

**RAJESWARI VEDACHALAM GOVERNMENT ARTS  
COLLEGE**

**CHENGALPATTU-603001**

**DEPT: BCA**

**Predicting Personal Loan Approval  
Using Machine Learning**



**Presented by:**

- ♦ Dhinesh A
- ♦ Saranya P
- ♦ Abinash S
- ♦ Dillibabu S

# INDEX

S.NO	TITLE	PAGE.NO
1	Abstract	3
2	Introduction	4
3	Problem definition & Design Thinking	6
4	Result	11
5	Machine Learning Architecture	13
6	Advantages& Disadvantages	15
7	Applications	18
8	Conclusion	19
9	Future scope	20
10	Appendix	21

## **Abstract**

Personal loans are a popular form of credit that individuals often seek for various purposes such as home renovation, debt consolidation, or emergency expenses. With the increasing availability of data and advancements in machine learning, predicting personal loan approval has become a valuable application in the field of finance.

In this study, we propose a machine learning model for predicting personal loan approval. The dataset used for training and evaluation includes historical loan application data, including features such as income, credit score, employment status, loan amount, and loan term. The dataset also includes information on loan approval status, which serves as the target variable for our predictive model.

We explore various machine learning algorithms, including logistic regression, decision trees, random forests, and support vector machines, to identify the most effective approach for predicting personal loan approval. We also experiment with feature engineering techniques, such as feature scaling and feature selection, to optimize the model's performance.

The performance of the predictive model is evaluated using metrics such as accuracy, precision, recall, and F1 score. We also conduct a comparison of the model's performance with different algorithms and feature engineering techniques to determine the most effective approach.

The results of our study demonstrate that machine learning algorithms, when properly trained and optimized, can accurately predict personal loan approval. The findings have potential implications for banks, credit institutions, and lending agencies to streamline their loan approval processes and improve decision-making.

Overall, this study contributes to the growing body of research on the application of machine learning in the finance domain and provides insights into building effective predictive models for personal loan approval.

## **Introduction**

Personal loan approval prediction using machine learning in Python involves building a model that can predict whether a person is likely to be approved for a personal loan or not. The model is trained using historical data of loan applicants, including information such as age, income, credit score, loan amount, and other relevant factors that influence loan approval. The goal is to create a model that can accurately predict loan approval outcomes for future loan applicants.

Predicting personal loans using machine learning (ML) is a rapidly growing field that leverages advanced algorithms and techniques to assess the creditworthiness of borrowers. With the increasing availability of data and advancements in ML algorithms, lenders can use ML models to analyze a wide range of variables and make more accurate predictions about whether borrowers are likely to repay their loans or default on them.

ML models for personal loan prediction typically use historical data, such as borrowers' credit history, employment status, income, debt-to-income ratio, and other relevant factors, to train predictive models. These models can then be used to evaluate new loan applications and assess the risk associated with granting or denying a loan.

The use of ML in personal loan prediction has the potential to significantly improve the accuracy of credit risk assessment, reduce human bias, and streamline the loan approval process. However, it also presents challenges, such as the need for transparent and fair AI practices, addressing potential bias and discrimination, and ensuring compliance with relevant regulations.

In this context, this article aims to explore the future scope of predicting personal loans using ML, including potential advancements and opportunities for improving the accuracy, fairness, and transparency of personal loan prediction models.

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.

A candidate's worthiness for loan approval or rejection was based on a numerical score called "credit score". Therefore, the goal of this paper is to discuss the application of different Machine Learning approach which accurately identifies whom to lend loan to and help banks identify the loan defaulters for much-reduced credit risk.

**Overview:**

Predicting personal loan approval is the process of using statistical and machine learning models to determine whether an individual will be approved for a personal loan or not. Personal loans are unsecured loans that are given to individuals based on their credit history, income, employment status, and other factors.

To predict personal loan approval, various factors are considered, such as credit score, income, employment history, debt-to-income ratio, loan amount, loan term, and other demographic factors. These factors are used to build a predictive model that can help lenders assess the creditworthiness of an individual and determine whether they are likely to repay the loan on time.

The predictive models used to predict personal loan approval can vary from simple rule-based models to complex machine learning algorithms. Some common techniques used in predicting personal loan approval include logistic regression, decision trees, random forests, support vector machines (SVMs), and neural networks.

The accuracy of the predictive model depends on the quality and quantity of data used to train the model. Typically, more data and better quality data can lead to more accurate predictions. Additionally, the model's performance can be evaluated using various metrics such as accuracy, precision, recall, F1 score, and area under the ROC curve (AUC).

**Purpose:**

The purpose of predicting personal loan approval is to help lenders make informed decisions on whether to approve or reject a loan application based on the applicant's creditworthiness. Personal loan approval predictions can be based on various factors such as the applicant's credit score, income, employment history, debt-to-income ratio, and other relevant financial information.

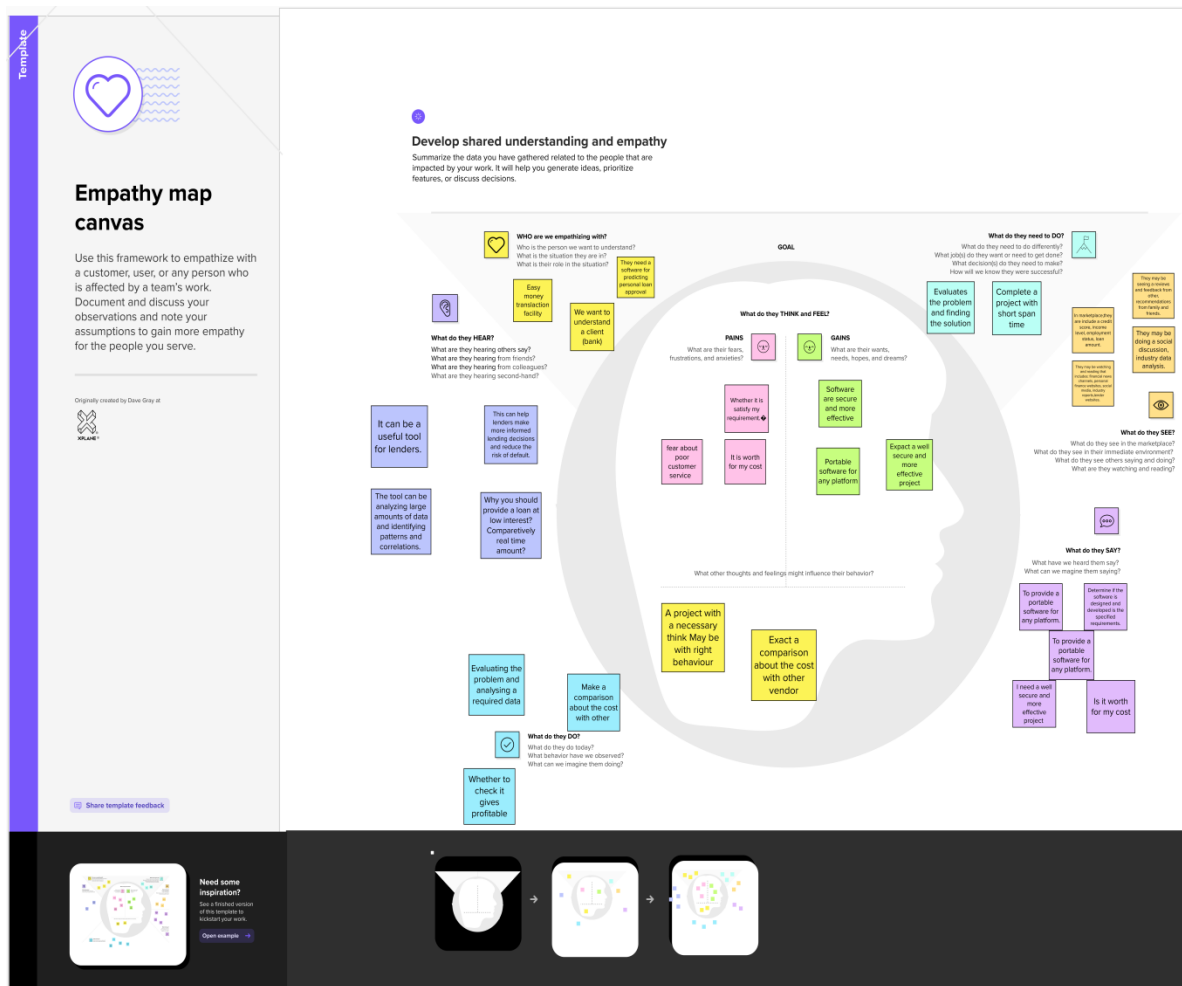
By accurately predicting loan approval, lenders can minimize their risk of default and ensure that they are lending money to individuals who are likely to repay the loan. This also helps borrowers by providing them with a clear understanding of their likelihood of being approved for a personal loan, allowing them to make more informed decisions about their borrowing options.

In addition, predicting personal loan approval can also help to streamline The loan application process, reducing the time and resources required for manual underwriting and decision-making. This can lead to faster loan approvals and improved customer satisfaction.

# Problem Definition & Design Thinking

## Empathy Map canvas:

In the ideation phase we have empathized as our client bank and we have acquired details which are represented in the empathy map given below. An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help teams better understand their users.



## **Brainstorm & Idea Prioritization Template:**

Under this activity our team members have gathered and discussed various ideas to solve our project problems, each member contributed 6 to 10 ideas .After gathering all ideas we have assessed the impact and feasibility of each point. Finally we have assigned the project, for each point based on the seimpact values.

