LOAN APPROVAL PREDICTION

S.NO	**Name of the Student**	**NM ID**
TEAM LEADER	S.Tamilarasi	3E7F2144219517ABD504A81C7D7EF88F
TEAM MEMBER	P.Babykala	A5DD2297BCAD6A5A20BD259D1DD40
	M.Mala	AB766288D97687497014648DAB59868A
	M.Nagammal	D07E40B8C4441CC87744628F46698D15
	R.Maheswari	5351B30DA06F7D4B1184B288317188A2

GOVERNMENT ARTS AND COLLEGE ,

KADAYANALLUR.

INDEX

1.INTRODUCTION

1.1 OVERVIEW

Loan approval prediction refers to the process of using machine learning algorithms to predict the likelihood of a loan being approved or rejected. This involves analyzing a variety of factors related to the borrower, such as their credit history, income, employment status, and other relevant information.

The goal of loan approval prediction is to help lenders make informed decisions about whether to approve or deny loan applications. By using advanced analytics techniques, lenders can improve their accuracy in predicting which borrowers are likely to default on their loans, and adjust their lending criteria accordingly.

Loan approval prediction can be applied in various industries, including banking, finance, and insurance. It has become increasingly important in recent years as more and more people are applying for loans and lenders need to efficiently manage their risk exposure.

1.2 PURPOSE

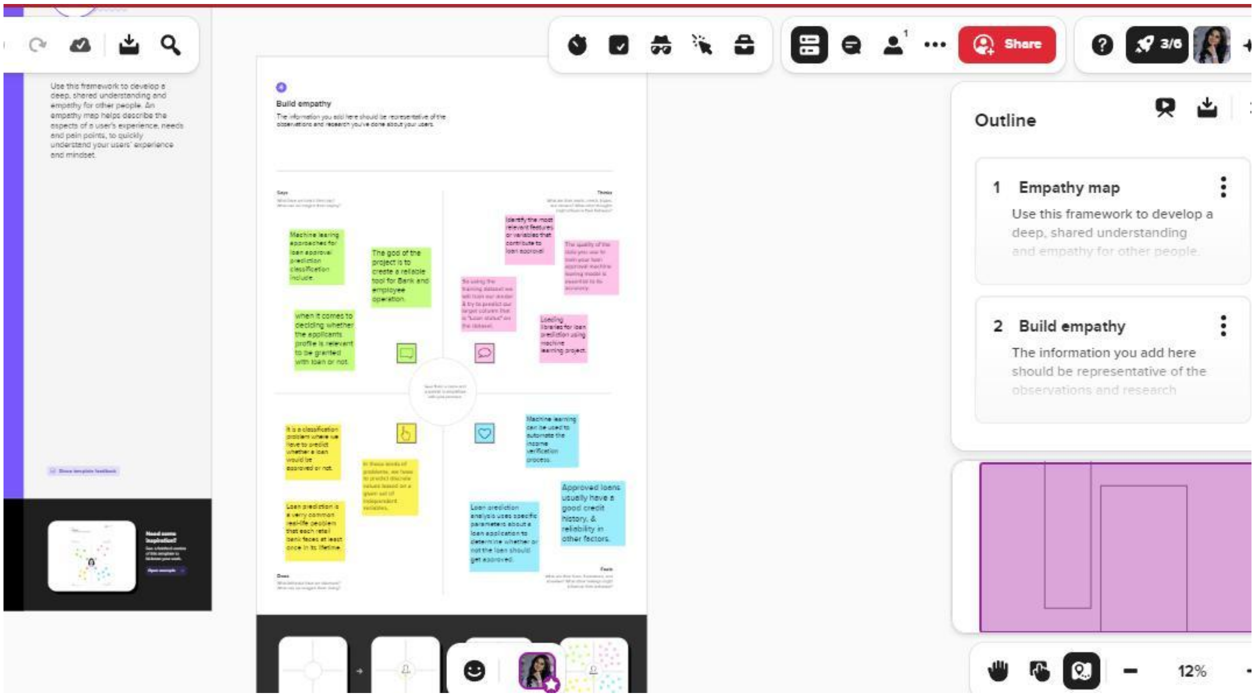
If you're looking to predict loan approvals, you can use machine learning algorithms to analyze data related to past loan applications and outcomes to create a model that can predict the likelihood of loan approval for a new application.

To do this, you would need a dataset that includes information about past loan applications, including features such as credit scores, income, employment status, loan amounts, and other relevant information. You can use this data to train a machine learning algorithm, such as logistic regression or a decision tree, to identify patterns in the data that are associated with loan approval or rejection.

Once you have trained your model, you can use it to predict the likelihood of loan approval for new loan applications. This can be useful for lenders who want to automate their loan approval process and reduce the time and cost associated with manual review of loan applications. It can also be useful for borrowers who want to get a sense of their likelihood of approval before applying for a loan.

2.PROBLEM DEFINITION &DESIGN THINKING

EMPATHY MAP



Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says

Machine learing approaches for loan approval prediction classification include.

The goal of the project is to create a reliable tool for Bank and employee operation.

when it comes to deciding whether the applicants profile is relevent to be granted with loan or not.

Thinks

Identify the most relevant features or variables that contribute to loan approval.

The quality of the data you use to train your loan approval machine learning model is essential to its accuracy.

So using the training dataset we will train our model & try to predict our target column that is "Loan status" on the dataset.

Loading libraries for loan prediction using machine learning project.

Does

It is a classification problem where we have to predict whether a loan would be approved or not.

In these kinds of problems, we have to predict discrete values based on a given set of independent variables.

Loan prediction is a verry common real-life peoblem that each retail bank faces at least once in its lifetime.

Feels

Machine learning can be used to automate the income verification process.

Loan prediction analysis uses specific parameters about a loan application to determine whether or not the loan should get approved.

Approved loans usually have a good credit history. & reliability in other factors.

Outline

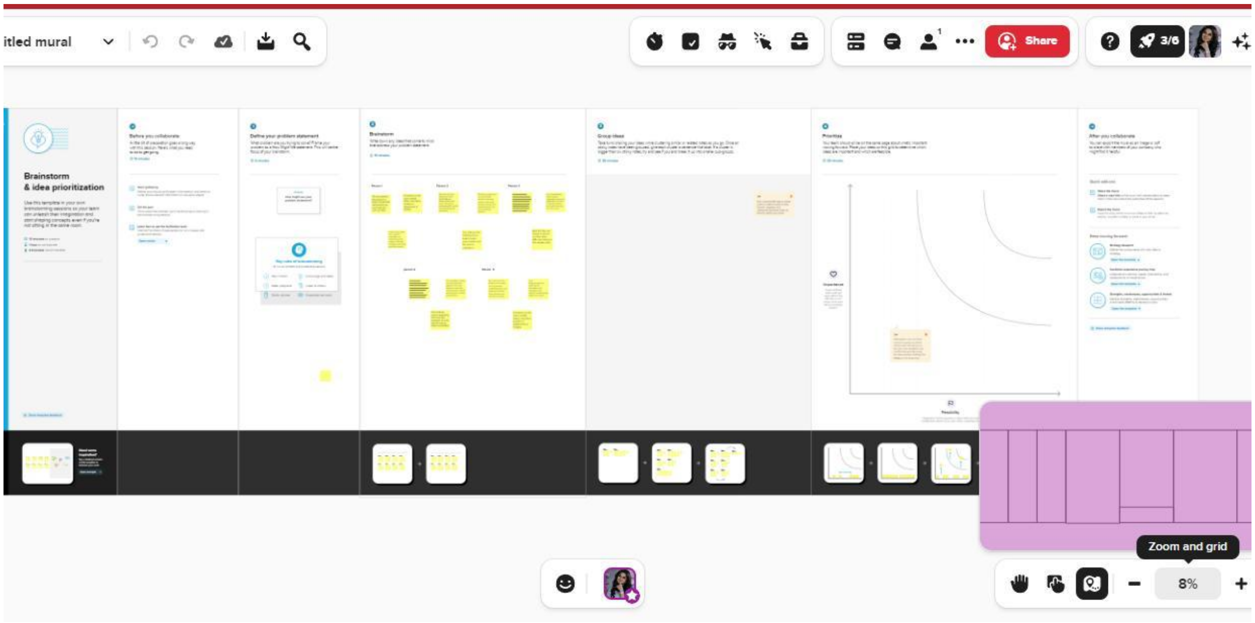
1 Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people.

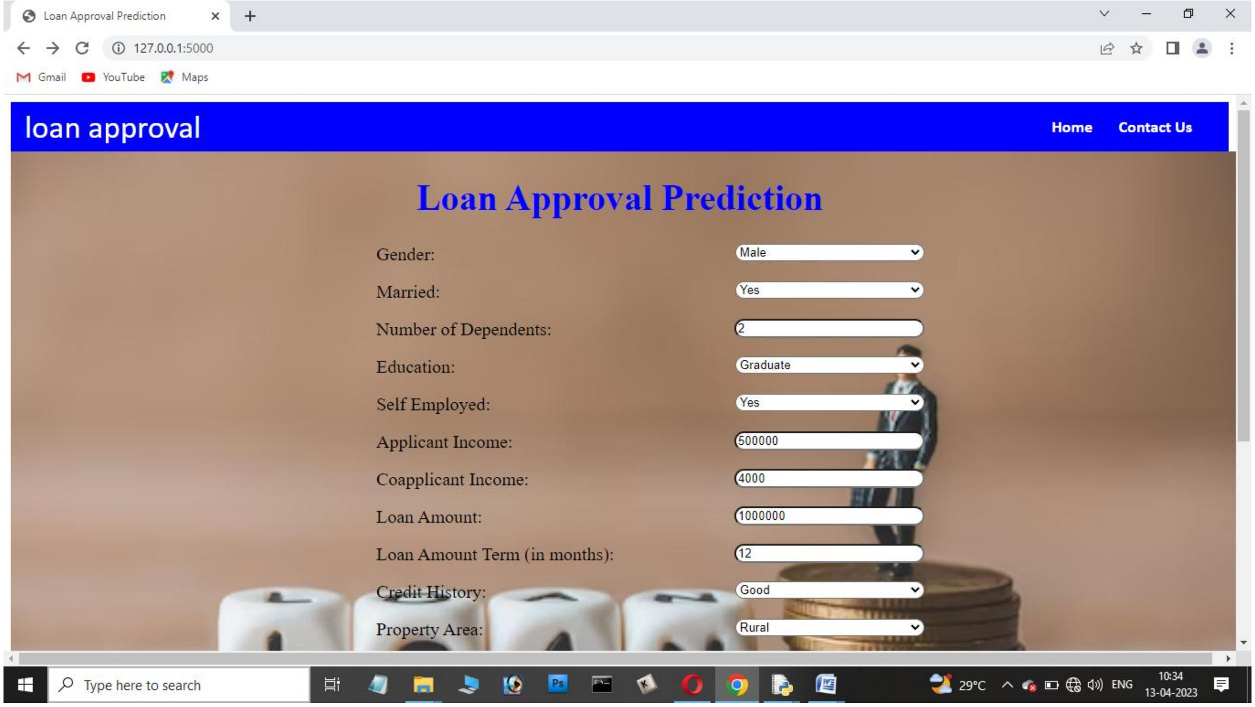
2 Build empathy

The information you add here should be representative of the observations and research

