Smart Lender - Applicant Credibility Prediction for Loan Approval

TEAM ID: PNT2022TMID37263

Team Leader

Harikrishnan S

Team Members

Arul Kumaran M

Aniruth V

Azhagu Rajesh L

CHAPTER -1
INTRODUCTION

1. INTRODUCTION

Banks make the majority of their income through loans. Loan approval is a critical step for financial institutions. It is extremely difficult to estimate the probability of loan repayment by customers due to a growing incidence of loan defaults, and banking authorities are finding it increasingly difficult to appropriately access loan requests and address the dangers of individuals defaulting on loans. Many scholars have focused on loan approval system prediction in recent years. Machine learning is a powerful tool for predicting outcomes from massive amounts of data. A large amount of a bank's assets are directly derived from interest earned on loans made. Lending loans has significant risks, including the borrower's inability to repay the loan within the time frame specified. It is known as "credit risk." The worthiness of an applicant for loan acceptance or rejection was determined by a numerical score known as a "credit score." As a result, the use of various Machine Learning approaches that properly identify people to lend to and assist banks in identifying loan defaulters for much-reduced credit risk.To anticipate client loan acceptance, four algorithms are used: the Random Forest method, the Decision Tree algorithm, the KNN algorithm, and the XGBoost algorithm. All four methods will be run on the same dataset to select the approach with the highest accuracy for deploying the model. We will now create a bank loan prediction system listing machine learning techniques, so that the system will automatically identify the most qualified people to authorize the loan.

1.1 Project Overview

Both bank workers and applicants benefit greatly from loan prediction. The purpose is to provide a quick and simple method for selecting competent applications. They are present in every urban, suburban, and rural area. After that company or bank verifies the consumer's loan eligibility, the customer choose whether or not to apply for a loan. Loan approval or rejection will be offered to applicants based on the criteria.

1.2 Purpose

The loan is among the most important financial tools. To entice customers to submit loan applications, every bank is working to develop effective marketing strategies. After their applications are approved, some customers exhibit poor behaviour. For banks to find a solution, customer behaviour prediction methods must be developed. Machine learning algorithms that are effective for this usage are often used in the banking sector. To predict loan behaviour in this case, I'll use machine learning models. the best model to use, and the identification of pertinent characteristic.

CHAPTER-2
LITERATURE SURVEY

2.1 Existing Problem

There are many existing solutions deployed for this use case. A. loan prediction model using Machine Learning (ML) algorithms • The dataset with features, namely, gender, marital status, education, number of dependents, employment status, income, co applicant‘s income, loan amount, loan tenure, credit history, existing loan status, and property area, are used for determining the loan eligibility regarding the loan sanctioning process • Various ML models adopted in the present method includes, Linear model, Decision Tree (DT), Neural Network (NN), Random Forest (RF), SVM, Extreme learning machines, Model tree, Multivariate Adaptive Regression Splines, Bagged Cart Model, NB and TGA. When evaluated these models using Environment in five runs, TGA resulted in better loan forecasting performance than the other methods. B. Loan prediction model based on the data mining techniques • Data mining techniques, such as Decision Tree , Naïve Bayes (NB) and Bayse Net approaches. • The procedure followed was training set preparation, building the model, Applying the model and finally. Evaluating the accuracy. • This approach was implemented using Weka Tool and considered a dataset with eight attributes, namely, gender, job, age, credit amount, credit history, purpose, housing, and class. Evaluating these models on the dataset, experimental results concluded that, Decision Tree based loan prediction approach resulted in better accuracy than the other methods.

2.2 References

1.Arun Kumar, Ishan Garg and Sanmeer Kaur, Loan Approval Prediction based on Machine Learning Approach.

2. Mohamed El Mohadab, Belaid Bouikhalene and Said Safi, "Predicting rank for scientific research papers using supervised learning", Applied Computing and Informatics, vol. 15, pp. 182-190, 2019.

3. K. Hanumantha Rao, G. Srinivas, A. Damodhar and M. Vikas Krishna, "Implementation of Anomaly Detection Technique Using Machine Learning Algorithms", International Journal of Computer Science and Telecommunications, vol. 2, no. 3, June 2011.

4. J.R. Quinlan, Induction of decision trees, Machine learning Springer, vol. 1, no. 1, pp. 81-106,1086.

5. S.S. Keerthi and E.G. Gilbert, Convergence of a generalize SMO algorithm for SVM classifierdesign, Machine Learning, Springer, vol. 46, no. 1, pp. 351-360, 2002.

6. J.M. Chambers, "Computational methods for data analysis" in Applied Statistics, Wiley, vol.1, no. 2, pp. 1-10.

2.3 Problem Statement Definition

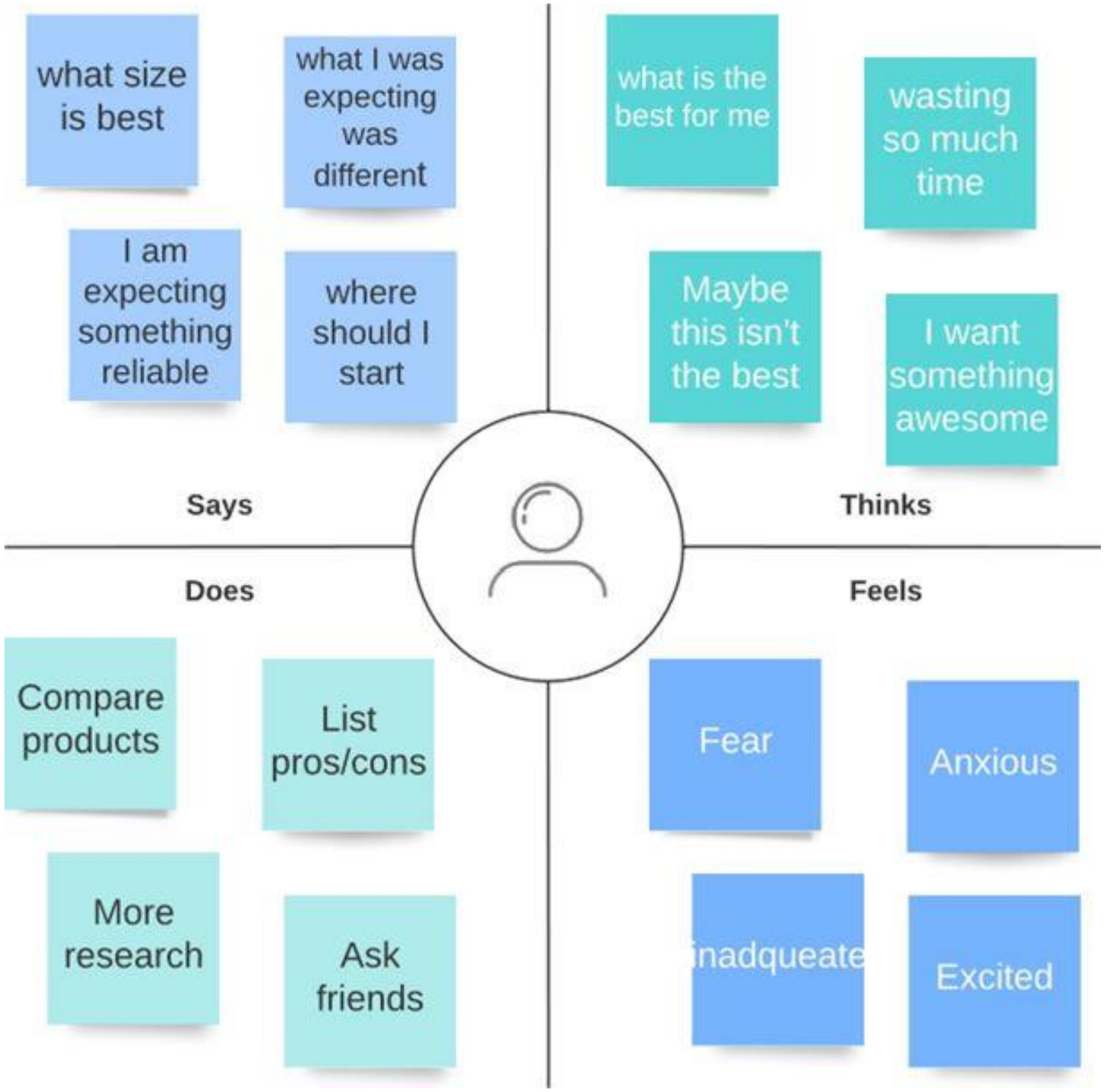
Every person must be prepared to face circumstances in which he/she might have to borrow money from a bank. While credit cards have made short-term financing quite easy, some individuals sometimes seek a particular type of loan. For example, a person building a house may seek a house-building loan or a person looking forward to purchasing a car might look for an option in that particular section. Thus, knowing the proper steps to applying for a loan is extremely necessary in order to select the best option and successfully achieving it. By this creditability testing by the use of machine learning bank can get source about the he/she. This will help them to provide the loan. Based on the credibility derived the bank can decide for the loan.

