

LOAN PREDICTION USING MACHINE LEARNING

A mini-project report submitted in partial fulfillment of the Academic requirements for the award of the Degree of

BACHELOR OF ENGINEERING IN INFORMATION TECHNOLOGY

By

T. VARUN KUMAR REDDY

(2451-20-737-123)

P. SAI NITHIN

(2451-20-737-124)

K. SAKETH

(2451-20-737-130)

Under the guidance of

M. Prathyusha

Assistant Professor, Dept of IT



DEPARTMENT OF INFORMATION TECHNOLOGY
MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE
(An Autonomous Institution)
(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)
Nadergul, Saroornagar Mandal, Hyderabad-501510

2022-2023

MATURI VENKATA SUBBA RAO (MVSR)
ENGINEERING COLLEGE
(An Autonomous Institution)
(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)
Nadargul, Saroornagar Mandal, Hyderabad-501510



DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the mini project work entitled “**LOAN PREDICTION USING MACHINE LEARNING**” is a bonafide work carried out by **T. Varun Reddy (2451-20-737-123)**, **P. Sai Nithin (2451-20-737-124)**, **K. Saketh (2451-20-737-130)** in partial fulfillment of the requirements for the award of degree of **Bachelor of Engineering in Information Technology** from **Maturi Venkata Subba Rao (M.V.S.R.) Engineering College**, affiliated to **OSMANIA UNIVERSITY, Hyderabad**.

Mini Project In-Charge

Signature of Guide

Signature of Head, ITD

Signature of External Examiner

DECLARATION

This is to certify that the work reported in the present mini-project entitled “**Loan Prediction using Machine Learning**” is a record of bonafide work done by us in the Department of Information Technology, M.V.S.R. Engineering College, Osmania University. This report is based on the project work done entirely by us and not copied from any other source.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

Roll Number

Student Name

Signature

2451-20-737-123

T. VARUN KUMAR REDDY

2451-20-737-124

P. SAI NITHIN

2451-20-737-130

K. SAKETH

ACKNOWLEDGEMENT

We with extreme jubilation and deepest gratitude, would like to thank our guide,

M. Prathyusha, Assistant Professor, Department of Information Technology, Maturi Venkata Subba Rao (MVSR) Engineering College, for her constant encouragement to us to complete our work in time.

With immense pleasure, we record our deep sense of gratitude to our beloved Head of the department **Dr. K. Venu Gopal Rao**, Dean-Academics & HOD, Department of Information Technology, Maturi Venkata Subba Rao Engineering College, for permitting and providing facilities to carry out this project.

We would like to extend our gratitude to **K. Devaki** and **D. Muninder**, Mini-Project-I coordinators, Department of Information Technology, Maturi Venkata Subba Rao Engineering College, for their valuable suggestions and timely help during the course of the project.

Finally, we express, from the bottom of our heart and deepest gratitude to the entire faculty, our parents and family for the support, dedication, comprehension and love.

T. VARUN KUMAR REDDY (2451-20-737-123)

P. SAI NITHIN (2451-20-737-124)

K. SAKETH (2451-20-737-130)

MVSR Engineering College
Department of Information Technology

COURSE NAME: MINI PROJECT I

COURSE CODE: PW 654 IT

VISION

To impart technical education to produce competent and socially responsible engineers in the field of Information Technology.

MISSION

- M1. To make the teaching-learning process effective and stimulating.
- M2. To provide adequate fundamental knowledge of sciences and Information Technology with positive attitude.
- M3. To create an environment that enhances skills and technologies required for industry.
- M4. To encourage creativity and innovation for solving real world problems.
- M5. To cultivate professional ethics in students and inculcate a sense of responsibility towards society

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

The Bachelor's program in Information Technology is aimed at preparing graduates who will:

- I. Apply knowledge of mathematics and Information Technology to analyze, design and implement solutions for real world problems in core or in multidisciplinary areas.
- II. Communicate effectively, work in a team, practice professional ethics and apply knowledge of computing technologies for societal development.
- III. Engage in Professional development or postgraduate education to be a life-long learner.

PROGRAM OUTCOMES (POs)

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Problem analysis: Identity, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.
3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Environment and sustainability: Understand the impact of the professional engineering Solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOS):

- (1) Hardware design: An ability to analyze, design, simulate and implement computer hardware/software and use basic analog/digital circuits, VLSI design for various computing and communication system applications.
- (2) Software design: An ability to analyze a problem, design algorithm, identify and define the computing requirements appropriate to its solution and implement the same.

COURSE OBJECTIVES:

1. To enhance practical & Professional skills.
2. To familiarize the tools and techniques of symmetric literature survey and documentation.
3. To expose students to industry practices and teamwork.
4. To encourage students to work with innovative and entrepreneurial ideas.

COURSE OUTCOMES:

On successful completion of this course students will be able to:

1. Define a problem of the recent advancements with applications towards society.
2. Outline requirements and perform requirement analysis for solving the problem.
3. Design and develop a software and/or hardware-based solution within the scope of project using contemporary technologies and tools.
4. Test and deploy the applications for use.
5. Develop the Project as a team and demonstrate the application, with effective written and oral communications.

ABSTRACT

Loan approval prediction is a critical task in the financial industry, as lenders need to assess the risk of granting loans to applicants. Machine learning techniques can be used to predict the likelihood of loan approval based on various features, such as credit history, income, employment status, and loan amount. In this task, the dataset is split into training and testing sets, and a machine learning algorithm is trained on the training set to learn the relationship between the features and loan approval status. The trained model is then evaluated on the testing set to determine its performance in predicting loan approval status. Various machine learning algorithms such as logistic regression, decision trees, random forests, and support vector machines can be used for loan approval prediction. The accuracy of the prediction model can be further improved by feature engineering, hyperparameter tuning, and ensemble learning techniques.

Key Terms: Loan Approval, lenders, Logistic Regression, Ensemble techniques

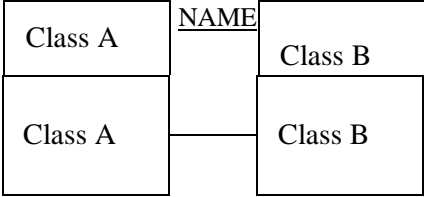
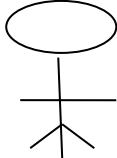
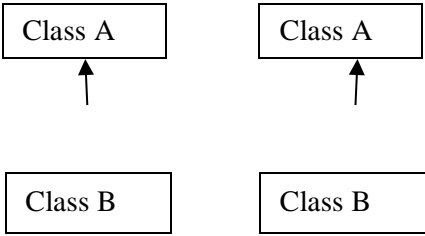
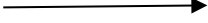
LIST OF FIGURES

FIGURE NO	NAME OF THE FIGURE	PAGE NO.
4.1	System Architecture	7
4.1.1	Architecture Description	8
4.2.1	Use Case Diagram	9
4.2.2	Class Diagram	10
4.2.3	Sequence Diagram	11
5.1	Data Collection	12
5.2	Data Exploration	14
5.3	Data	14
5.3.1	Reading Data	15
5.3.2	Model Building Part I	16
5.3.3	Logistic Regression	16
5.3.4	Random Forest	17
5.3.5	Decision Tree	18
5.3.6	Implementation of Model	19
5.3.7	Process of Loan Prediction	21
6.1	Anaconda Installation	22
6.1.1	Launching Jupyter notebook	23
6.1.2	Running Flask code in Anaconda Prompt	23
7.1	Machine Learning Models Accuracy	27
7.2	Algorithm Performance Comparison in Bar Graph	27
7.3	Web Interface	28
7.4	User Entering their values	28
7.5	Message	29
8.2.1	Person with Loan Approved	31
8.2.2	Person without Loan Approved	31

LIST OF TABLES

TABLE NO	NAME OF THE TABLE	PAGE NO.
2.1	LITERATURE SURVEY	3

LIST OF SYMBOLS

S.NO	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class	<div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 20px;"> + <i>public</i> - <i>private</i> </div> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>Class Name</i> - <i>attribute</i> - <i>attribute</i> </div> </div>	Represents a collection of similar entities grouped together.
2.	Association		Association represents static relationships between classes. Role represents the way the two classes see each other.
3.	Actor		It aggregates several classes into a single class.
4.	Aggregation		Interaction between the system and external environment
5.	Relation (extends)	<div style="text-align: center;"> extends  </div>	Extends relationship is used when one use case is similar to another use case but does a bit more.



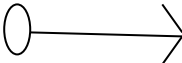
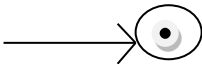
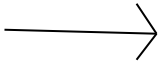
6.	Communication		Communication between various use cases.
7.	State		State of the processes.
8.	Initial State		Initial state of the object
9.	Final state		Final state of the object
10.	Control flow		Represents various control flow between the states

TABLE OF CONTENTS

<u>CONTENTS</u>	<u>PAGE NO.</u>
TITLE	i
CERTIFICATE	ii
DECLARATION	iii
ACKNOWLEDGEMENT	iv
VISION	v
MISSION	v
PROGRAM OUTCOMES	vi
COURSEOBJECTIVES	vii
COURSE OUTCOMES	vii
ABSTRACT	vii
LIST OF FIGURES	ix
LIST OF TABLES	x
LIST OF SYMBOLS	xi
TABLE OF CONTENTS	xiii
CHAPTER 1: INTRODUCTION	1
1.1 PROBLEM STATEMENT, 1.2 OBJECTIVES 1.3 MOTIVATION	
1.4 EXISTING SYSTEM 1.5 PROPOSED SYSTEM 1.6 SCOPE	
CHAPTER 2: LITERATURE SURVEY	4
CHAPTER 3: SYSTEM REQUIREMENT SPECIFICATION	5
3.1 SOFTWARE REQUIREMENTS	
3 HARDWARE REQUIREMENTS	
CHAPTER 4: SYSTEM DESIGN	6
4.1 SYSTEM ARCHITECTURE	6
4.1.1 ARCHITECTURE DESCRIPTION	6
4.1.2 FLOW OF THE PROJECT	7
4.2 UML DIAGRAMS	8
4.2.1 USE CASE DIAGRAM	9
4.2.2 CLASS DIAGRAM	10
4.2.3 SEQUENCE DIAGRAM	11

