PROJECT REPORT

Smart Lender - Applicant Credibility Prediction for Loan Approval

1. INTRODUCTION

1.1 PROJECT 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.

1.2 PURPOSE

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.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

[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%. [2]The two algorithms used were two class decision jungle and two class decision and their accuracy were 77.00% and 81.00% respectively. [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%. [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. [5]The process of Min-Max Normalization is used.The highest accuracy they got was 75.08% when the percentage of dataset split was 50-50% with k to be set as 30. In [6] LogisticRegression is the only algorithm used. They didn't calculate the accuracy of the algorithm. [7]checking manually individual consumer's credibility for the loan approval is difficult ,time consuming and risky. Thus, logistic regression is used as the tool to predict whether an applicant is eligible for the loan or not. [8]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 [9] Information mining calculations are utilized to concentrate on the advance supported information and precise examples, which would help in anticipating the sensible defaulters, subsequently helping the banks for pursuing better decisions later on. Information Mining is the interaction. [10] By provoking non-moneylenders, banks can decrease non-performing assets. This makes learning these things indispensable. Energy research shows that there are various approaches to focusing on reimbursement[11] . While examining the information catch and investigation for fair loaning, they represented a few starting key advances at present required for further developing data quality to all gatherings included.

2.2 REFERENCES

1. Ashwini S. Kadam, Shraddha R Nikam, Ankita A. Aher, Gayatri V. Shelke, Amar S. Chandgude, 2021, “Prediction for Loan Approval using Machine Learning Algorithm”, No “Apr” / ”2021”.

2. Sivasree M S, Rekha Sunny T, (2015), “Loan Credibility Prediction System Based on Decision Tree Algorithm”, No “September” / “2015”.

3. AnujaKadam, PragatiNamde, SonalShirke, SiddheshNandgaonkar, Dr.D.R Ingle, 2021, “Loan Credibility Prediction System using Data Mining Techniques” No “May” / “2021”.

4. PidikitiSupriya , MyneediPavani , NagarapuSaisushma , NamburiVimala Kumari , K Vikas, 2019, “Loan Prediction by using Machine Learning Models”, No “April” / “2019”.

5.https://medium.com/swlh/lending-club-data-web-app-ada56ff64cee

6.https://github.com/smartinternz02/SI-GuidedProject-48927-1652694502

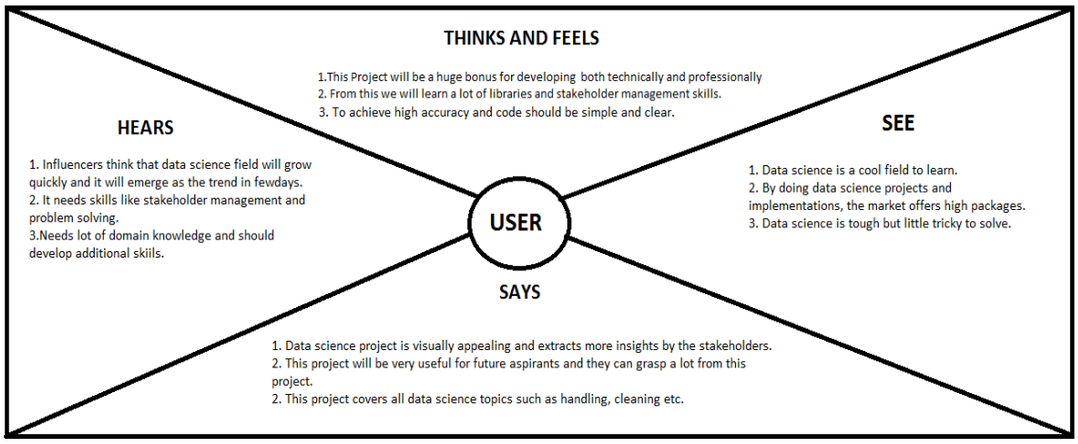
7.https://www.academia.edu/77162007/BANK_LOAN_PREDICTION_USING_MACHINE_LEARNING

2.3 PROBLEM STATEMENT DEFINITION

Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others.To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. It is a classification problem where we have to predict whether a loan would be approved or not.

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION AND BRAINSTORMING

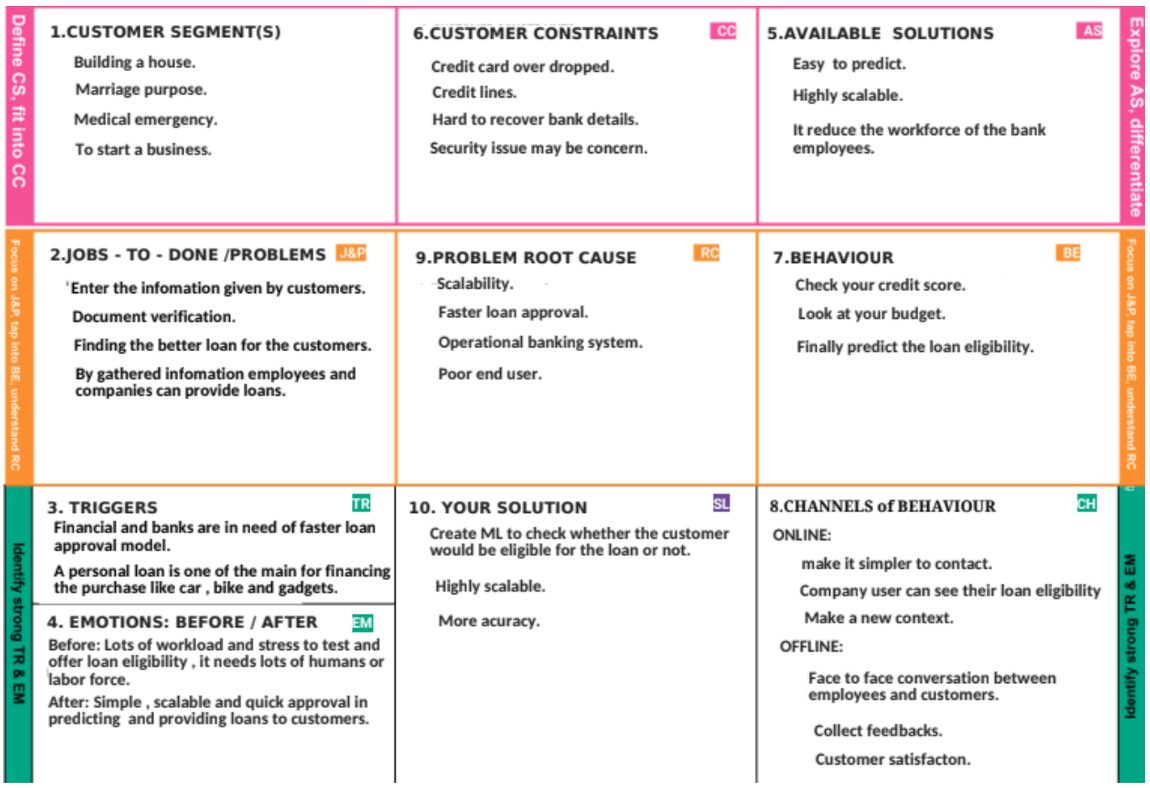
3.3 PROPOSED SOLUTION

These solution template relates the current situation to a desired result of this project and also describe the benefits acquire when desired result is achieved

