University for the Common Good

Coursework

Module Ref ID - *MHI224186-19-A*

Big Data

Student name: Kone Fanhatcha

Student ID: S1803435

"I declare that all work submitted for this coursework is the work of **Kone Fanhatcha** of my own except where explicitly stated otherwise."

Table of Contents

I. INTRODUCTION

Through this coursework report, we will do an in-depth analysis of a customer information dataset for a **bank(Bank dataset)** using **Logistic regression as a classifier.**
Indeed, this document begins with a description of the problem we have to investigate. Then we will build our classification-based collaborative filtering model with python. After this stage of construction, we will test the accuracy of our predictions. And finally, we'll have a discussion about the overall process we went through during this study.

II. DESCRIPTION OF THE PROBLEM

A. Overall description.

We have been hired as Marketing data scientists by MKL Holding which is a well-established financial institution in the UK. Our main goal is to decide whether one of the existing clients is a good candidate for a special term deposit offer that the bank is currently running. If the client is a good candidate then we will put his name on a list and a bank representative will reach out and make him the offer.
We will be using for this study **logistic regression** as a classifier in order to decide whether to recommend the client for the marketing outreach call.

B. Description of the dataset

The original dataset that has been provided by MKL holding is described in table 0 and 1 with the filename *original_bank_data.csv*

Table 0: Bank dataset overview

Dataset Property	**Value**
Title	Bank Dataset
File format	CSV
Number of instances	15111

Number of attributes	16 + class
Contains missing values	Yes
Number of missing values	464
Class	“subscribed”
Class distribution	“Yes”: 3106

Table 1: Description of attributes|Original dataset

Name	**Type**	**Description**	**Number of missing values**
age	Numeric	The age of the client	0
job	Categorical	This is the type of job the client has, and it is one of the following values: "admin.", "unknown", "unemployed", "management", "housemaid", "entrepreneur", "student", "blue-collar", "self-employed", "retired", "technician", "services"	69
marital	Categorical	This describes the marital status of the client, and is selected from the following list of values: "married", "divorced", "single". The value “ divorced ” means that the client is either divorced or widowed.	12
education	Categorical	This describes the level of education that the client has. Values	31

		are selected from the following: "unknown", "secondary", "primary", "tertiary"	
default	Binary	This attribute describes whether or not the client has credit in default, and this is stored as a binary with “yes” or “no”	51
balance	Numeric	This describes the average yearly balance of the client and the value is stored in euros.	1
housing	Binary	This attribute describes whether the client has a housing loan and this is stored as a binary with “yes” or “no”	15
loan	Binary	This attribute describes whether the client has a personal loan and this is stored as a binary with “yes” or “no”	0
contact	Categorical	This describes the communication type by which the client can be reached. The values are either of the following three: "unknown","telephone", "cellular"	30
day	Numeric	This number refers to the day of the month when the client was last contacted. The number,	123

		therefore, should be in the range 1-31.	
month	Categorical	This refers to the month of the year when the client was last contacted. The month is stored as a string chosen from the twelve months shortened(e.g “ jan ”, “ feb ”, “ march ”, etc)	112
duration	Numeric	This describes the duration of the last contact with the client, and it is stored in seconds	0
campaign	Numeric	This describes the number of contacts performed for a given client in a given campaign including the last contact	0
pdays	Numeric	This stores the number of days that have passed since the client was last contacted, and that number is -1 if the client was never contacted before.	20
previous	Numeric	This describes the number of contacts performed for a given client in before the current campaign	0
poutcome	categorical	This describes the outcome of the marketing campaign, and the value is chosen between the	0

		following: "unknown","other","failure","success"	
subscribed	Binary	This describes whether the client has subscribed to a term deposit and this is stored as a binary with “yes” or “no”. This is also the feature that the model will aim to predict.	0

III. DATA COLLECTION

The data collection process consists of importing the necessary libraries which are going to be used during our analysing. **Figure 1**

- The numpy library stands for numerical python and it is widely used to perform any scientific computation
- Pandas will be used for data analysis
- Seaborn for statistical plotting
- Matplotlib for plotting
- %matplotlib inline is used to be able to run matplotlib in Jupyter notebook
- Math is just for some basic mathematical operations

We libraries outside the red box will be explained in the training & testing section of this report.

Then we import our dataset using the read_csv() function from the pandas libraries and print out the first 10 instances of our dataset

I. Collection the data

Collect data: Import Librairies

```
In [58]:
# Importing the necessary Python libraries

import numpy as np #numerical python library
import pandas as pd #for data analysis
import seaborn as sns #statistical plotting
import matplotlib.pyplot as plt #for plotting

#to be able to run matplotlib in jupiter
%matplotlib inline

import math #calculate basic mathematical functions

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score


# Loading the original bank dataset in a dataframe
original_data = pd.read_csv('original_bank_data.csv')

#A quick overview of the first 10 records
original_data.head(10)
```

