PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

MANOJ SIRIGIRI
REG. NO. 39110604

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

NOVEMBER 2021

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI– 600119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **MANOJ SIRIGIRI (Reg. No: 39110604)** who carried out the project entitled “**Predicting Term Deposit Subscription by a client”**

under my supervision from June 2021 to November 2021.

Internal Guide
Dr. G.Kalaiarasi M.E., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on ____________________

InternalExaminer **ExternalExaminer**

DECLARATION

I, **MANOJ SIRIGIRI** hereby declare that the project report entitled **Predicting Term Deposit Subscription by a client** done by me under the guidance of **Dr. G.Kalaiarasi M.E., Ph.D.,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

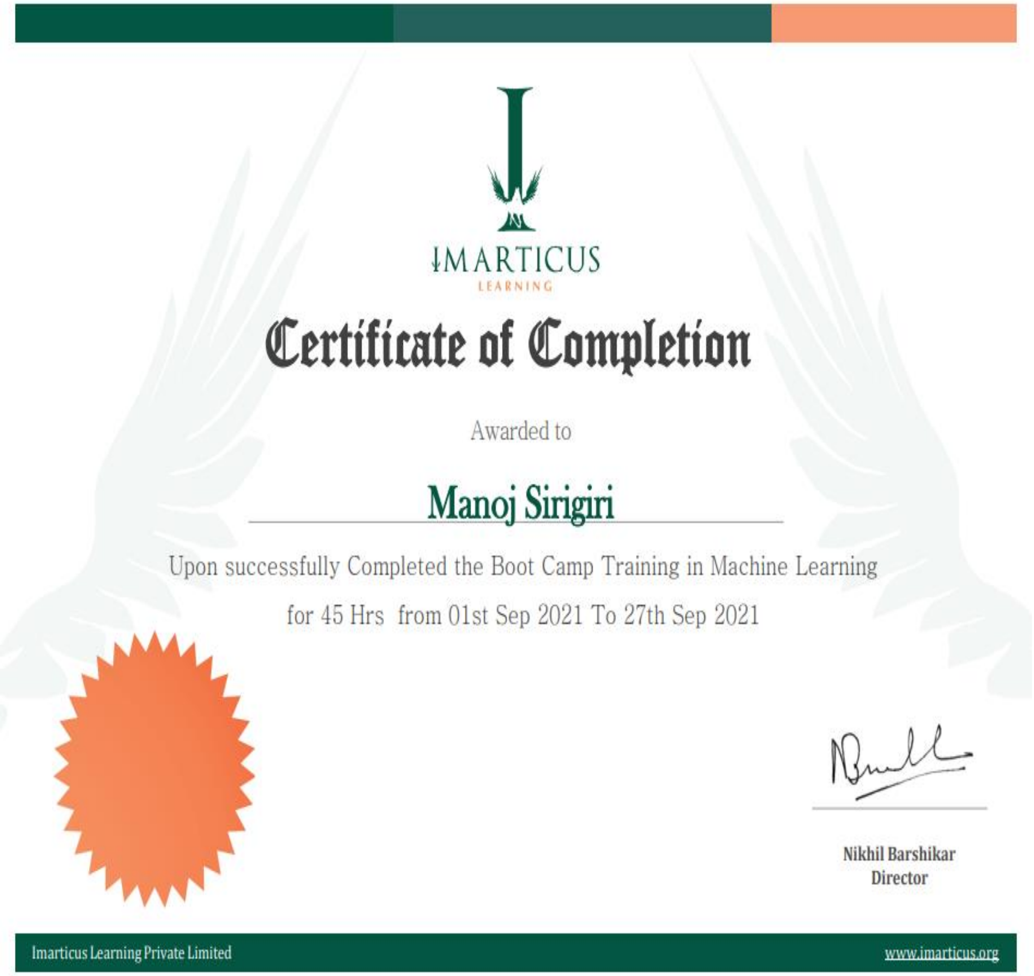
I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

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

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

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.

TRAINING CERTIFICATE



IMARTICUS
LEARNING

Certificate of Completion

Awarded to

Manoj Sirigiri

Upon successfully Completed the Boot Camp Training in Machine Learning

for 45 Hrs from 01st Sep 2021 To 27th Sep 2021

Nikhil Barshikar
Director

Imarticus Learning Private Limited

www.imarticus.org

ABSTRACT

I propose a data mining (DM) approach to predict the success of telemarketing calls for selling bank long-term deposits. A Portuguese retail bank was addressed, with data collected from 2008 to 2013, thus including the effects of the recent financial crisis. Ianalyzed a large set of 150 features related with bank client, product and social-economic attributes. A semi-automatic feature selection was explored in the modeling phase, performed with the data prior to July 2012 and that allowed to select a reduced set of 22 features. Ialso compared four DM models: logistic regression, decision trees (DT), neural network (NN) and support vector machine. Using two metrics, area of the receiver operating characteristic curve (AUC) and area of the LIFT cumulative curve (ALIFT), the four models were tested on an evaluation phase, using the most recent data (after July 2012) and a rolling windows scheme. The NN presented the best results (AUC = 0.8 and ALIFT = 0.7), allowing to reach 79% of the subscribers by selecting the half better classified clients. Also, two knowledge extraction methods, a sensitivity analysis and a DT, were applied to the NN model and revealed several key attributes (e.g., Euribor rate, direction of the call and bank agent experience). Such knowledge extraction confirmed the obtained model as credible and valuable for telemarketing campaign managers.

CONTENTS

LIST OF ABBREVIATIONS

S.NO	SHORTCUT	ABBREVATION
1	RFC	RANDOM FOREST CLASSIFICATION
2	LR	LOGISTIC REGRESSION
3	ML	MACHINE LEARNING
4	SVM	SUPPORT VECTOR MACHINE

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Marketing is technique of exposing the target clients to a product via suitable systems and channels. It ultimately facilitates the way to buy the product or service and even helps in determining the need of the product and persuade customers to buy it. The overall aim is to increase sales of products and services for enterprise, business and financial institutions. It also helps to maintain the reputation of the company.

Telemarketing is form of direct marketing in which salesperson approaches the customer either face to face or phone call and persuade him to buy the product. Telemarketing attains most popularity in 20th century and still gaining it. Nowadays, telephone (fixed-line or mobile) has been broadly used. It is cost effective and keeps the customers up to date. In Banking sector, marketing is the backbone to sell its product or service. Banking advertising and marketing is mostly based on an intensive knowledge of objective information about the market and the actual client needs for the bank profitable manner.

Making right decisions in organizational operations are sometimes proved a great challenge where the quality of decision really matters. Decision Support Systems (DSS) are classified as a particular class of computerized facts and figures that helps the organization or administration into their decision-making actions. The concept of DSS originates from a balance which lies between the data generated by computer and the judgment of human. According to Rupnik & Kukar (2007) the objective of decision support systems is to enhance the effectiveness of the decisions. This is a great tool which can analyze the sales data and provide further predictions. The purposes which can be established from the DSS are such as, analysis, optimization, forecasting and simulation. A study by Power (2008) found that research subjects who use DSS for the decision making, come-up with more effective decisions than those who did not use it. Nowadays, DSS is contributing a meaningful role in many fields such as for medical diagnosis, business and management, investment portfolios, command and control of military units, and statistics.

DSS uses statistical data to overcome the deficiencies and helps the decision makers to take the right decision. Data mining (DM) plays vital role to support the Decision support systems which are based on the data obtained from the data mining models: rules, patterns and relationship. Data mining is the process of selecting, discovering, and modeling high volume of data to find and clarify unknown patterns. The objective of data mining in decision support systems is to suggest a tool which is easily accessible for the business users to analyze the data mining models.

A specific technology used within the DSS is Machine learning (ML) that combines data and computer applications to accurately predicting the results. The fundamental principle of machine learning is to construct the algorithms that can obtain input data and then predict the results or outputs by using the statistical analysis within satisfactory interval. ML allows the DSS to obtain the new knowledge which helps it to make right decisions.

