Prediction of Term deposit Subscription in Marketing Campaign by different models

Dhanya G

27/07/2021

Introduction

In banking system, when account holder deposits money, the bank can use that money to loan or lend to other clients or business organisations. To use these funds for lending, the bank will pay the investor in consideration of their services as interest on their account balance. Deposit accounts like this, the banker can withdraw their money at any time. This makes hard for banking organisations to know ahead of time how much they may loan to other customers at any given time. To control this situation, bank offers term deposit accounts. This account withholds funds for a fixed period of time in return for high interest rate.

In term deposits, clients can deposit or invest but can't withdraw money for certain period of time. This helps bank to loan money to other customers at any time without worrying about clients withdrawal of money at any period. Also, bank uses term deposits, thereby receiving higher interest rate from borrowers compared to what bank is paying as an interest for term deposit. Thus, it's a win-win situation for banks.

The interest gained on term deposit account is higher compared to standard deposit account or interest - bearing checking accounts. Increased rate is because access to the money is limited to the time period in term deposit account. It is also an extremely safe investment intriguing to conservative, low-risk investors. Though term deposit has more benefits compared to savings account, many people are not aware of this account and its features and how to approach this. So, this project focuses on targeting customers by analysing their characteristics ,thereby improving the strategies of marketing campaign.

This project is based on term deposits given by Portuguese banking institution. Portuguese bank uses phone calls as marketing campaign to reach out customers for creating term deposit accounts. Previous campaigns data and results are collected to provide insights for better upcoming campaign. The data is obtained from UCI Machine Learning Repository. To overcome the problem of which customer to call or not, one can look at dataset which contains attributes of customer information, client's social and economic details and previous campaign details and whether client subscribed term deposit or not in that campaign?. By using this information, I built a model to predict whether client will subscribe term deposit or not by efficiently targeting intended customers. This is done by selecting which attributes contribute to clients term deposit subscription. The required attributes are found by predictive models to predict clients subscription. Various models with their accuracies are built in this project and the main goal is to find the best model with higher accuracy to predict term deposit subscription.

Executive Summary

The objective of this project is to predict which features of customers should be targeted in the upcoming marketing campaign of a Portuguese bank. The marketing campaign is based on the phone calls. More than one contact to the same client is required, in order to access if the product(bank term deposit) would be (*'yes'*) or not (*'no'*) subscribed. We can draw insights from the data to improve the strategies for the upcoming marketing campaign by choosing attributes that contribute to term deposit subscription. The

models built in this project are Logistic regression and 15 other classification models from rminer package. The aim is to choose the best model with highest accuracy to predict term deposit subscription to clients.

The initial phase of the project gives a picture about the dataset with its overview and the necessary techniques to be implemented for reorganising the data into a format that is ready to explore.

Dataset

The data is related with direct marketing campaigns of a Portuguese banking institution. The dataset found here is sourced from *UCI Machine Learning Repository* collected between May 2008 to November 2010, very close to the data analysed in [1]. The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The archive was created as an ftp archive in 1987 by David Aha and fellow graduate students at UC Irvine. Since that time, it has been widely used by students, educators, and researchers all over the world as a primary source of machine learning data sets.

The dataset can be found here:https://archive.ics.uci.edu/ml/machine-learning-databases/00222/. It consists of two zip files: *bank-additional.zip* and *bank.zip*. The *bank-additional.zip* file is used here because it contains additional information such as social and economic attributes of clients which is helpful in determining term deposit subscription compared to *bank.zip*. Downloaded from https://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank-additional.zip.

The *bank-additional-full.csv* csv file is unzipped from *bank-additional.zip* zip file. It consists of 41188 observations and *20 descriptive* and *1 target feature (yes/no)*.

Output attribute

The target attribute is *y*. It consists of two values, *yes* or *no*. I.e, Has the client subscribed term deposit or not?. Since *y* has two values, it is a binary classification problem. The classification goal is to predict if the client will subscribe (yes/no) a term deposit.

Descriptive attributes

Bank Client Data

1. age - clients age (numeric)
2. job - type of job (categorical: 'admin.',' blue-collar', 'entrepreneur',' housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician',' unemployed',' unknown'))
3. marital - marital status of client (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
4. education (categorical: 'basic.4y', 'basic.6y', 'basic.9y', 'high.school', 'illiterate', 'professional.course', 'university.degree', 'unknown')
5. default - has credit in default - failure to repay debt (categorical: 'no', 'yes', 'unknown')
6. housing - has client took housing loan? (categorical: 'no', 'yes', 'unknown')
7. loan - has personal loan? (categorical: 'no', 'yes', 'unknown')

Related with the last contact of the current campaign:

8. contact - how did bank communicate? (categorical: 'cellular', 'telephone')
9. month - last contact month of year (categorical: 'jan', 'feb', 'mar', . . . , 'nov', 'dec')
10. day_of_week: last contact day of the week (categorical: 'mon', 'tue', 'wed', 'thu', 'fri')
11. duration: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.

Campaign Attributes

12. campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
13. pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
14. previous: number of contacts performed before this campaign and for this client (numeric)
15. poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')

Social and Economic context Attributes

16. emp.var.rate: employment variation rate - quarterly indicator (numeric)
17. cons.price.idx: consumer price index - average change in prices over time that consumers pay for a basket of goods and services - monthly indicator (numeric)
18. cons.conf.idx: consumer confidence index - degree of optimism consumers are expressing through their activities of savings and spending - monthly indicator (numeric)
19. euribor3m: euribor 3 month rate - daily indicator (numeric)
20. nr.employed: number of employees - quarterly indicator (numeric)

Each row represents characteristic of a single customer. The class *unknown* denotes no known information about that customer attributes.

Overview

For this project, I built predictive model using Regression model (Logistic Regression model) and 14 other classification models such as Conditional Inference tree(ctree) model, Decision tree model(rpart), Generalized Linear Model(cv.glmnet), k-nearest neighbour model(knn), Support Vector Machine model(SVM), Least Squares Support Vector Machine model(lssvm), MultiLayer Perceptron with one hidden layer(mlp), Random Forest algorithm(randomForest), eXtreme Gradient Boosting (Tree) (xgboost), Bagging model, Adaboost model(boosting), Linear Discriminant Analysis model(LDA), Logistic regression model (multinom) and Naive Bayes Model(naiveBayes). These models predict whether client subscribe term deposit or not. All the models will be evaluated and the model with highest accuracy is chosen as a best model in predicting term deposit subscription.

Data Exploration and Visualization

Data Preprocessing

In this project, I use Bank Telemarketing dataset from UCI Machine Learning Repository. The dataset is divided into two parts namely, *bank_data* (90%) which is used for training the model and *validation* (10%) for predicting and testing the model. The *validation* set (final hold-out test set) should only be used at the end of the project to test the final algorithm which outputs maximum accuracy. Because of that, *bank_data* dataset should be further split into two parts, *bank_train_data* set for training and *bank_validation data* for testing the model until required accuracy is reached. Finally, for the final model with maximimum accuracy, we train the entire *bank_data* set with the *validation* set to predict term deposit subscription with highest accuracy value. For rminer classification models, *bank_train_data* data is split into training and test sets by *holdout()* function. The model with highest accuracy is chosen as the best model in predicting clients term deposit subscription.

1.Installing and loading packages

Installing required packages for the project

```r
if(!require(dplyr)) install.packages("dplyr")
if(!require(tidyverse)) install.packages("tidyverse")
if(!require(caret)) install.packages("caret")
if(!require(data.table)) install.packages("data.table")
if(!require(rminer)) install.packages("rminer")
```

Loading required libraries and data

```
library(dplyr)
library(tidyverse)
library(caret)
library(data.table)
library(rminer)
```

2.Data Extraction

The required Bank Marketing dataset can be extracted from UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/machine-learning-databases/00222/). Data can be downloaded from:"https://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank-additional.zip". Unzip *bank-additional-full.csv* from *bank-additional* zip file which consists of full observations compared to others. The csv file is imported in R using base R function read,csv() and copied into object *data*

```
dl <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank-additional.zip",dl)
data <- read.csv2(unzip(dl,"bank-additional/bank-additional-full.csv"),header = TRUE,
                       stringsAsFactors = TRUE)
```

3.Exploring the dataset

Bank Marketing dataset consists of 41188 rows and 21 columns. Each row represents characteristic about individual customer. Each column represents features needed to analyse customer behavior. It is represented as a tidy format below:

```
# Number of observations in the dataset.
dim(data)
```

```
## [1] 41188    21
```

```
# There are 41188 rows and 21 columns in the dataset.

# Overview of the dataset with initial 6 rows
head(data)
```

```
##   age       job marital   education default housing loan   contact month
## 1  56 housemaid married    basic.4y      no      no   no telephone   may
## 2  57  services married high.school unknown      no   no telephone   may
## 3  37  services married high.school      no     yes   no telephone   may
## 4  40    admin. married    basic.6y      no      no   no telephone   may
## 5  56  services married high.school      no      no  yes telephone   may
## 6  45  services married    basic.9y unknown      no   no telephone   may
##   day_of_week duration campaign pdays previous    poutcome emp.var.rate
## 1         mon      261        1   999        0 nonexistent          1.1
## 2         mon      149        1   999        0 nonexistent          1.1
## 3         mon      226        1   999        0 nonexistent          1.1
## 4         mon      151        1   999        0 nonexistent          1.1
## 5         mon      307        1   999        0 nonexistent          1.1
## 6         mon      198        1   999        0 nonexistent          1.1
##   cons.price.idx cons.conf.idx euribor3m nr.employed  y
## 1         93.994         -36.4     4.857        5191 no
## 2         93.994         -36.4     4.857        5191 no
## 3         93.994         -36.4     4.857        5191 no
## 4         93.994         -36.4     4.857        5191 no
## 5         93.994         -36.4     4.857        5191 no
## 6         93.994         -36.4     4.857        5191 no
```

```
# Structure of the dataset
str(data)
```

```
## '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  : Factor w/ 10 levels "-0.1","-0.2",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ cons.price.idx: Factor w/ 26 levels "92.201","92.379",..: 19 19 19 19 19 19 19 19 19 19 ...
##  $ cons.conf.idx : Factor w/ 26 levels "-26.9","-29.8",..: 10 10 10 10 10 10 10 10 10 10 ...
##  $ euribor3m     : Factor w/ 316 levels "0.634","0.635",..: 288 288 288 288 288 288 288 288 288 288
##  $ nr.employed   : Factor w/ 11 levels "4963.6","4991.6",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Summary of the dataset
summary(data)
```

```
##       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           loan      
##  university.degree  :12168   no     :32588   no     :18622   no     :33950  
##  high.school        : 9515   unknown: 8597   unknown:  990   unknown:  990  
##  basic.9y           : 6045   yes    :    3   yes    :21576   yes    : 6248  
##  professional.course: 5243                                                  
##  basic.4y           : 4176                                                  
##  basic.6y           : 2292                                                  
##  (Other)            : 1749                                                  
##       contact           month       day_of_week    duration     
##  cellular :26144   may    :13769   fri:7827    Min.   :   0.0  
##  telephone:15044   jul    : 7174   mon:8514    1st Qu.: 102.0  
##                    aug    : 6178   thu:8623    Median : 180.0  
##                    jun    : 5318   tue:8090    Mean   : 258.3  
##                     nov    : 4101   wed:8134    3rd Qu.: 319.0
##                     apr    : 2632               Max.   :4918.0
##                     (Other): 2016
##     campaign          pdays          previous            poutcome
##  Min.   : 1.000   Min.   :  0.0   Min.   :0.000   failure    : 4252
##  1st Qu.: 1.000   1st Qu.:999.0   1st Qu.:0.000   nonexistent:35563
##  Median : 2.000   Median :999.0   Median :0.000   success    : 1373
##  Mean   : 2.568   Mean   :962.5   Mean   :0.173
##  3rd Qu.: 3.000   3rd Qu.:999.0   3rd Qu.:0.000
##  Max.   :56.000   Max.   :999.0   Max.   :7.000
##
##   emp.var.rate   cons.price.idx cons.conf.idx    euribor3m      nr.employed
##  1.4    :16234   93.994 :7763   -36.4  :7763   4.857  : 2868   5228.1 :16234
##  -1.8   : 9184   93.918 :6685   -42.7  :6685   4.962  : 2613   5099.1 : 8534
##  1.1    : 7763   92.893 :5794   -46.2  :5794   4.963  : 2487   5191   : 7763
##  -0.1   : 3683   93.444 :5175   -36.1  :5175   4.961  : 1902   5195.8 : 3683
##  -2.9   : 1663   94.465 :4374   -41.8  :4374   4.856  : 1210   5076.2 : 1663
##  -3.4   : 1071   93.2   :3616   -42    :3616   4.964  : 1175   5017.5 : 1071
##  (Other): 1590   (Other):7781   (Other):7781   (Other):28933   (Other): 2240
##    y
##  no :36548
##  yes: 4640
##
##
##
##
##
```

It consists of 21 features namely, age, job, marital, education, default, housing, loan, contact, month, day_of_week, duration, campaign, pdays, previous, poutcome, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed and y(output variable). It is made of 20 descriptive attributes(client and campaign information) and 1 target attribute(y) Output variable y has two values yes or no, which denotes whether client has subscribed term deposit or not?

4. Data Cleaning - Scanning for NAs

The *data* dataset is scanned for missing values using is.na() function. Because, NAs can disrupt the model later while predicting the term deposit subsription.

```
colSums(is.na(data))
```

```
##            age            job        marital      education        default
##              0              0              0              0              0
##        housing           loan        contact          month    day_of_week
##              0              0              0              0              0
##       duration       campaign          pdays       previous       poutcome
##              0              0              0              0              0
##   emp.var.rate cons.price.idx  cons.conf.idx      euribor3m    nr.employed
##              0              0              0              0              0
##              y
##              0
```

It is proven that all attributes in the dataset doesn't have NA values. So, there is no problem in future to transform NAs into mean or median of the attribute to get a real predictive model.

5. Splitting the data

Bank Marketing Dataset is split into two parts namely, *bank_data* and *validation* sets. The *validation* set will be 10% of bank marketing data.

```
set.seed(1, sample.kind="Rounding")
# Splitting the dataset into training and testing sets
test_index <- createDataPartition(y = data$y, times = 1, p = 0.1, list = FALSE)
bank_data <- data[-test_index,]
bank_temp <- data[test_index,]
# Make sure euribor3m in validation set are also in bank_data set
validation <- bank_temp %>%
  semi_join(bank_data, by = "euribor3m")
# Bring back removed rows from validation to bank_data set
removed <- anti_join(bank_temp, validation)
bank_data <- rbind(bank_data, removed)
rm(dl, test_index, bank_temp, removed) #Clear out unwanted files
# Validation set should only be used at the end of the final model.
```

Data Exploration and Visualization

General overview of the dataset:

```
# Number of observations(rows and columns) in the dataset
dim(bank_data)
```

```
## [1] 37071    21
```

```
dim(validation)
```

```
## [1] 4117   21
```

```
# Glimpse of the dataset
glimpse(bank_data)
```

```
## Rows: 37,071
## Columns: 21
## $ age            <int> 56, 57, 37, 40, 56, 45, 59, 41, 24, 25, 41, 25, 29, 57,~
## $ job            <fct> housemaid, services, services, admin., services, servic~
## $ marital        <fct> married, married, married, married, married, married, m~
## $ education      <fct> basic.4y, high.school, high.school, basic.6y, high.scho~
## $ default        <fct> no, unknown, no, no, no, unknown, no, unknown, no, no, ~
## $ housing        <fct> no, no, yes, no, no, no, no, no, yes, yes, no, yes, no,~
## $ loan           <fct> no, no, no, no, yes, no, no, no, no, no, no, no, yes, n~
## $ contact        <fct> telephone, telephone, telephone, telephone, telephone, ~
## $ month          <fct> may, may, may, may, may, may, may, may, may, may, may, ~
## $ day_of_week    <fct> mon, mon, mon, mon, mon, mon, mon, mon, mon, mon, mon, ~
## $ duration       <int> 261, 149, 226, 151, 307, 198, 139, 217, 380, 50, 55, 22~
## $ campaign       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ pdays          <int> 999, 999, 999, 999, 999, 999, 999, 999, 999, 999, 999, ~
## $ previous       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ poutcome       <fct> nonexistent, nonexistent, nonexistent, nonexistent, non~
## $ emp.var.rate   <fct> 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, ~
## $ cons.price.idx <fct> 93.994, 93.994, 93.994, 93.994, 93.994, 93.994, 93.994,~
## $ cons.conf.idx  <fct> -36.4, -36.4, -36.4, -36.4, -36.4, -36.4, -36.4, -36.4,~
## $ euribor3m      <fct> 4.857, 4.857, 4.857, 4.857, 4.857, 4.857, 4.857, 4.857,~
## $ nr.employed    <fct> 5191, 5191, 5191, 5191, 5191, 5191, 5191, 5191, 5191, 5~
## $ y              <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no,~
```

There are 37071 rows and 21 columns in the training dataset. We have factor and integer as classes for data. Final holdout set *validation* consists of 4117 observations(about 10% of training set). The *validation* set (final hold-out test set) should only be used at the end of the project to test the final model.

Has the client subscribed a deposit?

The main goal is to predict *y* (term deposit subscription). Lets look at that:

```
unique(bank_data$y)
```

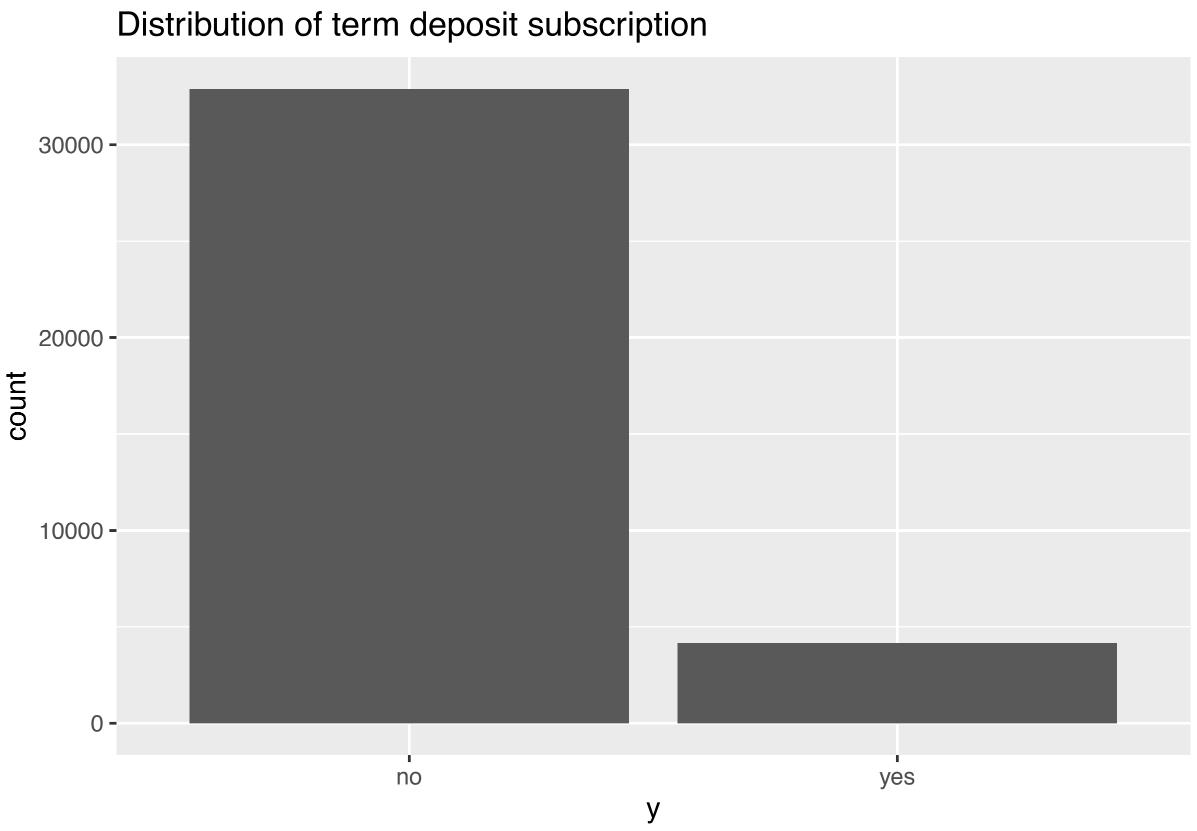
```
## [1] no  yes
## Levels: no yes
```

```
table(bank_data$y)
```

```
##
##    no   yes
## 32895  4176
```

The variable *y* has two levels of factor values - *yes* or *no*. *yes* - client subscribed term deposit *no* - client does not subscribe term deposit Table lists how many clients subscribed(yes) term deposit or not(no). Lets look at the distribution of y:

```
## # A tibble: 2 x 2
##   y     count
##   <fct> <int>
## 1 no    32895
## 2 yes    4176
```



This shows clients who does not subscribe term deposit are higher in count due to its large proportion of number in the dataset compared to clients who subscribed term deposit in this campaign.