CHAPTER-3

IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

The Empathy Map Canvas helps teams develop deep, shared understanding and empathy for other people. People use it to help them improve customer experience, to navigate organizational politics, to design better work environments, and a host of other things.



3.2 Ideation And Brainstroming

1. CREATE A BEAUTIFUL INTERFACE TO GIVE THE BEST USER EXPERIENCE .

2. User will enter their bank detail , salary details along with the loan money required .

3. These details will be forwarded to the backend and verification is done.

4. The model's algorithm will get the input and process .

5. ML algorithm predicts the creditability of the person using applied data science .

6. Model compares the creditability of the person with the loan amount requested .

7. ADS method will train the model to give more accurate predictions .

8. The result of the comparison is now transmitted to Front end

Proposed Solution

Overview

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

Goals

1. This feature will enable the banks to predict accurately if the customer can repay the loan on time or not.
2. To provide a loan to a deserving applicant out of all applicants.

Specifications

HARDWARE SPECIFICATION

The hardware requirements may serve as the basis for a contract for the implementation of the system and should

therefore be a complete engineer as the starting point for the system design.

Ram : 6GB Ram or more

Processor : Any Processor

GPU : 6GB or more

Hard Disk : 10GB or more

Speed : 1.4GHZ or more

SOFTWARE SPECIFICATION

The software requirements give detailed description of the system and all its features.

➜ Python

➜ Keras

➜ Tensorflow

➜ Numpy

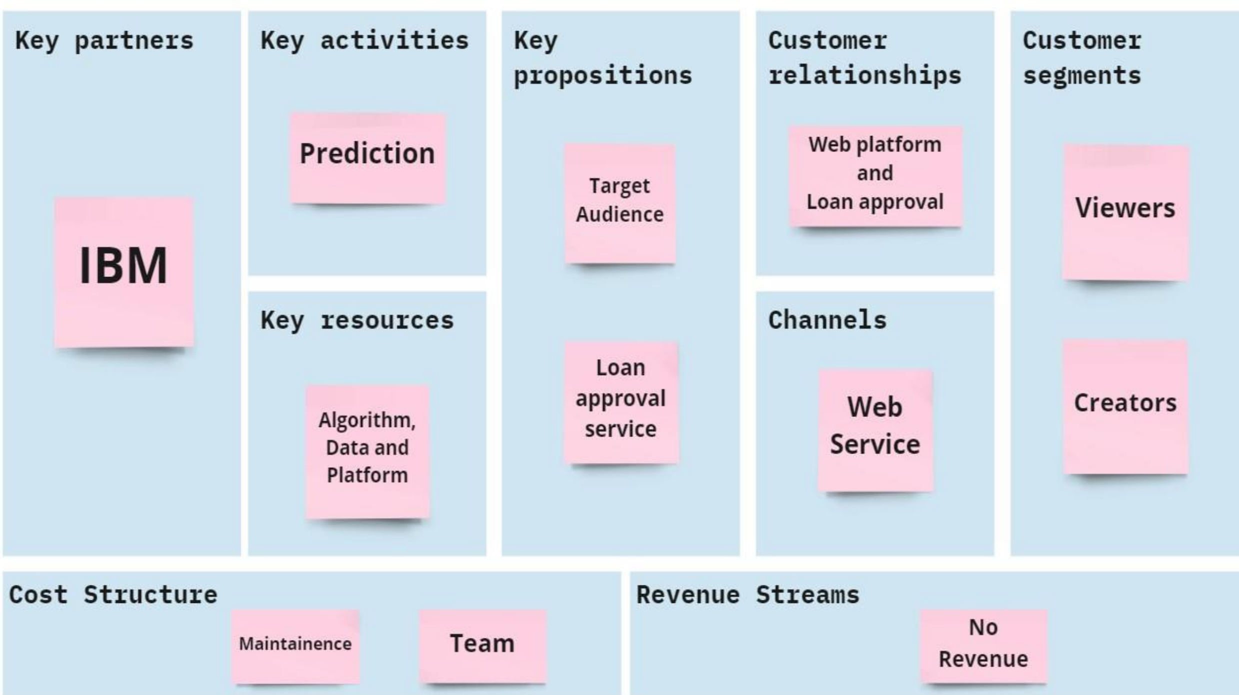
➜ Pandas 2

➜ Visual studio code

➜ Python-Flask

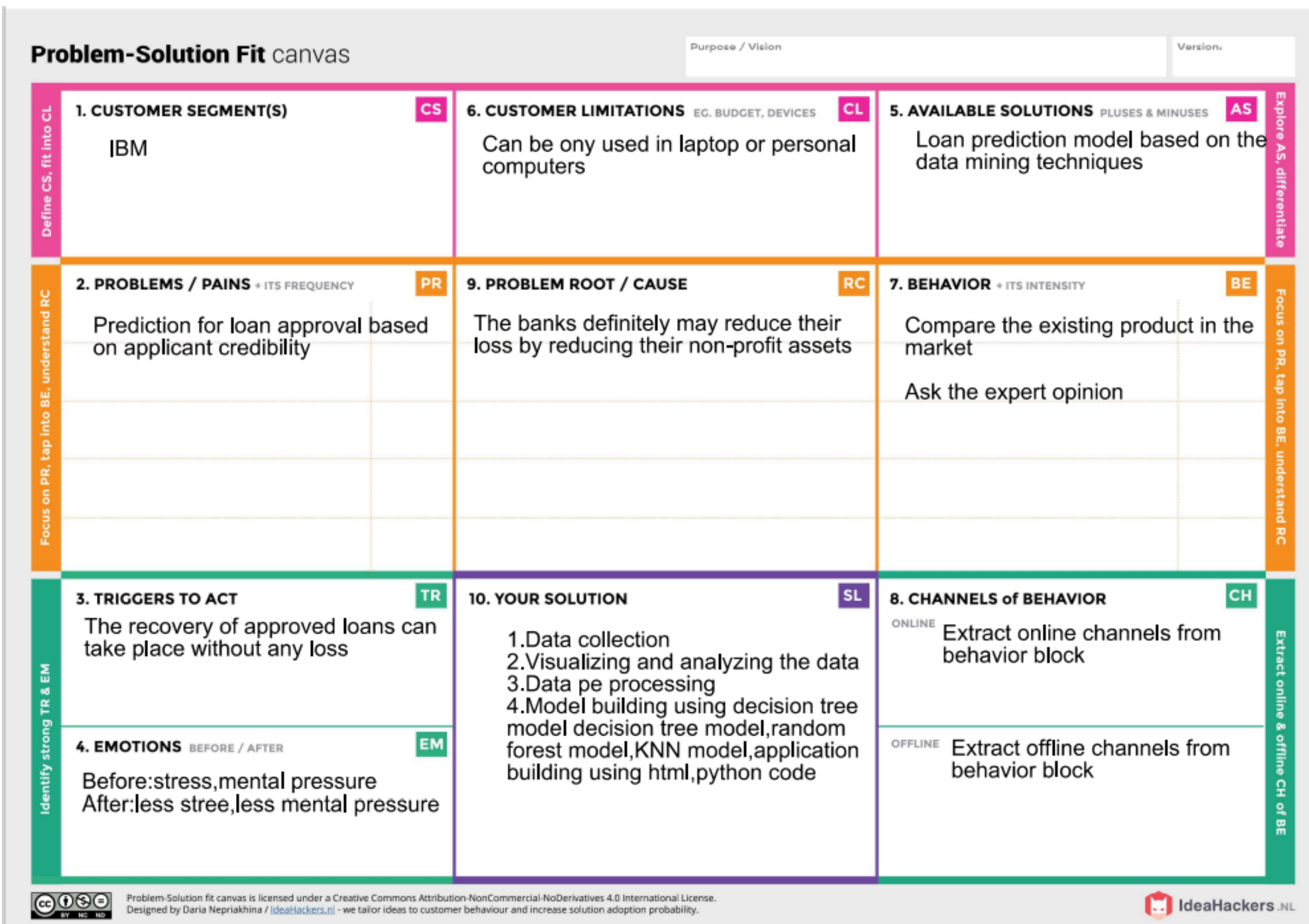
➜ IBM cloud

BUSINESS MODEL



3.3 Problem Solution Fit

The problem-Solution Fit basically implies that you identified a problem with your consumer and that the solution you devised genuinely solves the problem.Problem solution fit deals to have customer segments,Jobs to be done/Problems,Triggers, Customer Constraints,Problem root cause,General Solutions,Behavior and Available solutions.