Machine Learning can be mainly classified in 2 categories i.e. supervised learning and unsupervised learning. In supervised learning, the output of algorithm is already known and Iuse the input data to predict the output. The examples of supervised learning are regression and classification. In contrast, unsupervised learning Ionly have input data whereas no corresponding output variables are selected.

CHAPTER 2
AIM AND SCOPE OF THE PRESENT INVESTIGATION

2.1 AIM:

The main aim of the project is to predict whether a Bank client will subscribe to a term deposit.

2.2 PROJECT SCOPE:

The purpose of this study is to check the accuracy and performance of several models for classification. The full model which is used in this study consists of 16 independent variables. Feature selection approach has been used to select the best subsets of variables and then different type of classification algorithms have been utilized to check their accuracy and performance. The full model is then compared with the reduced model, obtained through feature selection, in terms of classification accuracy.

2.3 OVERALL OBJECTIVE:

This analysis aims to develop a predictive model that helps the firm to understand and predict which client is more likely to accept the subscription. For the same, perform data preprocessing tasks, split the data into train and test, build a model upon training data (70%) and evaluate the model on the test data (30%). Use the random_state =10 while splitting the data. Apply multiple algorithms to build the model, evaluate the model using various performance metrics and justify which is the best algorithm suitable for the data.

2.4 PROPOSED SOLUTION:

Machine learning is a computer system's method of learning by way of examples. There are many machine learning algorithms available to users that can be implemented on datasets. As the current project is about a supervised machine learning problem, implemented the supervised machine learning algorithms. The algorithms are given a particular attribute or set of attributes to predict. Data preprocessing process includes methods to remove any null values or infinite values which may affect the accuracy of the system. The main steps include Formatting, cleaning and sampling. Cleaning process is used for removal or fixing of some missing data there may be data that are incomplete. Implemented Random Forest and the Logistic machine learning models on the dataset after the preprocessing.

CHAPTER 3

3 EXPERIMENTAL OR MATERIALS AND METHODS, ALGORITHMS USED

3.1 MATERIAL REQUIREMENTS AND TECHNOLOGIES USED:

Requirements are the basic constrains that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements
2. Non-Functional requirements
3. Environment requirements
    A. Hardware requirements
    B. software requirements

3.1.1 FUNCTIONAL REQUIREMENTS:

The software requirements specification is a technical specification of requirements for the software product. It is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details to follow the special libraries like sk-learn, pandas, NumPy, matplotlib and seaborn.

3.1.2 NON-FUNCTIONAL REQUIREMENTS:

Process of functional steps,

1. Defining the business problem
2. Preparing data
3. Data Preprocessing
4. Data Visualization
5. Evaluating algorithms
6. Improving the results
7. Prediction of the result

3.1.3 ENVIRONMENTAL REQUIREMENTS:

1. Software Requirements:

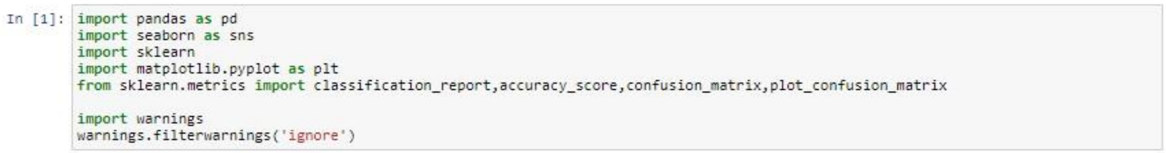
Operating system	:	windows
Tool	:	Anaconda with Jupyter notebook

2. Hardware requirements:

Processor	:	Pentium IV/III
Hard disk	:	minimum 80 GB
RAM	:	minimum 2 GB

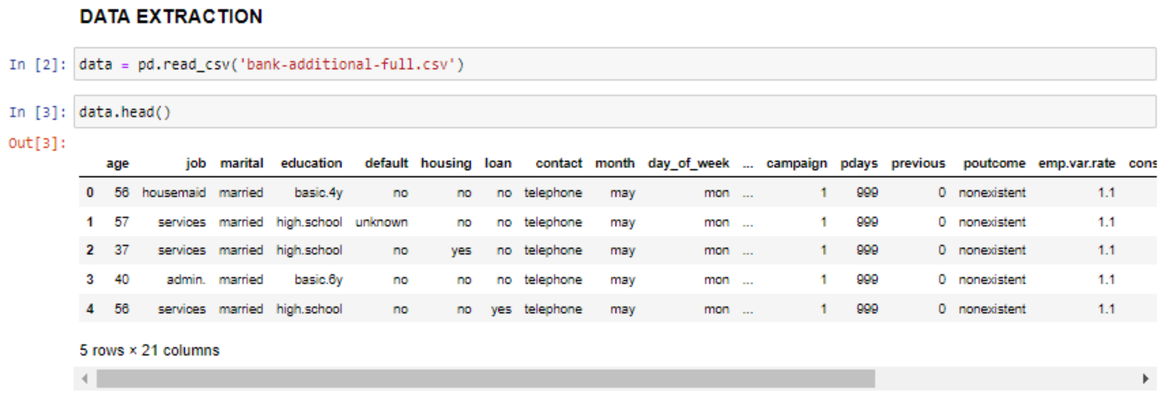
3.2 DATA PREPROCESSING:

- Importing the required modules



```
In [1]: import pandas as pd
        import seaborn as sns
        import sklearn
        import matplotlib.pyplot as plt
        from sklearn.metrics import classification_report,accuracy_score,confusion_matrix,plot_confusion_matrix

        import warnings
        warnings.filterwarnings('ignore')
```

- Importing csv file and extracting data



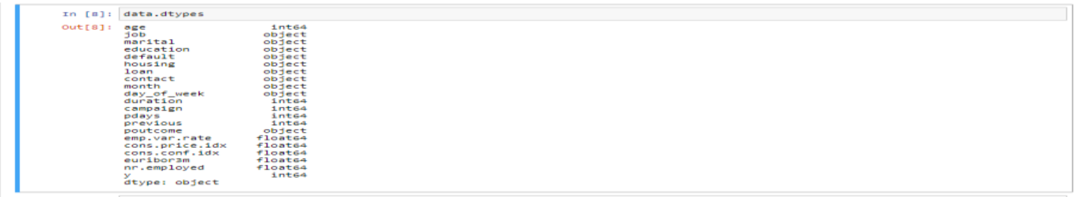
DATA EXTRACTION

```
In [2]: data = pd.read_csv('bank-additional-full.csv')

In [3]: data.head()

Out[3]:
```

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.var.rate	cons
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
2	37	services	married	high.school	no	yes	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
4	56	services	married	high.school	no	no	yes	telephone	may	mon	...	1	999	0	nonexistent	1.1	

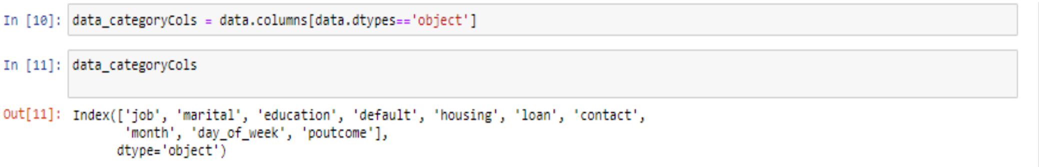
5 rows × 21 columns

- Checking data types of variables



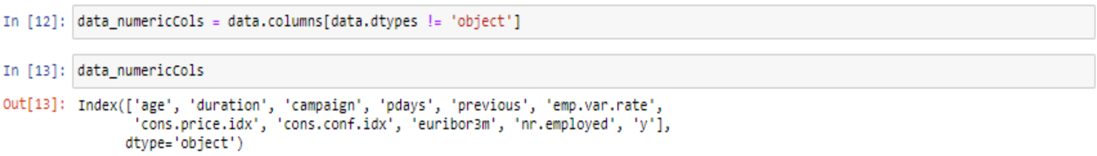
```
In [8]: data.dtypes
Out[8]: age                 int64
        job                object
        marital            object
        education          object
        default            object
        housing            object
        loan               object
        contact            object
        month              object
        day_of_week        object
        duration            int64
        campaign            int64
        pdays               int64
        previous            int64
        poutcome           object
        emp.var.rate      float64
        cons.price.idx    float64
        cons.conf.idx     float64
        euribor3m         float64
        nr.employed       float64
        y                   int64
        dtype: object
```

