Bank Marketing Data Analysis

Amy M Kalna

Bank Marketing Data Analysis & Modeling

1. Introduction

This project applies machine learning techniques that go beyond standard linear regression. I had the opportunity to use a publicly available dataset to solve the problem of my choice. I sifted through the datasets available on Kaggle and chose a finance/bank related dataset. I work at a bank so I was geared toward selecting a topic that's relevant to the banking business.

The goal of the project is to answer the following question: What kind of behaviors do potential customers exhibit that result in them more likely to subscribe to a term deposit?

The business problem is to devise a target marketing strategy for the bank based on the behavioral data collected. The dataset is included in one of the submission files and can be downloaded from Kaggle (https://www.kaggle.com/henriqueyamahata/bank-marketing).

The Dataset: It contains 41,188 customer data on direct marketing campaigns (phone calls) of a Portuguese banking institution.

It has the following variables:

Client: age, job, marital, education, default status, housing, and loan

Campaign: last contact type, last contact month of year, last contact day of the week, and last contact duration

Others: number of contacts performed in current campaign, number of days that passed by after the client was last contacted, number of contacts performed before this campaign, outcome of previous campaign, and whether a client has subscribed a term deposit

Key Steps Performed:

I first used Data Classification to examine the set related with direct marketing campaigns of a Portuguese banking institution. The objective of the classification is to predict if the client will subscribe to a Term Deposit. Data Classification is the use of machine learning techniques to organize datasets into related sub-populations, not previous specified in the dataset. This can uncover hidden characteristics within data, and identify hidden categories that new data belongs within. The rest of the key steps that were performed used the data science techniques of Exploratory Data Analysis, Data Classification basis Random Forest and K-Nearest Neighbor.

2. Data Analysis

2.1. Exploratory Analysis

Loading the required packages:

```
rm(list = ls())
options(warn=-1)

if(!require(readr)) install.packages("readr", repos = "")
if(!require(tidyverse)) install.packages("tidyverse", repos = "")
if(!require(GGally)) install.packages("GGally", repos = "")
if(!require(glmnet)) install.packages("glmnet", repos = "")
if(!require(Matrix)) install.packages("Matrix", repos = "")
if(!require(DataExplorer)) install.packages("DataExplorer", repos = "")
if(!require(corrplot)) install.packages("corrplot", repos = "")
if(!require(caret)) install.packages("caret", repos = "")
if(!require(randomForest)) install.packages("randomForest", repos = "")
if(!require(class)) install.packages("class", repos = "")
if(!require(gmodels)) install.packages("gmodels", repos = "")
if(!require(dplyr)) install.packages("dplyr", repos = "")
if(!require(psych)) install.packages("psych", repos = "")


library(readr)
library(tidyverse)
library(GGally)
library(glmnet)
library(Matrix)
library(ggplot2)
library(DataExplorer)
library(corrplot)
library(caret)
library(randomForest)
library(class)
library(gmodels)
library(dplyr)
library(psych)


set.seed(1)
```

Loading the dataset:

```
#setwd("C:/Users/1012233/Downloads/20191103 - R Studio Kaggle project")
#data.df <- read.csv("bank-additional-full.csv", header=TRUE, sep=";")
```

```r
data.df <- read.csv("https://raw.github.com/amymkalna/Kaggle-Bank-Marketing-Data/master/bank-additional-full.csv", header=TRUE, sep=";")
```

Viewing the column names of the dataset:

```r
names(data.df)

##  [1] "age"            "job"            "marital"        "education"
##  [5] "default"        "housing"        "loan"           "contact"
##  [9] "month"          "day_of_week"    "duration"       "campaign"
## [13] "pdays"          "previous"       "poutcome"       "emp.var.rate"
## [17] "cons.price.idx" "cons.conf.idx"  "euribor3m"      "nr.employed"
## [21] "y"
```

Column details of the dataset:

```r
str(data.df)

## 'data.frame':    41188 obs. of  21 variables:
##  $ age           : int  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 4 8 8 1 
8 8 1 2 10 8 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 2 2 2 2 2 
2 2 3 3 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4 2 4 
3 6 8 6 4 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2 1 2 1 
1 ...
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1 1 1 3 
3 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1 1 1 1 
1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2 
2 2 2 2 ...
##  $ month         : Factor w/ 10 levels "apr","aug","dec",..: 7 7 7 7 7 7 7 
7 7 7 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2 2 2 
2 2 2 ...
##  $ duration      : int  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2 
2 2 2 2 2 2 ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -3
6.4 -36.4 ...
```

```
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

Summary analysis of the dataset:

```
summary(data.df)

##       age                    job            marital     
##  Min.   :17.00   admin.     :10422   divorced: 4612  
##  1st Qu.:32.00   blue-collar: 9254   married :24928  
##  Median :38.00   technician : 6743   single  :11568  
##  Mean   :40.02   services   : 3969   unknown :   80  
##  3rd Qu.:47.00   management : 2924                   
##  Max.   :98.00   retired    : 1720                   
##                  (Other)    : 6156                   
##               education        default          housing     
##  university.degree  :12168   no     :32588   no     :18622  
##  high.school        : 9515   unknown: 8597   unknown:  990  
##  basic.9y           : 6045   yes    :    3   yes    :21576  
##  professional.course: 5243                                  
##  basic.4y           : 4176                                  
##  basic.6y           : 2292                                  
##  (Other)            : 1749                                  
##       loan            contact          month        day_of_week
##  no     :33950   cellular :26144   may    :13769   fri:7827   
##  unknown:  990   telephone:15044   jul    : 7174   mon:8514   
##  yes    : 6248                     aug    : 6178   thu:8623   
##                                    jun    : 5318   tue:8090   
##                                    nov    : 4101   wed:8134   
##                                    apr    : 2632              
##                                    (Other): 2016              
##     duration         campaign          pdays          previous    
##  Min.   :   0.0   Min.   : 1.000   Min.   :  0.0   Min.   :0.000  
##  1st Qu.: 102.0   1st Qu.: 1.000   1st Qu.:999.0   1st Qu.:0.000  
##  Median : 180.0   Median : 2.000   Median :999.0   Median :0.000  
##  Mean   : 258.3   Mean   : 2.568   Mean   :962.5   Mean   :0.173  
##  3rd Qu.: 319.0   3rd Qu.: 3.000   3rd Qu.:999.0   3rd Qu.:0.000  
##  Max.   :4918.0   Max.   :56.000   Max.   :999.0   Max.   :7.000  
##                                                                   
##         poutcome      emp.var.rate      cons.price.idx  cons.conf.idx  
##  failure    : 4252   Min.   :-3.40000   Min.   :92.20   Min.   :-50.8  
##  nonexistent:35563   1st Qu.:-1.80000   1st Qu.:93.08   1st Qu.:-42.7  
##  success    : 1373   Median : 1.10000   Median :93.75   Median :-41.8  
##                      Mean   : 0.08189   Mean   :93.58   Mean   :-40.5  
##                      3rd Qu.: 1.40000   3rd Qu.:93.99   3rd Qu.:-36.4  
##                      Max.   : 1.40000   Max.   :94.77   Max.   :-26.9  
##                                                                        
##    euribor3m      nr.employed     y        
##  Min.   :0.634   Min.   :4964   no :36548  
```

```
##   1st Qu.:1.344   1st Qu.:5099   yes: 4640  
##   Median :4.857   Median :5191              
##   Mean   :3.621   Mean   :5167              
##   3rd Qu.:4.961   3rd Qu.:5228              
##   Max.   :5.045   Max.   :5228              
##
```

2.2. Data Preparation

We check if there are any missing values that exists:

```
sum(is.na(data.df))

## [1] 0
```

There are no missing values in our dataset.

In the above exploratory analysis, we observed that there are many variables with class=int; hence, we convert them into numeric class

Converting quantitative values to numeric class:

```
data.df$age <- as.numeric(data.df$age)
data.df$duration <- as.numeric(data.df$duration)
data.df$campaign <- as.numeric(data.df$campaign)
data.df$pdays <- as.numeric(data.df$pdays)
data.df$previous <- as.numeric(data.df$previous)
```

Ordering the levels of month:

```
data.df$month<- factor(data.df$month, ordered = TRUE, levels = c("mar", "apr"
, "may", "jun", "jul", "aug", "sep", "oct", "nov", "dec"))
```

Since the target variable is a categorical variables with 2 possible values: yes, no; we transform it into a numerical denotation: 1,0 respectively.

Transforming the target variable as Yes=1 and No=0:

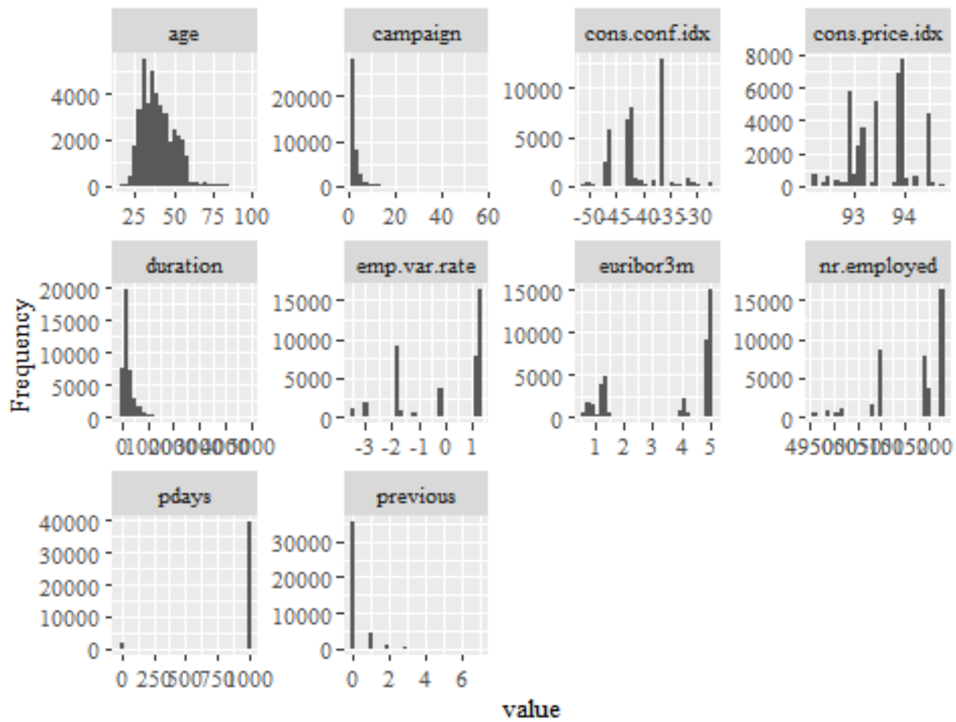
```
table(data.df$y)