S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	Inclined to human blunders. Time utilization is high, Parcel of paper works. Poor client care because of absence of labor. Tracking or checking the status is difficult.
2	Idea / Solution description	Following or checking the status turns out to be simple. Decrease the potential for human mistake. Time utilization of the interaction will be decreased. Diminishes the desk work to paperless. Work on the adequacy of client care groups. Fair qualification expectation. Profoundly adaptable and give information driven choices to partner and more significant position. We will prepare and test the information with these calculations, tune by hyper parameter tunning .From this the above thoughts are executed. A constant source of profits is the evidence for any lender that you are capable of repaying your non-public loan.
3	Novelty / Uniqueness	When the fundamental information are given, the model will anticipate whether to approve the loan or not - By utilization of transfer learning. An computerized customer support system will help the user and guide them through the loan approval process.
4	Social Impact / Customer Satisfaction	Quite possibly of the main component which influence our nation's economy and financial condition is the credit framework represented by the banks. As we probably are aware credit risk assessment is extremely essential, there is an assortment of techniques are utilized for risk level estimation. In expansion, credit risk is one of the fundamental elements of the banking community. In the absence of a job, make sure which you have other sources of income.

5	Business Model (Revenue Model)	This model can be created by least expense at a similar time it will give the max execution, higher exactness and the outcome will be more effective than traditional techniques. Can monetize capabilities like viewing multiple banks or applying for more than one banks or we can also have subscriptions once we hit a sure person fee.
6	Scalability of the Solution	Banks need not to go through the background verification process of the candidate by utilizing this model. The model will anticipate the clients information also, their characteristics like salary , credit score, etc.

3.4 PROBLEM SOLUTION FIT



4. REQUIREMENT ANALYSIS

4.1 Functional requirements

FR NO.	FUNTIONAL REQUIREMENT	SUB REQUIREMENT
FR-1	Registration of the User	Registration through bank website, Gmail, mobile application
FR-2	Confirmation of the User	Confirmation via OTP and via mobile notification
FR-3	Type of Loan	Personal Loan Education
FR-4	Details of the User	Name, Age, Address, Mobile Number, Income, Occupation
FR-5	Assets Proof	Gold, Agricultural Land

FR-6	Verification of the User Details	Verification of all the details provided by the user

4.2 Non-Functional requirements

FR NO.	NON-FUNCTIONAL REQUIREMENT	DESCRIPTION
NFR-1	Usability	Accessing is easy
NFR-2	Security	User proofs
NFR-3	Reliability	Customer income Basis
NFR-4	Performance	History of user's bank account
NFR-5	Availability	Based on user's address
NFR-6	Scalability	Based on user's assets proof

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

Data Flow Diagrams:

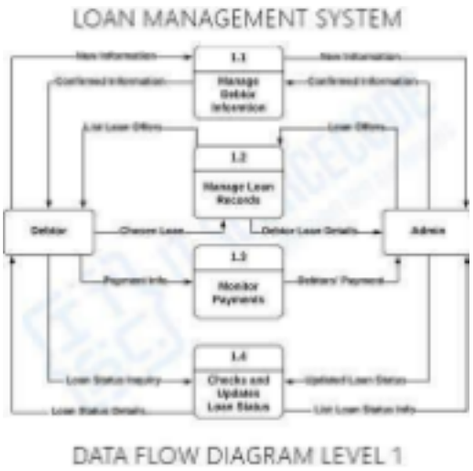
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is

stored.

Flow: (Simplified)

- The user can register in website by using Email and Mobile number.
- **The user can Login by using Email and password as Registered in the respective website.**
- **The user will provide personal and financial details.**
- **User should upload the scanned documents.**
- **Then it will goes to approval process.**
- **Finally they will get loan closure certificate**.

DFD Level 0 (Industry Standard)