IDEATION & BRAINSTORMING MAP

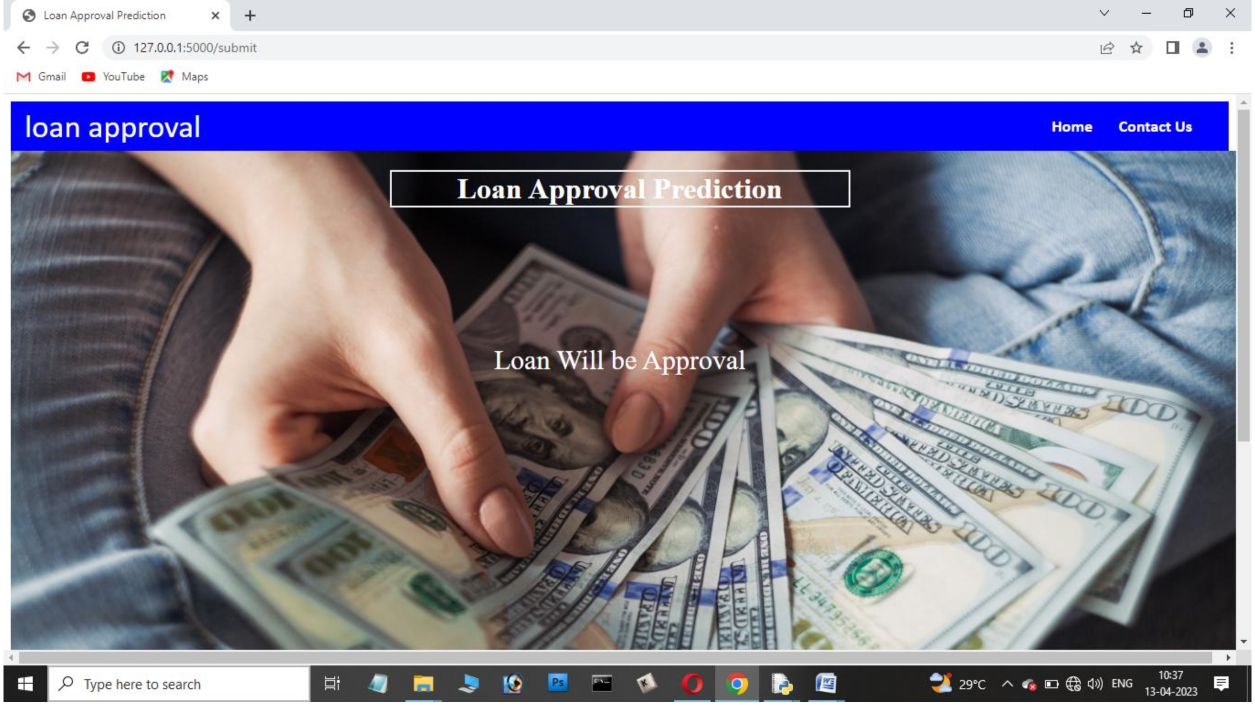


3.RESULT



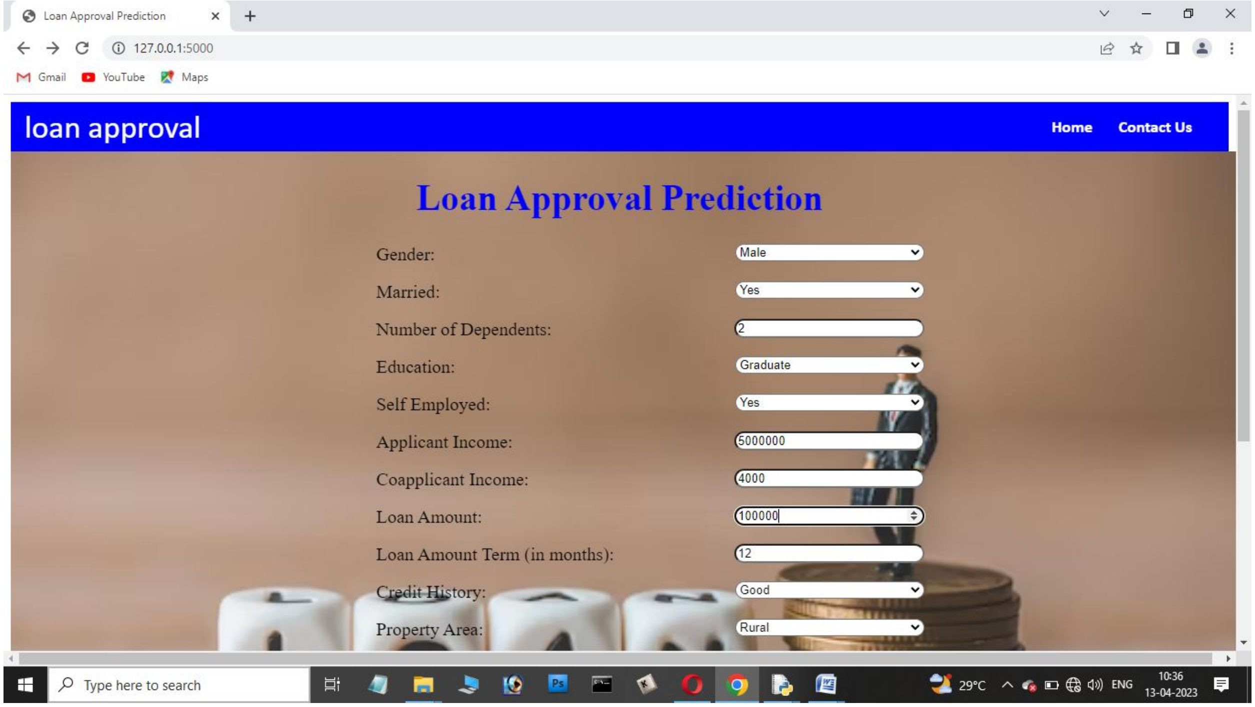
Loan Approval Prediction

Gender:	Male
Married:	Yes
Number of Dependents:	2
Education:	Graduate
Self Employed:	Yes
Applicant Income:	500000
Coapplicant Income:	4000
Loan Amount:	1000000
Loan Amount Term (in months):	12
Credit History:	Good
Property Area:	Rural



Loan Approval Prediction

Loan Will be Approval



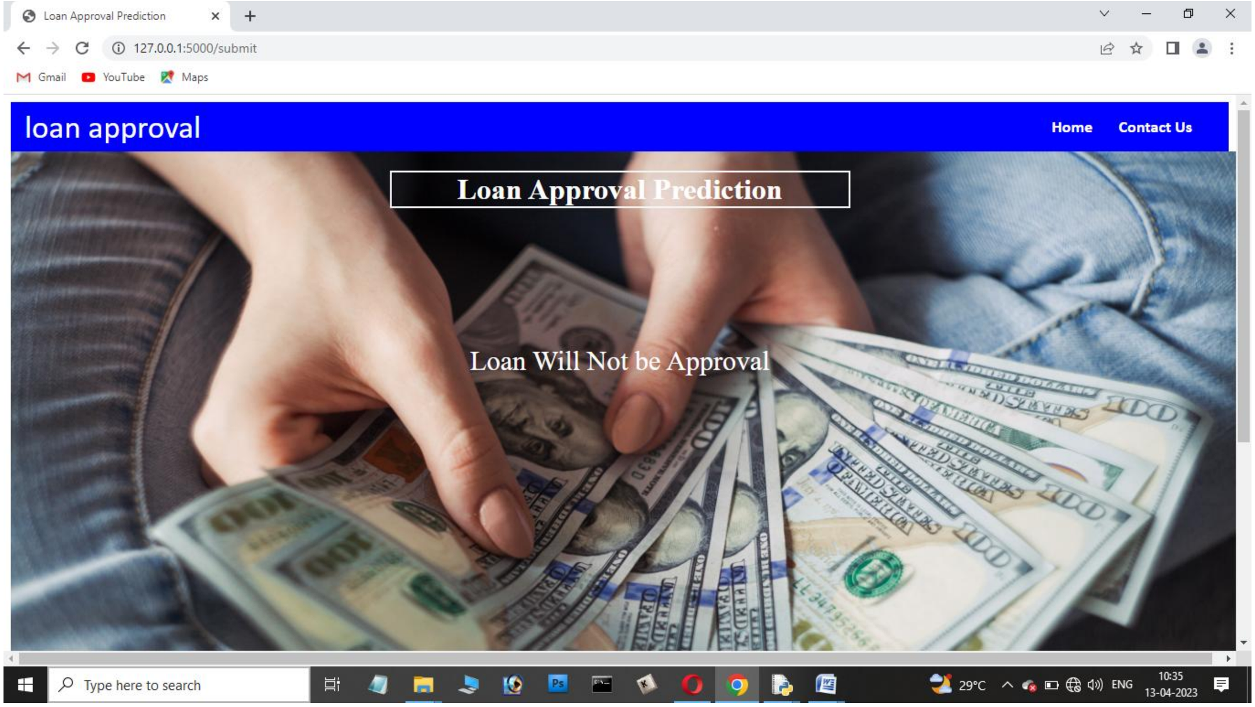
loan approval

Home

Contact Us

Loan Approval Prediction

Gender:	Male
Married:	Yes
Number of Dependents:	2
Education:	Graduate
Self Employed:	Yes
Applicant Income:	5000000
Coapplicant Income:	4000
Loan Amount:	100000
Loan Amount Term (in months):	12
Credit History:	Good
Property Area:	Rural



loan approval

Home

Contact Us

Loan Approval Prediction

Loan Will Not be Approval

4.ADVANDAGES AND DISADVANDAGES

ADVANDAGES:

Improved accuracy: Loan approval prediction can help lenders make more accurate decisions by analyzing a wide range of variables and factors that may impact the borrower's ability to repay the loan.

