

**SMART LENDER -APPLICANT**  
**CREDIBILITY PREDICTION FOR LOAN**  
**APPROVAL USING IBM WATSON**

# **1. INRODUCTION**

## **1.1 Overview**

One of the most important factors which affect our country's economy and financial condition is the credit system governed by the banks. The process of bank credit risk evaluation is recognized at banks across the globe. "As we know credit risk evaluation is very crucial, there is a variety of techniques are used for risk level calculation. In addition, credit risk is one of the main functions of the banking community.

The prediction of credit defaulters is one of the difficult tasks for any bank. But by forecasting the loan defaulters, the banks definitely may reduce their loss by reducing their non-profit assets, so that recovery of approved loans can take place without any loss and it can play as the contributing parameter of the bank statement. This makes the study of this loan approval prediction important. Machine Learning techniques are very crucial and useful in the prediction of these types of data.

We will be using classification algorithms such as Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment

## **1.2 Purpose**

Loans have made our life easier, providing us the financial leverage that extends beyond our earnings. Be it Credit Card, Home Loan, Personal Loan or Auto Loan etc. loans are the credit extended to us by lenders on fulfilling certain key parameters. However, getting a loan in India can often be a tedious process for the un-initiated, but not for individuals with a good credit score. Whenever you apply for a loan, banks check your CIBIL Score and Report to evaluate your credit history and credit worthiness. The higher your score the better are the chances of your loan application getting approved.

# **2. LITERATURE SURVEY**

## **2.1 Existing problem**

Anomaly detection relies on individuals' behaviour profiling and works by detecting any deviation from the norm. When it is used for online banking fraud detection, it suffers from three disadvantages. First, for an individual, the historical behaviour data are often too limited for profiling his/her behaviour pattern. Second, because of the

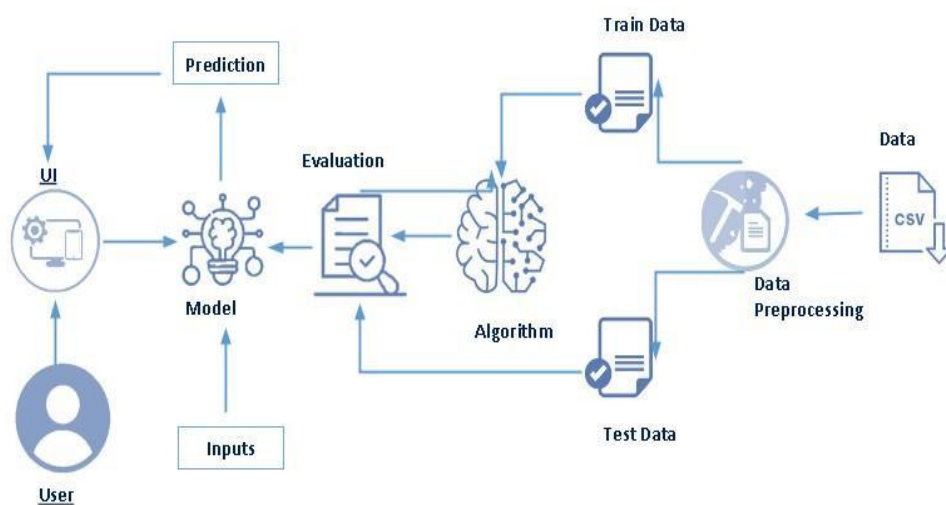
heterogeneous nature of transaction data, there is no uniform treatment to various attribute values, which will become a potential barrier for development of the model and for further usage. Third, the transaction data are highly skewed, and it becomes a challenge for utilizing the label information effectively. Anomaly detection often suffers from poor generalization ability and a very high false alarm rate. We argue that individuals limited historical data for behaviour profiling and fraud data's highly skewed nature could account for this defect. Since it is straightforward to use information from other similar individuals, similarity measurement itself becomes a great challenge due to heterogeneous nature of attribute values.

## 2.2 Proposed problem

In our proposed system, we combine datasets from different sources to form a generalized dataset and use four machine learning algorithms such as Random forest, Logistic regression, Decision tree and Naive bayes algorithm on the same dataset. The dataset we collected for predicting given data is split into training set and test set in the ratio of 7:3. The data model which was created using Machine learning algorithms are applied on training set and based on maximum test result from the four algorithms, the test set prediction is done using the algorithm that has maximum performance. After that, we deploy the model using Flask Framework.