CHAPTER 5: METHODOLOGY	13
5.1 DATA COLLECTION	13
5.2 DATA EXPLORATION	13
5.3 DATA PREPROCESSING	19
5.4 MACHINE LEARNING MODELS	21
CHAPTER 6: IMPLEMENTATION	22
6.1 ENVIRONMENTAL SETUP	22
6.2 ALGORITHM	24
6.3 MODULE DESCRIPTION	25
CHAPTER 7: RESULTS	27
CHAPTER 8: TESTS	30
8.1 TESTING	30
CHAPTER 9: CONCLUSION AND FUTURE ENHANCEMENTS	32
REFERENCES	33
APPENDIX	34

CHAPTER 1

INTRODUCTION

A loan is the core business part of banks. The main portion the bank's profit is directly come from the profit earned from the loans. Though bank approves loan after a regress process of verification and testimonial but still there's no surety whether the chosen hopeful is the right hopeful or not. This process takes fresh time while doing it manually. We can prophesy whether that particular hopeful is safe or not and the whole process of testimonial is automated by machine literacy style. Loan Prognostic is really helpful for retainer of banks as well as for the hopeful also

1.1 General

The immense increase in capitalism, the fast-paced development and instantaneous changes in the lifestyle has us in awe. Emi, loans at nominal rate, housing loans, vehicle loans, these are some of the few words which have skyrocketed from the past few years. The needs, wants and demands have never been increased this before. People gets loan from banks; however, it may be baffling for the bankers to judge who can pay back the loan nevertheless the bank shouldn't be in loss. Banks earn most of their profits through the loan sanctioning. Generally, banks pass loan after completing the numerous verification processes despite all these, it is still not confirmed that the borrower will pay back the loan or not. To get over the dilemma, I have built up a prediction model which says if the loan has been assigned in the safe hands or not. Government agencies like keep under surveillance why one person got a loan and the other person could not. In Machine Learning techniques which include classification and prediction can be applied to conquer this to a brilliant extent. Machine learning has eased today's world by developing these prediction models. Here we will be using the fine techniques of machine learning – Decision tree algorithm to build this prediction model for loan assessment. It is as so because decision tree gives accuracy in the prediction and is often used in the industry for these models.

1.2 PROBLEM STATEMENT

We are building a model to predict the approval of loan by using various machine learning algorithms such as logistic regression , decision trees, random forests.

1.3 OBJECTIVES

The objective of this project:

- To detect the accuracy of the prediction

- To allow only authorized person and ensure security.
- To send an alert sms and email to admin.

1.4 EXISTING SYSTEM

This model is trained on historical loan data to learn patterns and make predictions about whether a new loan applicant is likely to default or not. The system may also use various algorithms such as logistic regression, decision trees, or random forests to make predictions. The output of the model is a probability score that indicates the likelihood of loan approval. Based on this score, the lender can decide whether to approve the loan or not. However, the accuracy and reliability of the system depend on the quality of the data used to train the model, the selection of appropriate features, and the choice of the algorithm used for predictions.

DRAWBACKS OF EXISTING SYSTEM

- Limited data
- Biased training data
- Lack of interpretability
- Overfitting
- Lack of transparency
- Costly implementation

1.5 PROPOSED SYSTEM

The proposed system uses a machine learning approach to predict loan approval. The system will utilize historical data on loan applications and their outcomes to train a model that can accurately predict whether a new loan application is likely to be approved or not. The model will take into account various factors such as credit score, income, employment history, and other relevant information to make its prediction. The system will be designed to be user-friendly, with a simple interface that allows users to input their information and receive a quick decision on their loan application. It is hoped that the system will reduce the time and resources required for manual loan processing and improve the accuracy of loan decisions, the proposed method's final step is to build a user interface using Flask.

1.6 SCOPE

- The scope of this project is to predict the loan for an loan applicant with the maximum amount of accuracy in our prediction.
- Our present study mainly focused on the use of data for loan prediction and explore different ways of representing such data through our analysis.
- The ultimate goal is to develop a user interface where a user can check if they will get approved or not.

CHAPTER 2 : LITERATURE SURVEY

S.NO	NAME	YEAR	AUTHOR NAME	ALGORITHM /TECHNIQUE USED	ADVANTAGES	LIMITATIONS
1	Feature selection for loan approval prediction using machine learning algorithms	2022	A.S. Alghamdi and W.S. Alfares	Logistic Regression, Decision Tree, Random Forest, KNN, SVM, and ANN	Demonstrated that feature selection techniques can improve the accuracy and reduce the computational cost of loan approval prediction models .	Used only one dataset for evaluation
2	Loan approval prediction using deep learning models	2022	Nasmin Jiwani, Ketan Gupta, Neda Afreen	Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Hybrid models combining CNN and LSTM	Demonstrated that deep learning models can achieve high accuracy for loan approval prediction	Used only one dataset for evaluation, deep learning models require more computational resources than traditional machine learning models
3	Loan approval prediction using machine learning and fuzzy logic	2022	Rakshith DB, Mrigank Srivastava, Ashwani Kumar & Gururaj S P	Logistic Regression, Decision Tree, Random Forest, KNN, SVM, ANN, and XGBoost	Identified XGBoost as the most accurate model for loan approval prediction .	Used multiple datasets for evaluation, including a large real-world dataset
4	Loan approval prediction using machine learning and fuzzy logic	2021	Jayakumar Sadhasivam, J. Senthil, R.M. Ganesh, N. Chellapan	Decision Tree, Random Forest, KNN, SVM, and Fuzzy Logic .	Demonstrated that incorporating fuzzy logic with machine learning algorithms improved the accuracy of loan approval prediction compared to individual models	Used only one dataset for evaluation.
5	Loan approval prediction using machine learning algorithms	2021	H.V.Shetty and S.S. Shetty	Logistic Regression, Decision Tree, Random Forest, KNN, SVM, and ANN	Identified Random Forest as the most accurate model for loan approval prediction .	Used only one dataset for evaluation

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- ⌘ Processor : I3/Intel Processor
- ⌘ Hard Disk : 160GB
- ⌘ RAM : 8Gb

3.2 SOFTWARE REQUIREMENTS

- × Operating System : Windows 7/8/10/11
- × Libraries Used : NumPy, Pandas, Scikit learn, Matplot, Pickle
- × Technology : Python 3.6+ ,HTML
- × Interface : Flask

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

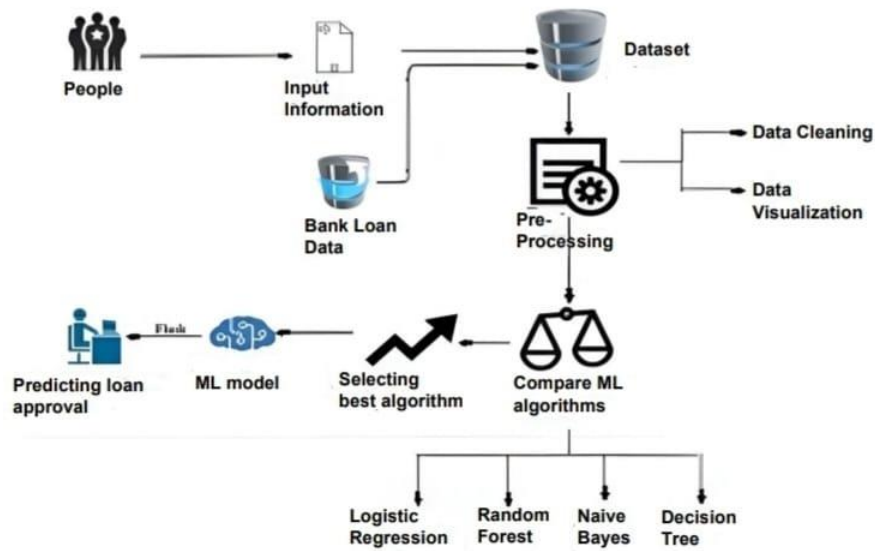


Fig-4.1 SYSTEM ARCHITECTURE

4.1.1 Architecture Description

- In the first step we collect the dataset and perform data preprocessing, cleaning and Exploratory Data Analysis.
- Then we go towards model creation of Machine Learning Algorithms, Logistic Regression, Random Forest, Support Vector Machine, K Nearest Neighbor, Gradient and Ada Boosting Classifier, Decision Tree, Naive Bayes.
- We select the model which has Highest Accuracy and import it into our flask code in our system.
- We then setup our environment of web page in system.
- User can check result by entering his/her data values and know whether he/she loan has approved or not.