Distribution of Clients Age

Number of unique age of clients in training dataset are:

```
length(unique(bank_data$age))
```

```
## [1] 78
```

```
min(bank_data$age)
```

```
## [1] 17
```

```
max(bank_data$age)
```

```
## [1] 98
```

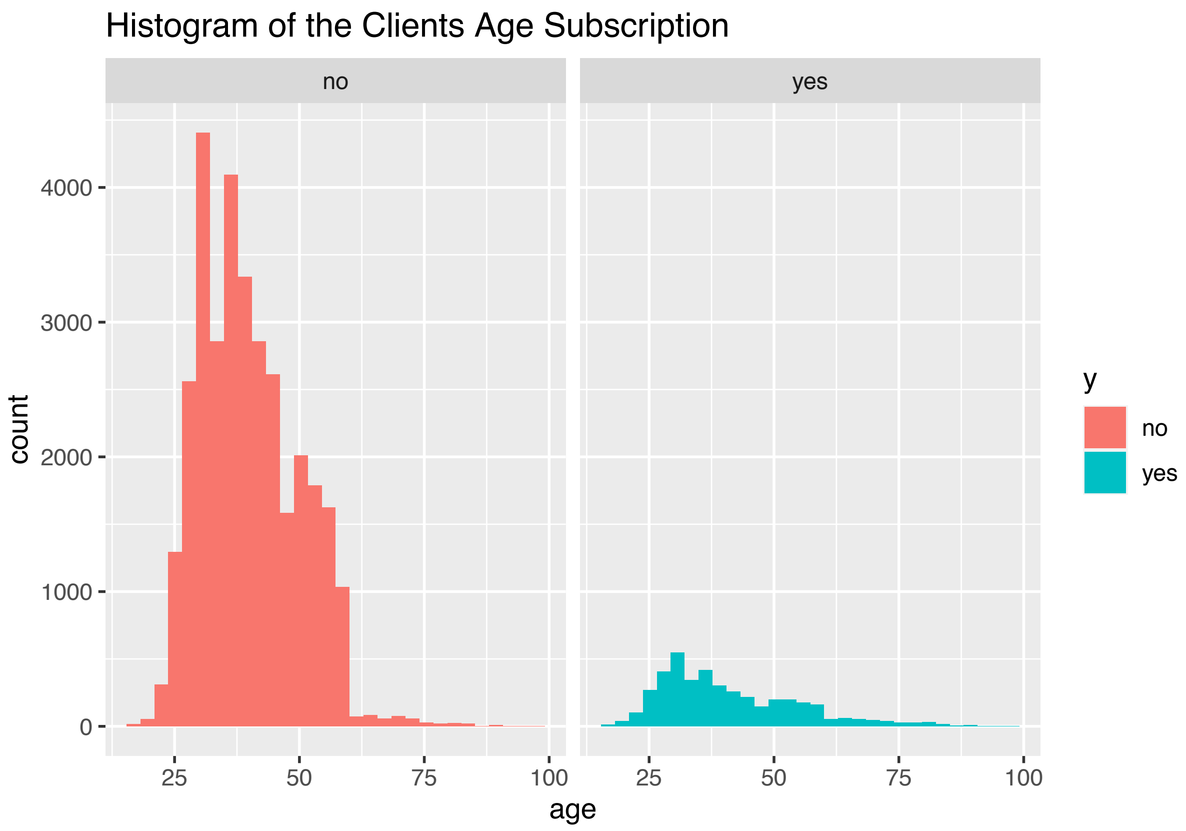
```
table(bank_data$age)
```

```
## 
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32 
##    4   27   36   57   89  123  200  422  534  608  764  887 1316 1545 1739 1671 
##   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48 
## 1654 1547 1578 1621 1311 1275 1303 1062 1141 1035  940  907 1004  921  842  887 
##   49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64 
##  756  789  663  696  667  624  583  642  576  521  419  257   65   59   46   55 
##   65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80 
##   43   55   24   31   31   42   50   33   32   31   23   32   16   22    9   25 
##   81   82   83   84   85   86   87   88   89   91   92   94   95   98 
##   15   16   15    7   13    7    1   19    2    1    4    1    1    2
```

This shows that in the marketing campaign, minimum age person who have been contacted are 17 whereas maximum age of the person is 98. There are 78 different ages of person who are client to banks. The table shows how many times the same type of age person has been contacted during the campaign

Does age play a role in predicting output variable, y? Lets look about that

Lets separate clients subscription yes and no against age in a histogram to clearly understand the marketing campaign:



Histogram Plot shows clients around the age group 30 to 40 are higher in number. This implies banking systems choose more people around 30 to 40 for marketing campaigns compared to others for coercing clients for term deposit subscription. And one can see decrease in number after age 60 drastically. This means clients age>60 does not subscribe term deposit much in number.

The age group between 30 to 40 subscribe term deposit(yes) more, although all age group people subscribes in a less manner. Surprisingly, the same age group(30 - 40) also have higher count in who did not subscribe term deposit(no) too. This leads to the fact that this is the most sought after group due to its highest proportion in total.

What are the different type of jobs of client?

Number of distinct jobs of clients in training dataset are:

```
n_distinct(bank_data$job)
```

```
## [1] 12
```

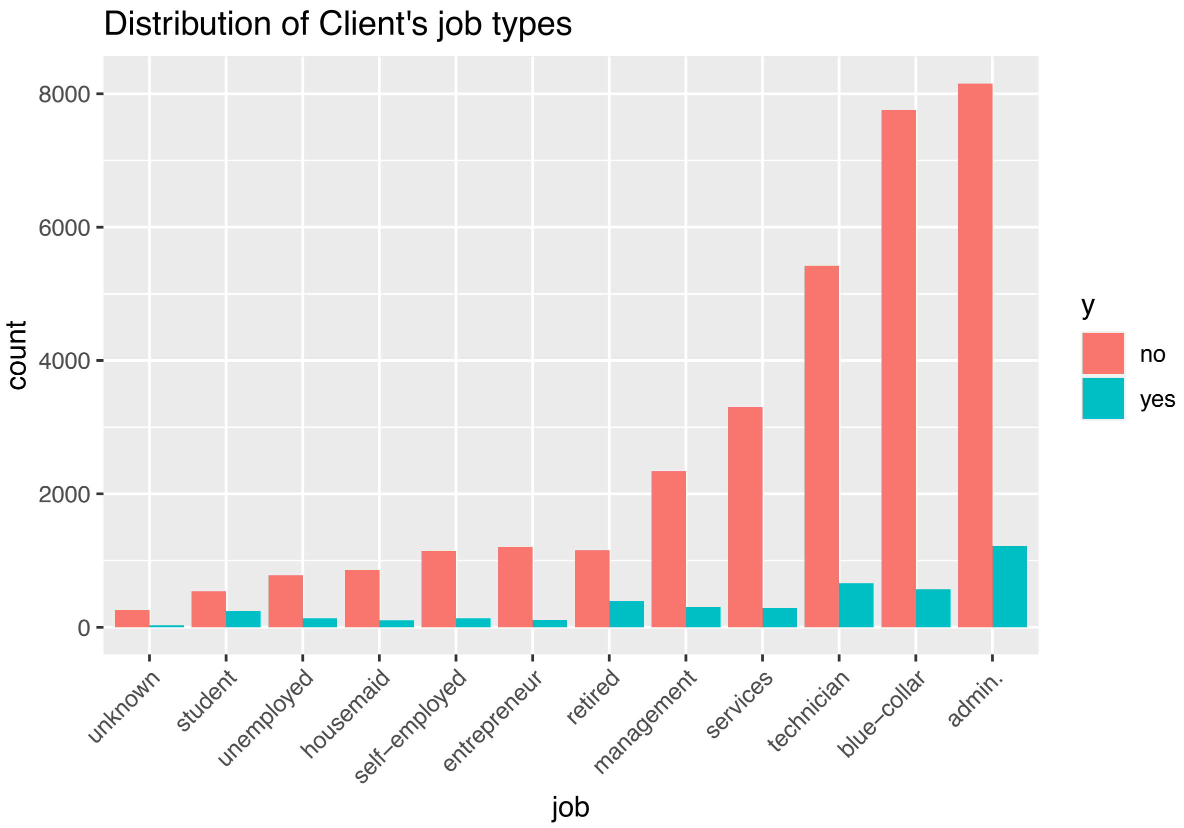
```
table(bank_data$job)
```

```
##
##        admin.   blue-collar  entrepreneur     housemaid    management
##          9374          8322          1310           957          2636
##       retired self-employed      services       student    technician
##          1549          1277          3586           776          6079
##    unemployed       unknown
##           913           292
```

There are 12 different types of job for client in the marketing campaign. Table lists what are the different types of job as well as how many clients are in the same job type in the campaign.

Distribution of Job types Of Clients

Does job of a client plays a role in predicting output variable, y? Lets look about that



From the plot, it is clear unknown and student are lower in number compared to others. People with admin. job who does not subscribe are larger in number (8153) in the dataset. This implies admin.job people are more active in deposit subscription in both "yes"(1221) and "no"(8153). People in admin subscribes term deposit more followed by technician(660) and blue-collar(565). This is due to higher proportion in total of admin jobs in the dataset.

To confirm if job attribute contributes to term deposit subscription, lets run chi-squared test and see,

```
##
##  Pearson's Chi-squared test
##
## data:  c_job
## X-squared = 883.78, df = 11, p-value < 2.2e-16
```

Since p-value is less than 0.05, it is clear that age is an important factor in predicting term deposit subscription. Because, p-value < 0.05 denotes that results are considered statistically significant. i.e, attribute job is significant(dependent) to attribute y(term deposit subscription). So, job feature contributes to deposit subscription and thus job feature should be included in predicting the model.

What are the distinct marital status of client?

Number of different jobs of clients in training dataset are:

```
n_distinct(bank_data$marital)
```

```
## [1] 4
```

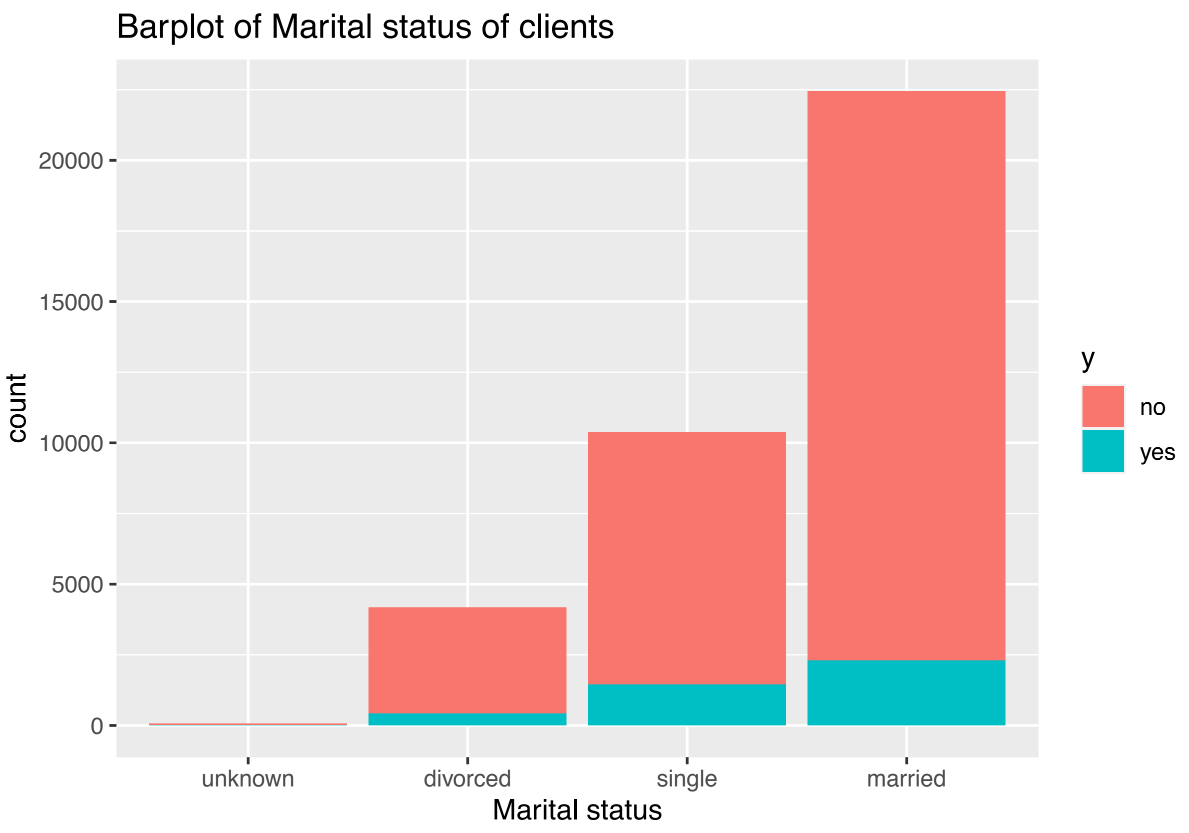
```
table(bank_data$marital)
```

```
##
## divorced  married   single  unknown
##     4178    22449    10372       72
```

There are 4 categories in the marital status of client. They are married, single, divorced and unknown.Table shows how many number of clients belong to the same marital category in the campaign. There are 72 unknown entries of clients for marital status in the marketing campaign. The value *unknown* refers that there is no information about the marital details for that client.

Distribution of Marital Status of clients

Does marital status of a client plays a role in predicting output variable, y? Lets look about that,



Married people avails the highest number of term deposits followed by single people. Plot shows Married people who does not subscribe term deposits are also higher in number(20157) followed by single people(y = "no") of 8922 respectively. This result is also due to higher proportion of people in that category. *Unknown* clients who subscribe term deposit are lower in number(12) followed by unknown clients who does not subscribe(y = "no").

To check if marital attribute contributes to term deposit subscription, lets run chi-squared test and see,

```
##
##  Pearson's Chi-squared test
##
## data:  c_mar
## X-squared = 109.26, df = 3, p-value < 2.2e-16
```

It is clear, p-value < 0.05 implies that attribute marital is statistically significant(dependent) to attribute y(term deposit subscription) which means that marital details of client is an important feature in term deposit subscription.

Basic Educational details of client:

Number of different education details of clients in trainng set are:

```
n_distinct(bank_data$education)
```

```
## [1] 8
```

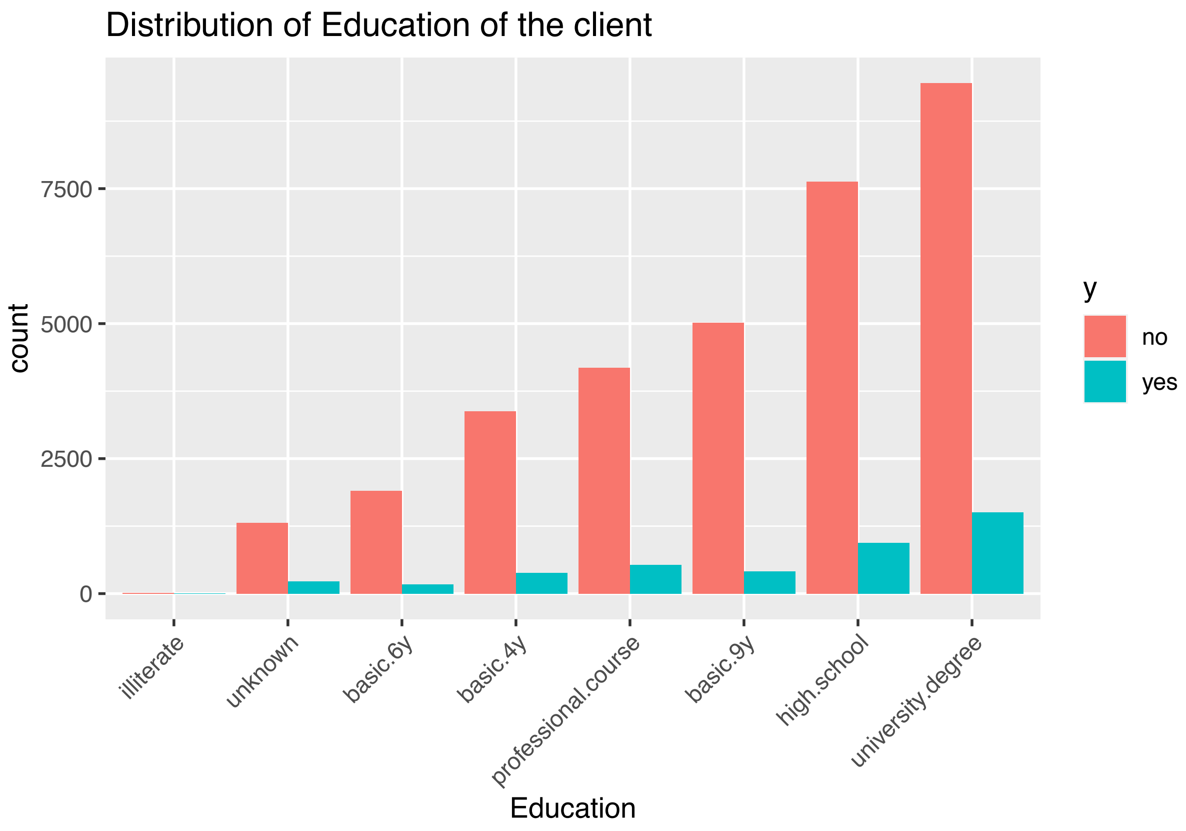
```
table(bank_data$education)
```

```
##
##           basic.4y            basic.6y            basic.9y         high.school
##               3764                2069                5427                8569
##         illiterate professional.course   university.degree             unknown
##                 16                4722               10963                1541
```

There are 8 categories of Education status in the dataset namely, basic.4y, high.school, basic.6y, basic.9y, professional.course, unknown, university.degree and illiterate. The table lists number of clients who are in the same education category of the campaign. There are 1541 unknown information of client education details in the dataset.

Distribution of Education details of clients

Does education status of a client plays a role in predicting output variable, y? Lets look about that,



University degree clients subscribe term deposit higher (1510) than others(High school-936 and Professional course - 535). Plot shows Clients who have University Degree who does not subscribe are also higher in number because of the large proportion of university entries in the dataset. Illiterate people have lower term deposit subscription compared to others.

To check if education of the client relates to term deposit subscription, run chi-squared test,

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  c_edu
## X-squared = 187.63, df = NA, p-value = 0.0004998
```

One can see p-value is less than 0.05 above for education chi-squared test with y. It is clear that education feature plays a role in subscribing term deposit subscription because education feature is dependent to deposit subscription in the marketing campaign.

Does client have a credit default?

Number of unique credit default status of client are:

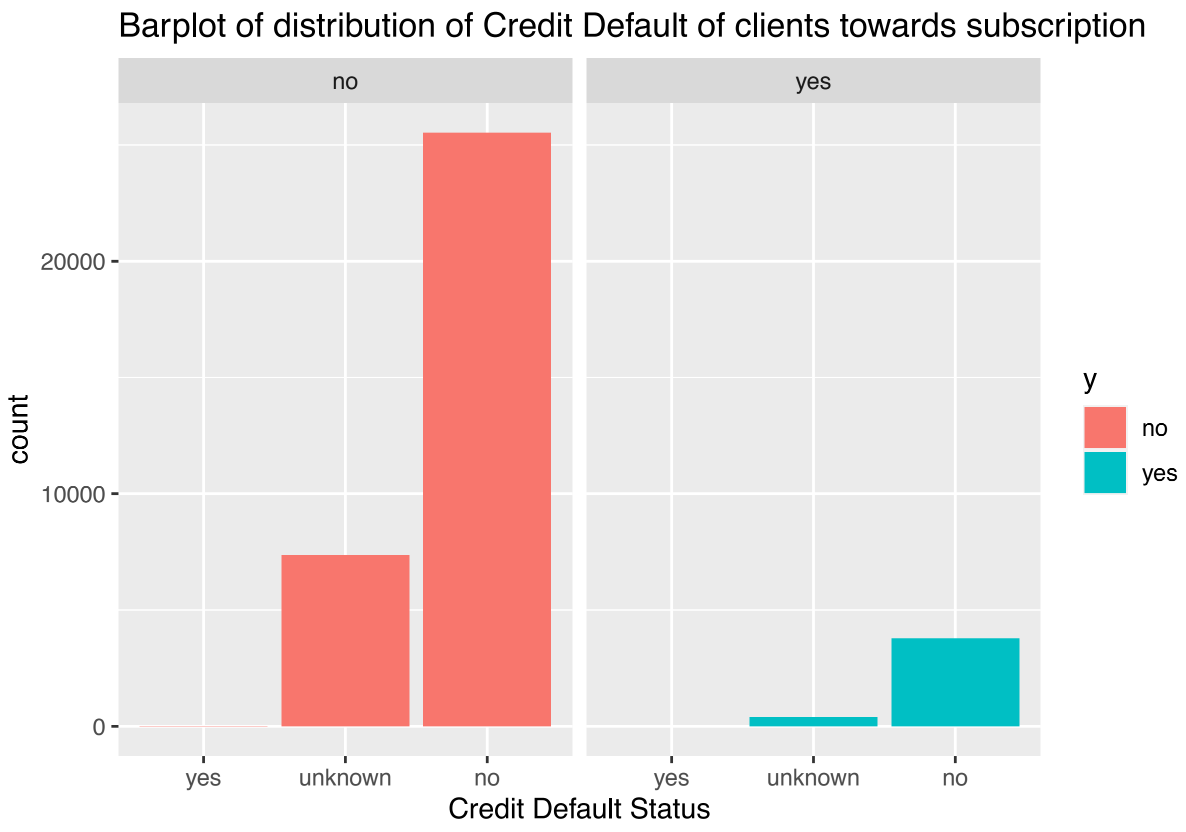
```
table(bank_data$default)
```

```
##
##      no unknown     yes
##   29311    7757       3
```

It have three default status as yes,no and unknown. *yes* is for client who has credit default, *no* is for clients who does not have credit default and *unknown* for no information about default status for that client. Table shows how many clients are in the same category of default status in the marketing campaign. Credit default is the term used when clients fail to repay loan or debt. We don't know credit default status for 7757 clients in the campaign.

How does credit default affects deposit subscription?

Does credit default status of a client plays a role in predicting output variable, y? Lets look about that,



From the plot, it is clear, client with credit default does not subscribe term deposit much. They are lesser (3) in count compared to others. This may be due to the fact, that they already have debt to pay. So, they can't make fixed term investment throughout the term correctly with some exceptions. Clients who does not have credit default avails term deposit higher compared to others. This may be because they don't have debt to repay. So, they can make fixed term investment following throughout the period.

Chi-squared test is run below to check whether default is an important variable in deciding term deposit subscription.

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  c_def
## X-squared = 374.42, df = NA, p-value = 0.0004998
```

Chi-squared test's statistical significant result shows default is an important feature in deciding term deposit subscription(y) preference. Because p-value is 0.0004998 which is less than 0.05. So, default status of client is dependent to output variable,y.

