Group Project
Phase 2

Data Mining Techniques– for Bank Marketing
Prediction of customers of bank who would subscribe to a term deposit

Group 5- members
Yasaswini Nallapula
Gopala Krishna Karnati
Mounika Anakanti

Background:

The finance industry is among the top industries exploiting the value of big data. As with any business, the effective use of digital marketing has become increasingly important for community banks focused on expanding their core customer base. According to 35+ years of research, the average US adult has had the same checking account for over 10 years. People like to stay in their bank for the long term. All community banks must continually find ways to encourage conversion of potential customers while retaining existing customers. That's why marketing is so important. The Audience needs to understand why switching to a bank is beneficial to them.

The Bank of Portugal wants to find a model that can predict future customers who will subscribe to their term deposits. Such effective predictive models can help increase the efficiency of campaigns by identifying customers who subscribe to term deposits and thereby direct marketing efforts towards them. This allows them to manage their resources better.

Business Definition: What is Term Deposit?

A term deposit is a fixed term deposit of money in an account of a financial institution. If a client or investor decides to deposit or invest in any of these accounts, they agree not to withdraw money for a period (from 1 month to 30 years) in exchange for a higher interest rate on that account. Banks can use this money to invest elsewhere or lend it to someone else for an agreed period. In other words, a term deposit guarantees that client will receive money at a fixed interest rate for a fixed time. As term deposits are an important source of income for banks, banks invest large sums of money and focus on marketing campaigns to attract more customers to commit to a term deposit. However, not everyone can afford to put their money away for a while or even want to, therefore it would be a waste of resources to include them in the marketing campaign. Identifying the target market, the group of customers who are likely to buy term deposits, is a key task that allows banks to focus resources only on those customers with high potential for sale. Targeting is the most time efficient and highest ROI marketing technique.

Introduction:

Marketing new potential customers and retaining them over the long term is a constant challenge for banks. To reach profitable customers, banks often use media such as social and digital media, customer service and strategic partnerships. But is it possible for banks to market to specific locations, communities, and groups of people? Fortunately, with the advent of machine learning technology, banking institutions are leveraging data and analytics solutions to target specific target customers and to predict which customers accurately and intelligently are likely to purchase financial products and services.

The goal of this project is to (1) explain how a banking institution can use its client data and machine learning techniques to predict which customers would subscribe for bank term deposits, (2) use three different machine learning algorithms to build up three predictive models and compare these three predictive models to see which algorithm is best suited for term deposit subscription prediction. With this, we aim to address two main research questions: (a) Is customer buying behavior predictable and how accurate is the prediction? (b) which machine learning algorithm is more effective in such a prediction task? It helps researchers increase their basic knowledge of various machine learning algorithms, better understand how to build various predictive models, and more effectively design and conduct studies on financial client behavior.

Data Mining Techniques that we will be using are:

- Random Forest Model

- Logistic Regression Model
- Naïve Bayes Model

Dataset source: https://datahub.io/machine-learning/bank-marketing#resource-bank-marketing

The Bank of Portugal has collected a huge amount of data that includes customers profiles of those who have subscribed to term deposits and the ones who did not subscribe to a term deposit. The data includes the following columns.
The data contains 45211 observations with 16 independent variables and 1 dependent variable. Often, more than one contact to the same client was required, to access if the product (bank term deposit) would be (or not) subscribed. Data is mix of Continuous and Categorical variables including demographic details. The predictor variable is probability (0 to 1) value. The dependent variable - term deposit subscription alone was recoded to 0 and 1 binary instead of two factors - yes and no.

Sl. No	Variable	Description
1	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 (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
4	education	(categorical: ‘unknown’,’secondary’,’primary’,’tertiary’)
5	default	has credit in default? (categorical: 'no','yes')
6	balance	average yearly balance, in euros (numeric)
7	housing	has housing loan? (categorical: 'no','yes','unknown')
8	loan	has personal loan? (categorical: 'no','yes','unknown')
9	contact	contact communication type (categorical: 'cellular','telephone', ‘unknown’)
10	day	last contact day of the month (numeric)

11	month	last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
12	duration	last contact duration, in seconds (numeric)
13	campaign	number of contacts performed during this campaign and for this client (numeric, includes last contact)
14	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)
15	previous	number of contacts performed before this campaign and for this client (numeric)
16	Poutcome	outcome of the previous marketing campaign (categorical: 'failure',unknown,'success', 'other')
17	y	has the client subscribed a term deposit? (numerical: 1, 2)

Challenges experienced and how these were resolved:

In the initial dataset variables are of string and number types, I had to convert them to factor type (as factor is categorical variable that stores both integer and string data values as levels) to plot the data visualization. Initially, I tried converting each variable and it is a repetitive task. Later, I found one source where all variables of string type can be converted to factor type in one line of code.

bank <- as.data.frame(unclass(bank), stringsAsFactors = TRUE)

The variable named as y (class) is a numeric type and it describes whether the client is subscribed to a term deposit or not i.e., the value ‘1’ means ‘no’ and the value ‘2’ means ‘yes’.

I need a categorical type target variable for data analysis and for model building (of naïve bayes, random forest). But I need numeric type target variable with binary format values for Logistic Regression(LR) model. So, I used one copy of bank data for LR and another copy for remaining tasks with the related changes in each of them.

Made below changes for LR model i.e., in ‘y’ variable value 1 → to 0 and 2 → to 1 (0 means ‘no’ and 1 means ‘yes’). Because binomial family in logistic regression generates errors (or sometimes warning messages) if I use anything other than 0 and 1. I used the below line of code for this purpose.

bank_lr <- bank
bank_lr$y <- ifelse(bank_lr$y==2, 1,0)

Converted numeric type target variable to factor type with the related labels ie., 'no' →for 1 and 'yes' →for 2. This is useful in visualization of plots and in model building.
bank$y <- factor(bank$y, levels = c(1,2), labels = c('no', 'yes'))

Implementation in R:

Step 1: Loading the dataset into R studio by using read.csv command.

Step 2: Understanding the data - by using its structure and summary commands. Installing the required libraries. Renaming the column names for better understanding.

Step 3: Data validation – checking for duplicates and null values. Dataset has no duplicates and has no null values. It is clean.

Step 4: Data Pre-processing - Converting the string to factor types and creating two copies of data for building different models. In one copy, changing the values of target variable to binary format (i.e., to 0 and 1). In another copy, converting the target variable to factor type and labeling it with ('yes', 'no') different levels.

Step 5: Exploratory data analysis – Here, I will try to use visualizations to understand the data, to understand the correlations between variables (univariate analysis, multivariate analysis).

Step 6: Data Preparation – Transforming the numeric variables using scale (Z-score standardization) to handle the distance calculation for the Logistic Regression model. Removing the duration feature for the Random Forest model.
Splitting the dataset into training and test data for the purpose of building Data Mining techniques which will use training data and then they will make predictions on the testing data.