CHAPTER-4

REQUIREMENT ANALYSIS

4.1 Functional Requirement

High Priority

1. Credit score and credit history are essential parameters for assessing the applicant's
lending risk

2. The history of their credit and debit card transactions are also taken in to the data processing and loan approval

3. The overall transactions and cibil score of the applicant's account are taken in to data
processing and loan approval

Medium Priority

1. The system shall provide following facility that will allow application that the user is permitted to access. The system must support the following facility:

a. Loan approval results.

b. Customer data management.

Low Priority

1. The system shall allow the user's status to be stored for the next time he returns to the
web site. This will save the user x minutes per visit by not having to reenter already supplied data.

2. The system shall provide information about the basic eligibility and requirements for the
loan approval

4.2 Non-Functional Rerquirements

A non-functional requirement (NFR) is one that defines criteria for judging the functioning of a system rather than particular behaviors. They differ from functional requirements, which describe precise behavior or functions. The system design includes a thorough plan for accomplishing functional requirements. Because non-functional needs are frequently architecturally significant, the plan for accomplishing them is outlined in the system architecture.

Reliability :

- The system shall be completely operational at least x% of the time.
- Down time after a failure shall not exceed x hours.

Usability :

- Customer should be able to use the system in his job for x days .
- A user who already knows what requirements should be required for the loan approval , the user should able to directly access the application.

Performance :

- The system should be able to support x simultaneous users.
- The mean time to view a application over a 56Kbps modem connection shall not exceed x seconds.

Security:

- The system shall provide password protected access to application that are to be viewed only by users.

Supportability :

- The system should be able to accommodate many customer datasets.
- The system application shall be viewable from chrome or any browser.

Interfaces :

- The system must interface with
- The cloudant db for customer and customer log information The acquired web site search engine.

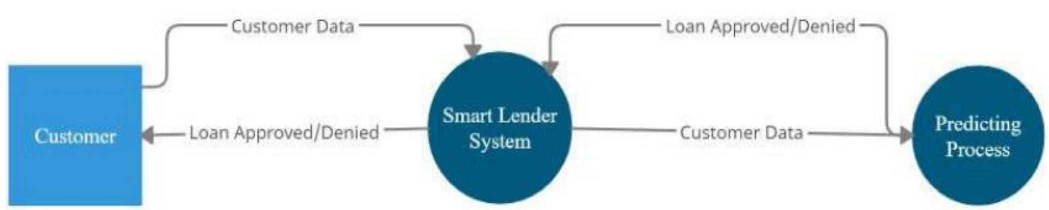
CHAPTER-5

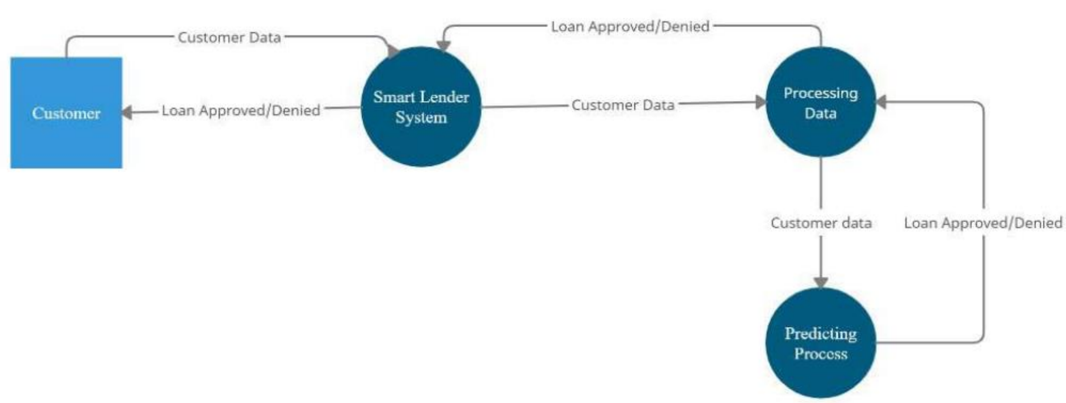
PROJECT

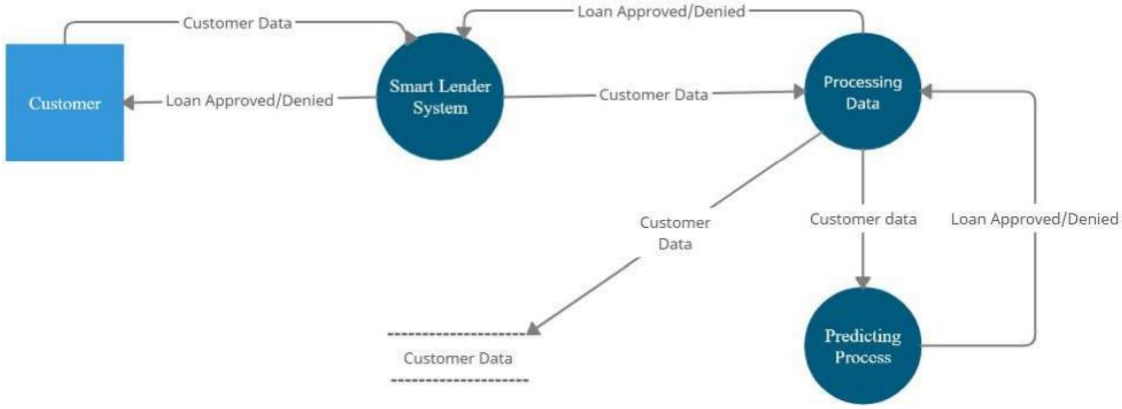
DESIGN

5.1 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the systemrequirement graphically. It shows how data entersand leaves the system, what changes the information, and where data is stored.Geneerally it shows clear view of thethe system requirements.The bank loan prediction deals to Manage the loan records,Monitor payments,Manage Applicant Information,Check and update the loan.





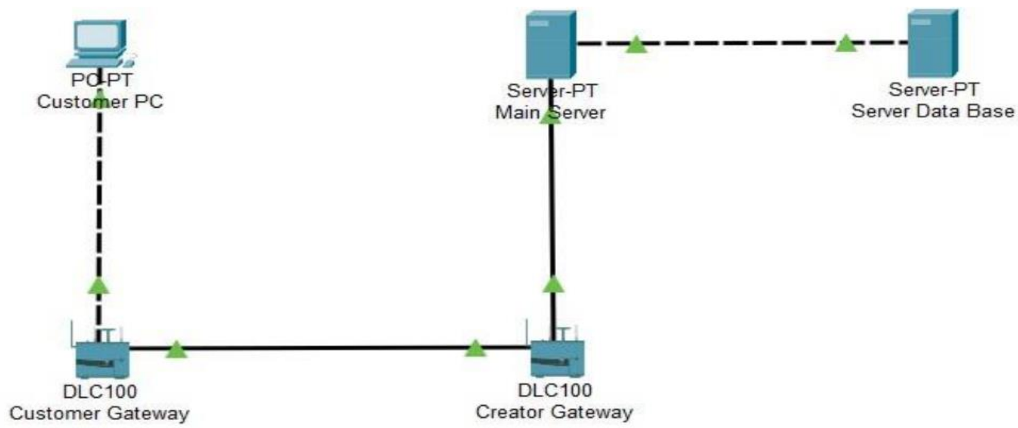


5.2 Solution And Technical ArchitectureSolution Architecture

The process of designing solutions based on predetermined procedures, rules, and best practises with the goal of ensuring that the generated solution fits inside the corporate architecture in terms of information architecture, system portfolios, integration needs, and other factors. It may therefore be defined as a set of roles, procedures, and documentation aimed at addressing specific business goals, requirements, or challenges through the design and development of applications and information systems.