Figure 1

Out[85]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	subscr
0	58	management	married	tertiary	no	2143.0	yes	no	unknown	5.0	may	261	1	-1.0	0	unknown	
1	44	technician	single	secondary	no	29.0	yes	no	unknown	5.0	may	151	1	-1.0	0	unknown	
2	33	entrepreneur	married	secondary	no	2.0	yes	yes	unknown	5.0	may	76	1	-1.0	0	unknown	
3	47	blue-collar	married	unknown	no	1506.0	yes	no	unknown	5.0	may	92	1	-1.0	0	unknown	
4	33	unknown	single	unknown	no	1.0	no	no	unknown	5.0	may	198	1	-1.0	0	unknown	
5	35	management	married	tertiary	no	231.0	yes	no	unknown	5.0	may	139	1	-1.0	0	unknown	
6	28	management	single	tertiary	no	447.0	yes	yes	unknown	5.0	may	217	1	-1.0	0	unknown	
7	42	entrepreneur	divorced	tertiary	yes	2.0	yes	no	unknown	5.0	may	380	1	-1.0	0	unknown	
8	58	retired	married	primary	no	121.0	yes	no	unknown	5.0	may	50	1	-1.0	0	unknown	
9	43	technician	single	secondary	no	593.0	yes	no	unknown	5.0	may	55	1	-1.0	0	unknown	

Figure 2

Finally, we will echo out the number of clients in the original dataset

```
In [87]:
#Number of clients in this dataset
print("Number of clients in original dataset:" + str(len(original_data.index)))
```

```
Number of clients in original dataset:15111
```

Figure 3

To summarize this section, we have collected the data, imported all the necessary libraries, and find out the total number of clients in the dataset. Now let's dive deep into the dataset by doing an in-depth analysis of the dataset

IV. ANALYZING DATA

This section consists of creating different plots to check the relationship between variables. In short, we would like to know how one variable in affecting the others.

A. Target column:subscribed

The ***subscribed*** column (reference) will be considered throughout the rest of this report as the target column as we want to know whether the client is going to **subscribe** or not to the special term deposit.

B. Data Analysis

- **Observation 1:** From figure 4, we can see that a lot of clients did not subscribe to the term deposit offer. Also, the divorcees were the ones who subscribed to the least to the offer. (Figure 5)

```
In [88]: #Number of clients in our dataset who subscribed or not to past term deposit offer
         sns.countplot(x="subscribed", data=original_data)
Out[88]: <matplotlib.axes._subplots.AxesSubplot at 0x13d84ed3320>
```

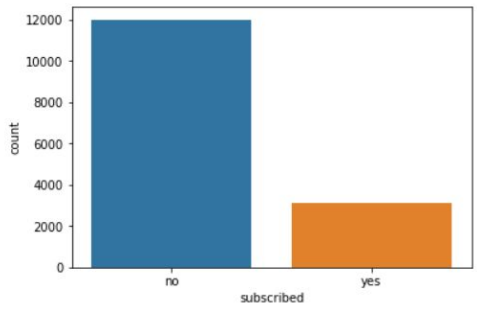


Figure 4

```
In [90]: #Marital status of the clients who subscribed or not
         sns.countplot(x="subscribed", hue="marital", data=original_data)
Out[90]: <matplotlib.axes._subplots.AxesSubplot at 0x13d86211e48>
```

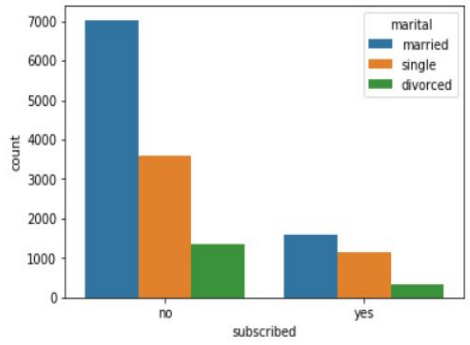


Figure 5

- **Observation 2:** Clients working in management tend to subscribed the most the term deposit (figure 6) as well as the one you got a least secondary education level(figure 7)

```
In [91]: #Job of the clients who subscribed or not
         sns.countplot(x="subscribed", hue="job", data=original_data)
```

```
Out[91]: <matplotlib.axes._subplots.AxesSubplot at 0x13d862825f8>
```

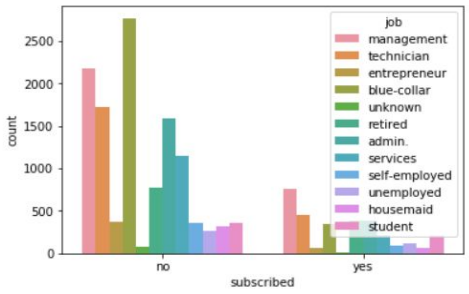


Figure 6

```
In [92]: #Level of education or the clients who subscribed or not
         sns.countplot(x="subscribed", hue="education", data=original_data)
```

```
Out[92]: <matplotlib.axes._subplots.AxesSubplot at 0x13d86282160>
```

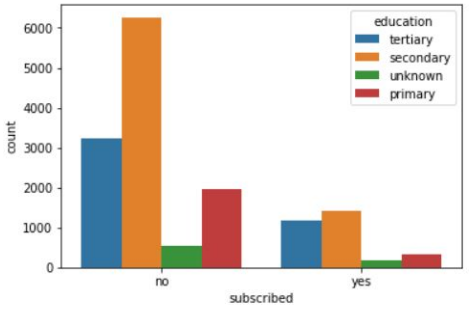


Figure 7

- **Observation 3:** From this observation, we have noticed that the clients who did not subscribe to the term deposit did not have either a credit in default, or personal loan. (Figure 8, 9). However, there a slightly high number of clients who did not subscribe to the term deposit already have a housing loan (figure 10)

```
In [96]: # does the clients who subscribed or not has a credit in default ?
         sns.countplot(x="subscribed", hue="default", data=original_data)
```

```
Out[96]: <matplotlib.axes._subplots.AxesSubplot at 0x13d864b86a0>
```

