

PROFFESIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology

(Deemed to be University)

Submitted in partial fulfilment of the requirements for the award

of Bachelor of Technology Degree in

Information Technology

By

SMRITHI.R

(Reg no:38120079)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600119

NOVEMBER 2021



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this project Report is the Bonafide work of who carried out the project entitled "TERM DEPOSIT PREDICTION USING MACHINE LEARNING" under our supervision from JULY 2021 to NOVEMBER 2021.

Internal Guide

Dr.V.Nirmalrani M.E., Ph.D.,

Head of the Department

Dr.R.Subhashini M.E., Ph.D.,

Submitted for viva voce Examination held on _____

Internal Examiner External Examiner

DECLARATION

I hereby declare that the project report entitled “**TERM DEPOSIT PREDICTION USING MACHINE LEARNING**” done by me under the guidance of Dr.V.Nirmalrani M.E., Ph.D., at Sathyabama Institute of Science and Technology (Deemed to be university), Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai -600119 is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:13/11/2021

Place: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge our 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 our thanks to **Dr. T. SASIKALA M.E., Ph.D., Dean, School of Computing** and **Dr. R. SUBHASHINI M.E., Ph.D., Head of the Department, Department of Information Technology** for providing us the necessary support and details at the right time during the progressive reviews.

I would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.V.NIRMALRANIM.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **INFORMATION TECHNOLOGY** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



INTERNSHIP CERTIFICATE



ABSTRACT

TERM DEPOSIT PREDICTION USING MACHINE LEARNING

In recent years, more and more data are being collected from a variety of sources for scientific researches. At the same time, data mining (DM) and machine learning (ML) are being utilized to analyze special features from the data. Meanwhile in business, this combination is able to produce comprehensive overviews to support human decision-making by showing profitable recommendations. In marketing, it is essential for organizations to know when they can provide their services by analyzing customer's data to make crucial strategies for their businesses, especially in banks. In this paper, with the problem of bank marketing, we will take a look at what types of experimental data are typically used, do a preliminary analysis on them, and generate a Term Deposit Subscription prediction model by using machine learning frameworks. Using a database with thousands of data points gathered in a marketing campaign, the accuracy rates of detection and classification are about 71% and 86% respectively. Marketing campaigns are characterized by focusing on the customer needs and their overall satisfaction. Nevertheless, there are different variables that determine whether a marketing campaign will be successful or not. There are certain variables that we need to take into consideration when making a marketing campaign. A Term deposit is a deposit that a bank or a financial institution offers with a fixed rate (often better than just opening a deposit account) in which your money will be returned back at a specific maturity time.

TABLE OF CONTENTS

CHAPTER NO	TITLE NAME	PAGE NO
	ABSTRACT	7
	LIST OF FIGURES	9
	LIST OF ABBREVIATION	10
1	INTRODUCTION	11
	1.1 GENERAL	11
	1.2 OUTLINE OF THE PROJECT	11
2	AIM AND SCOPE	12
	2.1 AIM	12
	2.2 PROBLEM STATEMENT	12
	2.3 SCOPE	12
3	SYSTEM ANALYSIS AND DESIGN	13
	3.1 GENERAL	13
	3.2 SOFTWARES USED	14
	3.3 ALGORITHM	15
	3.4 PROJECT DESCRIPTION	17
4	RESULT AND DISCUSSION	18
	4.1 SYSTEM REQUIREMENTS	18
	4.2 RESULT	18
5	CONCLUSION AND FUTURE WORK	20
	5.1 CONCLUSION	20
	5.2 FUTURE WORK	20
	5.3 REFERENCE	20
	APPENDIX	21
	A. SCREENSHOTS	21
	B. SOURCE CODE	22

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
3.1	DATA FLOW DIAGRAM	17
4.1	PIE CHART	19
4.1	BAR CHART	19

LIST OF ABBREVIATIONS

ABBREVIATIONS	EXPANSION
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
UI	USER INTERFACE
KNN	K NEAREST NEIGHBOUR
SVM	SUPPORT VECTOR MACHINE