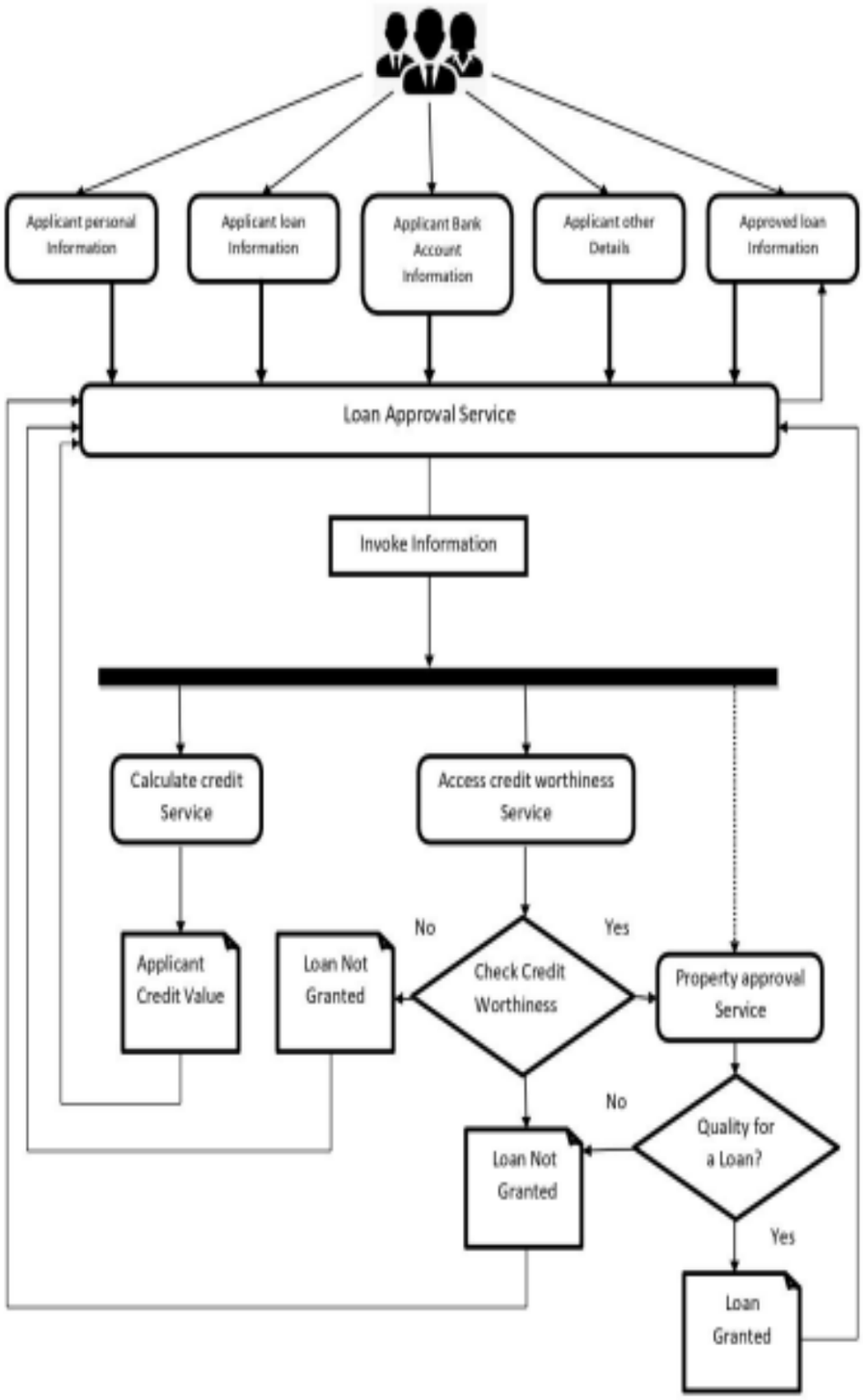


5.2 SOLUTION AND TECHNICAL ARCHITECTURE

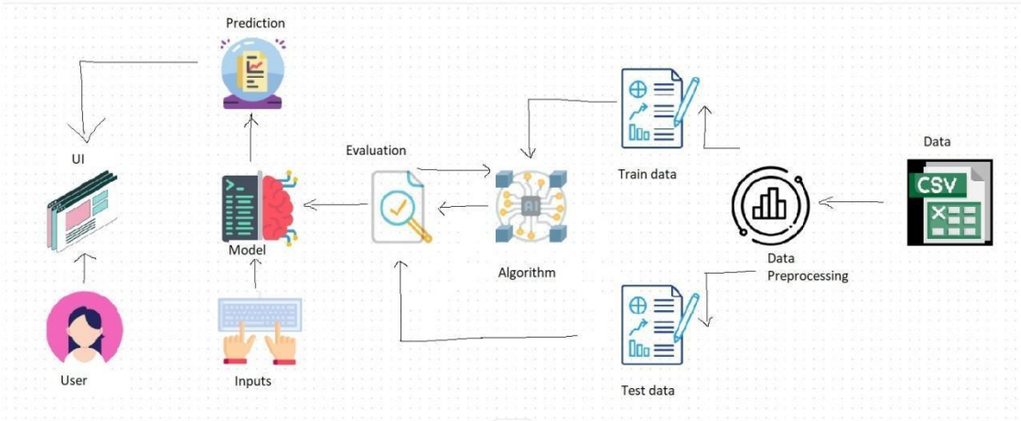
Solution Architecture:

- The essential objective in the financial business is to put their assets in safe hands. Thus, the framework needs to check the archives actually and ought to guarantee that as it were able individuals get the advance.
- The model ought to be prepared to create results with palatable precision, later which it produces precise outcomes regarding whether a borrower ought to be loaned cash or not with next to no drawn-out manual work.
- The clients can obtain the outcomes in the solace of their home.
- The framework ought to diminish chance to both the bank and the client.
- The model can expect results and is rapidly versatile to an extensive variety of inputs. Likewise, this system saves the financial business and its staff a huge measure of time.

Solution Architecture diagram:



Technical Architecture:



5.3 User Stories

Sprint	User Story Number	User Story / Task
Sprint-1	USN-1	As a user, I can visit the loan approval page.
Sprint-1	USN-2	As a user, I have to enter all my personal details and enter loan details.
Sprint-1	USN-3	As a user, I have to ensure whether all the details are correct
Sprint-1	USN-4	As a user, I will know if the analysis is Correct my loan will be approved.
Sprint-2	USN-1	As a user, I need a better UI and ux designs.
Sprint-2	USN-2	As a user, I should have a landing page.
Sprint-3	USN-1	As a user,I should have a better ux and ui designs.
Sprint-3	USN-2	As a user, I have a page for features of the web application.
Sprint-4	USN-1	As a user, I should be able to enter my name and email and do registration.
Sprint-4	USN-2	As a user, with my details loan approval will be predicted

6. CODING & SOLUTIONING

6.1 Feature 1

User can enter their personal details and loan details which will be taken as input, upon which our model will run and make predictions

```
<html>
    <head>
        <link rel="stylesheet"
href="static/css/result.css">
        <title>Smart Lender</title>
    </head>
    <body class="predict-page">
        <form action="/submit" method="post">
        <div class="container">
            <div class="left">
                <div class="header">
                    <h2 class="animation a1">Predict Your
Elgibility</h2>
                    <p></p>
                </div>
                <div class="form">
                    <label>Gender</label>
                    <select name="query1" class="form-field
animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option value="Male">Male</option>
                        <option
value="Female">Female</option>
                    </select><br><br>

                        <label>Married</label><br>
                        <select name="query2"
class="form-field animation a3 ">
                            <option value="none" selected
disabled hidden></option>
                            <option
value="Yes">Yes</option>
                            <option value="No">No</option>
                        </select><br><br>

                        <label>Dependents</label><br>
```

```
                    <select name="query3"
class="form-field animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option value="0">0</option>
                        <option value="1">1</option>
                        <option value="2">2</option>
                        <option value="3">3+</option>
                    </select><br><br>
                    <label>Education</label><br>
                    <select name="query4"
class="form-field animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option
value="Graduate">Graduate</option>
                        <option value="Not
Graduate">Not Graduate</option>
                    </select><br><br>

                    <label>Self Employed</label><br>
                    <select name="query5"
class="form-field animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option
value="Yes">Yes</option>
                        <option value="No">No</option>
                    </select><br><br>

                    <label>Applicant Income</label><br>
                    <input type="text" name="query6"
class="form-field animation a3" required><br><br>

                    <label>Coapplicant
Income</label><br>
                    <input type="text" name="query7"
class="form-field animation a3" required><br><br>

                    <label>Loan Amount</label><br>
                    <input type="text" name="query8"
class="form-field animation a3" required><br><br>

                    <label>Loan Amount Term</label><br>
```

```
                    <select name="query9"
class="form-field animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option value="12">12</option>
                        <option value="36">36</option>
                        <option value="60">60</option>
                        <option value="84">84</option>
                        <option
value="120">120</option>
                        <option
value="180">180</option>
                        <option
value="240">240</option>
                        <option
value="300">300</option>
                        <option
value="360">360</option>
                        <option
value="480">480</option>
                    </select><br><br>

                    <label>Credit history</label><br>
                    <select name="query10"
class="form-field animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option value="0">0</option>
                        <option value="1">1</option>
                    </select><br><br>
                    <label>Property area</label><br>
                    <select name="query11"
class="form-field animation a3">
                        <option value="none" selected
disabled hidden></option>
                        <option
value="Urban">Urban</option>
                        <option
value="Rural">Rural</option>
                        <option value="Semiurban">Semi
Urban</option>
                    </select><br><br>
                    <button type="submit"
class="sub">Submit</button>
```

```
<!--                        <a href = "mailto:
bhavoffl@gmail.com">Send Email</a>-->
                </div>
            </div>
            <div class="right">
<!--                    <img
src="static/images/piggy-bank.jpg">-->
            </div>
        </div>
    </form>
  </body>
</html>
```

