TEAM LEADER:ANANTHI.M

TEAM MEMBER:KAVIYA.K

TEAM MEMBER:KALAISELVI.V

TEAM MEMBER:ROJA.S

Subject	NALAIYATHIRAN
Date	19 November 2022
Team Id	PNT2022TMID39808
Project Name	Smart Lender- Applicant Credibility Prediction for Loan Approval

SmartLender – Applicant Credibility Prediction for Loan Approval

Introduction

A loan is the major source of income for the banking sector of financial risk for banks. Large portions of a bank's assets directly come from the interest earned on loans given. The activity of lending loans carry great risks including the inability of borrower to pay back the loan by the stipulated time. It is referred as "credit risk". A candidate's worthiness for loan approval or rejection was based on a numerical score called "credit score". Therefore, the goal of this paper is to discuss the application of different Machine Learning approach which accurately identifies whom to lend loan to and help banks identify the loan defaulters for much-reduced credit risk.

Literature Survey

In [1] they have used only one algorithm; there is no comparison of different algorithms. The algorithm used was Logistic Regression and the best accuracy they got was 81.11%. The final conclusion reached was only those who

have a good credit score, high income and low loan amount requirement will get their loan approved. Comparison of two machine learning algorithms was made in [2]. The two algorithms used were two class decision jungle and two class decision and their accuracy were 77.00% and 81.00% respectively. Along with these they also calculated parameters such as Precision, recall, F1 score and AUC. The [3] shows a comparison of four algorithms. The algorithms used were Gradient Boosting, Logistic Regression, Random Forest and CatBoost Classifier. Logistic Regression gave a very low accuracy of 14.96%. Random forest gave a good accuracy of 83.51%. The best accuracy we got was from CatBoost Classifier of 84.04%. There was not much difference between Gradient Boosting and CatBoost Classifier in terms of accuracy. Accuracy of Gradient Boosting was 84.03%. Logistic Regression, Support Vector Machine, Random Forest and Extreme Gradient Boosting algorithms are used in [4]. The accuracy percentage didn't vary a lot between all the algorithms. But the support vector Machine gave the lowest variance. The less the variance, the less is the fluctuation of scores and the model will be more precise and stable. Only the K Nearest Neighbor Classifier is used in [5]. The process of Min-Max Normalization is used. It is a process of decomposing the attributes values. The highest accuracy they got was 75.08% when the percentage of dataset split was 50-50% with k to be set as 30. Logistic Regression is the only algorithm used.
DESCRIPTION: Data mining techniques are becoming very popular nowadays because of the wide availability of huge quantity of data and the need for transforming such data into knowledge. Data mining techniques are implemented in various domains such as retail industry, biological data analysis, intrusion detection, telecommunication industry and other scientific applications. Techniques of data mining are also be used in the banking industry which help them compete in the market well equipped. In this paper, they introduced a prediction model for the bankers that will help them predict the credible customers who have applied for a loan. Decision Tree Algorithm is being applied to predict the attributes relevant for credibility. A prototype of the model has been described in this paper which can be used by the organizations for making the right decisions to approve or reject the loan request from the customers.

Advantages

1. Performance and accuracy of the algorithms can be calculated and compared.
2. Class imbalance can be dealt with machine learning approaches.

Disadvantages

1. They had proposed a mathematical model and machine learning algorithms were not used.
2. Class Imbalance problem was not addressed and the proper measure were not taken.

Visualizing And Analyzing The Data
Importing The Libraries

I am Import the necessary libraries as shown in the image Import the required libraries for the model to run. The first step is usually importing the libraries that will be needed in the program.

```
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
```

Data Pre-Processing

Handling Categorical Values

As we can see our dataset has categorical data we must convert the categorical data to integer encoding or binary encoding.

To convert the categorical features into numerical features we use encoding techniques. There are several techniques but in our project, we are using manual encoding with the help of list comprehension.

- In our project, Gender , married, dependents, self-employed, co applicants income, loan amount , loan amount term, credit history With list comprehension encoding is done.

In [22]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
oneh = OneHotEncoder()
data['Education'] = le.fit_transform(data['Education'])
data.head()
```

Out[22]:

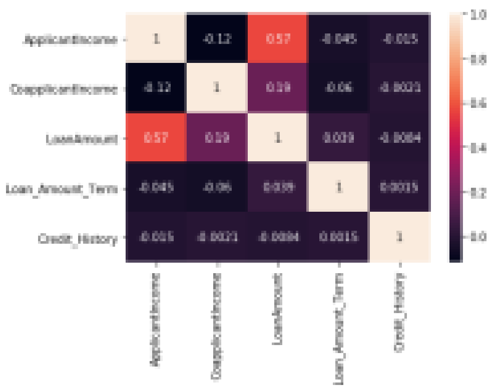
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	1	No	0	0	No	5849	0.0	NaN	360.0	1.0
1	LP001003	1	Yes	1	0	No	4583	1508.0	128.0	360.0	1.0
2	LP001005	1	Yes	0	0	Yes	3000	0.0	66.0	360.0	1.0
3	LP001006	1	Yes	0	1	No	2583	2358.0	120.0	360.0	1.0
4	LP001008	1	No	0	0	No	6000	0.0	141.0	360.0	1.0

Visualizing And Analyzing The Data

Multivariate Analysis

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used swarm plot from seaborn package.

```
In [6]: sns.heatmap(data.corr(), annot = True)
Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x257d639cf60>
```



From the above graph we are plotting the relationship between the Gender, applicants income and loan status of the person

Data Pre-Processing Splitting Data Into Train And Test

Now let’s split the Dataset into train and test sets

Changes: first split the dataset into x and y and then split the data set

Here x and y variables are created. On the x variable, df is passed by dropping the target variable. And on y target variable is passed. For splitting training and testing data, we are using the train_test_split() function from sklearn. As parameters, we are passing x, y, test_size, and random_state.

In [38]:
```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, OneHotEncoder,StandardScaler
le = LabelEncoder()
oneh = OneHotEncoder()
sc = StandardScaler()
data['Gender'] = le.fit_transform(data['Gender'])
data['Loan_ID'] = le.fit_transform(data['Loan_ID'])
data.head()
x = data.iloc[0:5, 0:2]
x_scaled = sc.fit_transform(x)
x_scaled
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size = 0.1, random_state = 0)
x_train
```

Visualizing And Analyzing The Data

Reading The Dataset

- Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.
- In pandas, we have a function called read_csv() to read the dataset. As a parameter, we have to give the directory of the CSV file.

In [2]:
```python
import pandas as pd
data = pd.read_csv(r"C:\Users\ELCOT\Downloads\Dataset\loan_prediction.csv")
data
```

Out[2]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_Histo
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1
5	LP001011	Male	Yes	2	Graduate	Yes	5417	4196.0	267.0	360.0	1
6	LP001013	Male	Yes	0	Not Graduate	No	2333	1516.0	95.0	360.0	1
7	LP001014	Male	Yes	3+	Graduate	No	3036	2504.0	158.0	360.0	0
8	LP001018	Male	Yes	2	Graduate	No	4006	1526.0	168.0	360.0	1

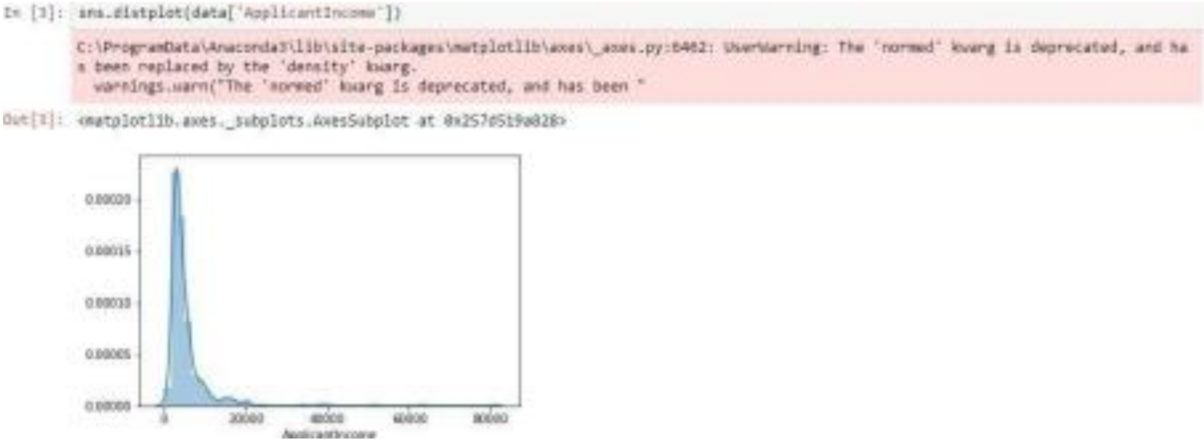
In [ ]:

Visualizing And Analyzing The Data

Uni-Variate Analysis

In simple words, univariate analysis is understanding the data with single feature. Here we have displayed two different graphs such as distplot and countplot.

- Seaborn package provides a wonderful function distplot. With the help of distplot, we can find the distribution of the feature. To make multiple graphs in a single plot, we use a subplot.



- In our dataset, we have some categorical features. With the count plot function, we are going to count the unique category in those.

Features. We have created a dummy data frame with categorical features. With for loop and subplot, we have plotted the below graph. • From the plot we came to know, Applicants' income is skewed towards the left side, whereas credit history is categorical with 1.0 and 0.0

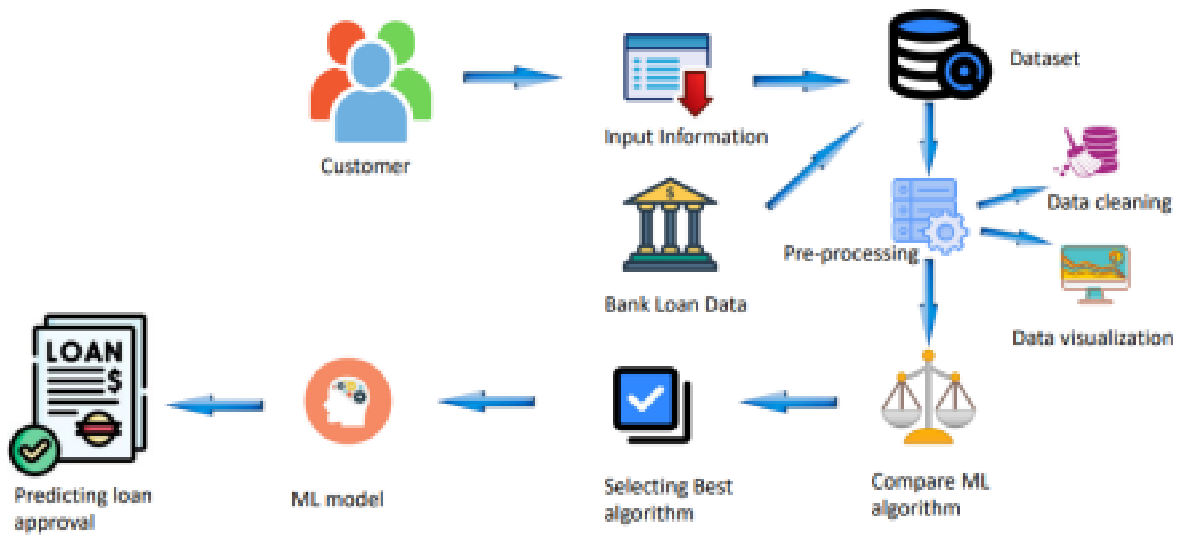
PROJECT DESIGN PHASE - I

SOLUTION ARCHITECTURE

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.

Solution Architecture



Project Design Phase-I

Proposed Solution

Proposed Solution

S.No.	Parameter	Description

1.	Problem Statement (Problem to be solved)	Loan Prediction is very helpful for employee of banks as well as for the applicant also. The aim of this Search is to provide quick, immediate and easy way to choose the deserving applicants. 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 best data same features are processed with respect to their associated weight.
2.	Idea / Solution description	A loan is a form of debt incurred by an individual or other entity. The lender— usually a corporation, financial institution, or government—advances a sum of money to the borrower. In return, the borrower agrees to a certain set of terms including any finance charges, interest, repayment date, and other conditions.
3.	Novelty / Uniqueness	The machine learning model uses several data points to make an accurate prediction of the credit eligibility of the person.
4.	Social Impact / Customer Satisfaction	Using credit score as a basis to judge a individuals loan taking capacity makes our country a credit based society and such a society has spending power

5.	Business Model (Revenue Model)	The AI based prediction model can be monetized by a subscription model that charges banks and other financial institutions a fee
6.	Scalability of the Solution	AI models can be easily scaled and software as a service(Saas). This software can be banking app

Setting Up Application Environment

Creating IBM Cloud Account

Step 1:

Type or click the below link in the browser
https://www.ibm.com/academic/home

Step 2:

Click register icon and give all the details one by one,

Step 3:

Already you got SI EMAIL and SI PASSWORD from your IBM student login. Login to your web mail account. Wait for 7-digit verification code in your web mail: https://sg2plmcpnl496936.prod.sin2.secureserver.net:2096/

Step 4:

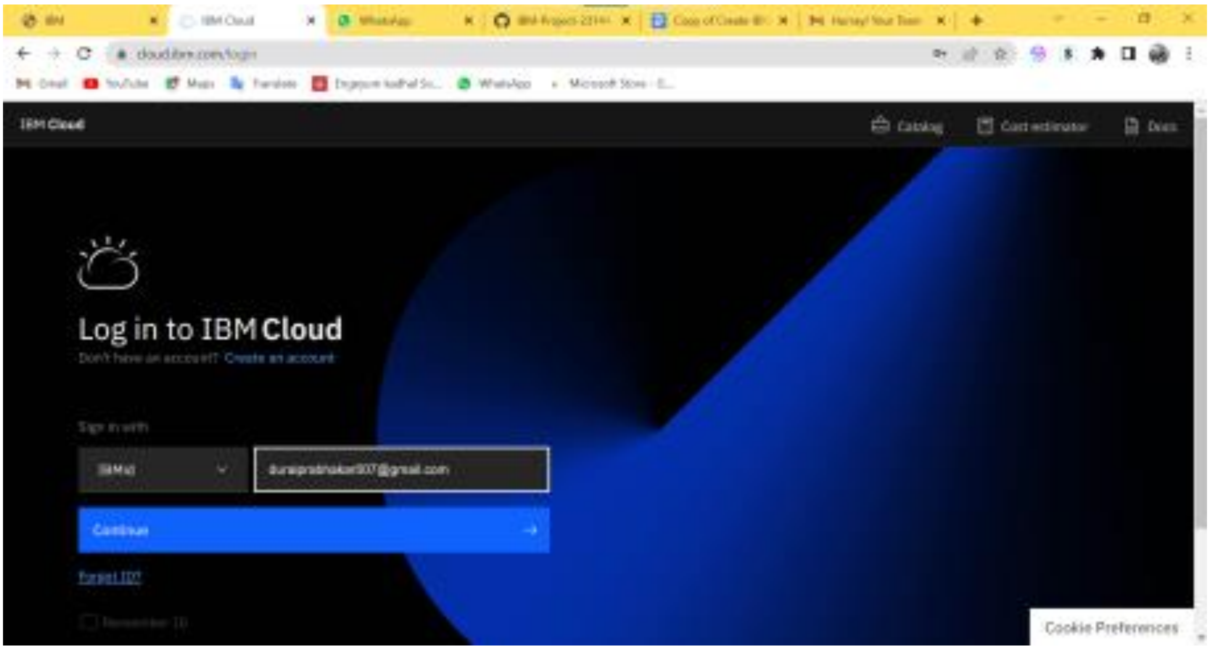
Get 7-digit verification code in webmail, then go to registration page and enter the verification code.

Step 5:

Now, you have successfully created the IBM Cloud Account.

Step 6:

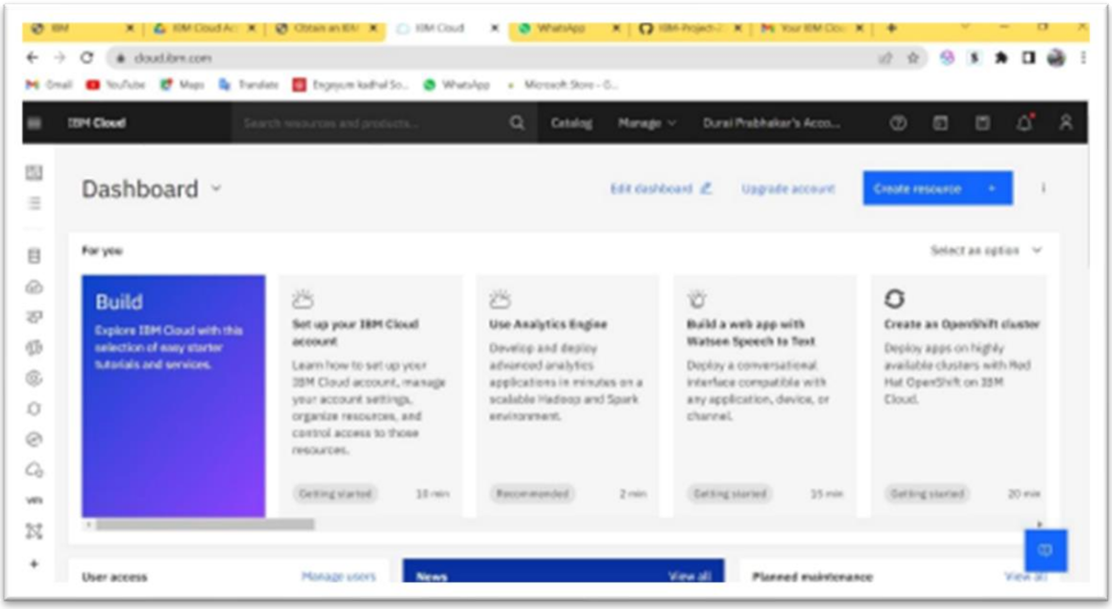
Go to IBM Cloud login page: [***https: //cloud.ibm.com/login***](https://cloud.ibm.com/login) ,then enter your IBM Id and Password.



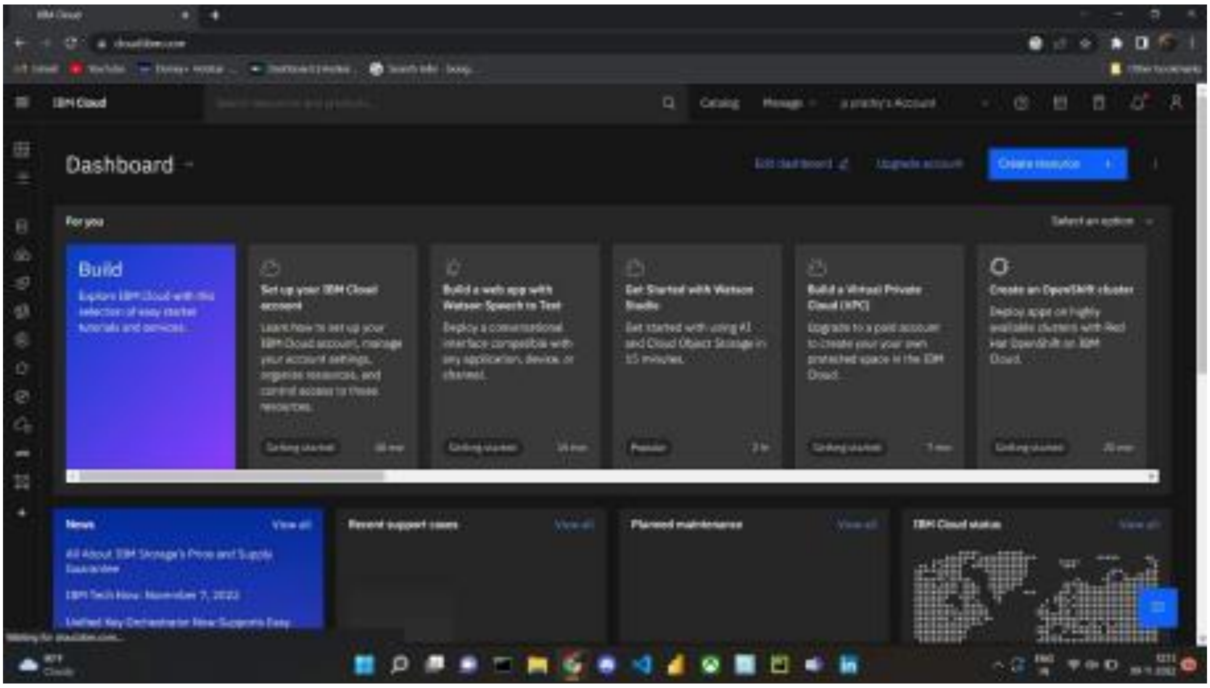
Step 7:

Finally, you will access the IBM cloud account.

Durai's Cloud Account



Prathy's Cloud Account



Project Design Phase-I - Solution Fit Template

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Our target customers are mostly banking firm, small financial firms that lends out loan and credit card companies because of the increasing rate of loan defaulter and also to increase the slow process of the loan approval.

6. CUSTOMER CONSTRAINTS CC

Banks are not to correctly handle the loan request. People within a protected class being clearly treated differently than those of non-protected classes for loan. There is an increasing rate of loan defaults. Banks identify the loan defaulters for much-reduced credit risk as large portions of a bank's assets directly come from the interest earned on loans given.

5. AVAILABLE SOLUTIONS AS

- Random forest, Logistic regression, Decision tree and Naive bayes algorithm are used
- Using data pre-processing data mining and data filtering
- Algorithms such as naïve bayes, k-nearest neighbors are used.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Needs to Support genuine Entrepreneur. That the process should be easier a time saving. To find an applicant which can give best interest. Needs to find a loan applicant with good credit score

9. PROBLEM ROOT CAUSE RC

The root cause of this problem is the banks identify the loan defaulters for much-reduced credit risk as large portions of a bank's assets directly come from the interest earned on loans given. . People within a protected class being clearly treated differently than those of non-protected classes for loan.

7. BEHAVIOUR BE

Directly related:
The customers who lends the loan and the banks that checks the credibility seek to do the process faster.

Indirectly associated:
The small finance sector that deals with middle class and poor class people seek to find the credibility.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

The slow and complex process of loan approval is affecting the business of our customer and it also decline the revenue of our customers. Due to the sudden surge in the number of loan defaulters our customers business is highly affected.

4. EMOTIONS: BEFORE / AFTER EM

Before:
Needs to Support genuine Entrepreneur. That the process should be easier a time saving. To find an applicant which can give best interest. Needs to find a loan applicant with good credit score.

After:
After implementing this project people can be able to face all these above-mentioned problems easily

10. YOUR SOLUTION SL

- There is an increasing rate of loan defaulters and banks are not able to correctly handle the loan request. To avoid this problem a machine learning algorithm is developed
- The system automatically selects the credible candidates to approve the loan and it will improve the speed, efficacy, and accuracy of loan approval processes.
- This help the user(Lender) to accurately identify whom to lend the loan and also help the banks to identify the loan defaulter for much-reduced credit risk.

8. CHANNELS OF BEHAVIOUR CH

ONLINE:
The customers needs to check the credibility of the client in an online mode.

OFFLINE:
The customer need to install the Machine Learning algorithm in their system to work efficiently.

Identify strong TR & EM

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sma
from statsmodels.stats.outliers_influence import
        variance_inflation_factor
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score

from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
import warnings
warnings.filterwarnings('ignore')
```