4.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: A Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.2.1 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

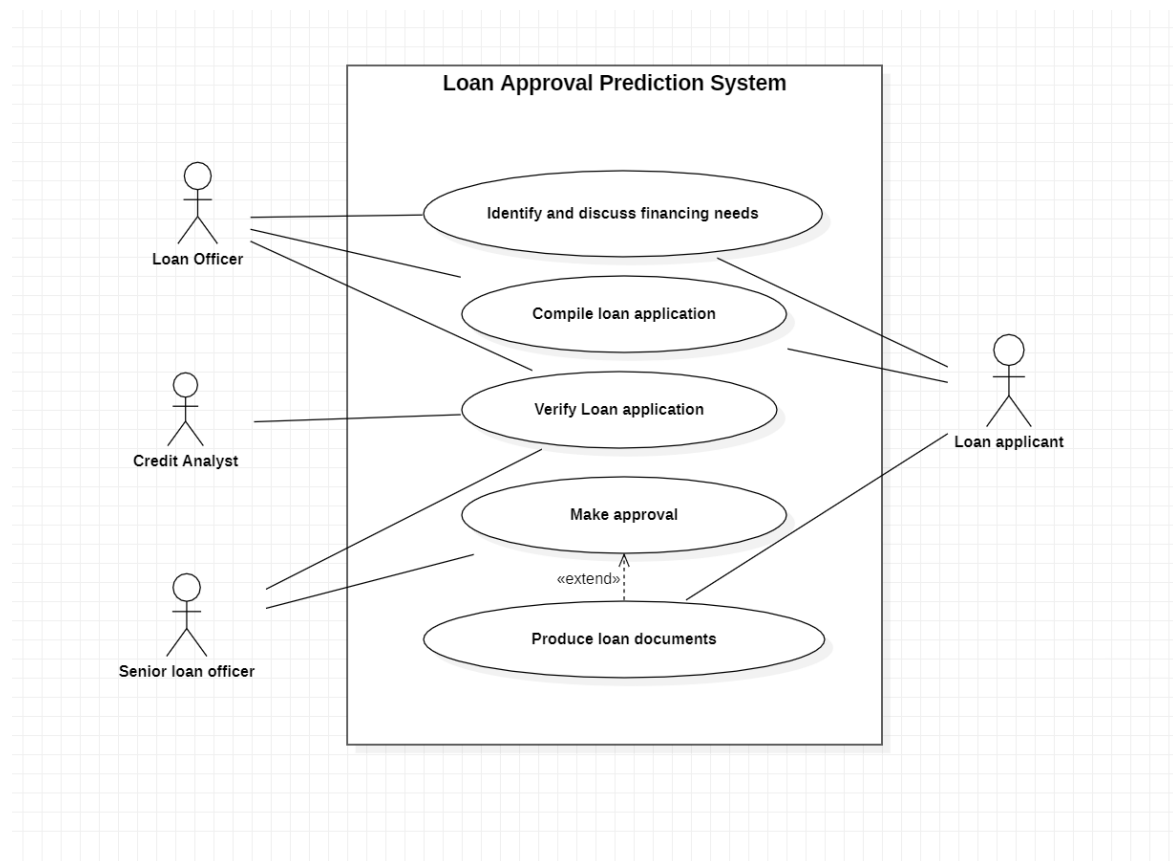


Fig-4.2.1 USE CASE DIAGRAM

4.2.2 Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains which information.

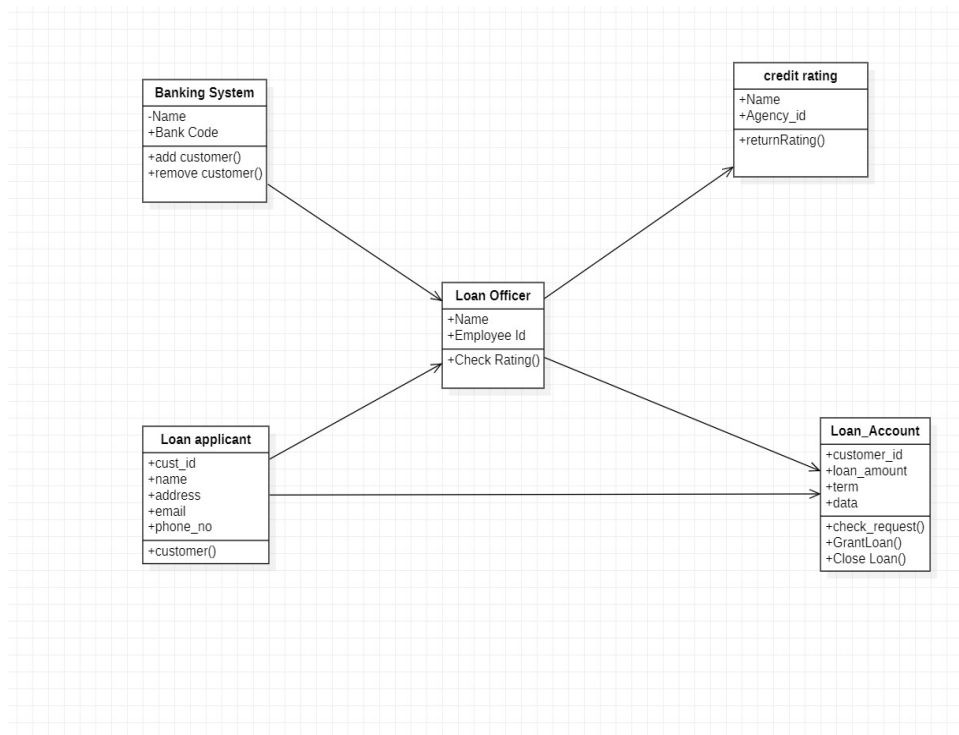


Fig-4.2.2 CLASS DIAGRAM

4.2.3 Sequence Diagram

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

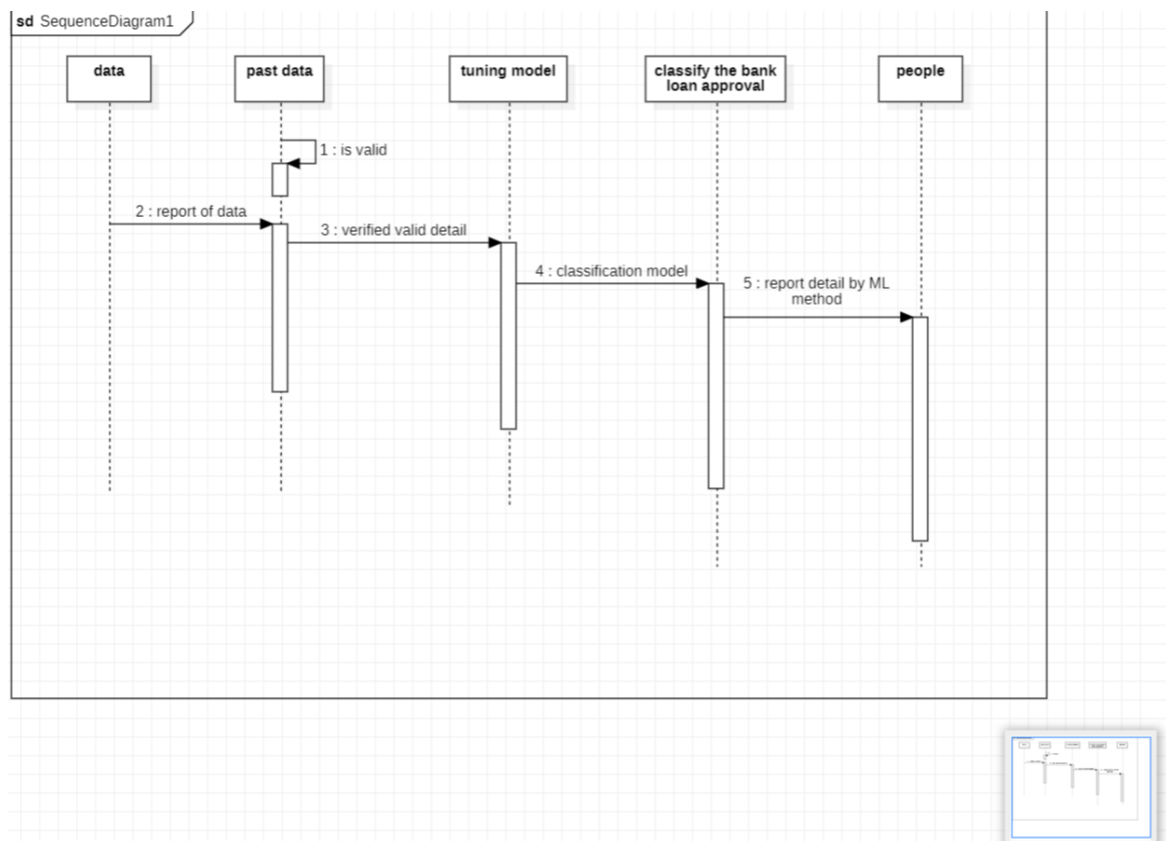


Fig-4.2.3 SEQUENCE DIAGRAM

CHAPTER 5

METHODOLOGY

5.1 DATA COLLECTION

In this experiment, we collect a dataset from the UCI Machine Learning Repository. In addition, the original dataset was collected from the northeast of Andhra Pradesh, India. This dataset consists of 583 liver patient's data whereas 75.64% male patients and 24.36% are female patients. This dataset has contained 11 particular parameters whereas we choose 10 parameters for our further analysis and 1 parameter as a target class. Such as,

- I.** Loan_id: Unique Loan Id
- II.** Gender: Male/ Female
- III.** Married: Applicant married (Y/N)
- IV.** Dependents: Number of dependents
- V.** Education: Applicant Education (Graduate/ Under Graduate)
- VI.** Self_Employed: Self employed (Y/N)
- VII.** Applicant Income: Applicant income
- VIII.** Coapplicant Income: Coapplicant income
- IX.** Loan Amount: Loan amount in thousands
- X.** Loan_Amount_Term: Term of loan in months
- XI.** Credit_History: credit history meets guidelines
- XII.** Property_Area: Urban/ Semi Urban/ Rural
- XIII.** Loan_Status: (Target) Loan approved (Y/N)

5.2 DATA EXPLORATION

Data exploration refers to the initial step in data analysis. Data analysts use data visualization and statistical techniques to describe dataset characterizations, such as size, quantity, and accuracy, to understand the nature of the data better.

5.3 DATA

For this problem, we have three CSV files: train, test, and sample submission.

- Train file will be used for training the model, i.e. our model will learn from this file. It contains all the independent variables and the target variable.

- Test file contains all the independent variables, but not the target variable. We will apply the model to predict the target variable for the test data.
- Sample submission file contains the format in which we have to submit our predictions

5.3.1 Reading data

```
train = pd.read_csv('Dataset/train.csv')
train.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0

```
test = pd.read_csv('Dataset/test.csv')
test.head()
```

5.3.2 Model Building Part I

Let us make our first model predict the target variable. We will start with Logistic Regression which is used for predicting binary outcome.

- Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables.
- Logistic regression is an estimation of Logit function. The logit function is simply a log of odds in favor of the event.
- This function creates an S-shaped curve with the probability estimate, which is very similar to the required stepwise function
- Let's drop the Loan_ID variable as it does not have any effect on the loan status. We will do the same changes to the test dataset which we did for the training dataset.

```
train=train.drop('Loan_ID',axis=1)
test=test.drop('Loan_ID',axis=1)
```

We will use scikit-learn (sklearn) for making different models which is an open source library for Python. It is one of the most efficient tools which contains many inbuilt functions that can be used for modeling in Python. Sklearn requires the target variable in a separate dataset. So, we will drop our target variable from the training dataset and save it in another dataset.

