Data Science and Analytics Project Report

A Data-Driven Approach to Predict the Success of Direct Marketing Campaign of a Bank

Report Presented by Team 6:

Nalini Kethineni

Theju Chikkathamme Gowda

Rajpal Singh Virk

Mahir Chowdhury

Submitted On: 12/10/2019

Professor: Yanchao Liu

Table of Contents

Abstract

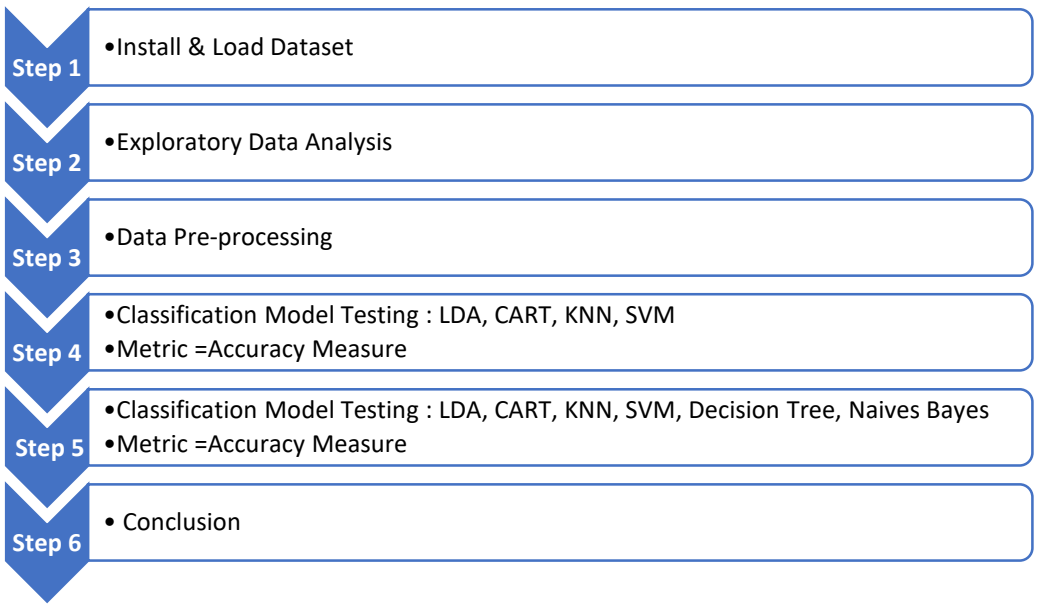
This document uses Data Classification to look at a dataset related with direct advertising efforts of a Portuguese financial organization. The goal of the arrangement is to anticipate if the customer will buy in to a Term Deposit. "Data Classification" is the utilization of Machine Learning systems to compose datasets into related sub-populaces. This can reveal concealed attributes inside information, and recognize hidden categories that new data belongs within.

The Data Science methods utilized inside this examination are Exploratory Data Analysis (EDA), Data Classification and Regression Trees, k-Nearest Neighbours, Support Vector Machines, Decision Tree, Naïve Bayes, and Linear Discriminant Analysis. The above types of examination are consecutively applied to refine the investigation of the Bank Marketing information, so as to decide whether a given bank client will have an affinity to choose a Term Deposit account as the form of saving money.

Introduction/Background

For the campaigning, Organizations rely mainly on either mass or direct campaign. Mass campaign is when a company campaigns with a very large population. Direct campaign is when organizations target a specific list of clients. Studies show that mass campaign is very inefficient and less effective with a 1% positive response. In contrast, the direct campaign focuses on potential client who will more than likely subscribe to the campaign. The issue is, direct campaigns are nearly impossible without the use of data analytics. With data analytics, organizations can scale down potential clients based off of information of the client. For example, a pet store should try to market with clients that own pets rather than the whole population. The bank in question is trying to campaign its new term deposit to potential clients. With data analytics and data classification, we can provide the bank the best model in order to decide which clients will more likely to subscribe to the term deposit. If classifier has very high accuracy, it can help the manager to filter clients and use available resources more efficiently to achieve the campaign goal. Proper strategy would reduce cost and improve long term relations with the clients.

Data Classification Flow Chart/Methodology



Install & Load Dataset/Libraries

Libraries

R-Studio was used in order to determine data classification on the provided dataset. In addition to the basic capabilities of the R programming language, several R packages were downloaded in order to effectively visualize the data. These packages are shown below:

```
# Load Libraries
library(ggplot2)
library(caret)
library(e1071)
library(rpart)
library(rpart.plot)
library(gmodels)
library(rattle)
library(randomForest)
library(caTools)
library(descr)
library(psych)
library(C50)
library(klaR)
library(descr)
library(Metrics)
library(mlbench)
library(gmodels)
library (MASS)
library(gmodels) # For Cross Tables
library(corrplot)
library(lattice)
```

Dataset

The data used in this project was directly taken from UCI Machine Learning Repository. The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Clients were contacted by the bank to market its new product - Term Deposit. Data was then recorded by the bank on the contacted clients and whether the client subscribed to its new product.

There are four datasets:

1) bank-additional-full.csv with all examples (41188) and 20 inputs
2) bank-additional.csv with 10% of the examples (4119), randomly selected from bank-additional-full.csv, and 20 inputs.
3) bank-full.csv with all examples and 17 inputs, ordered by date
4) bank.csv with 10% of the examples and 17 inputs, randomly selected from bank-full.csv

Since the classification goal is to predict if the client will subscribe to a term deposit, Group 6 administered the bank.csv to create the model. Again, these campaigns were based on phone calls. The bank employee would reach out to a client and determine whether a client would like to subscribe to the term deposit. Often, more than one call was administered to the client in order to determine if their product will be subscribed. The bank.csv data set has 17 variables and 4521 observations. 16 out of the 17 variables are input variables and the 17th variable in the data set is the output variable. In depth analysis of the variables is shown below:

```
names(df) # Column names
```

```
##  [1] "X"         "age"       "job"       "marital"   "education" "default"
##  [7] "balance"   "housing"   "loan"      "contact"   "day"       "month"
## [13] "duration"  "campaign"  "pdays"     "previous"  "poutcome"  "y"
```

Bank Client Data

Age (numeric)- Contains an age range from 19-87 years' old

Job (categorical): Type of job

Marital (categorical): Marital status

Education (categorical): Highest earned degree

Default (numeric): Has credit in default?

