PREDICTING PERSONAL LOAN APPROVAL USING MACHINE LEARNING

TEAM ID : NM2023TMID21630

HARISH.R (F558EA3EF92254986B84F94C2CE049E7)

SIVA.M (8A510E122AF6F88B6B4458689E5560AC)

JAYASURYA.K (FB5AD484E66F5F3B2E74FE679D65C3E9)

LOGANATHAN.S (24988B956FE6E8CA01943FCD530BE6E8)

PREDICTING PERSONAL LOAN APPROVAL USING

MACHINE LEARNING

INDRODUCTION:

*This is a classification problem in which we need to classifywhether the loan will be approved or not.

*The company wants to automate the loan eligibility process based on customer detail provided while filing out outlineapplication forms.

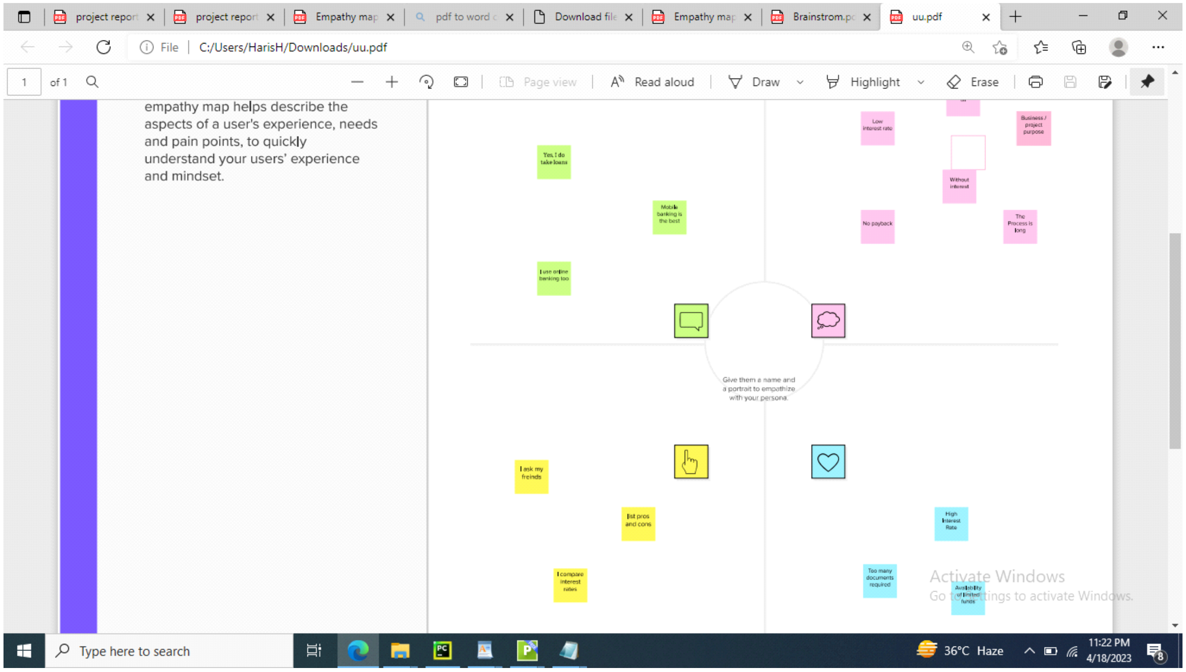
*To automate this process, they have providd a dataset to identify the customer segments that are eligible for loan amounts sothat they can specifically target these customer.