Does client take any housing loans in this or any banks?

Different types of housing loan status of the client are:

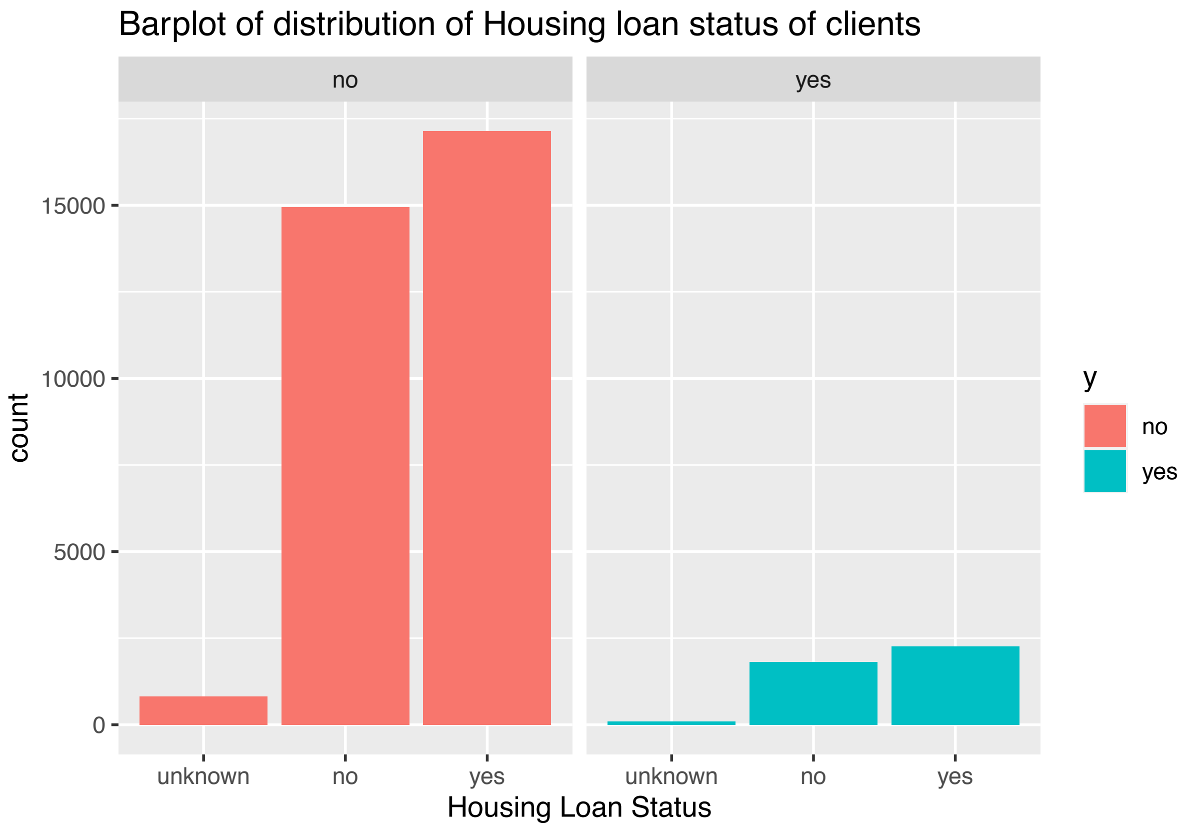
```
unique(bank_data$housing)
```

```
## [1] no      yes     unknown
## Levels: no unknown yes
```

It have 3 status of housing loans in the dataset- *"yes"* for taking house loan, *"no"* for no house loan and *"unknown"* for no known information about housing loan for that client. There are 909 unknown entries for housing loan details in this campaign.

How does housing loan affects term deposit subscription?

Does housing loan status of a client plays a role in predicting output variable, y? Lets look about that,



It appears people who take housing loans subscribes term deposit more compared to others followed by people who don't take hosing loans. Plot shows clients with housing loans who does not subscribe term deposits are also higher in number. Clients with unknown information of housing loans are the least in number(809 for y="no" and 100 for y="yes") in the bank marketing dataset.

Chi-squared test is run below to check whether housing loan is an important variable in deciding term deposit subscription.

```
##
##  Pearson's Chi-squared test
##
## data:  c_house
## X-squared = 5.7073, df = 2, p-value = 0.05763
```

Chi-squared test's statistical result shows housing loan feature is not significant to y variable. That is, housing loan status is not an important variable in deciding term deposit subscription preference because p-value is greater than 0.05. Here p-value is 0.05763. Lets analyze this feature in the model results and check whether chi-squared test result is true.

Does client take any personal loans?

What are the different status of personal loan for client?

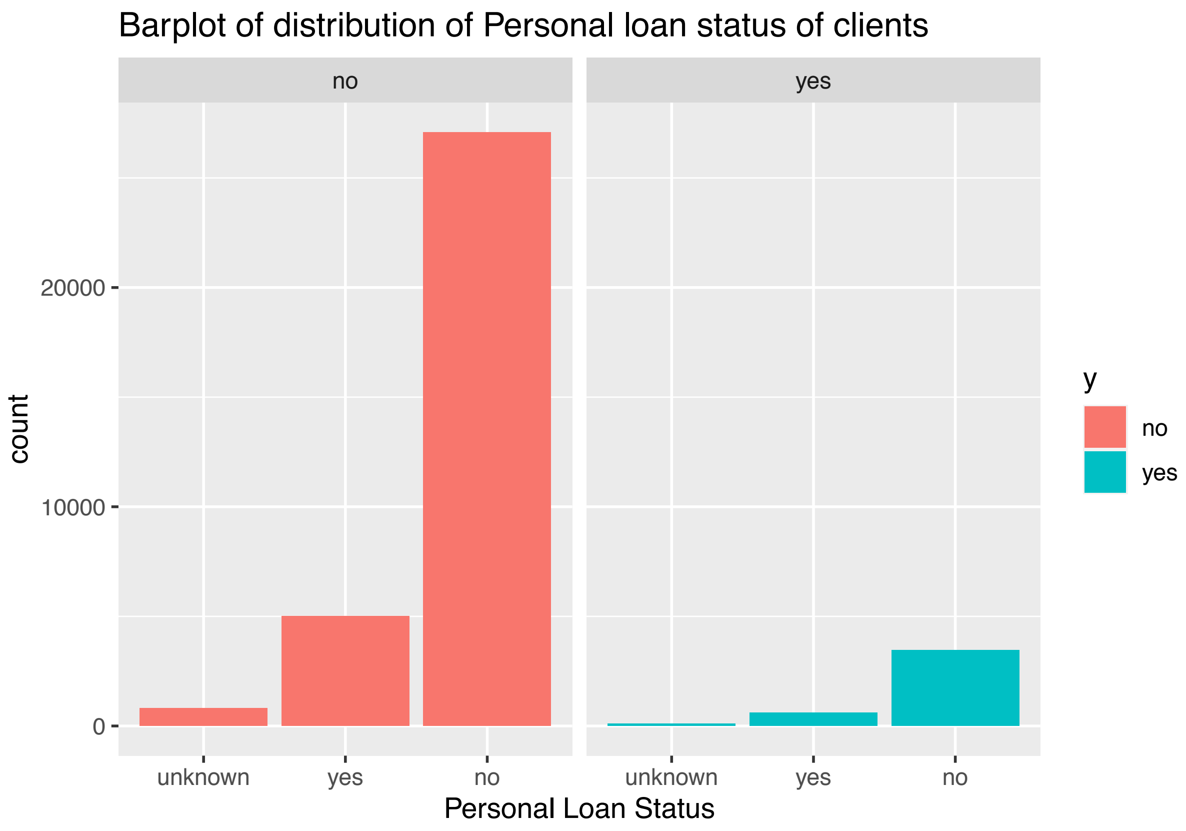
```
table(bank_data$loan)
```

```
##
##      no unknown     yes
##   30544     909    5618
```

It shows three status as yes, no and unknown for personal loan status for the client. *"yes"* for taking personal loan, *"no"* for no personal loan and *"unknown"* for no known information about personal loan for that client. Table lists how many clients belongs to the same criteria of personal loan details in the marketing campaign.

Distribution of personal loan details of the client

Does personal loan status of a client plays a role in predicting output variable, y? Lets look about that,



From the plot, it appears that person who does not take personal loans avails term deposit in large number followed by clients who take personal loans. This is because, people who take personal loan have enough money to pay the loan. So, they can't make fixed term investment in others. Clients with unknown details about personal loan are the ones who subscribe term deposit in low number in the campaign.

Chi-squared test is run below:

```
##
##  Pearson's Chi-squared test
##
## data:  c_loan
## X-squared = 0.84471, df = 2, p-value = 0.6555
```

Since p-value is 0.6555(greater than 0.05), personal loan feature does not decide subscription preference of clients. Lets analyze this in our model results section. From the above exploration of two chi-squared test results above, it is clear people who take loans(housing loan or personal loan), does not contribute to term deposit subscription as both p-value is greater than 0.05.

Contact details of client

What are the different ways bankers make contact with client for the campaign?

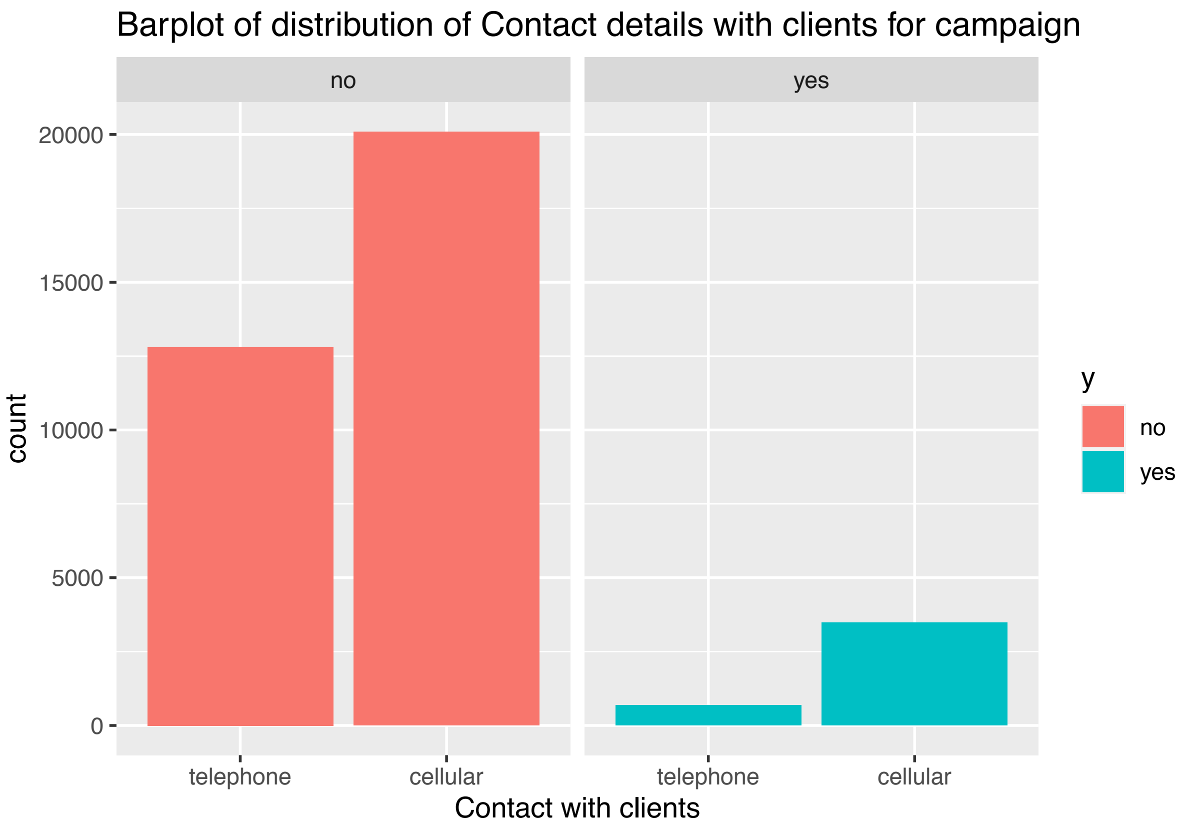
```
table(bank_data$contact)
```

```
##
##   cellular telephone
##      23578      13493
```

By two ways - telephone and cellular. Table shows number of clients who have been contacted by cellular and telephone

Distribution of contact status with client

Does contact status of a client plays a role in predicting output variable, y? Lets look about that,



Clients who have been contacted by cellular way subscribes term deposit higher in number. This is same with cellular contact who does not subscribe term deposit due to its large proportion of number. Clients contacted by telephone and subscribes term deposit are the least in the marketing campaign.

To check if contact attribute contributes to term deposit subscription, lets run chi-squared test and see,

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  c_contact
## X-squared = 800.11, df = 1, p-value < 2.2e-16
```

From the chi-squared test with Yates' continuity correction, it is clear that contact variable is an important factor in deciding term deposit subscription as p-value < 0.05. i.e, Contact attribute is statistically significant to term deposit subscription.So,we should consider this feature while predicting the model.

Exploration of attribute month

What are the different month bankers make contact with client for the campaign?

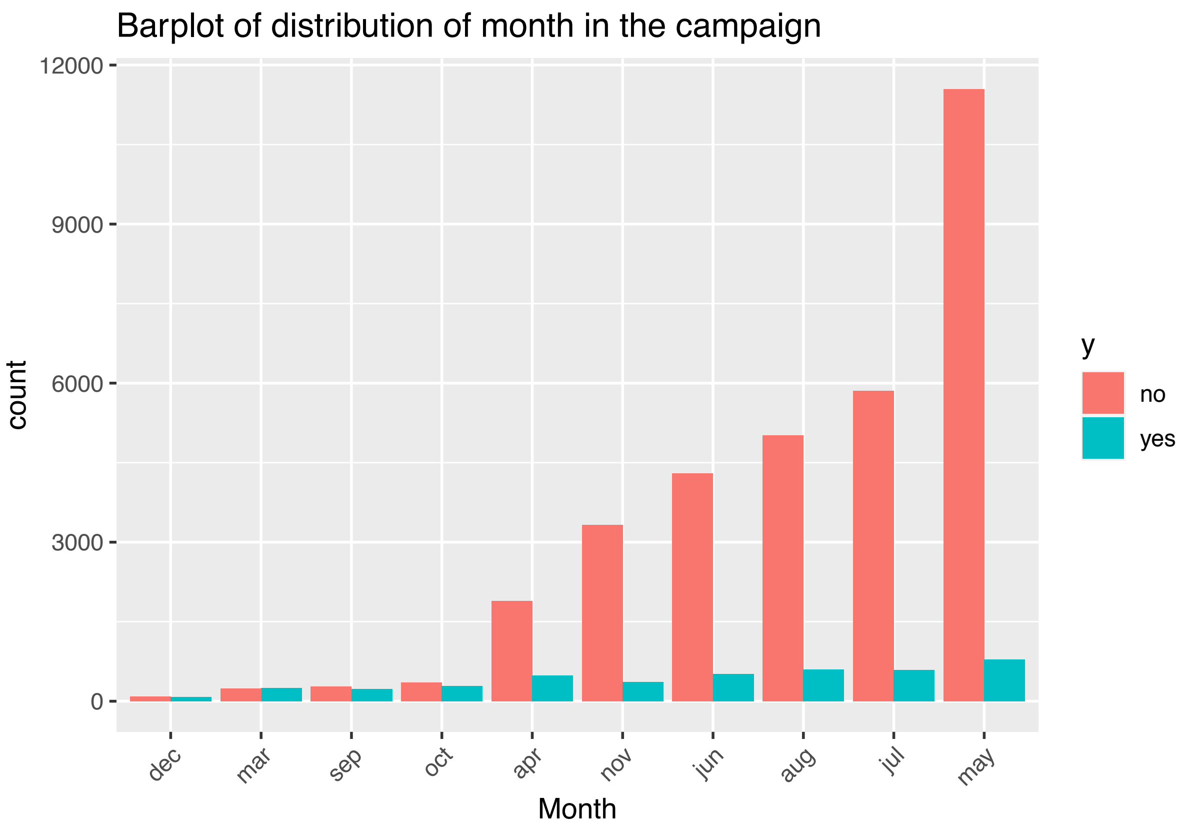
```
table(bank_data$month)
```

```
##
##   apr   aug   dec   jul   jun   mar   may   nov   oct   sep
##  2374  5611   164  6445  4806   485 12336  3690   644   516
```

Table shows how many clients has been contacted under same month in the campaign.

Which month client subscribes term deposit much?

Does attribute month is an important factor to be considered while predicting term deposit subscription? Lets see about that.



Plot shows in month may, clients subscribe term deposit more compared to other months in the campaign followed by august and july month. From this plot, we get to know which month client is subscribing more and which month not. This helps us to target customers during the marketing campaign. We can increase the number of calls and clients in month may and august and we can make sure there are enough opening accounts for term deposit subscription. December month is the one client subscribes term deposit in low number in the campaign.

Chi-squared test is run below,

```
##
##  Pearson's Chi-squared test
##
## data:  c_mon
## X-squared = 2848.1, df = 9, p-value < 2.2e-16
```

From the chi-squared test, it is clear that month variable is an important factor in deciding term deposit subscription as p-value < 0.05. Month attribute is statistically significant (dependent) to term deposit subscription. So, we can consider month attribute while predicting the model.

Exploration of attribute week

What are the days bankers make contact with client for the campaign?

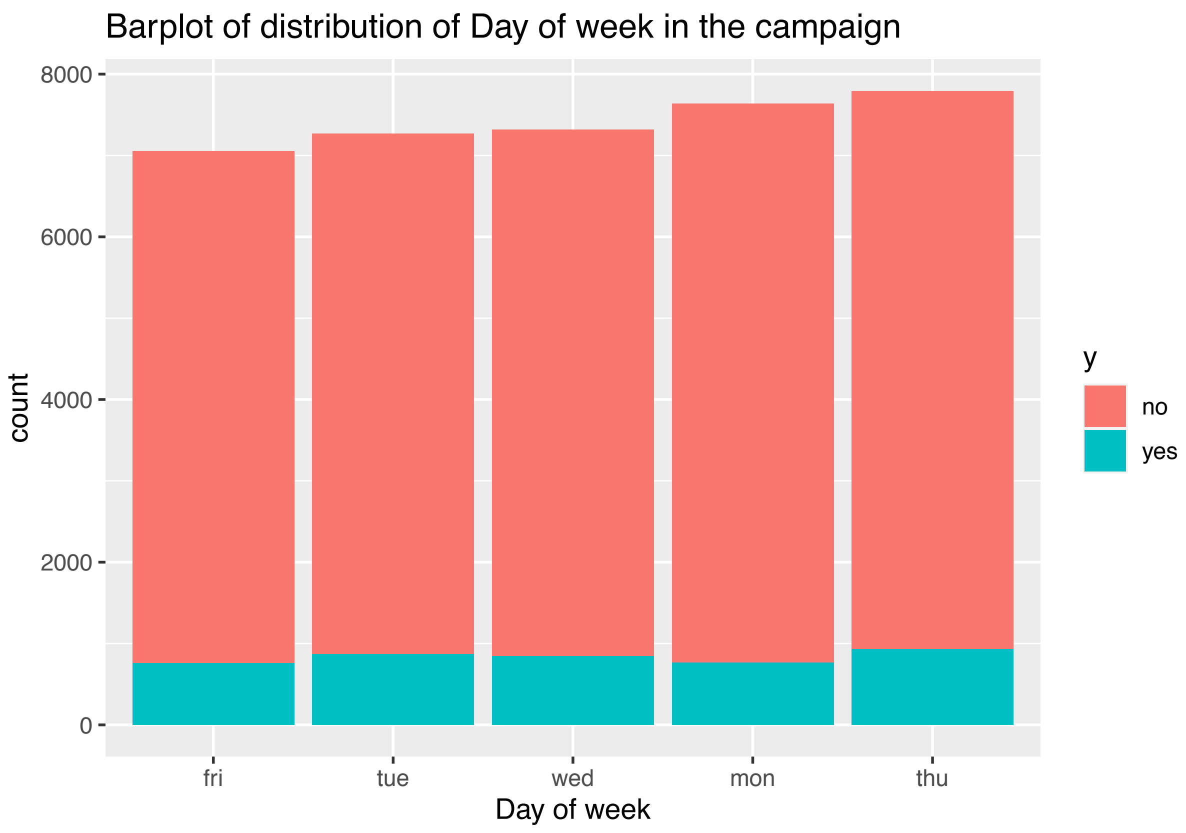
```
table(bank_data$day_of_week)
```

```
##
##  fri  mon  thu  tue  wed
## 7052 7639 7793 7271 7316
```

Table shows how many clients have been contacted in the same day in the campaign.

Which day of week client subscribes term deposit much?

Does attribute day_of_week is an important factor to be considered while predicting term deposit subscription? Lets see about that.



Plot shows Thursday is the day of week clients subscribe term deposit more compared to other days followed by Monday and Wednesday. There is not much difference in the days where client subscribes term deposit. The total numbers for day of week are closer to each other. Friday is the day client subscribes term deposit in lesser number in the campaign.

Chi-squared test is run below to check whether day of week is an important variable in deciding term deposit subscription.