1. Download the dataset: Dataset

2. Load the dataset into the tool.

In [4]:
```
df = pd.read_csv("E:\\IBM projects Assignment Sona
College\\abalone.csv")
```

3. Perform Below Visualizations. · Univariate Analysis

· Bi-Variate Analysis

· Multi-Variate Analysis

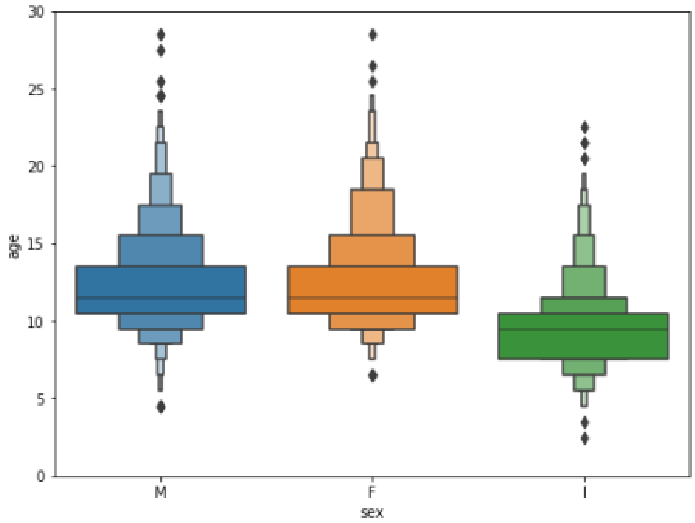
In [5]:
```
#rename output variable
df.rename(columns={"Sex":"sex", "Length":"length",
"Diameter":"diameter",  "Height":"height", "Whole
weight":"whole_weight",  "Shucked weight":"shucked_weight",
"Viscera  weight":"viscera_weight",
 "Shell weight":"shell_weight", "Rings":"rings"}, inplace =
True)
```

In [6]:
```
df[df['height'] == 0] #need to drop these rows.
df.drop(index=[1257,3996], inplace = True)
df.shape
```

Out[6]:
```
(4175, 9)
```

In [7]:
```
df['age'] = df['rings']+1.5 #AS per the
problem statement df.drop('rings', axis = 1,
inplace = True)
df.head()
#categorical features
temp = pd.concat([df['age'], df['sex']], axis=1)
```

```
f, ax = plt.subplots(figsize=(8, 6))
fig = sns.boxenplot(x='sex', y="age", data=df)
fig.axis(ymin=0, ymax=30);
```



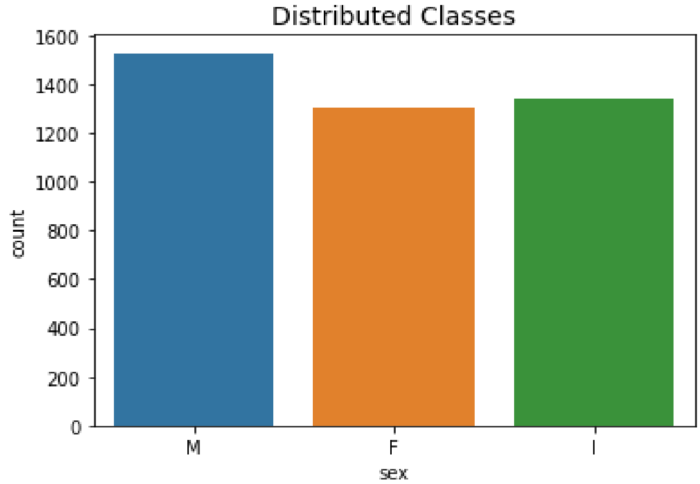
ANALYSIS

There is no difference in age of rings for male and female (8-19). But in infants, it lies between (5-10)

Count Plot

In [8]:

```
sns.countplot('sex', data=df)
plt.title('Distributed Classes', fontsize=14)
plt.show()
```



Histograms: Understanding the Distribution of the Numerical Features

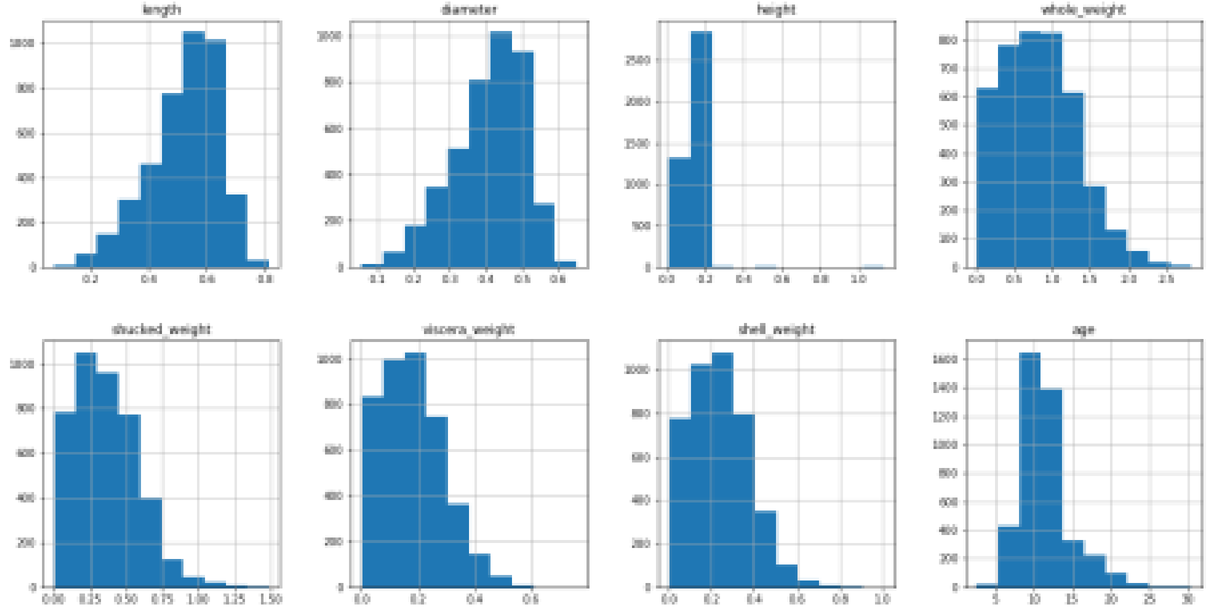
In [9]:

```
df.hist(figsize = (20,10), layout = (2,4))
```

Out[9]:

```
array([[,
 ,
 ,
 ],
```

```
[,
,
,
]], dtype=object)
```



ANALYSIS

- Skewness of the height is too high. (need to normalise later...)
- Need to check skewness for all varibles

Skewness of the Variables

In [10]:
```
df.skew().sort_values(ascending = False)
```

Out[10]:
```
height 3.166364
age 1.113754
shucked_weight 0.718735
shell_weight 0.621081
viscera_weight 0.591455
whole_weight 0.530549
diameter -0.610182
length -0.640993
dtype: float64
```

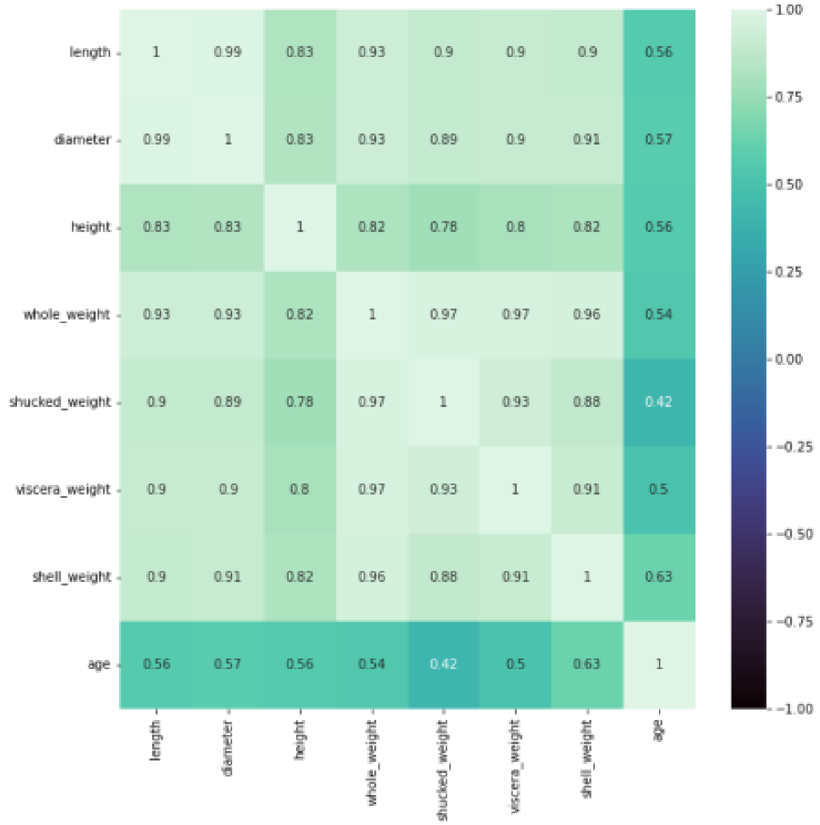
ANALYSIS:

• Skewness is close to 0 for Normal distribution curve.
• Height has the highest skewness of 3.17.
• May be there are outliers in height, we need to check that and remove them before modeling. • Will check the coorelation with the dependent variable (Rings)
• Will use IQR algorithm to remove outliers.

Coorelation Plot

In [11]:

```
corr = df.corr()
plt.figure(figsize = (10,10))
ax = sns.heatmap(corr, vmin = -1, center = 0, annot = True, cmap = 'mako')
```



ANALYSIS

- No Negative correlation found
- High coorelation between Length & Diameter
- High corelation between shucked weight, viscera weight Vs Whole_weight & Shell weight vs Whole_weight

In [12]:

```
upper_tri = corr.where(np.triu(np.ones(corr.shape),k=1).astype(np.bool))
columns_to_drop = [column for column in upper_tri.columns if any(upper_tri[column] > 0.95)] #highly correlated variables to be removed.

print("Columns to drop:\n", columns_to_drop)
```

```
Columns to drop:
['diameter', 'shucked_weight', 'viscera_weight', 'shell_weight']
```

ANALYSIS

- We will remove the above columns, before proceeding any further.

