

PROFESSIONAL TRAINING REPORT - II

Entitled

Analysing Term Deposits in Banking Sector by Performing Predictive Analysis

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering with
specialization in Artificial Intelligence and Robotics

by

Thejeshri Rajesh [41612041]



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

CATEGORY -1 UNIVERSITY BY UGC

**Accredited with Grade "A++" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

MAY 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

CATEGORY-I UNIVERSITY BY UGC

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **Ms. Thejeshri Rajesh (41612041)** who carried out the project entitled "**Analysing Term Deposits in Banking Sector by Performing Predictive Analysis**" under my supervision from January 2024 to May 2024.

Internal Guide

Mrs. V. Subapriya, M.E, (Ph.D.)

Head of the Department

Dr. S. VIGNESHWARI, M.E., (Ph.D.)

Submitted for Viva voce Examination held on 6/5/2024

Internal Examiner

External Examiner

DECLARATION

I, **Thejeshri Rajesh (41612041)**, hereby declare that the Professional Training Report-II entitled "**Predicting Term Deposit Subscriptions**" done by me under the guidance of **Mrs. V. Subapriya, M.E, (Ph.D.)** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering with specialization in Artificial Intelligence and Robotics.

DATE: 6/5/24

PLACE: Chennai


SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr. S.Vigneshwari M.E., Ph.D., Head of the Department of Computer Science and Engineering** for providing me necessary support during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Internal Guide **Mrs. V. Subapriya, M.E, (Ph.D.)** for her valuable guidance, suggestions and constant encouragement which paved way for the successful completion of my phase-2 Professional Training.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

COURSE CERTIFICATE



ABSTRACT

This study delves into the application of machine learning methodologies for predicting term deposit subscriptions within the framework of a bank's marketing campaign. Utilizing the Bank Marketing Dataset, which comprises an extensive range of client attributes and marketing campaign outcomes, the research aims to construct a robust predictive model. Employing rigorous data pre-processing techniques, including data cleaning, normalization, and categorical variable encoding, ensures the integrity and quality of the dataset. Subsequently, feature engineering methods are employed to extract relevant insights and transform raw data into meaningful predictors of term deposit subscriptions. Various machine learning algorithms, encompassing logistic regression, decision trees, random forests, gradient boosting, and neural networks, are explored and evaluated for their predictive performance. Evaluation metrics such as accuracy, precision, recall, and F1 score are employed to assess the effectiveness of each algorithm. Additionally, interpretability techniques are applied to unveil the underlying factors influencing term deposit subscriptions, providing valuable insights for stakeholders. The primary objective of this research is to deliver an effective and interpretable predictive model that enables banks to optimize marketing strategies, refine customer targeting efforts, and ultimately maximize term deposit subscription rates. This endeavour contributes not only to enhancing the financial performance and competitiveness of banks but also to fostering improved customer satisfaction and engagement.

CHAPTER NO.	TITLE		PAGE NO.
	TABLE OF CONTENTS		
	ABSTRACT		vi
	LIST OF FIGURES		vii
1	INTRODUCTION 1.1 Overview		1
2	LITERATURE SURVEY 2.1 Survey		2 - 3
3	REQUIREMENTS ANALYSIS		
	3.1	Objective	4
		Requirements	5
	3.2	3.2.1 Hardware Requirements	6-7
		3.2.2 Software Requirements	7-9
4	DESIGN DESCRIPTION OF PROPOSED PRODUCT		
	4.1	4.1.1 Ideation Map/Architecture Diagram	10
		4.1.2 Various stages	11
		4.1.3 Internal or Component design structure	12-13
		4.1.4 working principles	14
	4.2	4.2.1 Novelty of the Project	16
5	RESULTS AND DISCUSSION		17-19
	SUMMARY AND CONCLUSIONS		19
	REFERENCES		20
6	APPENDIX A. Research Paper B. Source Code with Output		21- 24 25- 46

LIST OF FIGURES

Figure No.

Figure Name

Page No.

1.	3.1 Flowchart	11
2.	4.1 Data Processing	12

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Marketing to potential clients is a critical challenge for banking institutions, requiring targeted strategies to reach specific demographics effectively. Traditionally, banks have utilized mediums such as social media, customer service, digital media, and strategic partnerships to engage customers. However, the advent of machine learning has revolutionized this process by leveraging data and analytics to provide precise targeting strategies. By analysing customer data, banks can identify individuals more likely to subscribe to financial products, thereby optimizing marketing efforts. In this project focused on bank marketing with machine learning, we explore how a Portuguese bank can utilize predictive analytics to prioritize customers for bank deposit subscriptions. The dataset used in this project originates from the direct marketing campaigns of a Portuguese banking institution, where phone calls were employed as the primary communication channel. These campaigns involved multiple contacts with clients to determine whether they subscribed to the bank's term deposit product. The classification objective is to predict whether a client will subscribe to the bank's term deposit (yes/no). Through this project, we aim to develop a predictive model that assists banks in targeting their marketing efforts more effectively, ultimately enhancing subscription rates and maximizing the bank's profitability.

CHAPTER 2

LITERATURE REVIEW

2.1 SURVEY

According to findings based on the paper of Advanced Data Analysis, Department of Statistics, Columbia University: Based on signs of coefficients of variables in logistic regression, “duration” has a positive effect on people saying “yes”. This is because the longer the conversations on the phone, the higher interest the customer will show to the term deposit. “nr.employed”, which is the number of employees in the bank, has a positive effect on turning people to subscribe the term deposit. This can be because the more employees the bank has, the more influential and prestigious this bank is. “Euribor” is another important variable, which denotes the Euribor 3-month rate. This indicator is based on the average interbank interest rates in Eurozone. It also has a positive effect since the higher the interest rates the more willingly customer will spend their money on financial tools. Employment variation rate (emp.var.rate) has a negative influence, which means the change in the employment rate will make customers less likely to subscribe to a term deposit. This makes sense because the employment rate is an indicator of the macroeconomy. A stable employment rate denotes a stable economic environment in which people are more confident to make their investments. Therefore, if banks want to improve their lead generation, what they should do is hire more people to work for them, improve the quality of conversation on the phone and run their campaigns when interest rates are high and the macroeconomic environment is stable.

According to the findings based on the paper in the International Journal of Computer Applications: The set of features that describes the data set can be discriminated from 16 features into the 3 features predominant set {Age, Duration, Balance}, which is considered as a CORE of that dataset. Thus, a huge number of more valuable decisions and predictive rules can be extracted based on such a CORE set, these rules, in turn, help decision-makers to have a provision about acquiring and targeting customers, make faster and better decisions about loan approvals, and minimize the management risk by the accuracy associated with the extracted rules. So, instead of analyzing the data set based on 16 features, which yield an intractable process for extracting meaningful patterns, only the CORE set that entails 3 features is used. Moreover, the Decision tree approach has been implemented on the same data set, the gain ratio for each attribute has been calculated and the C4.5 classifier has been used in the

classification process. The gain ratios showed that the “Duration” feature has the maximum gain ratio, however, the “age” feature was the 8th in the gain, and the “balance” was the 10th. DT also provides a huge number of decision and predictive rules using the 16 features associated with accuracy. Some of the DT extracted rules are more summarized than those rules extracted by the RST and others are not. Therefore, despite the decision tree being easy to be implemented as a classifier, and the hardness of the rough set theory implementation, RST yields a better summarization of the data set due to the feature reduction process that achieves the best minimal set of features to describe the information space and preserve its approximations.

CHAPTER 3

REQUIREMENTS ANALYSIS

3.1 OBJECTIVE OF THE PROJECT

The primary objective of this project is to develop a predictive model using machine learning techniques to identify clients who are more likely to subscribe to a bank's term deposit product. By leveraging historical data from the bank's marketing campaigns, our aim is to provide actionable insights that can inform targeted marketing strategies, thereby increasing subscription rates and optimizing the bank's marketing efforts. The ultimate result we are seeking is the development of a reliable and accurate predictive model that can effectively identify clients who are more likely to subscribe to the bank's term deposit product. This model will enable banks to prioritize their marketing efforts, target high-potential customers, and allocate resources more efficiently. By increasing subscription rates and optimizing marketing strategies, the project aims to enhance the bank's profitability and customer satisfaction.

3.2 REQUIREMENTS

To develop a successful model for predicting term deposit bank subscriptions, several critical steps must be taken. Firstly, high-quality data containing comprehensive client information, such as demographics and banking history, must be obtained. Secondly, relevant features that capture key aspects of client behavior and financial status must be identified. Data preprocessing techniques, including handling missing values and encoding categorical variables, ensure the data's suitability for analysis. Choosing the appropriate machine learning algorithm, training it using prepared data, and evaluating its performance using established metrics are essential. Interpretability techniques aid in understanding model predictions, while deployment considerations involve scalability and integration with existing systems. Ethical considerations, such as data privacy and bias mitigation, ensure responsible model usage.

3.2.1 *HARDWARE REQUIREMENTS*

COMPUTER:

A computer with sufficient processing power and memory to handle data processing, model training, and evaluation tasks. A multi-core processor and at least 8GB of RAM are recommended for efficient performance.

STORAGE:

Adequate storage space to store the dataset and any intermediate files generated during data pre-processing and model training. Depending on the size of the dataset, several gigabytes of storage may be required.

GRAPHICS PROCESSING UNIT (GPU)

While not strictly necessary, having access to a GPU can significantly accelerate model training for certain machine learning algorithms, especially deep learning models. If available, a GPU with CUDA support from NVIDIA can be utilized for faster computations.

CLOUD COMPUTING RESOURCES:

Alternatively, cloud computing platforms such as Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure can be leveraged to access scalable computing resources

for data analysis and model training. These platforms offer a variety of virtual machine instances with different specifications to meet specific requirements.

NETWORK CONNECTIVITY :

Stable internet connection is essential, especially when accessing cloud computing resources or collaborating with team members remotely. High-speed internet connectivity ensures efficient data transfer and communication during the project lifecycle.

PERIPHERALS :

Standard peripherals such as keyboard, mouse, monitor, and headset may be required for interacting with the computer and conducting data analysis tasks.

DATA SECURITY :

Implement appropriate security measures to protect sensitive data and ensure compliance with data privacy regulations. This may involve using encryption techniques, access controls, and secure data transmission protocols to safeguard data confidentiality and integrity.

3.2.2 SOFTWARE REQUIREMENTS

SOFTWARES USED:

KAGGLE

I utilized Kaggle as my primary platform for building a predictive model for term deposit bank subscriptions. With its extensive dataset repository, collaborative environment, and integrated Jupyter Notebooks, Kaggle provided a streamlined workflow for data exploration, model development, and experimentation. Leveraging its resources and community insights, I was able to efficiently train and evaluate the model, ultimately leading to the successful development of an accurate predictive tool.