Step 7: Model building – Logistic Regression, Naïve bayes and Random Forest on the training data.

Step 8: Evaluate the Prediction models - Applying trained models on the testing data to predict results.

Step 9: Comparing the accuracy of models.

Results in R:

Step 1: Loading the dataset.

```
Console  Terminal  Background Jobs
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> #Load the dataset into R
> bank <- read.csv("D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/bank_marketing.csv",
header=TRUE,sep=",")
>
```

Step 2: Renaming the column names and understanding the data.

```
Console  Terminal  Background Jobs
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> colnames(bank) = c("age", "job", "marital", "education", "default", "balance", "housing", "loan",
+                    "contact", "day", "month", "duration", "campaign", "pdays", "previous",
+                    "poutcome", "y")
> dim(bank)
[1] 45211    17
> str(bank)
'data.frame':   45211 obs. of  17 variables:
 $ age      : int  58 44 33 47 33 35 28 42 58 43 ...
 $ job      : chr  "management" "technician" "entrepreneur" "blue-collar" ...
 $ marital  : chr  "married" "single" "married" "married" ...
 $ education: chr  "tertiary" "secondary" "secondary" "unknown" ...
 $ default  : chr  "no" "no" "no" "no" ...
 $ balance  : int  2143 29 2 1506 1 231 447 2 121 593 ...
 $ housing  : chr  "yes" "yes" "yes" "yes" ...
 $ loan     : chr  "no" "no" "yes" "no" ...
 $ contact  : chr  "unknown" "unknown" "unknown" "unknown" ...
 $ day      : int  5 5 5 5 5 5 5 5 5 5 ...
 $ month    : chr  "may" "may" "may" "may" ...
 $ duration : int  261 151 76 92 198 139 217 380 50 55 ...
 $ campaign : int  1 1 1 1 1 1 1 1 1 1 ...
 $ pdays    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
 $ previous : int  0 0 0 0 0 0 0 0 0 0 ...
 $ poutcome : chr  "unknown" "unknown" "unknown" "unknown" ...
 $ y        : int  1 1 1 1 1 1 1 1 1 1 ...
>
```

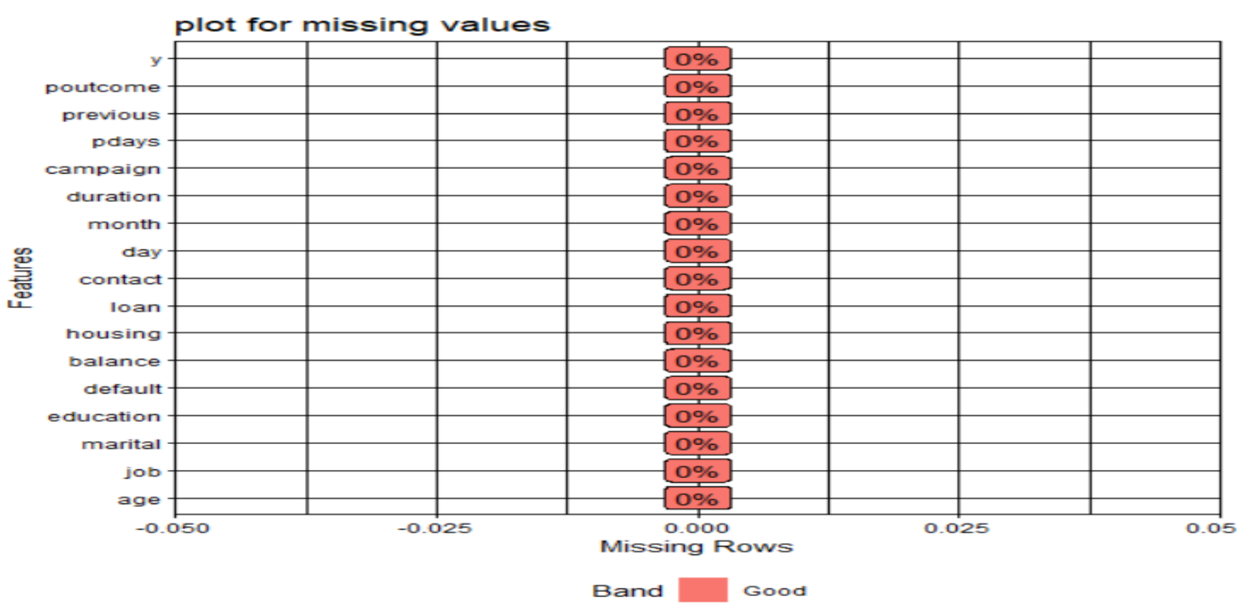
```
Console  Terminal  Background Jobs
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> summary(bank)
      age             job              marital           education           default         
 Min.   :18.00   Length:45211       Length:45211       Length:45211       Length:45211      
 1st Qu.:33.00   Class :character   Class :character   Class :character   Class :character  
 Median :39.00   Mode  :character   Mode  :character   Mode  :character   Mode  :character  
 Mean   :40.94                                                                               
 3rd Qu.:48.00                                                                               
 Max.   :95.00                                                                               
    balance         housing              loan             contact               day       
 Min.   : -8019   Length:45211       Length:45211       Length:45211       Min.   : 1.00  
 1st Qu.:    72   Class :character   Class :character   Class :character   1st Qu.: 8.00  
 Median :   448   Mode  :character   Mode  :character   Mode  :character   Median :16.00  
 Mean   :  1362                                                            Mean   :15.81  
 3rd Qu.:  1428                                                            3rd Qu.:21.00  
 Max.   :102127                                                            Max.   :31.00  
    month              duration         campaign          pdays          previous          poutcome        
 Length:45211       Min.   :   0.0   Min.   : 1.000   Min.   : -1.0   Min.   :  0.0000   Length:45211      
 Class :character   1st Qu.: 103.0   1st Qu.: 1.000   1st Qu.: -1.0   1st Qu.:  0.0000   Class :character  
 Mode  :character   Median : 180.0   Median : 2.000   Median : -1.0   Median :  0.0000   Mode  :character  
                    Mean   : 258.2   Mean   : 2.764   Mean   : 40.2   Mean   :  0.5803                     
                    3rd Qu.: 319.0   3rd Qu.: 3.000   3rd Qu.: -1.0   3rd Qu.:  0.0000                     
                    Max.   :4918.0   Max.   :63.000   Max.   :871.0   Max.   :275.0000                     
       y        
 Min.   :1.000  
 1st Qu.:1.000  
 Median :1.000  
 Mean   :1.117  
 3rd Qu.:1.000  
 Max.   :2.000  
>
```