Time-saving: Predictive models can analyze vast amounts of data and provide results quickly, which can save time for both lenders and borrowers.

Reduced bias: Loan approval prediction can help reduce bias in the lending process by relying on data-driven decisions rather than relying on subjective judgments.

Lower default rates: By accurately predicting the likelihood of loan repayment, lenders can reduce the risk of default and improve their overall loan portfolio performance.

DISADVANDAGES:

Over-reliance on data: Predictive models rely heavily on historical data, which may not always reflect current economic conditions or changes in borrower behavior.

Complexity: Predictive models can be complex and difficult to understand, which can make it challenging for lenders to explain their decisions to borrowers.

Data privacy concerns: Loan approval prediction requires access to sensitive data such as credit scores, income levels, and employment histories, which can raise privacy concerns.

Potential for errors: Predictive models may not always be accurate, and errors can lead to incorrect decisions that may negatively impact borrowers.

5.APPLICATIONS

Collect data: Collect relevant data about loan applicants, such as their credit score, income, employment history, loan amount requested, loan purpose, and other relevant factors.

Preprocess the data: Clean the data by removing missing or inconsistent values, normalize numeric values, and encode categorical variables.

Split the data: Split the data into a training set and a test set. The training set will be used to train the machine learning model, while the test set will be used to evaluate the model's performance.

Train the model: Train a machine learning model using the training set. There are several machine learning algorithms you can use, such as logistic regression, decision trees, random forests, or neural networks. Choose the algorithm that best fits your data and problem.

Evaluate the model: Evaluate the performance of the trained model on the test set. Measure metrics such as accuracy, precision, recall, and F1-score to determine how well the model is performing.

Deploy the model: Deploy the model in a production environment. You can either integrate the model into an existing loan approval system or build a new loan approval system around the model.

Monitor and improve the model: Monitor the performance of the deployed model and improve it over time. Collect feedback from users and use it to refine the model and make it more accurate.

6.CONCLUSION

So here, it can be concluded with confidence that the Naïve Bayes model is extremely efficient and gives a better result when compared to other models. It works correctly and fulfills all requirements of bankers. This system properly and accurately calculate the result. It predicts the loan is approve or reject to loan applicant or customer very accuratly.

7.REFERENCES

1) Kumar Arun, Garg Ishan, Kaur Sanmeet, ―Loan Approval Prediction based on Machine Learning Approach‖, IOSR Journal of Computer Engineering (IOSR-JCE), Vol. 18, Issue 3, pp. 79-81, Ver. I (May-Jun. 2016).

2) Aboobyda Jafar Hamid and Tarig Mohammed Ahmed,―Developing Prediction Model of Loan Risk in Banks using Data Mining‖, Machine Learning and Applications: An International Journal (MLAIJ), Vol.3, No.1, pp. 1-9, March 2016.

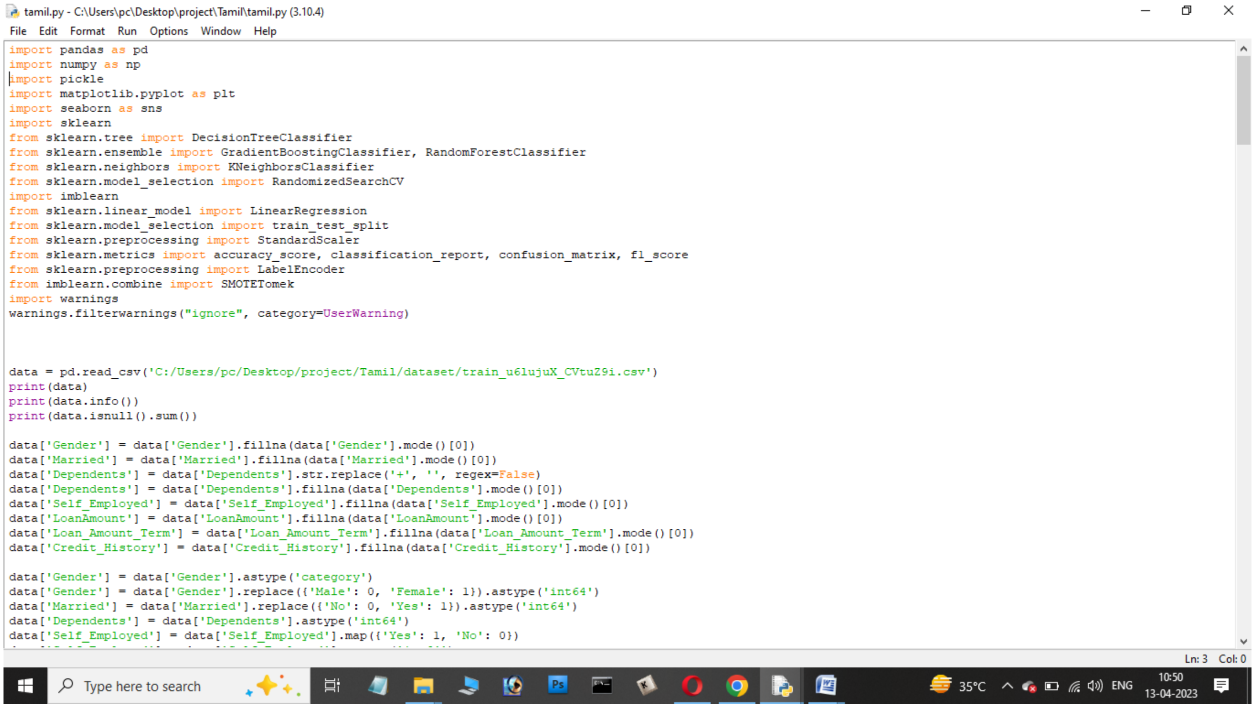
3) S. Vimala, K.C. Sharmili, ―Prediction of Loan Risk using NB and Support Vector Machine‖, International Conference on Advancements in Computing Technologies (ICACT 2018), vol. 4, no. 2, pp. 110-113, 2018.

4) Pidikiti Supriya, Myneedi Pavani, Nagarapu Saisushma, Namburi Vimala Kumari, kVikash,"Loan Prediction by using Machine Learning Models",

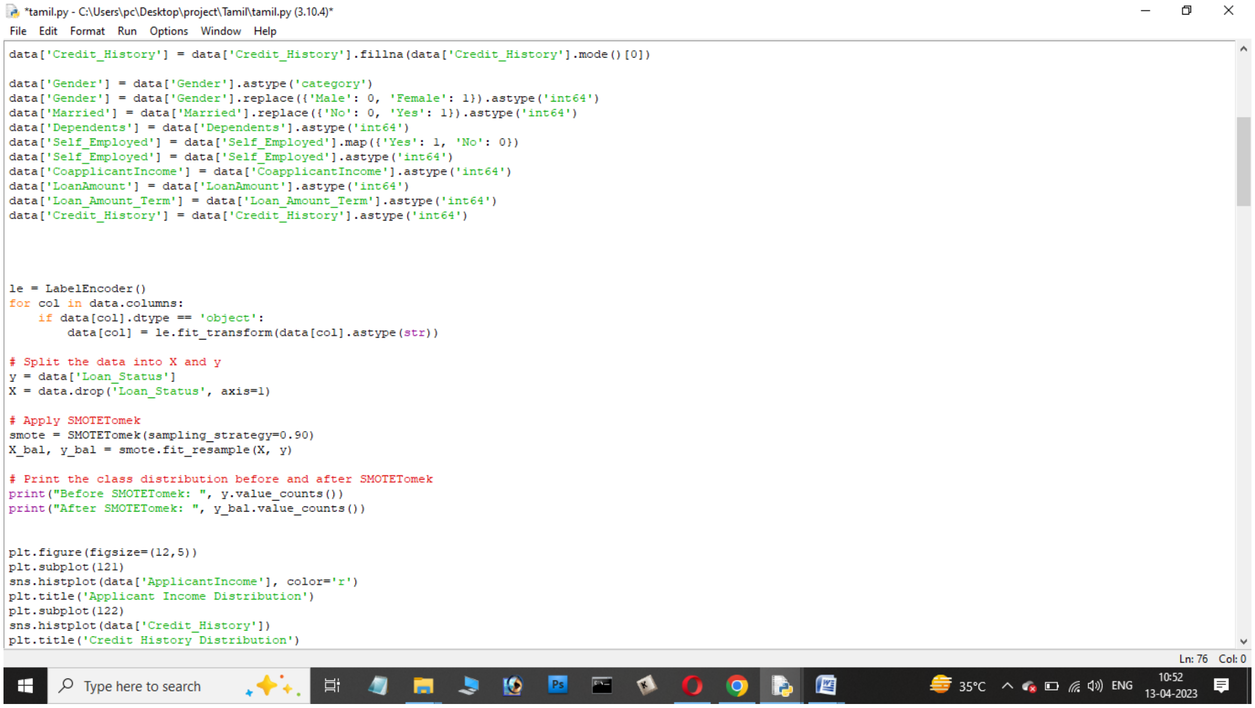
InternationalJournalofEngineering andTechniques.Volume 5 Issue 2, Mar-Apr 2019

5) Nikhil Madane, Siddharth Nanda,"Loan Prediction using Decision tree", Journal of the Gujrat Research History, Volume 21 Issue 14s, December 2019.