## 
##    no   yes 
## 36548  4640

data.df <- data.df %>%
  mutate(y = ifelse(y=="yes", 1, 0))

data.df$y <- as.factor(data.df$y)
table(data.df$y)
```

```
##
##     0     1
## 36548  4640
```

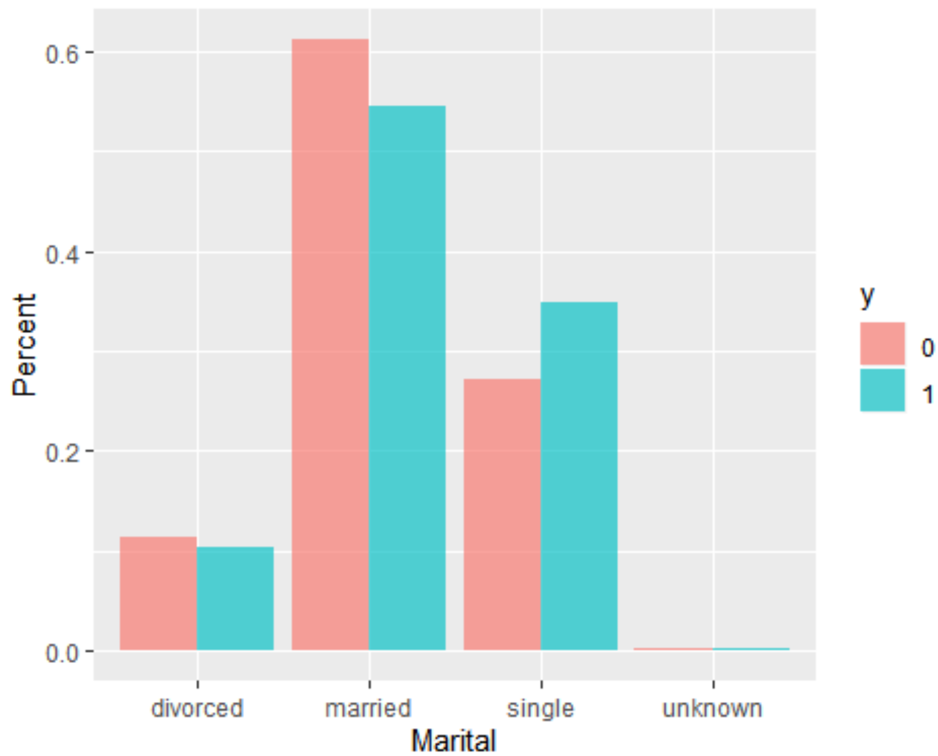
2.3. Descriptive Analysis

Let us look at the histogram of the input variables:

```
plot_histogram(data.df[,-21],ggtheme = theme_gray(base_size = 10, base_family
= "serif"))
```

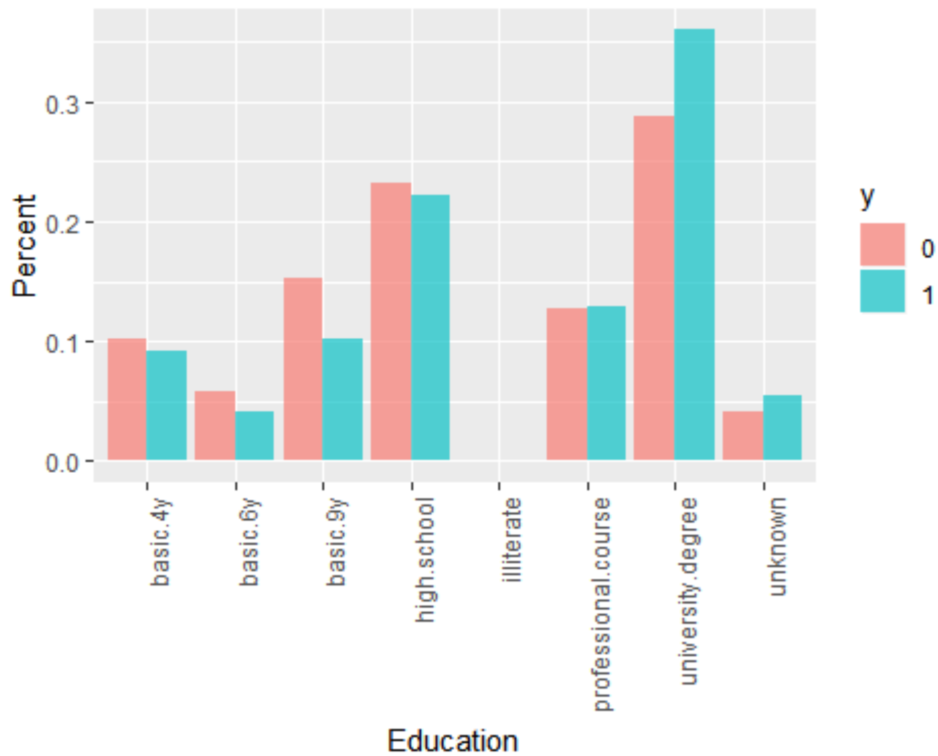



```
mytable <- table(data.df$marital, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("marital", "y", "perc")
ggplot(data = tab, aes(x = marital, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Marital")+ylab("Percent")
```



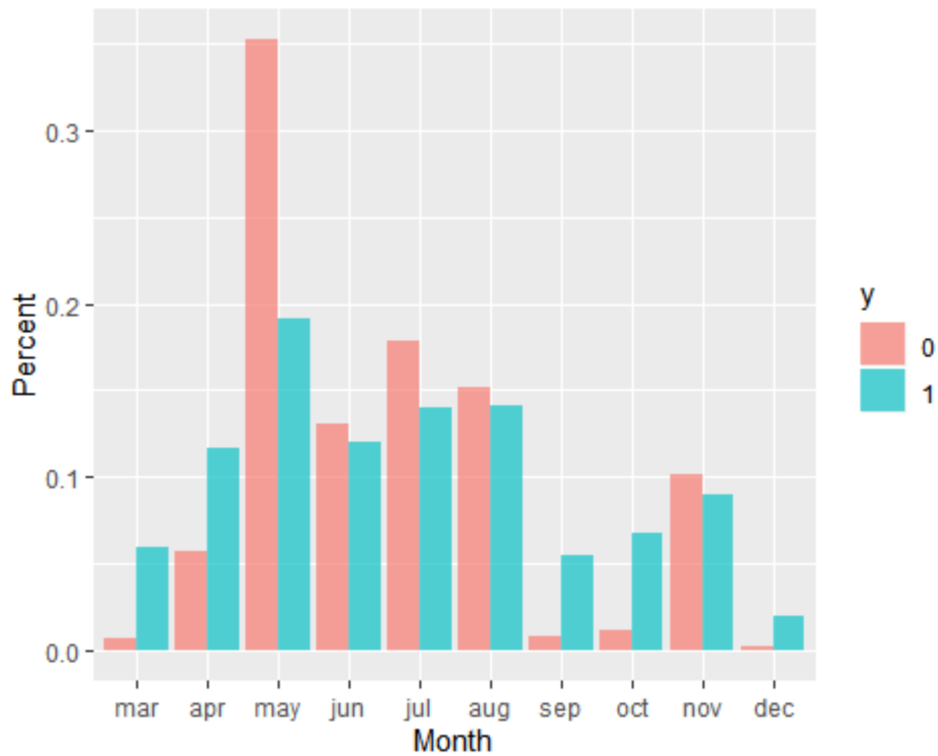
With respect to Marital Status there is not an observed large difference in the proportion of people subscribed to term deposits and people without term deposits.