Installing required Libraries

```r
#installing libraries
library(caret)
library(caTools)
install.packages("DataExplorer")
library(DataExplorer)
install.packages("ROCR")
library(ROCR)
library(dplyr)
library(glue)
#install.packages("randomForest")
library(e1071)
library(randomForest)
#library(partykit)
#library(rpart.plot)
#install.packages("naivebayes")
#library(naivebayes)
```

Step -3: Data Validation

```
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> ##check for duplicate rows
> sum(duplicated(bank))
[1] 0
> ##check for Missing values
> sapply(bank, function(x) sum(is.na(x)))
      age       job   marital education   default   balance   housing      loan   contact       day
        0         0         0         0         0         0         0         0         0         0
    month  duration  campaign     pdays  previous  poutcome         y
        0         0         0         0         0         0         0
> plot_missing(bank, title='plot for missing values', ggtheme=theme_linedraw(),
+              theme_config=list(legend.position=c("bottom")))
> 
```



Data validation is an important step, because if the data has duplicates/missing or null values, it gives wrong results while building the models. The bank dataset has no duplicates rows and it has no null values (no missing values). So, the data is clean.

Step -4: Data Pre-processing

String (character) type variables and target variable are converted to factor type for the purpose of plot visualizations and data analysis.

A copy of the data is created (i.e., bank_lr dataset). In its target variable is of numeric type only, but the values are changed to binary type (i.e., 0 and 1) as our Logistic Regression model uses binomial type as a classifer.

```r
#convert character to factor type
bank <- as.data.frame(unclass(bank), stringsAsFactors = TRUE)
bank_lr <- bank
#convert int to factor type
bank$y <- factor(bank$y, levels = c(1,2), labels = c('no', 'yes'))
bank_lr$y <- ifelse(bank_lr$y==2, 1,0)
#print all the levels of factor variables to find any missing values
levels(bank$job)
levels(bank$marital)
levels(bank$education)
levels(bank$default)
levels(bank$housing)
levels(bank$loan)
levels(bank$contact)
levels(bank$month)
levels(bank$poutcome)

```

```
Console   Terminal ×   Background Jobs ×
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> levels(bank$job)
 [1] "admin."        "blue-collar"   "entrepreneur"  "housemaid"     "management"    "retired"
 [7] "self-employed" "services"      "student"       "technician"    "unemployed"    "unknown"
> levels(bank$marital)
[1] "divorced" "married"  "single"
> levels(bank$education)
[1] "primary"   "secondary" "tertiary"  "unknown"
> levels(bank$y)
[1] "no"  "yes"
>
```

```
> str(bank)
'data.frame':	45211 obs. of  17 variables:
 $ age      : int  58 44 33 47 33 35 28 42 58 43 ...
 $ job      : Factor w/ 12 levels "admin.","blue-collar",..: 5 10 3 2 12 5 5 3 6 10 ...
 $ marital  : Factor w/ 3 levels "divorced","married",..: 2 3 2 2 3 2 3 1 2 3 ...
 $ education: Factor w/ 4 levels "primary","secondary",..: 3 2 2 4 4 3 3 3 1 2 ...
 $ default  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 2 1 1 ...
 $ balance  : int  2143 29 2 1506 1 231 447 2 121 593 ...
 $ housing  : Factor w/ 2 levels "no","yes": 2 2 2 2 1 2 2 2 2 2 ...
 $ loan     : Factor w/ 2 levels "no","yes": 1 1 2 1 1 1 2 1 1 1 ...
 $ contact  : Factor w/ 3 levels "cellular","telephone",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ day      : int  5 5 5 5 5 5 5 5 5 5 ...
 $ month    : Factor w/ 12 levels "apr","aug","dec",..: 9 9 9 9 9 9 9 9 9 9 ...
 $ duration : int  261 151 76 92 198 139 217 380 50 55 ...
 $ campaign : int  1 1 1 1 1 1 1 1 1 1 ...
 $ pdays    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
 $ previous : int  0 0 0 0 0 0 0 0 0 0 ...
 $ poutcome : Factor w/ 4 levels "failure","other",..: 4 4 4 4 4 4 4 4 4 4 ...
 $ y        : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
> 
```

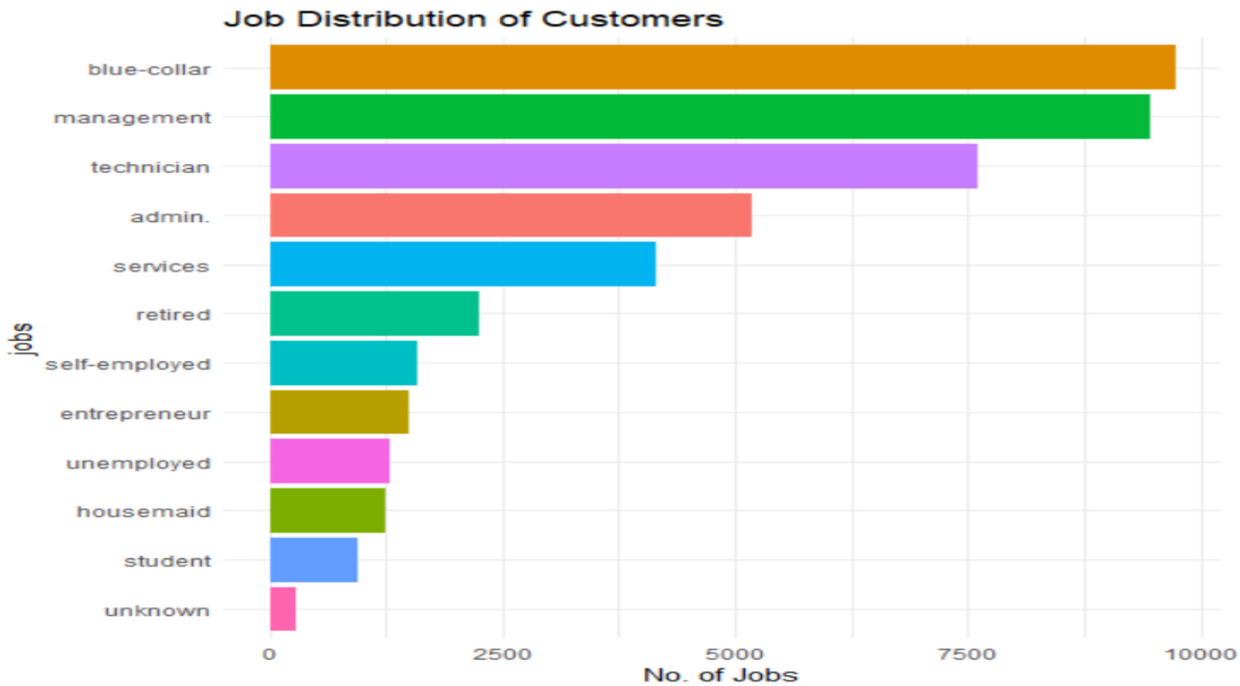
Step 5: Exploratory Data Analysis

```
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> table(bank$y)

   no   yes 
39922  5289 
> round(prop.table(table(bank$y)) * 100, digits = 1)

  no  yes 
88.3 11.7 
> 
```

In the data, the number of persons who subscribed to Term Deposit are 5289 and it is 11.7%. The number of people who are not subscribed is 39922 and it is 88.3%.

