PREDICTION LOAN APPROVAL

INDRODUCTION:

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

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

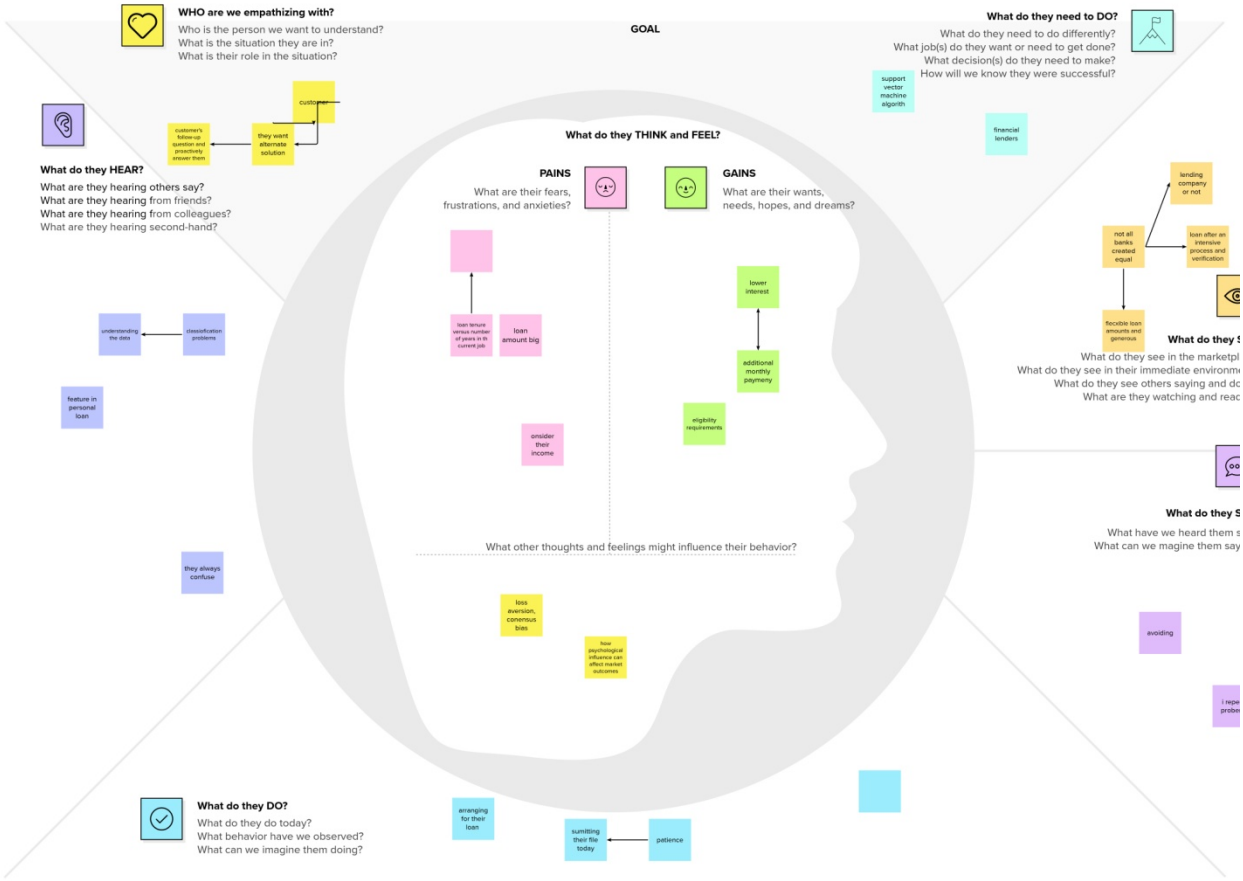
*To automate this process, they have providd a dataset to identify the customer segments that are eligible for loan amounts so that they can specifically target these customer.