6.2 Feature 2

User can see the features of the model, and also visit the project repository on github

```
<!DOCTYPE html>
<html lang="en">
<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">
  <link
href="https://unpkg.com/tailwindcss@^2/dist/tailwind.
min.css" rel="stylesheet">
  <title>Smart Lender</title>
</head>
<body>
  <header class="text-gray-400 bg-gray-900
body-font">
    <div class="container mx-auto flex flex-wrap
p-5 flex-col md:flex-row items-center" id="Home">
      <a class="flex title-font font-medium
items-center text-white mb-4 md:mb-0">
        <svg xmlns="http://www.w3.org/2000/svg"
fill="none" stroke="currentColor"
stroke-linecap="round" stroke-linejoin="round"
stroke-width="2" class="w-10 h-10 text-white p-2
bg-indigo-500 rounded-full" viewBox="0 0 24 24">
          <path d="M12 2L2 7l10 5 10-5-10-5zM2
17l10 5 10-5M2 12l10 5 10-5"></path>
        </svg>
```

```
            <span class="ml-3 text-xl">Loan
Prediction</span>
          </a>
          <nav class="md:ml-auto flex flex-wrap
items-center text-base justify-center">
            <!-- <a class="mr-5 hover:text-white"
href="#Home">Home</a> -->
            <a class="mr-5 hover:text-white"
href="#About">About</a>
            <a class="mr-5 hover:text-white"
href="#Features">Features</a>
            <a class="mr-5 hover:text-white"
href="#Predict">Predict</a>
          </nav>
        </div>
      </header>

      <section class="text-gray-400 bg-gray-900
body-font" id="About">
        <div class="container mx-auto flex px-5 py-24
md:flex-row flex-col items-center">
          <div class="lg:max-w-lg lg:w-full md:w-1/2
w-5/6 md:mb-0 mb-10">
            <img class="object-cover object-center
rounded" alt="hero" src="static/images/loan.jpg">
          </div>
          <div class="lg:flex-grow md:w-1/2 lg:pl-24
md:pl-16 flex flex-col md:items-start md:text-left
items-center text-center">
            <h1 class="title-font sm:text-4xl text-3xl
mb-4 font-medium text-white">Loan Approval
              <br class="hidden
lg:inline-block">Prediction System
            </h1>
            <p class="mb-8 leading-relaxed">
            One of the most important factors which
affect our country’s economy and financial condition
is the credit system governed by the banks. The
process of bank credit risk evaluation is recognized
at banks across the globe. “As we know credit risk
evaluation is very crucial, there is a variety of
techniques are used for risk level calculation. In
addition, credit risk is one of the main functions of
the banking community.
```

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

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

            </p>
            <div class="flex justify-center">
              <a class="inline-flex text-white bg-indigo-500 border-0 py-2 px-6 focus:outline-none hover:bg-indigo-600 rounded text-lg" href="https://github.com/IBM-EPBL/IBM-Project-17163-1659629444" target="_blank">Github</a>
<!--               <button class="ml-4 inline-flex text-gray-400 bg-gray-800 border-0 py-2 px-6 focus:outline-none hover:bg-gray-700 hover:text-white rounded text-lg">Video</button>-->
            </div>
          </div>
        </div>
      </section>

      <section class="text-gray-400 body-font bg-gray-900" id="Features">
        <div class="container px-5 py-24 mx-auto">
          <div class="flex flex-wrap w-full mb-20 flex-col items-center text-center">
            <h1 class="sm:text-3xl text-2xl font-medium title-font mb-2 text-white">Features</h1>
          </div>
          <div class="flex flex-wrap -m-4">
            <div class="xl:w-1/3 md:w-1/2 p-4">
```

```
            <div class="border border-gray-700
border-opacity-75 p-6 rounded-lg">
                <div class="w-10 h-10 inline-flex
items-center justify-center rounded-full bg-gray-800
text-indigo-400 mb-4">
                    <svg fill="none"
stroke="currentColor" stroke-linecap="round"
stroke-linejoin="round" stroke-width="2" class="w-6
h-6" viewBox="0 0 24 24">
                        <path d="M22 12h-4l-3 9L9 3l-3
9H2"></path>
                    </svg>
                </div>
                <h2 class="text-lg text-white
font-medium title-font mb-2">Easy to use</h2>
                <p class="leading-relaxed
text-base">Easy to use tool, simply put some features
and get to know whether you'll get an Approval or
not.</p>
            </div>
        </div>
        <div class="xl:w-1/3 md:w-1/2 p-4">
            <div class="border border-gray-700
border-opacity-75 p-6 rounded-lg">
                <div class="w-10 h-10 inline-flex
items-center justify-center rounded-full bg-gray-800
text-indigo-400 mb-4">
                    <svg fill="none"
stroke="currentColor" stroke-linecap="round"
stroke-linejoin="round" stroke-width="2" class="w-6
h-6" viewBox="0 0 24 24">
                        <circle cx="6" cy="6"
r="3"></circle>
                        <circle cx="6" cy="18"
r="3"></circle>
                        <path d="M20 4L8.12 15.88M14.47
14.48L20 20M8.12 8.12L12 12"></path>
                    </svg>
                </div>
                <h2 class="text-lg text-white
font-medium title-font mb-2">Type string values</h2>
                <p class="leading-relaxed text-base">We
can put categorical values</p>
            </div>
        </div>
```

```
            <div class="xl:w-1/3 md:w-1/2 p-4">
              <div class="border border-gray-700
border-opacity-75 p-6 rounded-lg">
                <div class="w-10 h-10 inline-flex
items-center justify-center rounded-full bg-gray-800
text-indigo-400 mb-4">
                  <svg fill="none"
stroke="currentColor" stroke-linecap="round"
stroke-linejoin="round" stroke-width="2" class="w-6
h-6" viewBox="0 0 24 24">
                    <path d="M20 21v-2a4 4 0 00-4-4H8a4
4 0 00-4 4v2"></path>
                    <circle cx="12" cy="7"
r="4"></circle>
                  </svg>
                </div>
                <h2 class="text-lg text-white
font-medium title-font mb-2">Accuracy</h2>
                <p class="leading-relaxed text-base">It
also has good Accuracy that makes good prediction
using low error.</p>
              </div>
            </div>
            <div class="xl:w-1/3 md:w-1/2 p-4">
              <div class="border border-gray-700
border-opacity-75 p-6 rounded-lg">
                <div class="w-10 h-10 inline-flex
items-center justify-center rounded-full bg-gray-800
text-indigo-400 mb-4">
                  <svg fill="none"
stroke="currentColor" stroke-linecap="round"
stroke-linejoin="round" stroke-width="2" class="w-6
h-6" viewBox="0 0 24 24">
                    <path d="M4 15s1-1 4-1 5 2 8 2 4-1
4-1V3s-1 1-4 1-5-2-8-2-4 1-4 1zM4 22v-7"></path>
                  </svg>
                </div>
                <h2 class="text-lg text-white
font-medium title-font mb-2">Model</h2>
                <p class="leading-relaxed
text-base">Tested on various model, but chosen,
Gradient Boost</p>
              </div>
            </div>
            <div class="xl:w-1/3 md:w-1/2 p-4">
```

```
            <div class="border border-gray-700
border-opacity-75 p-6 rounded-lg">
              <div class="w-10 h-10 inline-flex
items-center justify-center rounded-full bg-gray-800
text-indigo-400 mb-4">
                <svg fill="none"
stroke="currentColor" stroke-linecap="round"
stroke-linejoin="round" stroke-width="2" class="w-6
h-6" viewBox="0 0 24 24">
                  <path d="M21 12.79A9 9 0 1111.21 3
7 7 0 0021 12.79z"></path>
                </svg>
              </div>
              <h2 class="text-lg text-white
font-medium title-font mb-2">Interactive</h2>
              <p class="leading-relaxed text-base">A
very nice interactive and pleasant UI/UX.</p>
            </div>
          </div>
          <div class="xl:w-1/3 md:w-1/2 p-4">
            <div class="border border-gray-700
border-opacity-75 p-6 rounded-lg">
              <div class="w-10 h-10 inline-flex
items-center justify-center rounded-full bg-gray-800
text-indigo-400 mb-4">
                <svg fill="none"
stroke="currentColor" stroke-linecap="round"
stroke-linejoin="round" stroke-width="2" class="w-6
h-6" viewBox="0 0 24 24">
                  <path d="M12 22s8-4 8-10V5l-8-3-8
3v7c0 6 8 10 8 10z"></path>
                </svg>
              </div>
              <h2 class="text-lg text-white
font-medium title-font mb-2">Your shield!</h2>
              <p class="leading-relaxed
text-base">Use our magic predictor if you want to
whether you are eligible for a loan.</p>
            </div>
          </div>
        </div>