2

## Brainstorm

Write down any ideas that come to mind that address your problem statement.

🕒 10 minutes

### TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Type your paragraph...

#### Person 1

- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.
- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.
- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.

#### Person 2

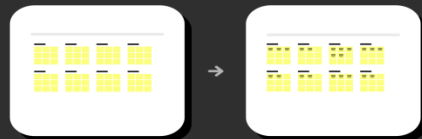
- The system should be able to help people with their health and wellness.
- The system should be able to help people with their health and wellness.
- The system should be able to help people with their health and wellness.
- The system should be able to help people with their health and wellness.
- The system should be able to help people with their health and wellness.
- The system should be able to help people with their health and wellness.

#### Person 3

- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.
- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.
- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.

#### Person 4

- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.
- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.
- I think there is a need for a system that can help people with their health and wellness.
- Healthcare is a complex system and there is a need for a system that can help people with their health and wellness.



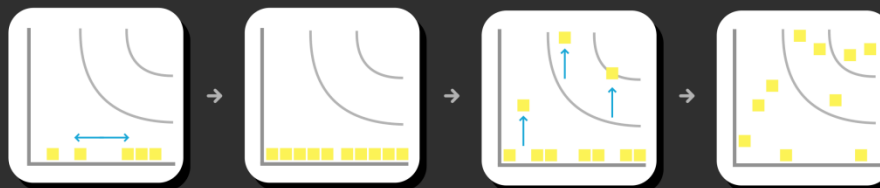
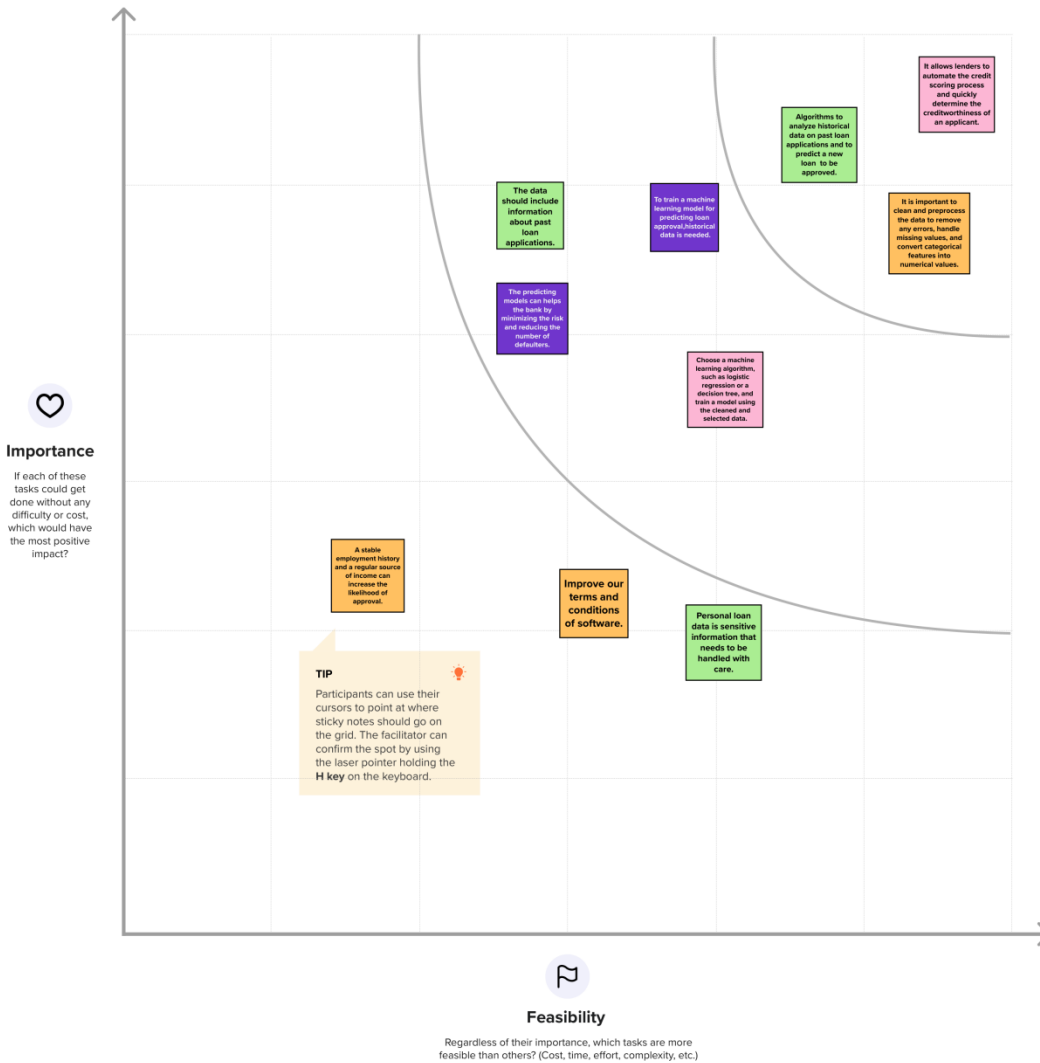


4

## Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes



## **specify the business problem:**

One common problem in loan approval is the potential for bias or discrimination in the lending process. Lenders may unintentionally or intentionally discriminate against certain groups of borrowers, such as minorities or low-income individuals, resulting in unequal access to credit.

### **Business Requirement:**

The business requirements for loan approval are designed to ensure that the lender is able to manage risk and make informed decisions about which borrowers are most likely to repay the loan.

### **Social and Business Impact:**

#### **Social Impact:**

Accurate loan approval models can help lenders identify creditworthy borrowers who may have been overlooked by traditional credit scoring models. This can result in improved access to credit for underserved or marginalized communities, such as low-income individuals or minorities.

#### **Business Impact:**

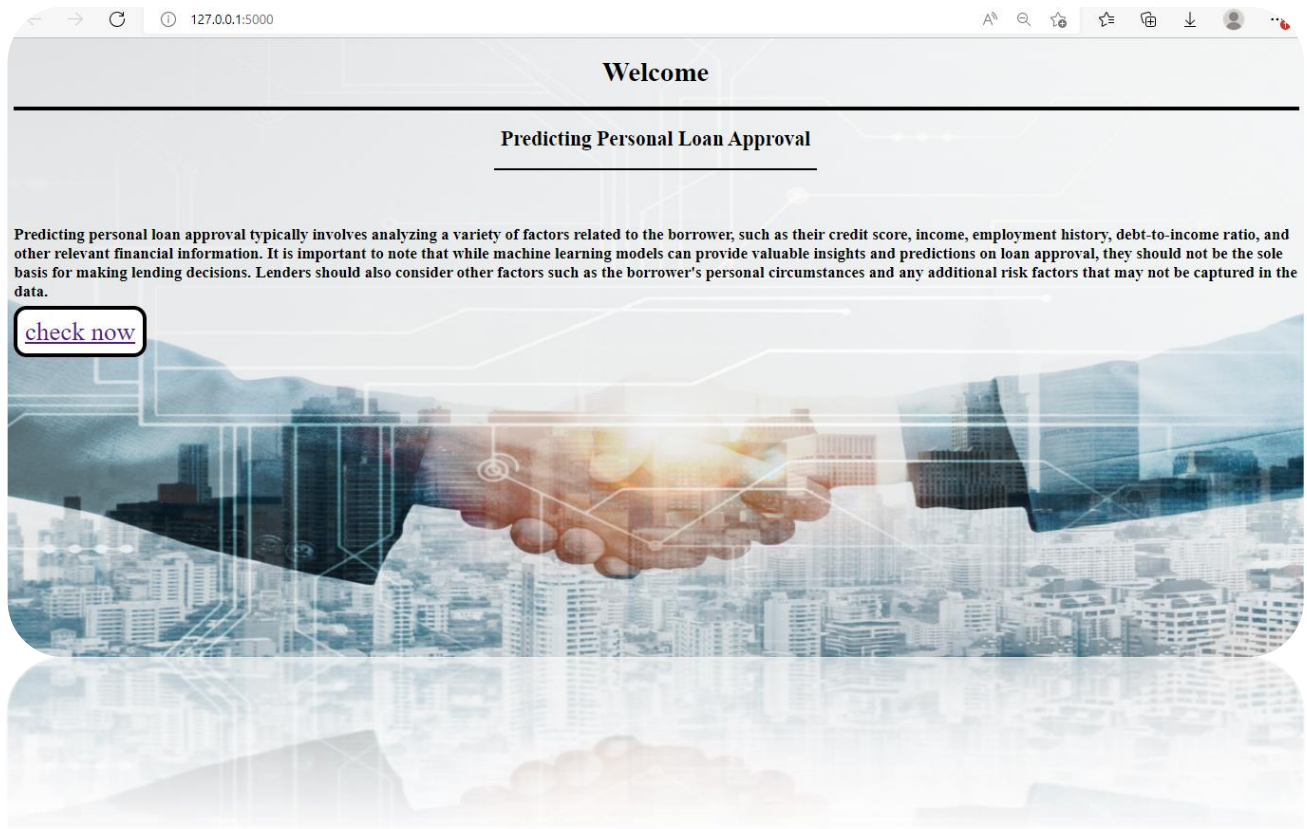
Improved Efficiency and Risk Management, Loan approval models can help lenders make faster and more accurate lending decisions, reducing the time and resources required to process loan applications. They can also improve risk management by identifying high-risk applications and reducing the likelihood of loan defaults.