Technical Architecture:

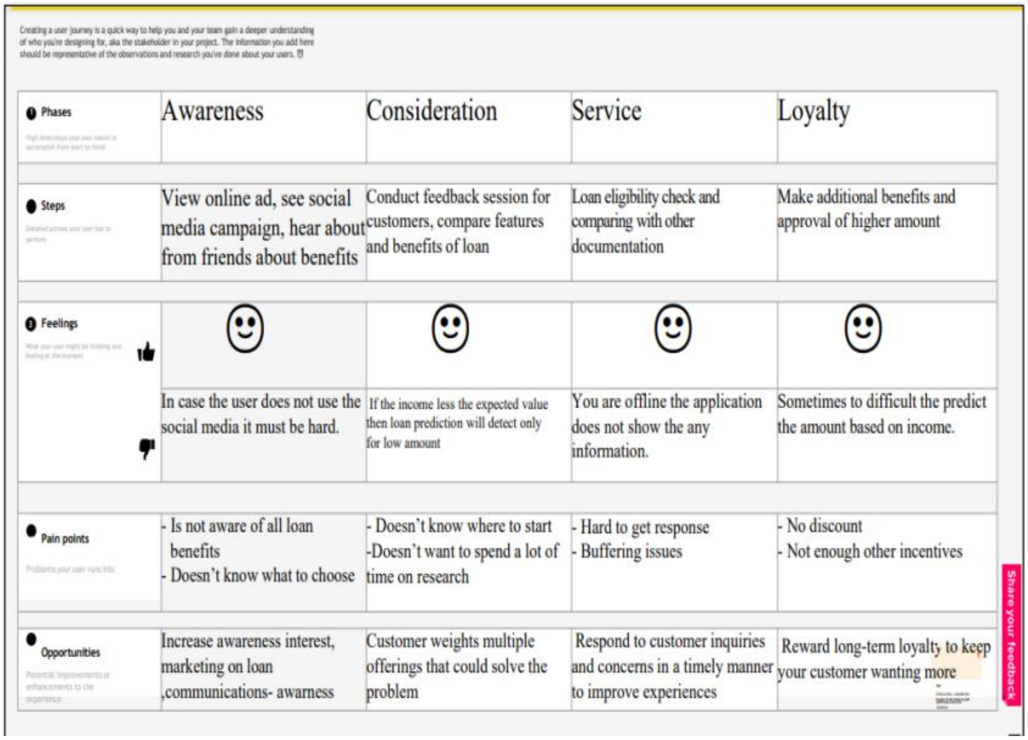
In Front End ,it involves to create the User Interface by using HTML,CSS.In Back End, it contains Customer Dashboard,Admin Dashboard,Approval Dashboard,Customer Dashboard connect to the Application Form and Application Status.Admin Dashboard connects to the manage application.Approval Dashboard connects to the Application Approval and Application Verification.In IBM Cloud which which contains Database and Loan Predicting Windows.Database connects to the Customer Dashboard,Admin Dashboard and Loan Predicting Window connects to the Approval Dashboard.



5.3 User Stories

It handles tasks such as logging into the IBM account in Sprint 1. Download the dataset and visualise it. It performs activities such as pre-processing the dataset in sprint 2. Model the algorithm Decision Tree modelbuilding, Knn modelbuilding, Random Forest modelbuilding,Xgboost modelbuilding, and then assess the models. In Sprint 3, it completes tasks such as integrating the model with Flask and Finally it deploy our project on IBM Cloud.

- To design a dashboard similar to the User Interface, As a user, you may fill out theapplication and access it through the user interface.
- You can also fill out the application and check for available sources.
- It conducts tasks such as registering all team members to IBM Cloud insprint 4.
- On the IBM Cloud, train the model.
- Install the website on IBM Cloud.
- The user applies for the loan (the loan can be checked by the user).



Creating a user journey is a quick way to help you and your team gain a deeper understanding of who you're designing for, aka the stakeholder in your project. The information you add here should be representative of the observations and research you've done about your users.

Phases	Awareness	Consideration	Service	Loyalty
Steps	View online ad, see social media campaign, hear about from friends about benefits	Conduct feedback session for customers, compare features and benefits of loan	Loan eligibility check and comparing with other documentation	Make additional benefits and approval of higher amount
Feelings	In case the user does not use the social media it must be hard.	If the income less the expected value then loan prediction will detect only for low amount	You are offline the application does not show the any information.	Sometimes to difficult the predict the amount based on income.
Pain points	- Is not aware of all loan benefits - Doesn't know what to choose	- Doesn't know where to start -Doesn't want to spend a lot of time on research	- Hard to get response - Buffering issues	- No discount - Not enough other incentives
Opportunities	Increase awareness interest, marketing on loan ,communications- awarness	Customer weights multiple offerings that could solve the problem	Respond to customer inquiries and concerns in a timely manner to improve experiences	Reward long-term loyalty to keep your customer wanting more

CHAPTER-6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning and Estimation

Sprint Planning

A sprint is essentially a predetermined length of time in which a development team needs to perform a specified amount of work. Sprints are often scheduled to last two weeks, although they can last as little as one week or as long as a month.The limited time span of a sprint forces developers to focus on sending out tiny, incremental improvements rather than massive, sweeping ones. As a result, significantly less debugging is necessary, and clients may have a more smooth experience with the programme.Generally it is used to create product backlog and contains sprint 1,2,3,4.Each performs some specific tasks to do so.

Sprint-1

In Sprint 1 which involve to create the functional requirement of User Registration and Login andDataset. It performs the task such as To login the IBM account,Download the dataset and visualize the dataset.

Sprint-2

In sprint 2 ,which involves to create the functional requirent of use model. It performs the tasks such as Pre-process the dataset,Model the algorithm Decision Tree model building,Knn model Random Forest model and Xgboost model and then evaluate the models.

Sprint-3

In Sprint 3,which involve to create the functional requirement of Dashboard (User Interface).It perform the task such as To integrate the model with flask,To create a dashboard as like User

Interface,As a user able to fill the application and accessthe application on the user interface,To fill the application and check for theavailability sources.

Sprint- 4

In Sprint 4 ,which involve to create the functional requirement of Deployed the websitein IBM Cloud. It performs the task such as Register all the team membersto IBM Cloud,Train the model on IBM Cloud,Deploy the website on IBM Cloud,User apply for the loan (user can check the loaneligibility or not).

Sprint Estimation

Sprint Estimation is part of the Sprint Turnover process, which happens at the end of the last sprint but before the next sprint starts. It make sure to check our JIRA for issues that were thrown out of the previous sprint or issues that emerged during the sprint time. To ensure thatthis process runs well or not.

Velocity:

Calculate the team’s average velocity (AV) periteration unit (story points per day) .AV= Sprint Duration/Velocity=20/10=2

Sprint-1 = 20/9=2.2
Sprint-2 =
20/6=3.33 Sprint-3
=20/6 = 3..33Sprint-
3 = 19/6 = 3.16

Total Velocity= 79/2 7= 2.92

6.2 Sprint Delivery Schedule

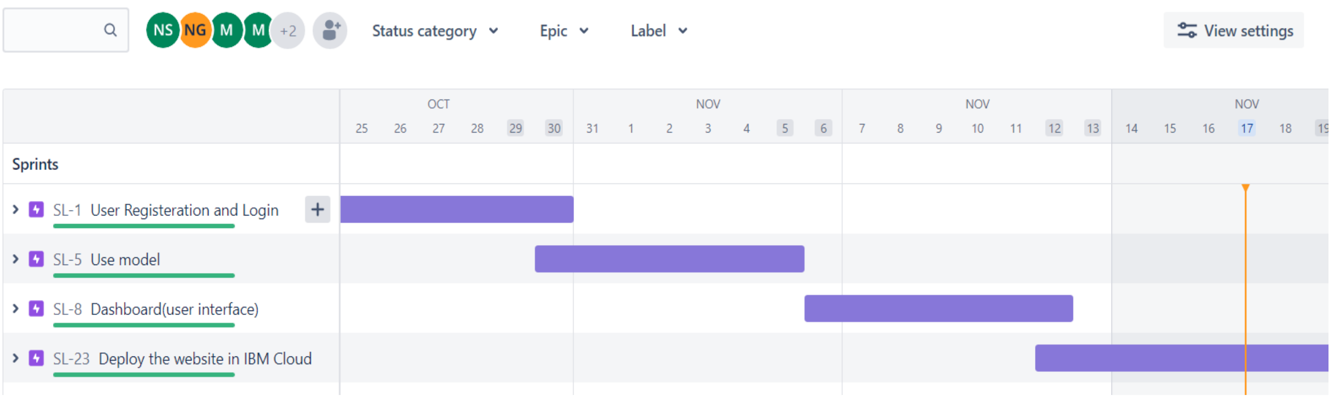
In Sprint 1 ,which involve to create the functional requirement of User Registration and Login and Dataset . It performs the task such as To login the IBM account,Download the dataset and

visualize the datase.Total duration required to complete sprint 1 was 9 days.

In sprint 2 ,which involves to create the functional requirent of use mode.It performs the tasks such as Pre-process the dataset,Model the algorithm Decision Tree modelbuilding,building of Knn model,Random Forest model,Decision Tree model,Xgboost model.and then evaluate the models.Total duration required to complete sprint 2 was 6 days.

In Sprint 3 ,which involve to create the functional requirement of Dashboard (User Interface).It performs the task such as To integrate the model with flask,To create a dashboard as like User Interface,As a user able to fill the application and access the application on the user interface,To fill the application and check for the availability sourcesToatal duration required to complete sprint 3 was 6 days.