```
mytable <- table(data.df$education, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("education", "y", "perc")
ggplot(data = tab, aes(x = education, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
theme(axis.text.x=element_text(angle=90,hjust=1)) +
  xlab("Education")+ylab("Percent")
```



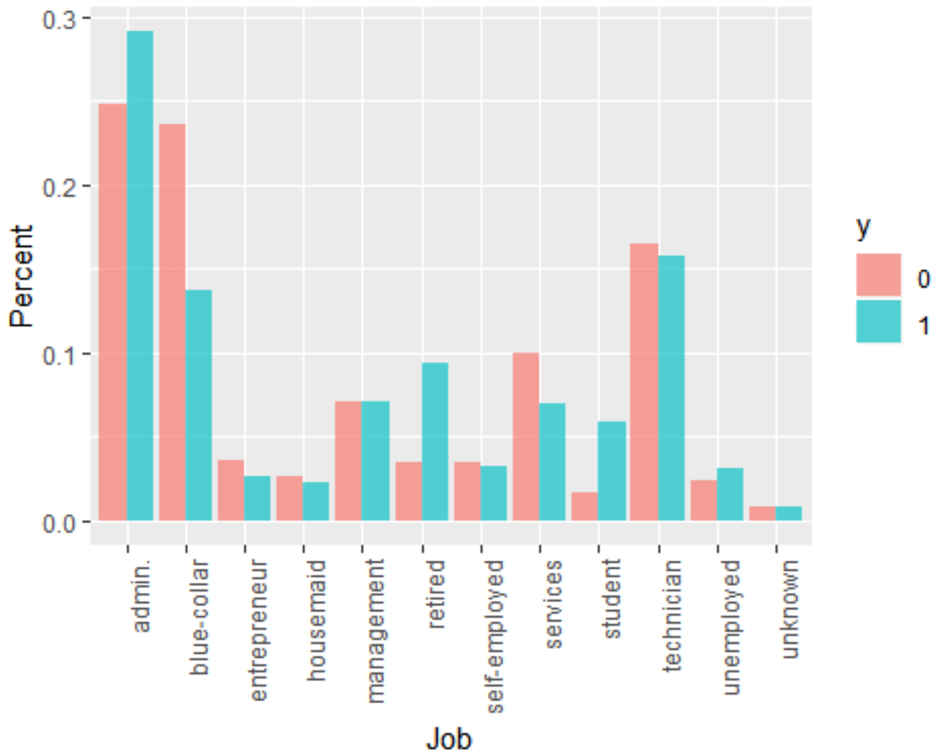
We can see that customers who sign up for bank deposits, proportionally, have achieved a higher level of education, than those who didn't sign up.

```
mytable <- table(data.df$month, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("month", "y", "perc")
ggplot(data = tab, aes(x = month, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Month")+ylab("Percent")
```



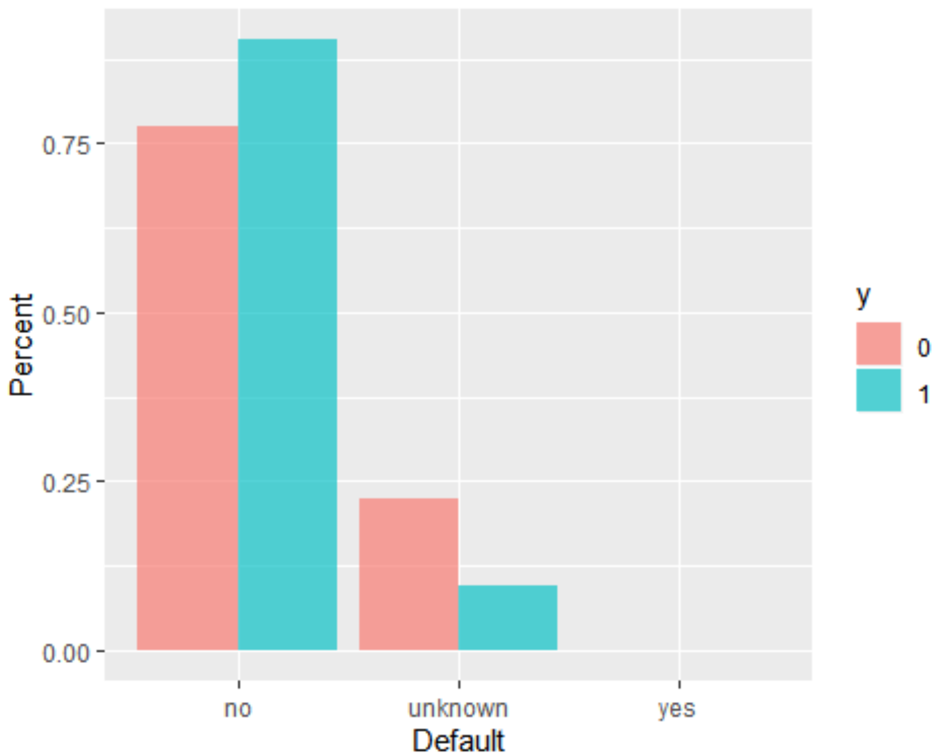
The month of May is when the highest number of calls were placed for marketing deposits. The months of April, September, October, and December is the time when a higher proportion of people subscribed for term deposits.

```
mytable <- table(data.df$job, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("job", "y", "perc")
ggplot(data = tab, aes(x = job, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
theme(axis.text.x=element_text(angle=90,hjust=1)) +
  xlab("Job")+ylab("Percent")
```



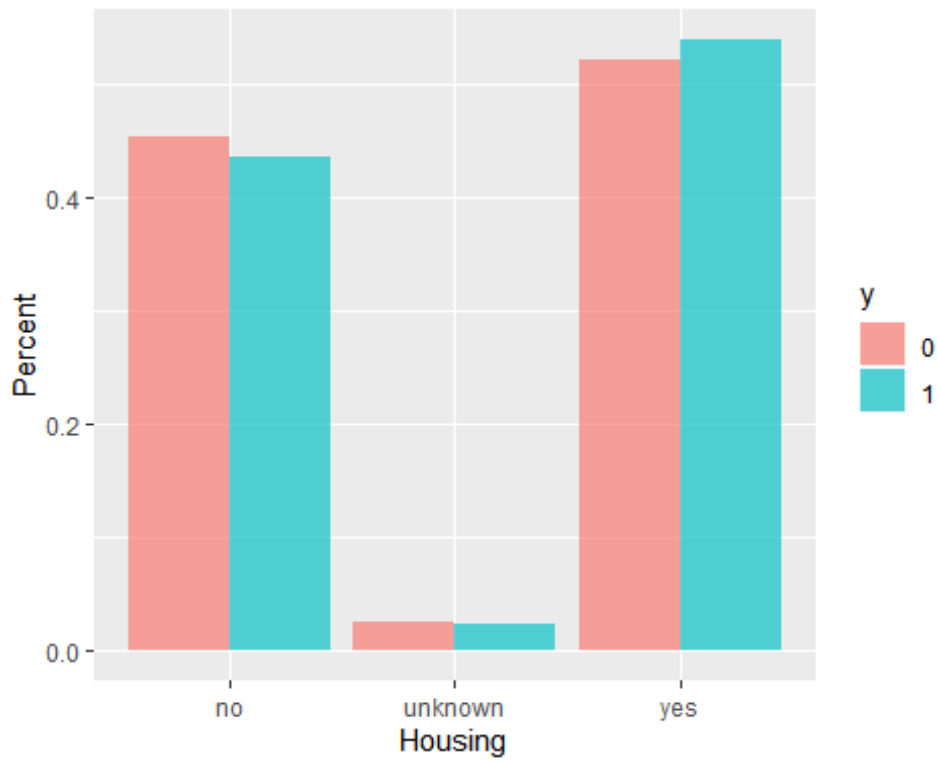
We see there are higher proportions for customers signing up for the term deposits who have the jobs of admin, retired, and students.

```
mytable <- table(data.df$default, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("default", "y", "perc")
ggplot(data = tab, aes(x = default, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Default")+ylab("Percent")
```



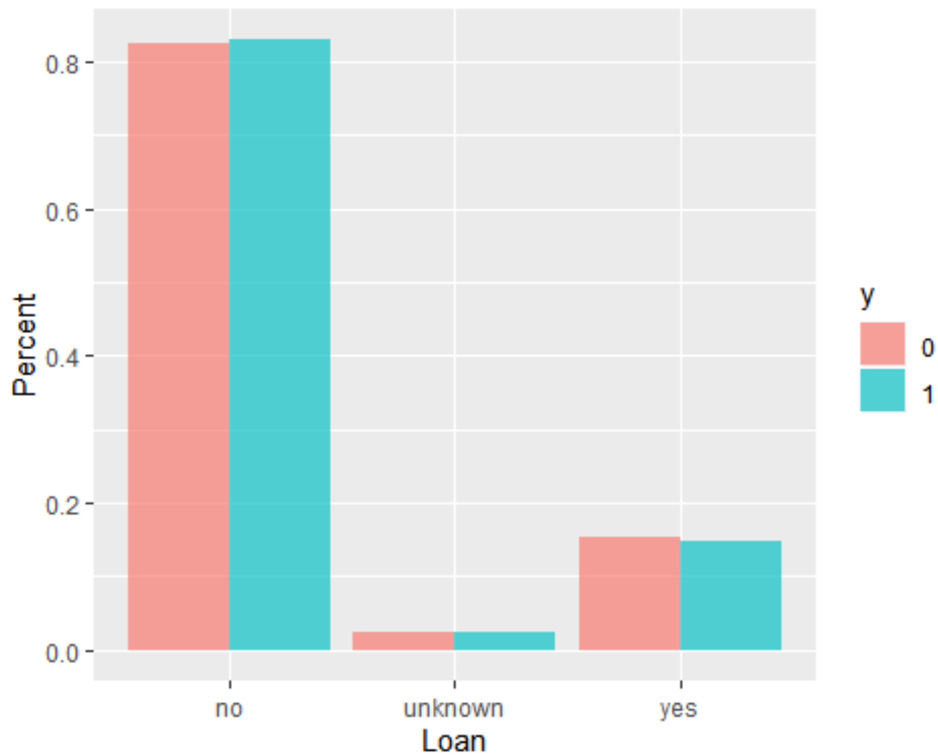
The data shows that people who aren't in default are a higher proportion of people who have subscribed for bank deposits.