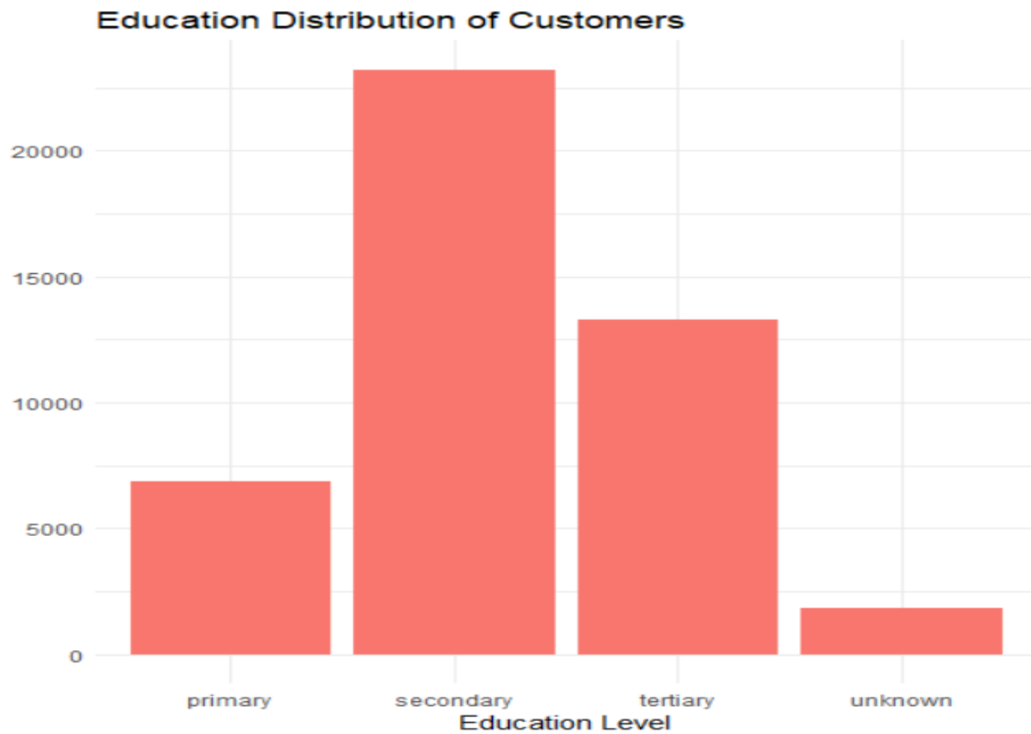
Univariate Analysis:

```
> job_distribution <- bank %>% group_by(job) %>% summarise(job_count = n()) %>% arrange(-job_count)
> job_distribution_plot <- ggplot(data = job_distribution, aes(x = job_count,
+                                                              y = reorder(job, job_count),
+                                                              text = glue("No. of customers: {job_count}")
+ )) +
+     geom_col(aes(fill = job)) +
+     labs( title = "Job Distribution of Customers",
+           x = "No. of Jobs",
+           y = "Jobs"
+     ) +
+     theme_minimal() +
+     theme(legend.position = "none")
> job_distribution_plot
> 
```



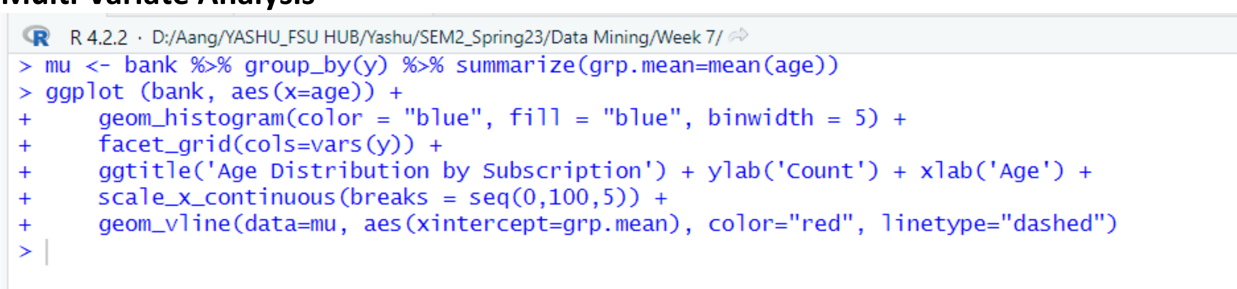
In the given bank data, most of the customers are under the job categories of blue-collar and management. There is minimal difference between the number of customers who are self-employed, entrepreneur, unemployed, housemaid and student. There are few customers who did not mention their job in the data.

```
> education_distribution <- bank %>% group_by(education) %>% summarise(ed_count = n())
> education_distribution_plot <- ggplot(data = education_distribution, aes(y = ed_count,
+                                                                           x = education,
+                                                                           text = glue("No. of customers: {ed_count}")
+ )) +
+     geom_col(aes(fill = "brickred")) +
+     labs( title = "Education Distribution of Customers",
+           x = "Education Level",
+           y = ""
+     ) +
+     theme_minimal() +
+     theme(legend.position = "none")
> education_distribution_plot
> 
```