In Sprint 4 ,which involve to create the functional requirement of Register,Deployed the website inIBM Cloud.It performs the task such as Register all the team members to IBM Cloud,Train the model on IBM Cloud,Deploy the website on IBM Cloud,User apply for the loan (user can check



the loan eligibility or not).Total duration required to complete sprint 4 was 6 days

CHAPTER-7

CODING & SOLUTIONING

FEATURE:

Feature

Engineering

data.info()

data.isnull().sum()

data['Gender'].fillna(data['Gender'].mode()[0],inplace=True)

data['Married'].fillna(data['Married'].mode()[0],inplace=True)

data['Dependents'].fillna(data['Dependents'].mode()[0],inplace=True)

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

data['LoanAmount'].fillna(data['LoanAmount'].mode()[0],inplace=True)

data['Loan_Amount_Term'].fillna(data['Loan_Amount_Term'].mode()[0],inplace=True)

data['Credit_History'].fillna(data['Credit_History'].mode()[0], inplace=True)

data.info()
```

Missing values in the column "Loan monthly payment" indicate that consumers did not make loan payments. In this case, instead of the mean or median, the missing values should be imputed with zero. The original data has a category target variable. It is divided into four categories, numbered A through D. To make the prediction, I must encode the category variable as 1 or 0, representing binary classes.By using the alforithm in machine learning is able to predict the loan approval.

Random Forest Algorithm

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

Comparison of Randim Forest Algorithm Vs KNN Vs Decision Tree:

```
RandomForest (X_train,X_test,y_train,y_test)
decisionTree(X_train,X_test,y_train,y_test)
KNN(X_train,X_test,y_train,y_test)
XGB (X_train,X_test,y_train,y_test)
```

CHAPTER-8

TESTING

8.1Test Cases

For checking the loan application, We have two testcase

- Eligible
- Not Eligible

This is based on the training and testing the model we used in our application.

This eligibility can be checked by using the details entered by the users. This includes the

details like

- Gender
- Status
- Dependants
- Education
- Employ
- Income
- Co-income(additional income)
- Loan amount
- Loan amount term(in days)
- Credit history
- Property area(type of location)

User Acceptance Testing

Purpose Of Document: The purpose of this document is to briefly explain the test coverage and open issues of the [Smart Lender - Applicant Credibility Prediction for Loan Approval] project at the time of the release to User AcceptanceTesting(UAT).

Defect AnalysiS: This report shows the number of resolved or closed bugs ateach severity level,and how they were resolved.

UAT Report Submission and usage of tools

	Total StoryPoints	**Duration**	**Sprint Start Date**	**Sprint End Date(Planned)**	**Story Point Completed(as an planned end date)**	**Sprint Release Date(Actual)**
Sprint-1	10	9 Days	21 Oct2022	30 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	11	6 Days	06 Nov2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	19	6 Days	12 Nov 2022	19 Nov 2022	19	19 Nov 2022

CHAPTER-9
RESULTS

9.1 Performance Metrics

FIS Financial View, for example, compiles useful indicators and KPIs and then helps organize and explain them so you can react to trends, uncover performance possibilities, and monitor financial health. In bank laon prediction ,the upside of the framework is that we present the prerequisites as a calculation, and while confirming the subtleties, we decide the necessities that have beenendorsed and that meet the rerequisites of the unlawful client.

Decision Tree

Decision trees may be used to forecast numerical values (regression) as well as categorise data. The decision tree which hold ,

Performance metrices of decision tree:

Confusion
Matrix[[49 14]

[20 58]]
Classification Report

precision recall f1-score
support0 0.71 0.78 0.74
63

1 0.81 0.74 0.77 78
accuracy 0.76 141

macro avg 0.76 0.76 0.76 141

weighted avg 0.76 0.76 0.76 141

Score:0.7588652482269503

Random Forest

In a random forest, the machine learning algorithm predicts a value or category by combining the results from a number of decision trees.The random forest algorithm is a bagging technique extension that use both bagging and feature randomization to produce an uncorrelated forest of decision trees.

Preformance matrices of Random forest algorithm:

Confusion

Matrix[[44 18]

[6 72]]

Classification Report

precision recall f1-score

support0 0.88 0.71 0.79

62

1 0.80 0.92 0.86 78

accuracy 0.83 140

macro avg 0.84 0.82 0.82 140

weighted avg 0.84 0.83 0.83 140

Score

0.82857142857142

86

K-Nearest Neighbors algorithm

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

Performance matrices of KNN algorithm:

Confusion

Matrix[[40 27]

[26 50]]

Classification Report

	precision	recall	f1-score	support
0	0.61	0.60	0.60	67
1	0.65	0.66	0.65	76
accuracy			0.63	143
macro avg	0.63	0.63	0.63	143
weighted avg	0.63	0.63	0.63	143

Score

0.6293706293706294

XGboost

XGBoost, or Extreme Gradient Boost, is a machine learning technique used to create gradient boosting decision trees. When it comes to unstructured data, such as photos and unstructured text data, ANN models (Artificial neural network) appear to be at the top of the list when it comes to prediction.

Performance matrices of Xgboost algorithm:

Confusion

Matrix[[53 16]

[25 44]]

Classification Report

	precision	recall	f1-score	support
0	0.68	0.77	0.72	69
1	0.73	0.64	0.68	69
accuracy			0.70	138
macro avg	0.71	0.70	0.70	138
weighted avg	0.71	0.70	0.70	138

score

0.7028985507246377

Evaluating Performance Of The Models:

When compared alla the other algorithms Random Forest Algorithm has the high est accuracy of 0.8285714285714286. By using this algorithm ,we obtain the prediction for the loan approval or rejection.

F1-Score:

0.78334173271636
04

Mean:

0.82281815296731

21

CHAPTER-10

ADVANTAGES & DISADVANTAGES

Advantages

Various sources to generate a generalised dataset and apply four machine learning algorithms to the dataset, including Random forest, Logistic regression, and Decision tree.

- The advantage of the framework is that we show the requirements as a calculation, and while checking the subtleties, we determine the demands that have been approved and fulfil the requirements of the illicit customer.
- The framework is rated higher than high even out information. The shown structure is similar to a good memory.
- The risk of spreading to the necessary framework is minimal.
- Slight changes in information have little effect on the hyper plant.
- Performance and accuracy of the algorithms can be calculated and compared.
- Class imbalance can be dealt with machine learning approaches

Disadvantages

- They provided a mathematical model and did not employ machine learning methods.
- The problem of class imbalance was not addressed, and appropriate measures were notadopted.

- Existing frameworks typically fail. Computations are undeniably difficult because manyof the outcomes are linked.

CHAPTER-11

CONCLUSION

The analysis begins with data cleansing and missing value processing, followed by exploratory analysis, model creation, and model evaluation. When we receive a better accuracy score and other performance indicators on the public test set, we will have the best accuracy. This papercan assist in predicting whether or not an applicant will be approved for a bank loan. When a consumer suffers a calamity, for example, the calculation cannot predict the outcome. This assessment paper can be used to determine whether a customer is capable.

CHAPTER-12

FUTURE SCOPE

Future enhancement of this research work on training bots to predict the loan eligibility areas by using machine learning techniques. Since, machine learning is similar to data mining advanced concept of machine learning can be used for better prediction. The data privacy, reliability, accuracy can be improved for enhanced prediction From the encouraging results, we believe that crime data mining has a promising future for increasing the effectiveness and efficiency of criminal and intelligence analysis. Visual and intuitive criminal and intelligence investigation techniques can be developed for loan credibility pattern. As we have applied machine learning technique of data mining for loan prediction we can also perform other techniques of data mining such as classification. Also we can perform analysis on various dataset such as enterprise survey dataset, poverty dataset, aid effectiveness dataset, etc.

CHAPTER-13

APPENDIX

Source Code

Home.html

```
<html>
<head>
<style>
    body{
        padding:50px 40px 50px 40px;
      background-color:   #00ffff;
      background-position: center;
      background-size: cover;
    }
    .predict{
        background-color: #04AA6D;
    color: white;
    padding: 3px 30px;
    margin: 8px ;
    border: none;
    cursor: pointer;
    width: 20%;
    opacity: 0.9;
      border-radius:10px;
}
h1{
    font-size: 50px;
}
</style>
<body>
<h1>Welcome to Loan Prediction</h1>
<h2><a href="/prediction" class="button">predict</a></h2>