CODE EDITORS USED COLAB

Collaboratory, or “Collab” for short, is a product from Google Research. Collab is a free Jupyter notebook environment that runs entirely in the cloud. It allows you to write and execute Python in your browser, with Zero configuration required, it provides free access to GPUs and it is also useful for easily sharing the codes. Collaboratory, or Collab allows

anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

PROGRAMMING LANGUAGES USED:

PYTHON

Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development.

LIBRARIES AND FRAMEWORKS USED:

PANDAS

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

MATPLOTLIB

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. It is better to represent the data through the graph where we can analyze the data more efficiently and make the specific decision according to data analysis.

NUMPY

NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements.

SEABORN

Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with data frames and the Pandas library. The graphs created can also be customized easily.

SKLEARN

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical reduction.

ALGORITHM USED:

LOGISTIC REGRESSION ALGORITHM:

Logistic regression, despite its name, is a linear model for classification rather than regression. Logistic regression is also known in the literature as logit regression, maximum-entropy classification (MaxEnt) or the log-linear classifier. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function.

K NEAREST NEIGHBOUR ALGORITHM:

K Nearest Neighbor (KNN) is a very simple, easy to understand machine learning algorithms. KNN used in the variety of applications such as finance, healthcare. In Credit ratings, financial institutes will predict the credit rating of customers. In loan disbursement, banking institutes will predict whether the loan is safe or risky. KNN algorithm used for both classification and regression problems. KNN algorithm based on feature similarity approach.

GAUSSIAN NB ALGORITHM:

A Gaussian Naive Bayes algorithm is a special type of NB(Naive Bayes) algorithm. It's specifically used when the features have continuous values. It's also assumed that all the features are following a gaussian distribution i.e., normal distribution.

SVM-SUPPORT VECTOR MACHINE ALGORITHM:

SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the

best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.

RANDOM FOREST CLASSIFIER ALGORITHM:

A random forest is a supervised machine learning algorithm that is constructed from decision tree algorithms. This algorithm is applied in various industries such as banking and e-commerce to predict behavior and outcomes.

CHAPTER 4

DESIGN DESCRIPTION OF PROPOSED PROJECT

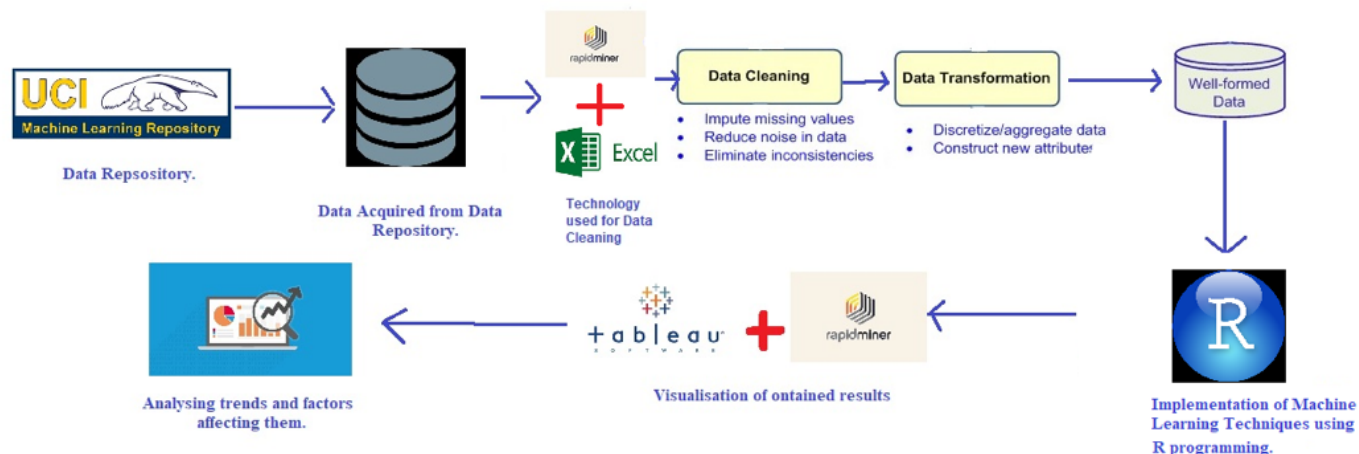
4.1 PROPOSED METHODOLOGY

The methodology is to build a model that predicts customers that would subscribe to a bank term deposit, to which we will be able to achieve by considering various models and using the best one for the prediction. The project focuses on the use of Logistic Regression and other Machine learning techniques to predict categorical values of whether customers or clients will subscribe to the term deposit or not.

The proposed methodology for this project begins with the acquisition of the Bank Marketing Dataset, containing extensive information about client attributes, campaign outcomes, and term deposit subscriptions. Following data acquisition, a thorough exploration of the dataset ensues, involving an in-depth analysis of its structure, features, and distributions to gain insights into potential patterns and relationships. Subsequently, data pre-processing techniques are applied to clean the dataset, handle missing values, and encode categorical variables, ensuring data quality and consistency. Feature engineering is then conducted to extract new insights and create additional features that may enhance predictive performance. This step involves techniques such as creating interaction terms, polynomial features, or deriving new features from existing ones. Once the dataset is prepared, multiple machine learning algorithms are evaluated for predicting term deposit subscriptions, including logistic regression, decision trees, random forests, gradient boosting, and neural networks. Each algorithm undergoes rigorous evaluation using appropriate performance metrics such as accuracy, precision, recall, and F1 score. Model interpretation techniques are employed to understand the factors influencing term deposit subscriptions and derive actionable insights for marketing strategies. Finally, the chosen model is deployed and integrated into the bank's marketing infrastructure for real-time predictions. Continuous monitoring and maintenance mechanisms are established to ensure the model's effectiveness and relevance over time, with periodic updates and retraining as needed. This comprehensive methodology aims to leverage data-driven insights to optimize marketing strategies, refine customer targeting efforts, and maximize term deposit subscription rates within the banking industry. By following this proposed methodology, we aim to develop an effective predictive model for predicting term

deposit subscriptions in the bank marketing dataset, providing valuable insights to inform marketing strategies and enhance customer targeting efforts.

4.1.1 Ideation Map/System Architecture



3.1 Flowchart processing

4.1.2 Various Stages

Various stages of the project include:

1. Data Exploration and Pre-processing: The first step involves exploring the dataset to understand its structure, features, and distributions. This includes handling missing values, encoding categorical variables, and performing any necessary data transformations to prepare the data for modelling.

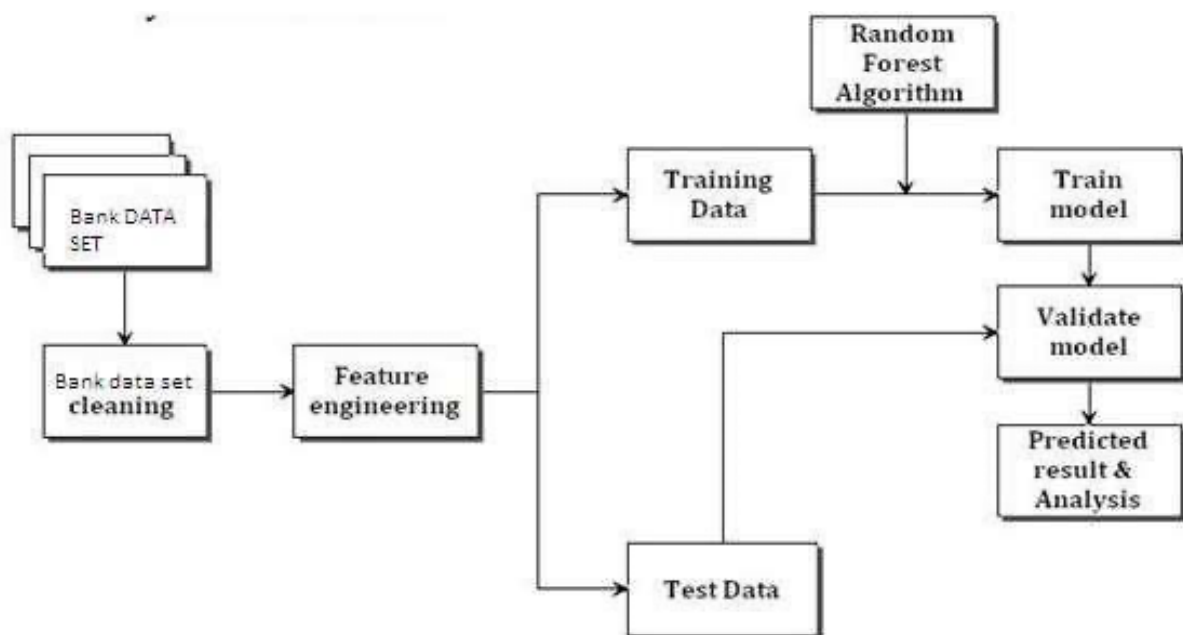
2. Feature Engineering: Next, we delve into feature engineering to extract relevant information from the dataset that may enhance predictive performance. This may involve creating new features, transforming existing ones, or selecting the most informative features for modelling.

3. Model Selection and Training: We then proceed to select appropriate machine learning algorithms for the task of predicting term deposit subscriptions. Various algorithms, such as logistic regression, decision trees, random forests, gradient boosting, and neural networks, will be explored and evaluated.

4. Model Evaluation and Validation: The selected models are evaluated using established performance metrics, including accuracy, precision, recall, and F1 score, to assess their predictive performance. Cross-validation techniques may be employed to ensure the robustness and generalizability of the models.

5. Interpretation and Insights: Finally, we interpret the results of the predictive models to gain insights into the factors influencing term deposit subscriptions. This includes identifying key features that contribute to subscription likelihood and understanding their impact on customer behaviour.

4.1.3 Internal or Component design structure



4.1 Data Flow

The system design include various stages like data collection, pre-processing, making training, testing data, implementing algorithm and last stage is predicted result. The collected raw data may be incomplete or noisy. The data must go through pre-processing phase to clean the data before using the data for learning, another step for training model is feature extraction. In machine learning feature engineering is a critical aspect it includes feature selection and extraction it boost the model performance. When we trying to predict the output first we need to train the model using a dataset and model try to learn the data to make accurate prediction.