Balance (numeric): Amount of Balance in bank

Housing (categorical): Has housing loan?

Loan (categorical): Has personal loan?

Related with the Last Contact of the Current Campaign:

Contact (categorical): Contact communication type

Month (categorical): Last contact month of year

Day_of_week (numeric): Last contact day of the week

Duration (numeric): 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.*

Other Attributes:

Campaign (numeric): Number of contacts performed during this campaign and for this client

Pdays (numeric): Number of days that passed by after the client was last contacted from a previous campaign

Previous (numeric): Number of contacts performed before this campaign and for this client

Poutcome (categorical): Outcome of the previous marketing campaign

Output Variable:

Y (categorical) - has the client subscribed a term deposit?

Source Link: http://archive.ics.uci.edu/ml/datasets/Bank+Marketing

Now that the variables of the dataset are outlined, we can start the initial data set classification process. The first step of the classification after the packages are imported is downloading the dataset into R. Before the data set can be downloaded into R, initial data

cleaning occurred on the original .CSV file. The data in the .CSV file was separated in excel through the use of column edit. Once the values were separated in respective column, the data was downloaded to R.

```
df <- read.csv("BankData.csv")
head(df)
```

	0	1	2	3	4
age	30	33	35	30	59
job	unemployed	services	management	management	blue-collar
marital	married	married	single	married	married
education	primary	secondary	tertiary	tertiary	secondary
default	no	no	no	no	no
balance	1787	4789	1350	1476	0
housing	no	yes	yes	yes	yes
loan	no	yes	no	yes	no
contact	cellular	cellular	cellular	unknown	unknown
day	19	11	16	3	5
month	oct	may	apr	jun	may
duration	79	220	185	199	226
campaign	1	1	1	4	1
pdays	-1	339	330	-1	-1
previous	0	4	1	0	0
poutcome	unknown	failure	failure	unknown	unknown
y	no	no	no	no	no

As you can see in the previous page, after the data was downloaded into R, the initial rows of the data set were printed in order to confirm the data downloaded correctly. In the printed dataset, we can see all 17 variables are present but only the first four observations are shown. An 'X' column is added which only contains integers to organize the data set. This column will not affect the data classification.

Exploratory Data Analysis

Reviewing Dataset

```
dim(df) # find the number of observations and variables of dataset
```

```
## [1] 4521   18
```

The imported dataset has 4521 rows and 18 columns. Again, original data has 17 columns and we added an extra column for organization purposes. This column will be taken before analysis models are created.

Checking for Missing Values

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

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

There are no missing data in our dataset. This is because the original data collectors recorded information not given by the client as "unknown." This can be seen when we take a look at the structure of the dataset shown below:

Dataset Structure

```
str(df) # Understanding the structure of dataset
```

```
## 'data.frame':    4521 obs. of  18 variables:
##  $ X         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ age       : int  30 33 35 30 59 35 36 39 41 43 ...
##  $ job       : Factor w/ 12 levels "admin.","blue-collar",..: 11 8 5 5 2 5 7 10 3 8
...
##  $ marital   : Factor w/ 3 levels "divorced","married",..: 2 2 3 2 2 3 2 2 2 2 ...
##  $ education: Factor w/ 4 levels "primary","secondary",..: 1 2 3 3 2 3 3 2 3 1 ...
##  $ default   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ balance   : int  1787 4789 1350 1476 0 747 307 147 221 -88 ...
##  $ housing   : Factor w/ 2 levels "no","yes": 1 2 2 2 2 1 2 2 2 2 ...
##  $ loan      : Factor w/ 2 levels "no","yes": 1 2 1 2 1 1 1 1 1 2 ...
##  $ contact   : Factor w/ 3 levels "cellular","telephone",..: 1 1 1 3 3 1 1 1 3 1
...
##  $ day       : int  19 11 16 3 5 23 14 6 14 17 ...
##  $ month     : Factor w/ 12 levels "apr","aug","dec",..: 11 9 1 7 9 4 9 9 9 1 ...
##  $ duration : int  79 220 185 199 226 141 341 151 57 313 ...
##  $ campaign : int  1 1 1 4 1 2 1 2 2 1 ...
##  $ pdays     : int  -1 339 330 -1 -1 176 330 -1 -1 147 ...
##  $ previous : int  0 4 1 0 0 3 2 0 0 2 ...
##  $ poutcome : Factor w/ 4 levels "failure","other",..: 4 1 1 4 4 1 2 4 4 1 ...
##  $ y         : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
#List of numeric columns
numerical_cols = list(bank.select_dtypes(exclude=['object']))
numerical_cols
```

```
['age', 'balance', 'day', 'duration', 'campaign', 'pdays', 'prev
ious']
```

```
#List of Categorical columns
category_cols = list(bank.select_dtypes(include=['object']))
category_cols
```

```
['job',
 'marital',
 'education',
 'default',
 'housing',
 'loan',
 'contact',
 'month',
 'poutcome',
 'y']
```

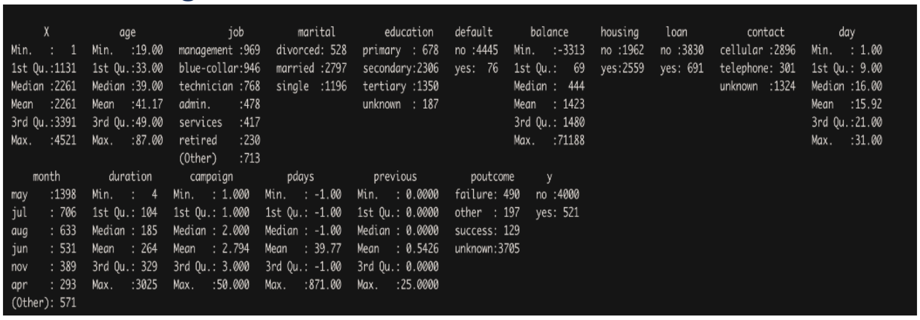
Original data set has 7 columns with quantitative values and 10 columns with qualitative values. The output column is qualitative since to observations is yes/no for the term deposit. Initial EDA of the output is shown below.

```
CrossTable(df$y) # Checking the output variable classes.
```

```
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |-------------------------|
##
## |       no |      yes |
## |----------|----------|
## |     4000 |      521 |
## |    0.885 |    0.115 |
## |----------|----------|
```

Output is a categorical variable of value either “yes” or “no”. Current dataset has 88.5% “no” values and 11.5% “yes” values. From above cross-table, we can see that there are more outputs labelled “no” than “yes”. This indicates that our dataset is imbalanced dataset.