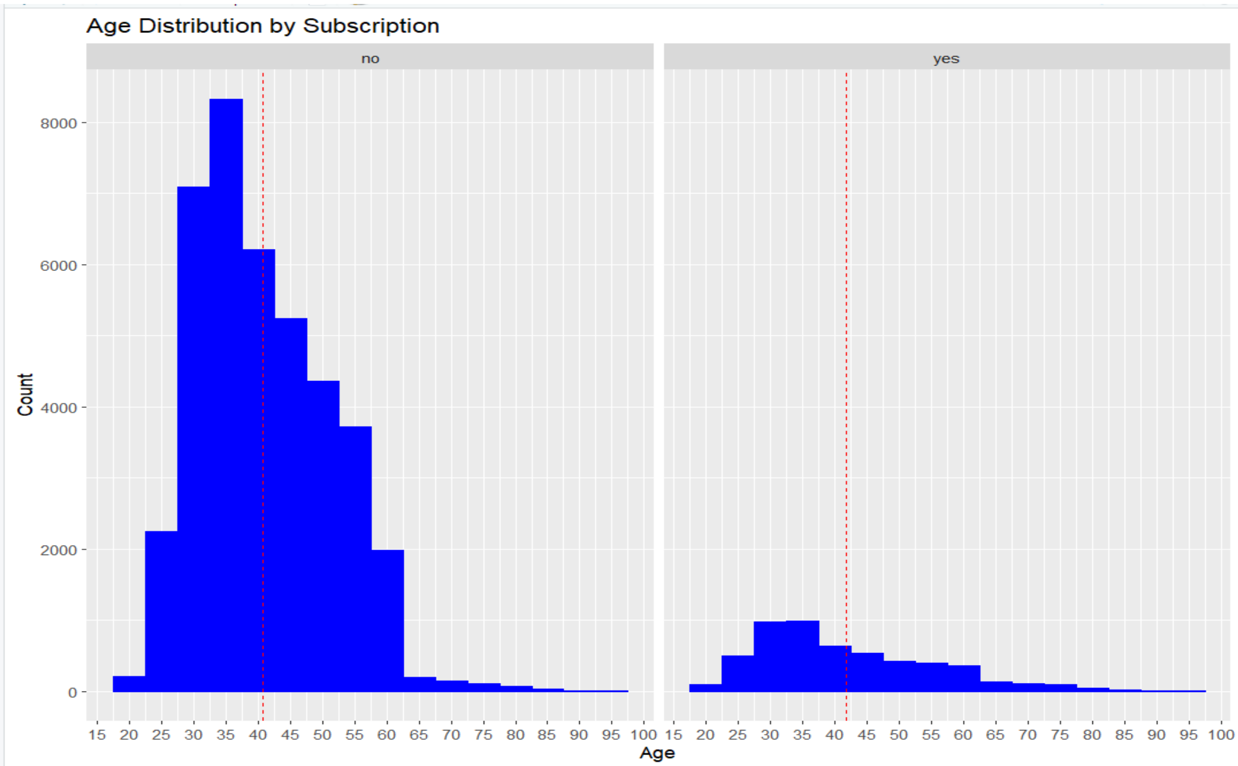


Most of the customers are under the category of secondary and tertiary in their education levels. It means, most of them are well educated. There are few customers (around 2000) who did not mention their education level.

Multi-variate Analysis

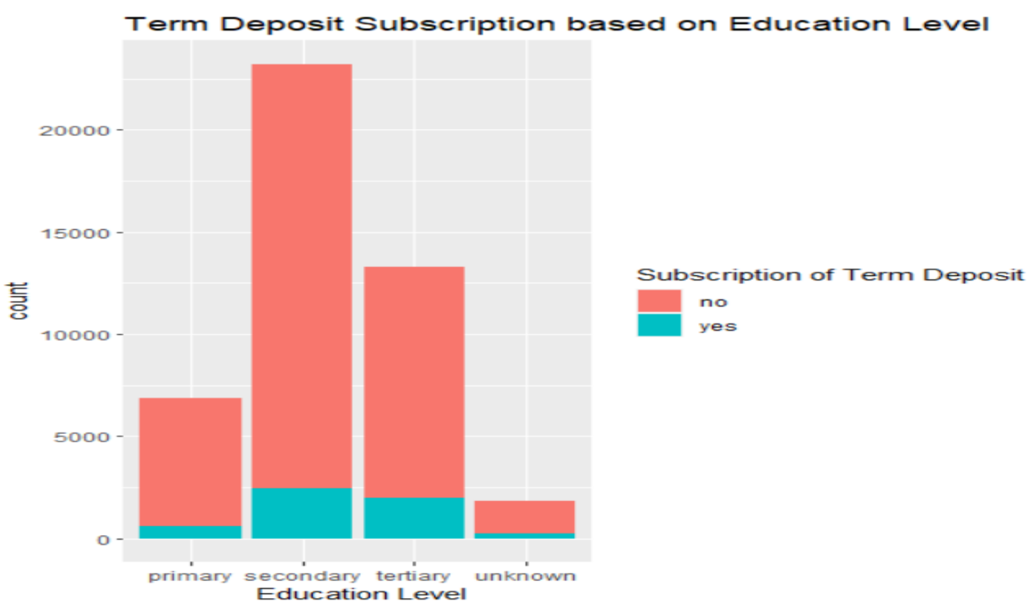



```
> mu <- bank %>% group_by(y) %>% summarize(grp.mean=mean(age))
> ggplot (bank, aes(x=age)) +
+     geom_histogram(color = "blue", fill = "blue", binwidth = 5) +
+     facet_grid(cols=vars(y)) +
+     ggtitle('Age Distribution by Subscription') + ylab('Count') + xlab('Age') +
+     scale_x_continuous(breaks = seq(0,100,5)) +
+     geom_vline(data=mu, aes(xintercept=grp.mean), color="red", linetype="dashed")
>
```



This bar plot divides the customers into two groups who are subscribed and who are not subscribed to term deposit based on their age. In both plots, most of the people under the age of 30-35 are the ones who are subscribed and who are not.

```
ggplot(data = bank, aes(x=education, fill=y)) +
  geom_bar() +
  ggtitle("Term Deposit Subscription based on Education Level") +
  xlab(" Education Level") +
  guides(fill=guide_legend(title="Subscription of Term Deposit"))
```