4. Perform descriptive statistics on the dataset.

In [13]:
```
df.head()
```
Out[13]:

se x
lengt h
diamete r
heigh t
whole_weigh t
shucked_weigh t
viscera_weigh t
shell_weigh

t age

0 M 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.150 16.5 **1** M 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.070 8.5 **2** F 0.530 0.420 0.135 0.6770 0.2565 0.1415

0.210 10.5 **3** M 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.155 11.5

4 I 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.055 8.5

In [14]:
```
df.shape
```
Out[14]:
```
(4175, 9)
```
In [15]:
```
df.describe()
```
Out[15]:

length diameter height whole_wei ght
shucked_wei ght
viscera_wei ght
shell_wei
ght age
cou nt mean
4175.000 000
4175.000 00
4175.000 000
4175.0000
00 4175.000000 4175.000000 4175.000000 4175.000 000
n 0.524065 0.40794 0.139583 0.829005 0.359476 0.180653 0.238834 11.435090 **std** 0.120069 0.09922 0.041725 0.490349 0.221954 0.109605
0.139212 3.224227 **min** 0.075000 0.05500 0.010000 0.002000 0.001000 0.000500 0.001500 2.500000
shell_wei
ght age 25
length diameter height whole_wei ght
shucked_wei ght
viscera_wei ght
% 0.450000 0.35000 0.115000 0.442250 0.186250 0.093500 0.130000 9.500000 **50**
% 0.545000 0.42500 0.140000 0.800000 0.336000 0.171000 0.234000 10.500000 **75**
% 0.615000 0.48000 0.165000 1.153500 0.502000 0.253000 0.328750 12.500000 **max** 0.815000 0.65000 1.130000 2.825500 1.488000 0.760000 1.005000 30.500000

In [16]:
```
df.info()
```
```
Int64Index: 4175 entries, 0 to 4176
Data columns (total 9 columns):
```

```
 # Column Non-Null Count Dtype
--- ------ -------------- -----
 0 sex 4175 non-null object
 1 length 4175 non-null float64
 2 diameter 4175 non-null float64
 3 height 4175 non-null float64
 4 whole_weight 4175 non-null float64
 5 shucked_weight 4175 non-null float64
 6 viscera_weight 4175 non-null float64
 7 shell_weight 4175 non-null float64
 8 age 4175 non-null float64
dtypes: float64(8), object(1)
memory usage: 326.2+ KB
```

5. Check for Missing values and deal with them.

In [17]:

```
df[df.duplicated()]
```

Out[17]:

sex length diameter height whole_weight shucked_weight viscera_weight shell_weight age

In [18]:

```
df.isna().sum()
```

Out[18]:

```
sex 0
length 0
diameter 0
height 0
whole_weight 0
shucked_weight 0
viscera_weight 0
shell_weight 0
age 0
dtype: int64
```

there is no missing values and duplicates in dataframe

6. Find the outliers and replace them outliers

In [19]:

```
for i in df:
 if df[i].dtype=='int64' or df[i].dtypes=='float64':
 q1=df[i].quantile(0.25)
 q3=df[i].quantile(0.75)
 iqr=q3-q1
 upper=q3+1.5*iqr
 lower=q1-1.5*iqr
 df[i]=np.where(df[i] >upper, upper, df[i])
 df[i]=np.where(df[i] <lower, lower, df[i])
```

After removing outliers, boxplot will be like

In [20]:

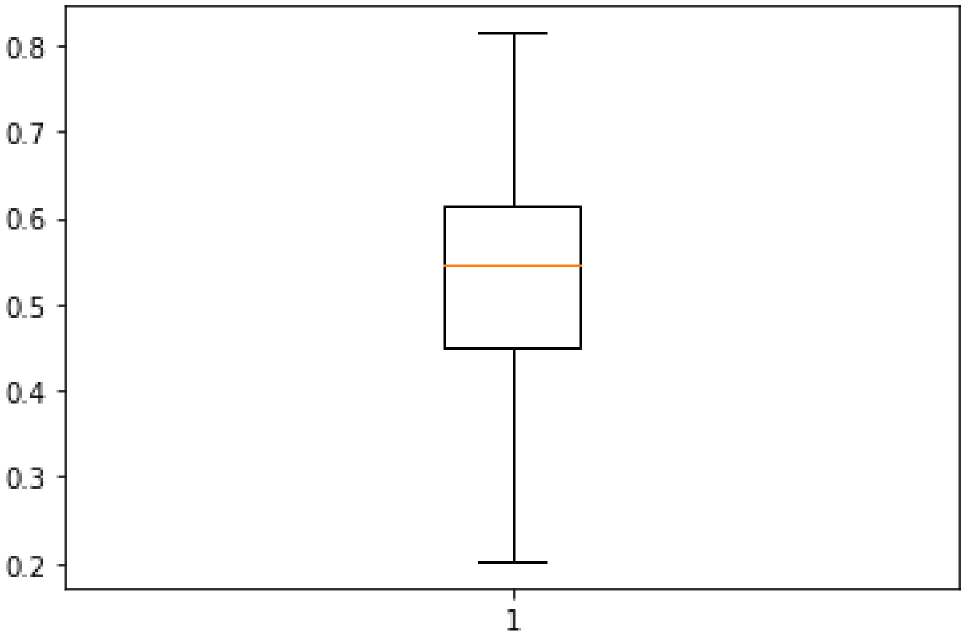
```
import matplotlib.pyplot as mtp
```

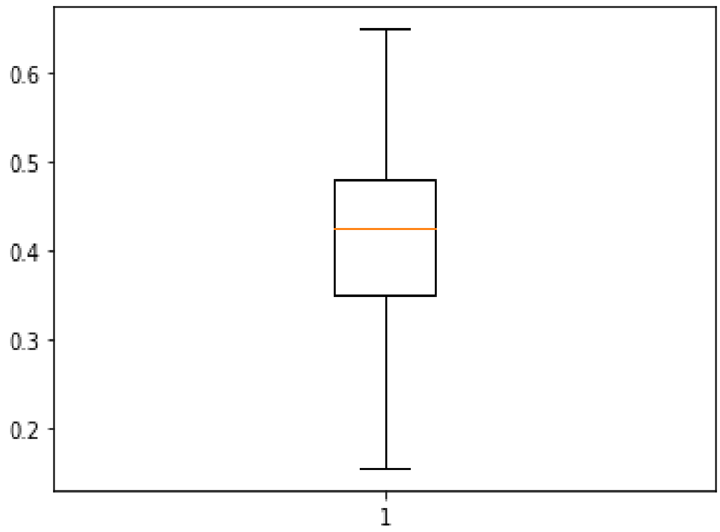
In [21]:

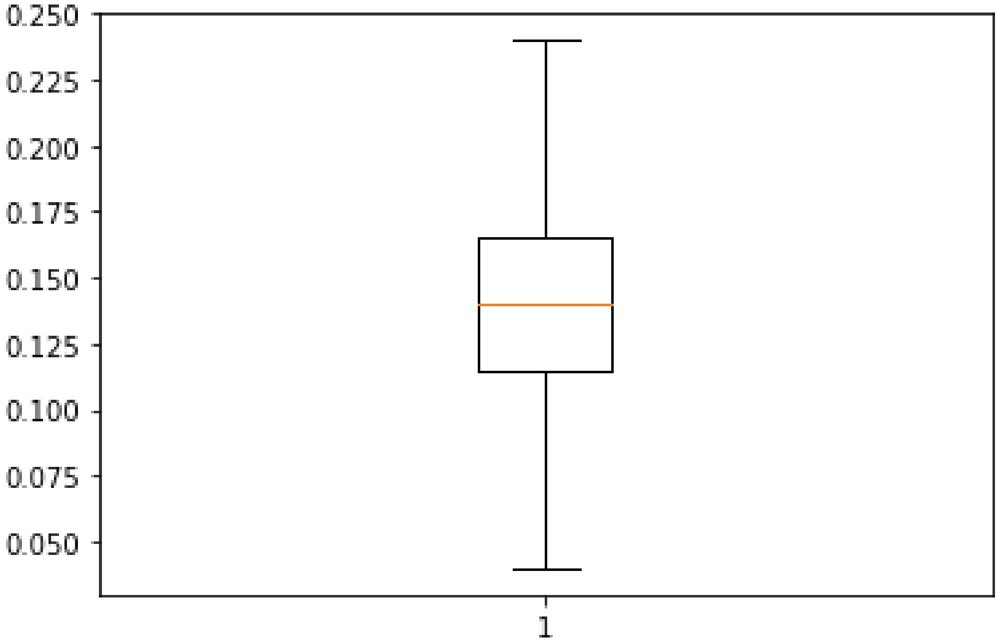
```
def box_scatter(data, x, y):
 fig, (ax1, ax2) = plt.subplots(nrows=2, ncols=1,
figsize=(16,6))  sns.boxplot(data=data, x=x, ax=ax1)
 sns.scatterplot(data=data, x=x,y=y,ax=ax2)
```

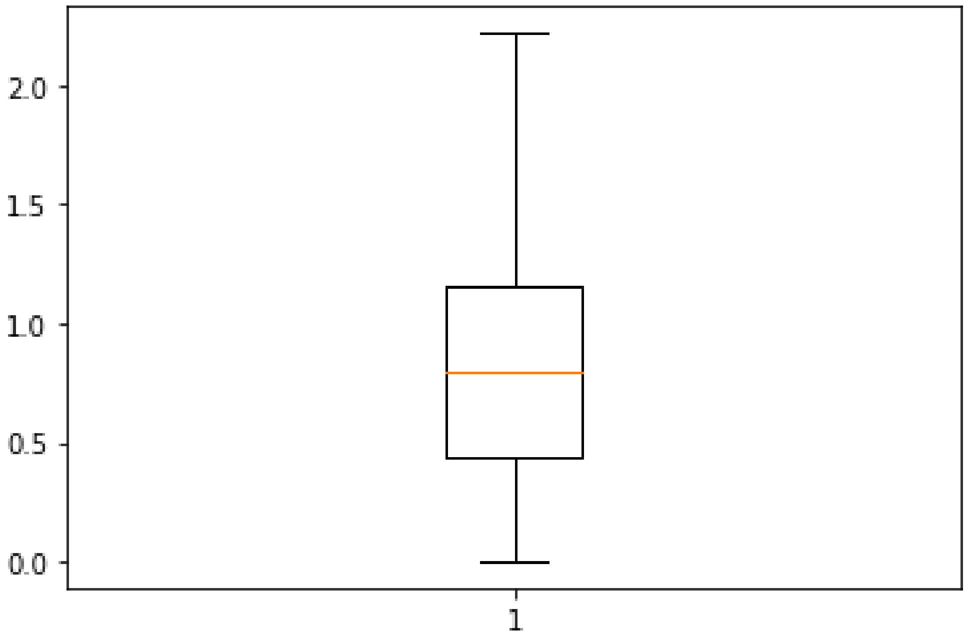
In [22]:

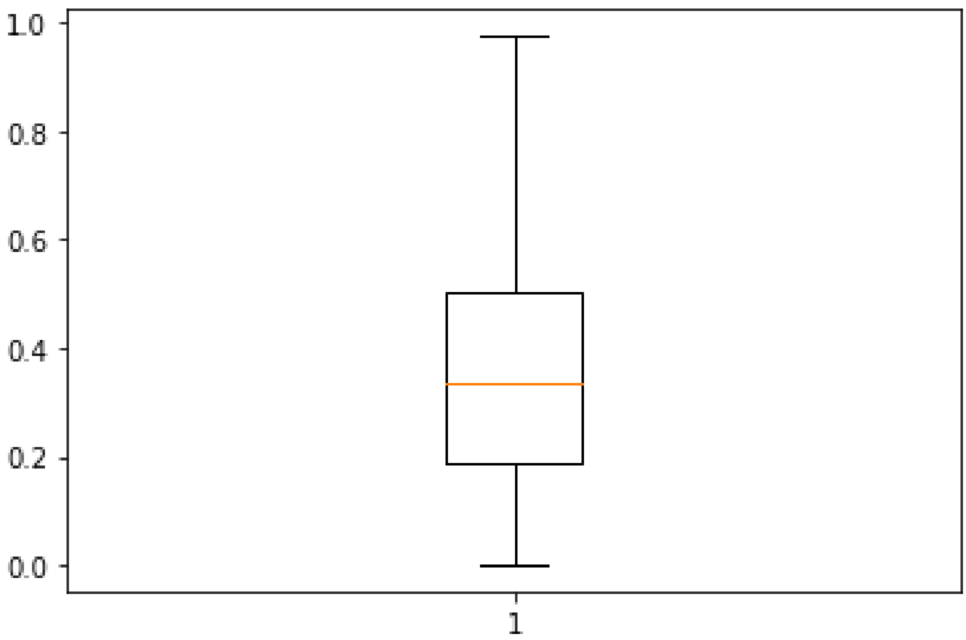
```
for i in df:
 if df[i].dtype=='int64' or df[i].dtypes=='float64':
 mtp.boxplot(df[i])
 mtp.show()
```

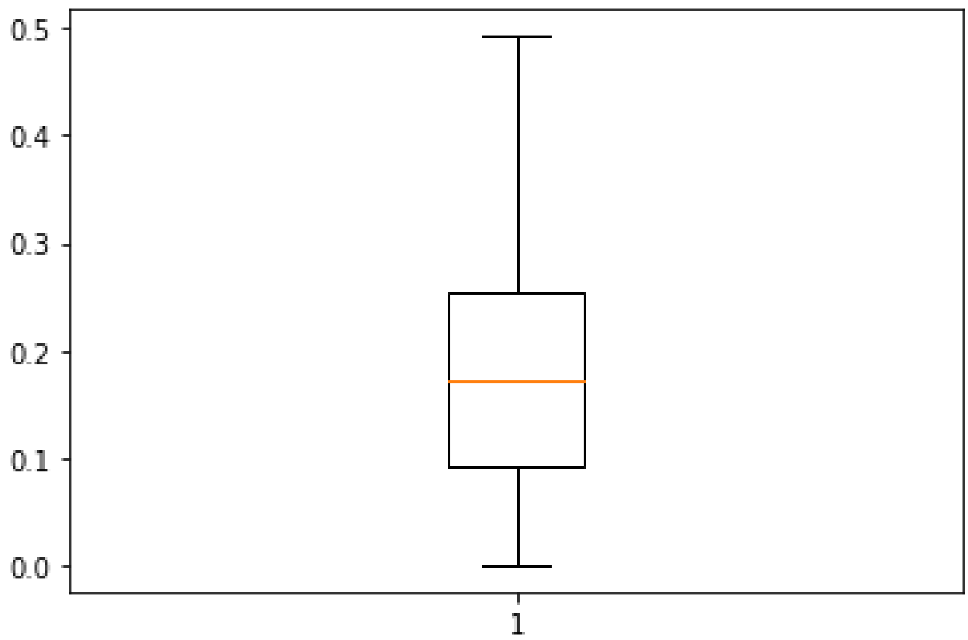


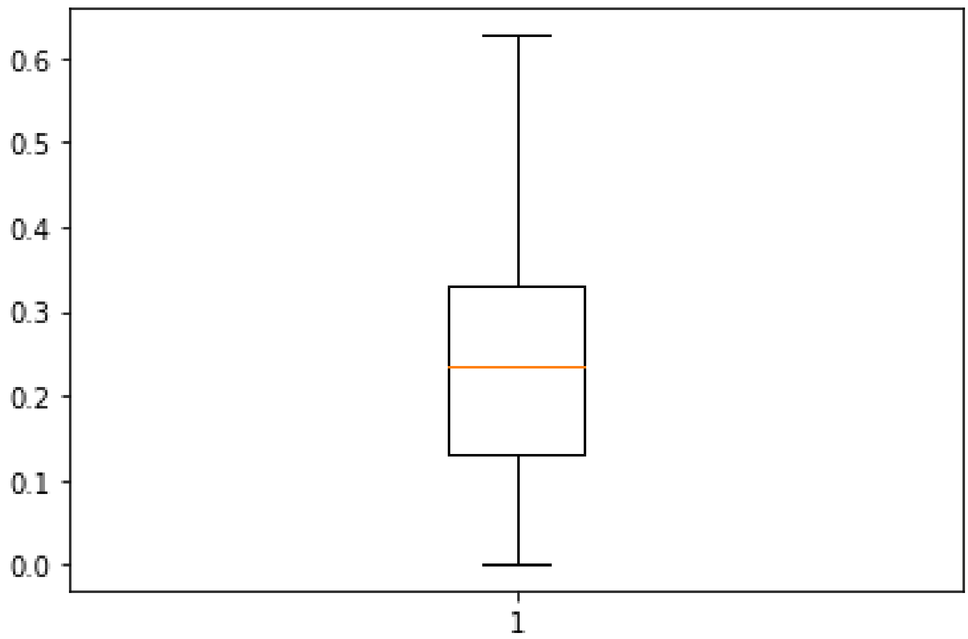

0.6
0.5
0.4
0.3
0.2
1

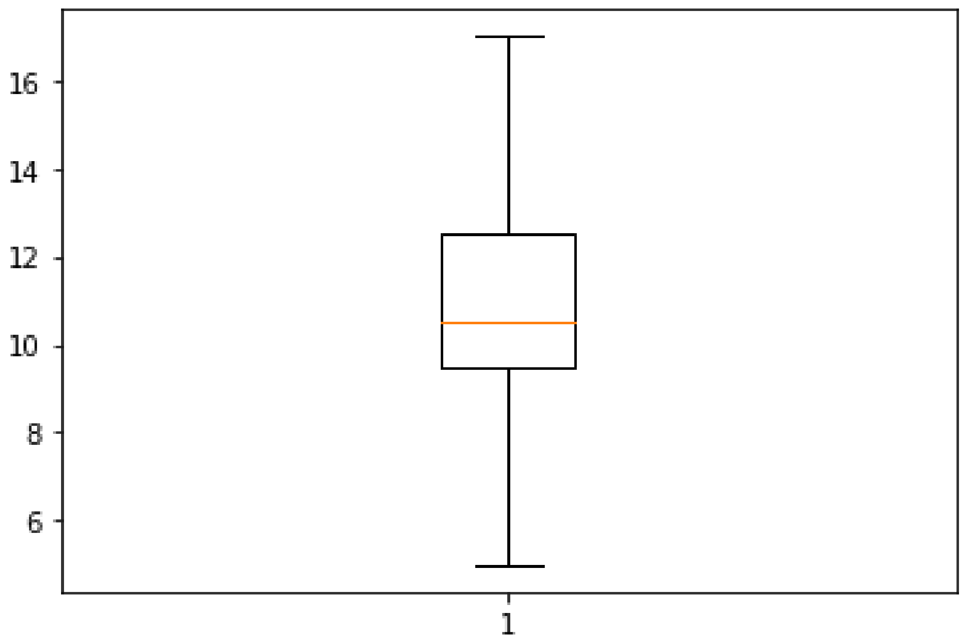

0.250
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
1


2.0
1.5
1.0
0.5
0.0
1


1.0
0.8
0.6
0.4
0.2
0.0
1


0.5
0.4
0.3
0.2
0.1
0.0
1


0.6
0.5
0.4
0.3
0.2
0.1
0.0
1



7. Check for Categorical columns and perform encoding.

In [23]:

```
df.head()
```

Out[23]:

se x
lengt h
diamete r
heigh t
whole_weigh t
shucked_weigh t
viscera_weigh t
shell_weigh

t age

0 M 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.150 16.5 **1** M 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.070 8.5 **2** F 0.530 0.420 0.135 0.6770 0.2565 0.1415 0.210 10.5 **3** M 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.155 11.5