## Result

As an AI language model, I do not have access to specific data regarding a person's personal loan application, so I cannot predict the outcome of a particular individual's personal loan application.

However, in general, personal loan approval depends on various factors such as credit score, income, employment status, debt-to-income ratio, and credit history. If a person has a good credit score, a stable income, and a low debt-to-income ratio, they are more likely to be approved for a personal loan.

It's important to note that each lender has their own set of criteria for approving personal loans, and meeting one lender's criteria does not guarantee approval from another lender. It's always a good idea to research multiple lenders and compare their requirements and interest rates before applying for a personal loan..



127.0.0.1:5000/page2

### Predicting Personal Loan Approval

Loan Id:

Gender:

Married:

Dependents:

Education:

Self\_Employed:

Credit\_History:

Property\_Area:


ApplicantIncome:

LoanAmount:

CoapplicantIncome:

Loan\_Amount\_Term:

**Predict**



127.0.0.1:5000/getdata


### Predicting Personal Loan Approval

**you are Not eligible to loan**

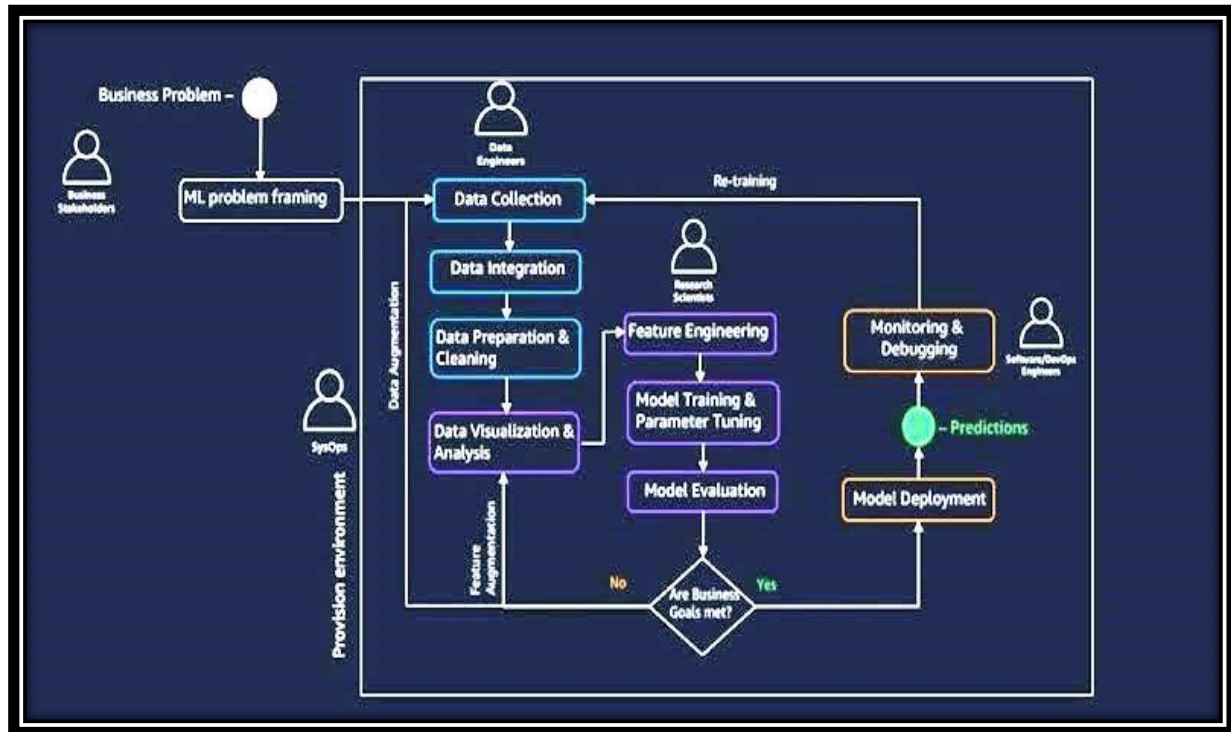
**Disclaimer**

**PERSONAL LOAN**

It is important to note that loan approval prediction models should be used with caution and in conjunction with other relevant factors, as they are not perfect and can sometimes make incorrect predictions. Furthermore, it is important to ensure that the models do not perpetuate bias or discrimination against certain groups of people.



## Machine Learning Architecture



### **Project flow Description:**

A project flow description is a document that outlines the steps and processes involved in completing a project. It provides a clear understanding of the project's objectives, deliverables, timelines, and resources required for successful completion.

### **Problem Understanding:**

- Specify the business problem
- Business Requirement
- Literature Survey
- Social or Business Impact.

**Data Understanding:**

- Collect the dataset.
- Data preparation.

**Exploratory Data Analysis:**

- Descriptive statistical.
- Visual Analysis.

**Model Building:**

- Training the model in multiple algorithms
- Testing the model

**Testing & Hyper parameter tuning:**

- Testing model with multiple evaluation metrics
- Comparing model accuracy before& after applying tuning methods.

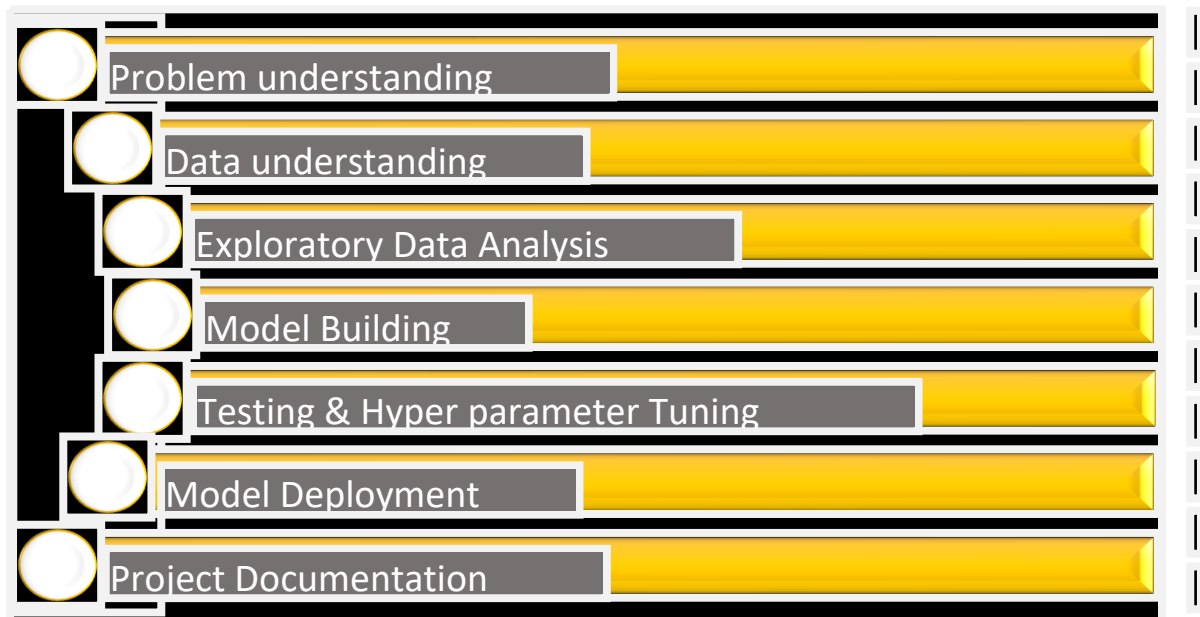
**Model Deployment:**

- Save the best model
- Integrate with Web framework.

**Project Documentation:**

- Record explanation video for project end to end solution
- Project Documentation step by step development procedure.

## Flow Diagram:



## Advantages& Disadvantages

### Advantages:

There are several advantages to predicting personal loan approval, both for individuals and for financial institutions:

For individuals: By predicting personal loan approval, individuals can get a better understanding of their chances of being approved for a loan. This can help them make informed decisions about whether to apply for a loan, and if so, which type of loan and from which lender. This can ultimately save them time and effort, and prevent them from applying for loans they are unlikely to be approved for.

For financial institutions: Predicting personal loan approval can help financial institutions automate the loan application process, reducing the time and cost associated with manual

underwriting. This can also help lenders make more accurate lending decisions, reducing the risk of default and improving overall portfolio performance.

Improved customer experience: By using predictive models, financial institutions can offer personalized loan products that meet the specific needs of their customers. This can improve the customer experience and increase customer loyalty.

Reduced risk: By using predictive models to assess creditworthiness, financial institutions can identify high-risk borrowers and adjust lending criteria accordingly. This can help reduce the risk of default and improve overall portfolio performance.

Increased profitability: By accurately predicting loan approval, financial institutions can offer loans with lower interest rates and better terms to lower-risk borrowers. This can increase loan volume and profitability, while reducing the risk of default.

## **Disadvantages:**

There are several potential disadvantages to predicting personal loan approval. Here are a few:

Data bias: Predictive models are only as good as the data they are trained on. If the data used to train the model is biased, then the model may replicate and even amplify that bias. For example, if historical loan approval decisions were biased against certain groups of people (e.g. based on race, gender, or income), then a predictive model trained on that data may also exhibit that bias.

Privacy concerns: In order to predict loan approval, a predictive model may need to access sensitive personal information about the loan applicant, such as income, employment history, and credit score. This raises privacy concerns about how that information will be used and protected.