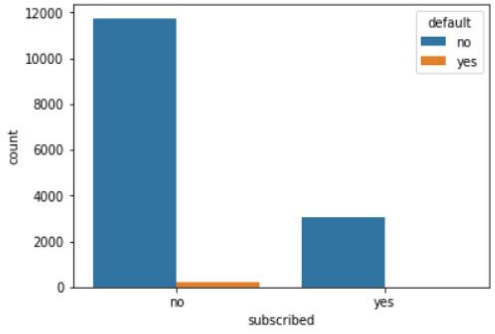


Figure 8

```
In [93]: # does the clients who subscribed or not has a loan personal ?
         sns.countplot(x="subscribed", hue="loan", data=original_data)
Out[93]: <matplotlib.axes._subplots.AxesSubplot at 0x13d8637eb70>
```

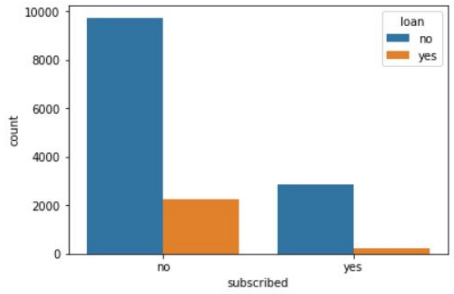


Figure 9

```
In [95]: # does the clients who subscribed or not has a housing loan ?
         sns.countplot(x="subscribed", hue="housing", data=original_data)
Out[95]: <matplotlib.axes._subplots.AxesSubplot at 0x13d8645f198>
```

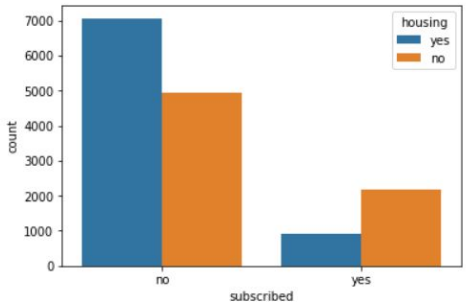


Figure 10

- **Conclusion of the observations:** Divorcees tend to subscribe the least to the term deposit as well as clients who do not have any credit in default and personal loan. However, Clients who already have a housing loan and working in management might be subscribed to the offer. As a result, it will be better to target clients with such a profile. Let's use more advanced modeling tools to verify this assumption

V. DATA WRANGLING

Data wrangling consists of cleaning the data by removing the NaN values and unnecessary columns in the dataset

A. Data cleaning

- The first step in our data cleaning is to check if there are any NaN values and other types of missing values in our dataset.

```
In [65]:
#Identifying missing values
missing_val = original_data.isnull().sum()

print("Total number of missing values is: " + str(missing_val.sum()) + "\n")
print("Find below the missing values for each attributes:")

missing_val
```

```
Total number of missing values is: 464
```

Figure 11

```
Find below the missing values for each attributes:

Out[65]: age                  0
         job                 69
         marital             12
         education           31
         default             51
         balance              1
         housing             15
         loan                 0
         contact             30
         day                123
         month              112
         duration             0
         campaign             0
         pdays               20
         previous             0
         poutcome             0
         subscribed           0
         binary_subscribed    0
         dtype: int64
```

Figure 12

This shows us the reference in our dataset which has missing values. Let's us a heatmap to better visualize it.

```
In [97]:  #Visualization of missing values through heatmap
          sns.heatmap(original_data.isnull(), yticklabels=False, cmap="viridis")

Out[97]: <matplotlib.axes._subplots.AxesSubplot at 0x13d8654cd68>
```

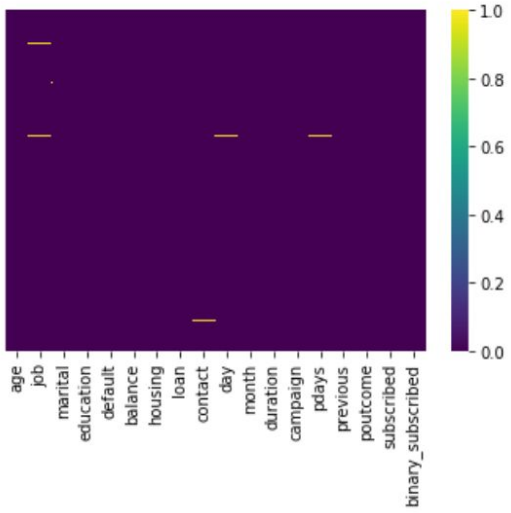


Figure 13

- The second step in cleaning process will be dropping the columns which as NaN values

```
In [99]:  #Drop missing values (NaN)
          original_data.dropna(inplace=True)

In [100]: #Check whether we still have missing(NaN) values or not

          sns.heatmap(original_data.isnull(), yticklabels=False, cmap="viridis")

Out[100]: <matplotlib.axes._subplots.AxesSubplot at 0x13d8661f1d0>
```