## 3. THEORITICAL ANALYSIS

### 3.1 BLOCK DIAGRAM



## **3.2 HARDWARE AND SOFTWARE REQUIREMENTS**

### **3.2.1 SOFTWARE REQUIREMENTS**

- **A Device with a constant internet connection.**

- **IBM Cloud**

IBM Cloud provides solutions that enable higher levels of compliance, security, and management, with proven architecture patterns and methods for rapid delivery for running mission-critical workloads

- **IBM Watson**

IBM Watson is AI for business. Watson helps organizations predict future outcomes, automate complex processes, and optimize employees' time.

- **Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It was created by Guido van Rossum, and first released on February 20, 1991. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

- **Anaconda Navigator**

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS. Conda is an open-source, cross-platform, package management system. Anaconda comes with so very nice tools like JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code. For this project, we will be using Jupyter notebook and Spyder.

- **Jupyter Notebook**

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter. Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

- **Spyder Spyder**

The Scientific Python Development Environment, is a free integrated development environment (IDE) that is included with Anaconda. It includes editing, interactive testing, debugging, and introspection features.

- **Tensor flow**

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers can easily build and deploy ML-powered applications.

- **Keras**

Keras leverages various optimization techniques to make high-level neural network API easier and more performant. It supports the following features: Consistent, simple, and extensible API.

- **Flask Web**

framework used for building. It is a web application framework written in python which will be running in local browser with a user interface. In this application, whenever the user interacts with UI and selects emoji, it will suggest the best and top movies of that genre to the user.

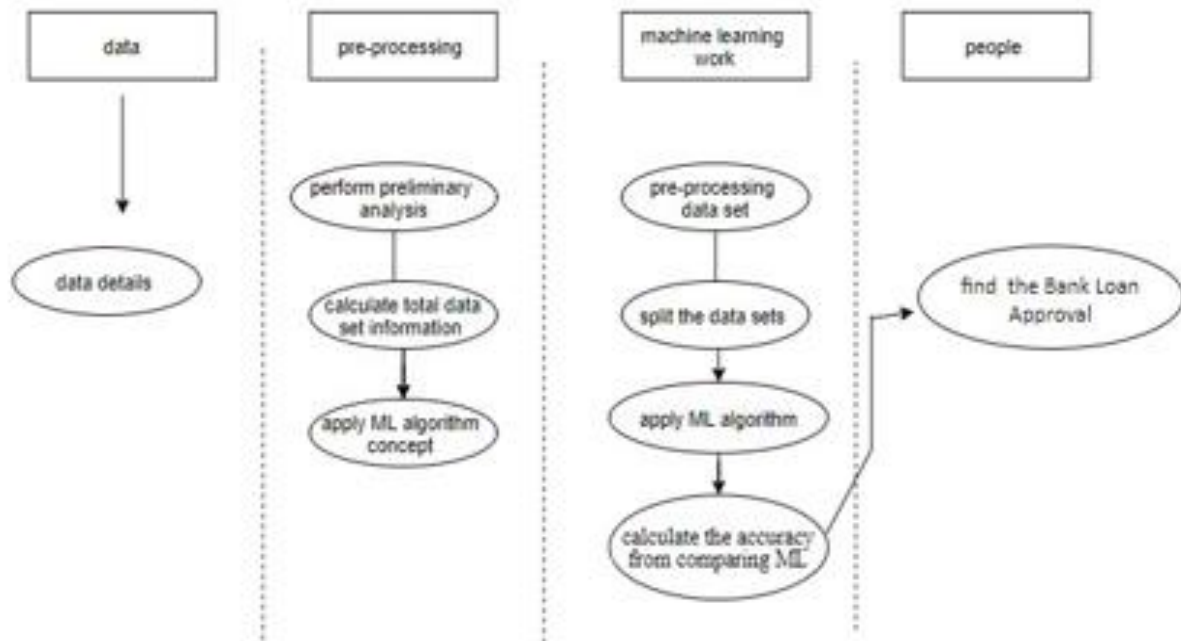
### **3.2.2 HARDWARE REQUIREMENTS**

- o Operating system: window 7 and above with 64bit
- o Processor Type -Intel Core i3-3220
- o RAM: 4Gb and above
- o Hard disk: min 100GB