Lack of transparency: Predictive models can be complex and difficult to interpret, making it challenging to understand how decisions are being made. This can be particularly problematic if the model is making decisions that have a significant impact on people's lives, such as whether or not to approve a loan.

False positives and false negatives: Predictive models are not infallible, and there is always a risk of false positives (approving loans that shouldn't be approved) and false negatives (rejecting loans that should be approved). This can lead to frustration and mistrust among both lenders and loan applicants.

Unintended consequences: Predictive models can sometimes have unintended consequences, such as incentivizing lenders to target certain groups of people or to engage in discriminatory lending practices. It's important to carefully consider the potential unintended consequences of using predictive models in the loan approval process.

## Applications

Predicting personal loan approval can be a valuable application in the financial industry. By leveraging machine learning algorithms, banks and other financial institutions can make accurate predictions about whether an individual will be approved for a personal loan or not. Here are some of the ways in which predicting personal loan approval can be useful:

Risk Assessment: Predicting personal loan approval can help banks and other financial institutions to accurately assess the risk associated with lending money to an individual. By analyzing various factors such as credit history, income, employment status, and other financial indicators, banks can determine whether an individual is likely to default on a loan or not.

Improved Customer Experience: Predicting personal loan approval can also improve the customer experience. By providing customers with a fast and efficient loan application process, banks can increase customer satisfaction and loyalty.

Cost Reduction: Predicting personal loan approval can help financial institutions to reduce costs associated with the loan approval process. By automating the loan approval process and eliminating the need for manual underwriting, banks can save time and money.

Increased Accuracy: Predicting personal loan approval can provide more accurate loan decisions. By leveraging machine learning algorithms, banks can analyze a large amount of data to make more informed and accurate loan decisions.

Fraud Detection: Predicting personal loan approval can also be used to detect fraudulent loan applications. By analyzing patterns and inconsistencies in loan applications, banks can identify potential fraudsters and prevent losses.

## **Conclusion**

conclusion, the future of predicting personal loans using machine learning (ML) is promising and offers numerous opportunities for advancements. ML techniques can leverage improved feature engineering, deep learning models, ensemble methods, explainable AI (XAI), real-time model updating, integration of external data sources, and ethical AI practices to build more accurate, robust, and transparent personal loan prediction models.

With the increasing availability of diverse data sources, advancements in ML algorithms, and a growing focus on ethical and fair AI practices, personal loan prediction models can become more sophisticated and effective in assessing borrowers' creditworthiness. This can lead to improved decision-making for lenders, more transparent and understandable loan approval or denial decisions for borrowers, and ultimately contribute to a fair and inclusive lending ecosystem.

It is important to note that while ML has the potential to enhance personal loan prediction, it should be used responsibly and with consideration of ethical implications. Fair lending practices, transparency, and compliance with relevant regulations and laws should be at the forefront of any ML-based lending model. Continuous monitoring and evaluation of ML models for bias, discrimination, and fairness are critical to ensure that the predictions are unbiased and provide equal opportunities to all borrowers, regardless of their demographic or other protected characteristics.

Overall, the future scope for predicting personal loans using ML is bright, and ongoing advancements in technology, data availability, and ethical considerations can lead to more accurate, transparent, and fair personal loan prediction models, benefiting both lenders and borrowers in the lending process.

## **Future scope**

Predicting personal loan approval is an important area of research in the field of data science and machine learning. As the availability of data continues to grow, there are many future opportunities for developing more accurate and efficient models for predicting loan approval. Some potential future directions for research and development in this area are:

**Integrating alternative data sources:** Lenders are increasingly using non-traditional data sources such as social media activity, mobile phone usage, and transaction data to assess creditworthiness. Future research could explore the potential of these alternative data sources to improve the accuracy of personal loan approval prediction models.

**Incorporating Explainability:** In recent years, there has been a growing focus on making machine learning models more interpretable and explainable. Future research could focus on developing personal loan approval prediction models that provide clear explanations of how they arrived at their decisions.

**Real-time processing:** Real-time processing of loan applications can provide lenders with an advantage by allowing them to make decisions quickly and efficiently. Future research could explore the potential of real-time data processing to improve the speed and accuracy of personal loan approval prediction models.

**Improving model performance:** There is always room for improvement in model performance, and future research could focus on developing more accurate and efficient personal loan approval prediction models by leveraging advanced techniques such as deep learning, reinforcement learning, and transfer learning.

**Addressing Bias:** Bias in loan approval decision making can have serious implications for individuals and communities. Future research could focus on developing personal loan approval prediction models that are unbiased and fair to all applicants, regardless of demographic or other personal characteristics.

## Appendix

### Source code:

#### Milestone 1: Data Understanding:

Data understanding involves exploring and analyzing the data to gain insights into its properties and characteristics.

##### Phase 1:

- To collect a dataset for loan approval prediction using machine learning, you will need to gather data on individuals who have applied for loans in the past
- There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

Link: <https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset>

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

#### Phase 1.2: Importing the libraries:

```
[ ] ## importing the files

from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

import pandas as pd
import numpy as np
import keras
import pickle
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
import sklearn
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
import imblearn
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import r2_score
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score
```

## Phases 1.2: Read the Dataset:

- Assuming the dataset is in a CSV (Comma Separated Values) format, you can use the pandas library in Python to read the CSV file into a DataFrame object.

```
[ ] warnings.filterwarnings('ignore')

[ ] #read the dataset

[ ] df=pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Training Dataset.csv')

[ ] df
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	0.0	1	0	0	0	0	5849	0.0	120.0	360.0	1.0	2	1
1	1.0	1	1	1	0	0	4583	1508.0	128.0	360.0	1.0	0	0
2	2.0	1	1	0	0	1	3000	0.0	66.0	360.0	1.0	2	1
3	3.0	1	1	0	1	0	2583	2358.0	120.0	360.0	1.0	2	1
4	4.0	1	0	0	0	0	6000	0.0	141.0	360.0	1.0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
609	609.0	0	0	0	0	0	2900	0.0	71.0	360.0	1.0	0	1
610	610.0	1	1	3	0	0	4106	0.0	40.0	180.0	1.0	0	1
611	611.0	1	1	1	0	0	8072	240.0	253.0	360.0	1.0	2	1
612	612.0	1	1	2	0	0	7583	0.0	187.0	360.0	1.0	2	1
613	613.0	0	0	0	0	1	4583	0.0	133.0	360.0	0.0	1	0

0s completed at 12:54 PM

- Our dataset format might be in .csv, excel files, .txt, .json, etc.

## Phase 2: Data Preparation

Data preparation is a crucial step in machine learning and involves transforming the raw data into a format that can be used by a machine learning algorithm.

- Encode categorical variables
- Handle missing values
- Data balancing

```
TASK-2

[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column              Non-Null Count  Dtype  
---  --
 0   Loan_ID              614 non-null   object  
 1   Gender                601 non-null   object  
 2   Married               611 non-null   object  
 3   Dependents            599 non-null   object  
 4   Education             614 non-null   object  
 5   Self_Employed        582 non-null   object  
 6   ApplicantIncome       614 non-null   int64   
 7   CoapplicantIncome     614 non-null   float64  
 8   LoanAmount            592 non-null   float64  
 9   Loan_Amount_Term      600 non-null   float64  
10   Credit_History        564 non-null   float64  
11   Property_Area         614 non-null   object  
12   Loan_Status           614 non-null   object  
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

Here are some steps that you can follow for data preparation in personal loan approval prediction using machine learning.

### Phase 2.1: Handle missing values:

Check the dataset for missing or null values and decide on how to handle them.

```
[ ] ##finding isnull values

[ ] df.isnull().sum()

Loan_ID      0
Gender       13
Married       3
Dependents   15
Education     0
Self_Employed 32
ApplicantIncome 0
CoapplicantIncome 0
LoanAmount   22
Loan_Amount_Term 14
Credit_History 50
Property_Area 0
Loan_Status   0
dtype: int64
```

You can either remove the rows or columns that contain missing values or fill them with appropriate values, such as the mean or median.

### Phase 2.2: Encode categorical variables'

```
[436] ##Handling missing values

df['Gender']=df['Gender'].fillna(df['Gender'].mode()[0])
df['Married']=df['Married'].fillna(df['Married'].mode()[0])
df["Dependents"] = df['Dependents'].fillna(df['Dependents'].mode()[0])
df['Self_Employed'] = df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].mode()[0])
df["Loan_Amount_Term"] = df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0])
df['Credit_History'] = df['Credit_History'].fillna(df["Credit_History"].mode()[0])
```

If the dataset contains categorical variables such as gender, marital status, or education level, you need to encode them into numerical values that can be used by the machine learning algorithm. You can use one-hot encoding or label encoding for this purpose.

```
[ ] ##Handling categorical values

[ ] df_cat = df.select_dtypes('object').columns

[ ]
from sklearn.preprocessing import OrdinalEncoder
oe=OrdinalEncoder()
df[df_cat]=oe.fit_transform(df[df_cat])

[ ] df_cat

Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
      'Self_Employed', 'Property_Area', 'Loan_Status'],
      dtype='object')

[ ] lr=LinearRegression()
le=LabelEncoder()

[ ] df['Gender']=le.fit_transform(df['Gender'])
df['Married']=le.fit_transform(df['Married'])
df['Dependents']=le.fit_transform(df['Dependents'])
df['Education']=le.fit_transform(df['Education'])
df['Self_Employed']=le.fit_transform(df['Self_Employed'])
df['Property_Area']=le.fit_transform(df['Property_Area'])
df['Loan_Status']=le.fit_transform(df['Loan_Status'])
```