```r
mytable <- table(data.df$housing, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("housing", "y", "perc")
ggplot(data = tab, aes(x = housing, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Housing")+ylab("Percent")
```



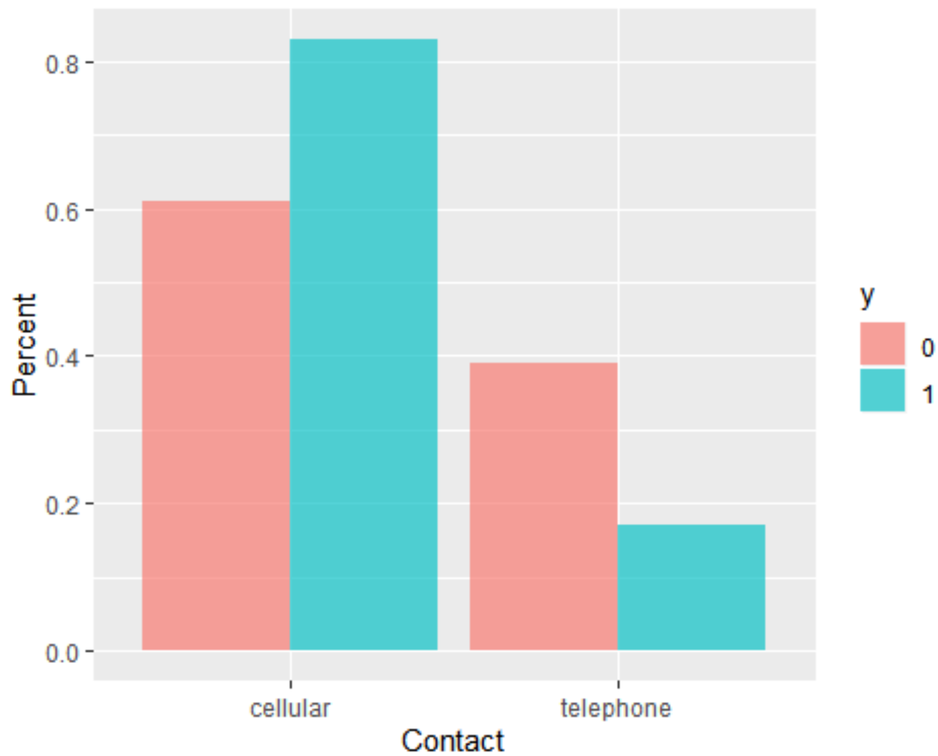
We see that a higher proportion of people who have subscribed for bank deposit are home owners versus ones that don't own their own houses.

```
mytable <- table(data.df$loan, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("loan", "y", "perc")
ggplot(data = tab, aes(x = loan, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Loan")+ylab("Percent")
```



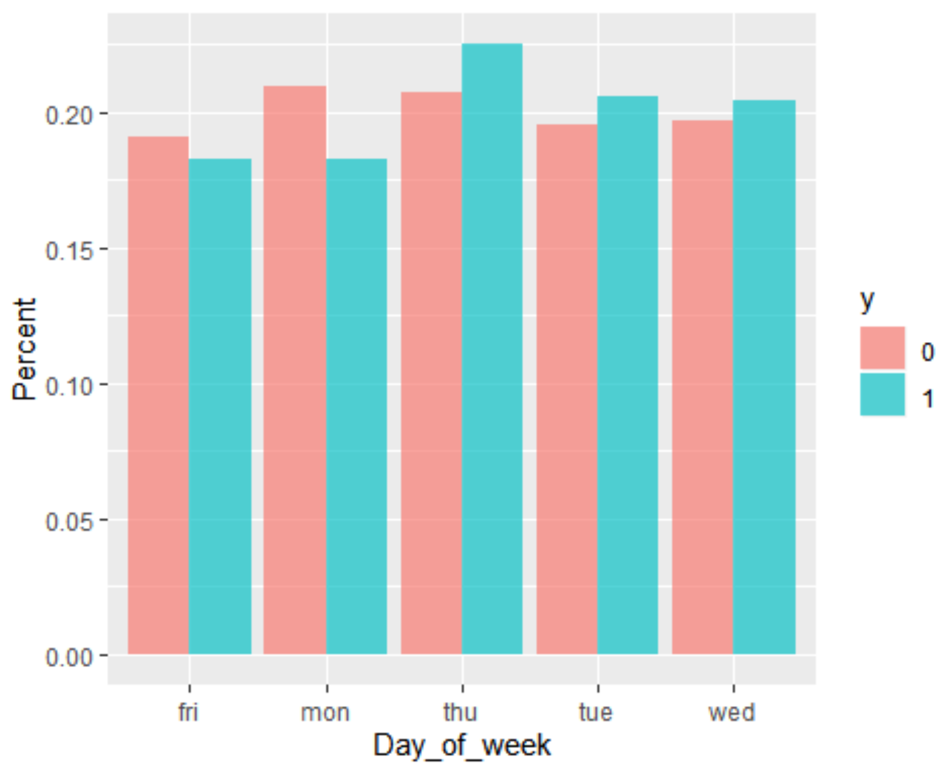
We see the proportion of people who have subscribed and not subscribed to a term deposit is the same for categories of the Loan.

```
mytable <- table(data.df$contact, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("contact", "y", "perc")
ggplot(data = tab, aes(x = contact, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Contact")+ylab("Percent")
```



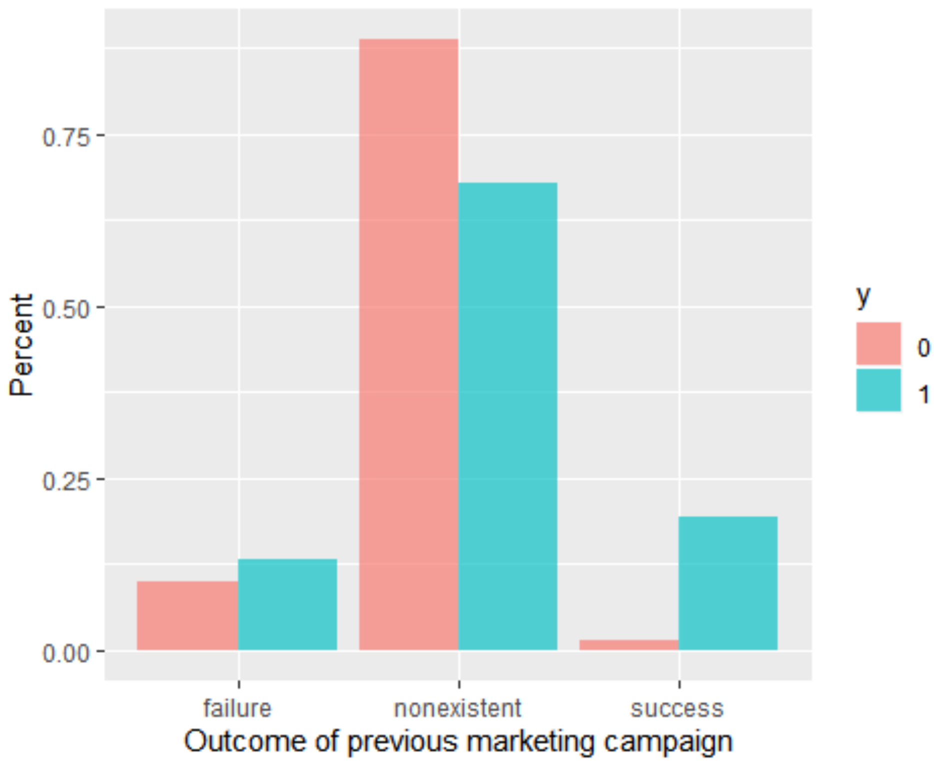
Customers who have cell phones, and therefore a more direct way of communicating, signed up for term deposits more than those who only had a landline telephone.

```
mytable <- table(data.df$day_of_week, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("day_of_week", "y", "perc")
ggplot(data = tab, aes(x = day_of_week, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Day_of_week")+ylab("Percent")
```



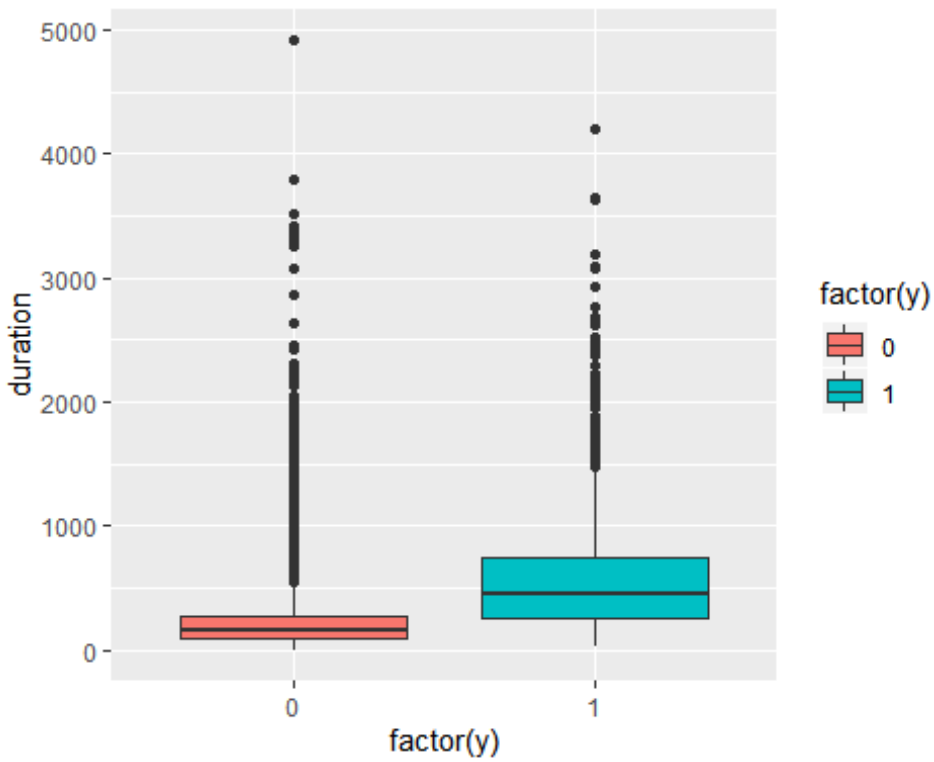
Campaigns that were performed midweek, on Tuesdays, Wednesdays, and Thursdays had a slightly higher proportion of people who subscribed for bank deposit.