- Separating columns with category data type



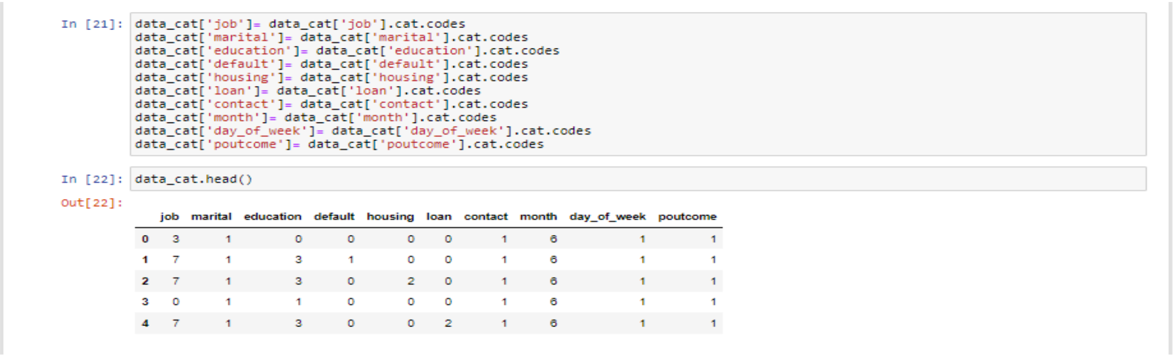
```
In [10]: data_categoryCols = data.columns[data.dtypes=='object']

In [11]: data_categoryCols

Out[11]: Index(['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact',
                'month', 'day_of_week', 'poutcome'],
               dtype='object')
```

- Separating columns with numerical data type



```
In [12]: data_numericCols = data.columns[data.dtypes != 'object']

In [13]: data_numericCols

Out[13]: Index(['age', 'duration', 'campaign', 'pdays', 'previous', 'emp.var.rate',
                'cons.price.idx', 'cons.conf.idx', 'euribor3m', 'nr.employed', 'y'],
               dtype='object')
```

- Converting categorical data numerical data

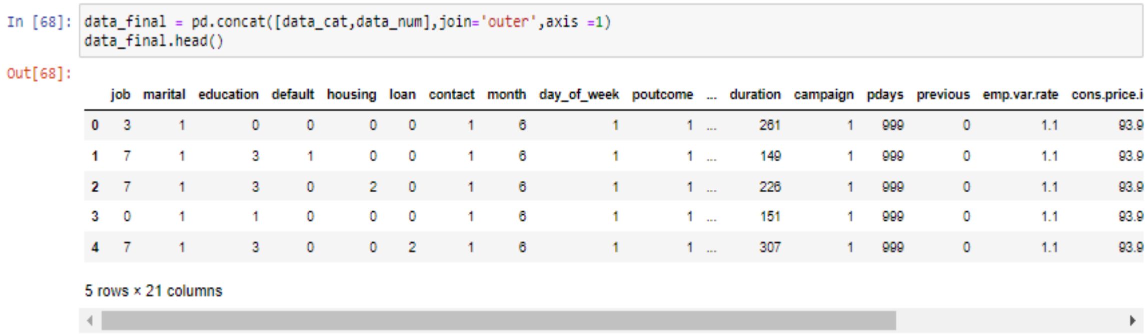


```
In [21]: data_cat['job']= data_cat['job'].cat.codes
         data_cat['marital']= data_cat['marital'].cat.codes
         data_cat['education']= data_cat['education'].cat.codes
         data_cat['default']= data_cat['default'].cat.codes
         data_cat['housing']= data_cat['housing'].cat.codes
         data_cat['loan']= data_cat['loan'].cat.codes
         data_cat['contact']= data_cat['contact'].cat.codes
         data_cat['month']= data_cat['month'].cat.codes
         data_cat['day_of_week']= data_cat['day_of_week'].cat.codes
         data_cat['poutcome']= data_cat['poutcome'].cat.codes

In [22]: data_cat.head()
```

Out[22]:

	job	marital	education	default	housing	loan	contact	month	day_of_week	poutcome
0	3	1	0	0	0	0	1	6	1	1
1	7	1	3	1	0	0	1	6	1	1
2	7	1	3	0	2	0	1	6	1	1
3	0	1	1	0	0	0	1	6	1	1
4	7	1	3	0	0	2	1	6	1	1

- Converting categorical data numerical data

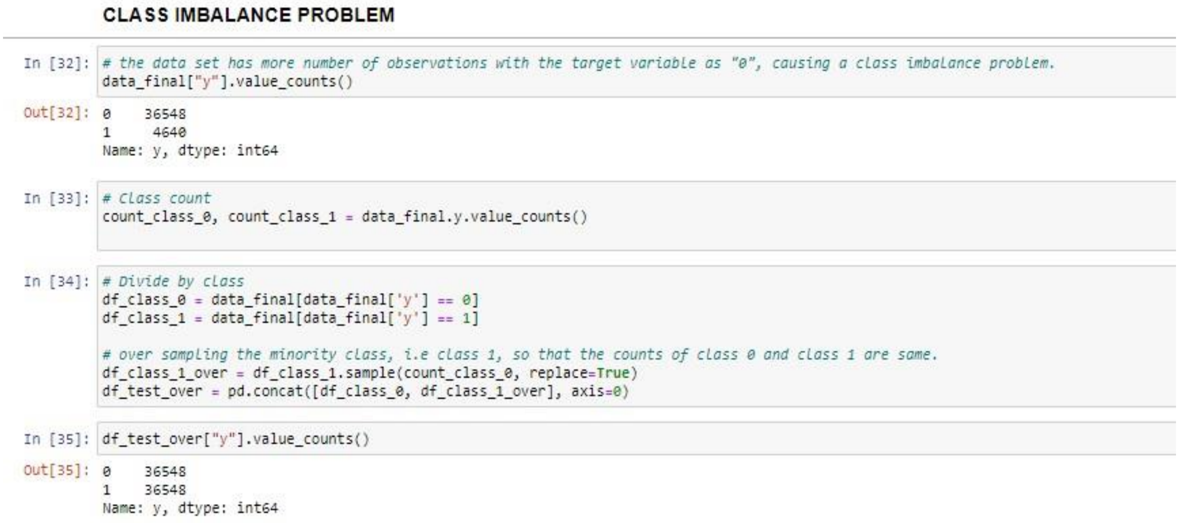


```
In [68]: data_final = pd.concat([data_cat,data_num],join='outer',axis =1)
         data_final.head()
Out[68]:
```

	job	marital	education	default	housing	loan	contact	month	day_of_week	poutcome	...	duration	campaign	pdays	previous	emp.var.rate	cons.price.i
0	3	1	0	0	0	0	1	6	1	1	...	261	1	999	0	1.1	93.9
1	7	1	3	1	0	0	1	6	1	1	...	149	1	999	0	1.1	93.9
2	7	1	3	0	2	0	1	6	1	1	...	226	1	999	0	1.1	93.9
3	0	1	1	0	0	0	1	6	1	1	...	151	1	999	0	1.1	93.9
4	7	1	3	0	0	2	1	6	1	1	...	307	1	999	0	1.1	93.9