Purpose:

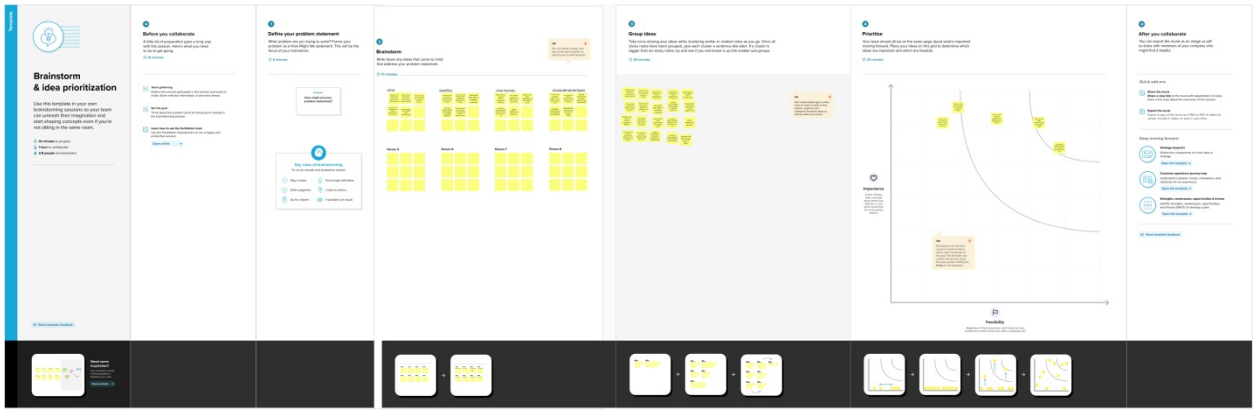
*The prediction model not only helps the applicant but also helps the bank by minimizing the risk and reducing the number of defaulters.

*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


WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?
GOAL
What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?
support vector machine algorith
financial lenders
customer
they want alternate solution
What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?
What do they THINK and FEEL?
PAINS
What are their fears, frustrations, and anxieties?
GAINS
What are their wants, needs, hopes, and dreams?
lending company or not
not all banks created equal
loan after an intensive process and verification
flecible loan amounts and generous
What do they see in the marketpl
What do they see in their immediate environm
What do they see others saying and do
What are they watching and reac
understanding the data
classification problems
feature in personal loan
loan amount big
lower interest
additional monthly paymeny
eligibility requirements
onsider their income
What other thoughts and feelings might influence their behavior?
What have we heard them s
What can we magine them say
they always confuse
loss aversion, conensus bias
how psychological influence can effect market outcomes
avoiding
What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?
arranging for their loan
sumitting their file today
patience

Brain Strom:


Brainstorm
& idea prioritization
Use this template to your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.
Before you collaborate
Define your problem statement
Brainstorm
Group ideas
Prioritize
After you collaborate

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 pay will 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 shares of its own as security to secure the finance

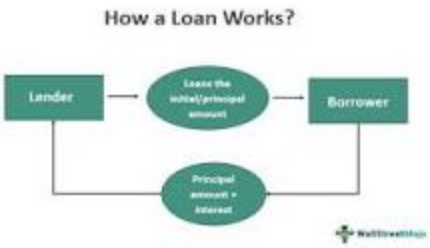
What is the advantage of loan portfolio

Portfolio lenders **focus more on cash flow and the individual's business history rather than the 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 could have 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**. The bank labels these loans as nonperforming assets (NPA). It can occur with either a commercial loan 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 as for the applicant also

Application:

Loan Prediction Project using Machine Learning in Python

1. Understanding the various features (columns) of the dataset: ...
2. Understanding Distribution of Categorical Variables: ...
3. Outliers of LoanAmount and Applicant Income: ...
4. Data Preparation for Model Building: ...
5. Generic Classification Function: ...
6. Model Building:

*We have data of some predicted loans from history. So when there is name 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 practice knowledge 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 … credit standards helped **predict loan** growth in both periods and that the total effect of **loan** growth on …

Source code:

Loan_prediction.ipynb

```
# -*- coding: utf-8 -*-
"""Loan Prediction.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/14vOj9-kHfGlwsIWLtWkC1lCJhlQCSlD3
"""

# Commented out IPython magic to ensure Python compatibility.
import pandas as pd
import numpy as np
```

```
import pickle
import matplotlib.pyplot as plt
# %matplotlib inline
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.model_selection import
train_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 Python
compatibility.
from google.colab import drive
drive.mount('/content/drive')
```

```
# %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 column
data.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.replace('
+','')

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'].mod
e()[0])

data['Loan_Amount_Term'] =
data['Loan_Amount_Term'].fillna(data['Loan_Amou
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'].astype
('int64')
```

```
data['CoapplicantIncome']=data['CoapplicationInco
me'].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'].astype(
'int64')

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

smote = SMOTETomek(0.90)

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

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

```
#printing the values of y before balancing the data
and 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['Edu
cation'])
plt.subplot(133)
sns.countplot(data['Property_Area'],hue=data['Loan
_Amount_Term'])

#visulaized based gender and income what would
be the application status
sns.swarmplot(data['Gender'],data['ApplicantIncom
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 balnmced
dataset
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_test)
    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,ypred))
    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,ypred))

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 Sequential
from 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='relu'))

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

#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 CoapplicantIncome LoanAmount 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 CoapplicantIncome LoanAmount 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 CoapplicantIncome LoanAmount 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 Python compatibility.
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_pred))
#print("Classification Report")
#print(classification_report(y_test,y_pred))
```

```
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='weighted')