```
mytable <- table(data.df$poutcome, data.df$y)
tab <- as.data.frame(prop.table(mytable, 2))
colnames(tab) <-  c("poutcome", "y", "perc")
ggplot(data = tab, aes(x = poutcome, y = perc, fill = y)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Outcome of previous marketing campaign")+ylab("Percent")
```



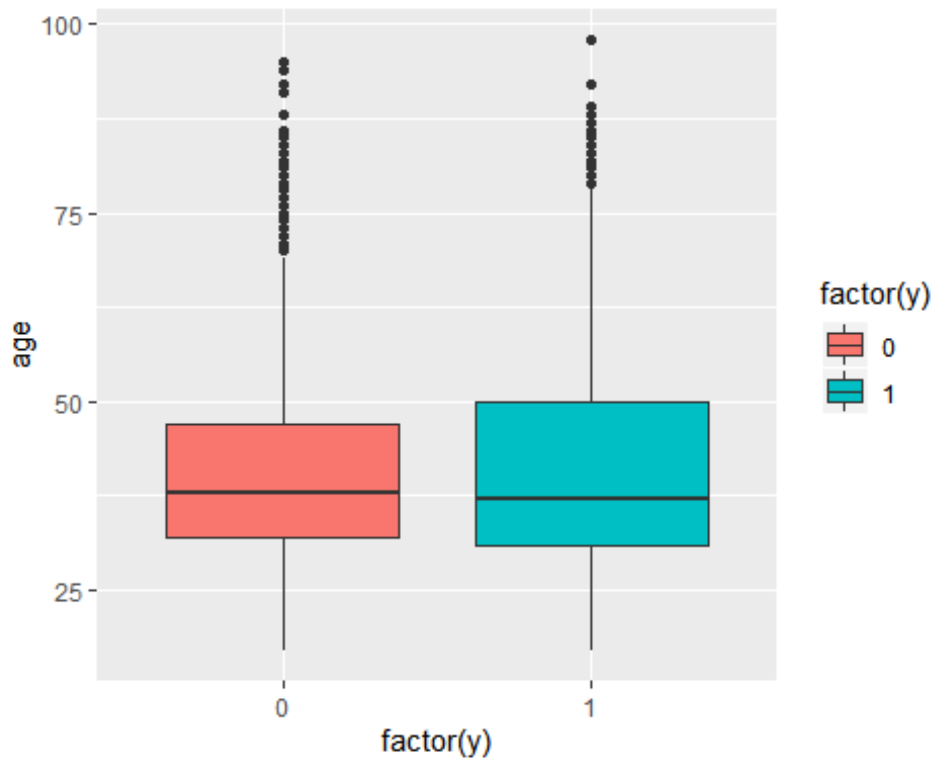
Potential customers who successfully connected and responded in previous campaigns had a higher proportion of signing up for the term deposit.

```
ggplot(data.df, aes(factor(y), duration)) + geom_boxplot(aes(fill = factor(y)
))
```



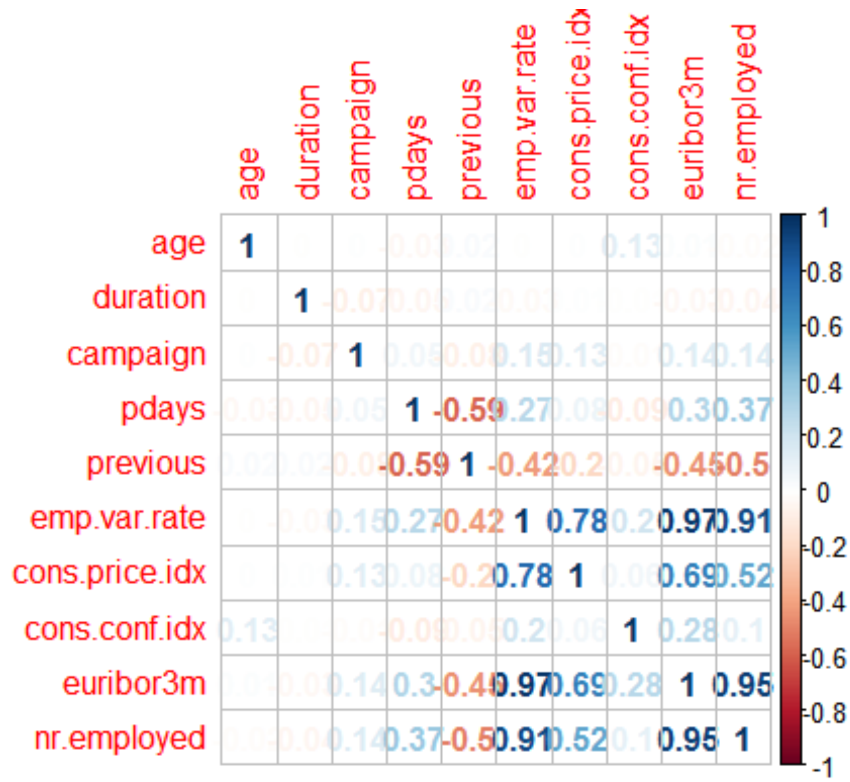
The longer the phone conversation the greater the conversion rate is for the potential customer to sign up for the term deposit. There are higher median and quartile ranges.

```
ggplot(data.df, aes(factor(y), age)) + geom_boxplot(aes(fill = factor(y)))
```



The age range for successful conversion has a slightly lower median, but higher quartile ranges.

```
df_cor <- select_if(data.df, is.numeric) %>% cor()
corrplot(df_cor, method = "number")
```



We see our target variable has a high positive correlation with duration and if the customer was involved and connected in a previous campaign, while there's negative correlation with Nr.employed (number of employees), pdays (number of days from last contact), Euribor3m (Euribor 3 month rate) and emp.var.rate (employee variation rate).

---

3. Results (includes Data Modeling and performance)

3.1. Data Preparation

Missing values for duration were filtered out (last contact duration, in seconds (numeric)) because if duration=0 then y="no" (no call was made). Thus, it doesn't make sense to have 0 second duration. I also filtered out education illiterate, and default yes because they only have 1 observation each. We can't predict these situations if they happen to be in the test data but not the train data.

```
data.df <- data.df %>%
  filter(duration != 0, education != "illiterate", default != "yes") %>%
  mutate(y = ifelse(y==1, 1, 0))
```

Split the data into training and test datasets:

```r
set.seed(123)
trainIndex <- createDataPartition(data.df$y,
                                  p = 0.8, # training contains 80% of data
                                  list = FALSE)
dfTrain <- data.df[ trainIndex,]
dfTest  <- data.df[-trainIndex,]

dim(dfTrain)

## [1] 32931    21

dim(dfTest)

## [1] 8232   21
```

The code and output above show that the trainData dataset has 8929 rows and 17 columns and the testData dataset gas 2233 rows and 17 columns. The number of columns remains the same because the dataset was split vertically.

3.2. Data Modeling using Random Forest

First the data set was divided into training and testing data with 80%-20% split respectively. A seed value was set using set.seed() function to make sure that the randomly split data could be regenerated. A random forest model was built using training data using randomforest package. We use 10 predictors for each split and grow 200 trees fully without pruning. A subset of predictors is randomly chosen without replacement at each split which helps in reducing the variance of the model overall. This is a prime advantage of random forest as compared to traditional decision trees.

In the below summary we can see that this model has an Out-Of-Bag error rate of 8.7%. The model also outputs a confusion matrix. We can see that random forest is doing a fairly good job in predicting the response variable i.e. deposit(Yes/No) field.

```r
set.seed(123)
# random forest
model_rf <- randomForest(as.factor(y)~.,
                  data = dfTrain,
                  ntree = 200,
                  mtry=10,
                  importance = TRUE)

print(model_rf)

##
## Call:
##  randomForest(formula = as.factor(y) ~ ., data = dfTrain, ntree = 200,
mtry = 10, importance = TRUE)
##                Type of random forest: classification
```

```
##                      Number of trees: 200
## No. of variables tried at each split: 10
##
##         OOB estimate of  error rate: 8.72%
## Confusion matrix:
##       0    1 class.error
## 0 27981 1244   0.0425663
## 1  1626 2080   0.4387480
```

```
pred_rf_prob <- predict(model_rf,
                          newdata = dfTest)
```

head(pred_rf_prob)

Model evaluation:

```
# put "pred_rf_prob" in a data frame
RF_outcome_test <- data.frame(dfTest$y)

# merge "model_rf" and "RF_outcome_test"
RF_comparison_df <- data.frame(pred_rf_prob, RF_outcome_test)

# specify column names for "RF_comparison_df"
names(RF_comparison_df) <- c("RF_Predicted_y", "RF_Observed_y")

RF_comparison_df$RF_Predicted_y <- as.factor(RF_comparison_df$RF_Predicted_y)
RF_comparison_df$RF_Observed_y <- as.factor(RF_comparison_df$RF_Observed_y)

# inspect "RF_comparison_df"
head(RF_comparison_df)
```

```
##    RF_Predicted_y RF_Observed_y
## 12              0             0
## 23              0             0
## 27              0             0
## 36              0             0
## 39              0             0
## 44              0             0
```

str(RF_comparison_df)

```
confusionMatrix(RF_comparison_df$RF_Observed_y,RF_comparison_df$RF_Predicted_
y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7029  273
##          1  415  515
##
```