</body>
</head>
</html>
```

prediction.html

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
 body {
  padding:50px 300px 50px 300px;
  font-family: Arial, Helvetica, sans-serif;
  background-image: url(bg.png);
  background-position: fixed;
   background-size: cover;
 }


 * {
  box-sizing: border-box;
 }

 /* Add padding to containers */
 .container {
  padding:50px 40px 50px 40px;
   background-color:   #00ffff;
   background-position: center;
   background-size: cover;
   border-radius: 10px;
 }

 /* Full-width input fields */
 input[type=text], input[type=integer] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
 }

 input[type=text]:focus, input[type=integer]:focus {
  background-color: #ddd;
  outline: none;
 }
   .h1{
     align-content: center;

   }

 /* Overwrite default styles of hr */
 hr {
```

```
  border: 1px solid #f1f1f1;
  margin-bottom: 25px;
}

/* Set a style for the submit button */
.registerbtn {
  background-color: #04AA6D;
  color: white;
  padding: 16px 20px;
  margin: 8px 0;
  border: none;
  cursor: pointer;
  width: 40%;
  opacity: 0.9;
    border-radius:10px;
}

.registerbtn:hover {
  opacity: 1;
}

/* Add a blue text color to links */
a {
  color: dodgerblue;
}

/* Set a grey background color and center the text of the "sign in" section */
.signin {
  background-color: #f1f1f1;
  text-align: center;
}
custom-select {
  position: relative;
  font-family: Arial;
}

.custom-select select {
  display: none; /*hide original SELECT element:*/
}

.select-selected {
  background-color: DodgerBlue;
}

/*style the arrow inside the select element:*/
.select-selected:after {
  position: absolute;
  content: "";
  top: 14px;
```

```
    right: 10px;
    width: 0;
    height: 0;
    border: 6px solid transparent;
    border-color: #fff transparent transparent transparent;
  }

  /*point the arrow upwards when the select box is open (active):*/
  .select-selected.select-arrow-active:after
 {
    border-color: transparent transparent #fff transparent;
    top: 7px;
  }

  /*style the items (options), including the selected item:*/
  .select-items div,.select-selected
{
    color: #ffffff;
    padding: 8px 16px;
    border: 1px solid transparent;
    border-color: transparent transparent rgba(0, 0, 0, 0.1) transparent;
    cursor: pointer;
    user-select: none;
  }

  /*style items (options):*/
  .select-items
 {
    position: absolute;
    background-color: DodgerBlue;
    top: 100%;
    left: 0;
    right: 0;
    z-index: 99;
  }

  /*hide the items when the select box is closed:*/
  .select-hide
{
    display: none;
  }

  .select-items div:hover, .same-as-selected
 {
    background-color: rgba(0, 0, 0, 0.1);
  }
  #Gender,#Education,#Credithistory,#Propertyarea,#LoanAmount,#LoanAmountTerm,#Maritalsta
tus,#co-applicantincome,#Dependents,#employed,#applicantincome{
    border-radius: 6px;
```

```
  }
</style>
</head>
<body>




  <div class="container">
    <h1>LOAN APPROVAL APPLICATION</h1>
    <p>Fill in the details to get your chances of Loan.</p>

    <hr>
 <form method="post" action="{{url_for('result')}}">
    <label for="email"><b>Gender</b></label></P>
    <select id="Gender" name="Gender">
    <option value="0">Select gender:</option>
    <option value="1">Male</option>
    <option value="2">Female</option>

  </select>

</p>
<label for="email"><b>Education</b></label></P>
    <select id="Education" name="Education">
    <option value="0">Select :</option>
    <option value="1">GRADUATED</option>
    <option value="2">NOT GRADUATED</option>

  </select>
  </p>


    <label for="psw"><b>Marital status</b></label></P>
      <select id="Maritalstatus" name="Marital status">
    <option value="0">Yes/No</option>
    <option value="1">Yes</option>
    <option value="2">No</option>

  </select>
</P>

   <label for="psw"><b>Dependents</b></label></P>
      <select id="Dependents" name="Dependents">
    <option value="0">No.of.Dependents</option>
    <option value="1">0</option>
    <option value="2">1</option>
    <option value="3">2</option><option value="">2+</option>

  </select>
```

```
</P>
   <label for="psw"><b>Self employed</b></label></P>
      <select id="employed" name="employed">
    <option value="0">Yes/No:</option>
    <option value="1">Yes</option>
    <option value="2">No</option>

  </select>
</P>
   <label for="psw"><b>Applicant Income</b></label></P>
   <input type="number" id="applicantincome" placeholder="Amount" name="applicant income"
min="10,000" max="10000000">
   </P>
   <label for="psw"><b>Co-applicant Income</b></label></P>
  <input type="number" id="co-applicantincome" placeholder="Amount" name="co-applicant
income" min="10,000" max="10000000">
  </P>
    <label for="psw"><b>Loan Amount </b></label></P>
 <input type="number" id="LoanAmount" placeholder="Amount" name="Loan Amount"
min="100000" max="100000000"></P>

  <label for="psw"><b>Loan Amount Term</b></label></P>
      <input type="number" id="LoanAmountTerm" placeholder="Amount" name="Loan Amount
Term" min="1" max="365"></P>

  <label for="psw"><b> Credit History</b></label></P>
      <select id="Credithistory" name="CreditHistory">
    <option value="0">Select :</option>
    <option value="1">1</option>
    <option value="2">0</option>

  </select></P>


  <label for="psw"><b>Property Area</b></label></P>
      <select id="Propertyarea" name="Propertyarea">
    <option value="0">Area:</option>
    <option value="1">rural</option>
    <option value="2">urban</option>
    <option value="3">semi-urban</option>

  </select></p>




  <hr>
    <p>Check the details before submit.</p>
```

```
  <button type="submit">submit</button>
</form>
  </div>



</body>
</html>
```

Result.html

```
<html>
<head>
    <style>
        body{
        padding:50px 40px 50px 40px;
      background-color:   #00ffff;
      background-position: center;
      background-size: cover;
    }
    </style>
<body>
 <h1></h1>
 <h2>{{result}}<h2>
</body>
</head>
</html>
```

app.py

```
from flask import Flask, render_template, request
import numpy as np
import pickle
import pandas
import os
app=Flask (__name__)
model = pickle.load(open ('rdf1.pkl', 'rb'))
scale = pickle.load(open('scale1.pkl','rb'))
@app.route('/') # rendering the html template
def home():
    return render_template('home.html')
@app.route('/prediction',methods=["POST","GET"])
def prediction():
```

```
    return render_template('prediction.html')

@app.route('/result', methods = [ "POST","GET"])# route to show the predictions in a web UI def
def result():
   input_feature=[int(x) for x in request.form.values() ]
    #input_feature = np.transpose(input_feature)
   input_feature=[np.array(input_feature)]
   print(input_feature)
   names = ['Gender', 'Married', 'Dependents', 'Education', 'Self Employed', 'Applicant Income',
'Coapplicant Income', 'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area']
   data = pandas.DataFrame(input_feature, columns=names)
   print(data)



   #data_scaled = scale.fit_transform(data) #data = pandas.DataFrame(, columns=names)
   # predictions using the loaded model file prediction=model.predict(data)
   prediction=model.predict(data)
   print (prediction)
   prediction = int(prediction)
   print(type(prediction))
   if (prediction == 0):
      return render_template("result.html", result = "Loan wiil Not be Approved")
   else:
    #showing the prediction results in a UI
      return render_template("result.html", result = "Loan will be Approved")



  if __name__=="__main__":
# app.run(host='0.0.0.0', port=8000, debug=True)
  port=int(os.environ.get('PORT',5000))
  app.run(debug=False)
```

Loan_Prediction.ipyn

```
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
```