```
X = train.drop('Loan_Status',1)
```

```
y = train.Loan_Status
```

Now we will make dummy variables for the categorical variables. The dummy variable turns categorical variables into a series of 0 and 1, making them a lot easier to quantify and compare. Let us understand the process of dummies first:

- Consider the “Gender” variable. It has two classes, Male and Female.
- As logistic regression takes only the numerical values as input, we have to change male and female into a numerical value.
- Once we apply dummies to this variable, it will convert the “Gender” variable into two variables (Gender_Male and Gender_Female), one for each class, i.e. Male and Female.
- Gender_Male will have a value of 0 if the gender is Female and a value of 1 if the gender is Male.

```
X = pd.get_dummies(X)
```

```
train=pd.get_dummies(train)
```

```
test=pd.get_dummies(test)
```

Now we will train the model on the training dataset and make predictions for the test dataset. But can we validate these predictions? One way of doing this is we can divide our train dataset into two parts: train and validation. We can train the model on this training part and using that make predictions for the validation part. In this way, we can validate our predictions as we have the true predictions for the validation part (which we do not have for the test dataset).

We will use the train_test_split function from sklearn to divide our train dataset. So, first, let us import train_test_split.

```
from sklearn.model_selection import train_test_split
x_train, x_cv, y_train, y_cv = train_test_split(X,y, test_size=0.3)
```

The dataset has been divided into training and validation part. Let us import LogisticRegression and accuracy_score from sklearn and fit the logistic regression model.

```
from sklearn.linear_model import LogisticRegression from sklearn.metrics
import accuracy_score model = LogisticRegression()
model.fit(x_train, y_train)LogisticRegression()
```

Here the C parameter represents the inverse of regularization strength. Regularization is applying a penalty to increasing the magnitude of parameter values in order to reduce overfitting. Smaller values of C specify stronger regularization.

Let's predict the Loan_Status for validation set and calculate its accuracy.

```
pred_cv = model.predict(x_cv)
accuracy_score(y_cv,pred_cv)
0.7891891891891892
```

So our predictions are almost 80% accurate, i.e. we have identified 80% of the loan status correctly.

Let's make predictions for the test dataset.

```
pred_test = model.predict(test)
```

Let's import the submission file which we have to submit on the solution checker.

```
submission = pd.read_csv('Dataset/sample_submission.csv')
submission.head()
```

	Loan_ID	Loan_Status
0	LP001015	N
1	LP001022	N
2	LP001031	N
3	LP001035	N
4	LP001051	N

We only need the Loan_ID and the corresponding Loan_Status for the final submission. we will fill these columns with the Loan_ID of the test dataset and the predictions that we made, i.e., pred_test respectively.

```
submission['Loan_Status']=pred_test  
submission['Loan_ID']=test_original['Loan_ID']
```

Remember we need predictions in Y and N. So let's convert 1 and 0 to Y and N.

```
submission['Loan_Status'].replace(0, 'N', inplace=True)  
submission['Loan_Status'].replace(1, 'Y', inplace=True)
```

Finally, we will convert the submission to .csv format.

```
pd.DataFrame(submission,  
              columns=['Loan_ID','Loan_Status']).to_csv('Output/logistic.csv')
```

Logistic Regression using stratified k-folds cross-validation

To check how robust our model is to unseen data, we can use Validation. It is a technique that involves reserving a particular sample of a dataset on which you do not train the model. Later, you test your model on this sample before finalizing it. Some of the common methods for validation are listed below:

- The validation set approach
- k-fold cross-validation
- Leave one out cross-validation (LOOCV)
- Stratified k-fold cross-validation

In this section, we will learn about stratified k-fold cross-validation. Let us understand how it works:

- Stratification is the process of rearranging the data so as to ensure that each fold is a good representative of the whole.
- For example, in a binary classification problem where each class comprises of 50% of the data, it is best to arrange the data such that in every fold, each class comprises of about half the instances.
- It is generally a better approach when dealing with both bias and variance.
- A randomly selected fold might not adequately represent the minor class, particularly in cases where there is a huge class imbalance.

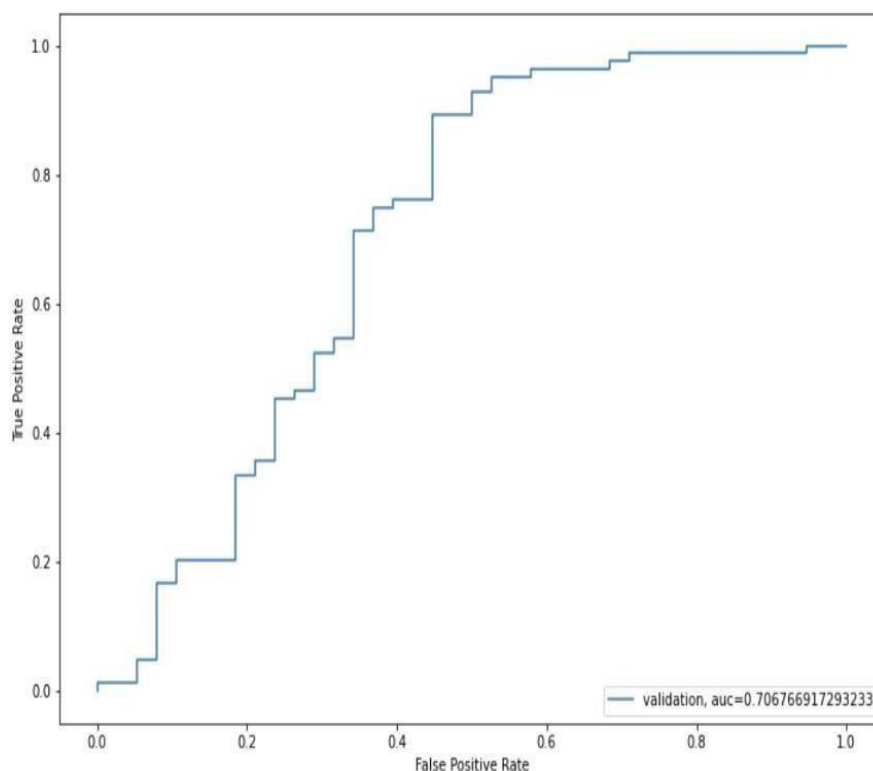
Let's import StratifiedKFold from sklearn and fit the model.

```
from sklearn.model_selection import StratifiedKFold
```

Now let's make a cross-validation logistic model with stratified 5 folds and make predictions for the test dataset

The mean validation accuracy for this model turns out to be 0.80. Let us visualize the roc curve.

```
from sklearn import metrics
fpr, tpr, _ = metrics.roc_curve(yvl, pred)
auc = metrics.roc_auc_score(yvl, pred)
plt.figure(figsize=(12,8))
plt.plot(fpr, tpr, label="validation, auc="+str(auc))
plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()
```



We got an auc value of 0.70

```
submission['Loan_Status']=pred_test
```

```
submission['Loan_ID']=test_original['Loan_ID']
```

Remember we need predictions in Y and N. So let's convert 1 and 0 to Y and N.

```
submission['Loan_Status'].replace(0, 'N', inplace=True)
submission['Loan_Status'].replace(1, 'Y', inplace=True)pd.DataFrame(submission,
columns=['Loan_ID','Loan_Status']).to_csv('Output/Log1.csv')
```

Feature Engineering

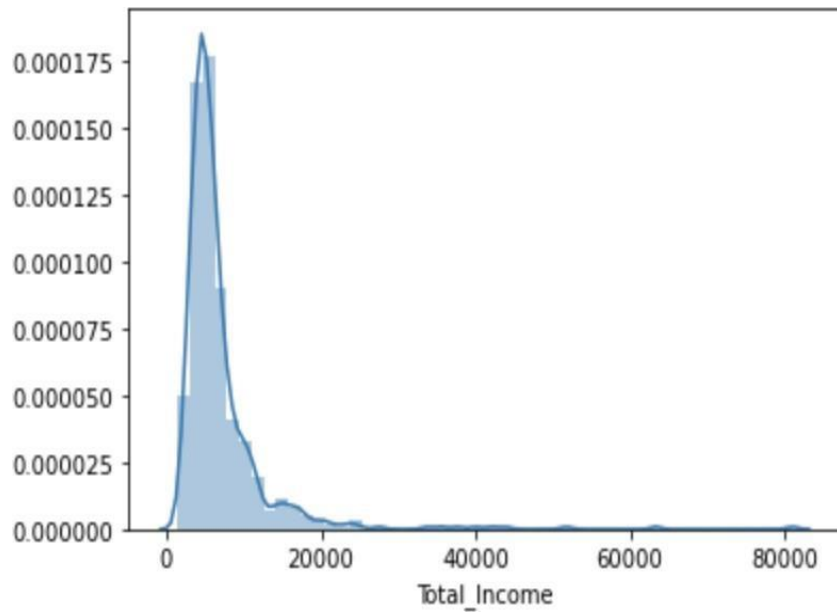
Based on the domain knowledge, we can come up with new features that might affect the target variable. We will create the following three new features:

- **Total Income** — As discussed during bivariate analysis we will combine the Applicant Income and Co-applicant Income. If the total income is high, the chances of loan approval might also be high.
- **EMI** — EMI is the monthly amount to be paid by the applicant to repay the loan. The idea behind making this variable is that people who have high EMI's might find it difficult to pay back the loan. We can calculate the EMI by taking the ratio of the loan amount with respect to the loan amount term.
- **Balance Income** — This is the income left after the EMI has been paid. The idea behind creating this variable is that if this value is high, the chances are high that a person will repay the loan and hence increasing the chances of loan approval.