```
##                Accuracy : 0.9164
##                  95% CI : (0.9102, 0.9223)
##     No Information Rate : 0.9043
##     P-Value [Acc > NIR] : 7.338e-05
##
##                   Kappa : 0.5532
##
##  Mcnemar's Test P-Value : 7.634e-08
##
##             Sensitivity : 0.9443
##             Specificity : 0.6536
##          Pos Pred Value : 0.9626
##          Neg Pred Value : 0.5538
##              Prevalence : 0.9043
##          Detection Rate : 0.8539
##    Detection Prevalence : 0.8870
##       Balanced Accuracy : 0.7989
##
##        'Positive' Class : 0
##
```

The RF test data consisted of 8232 observations. Out of which 7034 cases have been accurately predicted (TN->True Negatives) as negative class (0) which constitutes 85%. Also, 510 out of 8232 observations were accurately predicted (TP-> True Positives) as positive class (1) which constitutes 6%. Thus a total of 510 out of 8232 predictions where TP i.e, True Positive in nature.

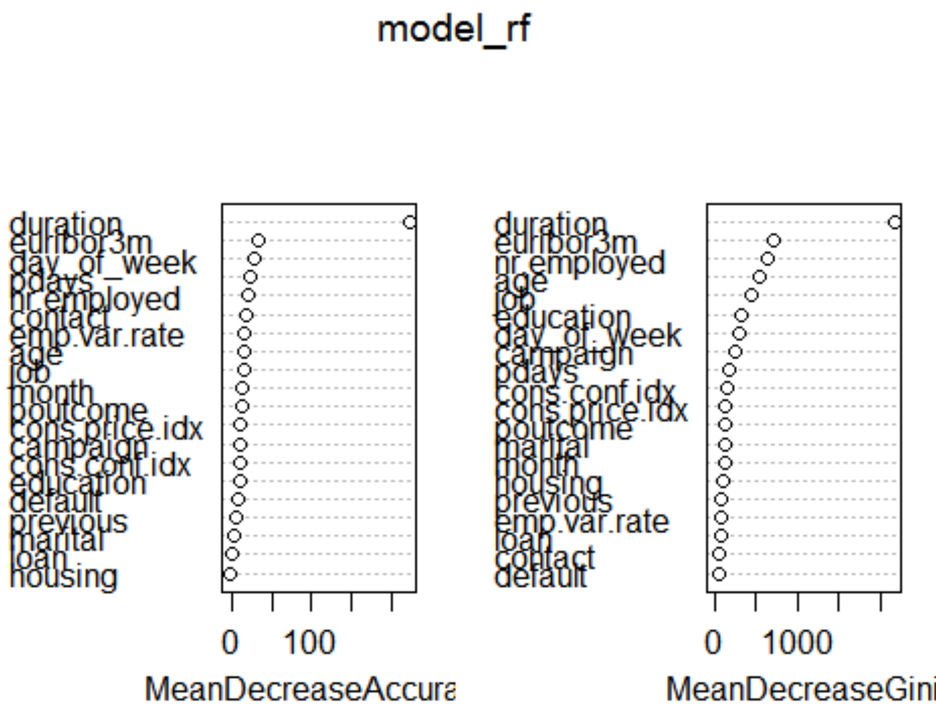
There were 420 cases of False Positives (FP) meaning 544 cases out of 8232 were actually negative but got predicted as positive.

There were 268 cases of False Negatives (FN) meaning 199 cases our of 8232 were actually positive in nature but got predicted as negative.

Accuracy of the model is the correctly classified positive and negative cases divided by all ther cases.The total accuracy of the model is 91.64%, which means the model prediction is very accurate.

Viewing the variable importance plot:

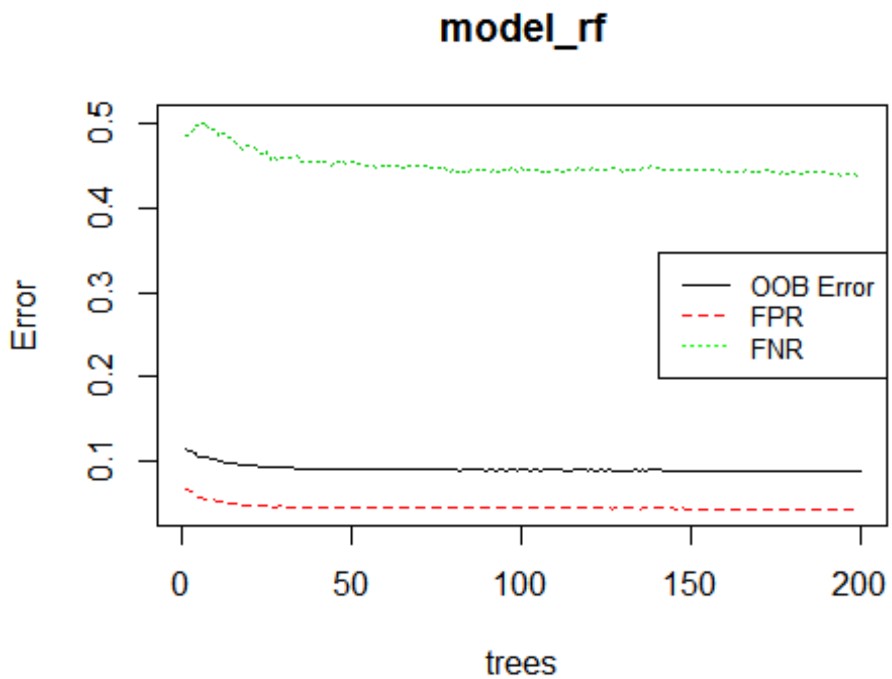
```
varImpPlot(model_rf)
```



By setting the importance argument on, we obtained the variable importance plot as above using varImpPlot() function and we can see that duration is highly significant in our data set.

We plot a graph for the error rate (False Positive Rate, False Negative Rate and Out-Of-Bag Error) with the increasing number of trees.

```
plot(model_rf)
legend("right", legend=c("OOB Error", "FPR", "FNR"),
       col=c("black", "red", "green"), lty=1:3, cex=0.8)
```



In the above plot, we can see the change of error with increasing number of trees. The False Negative Rate is higher compared to other error rate and False Positive Rate is lowest. The error rate starts dropping for at ntree~ 20. This says that our model is predicting 'Yes' cases more accurately than 'No' cases which can also be confirmed by the confusion matrix above.

3.2. Data Modeling using KNN

We will make a copy of our data set so that we can prepare it for our k-NN classification.

```
data_knn <- data.df

str(data_knn)

## 'data.frame':    41163 obs. of  21 variables:
##  $ age           : num  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 4 8 8 1
8 8 1 2 10 8 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 2 2 2 2 2
2 2 3 3 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4 2 4
3 6 8 6 4 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2 1 2 1
1 ...
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1 1 1 3
```

```
3 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1 1 1 1
1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2
2 2 2 2 ...
##  $ month         : Ord.factor w/ 10 levels "mar"<"apr"<"may"<..: 3 3 3 3 3
3 3 3 3 3 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2 2 2
2 2 2 ...
##  $ duration      : num  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : num  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2
2 2 2 2 2 2 ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -3
6.4 -36.4 ...
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : num  0 0 0 0 0 0 0 0 0 0 ...
```

Because k-NN algorithm involves determining distances between datapoints, we must use numeric variables only. This is applicable only to independent variables. The target variable for k-NN classification should remain a factor variable. First, we scale the data just in case our features are on different metrics. For example, if we had “duration” as a variable, it would be on a much larger scale than “age”, which could be problematic given the k-NN relies on distances. Note that we are using the ‘scale’ function here, which means we are scaling to a z-score metric.

We see that the variables “age”, “duration”, “campaign”, “pdays”, “previous”, “emp.var.rate”, “cons.price.idx”, “cons.conf.idx”, “euribor3m” and “nr.employed” are interger variables, which means they can be scaled.

```
data_knn[, c("age", "duration", "campaign", "pdays", "previous", "emp.var.rat
e", "cons.price.idx", "cons.conf.idx", "euribor3m","nr.employed")] <- scale(d
ata_knn[, c("age", "duration", "campaign", "pdays", "previous", "emp.var.rate
", "cons.price.idx", "cons.conf.idx", "euribor3m","nr.employed")])

head(data_knn)

##              age       job marital   education default housing loan
## 1  1.533684728 housemaid married    basic.4y      no      no   no
## 2  1.629657912  services married high.school unknown      no   no
## 3 -0.289805768  services married high.school      no     yes   no
## 4 -0.001886216    admin. married    basic.6y      no      no   no
## 5  1.533684728  services married high.school      no      no  yes
## 6  0.477979704  services married    basic.9y unknown      no   no
```

```
##      contact month day_of_week    duration   campaign     pdays   previous
## 1 telephone   may         mon  0.01036255 -0.5658418 0.1954061 -0.3494959
## 2 telephone   may         mon -0.42162181 -0.5658418 0.1954061 -0.3494959
## 3 telephone   may         mon -0.12463256 -0.5658418 0.1954061 -0.3494959
## 4 telephone   may         mon -0.41390781 -0.5658418 0.1954061 -0.3494959
## 5 telephone   may         mon  0.18778470 -0.5658418 0.1954061 -0.3494959
## 6 telephone   may         mon -0.23262865 -0.5658418 0.1954061 -0.3494959
##      poutcome emp.var.rate cons.price.idx cons.conf.idx euribor3m
## 1 nonexistent    0.6480509      0.7224735     0.8865513 0.7124339
## 2 nonexistent    0.6480509      0.7224735     0.8865513 0.7124339
## 3 nonexistent    0.6480509      0.7224735     0.8865513 0.7124339
## 4 nonexistent    0.6480509      0.7224735     0.8865513 0.7124339
## 5 nonexistent    0.6480509      0.7224735     0.8865513 0.7124339
## 6 nonexistent    0.6480509      0.7224735     0.8865513 0.7124339
##   nr.employed y
## 1   0.3317071 0
## 2   0.3317071 0
## 3   0.3317071 0
## 4   0.3317071 0
## 5   0.3317071 0
## 6   0.3317071 0
```