## **4. EXPERIMENTAL INVESTIGATIONS**

The dataset is obtained by gathering lot of required datasets and combining them to produce a generalised dataset. The dataset thus produced is pre-processed i.e., the dataset is cleaned before doing data visualization. Then the four algorithms are applied on the same pre-processed dataset and calculated for the best performed algorithms among them. Then the best algorithm is used to train the model and test it to check how accurate the algorithm can predict the output. Then we deploy that model to predict if bank loan can be approved or not for a specific candidate

## 5. FLOWCHART



## 6.RESULT

← → ↺ ⚙ 127.0.0.1:5000

### Welcome To Loan Prediction

Loan Approval is based on lot of this rather than going to a bank and getting rejected.we made it simple that you can get your loan approval prediction by our machine learning model for to predict, we need some of your information.

[Predict](#)

← → ↺ ⚙ 127.0.0.1:5000/predict

Graduate	▼
Self Employed	
Yes	▼
Applicant Income	
15000	
CO Applicant Income	
15000	
Loan Amount	
45000	⬇
Loan Amount Term	
1500	
Credit History	
1	
Property Area	
Urban	▼
<input type="button" value="Submit"/>	

# Loon Approval Prediction

Loan will Not be Approved



## Welcome To Loan Prediction

Loan Approval is based on lot of this rather than going to a bank and getting rejected.we made it simple that you can get your loan approval prediction by our machine learning model for to predict, we need some of your information.

[Predict](#)



← → ↺ 127.0.0.1:5000/predict

Graduate

Self Employed

Yes

Applicant Income

15000

CO Applicant Income

15000

Loan Amount

25000

Loan Amount Term

150

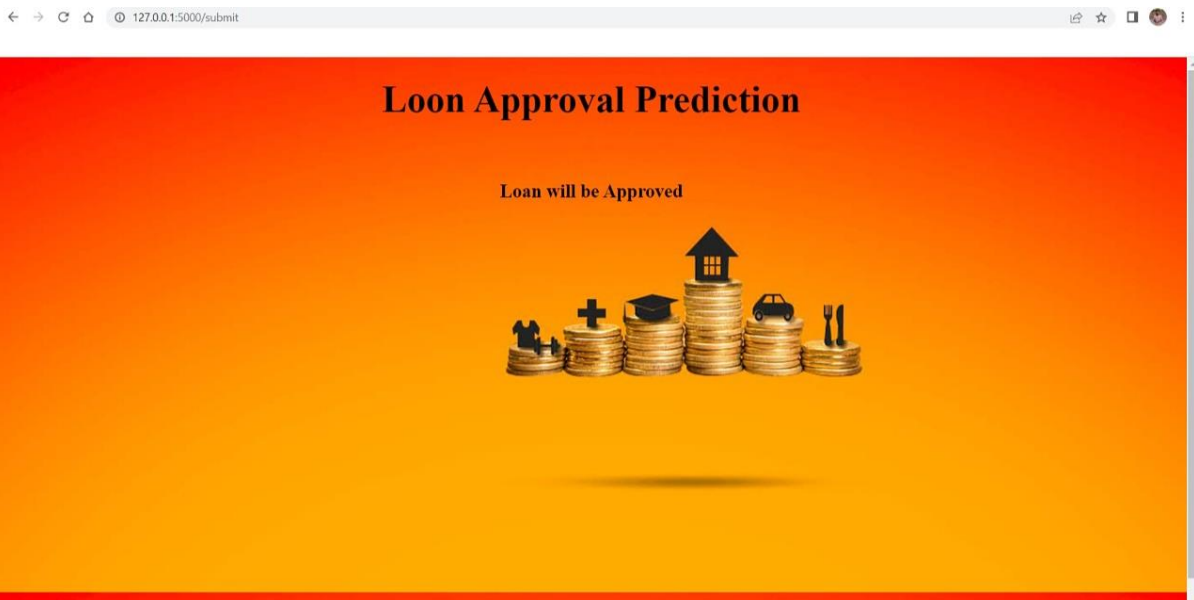
Credit History

1

Property Area

Urban

Submit



## **7. ADVANTAGES & DISADVANTAGES**

### **7.1 Advantages**

- Easy to use.
- Time efficient.
- Cost efficient.
- Safe to use
- Improve the quality of site investigations by calling for the minimum input data as classification parameters.
- Enable better engineering judgement and more effective communication on a project

It can provide special advantages to the bank. The Loan Prediction System can automatically calculate the weight of each features taking part in loan processing and on new test data same features are processed with respect to their associated weight .