      </div>
    </section>
    <section id="Predict">
```

```
    <form class="form-horizontal" action = "/predict"
method="post">
      <section class="text-gray-400 bg-gray-900
body-font">
        <div class="container px-5 py-24 mx-auto">
          <div class="flex flex-col text-center w-full
mb-12">
            <h1 class="sm:text-3xl text-2xl font-medium
title-font mb-4 text-white">It's your predictor!</h1>
            <p class="lg:w-2/3 mx-auto leading-relaxed
text-base">Now, Use the Predictor! .</p>
            <a class="flex mx-auto mt-16 text-white
bg-indigo-500 border-0 py-2 px-8 focus:outline-none
hover:bg-indigo-600 rounded text-lg"
href="predict">Predict</a>
          </div>
          </div>

      </section>
      </form>
</body>
</html>
```

7. TESTING

Performance Testing

S.No.	Parameter	Values
1.	Metrics	**Classification Model: Decision Tree** Accuracy Score - 84% **Classification Report** precision recall f1-score support 0 0.83 0.84 0.83 105 1 0.85 0.85 0.85 118
2.		**Classification Model: Random Forest** Accuracy Score - 84% **Classification Report** precision recall f1-score support 0 0.89 0.81 0.85 105 1 0.84 0.91 0.87 118
3.		**Classification Model: KNN** Accuracy Score - 78% **Classification Report** precision recall f1-score support 0 0.82 0.70 0.76 105 1 0.77 0.86 0.81 118
4.		**Classification Model: XGBoost** Accuracy Score - 85% **Classification Report** precision recall f1-score support 0 0.90 0.77 0.83 105 1 0.82 0.92 0.87 118

7.1 User Acceptance Testing and Test Cases

Sprint 1 test cases

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Expected Result	Actual Result	Status	TC for Automation(Y/N)
LoginPage_TC_OO 1	Functional	Predict page	Verify wheather loan details entered by the user is correctly received	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click submit	Prediction result page should appear	Working as expected	Pass	N

Sprint 2 test cases

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)
LoginPage_TC_OO 1	UI	Home Page	Verify wheather user is able to see the landing page	1.Enter URL and click go	Prediction result page should appear	Working as expected	Pass		
LoginPage_TC_OO 2	Functional	Predict Page	Verify wheather loan details entered by the user is correctly received	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click submit	Application should show below UI elements: • applicant_income • coapplicant_income • credit_history • dependents • education • loan_amount • loan_amt_term • married • property_area • self_employed	Working as expected	Pass	Steps are clear to follow	N

Sprint 3 test cases

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)
LoginPage_TC_OO 1	UI	Home Page	Verify wheather user is able to see the landing page	1.Enter URL and click go	Prediction result page should appear	Working as expected	Pass		
LoginPage_TC_OO 2	Functional	Predict Page	Verify wheather loan details entered by the user is correctly received	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click submit	Application should show below UI elements: • applicant_income • coapplicant_income • credit_history • dependents • education • loan_amount • loan_amt_term • married • property_area • self_employed	Working as expected	Pass	Steps are clear to follow	N
naturesPage_TC_OO	Functional	Features page	Verify whether the use can see features page at the click of the button	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click Go 5. View Features Page	1. User should see Features page 2. User should be able to go to our Github page when they click the "GITHUB" button 3. User should be able to navigate to the different sections of the page, including features, about and predict sections	Working as expected	Pass		N

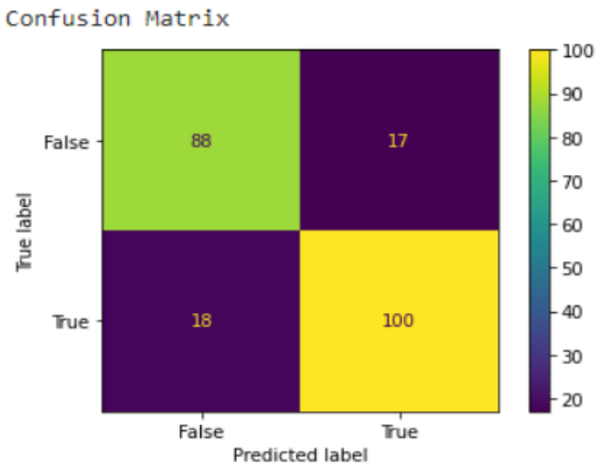
Sprint 4 test cases

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)
LoginPage_TC_OO 1	UI	Home Page	Verify wheather user is able to see the landing page	1.Enter URL and click go	Prediction result page should appear	Working as expected	Pass		
PredictPage_TC_O O2	Functional	Predict Page	Verify wheather loan details entered by the user is correctly received	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click submit	Application should show below UI elements: • applicant_income • coapplicant_income • credit_history • dependents • education • loan_amount • loan_amt_term • married • property_area • self_employed	Working as expected	Pass	Steps are clear to follow	N
eaturesPage_TC_OO	Functional	Features page	Verify whether the use can see features page at the click of the button	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click Go 5. View Features Page	1. User should see Features page 2. User should be able to go to our Github page when they click the "GITHUB" button 3. User should be able to navigate to the different sections of the page, including features, about and predict sections	Working as expected	Pass		N
Predictpage_TC_O O_02	Functional	Predict Page	Verify whether the user can go to predict page when they click predict button	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click Go 5. View Features Page 6. Click Predict button 7. View predict page	1. User should be taken to predict page when they click predict button	Working as expected	Pass		N
Predictpage_TC_O O2	Functional	Predict Page	Verify wheather loan details entered by the user is correctly received	1.Enter URL and click go 2.Click on predict button 3.Enter the details shown on the page 4.Click submit	Application should show below UI elements: • applicant_income • coapplicant_income • credit_history • dependents • education • loan_amount • loan_amt_term • married • property_area • self_employed • gender	Working as expected	Pass	Steps are clear to follow	N
Registerpage__TC_ _001	Functional	Register page	Verify whether use can see register page and enter details	1. Enter the URL and click register 2. User should enter first name, last name and email address			Pass		N
SubmitPage__TC__ 01	Functional	Submit page	Verify whether use is able to see the submit page which displays the result of the prediction	1. Enter the URL and click register 2. User should enter first name, last name and email address 3. User should go to features page 4. User should go to predict section and click PREDICT button 5. User should enter details in predict page 6. User should click SUBMIT button	User should see results of prediction on submit page	Working as expected	Pass		N

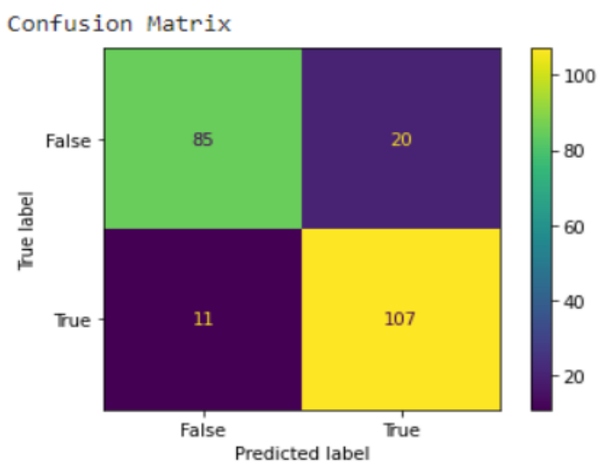
8. RESULTS

8.1 Performance Metrics

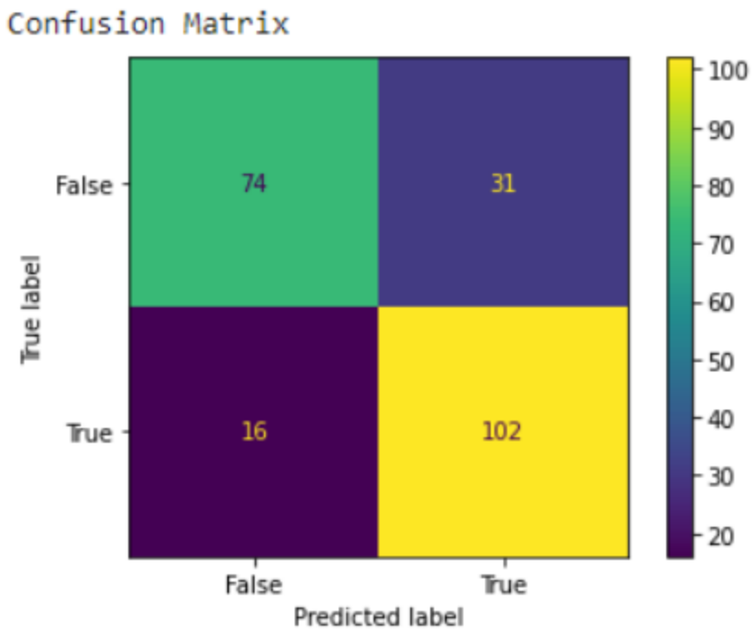
DECISION TREE CLASSIFIER



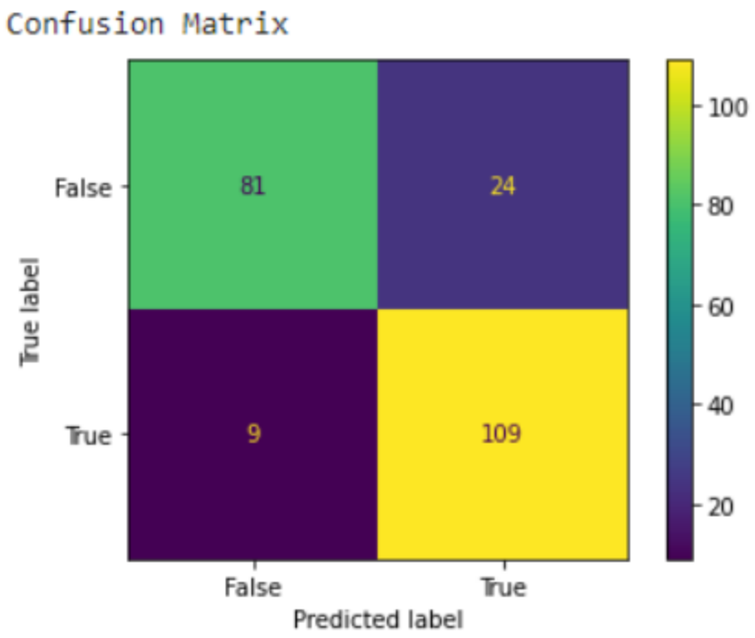
RANDOM FOREST CLASSIFIER



KNN



XG BOOST



9. ADVANTAGES & DISADVANTAGES

Advantages:

- Knowledge of Python Language with Machine Learning
- You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
- You will be able to know how to pre-process/clean the data using different data pre-processing techniques.
- Applying different algorithms according to the dataset and based on visualization.
- Real-Time Analysis of Project
- Building ease of User Interface (UI)
- Navigation of ideas towards other projects(creativeness)
- Knowledge of building ML models.
- How to build web applications using the Flask framework.

Disadvantages:

- As the given dataset is very small, the model is not able to make predictions on large values.
- For more accurate and real time results, we would need a dataset which contains millions of datapoints.

10. CONCLUSION

- After the Final Submission of test data, my accuracy score was 78%.
- Feature engineering helped me increase my accuracy.
- GradientBoost worked better than all other random forest, decision tree, regression models.

11. FUTURE SCOPE

More accuracy can be gained by increasing the size of the dataset by generating synthetic data which can be obtained by scaling the applicant income , co-applicant income and loan amount columns and adding it to the existing dataset and running our model on the new dataset. Although, synthetic data generation is a very tedious and difficult process, it can help achieve a better accuracy.

12. APPENDIX

Source Code

model.py