```r
c_day <- table(bank_data$day_of_week,bank_data$y)
chisq_day <- chisq.test(c_day)
chisq_day
```

```
##
##  Pearson's Chi-squared test
##
## data:  c_day
## X-squared = 21.208, df = 4, p-value = 0.000288
```

Chi-squared test tells p-value as 0.000288 which is less than 0.05. So, day of week variable affects term deposit subscription preference in the campaign.

Exploration of duration of last contact

How many different duration of time last contact is made for clients in the campaign?

```r
length(unique(bank_data$duration))
```

```
## [1] 1513
```

```r
min(bank_data$duration)
```

```
## [1] 0
```

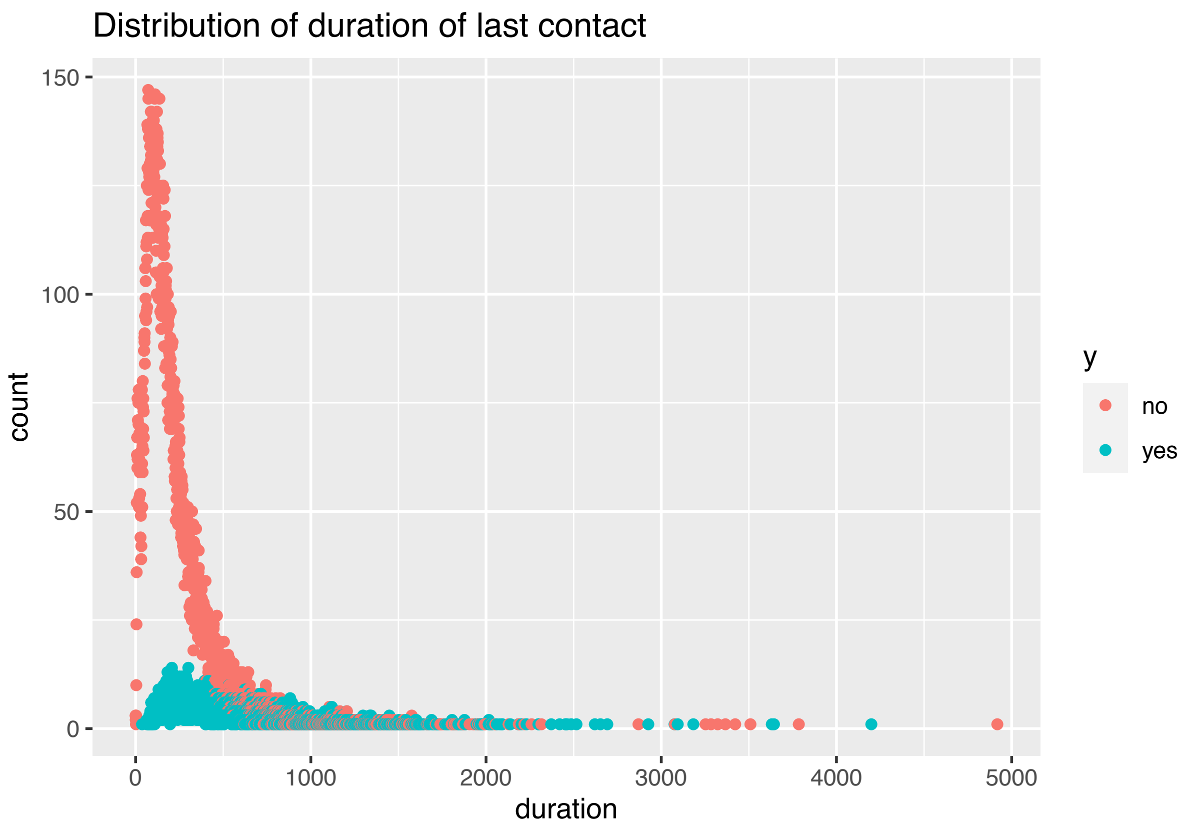
```r
max(bank_data$duration)
```

```
## [1] 4918
```

There are 1513 unique duration of phone calls during the campaign. Minimum duration of phone call is 0 whereas maximum duration of contact is 4918. If duration is 0, then there is no term deposit subscription as clients does not attend the call.

Distribution of duration of last contact

Does attribute duration is an important factor to be considered while predicting term deposit subscription? Lets see about that.



Plot shows duration of over 200 to 750 subscribes term deposit more compared to others. It is also clear longer duration of phone call guarantees term deposit subscription with some exceptions. Almost all people who does not subscribe term deposit decide within first 4 minutes and people who wish to subscribe sometimes take little longer in getting convinced and deciding.

Chi-squared test is run below,

```
c_dur <- table(bank_data$duration,bank_data$y)
chisq_dur <- chisq.test(c_dur,simulate.p.value = TRUE)
chisq_dur
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  c_dur
## X-squared = 9604.8, df = NA, p-value = 0.0004998
```

From the chi-squared test, it is clear that duration variable is an important factor in deciding term deposit subscription as p-value < 0.05. Duration attribute is statistically significant to term deposit subscription.

Exploration of previous outcome of the campaign

What are the different results of the last contact marketing campaign?

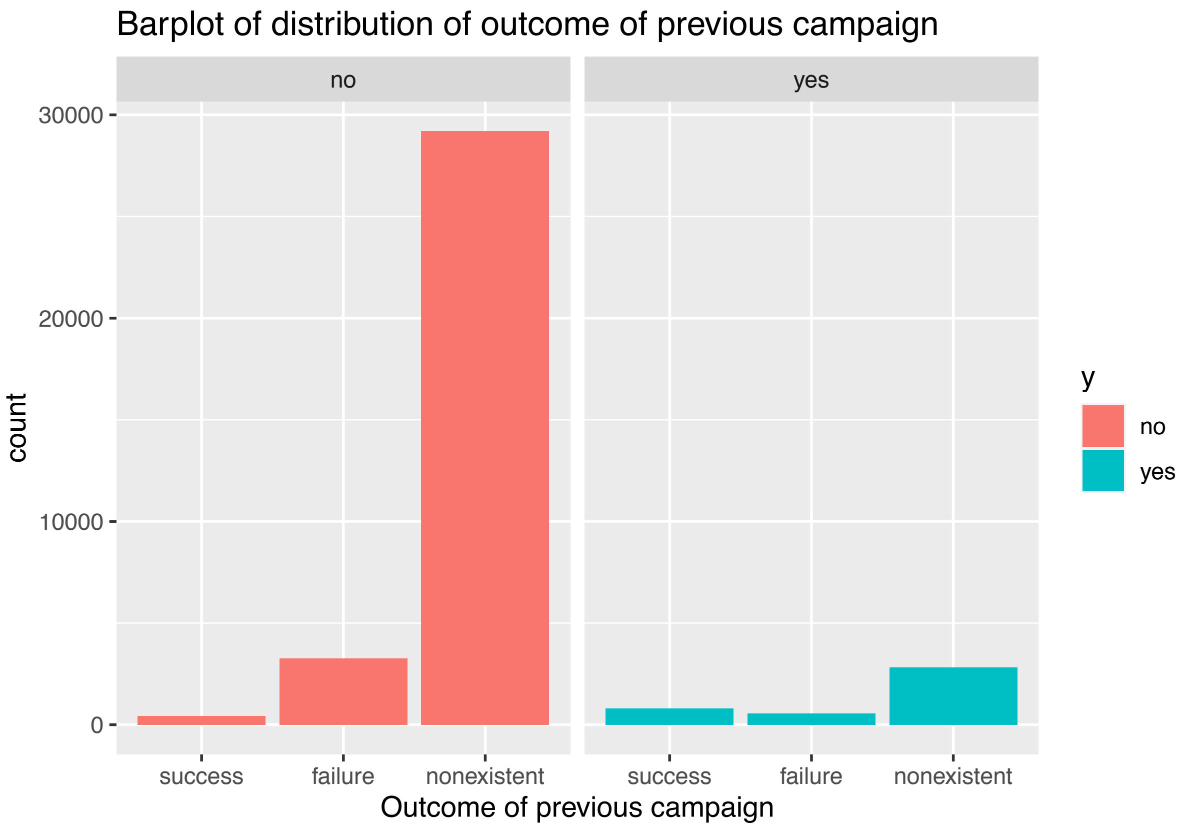
```
table(bank_data$poutcome)
```

```
##
##     failure nonexistent     success
##        3816       32032        1223
```

It has three results - success, failure and nonexistent(no previous contact). *success* for client who subscribed term deposit in the last campaign, *failure* for who doesn't subscribe in the last campaign and *non-existent* for no information about outcome of last campaign. There are 32032 nonexistent entries of client's previous marketing campaign results.

Distribution of outcome of the previous campaign

Does attribute poutcome is an important factor to be considered while predicting term deposit subscription? Lets see about that.



From the plot, it is clear success outcome of the previous campaign is the least in number compared to others because they subscribed term deposit last campaign. So, there is not a need to deposit once again this year. The nonexistent outcome of the previous campaign subscribes term deposit more followed by failure. Failure outcome of previous campaign clients who does not subscribe deposit can be targeted next year to coerce them into subscription.

Chi-squared test is run below,

```
##
##  Pearson's Chi-squared test
##
## data:  c_out
## X-squared = 3774.5, df = 2, p-value < 2.2e-16
```

One can see p-value is less than 0.05 above for previous outcome of campaign with y. It is clear that previous outcome feature plays a role in subscribing term deposit subscription.

Exploration of number of contacts performed during this campaign

What are the different number of contacts performed during this marketing campaign?

```
length(unique(bank_data$campaign))
```

```
## [1] 42
```

```
min(bank_data$campaign)
```

```
## [1] 1
```

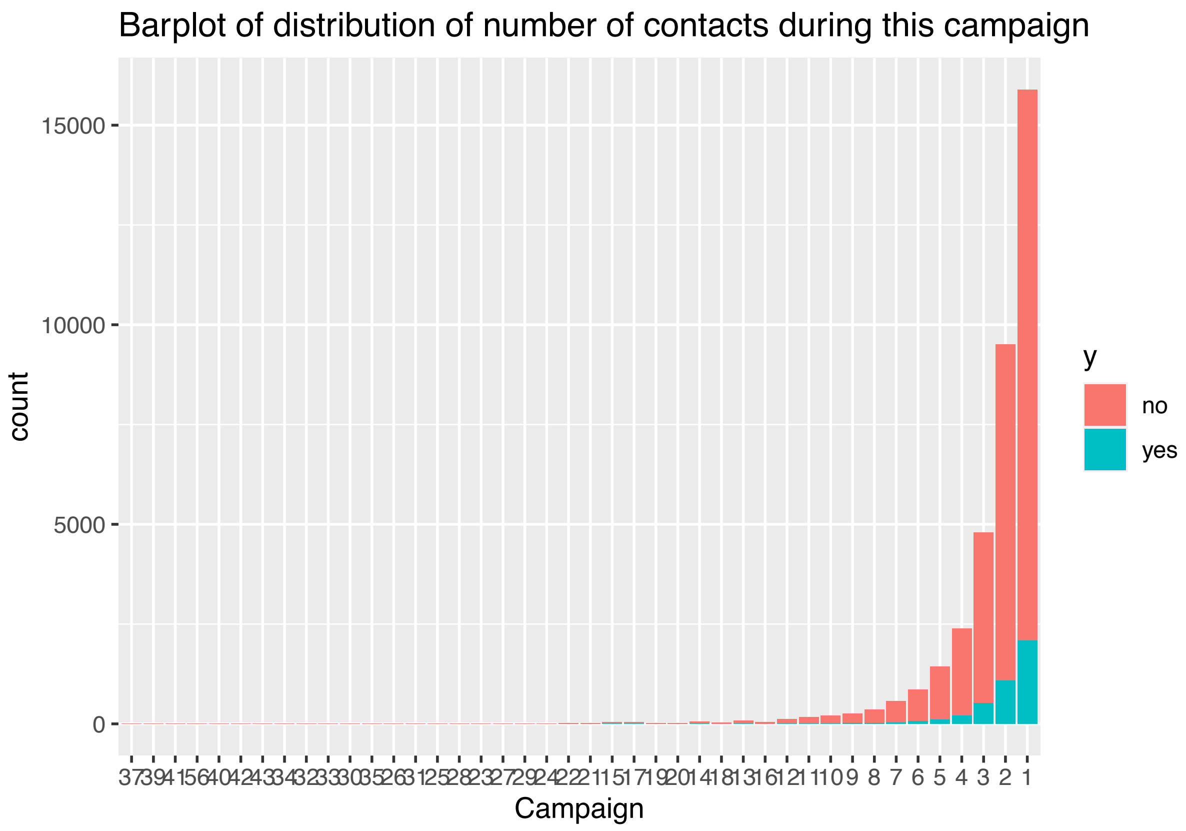
```
max(bank_data$campaign)
```

```
## [1] 56
```

42 different number of contacts were performed during the campaign. Minimum number of contact performed for the client during the campaign is 1 whereas maximum number of contact for client is 56.

Distribution of contacts during this campaign

Does attribute campaign is an important factor to be considered while predicting term deposit subscription? Lets see about that.



From the plot, it is clear that one contact with the client subscribes term deposit more followed by 2 and 3. This is due to large proportion of number for one contact during the campaign. This plays an important role in deciding term deposit subscription. Chi-squared test is not applicable for numeric categorical factors.

Exploration of number of days after last contact from previous campaign

What are the different number of days that passed by after the client was last contacted from a previous campaign?

```
length(unique(bank_data$pdays))
```

```
## [1] 26
```

```
min(bank_data$pdays)
```

```
## [1] 0
```

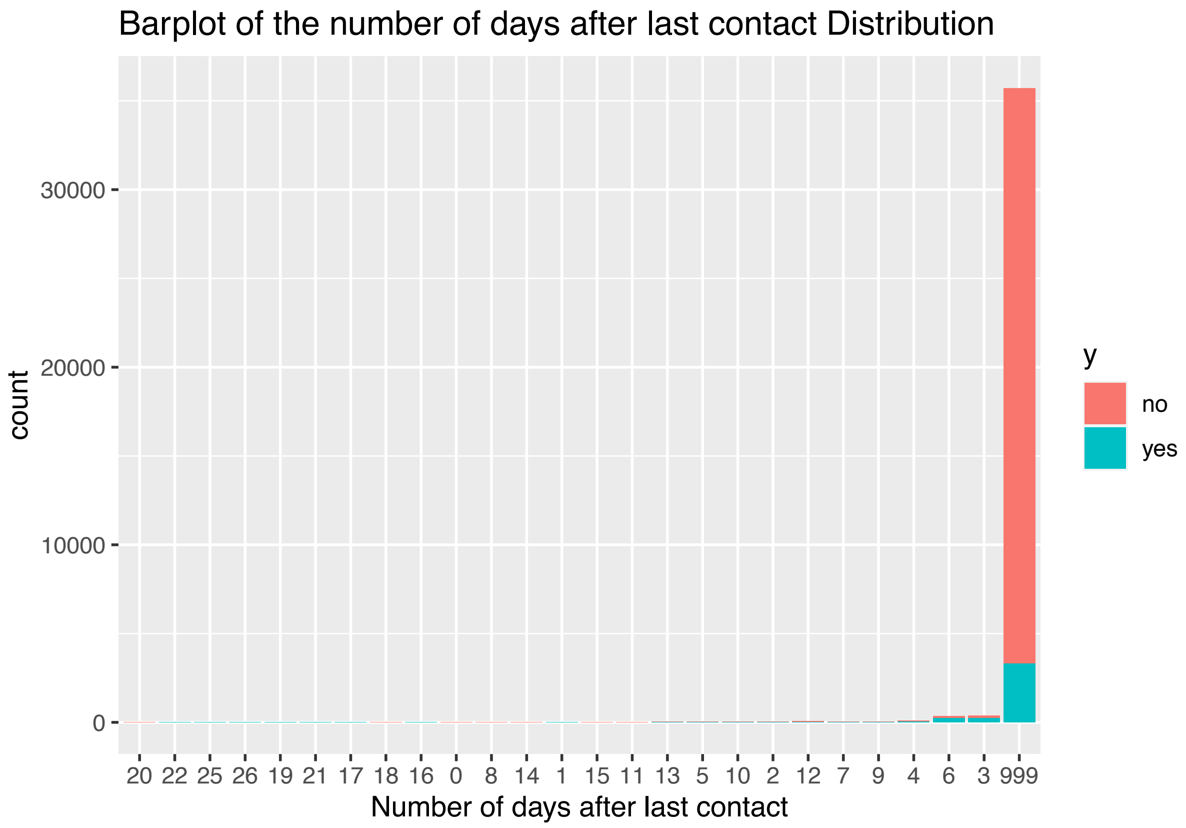
```
max(bank_data$pdays)
```

```
## [1] 999
```

There are 26 different number of days after last contact of the client from previous campaign. Minimum number of days after last contact is 0 whereas maximum number of days are 999. 999 means client was not previously contacted

Distribution of number of days after last contact in the previous campaign

Does attribute pdays is an important factor to be considered while predicting term deposit subscription? Lets see about that.



From the plot,it is clear that client who was not contacted(999) during previous campaign subscribes term deposit more as they did not subscribe last year. This is also due to the fact that 999 has large proportion of number in the campaign. pdays act as an important factor in term deposit subscription as many new clients who have not contacted during last campaign subscribes term deposit during this campaign.

Exploration of number of contacts for previous campaign for this client

What are the different number of contacts performed during previous marketing campaign for this client?

```
length(unique(bank_data$previous))
```

```
## [1] 8
```

```
min(bank_data$previous)
```

```
## [1] 0
```

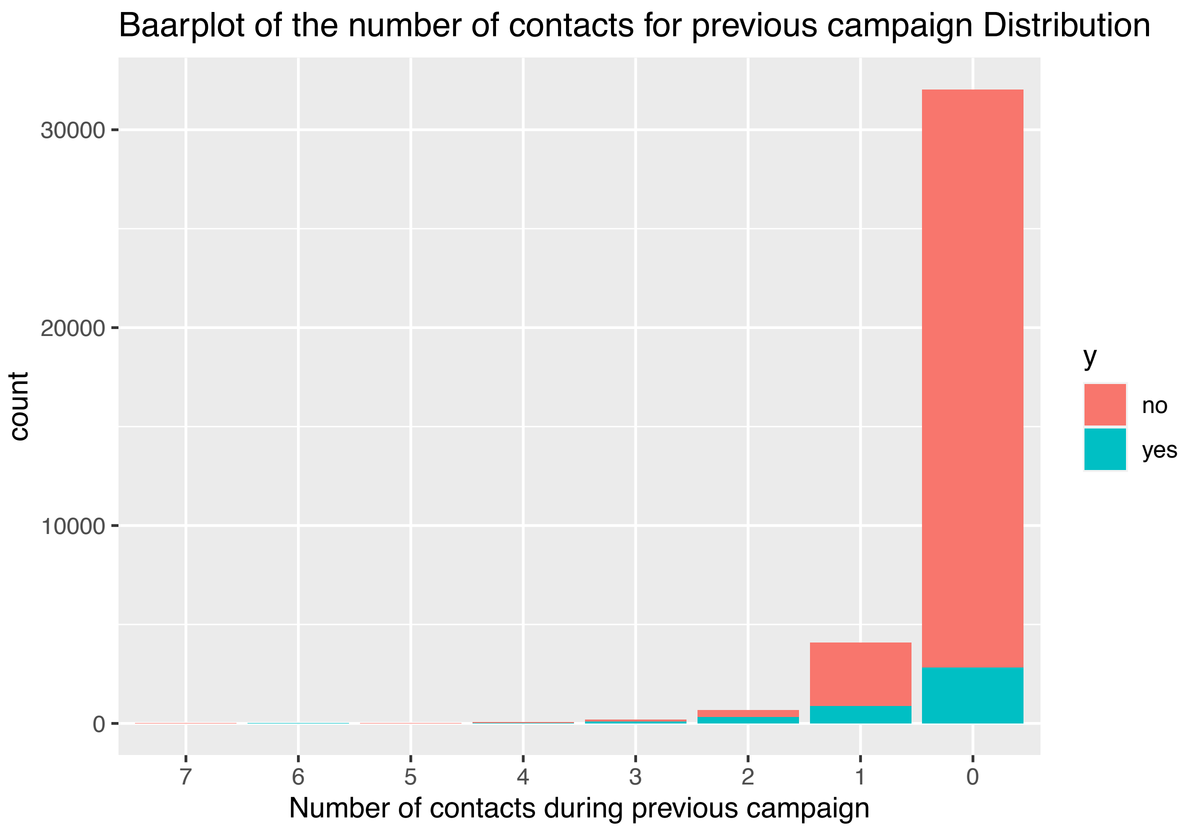
```
max(bank_data$previous)
```

```
## [1] 7
```

There are eight different number of contacts during previous campaign for the client. Minimum number of contact for the client during previous campaign is 0 whereas maximum number of contact for client is 7.

Distribution of number of contacts for previous campaign

Does attribute previous is an important factor to be considered while predicting term deposit subscription? Lets see about that.



From the plot, it is clear that clients who have zero contact during previous campaign subscribes term deposit more followed by one,two etc., Obviously, clients who have not contacted last campaign will be contacted this campaign to coerce them into term deposit subscription as well as due to high proportion of number in them. Also clients who have been contacted last year but did not subscribe last campaign subscribes term deposit this campaign with frequent marketing strategies and awareness. So, previous is an important term for term deposit subscription

Exploration of Employment variation rate

What are the different employment variation rate in the marketing campaign?