4 I 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.055 8.5

In [24]:

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
df['sex']=encoder.fit_transform(df['sex'])
```

In [25]:

```
df.head()
```

Out[25]:

se x
lengt h
diamete r
heigh t
whole_weigh t
shucked_weigh t
viscera_weigh t
shell_weigh

t**age**

0 2 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.150 16.5 **1** 2 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.070 8.5 **2** 0 0.530 0.420 0.135 0.6770 0.2565 0.1415 0.210 10.5 **3** 2 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.155 11.5 **4** 1 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.055 8.5

8. Split the data into dependent and independent variables.

In [26]:

```
x=df.iloc[:,:-1]
x.head()
```

Out[26]:

sex length diameter height whole_weight shucked_weight viscera_weight shell_weight

0 2 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.150 **1** 2 0.350 0.265 0.090 0.2255 0.0995 0.0485

0.070 **2** 0 0.530 0.420 0.135 0.6770 0.2565 0.1415 0.210 **3** 2 0.440 0.365 0.125 0.5160 0.2155

0.1140 0.155

4 1 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.055

In [27]:

```
y=df.iloc[:,-1]
y.head()
```

Out[27]:

```
0 16.5
1 8.5
2 10.5
3 11.5
4 8.5
Name: age, dtype: float64
```

9. Scale the independent variable

In [28]:

```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x=scaler.fit_transform(x)
```

10. Split the data into training and testing

In [29]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33)
```

In [30]:
```
x_train.shape
```

Out[30]:
```
(2797, 8)
```

In [31]:
```
x_test.shape
```

Out[31]:
```
(1378, 8)
```

11. Build the Model

In [32]:
```
from sklearn.ensemble import RandomForestRegressor
reg=RandomForestRegressor()
```

12. Train the Model

In [33]:
```
reg.fit(x_train,y_train)
```

Out[33]:
```
RandomForestRegressor()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

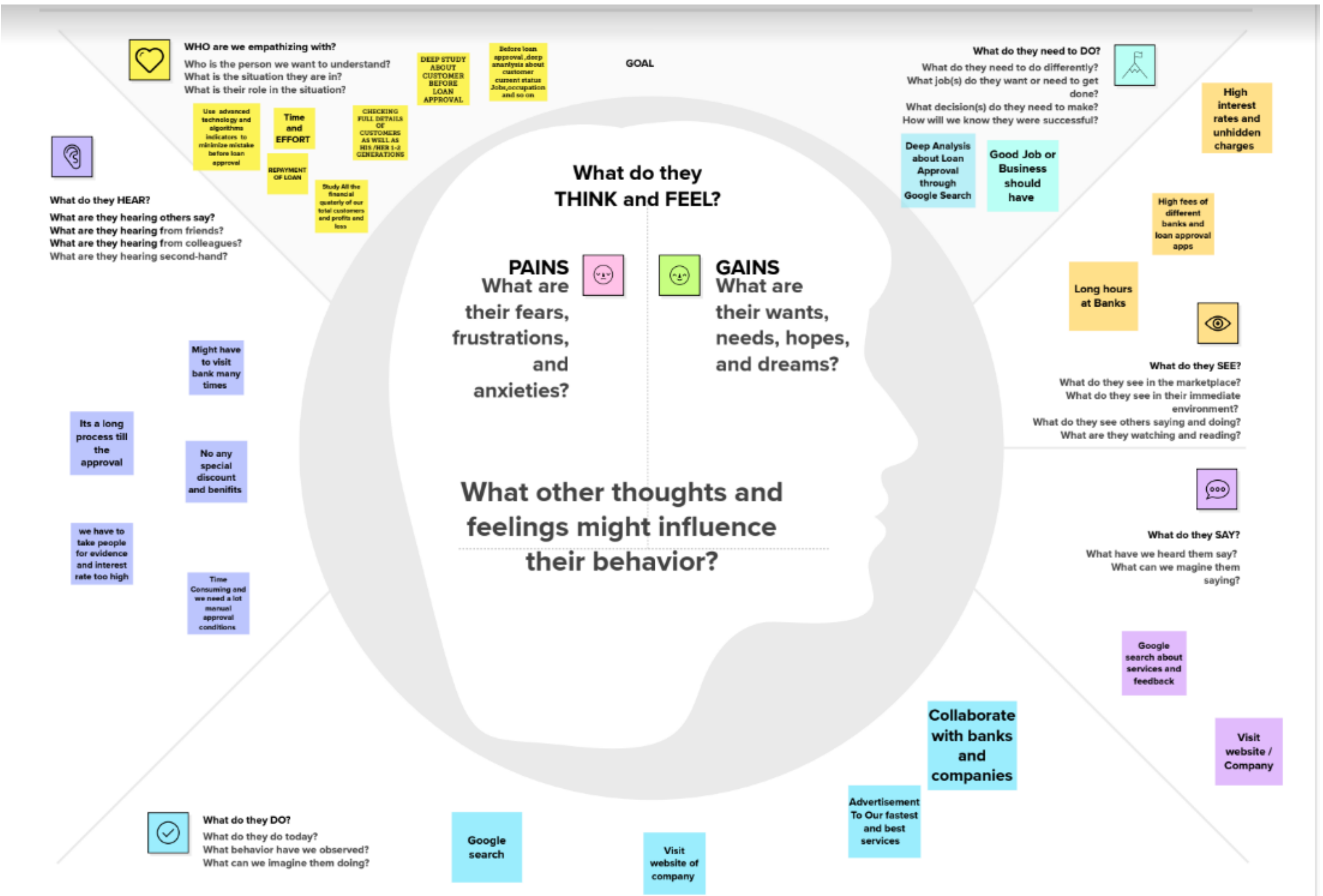
13. Test the Model

In [34]:
```
y_pred=reg.predict(x_test)
```

14. Measure the performance using Metrics.

In [35]:
```
from sklearn.metrics import mean_squared_error
import math
print(math.sqrt(mean_squared_error(y_test,y_pred)))
```

EMPATHY MAP



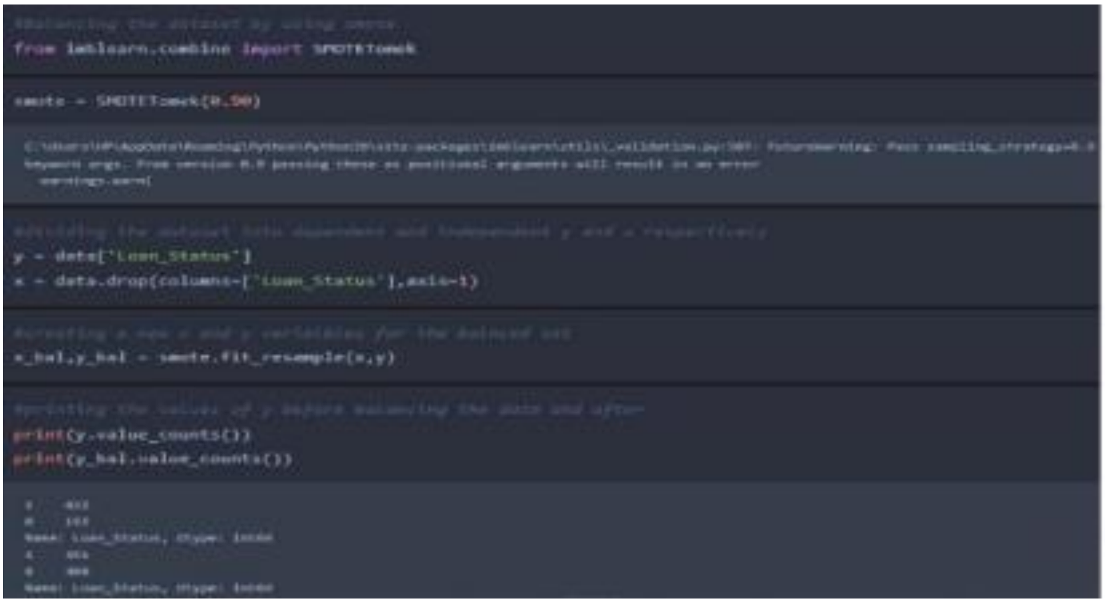
Data Pre-Processing

Balancing The Dataset

Data Balancing is one of the most important step, which need to be performed for classification models, because when we train our model on imbalanced dataset ,we will get biased results, which means our model is able to predict only one class element

For Balancing the data we are using SMOTE Method.

SMOTE: Synthetic minority over sampling technique, which will create new synthetic data points for under class as per the requirements given by us using KNN method.



From the above picture, we can infer that previously our dataset is having 492 class 1, and 192 class items, after applying smote technique on the dataset the size has been changed for minority class.

Data Pre-Processing

Handling Categorical Values

As we can see our dataset has categorical data we must convert the categorical data to integer encoding or binary encoding.

To convert the categorical features into numerical features we use encoding techniques. There are several techniques but in our project, we are using manual encoding with the help of list comprehension.

- In our project, Gender , married, dependents, self-employed, co applicants income, loan amount , loan amount term, credit history With list comprehension encoding is done.

In [22]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
oneh = OneHotEncoder()
data['Education'] = le.fit_transform(data['Education'])
data.head()
```

Out[22]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	1	No	0	0	No	5849	0.0	NaN	360.0	1.0
1	LP001003	1	Yes	1	0	No	4583	1508.0	128.0	360.0	1.0
2	LP001005	1	Yes	0	0	Yes	3000	0.0	66.0	360.0	1.0
3	LP001006	1	Yes	0	1	No	2583	2358.0	120.0	360.0	1.0
4	LP001008	1	No	0	0	No	6000	0.0	141.0	360.0	1.0

Data Pre-Processing

Checking For Null Values

- Let's find the shape of our dataset first, To find the shape of our data, df.shape method is used. To find the data type, df.info() function is used.

In [10]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
Loan_ID              614 non-null object
Gender               601 non-null object
Married              611 non-null object
Dependents           599 non-null object
Education            614 non-null object
Self_Employed        582 non-null object
ApplicantIncome      614 non-null int64
CoapplicantIncome    614 non-null float64
LoanAmount           592 non-null float64
Loan_Amount_Term     600 non-null float64
Credit_History       564 non-null float64
Property_Area        614 non-null object
Loan_Status          614 non-null object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.4+ KB
```

- For checking the null values, df.isnull() function is used. To sum thosenull values we use .sum() function to it. From the below image we found that there are no null values present in our dataset. So we can skip the handling of the missing values step.

```
In [9]: import pandas as pd
        data = pd.read_csv(r"C:\Users\ELCOT\Downloads\Dataset\loan_prediction.csv")
        data.isnull().any()

Out[9]: Loan_ID              False
        Gender                True
        Married               True
        Dependents            True
        Education            False
        Self_Employed         True
        ApplicantIncome      False
        CoapplicantIncome    False
        LoanAmount            True
        Loan_Amount_Term      True
        Credit_History        True
        Property_Area        False
        Loan_Status          False
        dtype: bool
```

From the above code of analysis, we can infer that columns such as gender, married, dependents, self-employed, loan amount, loan amounttern, and credit history are having the missing values, we need to treat them in a required way.

```
In [10]: data['Gender']=data['Gender'].fillna(data['Gender'].mode()[0])

In [11]: data['Married']=data['Married'].fillna(data['Married'].mode()[0])

In [12]: data['Dependents']=data['Dependents'].fillna(data['Dependents'].mode()[0])

In [13]: data['Self_Employed']=data['Self_Employed'].fillna(data['Self_Employed'].mode()[0])

In [14]: data['LoanAmount']=data['LoanAmount'].fillna(data['LoanAmount'].mode()[0])

In [15]: 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])
```

We will fill the missing values in numeric data type using the mean value ofthat particular column and categorical data type using the most repeated value.

Data Pre-Processing

Scaling The Data

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

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

```
In [34]: from sklearn.preprocessing import LabelEncoder, OneHotEncoder,StandardScaler
         le = LabelEncoder()
         oneh = OneHotEncoder()
         sc = StandardScaler()
         data['Gender'] = le.fit_transform(data['Gender'])
         data['Loan_ID'] = le.fit_transform(data['Loan_ID'])
         data.head()
         x = data.iloc[0:5, 0:2]
         x_scaled = sc.fit_transform(x)
         x_scaled

Out[34]: array([[-1.41421356,  0.        ],
                [-0.70710678,  0.        ],
                [ 0.        ,  0.        ],
                [ 0.70710678,  0.        ],
                [ 1.41421356,  0.        ]])
```

We will perform scaling only on the input values

Once the dataset is scaled, it will be converted into array and we need to convert it back to dataframe.

Data Pre-Processing

Splitting Data Into Train And Test

Now let’s split the Dataset into train and test sets

Changes: first split the dataset into x and y and then split the data set

Here x and y variables are created. On the x variable, df is passed by dropping the target variable. And on y target variable is passed. For splitting training and testing data, we are using the train_test_split() function from sklearn. As parameters, we are passing x, y, test_size, and random_state.

```
In [38]: from sklearn.model_selection import train_test_split
         from sklearn.preprocessing import LabelEncoder, OneHotEncoder,StandardScaler
         le = LabelEncoder()
         oneh = OneHotEncoder()
         sc = StandardScaler()
         data['Gender'] = le.fit_transform(data['Gender'])
         data['Loan_ID'] = le.fit_transform(data['Loan_ID'])
         data.head()
         x = data.iloc[0:5, 0:2]
         x_scaled = sc.fit_transform(x)
         x_scaled
         x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size = 0.1, random_state = 0)
         x_train
```

Login page

code:

```
<!-- login.html -->

<!DOCTYPE html>
```

```
<html>

<head>

        <title>Slide Navbar</title>

        <link rel="stylesheet" type="text/css" href="slide navbar style.css">

<link href="https://fonts.googleapis.com/css2?family=Jost:wght@500&display=swap"
rel="stylesheet"> </head>
<body>

        <div class="main">

                <input type="checkbox" id="chk" aria-hidden="true">

<div class="signup">

                        <form>

                            <label for="chk" aria-hidden="true">Sign up</label>

                            <input type="text" name="txt" placeholder="User name"
required="">

                            <input type="email" name="email" placeholder="Email"
required="">

                            <input type="password" name="pswd" placeholder="Password"
required="">

                            <button>Sign up</button>

                        </form>
                    </div>



                    <div class="login">

                        <form>

                            <label for="chk" aria-hidden="true">Login</label>

                            <input type="email" name="email" placeholder="Email"
required="">

                            <input type="password" name="pswd" placeholder="Password"
```

```
required="">
                                <button><a href="predict.html">Login</a></button>
                            </form>
                        </div>
            </div>
</body>
</html>
```

predict css:

```
@extend display-flex; */
display-flex, .display-flex, .display-flex-center, .signup-content, .signin-content, .social-login, .socials
{  display: flex;
 display: -webkit-flex; }

/* @extend list-type-ulli; */
list-type-ulli, .socials {
 list-style-type: none;
 margin: 0;
 padding: 0; }

/* poppins-300 - latin */
@font-face {
 font-family: 'Poppins';
 font-style: normal;
 font-weight: 300;
 src: url("../fonts/poppins/poppins-v5-latin-300.eot");
```

```
/* IE9 Compat Modes */

 src: local("Poppins Light"), local("Poppins-Light"),
url("../fonts/poppins/poppins-v5-latin-300.eot?#iefix") format("embedded-opentype"),  url("../fonts/poppins/poppins-v5-latin-300.woff2") format("woff2"),
url("../fonts/poppins/poppins-v5-latin-300.woff") format("woff"),
url("../fonts/poppins/poppins-v5-latin-300.ttf") format("truetype"),
url("../fonts/poppins/poppins-v5-latin-300.svg#Poppins") format("svg");

 /* Legacy iOS */ }
/* poppins-300italic - latin */

@font-face {

 font-family: 'Poppins';

 font-style: italic;

 font-weight: 300;

 src: url("../fonts/poppins/poppins-v5-latin-300italic.eot");

 /* IE9 Compat Modes */

 src: local("Poppins Light Italic"), local("Poppins-LightItalic"),
url("../fonts/poppins/poppins-v5-latin-300italic.eot?#iefix") format("embedded-opentype"),  url("../fonts/poppins/poppins-v5-latin-300italic.woff2")
format("woff2"),  url("../fonts/poppins/poppins-v5-latin-300italic.woff")
format("woff"),  url("../fonts/poppins/poppins-v5-latin-300italic.ttf")
format("truetype"),  url("../fonts/poppins/poppins-v5-latin-300italic.svg#Poppins")
format("svg");

 /* Legacy iOS */ }

/* poppins-regular - latin */

@font-face {

 font-family: 'Poppins';

 font-style: normal;

 font-weight: 400;

 src: url("../fonts/poppins/poppins-v5-latin-regular.eot");

