

PREDICTIVE ANALYTICS FOR LOAN APPROVAL SYSTEM USING MACHINE LEARNING TECHNIQUES

A project work

submitted in partial fulfilment of the requirements for the award

of

Master of Computer Applications

by

K. HIMA JAHNAVI

(Regd. No. 23FE1F0026)

Under the Esteemed Guidance of

KATTA KIRAN, M. Tech

Assistant Professor

Department of Master of Computer Applications



VIGNAN'S LARA

INSTITUTE OF TECHNOLOGY & SCIENCE

(AUTONOMOUS)

Approved by AICTE New Delhi & Affiliated to JNTUK Kakinada

Accredited by **NAAC 'A++'** and **NBA** | **ISO 9001 : 2015**

Vadlamudi - 522 213, Guntur District

April, 2025



VIGNAN'S LARA
INSTITUTE OF TECHNOLOGY & SCIENCE
(AUTONOMOUS)

Approved by AICTE New Delhi & Affiliated to JNTUK Kakinada
Accredited by **NAAC 'A+' and NBA** | **ISO 9001 : 2015**
Vadlamudi - 522 213, Guntur District

CERTIFICATE

This is to certify that the project report entitled “**Predictive analytics for loan approval system using ML techniques**” is a bona fide work done by **K. Hima Jahnavi (23FE1F0026)** under my guidance and submitted in partial fulfilment of the requirements for award of the degree of Master of Computer Applications from Jawaharlal Nehru Technological University, Kakinada. The work embodied in this project report is not submitted to any other university or institute for the award of any degree/diploma.

Project Guide

Katta Kiran, M. Tech

Assistant Professor

Head of the Department

R. VEERA BABU, M. Tech, (Ph. D)

Associate Professor & HOD

External Examiner

DECLARATION

I hereby declare that the project report entitled “**Predictive analytics for loan approval system using ML techniques**” submitted to the JNTUK, is a record of an original work done by **K. Hima Jahnavi** under the Guidance of **Katta Kiran, M. Tech** Assistant Professor of the Department of Master of Computer Applications and this project work is submitted in the partial fulfilment of requirements for the award of Degree of Master of Computer Applications. The results embodied in this project report are not submitted to any other University or Institute for the award of any Degree or Diploma.

Place: Vadlamudi

Date:

K. HIMA JAHNAVI

(Regd. No. 23FE1F0026)

ACKNOWLEDGEMENT

The satisfaction that accompanies with the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

I am glad to expense my deep sense of gratitude to. **KATTA KIRAN, M. Tech**, Assistant professor, Master of Computer Applications for guiding through this project and for encouraging right from the beginning of the project. Every interaction with him was an inspiration. At every step he was there to help me to choose right path.

I am glad to expense my deep sense of gratitude to **Mr. R. VEERA BABU**, Associate professor and Head of the Department, Master of Computer Applications for giving support in completion of project work.

I am glad to expense my deep sense of gratitude to the beloved **chairman Dr. L. RATHAIAH**, and the principal **Dr. K. PHANEENDRA KUMAR** for their encouragement and kind support in carrying out our work.

I thank my parents and others who have rendered help to me directly or indirectly in the completion of project work.

K. HIMA JAHNAVI
(Regd. No. 23FE1F0026)

ABSTRACT

Technology has boosted the existence of humankind the quality of life they live. Every day we are planning to create something new and different. We have a solution for every other problem we have machines to support our lives and make us somewhat complete in the banking sector candidate gets proofs/ backup before approval of the loan amount. The application approved or not approved depends upon the historical data of the candidate by the system. Every day lots of people applying for the loan in the banking sector but Bank would have limited funds. In this case, the right prediction would be very beneficial using some classes-function algorithm. An example the logistic regression, random forest classifier, support vector machine classifier, etc. A Bank's profit and loss depend on the amount of the loans that is whether the Client or customer is paying back the loan. Recovery of loans is the most important for the banking sector. The improvement process plays an important role in the banking sector. The historical data of candidates was used to build a machine learning model using different classification algorithms. The main objective of this paper is to predict whether a new applicant granted the loan or not using machine learning models trained on the historical data set.

CONTENTS

S.NO	TITLE	PAGE NO
1	INTRODUCTION	1
	1.1 Motivation	2
	1.2 Problem definition	2
	1.3 Objective of project	2
2	LITERATURE SURVEY	3
3	SYSTEM ANALYSIS	7
	3.1 Existing system	8
	3.2 Proposed system	9
	3.3 Software requirements	9
	3.4 System study	10
4	SYSTEM DESIGN	12
	4.1 System Architecture	13
	4.2 Data flow diagram	14
	4.3 UML diagram	16
	4.4 Implementation	23
5	SOFTWARE ENVIRONMENT	25
6	SYSTEM TEST	45
7	CODE	50
8	SCREENSHOTS	66
9	CONCLUSION	69
10	REFERENCES	71

LIST OF FIGURES

S NO	FIG NO	FIGURE NAME	PAGE NO
1	4.4.1	System Architecture	13
2	4.2.1	Data Flow Diagram	15
3	4.3.1	Use Case Diagram	17
4	4.3.2	Class Diagram	18
5	4.3.3	Object Diagram	19
6	4.3.4	State Diagram	20
7	4.3.5	Activity Diagram	21
8	4.3.6	Sequence Diagram	22

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

1.1 MOTIVATION

The motivation behind this project stems from the pivotal role technology plays in improving human life. Daily advancements prompt us to create innovative solutions, especially in the banking sector, where loan approvals are crucial. Limited funds necessitate accurate predictions, driving the use of machine learning algorithms like logistic regression and random forest for efficient and informed decision-making. The goal is to enhance the banking sector's efficiency, ensure prudent loan approvals, and contribute to the sector's overall financial health and sustainability.

1.2 PROBLEM DEFINITION

The challenge lies in optimizing loan approval processes within the banking sector. With numerous loan applications daily and limited funds, predicting approvals becomes vital. Utilizing machine learning algorithms like logistic regression, random forest, and support vector machines, the problem aims to enhance loan approval predictions based on historical candidate data for improved banking sector efficiency.

1.3 OBJECTIVE OF PROJECT

The primary objective of this project is to employ machine learning models, including logistic regression, random forest classifier, and support vector machine classifier, to analyze historical data of loan applicants in the banking sector. The goal is to predict loan approval outcomes accurately, optimizing decision-making processes and contributing to the sector's overall efficiency and profitability.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

[1] Amruta S. Aphale and R. Prof. Dr. Sandeep. R Shinde, “Predict Loan Approval in Banking System Machine Learning Approach for Cooperative Banks Loan Approval”, International Journal of Engineering Trends and Applications (IJETA), vol. 9, issue 8, 2020)

In today’s world, taking loans from financial institutions has become a very common phenomenon. Everyday a large number of people make application for loans, for a variety of purposes. But all these applicants are not reliable and everyone cannot be approved. Every year, we read about a number of cases where people do not repay bulk of the loan amount to the banks due to which they suffer huge losses. The risk associated with making a decision on loan approval is immense. So, the idea of this project is to gather loan data from multiple data sources and use various machine learning algorithms on this data to extract important information. This model can be used by the organizations in making the right decision to approve or reject the loan request of the customers. In this paper, we examine a real bank credit data and conduct several machine learning algorithms on the data for that determine credit worthiness of customers in order to formulate bank risk automated system.

[2] Loan Prediction Using Ensemble Technique, International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 3, March 2016

Extending credit to individuals is necessary for markets and society to function smoothly. Estimating the probability that an individual would default on their loan, is useful for banks to decide whether to sanction a loan to the individual or not. We introduce an effective prediction technique that helps the banker to predict the credit risk for customers who have applied for loan. A prototype is described in the paper which can be used by the organizations for making the correct or right decision for approve or reject the request for

loan of the customers. The paper uses three different models (SVM Model, Random Forest Network and Tree Model for Genetic Algorithm) and the Ensemble Model, which combines these three models and analyses the credit risk for optimum results.