```
str(data_knn)
```

```
## 'data.frame':    41163 obs. of  21 variables:
##  $ age           : num  1.53368 1.62966 -0.28981 -0.00189 1.53368 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 4 8 8 1
8 8 1 2 10 8 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 2 2 2 2 2
2 2 3 3 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4 2 4
3 6 8 6 4 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2 1 2 1
1 ...
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1 1 1 3
3 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1 1 1 1
1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2
2 2 2 2 ...
##  $ month         : Ord.factor w/ 10 levels "mar"<"apr"<"may"<..: 3 3 3 3 3
3 3 3 3 3 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2 2 2
2 2 2 ...
##  $ duration      : num  0.0104 -0.4216 -0.1246 -0.4139 0.1878 ...
##  $ campaign      : num  -0.566 -0.566 -0.566 -0.566 -0.566 ...
##  $ pdays         : num  0.195 0.195 0.195 0.195 0.195 ...
##  $ previous      : num  -0.349 -0.349 -0.349 -0.349 -0.349 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2
2 2 2 2 2 2 ...
```

```
##  $ emp.var.rate  : num  0.648 0.648 0.648 0.648 0.648 ...
##  $ cons.price.idx: num  0.722 0.722 0.722 0.722 0.722 ...
##  $ cons.conf.idx : num  0.887 0.887 0.887 0.887 0.887 ...
##  $ euribor3m     : num  0.712 0.712 0.712 0.712 0.712 ...
##  $ nr.employed   : num  0.332 0.332 0.332 0.332 0.332 ...
##  $ y             : num  0 0 0 0 0 0 0 0 0 0 ...
```

We can see that the variables "job", "marital", "education", "default", "housing", "loan", "contact", "month", "day_of_week" and "poutcome" are factor variables that have two or more levels.

** Then dummy code variables that have two levels, but are not numeric. **

```
data_knn$contact <- dummy.code(data_knn$contact)
```

Next we dummy code variables that have three or more levels.

```
job <- as.data.frame(dummy.code(data_knn$job))
marital <- as.data.frame(dummy.code(data_knn$marital))
education <- as.data.frame(dummy.code(data_knn$education))
default <- as.data.frame(dummy.code(data_knn$default))
housing <- as.data.frame(dummy.code(data_knn$housing))
loan <- as.data.frame(dummy.code(data_knn$loan))
month <- as.data.frame(dummy.code(data_knn$month))
day_of_week <- as.data.frame(dummy.code(data_knn$day_of_week))
poutcome <- as.data.frame(dummy.code(data_knn$poutcome))
```

Rename "unknown" columns, so we don't have duplicate columns later).

```
job <- rename(job, unknown_job = unknown)
marital <- rename(marital, unknown_marital = unknown)
education <- rename(education , unknown_education  = unknown)
default <- rename(default , unknown_default  = unknown)
housing <- rename(housing , unknown_housing  = unknown)
loan <- rename(loan , unknown_loan  = unknown)

default <- rename(default , yes_default  = yes)
default <- rename(default , no_default  = no)

housing <- rename(housing , yes_housing  = yes)
housing <- rename(housing , no_housing  = no)

loan <- rename(loan , yes_loan  = yes)
loan <- rename(loan , no_loan  = no)
```

Combine new dummy variables with original data set.

```
data_knn <- cbind(data_knn, job, marital, education, default, housing, loan,
month, day_of_week,poutcome)

str(data_knn)
```

```
## 'data.frame':    41163 obs. of  72 variables:
##  $ age              : num  1.53368 1.62966 -0.28981 -0.00189 1.53368 ...
##  $ job              : Factor w/ 12 levels "admin.","blue-collar",..: 4 8
8 1 8 8 1 2 10 8 ...
##  $ marital          : Factor w/ 4 levels "divorced","married",..: 2 2 2
2 2 2 2 2 3 3 ...
##  $ education        : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4
2 4 3 6 8 6 4 ...
##  $ default          : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2
1 2 1 1 ...
##  $ housing          : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1
1 1 3 3 ...
##  $ loan             : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1
1 1 1 1 ...
##  $ contact          : num [1:41163, 1:2] 0 0 0 0 0 0 0 0 0 0 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr  "cellular" "telephone"
##  $ month            : Ord.factor w/ 10 levels "mar"<"apr"<"may"<..: 3 3
3 3 3 3 3 3 3 3 ...
##  $ day_of_week      : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2
2 2 2 2 2 ...
##  $ duration         : num  0.0104 -0.4216 -0.1246 -0.4139 0.1878 ...
##  $ campaign         : num  -0.566 -0.566 -0.566 -0.566 -0.566 ...
##  $ pdays            : num  0.195 0.195 0.195 0.195 0.195 ...
##  $ previous         : num  -0.349 -0.349 -0.349 -0.349 -0.349 ...
##  $ poutcome         : Factor w/ 3 levels "failure","nonexistent",..: 2 2
2 2 2 2 2 2 2 2 ...
##  $ emp.var.rate     : num  0.648 0.648 0.648 0.648 0.648 ...
##  $ cons.price.idx   : num  0.722 0.722 0.722 0.722 0.722 ...
##  $ cons.conf.idx    : num  0.887 0.887 0.887 0.887 0.887 ...
##  $ euribor3m        : num  0.712 0.712 0.712 0.712 0.712 ...
##  $ nr.employed      : num  0.332 0.332 0.332 0.332 0.332 ...
##  $ y                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ admin.           : num  0 0 0 1 0 0 1 0 0 0 ...
##  $ blue-collar      : num  0 0 0 0 0 0 0 1 0 0 ...
##  $ entrepreneur     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ housemaid        : num  1 0 0 0 0 0 0 0 0 0 ...
##  $ management       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ retired          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ self-employed    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ services         : num  0 1 1 0 1 1 0 0 0 1 ...
##  $ student          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ technician       : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ unemployed       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ unknown_job      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ divorced         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ married          : num  1 1 1 1 1 1 1 1 0 0 ...
##  $ single           : num  0 0 0 0 0 0 0 0 1 1 ...
##  $ unknown_marital  : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ basic.4y           : num  1 0 0 0 0 0 0 0 0 0 ...
##  $ basic.6y           : num  0 0 0 1 0 0 0 0 0 0 ...
##  $ basic.9y           : num  0 0 0 0 0 1 0 0 0 0 ...
##  $ high.school        : num  0 1 1 0 1 0 0 0 0 1 ...
##  $ illiterate         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ professional.course: num  0 0 0 0 0 0 1 0 1 0 ...
##  $ university.degree  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ unknown_education  : num  0 0 0 0 0 0 0 1 0 0 ...
##  $ no_default         : num  1 0 1 1 1 0 1 0 1 1 ...
##  $ unknown_default    : num  0 1 0 0 0 1 0 1 0 0 ...
##  $ yes_default        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ no_housing         : num  1 1 0 1 1 1 1 1 0 0 ...
##  $ unknown_housing    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ yes_housing        : num  0 0 1 0 0 0 0 0 1 1 ...
##  $ no_loan            : num  1 1 1 1 0 1 1 1 1 1 ...
##  $ unknown_loan       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ yes_loan           : num  0 0 0 0 1 0 0 0 0 0 ...
##  $ mar                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ apr                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ may                : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ jun                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ jul                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ aug                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ sep                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ oct                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ nov                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ dec                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ fri                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ mon                : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ thu                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ tue                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ wed                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ failure            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ nonexistent        : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ success            : num  0 0 0 0 0 0 0 0 0 0 ...
```

Remove original variables that had to be dummy coded.

```
data_knn <- data_knn %>% select(-one_of(c("job", "marital", "education", "def
ault", "housing", "loan", "month", "day_of_week", "poutcome")))

head(data_knn)
```

```
##            age contact.cellular contact.telephone    duration   campaign
## 1  1.533684728                0                 1  0.01036255 -0.5658418
## 2  1.629657912                0                 1 -0.42162181 -0.5658418
## 3 -0.289805768                0                 1 -0.12463256 -0.5658418
## 4 -0.001886216                0                 1 -0.41390781 -0.5658418
## 5  1.533684728                0                 1  0.18778470 -0.5658418
## 6  0.477979704                0                 1 -0.23262865 -0.5658418
```