 /* IE9 Compat Modes */

 src: local("Poppins Regular"), local("Poppins-Regular"),
url("../fonts/poppins/poppins-v5-latin-regular.eot?#iefix") format("embedded-opentype"),  url("../fonts/poppins/poppins-v5-latin-regular.woff2")
format("woff2"),  url("../fonts/poppins/poppins-v5-latin-regular.woff") format("woff"),
```

url("../fonts/poppins/poppins-v5-latin-regular.ttf")
format("truetype"), url("../fonts/poppins/poppins-v5-latin-
regular.svg#Poppins") format("svg");

/* Legacy iOS */ }

/* poppins-italic - latin */
@font-face {

font-family: 'Poppins';

font-style: italic;

font-weight: 400;

src: url("../fonts/poppins/poppins-v5-latin-italic.eot");

/* IE9 Compat Modes */

src: local("Poppins Italic"), local("Poppins-Italic"),
url("../fonts/poppins/poppins-v5-latin-italic.eot?#iefix") format("embedded-
opentype"), url("../fonts/poppins/poppins-v5-latin-italic.woff2")
format("woff2"), url("../fonts/poppins/poppins-v5-latin-italic.woff")
format("woff"), url("../fonts/poppins/poppins-v5-latin-italic.ttf")
format("truetype"), url("../fonts/poppins/poppins-v5-latin-italic.svg#Poppins")
format("svg");

/* Legacy iOS */ }

/* poppins-500 - latin */

@font-face {

font-family: 'Poppins';

font-style: normal;

font-weight: 500;

src: url("../fonts/poppins/poppins-v5-latin-500.eot");

/* IE9 Compat Modes */

src: local("Poppins Medium"), local("Poppins-Medium"),
url("../fonts/poppins/poppins-v5-latin-500.eot?#iefix") format("embedded-
opentype"), url("../fonts/poppins/poppins-v5-latin-500.woff2")
format("woff2"), url("../fonts/poppins/poppins-v5-latin-500.woff") format("woff"),
url("../fonts/poppins/poppins-v5-latin-500.ttf")
format("truetype"), url("../fonts/poppins/poppins-v5-latin-
500.svg#Poppins") format("svg");

/* Legacy iOS */ }

```
/* poppins-500italic - latin */

@font-face {
 font-family: 'Poppins';

 font-style: italic;

 font-weight: 500;

 src: url("../fonts/poppins/poppins-v5-latin-500italic.eot");

 /* IE9 Compat Modes */

 src: local("Poppins Medium Italic"), local("Poppins-
MediumItalic"),  url("../fonts/poppins/poppins-v5-latin-500italic.eot?#iefix")
format("embedded-opentype"),  url("../fonts/poppins/poppins-v5-latin-500italic.woff2")
format("woff2"),  url("../fonts/poppins/poppins-v5-latin-500italic.woff")
format("woff"),  url("../fonts/poppins/poppins-v5-latin-500italic.ttf")
format("truetype"),  url("../fonts/poppins/poppins-v5-latin-500italic.svg#Poppins")
format("svg");

 /* Legacy iOS */ }

/* poppins-600 - latin */

@font-face {

 font-family: 'Poppins';

 font-style: normal;

 font-weight: 600;

 src: url("../fonts/poppins/poppins-v5-latin-600.eot");

 /* IE9 Compat Modes */

 src: local("Poppins SemiBold"), local("Poppins-SemiBold"),
url("../fonts/poppins/poppins-v5-latin-600.eot?#iefix") format("embedded-
opentype"),  url("../fonts/poppins/poppins-v5-latin-600.woff2") format("woff2"),
url("../fonts/poppins/poppins-v5-latin-600.woff") format("woff"),
url("../fonts/poppins/poppins-v5-latin-600.ttf") format("truetype"),
url("../fonts/poppins/poppins-v5-latin-600.svg#Poppins")

format("svg");  /* Legacy iOS */ }

/* poppins-700 - latin */

@font-face {

 font-family: 'Poppins';
 font-style: normal;
```

```
font-weight: 700;

src: url("../fonts/poppins/poppins-v5-latin-700.eot");

/* IE9 Compat Modes */

src: local("Poppins Bold"), local("Poppins-Bold"),
url("../fonts/poppins/poppins-v5-latin-700.eot?#iefix") format("embedded-
opentype"),  url("../fonts/poppins/poppins-v5-latin-700.woff2") format("woff2"),
url("../fonts/poppins/poppins-v5-latin-700.woff") format("woff"),
url("../fonts/poppins/poppins-v5-latin-700.ttf") format("truetype"),
url("../fonts/poppins/poppins-v5-latin-700.svg#Poppins")

format("svg");  /* Legacy iOS */ }

/* poppins-700italic - latin */

@font-face {

font-family: 'Poppins';

font-style: italic;

font-weight: 700;

src: url("../fonts/poppins/poppins-v5-latin-700italic.eot");

/* IE9 Compat Modes */

src: local("Poppins Bold Italic"), local("Poppins-BoldItalic"),
url("../fonts/poppins/poppins-v5-latin-700italic.eot?#iefix") format("embedded-
opentype"),  url("../fonts/poppins/poppins-v5-latin-700italic.woff2")
format("woff2"),  url("../fonts/poppins/poppins-v5-latin-700italic.woff")
format("woff"),  url("../fonts/poppins/poppins-v5-latin-700italic.ttf")
format("truetype"),  url("../fonts/poppins/poppins-v5-latin-700italic.svg#Poppins")
format("svg");

/* Legacy iOS */ }

/* poppins-800 - latin */

@font-face {

font-family: 'Poppins';

font-style: normal;
font-weight: 800;

src: url("../fonts/poppins/poppins-v5-latin-800.eot");

/* IE9 Compat Modes */
```

src: local("Poppins ExtraBold"), local("Poppins-ExtraBold"),
url("../fonts/poppins/poppins-v5-latin-800.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-800.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-800.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-800.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-800.svg#Poppins") format("svg");

/* Legacy iOS */ }

/* poppins-800italic - latin */

@font-face {

font-family: 'Poppins';

font-style: italic;

font-weight: 800;

src: url("../fonts/poppins/poppins-v5-latin-800italic.eot");

slide navbar style:

body{

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

font-family: 'Jost', sans-serif;

background: #DDA0DD;

}

.main{

width: 350px;

height: 500px;

background: red;

```
            overflow: hidden;

            background:
url("https://doc-08-2c-
docs.googleusercontent.com/docs/securesc/68c90smiglihng9534mvqmq1946d
mis5/fo0picsp1nhiucmc0l25s29respgpr4j/1631524275000/03522360960922298374/03522360960
922
298374/1Sx0jhdpEpnNIydS4rnN4kHSJtU1EyWka?e=view&authuser=0&nonce=gcrocepgbb17m&us
er= 03522360960922298374&hash=tfhgbs86ka6divo3llbvp93mg4csvb38") no-repeat center/
cover;

            border-radius: 10px;

            box-shadow: 5px 20px 50px #000;

}

#chk{

            display: none;
}

.signup{

            position: relative;

            width:100%;

            height: 100%;

}

label{

            color: #fff;

            font-size: 2.3em;

            justify-content: center;

            display: flex;

            margin: 60px;

            font-weight: bold;

            cursor: pointer;

            transition: .5s ease-in-

out; }

input{
```

```
        width: 60%;

        height: 24px;

        background: #fff;

        justify-content: center;

        display: flex;

        margin: 20px auto;

        padding: 10px;

        border: none;
        outline: none;

        border-radius: 5px;

}

button{

        width: 60%;

        height: 40px;

        margin: 10px auto;

        justify-content: center;

        display: block;

        color: darkmagenta;

        background:white;

        font-size: 1em;

        font-weight: bold;

        margin-top: 20px;

        outline: none;

        border: none;

        border-radius: 5px;

        transition: .2s ease-in;

        cursor: pointer;
```

```
}
.login a{
 text-decoration: none;
}

button:hover
{
		background:#D8BFD8;
}
.login{
		height: 460px;
		background: #eee;
		border-radius: 60% / 10%;
		transform: translateY(-
		180px); transition: .8s
		ease-in-out;
}
.login label{
		color: darkmagenta;
		transform: scale(.6);
}

#chk:checked ~ .login{
		transform: translateY(-
500px); }
#chk:checked ~ .login label{
		transform: scale(1);
```

```
}
#chk:checked ~ .signup label{
	transform: scale(.6);
	}
```

Visualizing And Analyzing The Data

Descriptive Analysis

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas have a worthy function called describe. With this describe function we can understand the unique, top, and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

```
In [7]: data.describe()
Out[7]:
```

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

CONTACT HTML

```
<!DOCTYPE html>
<html lang="en">
 <title>Contact</title>
<head>
 <meta charset="UTF-8">
```

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">  <title>Home Page</title>
<link rel="stylesheet" href="static/contact.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font
awesome.min.css">
<script src='https://kit.fontawesome.com/a076d05399.js' crossorigin='anonymous'></script>
</head>
<body>

<a href="home.html"><button class="but">Back</button></a>
<div class="contact-section">
<div class="contact-info">
<div>Address : Vellore,Tamilnadu,India</div>
<div>Email : Bank@gmail.com</div>
<div>Mobile No : 9000033456</div>
<div>Working Hours : Mon - Fri 10:00 AM to 4:30 PM</div>
</div>
<div class="box">
<div class="title">
<h1>Contact Us</h1><br>
<h2>We are ready</h2>
</div>
<form action="" name="contact-us">
<input type="text" name="name" class="form-control" id="name" placeholder="Your
Name"  required><br>
<input type="text" name="phone" class="form-control" id="phone"
placeholder="Your  mobile Number" required><br>
<input type="email" name="email" class="form-control" id="email" placeholder="Your Email  id"
required><br>
<textarea name="message" class="form-control" id="message"
rows="4"  placeholder="Message"></textarea><br>
```

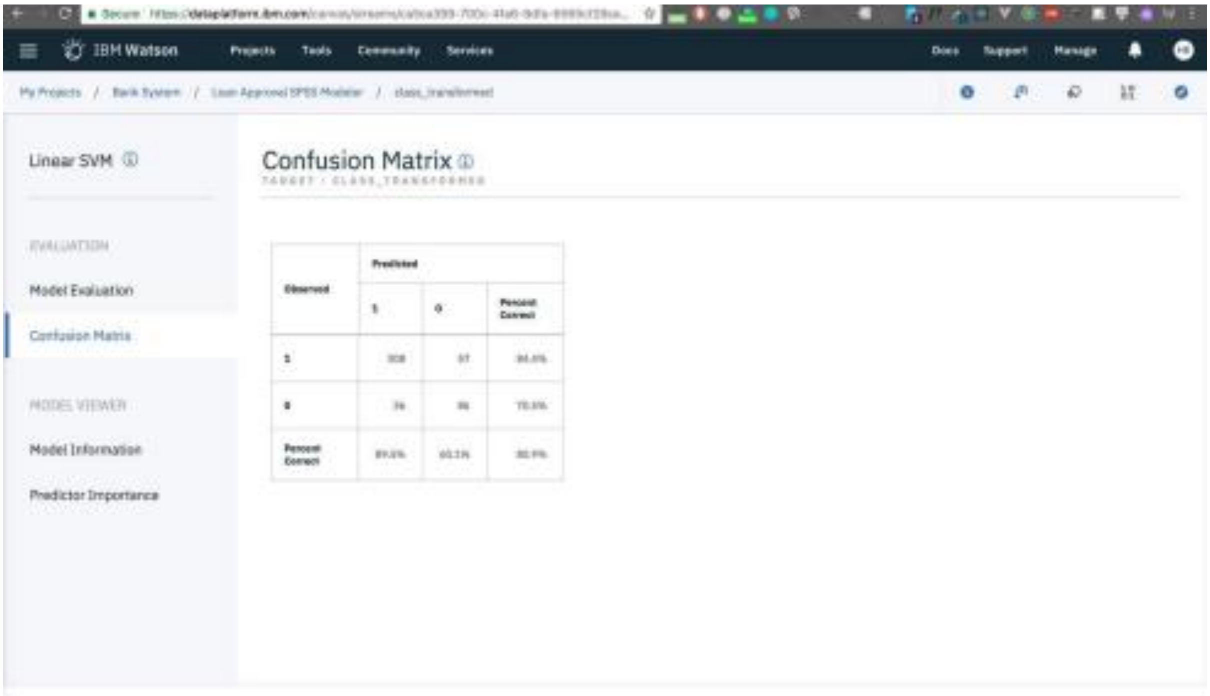
```
<input type="submit" name="" class="form-control submit"
value="SEND">  </form>
</div>
</div>
<script>
const scriptURL =
'https://script.google.com/macros/s/AKfycbyE5Kll2s6QbBSUMinwFcLiWih1xpz6px9YqCsQTYOBbYt0I
82dxixhN1uT1bX4fRPMbA/exec'
const form = document.forms['contact-us']


form.addEventListener('submit', e => {
e.preventDefault()
fetch(scriptURL, { method: 'POST', body: new FormData(form)})
.then(response => console.log('Success!', response))
.catch(error => console.error('Error!', error.message))
form.reset()
alert('Success!')
})
</script>
</body>
</html>
```

MODEL BUILDING COMPARE THE MODEL

Compare models

Now there are many models and you want to select the best one for deployment. A comparison will give you detailed information about each model used. Right-click the LSVM model node and choose View model to see more information about different performance metrics.

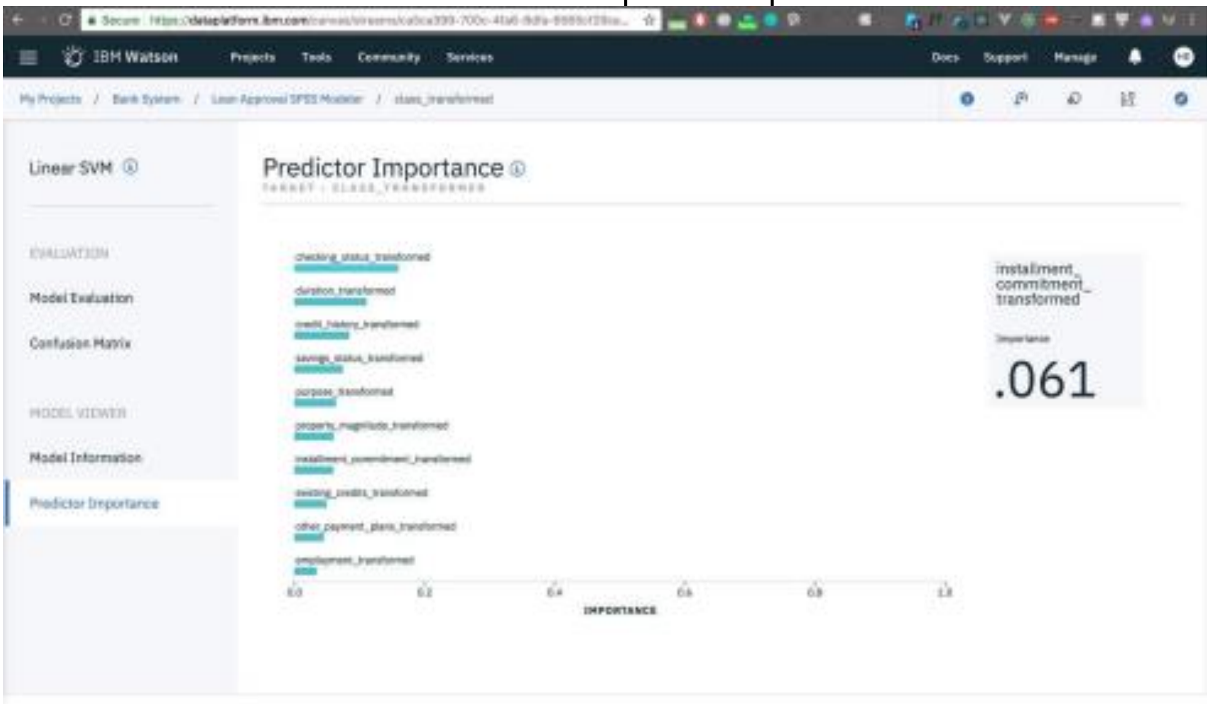
First, from the Model Evaluation tab, you will see details about overall model accuracy. These details include false positives, false negatives, model precision, recall, and f1 score. The overall accuracy of the LSVM model here is 80.9% which is fair. model evaluation: Model Evaluation