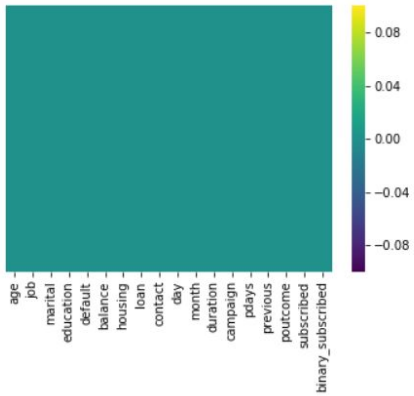


Figure 14

- Heatmap before and after cleaning the dataset

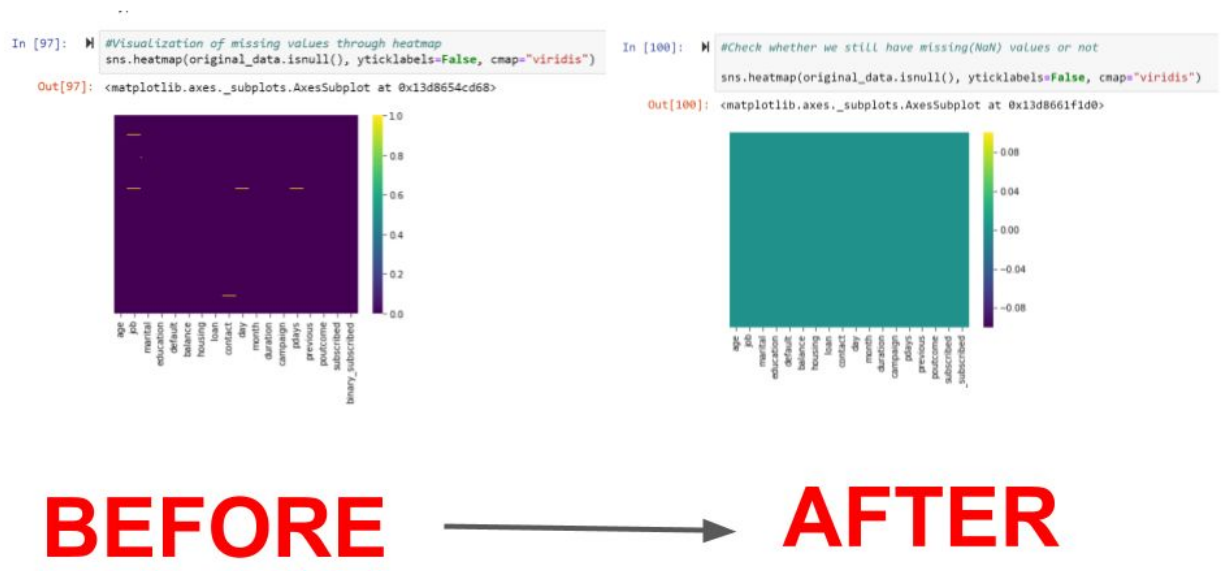


Figure 15

As we can see there are no missing values

```
In [65]:  #This confirms that our dataset is clean

          original_data.isnull().sum()

Out[65]:  age           0
          job           0
          marital       0
          education     0
          default       0
          balance       0
          housing       0
          loan          0
          contact       0
          day           0
          month         0
          duration      0
          campaign      0
          pdays         0
          previous      0
          poutcome      0
          subscribed    0
          dtype: int64
```

Figure 16

Before moving forward in our data wrangling, let's decide which algorithm we will use to build our model. By identified right now the algorithm, we can prepare our dataset consequently to the dataset preparation of that specific algorithm.

B. Overview of machine learning

The first step in identifying which type of machine learning algorithm to be used during our modeling is to identify the different types of machine learning. In fact, machine learning can be divided into 3 main categories:

1. *Supervised learning*: Which consists of giving the machine sets of labelled data and telling the algorithm how the output must or should look like.
2. *Unsupervised learning:* Consists of giving the machine sets of data and letting the machine figured out the output and hidden patterns.
3. *Reinforcement learning:* : The machine is exposed to an environment where it trains itself continually using trial and error according to analyticsvidhya.com**.**

The tree (figure 17) below give us a better understanding of machine learning.
For the scope of this report, we will focus only on supervised learning as we will give some data to our model then show it how the result has to look like. Find more details about this approach in the train and test section of this report.