### **7.2 Disadvantages**

- 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.
- 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

## **8. APPLICATIONS**

By using data from previous loan applications,

- tax returns, bank statements
- machine learning models can learn to identify patterns that are predictive of loan default.

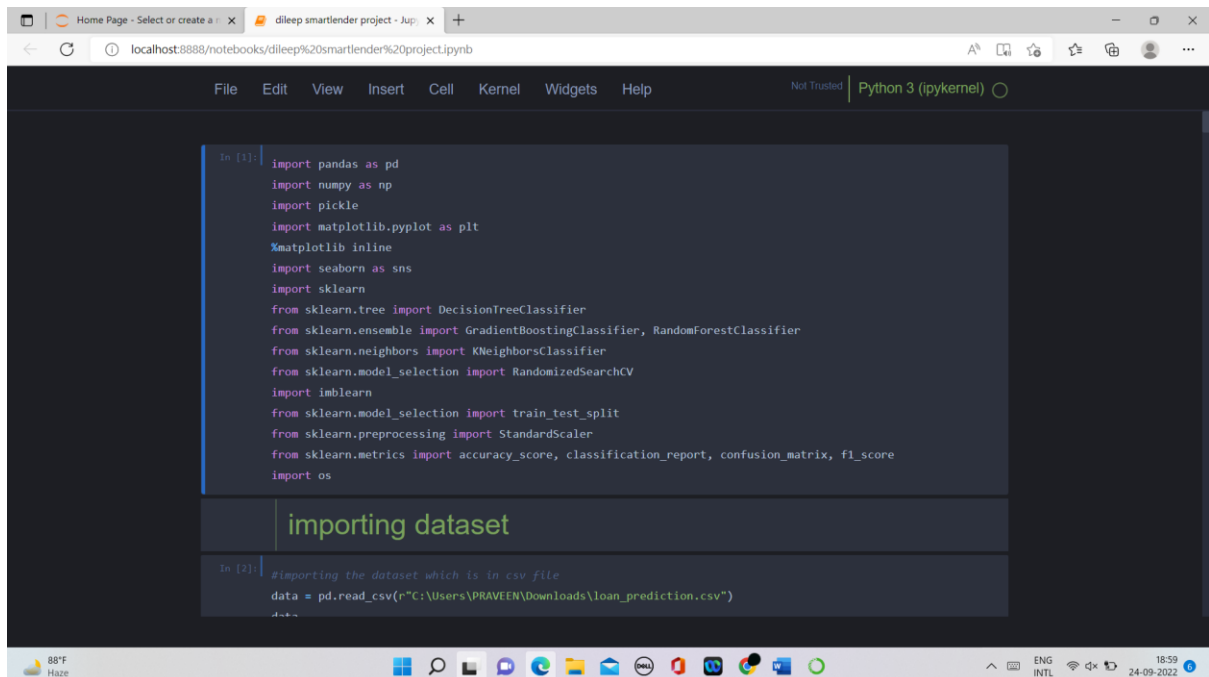
## 9. CONCLUSION

The analysis starts from data cleaning and processing missing value, exploratory analysis and finally model building and evaluation of the model. The best accuracy on public test set is when we get higher accuracy score and other performance metrics which will be found out. This paper can help to predict the approval of bank loan or not for a candidate.

## 10. FUTURE HOPE

In future, this model can be used to compare various machine learning algorithm generated prediction models and the model which will give higher accuracy will be chosen as the prediction model.