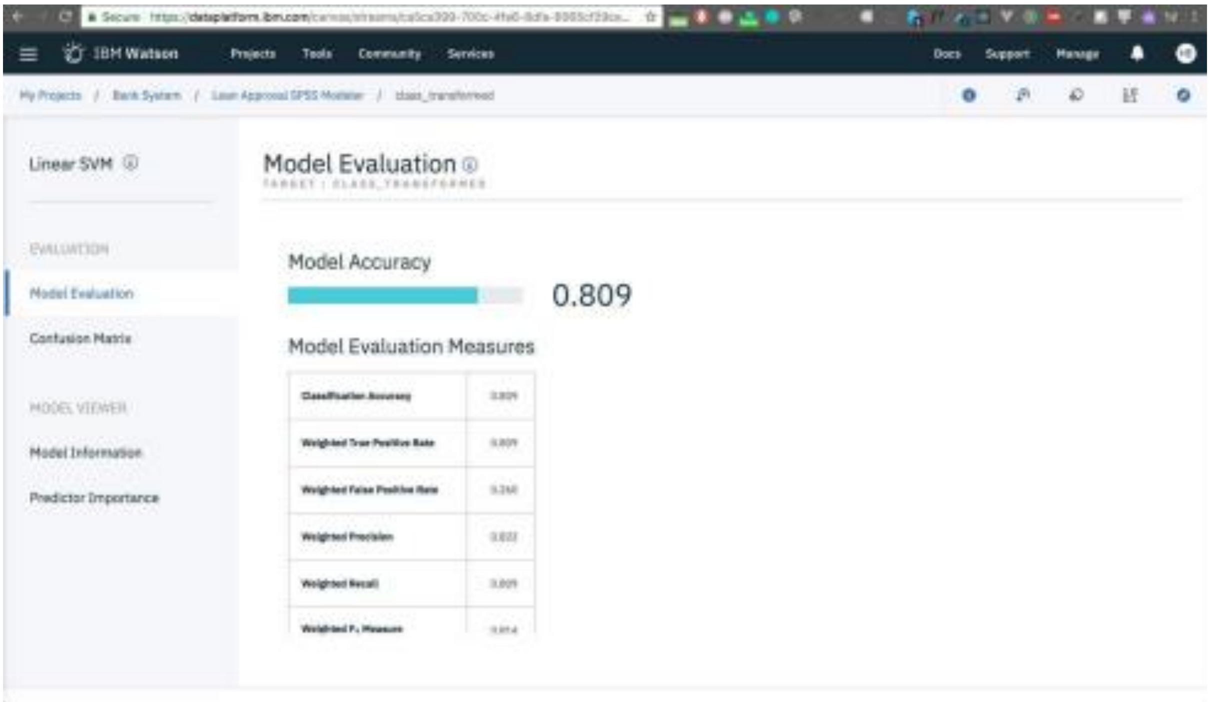


The Confusion Matrix tab shows you the percentages of correct predictions for each class. model evaluation: Confusion Matrix

From the Predictor Importance tab you can see the order of fields that had the highest imapct on the predictions or outputs. model evaluation:Predictor Importance

Use the steps above in the Analyze the model ouput section to check the Random Forest Classifier model's performance. Right-click the model node on the canvas and select View Model. spss: Compare Models





MODEL BUILDING

EVALUATING THE PERFORMANCE OF THE MODEL AND SAVING THE MODEL

When it comes to evaluating a Binary Classifier, Accuracy is a well-known performance metric that is used to tell a strong classification model from one that is weak. Accuracy is, simply put, the total proportion of observations that have been correctly predicted. There are four (4) main components that comprise the mathematical formula for calculating Accuracy, viz. TP, TN, FP, FN, and these components grant us the ability to explore other ML Model Evaluation Metrics. The formula for calculating accuracy is as follows

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Where:

- TP represents the number of True Positives. This refers to the total number of observations that belong to the positive class and have been predicted correctly.
- TN represents the number of True Negatives. This is the total number of observations that belong to the negative class and have been predicted correctly.

- FP is the number of False Positives. It is also known as a Type 1 Error. This is the total number of observations that have been predicted to belong to the positive class, but instead, actually, belong to the negative class.
- FN is the number of False Negatives. It may be referred to as a Type 2 Error. This is the total number of observations that have been predicted to be a part of the negative class but instead belong to the positive class.

The main reason for individuals to utilize the Accuracy Evaluation Metric is for ease of use. This Evaluation Metric has a simple approach and explanation. It is, as discussed before, simply the total proportion (total number) of observations that have been predicted correctly. Accuracy, however, is an Evaluation Metric that does not perform well when the presence of imbalanced classes-when in the presence of imbalanced classes, Accuracy suffers from a paradox; i.e., where the Accuracy value is high but the model lacks predictive power and most, if not all, predictions are going to be incorrect.

For the above reason, when we are unable to use the Accuracy Evaluation Metric, we are compelled to turn to other evaluation metrics in the scikit-learn arsenal. These include, but are not limited to, the following Evaluation Metrics:

Precision

This refers to the proportion (total number) of all observations that have been predicted to belong to the positive class and are actually positive. The formula for Precision Evaluation Metric is as follows:

$$Precision = \frac{TP}{TP + FP}$$

This is the proportion of observation predicted to belong to the positive class, that truly belongs to the positive class. It indirectly tells us the model's ability to randomly identify an observation that belongs to the positive class. The formula for Recall is as follows:

$$Recall = \frac{TP}{TP + FN}$$

F1 Score.

This is an averaging Evaluation Metric that is used to generate a ratio. The F1 Score is also known as the Harmonic Mean of the precision and recall

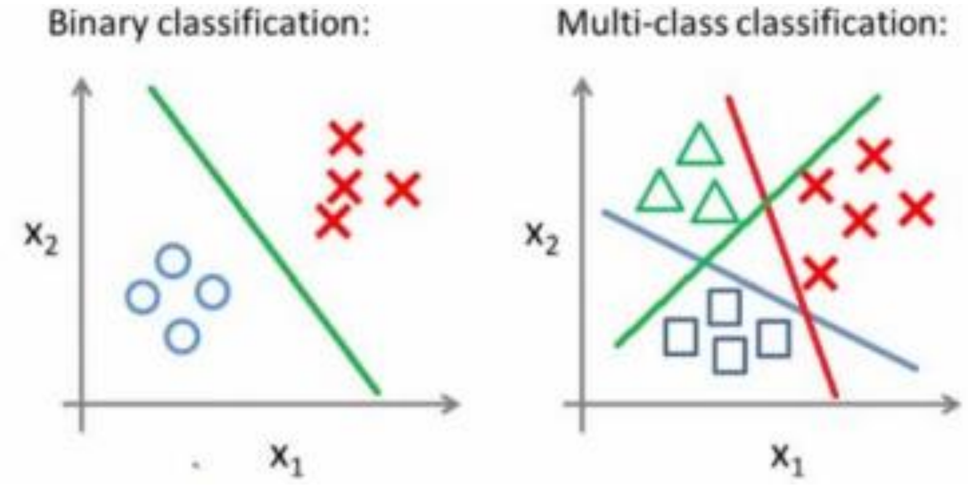
Evaluation Metrics. This Evaluation Metric is a measure of overall correctness that our model has achieved in a positive prediction environment
i.e., Of all observations that our model has labeled as positive, how many of these observations are actually positive. The formula for the F1 Score Evaluation Metric is as follows:

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Evaluating Multiclass Classifier Predictions.

As we have learned from earlier information in the article, in Machine Learning, all input data is not balanced, hence the issue of Imbalanced Classes. With the Accuracy Evaluation Metric removed from our options, we specifically turn to Precision, Recall, and F1 Scores. We use parameter options in Python, which are used for aggregating the evaluation values by averaging them. The three main options that we have available to us are:

1. _macro – Here we specify to the compiler to calculate the mean of metric scores for each class in the dataset, weighting each class equally. 2. _weighted – We calculate the mean of metric scores for each class, and we weigh each class directly proportional to its size in the dataset. 3. _micro – Here we calculate the mean of metric scores for each OBSERVATION in the dataset.



Visualizing a Classifier's Performance.

Currently, the most popular way to visualize a classifier's performance is through a Confusion Matrix. A Confusion Matrix may be referred to as an Error Matrix. A Confusion Matrix has a high level of interpretability. It comprises a simple tabular format, which is often generated and visualized as a Heatmap. Each Column of the Confusion Matrix represents the predicted classes, while every row shows the true (or actual) classes.

There are three important facts to be aware of about a Confusion Matrix:

1. A Perfect Confusion Matrix will have values along the main diagonal (from left to right), and there will zeroes (0) everywhere else in the confusion matrix.
2. A Confusion Matrix does not only show us where the Machine Learning Model faltered but also how it reached those conclusions.
3. A Confusion Matrix will function with any number of classes, i.e., Having a dataset containing 50 classes, will not affect model performance nor the Confusion Matrix- it just means your Visualized Matrix will be very large in size.

Evaluating a Regression Model's Performance.

For a Regressor, you will find that one of the most used and well-known Evaluation Metrics is MSE. MSE stands for Mean Squared Error. Put into a mathematical representation, MSE is calculated as follows:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

Where:

- n represents the number of observations in the dataset.
- y_i is the true value of the target value we are trying to predict for observation I.
- $\hat{y}_i$ is the model's predicted value for y_i.

MSE is a calculation that involves finding the squared sum of all the distances between predicted and true values. The higher the output value for MSE, the greater the sum of squared error present in the model, and hence, the worse the quality of model predictions. There are advantages of squaring the error margins, as seen in the model:

Project Design Phase-II
Solution Requirements (Functional & Non-functional)

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Requirement	To check the loan eligibility using the credit score, prediction for loan approval.
FR-2	User Confirmation	Through one time verification and using captcha etc…
FR-3	Profile Updation	The user can update their profile when their is need to add any add-ons to it.
FR-4	User Registration	The user gets login or signup using Gmail account or by using mobile number.
FR-5	User Authentication	By OTP or verification code the user gets authenticated and OTP is used for mobile number registration.
FR-6	Feedback Evaluation	The user provided feedbacks are used for evaluation of app performance and updation is made over that.

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This application is mainly used to analyse cibil score and predict the eligibility for users to avail for loan approval by following community guidelines.
NFR-2	**Security**	It uses OTP and verify code verification for each user and uses hybrid security features over internet to safely maintain the updated documents of user .
NFR-3	**Reliability**	Maintaining the app up to date for reliant features ,durability and efficiency of the mobile app by releasing patch fix and software updates.

NFR-4	**Performance**	It has a user friendly interface and can check multiple persons cibil score parallels irrespective of server traffic .It stores the data collected over in a efficient database
NFR-5	**Availability**	It is platform independent and it is available where the users are able to wish it want to be.Depending upon the user requirements all services get offered.

NFR-6	**Scalability**	Provides accurate prediction for user eligibility by using high efficient algorithms and testing all the documents uploaded by the user at high efficient rate.

STATIC

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

from sklearn.preprocessing import scale

app= Flask(__name__, template_folder='templates')



model = pickle.load(open("Rfmodel.pkl",'rb'))







@app.route('/')

def home():

 return render_template('home.html')

@app.route('/login.html')

def login():

 return render_template('login.html')

@app.route('/procedure.html')

def procedure():

 return render_template('procedure.html')

@app.route('/bank login.html')
```

```
def bank():

 return render_template('bank login.html')

@app.route('/About.html')

def about():

 return render_template('About.html')

@app.route('/terms.html')

def terms():

 return render_template('terms.html')

@app.route('/register.html')

def register():
 return render_template('register.html')

@app.route('/contact.html')

def contact():

 return render_template('contact.html')

@app.route('/home.html')

def home1():

 return render_template('home.html')

@app.route('/prediction.html')

def formpg():

 return render_template('prediction.html')

@app.route('/rating.html')

def rat():

 return render_template('rating.html')

@app.route('/prediction.html',methods = ['POST'])

def predict():

 if request.method=='POST':

 name=request.form['Name']

 gender=request.form['gender']

 married=request.form['married']

 dependents=request.form['dependents']

 education=request.form['education']
```

```
employed=request.form['employed']

credit=request.form['credit']

proparea=request.form['proparea']

ApplicantIncome=float(request.form['ApplicantIncome'])  Coapplica

ntIncome=float(request.form['CoapplicantIncome'])  LoanAmount=fl

oat(request.form['LoanAmount'])  Loan_Amount_Term=float(request

.form['Loan_Amount_Term'])  if gender == 'Male':

gender = 1

else:
gender = 0


if married == 'Yes':

married = 1

else:

married = 0


if education ==

'Graduate':  education = 0

else:

education = 1


if employed ==

'Yes':  employed = 1

else:

employed = 0


if dependents ==

'3+':  dependents = 3

if credit == 'Yes':
```

```
credit = 1
else:
credit = 0
if proparea ==
'Urban':  proparea = 2
 elif proparea ==
'Rural':  proparea = 0
else:
proparea = 1



features =
[gender,married,dependents,education,employed,ApplicantIncome,CoapplicantIncome,LoanAmoun
t,Loan_Amount_Term,credit,proparea]


con_features = [np.array(features)]




prediction = model.predict(con_features)
print(prediction)
if prediction==1:
return render_template('approve.html',prediction_text ='Congratulations! '+name+' You
are  eligible for loan')
else:
return render_template('reject.html',prediction_text ='Sorry '+name+' You are not eligible
for  loan')



if __name__ == "__main__":
app.run(debug=True)
```

Project Design Phase-II
Technology Stack (Architecture & Stack)

Technical Architecture:

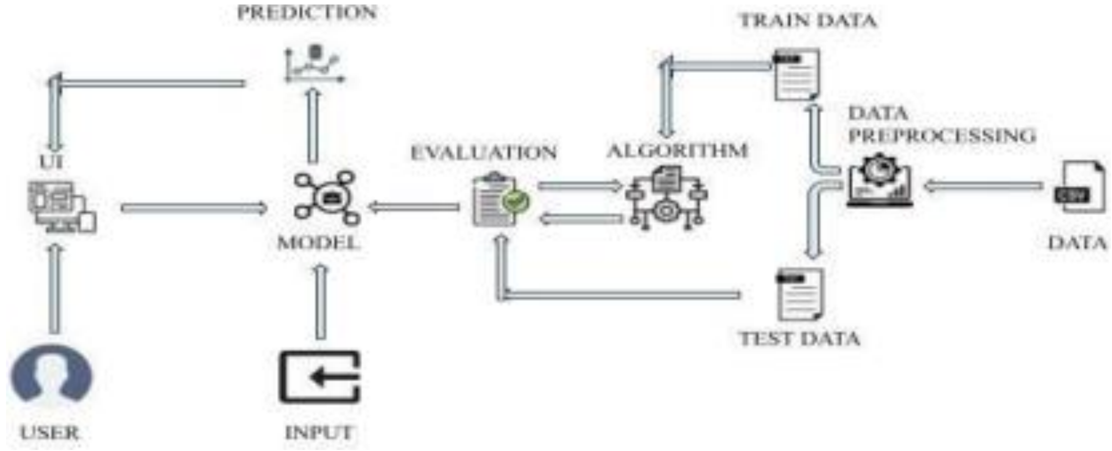


Table-1 : Components & Technologies:

1.	User Interface	Users interact with the application with the help of a web UI.	HTML, CSS,Javascript
2.	Building application	Getting user information from UI and feeding it to ML model	Python Flask
3.	Visualizing and analysing data	Reading and understanding the data properly with the help of visualization and analysing techniques	Python pandas, numpy, matplotlib, seaborn
4.	Pre-processing or cleaning data	Handling missing values, Handling categorical data, Handling outliers, Scaling Techniques	Python pandas
5.	Database	Loan Approval dataset.	csv file

S.No	Component	Description	Technology
6.	Cloud Database	Deploying the model on cloud	IBM cloud
7.	Machine Learning Model	Using machine learning model for predicting loan approval	Model building using classification algorithms such as Decision tree, Random forest, KNN, and xg boost

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask is used to host the website. Scikit, numpy and tensorflow are all open source python machine learning frameworks.	Scikit, Numpy
2.	Security Implementations	OpenSSL is a program and library that supports many different cryptographic operations, including: Symmetric key	OpenSSL Encryption
		encryption. Public/private key pair generation. Public key encryption. Hash functions.	

3.	Scalable Architecture	Since the application servers can be deployed on many machines. Also, the database does not make longer connections with every client – it only requires connections from a smaller number of application servers. It improves data integrity.	3 Tier Architecture

4.	Availability	Decentralized storage and distribution along-with web application approach make the service highly available.	IBM Cloud file storage, MySQL Online
5.	Performance	Long term header expiration. Cacheable AJAX Cookie Free Domain Compress zip components.	AJAX, CDN

PREDICTION HTML

```
<!doctype html>

<html lang="en">

 <head>

 <!-- Required meta tags -->

 <meta charset="utf-8">

 <meta name="viewport" content="width=device-width, initial-scale=1">


 <!-- Bootstrap CSS -->

 <link rel="stylesheet" href="static/prediction.css">

 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css"  rel="stylesheet" integrity="sha384-
eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwra6"
  crossorigin="anonymous">
```

```
<link href="https://unpkg.com/tailwindcss@^2/dist/tailwind.min.css"
rel="stylesheet">
<title>prediction</title>
</head>
<body>
<script>
function valid(){
var Ai=document.getElementById("ApplicantIncome").value;
var Co=document.getElementById("CoapplicantIncome").value;
var LA=document.getElementById("LoanAmount").value;
var LT=document.getElementById("Loan_Amount_Term").value;
if(Ai > 100000000000000000000000000000000000000){
alert("Applicant income is too large enter a valid number")
return false;
}
if(Co > 100000000000000000000000000000000000000){
alert("Coapplicant income is too large enter a valid number")
return false;
}
if(LA >
100000000000000000000000000000000000000){  alert("L
oan Amount is too large enter a valid number")  return
false;
}
if(LT >
100000000000000000000000000000000000000){  alert("loan
amount term is too large enter a valid number")  return false;
}
```

```
var
name=document.getElementById("Name").value;  var
letters=/^[a-zA-Z]*$/;
if(!name.match(letters)){
alert("Name must contain only alphabets")
return false;
}
var num=/^[0-9]+$/;
if(!Ai.match(num)){
alert("Enter only valid numbers alphabets are not allowed
")  return false;
}
if(!Co.match(num)){
alert("Enter only valid numbers alphabets are not allowed
")  return false;
}
if(!LA.match(num)){
alert("Enter only valid numbers alphabets are not allowed
")  return false;
}
if(!LT.match(num)){
alert("Enter only valid numbers alphabets are not allowed
")  return false;
}
var mo=document.getElementById("mon").value;
var mn=/^[0-9]{10}$/;
if(!mo.match(mn)){
alert("Please enter only 10 digit mobile number")
```