```
import pandas as pd
from matplotlib import pyplot as plt
import missingno as msno
import seaborn as sns
import numpy as np
from imblearn.combine import SMOTETomek
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix,
ConfusionMatrixDisplay
import warnings

warnings.filterwarnings("ignore")

loan_data = pd.read_csv(r'C:\Users\bhaav\OneDrive\Desktop\My_Projects_DS\LoanDataset.csv')


"""## EDA

## Uni variate Analysis
"""

plt.figure(figsize=(20, 12))
plt.subplot(231)
sns.distplot(loan_data['ApplicantIncome'], color='r')
plt.subplot(232)
sns.distplot(loan_data['Credit_History'])
plt.subplot(233)
```

```
sns.distplot(loan_data['CoapplicantIncome'], color='r')
plt.subplot(234)
sns.distplot(loan_data['LoanAmount'])
plt.subplot(235)
sns.distplot(loan_data['Loan_Amount_Term'], color='r')
# plt.show()

plt.figure(figsize=(12, 4))
plt.subplot(121)
sns.countplot(loan_data['Gender'])
plt.subplot(122)
sns.countplot(loan_data['Education'])
# plt.show()

plt.figure(figsize=(20, 5))
plt.subplot(131)
sns.countplot(loan_data['Gender'], hue=loan_data['Education'])
plt.subplot(132)
sns.countplot(loan_data['Married'], hue=loan_data['Gender'])
plt.subplot(133)
sns.countplot(loan_data['Self_Employed'], hue=loan_data['Education'])
# plt.show()
plt.figure(figsize=(20, 10))
sns.countplot(loan_data['Property_Area'], hue=loan_data['Loan_Amount_Term'])
# plt.show()

temp_data = loan_data.drop(
   ['Loan_ID', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term',
'Loan_Status'], axis=1)
cols = temp_data.columns
for c in cols:
   plt.figure(c, figsize=(7, 4))
   sns.countplot(temp_data[c], hue=loan_data['Loan_Status'])