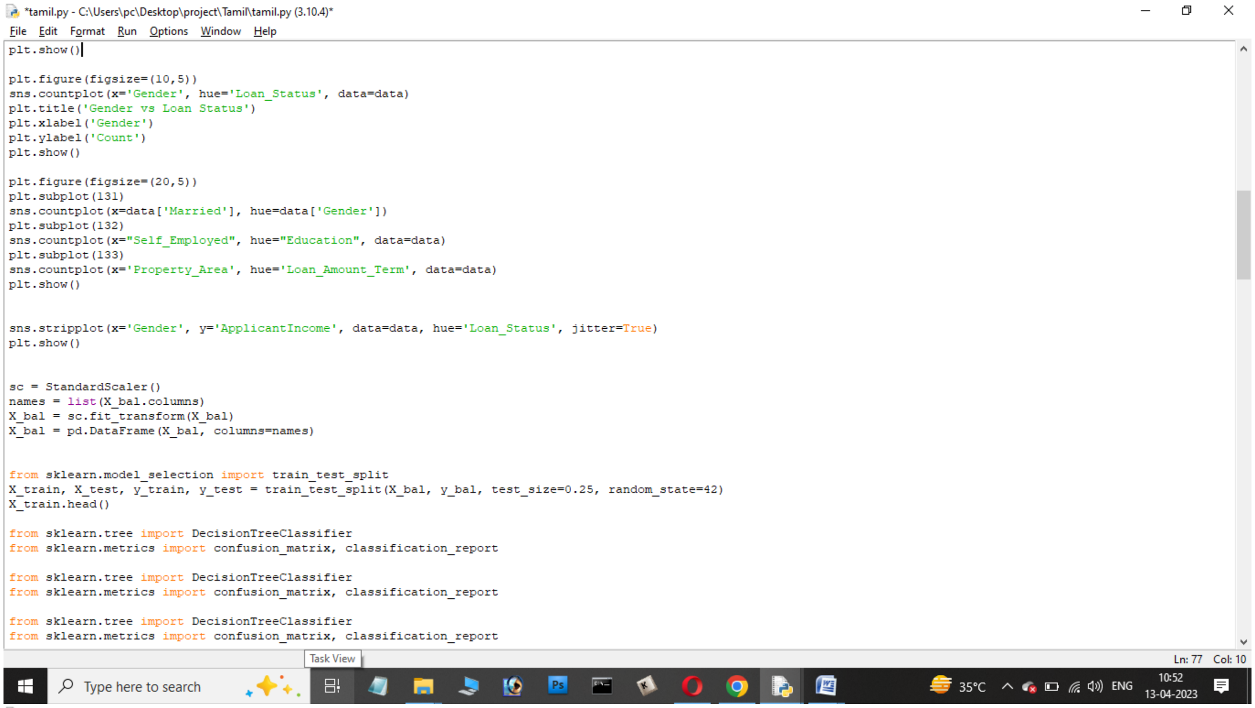
8.APPENDIX


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 3 Col: 0
Type here to search
35°C
ENG
10:50
13-04-2023

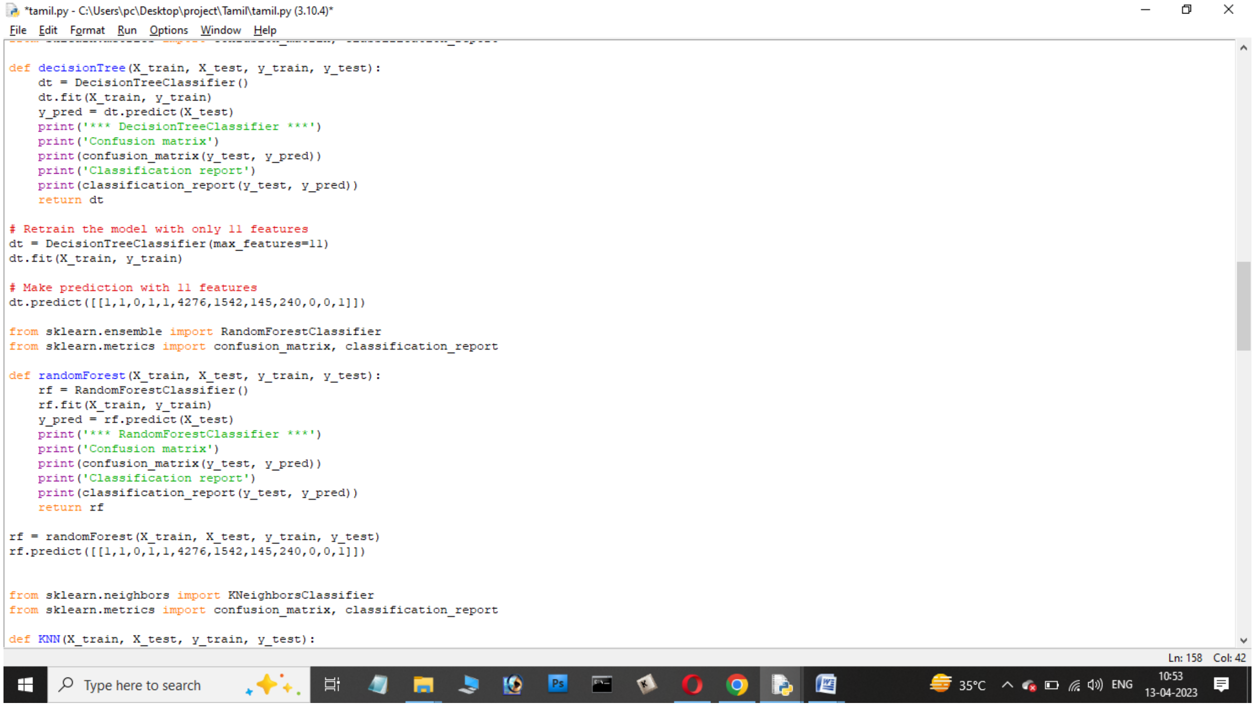
```
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
import imblearn
from sklearn.linear_model import LinearRegression
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
from sklearn.preprocessing import LabelEncoder
from imblearn.combine import SMOTETomek
import warnings
warnings.filterwarnings("ignore", category=UserWarning)



data = pd.read_csv('C:/Users/pc/Desktop/project/Tamil/dataset/train_u6lujuX_CVtuZ9i.csv')
print(data)
print(data.info())
print(data.isnull().sum())

data['Gender'] = data['Gender'].fillna(data['Gender'].mode()[0])
data['Married'] = data['Married'].fillna(data['Married'].mode()[0])
data['Dependents'] = data['Dependents'].str.replace('+', '', regex=False)
data['Dependents'] = data['Dependents'].fillna(data['Dependents'].mode()[0])
data['Self_Employed'] = data['Self_Employed'].fillna(data['Self_Employed'].mode()[0])
data['LoanAmount'] = data['LoanAmount'].fillna(data['LoanAmount'].mode()[0])
data['Loan_Amount_Term'] = data['Loan_Amount_Term'].fillna(data['Loan_Amount_Term'].mode()[0])
data['Credit_History'] = data['Credit_History'].fillna(data['Credit_History'].mode()[0])

data['Gender'] = data['Gender'].astype('category')
data['Gender'] = data['Gender'].replace({'Male': 0, 'Female': 1}).astype('int64')
data['Married'] = data['Married'].replace({'No': 0, 'Yes': 1}).astype('int64')
data['Dependents'] = data['Dependents'].astype('int64')
data['Self_Employed'] = data['Self_Employed'].map({'Yes': 1, 'No': 0})
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 76 Col: 0
Type here to search
35°C
ENG
10:52
13-04-2023

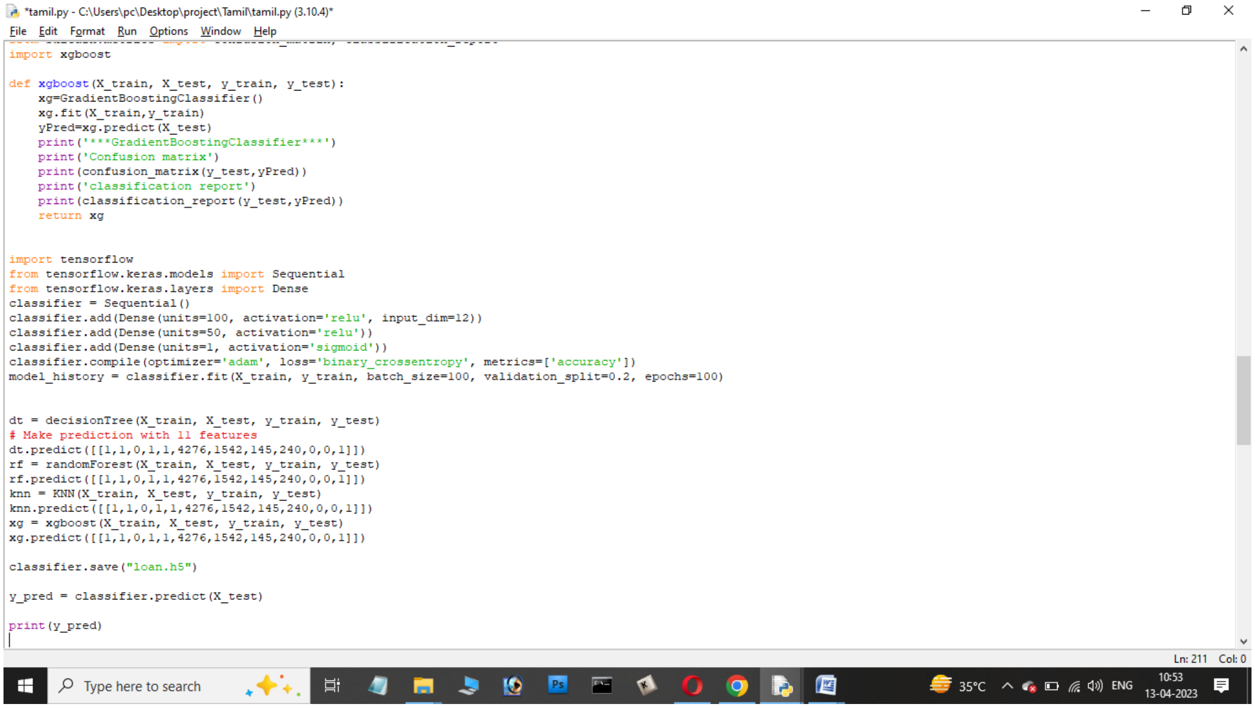
```
data['Credit_History'] = data['Credit_History'].fillna(data['Credit_History'].mode()[0])

data['Gender'] = data['Gender'].astype('category')
data['Gender'] = data['Gender'].replace({'Male': 0, 'Female': 1}).astype('int64')
data['Married'] = data['Married'].replace({'No': 0, 'Yes': 1}).astype('int64')
data['Dependents'] = data['Dependents'].astype('int64')
data['Self_Employed'] = data['Self_Employed'].map({'Yes': 1, 'No': 0})
data['Self_Employed'] = data['Self_Employed'].astype('int64')
data['CoapplicantIncome'] = data['CoapplicantIncome'].astype('int64')
data['LoanAmount'] = data['LoanAmount'].astype('int64')
data['Loan_Amount_Term'] = data['Loan_Amount_Term'].astype('int64')
data['Credit_History'] = data['Credit_History'].astype('int64')




le = LabelEncoder()
for col in data.columns:
    if data[col].dtype == 'object':
        data[col] = le.fit_transform(data[col].astype(str))