```
return false;

}


}

</script>

<section class="text-gray-600 body-font">

<div class="container px-5 py-24 mx-auto">

<div class="flex flex-col text-center w-full mb-20">


<h1 class="Heading">LOAN ELIGIBILITY PREDICTION</h1><br>

<p class="fill">Fill the form for prediction</p>

</div>

<div>


</div>

<div class="mb-3">

<a class="btn btn-primary" href="./" id="back" role="button">Back</a></div>


<form action='/prediction.html' method="post" onsubmit="return

valid()">  <div class="mb-3">

<label for="exampleFormControlInput1" class="form-label">Name</label>

 <input type="text" class="form-control" id="Name" name="Name" placeholder="Enter
your  Name" required >

</div>

<div class="mb-3">

<label for="exampleFormControlInput1" class="form-label"> Email ID</label>
<input type="email" class="form-control" id="email" name="email" placeholder="Enter
your  Email ID" required >

</div>

<div class="mb-3">

<label for="exampleFormControlInput1" class="form-label">Mobile Number</label>
```

```
 <input type="text" class="form-control" id="mon" name="mon" placeholder="Enter your
Mobile  number" required>

</div>

<div class="mb-3">

 <label for="exampleFormControlInput1" class="form-label"> Gender</label>

 <select class="form-select" id="gender" name="gender" aria-label="Default select
example"  required >

 <option selected>-- select gender --</option>

 <option value="Male">Male</option>

 <option value="Female">Female</option>

</select>

</div>

<div class="mb-3">

 <label for="exampleFormControlInput1" class="form-label"> Married status</label>

 <select class="form-select" id="married" name="married" aria-label="Default select
example"  required >

 <option selected>-- select married status --</option>

 <option value="Yes">Yes</option>

 <option value="No">No</option>

</select>

</div>

<div class="mb-3">

 <label for="exampleFormControlInput1" class="form-label">Dependents</label>

 <select class="form-select" id="dependents" name="dependents" aria-label="Default
select  example" required>

 <option selected>-- select dependents --</option>

 <option value="0">0</option>

 <option value="1">1</option>
 <option value="2">2</option>

 <option value="3+">3+</option>

</select>

</div>
```

```
<div class="mb-3">
 <label for="exampleFormControlInput1" class="form-label">Education</label>
 <select class="form-select" id="education" name="education" aria-label="Default select example"  required>
 <option selected>-- select education --</option>
 <option value="Graduate">Graduate</option>
 <option value="Not Graduate">Not Graduate</option>
</select>
</div>
<div class="mb-3">
 <label for="exampleFormControlInput1" class="form-label">Self_Employed</label>
 <select class="form-select" id="employed" name="employed" aria-label="Default select example"  required>
 <option selected>-- select Self_Employed --</option>
  <option value="Yes">Yes</option>
 <option value="No">No</option>
</select>
</div>
<div class="mb-3">
 <label for="exampleFormControlInput1" class="form-label">Credit_History</label>
 <select class="form-select" id="credit" name="credit" aria-label="Default select example" required  >
 <option selected >-- select Credit_History --</option>
 <option value="Yes">Yes</option>
 <option value="No">No</option>


</select>
</div>
<div class="mb-3">
 <label for="exampleFormControlInput1" class="form-label">Property_Area</label>
 <select class="form-select" id="proparea" name="proparea" aria-label="Default select example"  required>
```

```
<option selected>-- select Property_Area --</option>

<option value="Semiurban">Semiurban</option>

<option value="Urban">Urban</option>

<option value="Rural">Rural</option>

</select>

</div>

<div class="mb-3">

<label for="exampleFormControlInput1" class="form-label">Enter ApplicantIncome</label>

<input type="text" class="form-control" id="ApplicantIncome"
name="ApplicantIncome"  placeholder="ApplicantIncome" required>


</div>

<div class="mb-3">

<label for="exampleFormControlInput1" class="form-label">Enter CoapplicantIncome</label>

<input type="text" class="form-control" id="CoapplicantIncome"
name="CoapplicantIncome"  placeholder="CoapplicantIncome" required>

</div>

<div class="mb-3">

<label for="exampleFormControlInput1" class="form-label">Purpose of loan</label>  <select

class="form-select" id="pur" name="pur" aria-label="Default select example"

required>  <option selected>-- select the purpose of loan --</option>

<option value="person">Personal loan</option>

<option value="Bussiness">Bussiness loan</option>

<option value="Education">Education loan</option>

<option value="Home">Home loan</option>

<option value="Other">other</option>

</select>

</div>
<div class="mb-3">

<label for="exampleFormControlInput1" class="form-label">Enter LoanAmount</label>
```

```
 <input type="text" class="form-control" id="LoanAmount"
name="LoanAmount"  placeholder="LoanAmount" required>

</div>

<div class="mb-3">

 <label for="exampleFormControlInput1" class="form-label">Enter Loan_Amount_Term</label>

 <input type="text" class="form-control" id="Loan_Amount_Term"
name="Loan_Amount_Term"  placeholder="Loan_Amount_Term" required>

</div>

<div class="mb-3">

 <label for="exampleFormControlInput1" class="form-label">Enter Adhar Number</label>

 <input type="text" class="form-control" id="Adhar" name="Adhar" placeholder="Adhar
Number"  required >

</div>

<div class="mb-3">

 <label for="exampleFormControlInput1" class="form-label">Enter PAN card ID</label>

 <input type="text" class="form-control" id="PAN " name="PAN " placeholder="PAN card
ID"  required>

</div>

<div class="mb-3">

<label for="property document" class="form-label">Property
Document</label><br><input  type="file" required >

</div>

<div class="mb-3">

<label for="Govt ID proof" class="form-label">Govet ID proof</label><br><input
type="file"  required>

</div>

<div class="mb-3">

 <input type="checkbox" required>

 I accept the <a href="terms.html">Terms and conditions</a>

 </div>

<br><br>
<div class="mb-3">

<input type="submit" class="but" value="PREDICT">
```

```
</div>

</form>



</div>

</section>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js"  integrity="sha384-
JEW9xMcG8R+pH31jmWH6WWP0WintQrMb4s7ZOdauHnUtxwoG2vI5DkLtS3qm9Ekf"
  crossorigin="anonymous"></script>


</body>

</html>
```

Training the machine learning model in ibm

In IBM Watson Knowledge Studio , the creation of the machine learning model involves training the machine learning model and evaluating how well the model performed when annotating test data and blind data.

Creating a machine learning model

When you create a machine learning model, you select the document sets that you want to use to train the model and specify the percentage of documents that are to be used as training data, test data, and blind data.

About this task

By exploring the performance metrics, you can identify ways to improve the

model's accuracy. **Procedure**

To create a machine learning model:

Log in as a Knowledge Studio administrator and select your workspace.

Select Machine Learning Model > Performance.

Verify that all of the document sets have been approved and that all annotation conflicts have been resolved through adjudication. Only documents that have become ground truth through adjudication or approval can be used to train the model.

Click Train and evaluate.

Click Train and evaluate.

See Document set management for help determining which ratios to apply.

Click Train to train the model, or click Train & Evaluate to train the model, evaluate annotations added by the machine learning model, and analyze the performance statistics.
Select the document sets that you want to use for training the model.

Evaluating annotations added by the model

You can compare the ground truth view for annotations added by human annotators to the annotations on the model.

BANK LOGIN

```
<!DOCTYPE html>

<html>

<head>

	<title>LogIn Page</title>

	<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"  crossorigin="anonymous">

</head>

<style>

 .group{

 padding-top: 100px;

 }

</style>

<body>




<div class="container">

	<div class="row">

		<div style="width: 40%; margin: 25px auto;">
```

```
<div class="group">

                        <h3 style="text-align: center;">Bank Login Page</h3>

                        <form method="POST" action="bank1.php">

                                    <div class="form-group">

                                                <label>Bank user ID:</label><input type="text"
name="BankUserName" class="form-control" autofocus placeholder="Enter the Bank User
ID"  required>

                                    </div>

                                    <div class="form-group">

                                                <label>Bank Email ID:</label><input type="email"
name="bankemail" class="form-control" autofocus placeholder="Enter the Bank Email ID" required>
                                    </div>

<div class="form-group">

                                                <label>Password:</label><input type="Password"
name="Password" class="form-control" autofocus placeholder="Password" required>

                                                            </div>



<label>Enter Captcha:</label>

<div class="form-row">

<div class="form-group col-md-6">

<input type="text" class="form-control" readonly id="capt"

required>  </div>

<div class="form-group col-md-6">

<input type="text" class="form-control" id="textinput" required>

</div>

</div>


            <div class="form-group">

                                                <button onclick="validcap()" name="Submit" class="btn btn-lg btn
success btn-block">Submit</button>

                                    </div>

                        </form>
```

```
 <h6>Captcha not visible <img src="refresh.png" width="40px" onclick="cap()"></h6>

            </div>

            </div>

</div>

</div>

<script type="text/javascript">

 function cap(){

 var alpha =

['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V'  ,'W','X','Y','Z','1','2',

'3','4','5','6','7','8','9','0','a','b','c','d','e','f','g','h','i',  'j','k','l','m','n','o','p','q','r','s','t','u','v','

w','x','y','z', '!','@','#','$','%','^','&','*','+'];

 var a = alpha[Math.floor(Math.random()*71)];

 var b = alpha[Math.floor(Math.random()*71)];

 var c = alpha[Math.floor(Math.random()*71)];

 var d = alpha[Math.floor(Math.random()*71)];

 var e = alpha[Math.floor(Math.random()*71)];

 var f = alpha[Math.floor(Math.random()*71)];


 var final = a+b+c+d+e+f;

 document.getElementById("capt").value=final;

 }

 function validcap(){

 var stg1 = document.getElementById('capt').value;

 var stg2 = document.getElementById('textinput').value;

 if(stg1==stg2){

 alert("Form is validated Succesfully");

 return true;

 }else{

 alert("Please enter a valid captcha");
```

```
 return false;

 }

 }

</script>

</body>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"  c
rossorigin="anonymous"></script>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"  integrity="sha
384-
ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"  crossorigin="anon
ymous"></script>

</html>
```

APPROVE HTML

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

 <head>

 <meta charset="utf-8">

 <title>Loan approva status</title>

 <link rel="stylesheet" href="static/approve.css">

 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font
awesome/5.15.3/css/all.min.css"/>

 </head>

 <body>

 <h1>LOAN APPROVAL STATUS</h1>

 <h2>{{prediction_text}}</h2>

 <img src="static/approve.jpg" width="150px" height="150px">  <h3>Please provide

your feedback</h3>

 <div class="container">



 <div class="post">
```

```
<div class="text">Thanks for rating us!</div>

<div class="edit">EDIT</div>

</div>

<div class="star-widget">

<input type="radio" name="rate" id="rate-5">

<label for="rate-5" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-4">

<label for="rate-4" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-3">

<label for="rate-3" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-2">

<label for="rate-2" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-1">
<label for="rate-1" class="fas fa-star"></label>

<form action="#">

<header></header>

<div class="textarea">

<textarea cols="30" placeholder="Describe your experience.."></textarea>  </div>

<div class="btn">

<button type="submit">Post</button>

</div>

</form>

</div>

</div>

<script>

const btn = document.querySelector("button");

const post = document.querySelector(".post");

const widget = document.querySelector(".star-widget");

const editBtn = document.querySelector(".edit");

btn.onclick = ()=>{
```

```
widget.style.display = "none";
post.style.display = "block";
editBtn.onclick = ()=>{
widget.style.display = "block";
post.style.display = "none";
}
return false;
}
</script>
</body>
</html>
```

HOME HTML

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Loan Prediction</title>
	<link rel="stylesheet" href="static/home.css">
</head>
<body>
	<div class="container">
	<div class="navbar">

		<nav>
			<ul>
			 <li><a href="home.html">Home</a></li>
					<li><a href="About.html">About</a></li>
<li><a href="procedure.html">Procedure</a></li>
					<li><a href="contact.html">Contact Us</a></li>
```

```
<li><a href="login.html">User login</a></li>
<li><a href="bank login.html">Bank login</a></li>
</ul>
</nav>

</div>
<div class="content">

<h1>Smart Lender - Applicant Credibility Prediction For Loan Approval </h1> <p>
Predit your loan eligibility here</p><br><br>
<a href="prediction.html" class="btn">PREDICT</a>
<br><br>
<h2>Team ID -PNT2022TMID39687</h2><br>
<h3>Team members</h3>
<p>KAMALI K</p>
<p>KOMALA DEVI M</p>
<p>ASIK A</p>
<p>DHIVYA M</p>

</div>

</div>
</body>
</html>
```

REGISTER HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Register</title>
<link rel="stylesheet" type="text/css"
```

```
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.c
ss">  <link rel="stylesheet" type="text/css" href="static/register.css">
 <script >
function check(x)
{

 var number=/^([0-9]{10})+$/;
 if(x.value.match(number)){
 alert("Valid email address!");
 document.myform.mon.focus();
 return true;

 }
 else{
 alert("Please enter only your 10 digit mobile number");
 document.myform.mon.focus();
 return false;

 }

}
function ValidateEmail(input) {

var validRegex = /^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/;
if (input.value.match(validRegex)) {

alert("Valid email address!");

document.myform.email.focus();

return true;
```

```
} else {

alert("Invalid email address!");

document.myform.email.focus();

return false;

}
 }

 </script>

</head>
<body>

 <div class="container">
 <form name="myform" method="post" class="form-signup"
onsubmit="return  check(document.myform.mon)" onsubmit="return
ValidateEmail(document.myform.email)">
 <h1 class="reg">Register</h1>
 <p>Create your account</p>
 <div class="form-group">
 <input type="text" class="form-control" name="name" placeholder="Enter your
name"  required >
 </div>

 <div class="form-group">
 <input type="email" class="form-control" name="email" placeholder="Enter your
emailID"  required>
 </div>

 <div class="form-group">
```

```
 <input type="user name" class="form-control" name="username" placeholder="Enter
your  username" required>

 </div>

 <div class="form-group">

 <input type="password" class="form-control" name="password" placeholder="Enter
your  password" required>

 </div>

 <div class="form-group">

 <input type="text" class="form-control" name="mon" placeholder="Enter your
mobile  number" required >

 </div>

 <div class="form-group">

 <label>

 <input type="checkbox">

 I accept the <a href="terms.html">Terms and conditions</a>

 </label>

 </div>

 <input type="submit" class="btn btn-success btn-block" name="" value="submit">
 </form>  </

div>

</body>

</html>
```

REJECT HTML

```
<!DOCTYPE html>

<html lang="en" dir="ltr">
```

```
<head>

<meta charset="utf-8">

<title>Loan approval status</title>

<link rel="stylesheet" href="static/reject.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font
awesome/5.15.3/css/all.min.css"/>

</head>

<body>

<h1>LOAN APPROVAL STATUS</h1>

<h2>{{prediction_text}}</h2>

<img src="static/reject.jpg" width="200px" height="200px">  <h3>Please provide

your feedback</h3>

<div class="container">


<div class="post">

<div class="text">Thanks for rating us!</div>

<div class="edit">EDIT</div>

</div>

<div class="star-widget">

<input type="radio" name="rate" id="rate-5">

<label for="rate-5" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-4">

<label for="rate-4" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-3">

<label for="rate-3" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-2">

<label for="rate-2" class="fas fa-star"></label>

<input type="radio" name="rate" id="rate-1">
<label for="rate-1" class="fas fa-star"></label>

<form action="#">

<header></header>
```

```
<div class="textarea">
<textarea cols="30" placeholder="Describe your experience.."></textarea>  </div>
<div class="btn">
<button type="submit">Post</button>
</div>
</form>
</div>
</div>
<script>
const btn = document.querySelector("button");
const post = document.querySelector(".post");
const widget = document.querySelector(".star-widget");
const editBtn = document.querySelector(".edit");
btn.onclick = ()=>{
widget.style.display = "none";
post.style.display = "block";
editBtn.onclick = ()=>{
widget.style.display = "block";
post.style.display = "none";
}
return false;
}
</script>
</body>
</html>
```

SPRINT

```
import numpy as np
import seaborn as sb
import pandas as pd
from pandas_profiling import ProfileReport
```

```
import plotly.express as px
import plotly.graph_objects as go
from matplotlib import pyplot as plt
```

In [2]:

```
df = pd.read_csv("loan_prediction.csv")
df.head()
```

Out[2]:

Loa n_I D

LP0
Ge nd er

M
Ma rri ed
Dep ende nts
Edu cati on

Gra
Self_ Empl oyed
Applic antInc ome
Coappli cantInc ome
Loan Amo unt
Loan_A mount_ Term
Credi t_Hist ory
Prope rty_A rea
Loa n_St atus

0
010 02
ale No 0
duat e
No 5849 0.0 NaN 360.0 1.0 Urban Y

1 2

3
LP0 010 03

LP0 010 05

LP0 010 06

LP0
M ale

M ale

M ale

M

Ye
s1

Ye
s0

Ye
s0
Gra
duat e

Gra
duat e

Not Gra
duat e

Gra

No 4583 1508.0 128.0 360.0 1.0 Rural N Yes 3000 0.0 66.0 360.0 1.0 Urban Y

No 2583 2358.0 120.0 360.0 1.0 Urban Y

4
010 08
ale No 0
duat e

No 6000 0.0 141.0 360.0 1.0 Urban Y In [3]:

```
df.info()
```

```
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
```