```
length(unique(bank_data$emp.var.rate))
```

```
## [1] 10
```

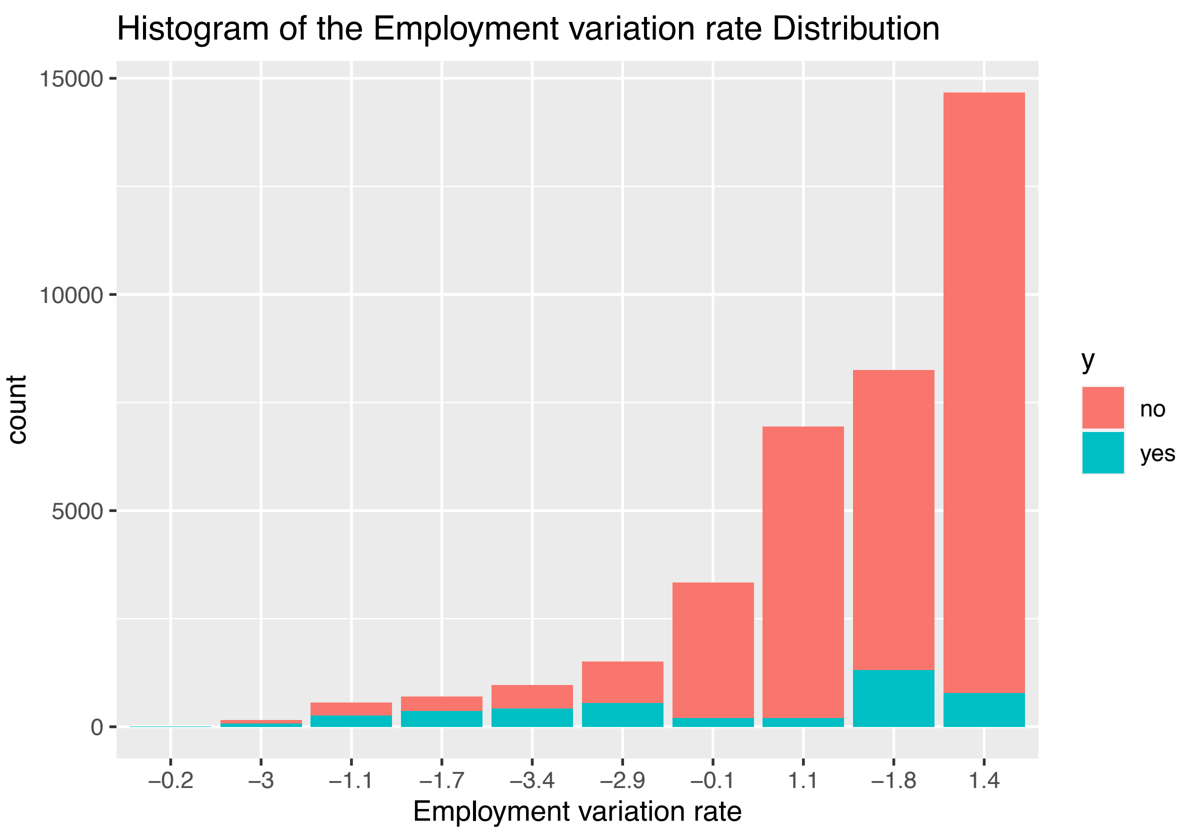
```
table(bank_data$emp.var.rate)
```

```
##
##  -0.1  -0.2  -1.1  -1.7  -1.8  -2.9    -3  -3.4   1.1   1.4
##  3333    10   553   695  8250  1504   154   964  6945 14663
```

There are 10 different types of employment variation rate for clients in the campaign. Table lists how many clients have same employment variation rate in the campaign

Distribution of employment variation rate

Does employment variation rate of the client affects term deposit subscription?



Plot shows employment variation rate of 1.4 subscribes term deposit much followed by -1.8 and 1.1. This figure may be due to large proportion of number. Employment variation rate of -0.2 subscribes term deposit in low number

Exploration of Consumer price index

What are the different consumer price indexes for client in the campaign?

```
length(unique(bank_data$cons.price.idx))
```

```
## [1] 26
```

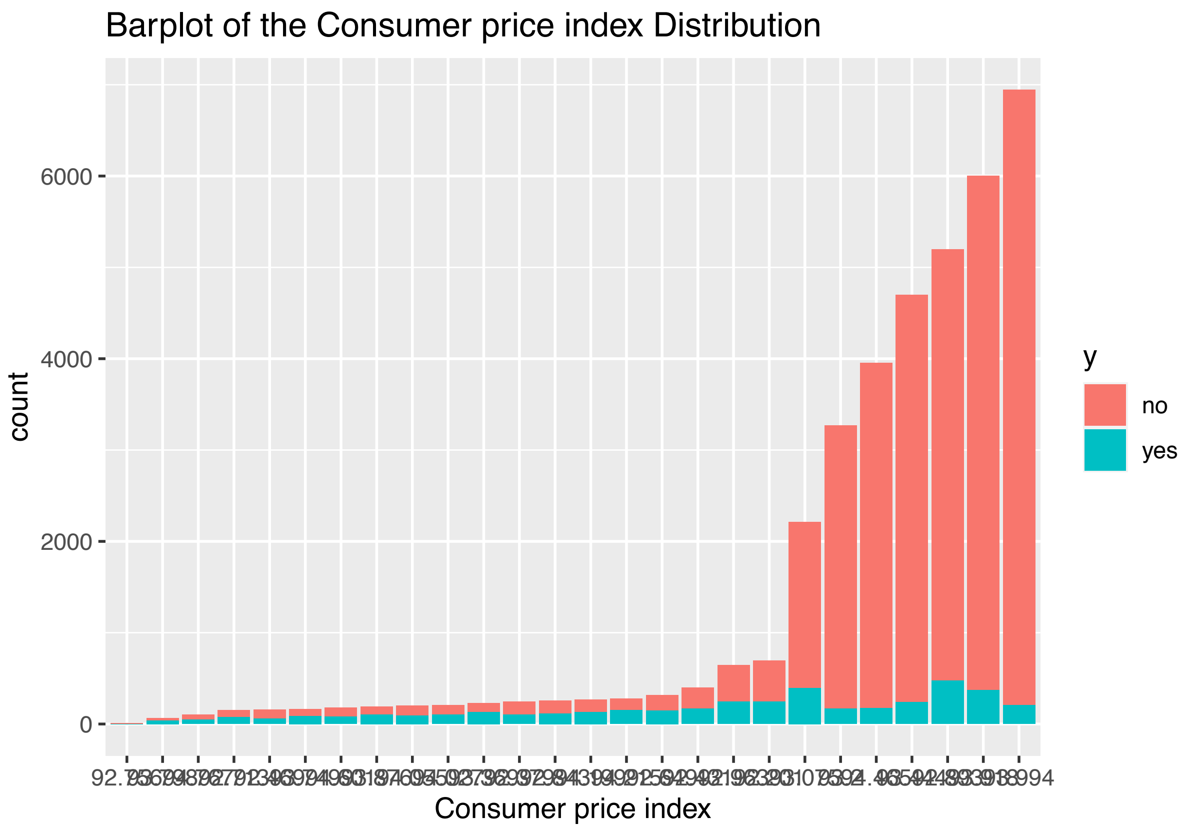
```
table(bank_data$cons.price.idx)
```

```
##
## 92.201 92.379 92.431 92.469 92.649 92.713 92.756 92.843 92.893 92.963 93.075
##    699    247    402    159    315    154     10    255   5198    646   2213
##   93.2 93.369 93.444 93.749 93.798 93.876 93.918 93.994 94.027 94.055 94.199
##   3271    230   4702    161     62    193   6005   6945    210    204    269
## 94.215 94.465 94.601 94.767
##    281   3956    180    104
```

There are 26 different types of consumer price index for clients in the campaign. Table lists how many clients have same consumer price index in the campaign

Distribution of Consumer price index

Does consumer price index affects term deposit subscription?



Plot shows consumer price index of 92.893 subscribes term deposit much followed by 93.095. Consumer price index of 92.756 subscribes term deposit in low number. From the figure one can observe changes in consumer price index increases term deposit subscription. So, consumer price index affects term deposit subscription

Exploration of Consumer Confidence index

What are the different consumer confidence indexes for client in the campaign?

```
length(unique(bank_data$cons.conf.idx))
```

```
## [1] 26
```

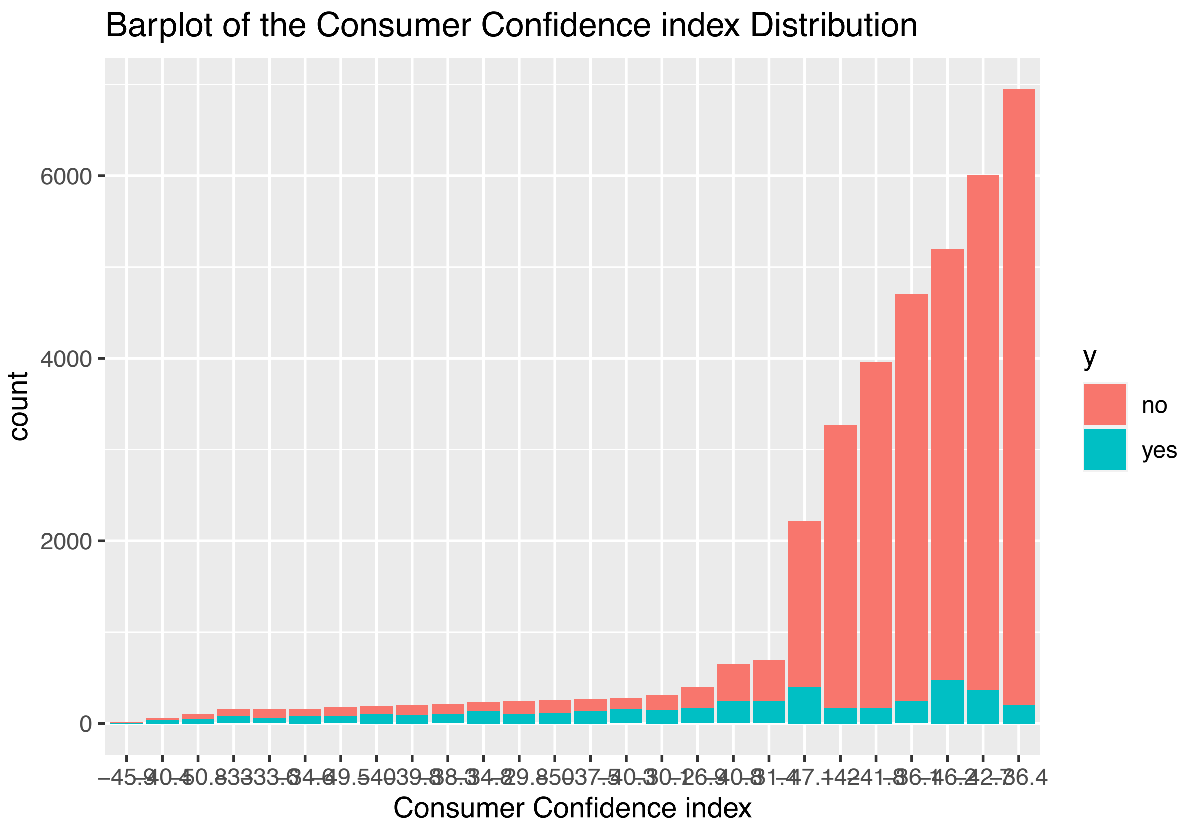
```
table(bank_data$cons.conf.idx)
```

```
##
## -26.9 -29.8 -30.1 -31.4   -33 -33.6 -34.6 -34.8 -36.1 -36.4 -37.5 -38.3 -39.8
##   402   247   315   699   154   159   161   230  4702  6945   269   210   204
##   -40 -40.3 -40.4 -40.8 -41.8   -42 -42.7 -45.9 -46.2 -47.1 -49.5   -50 -50.8
##   193   281    62   646  3956  3271  6005    10  5198  2213   180   255   104
```

There are 26 different types of consumer confidence index for clients in the campaign. Table lists how many clients have same consumer confidence index in the campaign

Distribution of Consumer Confidence index

Does consumer price index affects term deposit subscription?



Plot shows consumer price index of -46.02 subscribes term deposit much followed by -47.1. Consumer price index of -40.4 subscribes term deposit in low number. From the figure one can observe changes in consumer confidence index increases term deposit subscription

Exploration of Euribor 3 month rate

What are the different Euribor 3 month rate for client in the campaign?

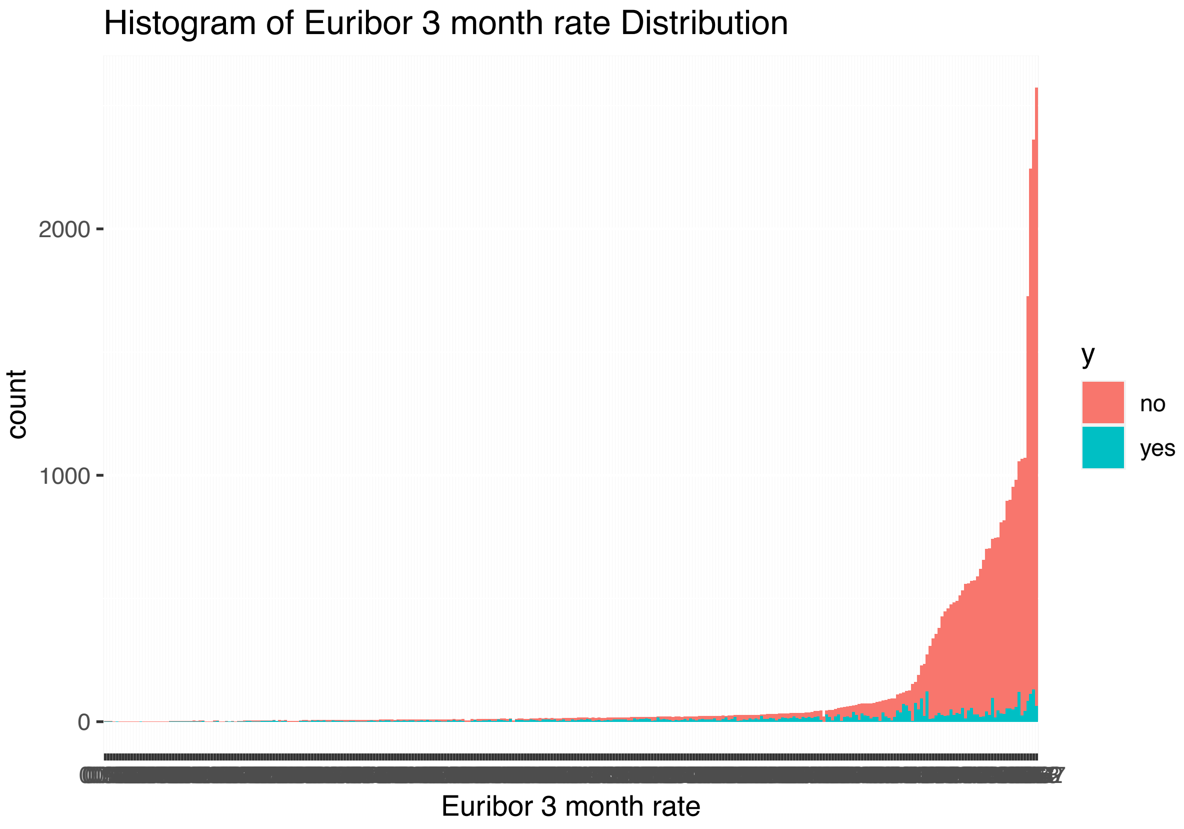
```
length(unique(bank_data$euribor3m))
```

```
## [1] 316
```

There are 316 different types of euribor 3month rate for clients in the campaign.

Distribution of Euribor 3 month rate

Does Euribor 3 month rate affects term deposit subscription?



From the plot, one can observe changes in changes in euribor 3month rate increases term deposit subscription. As well as low euribor rate has low term deposit subscription.

Exploration of number of employees

How many different number of employees are in the campaign?

```
length(unique(bank_data$nr.employed))
```

```
## [1] 11
```

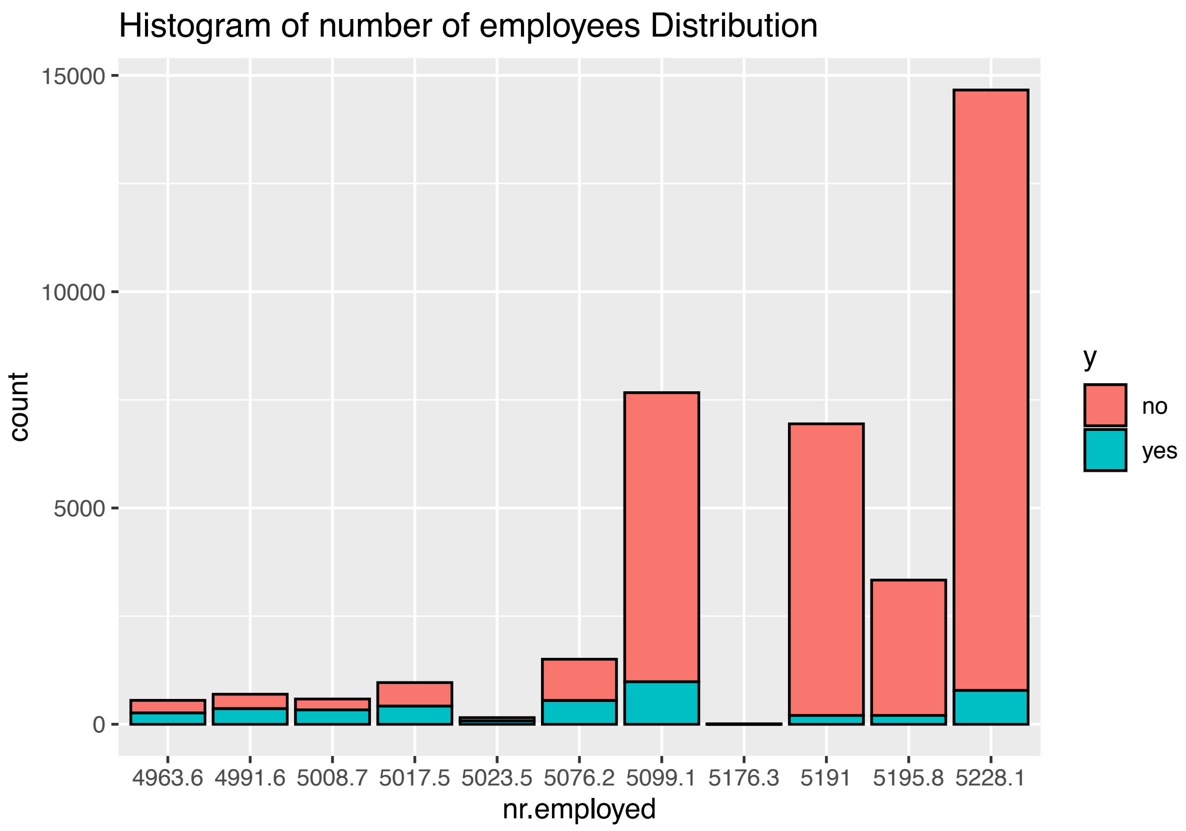
```
table(bank_data$nr.employed)
```

```
##
## 4963.6 4991.6 5008.7 5017.5 5023.5 5076.2 5099.1 5176.3   5191 5195.8 5228.1
##    553    695    584    964    154   1504   7666     10   6945   3333  14663
```

Table shows how many clients belong to the same number of employees in the campaign.

Distribution of number of employees

Does number of employees affects term deposit subscription?



From the plot, it is clear that 5099.1 no. of employees subscribes term deposit higher compared to others.

Model Analysis and Results

To compare different models with their accuracies in predicting term deposit subscription, split bank_data further into training and testing sets as bank_train_data & bank_validation_data. The model which gives highest accuracy is taken and tested against validation set(final holdout test) to determine accuracy and to evaluate model performance.