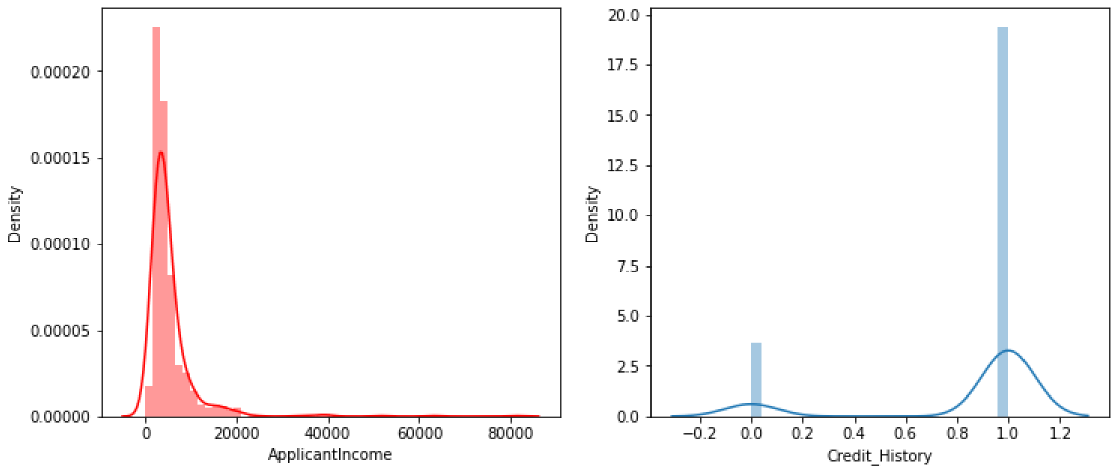
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
data=pd.read_csv("/content/drive/MyDrive/train_u6lujuX_CVtuZ9i.csv")
```

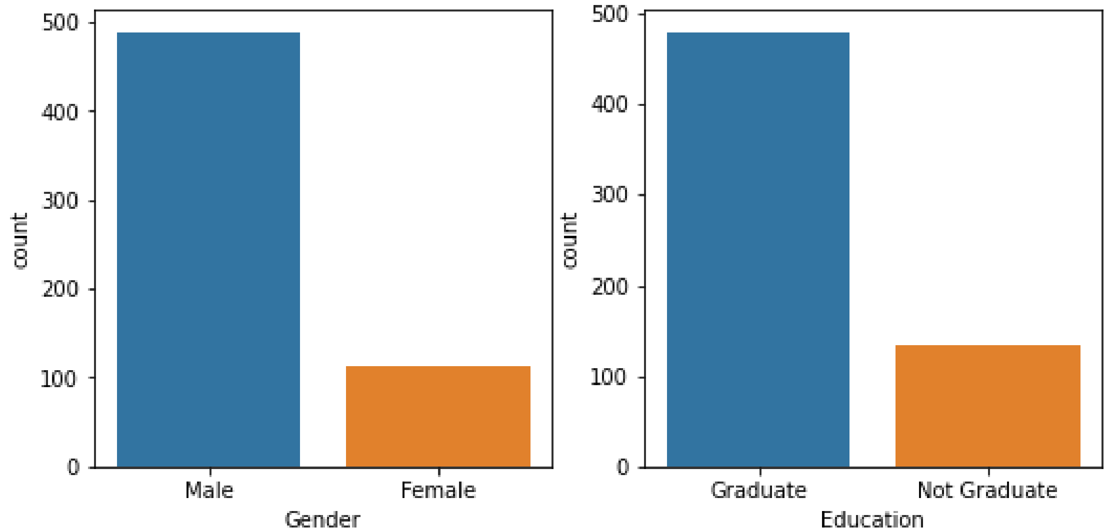
```
#plotting the using distplot
plt.figure(figsize=(12,5))
plt.subplot(121)
sns.distplot(data['ApplicantIncome'], color='r')
plt.subplot(122)
sns.distplot(data['Credit_History'])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an
axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an
axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```




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

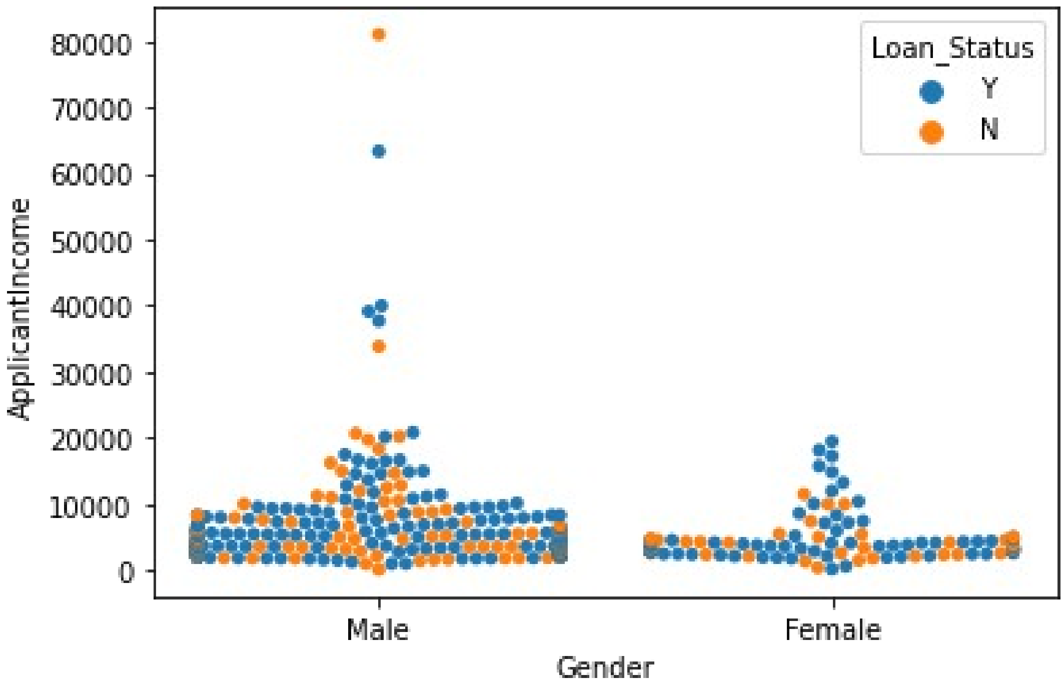
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```




```
#visulaized based gender and income what would be the appplication
status
sns.swarmplot(data['Gender'], data['ApplicantIncome'], hue =
data['Loan_Status'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296:
UserWarning: 67.5% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296:
UserWarning: 33.0% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)

<matplotlib.axes._subplots.AxesSubplot at 0x7f927c385810>
```




```
data.describe()

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

       Credit_History
count      564.000000
mean         0.842199
std          0.364878
min          0.000000
25%          1.000000
50%          1.000000
```

```
75%          1.000000
max          1.000000
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    object
 1   Gender             601 non-null    object
 2   Married            611 non-null    object
 3   Dependents         599 non-null    object
 4   Education          614 non-null    object
 5   Self_Employed      582 non-null    object
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
 10  Credit_History     564 non-null    float64
 11  Property_Area      614 non-null    object
 12  Loan_Status        614 non-null    object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

```
from sklearn.compose import make_column_selector as selector

numerical_columns_selector = selector(dtype_exclude=object)
categorical_columns_selector = selector(dtype_include=object)

numerical_columns = numerical_columns_selector(data)
categorical_columns = categorical_columns_selector(data)
```

```
numerical_columns
```

```
['ApplicantIncome',
 'CoapplicantIncome',
 'LoanAmount',
 'Loan_Amount_Term',
 'Credit_History']
```

```
categorical_columns
```

```
['Loan_ID',
 'Gender',
 'Married',
 'Dependents',
 'Education',
 'Self_Employed',
```

```
 'Property_Area',
 'Loan_Status']
```

```
data['Gender'].unique()
```

```
array(['Male', 'Female', nan], dtype=object)
```

```
data['Married'].unique()
```

```
array(['No', 'Yes', nan], dtype=object)
```

```
data['Dependents'].unique()
```

```
array(['0', '1', '2', '3+', nan], dtype=object)
```

```
data['Education'].unique()
```

```
array(['Graduate', 'Not Graduate'], dtype=object)
```

```
data['Self_Employed'].unique()
```

```
array(['No', 'Yes', nan], dtype=object)
```

```
data['Property_Area'].unique()
```

```
array(['Urban', 'Rural', 'Semiurban'], dtype=object)
```

```
data['Loan_Status'].unique()
```

```
array(['Y', 'N'], dtype=object)
```

```
data['Loan_ID']= label_encoder.fit_transform(data['Loan_ID'])
data['Loan_ID'].unique()
```

```
array([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,
12,
        13,  14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,
25,
        26,  27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,
38,
        39,  40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,
51,
        52,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,
64,
        65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,
77,
        78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,
90,
        91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102,
103,
       104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115,
116,
       117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128,
129,
```

454,

130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141,
142,
143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154,
155,
156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167,
168,
169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180,
181,
182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193,
194,
195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206,
207,
208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219,
220,
221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232,
233,
234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245,
246,
247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258,
259,
260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271,
272,
273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284,
285,
286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297,
298,
299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310,
311,
312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323,
324,
325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336,
337,
338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349,
350,
351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362,
363,
364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375,
376,
377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388,
389,
390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401,
402,
403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414,
415,
416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427,
428,
429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440,
441,
442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453,
454,

```
       455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466,
467,
       468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479,
480,
       481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492,
493,
       494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505,
506,
       507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518,
519,
       520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531,
532,
       533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544,
545,
       546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557,
558,
       559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570,
571,
       572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583,
584,
       585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596,
597,
       598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609,
610,
       611, 612, 613])
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Loan_ID            614 non-null    int64  
 1   Gender             614 non-null    int64  
 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(3), object(6)
memory usage: 62.5+ KB
```

```
# Import label encoder
```

```
from sklearn import preprocessing

# label_encoder object knows how to understand word labels.

label_encoder = preprocessing.LabelEncoder()


# Encode labels in column 'Gender'.

data['Gender']= label_encoder.fit_transform(data['Gender'])



data['Gender'].unique()

array([1, 0, 2])

data['Married']= label_encoder.fit_transform(data['Married'])
data['Married'].unique()

array([0, 1, 2])

data['Dependents']= label_encoder.fit_transform(data['Dependents'])
data['Dependents'].unique()

array([0, 1, 2, 3, 4])

data['Education']= label_encoder.fit_transform(data['Education'])
data['Education'].unique()

array([0, 1])

data['Self_Employed']=
label_encoder.fit_transform(data['Self_Employed'])
data['Self_Employed'].unique()

array([0, 1, 2])

data['Property_Area']=
label_encoder.fit_transform(data['Property_Area'])
data['Property_Area'].unique()

array([2, 0, 1])

data['Loan_Status']= label_encoder.fit_transform(data['Loan_Status'])
data['Loan_Status'].unique()

array([1, 0])

data['LoanAmount']=data['LoanAmount'].apply('int64')
data['CoapplicantIncome'] = data['CoapplicantIncome'].astype('int64')
```