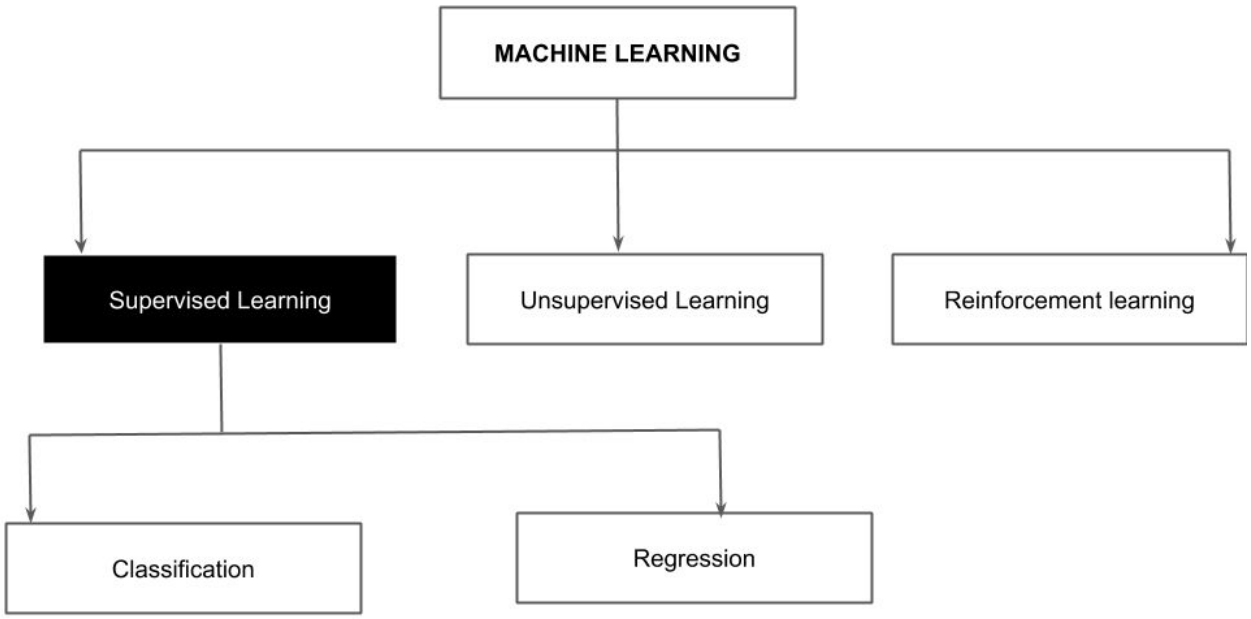


Figure 17

Supervised Machine Learning algorithms solve two(2) major type of problems namely: regression and classification problems

1. **Classification Problems:** Deals with predicting a discrete class label.
    a. *Binary Classification*

    E.g: Predict if an email is going to be spam or Not
    b. *Multi-class classification*

    E.g: Classify if an email will be classed as spam, not spam, or draft.

2. **Regression Problems:** Deals with predicting a continuous quantity.
   E.g: Predict someone's weight or stock price.

*From these definitions we can clearly conclude that we are dealing with a **binary classification problems.** In fact, we want to see if the client is going to subscribe or not to the term deposit.*

C. Logistic Regression as a classifier

Before diving into Logistic regression, let's first of all, understand what regression is exactly:

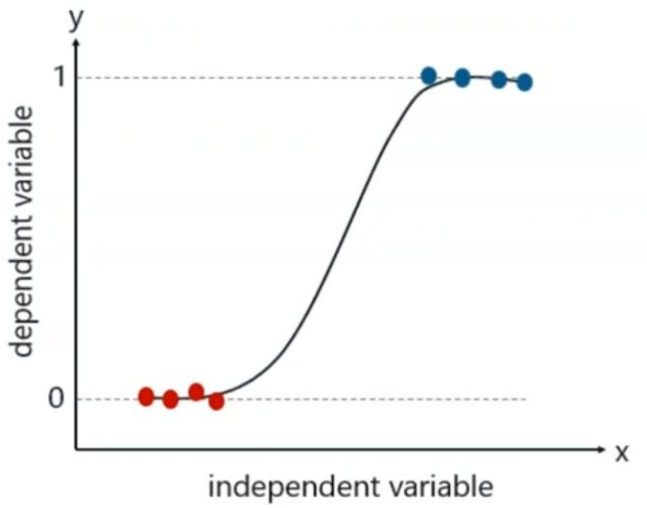
- Regression: It is predictive modeling technique that estimates the relationship between a dependent (target) and an independent variable(predictor). (edureka.co)
- Regression is divided into 3 major categories namely: linear regression, Logistic regression, and Polynomial regression. Let's have a look at the two(2) most widely used regression analysis methods:
    - Linear regression: is a method to predict the dependent variable(Y) based on the values of independent variables (X). It can be used for the cases where we want **to predict some continuous quantity** (edureka.co)
    - Logistic regression: is a method used to predict a dependent variable, given a set of independent variables, such that **the dependent variable is categorical** (binary type of variable)

*As we want to predict whether the client will subscribe or not to the offer, **Logistic regression** came to us as the appropriate binary classification method to used.However we could have used other classification algorication such as Support vector machine (SVM), Artificial Neural Network (ANN), Decision Tree etc.*

A logistic regression classifier can be model using the following equation:

$$\log\left(\frac{Y}{1-Y}\right) = C + B1X1 + B2X2 + \ldots.$$

This function gives us the following graphical representation.