[3] Exploratory data analysis https://en.wikipedia.org/wiki/Exploratory_data_analysis

In statistics, exploratory data analysis (EDA) is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling and thereby contrasts traditional hypothesis testing. Exploratory data analysis has been promoted by John Tukey since 1970 to encourage statisticians to explore the data, and possibly formulate hypotheses that could lead to new data collection and experiments. EDA is different from initial data analysis (IDA), which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing, and handling missing values and making transformations of variables as needed. EDA encompasses IDA.

[4] ACCURATE LOAN APPROVAL PREDICTION BASED ON MACHINE LEARNING APPROACH AUTHORS: J. Tejaswini¹, T. MohanaKavya, R. Devi Naga Ramya, P. Sai TriveniVenkata Rao Maddumala.

Loan approval is a very important process for banking organizations. Banking Industry always needs a more accurate predictive modeling system for many issues. Predicting credit defaulters is a difficult task for the banking industry. The system approved or rejects the loan applications. Recovery of loans is a major contributing parameter in the financial statements of a bank. It is very difficult to predict the possibility of payment of loan by the customer. Machine Learning (ML) techniques are very useful in predicting outcomes for large amount of data. In this paper three machine learning algorithms, Logistic Regression (LR), Decision Tree (DT) and Random Forest (RF) are applied to predict the loan approval

of customers. The experimental results conclude that the accuracy of Decision Tree machine learning algorithm is better as compared to Logistic Regression and Random Forest machine learning approached.

[5] Predictive and probabilistic approach using logistic regression: Application to prediction of loan approval AUTHORS: Vaidya

Decision taking is attained by probabilistic and predictive approaches developed by various machine learning algorithms. This paper discusses about logistic regression ad its mathematical representation. This paper adheres to logistic regression as a machine learning tool in order to actualize the predictive and probabilistic approaches to a given problem of loan approval prediction. Using logistic regression as a tool, this paper specifically delineates about whether or not loan for a set of records of an applicant will be approved. Furthermore, it also discusses about other real-world applications of this machine learning mode.

CHAPTER-3

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

There is a major problem that many people not able to back the loans to banks. And banks are going in losses. Banks received many applications for loan approval day by day and not everyone gets approved. Most of the banks have their own credit score and risk assessment techniques so as to check that the loan is approved or not. Why this loan problem arises this question will get resolved in just a few minutes. The main reason to get a loan is to fulfill the needs of something. For a businessman he/she wants to increase the business or if that company is at loss to get over from that he/she needs a loan. In middle-class people wants to fulfill their needs so they want a loan. So, the main thing of this to fulfill the needs of someone or for something. The answer to this question that not everybody can loan because if he/she is not able to return then who is providing the loan he/she or the company or the bank that is providing the loan will get in the loss. So, first who is providing the loan they have to verify or set some criteria that who is taking the loan is able to return or not. Like in banks like we have a credit card facility but not everybody gets a credit card. For that, a credit score is there to check whether eligible or not. For credit score one should have a good credit score then he/she be able to get a loan. Some criteria like a source of income should be there for getting a credit card. Banks provide loans on behalf of one who is taking the loan he/she should provide some documents and verify. Like some company not able to provide the loans then banks get in loss and they called it NBFC's. During this project data processing algorithm are going to study loan-approved data might help in predicting. The existing system employs Support Vector Machine (SVM) as a key classification algorithm to predict loan approval based on historical data. SVM analyzes candidate profiles, considering factors like credit history, income, and more. This machine learning model aids banks in decision-making, optimizing loan approval processes, and enhancing overall efficiency in the banking sector.

DISADVANTAGES

1. SVM may struggle with large datasets and is sensitive to noise.
2. Potentially leading to sub optimal performance

3.2 PROPOSED SYSTEM

In this, we are going to discuss the advantage of loan prediction. In this system, we are going to predict that the person who is applying for a loan can repay or not. If the client can repay then we predict that yes, eligible for a loan. And if the candidate fails then we predict that client is not eligible. The advantage of this system is that we provided some conditions by setting the algorithms and just by evaluating the details, we get to know eligibility criteria that client is eligible or not. This system may be built which is able to take various inputs from the users like salary, address, loan amount, loan duration, etc and provide a prediction of whether their application will be approved by the bank or not.

ADVANTAGES

1. Fraud Prevention
2. Personalization
3. Automating Processes

3.3 SOFTWARE REQUIREMENT SPECIFICATION

H/W System Configuration:-

Processor	- P-IV
RAM	- 2 GB(min)
Hard Disk	- 40 GB
Keyboard	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA 21"

Software Requirements:

Operating system	- Windows 7 Ultimate or above.
Coding Language	- Python
Python IDE	- pycharm, jupyter note book