5 rows × 21 columns

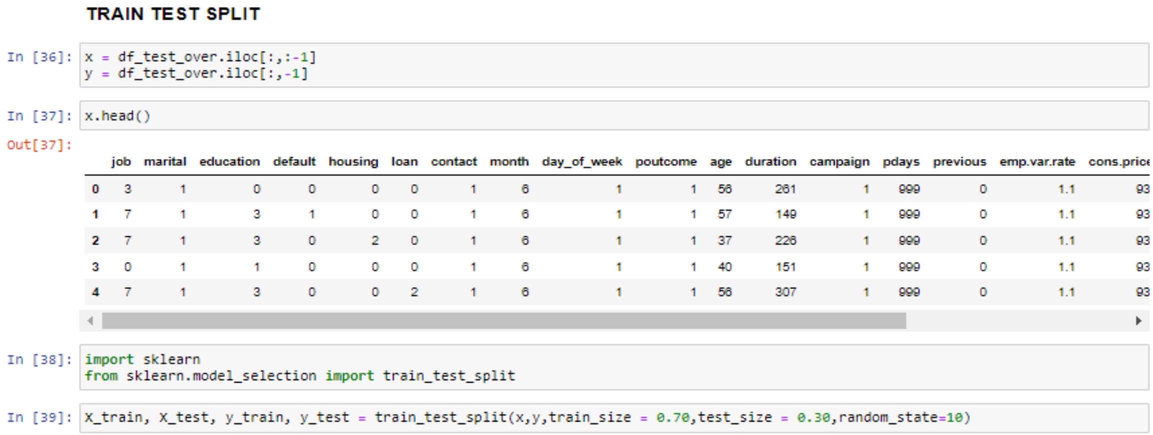
- class imbalance problem



CLASS IMBALANCE PROBLEM

```
In [32]: # the data set has more number of observations with the target variable as "0", causing a class imbalance problem.
         data_final["y"].value_counts()

Out[32]: 0    36548
         1     4640
         Name: y, dtype: int64

In [33]: # Class count
         count_class_0, count_class_1 = data_final.y.value_counts()

In [34]: # Divide by class
         df_class_0 = data_final[data_final['y'] == 0]
         df_class_1 = data_final[data_final['y'] == 1]

         # over sampling the minority class, i.e class 1, so that the counts of class 0 and class 1 are same.
         df_class_1_over = df_class_1.sample(count_class_0, replace=True)
         df_test_over = pd.concat([df_class_0, df_class_1_over], axis=0)

In [35]: df_test_over["y"].value_counts()

Out[35]: 0    36548
         1    36548
         Name: y, dtype: int64
```

- splitting test data and train data



TRAIN TEST SPLIT

```
In [36]: x = df_test_over.iloc[:,:-1]
         y = df_test_over.iloc[:,-1]

In [37]: x.head()

Out[37]:
```

	job	marital	education	default	housing	loan	contact	month	day_of_week	poutcome	age	duration	campaign	pdays	previous	emp.var.rate	cons.price
0	3	1	0	0	0	0	1	6	1	1	56	261	1	999	0	1.1	93
1	7	1	3	1	0	0	1	6	1	1	57	149	1	999	0	1.1	93
2	7	1	3	0	2	0	1	6	1	1	37	226	1	999	0	1.1	93
3	0	1	1	0	0	0	1	6	1	1	40	151	1	999	0	1.1	93
4	7	1	3	0	0	2	1	6	1	1	56	307	1	999	0	1.1	93

```
In [38]: import sklearn
         from sklearn.model_selection import train_test_split

In [39]: x_train, x_test, y_train, y_test = train_test_split(x,y,train_size = 0.70,test_size = 0.30,random_state=10)
```

3.3 METHODS AND ALGORITHMS USED:

Since this is a classification problem with binary response, the method Iattempt to try includes logistic regression, random Forest. In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class. (like identifying whether the person is subscribed to term deposit or not, whether person is interested to subscribe) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc. In Supervised Learning, algorithms learn from labeled data. After understanding the data, the algorithm determines which label should be given to new data based on pattern and associating the patterns to the unlabeled new data.

Used Python Packages:

sklearn:

- In python, sklearn is a machine learning package which include a lot of ML algorithms.
- Here, Iare using some of its modules like train_test_split, Logistic Regression and accuracy_score.

NumPy:

- It is a numeric python module which provides fast math functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

Pandas:

- Used to read and write different files.
- Data manipulation can be done easily with data frames.

Matplotlib:

- Data visualization is a useful way to help with identify the patterns from given dataset.

- Data manipulation can be done easily with data frames.

3.4 Logistic Regression:

Logistic Regression, or logit regression, is a kind of probabilistic statistical classification model. It is also used to predict a binary response from a binary predictor. What’s better, logistics model doesn’t suffer a lot from severe class imbalance.

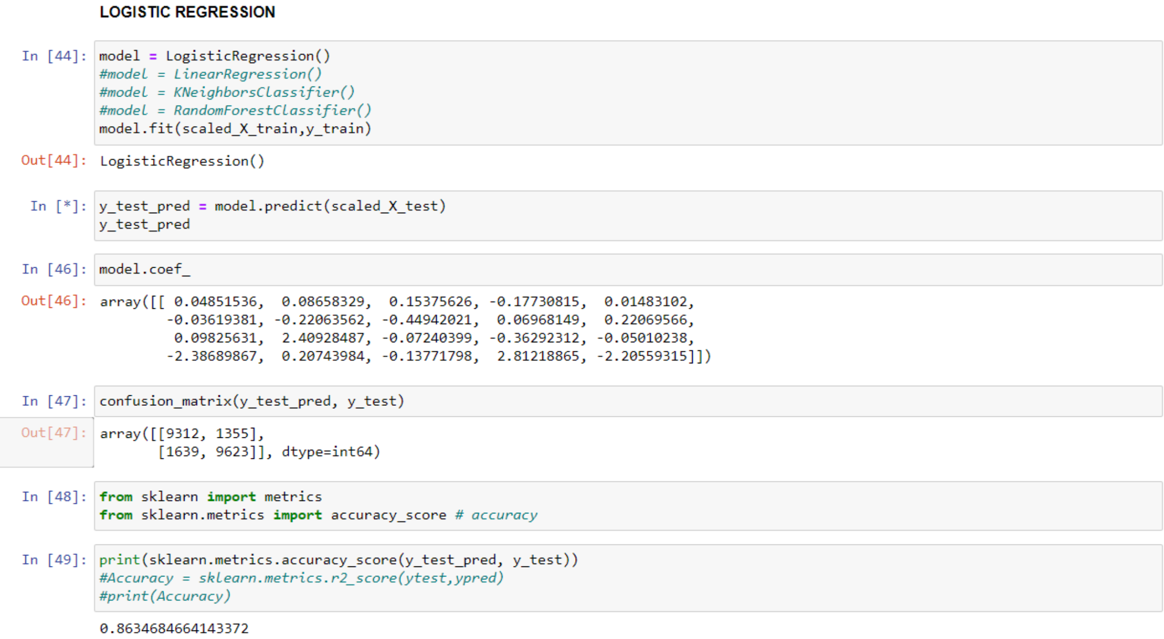
Logistic Regression models the log odds of the event as a linear function:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_P x_P$$

$$p = \frac{1}{1+\exp\left[-(\beta_0 + \beta_1 x_1 + \cdots + \beta_P x_P)\right]}$$

This nonlinear function is a sigmoidal function of the model terms and constraints the probability estimates to between 0 and 1. Also, this model produces linear class boundaries, unless the predictors used in the model are nonlinear transformations of the original features. Many of the categorical predictors in our project have sparse and unbalanced distributions. Because of this, Iwould expect that a model using the full set of predictors would perform worse than the set that has near-zero variance predictors removed.