Purpose:

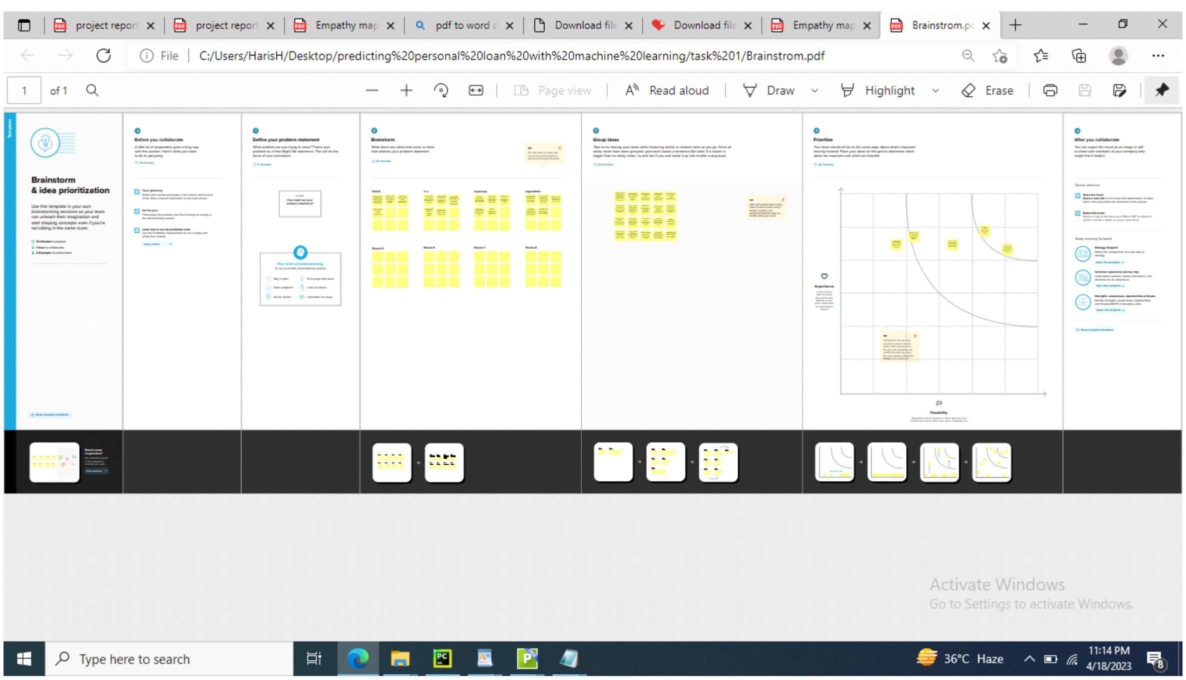
*The prediction model not only helps the applicant but alsohelps the bank by minimizing the risk and reducing the number of defaulters.

*It is done by predicting if the loan can be given to the person and basis of veries parameters like credit score ,income ,age ,marital status , gender , etc.

Empathy map:



Brain Strom:



Advantages :

- you to consolidate high-interest debt. ...
- You can use them to finance your wedding or dream vacation. ...
- They have predictable payment schedules. ...
- Personal loans are flexible in their uses

- They help you pay for emergency expenses without draining your savings. ...

They enable

Low Interest Rates: Generally, bank loans have the cheapest interest rates. The rates you paywill be cheaper than other types of high interest loans, such as venture capital. As Bizfluent says, bank loans offer significantly lower interest rates than you will find with credit cards or overdraft.

Advantages of Loan Stock

The money raised from the market does not have to be repaid, unlike debt financing which has a definite repayment schedule. read more. In the stock, the finance business keeps sharesof its own as security to secure the finance

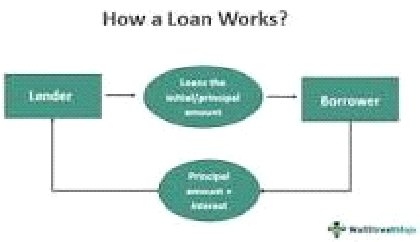
What is the advantage of loan portfolio

Portfolio lenders **focus more on cash flow and the individual's business history rather thanthe borrower's income and other personal metrics**. In some instances, investors may not have to provide personal tax returns if the cash flow being considered by the portfolio lender is based on rent rather than personal income.

Disadvantages:

Loans are not very flexible - you could be paying interest on funds you're not using. You couldhave trouble making monthly repayments if your customers don't pay you promptly, causing cashflow problems

What is the problem of loans?