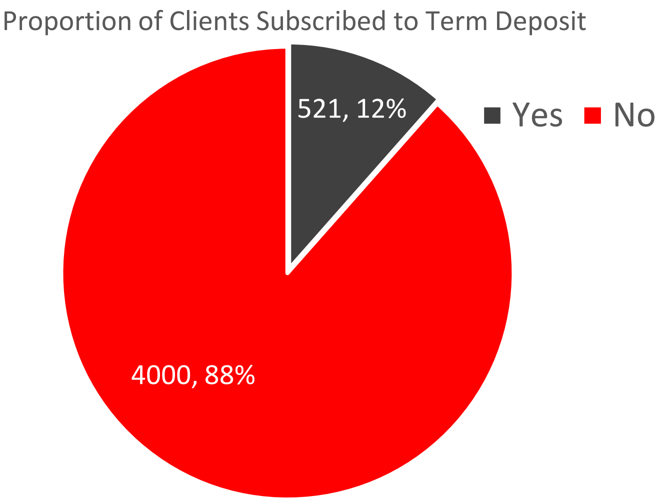
Summarizing Dataset



```
       X              age                  job           marital          education    default      balance       housing    loan           contact           day
 Min.   :   1   Min.   :19.00   management :969   divorced: 528   primary  : 678   no :4445   Min.   :-3313   no :1962   no :3830   cellular :2896   Min.   : 1.00
 1st Qu.:1131   1st Qu.:33.00   blue-collar:946   married :2797   secondary:2306   yes:  76   1st Qu.:   69   yes:2559   yes: 691   telephone: 301   1st Qu.: 9.00
 Median :2261   Median :39.00   technician :768   single  :1196   tertiary :1350              Median :  444                         unknown  :1324   Median :16.00
 Mean   :2261   Mean   :41.17   admin.     :478                   unknown  : 187              Mean   : 1423                                          Mean   :15.92
 3rd Qu.:3391   3rd Qu.:49.00   services   :417                                               3rd Qu.: 1480                                          3rd Qu.:21.00
 Max.   :4521   Max.   :87.00   retired    :230                                               Max.   :71188                                          Max.   :31.00
                                (Other)    :713
     month         duration        campaign          pdays           previous           poutcome      y
 may    :1398   Min.   :   4   Min.   : 1.000   Min.   : -1.00   Min.   : 0.0000   failure: 490   no :4000
 jul    : 706   1st Qu.: 104   1st Qu.: 1.000   1st Qu.: -1.00   1st Qu.: 0.0000   other  : 197   yes: 521
 aug    : 633   Median : 185   Median : 2.000   Median : -1.00   Median : 0.0000   success: 129
 jun    : 531   Mean   : 264   Mean   : 2.794   Mean   : 39.77   Mean   : 0.5426   unknown:3705
 nov    : 389   3rd Qu.: 329   3rd Qu.: 3.000   3rd Qu.: -1.00   3rd Qu.: 0.0000
 apr    : 293   Max.   :3025   Max.   :50.000   Max.   :871.00   Max.   :25.0000
 (Other): 571
```

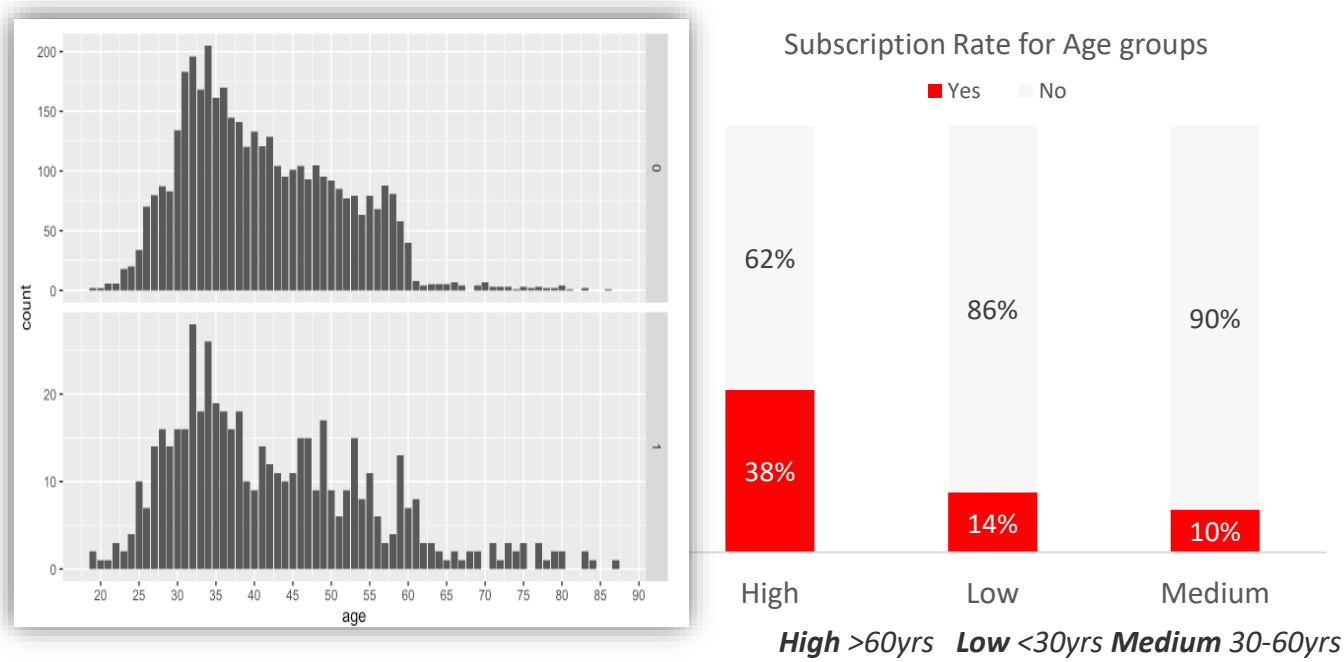
Data Exploration

Output (Y) : What proportion of clients those are contacted actually subscribed to “Term Deposit”?