- The Sigmoid curve in this graph is mapping the independent and dependant variables
- By looking at this graph, we first noticed that it is not a straight linear, in fact, the shape is a type of (S). This type of curve is called the Sigmoid curve.
- The Sigmoid curve in this graph is mapping the independent and dependent variables
- This curve determine the likelihood or probability for an event to occur. It determine

To sum the logistic regression algorithm will help us to determine the probability that the client subscribe or not to the term deposit. This result value will be set between 0 - 1.
If the value is between 0 to 0.5 it means that the client will not subscribe. Else if the is between 0.5 to 1 it means the client will subscribe **Figure 18**

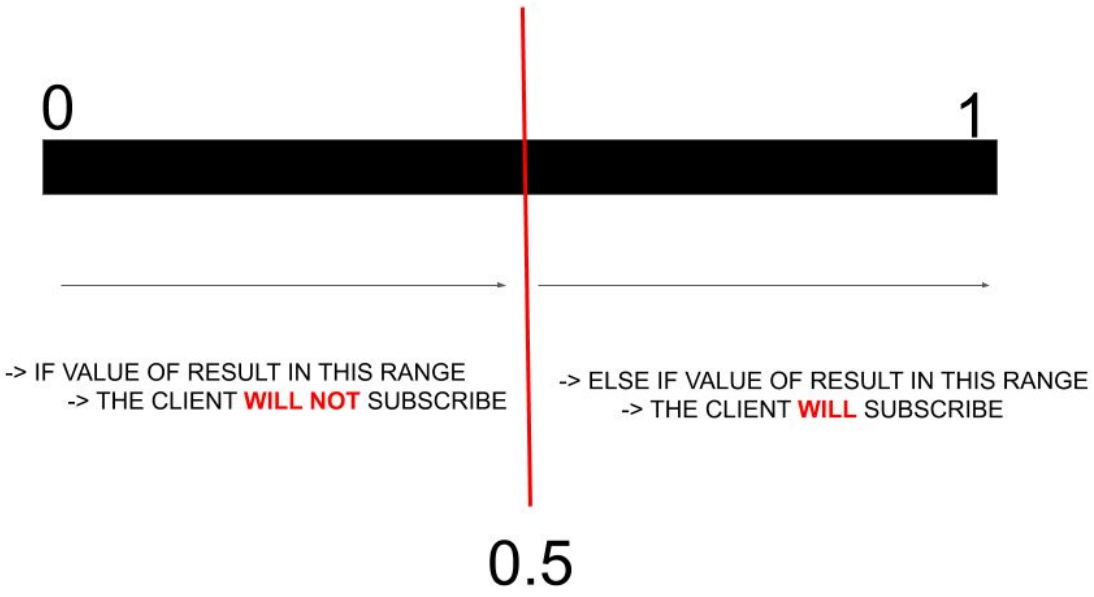


Figure 18

As we know now which algorithm we will used to model our data, let's continue our data wrangling process.

In fact, we will convert the **non-numerical** values in our database to **numerical** values and we will use both b**inary encoding and one hot encoding**

- First of all we manually converted our target(**subscribed**) variable to binary value (0 -1). We have applied the following excel formula to do it: =IF(Q2="yes",1,IF(Q2="no",0)) . However this could also be done using the following python function from the pandas library: **pd.Series(np.where(original_data.subscribed.values == 'yes', 1, 0), sample.index)**

subscribed	binary_subscribed
no	0
no	0
no	0
no	0
no	0
no	0
no	0
no	0
no	0
no	0

Figure 19

- The second process was deleting redundant columns and keep only the necessary attributes to perform the analysis.

In [22]:
```
#Marital status

marital_status_1 = pd.get_dummies(original_data['marital'])
marital_status_1.head(5)

#if client is not married or divorced that means his single. So let's drop the single column
marital_status_2 = marital_status_1.drop(columns=['single'])
marital_status_2.head(5)
```

Out[22]:

	divorced	married
0	0	1
1	0	0
2	0	1
3	0	1
4	0	0

In [23]:
```
#Education level

#if client level of education is not primary, secondary or tertiary that means his education's level is unknown.
#Let's drop the unknown column
education_level = pd.get_dummies(original_data['education']).drop(columns=['unknown'])
education_level.head(5)
```

Figure 19

- The third first was concatenating all the new references we have created with the original dataset

3. Concatenate all the new references into the original dataset

```
In [85]: original_data = pd.concat([original_data, job, marital_status_2, has_credit_default, education_level, has_housing_loan, has_p
         original_data.head(10)
```

Out[85]:

marital	education	default	balance	housing	loan	contact	day	...	primary	secondary	tertiary	yes	yes	cellular	telephone	failure	other	success
married	tertiary	no	2143.0	yes	no	unknown	5.0	...	0	0	1	1	0	0	0	0	0	0
single	secondary	no	29.0	yes	no	unknown	5.0	...	0	1	0	1	0	0	0	0	0	0
married	secondary	no	2.0	yes	yes	unknown	5.0	...	0	1	0	1	1	0	0	0	0	0
married	unknown	no	1506.0	yes	no	unknown	5.0	...	0	0	0	1	0	0	0	0	0	0
single	unknown	no	1.0	no	no	unknown	5.0	...	0	0	0	0	0	0	0	0	0	0
married	tertiary	no	231.0	yes	no	unknown	5.0	...	0	0	1	1	0	0	0	0	0	0
single	tertiary	no	447.0	yes	yes	unknown	5.0	...	0	0	1	1	1	0	0	0	0	0
divorced	tertiary	yes	2.0	yes	no	unknown	5.0	...	0	0	1	1	0	0	0	0	0	0
married	primary	no	121.0	yes	no	unknown	5.0	...	1	0	0	1	0	0	0	0	0	0
single	secondary	no	593.0	yes	no	unknown	5.0	...	0	1	0	1	0	0	0	0	0	0