A problem loan is a scenario where **borrowers fail to repay monthly loan installments**. Thebank labels these loans as nonperforming assets (NPA). It can occur with either a commercialloan or a consumer loan. The loan is considered a default when borrowers miss consecutive repayments beyond the delinquency periods.

What are the disadvantages of loan prediction system?

The disadvantage of this model is that **it emphasize different weights to each factor** but in real life sometime loan can be approved on the basis of single strong factor only, which is not possible through this system. Loan Prediction is very helpful for employee of banks as well asfor the applicant also

Application:

Loan Prediction Project using Machine Learning in Python

- Understanding the various features (columns) of the dataset: ...
- Understanding Distribution of Categorical Variables: ...
- Outliers of LoanAmount and Applicant Income: ...
- Data Preparation for Model Building: ...
- Generic Classification Function: ...
- Model Building:

*We have data of some predicted loans from history. So when there isname of some **'Data'** there is a lot interesting for **'Data Scientists'.**

Introduction Loan Prediction Problem

Welcome to this article on Loan Prediction Problem. Below is a brief introduction to this topic to get you acquainted with what you will be learning.

The Objective of the Article

This article is designed for people who want to solve binary classification problems using Python. By the end of this article, you will have the necessary skills and techniques required to solve such problems. This article provides you with sufficient theory and practiceknowledge to hone you

Problem Statement

Understanding the problem statement is the first and foremost step. This would help you give an intuition of what you will face ahead of time. Let us see the problem statement.

Dream Housing Finance company deals in all home loans. They have a presence across all urban, semi-urban and rural areas. Customers first apply for a home loan after that company validates the customer's eligibility for a loan. The company wants to automate the loan eligibility process (real-time) based on customer detail provided while filling out the online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History, and others. To automate this process, they have given a problem to identify the customer segments, that are eligible for loan amounts so that they can specifically target these customers.

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 (s). Classification can be of two types:

- **Binary Classification**:- In this, we have to predict either of the two given classes. For example: classifying the "gender" as male or female, predicting the "result" as to win or loss,etc.

- **MultiClass Classification**:- Here we have to classify the data into three or more classes. For example: classifying a "movie's genre" as comedy, action, or romantic, classifying "fruits" like oranges, apples, pears, etc.

Loan prediction is a very common real-life problem that each retail bank faces at least once in its lifetime. If done correctly, it can save a lot of man-hours at the end of a retail bank.

Although this course is specifically built to give you a walkthrough of the Loan Prediction problem, you can always refer to the content to get a comprehensive overview to solve a classification problem.

Conclusion:

… The **conclusion** derived from such assessments helps banks and other financial institutions

… **CONCLUSION** In this paper, various algorithms were implemented to **predict loan** defaulters. …

This **conclusion** follows from the first and third columns of the table, which show that … creditstandards helped **predict loan** growth in both periods and that the total effect of **loan** growth on …

Source code:

Loan_prediction.i
pynb # -*- coding:
utf-8 -*-
"""Loan Prediction.ipynb

Automatically generated by Colaboratory.

Original file is located at
https://colab.research.google.com/drive/14vOj9-
kHfGlwsIWLtWkC1lCJhlQC
SlD3 """

```
# Commented out IPython magic to ensure
Pythoncompatibility.
import pandas
as pdimport
numpy as np

import pickle


import matplotlib.pyplot
as plt# %matplotlib inline
import seaborn as
snsimport sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import
GradientBoostingClassifier,RandomForestClassif
ier from sklearn.neighbors import
KNeighborsClassifierfrom
sklearn.model_selection import
RandomizedSearchCV
import imblearn
from sklearn.model_selection
importtrain_test_split
```

```
from sklearn.preprocessing import
StandardScaler from sklearn.metrics import
accuracy_score,classification_report,confusion_
mat rix, f1_score

# Commented out IPython magic to ensure
Pythoncompatibility.
from google.colab import
drive
drive.mount('/content/driv
e')

# %cp '/content/drive/MyDrive/Colab
Notebooks/loan_prediction.csv' '/content/'

#importing the dataset which is in csv
file                data                =
pd.read_csv('loan_prediction.csv')
data

data.info

#fining rthe sum of null values un each
columndata.isnull().sum()
```