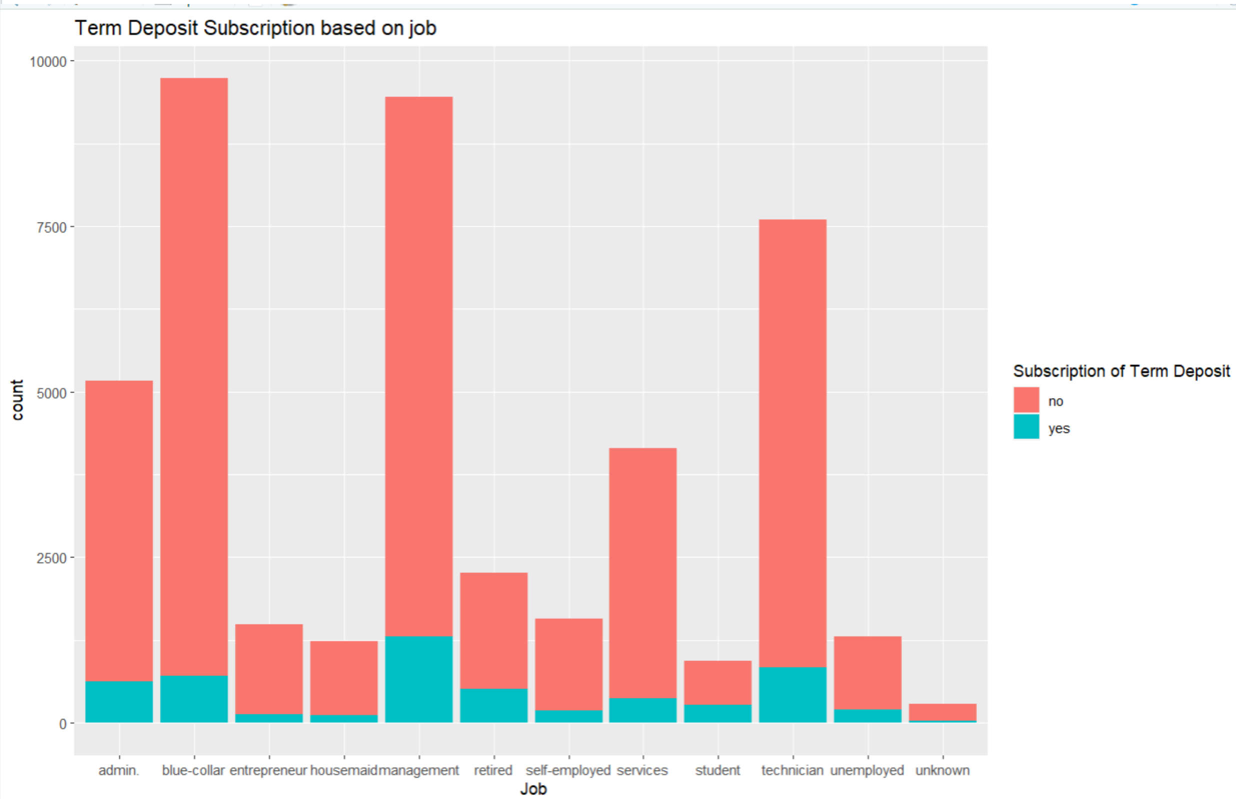


In this par plot also most of the people who are subscribed and who are not subscribed comes under the category of secondary and tertiary levels.

Console Terminal × Background Jobs ×

R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/

```
> ggplot(data = bank, aes(x=job, fill=y)) +
+     geom_bar() +
+     ggtitle("Term Deposit Subscription based on job") +
+     xlab(" Job") +
+     guides(fill=guide_legend(title="Subscription of Term Deposit"))
>
```



In this plot, most of the people who are subscribed are under the job categories of management, technician and blue-collar.

Step 6: Data Preparation (Splitting the data into train and test for Logistic Regression)

The distance calculation will be heavily dependent upon the measurement scale of the input variables. If the range of numeric variables is larger, this could potentially cause problems for classifier, hence rescaling/transforming the numeric variables to a standard range of values is necessary. I will be using Z-score standardization here.

Splitting the data into two portions: a training dataset that will be used to build the two models and a test dataset that will be used to estimate the predictive accuracy of the model.

Splitting a copy of bank dataset(bank_lr) that will be used in the Logistic Regression model.

```
Console   Terminal ×   Background Jobs ×
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> #transforming the numerical variables using scale
> bank_lr[c(1,6,10,12,13)] <- scale(bank_lr[c(1,6,10,12,13)])
> #split the dataset into training and testing for logistic regression
> set.seed(112)
> split = sample.split(bank_lr$y,SplitRatio = 0.70)
> bank_lr_training = subset(bank_lr, split == TRUE)
> bank_lr_test = subset(bank_lr, split == FALSE)
>
```

Step 7: Logistic Regression (LR) Model Building

Logistic Regression is a powerful statistical way of modeling a binomial outcome with one or more explanatory variables. It measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution.

<u>What is Logistic Regression with glm(family = binomial)?</u>

The most common non-normal regression analysis is logistic regression, where your dependent variable is just 0s and 1. To do a logistic regression analysis with glm(), use the family = binomial argument.

Creating custom function For Binary Class Performance Evaluation (it will be useful in the LR classifier of binomial type)

```
Console   Terminal ×   Background Jobs ×
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> binclass_eval = function (actual, predict) {
+     cm = table(as.integer(actual), as.integer(predict), dnn=c('Actual','Predicted'))
+     ac = (cm['1','1']+cm['0','0'])/(cm['0','1'] + cm['1','0'] + cm['1','1'] + cm['0','0'])
+     pr = cm['1','1']/(cm['0','1'] + cm['1','1'])
+     rc = cm['1','1']/(cm['1','0'] + cm['1','1'])
+     fs = 2* pr*rc/(pr+rc)
+     list(cm=cm, recall=rc, precision=pr, fscore=fs, accuracy=ac)
+ }
>
```

The below code creates a distribution plot function which will be used to plot the prediction distribution.

```
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> plot_pred_type_distribution <- function(df, threshold) {
+     v <- rep(NA, nrow(df))
+     v <- ifelse(df$pred >= threshold & df$y == 1, "TP", v)
+     v <- ifelse(df$pred >= threshold & df$y == 0, "FP", v)
+     v <- ifelse(df$pred < threshold & df$y == 1, "FN", v)
+     v <- ifelse(df$pred < threshold & df$y == 0, "TN", v)
+
+     df$pred_type <- v
+
+     ggplot(data=df, aes(x=y, y=pred)) +
+         geom_violin(fill='black', color=NA) +
+         geom_jitter(aes(color=pred_type), alpha=0.6) +
+         geom_hline(yintercept=threshold, color="red", alpha=0.6) +
+         scale_color_discrete(name = "type") +
+         labs(title=sprintf("Threshold at %.2f", threshold))
+ }
>
```

Creating Logistic Regression Classifier by using training dataset and classifier is of binomial family type.

```
Console   Terminal ×   Background Jobs ×
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> #creating the LR classifer
> classifier.lm = glm(formula = y ~ .,
+                     family = binomial,
+                     data = bank_lr_training)
>
```