```
train['Total_Income']=train['ApplicantIncome']+train['CoapplicantIncome']
test['Total_Income']=test['ApplicantIncome']+test['CoapplicantIncome']
```

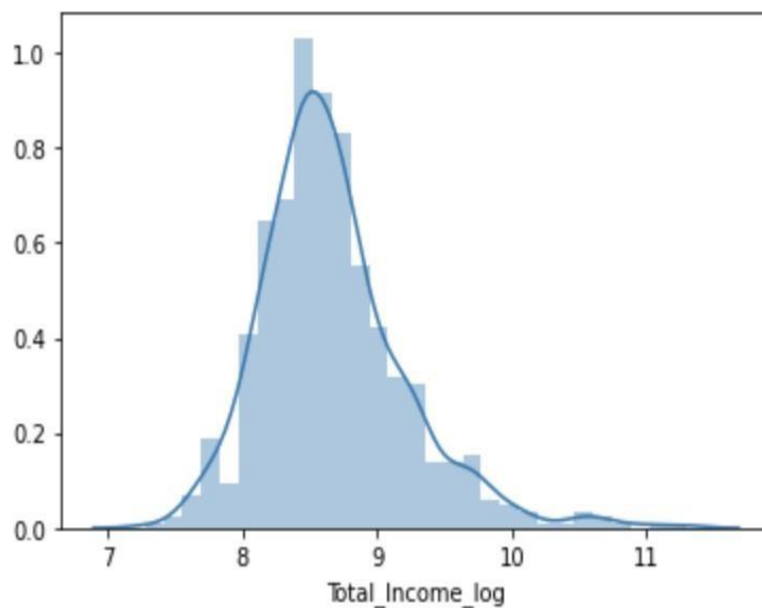
Let's check the distribution of Total Income.

```
sns.distplot(train['Total_Income'])
```



We can see it is shifted towards left, i.e., the distribution is right-skewed. So, let's take the log transformation to make the distribution normal.

```
train['Total_Income_log'] = np.log(train['Total_Income'])
sns.distplot(train['Total_Income_log'])
test['Total_Income_log'] = np.log(test['Total_Income'])
```

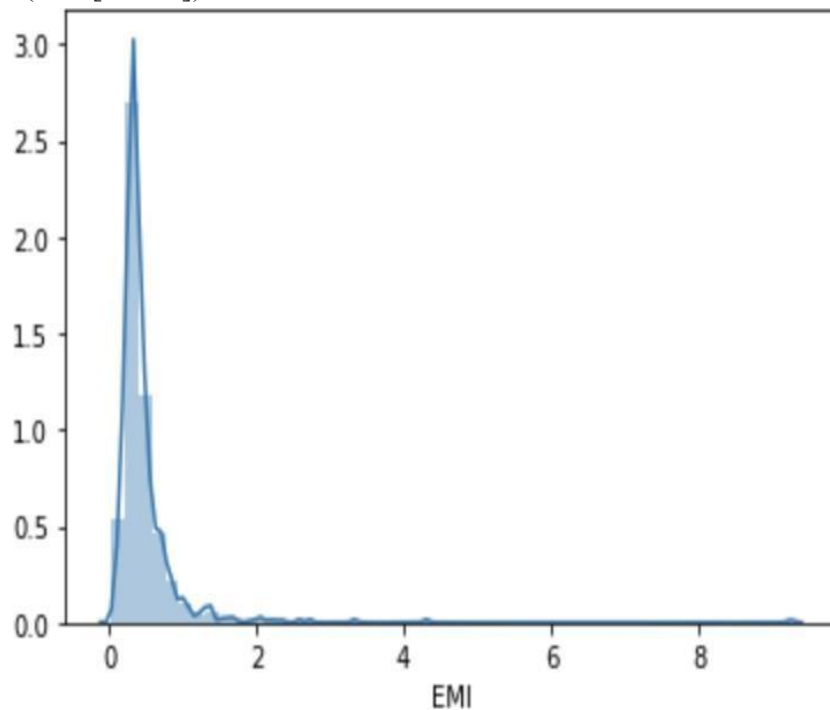


Now the distribution looks much closer to normal and the effect of extreme values has been significantly subsided. Let's create the EMI feature now

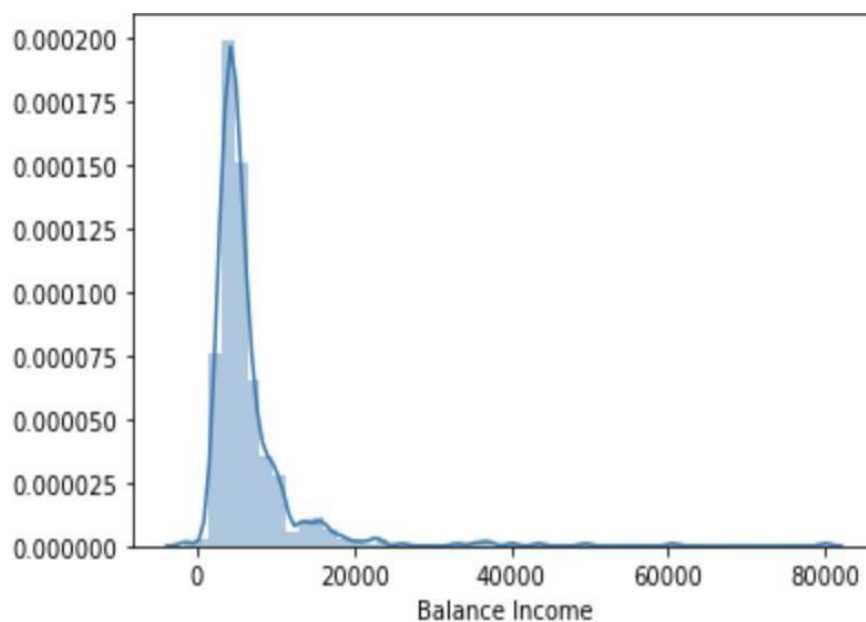
```
train['EMI']=train['LoanAmount']/train['Loan_Amount_Term']  
test['EMI']=test['LoanAmount']/test['Loan_Amount_Term']
```

Let's check the distribution of the EMI variable.

```
sns.distplot(train['EMI'])
```



```
train['Balance Income'] = train['Total_Income']-(train['EMI']*1000)  
test['Balance Income'] = test['Total_Income']-(test['EMI']*1000)  
sns.distplot(train['Balance Income'])
```



Let us now drop the variables which we used to create these new features. The reason for doing this is, the correlation between those old features and these new features will be very high, and logistic regression assumes that the variables are not highly correlated. We also want to remove the noise from the dataset, so removing correlated features will help in reducing the noise too.

```
train=train.drop(['ApplicantIncome', 'CoapplicantIncome',  
                 'LoanAmount', 'Loan_Amount_Term'], axis=1)  
test=test.drop(['ApplicantIncome', 'CoapplicantIncome',  
               'LoanAmount', 'Loan_Amount_Term'], axis=1)
```

After creating new features, we can continue the model building process. So we will start with the logistic regression model and then move over to more complex models like RandomForest and XGBoost. We will build the following models in this section.

- Logistic Regression
- Decision Tree
- Random Forest
- XGBoost

Let's prepare the data for feeding into the models.

```
X = train.drop('Loan_Status',1)  
y = train.Loan_Status
```

5.3.3 Logistic Regression

```
i=1 mean = 0 kf = StratifiedKFold(n_splits=5,random_state=1,shuffle=True)
for train_index,test_index in kf.split(X,y):
    print ('\n{} of kfold {}'.format(i,kf.n_splits))
    xtr,xvl = X.loc[train_index],X.loc[test_index]
    ytr,yvl = y[train_index],y[test_index]
    model = LogisticRegression(random_state=1)
    model.fit(xtr,ytr)
    pred_test=model.predict(xvl)
    score=accuracy_score(yvl,pred_test)
    mean += score print ('accuracy_score',score)
    i+=1
    pred_test = model.predict(test)
    pred = model.predict_proba(xvl)[:,-1]
    print ('\n Mean Validation Accuracy',mean/(i-1))

1of kfold 5 accuracy_score 0.7967479674796748

2of kfold 5 accuracy_score 0.6910569105691057
3of kfold 5 accuracy_score 0.6666666666666666
4of kfold 5 accuracy_score 0.7804878048780488
5of kfold 5 accuracy_score 0.680327868852459

Mean Validation
Accuracy 0.7230574436891909
submission['Loan_Status']=pred_test submission['Loan_ID']=test_original['Loan_ID']
submission['Loan_Status'].replace(0, 'N', inplace=True)
submission['Loan_Status'].replace(1, 'Y',
inplace=True)pd.DataFrame(submission,columns=['Loan_ID','Loan_Status']).to_
csv('Output/Log2.csv')
```

5.3.4 Decision Tree

Decision tree is a type of supervised learning algorithm(having a pre- defined target variable) that is mostly used in classification problems. In this technique, we split the population or sample into two or more homogeneous sets(or sub-populations) based on the most significant splitter/differentiator in input variables.

Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable.

Let's fit the decision tree model with 5 folds of cross-validation.