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

data['Marrieed'] =
data['Married'].fillna(data['Married'].mode()
[0])

#replacing + with space for filling the non values
data['Dependents']=data['Dependents'].str.repl
ace('
+','')

data['Dependents']=data['Dependents'].fillna(dat
a[' Dependents'].mode()[0])

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

```
data['LoanAmount'] =
data['LoanAmount'].fillna(data['LoanAmount'].m
od e()[0])

data['Loan_Amount_Term'] =
data['Loan_Amount_Term'].fillna(data['Loan_Am
ou nt_Term'].mode()[0])

data['Credit_History'] =
data['Credit_History'].fillna(data['Credit_History']
.m ode()[0])

#changing th datatype of each float column to
int
data['Gender']=data['Gender'].astype('int64')
data['Married']=data['Married'].astype('int64')
data['Dependents']=data['Dependents'].astype
('int6 4')
data['Self_Employed']=data['Self_Employed'].asty
pe ('int64')

data['CoapplicantIncome']=data['CoapplicationIn
come'].astype('int64')
```

```
data['LoanAmount']=data['LoanAmount'].astype('
in t64')
data['Loan_Amount_Term']=data['Loan_Amount
_T erm'].astype('int64')
data['Credit_History']=data['Credit_History'].asty
pe( 'int64')

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

SMOTETomeksmote =

SMOTETomek(0.90)

#dividing the dataset into dependent
andindependent y and x respectively
y = data['Loan_Status']
x = data(columns=['Loan_Status'],axis=1)

#creating a new x and y variables for the
balncedset
x_bal,y_bal, = smote.fit_resample(x,y)
```

```
#printing the values of y before balancing the
dataand after
print(y.value_counts())
print(y_bal.value_counts())

data.describe()

#plotting the using
displot
plt.figure(figsize=(12,5
) plt.subplot(121)
sns.distplot(data['ApplicantIncome'],
color='r')plt.subplot(122)
sns.distplot(data[Credit_History'])
plt.show


#plotting the count plot
plt.figure(figsize=(18,4))
plt.subplot(1,4,1)
sns.countplot(data['Gender'])
plt.sublot(1,4,2)
```

```
sns.countplot(data['Education'])
plt.show()

#visualsing two colunms againist each other

plt.figure(figsize=(20,5))
plt.subplot(131)
sns.countplot(data['Married'],hue=data['Gender']
) plt.subplot(132)
sns.countplot(data['Self_Employed'],hue=data['E
du cation'])
plt.subplot(133)
sns.countplot(data['Property_Area'],hue=data['L
oan
_Amount_Term'])


#visulaized based gender and income what
would be the application status
sns.swarmplot(data['Gender'],data['ApplicantInc
om e'],hue = data['Loan_Status'])

#perfroming feature scaling operation using
standard scaller on x part of the dataset
```

because#there different type of values in the colunms sc=StandardScaler()
x_bal=sc.fit_transform(x_bal)

x_bal = pd.DataFrame(x_bal,colunms=names)