Test data is independent of training data, if a model fit to the train data then it also fit to the test data, least overfitting has taken place. Random forest is most used algorithm in ML most of time this algorithm give the best accuracy result. Training the model using ML algorithm with the training data. The process of trained model is evaluated with the test data is validate data, finally, the model give the best predicted result.

4.1.4 *working principles*

The project operates on the foundational principle of leveraging data-driven insights to predict term deposit subscriptions within the context of a bank's marketing campaign. Beginning with a thorough exploration of the Bank Marketing Dataset, the project meticulously engineers features and pre-processes data to extract meaningful patterns and relationships. Through a systematic approach, various machine learning algorithms, including logistic regression, decision trees, random forests, gradient boosting, and neural networks, are trained and evaluated. Evaluation metrics such as accuracy, precision, recall, and F1 score guide the selection of the most effective model for predicting term deposit subscriptions. Furthermore, the project emphasizes model interpretation, seeking to uncover the underlying factors influencing subscription likelihood. This interpretability aspect facilitates actionable insights for refining marketing strategies and targeting high-potential customers. Upon model deployment, continuous monitoring and periodic maintenance ensure the model's longevity and relevance in real-world applications. Overall, the project's detailed methodology, from data pre-processing to model interpretation, is grounded in the pursuit of actionable insights to optimize marketing efforts and enhance customer engagement for the bank.

4.2 FEATURES

The project encompasses a comprehensive approach to developing a predictive model for term deposit subscriptions within the banking sector. It begins with a meticulous exploration and pre-processing of the Bank Marketing Dataset to ensure data quality and uniformity. Feature engineering techniques are then employed to extract valuable insights and enhance predictive performance. Through rigorous evaluation and selection of machine learning algorithms, including logistic regression, decision trees, random forests, and gradient boosting, the most effective model is identified. Model interpretation techniques provide understanding into the

underlying factors influencing term deposit subscriptions, enabling actionable insights for marketing strategies. The finalized model is seamlessly deployed and integrated into the bank's infrastructure, with continuous monitoring and maintenance mechanisms established to ensure its effectiveness and relevance over time. This holistic approach aims to optimize marketing efforts, refine customer targeting, and maximize term deposit subscription rates, thus enhancing overall business outcomes within the banking industry.

Data Description

This is the classic marketing bank dataset uploaded originally in the UCI Machine Learning Repository. The dataset gives you information about a marketing campaign of a financial institution in which you will have to analyze in order to find ways to look for future strategies in order to improve future marketing campaigns for the bank.

Feature

1. Demographic information:

Features such as age, gender, marital status, education level, and occupation provide insights into the client's background and may influence their likelihood of subscribing to a term deposit.

2. Banking History

data on previous banking transactions, account balance, credit history, and tenure with the bank can indicate the client's financial behavior and stability, which may impact their decision to subscribe to a term deposit.

3. Marketing Interactions:

information about past marketing interactions, such as responses to promotional campaigns, calls made by the bank, or emails opened, can reveal the client's engagement level and receptiveness to term deposit offers.

4. Economic Indicators:

external factors such as interest rates, inflation rates, unemployment rates, and gdp growth can affect a client's financial decisions and propensity to invest in term deposits.

5. Customer Behaviour:

features derived from customer behaviour data, such as frequency of visits to the bank's website, engagement with online banking services, or usage patterns of mobile apps, provide insights into client preferences and digital banking habits.

6. Geographic Location:

regional or local factors may influence term deposit subscriptions, such as proximity to bank branches, economic conditions, or cultural preferences for saving habits.

7. Time-based Variables:

temporal features such as month of the year, day of the week, or seasonality may capture variations in term deposit subscriptions over time and enable the model to account for temporal dynamics.

8. Derived Features:

additional derived features such as average transaction amount, total number of transactions, or account activity patterns can provide deeper insights into client behavior and financial preferences.

By incorporating these diverse features into the predictive model, banks can gain a comprehensive understanding of client behaviour and improve the accuracy of term deposit subscription predictions.

4.2.1 Novelty of the proposal

The proposal for this project introduces novelty by leveraging machine learning techniques to optimize marketing strategies within the banking sector. Through the utilization of the Bank Marketing Dataset and comprehensive feature engineering, hidden patterns and trends are unearthed, providing a rich foundation for predictive modeling of term deposit subscriptions. The project innovates by evaluating multiple machine learning algorithms, prioritizing model interpretability, and emphasizing the derivation of actionable insights. Additionally, the incorporation of continuous monitoring and maintenance practices ensures the relevance and effectiveness of the predictive model in real-world applications. This holistic approach, encompassing data exploration, feature engineering, model selection, interpretation, and maintenance, introduces a novel paradigm for enhancing marketing campaigns, refining customer targeting efforts, and maximizing term deposit subscription rates within the banking industry.

CHAPTER 5

RESULTS AND DISCUSSION

The main objective of this project is to increase the effectiveness of the bank's telemarketing campaign, which was successfully met through data analysis, visualization and analytical model building. A target customer profile was established while classification and regression models were built to predict customers' response to the term deposit campaign.

According to previous analysis, a target customer profile can be established. The most responsive customers possess these features:

Feature 1: age < 30 or age > 60

Feature 2: students or retired people

Feature 3: a balance of more than 5000 euros

By applying logistic and ridge regression algorithms, classification and estimation model were successfully built. With these two models, the bank will be able to predict a customer's response to its telemarketing campaign before calling this customer. In this way, the bank can allocate more marketing efforts to the clients who are classified as highly likely to accept term deposits, and call less to those who are unlikely to make term deposits. In addition, predicting duration before calling and adjusting marketing plan benefit both the bank and its clients. On the one hand, it will increase the efficiency of the bank's telemarketing campaign, saving time and efforts. On the other hand, it prevents some clients from receiving undesirable advertisements, raising customer satisfaction. With the aid of logistic and ridge regression models, the bank can enter a virtuous cycle of effective marketing, more investments and happier customers.

5.2 Future Work

Future work for this project could involve several avenues for further improvement and exploration. Firstly, the integration of more advanced machine learning techniques, such as deep learning architectures like recurrent neural networks (RNNs) or transformers, could enhance the model's predictive performance by capturing complex temporal dependencies and nonlinear relationships in the data. Additionally, incorporating external data sources, such as social media activity or macroeconomic indicators, could provide additional context and improve the model's accuracy. Furthermore, exploring the use of ensemble methods or meta-learning approaches to combine multiple models could help mitigate overfitting and improve generalization. Moreover, deploying the model in a real-world setting and conducting rigorous A/B testing to evaluate its effectiveness in driving marketing strategies and increasing term deposit subscriptions would be crucial. Finally, continuous monitoring and updating of the model based on feedback and changing market conditions would ensure its long-term relevance and effectiveness in the dynamic banking landscape.

CONCLUSION

In conclusion, the project to predict term deposit bank subscriptions from clients represents a significant step forward in leveraging machine learning for enhancing marketing strategies and customer engagement in the banking sector. By harnessing comprehensive client data and employing advanced modelling techniques, we have developed a predictive model capable of accurately identifying potential term deposit subscribers. The project's success underscores the importance of data-driven decision-making in financial institutions and highlights the potential for predictive analytics to drive actionable insights and improve business outcomes. Moving forward, continued refinement and optimization of the model, along with its integration into operational workflows, will be essential to realize its full potential in driving customer acquisition and retention efforts. This project serves as a testament to the power of data science in transforming traditional banking practices and adapting to the evolving needs of clients in today's digital age.

REFERENCES

- 1) Y Dasril, A Alamsyah and T Mustaqim,” Bank predictions for prospective long-term deposit investors using machine learning LightGBM and SMOTE”, Published under licence by IOP Publishing Ltd(2021)
- 2) Phan Dun Yung, Tran Duc Hanh and Ta Duc Hanh, “Term Deposit Subscription Prediction Using Spark MLlib and ML Packages“, ICEBA 2019: Proceedings of the 2019 5th International Conference on E-Business and Applications (2019)
- 3) Sipu Hou, Zongzhen Cai, Jiming Wu, Hongwei Du and Peng Xie, ” Applying Machine Learning to the Development of Prediction Models for Bank Deposit Subscription“, published under International Journal of Business Analytics(2016)
- 4) Ashalata Panigrahi and Manik Chand Patnaik,” Customer Deposit Prediction Using Neural Network Techniques”, published under International Journal of Applied Engineering Research ISSN 0973-4562 Volume 15, Number 3 (2020)

B. SOURCE CODE WITH OUTPUT:

1.Load dataset

```
import pandas as pd
import numpy as np
from pandas import read_csv
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
import os

linkname = '../input/bank-full.csv'
dataset1 = pd.read_csv(linkname, sep = ';')
```

```
# View the first 5 rows in the dataset
dataset1.head()
```

```
[2]:
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no

2.1 Clean the dataset

```
# Step 1: Delete the rows which column 'poutcome' contains 'other'
condition = dataset1.poutcome == 'other'
dataset2 = dataset1.drop(dataset1[condition].index, axis = 0, inplace =
False)
```

```
# Step 2: Replace 'unknown' in job and education with 'other'
dataset2[['job', 'education']] =
dataset2[['job', 'education']].replace(['unknown'], 'other')
```

2.2 Drop outliers in the column 'balance'

```
from scipy.stats import zscore

dataset2[['balance']].mean()
dataset2[['balance']].mean()

dataset2['balance_outliers'] = dataset2['balance']
```

```
dataset2['balance_outliers']= zscore(dataset2['balance_outliers'])

condition1 = (dataset2['balance_outliers']>3) |
(dataset2['balance_outliers']<-3 )
dataset3 = dataset2.drop(dataset2[condition1].index, axis = 0, inplace
= False)
```

```
dataset4 = dataset3.drop('balance_outliers', axis=1)
```

2.3 Creating and transforming data

```
# Step 1: Change column name: 'y' to 'response'
dataset4.rename(index=str, columns={'y': 'response'}, inplace = True)

def convert(dataset4, new_column, old_column):
    dataset4[new_column] = dataset4[old_column].apply(lambda x: 0 if x
== 'no' else 1)
    return dataset4[new_column].value_counts()

convert(dataset4, "response_binary", "response")
```

```
0  37785
1   4870
Name: response_binary, dtype: int64
```

```
# Step 2: Drop column "contact" which is useless
dataset5 = dataset4.drop('contact', axis=1)
```

```
# Step 3: Change the unit of 'duration' from seconds to minutes
dataset5['duration'] = dataset5['duration'].apply(lambda
n:n/60).round(2)
```