## 11. APPENDIX



The screenshot displays a Jupyter Notebook environment. The top bar shows the file name 'dileep smartlender project - Jupyter' and the kernel 'Python 3 (ipykernel)'. The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar. The code is written in a dark-themed editor. The first cell, labeled 'In [1]:', contains a series of import statements for various Python libraries. The second cell, labeled 'In [2]:', contains a comment and a line of code to read a CSV file.

```
In [1]: 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_matrix, f1_score
import os

importing dataset

In [2]: #importing the dataset which is in csv file
data = pd.read_csv(r"C:\Users\PRAVEEN\Downloads\loan_prediction.csv")
data
```





```
9 Credit_History    614 non-null    float64
10 Property_Area    614 non-null    int64
11 Loan_Status      614 non-null    int64
dtypes: float64(7), int64(4), object(1)
memory usage: 37.7+ KB

In [20]: #changing the datatype of each float column to int
data['Gender'] = data['Gender'].astype('int64')
data['Married'] = data['Married'].astype('int64')
data['Dependents'] = data['Dependents'].astype('int64')
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')

In [21]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   Gender                 614 non-null   int64
1   Married                614 non-null   int64
2   Dependents             614 non-null   int64
3   Education              614 non-null   int64
4   Self_Employed          614 non-null   int64
5   ApplicantIncome        614 non-null   int64
6   CoapplicantIncome      614 non-null   int64
```

```
In [16]: data["Loan_Amount_Term"] = data["Loan_Amount_Term"].fillna(data["Loan_Amount_Term"].mode()[0])

In [17]: data["Credit_History"] = data["Credit_History"].fillna(data["Credit_History"].mode()[0])

In [18]: data.isnull().sum()

Gender                0
Married               0
Dependents            0
Education             0
Self_Employed         0
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount            0
Loan_Amount_Term      0
Credit_History        0
Property_Area         0
Loan_Status           0
dtype: int64

In [19]: #getting bthe total info of the data after performing categorical to numerical and replacing missing value
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   Gender                 614 non-null   float64
1   Married                614 non-null   float64
```

```
9 Credit_History    614 non-null    float64
10 Property_Area    614 non-null    int64
11 Loan_Status      614 non-null    int64
dtypes: float64(7), int64(4), object(1)
memory usage: 37.7+ KB

In [20]: #changing the datatype of each float column to int
data['Gender']=data['Gender'].astype('int64')
data['Married']=data['Married'].astype('int64')
data['Dependents']=data['Dependents'].astype('int64')
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')

In [21]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype  
---  --
0   Gender                 614 non-null   int64   
1   Married                614 non-null   int64   
2   Dependents             614 non-null   int64   
3   Education              614 non-null   int64   
4   Self_Employed          614 non-null   int64   
5   ApplicantIncome        614 non-null   int64   
6   CoapplicantIncome      614 non-null   int64
```

```
x = data.drop(columns=['Loan_Status'],axis=1)

In [30]: #shape of x after seperating from the total data set
x.shape

(614, 11)

In [31]: #shape of y
y.shape

(614,)

In [32]: #creating a new x and y variables for the balanced set
x_bal,y_bal = smote.fit_resample(x,y)

In [33]: #printing the values of y before balancing the data and after
print(y.value_counts())
print(y_bal.value_counts())

1    422
0    192
Name: Loan_Status, dtype: int64
1    358
0    315
Name: Loan_Status, dtype: int64
```

Home Page - Select or create a... x dileep smartlender project - Jup... x Student Dashboard x SI-133321-1661496027 x Telegram Web x + -

localhost:8886/notebooks/dileep%20smartlender%20project.ipynb

Dell Gmail YouTube Maps sathyabama lms Sathyabama Place...

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

## scaling the dataset

```
In [35]: # performing feature Scaling operation using standard scaler on X part of the dataset because
# there different type of values in the columns
sc=StandardScaler()
x_bal=sc.fit_transform(x_bal)

In [36]: x_bal = pd.DataFrame(x_bal,columns=names)

In [37]: #splitting the dataset in train and test on balncced dataset
X_train, X_test, y_train, y_test = train_test_split(
    x_bal, y_bal, test_size=0.33, random_state=42)

In [38]: X_train.shape

(450, 11)

In [39]: X_test.shape

(223, 11)

In [40]: y_train.shape, y_test.shape
```

88°F Haze 20:48 24-09-2022

Home Page - Select or create a... x dileep smartlender project - Jup... x Student Dashboard x SI-133321-1661496027 x Telegram Web x + -

localhost:8886/notebooks/dileep%20smartlender%20project.ipynb

Dell Gmail YouTube Maps sathyabama lms Sathyabama Place...