# Split the data into X and y
y = data['Loan_Status']
X = data.drop('Loan_Status', axis=1)

# Apply SMOTETomek
smote = SMOTETomek(sampling_strategy=0.90)
X_bal, y_bal = smote.fit_resample(X, y)

# Print the class distribution before and after SMOTETomek
print("Before SMOTETomek: ", y.value_counts())
print("After SMOTETomek: ", y_bal.value_counts())


plt.figure(figsize=(12,5))
plt.subplot(121)
sns.histplot(data['ApplicantIncome'], color='r')
plt.title('Applicant Income Distribution')
plt.subplot(122)
sns.histplot(data['Credit_History'])
plt.title('Credit History Distribution')
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 77 Col: 10
Task View
Type here to search
35°C
ENG
10:52
13-04-2023

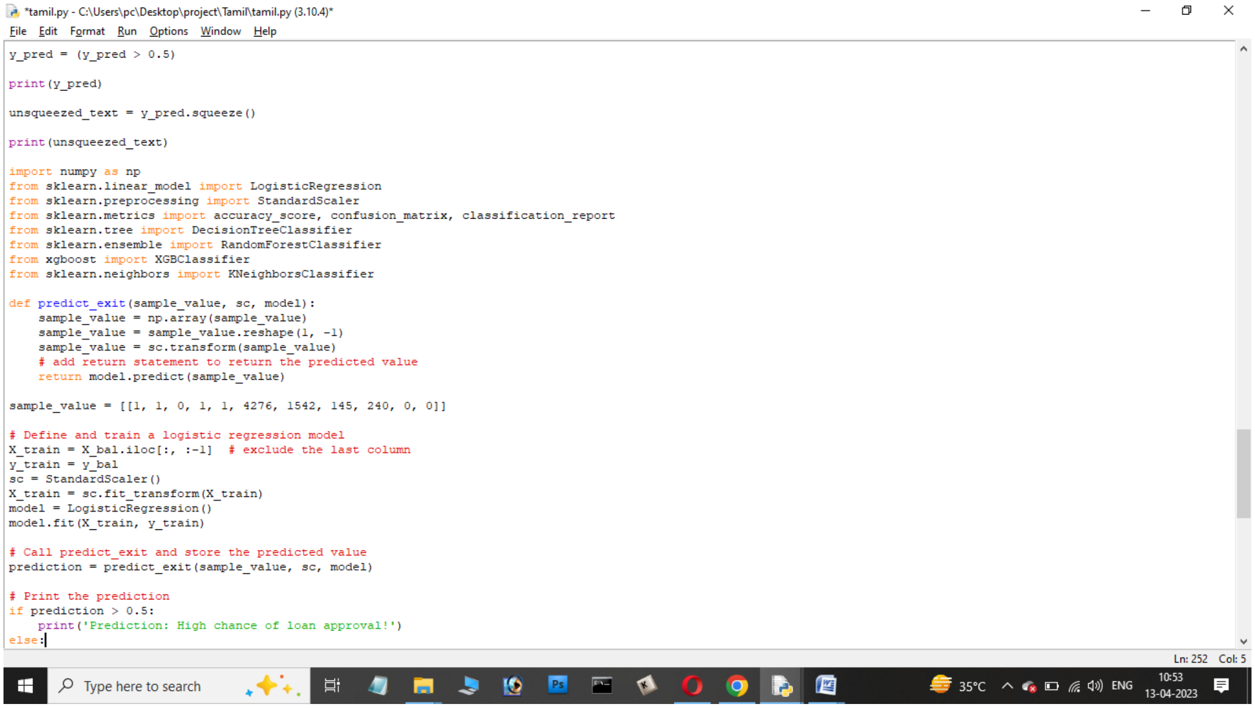
```
plt.show()

plt.figure(figsize=(10,5))
sns.countplot(x='Gender', hue='Loan_Status', data=data)
plt.title('Gender vs Loan Status')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()

plt.figure(figsize=(20,5))
plt.subplot(131)
sns.countplot(x=data['Married'], hue=data['Gender'])
plt.subplot(132)
sns.countplot(x="Self_Employed", hue="Education", data=data)
plt.subplot(133)
sns.countplot(x='Property_Area', hue='Loan_Amount_Term', data=data)
plt.show()


sns.stripplot(x='Gender', y='ApplicantIncome', data=data, hue='Loan_Status', jitter=True)
plt.show()


sc = StandardScaler()
names = list(X_bal.columns)
X_bal = sc.fit_transform(X_bal)
X_bal = pd.DataFrame(X_bal, columns=names)


from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_bal, y_bal, test_size=0.25, random_state=42)
X_train.head()

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, classification_report

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, classification_report

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, classification_report
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 158 Col: 42
Type here to search
35°C
ENG
10:53
13-04-2023

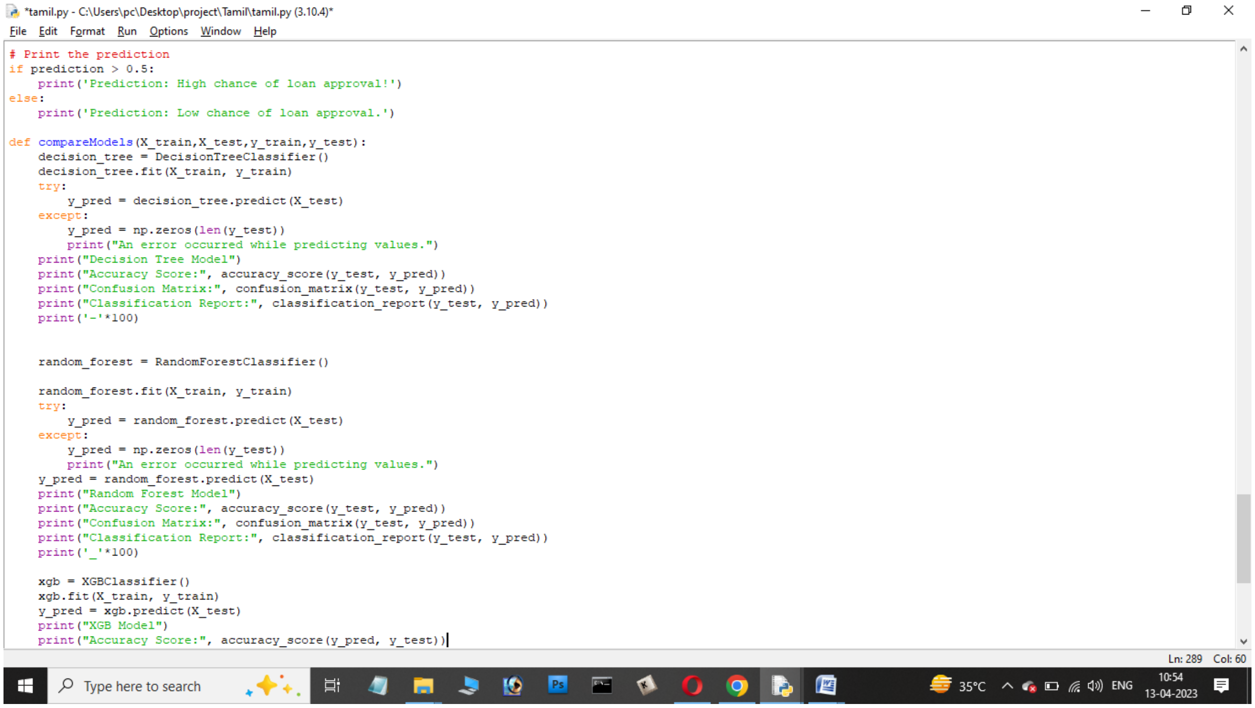
```
def decisionTree(X_train, X_test, y_train, y_test):
    dt = DecisionTreeClassifier()
    dt.fit(X_train, y_train)
    y_pred = dt.predict(X_test)
    print('*** DecisionTreeClassifier ***')
    print('Confusion matrix')
    print(confusion_matrix(y_test, y_pred))
    print('Classification report')
    print(classification_report(y_test, y_pred))
    return dt

# Retrain the model with only 11 features
dt = DecisionTreeClassifier(max_features=11)
dt.fit(X_train, y_train)

# Make prediction with 11 features
dt.predict([[1,1,0,1,1,4276,1542,145,240,0,0,1]])

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report

def randomForest(X_train, X_test, y_train, y_test):
    rf = RandomForestClassifier()
    rf.fit(X_train, y_train)
    y_pred = rf.predict(X_test)
    print('*** RandomForestClassifier ***')
    print('Confusion matrix')
    print(confusion_matrix(y_test, y_pred))
    print('Classification report')
    print(classification_report(y_test, y_pred))
    return rf

rf = randomForest(X_train, X_test, y_train, y_test)
rf.predict([[1,1,0,1,1,4276,1542,145,240,0,0,1]])


from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, classification_report

def KNN(X_train, X_test, y_train, y_test):
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 211 Col: 0
Type here to search
35°C
ENG
10:53
13-04-2023