```
# Step 4: Change 'month' from words to numbers for easier analysis
lst = [dataset5]
for column in lst:
    column.loc[column["month"] == "jan", "month_int"] = 1
    column.loc[column["month"] == "feb", "month_int"] = 2
    column.loc[column["month"] == "mar", "month_int"] = 3
    column.loc[column["month"] == "apr", "month_int"] = 4
    column.loc[column["month"] == "may", "month_int"] = 5
    column.loc[column["month"] == "jun", "month_int"] = 6
    column.loc[column["month"] == "jul", "month_int"] = 7
    column.loc[column["month"] == "aug", "month_int"] = 8
    column.loc[column["month"] == "sep", "month_int"] = 9
    column.loc[column["month"] == "oct", "month_int"] = 10
    column.loc[column["month"] == "nov", "month_int"] = 11
    column.loc[column["month"] == "dec", "month_int"] = 12
```

2.4 Filtering

```
# Step 1: Drop rows that 'duration' < 5s
condition2 = (dataset5['duration'] < 5/60)
dataset6 = dataset5.drop(dataset5[condition2].index, axis = 0, inplace
= False)

# Step 2: Drop customer values with 'other' education
condition3 = (dataset6['education'] == 'other')
dataset7 = dataset6.drop(dataset6[condition3].index, axis = 0, inplace
= False)
```

Exploratory Data Analysis

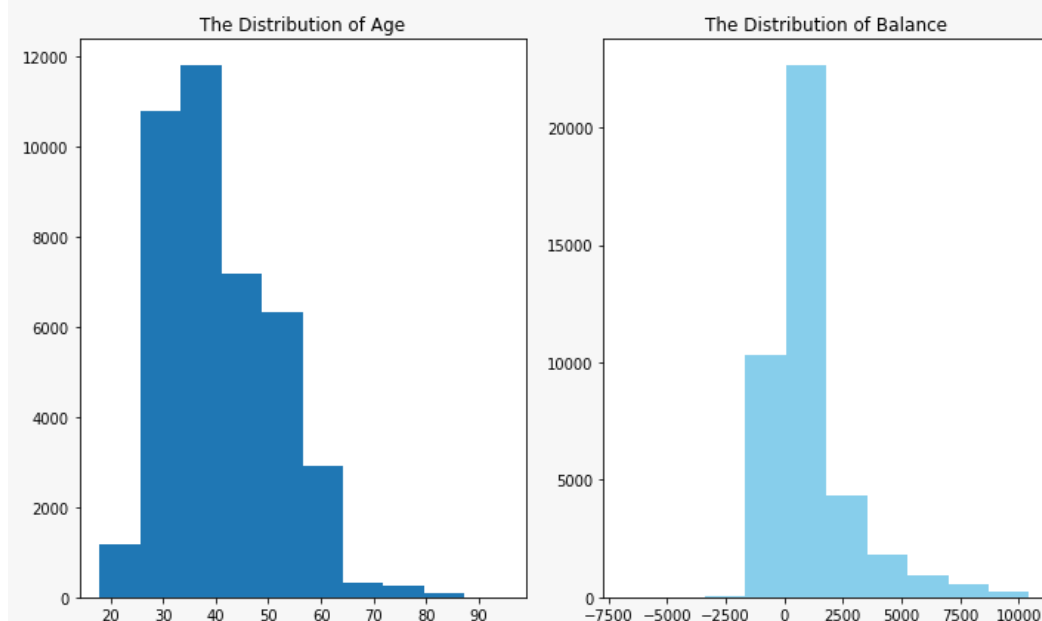
3.1 Visualize the distribution of 'age' and 'balance'

```
dist_age_balance = plt.figure(figsize = (10,6))

ra1 = dist_age_balance.add_subplot(1,2,1)
ra2 = dist_age_balance.add_subplot(1,2,2)

ra1.hist(dataset7['age'])
ra1.set_title('The Distribution of Age')

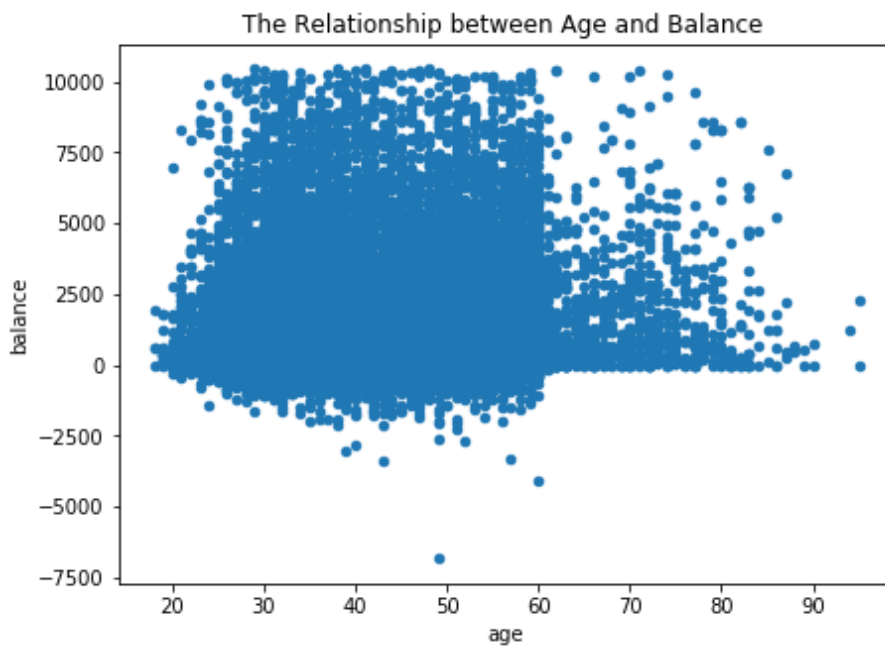
ra2.hist(dataset7['balance'], color = 'skyblue')
ra2.set_title('The Distribution of Balance')
plt.tight_layout()
plt.show()
```



3.2 Visualize the relationship between 'age' and 'balance'

```
scatter_age_balance = dataset7.plot.scatter('age','balance',figsize =
(7,5))

plt.title('The Relationship between Age and Balance ')
plt.show()
```



3.3 Visualize the distribution of 'duration' & 'campaign'

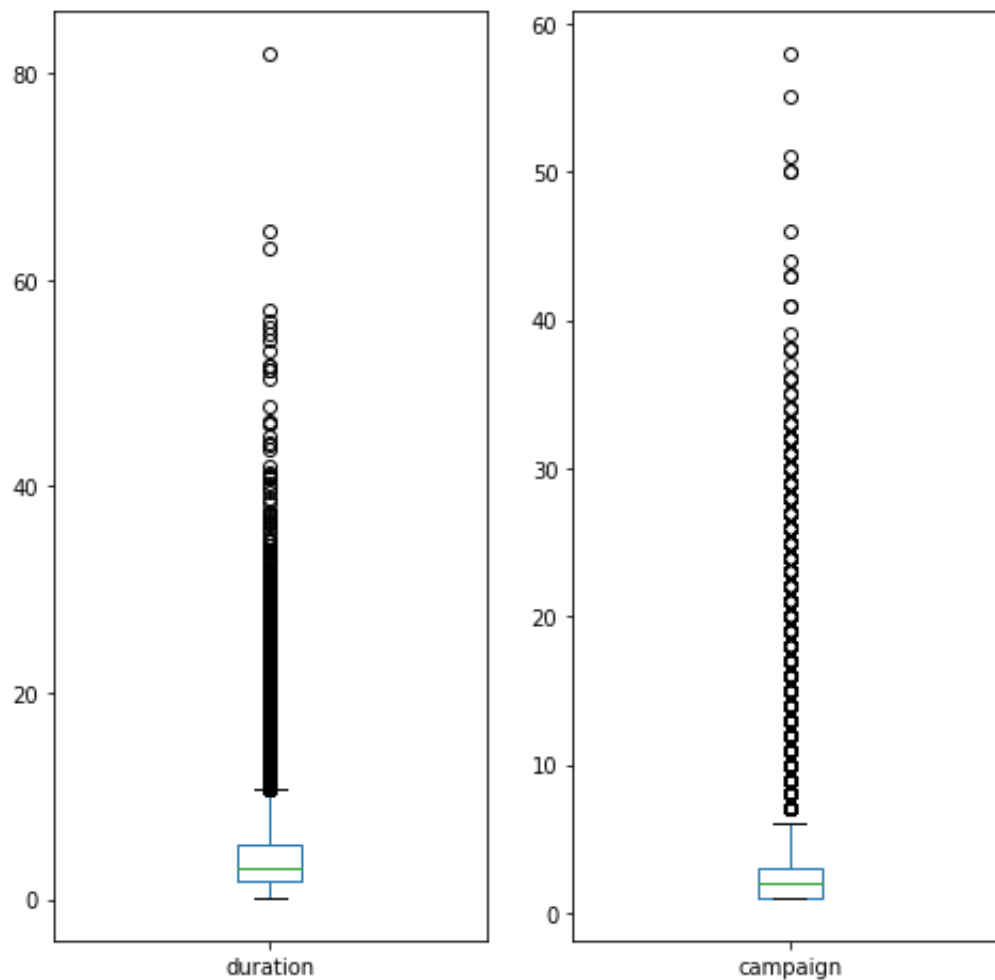
```
dist_dur_cam = dataset7[['duration','campaign']].plot(kind = 'box',
figsize = (8,8),
subplots = True,

sharex = False,

sharey = False,

title='The
Distribution of Duration and Campaign')
plt.show()
```

The Distribution of Duration and Campaign



```
import seaborn as sns
dur_cam = sns.lmplot(x='duration', y='campaign', data = dataset7,
                    hue = 'response',
                    fit_reg = False,
                    scatter_kws={'alpha':0.6}, height =7)

plt.axis([0,65,0,65])

plt.ylabel('Number of Calls')