In [4]:

```
profile = ProfileReport(df, title="Analysis report for data
Analysis") profile.to_notebook_iframe()
profile.to_file("data_analysis.html")
```

```
Summarize dataset: 0%| | 0/5 [00:00, ?it/s]
Generate report structure: 0%| | 0/1 [00:00, ?it/s] Render
HTML: 0%| | 0/1 [00:00, ?it/s]
```

Analysis report for data Analysis
[Analysis report for data Analysis]

- [Overview]
- [Variables]
- [Interactions]
- [Correlations]
- [Missing values]
- [Sample]

Dataset statistics

Number of variables 13

Number of observations 614

Missing cells 149

Missing cells (%) 1.9%

Duplicate rows 0

Duplicate rows (%) 0.0%

Total size in memory 62.5 KiB

Average record size in memory 104.2 B

Variable types

Categorical 6

Boolean 3

Numeric 4

Alerts

Loan_ID has a high cardinality: 614 distinct values High cardinality

ApplicantIncome is highly correlated with LoanAmount High correlation

LoanAmount is highly correlated with ApplicantIncome High correlation

ApplicantIncome is highly correlated with LoanAmount High correlation

LoanAmount is highly correlated with ApplicantIncome High correlation

Loan_Status is highly correlated with Credit_History High correlation

Credit_History is highly correlated with Loan_Status High correlation

Gender is highly correlated with Married High correlation Married is highly correlated with Gender and 1 other fields High correlation Dependents is highly correlated with Married High correlation ApplicantIncome is highly correlated with LoanAmount High correlation LoanAmount is highly correlated with ApplicantIncome High correlation Credit_History is highly correlated with Loan_Status High correlation Loan_Status is highly

correlated with Credit_History High correlation Gender has 13 (2.1%)

missing values Missing

Dependents has 15 (2.4%) missing values Missing Self_Employed

has 32 (5.2%) missing values Missing LoanAmount has 22 (3.6%)

missing values Missing Loan_Amount_Term has 14 (2.3%) missing

values Missing Credit_History has 50 (8.1%) missing values Missing

Loan_ID is uniformly distributed Uniform

Loan_ID has unique values Unique

CoapplicantIncome has 273 (44.5%) zeros Zeros

Reproduction

Analysis started 2022-11-09 09:43:02.856825

Analysis finished 2022-11-09 09:43:17.092955

Duration 14.24 seconds

Software version pandas-profiling v3.2.0

Download configuration config.json

Variables

Loan_ID
Categorical

HIGH CARDINALITY
UNIFORM
UNIQUE

Distinct 614

Distinct (%) 100.0%
Missing 0

Missing (%) 0.0%

Memory size 4.9 KiB

LP001002	1
LP002328	1
LP002305	1
LP002308	1
LP002314	1
Other values (609)	609

- Overview
- Categories
- Words
- Characters

Length

Max length 8

Median length 8

Mean length 8

Min length 8

Characters and Unicode

Total characters 4912
Distinct characters 12

Distinct categories 2 ?

Distinct scripts 2 ?

Distinct blocks 1 ?

The Unicode Standard assigns character properties to each code point, which can be used to analyse textual variables.
Unique

Unique 614 ?

Unique (%) 100.0%

Sample

1st row LP001002

2nd row LP001003

3rd row LP001005

4th row LP001006

5th row LP001008

Common Values

Value	Count	Frequency (%)
LP001002	1	0.2%
LP002328	1	0.2%
LP002305	1	0.2%

Value	Count	Frequency (%)
LP002308	1	

0.2%

LP002314 1
0.2%

LP002315 1
0.2%

LP002317 1
0.2%

LP002318 1
0.2%

LP002319 1
0.2%

LP002332 1
0.2%

Other values (604) 604 98.4%

Length

Histogram of lengths of the category Value Count
Frequency (%)

lp001002 1
0.2%

lp001014 1
0.2%
Value Count Frequency (%)

lp001038 1
0.2%

lp001036 1
0.2%

lp001005 1
0.2%

lp001006 1
0.2%

lp001008 1
0.2%

lp001011 1
0.2%

lp001013 1
0.2%

lp001018 1
0.2%

Other values (604) 604 98.4%

- Characters
- Categories
- Scripts
- Blocks

Most occurring characters

Value Count

Frequency (%)

0 1403 28.6%
Value Count Frequency (%)

L 614 12.5%

P 614 12.5%

1 491
10.0%

2 478
9.7%

4	203	4.1%
3	198	4.0%
8	189	3.8%
7	183	3.7%
9	182	3.7%
Other values (2)	357	7.3%

Most occurring categories

Value	Count	Frequency (%)
Decimal Number	3684	75.0%
Uppercase Letter	1228	25.0%

Most frequent character per category ***Decimal Number***

Value	Count	Frequency (%)
0	1403	38.1%
1	491	13.3%

Value	Count	Frequency (%)
2	478	13.0%
4	203	5.5%
3	198	5.4%
8	189	5.1%
7	183	5.0%
9	182	4.9%
6	181	4.9%
5	176	4.8%

Uppercase Letter

Value	Count	Frequency (%)
L	614	50.0%
P	614	50.0%

Most occurring scripts

Value	Count	Frequency (%)
Common	3684	75.0%

Latin	1228	25.0%

Most frequent character per script *Common*

Value	Count	Frequency (%)
0	1403	38.1%
1	491	13.3%
2	478	13.0%
4	203	5.5%
3	198	5.4%
8	189	5.1%
7	183	5.0%
9	182	4.9%
6	181	4.9%
5	176	4.8%

Latin

Value	Count	Frequency (%)
L	614	50.0%
P	614	50.0%

Most occurring blocks

Value	Count	Frequency (%)
ASCII	4912	100.0%

Most frequent character per block

ASCII

Value	Count	Frequency (%)
0	1403	28.6%
L	614	12.5%
P	614	12.5%
1	491	10.0%
2	478	9.7%
4	203	4.1%
3	198	4.0%
8	189	3.8%

7 183	3.7%
9 182	3.7%
Other values (2) 357	7.3%

[Gender](#)
Categorical

HIGH CORRELATION
MISSING

Distinct 2

Distinct (%) 0.3%

Missing 13

Missing (%) 2.1%

Memory size 4.9 KiB

Male 489

Female 112

- [Overview](#)
- [Categories](#)
- [Words](#)
- [Characters](#)

Length

Max length 6

Median length 4

Mean length 4.372712146

Min length 4

Characters and Unicode

Total characters 2628

Distinct characters 6

Distinct categories 2 ?

Distinct scripts 1 ?

Distinct blocks 1 ?

The Unicode Standard assigns character properties to each code point, which can be used to analyse textual variables.

Unique

Unique 0 ?

Unique (%) 0.0%

Sample

1st row Male

2nd row Male

3rd row Male

4th row Male

5th row Male

Common Values

Value	Count	Frequency (%)
Male	489	79.6%
Female	112	18.2%
(Missing)	13	2.1%

Length

Histogram of lengths of the

category **Category Frequency**

Plot

Value Count Frequency (%)

male 489 81.4%
Value Count Frequency (%)

female 112
18.6%

- [Characters](#)
- [Categories](#)
- [Scripts](#)
- [Blocks](#)

Most occurring characters

Value Count Frequency (%)

e 713 27.1%

a 601 22.9%

l 601 22.9%

M 489 18.6%

F 112
4.3%

m 112
4.3%

Most occurring categories

Value Count Frequency (%)

Lowercase Letter	2027	77.1%
Uppercase Letter	601	22.9%

Most frequent character per category

Lowercase Letter

Value	Count	Frequency (%)
e	713	35.2%
a	601	29.6%
l	601	29.6%
m	112	5.5%

Uppercase Letter

Value	Count	Frequency (%)
M	489	81.4%
F	112	18.6%

Most occurring scripts

Value	Count	Frequency (%)
Latin	2628	100.0%

Most frequent character per script *Latin*

Value	Count	Frequency (%)

e 713 27.1%

a 601 22.9%
Value Count Frequency (%)

l 601 22.9%

M 489 18.6%

F 112
4.3%

m 112
4.3%

Most occurring blocks

Value	Count	Frequency (%)
ASCII	2628	100.0%

Most frequent character per block *ASCII*

Value	Count	Frequency (%)
e	713	27.1%
a	601	22.9%
l	601	22.9%
M	489	18.6%
F	112	4.3%
m	112	

4.3%

Married
Boolean

HIGH CORRELATION

Distinct 2

Distinct (%) 0.3%

Missing 3

Missing (%) 0.5%

Memory size 1.3 KiB

True 398

False 213

(Missing)

3

- Common Values
- Category Frequency Plot

Value	Count	Frequency (%)
True	398	64.8%
False	213	34.7%
(Missing)	3	0.5%

Dependents
Categorical

HIGH CORRELATION
MISSING

Distinct 4
Distinct (%) 0.7%

Missing 15

Missing (%) 2.4%

Memory size 4.9 KiB

0 345

1 102

2 101

3+ 51

- Overview
- Categories
- Words
- Characters

Length

Max length 2

Median length 1

Mean length 1.085141903

Min length 1

Characters and Unicode

Total characters 650

Distinct characters 5

Distinct categories 2 ?

Distinct scripts 1 ?
Distinct blocks 1 ?

The Unicode Standard assigns character properties to each code point, which can be used to analyse textual variables.
Unique

Unique 0 ?

Unique (%) 0.0%

Sample

1st row 0

2nd row 1

3rd row 0

4th row 0

5th row 0

Common Values

Value	Count	Frequency (%)
0	345	56.2%
1	102	16.6%
2	101	16.4%
3+	51	8.3%
(Missing)	15	2.4%

Length

Histogram of lengths of the category **Category Frequency Plot**

Value	Count	Frequency (%)
0	345	57.6%
1	102	17.0%
2	101	16.9%
3	51	8.5%

- Characters
- Categories
- Scripts
- Blocks

Most occurring characters

Value	Count	Frequency (%)
0	345	53.1%
1	102	15.7%
2	101	15.5%
3	51	7.8%

Value	Count	Frequency (%)
+	51	7.8%

Most occurring categories

Value	Count	Frequency (%)

Decimal Number	599	92.2%
Math Symbol	51	7.8%

Most frequent character per category

Decimal Number

Value	Count	Frequency (%)
0	345	57.6%
1	102	17.0%
2	101	16.9%
3	51	8.5%

Math Symbol

Value	Count	Frequency (%)
+	51	100.0%

Most occurring scripts

Value	Count	Frequency (%)
Common	650	100.0%

Most frequent character per script

Common

Value	Count	Frequency (%)

0 345 53.1%

1 102 15.7%

2 101 15.5%

3 51 7.8%

+ 51 7.8%

Most occurring blocks

Value	Count	Frequency (%)
ASCII	650	100.0%

Most frequent character per block *ASCII*

Value	Count	Frequency (%)
0	345	53.1%
1	102	15.7%
2	101	15.5%
3	51	7.8%
+	51	

7.8%

Education
Categorical

Distinct 2

Distinct (%) 0.3%

Missing 0

Missing (%) 0.0%

Memory size 4.9 KiB

Graduate 480 **Not Graduate**

134

- Overview
- Categories
- Words
- Characters

Length
Max length 12

Median length 8

Mean length 8.872964169

Min length 8

Characters and Unicode

Total characters 5448

Distinct characters 10

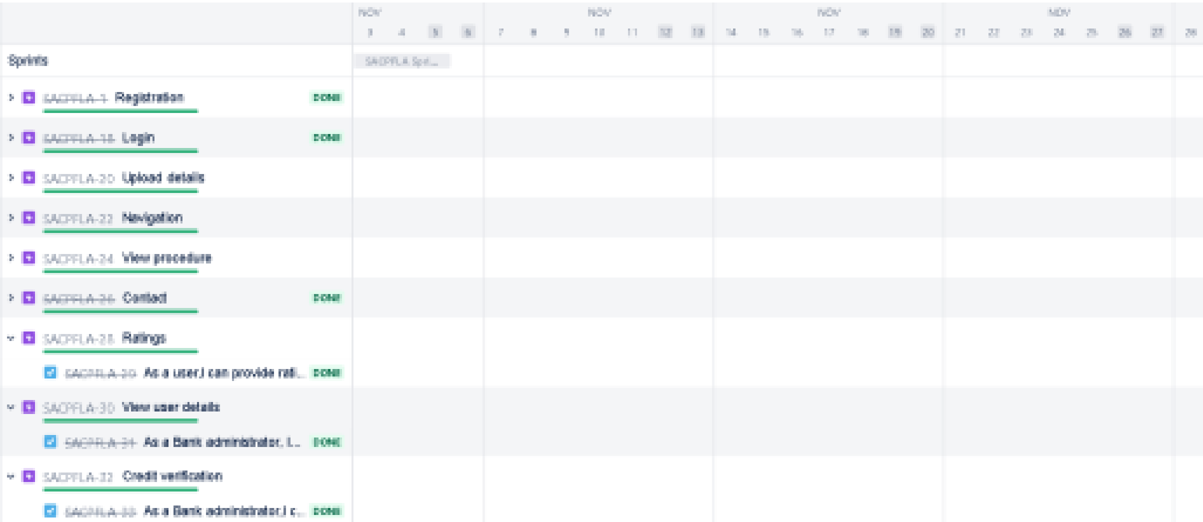
Distinct categories 3 ?

Distinct scripts 2 ?

Distinct blocks 1 ?

The Unicode Standard assigns character properties to each code point, which can be used to analyse textual variables.
Unique

Sprint



APPS:

```
from flask import render_template,Flask,request

import numpy as np

import pickle

from sklearn.preprocessing import scale app=

Flask(__name__, template_folder='templates')



model = pickle.load(open("Rfmodel.pkl",'rb'))




@app.route('/')
```

```
def home():
 return render_template('home.html')
@app.route('/login.html')
def login():
 return render_template('login.html')
@app.route('/procedure.html')
def procedure():
 return render_template('procedure.html')
@app.route('/bank login.html')
def bank():
 return render_template('bank login.html')
@app.route('/About.html')
def about():
 return render_template('About.html')
@app.route('/terms.html')
def terms():
 return render_template('terms.html')
@app.route('/register.html')
def register():
 return render_template('register.html')
@app.route('/contact.html')
def contact():
 return render_template('contact.html')
@app.route('/home.html')
def home1():
 return render_template('home.html')
@app.route('/prediction.html')
def formpg():
 return render_template('prediction.html')
```

```
@app.route('/rating.html')
def rat():
 return render_template('rating.html')
@app.route('/prediction.html',methods = ['POST'])
def predict():
 if request.method=='POST':
 name=request.form['Name']
 gender=request.form['gender']
 married=request.form['married']
 dependents=request.form['dependents']
 education=request.form['education']
 employed=request.form['employed']
 credit=request.form['credit']
 proparea=request.form['proparea']
 ApplicantIncome=float(request.form['ApplicantIncome'])  Coapplica
ntIncome=float(request.form['CoapplicantIncome'])  LoanAmount=fl
oat(request.form['LoanAmount'])  Loan_Amount_Term=float(request
.form['Loan_Amount_Term'])  if gender == 'Male':
 gender = 1
 else:
 gender = 0
 if married == 'Yes':
 married = 1
 else:
 married = 0


 if education ==
'Graduate':  education = 0
 else:
```

education = 1

if employed ==

'Yes': employed = 1

else:

employed = 0

if dependents ==

'3+': dependents = 3

if credit == 'Yes':

credit = 1

else:

credit = 0

if proparea ==

'Urban': proparea = 2

elif proparea ==

'Rural': proparea = 0

else:

proparea = 1

features =
[gender,married,dependents,education,employed,ApplicantIncome,CoapplicantIncome,LoanAmoun
t,Loan_Amount_Term,credit,proparea]

con_features = [np.array(features)]

```
prediction = model.predict(con_features)

print(prediction)

if prediction==1:

return render_template('approve.html',prediction_text ='Congratulations! '+name+' You
are  eligible for loan')

else:

return render_template('reject.html',prediction_text ='Sorry '+name+' You are not eligible
for  loan')



if __name__ == "__main__":

app.run(debug=True)
```

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.

IBM-EPBL / **IBM-Project-117-1658211897** Public

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights

main | 1 branch | 0 tags | Go to file | Add file | Code

kamalikumar123 Add files via upload — cc4a6fd 8 days ago — **23** commits

Application building	Add files via upload	8 days ago
IBM/Iteration phase	Add files via upload	21 days ago
Train the model on ibm	Add files via upload	8 days ago
assignment view	Add files via upload	21 days ago
data development phase	Add files via upload	8 days ago
data pre-processing	Add files via upload	8 days ago
dataset	Add files via upload	16 days ago
model building	Add files via upload	8 days ago
pre requities	Add files via upload	15 days ago
project design and plan	Add files via upload	21 days ago

pre requities	Add files via upload	15 days ago
project design and plan	Add files via upload	21 days ago
project planning phase	Add files via upload	15 days ago
visualizing and analyzing data	Add files via upload	8 days ago
PROJECT FLOW.pdf	Add files via upload	15 days ago
PROJECT OBJECTIVE smart lender.pdf	Add files via upload	15 days ago
PROJECT STRUCTURES.pdf	Add files via upload	15 days ago
Prior Knowledges.pdf	Add files via upload	15 days ago
Project Plannings.pdf	Add files via upload	16 days ago
README.md	Create README.md	21 days ago
Smart_Lender_Applicant_Credibility_P...	Create Smart_Lender_Applicant_Credibility_Prediction_For_Loan_Approv...	16 days ago
code	Create code	16 days ago
project development	Create project development	16 days ago

README.md

IBM-Project-117-1658211897