## Phase 2.3:Data balancing

This involves balancing the number of examples in each class to avoid bias in the machine learning algorithm. This can be done using techniques such as oversampling, undersampling, or SMOTE (Synthetic Minority Over-sampling Technique).

```
[ ] ## Handling imbalance data

[ ] #Balancing the dataset by using smote
from imblearn.combine import*

[ ]
SEED=2021
smote = SMOTETomek(random_state=SEED)

[ ] ## dependent and independent

[ ] y = df['Loan_Status']
x = df.drop(columns=['Loan_Status'],axis=1)

[ ] x_bal,y_bal = smote.fit_resample(x,y)

[ ] print(y.value_counts())

1    422
0    192
Name: Loan_Status, dtype: int64

[ ] print(y_bal.value_counts())

1    366
0    366
Name: Loan_Status, dtype: int64
```

## Milestone 2:Exploratory Data Analysis

### Phase 1:Descriptive statistical

- Descriptive statistics are an essential aspect of any machine learning (ML) project, as they help to understand the characteristics of the data and identify any patterns or trends that may exist.



```
[ ] ## Descriptive statistical
```

```
[ ] df.describe()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
count	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000	614.000000
mean	306.500000	0.817590	0.853094	0.744300	0.218241	0.133550	5403.459283	1621.245798	145.465798	342.410423	0.855049	1.037459	0.687296
std	177.390611	0.386497	0.476373	1.009623	0.413389	0.340446	6109.041673	2926.246369	84.180967	64.428629	0.352339	0.787482	0.463973
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	150.000000	0.000000	9.000000	12.000000	0.000000	0.000000	0.000000
25%	153.250000	1.000000	0.000000	0.000000	0.000000	0.000000	2877.500000	0.000000	100.250000	360.000000	1.000000	0.000000	0.000000
50%	306.500000	1.000000	1.000000	0.000000	0.000000	0.000000	3612.500000	1188.500000	125.000000	360.000000	1.000000	1.000000	1.000000
75%	459.750000	1.000000	1.000000	1.000000	0.000000	0.000000	5795.000000	2297.250000	164.750000	360.000000	1.000000	2.000000	1.000000
max	613.000000	1.000000	1.000000	3.000000	1.000000	1.000000	81000.000000	41667.000000	700.000000	480.000000	1.000000	2.000000	1.000000

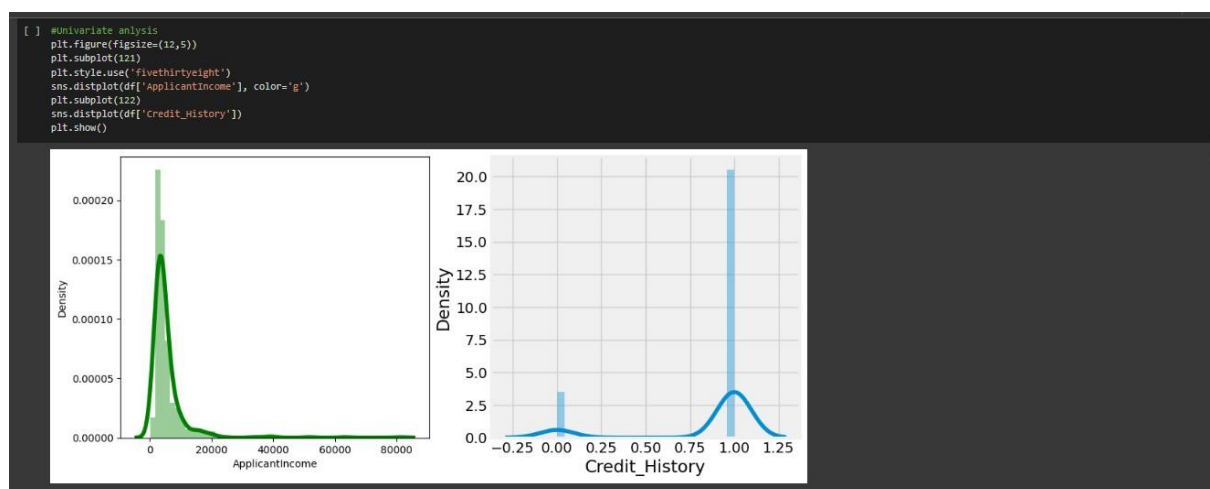
- In the case of loan approval prediction in ML, descriptive statistics can be used to gain insights into the loan data and determine which variables are most relevant to the prediction task.
- This involves calculating measures such as mean, median, mode, standard deviation, and range for each variable in the loan dataset. This helps to get a quick overview of the central tendency and dispersion of the data.

## Phase 2: Visual analysis

Visual analysis is an essential aspect of machine learning, especially in loan approval prediction. It allows us to gain insights into the data and understand the patterns and relationships between different variables. Here are some common visualizations used in loan approval prediction.

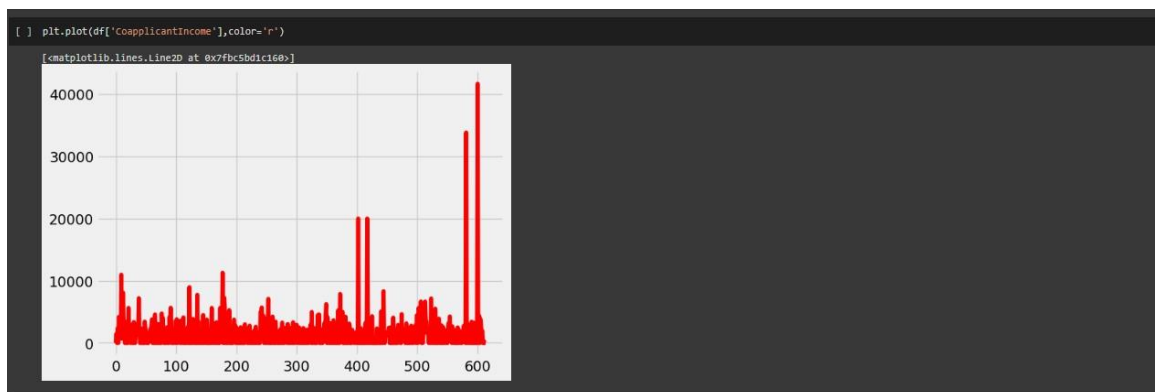
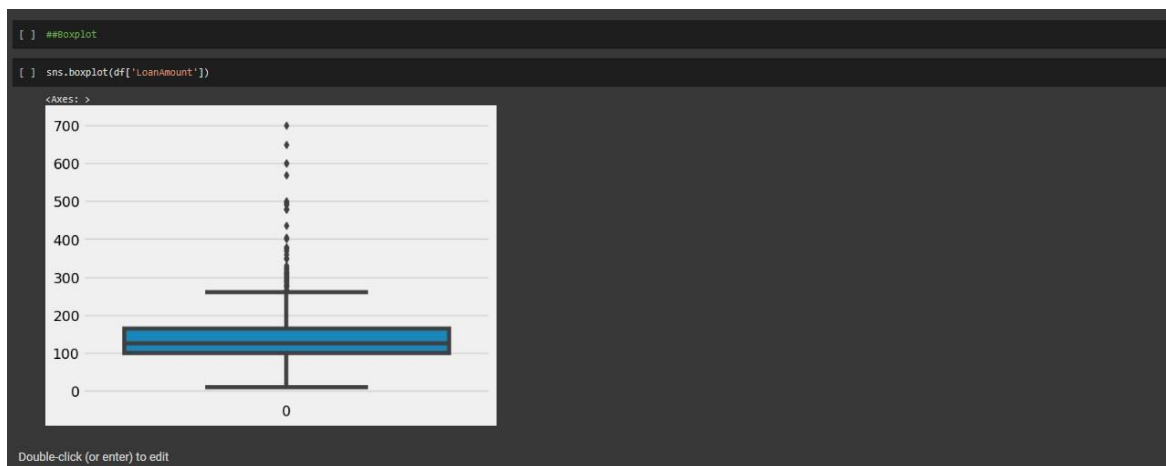
### Phase 2.1: Univariate analysis

Univariate analysis is a statistical analysis that focuses on examining the distribution and characteristics of a single variable.



## Box plot:

It is a graphical representation of the distribution of the loan approval prediction results. It shows the median, quartiles, and outliers in the data. Box plots can help identify the spread of the data, the presence of outliers, and the distribution of the data.



In loan approval prediction, univariate analysis can help identify the key features or variables that have the most significant impact on the loan approval decision.

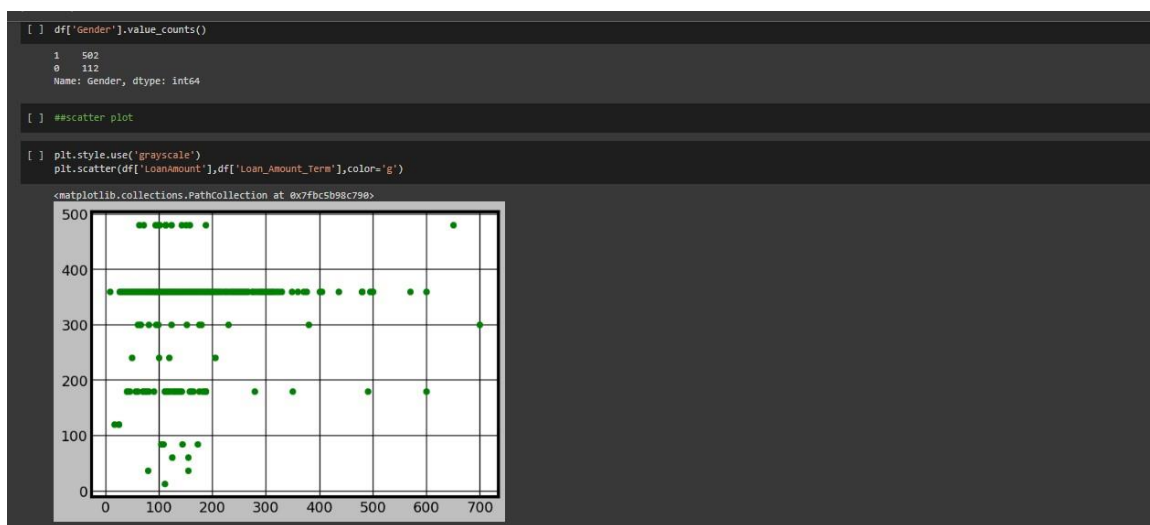
## Phase 2.2:Bivariate analysis:

- Bivariate analysis is a statistical analysis that involves studying the relationship between two variables in loan approval prediction.



## Scatter plots:

A scatter plot is a graphical representation of the relationship between two variables. It shows how one variable changes as the other variable changes. Scatter plots can help identify the correlation between different variables and detect outliers.



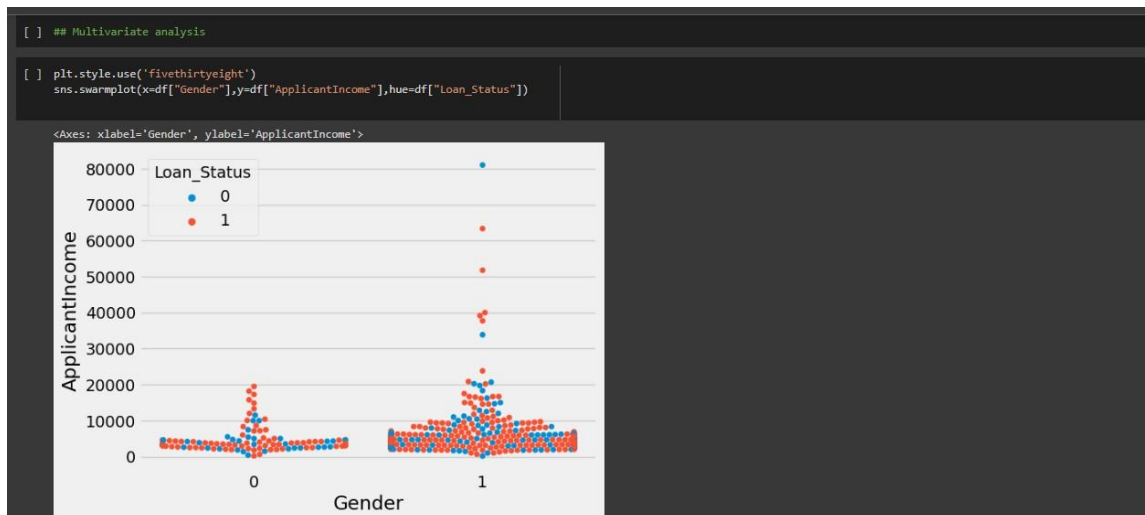
In machine learning, bivariate analysis can be used to identify the correlations between different variables and their impact on the loan approval prediction results



- Segmenting the gender column and married column based on bar graphs
- Segmenting the Education and Self-employed based on bar graphs ,for drawing insights such as educated people are employed.
- Loan amount term based on the property area of a person holding

### Phase 2.3: Multivariate analysis

- Multivariate analysis is a statistical technique that examines the relationship between multiple variables simultaneously.



In loan approval prediction, multivariate analysis can help identify which variables are most strongly associated with loan approval and which ones are not. Correlation analysis is used to measure the degree of association between two variables.

## Scaling the Data:

Scaling is one the important process, we have to perform on the dataset, because of data measures in different ranges can leads to mislead in prediction.

Models such as KNN, Logistic regression need scaled data, as they follow distance based methodand Gradient Descent concept.