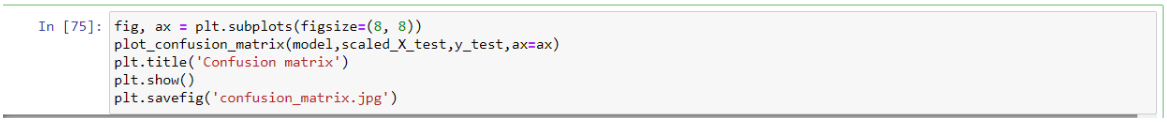
3.4.1Using logistic regression on client data set:



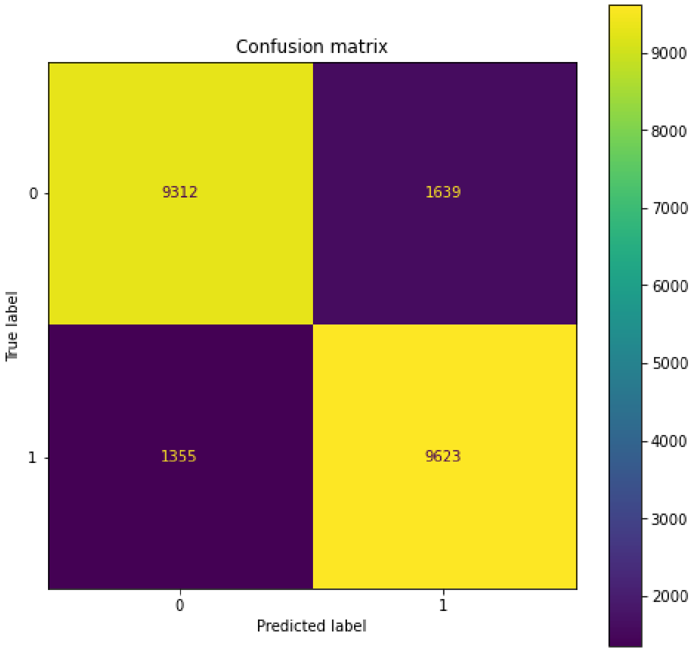
LOGISTIC REGRESSION

```
In [44]: model = LogisticRegression()
         #model = LinearRegression()
         #model = KNeighborsClassifier()
         #model = RandomForestClassifier()
         model.fit(scaled_X_train,y_train)

Out[44]: LogisticRegression()

 In [*]: y_test_pred = model.predict(scaled_X_test)
         y_test_pred

In [46]: model.coef_

Out[46]: array([[ 0.04851536,  0.08658329,  0.15375626, -0.17730815,  0.01483102,
                 -0.03619381, -0.22063562, -0.44942021,  0.06968149,  0.22069566,
                  0.09825631,  2.40928487, -0.07240399, -0.36292312, -0.05010238,
                 -2.38689867,  0.20743984, -0.13771798,  2.81218865, -2.20559315]])

In [47]: confusion_matrix(y_test_pred, y_test)

Out[47]: array([[9312, 1355],
                [1639, 9623]], dtype=int64)

In [48]: from sklearn import metrics
         from sklearn.metrics import accuracy_score # accuracy

In [49]: print(sklearn.metrics.accuracy_score(y_test_pred, y_test))
         #Accuracy = sklearn.metrics.r2_score(ytest,ypred)
         #print(Accuracy)

         0.8634684664143372
```

3.4.2 Confusion matrix for logistic regression :-



```
In [75]: fig, ax = plt.subplots(figsize=(8, 8))
         plot_confusion_matrix(model,scaled_X_test,y_test,ax=ax)
         plt.title('Confusion matrix')
         plt.show()
         plt.savefig('confusion_matrix.jpg')
```



3.5 Random Forest for Classification:

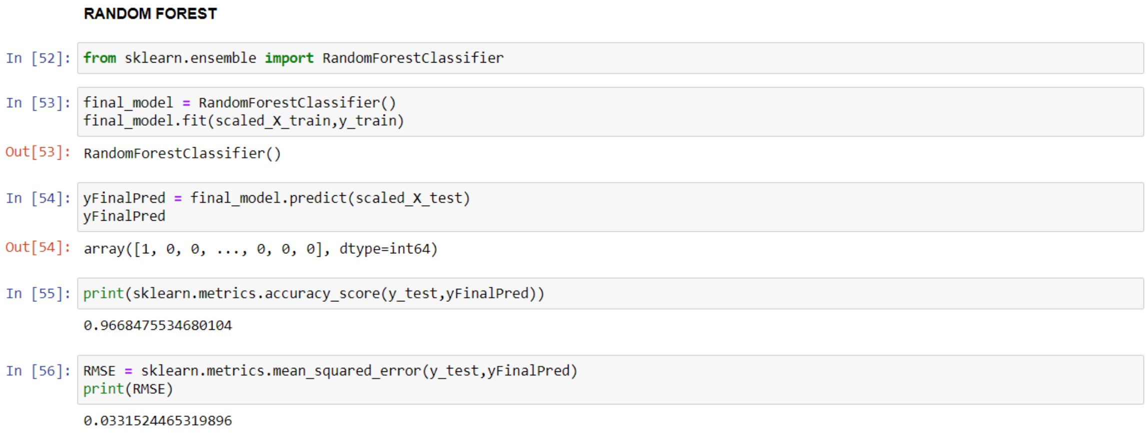
Random forest can also be used for the purpose of classification. It is one of the most broadly used machine learning algorithm for classification. Either the response variable is continuous or categorical, it works in both cases. According to Friedman et al. random forests starts to become stable at around 200 trees, whereas at 1000 trees the boosting of this still keeps on improving. If trees are much smaller or there is a presence of shrinkage then process of boosting starts to reduce. The important function of the RF is the utilization of out of bag (OOB) samples. For each value zi = (xi , yi), in which term zi not appears that makes the RF predictor by averaging only those trees which are consistent to the bootstrap samples. The OOB error estimate is then nearly identical of that which is getting by N fold cross validation. In contrast to the other non linear estimators, it is possible to fit the RF in one sequence with the cross validation being completed. The training can be finished if the OOB error stabilizes itself.

3.5.1 Algorithm of RF:

The algorithm of RF is considered as best in term of accuracy. Even the data is too large or includes thousands of input variables, though the efficiency does not decrease and at the same time prevent to be overfitting as well and there is no need of data pruning in it. It can be used for methods i.e. selection of best subset as well as imputation of the missing values and in both cases it performs very fine and efficient. The forest which is produced as an output is also proficient for adding the data for future.

In RF Ihave a learning set which is L = {(X1, Y1), ...,(Xn, Yn)} which should contain the observations of independent random vector (X,Y), where X is a vector of explanatory variables i.e. X = (X1 , ..., XP) and X< p and Y is the class label if the in the case of classification.

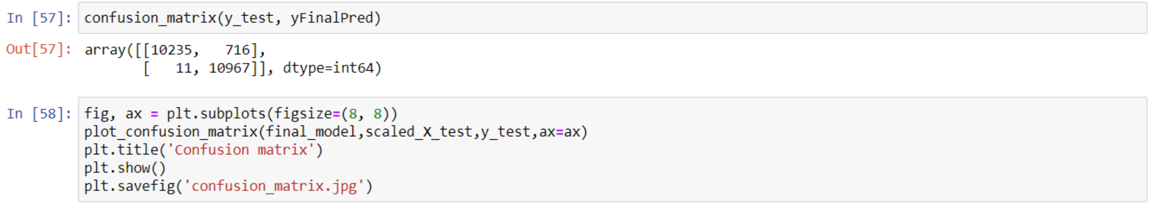
3.5.2 Using random forest on client data set:



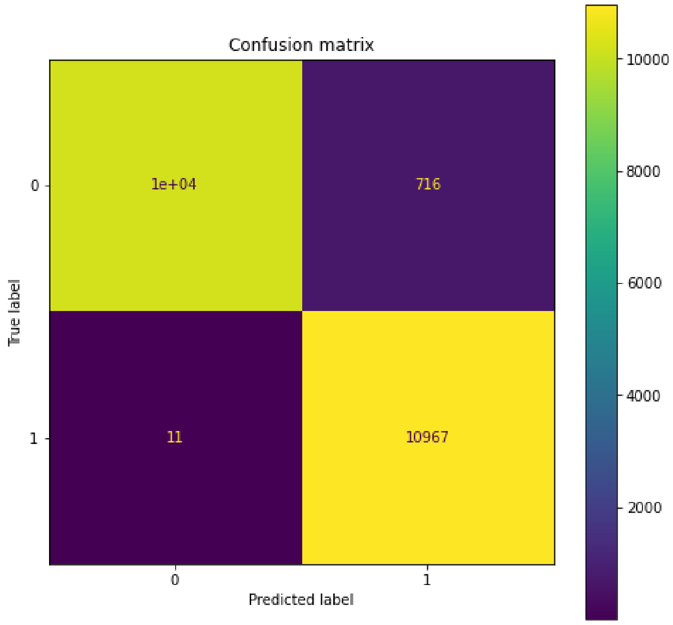
RANDOM FOREST

```
In [52]: from sklearn.ensemble import RandomForestClassifier

In [53]: final_model = RandomForestClassifier()
         final_model.fit(scaled_X_train,y_train)

Out[53]: RandomForestClassifier()

In [54]: yFinalPred = final_model.predict(scaled_X_test)
         yFinalPred

Out[54]: array([1, 0, 0, ..., 0, 0, 0], dtype=int64)

In [55]: print(sklearn.metrics.accuracy_score(y_test,yFinalPred))

         0.9668475534680104

In [56]: RMSE = sklearn.metrics.mean_squared_error(y_test,yFinalPred)
         print(RMSE)

         0.0331524465319896
```

3.5.3 Confusion matrix for random forest:-



```
In [57]: confusion_matrix(y_test, yFinalPred)

Out[57]: array([[10235,   716],
                [   11, 10967]], dtype=int64)

In [58]: fig, ax = plt.subplots(figsize=(8, 8))
         plot_confusion_matrix(final_model,scaled_X_test,y_test,ax=ax)
         plt.title('Confusion matrix')
         plt.show()
         plt.savefig('confusion_matrix.jpg')
```



CHAPTER 4
RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

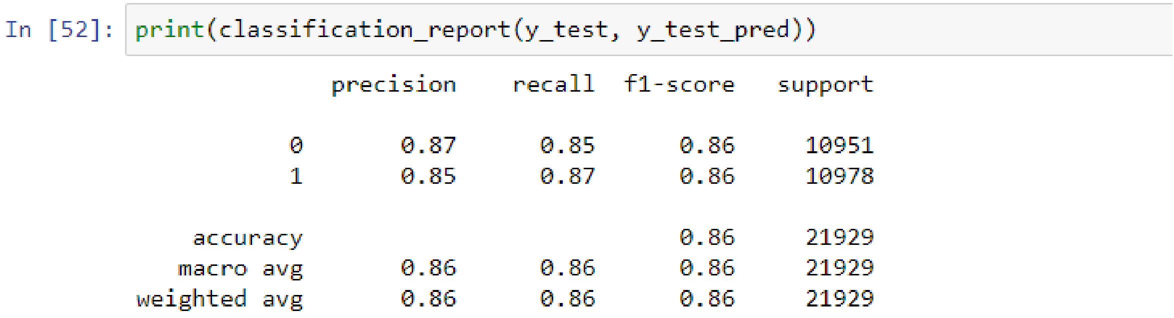
This section consists of outputs and tables of features selection and different classification approaches which have been discussed in the chapter 3.

4.1 Evaluation Metrics:

- **Accuracy:** The number of correctly predicted data points. This can be a misleading metric for an imbalanced dataset. Therefore, it is advisable to consider other evaluation metrics.
- **Precision:** It is calculated as the ratio of correctly predicted positive examples divided by the total number of positive examples that were predicted.
- **Recall:** It refers to the percentage of total relevant results correctly classified by your algorithm.
- **F1 score:** This is the weighted average of Precision and Recall.

4.2 Analysis and Interpretation of Results:

In the below fig No:4.2.1 shown different metrics values for Logistic Regression classification

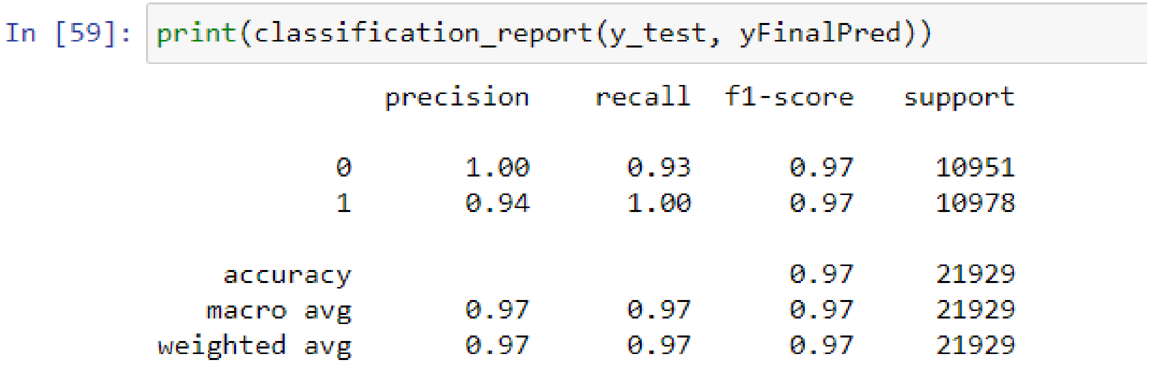


```
In [52]: print(classification_report(y_test, y_test_pred))

              precision    recall  f1-score   support

           0       0.87      0.85      0.86     10951
           1       0.85      0.87      0.86     10978

    accuracy                           0.86     21929
   macro avg       0.86      0.86      0.86     21929
weighted avg       0.86      0.86      0.86     21929
```

FIG 4.2.1 CLASSIFICATION REPORT OUTPUT FOR LOGISTIC REGRESSION

from above Fig no.4.2.1 Logistic has f1 score of 86 percent

In the below fig No:4.2.2 shown different metrics values for Random forest classification.



```
In [59]: print(classification_report(y_test, yFinalPred))

              precision    recall  f1-score   support

           0       1.00      0.93      0.97     10951
           1       0.94      1.00      0.97     10978

    accuracy                           0.97     21929
   macro avg       0.97      0.97      0.97     21929
weighted avg       0.97      0.97      0.97     21929
```

FIG 4.2.2 CLASSIFICATION REPORT OUTPUT FOR RANDOM FOREST

Therefore, from above Fig no.4.2.2 Random Forest has highest accuracy and f1 score. As random forest has given better results compared to the logistic regression, Random forest is considered as the best machine learning model for this business problem.

CHAPTER 5
SUMMARY AND CONCLUSIONS

5.1 SUMMARY:

This research demonstrates the different data mining methods which is a great tool in the decision making. For this study work, sample dataset **BANK ADDITIONAL FULL** was taken from the direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to assess if the product (bank term deposit) would be ('yes') or not ('no') subscribed. The dataset consists of several predictor variables and one target variable, Outcome. Predictor variables includes the age, job, marital status, and so on . Direct science website which is open source. In general there are 2 steps involved in this is work. In first step the process of feature selection has been done by using 3 statistical approaches i.e. **Logistic Regression, Random Forest classification etc**. performing data preprocessing tasks, split the data into train and test, build a model upon training data (70%) and evaluate the model on the test data (30%). Use the **random_state=10** while splitting the data. In the second step, the best subset of variables which are selected by these methods are go through for the classification. For classification purpose there are 4 computational algorithms for classification have been used that are Support vector Machines, Decision Trees, and Random Forest for classification, Artificial neural network. The aim was to check whether these classification methods give same number of accuracy and performance by using the feature selection approach. The results indicated that Ido not need to go for the full model as reduced subset of all variables can provide almost the same accuracy. Regarding feature subset, the best subset of variables is chosen by the random forest and regarding classification, also random forest comes up with the most **accurate results with 96.5023 % accuracy** on the subset selected by random forest. As accuracy of reduced model is almost the same so one can rely on the subset of variables selected by random forest instead of full set of variables. RF reveals that the most impacting attribute is contact, afterwards there are duration and age respectively.