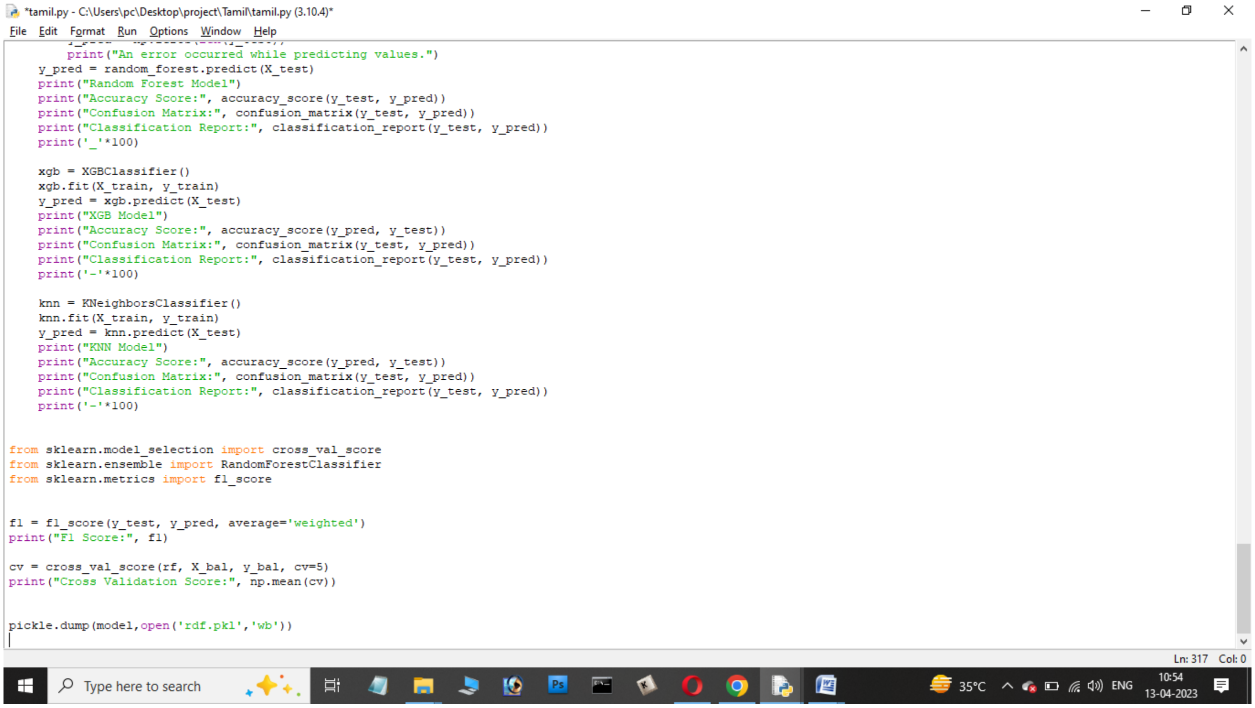
```
import xgboost

def xgboost(X_train, X_test, y_train, y_test):
    xg=GradientBoostingClassifier()
    xg.fit(X_train,y_train)
    yPred=xg.predict(X_test)
    print('***GradientBoostingClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print('classification report')
    print(classification_report(y_test,yPred))
    return xg


import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
classifier = Sequential()
classifier.add(Dense(units=100, activation='relu', input_dim=12))
classifier.add(Dense(units=50, activation='relu'))
classifier.add(Dense(units=1, activation='sigmoid'))
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
model_history = classifier.fit(X_train, y_train, batch_size=100, validation_split=0.2, epochs=100)


dt = decisionTree(X_train, X_test, y_train, y_test)
# Make prediction with 11 features
dt.predict([[1,1,0,1,1,4276,1542,145,240,0,0,1]])
rf = randomForest(X_train, X_test, y_train, y_test)
rf.predict([[1,1,0,1,1,4276,1542,145,240,0,0,1]])
knn = KNN(X_train, X_test, y_train, y_test)
knn.predict([[1,1,0,1,1,4276,1542,145,240,0,0,1]])
xg = xgboost(X_train, X_test, y_train, y_test)
xg.predict([[1,1,0,1,1,4276,1542,145,240,0,0,1]])

classifier.save("loan.h5")

y_pred = classifier.predict(X_test)

print(y_pred)
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 252 Col: 5
Type here to search
35°C
ENG
10:53
13-04-2023

```
y_pred = (y_pred > 0.5)

print(y_pred)

unsqueezed_text = y_pred.squeeze()

print(unsqueezed_text)

import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.neighbors import KNeighborsClassifier

def predict_exit(sample_value, sc, model):
    sample_value = np.array(sample_value)
    sample_value = sample_value.reshape(1, -1)
    sample_value = sc.transform(sample_value)
    # add return statement to return the predicted value
    return model.predict(sample_value)

sample_value = [[1, 1, 0, 1, 1, 4276, 1542, 145, 240, 0, 0]]

# Define and train a logistic regression model
X_train = X_bal.iloc[:, :-1]  # exclude the last column
y_train = y_bal
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
model = LogisticRegression()
model.fit(X_train, y_train)

# Call predict_exit and store the predicted value
prediction = predict_exit(sample_value, sc, model)

# Print the prediction
if prediction > 0.5:
    print('Prediction: High chance of loan approval!')
else:
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 289 Col: 60
Type here to search
35°C
ENG
10:54
13-04-2023

```python
# Print the prediction
if prediction > 0.5:
    print('Prediction: High chance of loan approval!')
else:
    print('Prediction: Low chance of loan approval.')

def compareModels(X_train,X_test,y_train,y_test):
    decision_tree = DecisionTreeClassifier()
    decision_tree.fit(X_train, y_train)
    try:
        y_pred = decision_tree.predict(X_test)
    except:
        y_pred = np.zeros(len(y_test))
        print("An error occurred while predicting values.")
    print("Decision Tree Model")
    print("Accuracy Score:", accuracy_score(y_test, y_pred))
    print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
    print("Classification Report:", classification_report(y_test, y_pred))
    print('-'*100)


    random_forest = RandomForestClassifier()

    random_forest.fit(X_train, y_train)
    try:
        y_pred = random_forest.predict(X_test)
    except:
        y_pred = np.zeros(len(y_test))
        print("An error occurred while predicting values.")
    y_pred = random_forest.predict(X_test)
    print("Random Forest Model")
    print("Accuracy Score:", accuracy_score(y_test, y_pred))
    print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
    print("Classification Report:", classification_report(y_test, y_pred))
    print('_'*100)

    xgb = XGBClassifier()
    xgb.fit(X_train, y_train)
    y_pred = xgb.predict(X_test)
    print("XGB Model")
    print("Accuracy Score:", accuracy_score(y_pred, y_test))
```


tamil.py - C:\Users\pc\Desktop\project\Tamil\tamil.py (3.10.4)
File Edit Format Run Options Window Help
Ln: 317 Col: 0
Type here to search
35°C
ENG
10:54
13-04-2023

```python
        print("An error occurred while predicting values.")
    y_pred = random_forest.predict(X_test)
    print("Random Forest Model")
    print("Accuracy Score:", accuracy_score(y_test, y_pred))
    print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
    print("Classification Report:", classification_report(y_test, y_pred))
    print('_'*100)

    xgb = XGBClassifier()
    xgb.fit(X_train, y_train)
    y_pred = xgb.predict(X_test)
    print("XGB Model")
    print("Accuracy Score:", accuracy_score(y_pred, y_test))
    print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
    print("Classification Report:", classification_report(y_test, y_pred))
    print('-'*100)

    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    print("KNN Model")
    print("Accuracy Score:", accuracy_score(y_pred, y_test))
    print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
    print("Classification Report:", classification_report(y_test, y_pred))
    print('-'*100)


from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import f1_score


f1 = f1_score(y_test, y_pred, average='weighted')
print("F1 Score:", f1)

cv = cross_val_score(rf, X_bal, y_bal, cv=5)
print("Cross Validation Score:", np.mean(cv))


pickle.dump(model,open('rdf.pkl','wb'))
```

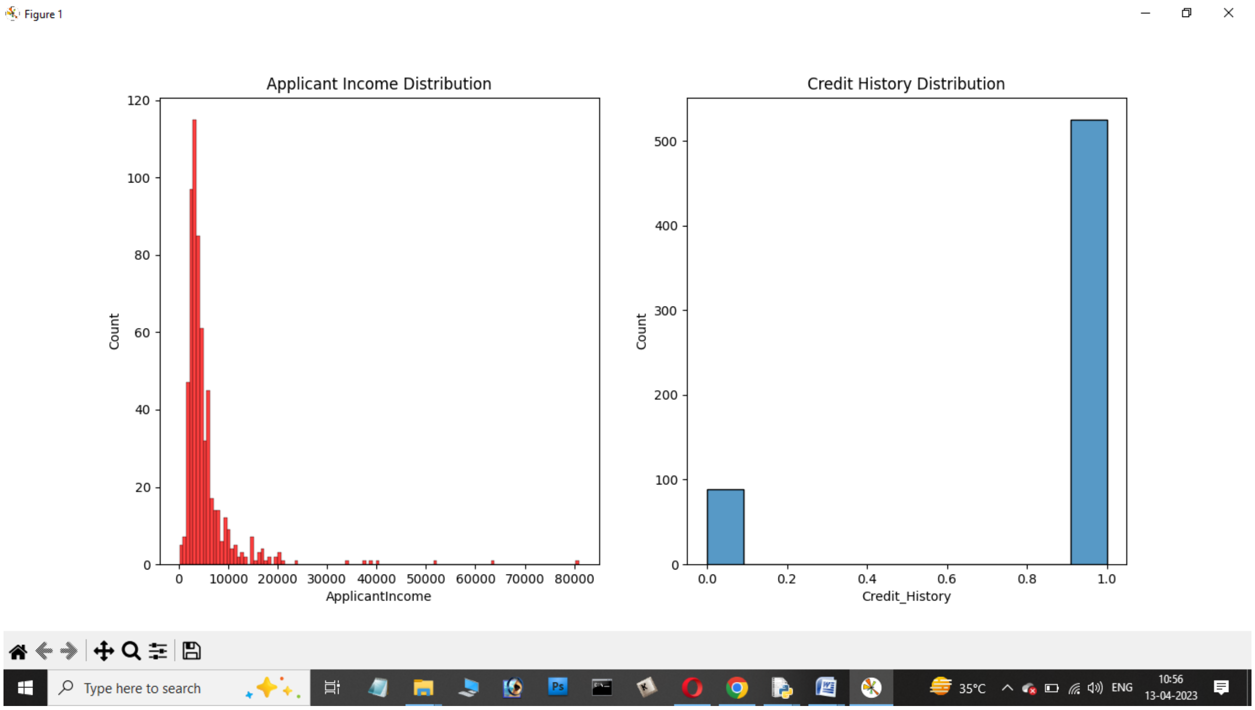


Figure 1
Applicant Income Distribution
Count
120
100
80
60
40
20
0
0
10000
20000
30000
40000
50000
60000
70000
80000
ApplicantIncome
Credit History Distribution
Count
500
400
300
200
100
0
0.0
0.2
0.4
0.6
0.8
1.0
Credit_History
Type here to search
35°C
ENG
10:56
13-04-2023