3.4 SYSTEM STUDY**FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

◆ ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

◆ TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

◆ SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER-4

SYSTEM DESIGN

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

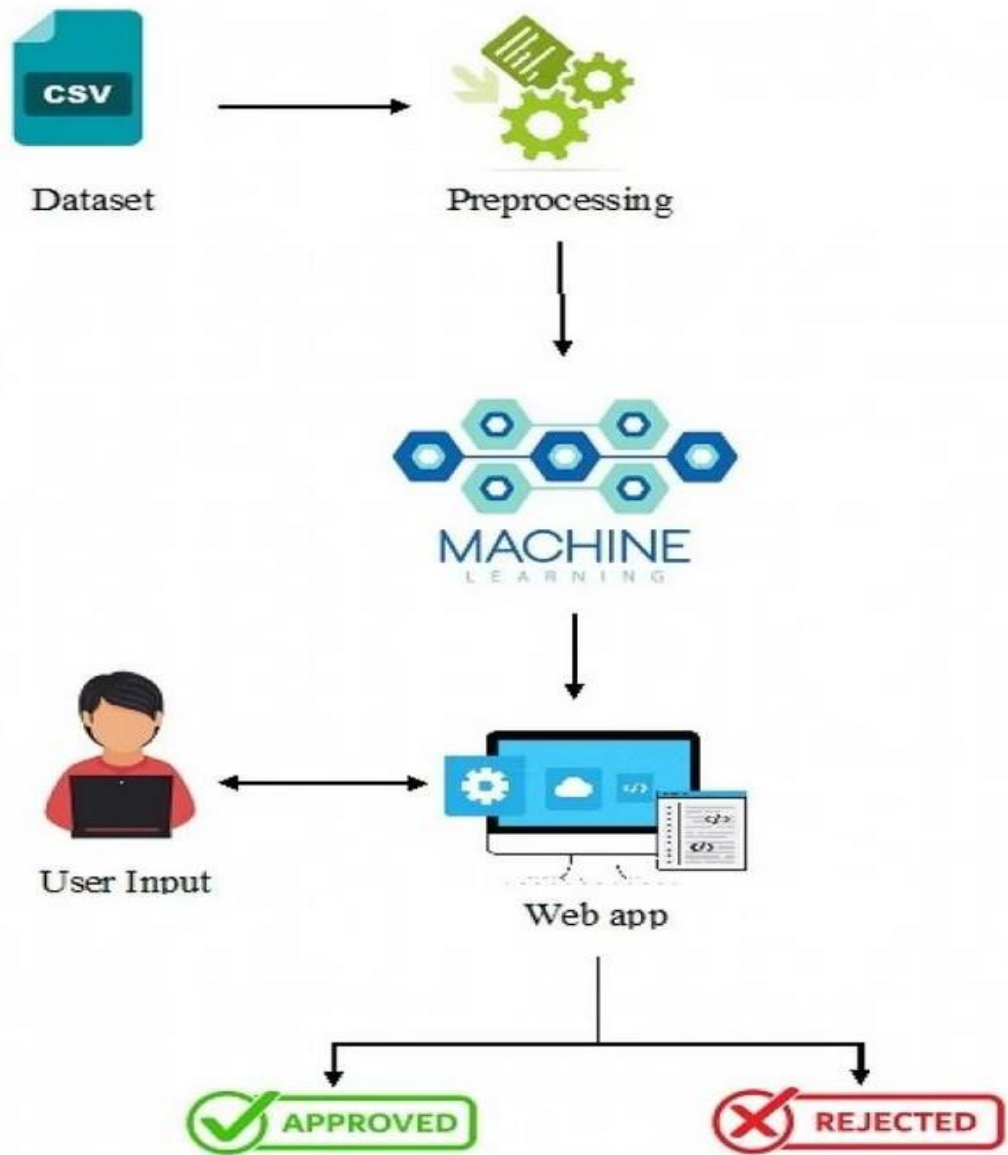
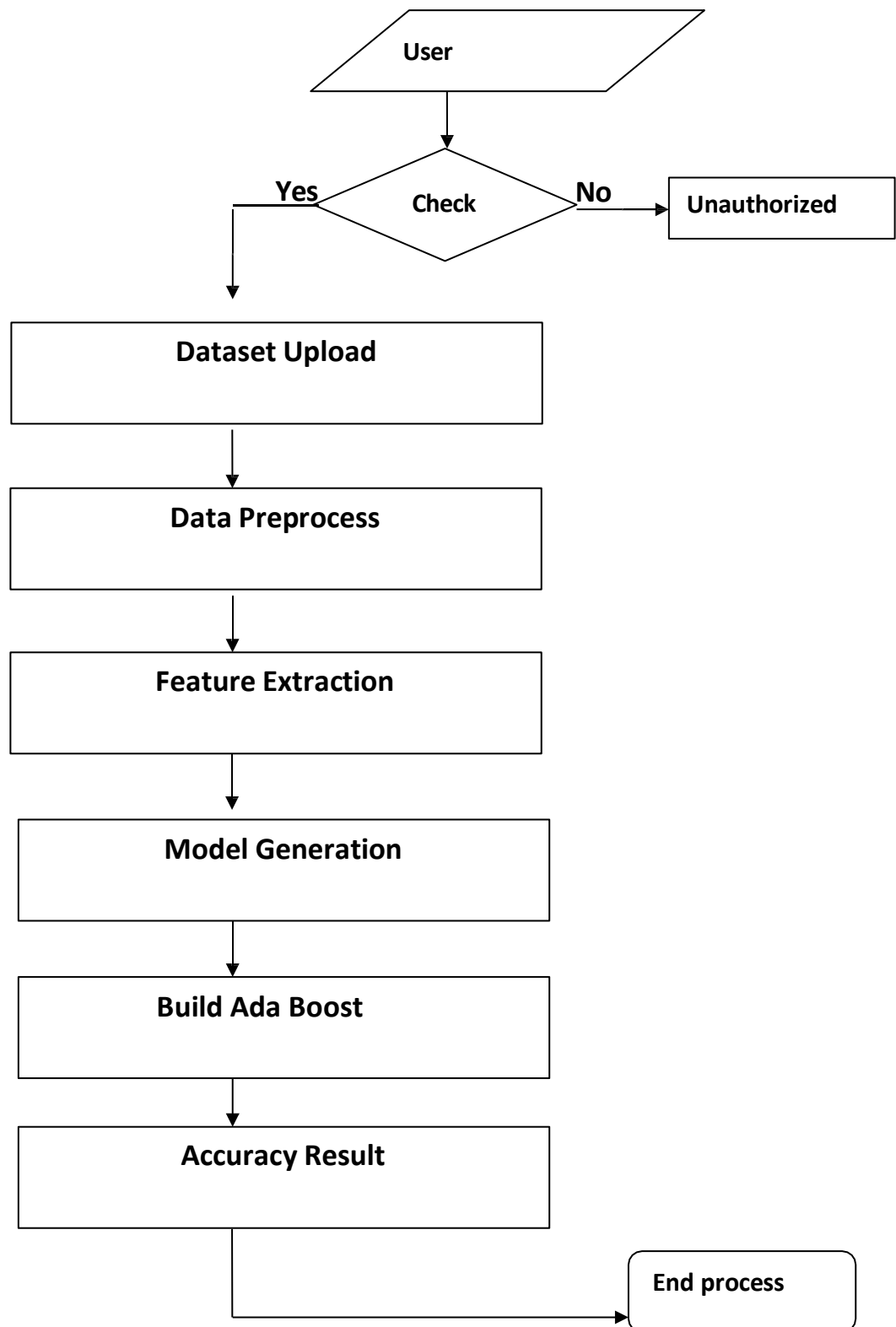


Fig:4.4.1 System Architecture

4.2 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Fig:4.2.1 Data Flow Diagram



4.3 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

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

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.

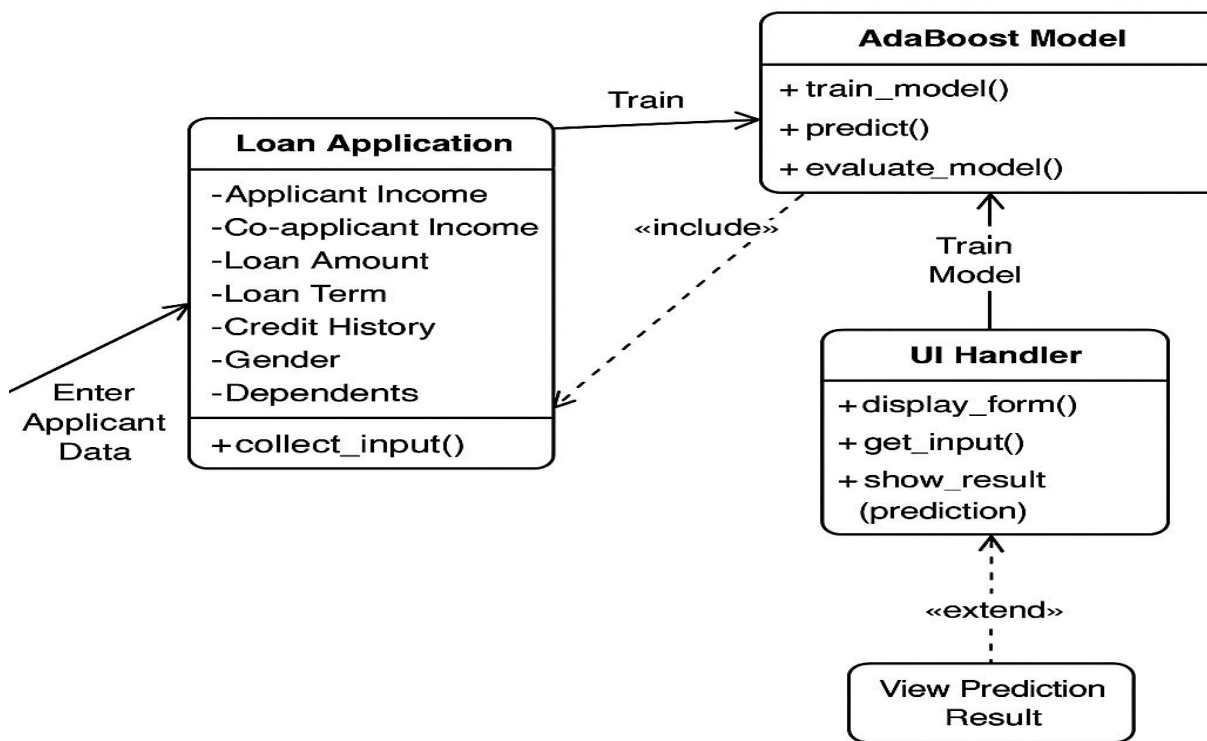


Fig:4.3.1 Use Case Diagram

CLASS DIAGRAM:

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

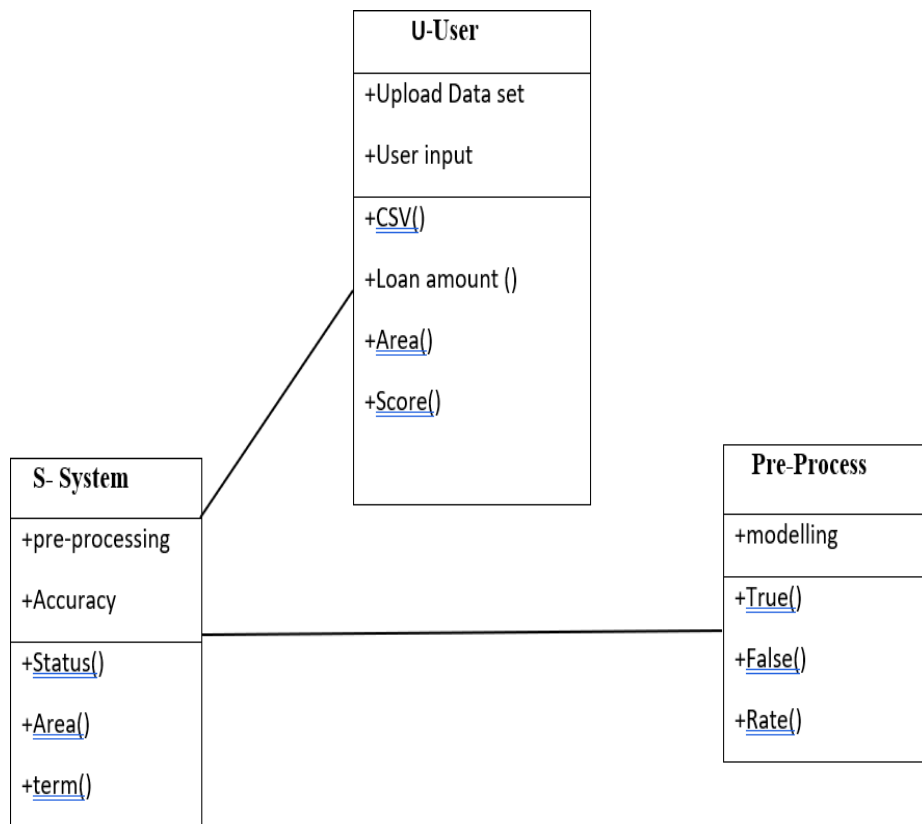


Fig:4.3.2 Class Diagram

OBJECT DIAGRAM:

The object diagram is a special kind of class diagram. An object is an instance of a class. This essentially means that an object represents the state of a class at a given point of time while the system is running. The object diagram captures the state of different classes in the system and their relationships or associations at a given point of time.

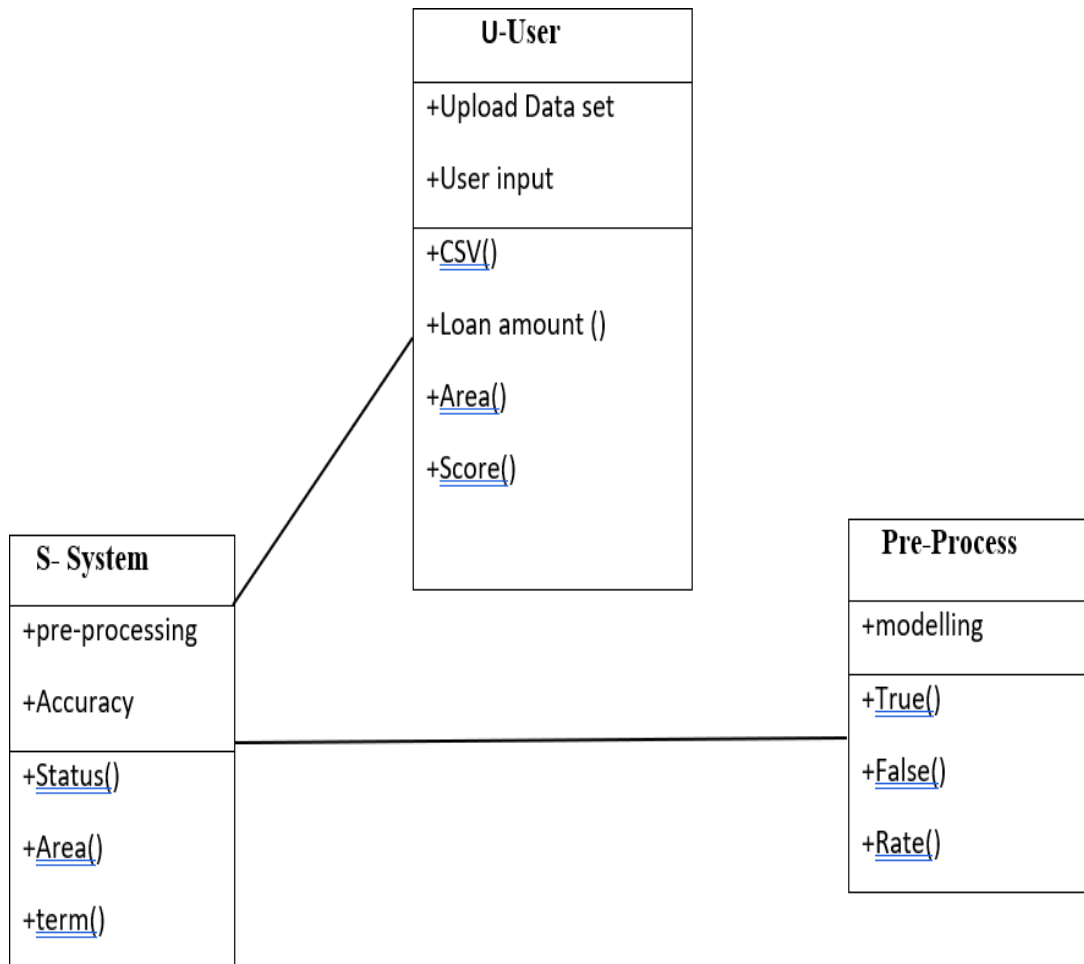


Fig:4.3.3 Object Diagram

STATE DIAGRAM:

A state diagram, as the name suggests, represents the different states that objects in the system undergo during their life cycle. Objects in the system change states in response to events. In addition to this, a state diagram also captures the transition of the object's state from an initial state to a final state in response to events affecting the system.

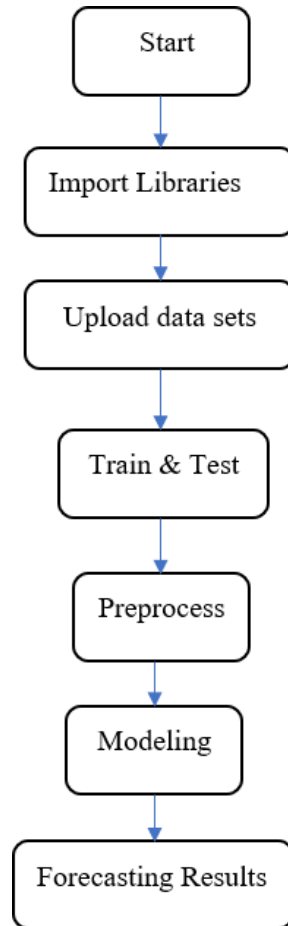


Fig:4.3.4 State Diagram

ACTIVITY DIAGRAM:

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.

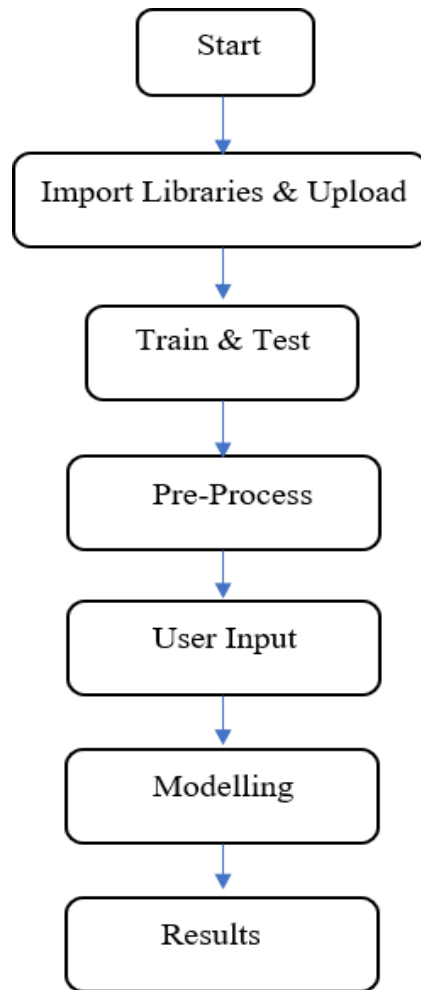


Fig:4.3.5 Activity Diagram

SEQUENCE DIAGRAM:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".