Figure 20

3. Concatenate all the new references into the original dataset

```
In [85]: original_data = pd.concat([original_data, job, marital_status_2, has_credit_default, education_level, has_housing_loan, has_p
         original_data.head(10)
```

Out[85]:

marital	education	default	balance	housing	loan	contact	day	...	primary	secondary	tertiary	yes	yes	cellular	telephone	failure	other	success
married	tertiary	no	2143.0	yes	no	unknown	5.0	...	0	0	1	1	0	0	0	0	0	0
single	secondary	no	29.0	yes	no	unknown	5.0	...	0	1	0	1	0	0	0	0	0	0
married	secondary	no	2.0	yes	yes	unknown	5.0	...	0	1	0	1	1	0	0	0	0	0
married	unknown	no	1506.0	yes	no	unknown	5.0	...	0	0	0	1	0	0	0	0	0	0
single	unknown	no	1.0	no	no	unknown	5.0	...	0	0	0	0	0	0	0	0	0	0
married	tertiary	no	231.0	yes	no	unknown	5.0	...	0	0	1	1	0	0	0	0	0	0
single	tertiary	no	447.0	yes	yes	unknown	5.0	...	0	0	1	1	1	0	0	0	0	0
divorced	tertiary	yes	2.0	yes	no	unknown	5.0	...	0	0	1	1	0	0	0	0	0	0
married	primary	no	121.0	yes	no	unknown	5.0	...	1	0	0	1	0	0	0	0	0	0
single	secondary	no	593.0	yes	no	unknown	5.0	...	0	1	0	1	0	0	0	0	0	0

Figure 22

- Finally, we dropped all the non-numerical reference to keep only the numerical references

```
In [86]: #Dropping the irrelevant columns

original_data.drop(['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact', 'month', 'poutcome', 'subscribed'
```

```
In [33]: original_data.head(10)
```

Out[33]:

	age	balance	day	duration	campaign	pdays	previous	binary_subscribed	blue-collar	entrepreneur	...	primary	secondary	tertiary	yes	yes	cellula
0	58	2143.0	5.0	261	1	-1.0	0	0	0	0	...	0	0	1	1	0	
1	44	29.0	5.0	151	1	-1.0	0	0	0	0	...	0	1	0	1	0	
2	33	2.0	5.0	76	1	-1.0	0	0	0	1	...	0	1	0	1	1	
3	47	1506.0	5.0	92	1	-1.0	0	0	1	0	...	0	0	0	1	0	
4	33	1.0	5.0	198	1	-1.0	0	0	0	0	...	0	0	0	0	0	
5	35	231.0	5.0	139	1	-1.0	0	0	0	0	...	0	0	1	1	0	
6	28	447.0	5.0	217	1	-1.0	0	0	0	0	...	0	0	1	1	1	
7	42	2.0	5.0	380	1	-1.0	0	0	0	1	...	0	0	1	1	0	
8	58	121.0	5.0	50	1	-1.0	0	0	0	0	...	1	0	0	1	0	
9	43	593.0	5.0	55	1	-1.0	0	0	0	0	...	0	1	0	1	0	

Figure 23

```
, 'education', 'default', 'housing', 'loan', 'contact', 'month', 'poutcome', 'subscribed'], axis=1, inplace=True)
```

```
a.head(10)
```

previous	binary_subscribed	blue-collar	entrepreneur	...	primary	secondary	tertiary	yes	yes	cellular	telephone	failure	other	success
0	0	0	0	...	0	0	1	1	0	0	0	0	0	0
0	0	0	0	...	0	1	0	1	0	0	0	0	0	0
0	0	0	1	...	0	1	0	1	1	0	0	0	0	0
0	0	1	0	...	0	0	0	1	0	0	0	0	0	0
0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
0	0	0	0	...	0	0	1	1	0	0	0	0	0	0
0	0	0	0	...	0	0	1	1	1	0	0	0	0	0
0	0	0	1	...	0	0	1	1	0	0	0	0	0	0
0	0	0	0	...	1	0	0	1	0	0	0	0	0	0
0	0	0	0	...	0	1	0	1	0	0	0	0	0	0

Figure 24

VI. Train & Test

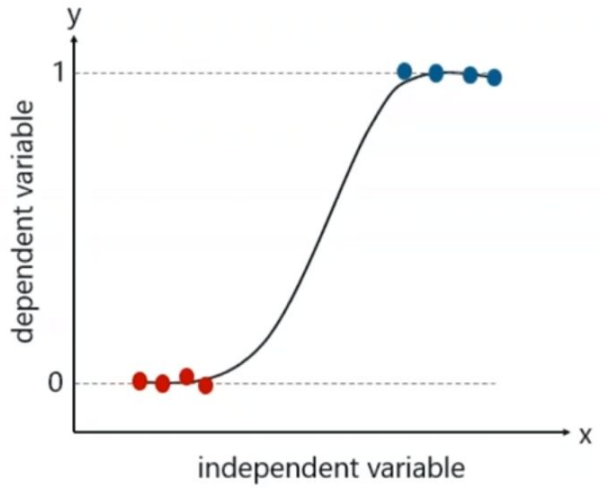
This section consists of building the model on the train data and predict the output on the test data.

