                   as.factor)
# Creating Training and Testing Sets
library(caTools)
```

```
# Normalization
library(caret)


#mean and sd of each variable
preproc = preProcess(bank)

#normalize the data
BankNorm = predict(preproc, bank)

# Creating Training and Testing Sets
set.seed(88)
split = sample.split(BankNorm$y, SplitRatio = 0.75)
Train = subset(bank, split==TRUE)
Test = subset(bank, split==FALSE)

#Random Forest Model
library(randomForest)

Train$y = as.factor(Train$y)
Test$y = as.factor(Test$y)
```

```
#Random Forest Model
rfmdel <-randomForest(y ~age+default+balance+housing+lo
an+contact+day+duration+campaign+pdays+previous+marit
al+education+poutcome+job+month,data = Train)
varImpPlot(rfmdel) #To Check Variable Importance
prediction_rf<-predict(rfmdel,Test)
prediction_rf

#Confusion matrix to validate it
conf_mat<-confusionMatrix(prediction_rf,Test$y)

#Table for Validation
rftable = table(Test$y,prediction_rf)
Accuracy = sum(diag(rftable))/sum(rftable)
Accuracy

#ROC Curve
library(ROCR)
ROCRpredrf = as.numeric(predict(rfmdel,Test,type="respon
se"))
ROCrf = prediction(ROCRpredrf,Test$y)
ROCCurverf = performance(ROCrf,measure = "tpr", x.mea
sure = "fpr")
plot(ROCCurverf)
plot(ROCCurverf, colorize=TRUE, print.cutoffs.at=seq(0,1,0
.1), text.adj=c(-0.2,0.7))
AUC_rf <- as.numeric(performance(ROCrf, "auc")@y.value
s) # AUC value
AUC_rf
```