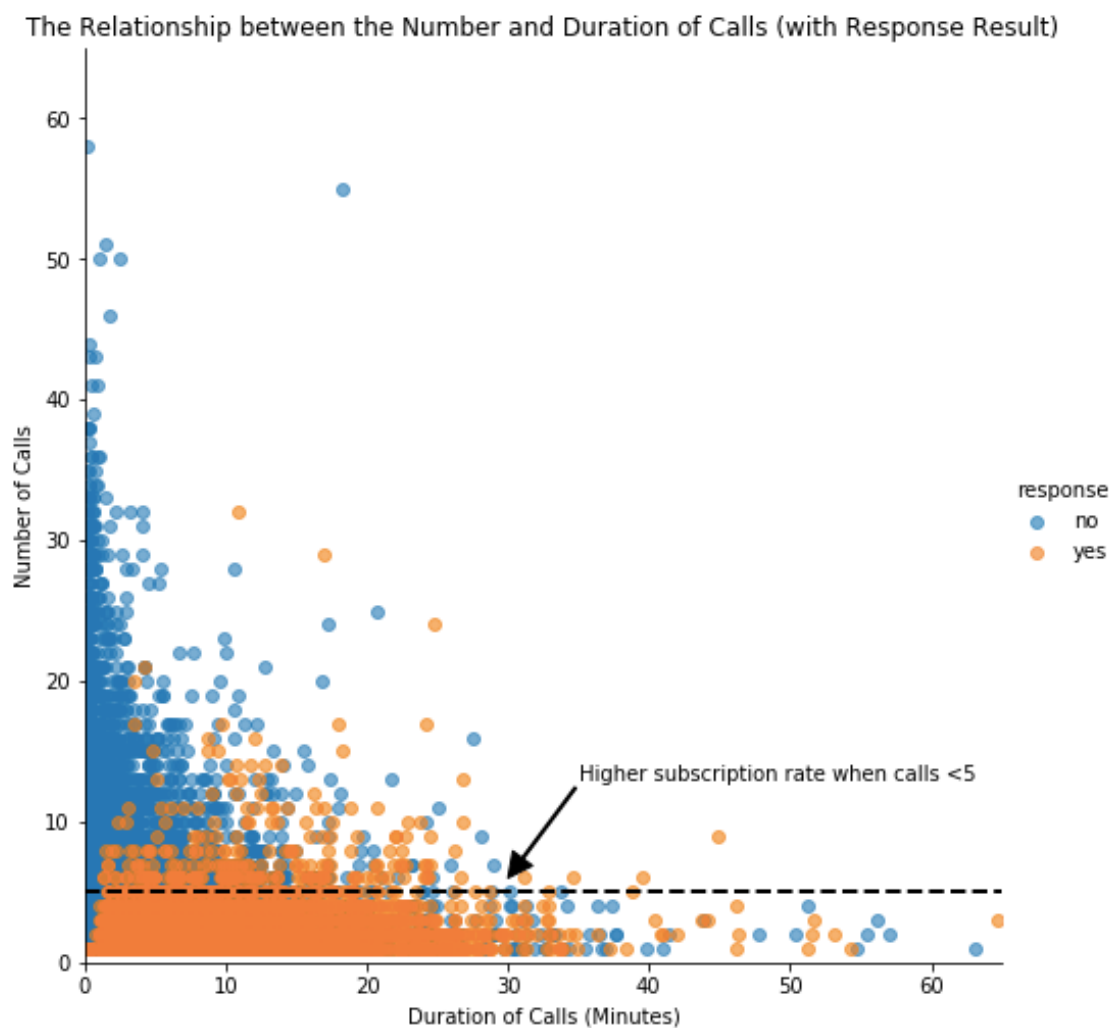
plt.xlabel('Duration of Calls (Minutes)')
```

```
plt.title('The Relationship between the Number and Duration of Calls
(with Response Result)')

# Annotation
plt.axhline(y=5, linewidth=2, color="k", linestyle='--')

plt.annotate('Higher subscription rate when calls <5',xytext = (35,13),
            arrowprops=dict(color = 'k', width=1),xy=(30,6))

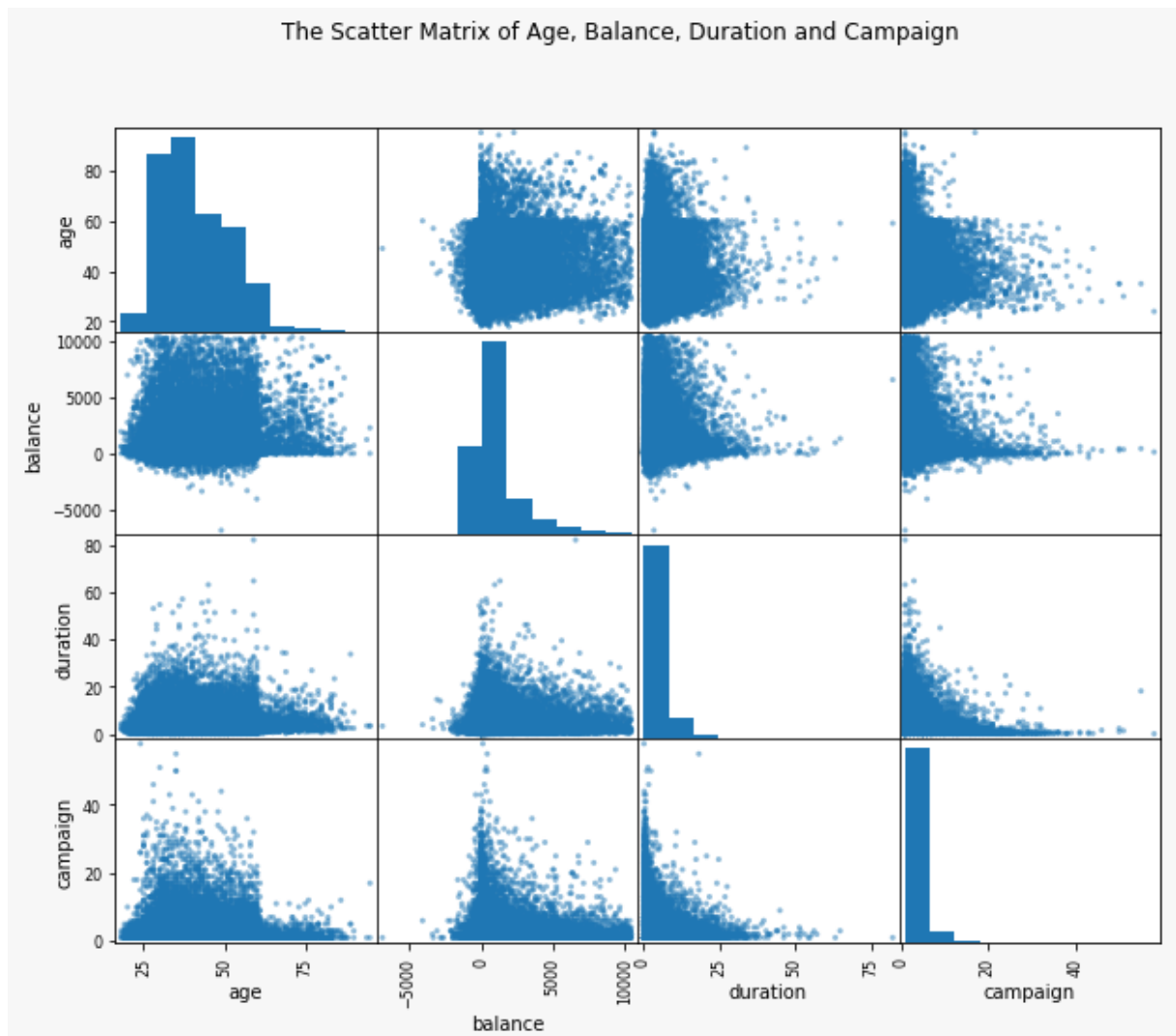
plt.show()
```



3.5 Scatter matrix and Correlation matrix

```
from pandas.plotting import scatter_matrix
matrix=scatter_matrix(dataset7[['age', 'balance', 'duration', 'campaign']],
figsize=(10,8))
```

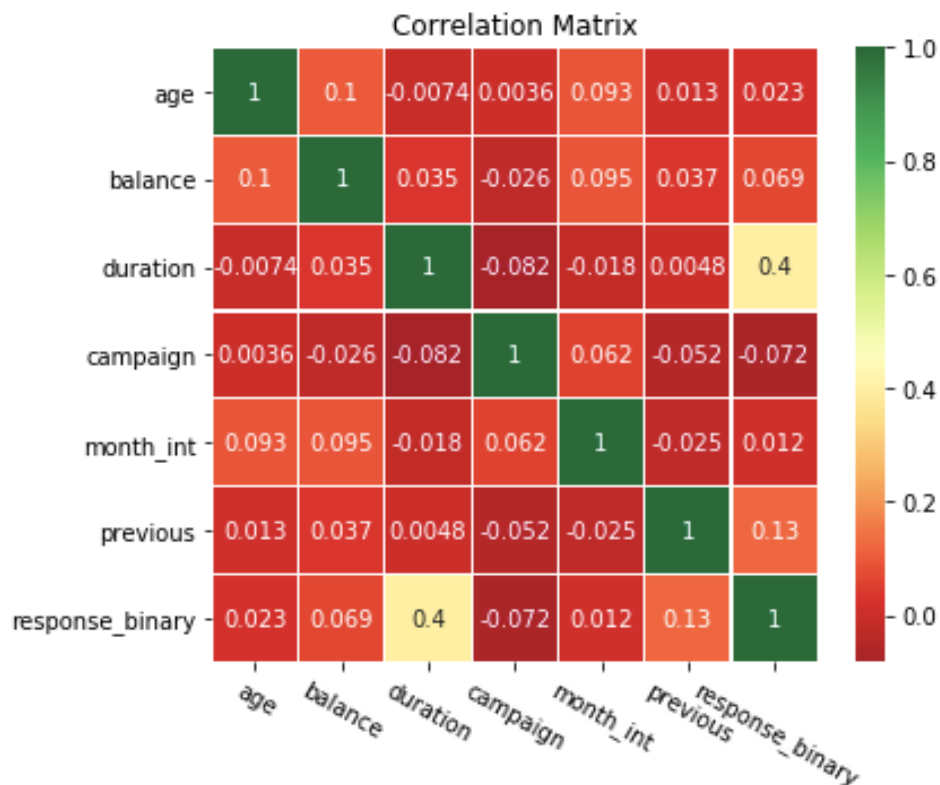
```
plt.suptitle('The Scatter Matrix of Age, Balance, Duration and Campaign')
plt.show()
```



```
corr_data=dataset7[['age', 'balance', 'duration', 'campaign', 'month_int', 'previous', 'response_binary']]
corr = corr_data.corr()

cor_plot=sns.heatmap(corr,annot=True,cmap='RdYlGn',linewidths=0.2,annot_kws={'size':10})
fig=plt.gcf()
fig.set_size_inches(6,5)
plt.xticks(fontsize=10,rotation=-30)
plt.yticks(fontsize=10)
plt.title('Correlation Matrix')
```