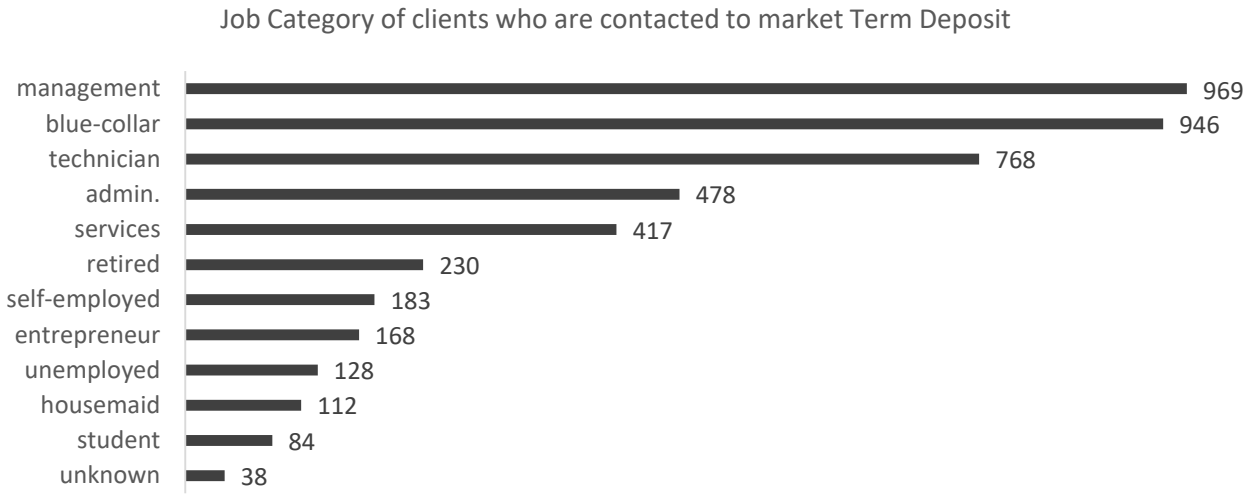
Output is a categorical variable of value either “yes” or “no”. Current dataset has 88.5% “no” values and 11.5% “yes” values. From above Pie graph, we can see that there are more outputs labelled “no” than “yes”. This indicates that our dataset is imbalanced dataset.

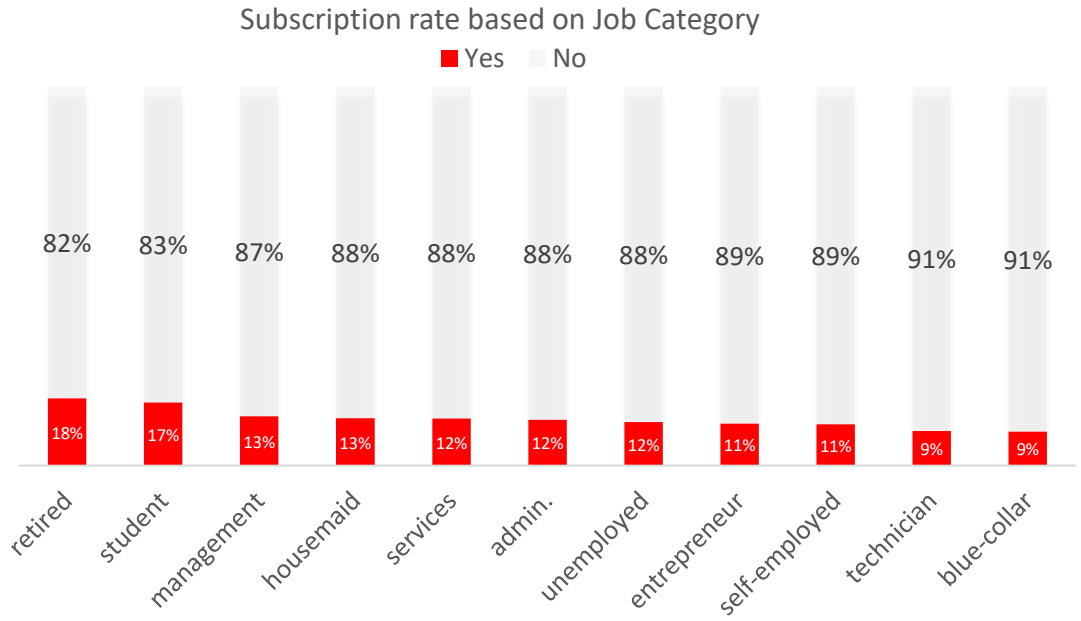
Input (Age): Clients of which age are targeted and which targeted age group has highest subscription rate?



From above plots, it can be visualized that the clients with age 30 years old has highest number of subscriptions. However, this age also has a considerable number of rejection to subscribing. Similar pattern is visible with other age categories. Thus, it would be appropriate to group the ages to collect better insights. The three sections the age was separated into is high, medium, and low. “High” is characterized as clients over the age of 60. “Medium” is characterized as clients between the age of 30 and 60 years old. Finally, “low” is characterized as clients less than 30 years old. We can conclude that less clients with age above 60 are contacted for Term Deposit. But, surprisingly, the subscription rate of clients above age 60 is much higher than that of clients aged between 30 and 60. In other words, clients with age 60 or above are more likely to subscribe to a Term Deposit.