5.2 CONCLUSION:

The main objective of this project is to build a model that predicts customers that would subscribe to a bank term deposit, and Iwere able to achieve that by considering two different models and using the best one for the prediction. I also went through rigorous steps of preparing I data for the model and choosing various evaluation metrics to measure the performance of I models. In the result achieved, observed that Random Forest is the best model with high f1 score and accuracy compared to the other models and with accuracy score of 96.534%.

REFERENCES:

[1] S. Moro, P. Cortez and P. Rita, "A Data-Driven Approach to Predict the Success of Bank Telemarketing", *Decision Support Systems Elsevier*, vol. 62, pp. 22-31, June 2014.

[2] A. Keles and A. Keles, "IBMMS Decision Support Tool for Management of Bank Telemarketing Campaigns", *International Journal of Database Management Systems*, vol. 7, no. 5, pp. 1-15, 2015.

[3] O.Apampa, "Evaluation of Classification and Ensemble Algorithms for Bank Customer Marketing Response Prediction", *Journal of International Technology and Information Management*, vol. 25, no. 4, pp. 6, 2016.

[4] Analytics Vidhya. (2016). https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-basedmodeling-scratch-in-python/

[5] Kodali, T. (2016). Prediction wine quality using Random Forests. R-bloggers. Retrieved from: https://www.rbloggers.com/predicting-wine-quality-using-random-forests/

[6] Moro, Cortez, & Rita. (2014). A data-driven approach to predict the success of bank telemarketing. Decision Support Systems.

[7] Pandya, R. (2015). C5.0 Algorithm to Improved Decision Tree with Feature Selection and Reduced Error Pruning. International Journal of Computer Applications, 117(16).

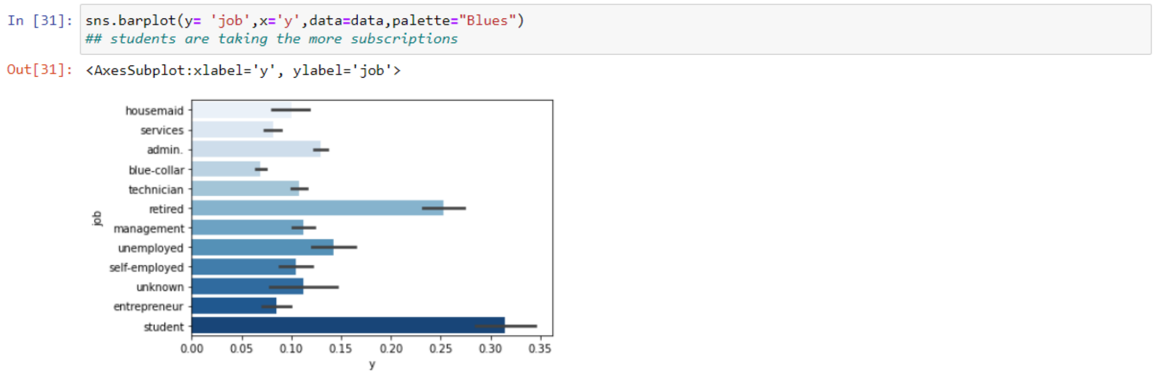
[8] *Amazon Machine Learning:* www.aws.amazon.com

APPENDIX:

A. SCREENSHOTS:

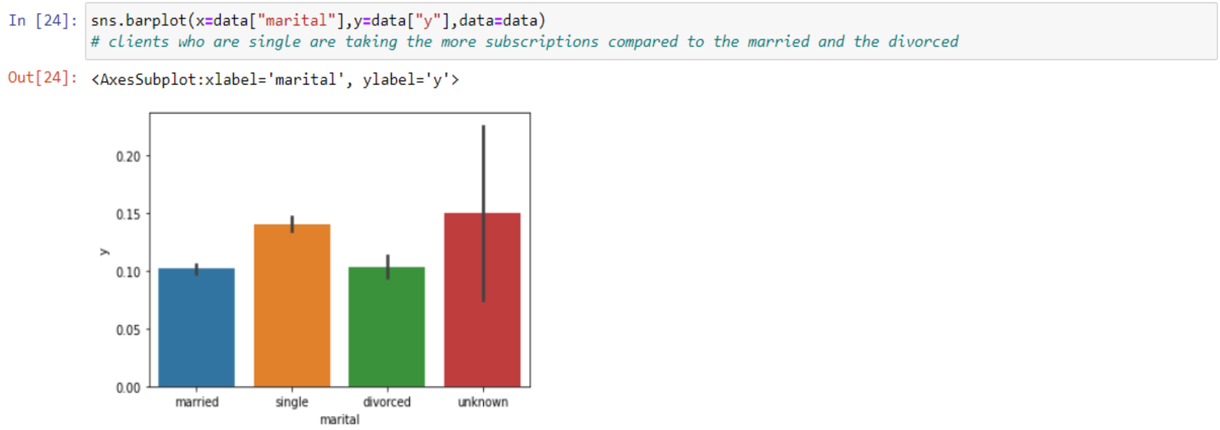
Data visualization:

Analysis of job, and target variable using bar plot



```
In [31]: sns.barplot(y= 'job',x='y',data=data,palette="Blues")
         ## students are taking the more subscriptions

Out[31]: <AxesSubplot:xlabel='y', ylabel='job'>
```

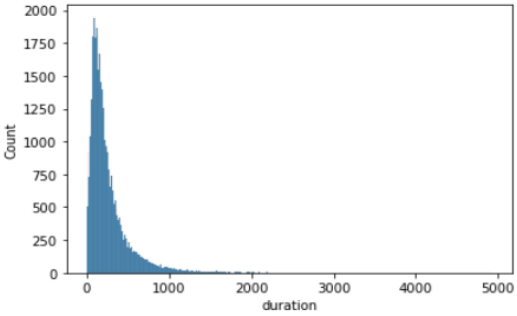
Analysis of marital, and target variable using bar plot



```
In [24]: sns.barplot(x=data["marital"],y=data["y"],data=data)
         # clients who are single are taking the more subscriptions compared to the married and the divorced

Out[24]: <AxesSubplot:xlabel='marital', ylabel='y'>
```

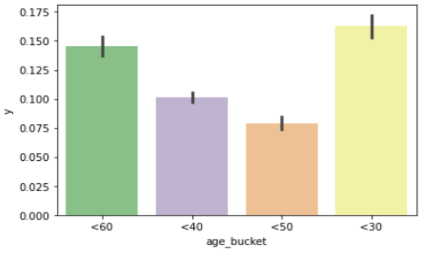
Analysis of duration of call, and target variable using hist plot

```
In [25]: sns.histplot(data['duration'])
         ## duration of phone call between clinet and agent is mostly less than 1000

Out[25]: <AxesSubplot:xlabel='duration', ylabel='Count'>
```



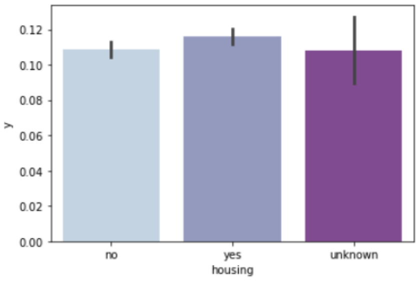
Analysis of age, and target variable using bar plot

```
In [26]: temp = ["<30" if i<30 else "<40" if i<40 else "<50" if i<50 else "<60" for i in data["age"]]
         k = pd.DataFrame(temp)
         k[0].value_counts()
         data["age_bucket"] = k[0]

In [27]: sns.barplot(x="age_bucket",y="y",data=data,palette = "Accent")
         # the clients with age less than 30 are taking more subscriptions

Out[27]: <AxesSubplot:xlabel='age_bucket', ylabel='y'>
```