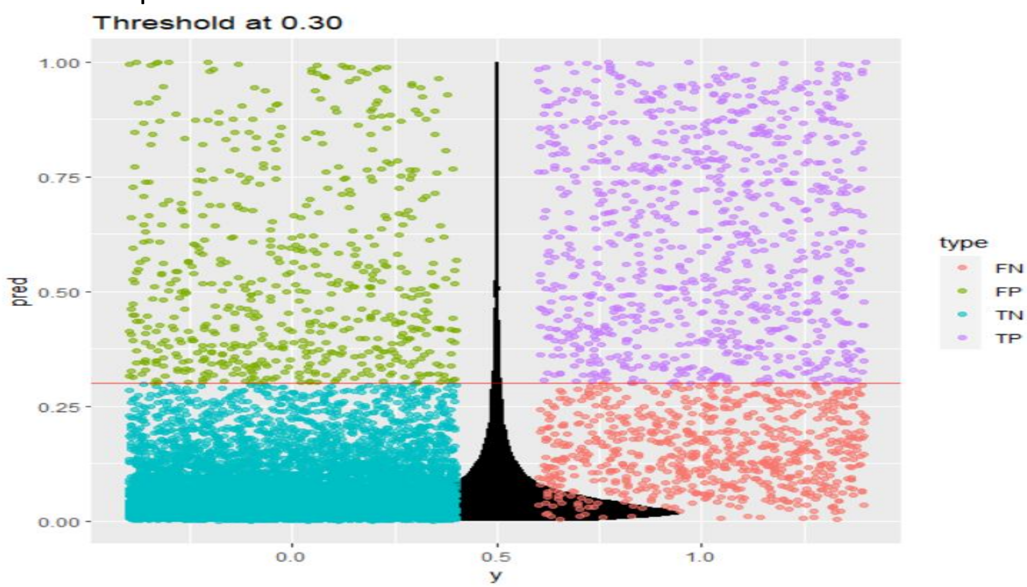
Step 8: Evaluate the Logistic Regression Model and Apply test dataset to it

```
#Evaluating the LR model
pred_lm = predict(classifier.lm, type='response', newdata=bank_lr_test[-17])
predictions_LR <- data.frame(y = bank_lr_test$y, pred = NA)
predictions_LR$pred <- pred_lm
plot_pred_type_distribution(predictions_LR,0.30)
```

```
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> test.eval.LR = binclass_eval(bank_lr_test[, 17], pred_lm > 0.30)
> test.eval.LR$cm
       Predicted
Actual     0     1
     0 11372   605
     1   720   867
> acc_LR=test.eval.LR$accuracy
> prc_LR=test.eval.LR$precision
> rc_LR=test.eval.LR$recall
> cat("Accuracy:  ",   acc_LR,
+     "\nPrecision: ", prc_LR,
+     "\nRecall: ",rc_LR)
Accuracy:    0.902315
Precision:   0.5889946
Recall:   0.5463138
>
```

From the actual and predicted table – 0 is for ‘no’ and 1 is for ‘yes’.

Actually the customers who are not subscribed and the exact predicted is – 11372 (it is True Negative). The number of customers who are not subscribed but predicted as otherwise is – 605(it is False Positive). The customers who are subscribed but predicted as otherwise is – 720 (it is false negative). The customers who are subscribed and the exact predicted is – 867(True positive).
The visual plot of this table is the below:

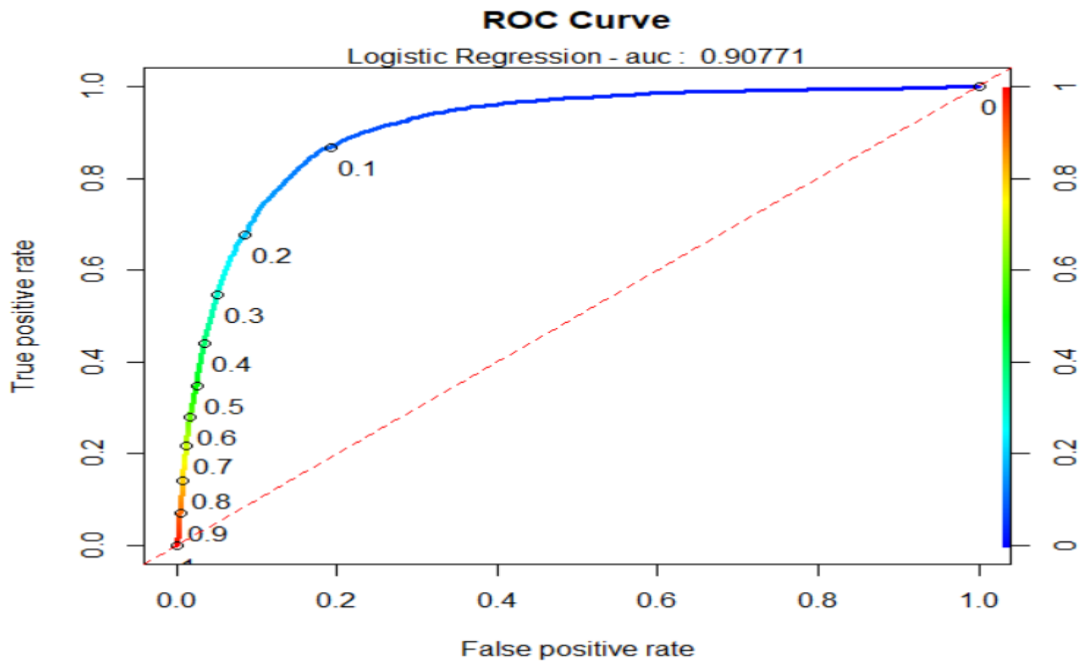


Here, FN- False Negative; FP- False Positive, TN-True Negative, TP-True Positive.

```
Console   Terminal ×   Background Jobs ×
R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/
> rocr.pred.lr = prediction(predictions = pred_lm, labels = bank_lr_test$y)
> rocr.perf.lr = performance(rocr.pred.lr, measure = "tpr", x.measure = "fpr")
> rocr.auc.lr = as.numeric(performance(rocr.pred.lr, "auc")@y.values)
> rocr.auc.lr
[1] 0.9077088
> plot(rocr.perf.lr,
+      lwd = 3, colorize = TRUE,
+      print.cutoffs.at = seq(0, 1, by = 0.1),
+      text.adj = c(-0.2, 1.7),
+      main = 'ROC Curve')
> mtext(paste('Logistic Regression - auc : ', round(rocr.auc.lr, 5)))
> abline(0, 1, col = "red", lty = 2)
> 
```

<u>ROC and AUC curve:</u>
ROC (receiver operator characteristic) curve is graphical plot used to show the diagnostic ability of binary classifiers. It is constructed by plotting True Positive Rate (TPR) with False Positive Rate (FPR) - TPR: The true positive rate is the proportion of observations that were correctly predicted to be positive out of all positive observations (TP/(TP + FN)).
- FPR: the proportion of observations that are incorrectly predicted to be positive out of all negative observations (FP/(TN + FP))
The ROC curve shows the trade-off between sensitivity (or TPR) and specificity (1 – FPR). Generally, if the curve is closer to the top-left corner, then the classifiers meant that is has better performance (because “True positive is high while false negative is low”), which

is not the case with the graphic above. ROC itself does not depend on the class distribution.



Result: The model achieved accuracy of 90.23 %, but the recall value is very small at 54.6%. Recall measures how good our model is at correctly predicting positive classes.

Step 6 & 7: Naïve Bayes Model and Splitting the data into train and test

Naive Bayes is a probabilistic algorithm that makes predictions based on Bayes' theorem of conditional probability. It is called "naive" because it makes a simplifying assumption that all features used to describe a sample are independent of each other, even though this assumption may not be true in reality.