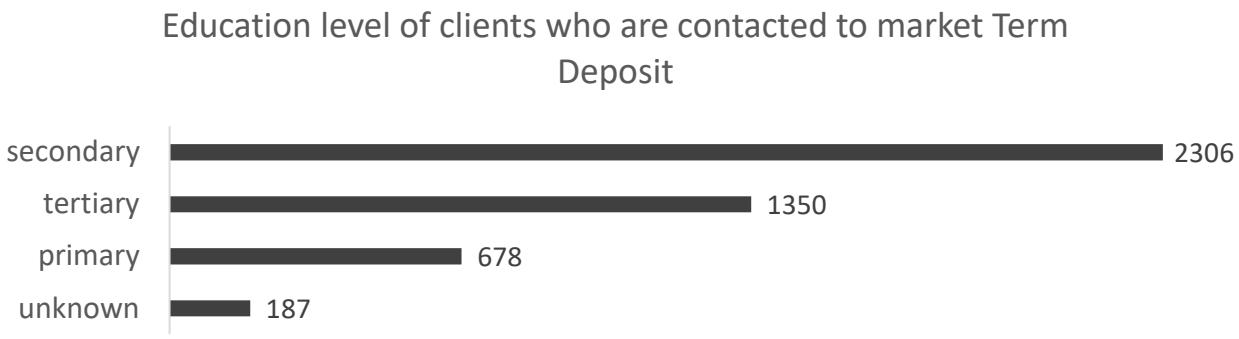
Input (Job): Clients belonging to which job category are contacted and which category resulted in highest subscription?

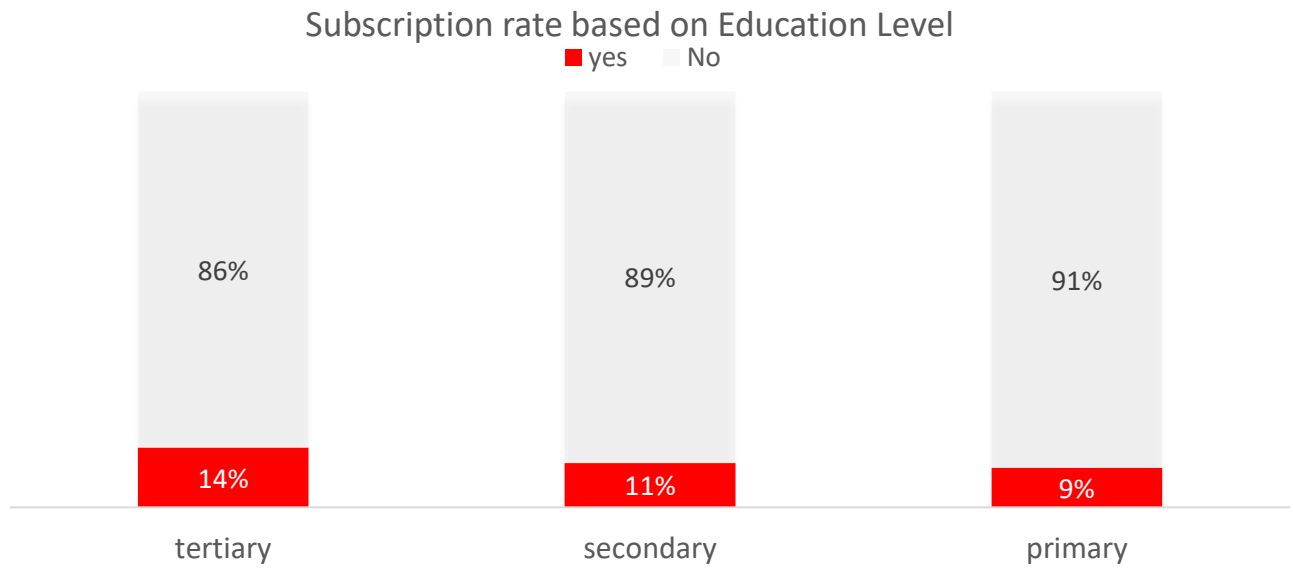




Mostly clients, who are contacted, are in management positions followed by those who are in blue-collar jobs. However, we do have 38 entries with no job information. We can conclude that Retired clients has the highest percentage (23.5%) of subscription rate to term deposit, followed by proportion of Student clients (22.6%).

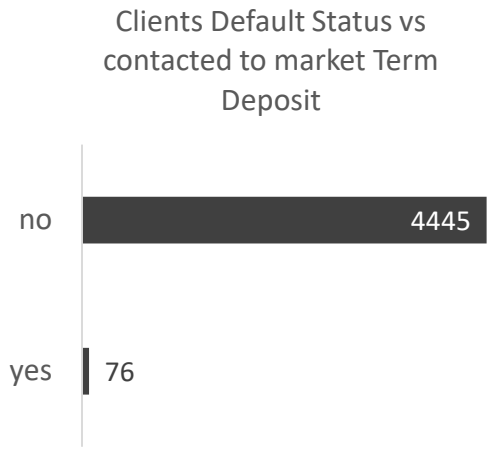
Input (Education Level): Education level of contacted clients and Who had the highest subscription rate?

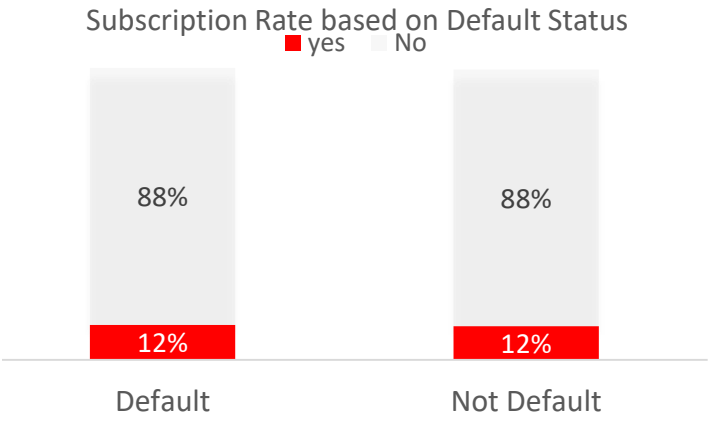




Three key categories of education are - Primary, Secondary and Tertiary education. However, we do have “unknown”, education observation, which is only 4% of the total observations and have only 10% subscription rate. Thus, we can safely remove this unknown entry to visualize the education category. The highest level of education in this category is tertiary. They have the highest subscription rate of 14% followed by clients with the highest level of education as secondary at 11%. Finally, primary education has the lowest percent of subscription at 9%. We can conclude, the higher the education, the more likely they would subscribe.