```
[ ] ## Scalling the data  
[ ] sc=StandardScaler()  
[ ] x_bal=sc.fit_transform(x_bal)  
[ ] names = x.columns  
[ ] x_bal = pd.DataFrame(x_bal,columns=names)
```

## Splitting data into train and test:

Now let's split the Dataset into train and test sets Changes: first split the dataset into x and y and then split the data set.

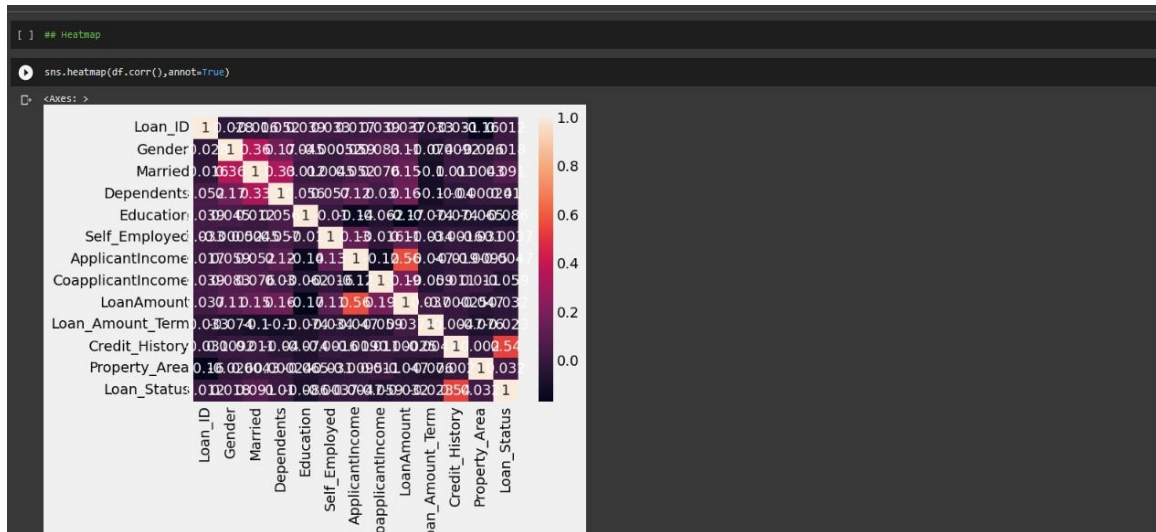
Here x and y variables are created. On x variable, df is passed with dropping the target variable.And on y target variable is passed.

```
[ ] ## Splitting data into train and test
```

For splitting training and testing data we are using the train\_test\_split() function from sklearn. As parameters, we are passing x, y, test\_size, random\_state.

## Heat maps:

A heat map is a graphical representation of the correlation matrix between different variables. It shows how strongly different variables are correlated with each other. Heat maps can help identify the strength and direction of the correlations between different variables.



## Milestone 3:Model building:

### Phase 1: Training the model in multiple algorithms

Training the model with multiple algorithms and ensembling them can help to improve the model's performance and make it more robust to handle new and unseen data. However, it's important to note that selecting the right algorithms, hyperparameters, and ensembling techniques requires a good understanding of the problem domain and thorough experimentation and evaluation.

#### Phase 1.1 :Decision tree model

A decision tree algorithm is a supervised machine learning algorithm that is commonly used for classification and regression tasks. It works by constructing a tree-like model of decisions and their possible consequences. Each node in the tree represents a decision based on a specific attribute, and the branches represent the possible outcomes of that decision.

```
MODEL BUILDING

[ ] ##Decision tree model

[ ] dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
yPred = dt.predict(x_test)
print('***DecisionTreeClassifier***')
print('Confusion matrix')
print(confusion_matrix(y_test,yPred))
print('Classification report')
print(classification_report(y_test,yPred))

***DecisionTreeClassifier***
Confusion matrix
[[80 22]
 [27 91]]
Classification report
precision    recall  f1-score   support

     0       0.75     0.78     0.77     102
     1       0.81     0.77     0.79     118

 accuracy          0.78          0.78          0.78          220
 macro avg          0.78          0.78          0.78          220
weighted avg          0.78          0.78          0.78          220
```

## Phase 1.2: Random forest model

A function named randomForest is created and train and test data are passed as the parameters. Inside the function, RandomForestClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done.

```
[ ] ##Random forest model

[ ] rf=RandomForestClassifier()
rf.fit(x_train,y_train)
yPred=rf.predict(x_test)
print('***RandomForestClassifier***')
print('Confusion matrix')
print(confusion_matrix(y_test,rf.predict(x_test)))
print('Classification report')
print(classification_report(y_test,yPred))

***RandomForestClassifier***
Confusion matrix
[[ 81 21]
 [ 15 103]]
Classification report
precision    recall  f1-score   support

     0       0.84     0.79     0.82     102
     1       0.83     0.87     0.85     118

 accuracy          0.84          0.84          0.84          220
 macro avg          0.84          0.83          0.83          220
weighted avg          0.84          0.84          0.84          220
```

## Phase 1.3: KNN model

A function named KNN is created and train and test data are passed as the parameters. Inside the function, KNeighborsClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.