```
plt.show()
```



Part 4: Data Visualization

```
lst = [dataset7]
for column in lst:
    column.loc[column["age"] < 30, 'age_group'] = 20
    column.loc[(column["age"] >= 30) & (column["age"] <= 39),
'age_group'] = 30
    column.loc[(column["age"] >= 40) & (column["age"] <= 49),
'age_group'] = 40
    column.loc[(column["age"] >= 50) & (column["age"] <= 59),
'age_group'] = 50
    column.loc[column["age"] >= 60, 'age_group'] = 60
```

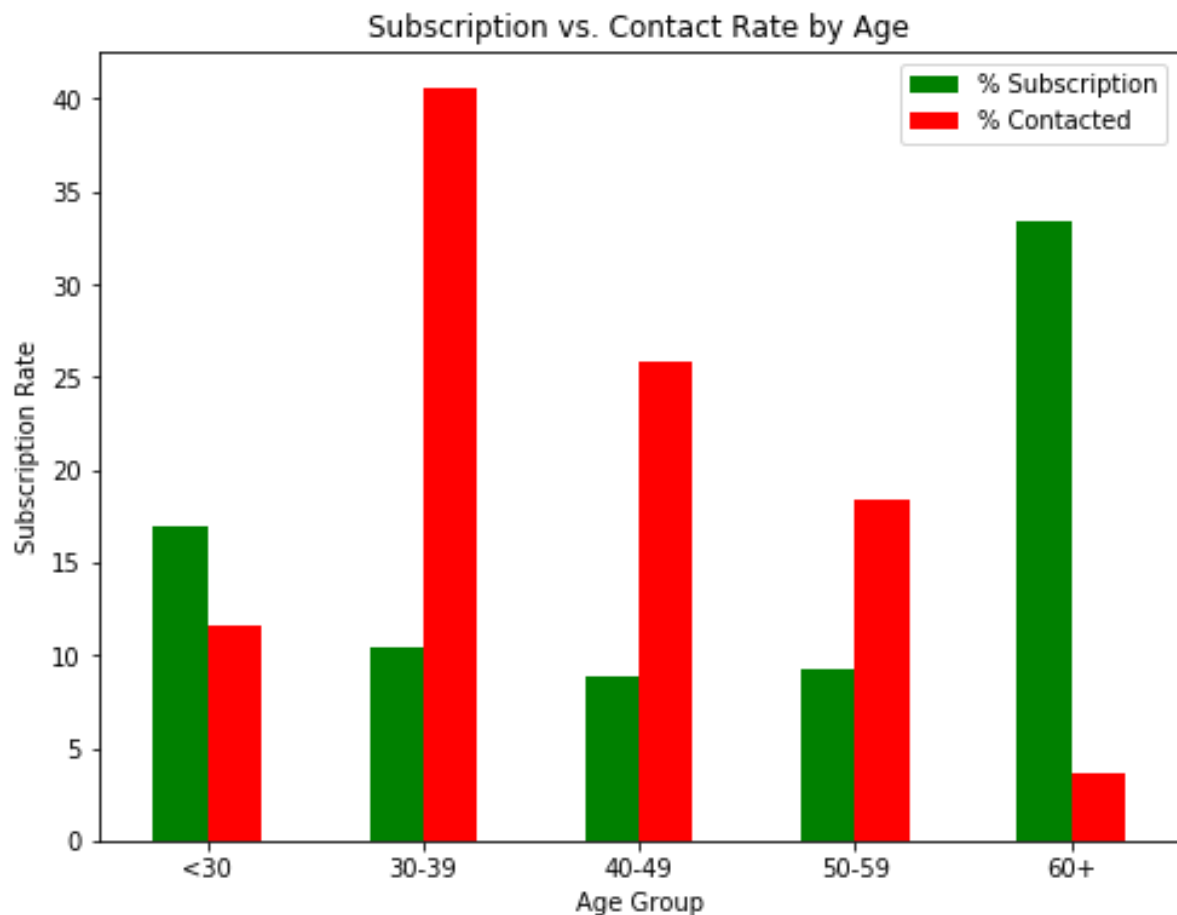
```
count_age_response_pct =
pd.crosstab(dataset7['response'], dataset7['age_group']).apply(lambda x:
x/x.sum() * 100)
count_age_response_pct = count_age_response_pct.transpose()
```

```
age = pd.DataFrame(dataset7['age_group'].value_counts())
age['% Contacted'] = age['age_group']*100/age['age_group'].sum()
```

```
age['% Subscription'] = count_age_response_pct['yes']
age.drop('age_group',axis = 1,inplace = True)

age['age'] = [30,40,50,20,60]
age = age.sort_values('age',ascending = True)
```

```
plot_age = age[['% Subscription','% Contacted']].plot(kind = 'bar',
                                                    figsize=(8,6), color =
('green','red'))
plt.xlabel('Age Group')
plt.ylabel('Subscription Rate')
plt.xticks(np.arange(5), ('<30', '30-39', '40-49', '50-59',
'60+'),rotation = 'horizontal')
plt.title('Subscription vs. Contact Rate by Age')
plt.show()
```



4.2 Visualize the subscription rate by balance level

```
lst = [dataset7]
for column in lst:
    column.loc[column["balance"] <= 0, 'balance_group'] = 'no balance'
```

```

column.loc[(column["balance"] > 0) & (column["balance"] <= 1000),
'balance_group'] = 'low balance'
column.loc[(column["balance"] > 1000) & (column["balance"] <=
5000), 'balance_group'] = 'average balance'
column.loc[(column["balance"] > 5000), 'balance_group'] = 'high
balance'

```

```

count_balance_response_pct =
pd.crosstab(dataset7['response'],dataset7['balance_group']).apply(lambd
a x: x/x.sum() * 100)
count_balance_response_pct = count_balance_response_pct.transpose()
bal = pd.DataFrame(dataset7['balance_group'].value_counts())
bal['% Contacted'] =
bal['balance_group']*100/bal['balance_group'].sum()
bal['% Subscription'] = count_balance_response_pct['yes']
bal.drop('balance_group',axis = 1,inplace = True)

bal['bal'] = [1,2,0,3]
bal = bal.sort_values('bal',ascending = True)

```

```

plot_balance = bal[['% Subscription','% Contacted']].plot(kind = 'bar',
color =
('royalblue','skyblue'),
figsize = (8,6))

plt.title('Subscription vs Contact Rate by Balance Level')

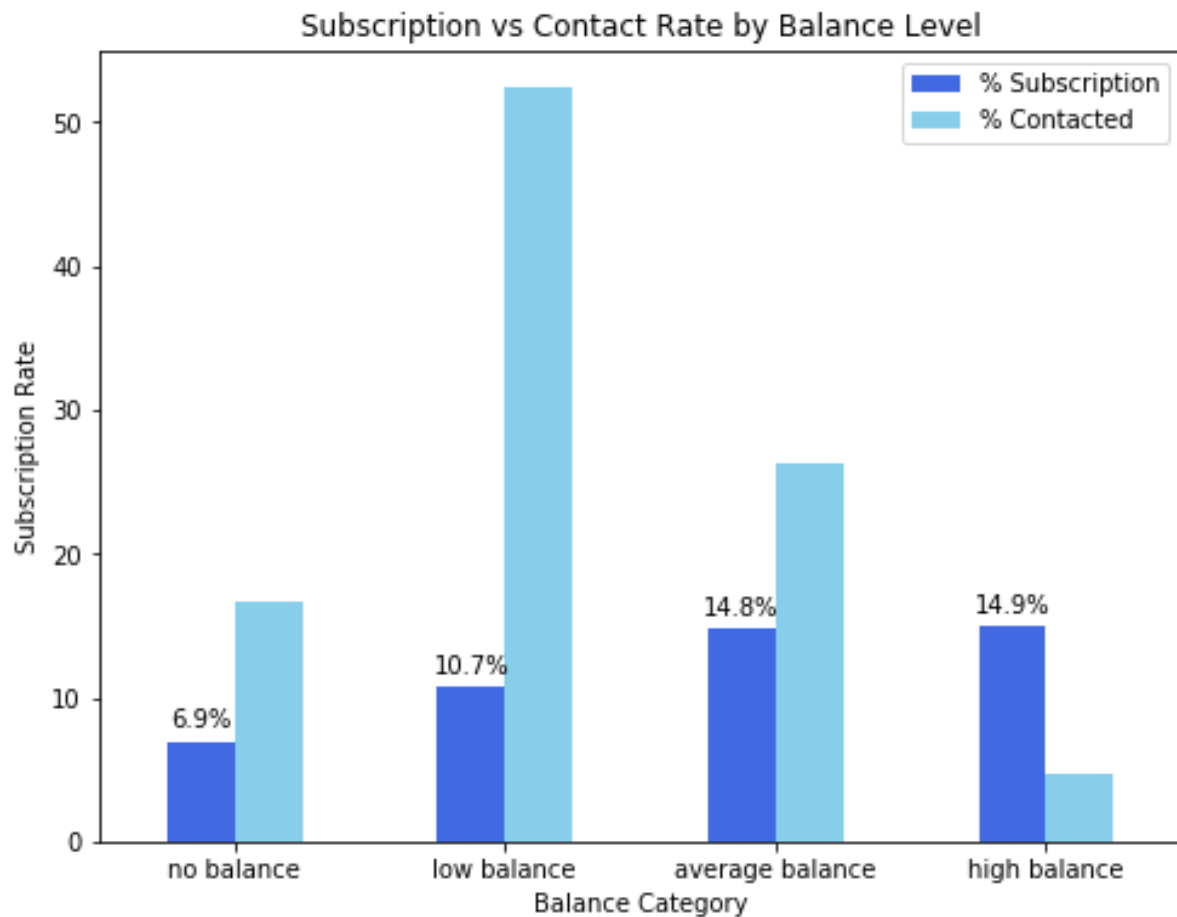
plt.ylabel('Subscription Rate')

plt.xlabel('Balance Category')

plt.xticks(rotation = 'horizontal')

# label the bar
for rec, label in zip(plot_balance.patches,
bal['% Subscription'].round(1).astype(str)):
plot_balance.text(rec.get_x() + rec.get_width()/2,
rec.get_height() + 1,
label+'%',
ha = 'center',
color = 'black')

```



4.3 Visualize the subscription rate by age and balance

```
age_balance1 =
pd.DataFrame(dataset7.groupby(['age_group', 'balance_group'])['response_
binary'].sum())
age_balance2 =
pd.DataFrame(dataset7.groupby(['age_group', 'balance_group'])['response'
].count())

age_balance1['response'] = age_balance2['response']
age_balance1['response_rate'] = age_balance1['response_binary']/
(age_balance1['response'])
age_balance1 = age_balance1.drop(['response_binary', 'response'], axis
=1)

age_balance1 = age_balance1.unstack()