#splitting the dataset in train and test on balnmceddataset
x_train, x_test, y_train, y_test = train_test_split(x_bal, y_bal, test_size=0.33,random_state=42)

```
def
   decisionTree(x_train,x_test,y_train,y_te
   st)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))
```

```
def randomForest(x_train, x_test, y_train,
   y_test):rf = RandomForestClassifier()
   rf.fit(x_train,y_train)
   yperd = rf.predict(x_test)
   print('***RandomForestClassifier**
   *') print('Confusion matrix')
   print(confusion_matrix(y_test,ypre
   d))print('Classification report')

   print(classification_report(y_test,ypred))


def KNN(x_train, x_test, y_train,
   y_test):knn =
   KNeighborsClassifier()
   knn,fit(x_train,y_train)
   ypred = knn.predict(x_test)
   print('***KNeighborsClassifier***')
   print('confusion  matrix')
   print('Confusion matrix')
   print(confusion_matrix(y_test,ypred)
   ) print('Classification report')
   print(classification_report(y_test,ypr
   ed))
```

```
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)

#Importing the keras libraries and packages

import imblearn.tensorflow


from tensorflow.keras.models import
Sequentialfrom tensorflow.keras.layers
import Dense

#Initialising the
ANN classifier =
Sequential()
```

#Adding the input layer and the first hidden layer classifier.add(Dense(units=100, activation='relu',input_dim=11))

#Adding the second hidden layer classifier.add(Dense(units=50,activation='rel u'))

#Adding the output layer classifier.add(Dense(units=1,activation='sigmo id'))

#compiling the ANN classifier.compile(optimizer='adam',loss='binary' _cr ossentropy',metrics=['accuracy'])

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

#Gender Married Dependents Education Self_Employed Application CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History Property_Area
dtr.predict([[1,1,0,1,1,4276,1542,145,240,0,1
]])

#Gender Married Dependents Education Self_Employed ApplicantIncome CoapplicantIncomeLoanAmount Loan_Amount_Term Credit_History Property_Area
rfr.Predict([[1,1,0,1,1,4276,1542,145,240,0,1]])

#Gender Married Dependents Education Self_Employed ApplicantIncome CoapplicantIncomeLoanAmount Loan_Amount_Term Credit_History property_Area
knn.predict([[1,1,0,1,1,4276,145,240,0,1]])

#Gender Married Dependents Education Self_Employed ApplicantIncome

```
CoapplicantIncomeLoanAmount
Loan_Amount_Term Credit_History
property_Area
xgb.predict([[1,1,0,1,1,4276,145,240,0,1]])

# Commented out IPython magic to ensure
Pythoncompatibility.
classifier.save("loan.

h5") # %cp

'/content/loan.h5'

#predicting the Test set rsults
y_pred =
classifier.predict(x_test)

y_pred

y_pred = (y_pred >
0.5)y_pred
```

```
def  predit_exit(sample_value):
  sample_value = np.array(sample_value)
  sample_value =
  sample_value.reshape(1,-1)
  sample_value =
  sc.transform(sample_data) return
  classifier.predict(sample_value)

sample_value =
[[1,1,0,1,1,4276,1542,145,240,0,1]]
if predict_exit(sample_values)>0.5:
  print('prediction:High chance of Loan
  Approval!')

else:


  print('prediction:Low chance Loan Approval.')


sample_value=[[1,0,1,1,45,14,45,240,1,1]]
if predict_exit(sample_value)>0.5:
print('prediction:High chance of Loan
Approval!') else:
  print('prediction:Low chance of Loan Approval.')


def
  compareModel(x_train,x_test,y_train,y_test)
```

```
:decisionTree(x_train,x_test,y_train,y_test)
print('_'*100)
RandomForest(x_train,x_test,y_train,y_test)
print('_'*100)
xGB(x_train,_test,y_train,y_test)
print('_'*100)
KNN(x_train.x_test,y_train,y_test)
# print('_'*100_)

#compareModel(x_train,x_test,y_train,y_test_)

#ypred = classifier.predict(x_test)
#print(accuracy_score(y_pred,y_test))
#print("ANN Model")
#print("Confusion_Matrix")
#print(confusion_matrix(y_test,y_pre
d)) #print("Classification Report")
#print(classification_report(y_test,y_p
red))

from sklearn.model_selection import cross_val_score

#rf =
RandomForestClassifier()
#rf.fit(x_train,y_train)
#ypred = rf.predict(x_test)

#f1_score(ypred,y_test,average='wei
```

ghted') #cv =

cross_val_score(rf,x,y,cv=5)