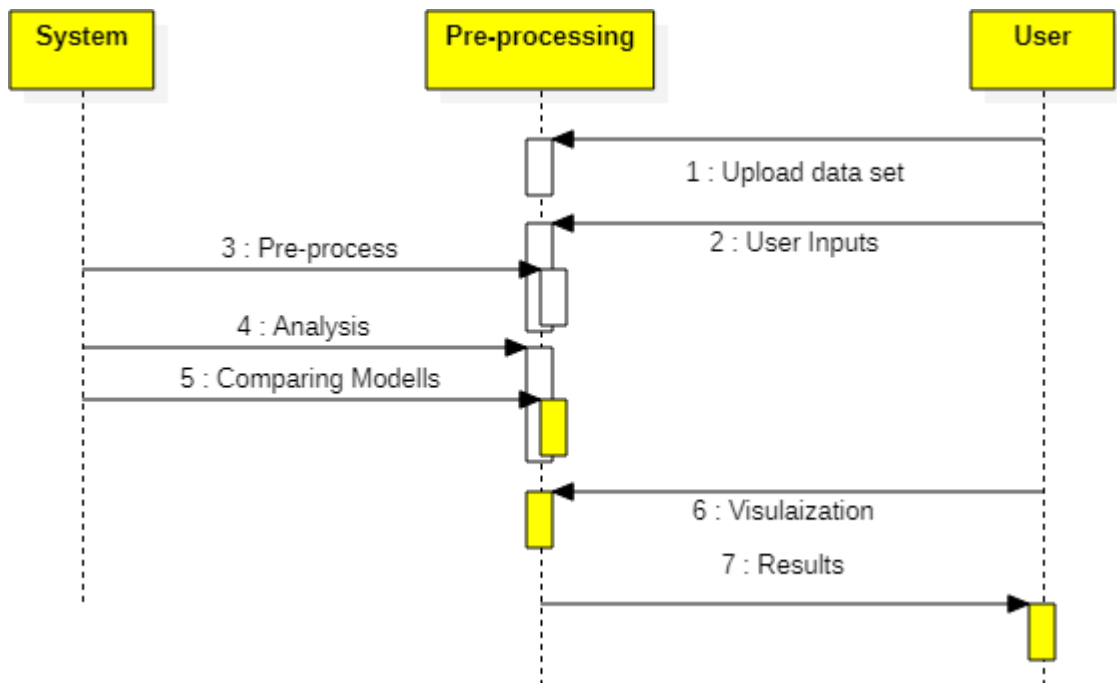


Fig:4.3.6 Sequence Diagram

4.4 IMPLEMENTATION:

MODULES:

Data Collection: The first step is to collect data from various sources like loan applications, credit history, income details, etc. This data can be collected from various sources like databases, API endpoints, etc. and stored in a data warehouse.

Data Pre-processing: The collected data needs to be cleaned and processed before it can be used for training machine learning models. This step involves handling missing values, outliers, and removing duplicates.

Feature Engineering: Once the data is cleaned, the next step is to extract relevant features from the data. This involves selecting the most important features for the loan approval process, like credit score, income, debt-to-income ratio, employment status, etc.

Model Selection: After feature engineering, the next step is to select the appropriate machine learning models to train on the data. This can include classification algorithms like logistic regression, decision trees, or random forests.

Model Training: Once the appropriate machine learning models are selected, the next step is to train them on the pre-processed data. The data is split into training and validation sets to evaluate the performance of the models.

Model Evaluation: The trained models are evaluated using various evaluation metrics like accuracy, precision, recall, and F1-score. The best-performing model is selected for deployment.

Model Deployment: The final step is to deploy the selected model into the loan approval system. The model can be integrated into the loan application process to automatically approve or reject loan applications based on the input features.

CHAPTER – 5

SOFTWARE ENVIRONMENT

5. SOFTWARE ENVIRONMENT

What is Python :-

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.
- The biggest strength of Python is huge collection of standard libraries which can be used for the following –
 - Machine Learning
 - GUI Applications (like Kivy, Tkinter, PyQtetc.)
 - Web frameworks like Django (used by YouTube, Instagram, Dropbox)
 - Image processing (like Opencv, Pillow)
 - Web scraping (like Scrapy, BeautifulSoup, Selenium)
 - Test frameworks
 - Multimedia

Advantages of Python:-

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contains code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print ‘Hello World’. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write

Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support. The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle. The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java Data Base Connectivity) and ODBC (Open Data Base Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python : -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning :

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories of Machine Learning

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning. Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in

regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machines Learning

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to

go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are –

Quality of data – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons – As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting – If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality – Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment – Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machine Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

How to Start Learning Machine Learning?

Arthur Samuel coined the term “Machine Learning” in 1959 and defined it as a “Field of study that gives computers the capability to learn without being explicitly programmed”. And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a 344% growth and an average base salary of **\$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let’s get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!! Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking

is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.

- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website

like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As **ML algorithms** gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning :-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

Python Development Steps :-

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a

Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools `lambda`, `map`, `filter` and `reduce`, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting Unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 3.0:

- `print` is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. `int`. `long` is `int` as well.
- The division of two integers returns a float instead of an integer. `//` can be used to have the "old" behavior.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Modules Used in Project :-

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing

and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

CHAPTER – 6

SYSTEM TEST

6. SYSTEM TEST

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input** - identified classes of valid input must be accepted.
- Invalid Input** - identified classes of invalid input must be rejected.
- Functions** - identified functions must be exercised.
- Output** - identified classes of application outputs must be exercised.
- Systems/Procedures** - interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

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
- No duplicate entries should be allowed
- All links should take the user to the correct page.

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.

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.

CHAPTER – 7

CODE

7. CODE

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.metrics import f1_score

train_df = pd.read_csv("/content/train.csv")

test_df = pd.read_csv("/content/test.csv")

print('Train Set:',train_df.shape,'\nTest Set:',test_df.shape)

train_df.head()

test_df.head()

train_df.drop('Unnamed: 0',axis=1,inplace=True)

train_df.info()

test_df.info()

train_df.describe(include='all')

test_df.describe(include='all')

train_df.drop('Loan_ID',axis=1,inplace=True)
```

```
test_df.drop('Loan_ID',axis=1,inplace=True)

train_df.isnull().sum()

test_df.isnull().sum()

train_df.shape[0] - train_df.dropna().shape[0]

train_df[train_df.Gender.isnull()]

test_df[test_df.Gender.isnull()]

train_df.Gender.value_counts()

mode_gender=train_df.Gender.mode()[0]

print(mode_gender)

train_df.Gender.fillna(mode_gender,inplace=True)

test_df.Gender.fillna(mode_gender,inplace=True)

print(train_df.loc[59])

print(test_df.loc[66])

train_df[train_df.Married.isnull()]

test_df[test_df.Married.isnull()]

mode_married=train_df.Married.mode()[0]

print(mode_married)

train_df.Married.fillna(mode_married,inplace=True)

test_df.Married.fillna(mode_married,inplace=True)
```

```
test_df.loc[29]

train_df[train_df.Dependents.isnull()]

test_df[test_df.Dependents.isnull()]

train_df.Dependents.value_counts()

mode_Dependents=train_df.Dependents.mode()[0]

print(mode_Dependents)

train_df.Dependents.fillna(mode_Dependents,inplace=True)

test_df.Dependents.fillna(mode_Dependents,inplace=True)

print(train_df.loc[11])

print(test_df.loc[26])

train_df[train_df.Self_Employed.isnull()]

test_df[test_df.Self_Employed.isnull()]

mode_Self_Employed=train_df.Self_Employed.mode()[0]

print(mode_Self_Employed)

train_df.Self_Employed.fillna(mode_Self_Employed,inplace=True)

test_df.Self_Employed.fillna(mode_Self_Employed,inplace=True)

print(train_df.loc[39])

print(test_df.loc[42])
```

```
train_df[train_df.LoanAmount.isnull()]

test_df[test_df.LoanAmount.isnull()]

train_df.LoanAmount.hist()

median_loanamount=train_df.LoanAmount.median()