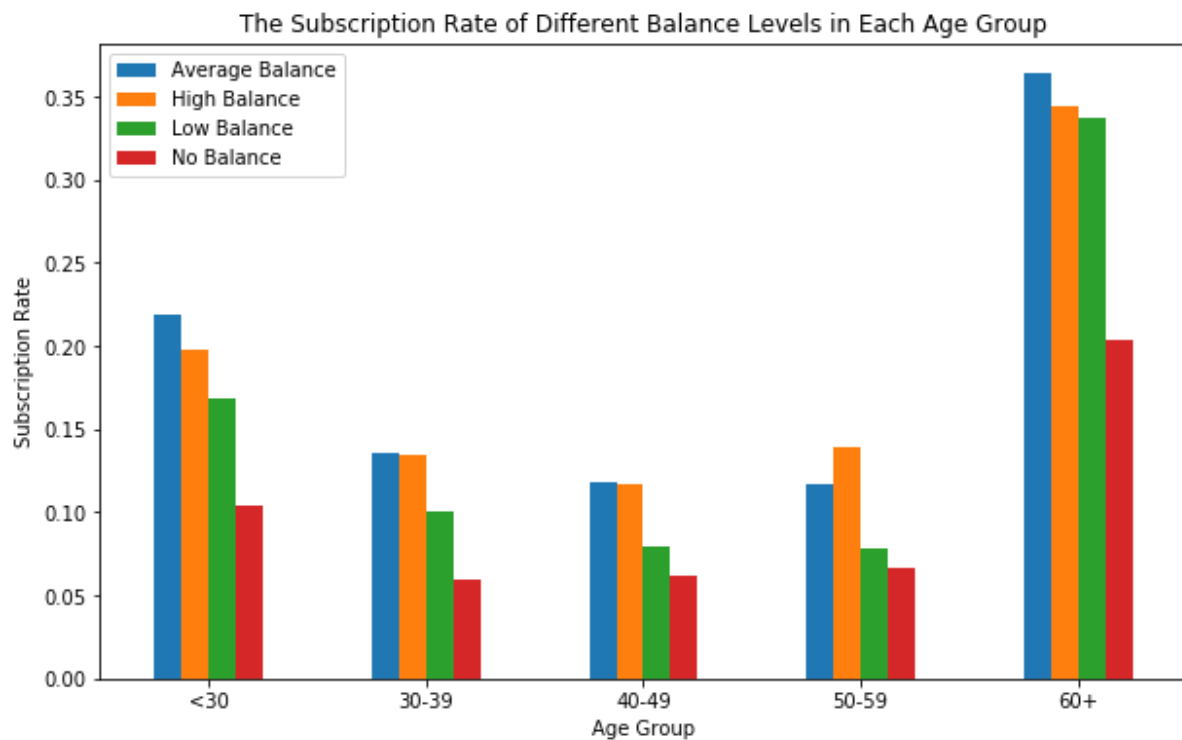
age_bal = age_balance1.plot(kind='bar', figsize = (10,6))

# Set x ticks
```

```
plt.xticks(np.arange(5), ('<30', '30-39', '40-49', '50-59',
'60+'), rotation = 'horizontal')

# Set legend
plt.legend(['Average Balance', 'High Balance', 'Low Balance', 'No
Balance'], loc = 'best', ncol = 1)

plt.ylabel('Subscription Rate')
plt.xlabel('Age Group')
plt.title('The Subscription Rate of Different Balance Levels in Each
Age Group')
plt.show()
```



4.4 Visualize the subscription rate by job

```
count_job_response_pct=pd.crosstab(dataset7['response'],dataset7['job']
).apply(lambda x: x/x.sum() * 100)
count_job_response_pct = count_job_response_pct.transpose()
```

```
plot_job = count_job_response_pct['yes'].sort_values(ascending =
True).plot(kind = 'barh',

figsize = (12,6))

plt.title('Subscription Rate by Job')
plt.xlabel('Subscription Rate')
```

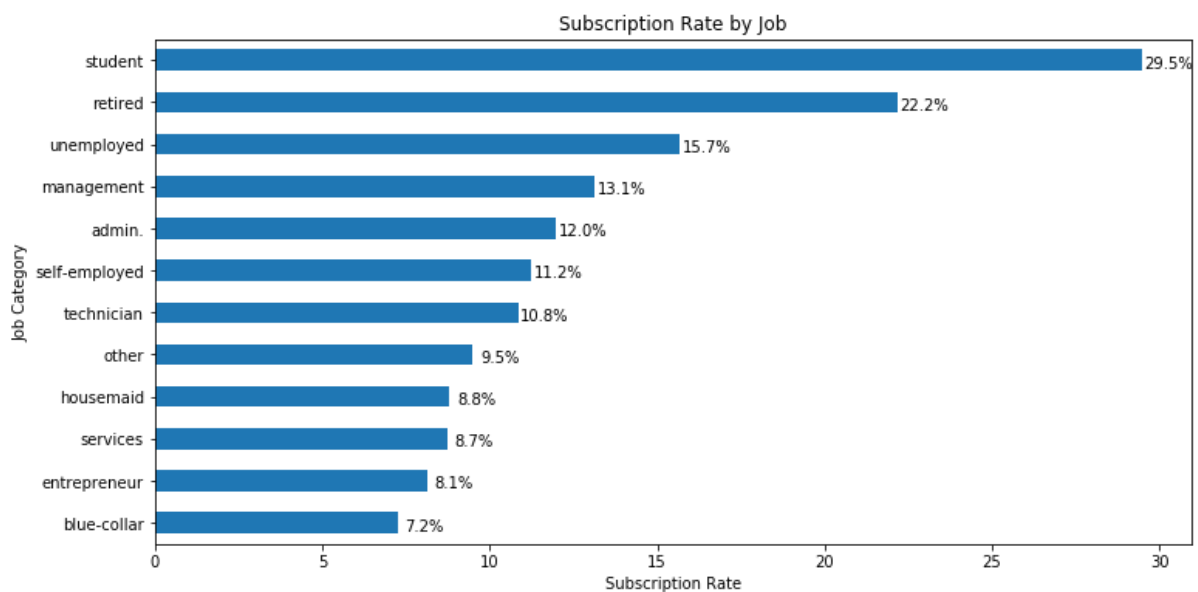
```

plt.ylabel('Job Category')

# Label each bar
for rec, label in zip(plot_job.patches,

count_job_response_pct['yes'].sort_values(ascending =
True).round(1).astype(str)):
    plot_job.text(rec.get_width()+0.8,
                  rec.get_y()+ rec.get_height()-0.5,
                  label+'%',
                  ha = 'center',
                  va='bottom')

```



```

count_month_response_pct =
pd.crosstab(dataset7['response'],dataset7['month_int']).apply(lambda x:
x/x.sum() * 100)
count_month_response_pct = count_month_response_pct.transpose()
month = pd.DataFrame(dataset7['month_int'].value_counts())
month['% Contacted'] = month['month_int']*100/month['month_int'].sum()
month['% Subscription'] = count_month_response_pct['yes']
month.drop('month_int',axis = 1,inplace = True)

month['Month'] = [5,7,8,6,11,4,2,1,10,9,3,12]
month = month.sort_values('Month',ascending = True)

```

```

plot_month = month[['% Subscription','% Contacted']].plot(kind='line',
figsize =
(10,6),
marker = 'o')

```

```

plt.title('Subscription vs. Contact Rate by Month')
plt.ylabel('Subscription and Contact Rate')
plt.xlabel('Month')

ticks = np.arange(1,13,1)
plt.xticks(ticks)

# Annotation: peak of contact
y = month['% Contacted'].max()
x = month['% Contacted'].idxmax()
plt.annotate('May: Peak of contact', xy=(x+0.1, y+0.1),
xytext=(x+1,y+4), arrowprops=dict(facecolor='black', headwidth=6,
width=1, headlength=4), horizontalalignment='left',
verticalalignment='top')

# Annotation: peak of subscription rate
y = month['% Subscription'].max()
x = month['% Subscription'].idxmax()
plt.annotate('March: Peak Subscription rate', xy=(x+0.1, y+0.1),
xytext=(x+1,y+1), arrowprops=dict(facecolor='black', headwidth=6,
width=1, headlength=4), horizontalalignment='left',
verticalalignment='top')

plt.show()

```



Part 5: Machine Learning Classification

```
linkname = '../input/bank_cleaned.csv'
dataset = read_csv(linkname)
dataset = dataset.drop(['Unnamed: 0'], axis=1)
```

```
# View the first 5 rows of cleaned data
dataset.head()
```

```
[35...
  age      job  marital  education  default  balance  housing  loan  day  month  duration  campaign  pdays  previous  poutcome  response  response_binary
0   58  management   married    tertiary    no    2143     yes   no   5   may     4.35         1     -1         0   unknown     no             0
1   44  technician   single    secondary    no      29     yes   no   5   may     2.52         1     -1         0   unknown     no             0
2   33  entrepreneur  married    secondary    no       2     yes  yes   5   may     1.27         1     -1         0   unknown     no             0
3   35  management   married    tertiary    no    231     yes   no   5   may     2.32         1     -1         0   unknown     no             0
4   28  management   single    tertiary    no    447     yes  yes   5   may     3.62         1     -1         0   unknown     no             0
```

Prepare Data for Classification

5.1 Select variables relevant to customers

```
dataset.drop(['marital'],axis=1, inplace=True)
dataset1 = dataset.iloc[:, 0:7]
```

5.2 Tranform categorical data into dummy variables

```
dataset2 = pd.get_dummies(dataset1, columns = ['job'])
dataset2 = pd.get_dummies(dataset2, columns = ['education'])
dataset2['housing'] = dataset2['housing'].map({'yes': 1, 'no': 0})
dataset2['default'] = dataset2['default'].map({'yes': 1, 'no': 0})
dataset2['loan'] = dataset2['loan'].map({'yes': 1, 'no': 0})
dataset_response = pd.DataFrame(dataset['response_binary'])
dataset2 = pd.merge(dataset2, dataset_response, left_index = True,
right_index = True)
```

5.3 Feature Selection

```
array = dataset2.values

# Features: first 20 columns
X = array[:,0:-1]

# Target variable: 'response_binary'
Y = array[:, -1]
```

5.4 Train/Test Split

```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.metrics import accuracy_score
```