#np.mean(cv)

#saving the model by using pickle
funtion
#pickle.dump(model,open('rdf.pkl','
wb'))

app.py

-*- coding: utf-8 -*-
"""app.py

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1cKrF6Vhi LVh1wOgorY67UX4kFHZkTFys

"""

```
from flask import Flask,
render_template,requestimport numpy as np
import pickle


!pip install pyngrok
from pyngrok import ngrok

# Commented out IPython magic to ensure
Pythoncompatibility.
from google.colab import
drive
drive.mount('/content/driv
e')

# %cp -r
'/content/drive/MyDrive/predicting
personal loan approval using machine
learning/Flask/templates/' '/content/'

app = Flask(__name__)
ngrok.set_auth_token("2OGw99kjZxSOAJbHZ
yWvR wcBw4U_6ixdhHh61sGML3gSQT91K")
```

```
#model = pickle.load(open(r'rdf.pkl',
'rb'))                    #scale                    =
pickle.load(open(r'scale.pkl',         'rb'))
public_url = ngrok.connect(5000)

print(public_url)


@app.route('/') #rendering the html
templatedef home():
    return render_template('home.html')


@app.route('/submit',methods=["POST","GET"])
#ro ute to show the prediction in a web UI
def submit():

  # reading  the inputs given by the
userinput_faeature=[int(x)for x in
request.form.values() ]
  #input_feature =
  np.transpose(input.feature)
```

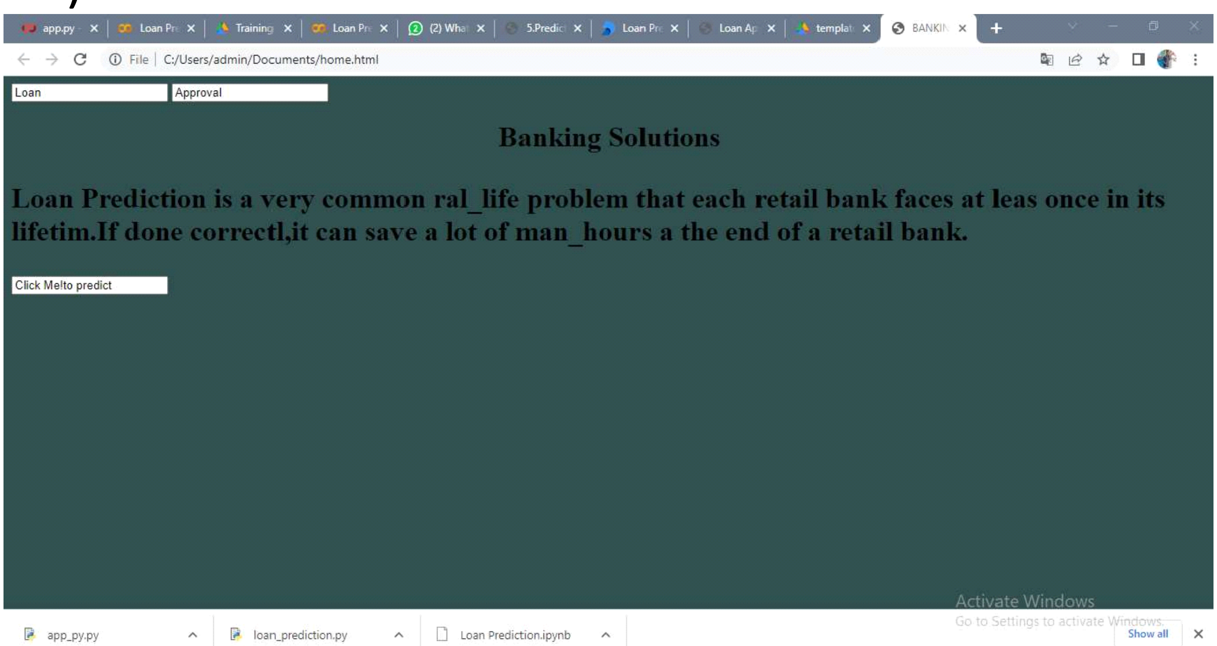
```
 input_feature-[np.arry(input_feature)]
 print(input_feature)
 names = ['Gender', 'Married',
'Dependents', 'Education', 'Self_Employed',
'ApplicantIncome','CoapplicantIncome',
'Loan_Amount_Term', 'Credit_History',
'Property_Area']
 data =
pandas.DataFrame(input_feature,columns=nam
es) print(data)
 #data_scaled = scale.fit_transform(data)
 #data =
 pandas.DataFrame(,columns=names)

 #predictions using the loaded model
 fileprediction=mode.predict(data)
 print(prediction)
 prediction =
 int(prediction)
 print(type(prediction))
 if(prediction == 0):

   return render_template("output.html",result =
```