# plt.show()

loan_data['ApplicantIncome'].max()
maximum_income = loan_data['ApplicantIncome'].unique().max()
minimum_income = loan_data['ApplicantIncome'].unique().min()
print(maximum_income, minimum_income)

loan_data['CoapplicantIncome'].max()
maximum_income = loan_data['CoapplicantIncome'].unique().max()
minimum_income = loan_data['CoapplicantIncome'].unique().min()
print(maximum_income, minimum_income)

copy_data = loan_data.copy()
bin_range = [150, 10000, 20000, 30000, 40000, 50000, 60000, 70000, 81000]
copy_data['Income_group'] = pd.cut(x=copy_data['ApplicantIncome'], bins=bin_range)
copy_data['Income_group'].value_counts()

plt.figure(figsize=(20, 5))
sns.countplot(x=copy_data['Income_group'], hue=copy_data['Loan_Status'])
# plt.show()

bin_range = [1, 10000, 20000, 30000, 42000]
copy_data['CoIncome_group'] = pd.cut(x=copy_data['CoapplicantIncome'], bins=bin_range)
copy_data['CoIncome_group'].value_counts()

plt.figure(figsize=(20, 5))
sns.countplot(x=copy_data['CoIncome_group'], hue=copy_data['Loan_Status'])
# plt.show()

"""### Multivariate Analysis"""
```

```
sns.swarmplot(loan_data['Gender'], loan_data['ApplicantIncome'], hue=loan_data['Loan_Status'])

"""### Descriptive Analysis"""

loan_data.describe()

stat_data = loan_data[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term',
'Credit_History']]
for i in stat_data:
    plt.hist(loan_data[i])
    # plt.show()

"""## Data Preprocessing"""

loan_data = loan_data.drop(columns=['Loan_ID'], axis=1)
loan_data.head()

"""## Handling Missing Values"""

loan_data.isna()

null_data = loan_data.isna().sum()
null_data.sort_values()

msno.bar(loan_data)

msno.matrix(loan_data)

unique_cols = loan_data.drop(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount'], axis=1)
for col in unique_cols:
    print(col, loan_data[col].unique())

loan_data['Gender'] = loan_data['Gender'].fillna(loan_data['Gender'].mode()[0])
loan_data['Married'] = loan_data['Married'].fillna(loan_data['Married'].mode()[0])
loan_data['Dependents'] = loan_data['Dependents'].str.replace('+', '')
loan_data['Self_Employed'] = loan_data['Self_Employed'].fillna(loan_data['Self_Employed'].mode()[0])
loan_data['LoanAmount'] = loan_data['LoanAmount'].fillna(loan_data['LoanAmount'].mode()[0])
loan_data['Loan_Amount_Term'] =
loan_data['Loan_Amount_Term'].fillna(loan_data['Loan_Amount_Term'].mode()[0])
loan_data['Credit_History'] = loan_data['Credit_History'].fillna(loan_data['Credit_History'].mode()[0])
# loan_data

loan_data['CoapplicantIncome'] = loan_data['CoapplicantIncome'].astype('int64')
loan_data['LoanAmount'] = loan_data['LoanAmount'].astype('int64')
loan_data['Loan_Amount_Term'] = loan_data['Loan_Amount_Term'].astype('int64')
loan_data['Credit_History'] = loan_data['Credit_History'].astype('int64')
# loan_data

label_encoder = preprocessing.LabelEncoder()
label_encoding_columns = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed',
'Property_Area',
                          'Loan_Status']
for col in label_encoding_columns:
    loan_data[col] = label_encoder.fit_transform(loan_data[col])

# loan_data

smk = SMOTETomek(0.90)

y = loan_data['Loan_Status']
x = loan_data.drop(columns=['Loan_Status'], axis=1)

x_bal, y_bal = smk.fit_resample(x, y)
```

```
print(y.value_counts())
print(y_bal.value_counts())

sc = StandardScaler()
x_bal = sc.fit_transform(x_bal)
x_bal = pd.DataFrame(x_bal)

# loan_data

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

"""## MODEL BUILDING"""

acc = {}


def Decision_Tree(X_train, X_test, y_train, y_test):
    dt = DecisionTreeClassifier()
    dt.fit(X_train, y_train)

    # predicting the outcome
    y_pred = dt.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['Decision Tree Classifier'] = accuracy_score(y_test, y_pred)


Decision_Tree(X_train, X_test, y_train, y_test)


def Random_Forest(X_train, X_test, y_train, y_test):
    rf = RandomForestClassifier()
    rf.fit(X_train, y_train)

    # predicting the outcome
    y_pred = rf.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['Random Forest Classifier'] = accuracy_score(y_test, y_pred)


Random_Forest(X_train, X_test, y_train, y_test)


def KNN(X_train, X_test, y_train, y_test):
    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)

    # predicting the outcome
    y_pred = knn.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
```

```
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['KNN'] = accuracy_score(y_test, y_pred)


KNN(X_train, X_test, y_train, y_test)


def XGboost(X_train, X_test, y_train, y_test):
    xg = GradientBoostingClassifier()
    xg.fit(X_train, y_train)

    # predicting the outcome
    y_pred = xg.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['Gradient Boost'] = accuracy_score(y_test, y_pred)


XGboost(X_train, X_test, y_train, y_test)
xg = GradientBoostingClassifier()
xg.fit(X_train, y_train)

# input_feature = [1,1,1,1,1,5849,1508,120,360,1,1]
input_feature = ['Female', 'Yes', 0, 'Graduate', 'Yes', 584900, 0, 120, 360, 1, 'Urban']
input_feature = [np.array(input_feature)]
print(input_feature)
names = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome',
'CoapplicantIncome',
        'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area']
data = pd.DataFrame(input_feature, columns=names)
print(data)
new = data.copy()
label_encoder = preprocessing.LabelEncoder()
label_encoding_columns = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed',
'Property_Area']
for col in label_encoding_columns:
    new[col] = label_encoder.fit_transform(new[col])
preed = xg.predict(new)
print(preed)
print(acc)

# import pickle
#
# xg = GradientBoostingClassifier()
# xg.fit(X_train, y_train)
# # save the model to disk
# pickle.dump(xg, open('model3.pkl','wb'))
#
# model = pickle.load(open('model3.pkl','rb'))
#
# print(model.predict(X_test))
```

server.py

```
import pandas as pd
from matplotlib import pyplot as plt
import missingno as msno
import seaborn as sns
import numpy as np
from imblearn.combine import SMOTETomek
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix,
ConfusionMatrixDisplay
import warnings

warnings.filterwarnings("ignore")

loan_data = pd.read_csv(r'LoanDataset.csv')


"""## EDA

## Uni variate Analysis
"""

plt.figure(figsize=(20, 12))
plt.subplot(231)
sns.distplot(loan_data['ApplicantIncome'], color='r')
plt.subplot(232)
sns.distplot(loan_data['Credit_History'])
plt.subplot(233)
sns.distplot(loan_data['CoapplicantIncome'], color='r')
plt.subplot(234)
sns.distplot(loan_data['LoanAmount'])
plt.subplot(235)
sns.distplot(loan_data['Loan_Amount_Term'], color='r')
# plt.show()

plt.figure(figsize=(12, 4))
plt.subplot(121)
sns.countplot(loan_data['Gender'])
plt.subplot(122)
sns.countplot(loan_data['Education'])
# plt.show()

plt.figure(figsize=(20, 5))
plt.subplot(131)
sns.countplot(loan_data['Gender'], hue=loan_data['Education'])
plt.subplot(132)
sns.countplot(loan_data['Married'], hue=loan_data['Gender'])
plt.subplot(133)
sns.countplot(loan_data['Self_Employed'], hue=loan_data['Education'])
# plt.show()
plt.figure(figsize=(20, 10))
sns.countplot(loan_data['Property_Area'], hue=loan_data['Loan_Amount_Term'])
# plt.show()

temp_data = loan_data.drop(
    ['Loan_ID', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term',
'Loan_Status'], axis=1)
cols = temp_data.columns
for c in cols:
```