Input (Credit Default): Does the client have a credit in default and Does it affect subscription?





The proportion of clients who subscribed is same for both clients who have defaulted and clients who have not even though the count of defaulter and non-defaulters have a vast difference.

Pre-processing Dataset

Converting quantitative (integer) values as numeric.

```
col_num<-c(2,7,11,14,15,16,17)
df[,col_num]<-lapply(df[,col_num],as.numeric)
str(df)
```

```
## 'data.frame':    4521 obs. of  18 variables:
##  $ X         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ age       : num  30 33 35 30 59 35 36 39 41 43 ...
##  $ job       : Factor w/ 12 levels "admin.","blue-collar",..: 11 8 5 5 2 5 7 10 3 8
...
##  $ marital   : Factor w/ 3 levels "divorced","married",..: 2 2 3 2 2 3 2 2 2 2 ...
##  $ education: Factor w/ 4 levels "primary","secondary",..: 1 2 3 3 2 3 3 2 3 1 ...
##  $ default   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ balance   : num  1787 4789 1350 1476 0 ...
##  $ housing   : Factor w/ 2 levels "no","yes": 1 2 2 2 2 1 2 2 2 2 ...
##  $ loan      : Factor w/ 2 levels "no","yes": 1 2 1 2 1 1 1 1 1 2 ...
##  $ contact   : Factor w/ 3 levels "cellular","telephone",..: 1 1 1 3 3 1 1 1 3 1
...
##  $ day       : num  19 11 16 3 5 23 14 6 14 17 ...
##  $ month     : Factor w/ 12 levels "apr","aug","dec",..: 11 9 1 7 9 4 9 9 9 1 ...
##  $ duration  : int  79 220 185 199 226 141 341 151 57 313 ...
##  $ campaign  : num  1 1 1 4 1 2 1 2 2 1 ...
##  $ pdays     : num  -1 339 330 -1 -1 176 330 -1 -1 147 ...
##  $ previous  : num  0 4 1 0 0 3 2 0 0 2 ...
##  $ poutcome  : num  4 1 1 4 4 1 2 4 4 1 ...
##  $ y         : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

Normalization of Numeric Variables

Since the numeric value ranges differ from variable to variable, these cannot be used for model training and testing without normalization.

```
normal=function(x)
{
  return((x-min(x))/(max(x)-min(x)))

}
col_list=c(2,7,11,17,14,15,16)
for(i in col_list)
{
  df[(i)]=normal(df[(i)])
  df<-df
}
```

Splitting Dataset

We split dataset in training and testing datasets using 80-20 split ration.

There is a similar split of “no” and “yes” labels for the output variable in both testing and training dataset.

Classification Model Testing/Results

Since response variable is categorical Logistic regression can be applied. Data is very structured and for major columns data is categorical. Decision tree can be more appropriate to exploit sub-feature space i.e. categories. Support vector machine can also be used for classification. Here we will limit our discussion to four models. The following four models will be used and compared with, for class identification.

1. Linear Discriminant Analysis (LDA)
2. Classification and Regression Trees (CART)
3. k-Nearest Neighbours (kNN).
4. Support Vector Machines (SVM) with a linear Kernel

We decided on these four models because they have a mixture of simple linear (LDA), non-linear (CART, kNN), and complex non-linear method (SVM). We will also use a k-fold of value 10 for cross validation.

Model Training for Accuracy Measure

```
control<- trainControl(method="cv", number=10)
metric <- "Accuracy"
# a) linear algorithms
set.seed(7)
fit.lda <- train(y~., data=train_df, method="lda", metric=metric, trControl=control) #Linear Discriminant An
alysis (LDA)

# b) nonlinear algorithms
set.seed(7)
fit.cart <- train(y~., data=train_df, method="rpart", metric=metric, trControl=control) #Classification and
Regression Trees (CART)

set.seed(7)
fit.knn <- train(y~., data=train_df, method="knn", metric=metric, trControl=control) #k-Nearest Neighbors (k
NN)
# c) advanced algorithms
set.seed(7)
fit.svm <- train(y~., data=train_df, method="svmRadial", metric=metric, trControl=control) #Support Vector M
achines (SVM) with a linear kernel
```

Comparing Model Accuracy

This is only applicable to balanced dataset only. For a balanced dataset, we can assess and compare the accuracy of each model and select the one with highest accuracy. Though our data is imbalanced, we have displayed the process below. We have 4 models and accuracy estimations for each model. We need to compare the models to each other and

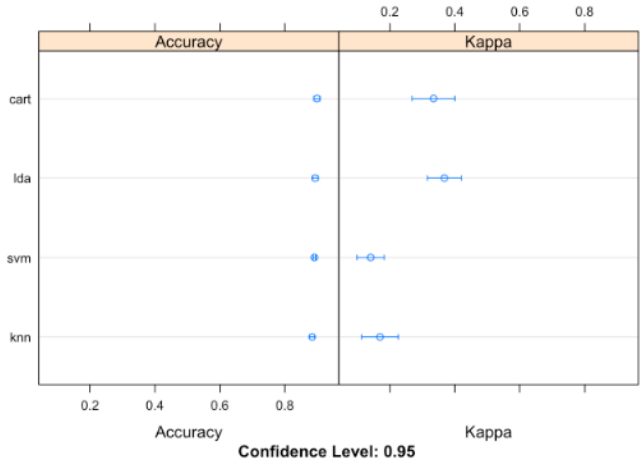
select the most accurate. We can report the accuracy of each model by creating a list of accuracies of each model and using the summary function on this list.

```
# summarize accuracy of models
results <- resamples(list(lda=fit.lda, cart=fit.cart, knn=fit.knn, svm=fit.svm))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: lda, cart, knn, svm
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.8756906 0.8871191 0.8936464 0.8932584 0.8991713 0.9168975    0
## cart 0.8670360 0.9011761 0.9031810 0.8990550 0.9051916 0.9143646    0
## knn  0.8670360 0.8769896 0.8865911 0.8843957 0.8922652 0.8977901    0
## svm  0.8808864 0.8891967 0.8908840 0.8910386 0.8964088 0.9002770    0
##
## Kappa
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.26975977 0.3277196 0.3613232 0.3679431 0.3929645 0.5378115    0
## cart 0.18202417 0.2858638 0.3425735 0.3344502 0.3745100 0.4647525    0
## knn  0.03733333 0.1215080 0.1626139 0.1697975 0.2341429 0.2881844    0
## svm  0.02039018 0.1121136 0.1495257 0.1408284 0.1832320 0.2129510    0
```

We will create a plot of this summary result to evaluate the models. From the plot, we can compare spread and mean accuracy of each model.