```
From sklearn import tree i=1 mean = 0
kf=StratifiedKFold(n_splits=5,random_state=1,shuffle=True) for train_index,test_index in
kf.split(X,y):
print ("\n{ } of kfold { } '.format(i,kf.n_splits))
xtr,xvl = X.loc[train_index],X.loc[test_index]
ytr,yvl = y[train_index],y[test_index]
model = tree.DecisionTreeClassifier(random_state=1)
model.fit(xtr,ytr)
pred_test=model.predict(xvl)
score=accuracy_score(yvl,pred_test) mean += score print ('accuracy_score',score) i+=1
pred_test = model.predict(test)
pred = model.predict_proba(xvl)[: ,1]
print ("\n Mean Validation Accuracy',mean/(i-1))

1 of kfold 5 accuracy_score 0.7398373983739838

2 of kfold 5 accuracy_score 0.6991869918699187
3 of kfold 5 accuracy_score 0.7560975609756098
of kfold 5
```

```

accuracy_score 0.7073170731707317

5 of kfold 5 accuracy_score
0.6721311475409836
Mean Validation Accuracy 0.7149140343862455
submission['Loan_Status']=pred_test submission['Loan_ID']=test_original['Loan_ID']
submission['Loan_Status'].replace(0, 'N', inplace=True)
submission['Loan_Status'].replace(1, 'Y', inplace=True)
pd.DataFrame(submission, columns=['Loan_ID','Loan_Status']).to_csv('Output/DecisionTree.csv')

```

5.3.5 Random Forest

- RandomForest is a tree-based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) are combined to make a powerful prediction model.
- For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model.
- Final prediction can be a function of all the predictions made by the individual learners.
- In the case of a regression problem, the final prediction can be the mean of all the predictions.

For a detailed explanation visit this article

<https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial- tree-based-modeling-scratch-in-python/>

```

from sklearn.ensemble import RandomForestClassifier i=1 mean = 0
kf = StratifiedKFold(n_splits=5,random_state=1,shuffle=True)
for train_index,test_index in kf.split(X,y): print ('{ } of kfold { }
'.format(i,kf.n_splits))
xtr,xvl = X.loc[train_index],X.loc[test_index]
ytr,yvl = y[train_index],y[test_index]
model = RandomForestClassifier(random_state=1, max_depth=10)
model.fit(xtr,ytr) pred_test=model.predict(xvl)

```

```

score=accuracy_score(yvl,pred_test) mean += score
print ('accuracy_score',score) i+=1
pred_test = model.predict(test)
pred = model.predict_proba(xvl)[:,1]
print ('\n Mean Validation Accuracy',mean/(i-1))
1 of kfold 5 accuracy_score 0.8292682926829268

2 of kfold 5 accuracy_score
0.8130081300813008

3 of kfold 5 accuracy_score
0.7723577235772358

4 of kfold 5 accuracy_score
0.8048780487804879

5 of kfold 5 accuracy_score
0.7540983606557377

Mean Validation Accuracy 0.7947221111555378

```

We will try to improve the accuracy by tuning the hyperparameters for this model. We will use a grid search to get the optimized values of hyper parameters. Grid-search is a way to select the best of a family of hyper parameters, parametrized by a grid of parameters.

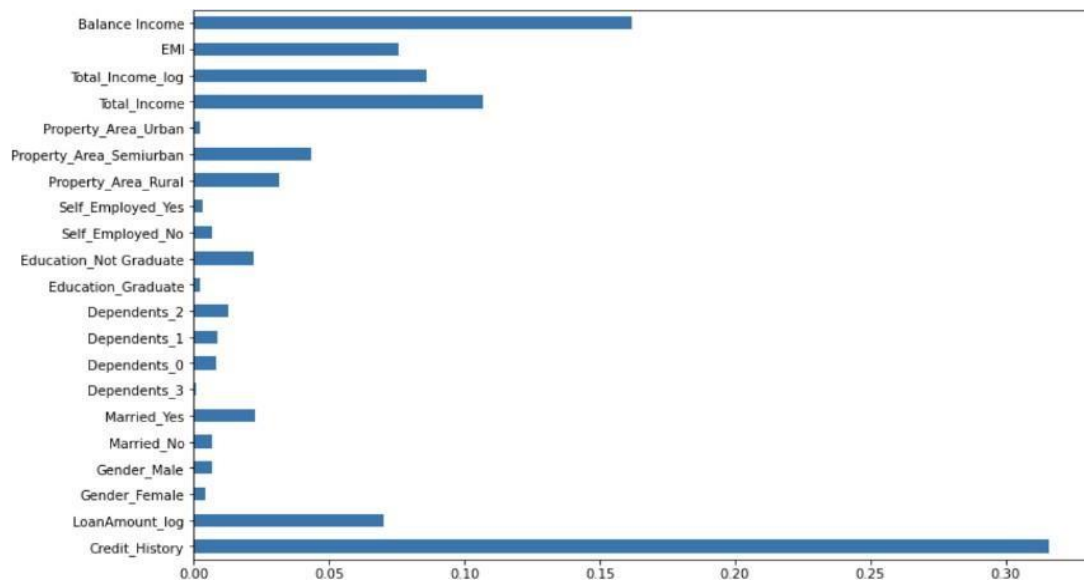
We will tune the `max_depth` and `n_estimators` parameters. `max_depth` decides the maximum depth of the tree and `n_estimators` decides the number of trees that will be used in the random forest model.

Let us find the feature importance now, i.e. which features are most important for this problem. We will use the `feature_importances_` attribute of sklearn to do so.

```

importances=pd.Series(model.feature_importances_, index=X.columns)
importances.plot(kind='barh', figsize=(12,8))

```



We can see that Credit_History is the most important feature followed by Balance Income, Total Income, EMI. So, feature engineering helped us in predicting our target variable.

5.3.6 IMPLEMENTATION OF MODEL

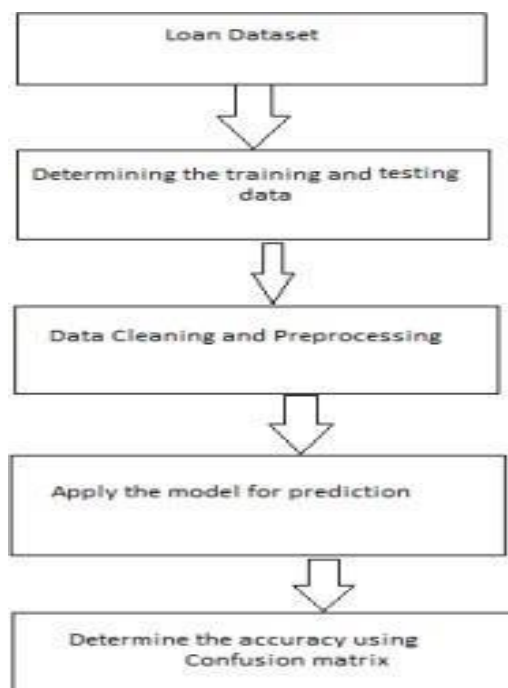


Fig 11: Diagram of Prediction Model

In above exploratory analysis of features, it basically shows the dependency or association of features of defaults and non- default attributes. In fig 2, graph shows the relationship of age between defaulters and non-defaulters and shows the age group of them. Similarly, other plots of features also show connection between them. Suppose, if someone is already in debt and income of customer is not that much, the probability of repaying his loan will be low. There are 850 observations and 9 features in the data set - All 9 features are numerical in nature - There are no missing values in the data set - Out of 850 customers data, 700 are existing customers and 150 are new customers - In the 700 existing customers, 517 customers are tagged as non-defaulters and remaining 183 are tagged as defaulters - The data is highly imbalanced - From VIF check, found out that the correlation between the variables is within the acceptable limits.

5.3.7 Processes for Loan Prediction:



Separating the target variable