print(median_loanamount)

train_df.LoanAmount.fillna(median_loanamount,inplace=True)

test_df.LoanAmount.fillna(median_loanamount,inplace=True)

print(train_df.loc[104])

print(test_df.loc[11])

train_df[train_df.Loan_Amount_Term.isnull()]

test_df[test_data.Loan_Amount_Term.isnull()]

median_Loan_Amount_Term=train_df.Loan_Amount_Term.mode()[0]

print(median_Loan_Amount_Term)

train_df.Loan_Amount_Term.fillna(median_Loan_Amount_Term,inplace=True)

test_df.Loan_Amount_Term.fillna(median_Loan_Amount_Term,inplace=True)

print(train_df.loc[3])
```

```
print(test_df.loc[25])

train_df.dropna(inplace=True)

train_df.reset_index(drop=True)

train_df[train_df.Credit_History.isnull()]

test_df[test_df.Credit_History.isnull()]

mode_Credit_History=train_df.Credit_History.mode()[0]

print(mode_Credit_History)


train_df.Credit_History.fillna(mode_Credit_History,inplace=True)

test_df.Credit_History.fillna(mode_Credit_History,inplace=True)


print(test_df.loc[31])

train_df.isnull().sum().sum()

test_df.isnull().sum().sum()

plt.figure(figsize=(20,6))

sns.heatmap(loan_data.corr(), cmap="YlGnBu", annot=True)

plt.show()

plt.figure(figsize=(10,4))

plt.subplot(1,2,1)
```

```

train_df.ApplicantIncome.hist()

plt.title('ApplicantIncome')

plt.subplot(1,2,2)

train_df.CoapplicantIncome.hist()

plt.title('CoapplicantIncome')

plt.tight_layout()

plt.show()

train_df.head()

train_df.columns

train_df.reset_index(drop=True,inplace=True)

cat_cols=['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed','Credit_History',
'Property_Area']

for i in cat_cols:

    print(i)

    print('Train set:',sorted(train_data[i].unique()))

    print('Test set:',sorted(test_data[i].unique()))

from sklearn.preprocessing import OneHotEncoder

ohe = OneHotEncoder(handle_unknown = 'ignore')

load_df_encoded = ohe.fit_transform(train_df[cat_cols]).toarray()

```

```

test_df_encoded = ohe.transform(test_df[cat_cols]).toarray()

load_df_encoded

len(load_df_encoded)

len(train_df)

loan_df=train_df.drop(cat_cols,axis=1);print(loan_df.shape)

test_df=test_df.drop(cat_cols,axis=1);print(test_df.shape)

loan_df

loan_df=pd.concat([loan_df,pd.DataFrame(load_df_encoded,
columns=ohe.get_feature_names_out(loan_data[cat_cols].columns))],axis=1);print(loan_
df.shape)

test_df=pd.concat([test_df,pd.DataFrame(test_df_encoded,
columns=ohe.get_feature_names_out(test_data[cat_cols].columns))],axis=1);print(test_df
.shape)

loan_df.tail()

round(loan_df.Loan_Status.value_counts()/loan_df.shape[0]*100,2)

target_column = train_df.columns[-1]

X = train_df.drop(columns=[target_column])

y = train_df[target_column]

label_encoders = {}

for col in X.select_dtypes(include=['object']).columns:

    le = LabelEncoder()

    X[col] = le.fit_transform(X[col].astype(str))

```

```

label_encoders[col] = le

if y.dtype == 'object':

target_encoder = LabelEncoder()

y = target_encoder.fit_transform(y)

imputer = SimpleImputer(strategy='mean')

X = imputer.fit_transform(X)

smote = SMOTE(random_state=42)

X, y = smote.fit_resample(X, y)

scaler = StandardScaler()

X = scaler.fit_transform(X)

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)

models = {

"Random Forest": RandomForestClassifier(n_estimators=100, random_state=42),

"Logistic Regression": LogisticRegression(max_iter=5000), # Increased max_iter

"XGBoost": XGBClassifier(use_label_encoder=False, eval_metric='logloss'),

"AdaBoost": AdaBoostClassifier(),

"Gradient Boost": GradientBoostingClassifier()

}

# Random Forest Model

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

```

```
rf_model.fit(X_train, y_train)

y_pred_rf = rf_model.predict(X_val)

accuracy_rf = accuracy_score(y_val, y_pred_rf)

print(f'Random Forest Accuracy: {accuracy_rf * 100:.2f}%')

# Logistic Regression Model

lr_model = LogisticRegression(max_iter=5000)

lr_model.fit(X_train, y_train)

y_pred_lr = lr_model.predict(X_val)

accuracy_lr = accuracy_score(y_val, y_pred_lr)

print(f'Logistic Regression Accuracy: {accuracy_lr * 100:.2f}%')

# XGBoost Model

xgb_model = XGBClassifier(use_label_encoder=False, eval_metric='logloss')

xgb_model.fit(X_train, y_train)

y_pred_xgb = xgb_model.predict(X_val)

accuracy_xgb = accuracy_score(y_val, y_pred_xgb)

print(f'XGBoost Accuracy: {accuracy_xgb * 100:.2f}%')

# AdaBoost Model

ada_model = AdaBoostClassifier()

ada_model.fit(X_train, y_train)

y_pred_ada = ada_model.predict(X_val)
```

```

accuracy_ada = accuracy_score(y_val, y_pred_ada)

print(f'AdaBoost Accuracy: {accuracy_ada * 100:.2f}%')

# Gradient Boost Model

gb_model = GradientBoostingClassifier()

gb_model.fit(X_train, y_train)

y_pred_gb = gb_model.predict(X_val)

accuracy_gb = accuracy_score(y_val, y_pred_gb)

print(f'Gradient Boost Accuracy: {accuracy_gb * 100:.2f}%')

# SVM Model

from sklearn.svm import SVC # Import the SVC class

svm_model = SVC()

svm_model.fit(X_train, y_train)

y_pred_svm = svm_model.predict(X_val)

accuracy_svm = accuracy_score(y_val, y_pred_svm)

print(f'SVM Accuracy: {accuracy_svm * 100:.2f}%')

# Identify the best model

accuracies = {

    "Random Forest": accuracy_rf,

    "Logistic Regression": accuracy_lr,

    "XGBoost": accuracy_xgb,

```

```

    "AdaBoost": accuracy_ada,

    "Gradient Boost": accuracy_gb,

    "SVM": accuracy_svm

}

best_model = max(accuracies, key=accuracies.get)

print(f'Best Model: {best_model} with Accuracy: {accuracies[best_model] * 100:.2f}%')


import pickle

with open('loan_status_model.pkl','wb') as f:

    pickle.dump(rf_model,f)

import json

columns = {

    'data_columns' : [col.lower() for col in train_df.drop(columns=[target_column]).columns]

}

with open("columns.json","w") as f:

    f.write(json.dumps(columns))

original_columns = train_df.drop(columns=[target_column]).columns

print(original_columns)

train_df.Property_Area.value_counts()

```

```

!pip install streamlit

import streamlit as st

import numpy as np

import pandas as pd

import base64

import pickle

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

st.set_page_config(page_title="Loan Prediction App", layout="wide")

def add_bg_from_local(image_file):

    with open(image_file, "rb") as img_file:

        encoded = base64.b64encode(img_file.read()).decode()

    st.markdown(

        f"""

        <style>

        .stApp {{

            background-image: url("data:image/jpg;base64,{encoded}");

            background-size: cover;

            background-position: center;

            background-repeat: no-repeat;

        }}

```

```

</style>

"""

unsafe_allow_html=True

)

add_bg_from_local('loan_image.jpg')

st.markdown("<h1 style='text-align: center; color: black;'>Loan Application Evaluator</h1>",
unsafe_allow_html=True)

col1, col2 = st.columns(2)

with col1:

    gender = st.selectbox("Gender", ["Male", "Female"])

    married = st.selectbox("Marital Status", ["No", "Yes"])

    dependents = st.number_input("Number of Dependents", min_value=0, max_value=4)

    education = st.selectbox("Education", ["Not Graduate", "Graduate"])

    self_employed = st.selectbox("Self Employed", ["No", "Yes"])

    property_area = st.selectbox("Property Area", ["Rural", "Semiurban", "Urban"])

with col2:

    applicant_income = st.number_input("Applicant Income (in thousands)", min_value=0, value=0)

    coapplicant_income = st.number_input("Co-applicant Income (in thousands)", min_value=0,
value=0)

    loan_amount = st.number_input("Loan Amount (in thousands)", min_value=0, value=0)

```