#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/1cKrF6VhiLVh1wOgorY67UX4kFHZkTFys
```

"""

from flask import Flask, render_template,request
import numpy as np
import pickle

!pip install pyngrok
from pyngrok import ngrok

# Commented out IPython magic to ensure Python compatibility.
from google.colab import drive
drive.mount('/content/drive')

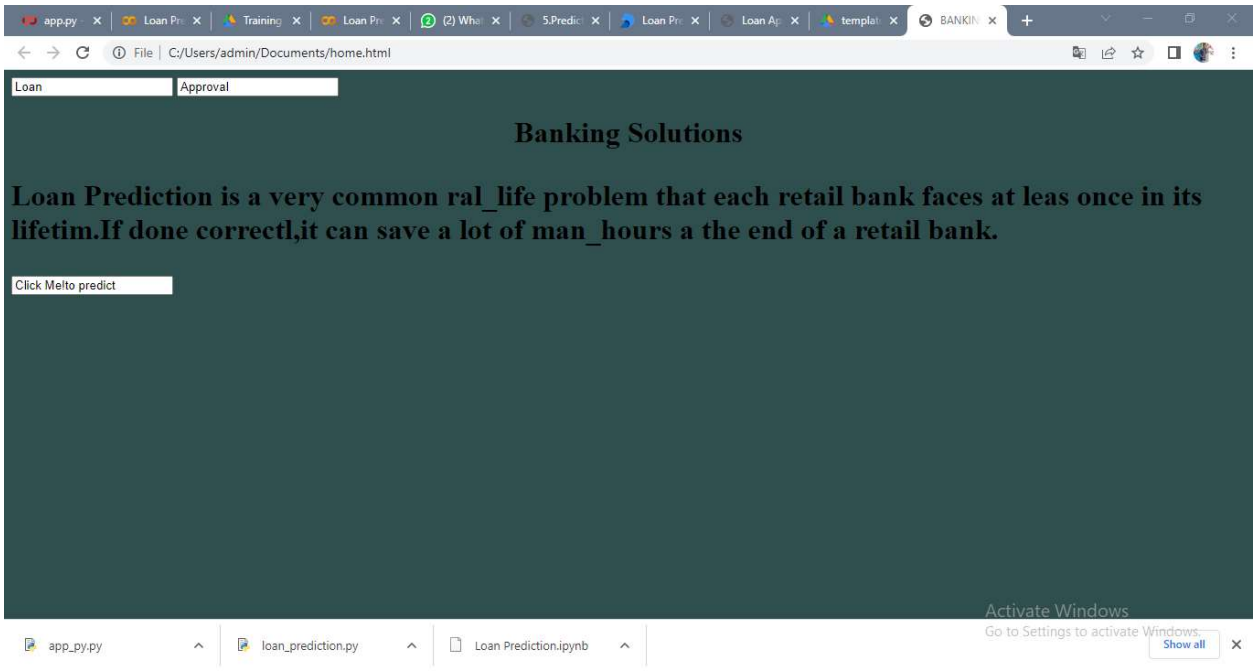
%cp -r '/content/drive/MyDrive/predicting personal loan approval using machine learning/Flask/templates/' '/content/'

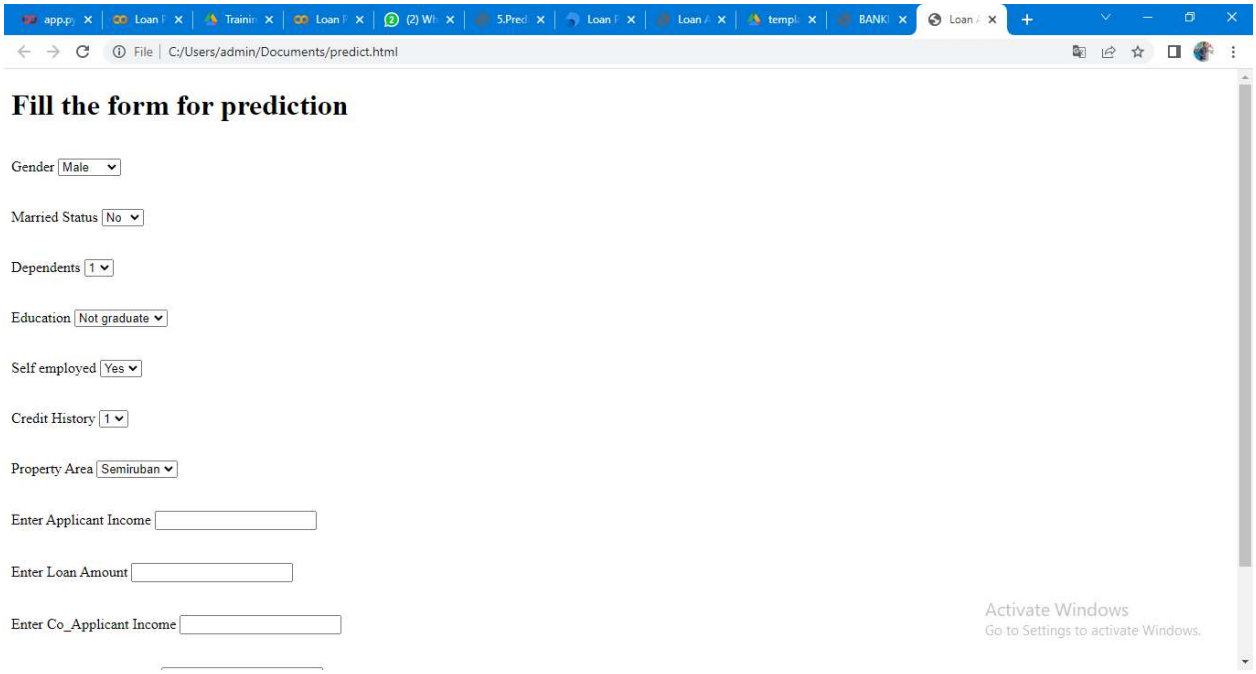
app = Flask(__name__)
ngrok.set_auth_token("2OGw99kjZxSOAJbHZyWvRwcBw4U_6ixdhHh61sGML3gSQT91K")
#model = pickle.load(open(r'rdf.pkl', 'rb'))
#scale = pickle.load(open(r'scale.pkl', 'rb'))
public_url = ngrok.connect(5000)

```
print(public_url)

@app.route('/') #rendering the html template
def 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 user
  input_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=names)
  print(data)
  #data_scaled = scale.fit_transform(data)
  #data = pandas.DataFrame(,columns=names)
```

```
  #predictions using the loaded model file
  prediction=mode.predict(data)
  print(prediction)
  prediction = int(prediction)
  print(type(prediction))
  if(prediction == 0):
     return render_template("output.html",result =
"Loan will Not be Approved")
  else:
     return render_template("output.html",result =
"Loan will be Aproved")

app.run(debug=False)
```

Result:




File | 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
Activate Windows
Go to Settings to activate Windows.