```
data['Loan_Amount_Term']=data['Loan_Amount_Term'].apply('int64')
data['Credit_History']=data['Credit_History'].apply('int64')

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

smote = SMOTETomek (0.90)
```

```
/usr/local/lib/python3.7/dist-packages/imblearn/utils/
_validation.py:591: FutureWarning: Pass sampling_strategy=0.9 as
keyword args. From version 0.9 passing these as positional arguments
will result in an error
  FutureWarning,
```

```
x = data.drop(columns=['Loan_Status','Loan_ID'],axis=1)
y = data['Loan_Status']
x_bal,y_bal=smote.fit_resample(x,y)

x
```

```
     Gender  Married  Dependents  Education  Self_Employed
ApplicantIncome  \
0         1        0           0          0              0
5849
1         1        1           1          0              0
4583
2         1        1           0          0              1
3000
3         1        1           0          1              0
2583
4         1        0           0          0              0
6000
..      ...      ...         ...        ...            ...
...
609       0        0           0          0              0
2900
610       1        1           3          0              0
4106
611       1        1           1          0              0
8072
612       1        1           2          0              0
7583
613       0        0           0          0              1
4583

     CoapplicantIncome           LoanAmount  Loan_Amount_Term
Credit_History  \
0                    0 -9223372036854775808               360
1
```

```
1                  1508                      128                  360
1
2                     0                       66                  360
1
3                  2358                      120                  360
1
4                     0                      141                  360
1
..                  ...                      ...                  ...
...
609                   0                       71                  360
1
610                   0                       40                  180
1
611                 240                      253                  360
1
612                   0                      187                  360
1
613                   0                      133                  360
0

     Property_Area
0                2
1                0
2                2
3                2
4                2
..             ...
609              0
610              0
611              2
612              2
613              1

[614 rows x 11 columns]

y

0      1
1      0
2      1
3      1
4      1
      ..
609    1
610    1
611    1
612    1
613    0
Name: Loan_Status, Length: 614, dtype: int64
```

```
print(y.value_counts())

print(y_bal.value_counts())
```

```
1    422
0    192
Name: Loan_Status, dtype: int64
1    350
0    307
Name: Loan_Status, dtype: int64
```

```
sc=StandardScaler()

x_bal=sc.fit_transform(x_bal)

x_bal = pd.DataFrame(x_bal,
columns=['Gender','Married','Education','Dependents','Self_Employed','
ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term','
Credit_History','Property_Area'])



X_train, X_test, y_train, y_test= train_test_split( x_bal, y_bal,
test_size=0.33, random_state=42)

X_train
```

```
       Gender   Married  Education  Dependents  Self_Employed  \
203  0.483458  0.843505   0.236638    2.063008      -0.407244
196 -1.753385 -1.142813  -0.711356   -0.484729       1.574677
286 -1.753385 -1.142813  -0.711356   -0.484729      -0.407244
93   0.483458  0.843505  -0.711356    2.063008      -0.407244
586  0.483458 -1.142813  -0.711356   -0.484729      -0.407244
..        ...       ...        ...         ...            ...
71   0.483458  0.843505   1.184631    2.063008      -0.407244
106  2.720300  0.843505   2.132625   -0.484729      -0.407244
270  0.483458  0.843505   0.236638   -0.484729       1.574677
435  0.483458 -1.142813  -0.711356   -0.484729      -0.407244
102 -1.753385 -1.142813  -0.711356   -0.484729      -0.407244

     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term
\
203         0.197238          -0.537302    0.210986          0.087571

196        -0.332645          -0.537302    0.210986          0.087571

286        -0.515450          -0.537302    0.210986          0.087571

93         -0.314280           0.922234    0.210986        -11.419282

586        -0.312258          -0.537302    0.210986          0.087571
```

```
..              ...                ...         ...                ...
71        -0.354548           0.172244    0.210986           0.087571
106        3.094321          -0.537302    0.210986           0.087571
270        0.543978           1.230532    0.210986           0.087571
435       -0.127600           0.203109   -4.739650           0.087571
102       -0.556055           2.648559    0.210986           0.087571

     Credit_History  Property_Area
203        0.305251       1.300985
196       -3.275995       1.300985
286        0.305251      -1.265823
93         0.305251      -1.265823
586        0.305251       0.017581
..              ...            ...
71        -3.275995       0.017581
106        0.305251      -1.265823
270        0.305251      -1.265823
435        0.305251       0.017581
102        0.305251       0.017581

[440 rows x 11 columns]
```

```
y_train
```

```
203    1
196    1
286    0
93     0
586    0
      ..
71     1
106    1
270    0
435    0
102    1
Name: Loan_Status, Length: 440, dtype: int64
```

```
def decisionTree(x_train, x_test, y_train, y_test):
   dt=DecisionTreeClassifier()
   dt.fit(x_train,y_train)
   yPred = dt.predict(x_test)
   print("***DecisionTreeClassifier***")
   print('Confusion matrix')
```

```python
   print(confusion_matrix(y_test,yPred))
   print('Classification report')
   print(classification_report (y_test, yPred))

def randomForest (x_train, x_test, y_train, y_test):
    rf = RandomForestClassifier()
    rf.fit(x_train,y_train)
    yPred = rf.predict(x_test)
    print ("***RandomForestClassifier***")
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print('Classification report')
    print(classification_report (y_test,yPred))

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

def xgboost(x_train, x_test, y_train, y_test):
  xg = GradientBoostingClassifier()
  xg.fit(x_train,y_train)
  yPred = xg.predict(x_test)
  print("***Gradient BoostingClassifier***")
  print('Confusion matrix')
  print(confusion_matrix(y_test,yPred))
  print('Classification report')
  print(classification_report (y_test, yPred))

randomForest(X_train,X_test,y_train,y_test)
```

```
***RandomForestClassifier***
Confusion matrix
[[65 41]
 [27 84]]
Classification report
              precision    recall  f1-score   support

           0       0.71      0.61      0.66       106
           1       0.67      0.76      0.71       111

    accuracy                           0.69       217
   macro avg       0.69      0.68      0.68       217
weighted avg       0.69      0.69      0.68       217
```

```
decisionTree(X_train,X_test,y_train,y_test)

***DecisionTreeClassifier***
Confusion matrix
[[66 40]
 [31 80]]
Classification report
              precision    recall  f1-score   support

           0       0.68      0.62      0.65       106
           1       0.67      0.72      0.69       111

    accuracy                           0.67       217
   macro avg       0.67      0.67      0.67       217
weighted avg       0.67      0.67      0.67       217


KNN(X_train,X_test,y_train,y_test)

***KNeighborsClassifier***
Confusion matrix
[[54 52]
 [46 65]]
              precision    recall  f1-score   support

           0       0.54      0.51      0.52       106
           1       0.56      0.59      0.57       111

    accuracy                           0.55       217
   macro avg       0.55      0.55      0.55       217
weighted avg       0.55      0.55      0.55       217

Classification report
              precision    recall  f1-score   support

           0       0.54      0.51      0.52       106
           1       0.56      0.59      0.57       111

    accuracy                           0.55       217
   macro avg       0.55      0.55      0.55       217
weighted avg       0.55      0.55      0.55       217


xgboost(X_train,X_test,y_train,y_test)

***Gradient BoostingClassifier***
Confusion matrix
[[61 45]
 [32 79]]
Classification report
```

```
              precision    recall  f1-score   support

           0       0.66      0.58      0.61       106
           1       0.64      0.71      0.67       111

    accuracy                           0.65       217
   macro avg       0.65      0.64      0.64       217
weighted avg       0.65      0.65      0.64       217
```

```
from sklearn.model_selection import cross_val_score
# Random forest model is selected
rf = RandomForestClassifier()
rf.fit(X_train,y_train)
yPred=rf.predict(X_test)
f1_score(yPred,y_test, average='weighted')
```

```
0.679255651098188
```

```
cv = cross_val_score(rf,x,y,cv=5)
np.mean(cv)
```

```
0.7833933093429295
```

```
pickle.dump(rf, open('rdf1.pkl','wb'))
pickle.dump(sc, open('scale.pkl','wb'))
```

GITHUB: https://github.com/IBM-EPBL/IBM-Project-15375-1659597829

DEMO LINK: https://www.youtube.com/watch?v=XUixoVVP4tA