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Marketing to potential clients has always been a crucial challenge in attaining success for banking institutions. It's not a surprise that banks usually deploy mediums such as social media, customer service, digital media and strategic partnerships to reach out to customers. The investment and portfolio department of the Bank would want to be able to identify their customers who potentially would subscribe to their term deposits. As there has been heightened interest of marketing managers to carefully tune their directed campaigns to the rigorous selection of contacts, the task at hand is to find a model that can predict which future clients who would subscribe to the Bank's term deposit. In this post, am going to talk about data exploration, data cleaning, feature extraction, and dealing with class imbalance and developing a robust machine learning algorithm for predicting which potential customers would subscribe to the term deposit using different Machine learning library.

1.3 OUTLINE OF THE PROJECT

This project aims to predict if a customer subscribes to a term deposit or not, when contacted by a marketing agent, by understanding the different features and performing predictive analytics.

CHAPTER 2

AIM AND SCOPE

2.1 AIM:

The main objective of this project is to build a model that predicts the customers that would or would not subscribe to bank term deposits using Machine Learning Techniques.

2.2 PROBLEM STATEMENT:

To Predict if a customer subscribes to a term deposit or not, when contacted by a marketing agent, by understanding the different features and performing predictive analytics

2.3 SCOPE:

This project will investigate how machine learning techniques can be used to predict the customer's subscriptions to a term deposit as to whether they will subscribe or not. This aims to develop a model in which there will be a dataset storing all the dataset consists of several predictor variables and one target variable, Outcome. Predictor variables include the age, job, marital status, and so on. The main purpose of the prediction is to reduce uncertainty associated to investment decision making and to avoid discrepancy in the deposition of money. The machine learning algorithm that was considered in this project includes Logistic Regression and several other methods which will predict data in an accurate manner.

CHAPTER 3

SYSTEM ANALYSIS

3.1 GENERAL:

3.1.1 MACHINE LEARNING:

Machine Learning is said as a subset of **artificial intelligence** that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own.

3.1.2 PYTHON:

Python is an interpreted, high-level and general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

3.1.3 IDE (INTEGRATED DEVELOPMENT ENVIRONMENT)

IDE stands for Integrated Development Environment. It's a coding tool which allows you to write, test, and debug your code in an easier way, as they typically offer code completion or code insight by highlighting, resource management, debugging tools. And even though the IDE is a strictly defined concept, it's starting to be redefined as other tools such as notebooks start gaining more and more features that traditionally belong to IDEs. The IDE used in this project is Colab.

3.1.4 USER INTERFACE

The user interface (UI) is the point of human-computer interaction and communication in a device. This can include display screens, keyboards, a mouse

and the appearance of a desktop. It is also the way through which a user interacts with an application or a website.

3.2 SOFTWARES USED

3.2.1 CODE EDITORS USED

COLAB

Colaboratory, or “Colab” for short, is a product from Google Research. Colab 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. Colaboratory, or Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

3.2.2 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.

3.2.3 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 analyse 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 dataframes 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 modeling including classification, regression, clustering and dimensionality reduction.

3.3 ALGORITHM

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.

LINEAR DISCRIMINANT ANALYSIS ALGORITHM:

Linear Discriminant Analysis or LDA is a dimensionality reduction technique. It is used as a pre-processing step in Machine Learning and applications of pattern classification. ... LDA is a supervised classification technique that is considered a part of crafting competitive machine learning models.

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.

CART ALGORITHM:

The CART algorithm is a type of classification algorithm that is required to build a decision tree on the basis of Gini'simpurityindex. It is a basic machine learning algorithm and provides a wide variety of use cases.It is used to describe Decision Tree algorithms that may be used for classification or regression predictive modeling issues.

GAUSSIAN NB ALGORITHM:

A Gaussian Naive Bayes algorithm is a special type of NB(Naïve 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

3.4 PROJECT DESCRIPTION

The project aims 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.