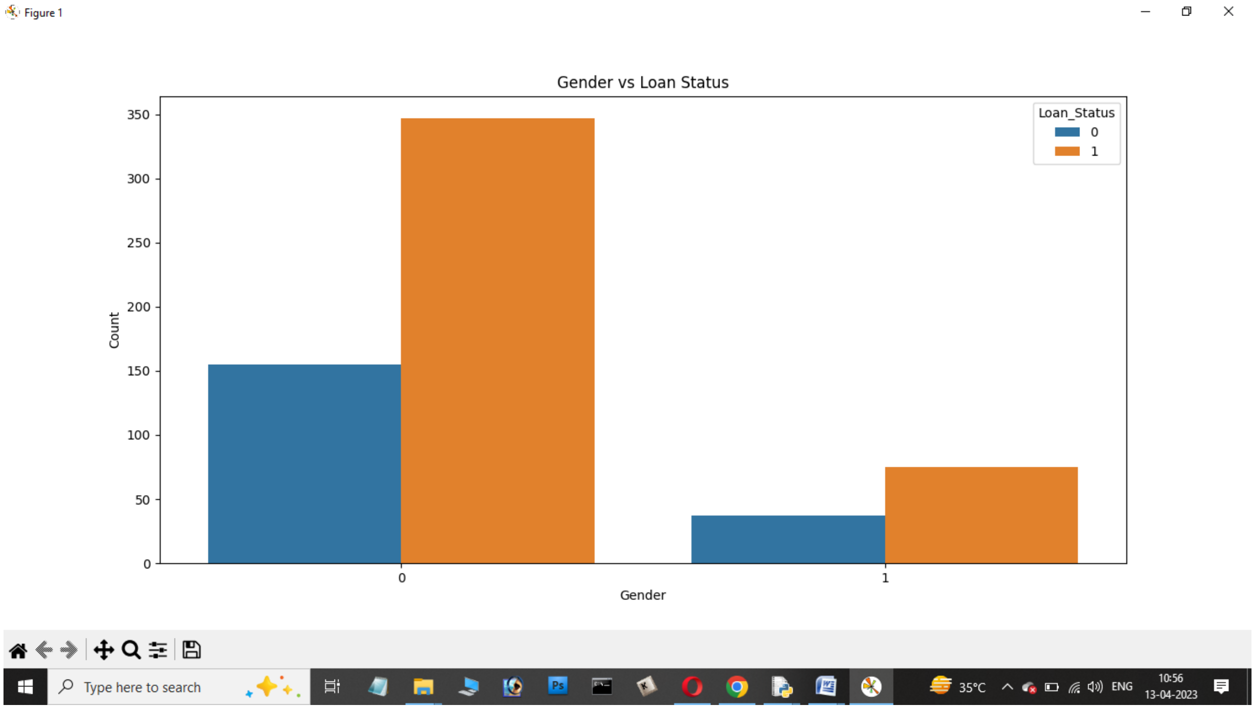


Figure 1
Gender vs Loan Status
Loan_Status
0
1
Count
350
300
250
200
150
100
50
0
0
1
Gender
Type here to search
35°C
ENG
10:56
13-04-2023

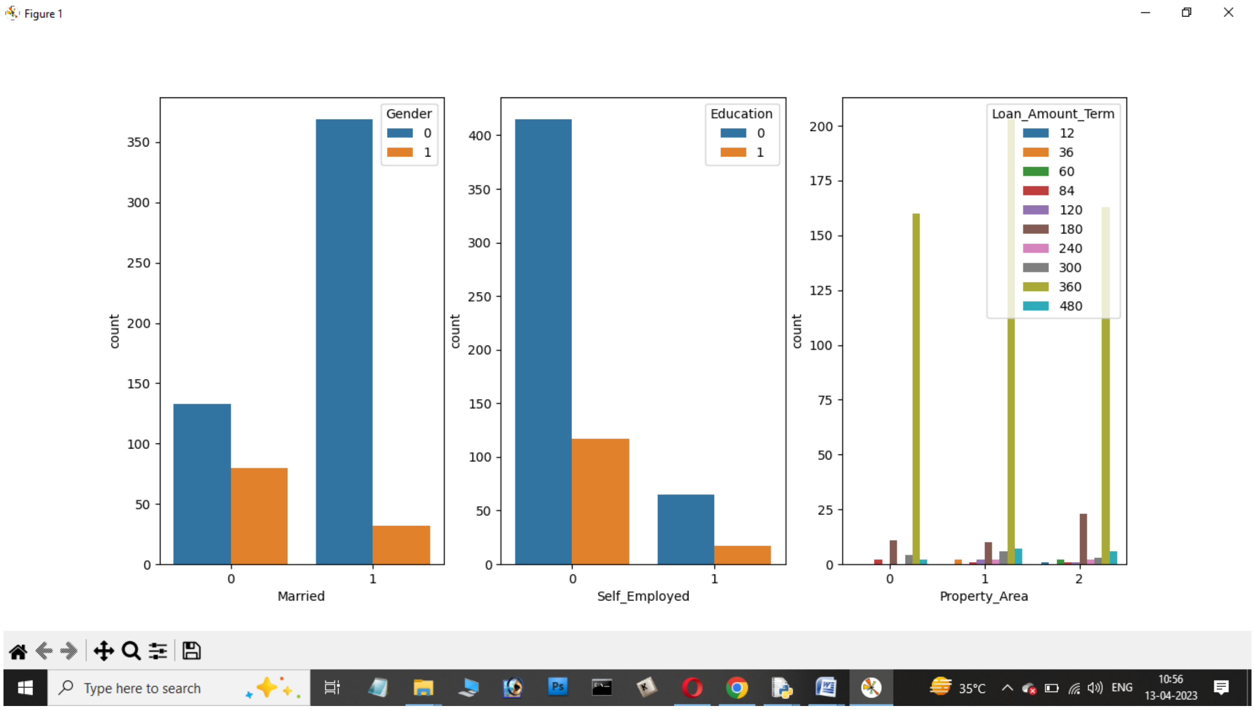


Figure 1
Gender
0
1
count
350
300
250
200
150
100
50
0
0
1
Married
Education
0
1
count
400
350
300
250
200
150
100
50
0
0
1
Self_Employed
Loan_Amount_Term
12
36
60
84
120
180
240
300
360
480
count
200
175
150
125
100
75
50
25
0
0
1
2
Property_Area
Type here to search
35°C
ENG
10:56
13-04-2023

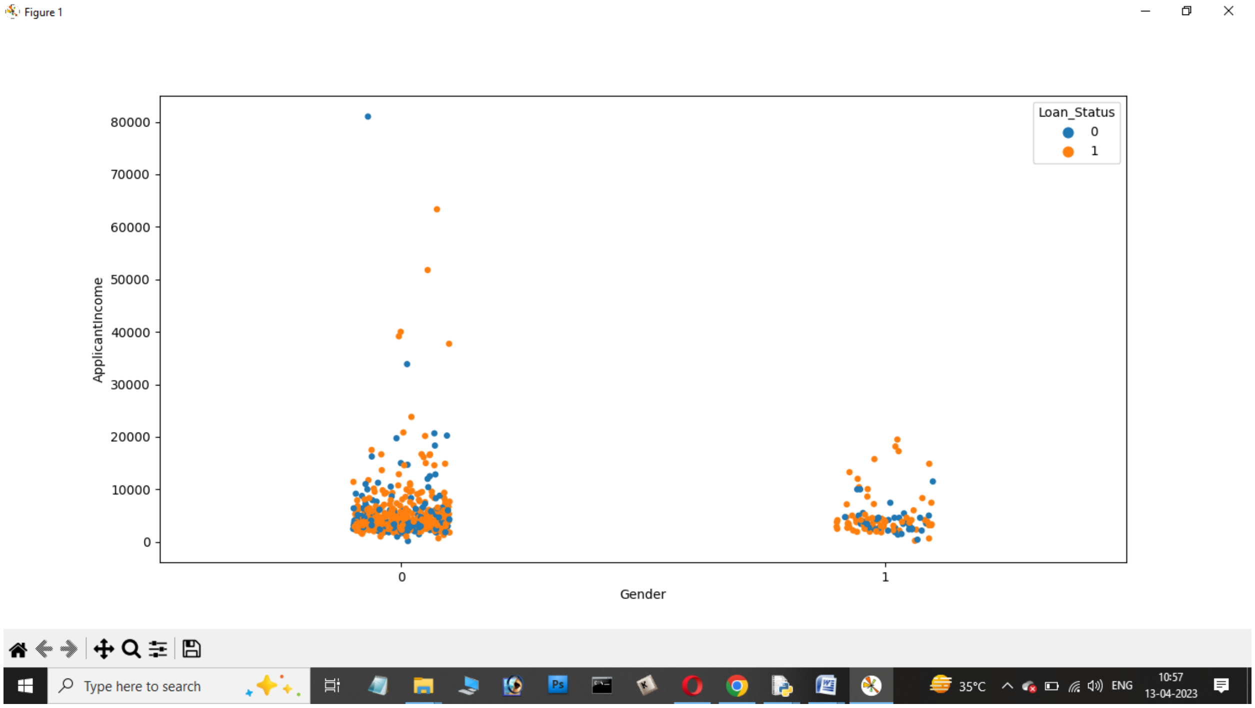
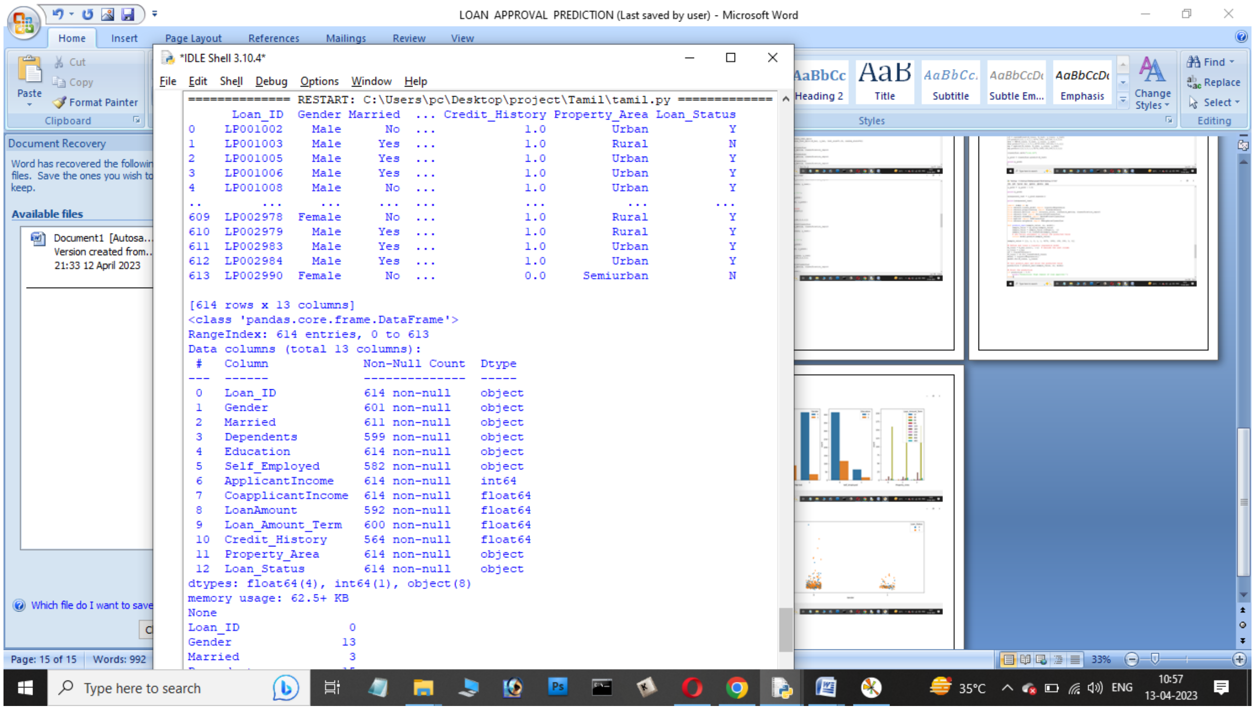