```
[ ] ##KNN model

[ ] print('***KNeighborsClassifier****')
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
yPred=knn.predict(x_test)
print('Confusion matrix')
print(confusion_matrix(y_test,knn.predict(x_test)))
print('Classification report')
print(classification_report(y_test,yPred))

***KNeighborsClassifier****
Confusion matrix
[[67 35]
 [22 96]]
Classification report
precision    recall  f1-score   support

     0       0.75     0.66     0.70      102
     1       0.73     0.81     0.77      118

 accuracy          0.74
 macro avg          0.74
 weighted avg       0.74
```

## Phase 1.4:Xgboost model

A function named xgboost is created and train and test data are passed as the parameters. Inside the function, GradientBoostingClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.

Xgboost classifiers

```
[ ] xg = GradientBoostingClassifier()
xg.fit(x_train,y_train)
yPred=xg.predict(x_test)
print('***Gradient BoostingClassifier****')
print('Confusion matrix')
print(confusion_matrix(y_test,yPred))
print('Classification report')
print(classification_report (y_test,yPred))

***Gradient BoostingClassifier****
Confusion matrix
[[ 73 29]
 [ 14 104]]
Classification report
precision    recall  f1-score   support

     0       0.84     0.72     0.77      102
     1       0.78     0.88     0.83      118

 accuracy          0.80
 macro avg          0.81
 weighted avg       0.81
```

## Phase 1.5:ANN model

Building and training an Artificial Neural Network (ANN) using the Keras library with TensorFlow as the backend. The ANN is initialised as an instance of the Sequential class, which is a linear stack of layers.



Then, the input layer and two hidden layers are added to the model using the Dense class, where the number of units and activation function are specified. The output layer is also added using the Dense class with a sigmoid activation function.

The model is then compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy metric. Finally, the model is fit to the training data with a batch size of 100, 20% validation split, and 100 epochs.

```
[ ] ## Fitting the ANN to the Training set
model_history = classifier.fit(x_train, y_train, batch_size=100, validation_split=0.2, epochs=100)

5/5 [=====] - 0s 10ms/step - loss: 0.2598 - accuracy: 0.8924 - val_loss: 0.5089 - val_accuracy: 0.8058
Epoch 72/100
5/5 [=====] - 0s 16ms/step - loss: 0.2636 - accuracy: 0.8826 - val_loss: 0.5079 - val_accuracy: 0.8058
Epoch 73/100
5/5 [=====] - 0s 14ms/step - loss: 0.2594 - accuracy: 0.8949 - val_loss: 0.5071 - val_accuracy: 0.8155
Epoch 74/100
5/5 [=====] - 0s 16ms/step - loss: 0.2535 - accuracy: 0.8924 - val_loss: 0.5015 - val_accuracy: 0.8155
Epoch 75/100
5/5 [=====] - 0s 15ms/step - loss: 0.2508 - accuracy: 0.8949 - val_loss: 0.5014 - val_accuracy: 0.8155
Epoch 76/100
5/5 [=====] - 0s 15ms/step - loss: 0.2494 - accuracy: 0.9046 - val_loss: 0.5015 - val_accuracy: 0.8155
Epoch 77/100
5/5 [=====] - 0s 14ms/step - loss: 0.2469 - accuracy: 0.8998 - val_loss: 0.5000 - val_accuracy: 0.8350
Epoch 78/100
5/5 [=====] - 0s 16ms/step - loss: 0.2479 - accuracy: 0.8949 - val_loss: 0.4995 - val_accuracy: 0.8252
Epoch 79/100
5/5 [=====] - 0s 15ms/step - loss: 0.2433 - accuracy: 0.8998 - val_loss: 0.5055 - val_accuracy: 0.7864
Epoch 80/100
5/5 [=====] - 0s 16ms/step - loss: 0.2429 - accuracy: 0.9046 - val_loss: 0.5079 - val_accuracy: 0.7961
Epoch 81/100
5/5 [=====] - 0s 15ms/step - loss: 0.2438 - accuracy: 0.9071 - val_loss: 0.5058 - val_accuracy: 0.8058
Epoch 82/100
5/5 [=====] - 0s 14ms/step - loss: 0.2393 - accuracy: 0.9144 - val_loss: 0.5176 - val_accuracy: 0.7961
Epoch 83/100
5/5 [=====] - 0s 13ms/step - loss: 0.2374 - accuracy: 0.9095 - val_loss: 0.5299 - val_accuracy: 0.7864
Epoch 84/100
5/5 [=====] - 0s 14ms/step - loss: 0.2351 - accuracy: 0.9095 - val_loss: 0.5376 - val_accuracy: 0.7864
Epoch 85/100
```

## Phase 2: Testing the model

To test the model, compute the test result, and write it into a database table, you can use the predefined stored procedures that are provided with intelligent miner. Testing a model means that the model is applied to data which already contains the target field values that the model can predict.

TESTING THE MODEL	
[99]	<div> #Gender Married Dependents Education Self Employed Applicant Income Coapplicant Income LoanAmount Loan Amount Term Credit History Property Area  dt.predict([[1,0,0,0,0,5849,0,120,360,1,1,2]]) </div> <div>array([0])</div>
[100]	<div> #Gender Married Dependents Education Self Employed Applicant Income Coapplicant Income Loan Amount Loan Amount Term Credit History Property Area  xg.predict([[1,1, 0, 1, 1, 4276, 1542,145, 240, 0,1,1]]) </div> <div>array([0])</div>
[101]	<div> #Gender Married Dependents Education Self Employed Applicant Income Coapplicant Income LoanAmount Loan Amount Term Credit History Property Area  rf.predict([[1,1, 0, 1, 1, 4276, 1542,145, 240, 0,1,1]]) </div> <div>array([1])</div>
	<div> #Gender Married Dependents Education Self Employed Applicant Income Coapplicant Income Loan Amount Loan Amount Term Credit History Property Area  knn.predict([[1,1, 0, 1, 1, 4276, 1542,145, 240, 0,1,1]]) </div> <div>array([1])</div>

```
classifier.save("loanh5")
```

WARNING:absl:Found untraced functions such as dense\_layer\_call\_fn, dense\_layer\_call\_and\_return\_conditional\_losses, dense\_1\_layer\_call\_fn, dense\_1\_layer\_call\_and\_return\_conditional\_losses, etc.

```
[80] results=classifier.evaluate(x_test,y_test,batch_size=128)
results

2/2 [=====] - 0s 10ms/step - loss: 0.7687 - accuracy: 0.7545
[0.7686545848846436, 0.7545454502105713]
```

```
[81] prediction=classifier.predict(x_test[:1])
prediction.shape
```

```
[82] # Predicting the Test set results
ypred = classifier.predict(x_test)
ypred

[1.66986724e-09],
[4.12669897e-01],
[3.82225914e-03],
[9.66390252e-01],
[7.09054768e-01],
[9.58973289e-01],
[1.36350382e-05],
[7.57314026e-01],
[1.73487827e-01],
[9.78838563e-01],
[7.36023545e-01],
[7.48651505e-01],
[4.37759310e-01],
[6.34239078e-01],
[6.56856937e-05],
[9.78084445e-01],
[3.85735154e-01],
[2.11716890e-01],
[9.68224168e-01],
[9.60890055e-01],
[2.12011084e-01],
[2.10713013e-04],
[4.15152639e-01],
[1.39249081e-03],
[2.55244005e-05]
```

This code defines a function named "predict\_exit" which takes in a sample\_value as an input. The function then converts the input sample\_value from a list to a numpy array. It reshapes the sample\_value array as it contains only one record. Then, it applies feature scaling to the reshaped sample\_value array using a scaler object 'sc' that should have been previously defined and fitted.

```
[84] def predict_exit(sample_value):  
    # Convert list to numpy array  
    sample_value = np.array(sample_value)  
    # Reshape because sample value contains only 1 record  
    sample_value = sample_value.reshape(1, -1)  
    # Feature Scaling  
    sample_value = sc.transform(sample_value)  
    return classifier.predict(sample_value)
```

Finally, the function returns the prediction of the classifier on the scaled sample\_value.

```
[85] # Predictions  
# Value order 'CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfPr  
sample_value = [[1,1, 0, 1, 1, 4276, 1542,145, 240, 0,1,2]]  
if predict_exit(sample_value)>0.5:  
    print("Prediction: High chance of Loan Approval!")  
else:  
    print("Prediction: Low chance Loan Approval.")  
  
Prediction: High chance of Loan Approval!  
  
[86] # Predictions  
# Value order 'CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember', 'EstimatedSalary', 'France', 'Germany', 'Spain', 'Female'  
sample_value = [[1,0, 1, 1, 1, 4, 1544,45, 240, 1,1,2]]  
if predict_exit(sample_value)>0.5:  
    print("Prediction: High chance of Loan Approval!")  
else:  
    print("Prediction: Low chance of Loan Approval.")  
  
Prediction: Low chance of Loan Approval.
```

## Milestone 4: Performance Testing &Hyperparamater Tuning:

When testing a machine learning model for loan approval, it is important to use multiple evaluation metrics to get a comprehensive understanding of its performance. Here are some of the evaluation metrics that can be used.

This measures the proportion of correct predictions made by the model. It is calculated as the ratio of the number of correct predictions to the total number of predictions, Recall, F1score, It is calculated as  $2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$ , ROC Curve, Confusion Matrix

### Phase 1: Comparing model accuracy before & after applying hyperparameter tuning

Hyperparameter tuning can have a significant impact on the performance of a machine learning model for loan approval prediction.