```
set.seed(1, sample.kind="Rounding")
test_index1 <- createDataPartition(y = bank_data$y, times = 1, p = 0.1, list = FALSE)
bank_train_data <- bank_data[-test_index1,]
bank_temp_data <- bank_data[test_index1,]
# Make sure euribor3m in bank_validation_data set are also in bank_train_data set
bank_validation_data <- bank_temp_data %>%
  semi_join(bank_train_data, by = "euribor3m")
# Bring back removed rows from bank_validation_data to bank_train_data set
removed <- anti_join(bank_temp_data, bank_validation_data)
bank_train_data <- rbind(bank_train_data, removed)
rm( test_index1, bank_temp_data, removed) #Clear out unwanted files
```

General overview of the dataset:

```
# Number of observations(rows and columns) in the dataset
dim(bank_train_data)
```

```
## [1] 33366    21
```

```
dim(bank_validation_data)
```

```
## [1] 3705   21
```

There are 33366 rows and 21 columns in training set. There are 3705 rows and 21 columns in test set.

Depending on the data exploration and visualization of the bank marketing data, we get to know which attribute reflects on term deposit subscription. By considering the attributes which contribute to subscription, I built a system made of classification and regression models. The former one is done with the help of *rminer* package while the latter is done by caret package. I have constructed 14 rminer classification models and one base caret regression model. Classification models are Naive Bayes model, KNN(K- Nearest Neighbor) model, Conditional inference tree model, eXtreme Gradient Boosting model, Support Vector Machine, Least Squares Support Vector Machine, Random Forest method, Linear Discriminant Analysis, Generalised Linear model, Multilayer perceptron, Logistic Regression(multinom), Bagging, Adaboost, Decision tree. Regression model is Logistic regression from caret package.

The main objective of the project is to predict term deposit subscription. This is done by implementing every model and analyzing the obtained results and choosing the best one which gives the highest accuracy in predicting deposit subscription.

Evaluation metrics:

After building the model and predicting term deposit subscription, one of the final steps is evaluating the model performance. The evaluation metrics are Confusion matrix, Accuracy, Sensitivity, Specificity, AUC, F1 score etc.,

Confusion Matrix : It is an N X N matrix, where N is the number of classes to be predicted. In our project, target variable(y) is a binary class.i.e, N=2, and hence we get a 2 X 2 matrix. It is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. It is used to visualize important predictive analytics like recall, specificity, accuracy, and precision. Confusion matrices are useful because they give direct comparisons of values like True Positives, False Positives, True Negatives and False Negatives.

Accuracy : the proportion of the total number of predictions that were correct. The problem with using accuracy as your main performance metric is that it does not do well when you have a severe class imbalance.

Positive Predictive Value or Precision : indicates the rate at which positive predictions are correct. Precision tells us how many of the correctly predicted cases actually turned out to be positive. This would determine whether our model is reliable or not.

Sensitivity or Recall or true positive rate : the proportion of actual positive cases which are correctly identified. How many correct items are predicted?

Specificity or True Negative Rate : the proportion of actual negative cases which are correctly identified.

Ideally, a test should provide a high sensitivity and specificity. A highly sensitive test means that there are few false negative results, and thus fewer cases of disease are missed. A highly specific test means that there are few false positive results.

F1-score : F1-Score is the harmonic mean of precision and recall values for a classification problem. A good F1 score means that you have low false positives and low false negatives, so you're correctly identifying real threats and you are not disturbed by false alarms. An F1 score is considered perfect when it is 1, while the model is a total failure when it is 0.

ROC curve : The Receiver Operator Characteristic(ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots TPR against FPR at various threshold values and separates signal from the noise.

AUC: It is an area below the ROC curve. It is the measure of ability of a classifier to distinguish between classes. Higher the AUC, better the performance of the model.

In this section, each algorithm model with their effects are evaluated for model performance:

Regression model

Regression in machine learning consists of atatistical and mathematical methods that allow data scientists to predict a continuous outcome (y) based on the value of one or more predictor variables (x). Linear regression is probably the most popular form of regression analysis because of its ease-of-use in predicting the model. It is a supervised technique.It is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values. In this project, I use base logistic regression to predict term deposit subscription by selecting appropriate attributes that influence the model.

Logistic Regression

Logistic regression is another supervised learning algorithm which is used to solve the classification problems. Logistic regression algorithm works with the categorical variable such as 0 or 1, Yes or No, True or False, Spam or not spam, etc. It is a predictive analysis algorithm which works on the concept of probability. To avoid false predictions, the variance should be low. For that reason, the model should be generalized to accept unseen features of temperature data and produce better predictions. For a model to be ideal, it's expected to have low variance, low bias and low error.

To improve the model, features of the model should be evaluated and check which attributes affect the model thereby removing the unnecessary attributes which is withhelding the improvement of the model.

Data Transformation

Since Logistic regression takes numeric categorical values, transform the target variable y(term deposit subscription) into numeric from factor. The levels of y should be 0 for no and 1 for yes. It is done by using base as.numeric() function.

```
bank_train_data <- bank_train_data %>% mutate(y = as.numeric(bank_train_data$y=="yes"))
bank_validation_data <- bank_validation_data %>%
  mutate(y = as.numeric(bank_validation_data$y=="yes"))
str(bank_train_data$y)
```

```
##  num [1:33366] 0 0 0 0 0 0 0 0 0 0 ...
```

```
str(bank_validation_data$y)
```

```
##  num [1:3705] 0 0 0 0 0 0 0 0 0 0 ...
```

As one can see, target variable y has been converted into numeric for making prediction for regerssion method.

As we already know, duration attribute cannot be used since it is not known before a call is performed. This makes difficult to have a real predictive model. So, duration variable is discarded from the dataset.

```
bank_glm <- glm(y ~. -duration, data = bank_train_data, family = "binomial")
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

To remove warnings from the above model, we should remove NAs from dataset. Consumer confidence index and Consumer price index attributes have NAs. So, remove those attributes and fit the model once again and see whether the model gives warning.

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.9811378            0.3086124            0.9177575
##       Neg Pred Value            Precision               Recall
##            0.6753927            0.9177575            0.9811378
##                   F1           Prevalence       Detection Rate
##            0.9483899            0.8871795            0.8704453
## Detection Prevalence    Balanced Accuracy
##            0.9484480            0.6448751
```

Model	Accuracy
Base Logistic Regression model	0.9052632

It produces accuracy of about 0.9053. This model has high sensitivity value(0.9811378). 98% sensitivity will identify 98% of clients who subscribes term deposit subscription. F1 score is 0.9483899. F1 score > 0.90 is considered as a good model. 0.9177575, Precision denotes that this model is a reliable model.

The step function removes all features which are not statistically dependent to target variable. i.e, it selects only features which contributes to prediction of term deposit subscription. After running step function, 17 attributes are further reduced to 10 attributes which primarily contributes in predicting term deposit subscription. These attributes are then modeled once again to get better prediction. The above variables tends to subscribe bank's term deposit.

```
## Start:  AIC=18287.34
## y ~ (age + job + marital + education + default + housing + loan +
##     contact + month + day_of_week + duration + campaign + pdays +
##     previous + poutcome + emp.var.rate + cons.price.idx + cons.conf.idx +
##     euribor3m + nr.employed) - duration - cons.price.idx - cons.conf.idx
##
##
## Step:  AIC=18287.34
## y ~ age + job + marital + education + default + housing + loan +
##     contact + month + day_of_week + campaign + pdays + previous +
##     poutcome + emp.var.rate + euribor3m
##
##                 Df Deviance   AIC
## - education      7    17562 18278
## - job           11    17572 18280
## - marital        3    17559 18283
## - loan           1    17558 18286
## - age            1    17558 18286
## - housing        1    17558 18286
## <none>                17557 18287
## - default        2    17562 18288
## - previous       1    17561 18289
## - day_of_week    4    17579 18301
## - month          5    17581 18301
## - poutcome       2    17576 18302
## - pdays          1    17577 18305
## - campaign       1    17577 18305
## - emp.var.rate   4    17596 18318
## - contact        1    17656 18384
## - euribor3m    312    18554 18660
##
## Step:  AIC=18278.13
## y ~ age + job + marital + default + housing + loan + contact +
##     month + day_of_week + campaign + pdays + previous + poutcome +
##     emp.var.rate + euribor3m
##
##                Df Deviance   AIC
## - job          11    17578 18272
## - marital       3    17564 18274
## - loan          1    17562 18276
## - age           1    17563 18277
## - housing       1    17563 18277
## <none>               17562 18278
## - default       2    17567 18279
## - previous      1    17566 18280
## - month         5    17586 18292
## - day_of_week   4    17584 18292
## - poutcome      2    17580 18292
## - pdays         1    17582 18296
## - campaign      1    17582 18296
## - emp.var.rate  4    17601 18309
## - contact       1    17660 18374
## - euribor3m   312    18561 18653
##
## Step:  AIC=18272.01
## y ~ age + marital + default + housing + loan + contact + month +
##     day_of_week + campaign + pdays + previous + poutcome + emp.var.rate +
##     euribor3m
##
##                Df Deviance   AIC
## - marital       3    17580 18268
## - age           1    17578 18270
## - loan          1    17578 18270
## - housing       1    17579 18271
## <none>               17578 18272
## - previous      1    17582 18274
## - default       2    17585 18275
## - poutcome      2    17596 18286
## - month         5    17602 18286
## - day_of_week   4    17601 18287
## - campaign      1    17597 18289
## - pdays         1    17598 18290
## - emp.var.rate  4    17617 18303
## - contact       1    17676 18368
## - euribor3m   312    18606 18676
##
## Step:  AIC=18268.23
## y ~ age + default + housing + loan + contact + month + day_of_week +
##     campaign + pdays + previous + poutcome + emp.var.rate + euribor3m
##
##                Df Deviance   AIC
## - age           1    17580 18266
## - loan          1    17580 18266
## - housing       1    17581 18267
## <none>               17580 18268
## - previous      1    17584 18270
## - default       2    17587 18271
## - poutcome      2    17598 18282
## - month          5    17604 18282
## - day_of_week    4    17603 18283
## - campaign       1    17599 18285
## - pdays          1    17600 18286
## - emp.var.rate   4    17620 18300
## - contact        1    17678 18364
## - euribor3m    312    18613 18677
##
## Step:  AIC=18266.23
## y ~ default + housing + loan + contact + month + day_of_week +
##     campaign + pdays + previous + poutcome + emp.var.rate + euribor3m
##
##                 Df Deviance   AIC
## - loan           1    17580 18264
## - housing        1    17581 18265
## <none>                17580 18266
## - previous       1    17584 18268
## - default        2    17587 18269
## - poutcome       2    17598 18280
## - month          5    17604 18280
## - day_of_week    4    17603 18281
## - campaign       1    17599 18283
## - pdays          1    17600 18284
## - emp.var.rate   4    17620 18298
## - contact        1    17678 18362
## - euribor3m    312    18616 18678
##
## Step:  AIC=18264.44
## y ~ default + housing + contact + month + day_of_week + campaign +
##     pdays + previous + poutcome + emp.var.rate + euribor3m
##
##                 Df Deviance   AIC
## - housing        2    17581 18261
## <none>                17580 18264
## - previous       1    17584 18266
## - default        2    17588 18268
## - poutcome       2    17599 18279
## - month          5    17605 18279
## - day_of_week    4    17603 18279
## - campaign       1    17600 18282
## - pdays          1    17600 18282
## - emp.var.rate   4    17620 18296
## - contact        1    17679 18361
## - euribor3m    312    18616 18676
##
## Step:  AIC=18261.17
## y ~ default + contact + month + day_of_week + campaign + pdays +
##     previous + poutcome + emp.var.rate + euribor3m
##
##                 Df Deviance   AIC
## <none>                17581 18261
## - previous       1    17585 18263
## - default        2    17588 18264
## - poutcome       2    17599 18275
## - month          5    17606 18276
## - day_of_week    4    17604 18276
## - campaign       1    17600 18278
## - pdays          1    17601 18279
## - emp.var.rate   4    17621 18293
## - contact        1    17679 18357
## - euribor3m    312    18617 18673
```

```
## glm(formula = y ~ default + contact + month + day_of_week + campaign +
##     pdays + previous + poutcome + emp.var.rate + euribor3m, family = binomial,
##     data = bank_train_data)
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.9808336            0.2990431            0.9166904
##       Neg Pred Value            Precision               Recall
##            0.6648936            0.9166904            0.9808336
##                   F1           Prevalence       Detection Rate
##            0.9476778            0.8871795            0.8701754
## Detection Prevalence    Balanced Accuracy
##            0.9492578            0.6399383
```

Model	Accuracy
Base Logistic Regression model	0.9052632
Step Logistic Regression model	0.9039136

The goal of this model is to remove the unnecessary attributes withholding the model performance and to find the required attributes that affects prediction of term deposit subscription.

From the code implementation in R file, The accuracy is 0.9039136. It has high sensitivity value(0.9808336) and high precision value(0.9177575) thereby making a reliable model. Model's F1 score is 0.9476778. Though this model has 90% accuracy and F1 score(0.947), it takes 2 hours for implementation and execution. Hence, this model is computationally time inefficient. The step logistic regression model's accuracy and F1 score is lower compared to Base Logistic Regression model.

Classfication model- Rminer

Classification model is implemented with the help of rminer package. Rminer package can execute classification and regression data mining tasks, including in particular three CRISP-DM stages: data preparation, modeling and evaluation. The particular case of time series forecasting can also be evaluated if needed. The goal is to provide reduced and coherent set of R functions to perform classification and regression. It saves more time because of fast implementation of the model and by executing full data mining tasks with few lines of code.

The data preparation stage of CRISP-DM may include several tasks, such as data selection, cleaning and transformation. Since data selection and cleaning is done in the first phase of the data exploration and visualization section, it is not necessary to repeat the same data extraction and cleaning(scanning NA values) process once again. Now, we move to data transformation. Data transformation is a crucial step for achieving a successful data mining project and it includes several operations, such as attribute and instance selection, assuring data quality and transforming attributes to a required format for model implementation.

Data Transformation

Since, regression model converts output variable into numeric, classification model can't work. Because, classification model is based on factor levels. So, convert numeric to factor data type for target variable(y). It involves two steps: 1. Convert numeric to character and 2. Convert character to factor

```
bank_train_data$y <- ifelse(bank_train_data$y == 1, "yes","no")
str(bank_train_data$y) # character
```

```
##  chr [1:33366] "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" ...
```

```
bank_train_data <- mutate(bank_train_data, y = as.factor(y))
str(bank_train_data$y) # factor
```

```
##  Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
bank_validation_data$y <- ifelse(bank_validation_data$y == 1, "yes","no")
str(bank_validation_data$y) #character
```

```
##  chr [1:3705] "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" ...
```

```
bank_validation_data <- mutate(bank_validation_data, y = as.factor(y))
str(bank_validation_data$y) # factor
```

```
##  Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

Since R cannot handle categorical predictors more than 53 in fit() function for some models, check all the attributes factor levels with the help of str() function. We find attribute euribor3m is a factor with 316 levels of categories. So, convert factor to numeric for variable euribor3m in the training and testing data set.

```
## 'data.frame':    33366 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  : Factor w/ 10 levels "-0.1","-0.2",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ cons.price.idx: Factor w/ 26 levels "92.201","92.379",..: 19 19 19 19 19 19 19 19 19 19 ...
##  $ cons.conf.idx : Factor w/ 26 levels "-26.9","-29.8",..: 10 10 10 10 10 10 10 10 10 10 ...
##  $ euribor3m     : Factor w/ 316 levels "0.634","0.635",..: 288 288 288 288 288 288 288 288 288 288
##  $ nr.employed   : Factor w/ 11 levels "4963.6","4991.6",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  num [1:33366] 4.86 4.86 4.86 4.86 4.86 ...
```

```
##  num [1:3705] 4.86 4.86 4.86 4.86 4.86 ...
```

```
##  num [1:37071] 4.86 4.86 4.86 4.86 4.86 ...
```

From the above results, we can see variable euribor3m of training and validation sets has been converted from factor to numeric for model implementation purposes. Next, we proceed to implementation of every rminer classification model and evaluating their model performance by training and testing the dataset. The optimal model is identified by maximum accuracy of the model.

In rminer package, holdout() function splits the banking data into training and test sets and computes indexes for them to use for fitting and predicting the model for future. The fit() function in rminer is a supervised data mining model. It is a wrapper function that consists of 16 classification and 18 regression methods inside it under the same coherent function structure. Also, it tunes the hyperparameters of the models (e.g., kknn, mlpe and ksvm) and performs some feature selection methods. Since fit() wraps everything around same structure, it is easy to implement the model just by calling the model name thereby saving time for writing code and executing it. The mmetric function computes classification or regression error metrics. The mmetric function predicts all metrics such as confusion matrix, accuracy rate, weighted average ACC score, classification error, Mean Spearman's rho coefficient for ranking, kappa index, accuracy rate per class, true positive rate, mean absolute error etc.,

1. Naive Bayes model

The Naive Bayes model is based on conditional probabilities. It uses Bayes theorem, that calculates probability by counting the frequency of values and combinations of values in the historical data. The fundamental assumption of Naive Bayes is that, given the value of the label (the class), the value of any Attribute is independent of the value of any other Attribute. The independence assumption vastly simplifies the calculations needed to build the Naive Bayes probability model. Naive Bayes can be used for both binary and multiclass classification problems. It supports parallel execution. Thus it is fast to implement the model.

Bayes' theorem finds the probability of an event occurring given the probability of another event that has already occurred. If B represents the dependent event and A represents the previous event, Bayes' theorem can be stated as follows:

$$P(B|A) = P(AandB)/P(A)$$

To calculate the probability of B given A, the algorithm counts the number of cases where A and B occur together and divides it by the number of cases where A occurs alone.

In rminer Naive Bayes model, *bank_data* is split into two parts as testing and training datasets by holdout() function. It splits the data set into two ways, 2/3 for training the model and 1/3 for testing the model. The fit() function accomodates naive bayes model into it and we can execute it by calling the model name(naiveBayes or naivebayes). The fit() uses naiveBayes from e1071 package for implementation. The model is trained using fit() function and predicted against test(validation) dataset. For finding accuracy and confusion matrix, mmetric() function is used.

```
## [1] "AUC of ROC curve:"
```

```
## [1] 0.8632768
```

```
## $res
## NULL
##
## $conf
##       pred
## target   no  yes
##    no  9755 1210
##    yes  504  888
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "All metrics:"
```

```
##         ACC          CE         BER       KAPPA     CRAMERV    ACCLASS1
##  86.1293194  13.8706806  23.6210041  43.1947369   0.4439393  86.1293194
##    ACCLASS2    BAL_ACC1    BAL_ACC2        TPR1        TPR2        TNR1
##  86.1293194  76.3789959  76.3789959  88.9648883  63.7931034  63.7931034
##        TNR2  PRECISION1  PRECISION2         F11         F12        MCC1
##  88.9648883  95.0872405  42.3260248  91.9242367  50.8882521   0.5115727
##        MCC2       BRIER BRIERCLASS1 BRIERCLASS2         AUC   AUCCLASS1
##   0.5115727   0.1253224   0.1253224   0.1253224   0.8632768   0.8632768
##   AUCCLASS2        NAUC    TPRATFPR       ALIFT      NALIFT ALIFTATPERC
##   0.8632768   0.8632768   1.0000000   0.8222342   0.8222342   1.0000000
```

```
## [1] "Accuracy:"
```

```
## [1] 86.13
```

Though we get a good accuracy value, we see attribute duration highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.

So, remove attribute duration from the dataset for every model and predict term deposit subscription.

```
## [1] "AUC of ROC curve:"
```

```
## [1] 0.7791065
```

```
## $res
## NULL
##
## $conf
##       pred
## target   no  yes
##    no  9740 1225
##    yes  601  791
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 68.92492
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492

By Naive Bayes model, we get overall area under the curve of ROC as 0.7791065. The accuracy is 85.22%. Though accuracy has been reduced by around 1 point, it is necessary to remove duration attribute from the dataset. Also, F1 score is average(68.924916%). It is not the better one. We can implement further models and see whether accuracy has been increased or not. From the above, one can say Naive Bayes model is simple to use and computationally inexpensive.

2. KNN(K-Nearest Neighbor) model

The k-nearest neighbors (KNN) model is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. Even with such simplicity, it can give highly competitive results. The KNN algorithm is implemented by assuming that similar things are nearer to each other. "Closeness" or "Nearness" is defined in terms of a distance in the n-dimensional space, defined by the n Attributes in the training set. It calculates the distance between the unknown point and the training points by different metrics such as Euclidean distance etc.,

It is implemented in rminer package similar to naivebayes model using fit() and predict functions. The fit function trains the model whereas predict function tests the model. The mmetric function finds accuracy, confusion matrix and all metrics. The KNN model in rminer package uses kknn from kknn package. KNN model is implemented as:

```
## [1] "AUC of ROC:"
```

```
## [1] 0.7393618
```

```
## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10466   499
##    yes   958   434
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 65.41284
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284

By K-Nearest Neighbor(KNN) model, the accuracy has been increased to 88.21 compared to previous model and the overall area under the curve(AUC) of ROC is 0.7393618%. F1 score is 65.4128367. Though this model has increased accuracy , AUC and ROC is lower compared to previous ones. This implies knn is not a better model in predicting term deposit subscription.