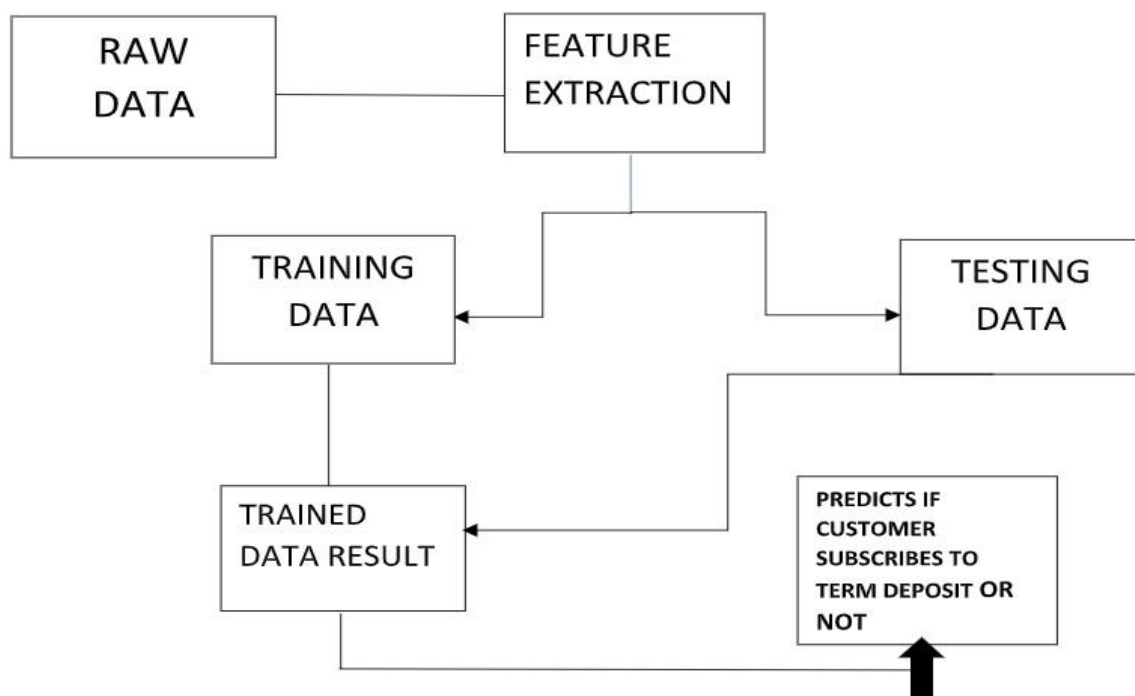


FIG 3.1 DATA FLOW

CHAPTER 4

RESULTS AND DISCUSSION

4.1 SYSTEM REQUIREMENTS

- PROCESSOR: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz
- RAM: 8GB
- SYSTEM TYPE: 64 bit Operating System
- STORAGE: 500 GB

4.2 RESULT

As a result, The main aim of the research was to analyze the Customer behavior for making a Term deposit in banks, by considering some basic attributes that can be collected and analyzed easily. The research was evaluated using seven different machine learning techniques (Logistic Regression, LDA, KNN, SVM, Decision Trees (CART), Gaussian NB, SVM, Random Forest Classifier,). Also, all these techniques were tested using Python. The best results were obtained for performing Random Forest Classifier with the highest predictive accuracy of 91 percent. The implemented system performs a predictive analysis for the customers who are more responsive to apply for the term-deposits in the bank. The model can help banks in focusing and providing services to the customers who have more probability to apply for term-deposits in the bank. In Future, the model can be improved by considering a large dataset as compared to our model. Also, some ensemble models can be used to get more accuracy.