```

loan_amount_term = st.number_input("Loan Amount Term (in months)", min_value=0, value=0)

credit_history = st.selectbox("Credit History (0 = No history, 1 = Good history)", [0, 1])

user_data = {

    'Gender': gender,

    'Married': married,

    'Dependents': dependents,

    'Education': education,

    'Self_Employed': self_employed,

    'ApplicantIncome': applicant_income,

    'CoapplicantIncome': coapplicant_income,

    'LoanAmount': loan_amount,

    'Loan_Amount_Term': loan_amount_term,

    'Credit_History': credit_history,

    'Property_Area': property_area

}

def preprocess_features(features):

    features.interpolate(method='linear', inplace=True)

    features['Gender'].fillna(features['Gender'].mode()[0], inplace=True)

    features['Married'].fillna(features['Married'].mode()[0], inplace=True)

    features['Dependents'].fillna(features['Dependents'].mode()[0], inplace=True)

```

```

features['Self_Employed'].fillna(features['Self_Employed'].mode()[0], inplace=True)

features.replace({'Married': {'No': 0, 'Yes': 1},

                  'Gender': {'Male': 1, 'Female': 0},

                  'Self_Employed': {'No': 0, 'Yes': 1},

                  'Property_Area': {'Rural': 0, 'Semiurban': 1, 'Urban': 2},

                  'Education': {'Graduate': 1, 'Not Graduate': 0}}, inplace=True)

features = features.replace(to_replace='3+', value=4)

return features

processed_data = preprocess_features(pd.DataFrame(user_data, index=[0]))

if st.button("Predict Loan Status"):

    prediction = model.predict(processed_data)

    if prediction[0] == 1:

        st.success("🎉 Congratulations! Your loan is likely to be approved.")

    else:

        st.error(" Sorry, your loan is likely to be rejected.")