We have imported our logistic regression function from sklearn all the other necessary libraries to test the accuracy of our model.

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

Figure 25

Then, as stipulated by the logistic regression function curve below: We have identified our target variable (X), and the independent variables (y) that will help determine whether the client will subscribe or not.



The following figure 26 displays how we have implemented the model using python logistic regression model from the sklearn library.

```
In [87]:
#Spliting Data into training and testing subset

X = original_data.drop("binary_subscribed", axis=1) #Independent variables
y = original_data["binary_subscribed"] #Value we need to predict || subscribed or not

#print(y)
```

```
In [100]:
#from sklearn.cross_validation import train_test_split
# X_train, x_test, Y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

```
In [101]:
logmodel = LogisticRegression()
```

```
In [102]:
logmodel.fit(X_train, y_train)
```

```
C:\Users\23058\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: Default solver will be chang
ed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
Out[102]:
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

```
In [103]:

predictions = logmodel.predict(X_test)
```

Figure 26

VII. Accuracy check

A. Classification report

A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False. (muthu.co)
In our case we are having an average weighted accuracy of 0.82 which is pretty good a score

1.Evaluation using Classification report

```
In [108]:
#Evaluate how well the model is doing
print(classification_report(y_test, predictions))
```

	precision	recall	f1-score	support
0	0.86	0.95	0.90	3521
1	0.67	0.42	0.52	925
micro avg	0.84	0.84	0.84	4446
macro avg	0.77	0.68	0.71	4446
weighted avg	0.82	0.84	0.82	4446

Figure 27

B. Confusion matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.(geeksforgeeks.org)

Predictive Model: Evaluation

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

predictive result / classification		actual result / classification: yes	actual result / classification: no
	yes	**tp** (true positive)	**fp** (false positive) ← Type 1 error
	no	**fn** (false negative)	**tn** (true negative)

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \qquad \text{True Negative Rate} = \frac{tn}{tn + fp}$$

Definition of the Terms:

• Positive (P) : Observation is positive (for example: is an apple).

• Negative (N) : Observation is not positive (for example: is not an apple).

• True Positive (TP) : Observation is positive, and is predicted to be positive.

• False Negative (FN) : Observation is positive, but is predicted negative.

• True Negative (TN) : Observation is negative, and is predicted to be negative.

• False Positive (FP) : Observation is negative, but is predicted positive.

In our case using the confusing matrix we have a score of 0.83 as accuracy.

2.Evaluation confusion matrix

```
In [110]: #Determine accuracy using the confusion metric

          print(confusion_matrix(y_test, predictions))
```

```
[[3331  190]
 [ 535  390]]
```

Accuracy matrix

Calculate accuracy score manually

(3331 + 390) / (3331 + 190 + 535 + 390)

```
In [113]: #Calculate accuracy score manually using the accuracy_score function in python
          print("The accuracy score is:" + str( accuracy_score(y_test, predictions)) )
```

```
The accuracy score is:0.836932073774179
```

Figure 27

VIII. Conclusion

To conclude, the journey of building a model which will allow MLK holding to decide which client will be contacted for the special term deposit has been insightful. After acquiring the data, we describes its attributes, then we analysis the data and preparing it to fit our logistic regression model. Our algorithm gave us on average 0.83 as accuracy rate which is pretty good as a score. However, even we have dropped my reference from the model we still have 0.82 as score (figure 27, 28) which is a **82.0%** accuracy which is a pretty good performance.

Figure 27

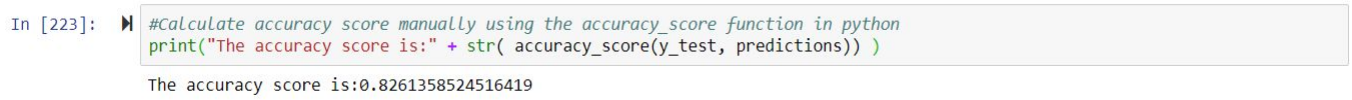


Figure 28

Works Cited

Chauhan, Nagesh Singh. “Real World Implementation of Logistic Regression.” *Medium*, Towards Data Science, 4 Apr. 2019, towardsdatascience.com/real-world-implementation-of-logistic-regression-5136cefb8125.

“Confusion Matrix in Machine Learning.” *GeeksforGeeks*, 7 Feb. 2018,

www.geeksforgeeks.org/confusion-matrix-machine-learning/.

“Data Science Tutorial Videos.” *YouTube*, YouTube, www.youtube.com/playlist?list=PL9ooVrP1hQOFZ1W2m8zYMxUiQhHywHxYm.

“Data Science With R Programming Certification Training By Edureka.” *Edureka.co*, www.edureka.co/data-science-r-programming-certification-course.

Krishnan, Muthu. “Understanding the Classification Report through Sklearn.” *Muthukrishnan*, 19 Oct. 2019, muthu.co/understanding-the-classification-report-in-sklearn/.

Ray, Sunil, and Business Analytics and Intelligence. “Essentials of Machine Learning Algorithms (with Python and R Codes).” *Analytics Vidhya*, 3 Sept. 2019, www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/.