```
# compare accuracy of models
dotplot(results)
```



From above plot, we can conclude that, assuming our dataset is balanced, accuracy level is quite similar of each model. However, LDA provides highest Kappa value. In case of imbalanced dataset, we use Confusion Matrix to assess the model performance.

Confusion Matrix

The summary of a confusion Matrix represents Precision (sensitivity) and recall. Model selection based on Confusion Matrix depends on whether the need is to minimize false negatives or false positives.

Minimize false negative: When the actual class is True (1) but model predicts it False (0), then we try to minimize false negative. Aim is to select model which results Recall close to 100% with highest possible precision.

Minimising false positives: When the actual class is False (0) but model predicts it True (1), then we try to minimize false positives. Aim is to select model which results highest possible precision.

In this analysis, our goal is to select model which has lowest false positive rate, i.e. we want to make sure that the selected model shows minimum number of certain clients who actually did not subscribe but model predicted that those clients subscribed. However, at the same time we want to make sure that True positive rate is not too low.

Model Training and Testing(Imbalanced Dataset)/Results

The following six models will be used and compared with, for class identification.

1. Classification and Regression Trees (CART)
2. k-Nearest Neighbours (kNN)
3. Support Vector Machines (SVM) with a linear kernel
4. Decision tree using C5.0 Algorithm (DT)
5. Naive Bayes (NB)
6. Linear Discriminant Analysis (LDA)

The confusion matrix results are show respectively for each. This can be better seen in our R file.

Confusion Matrix - CART

```
confusionMatrix(cart_pred , test_df$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  758  80
##        yes  26  41
##
##               Accuracy : 0.8829
##                 95% CI : (0.8601, 0.9031)
##    No Information Rate : 0.8663
##    P-Value [Acc > NIR] : 0.07647
##
##                  Kappa : 0.3768
##
##  Mcnemar's Test P-Value : 2.635e-07
##
##             Sensitivity : 0.9668
##             Specificity : 0.3388
##          Pos Pred Value : 0.9045
##          Neg Pred Value : 0.6119
##              Prevalence : 0.8663
##          Detection Rate : 0.8376
##    Detection Prevalence : 0.9260
##       Balanced Accuracy : 0.6528
##
##        'Positive' Class : no
##
```

1

Confusion Matrix - kNN

```
confusionMatrix(predictedkNN , test_df$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  766 101
##        yes  18  20
##
##               Accuracy : 0.8685
##                 95% CI : (0.8447, 0.8899)
##    No Information Rate : 0.8663
##    P-Value [Acc > NIR] : 0.4464
##
##                  Kappa : 0.2005
##
##  Mcnemar's Test P-Value : 5.608e-14
##
##             Sensitivity : 0.9770
##             Specificity : 0.1653
##          Pos Pred Value : 0.8835
##          Neg Pred Value : 0.5263
##              Prevalence : 0.8663
##          Detection Rate : 0.8464
##    Detection Prevalence : 0.9580
##       Balanced Accuracy : 0.5712
##
##        'Positive' Class : no
##
```

2

Confusion Matrix - SVM

```
confusionMatrix(SVMPredictions, test_df$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  780 119
##        yes   4   2
##
##               Accuracy : 0.8641
##                 95% CI : (0.84, 0.8858)
##    No Information Rate : 0.8663
##    P-Value [Acc > NIR] : 0.6008
##
##                  Kappa : 0.0191
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99490
##             Specificity : 0.01653
##          Pos Pred Value : 0.86763
##          Neg Pred Value : 0.33333
##              Prevalence : 0.86630
##          Detection Rate : 0.86188
##    Detection Prevalence : 0.99337
##       Balanced Accuracy : 0.50571
##
##        'Positive' Class : no
##
```

3

Confusion Matrix - DT

```
confusionMatrix(dectree_c5_pred, test_df$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  771  89
##        yes  13  32
##
##               Accuracy : 0.8873
##                 95% CI : (0.8649, 0.9072)
##    No Information Rate : 0.8663
##    P-Value [Acc > NIR] : 0.03315
##
##                  Kappa : 0.3375
##
##  Mcnemar's Test P-Value : 1.118e-13
##
##             Sensitivity : 0.9834
##             Specificity : 0.2645
##          Pos Pred Value : 0.8965
##          Neg Pred Value : 0.7111
##              Prevalence : 0.8663
##          Detection Rate : 0.8519
##    Detection Prevalence : 0.9503
##       Balanced Accuracy : 0.6239
##
##        'Positive' Class : no
##
```

4

Confusion Matrix - NB

```
confusionMatrix(NBPredictions, test_df$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  751  87
##        yes  33  34
##
##               Accuracy : 0.8674
##                 95% CI : (0.8436, 0.8888)
##    No Information Rate : 0.8663
##    P-Value [Acc > NIR] : 0.4853
##
##                  Kappa : 0.2945
##
##  Mcnemar's Test P-Value : 1.31e-06
##
##             Sensitivity : 0.9579
##             Specificity : 0.2810
##          Pos Pred Value : 0.8962
##          Neg Pred Value : 0.5075
##              Prevalence : 0.8663
##          Detection Rate : 0.8298
##    Detection Prevalence : 0.9260
##       Balanced Accuracy : 0.6194
##
##        'Positive' Class : no
##
```

5

Model training and testing - LDA

```
lda_fit <- lda(train_df$y~.,data = train_df)
lda_pred <- predict(lda_fit,test_df)
lda.class <- lda_pred$class
```

Confusion Matrix - LDA