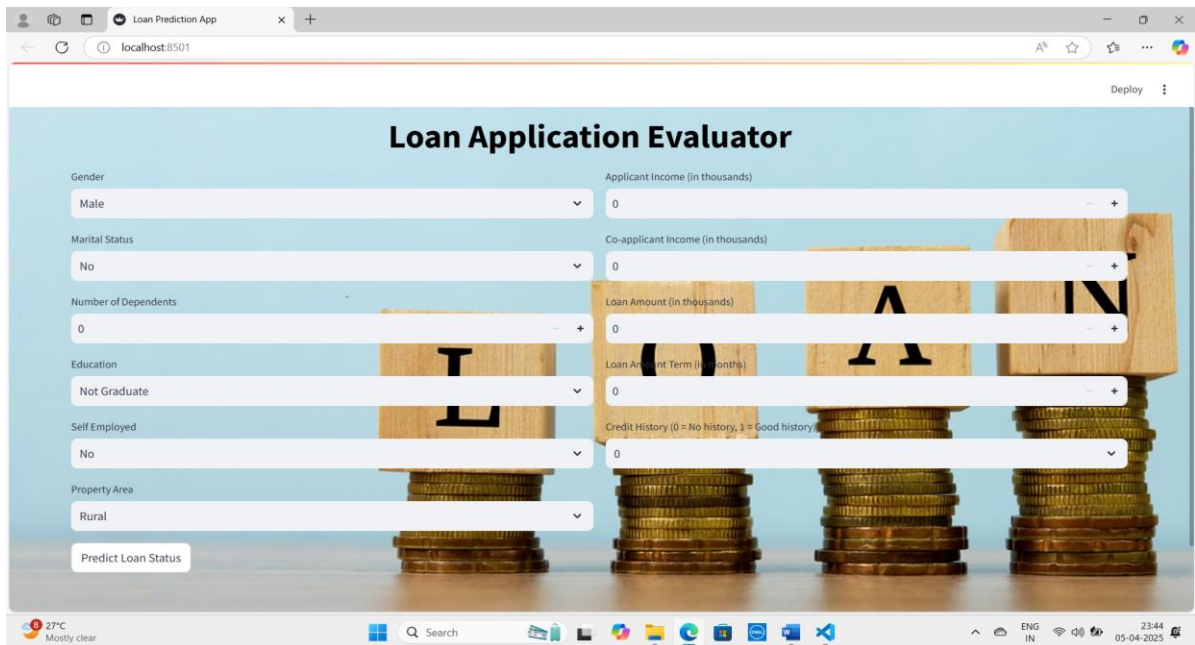
```


CHAPTER – 8

SCREENSHOTS

8. SCREENSHOTS

STEP-1: Open the Web-app and navigate to the prediction form page where we fill our data.

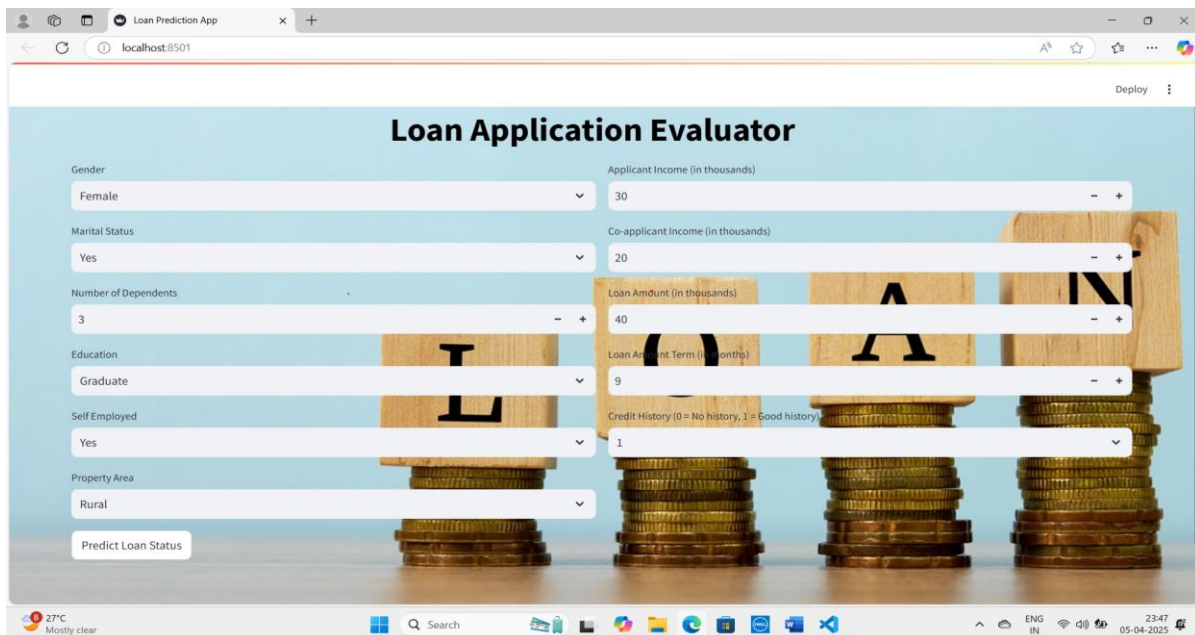


The screenshot shows a web browser window with the title "Loan Prediction App" and the address bar displaying "localhost:8501". The page features a "Loan Application Evaluator" form with the following fields and values:

Field	Value
Gender	Male
Applicant Income (in thousands)	0
Marital Status	No
Co-applicant Income (in thousands)	0
Number of Dependents	0
Loan Amount (in thousands)	0
Education	Not Graduate
Loan Amount Term (in months)	0
Self Employed	No
Credit History (0 = No history, 1 = Good history)	0
Property Area	Rural

A "Predict Loan Status" button is located at the bottom left of the form. The background of the form is a light blue gradient with a stack of gold coins and wooden blocks with letters "L", "A", and "N" visible.

STEP-2: Fill all the data in the given form and press predict button to get the prediction.(All the data will then be sent to the backend script and then to the prediction model for processing.)



The screenshot shows the same web browser window as the previous one, but with the form fields filled with data:

Field	Value
Gender	Female
Applicant Income (in thousands)	30
Marital Status	Yes
Co-applicant Income (in thousands)	20
Number of Dependents	3
Loan Amount (in thousands)	40
Education	Graduate
Loan Amount Term (in months)	9
Self Employed	Yes
Credit History (0 = No history, 1 = Good history)	1
Property Area	Rural

The "Predict Loan Status" button remains at the bottom left. The background image is consistent with the previous screenshot.

STEP-3: View the predicted result

Predicted result for Loan Approval

The screenshot shows a web browser window with the URL `localhost:8501`. The application is titled "Loan Application Evaluator" and features a "Deploy" button in the top right corner. The form contains the following fields and values:

Field	Value
Gender	Female
Marital Status	Yes
Number of Dependents	4
Education	Graduate
Self Employed	Yes
Property Area	Semiurban
Applicant Income (in thousands)	30
Co-applicant Income (in thousands)	20
Loan Amount (in thousands)	50
Loan Amount Term (in months)	12
Credit History (0 = No history, 1 = Good history)	1

A red "Predict Loan Status" button is located below the form. At the bottom of the form, a green message states: "Congratulations! Your loan is likely to be approved." The background of the application features a stack of gold coins and wooden blocks with the letters "L", "A", and "N". The Windows taskbar at the bottom shows the date as 05-04-2025 and the time as 23:34.

Predicted result for Loan Rejection

The screenshot shows the same "Loan Application Evaluator" web application, but with different input values. The "Predict Loan Status" button is now red. At the bottom of the form, a red message states: "Sorry, your loan is likely to be rejected." The input values are as follows:

Field	Value
Gender	Male
Marital Status	No
Number of Dependents	0
Education	Not Graduate
Self Employed	No
Property Area	Rural
Applicant Income (in thousands)	0
Co-applicant Income (in thousands)	0
Loan Amount (in thousands)	0
Loan Amount Term (in months)	0
Credit History (0 = No history, 1 = Good history)	0

The background and Windows taskbar are identical to the previous screenshot, showing the date as 05-04-2025 and the time as 21:48.

CHAPTER – 9

CONCLUSION

9. CONCLUSION

Prediction accuracy is sweet for both datasets. In some situations like client going through some disaster so here the algorithm cannot predict the appropriate result. This research paper can find out the client is potential and repay the loan and the accuracy is good. loan duration, loan amount, age, income are the most important factors for finding out there. The system approved or rejects the loan applications. Recovery of loans is a major contributing parameter in the financial statements of a bank. It is very difficult to predict the possibility of payment of loan by the customer. Machine Learning (ML) techniques are very useful in predicting outcomes for large amount of data. In our project, three machine learning algorithms, Logistic Regression (LR), Decision Tree (DT), Random Forest (RF) and Adaptive Boost are applied to predict the loan approval of customers. The experimental results conclude that the accuracy of Adaptive Boost machine algorithm is better than compared to Random Forest, Logistic Regression and SVM machine learning approaches.

CHAPTER – 10

REFERENCES

10. REFERENCES

- [1] Amruta S. Aphale and R. Prof. Dr. Sandeep. R Shinde, “Predict Loan Approval in Banking System Machine Learning Approach for Cooperative Banks Loan Approval”, International Journal of Engineering Trends and Applications (IJETA), vol. 9, issue 8, 2020)
- [2] Loan Prediction Using Ensemble Technique, International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 3, March 2016
- [3] Exploratory data analysis https://en.wikipedia.org/wiki/Exploratory_data_analysis
- [4] Pandas Library <https://pandas.pydata.org/pandas-docs/stable/>
- [5] Mean Decrease Accuracy .
- [6] G. McLachlan, K.-A. Do, and C. Ambroise, Analyzing microarray gene expression data, vol. 422. John Wiley & Sons, 2005.
- [7] E. Ngai, Y. Hu, Y. Wong, Y. Chen, and X. Sun, The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature, Decision Support Systems, vol. 50, no. 3, pp. 559569, 2011.
- [8] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern classification. John Wiley & Sons, 2012.
- [9] I. H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.
- [10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Van- derplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research, vol. 12, pp. 28252830, 2011.

- [11] J. Li, H. Wei, and W. Hao, Weight-selected attribute bagging for credit scoring, *Mathematical Problems in Engineering*, vol. 2013, 2013.
- [12] S. Moro, P. Cortez, and P. Rita, A data-driven approach to predict the success of bank telemarketing, *Decision Support Systems*, vol. 62, pp. 2231, 2014.
- [13] I. Bose and R. K. Mahapatra, Business data mininga machine learning perspective, *Information & management*, vol. 39, no. 3,pp. 211225, 2001.
- [14] Haque, F. M. A., & Hassan, Md. M. (2024). Bank Loan Prediction Using Machine Learning Techniques. *American Journal of Industrial and Business Management*, 14(12), 1690-1711.
- [15] Bhargav, H., & Santhosh, S. (2023). An Ensemble Machine Learning Based Bank Loan Approval Predictions. *Materials Today: Proceedings*, 72(3), 746-750.
- [16] Kathe, R. P., Dapse, P. L., Panhale, S. D., & Avhad, P. P. (2021). An Approach for Prediction of Loan Approval Using Machine Learning Algorithm. *International Journal of Creative Research Thoughts (IJCRT)*, 9(6), 568-573.
- [17] Tyagi, S. (2022). Analyzing Machine Learning Models for Credit Scoring with Explainable AI and Optimizing Investment Decisions. *arXiv preprint arXiv:2209.09362*.
- [18] Innan, N., Marchisio, A., Bennai, M., & Shafique, M. (2024). LEP-QNN: Loan Eligibility Prediction Using Quantum Neural Networks. *arXiv preprint arXiv:2412.03158*.
- [19] Khare, K. (2022). Loan Prediction Project: A Comprehensive Analysis Using Machine Learning Techniques. *GitHub Repository*.
- [20] Arora, Y. (2022). Loan Prediction with Machine Learning. *GitHub Repository*.
- [21] Balaji, A. (2022). Loan Approval Prediction. *GitHub Repository*.

- [22] Sai Priya, T. K., & Lavanya Kumari, N. P. (2024). Loan Approval Prediction Using Machine Learning. *Journal of Nonlinear Analysis and Optimization*, 15(2), 1906-9685.
- [23] Haque, F. M. A., & Hassan, Md. M. (2024). Bank Loan Prediction Using Machine Learning Techniques. Scientific Research Publishing.
- [24] Kathe, R. P., Panhale, S. D., Avhad, P. P., Dapse, P. L., & Ghorpade, D. B. (2021). Prediction of Loan Approval Using Machine Learning Algorithm: A Review Paper. *International Journal of Creative Research Thoughts (IJCRT)*, 9(7), a76.
- [25] Bose, I., & Mahapatra, R. K. (2001). Business Data Mining—A Machine Learning Perspective. *Information & Management*, 39(3), 211-225.
- [26] Li, J., Wei, H., & Hao, W. (2013). Weight-Selected Attribute Bagging for Credit Scoring. *Mathematical Problems in Engineering*, 2013.
- [27] Moro, S., Cortez, P., & Rita, P. (2014). A Data-Driven Approach to Predict the Success of Bank Telemarketing. *Decision Support Systems*, 62, 22-31.
- [28] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.
- [29] Viswanatha V., Ramachandra A.C., Vishwas K. N., & Adithya G. (2023). Prediction of Loan Approval in Banks using Machine Learning Approach. *International Journal of Engineering and Management Research*, 13(4), 7–19.
- [30] Krishnaraj P., Rita S., & Jaiswal J. (2024). Comparing Machine Learning Techniques for Loan Approval Prediction. *Proceedings of the 1st International Conference on Artificial Intelligence, Communication, IoT, Data Engineering and Security (IACIDS 2023)*, 23-25 November 2023, Lavasa, Pune, India.