```
# 20% of the data will be used for testing
test_size= 0.20
seed = 7
X_train, X_test, Y_train, Y_test= train_test_split(X, Y,
test_size=test_size, random_state=seed)
```

Compare classification algorithms:

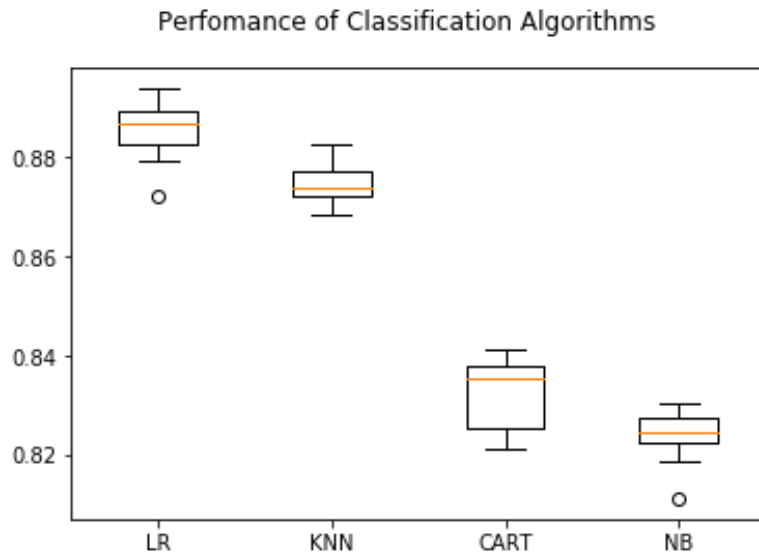
```
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
```

```
models = []
models.append(('LR', LogisticRegression()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
results_c = []
names_c = []

for name, model in models:
    # define how to split off validation data ('kfold' how many folds)
    kfold = KFold(n_splits=10, random_state=seed)
    # train the model
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold,
scoring='accuracy')
    results_c.append(cv_results)
    names_c.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

LR: 0.885284 (0.005933)
KNN: 0.874938 (0.004055)
CART: 0.832394 (0.007199)
NB: 0.823855 (0.005429)

```
fig = plt.figure()
fig.suptitle('Perfomance of Classification Algorithms')
ax = fig.add_subplot(111)
plt.boxplot(results_c)
ax.set_xticklabels(names_c)
plt.show()
```



Logistic regression is the best performing model

Test LR model on test set:

```
LR = LogisticRegression()
LR.fit(X_train, Y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
    intercept_scaling=1, l1_ratio=None, max_iter=100,
    multi_class='warn', n_jobs=None, penalty='l2',
    random_state=None, solver='warn', tol=0.0001, verbose=0,
    warm_start=False)
```

Evaluate LR Model

```
predictions = LR.predict(X_test)
```

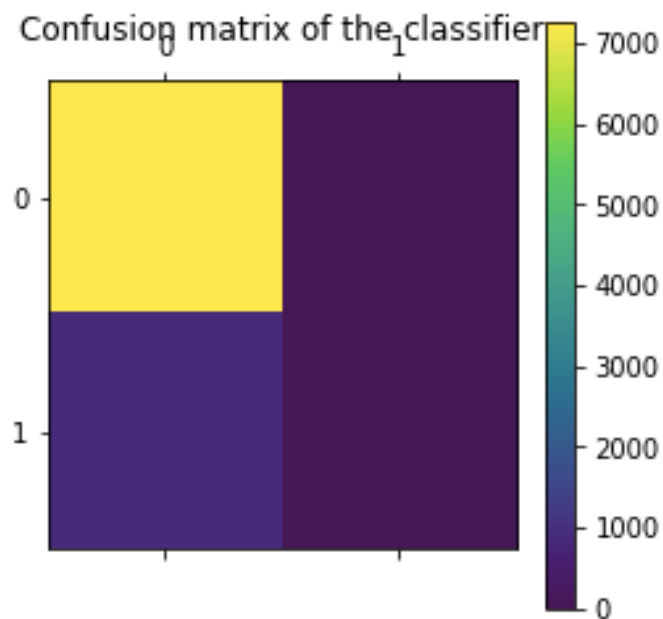
```
# Accuracy Score
print(accuracy_score(Y_test, predictions))
```

0.890806708287428

```
# Confusion Matrix
from sklearn.metrics import confusion_matrix
import pylab as pl

print(confusion_matrix(Y_test, predictions))

cm = confusion_matrix(Y_test, predictions)
pl.matshow(cm)
pl.title('Confusion matrix of the classifier')
pl.colorbar()
pl.show()
```



```
# Classification Report
from sklearn.metrics import classification_report
print(classification_report(Y_test, predictions))
```

	precision	recall	f1-score	support
0	0.89	1.00	0.94	7278
1	0.00	0.00	0.00	891
accuracy			0.89	8169
macro avg	0.45	0.50	0.47	8169
weighted avg	0.79	0.89	0.84	8169

Part 6: Machine Learning Regression:

Prepare Data for Regression

```
dataset4 = dataset2.drop(['response_binary'],axis = 1)
dataset4['duration'] = dataset['duration']
```

Feature Selection

```
array = dataset4.values
X = array[:,0:20]
Y = array[:,20]
```

6.2 Train/test split

```
test_size= 0.20
seed = 10
X_train, X_test, Y_train, Y_test= train_test_split(X, Y,
test_size=test_size, random_state=seed)
```

Compare Regression algorithms

```
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.linear_model import Ridge
from sklearn.linear_model import ElasticNet
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
```

```
models = []
models.append(('LR', LinearRegression()))
models.append(('LASSO', Lasso()))
models.append(('RIDGE', Ridge()))
models.append(('EN', ElasticNet()))
models.append(('KNN', KNeighborsRegressor()))
models.append(('CART', DecisionTreeRegressor()))
```

```
results_e2 = []
names_e2 = []

for name, model in models:
    # define how to split off validation data
    kfold = KFold(n_splits=10, random_state=seed)

    # train the model
```

```

cv_results = cross_val_score(model, X_train, Y_train, cv=kfold,
scoring='neg_mean_squared_error')
results_e2.append(cv_results)
names_e2.append(name)
msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
print(msg)

```

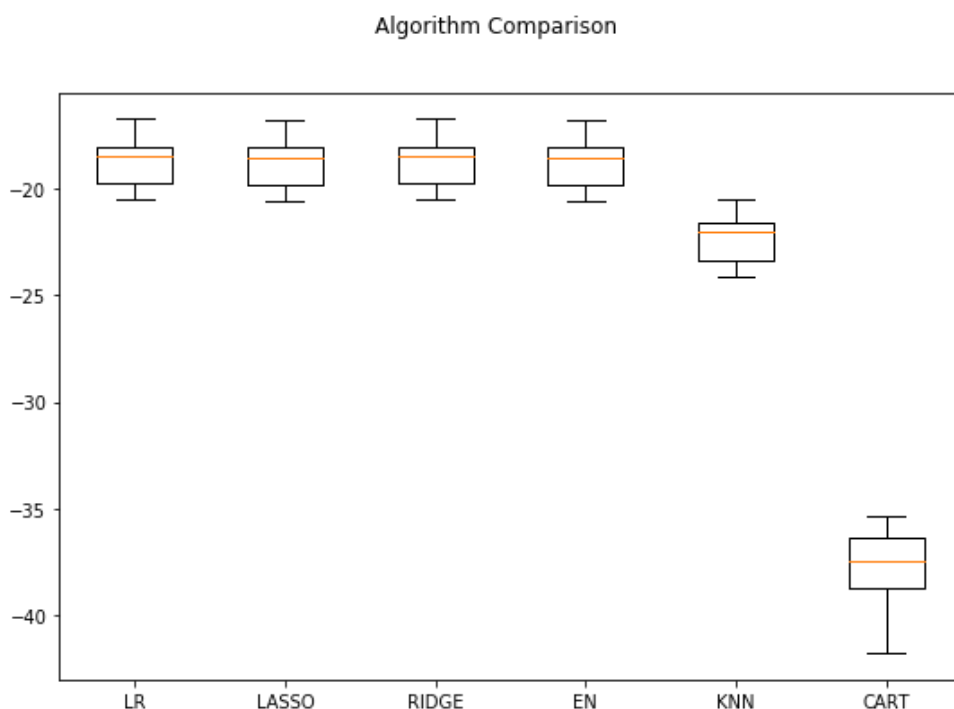
LR: -18.667426 (1.243464)
 LASSO: -18.701201 (1.256921)
 RIDGE: -18.667404 (1.243472)
 EN: -18.700962 (1.256775)
 KNN: -22.237990 (1.196449)
 CART: -37.761878 (1.834286)

```

# Plot results
fig = plt.figure(figsize=(9,6))
ax = fig.add_subplot(111)
plt.boxplot(results_e2)

fig.suptitle('Algorithm Comparison')
ax.set_xticklabels(names_e2)
plt.show()

```



Ridge regression slightly outperforms other models.

Standardize Data

```

from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

pipelines = []
pipelines.append(('ScaledLR', Pipeline([('Scaler',
StandardScaler()), ('LR', LinearRegression())])))
pipelines.append(('ScaledLASSO', Pipeline([('Scaler',
StandardScaler()), ('LASSO', Lasso())])))
pipelines.append(('ScaledRIDGE', Pipeline([('Scaler',
StandardScaler()), ('RIDGE', Ridge())])))
pipelines.append(('ScaledEN', Pipeline([('Scaler',
StandardScaler()), ('EN', ElasticNet())])))
pipelines.append(('ScaledKNN', Pipeline([('Scaler',
StandardScaler()), ('KNN', KNeighborsRegressor())])))
pipelines.append(('ScaledCART', Pipeline([('Scaler',
StandardScaler()), ('CART', DecisionTreeRegressor())])))

results_e2_normalized = []
names_e2_normalized = []

for name, model in pipelines:
    # define how to split off validation data
    kfold = KFold(n_splits=10, random_state=seed)

    # train the model
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold,
scoring='neg_mean_squared_error')
    results_e2_normalized.append(cv_results)
    names_e2_normalized.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)

```

```

ScaledLR: -18.667448 (1.244456)
ScaledLASSO: -18.721218 (1.268503)
ScaledRIDGE: -18.667425 (1.243465)
ScaledEN: -18.721218 (1.268503)

```

Test RIDGE model on test set:

```

# Plot results
fig = plt.figure(figsize=(9,6))
ax = fig.add_subplot(111)
plt.boxplot(results_e2_normalized)

```

```
fig.suptitle('Algorithm Comparison')
ax.set_xticklabels(names_e2_normalized)
plt.show()
ridge = Ridge()
ridge.fit(X_train, Y_train)
predicted_y = ridge.predict(X_test)
```

Evaluate RIDGE Model

```
from sklearn.metrics import mean_squared_error
print('The MSE is', mean_squared_error(Y_test, predicted_y))
```