```
##       pdays   previous emp.var.rate cons.price.idx cons.conf.idx euribor3m
## 1 0.1954061 -0.3494959    0.6480509      0.7224735     0.8865513 0.7124339
## 2 0.1954061 -0.3494959    0.6480509      0.7224735     0.8865513 0.7124339
## 3 0.1954061 -0.3494959    0.6480509      0.7224735     0.8865513 0.7124339
## 4 0.1954061 -0.3494959    0.6480509      0.7224735     0.8865513 0.7124339
## 5 0.1954061 -0.3494959    0.6480509      0.7224735     0.8865513 0.7124339
## 6 0.1954061 -0.3494959    0.6480509      0.7224735     0.8865513 0.7124339
##   nr.employed y admin. blue-collar entrepreneur housemaid management
## 1   0.3317071 0      0           0            0         1          0
## 2   0.3317071 0      0           0            0         0          0
## 3   0.3317071 0      0           0            0         0          0
## 4   0.3317071 0      1           0            0         0          0
## 5   0.3317071 0      0           0            0         0          0
## 6   0.3317071 0      0           0            0         0          0
##   retired self-employed services student technician unemployed unknown_job
## 1       0             0        0       0          0          0           0
## 2       0             0        1       0          0          0           0
## 3       0             0        1       0          0          0           0
## 4       0             0        0       0          0          0           0
## 5       0             0        1       0          0          0           0
## 6       0             0        1       0          0          0           0
##   divorced married single unknown_marital basic.4y basic.6y basic.9y
## 1        0       1      0               0        1        0        0
## 2        0       1      0               0        0        0        0
## 3        0       1      0               0        0        0        0
## 4        0       1      0               0        0        1        0
## 5        0       1      0               0        0        0        0
## 6        0       1      0               0        0        0        1
##   high.school illiterate professional.course university.degree
## 1           0          0                   0                 0
## 2           1          0                   0                 0
## 3           1          0                   0                 0
## 4           0          0                   0                 0
## 5           1          0                   0                 0
## 6           0          0                   0                 0
##   unknown_education no_default unknown_default yes_default no_housing
## 1                 0          1               0           0          1
## 2                 0          0               1           0          1
## 3                 0          1               0           0          0
## 4                 0          1               0           0          1
## 5                 0          1               0           0          1
## 6                 0          0               1           0          1
##   unknown_housing yes_housing no_loan unknown_loan yes_loan mar apr may
## 1               0           0       1            0        0   0   0   1
## 2               0           0       1            0        0   0   0   1
## 3               0           1       1            0        0   0   0   1
## 4               0           0       1            0        0   0   0   1
## 5               0           0       0            0        1   0   0   1
## 6               0           0       1            0        0   0   0   1
##   jun jul aug sep oct nov dec fri mon thu tue wed failure nonexistent
```

```
## 1   0   0   0   0   0   0   0   0   1   0   0   0       0           1
## 2   0   0   0   0   0   0   0   0   1   0   0   0       0           1
## 3   0   0   0   0   0   0   0   0   1   0   0   0       0           1
## 4   0   0   0   0   0   0   0   0   1   0   0   0       0           1
## 5   0   0   0   0   0   0   0   0   1   0   0   0       0           1
## 6   0   0   0   0   0   0   0   0   1   0   0   0       0           1
##   success
## 1       0
## 2       0
## 3       0
## 4       0
## 5       0
## 6       0
```

We are now ready for k-NN classification. We split the data into training and test sets. We partition 80% of the data into the training set and the remaining 20% into the test set.

Splitting the dataset into Test and Train:

```r
set.seed(1234) # set the seed to make the partition reproducible

# 80% of the sample size
sample_size <- floor(0.8 * nrow(data_knn))


train_index <- sample(seq_len(nrow(data_knn)), size = sample_size)

# put outcome in its own object
knn_outcome <- data_knn %>% select(y)

# remove original variable from the data set
data_knn <- data_knn %>% select(-y)



# creating test and training sets that contain all of the predictors
knn_data_train <- data_knn[train_index, ]
knn_data_test <- data_knn[-train_index, ]

# Split outcome variable into training and test sets using the same partition
as above.
knn_outcome_train <- knn_outcome[train_index, ]
knn_outcome_test <- knn_outcome[-train_index, ]
```

Using ‘class’ package, we run k-NN classification on our data. We have to decide on the number of neighbors (k).This is an iterative exercise as we need to keep changing the value of k to dtermine the optimum performance. In our case, we started with k=10 till k=20, and finally got an optimum performance at k=17.

```r
model_knn <- knn(train = knn_data_train, test = knn_data_test, cl = knn_outco
me_train, k=17)
```

Model evaluation:

```r
# put "knn_outcome_test" in a data frame
knn_outcome_test <- data.frame(knn_outcome_test)

# merge "model_knn" and "knn_outcome_test"
knn_comparison_df <- data.frame(model_knn, knn_outcome_test)

# specify column names for "knn_comparison_df"
names(knn_comparison_df) <- c("KNN_Predicted_y", "KNN_Observed_y")

knn_comparison_df$KNN_Predicted_y <- as.factor(knn_comparison_df$KNN_Predicte
d_y)
knn_comparison_df$KNN_Observed_y <- as.factor(knn_comparison_df$KNN_Observed_
y)

# inspect "knn_comparison_df"
head(knn_comparison_df)

##   KNN_Predicted_y KNN_Observed_y
## 1               0              0
## 2               0              0
## 3               0              0
## 4               0              0
## 5               0              0
## 6               0              0
```

Next, we compare our predicted values of deposit to our actual values. The confusion matrix gives an indication of how well our model predicted the actual values. The confusion matrix output also shows overall model statistics and statistics by class

```r
confusionMatrix(knn_comparison_df$KNN_Observed_y,knn_comparison_df$KNN_Predic
ted_y)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7071  179
##          1  586  397
##
##                Accuracy : 0.9071
##                  95% CI : (0.9006, 0.9133)
##     No Information Rate : 0.93
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.4618
```

```
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9235
##             Specificity : 0.6892
##          Pos Pred Value : 0.9753
##          Neg Pred Value : 0.4039
##              Prevalence : 0.9300
##          Detection Rate : 0.8589
##    Detection Prevalence : 0.8806
##       Balanced Accuracy : 0.8064
##
##        'Positive' Class : 0
##
```

The K-nn test data consisted of 8238 observations. Out of which 7128 cases have been accurately predicted (TN->True Negatives) as negative class (0) which constitutes 87%. Also, 367 out of 8238 observations were accurately predicted (TP-> True Positives) as positive class (1) which constitutes 4%. Thus a total of 367 out of 8238 predictions where TP i.e, True Positive in nature.

There were 544 cases of False Positives (FP) meaning 544 cases out of 8238 were actually negative but got predicted as positive.

There were 199 cases of False Negatives (FN) meaning 199 cases were actually positive in nature but got predicted as negative.

Accuracy of the model is the correctly classified positive and negative cases divided by all the cases. The total accuracy of the model is 91.13%, which means the model prediction is very accurate.

---

4. Conclusion

Model Comparison

Both the algorithms namely Random Forest and K Nearest Neighbor are generating high accuracy when trained with the bank marketing dataset.

The parameter comparison for both the model is:

Parameter	Random Forest	K-nn Model
Accuracy	91.64%	91.13%
Sensitivity	94.37%	93.03%

Specificity	65.55%	65.68%
Pos Pred Value	96.33%	97.31%
Neg Pred Value	54.84%	41.38%

- The accuracy of both the algorithms is very similar, and random forest model has a slightly higher accuracy compared to K-nn model.
- The sensitivity and specificity of both the algorithms is also very close, and random forest model has a slightly higher sensitivity compared to K-nn model.
- The Positive Pred Value of random forest model is a little lower as compared to K-nn model.
- The Negative Pred Value of random forest model is nearly 10% higher as compared to K-nn model.

At an overall level, the performance of both the model is similar, however Random Forest model has a better prediction performance for Negative classes and hence, we can go forward with selecting Random Forest as a better model for our objective.

Analysis Summary

The key insights derived from the overall analysis are:

-With respect to Marital Status, there is not an observed large difference in the proportion of people subscribed to term deposits and people without term deposits.

- Customers who sign up for bank deposits, proportionally, have achieved a higher level of education, than those who didn't sign up.

- The months of April, September, October, and December is the time when a higher proportion of people subscribed for term deposits.

- There are higher proportions for customers signing up for the term deposits who have the jobs of admin, retired, and students.

- People who aren't in default are a higher proportion of people who have subscribed for bank deposits.

- Higher proportion of people who have subscribed for bank deposit are home owners versus ones that don't own their own houses.

- The proportion of people who have subscribed and not subscribed to a term deposit is the same for categories of the Loan.

- Customers who have cell phones, and therefore a more direct way of communicating, signed up for term deposits more than those who only had a landline telephone.

- Campaigns that were performed midweek, on Tuesdays, Wednesdays, and Thursdays had a slightly higher proportion of people who subscribed for bank deposit.

- Potential customers who successfully connected and responded in previous campaigns had a higher proportion of signing up for the term deposit.

- The longer the phone conversation the greater the conversion rate is for the potential customer to sign up for the term deposit. There are higher median and quartile ranges.

- The age range for successful conversion has a slightly lower median, but higher quartile ranges.

- Subscribing to term deposit has a high positive correlation with duration and if the customer was involved and connected in a previous campaign, while there's negative correlation with Nr.employed (number of employees), pdays (number of days from last contact), Euribor3m (Euribor 3 month rate) and emp.var.rate (employee variation rate).

Target Market Strategy

The business problem defined in the introduction is to devise a target marketing strategy for the bank based on the behavioral data collected. We discovered the kinds of observations and behaviors of potential customers that result in them more likely to subscribe to a term deposit.

Using the above insights, the bank should devise a target marketing strategy that is customized towards potential customers with those who already have an existing account with the bank and have higher education. Those customers who generally are either employed in admin related jobs, or are students, or retired are those the bank can further pull. Those customers who are easily accessible with a mobile number will be the ones to answer the call, the key is to have an engaging and personable conversation with the customer and establish a relationship where they feel comfortable signing up for a term deposit with the bank. Those customers who were part of the previous campaign should be contacted by the bank again because we saw following up and having a continued dialogue and relationship results in a higher number of those who sign up. Also, in order to improve the probability of success, the campaigns should be launched in the last third of the calendar year when people are thinking of saving for the future and preparing for year-end taxes.

Future Work

Data analytics is usually used to analyze and work with big data such as the one we provided in the project here. It eases the cross-examination of the data and the methods of finding relationships within the data, so it becomes easier. There is a lot of things that we can do in future upon the existing model such as determining the right day of the week and time for each of the target audience or build custom models for individual clusters to further improve the prediction rate and reduce the error rate.

The work that has been done on this modeling and analysis is a great start for the bank to acquire more customers in an efficient way. There can always be improvements and tweaking as more data comes in, as well as exploring niches and clusters within the data. What a great start to enhancing the target market strategy!