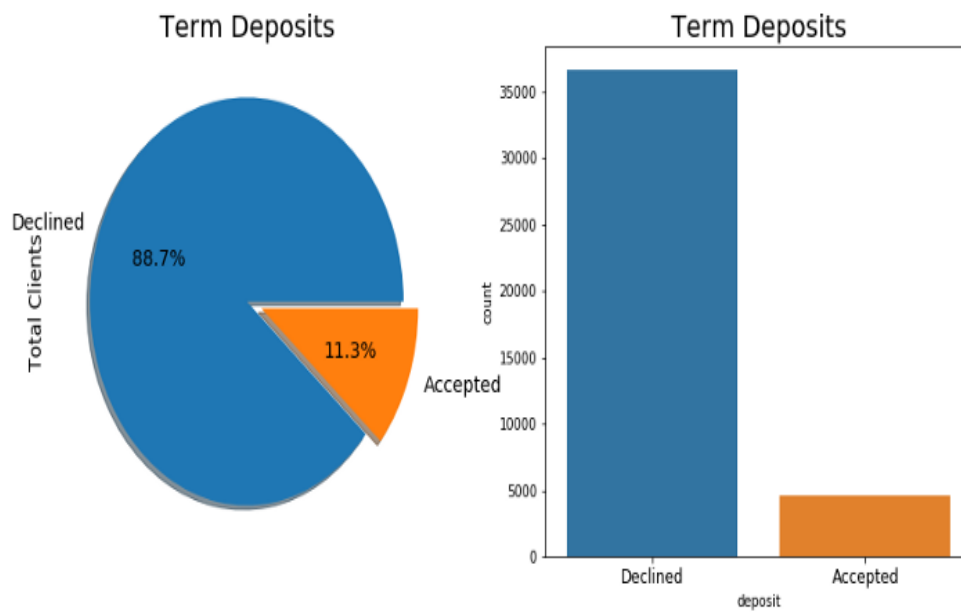


FIG 4.1 PIE CHART AND BAR CHART

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

This project is undertaken using machine learning and evaluates the performance of prediction by using logistic regression algorithm, Linear Discriminant Analysis(LDA) algorithm, KNN, CART, SVM, Gaussian NB and Random Forest Classifier Algorithm. In our proposed model, all the methods are involved. The implementation of above system would help in better prediction of whether the customer is interested to subscribe to a term deposit or not, than by assumptions or manual methods.

5.2 FUTURE WORK

This project describes prediction of subscriptions to term deposits in Banks. In future the Bank tellers can predict the subscription of customers to the term deposit using the computerised method unlike the normal method using the machine Learning techniques discussed proposed in this project.

5.3 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)

APPENDIX

A.SCREENSHOTS

Splitting Of Data

```
[ ] #splitting of data
from sklearn.model_selection import train_test_split

Xtrain,Xtest,ytrain,ytest = train_test_split(X,y,test_size=0.3,random_state=10)

Xtrain.shape,Xtest.shape,ytrain.shape,ytest.shape

((28831, 63), (12357, 63), (28831,), (12357,))

[ ] #Logistic Regression
from sklearn.linear_model import LogisticRegression

[ ] model = LogisticRegression()

[ ] model

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

Building and Evaluation Of Models

```
[ ] # Building models
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

# Spot Checking Algorithms
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
models.append(('DecisionTree', DecisionTreeClassifier()))
models.append(('Random Forest', RandomForestClassifier()))

[ ] # evaluation of each model
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=None)
    cv_results = model_selection.cross_val_score(model, Xtrain, ytrain, cv=kfold, scoring='accuracy')
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

Showing the prediction

```
[ ] # Making predictions on validation dataset
rf = RandomForestClassifier()
rf.fit(Xtrain, ytrain)
predictions = rf.predict(Xtest)
print(accuracy_score(ytest, predictions))
print(confusion_matrix(ytest, predictions))
print(classification_report(ytest, predictions))

0.9116290361738286
[[10629  306]
 [ 786  636]]
      precision    recall  f1-score   support

     0       0.93      0.97      0.95     10935
     1       0.68      0.45      0.54      1422

 accuracy          0.80          0.91     12357
 macro avg          0.80          0.71     12357
 weighted avg          0.90          0.90     12357

[ ]
```

B. SOURCE CODE

#importing Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
data = pd.read_csv('/content/bank-additional-file.csv')
data.dtypes
data.columns
```

#columns which are of category type

```
data_category_column = data.columns[data.dtypes=='object']
```

#columns which are of numeric type