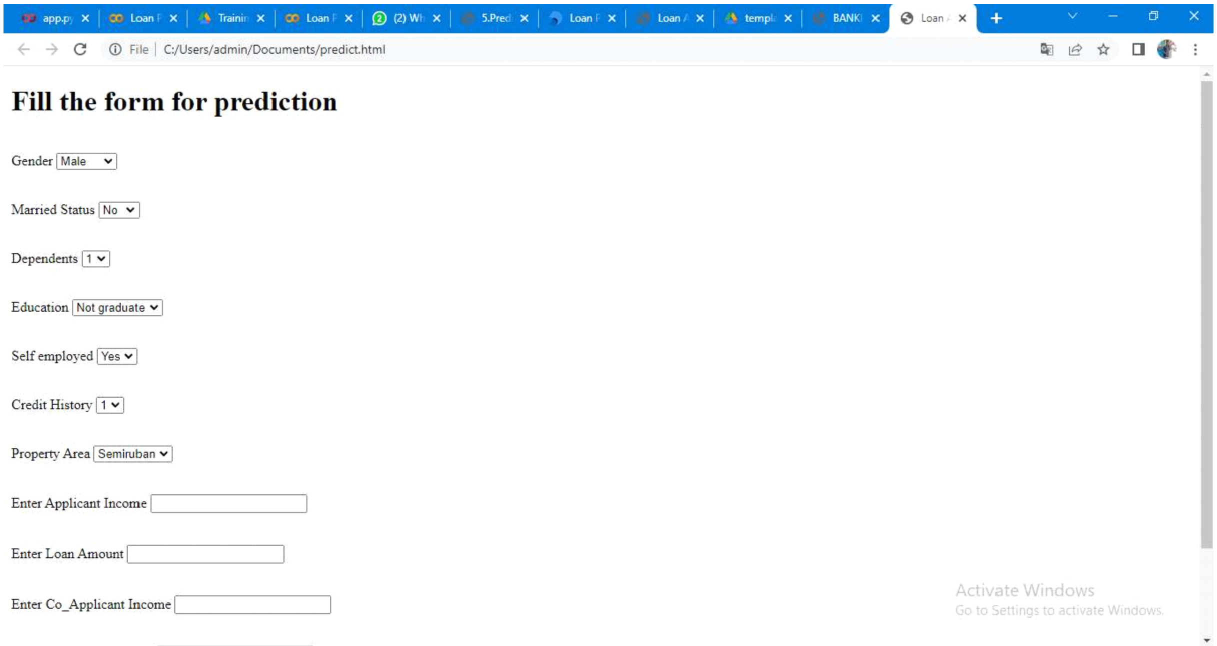
"Loan will Not beelse:

Approved")

return render_template("output.html",result ="Loan will be Aproved")

app.run(debug=False) Result:





C:/Users/admin/Documents/predict.html

Fill the form for prediction

Gender Male

Married Status No

Dependents 1

Education Not graduate

Self employed Yes

Credit History 1

Property Area Semiruban

Enter Applicant Income

Enter Loan Amount

Enter Co_Applicant Income