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
In [36]: x_bal = pd.DataFrame(x_bal,columns=names)

In [37]: #splitting the dataset in train and test on balncced dataset
X_train, X_test, y_train, y_test = train_test_split(
    x_bal, y_bal, test_size=0.33, random_state=42)

In [38]: X_train.shape

(450, 11)

In [39]: X_test.shape

(223, 11)

In [40]: y_train.shape, y_test.shape

((450,), (223,))
```

## model building

```
In [41]: #importing and building the random forest model
def RandomForest(X_train,X_test,y_train,y_test):
    model = RandomForestClassifier()
```

88°F Haze 20:50 24-09-2022



model building

```
In [41]: #importing and building the random forest model
def RandomForest(X_train,X_test,y_train,y_test):
    model = RandomForestClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))

In [42]: #printing the train accuracy and test accuracy respectively
RandomForest(X_train,X_test,y_train,y_test)

1.0
0.8609865470852018

In [43]: #importing and building the Decision tree model
def decisionTree(X_train,X_test,y_train,y_test):
    model = DecisionTreeClassifier()
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
```

Battery saver

Battery saver is on  
Consider plugging in your device.

88°F  
Haze

20:52  
24-09-2022

```
#printing the train accuracy and test accuracy res
RandomForest(X_train,X_test,y_train,y_test)

1.0
0.8165137614678899
```

```
'max_features': 'log2',
'max_depth': 10,
'criterion': 'entropy'})

In [58]: bt_score

0.7933333333333333

In [59]: # training and test the xg boost model on the best parameters got from the randomized cv
def RandomForest(X_train,X_test,y_train,y_test):
    model = RandomForestClassifier(verbose= 10, n_estimators= 120, max_features= 'log2',max_depth= 10,criteri
    model.fit(X_train,y_train)
    y_tr = model.predict(X_train)
    print(accuracy_score(y_tr,y_train))
    yPred = model.predict(X_test)
    print(accuracy_score(yPred,y_test))

In [60]: model = RandomForestClassifier(verbose= 10, n_estimators= 120, max_features= 'log2',max_depth= 10,criteri
model.fit(X_train,y_train)

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done 1 out of 1 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=1)]: Done 2 out of 2 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=1)]: Done 3 out of 3 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=1)]: Done 4 out of 4 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=1)]: Done 6 out of 6 | elapsed: 0.0s remaining: 0.0s
```

88°F  
Haze

21:01  
24-09-2022

Home Page - Select x dileep smartlender x Student Dashboard x SI-133321-166149 x Writer - Extension x bvt1Q48GZk5X04G x Telegram Web x

localhost:8886/notebooks/dileep%20smartlender%20project.ipynb

Dell Gmail YouTube Maps sathyabama lms Sathyabama Place...

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

203	-0.429953	0.870161	0.383532	0.627196	-0.335256	-0.331763	-0.514931	-0.582777	0.276204
196	-0.429953	0.870161	-0.687494	0.627196	-0.335256	-0.261068	-0.514931	-1.264330	-2.473804
...	...	...	...	...	...	...	...	...	...
71	-0.429953	0.870161	1.454559	-1.594398	-0.335256	-0.328590	0.151161	-0.544913	0.276204
106	2.325838	-1.149213	-0.687494	0.627196	-0.335256	-0.265299	-0.514931	-0.620641	0.276204
270	-0.429953	0.870161	2.525585	0.627196	-0.335256	-0.150705	-0.451985	-0.229379	0.276204
435	-0.429953	0.870161	2.525585	0.627196	2.982794	0.084654	-0.040673	-0.532291	0.276204
102	2.325838	-1.149213	-0.687494	0.627196	-0.335256	-0.181733	-0.514931	-1.239087	0.276204

458 rows x 11 columns

### saving the model

```
In [63]: #saving the model by using pickle function
pickle.dump(model,open('rdf.pkl','wb'))

In [ ]:
```

88°F Haze

21:03 24-09-2022