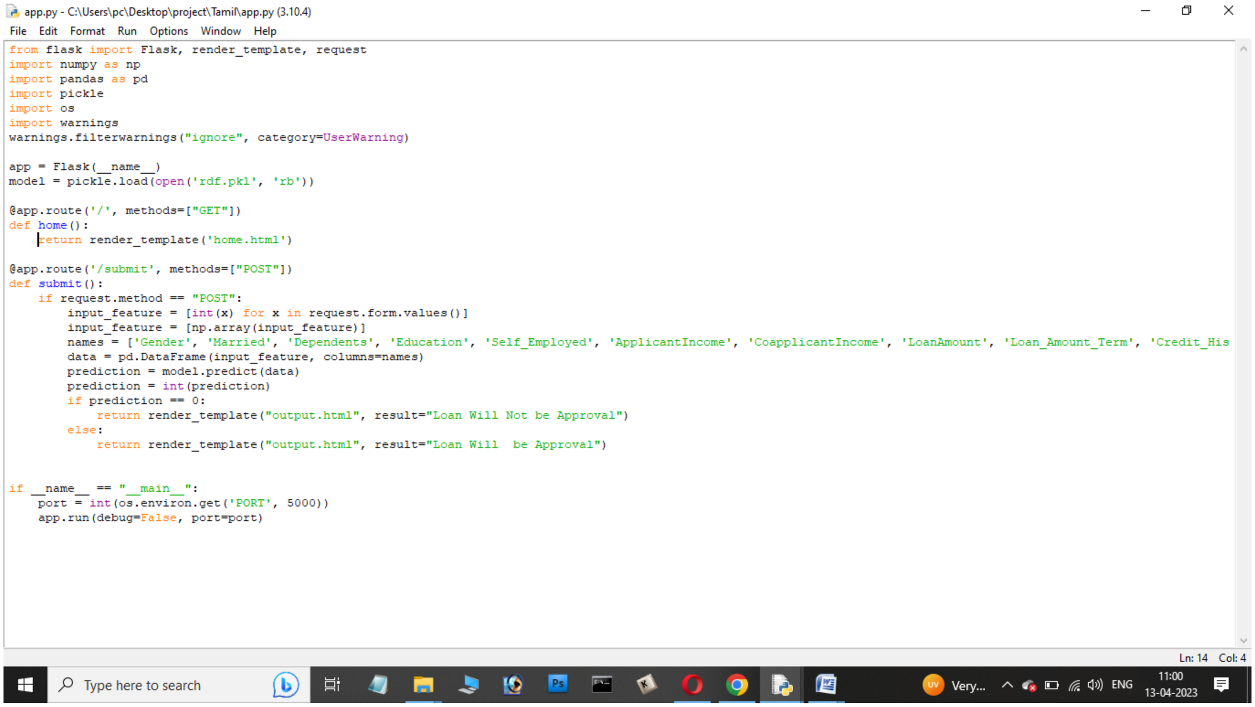
Figure 1
Loan_Status
0
1
ApplicantIncome
80000
70000
60000
50000
40000
30000
20000
10000
0
0
1
Gender
Type here to search
35°C
ENG
10:57
13-04-2023



LOAN APPROVAL PREDICTION (Last saved by user) - Microsoft Word
IDLE Shell 3.10.4
File Edit Shell Debug Options Window Help

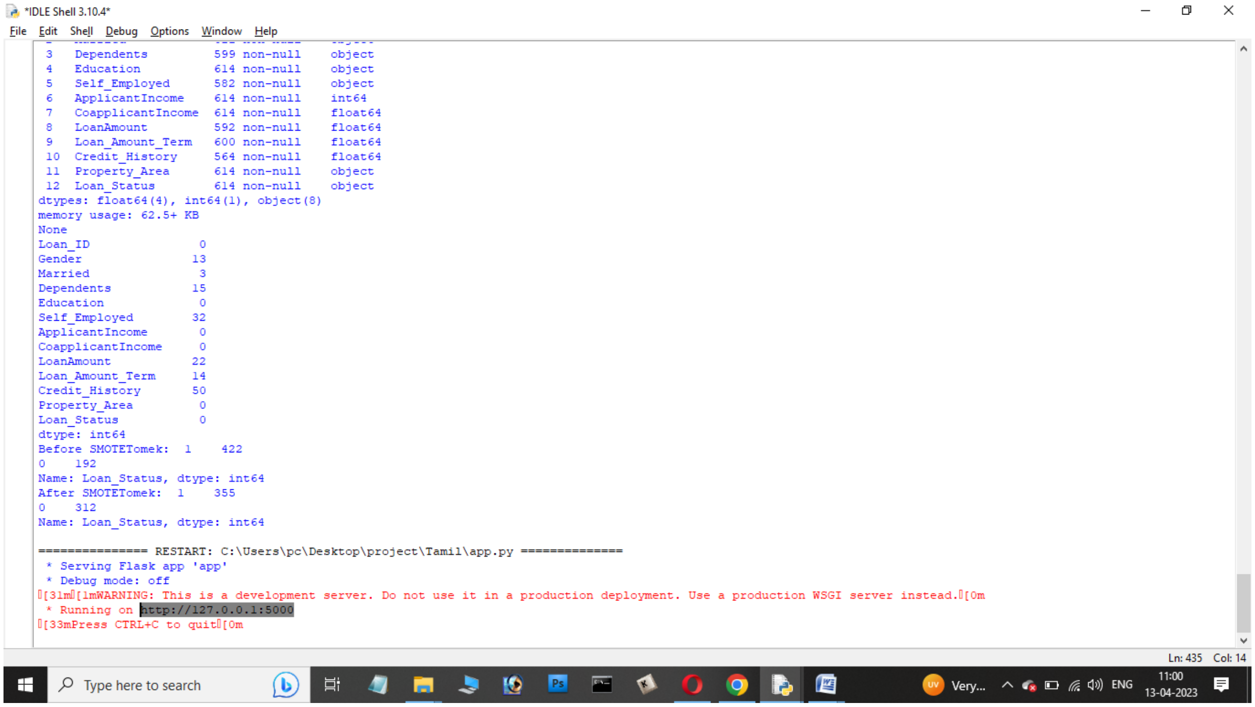
```
============== RESTART: C:\Users\pc\Desktop\project\Tamil\tamil.py =============
      Loan_ID  Gender Married  ... Credit_History Property_Area Loan_Status
0    LP001002    Male      No  ...            1.0         Urban           Y
1    LP001003    Male     Yes  ...            1.0         Rural           N
2    LP001005    Male     Yes  ...            1.0         Urban           Y
3    LP001006    Male     Yes  ...            1.0         Urban           Y
4    LP001008    Male      No  ...            1.0         Urban           Y
..        ...     ...     ...  ...            ...           ...         ...
609  LP002978  Female      No  ...            1.0         Rural           Y
610  LP002979    Male     Yes  ...            1.0         Rural           Y
611  LP002983    Male     Yes  ...            1.0         Urban           Y
612  LP002984    Male     Yes  ...            1.0         Urban           Y
613  LP002990  Female      No  ...            0.0     Semiurban           N

[614 rows x 13 columns]
<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
None
Loan_ID               0
Gender               13
Married               3
```

Python flask file:



app.py - C:\Users\pc\Desktop\project\Tamil\app.py (3.10.4)
File Edit Format Run Options Window Help

```python
from flask import Flask, render_template, request
import numpy as np
import pandas as pd
import pickle
import os
import warnings
warnings.filterwarnings("ignore", category=UserWarning)

app = Flask(__name__)
model = pickle.load(open('rdf.pkl', 'rb'))

@app.route('/', methods=["GET"])
def home():
    return render_template('home.html')

@app.route('/submit', methods=["POST"])
def submit():
    if request.method == "POST":
        input_feature = [int(x) for x in request.form.values()]
        input_feature = [np.array(input_feature)]
        names = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term', 'Credit_His
        data = pd.DataFrame(input_feature, columns=names)
        prediction = model.predict(data)
        prediction = int(prediction)
        if prediction == 0:
            return render_template("output.html", result="Loan Will Not be Approval")
        else:
            return render_template("output.html", result="Loan Will  be Approval")


if __name__ == "__main__":
    port = int(os.environ.get('PORT', 5000))
    app.run(debug=False, port=port)
```



```
 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
None
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
Before SMOTETomek:  1    422
0    192
Name: Loan_Status, dtype: int64
After SMOTETomek:  1    355
0    312
Name: Loan_Status, dtype: int64

=============== RESTART: C:\Users\pc\Desktop\project\Tamil\app.py ==============
 * Serving Flask app 'app'
 * Debug mode: off
[31m[1mWARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.[0m
 * Running on http://127.0.0.1:5000
[33mPress CTRL+C to quit[0m
```