```
data_numeric_column = data.columns[data.dtypes!='object']
```

#category columns names

```
print(data_category_column)
```

#numeric columns names

```
print(data_numeric_column)
```

Creating data sub sets

#Category Data

```
data_category = data[data_category_column]
```

#Numeric Data

```
data_numeric = data[data_numeric_column]
```

```
data_category.shape
```

```
data_numeric.shape
```

```
data_category.head()
```

```
data_numeric.head()
```

Checking for missing data

```
data_category.isna().sum().sort_values(ascending=False)
data_category.isna().sum().sort_values(ascending=False)
data_category.isna().sum().sort_values(ascending=False)
data_category['y'] = data_category['y'].map({'yes':1,"no":0}) #Encoding y
data_category['y'].value_counts()
```

Using a loop for converting all object columns to category

```
for column in data_category.columns:
    data_category[column] = data_category[column].astype('category')
data_category.dtypes
data_category.head()
data_numeric.head()
data_numeric.dtypes
category_count = data_category.columns.value_counts()
```

#barplot for category count

```
sns.barplot(category_count.index,category_count.values,alpha=0.9)
plt.show()
```

#barplot for numeric count

```
numeric_count = data_numeric.columns.value_counts()
sns.barplot(numeric_count.index,numeric_count.values,alpha=0.9)
plt.show()
data_category_X = data_category.drop(columns=['y'])
data_category_Y = data_category['y']
data_category_X = pd.get_dummies(data_category_X)
data_category_X.shape,data_category_Y.shape,data_numeric.shape
data_category_X.dtypes
data_category_Y.value_counts()
```

#Merging all data

Integrating category data with numeric data

```
termdeposit_data = pd.concat([data_numeric,data_category_X,data_category_Y],
join='outer',axis=1)
```

```
y = termdeposit_data.head()
```

```
X = termdeposit_data.drop(columns=['y'])
```

```
y = termdeposit_data['y']
```

```
X.shape,y.shape
```

#Displaying data of X in the form of heatmap

```
sns.heatmap(X)
```

Converting X and y to arrays

```
X = termdeposit_data.iloc[:, :-1].values #features
```

```
y = termdeposit_data.iloc[:, -1].values #target
```

```
X,y,X.shape,y.shape
```

#splitting of data

```
from sklearn.model_selection import train_test_split
```

```
Xtrain,Xtest,ytrain,ytest = train_test_split(X,y,test_size=0.3,random_state=10)
```

```
Xtrain.shape,Xtest.shape,ytrain.shape,ytest.shape
```

#Logistic Regression

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model
model.fit(Xtrain,ytrain)
ypred=model.predict(Xtest)
ytest
model.coef_ #weights or coefficient
model.intercept_
#model selection
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
#accuracy score
print(sklearn.metrics.accuracy_score(ytest,ypred))
#classification_report
print(sklearn.metrics.classification_report(ytest,ypred))
#confusion_matrix
print(sklearn.metrics.confusion_matrix(ytest,ypred))
depositors = pd.DataFrame(ypred)
depositors.value_counts
print('Default Logistic Regression Accuracy Training Score',model.score(Xtrain,ytrain))
print('Default Logistic Regression Accuracy Test Score',model.score(Xtest,ytest))
Comparison of Various Models
```

Building models

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
```

Spot Checking Algorithms

```
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
models.append(('DecisionTree',DecisionTreeClassifier()))
models.append(('Random Forest',RandomForestClassifier()))
```

evaluation of each model

```
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=None)
```

```

cv_results = model_selection.cross_val_score(model, Xtrain, ytrain, cv=kfold, scor
ing='accuracy')
results.append(cv_results)
names.append(name)
msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
print(msg)
# Making predictions on validation dataset
rf = RandomForestClassifier()
rf.fit(Xtrain, ytrain)
predictions = rf.predict(Xtest)
print(accuracy_score(ytest, predictions))
print(confusion_matrix(ytest, predictions))
print(classification_report(ytest, predictions))

```