```
table(lda.class,test_df$y)
```

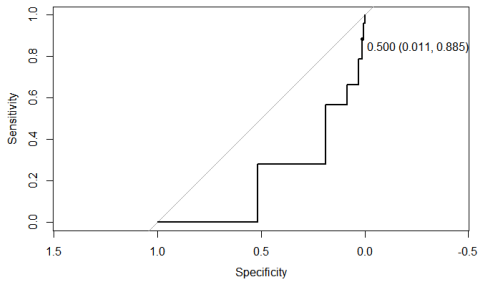
```
##
## lda.class  no yes
##       no  751  72
##       yes  33  49
```

6

Receiver Operating Characteristic Curve

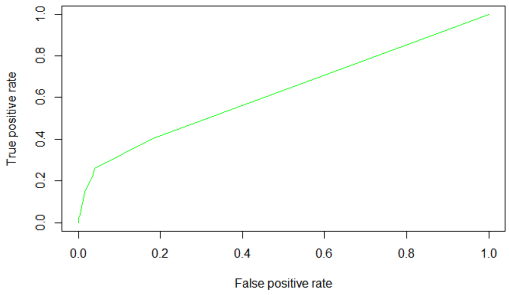
A receiver operating curve, or ROC curve, is a plot that illustrates the diagnostic ability of a classifier system as its threshold is varied. Due to our project being a “data classification” project, we created an ROC curve for each of the tests to determine the best model to sue to predict if a client would subscribe to a “term deposit.”

ROC-KNN



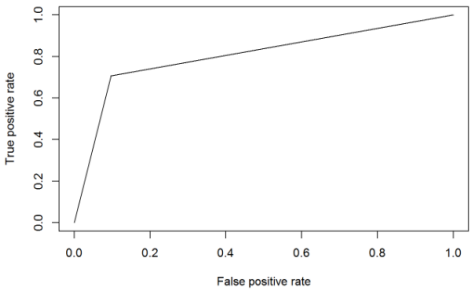
Area under the curve is approximately 0.335 meaning that the model is actually reciprocating the classes. The model is predicting negative class as a positive class and vice versa. It has no discrimination capacity to distinguish between positive and negative class.

ROC-CART



ROC curve on the CART model suggests that the model may not be the best but it is still good for prediction.

ROC-LDA



The AUC here is 0.81 which means there is an 81% chance that the model will be able to distinguish between positive class and negative class. In conclusion the ROC curves tell us the best models for prediction would be between the CART model and the LDA model. In order to determine which, one is best we took a look at the highest F1 score to decide.

Conclusion

```
Model <- c("CART","KNN","SVM","Decision Tree","Naive Bayes","LDA")
FP <- c(26,18,4,13,33,31)
FN <- c(80,101,119,89,87,78)
TP <- c(41,20,2,32,34,43)
TN <- c(758,766,780,771,751,753)
Sensitivity = TP / (TP + FN)
Specificity = TN / (TN + FP)
Precision = TP / (TP + FP)
FPR <- 1- (TN/(TN+FP)) #1-specificity
Accuracy <- (TP + TN) / (TP + TN + FP + FN)
F1 = 2 * (Precision * Sensitivity) / (Precision + Sensitivity)
Eval_Table <- data.frame(Model,TN,FP,FN,TP,Sensitivity,Specificity,Precision,FPR, Accuracy,F1)
Eval_Table
```

```
##           Model  TN FP  FN TP Sensitivity Specificity Precision         FPR
## 1          CART 758 26  80 41  0.33884298   0.9668367 0.6119403 0.033163265
## 2           KNN 766 18 101 20  0.16528926   0.9770408 0.5263158 0.022959184
## 3           SVM 780  4 119  2  0.01652893   0.9948980 0.3333333 0.005102041
## 4 Decision Tree 771 13  89 32  0.26446281   0.9834184 0.7111111 0.016581633
## 5   Naive Bayes 751 33  87 34  0.28099174   0.9579082 0.5074627 0.042091837
## 6           LDA 753 31  78 43  0.35537190   0.9604592 0.5810811 0.039540816
##    Accuracy         F1
## 1 0.8828729 0.43617021
## 2 0.8685083 0.25157233
## 3 0.8640884 0.03149606
## 4 0.8872928 0.38554217
## 5 0.8674033 0.36170213
## 6 0.8795580 0.44102564
```

Model <fctr>	TN <dbl>	FP <dbl>	FN <dbl>	TP <dbl>	Sensitivity <dbl>	Specificity <dbl>	Precision <dbl>	FPR <dbl>	Accuracy <dbl>	F1 <dbl>
CART	758	26	80	41	0.33884298	0.9668367	0.6119403	0.033163265	0.8828729	0.43617021
KNN	766	18	101	20	0.16528926	0.9770408	0.5263158	0.022959184	0.8685083	0.25157233
SVM	780	4	119	2	0.01652893	0.9948980	0.3333333	0.005102041	0.8640884	0.03149606
Decision Tree	771	13	89	32	0.26446281	0.9834184	0.7111111	0.016581633	0.8872928	0.38554217
Naive Bayes	751	33	87	34	0.28099174	0.9579082	0.5074627	0.042091837	0.8674033	0.36170213
LDA	753	31	78	43	0.35537190	0.9604592	0.5810811	0.039540816	0.8795580	0.44102564

The lowest possible false positive rate is predicted by Support Vector Machine (SVM) model. However, the Precision is very low in case of SVM. F1 Score, which is another important measure to assess the model, is high for Linear Discriminant Analysis (LDA). Hence, we'll select Linear Discriminant Analysis (LDA) model for future predictions for Term deposit subscriptions. Notes: F1 score summarizes both precision and recall. An F1 score of 1 indicates perfect precision and recall, therefore the higher the F1 score, the better the model.

References

Dean F. Amel and Martha Starr-McCluer, “Market Definition in Banking: Recent Evidence,” The Antitrust Bulletin 47 (2002), pp. 63-89.

“Financial Services Used by Small Businesses: Evidence from the 1998 Survey of Small Business Finances,” Federal Reserve Bulletin 87 (2001), pp 183-205.

[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

Myron L. Kwast, Martha Starr-McCluer and John D. Wolken, “Market Definition and the Analysis of Antitrust in Banking,” The Antitrust Bulletin, 44 (1997), pp. 973-995, and Marianne P. Bitler, Alicia M. Robb and John D. Wolken.