Despite this limitation, Naive Bayes is a popular algorithm in many classification problems because it is simple, efficient, and often performs well in practice, especially when the number of features is high and the number of training samples is relatively small. Naive Bayes can be trained on labeled data, and once trained, it can quickly classify new samples into one of several predefined classes based on their features.

- Split the data into training and testing sets: Randomly divide the pre-processed data into two sets: a training set and a testing set. The training set is used to train the Naive Bayes model, while the testing set is used to evaluate its performance.
- Train the Naive Bayes model: Use the training set to train the Naive Bayes model.

```
# Split the data into training and testing sets for the Naive Bayes model
set.seed(1234)  # for reproducibility
trainIndex <- createDataPartition(bank$y, p = 0.7, list = FALSE)
bank_nb_train <- bank[trainIndex, ]
bank_nb_test <- bank[-trainIndex, ]

# Train the Naive Bayes model
nb_model <- naiveBayes(y ~ ., data = bank_nb_train)
nb_model
```

Step 8: Evaluate the Naïve Bayes model and Apply test dataset to it

Evaluate the model on the testing set: Use the trained model to make predictions on the testing set and compare them with the true labels to compute various evaluation metrics such as accuracy.

Overall, Naive Bayes can be a useful algorithm for predicting whether a customer subscribes to term deposit in the Bank Marketing dataset, especially if the dataset has many features and not enough samples to train more complex models.

```
# Evaluate the model on the testing set
predictions <- predict(nb_model, newdata = bank_nb_test,na.action = na.pass)
confusionMatrix(predictions, bank_nb_test$y)
```

```
> predictions <- predict(nb_model, newdata = bank_nb_test,na.action = na.pass)
> confusionMatrix(predictions, bank_nb_test$y)
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11101   752
       yes   875   834

               Accuracy : 0.88
                 95% CI : (0.8744, 0.8855)
    No Information Rate : 0.8831
    P-Value [Acc > NIR] : 0.86611

                  Kappa : 0.4381

 Mcnemar's Test P-Value : 0.00249

            Sensitivity : 0.9269
            Specificity : 0.5259
         Pos Pred Value : 0.9366
         Neg Pred Value : 0.4880
             Prevalence : 0.8831
         Detection Rate : 0.8185
   Detection Prevalence : 0.8740
      Balanced Accuracy : 0.7264

       'Positive' Class : no

>
```

Result: This model has 88% accuracy and has good to predict the future datasets.

Step 6 & 7: Random Forest Model and splitting the data into train and test

Random Forest is a popular machine learning algorithm that is used for both classification and regression tasks. It is an ensemble learning technique that combines multiple decision trees to make a final prediction. In a Random Forest, a set of decision trees is generated using a random subset of features and training data. Each decision tree is trained on a randomly selected subset of the training data and a random subset of features. The output of the Random Forest is the majority vote of all the decision trees.

Random Forest is considered to be one of the most accurate and reliable machine learning algorithms due to its ability to handle noisy and high-dimensional data, avoid overfitting, and provide feature importance measures .Random Forest can be used for a wide range of applications, such as fraud detection, image classification, stock price prediction, and customer churn prediction, among others.

Overall, Random Forest is a powerful and versatile algorithm that is widely used in the field of machine learning and data science.

Preprocess the data by removing the 'duration' feature.

We split the data into training and testing sets using createDataPartition() function with 70% for training data and 30% for testing data.

```
#split the data into training and testing for the Random Forest
bank_rf <- subset(bank, select = -c(duration))
set.seed(42) # Set random seed for reproducibility
train_indices_rf <- createDataPartition(bank_rf$y, p = 0.7, list = FALSE)
bank_rf_train <- bank_rf[train_indices_rf, ]
bank_rf_test <- bank_rf[-train_indices_rf, ]

# Creating a random forest classifier with 100 trees
rf_model <- randomForest(y ~ ., data = bank_rf_train, ntree = 100)
rf_model
```

We create a random forest classifier with 100 trees using randomForest() function and fit the model on the training data.

```
Call:
 randomForest(formula = y ~ ., data = bank_rf_train, ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 3

        OOB estimate of  error rate: 10.7%
Confusion matrix:
       no yes class.error
no  27363 583  0.02086166
yes  2805 898  0.75749392
> 
```

Step 8: Evaluate the Random Forest Model and Apply test dataset to it

We then make predictions on the testing data using predict() function, and evaluate the accuracy of the model by comparing predicted values to the actual values using confusionMatrix() function from caret package.

```
# Making predictions on the testing data
y_pred <- predict(rf_model, bank_rf_test)

# Evaluating the accuracy of the model
accuracy <- confusionMatrix(y_pred, bank_rf_test$y)$overall["Accuracy"]
print(paste("Accuracy:", round(accuracy, 4)))
```

226:1 (Top Level)

Console Terminal Background Jobs

R 4.2.2 · D:/Aang/YASHU_FSU HUB/Yashu/SEM2_Spring23/Data Mining/Week 7/

```
> y_pred <- predict(rf_model, bank_rf_test)
> accuracy <- confusionMatrix(y_pred, bank_rf_test$y)$overall["Accuracy"]
> print(paste("Accuracy:", round(accuracy, 4)))
[1] "Accuracy: 0.8912"
>
```

Step 9: Compare the accuracy of three models.

Model	**Accuracy**
Logistic Regression	90.23%
Naive Bayes	88%
Random Forest	89.12%

Conclusion:

Independent variables used in all three models are the same. There might be a difference in the number of customers under training and test datasets (difference in the count of splitting). As, Logistic Regression has an accuracy of 90.23% which is the highest and hence is the best one among three models that are built.

Tasks carried out by each group member:

Task	**Group Member**
Background and Introduction	Mounika Anakanti
Understanding data and Data Pre-processing	Gopala Krishna Karnati
Exploratory Data Analysis	Yasaswini Nallapula
Model building for Logistic Regression	Yasaswini Nallapula
Model building for Naïve bayes	Gopala Krishna Karnati
Model building for Random Forest	Mounika Anakanti
Review, Evaluation and Final Documentation	Yasaswini Nallapula

References:

https://scholarworks.rit.edu/cgi/viewcontent.cgi?article=12514&context=theses

https://www.investopedia.com/terms/t/termdeposit.asp

https://rstudio-pubs-static.s3.amazonaws.com/463779_7be86938710149cbb44633b2466cef7a.html

https://medium.com/analytics-vidhya/a-machine-learning-approach-to-identifying-customers-of-bank-of-portugal-who-would-subscribe-to-a-8bd04387aac2

https://statisticsglobe.com/convert-character-to-factor-in-r

https://rpubs.com/Alvian2022/predicting-term_deposit

https://bookdown.org/ndphillips/YaRrr/logistic-regression-with-glmfamily-binomial.html

https://research.aimultiple.com/machine-learning-accuracy/