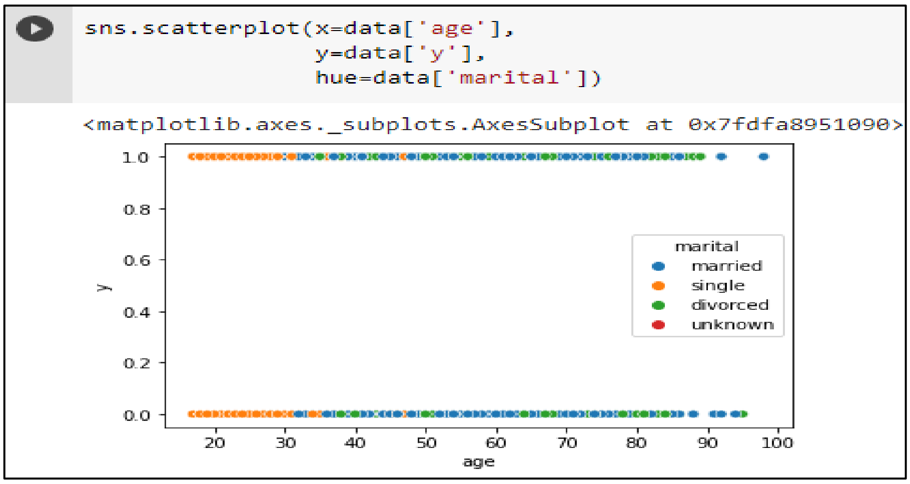


Analysis of loan, and target variable using bar plot

```
In [30]: sns.barplot(x="housing",y="y",data=data,palette = "BuPu")
         ## the clients who already have housing loan taking more subscriptions

Out[30]: <AxesSubplot:xlabel='housing', ylabel='y'>
```



2. Analysis of age, marital and independent variable using scatterplot



```
sns.scatterplot(x=data['age'],
                y=data['y'],
                hue=data['marital'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdfa8951090>
```

SOURCE CODE:

```
import pandas as pd
import seaborn as sns
import sklearn
import matplotlib.pyplot as plt
from sklearn.metrics import
classification_report,accuracy_score,confusion_matrix,plot_confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
import warnings
warnings.filterwarnings('ignore')


# ### DATA EXTRACTION

data = pd.read_csv('bank-additional-full.csv')


data.head()


# ### DATA PREPROCESSING

data.isnull().sum()
# no missing values observed

data.describe()
data.shape

data['y'] = data['y'].map({'yes':1,"no":0}) #encoding y

data.dtypes


data.shape

data_categoryCols = data.columns[data.dtypes=='object']

data_categoryCols


data_numericCols = data.columns[data.dtypes != 'object']




data_numericCols



data_cat = data[data_categoryCols]

data_num = data[data_numericCols]

data_cat.head(2)

data_num.head(2)

data_cat.columns

data_cat['job'] = data_cat['job'].astype('category')
data_cat['marital'] = data_cat['marital'].astype('category')
data_cat['education'] = data_cat['education'].astype('category')
data_cat['default'] = data_cat['default'].astype('category')
data_cat['housing'] = data_cat['housing'].astype('category')
data_cat['loan'] = data_cat['loan'].astype('category')
data_cat['contact'] = data_cat['contact'].astype('category')
data_cat['month'] = data_cat['month'].astype('category')
data_cat['day_of_week'] = data_cat['day_of_week'].astype('category')
data_cat['poutcome'] = data_cat['poutcome'].astype('category')

data_cat['job'].value_counts()

data_cat.dtypes


# convert category to num(0,1,2,3...) as per need
```

```
data_cat['job']= data_cat['job'].cat.codes
data_cat['marital']= data_cat['marital'].cat.codes
data_cat['education']= data_cat['education'].cat.codes
data_cat['default']= data_cat['default'].cat.codes
data_cat['housing']= data_cat['housing'].cat.codes
data_cat['loan']= data_cat['loan'].cat.codes
data_cat['contact']= data_cat['contact'].cat.codes
data_cat['month']= data_cat['month'].cat.codes
data_cat['day_of_week']= data_cat['day_of_week'].cat.codes
data_cat['poutcome']= data_cat['poutcome'].cat.codes

data_cat.head()


# merge

data_final = pd.concat([data_cat,data_num],join='outer',axis =1)
data_final.head()



# ### DATA VISUALIZATION

sns.barplot(x=data["marital"],y=data["y"],data=data)
# clients who are single are taking the more subscriptions compared to the married
and the divorced

sns.histplot(data['duration'])
## duration of phone call between clinet and agent is mostly less than 1000

temp = ["<30" if i<30 else "<40" if i<40 else "<50" if i<50 else "<60" for i in data["age"]]
k = pd.DataFrame(temp)
k[0].value_counts()
data["age_bucket"] = k[0]

sns.barplot(x="age_bucket",y="y",data=data,palette = "Accent")
# the clients with age less than 30 are taking more subscriptions

del data["age_bucket"]


sns.barplot(x="loan",y="y",data=data,palette = "BuPu")
## the clients who already have personal loan taking less subscriptions##


# clients who have housing loan

sns.barplot(x="housing",y="y",data=data,palette = "BuPu")
## the clients who already have housing loan taking more subscriptions
```

```
# current working job of client

sns.barplot(y= 'job',x='y',data=data,palette="Blues")
## students are taking the more subscriptions


# ### CLASS IMBALANCE PROBLEM


# the data set has more number of observations with the target variable as "0",
causing a class imbalance problem.
data_final["y"].value_counts()


# Class count
count_class_0, count_class_1 = data_final.y.value_counts()



# Divide by class
df_class_0 = data_final[data_final['y'] == 0]
df_class_1 = data_final[data_final['y'] == 1]

# over sampling the minority class, i.e class 1, so that the counts of class 0 and class
1 are same.
df_class_1_over = df_class_1.sample(count_class_0, replace=True)
df_test_over = pd.concat([df_class_0, df_class_1_over], axis=0)


df_test_over["y"].value_counts()


# ### TRAIN TEST SPLIT


x = df_test_over.iloc[:,:-1]
y = df_test_over.iloc[:,-1]

x.head()


import sklearn
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(x,y,train_size = 0.70,test_size =
0.30,random_state=10)


# ### FEATURE SCALING
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaled_X_train = scaler.fit_transform(X_train)
scaled_X_test = scaler.transform(X_test)


# ### MODEL DEVELOPMENT

from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier







# #### LOGISTIC REGRESSION

model = LogisticRegression()
#model = LinearRegression()
#model = KNeighborsClassifier()
#model = RandomForestClassifier()
model.fit(scaled_X_train,y_train)

y_test_pred = model.predict(scaled_X_test)
y_test_pred

model.coef_


confusion_matrix(y_test_pred, y_test)

from sklearn import metrics
from sklearn.metrics import accuracy_score # accuracy

print(sklearn.metrics.accuracy_score(y_test_pred, y_test))
#Accuracy = sklearn.metrics.r2_score(ytest,ypred)
#print(Accuracy)

fig, ax = plt.subplots(figsize=(8, 8))
plot_confusion_matrix(model,scaled_X_test,y_test,ax=ax)
plt.title('Confusion matrix')
plt.show()
plt.savefig('confusion_matrix.jpg')
```

```
print(classification_report(y_test, y_test_pred))


# #### RANDOM FOREST

from sklearn.ensemble import RandomForestClassifier

final_model = RandomForestClassifier()
final_model.fit(scaled_X_train,y_train)

yFinalPred = final_model.predict(scaled_X_test)
yFinalPred

print(sklearn.metrics.accuracy_score(y_test,yFinalPred))

RMSE = sklearn.metrics.mean_squared_error(y_test,yFinalPred)
print(RMSE)




confusion_matrix(y_test, yFinalPred)




fig, ax = plt.subplots(figsize=(8, 8))
plot_confusion_matrix(final_model,scaled_X_test,y_test,ax=ax)
plt.title('Confusion matrix')
plt.show()
plt.savefig('confusion_matrix.jpg')


print(classification_report(y_test, yFinalPred))


# **LogisticRegression**
# *   Accuracy - 0.8634684664143372

# **RandomForest**
# *   Accuracy - 0.9668475534680104
```