```
    plt.figure(c, figsize=(7, 4))
    sns.countplot(temp_data[c], hue=loan_data['Loan_Status'])

# plt.show()

loan_data['ApplicantIncome'].max()
maximum_income = loan_data['ApplicantIncome'].unique().max()
minimum_income = loan_data['ApplicantIncome'].unique().min()
print(maximum_income, minimum_income)

loan_data['CoapplicantIncome'].max()
maximum_income = loan_data['CoapplicantIncome'].unique().max()
minimum_income = loan_data['CoapplicantIncome'].unique().min()
print(maximum_income, minimum_income)

copy_data = loan_data.copy()
bin_range = [150, 10000, 20000, 30000, 40000, 50000, 60000, 70000, 81000]
copy_data['Income_group'] = pd.cut(x=copy_data['ApplicantIncome'], bins=bin_range)
copy_data['Income_group'].value_counts()

plt.figure(figsize=(20, 5))
sns.countplot(x=copy_data['Income_group'], hue=copy_data['Loan_Status'])
# plt.show()

bin_range = [1, 10000, 20000, 30000, 42000]
copy_data['CoIncome_group'] = pd.cut(x=copy_data['CoapplicantIncome'], bins=bin_range)
copy_data['CoIncome_group'].value_counts()

plt.figure(figsize=(20, 5))
sns.countplot(x=copy_data['CoIncome_group'], hue=copy_data['Loan_Status'])
# plt.show()

"""### Multivariate Analysis"""

sns.swarmplot(loan_data['Gender'], loan_data['ApplicantIncome'], hue=loan_data['Loan_Status'])

"""### Descriptive Analysis"""

loan_data.describe()

stat_data = loan_data[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term',
'Credit_History']]
for i in stat_data:
    plt.hist(loan_data[i])
    # plt.show()

"""## Data Preprocessing"""

loan_data = loan_data.drop(columns=['Loan_ID'], axis=1)
loan_data.head()

"""## Handling Missing Values"""

loan_data.isna()

null_data = loan_data.isna().sum()
null_data.sort_values()

msno.bar(loan_data)

msno.matrix(loan_data)

unique_cols = loan_data.drop(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount'], axis=1)
for col in unique_cols:
```

```
    print(col, loan_data[col].unique())

loan_data['Gender'] = loan_data['Gender'].fillna(loan_data['Gender'].mode()[0])
loan_data['Married'] = loan_data['Married'].fillna(loan_data['Married'].mode()[0])
loan_data['Dependents'] = loan_data['Dependents'].str.replace('+', '')
loan_data['Self_Employed'] = loan_data['Self_Employed'].fillna(loan_data['Self_Employed'].mode()[0])
loan_data['LoanAmount'] = loan_data['LoanAmount'].fillna(loan_data['LoanAmount'].mode()[0])
loan_data['Loan_Amount_Term'] =
loan_data['Loan_Amount_Term'].fillna(loan_data['Loan_Amount_Term'].mode()[0])
loan_data['Credit_History'] = loan_data['Credit_History'].fillna(loan_data['Credit_History'].mode()[0])
# loan_data

loan_data['CoapplicantIncome'] = loan_data['CoapplicantIncome'].astype('int64')
loan_data['LoanAmount'] = loan_data['LoanAmount'].astype('int64')
loan_data['Loan_Amount_Term'] = loan_data['Loan_Amount_Term'].astype('int64')
loan_data['Credit_History'] = loan_data['Credit_History'].astype('int64')
# loan_data

label_encoder = preprocessing.LabelEncoder()
label_encoding_columns = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed',
'Property_Area',
                          'Loan_Status']
for col in label_encoding_columns:
    loan_data[col] = label_encoder.fit_transform(loan_data[col])

# loan_data

smk = SMOTETomek(0.90)

y = loan_data['Loan_Status']
x = loan_data.drop(columns=['Loan_Status'], axis=1)

x_bal, y_bal = smk.fit_resample(x, y)
print(y.value_counts())
print(y_bal.value_counts())

sc = StandardScaler()
x_bal = sc.fit_transform(x_bal)
x_bal = pd.DataFrame(x_bal)

# loan_data

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

"""## MODEL BUILDING"""

acc = {}


def Decision_Tree(X_train, X_test, y_train, y_test):
    dt = DecisionTreeClassifier()
    dt.fit(X_train, y_train)

    # predicting the outcome
    y_pred = dt.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['Decision Tree Classifier'] = accuracy_score(y_test, y_pred)
```

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


def Random_Forest(X_train, X_test, y_train, y_test):
    rf = RandomForestClassifier()
    rf.fit(X_train, y_train)

    # predicting the outcome
    y_pred = rf.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['Random Forest Classifier'] = accuracy_score(y_test, y_pred)


Random_Forest(X_train, X_test, y_train, y_test)


def KNN(X_train, X_test, y_train, y_test):
    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)

    # predicting the outcome
    y_pred = knn.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['KNN'] = accuracy_score(y_test, y_pred)


KNN(X_train, X_test, y_train, y_test)


def XGboost(X_train, X_test, y_train, y_test):
    xg = GradientBoostingClassifier()
    xg.fit(X_train, y_train)

    # predicting the outcome
    y_pred = xg.predict(X_test)
    print("Confusion Matrix")
    cm = confusion_matrix(y_test, y_pred)
    cm_display = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[False, True])
    cm_display.plot()
    # plt.show()
    print("Classification Report")
    cr = classification_report(y_test, y_pred)
    print(cr)
    acc['Gradient Boost'] = accuracy_score(y_test, y_pred)


XGboost(X_train, X_test, y_train, y_test)
xg = GradientBoostingClassifier()
```

```
xg.fit(X_train, y_train)

# input_feature = [1,1,1,1,1,5849,1508,120,360,1,1]
input_feature = ['Female', 'Yes', 0, 'Graduate', 'Yes', 584900, 0, 120, 360, 1, 'Urban']
input_feature = [np.array(input_feature)]
print(input_feature)
names = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome',
'CoapplicantIncome',
       'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area']
data = pd.DataFrame(input_feature, columns=names)
print(data)
new = data.copy()
label_encoder = preprocessing.LabelEncoder()
label_encoding_columns = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed',
'Property_Area']
for col in label_encoding_columns:
   new[col] = label_encoder.fit_transform(new[col])
preed = xg.predict(new)
print(preed)
print(acc)

# import pickle
#
# xg = GradientBoostingClassifier()
# xg.fit(X_train, y_train)
# # save the model to disk
# pickle.dump(xg, open('model3.pkl','wb'))
#
# model = pickle.load(open('model3.pkl','rb'))
#
# print(model.predict(X_test))
```

GitHub& Project Demo Link

https://github.com/IBM-EPBL/IBM-Project-17163-1659629444/tree/main/Final%20Deliverables