```
X= balance_data. values[:, 1:5]
```

```
Y= balance_data. values[:, 0]
```

Splitting the dataset into test and train

```
X_train, X_test, Y_train, y_test= train_test_split (X, Y, test_size = 0.3,  
random_state= 100)
```

CHAPTER 6

IMPLEMENTATION

6.1 ENVIRONMENTAL SETUP

Installing Anaconda Navigator : Anaconda Navigator is a desktop graphical program that permits you to launch environment and supervises conda packages, and channels while not the necessity to use statement commands

1. To download and install Anaconda Navigator visit the official website of <https://www.anaconda.com/download> and choose your version.

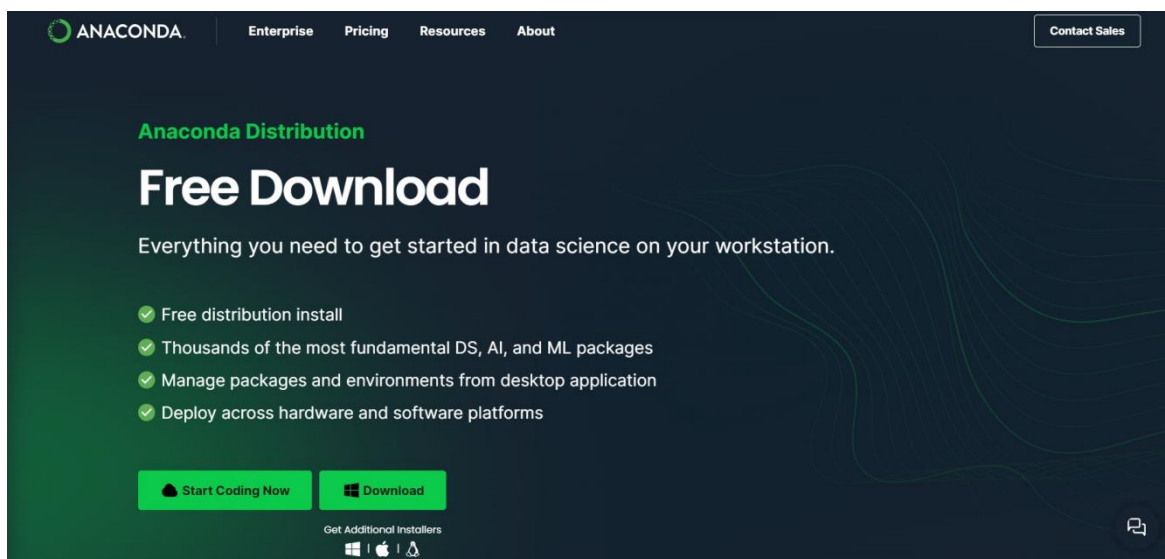


FIG-6.1 ANACONDA NAVIGATOR INSTALLATION

2. Once the download is complete, open and install.
3. You can see Anaconda installing at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".
5. On the next screen, you can create a desktop shortcut if you want and click on "Next"

Launching Jupyter notebook:

1. We need to launch jupyter notebook in Anaconda Navigator

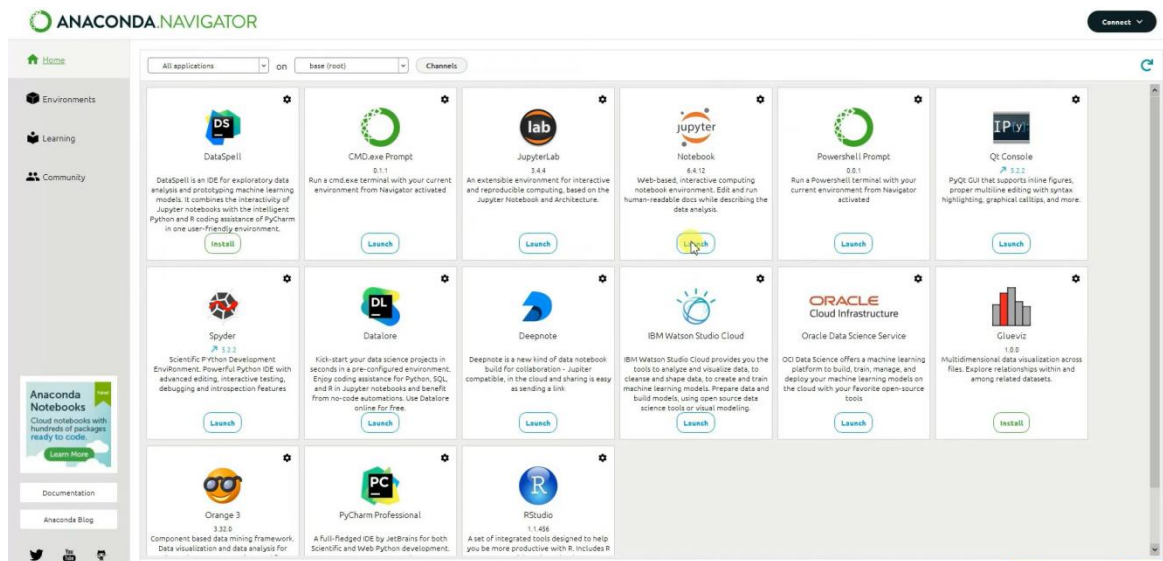
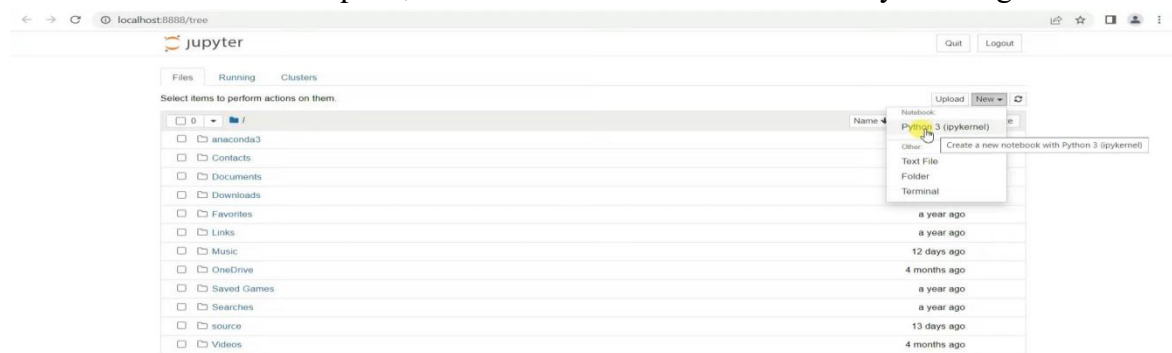


FIG-6.1.1 JUPYTER NOTEBOOK LAUNCH

2. Once the launch is complete, we can create a new notebook by clicking on 'New'



3. Under New by selecting Python 3 a new notebook is created.
4. We can start our programming in the new notebook
5. After coding and selecting the best model we import it in our flask code and keep them all in same directories
6. Now for the final result we need to open our Anaconda prompt and run flask code.

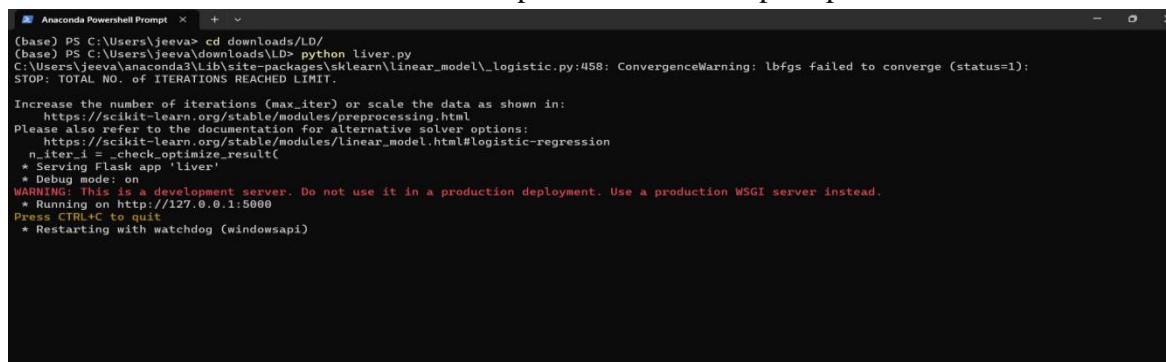


FIG-6.1.2 RUNNING FLASK CODE IN ANACONDA PROMPT

6.2 ALGORITHM

- **Logistic Regression (LR):** Logistic regression is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- **Random Forest Classifier:** Random Forest is a supervised learning based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. The variables are selected at random and models are trained on bootstrap samples. The final outcome will be calculated by aggregating the output from these trees.
- **Decision Tree Classifier :** It is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome
- **The Naïve Bayes Classifier :** It is a popular supervised machine learning algorithm used for classification tasks such as text classification. It belongs to the family of generative learning algorithms, which means that it models the distribution of inputs for a given class or category. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions

6.3 DESCRIPTION

- Pandas
- NumPy
- Scikit-learn
- Flask

Pandas

- Pandas provide us with many Series and Data Frames. It allows you to easily organize, explore, represent, and manipulate data.
- Smart alignment and indexing featured in Pandas offer you a perfect organization and data labeling.
- Pandas has some special features that allow you to handle missing data or value with a proper measure.

- It provides a collection of built-in tools that allows you to both read and write data in different web services, data-structure, and databases as well.
- Pandas can support JSON, Excel, CSV, HDF5, and many other formats. In fact, you can merge different databases at a time with Pandas.

Numpy

- Arrays of Numpy offer modern mathematical implementations on huge amount of data. Numpy makes the execution of these projects much easier and hassle-free.
- Numpy provides masked arrays along with general array objects. It also comes with functionalities such as manipulation of logical shapes, discrete Fourier transform, general linear algebra, and many more.
- This python package provides useful tools for integration. You can easily integrate Numpy with programming languages such as C, C++, and Fortran code.
- Numpy provides such functionalities that are comparable to MATLAB. They both allow users to get faster with operations.

Scikit-Learn

- The random module is a simple and efficient tool for predictive data analysis
- The functionality that scikit-learn provides include:
- Regression, including Linear and Logistic Regression
- Classification, including K-Nearest Neighbors
- Clustering, including K-Means and K-Means++
- Model selection
- Preprocessing, including Min-Max Normalization

Flask

- Flask is a lightweight web framework for building web applications in Python. It provides us with the tools and libraries to create web pages, handle HTTP requests and responses, and manage routing and sessions.
- Flask offers a simple and intuitive interface for creating web APIs and serving dynamic content. It allows us to define routes and associate them with functions that handle the requests and generate the responses.
- Flask provides a flexible templating system that allows us to create dynamic HTML pages by inserting data into predefined templates. This makes it easy to generate dynamic content and display data from a database or other sources.
- Flask is known for its simplicity and ease of use, making it a popular choice for beginners and experienced developers alike. Its lightweight nature and minimalistic design make it easy to understand and get started .

CHAPTER 7

RESULTS

The Confusion Matrix (CM) is used to analyze and determine the performance of the proposed loan prediction model.

- True Positive (TP), when both the actual and predicted values are positive (1)
- True Negative (TN), when both the actual and predicted values are negative (0)
- False Positive (FP), when the actual value is negative and the predicted value is positive (1)
- False Negative (FN), when the actual value is positive (1) and the predicted value is negative(0).

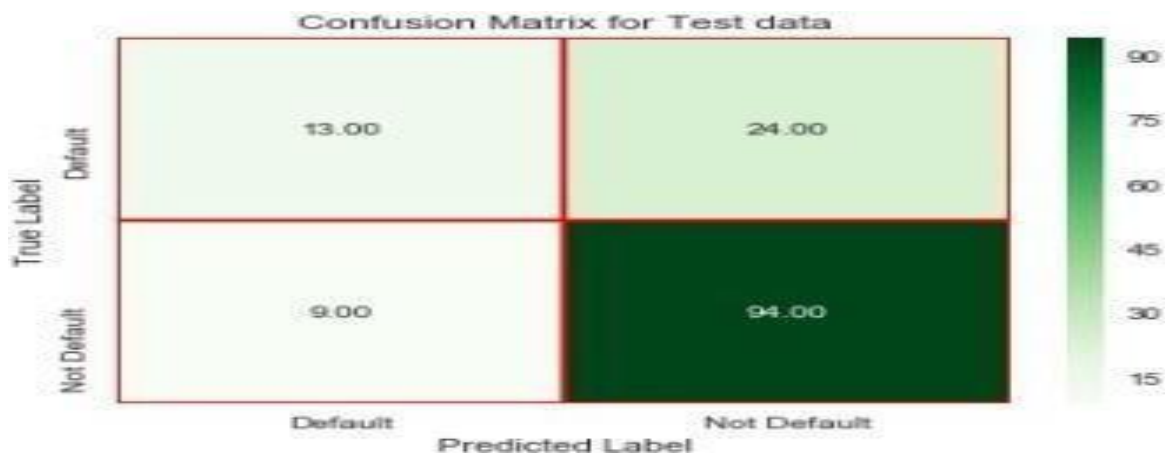


Fig 20: Generated Confusion Matrix for the Model

Confusion Matrix is given for the test data after the prediction of the model for loan. It shows the performance of the model developed for the prediction.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy for the model:

After the model is built, best score for the model is 0.7893. Then the model performance evaluations of test datasets are done and datasets passed through the model. Precision score of the test data is 0.591. Overall Accuracy of the model is 0.764

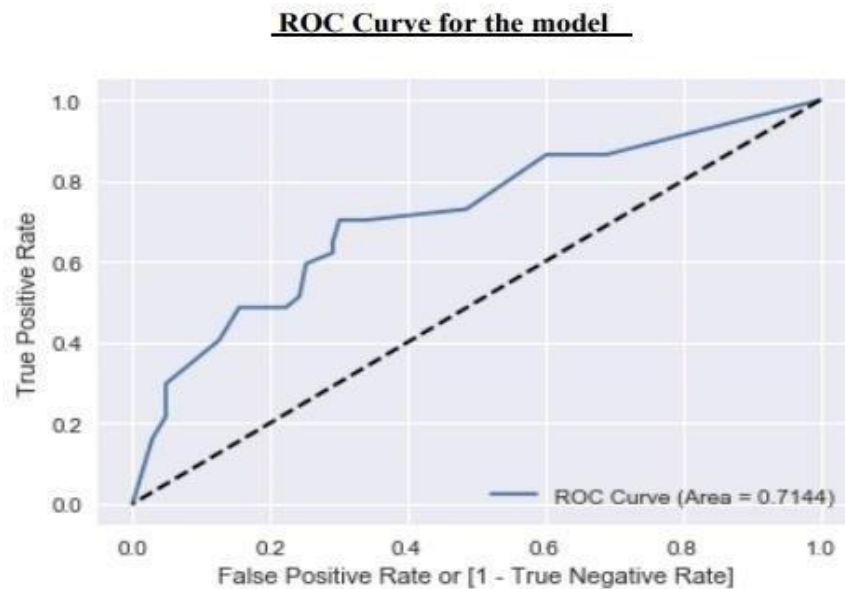


Fig 21: ROC Curve for the model

ROC Curves summarize the trade-off between the true positive rate and false positive rate for a predictive model using different probability thresholds

CHAPTER 8

TESTS

8.1 TESTING

8.1.1 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- All links should take the user to the correct page.

8.1.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

8.1.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant

participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

8.2 TEST CASES

- **PERSON WHO GOT APPROVED**

The image displays two screenshots of a web application. The top screenshot shows the 'LOAN PREDICTION' interface with a 'REGISTRATION' form. The form is set against a background of various Canadian coins. The form fields are as follows:

REGISTRATION	
Name: Samaira	Address: Hno. 11-1-146, Vidhyuth Na
Email: Sam77@gmail.com	Mobile Number: 7796177077
Applicant Income: 12000	Co-applicant Income: 5639
Loan Amount: 120	Loan Amount Term: 360
Credit History: Clear	Married: YES
Gender: MALE	Dependents: 0
Education: Graduate	Self Employed: YES
Property: Rural	
<input type="button" value="Submit"/>	

The bottom screenshot shows the result page with a white brick wall background. It displays the text 'WOW YOUR LOAN IS APPROVED' and a link labeled 'HOME PAGE'.

- PERSON WHO GOT REJECTED

Document x +

127.0.0.1:5000

LOAN PREDICTION

REGISITRATION

Name Sameer	Address Hno. 11-1-146, Vidhyuth Na
Email Sameer97@gmail.com	Mobile Number 7799177066
Applicant Income 3000	Co-applicant Income 0
Loan Amount 90	Loan Amount Term 360
Credit History : Clear	Married : YES
Gender : MALE	Dependents : 0
Education : Graduate	Self Employed : YES
Property : Urban	

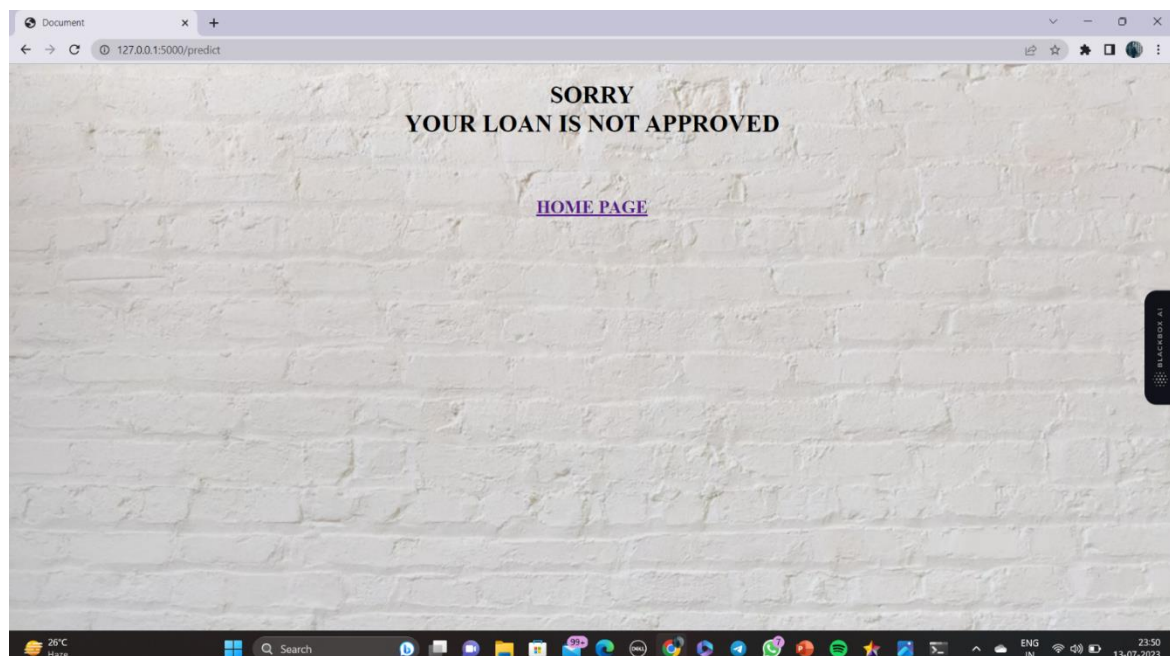
Submit

25°C Haze

Search

ENG IN

23:48 13-07-2023



CHAPTER 9

CONCLUSION and FUTURE ENHANCEMENTS

After this work, we are able to conclude that Decision tree version is extraordinary efficient and gives a higher end result. We have developed a model which can easily predict that the person will repay its loan or not. we can see our model has reduced the efforts of bankers. Machine learning has helped a lot in developing this model which gives precise results.

In future, this model can be used to compare various machine learning algorithm generated prediction models and the model which will give higher accuracy will be chosen as the prediction model.

REFERENCES

- 1 M. A. Sheikh, A. K. Goel and T. Kumar, "An Approach for Prediction of Loan Approval using Machine Learning Algorithm," 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2020, pp. 490-494, doi: 10.1109/ICESC48915.2020.9155614.
- 2 Gurlove Singh, Amit Kumar Goel ,”Face Detection and Recognition System using Digital Image Processing” , 2nd International conference on Innovative Mechanism for Industry Application ICMIA 2020, 5-7 March 2020, IEEE Publisher.
- 3 Raj, J. S., Ananthi, J. V., “Recurrent neural networks and nonlinear prediction in support vector machine” Journal of Soft Computing Paradigm (JSCP),1(01), 33-40, 2019.
- 4 Aakanksha Saha, Tamara Denning, Vivek Srikumar, Sneha Kumar Kasera. ”Secrets in Source Code: Reducing False Positives using Machine Learning”, 2020 International Conference on Communication Systems Networks (COMSNETS), 2020.
- 5 Pidikiti Supriya, Myneedi Pavani, Nagarapu Saisushma, Namburi Vimala Kumari, k Vikash, “Loan Prediction by using Machine Learning Models”, International Journal of Engineering and Techniques. Volume 5 Issue 2, Mar-Apr 2019.

APPENDIX

SOURCE CODE

FLASK CODE:

```
from flask import Flask, render_template, request
import pickle
import numpy as np
import warnings
import sys

if not sys.warnoptions:
    warnings.simplefilter("ignore")
warnings.filterwarnings("ignore", category=DeprecationWarning)

model = pickle.load(open('mini.pkl', 'rb'))

app = Flask(__name__, template_folder='templates')

@app.route('/')
def man():
    return render_template('home.html')

@app.route('/predict', methods=['POST'])
def home():
    name = request.form['NAME']
    address = request.form['ADDRESS']
    email = request.form['EMAIL']
    app_income1 = request.form['APP_INCOME']
    coapp_income1 = request.form['COAPP_INCOME']
    loan_amount1 = request.form['LOAN_AMOUNT']
    la_term1 = request.form['LA_TERM']
    cr_history1 = request.form['CR_HISTORY']
    married1 = request.form['MARRIED']
    gender1 = request.form['GENDER']
    dependents1 = request.form['DEPENDENTS']
    education1 = request.form['EDUCATION']
```

```

self_emp1 = request.form['SELF_EMP']
mobile1 = request.form['MOBILE']
property1 = request.form['PROPERTY']
app_income = int(app_income1)
coapp_income = int(coapp_income1)
loan_amount = int(loan_amount1)
la_term = int(la_term1)
if (cr_history1 == 'Clear'):
    cr_history = 1.0
else:
    cr_history = 0.0
if (married1 == 'YES'):
    married = 1
else:
    married = 0
if (gender1 == 'MALE'):
    gender = 1
else:
    gender = 0
dependents_0 = 0
dependents_1 = 0
dependents_2 = 0
dependents_3 = 0
if (dependents1 == 'zero'):
    dependents_0 = 1
if (dependents1 == 'one'):
    dependents_1 = 1
if (dependents1 == 'two'):
    dependents_2 = 1
if (dependents1 == 'three_plus'):
    dependents_3 = 1
if (education1 == 'Graduate'):
    education = 1
else:
    education = 0
if (self_emp1 == 'yes'):
    self_emp = 1
else:
    self_emp = 0
mobile = int(mobile1)
p_rural = 0
p_semi_urban = 0
p_urban = 0

```

```

if (property1 == "rural"):
    p_rural = 1
if (property1 == "semi-urban"):
    p_semi_urban = 1
if (property1 == "urban"):
    p_urban = 1
# arr=np.array([3366, 2200.0, 135.0, 360.0, 1.0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0])
arr = np.array(
    [app_income, coapp_income, loan_amount, la_term, cr_history, gender, married,
dependents_0, dependents_1,
    dependents_2, dependents_3, education, self_emp, p_rural, p_semi_urban, p_urban])
pred = model.predict(arr.reshape(1, -1))
return render_template('prediction_page.html', data=pred)
if __name__ == "__main__":
app.run(debug=True)

```