3.1 Conditional inference Tree model(cTree)

cTree is a non-parametric class of classification and regression trees embedding tree-structured models into a well defined theory of conditional inference procedures. It is a statistical approach [to recursive partitioning] which takes into account the distributional properties of the measures. The conditional distribution of statistics measuring the association between responses and covariates is the basis for an unbiased selection among covariates measured at different scales. Moreover, multiple test procedures are applied to determine whether no significant association between any of the covariates and the response can be stated and the recursion needs to stop.

In rminer package, Conditional inference Tree model(cTree) uses ctree from party package. The fit function trains the model and predict function predicts term deposit subscription. The mmetric function finds out all error metrics. Ctree model is implemented as:

```
## $res
## NULL
##
```

```
## $conf
##       pred
## target   no  yes
##    no  3152  135
##    yes  178  240
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "AUC of ROC:"
```

```
## [1] 0.9439193
```

```
## [1] "F1 score: "
```

```
## [1] 77.8997
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970

The accuracy of the cTree model is 91.55%. One can see good change of accuracy compared to previous models(increased by 3 points). Also, F1 score is around 78% and AUC of ROC is 0.94391928. AUC > 0.9 is considered as a better model. Hence, ctree is a good model with increased accuracy, F1 score and AUC.

3.2 Conditional inference Tree model(cTree) by internal validation

cTree model by internal validation is a resampling procedure used to evaluate machine learning models on a limited data sample. Internal validation came into picture because the estimate obtained on the training set is generally lower than the estimate obtained with the test set, with the difference larger for smaller values of k due to over-training. Internal validation helps us to avoid over-training and overfitting of the model. It has single parameter k, that refers to the number of groups that a given data sample is split into. Therefore, it is called k-fold validation. When a specific value for k is chosen, such as k=10 becomes 10-fold validation.

In rminer package, cTree by internal validation uses mparheuristic function that returns a list of searching (hyper)parameters for a particular model or for a multiple list of models. The result is to be put in a search argument, used by fit or mining functions. The n arguement in mparheuristic() is number of searches or heuristic for the model. The lower and upper argument is a sequence of window(values) for k-value to be chosen.

Here, Internal validation uses 10-fold validation for better accuracy and to avoid overtraining and overfitting of the model. Internal validation trains the model with the evaluation of AUC(overall area under the curve) of ROC at every search.

```
## grid with: 8 searches (AUC values)
## i: 1 eval: 0.923132 best: 0.923132
## i: 2 eval: 0.9297599 best: 0.9297599
## i: 3 eval: 0.9341077 best: 0.9341077
## i: 4 eval: 0.9369744 best: 0.9369744
## i: 5 eval: 0.9403008 best: 0.9403008
## i: 6 eval: 0.9418541 best: 0.9418541
## i: 7 eval: 0.9414226 best: 0.9418541
## i: 8 eval: 0.9409276 best: 0.9418541
## $res
## NULL
##
## $conf
##       pred
## target   no  yes
##    no  3158  129
##    yes  178  240
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "AUC of ROC:"

## [1] 0.9434025

## [1] "F1 score: "

## [1] 78.17787
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787

The accuracy of the model is increased to 91.71% by 10-fold internal validation of the conditional inference tree. Here, at each fold, it picks observations at random with replacement (which means the same observation can appear twice). F1 score(78.1778719%) is slightly improved and AUC is 0.9434025.

4. eXtremeGradientBoosting(Tree) model

XGBoost stands for eXtreme Gradient Boosting. This algorithm came into existence to push the limit of computations resources for boosted tree algorithms. It uses Parallelization of tree construction technique by using all the CPU cores during training of the model. The implementation of the algorithm was engineered for efficiency of compute time and memory resources.

Gradient boosting refers to a class of ensemble machine learning algorithms that can be used for classification or regression predictive modeling problems. Ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. This type of ensemble machine learning model is referred as boosting. Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, "gradient boosting," as the loss gradient is minimized as the model is fit, much like a neural network.

In rminer package, xgboost algorithm is implemented from xgboost package. Here, nrounds parameter is set by default to 2. The parameter nrounds implies number of decision trees in the final model. The fit and predict function trains and test the model performance. XGBoost is implemented as:

```
## [11:21:14] WARNING: amalgamation/../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalu
## [1]  train-logloss:0.515195
## [2]  train-logloss:0.418863

## [1] "Overall area under the curve of ROC:"
## [1] 0.806813
```

```
## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10807   158
##    yes  1084   308
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 63.85995
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995

The accuracy value for this model is 89.95% and Overall area under the curve of ROC is 0.806813. When compared to cTree model, XGBoost model accuracy is lesser by 3 points but higher than KNN model. F1 score is63.8599536%. Compared to knn and naive bayes model, F1 score is low for xgboost model.

To check if accuracy improves, when number of decision trees in the model is changed to 3. Lets implement the model below:

```
## [11:21:14] WARNING: amalgamation/../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default eval
## [1]  train-logloss:0.515195
## [2]  train-logloss:0.418863
## [3]  train-logloss:0.360665
```

```
## [1] "Overall area under the curve of ROC:"
```

```
## [1] 0.8067975
```

```
## [1] "F1 score: "
```

```
## [1] 63.77938
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938

One can see, after changing nrounds parameter as 3 in XGBoost model, there is a very slight change in accuracy of the model. The accuracy is changed as 89.92% and AUC value is improved to 0.8067975 compared to knn model. But F1 score is decreased(around 63.7793765%).

5. Support Vector Machine model

Support Vector Machine is a supervised machine learning model with associated learning algorithms that analyze data for classification and regression analysis. It is widely used for classification models. Given a set of training examples, each belonging to one of two categories, SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. SVM maps training examples to points in space so as to maximise the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition, SVM can efficiently perform a non-linear classification using kernel trick by implicitly mapping their inputs into high-dimensional feature spaces.

In the SVM algorithm, to maximize the margin between the data points and the hyperplane, loss function named hinge loss is used. It is represented as,

$$c(x, y, f(x)) = (1 - y * f(x))_+$$

The cost is 0 if the predicted value and the actual value are of the same sign. If they are not, we then calculate the loss value. We also add a regularization parameter to the cost function. The objective of the regularization parameter is to balance the margin maximization and loss. After adding the regularization parameter, the cost functions looks as below.

$$min_w \lambda ||w||^2 + \sum_{i=1}^{n} (1 - y_i(x_i, w))_+$$

In rminer package, Support Vector machine is a classification model with discrete classes and probabilities. It uses ksvm from kernlab package. The fit and predict function trains and test the model performance. SVM model is implemented as:

```
## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10811   154
##    yes  1058   334
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 65.11202
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995

Model	Accuracy	AUC	F1_score
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202

Here, svm uses class as a task parameter. I.e, svm is a two class classification model. This model's accuracy in predicting term deposit subscription is 90.19%. SVM model prediction has slight varied accuracy when compared to XGBoost model. It has increased F1 score as 65.1120203% compared to xgboost models. SVM with classes can't determine overall area under the curve of ROC.

When svm uses probability as a task parameter, does accuracy improves for predicting term deposit subscription? It can be implemented as:

```
## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10826   139
##    yes  1079   313
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "Overall area under the curve of ROC:"
```

```
## [1] 0.7131144
```

```
## [1] "F1 score: "
```

```
## [1] 64.31109
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109

Support vector machine is an elegant and powerful algorithm. It predicts term deposit subscription of accuracy about 90.14%. The accuracy of the model has very slightly varied when task becomes probability compared to classes. F1 score(64.3110925%) is smaller than the svm with classes. AUC of ROC is lesser(0.7131144) compared to knn and naive bayes model.

6. LSSVM model

LSSVM stands for Least Squares Support Vector Machines. It uses least squares versions of support-vector machines (SVM), which are a set of related supervised learning methods that analyze data and recognize patterns. In this model, a set of linear equations is solved instead of a convex quadratic programming (QP) problem for classical SVMs. The algorithm is based on the minimization of a classical penalized least-squares cost function.

LSSVM uses a set of linear equations during the training process and chooses all training data as support vectors, so it has excellent generalization and low computation complexity. According to the parallel test results, LS-SVM is preferred especially for large scale problem, because its solution procedure has high efficiency and after pruning both sparseness and performance of LS-SVM are comparable with those of SVM[2].

LS-SVM classifier is represented as:

$$J_2(w, b, e) = \mu E_W + \zeta E_D$$

with

$$E_W = \frac{1}{2} w^T w$$

and

$$E_D = \frac{1}{2} \sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - (w^T \phi(x_i) + b))^2$$

.

Both μ and ζ should be considered as hyperparameters to tune the amount of regularization versus the sum squared error. The solution does only depend on the ratio

$$\gamma = \zeta / \mu$$

, therefore the original formulation uses only γ as tuning parameter.

In rminer package, lssvm model uses lssvm from kernlab package. It is a pure classification model. lssvm includes a reduced version of Least Squares SVM using a decomposition of the kernel matrix which is calculated by the csi function. The fit and predict function trains and test the model performance. LSSVM model is implemented as:

```
## $res
## NULL
##
## $conf
##        pred
## target    no   yes
##    no  10864   101
##    yes  1135   257
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 61.99455
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995

Model	Accuracy	AUC	F1_score
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455

LSSVM model does not improve accuracy compared to SVM with classes and probability. So, this is not considered as the best model in predicting term deposit subscription. Its accuracy is 90%. The F1 score is greatly decreased to 61.9945455%. F1 score near 60% is the average model. Therefore this model doesnot predict term deposit correctly. LSSVM with classes does not examine Overall area under the curve of ROC.

7.1 Random Forest

Random forest otherwise known as Random Decision forests, is an ensemble of many decision trees mainly used for classification and regression tasks. It constructs multiple of decision trees at a training time. For classification tasks, the output of the random forest is the class selected by most trees. Random forests selects random subset of the features at each candidate split in the learning process. Typically, for a classification problem with p features, $\sqrt{p}$ (rounded down) features are used in each split.

It uses bagging and feature randomness when building each individual tree whose prediction is more accurate than that of any individual tree. It randomize subsets of features and take average voting to make predictions. It merges multiple decision trees to get more accurate and stable prediction. It adds additional randomness to the model, while growing the trees. So in random forest, output trees are not only trained on different sets of data but also use different features to make decisions.

Random forest in the rminer package uses randomForest from randomForest package. The fit and predict function trains and test the model performance. Random Forest model is implemented as:

```
## $res
## NULL
##
## $conf
##       pred
## target   no  yes
##    no  3196   91
##    yes  280  138
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "Overall area under the curve of ROC:"

## [1] 0.7796379

## [1] "F1 score: "

## [1] 68.58635
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787

Model	Accuracy	AUC	F1_score
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635

The accuracy of the random forest model in predicting term deposit subscription is lower than ctree model. Yet, it is a good model and faster model to implement. The model produces accuracy of about 89.99%. This model is a better model with improved accuracy, overall area under the curve(0.7796379) and F1 score(68.5863462%) compared to SVM and LSSVM.

7.2 Tuning Random Forest

The hyper-parameters of the Random forest algorithm can be tuned by setting up the number of samples for tree building at each replacement. This can reduce overfitting and overtraining of the model with improved accuracy. In random forest model, pre-understanding the result can't be done because models are randomly processing. Tuning the algorithm will helps to control the training process and gain better result. There are many parameters to tune, but to have the biggest affect in model accuracy, two parameters mtry and ntree is used.

The parameter mtry denotes number of variables randomly collected to be sampled at each split time and ntree denotes number of branches that will grow after each time split. I used grid search() function to divide the dataset into 3 folds cross validation using AUC of ROC curve as an evaluation metric at each split. It is implemented as:

```
## grid with: 9 searches (AUC values)
## i: 1 eval: 0.7598384 best: 0.7598384
## i: 2 eval: 0.7754541 best: 0.7754541
## i: 3 eval: 0.7832832 best: 0.7832832
## i: 4 eval: 0.7681064 best: 0.7832832
## i: 5 eval: 0.77975 best: 0.7832832
## i: 6 eval: 0.785655 best: 0.785655
## i: 7 eval: 0.7706556 best: 0.785655
## i: 8 eval: 0.7816846 best: 0.785655
## i: 9 eval: 0.7856978 best: 0.7856978
```

```
## $mtry
## [1] 3
##
## $ntree
## [1] 500
##
## $importance
## [1] TRUE
```

```
## [1] "AUC: "
```

```
## [1] 0.7824564
```

```
## $res
## NULL
##
## $conf
##       pred
## target   no  yes
##    no  3216   71
##    yes  285  133
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 68.76036
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036

The accuracy of the model has been slightly improved to 90.39%. The main objective of this tuning is to control the training of the model and to improve the accuracy without overfitting and overtraining of the model. The overall curve of the ROC is 0.7824564 and F1 score is 68.7603621%. F1 scores and AUC is slightly improved compared to standard random forest.

8. LDA model

LDA stands for Linear Discriminant Analysis. It is a classifier with a linear decision boundary, generated by fitting class conditional densities to the data and using Bayes rule. The model fits a Gaussian density to each class, assuming that all classes share the same covariance matrix. The fitted model can also be used to reduce the dimensionality of the input by projecting it to the most discriminative directions. It is a dimensionality reduction technique. As the name implies dimensionality reduction techniques reduce the number of dimensions (i.e, variables) in a dataset while retaining as much information as possible. It is a two-class technique.

It is constructed as: 1) Calculating the distance between the mean of different classes(between-class variance). 2) Calculating distance between the mean and the sample of every class(within-class variance). 3) Constructing lower-dimensional space that maximizes between-class variance and minimizes within-class variance.

In rminer package, LDA model uses lda from MASS package. The fit and predict function trains and test the model performance. LDA model is implemented as:

```
## [1] "AUC: "
```

```
## [1] 0.7901313
```

```
## $res
## NULL
##
```

```
## $conf
##       pred
## target    no   yes
##    no  10405   560
##    yes   829   563
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 69.25717
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717

The LDA model prunes the attributes thereby selecting appropriate attributes that is important in predicting term deposit subscription. The accuracy of the model is 88.76% which is lower compared to random forest and svm model. It is a dimensionality reduction model. Therefore,we can't expect to improve the model quite high. Though accuracy is smaller, F1 scores are higher for LDA model compared to svm and lssvm model. F1 score is 69.2571666% and AUC of ROC is 0.7901313.Therefore, it is a average model in predicting term deposit subscription.

9. Generalized Linear model(GLM)

GLM model is an extension of linear regression model but with flexibility. It allows response variable y to have an error distribution other than a normal distribution. The General Linear Model (GLM) is a useful framework for comparing how several variables affect different continuous variables. It consists of three components: random component, systematic component, and a link function. The model should assume that Y is normally distributed, errors are normally distributed and independent, and X is fixed and constant variance. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value.

In rminer package, GLM model uses cv.glmnet from glmnet package. Here, Generalized Linear Model (GLM) uses lasso or elasticnet regularization. Also, cross-validation is used automatically to set the lambda parameter that is needed to compute the predictions.GLM model is implemented as:

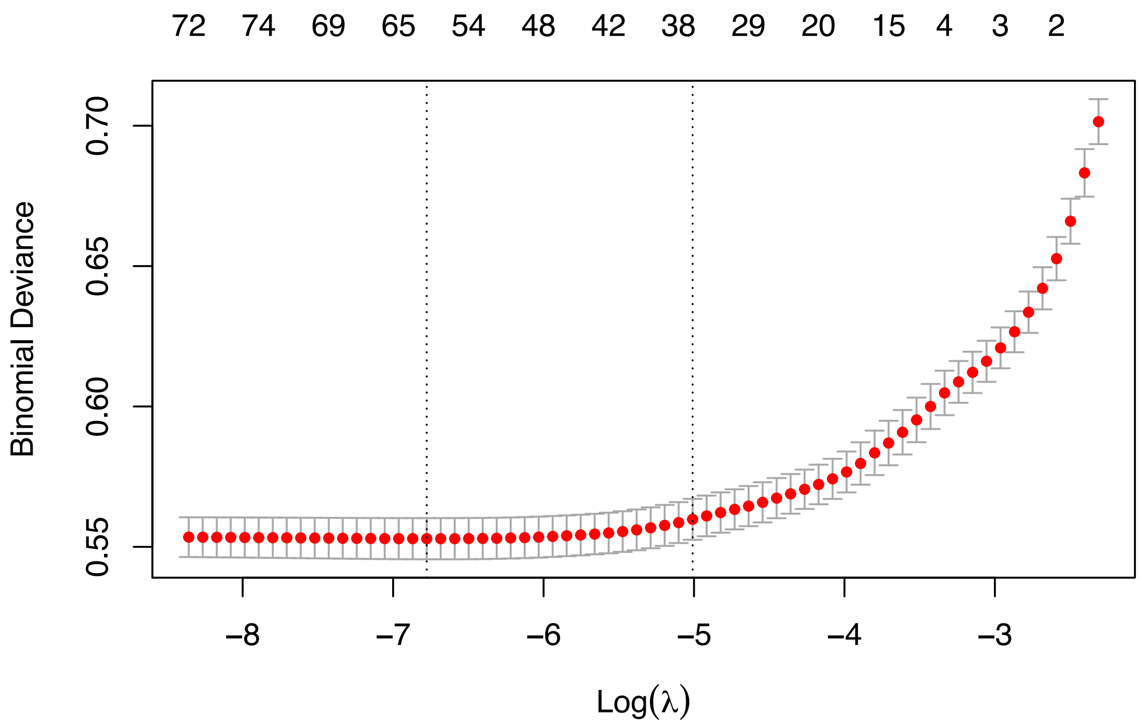
```
## [1] "AUC: "
```

```
## [1] 0.795586
## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10866    99
##    yes  1124   268
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "F1 score: "

## [1] 62.57202
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202

The accuracy of the model in predicting term deposit subscription is 90.1%. It is slightly better than LDA model. F1 score is low (i.e, 62.5720219%) and AUC of ROC is higher than random forest and lda models.AUC is 0.795586.

To check if GLM model's accuracy increases by k-fold validation method, lets implement that below:



```
## [1] "AUC: "

## [1] 0.7966625

## $res
## NULL
##
## $conf
##        pred
## target    no   yes
##    no  10858   107
##    yes  1120   272
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "F1 score: "

## [1] 62.68454
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109

Model	Accuracy	AUC	F1_score
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454

The accuracy of the model in predicting term deposit subscription is 90.07%. It does not quite improve the model compared to normal GLM model. There is no big difference between AUCs(0.7966625). Very slight improvement in F1 scores(62.6845363).

10. Multilayer Perceptron model

Multi-layer Perceptron (MLP) is a supervised machine learning algorithm that learns a function $f(.) : R_m - > R_n$ by training on a dataset, where m is the number of dimensions for input and n is the number of dimensions for output. Given a set of features X = x1,x2,..,xm and a target y, it can learn a non-linear function for either classification or regression. The standard multilayer perceptron (MLP) is a cascade of single-layer perceptrons. There is a layer of input nodes, a layer of output nodes, and one or more intermediate layers. The interior layers are sometimes called "hidden layers" because they are not directly observable from the systems inputs and outputs.

Since MLPs are fully connected, each node in one layer connects with a certain weight w_{ij} to every node in the following layer. Learning occurs in the perceptron by changing connection weights after each piece of data is processed, based on the amount of error in the output compared to the expected result.

Here, in rminer package, Multilayer perceptron has single hidden layer. It uses nnet from nnet package. In default, the maximum number of weights was increased and fixed to MaxNWts=10000). The fit and predict function trains and test the model performance. MLP model is implemented as:

```
## [1] "AUC: "
```

```
## [1] 0.7761738
```

```
## $res
## NULL
##
## $conf
##        pred
## target    no   yes
##    no  10616   349
##    yes   975   417
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 66.38853
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284

Model	Accuracy	AUC	F1_score
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454
MLP model	89.29	0.7761738	66.38853

The accuracy of the MLP model is 89.29%. MLP model does not improve the model performance compared to GLM model. Hence, it is not the best model in predicting term deposit subscription. There is no improvement in accuracy compared to GLM and KNN models. F1 scores(66.3885301%) are higher than GLM models. AUC of ROC is lower(0.7761738) compared to GLM models

11. Mutinom(Logistic Regression) model

Multinomial logistic regression is used to model nominal outcome variables, in which the log odds of the outcomes are modeled as a linear combination of the predictor variables. It Fits multinomial log-linear models via neural networks. It is a classification method that generalizes logistic regression to multi-class problems, i.e. with more than two possible discrete outcomes. It is used to predict the probabilities of the different possible outcomes of a categorically distributed dependent variable, given a set of independent variables. Multinomial logistic regression is used when the dependent variable in question is categorical.

The goal is to construct a linear predictor function that constructs a score from a set of weights that are linearly combined with the explanatory variables (features) of a given observation using a dot product:

$$score(X_i, k) = \beta_k . X_i$$

where X_i is the vector of explanatory variables describing observation i, β_k is a vector of weights (or regression coefficients) corresponding to outcome k, and $score(X_i, k)$ is the score associated with assigning observation i to category k.

In rminer package, Multinomial Logistic Regression model uses multinom from nnet package. The fit and predict function trains and test the model performance. The model is implemented as:

```
## [1] "AUC: "
```

```
## [1] 0.7963361
```

```
## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10794   171
##    yes  1048   344
##
## $roc
## NULL
```

```
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 65.36638
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454
MLP model	89.29	0.7761738	66.38853
Multinomial Logistic Regression model	90.14	0.7963361	65.36638

The accuracy of the model in predicting term deposit subscription is 90.14%. Multinomial Logistic Regression model improved the accuracy quite high compared to GLM and SVM models. The overall area under the curve of ROC is higher(0.7963361) than mlp and random forest models. F1_score(65.3663793%) is lower than mlp models.