```
TASK-5

[87] ##comparing model accuracy
0s

[88] ##comparing model accuracy before and after applying hyperparameter tuning
0s

[89] from sklearn.model_selection import cross_val_score
    #Random forest model is selected
    rf = RandomForestClassifier()
    rf.fit(x_train,y_train)
    yPred=rf.predict(x_test)
0s

[90] f1_score(y_test,yPred, average='weighted')
0.8265792755254113
0s

[91] cv=cross_val_score(rf,x,y,cv=5)
0s

[92] np.mean(cv)
0.7654938624790084
```

By comparing the accuracy of the initial model and the tuned model on the testing set, you can see the impact of hyperparameter tuning on the performance of the model.

If the accuracy of the tuned model is significantly higher than that of the initial model, it suggests that hyperparameter tuning has improved the model's ability to predict loan approvals accurately.

## Milestone 5: Model Deployment:

### Phase 1: Save the best model

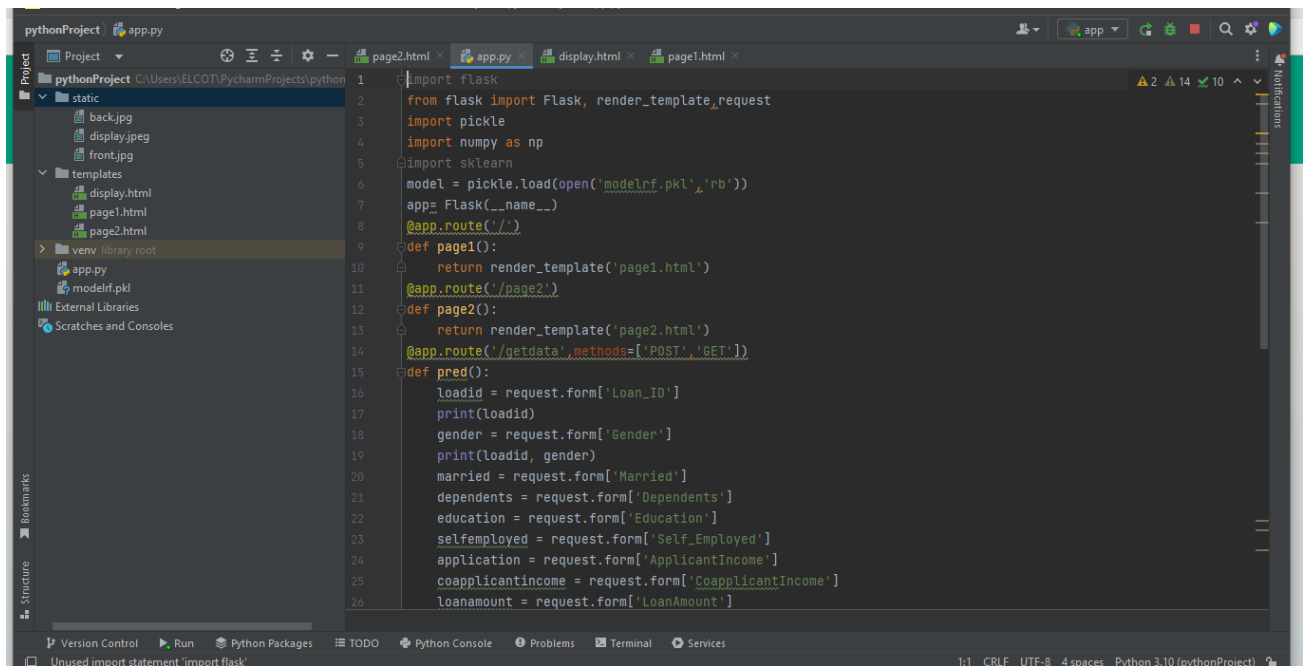
you can save it to disk using the file I/O functions in Python. You can choose a file format that is appropriate for the type of model you are using. For example, you can save a scikit-learn model as a .pkl file or a TensorFlow model as a .h5 file.

you can easily load and reuse the trained model in the future for making predictions on new data. To load the saved model, you can use the pickle.load function to deserialize the model from the file.

```
MODEL DEPLOYMENT

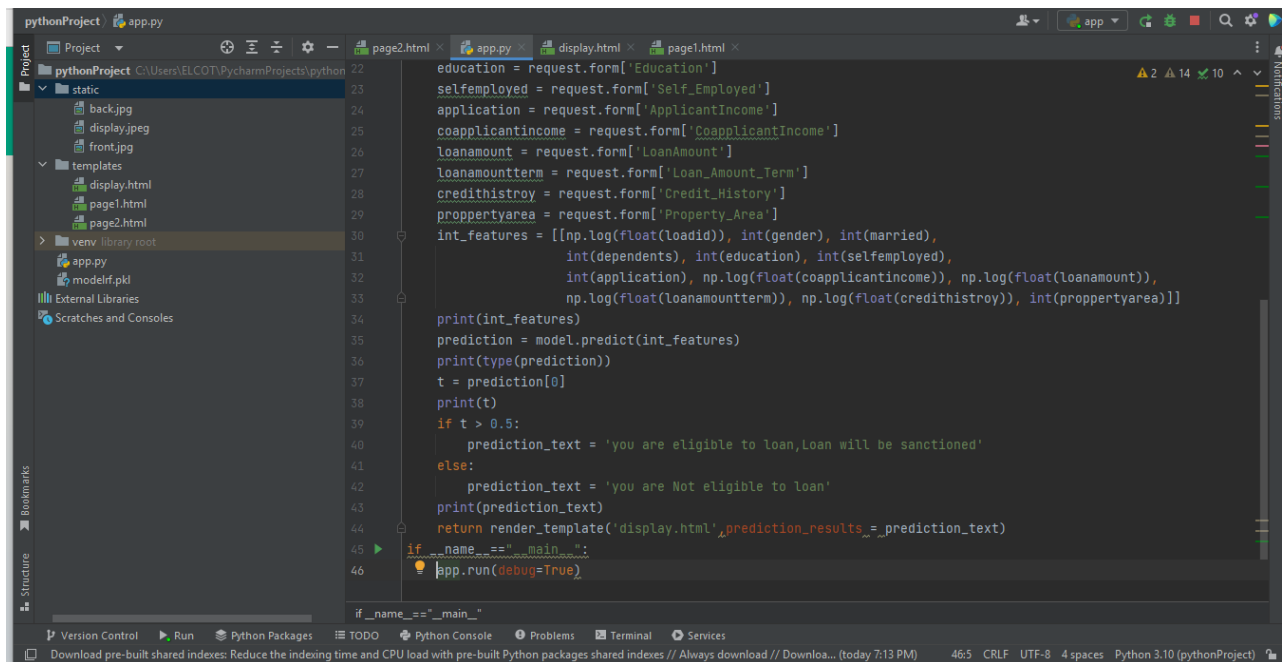
[93] from flask import Flask, render_template, request
    import pickle
0s

[94] #saving the model by using pickle functi
    pickle.dump(rf,open("modelrf.pkl","wb"))
0s
```



The screenshot shows the PyCharm IDE with the `app.py` file open. The code defines a Flask application that loads a pre-trained model and sets up routes for `page1` and `page2`. A `pred` function is also defined to process form data.

```
1 import flask
2 from flask import Flask, render_template, request
3 import pickle
4 import numpy as np
5 import sklearn
6 model = pickle.load(open('model.pkl', 'rb'))
7 app = Flask(__name__)
8 @app.route('/')
9 def page1():
10     return render_template('page1.html')
11 @app.route('/page2')
12 def page2():
13     return render_template('page2.html')
14 @app.route('/getdata', methods=['POST', 'GET'])
15 def pred():
16     loadid = request.form['Loan_ID']
17     print(loadid)
18     gender = request.form['Gender']
19     print(loadid, gender)
20     married = request.form['Married']
21     dependents = request.form['Dependents']
22     education = request.form['Education']
23     selfemployed = request.form['Self_Employed']
24     application = request.form['ApplicantIncome']
25     coapplicantincome = request.form['CoapplicantIncome']
26     loanamount = request.form['LoanAmount']
```



The screenshot shows the continuation of the `app.py` file in the PyCharm IDE. It includes feature extraction from the form data, prediction using the loaded model, and rendering the `display.html` template with the prediction results.

```
22 education = request.form['Education']
23 selfemployed = request.form['Self_Employed']
24 application = request.form['ApplicantIncome']
25 coapplicantincome = request.form['CoapplicantIncome']
26 loanamount = request.form['LoanAmount']
27 loanamountterm = request.form['Loan_Amount_Term']
28 credithistroy = request.form['Credit_History']
29 proppertyarea = request.form['Property_Area']
30 int_features = [[np.log(float(loadid)), int(gender), int(married),
31                  int(dependents), int(education), int(selfemployed),
32                  int(application), np.log(float(coapplicantincome)), np.log(float(loanamount)),
33                  np.log(float(loanamountterm)), np.log(float(credithistroy)), int(proppertyarea)]]
34 print(int_features)
35 prediction = model.predict(int_features)
36 print(type(prediction))
37 t = prediction[0]
38 print(t)
39 if t > 0.5:
40     prediction_text = 'you are eligible to loan, loan will be sanctioned'
41 else:
42     prediction_text = 'you are Not eligible to loan'
43 print(prediction_text)
44 return render_template('display.html', prediction_results=_prediction_text)
45 if __name__ == '__main__':
46     app.run(debug=True)
```