12. Bagging model

Bagging, also known as bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy dataset. Each base classifier is trained in parallel with a random sample of data selected with replacement. i.e, the individual data points can be chosen more than once, N data from the original training dataset , where N is the size of the original training set. Training set for each of the base classifiers is independent of each other. Bagging helps to avoid overfitting. Each sample of data is used to train their decision trees. As a result, an ensemble of different models is achieved. Average of all the predictions from different trees are used which is more robust than a single decision tree classifier.

Suppose there are N observations and M features in training data set, a sample is taken randomly with replacement. Then, subset of M features are selected randomly and the feature which gives the best split is used to split the node iteratively till the tree is grown to the largest. Above steps are repeated n times and prediction is given based on the aggregation of predictions from n number of trees.

In rminer package, Bagging model uses bagging from adabag package. The fit and predict function in rminer trains and test the model performance. Bagging model is implemented as:

```
## [1] "AUC: "
```

```
## [1] 0.6349442
```

```
## $res
## NULL
##
```

```
## $conf
##       pred
## target    no   yes
##    no  10867    98
##    yes  1139   253
##
## $roc
## NULL
##
## $lift
## NULL
```

```
## [1] "F1 score: "
```

```
## [1] 61.82268
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454
MLP model	89.29	0.7761738	66.38853
Multinomial Logistic Regression model	90.14	0.7963361	65.36638
Bagging model	89.99	0.6349442	61.82268

The accuracy of the bagging model is 89.99%. The overall area under the curve is 0.6349442. Bagging model's AUC of ROC is the least value in this project. F1-score(61.8226783 %) is also lower compared to mlp and multinom model. There is no significant difference in accuracy compared to previous models.

13. Boosting model

Boosting is the ensemble machine learning model which reduces variance similar to Bagging model. Instead of random replacement like bagging, boosting model create collection of predictors. The goal is to build a strong classifier from the number of weak classifiers. A weak classifier is one which is slightly correlated with the true classification whereas strong classifier is arbitrarily well-correlated with the true classification.

At first, the model is built from the training data. The second model is built by analysing the first model for errors and trying to correct it. Consecutive trees (random sample) are fit at every step and the goal is to improve the accuracy from the prior tree. This method is continued and models are added until complete dataset is predicted or maximum number of models are added. When an input is misclassified by a hypothesis, its weight is increased so that next hypothesis is more likely to classify it correctly.

In rminer package, Boosting model uses boosting from adabag package. AdaBoost was the first really successful boosting algorithm developed for the purpose of binary classification which combines multiple "weak classifiers" into a single "strong classifier". The fit and predict function in rminer trains and test the model performance. Boosting model is implemented as:

```
## [1] "AUC: "

## [1] 0.7987788

## $res
## NULL
##
## $conf
##       pred
## target    no   yes
##    no  10791   174
##    yes  1034   358
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "F1 score: "

## [1] 65.95678
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454
MLP model	89.29	0.7761738	66.38853
Multinomial Logistic Regression model	90.14	0.7963361	65.36638
Bagging model	89.99	0.6349442	61.82268
Boosting model	90.22	0.7987788	65.95678

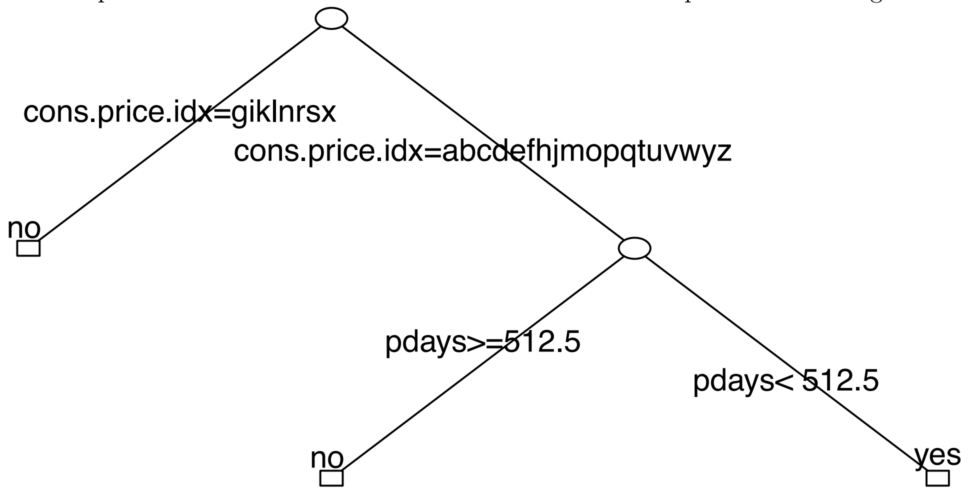
One can see boosting model is not better than bagging model although every step of boosting tries to improve the accuracy by identifying the errors. Thus, the accuracy of the model is 90.22%. Both AUC of ROC(0.7987788) and F1_scores (65.9567834) are lower than multinom and mlp models.

14. Decision tree model

Decision tress are represented in tree like structure where each internal node represents feature and leaf node represents class label of the model. It is a Supervised Machine Learning algorithm that helps us to determine a course of action or a statistical probability outcome. The paths from root to leaf represent classification rules. It identifies ways to split a data set based on different conditions or rules. Tree models where the target variable can take a discrete set of values are called classification trees.

To make a decision tree, start with a decision that needs to be made. Then, draw lines outward from the decision; each line moves from left to right, representing a potential option. At the end of each option, analyze the results. If the result of an option is a new decision, draw new lines out of that decision, representing the new options, and labeling them accordingly. If the result of an option is unclear, it denotes potential risk. Continue to expand until every line reaches an endpoint stating every choice or outcome is covered.

In rminer package, Decision tree uses rpart from rpart package. It is easy to read and interpret without requiring statistical knowledge. The fit and predict function in rminer trains and test the model performance. Bagging model is implemented as:



```
## [1] "AUC: "

## [1] 0.7390903

## $res
## NULL
##
## $conf
##         pred
## target   no  yes
##    no  3250   37
##    yes  332   86
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "F1 score: "

## [1] 63.21051
```

Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995

Model	Accuracy	AUC	F1_score
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454
MLP model	89.29	0.7761738	66.38853
Multinomial Logistic Regression model	90.14	0.7963361	65.36638
Bagging model	89.99	0.6349442	61.82268
Boosting model	90.22	0.7987788	65.95678
Decision Tree	90.04	0.7390903	63.21051

The accuracy of the decision tree model is 90.04%. Decision tree model is slightly better than Boosting and GLM model. But its AUC of ROC(0.7390903) and F1 scores(63.2105075%) are lower compared to boosting model.

Final model against validation set by ctree model(Internal validation)

By summing up the prediction of term deposit subscription by different models, it is concluded that cTree model by internal validation gives best accuracy, F1 score and AUC of the ROC. Hence, ctree model is used as a final model against validation set to predict term deposit subscription for clients. Here, we use the entire *bank_data* dataset for training against *validation*(testing) set.

This model is chosen because it has the highest accuracy along with large overall area under the curve of ROC and highest F1 score. Accuracy alone is not considered for choosing this model. Along with it F1 scores and AUC is also examined. It is said that higher the AUC, better the performance of the model at distinguishing between positive and negative classes. Here, AUC is above 0.9(around 0.94) and F1 score is around 78% which is the highest score in this project model.Hence, cTree by internal validation is considered as the best model.

```
validation <- validation %>% mutate(euribor3m = as.numeric(as.character(euribor3m)))
str(validation$euribor3m)
```

```
##  num [1:4117] 4.86 4.86 4.86 4.86 4.86 ...
```

Here, euribor of validation is convereted into numeric same as bank_data set.

cTree by internal validation uses mparheuristic function in rminer package that returns a list of searching (hyper)parameters for a particular model or for a multiple list of models. The result is to be put in a search argument, used by fit or mining functions. The n arguement in mparheuristic() is number of searches or heuristic for the model. The lower and upper argument is a sequence of window(values) for k-value to be chosen.

Here, Internal validation uses 10-fold validation for better accuracy and to avoid overtraining and overfitting of the model. Internal validation trains the model with the evaluation of AUC(overall area under the curve) of ROC at every search.

```
## grid with: 8 searches (AUC values)
## i: 1 eval: 0.9268852 best: 0.9268852
## i: 2 eval: 0.9346306 best: 0.9346306
## i: 3 eval: 0.9376976 best: 0.9376976
## i: 4 eval: 0.9399884 best: 0.9399884
## i: 5 eval: 0.940587 best: 0.940587
## i: 6 eval: 0.9410226 best: 0.9410226
## i: 7 eval: 0.9417786 best: 0.9417786
## i: 8 eval: 0.9415891 best: 0.9417786
```

```
## [1] "Heurestic parameter object:"
```

```
## $controls
## An object of class "TreeControl"
## Slot "varctrl":
## An object of class "VariableControl"
## Slot "teststat":
## [1] quad
## Levels: max quad
##
## Slot "pvalue":
## [1] TRUE
##
## Slot "tol":
## [1] 1e-10
##
## Slot "maxpts":
## [1] 25000
##
## Slot "abseps":
## [1] 1e-04
##
## Slot "releps":
## [1] 0
##
##
## Slot "splitctrl":
## An object of class "SplitControl"
## Slot "minprob":
## [1] 0.01
##
## Slot "minsplit":
## [1] 20
##
## Slot "minbucket":
## [1] 7
##
## Slot "tol":
## [1] 1e-10
##
## Slot "maxsurrogate":
## [1] 0
##
##
## Slot "gtctrl":
## An object of class "GlobalTestControl"
## Slot "testtype":
## [1] Bonferroni
## Levels: Bonferroni MonteCarlo Aggregated Univariate Teststatistic
##
```

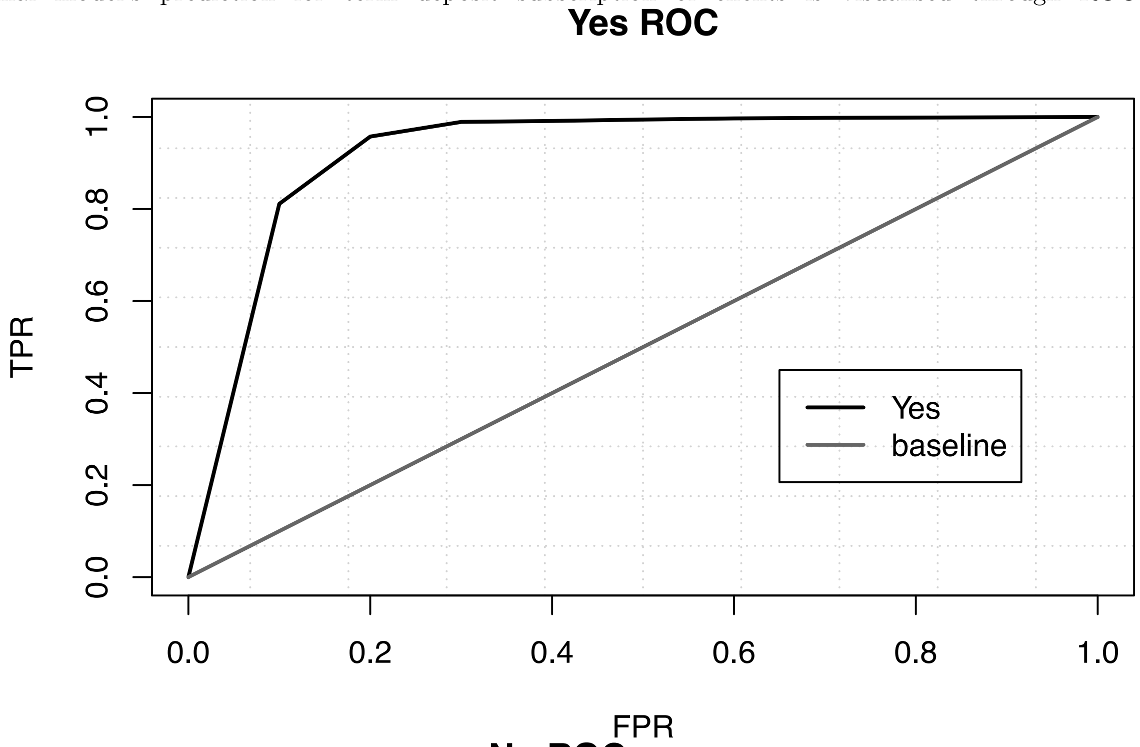
```
## Slot "nresample":
## [1] 9999
##
## Slot "randomsplits":
## [1] FALSE
##
## Slot "mtry":
## [1] 0
##
## Slot "mincriterion":
## [1] 0.8628571
##
##
## Slot "tgctrl":
## An object of class "TreeGrowControl"
## Slot "stump":
## [1] FALSE
##
## Slot "maxdepth":
## [1] 0
##
## Slot "savesplitstats":
## [1] TRUE
##
## Slot "remove_weights":
## [1] FALSE

## $res
## NULL
##
## $conf
##       pred
## target   no  yes
##    no  3507  146
##    yes  209  255
##
## $roc
## NULL
##
## $lift
## NULL

## [1] "AUC of ROC:"

## [1] 0.9409127

## [1] "All metrics:"

##          ACC          CE         BER       KAPPA     CRAMERV    ACCLASS1
## 91.37721642  8.62278358 24.51990924 54.17059544  0.54221125 91.37721642
##    ACCLASS2    BAL_ACC1    BAL_ACC2        TPR1        TPR2        TNR1
## 91.37721642 75.48009076 75.48009076 96.00328497 54.95689655 54.95689655
##        TNR2  PRECISION1  PRECISION2         F11         F12        MCC1
## 96.00328497 94.37567277 63.59102244 95.18252137 58.95953757  0.58190687
##        MCC2       BRIER BRIERCLASS1 BRIERCLASS2         AUC   AUCCLASS1
##  0.58190687  0.05871276  0.05871276  0.05871276  0.94091270  0.94091270
##   AUCCLASS2        NAUC    TPRATFPR       ALIFT      NALIFT ALIFTATPERC
##  0.94091270  0.94091270  1.00000000  0.89025862  0.89025862  1.00000000

## [1] "F1 score: "

## [1] 77.07103
```

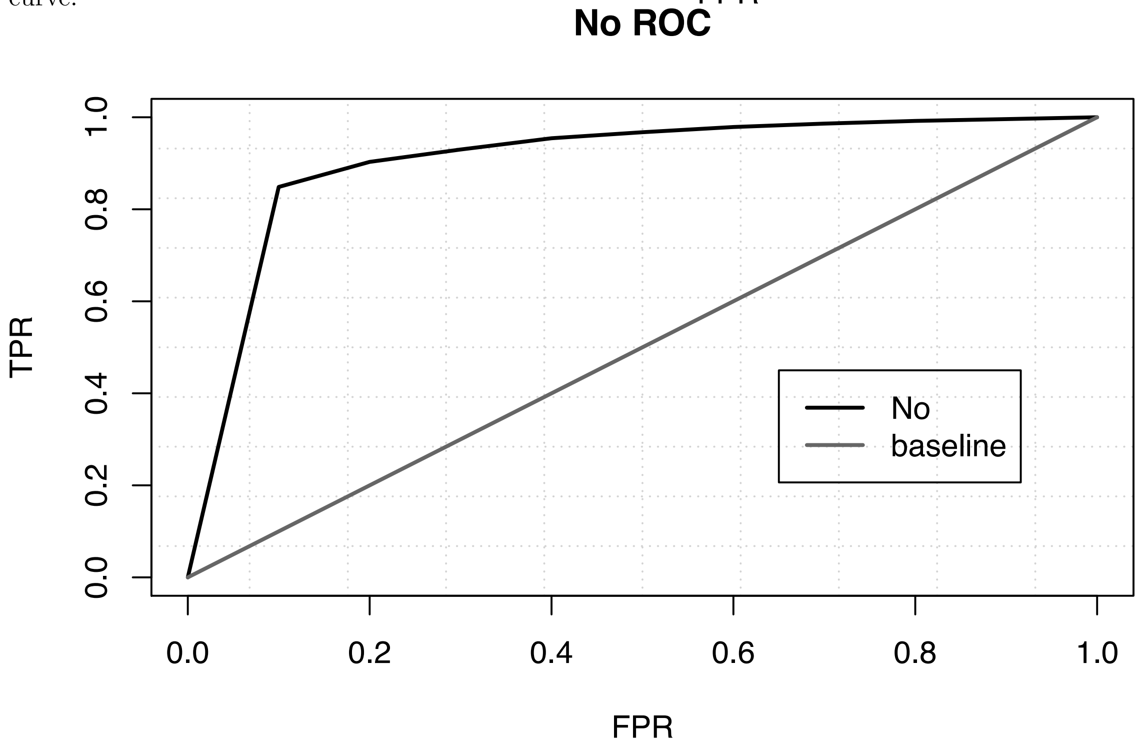
Model	Accuracy	AUC	F1_score
Naive Bayes Model	85.22	0.7791065	68.92492
KNN Model	88.21	0.7393618	65.41284
cTree Model	91.55	0.9439193	77.89970
cTree Model by 10-fold internal validation	91.71	0.9434025	78.17787
eXtremeGradientBoosting model	89.95	0.8068130	63.85995
XGBoost Model(when nrounds = 3)	89.92	0.8067975	63.77938
SVM model with classes	90.19	NA	65.11202
SVM model with probability	90.14	0.7131144	64.31109
LSSVM model with classes	90.00	NA	61.99455
Random Forest	89.99	0.7796379	68.58635
Random Forest Tuning	90.39	0.7824564	68.76036
LDA model	88.76	0.7901313	69.25717
GLM model	90.10	0.7955860	62.57202
GLM model tuning	90.07	0.7966625	62.68454
MLP model	89.29	0.7761738	66.38853
Multinomial Logistic Regression model	90.14	0.7963361	65.36638
Bagging model	89.99	0.6349442	61.82268
Boosting model	90.22	0.7987788	65.95678
Decision Tree	90.04	0.7390903	63.21051
Final model by cTree(internal validation)	91.38	0.9409127	77.07103

The accuracy of the final model is 91.38 by 10-fold internal validation of the conditional inference tree. Here, at each fold, it picks observations at random with replacement (which means the same observation can appear twice). F1 score(77.0710295) and AUC(0.9409127) are higher for this model.

Visualisation of final model

The final model's prediction for term deposit subscription of clients is visualised through ROC curve.





It is said that good ROC curve will bow up to the top left of the plot. Here, in both ROC plots for two classes yes and no, ROC curve bent near the top left corner compared to any other models. Hence this model is the best model in predicting term deposit subscription. It also predicts term deposit subscription accurate than any model.

Conclusion

Bank marketing campaign dataset has 20 descriptive and 1 target feature. All attributes are considered in the model except duration which is not known before a call is performed. In data exploration, chi-square test determines personal and housing loan attributes does not play an important role in predicting term deposit subscription by p-value. So, duration, personal and housing loan features are removed. Later, in base Logistic regression model, all other features are explored individually and found nr.employed, consumer price and confidence indexes, age, job, marital and education have no significant feature in the prediction of deposit subscription. So, they are omitted and accuracy of the model for prediction is estimated for the model. In second phase of the project, classification models are explored using rminer package. 14 classification models are executed one by one and their accuracy are compared with each other to choose the best model with highest accuracy. The highest accuracy produced by the model is Ctree(about 91%) in predicting term deposit subscription.

Hence, ctree by internal validation model is chosen as the best and final model because of its increased accuracy, overall area under the curve and F1 scores compared to other models. Thus, final model uses entire bank_data set against validation set(final holdout test). The accuracy produced by the final ctree model is 91.38%. Hence, it is proven that this model predicts term deposit subscription of clients better than any other models.

This project can be extended in future by handling outliers which may be part of the dataset and not just random figures. Removing them can modify the dataaset, so domain knowledge of that area is required to handle those outliers. They may have an effect in predicting term deposit subscription. And powerful neural networks can also be used for making good prediction of deposit subscription in the near future.

References

[1]. [Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014

[2]. Haifeng Wang and Dejin Hu, “Comparison of SVM and LS-SVM for Regression,” 2005 International Conference on Neural Networks and Brain, 2005, pp. 279-283, doi: 10.1109/ICNNB.2005.1614615.