Project Report

Date	21st Nov 2023
Team ID	Team-591689
Project Name	Smart Lender - Applicant Credibility Prediction for Loan Approval

<u>INDEX</u>

INTRODUCTION

1.1 PROJECT OVERVIEW:

The Smart Lender project is a creative endeavour to create a thorough and reliable prediction model that can evaluate loan applicants' credibility and forecast the possibility of loan acceptance. The project uses machine learning to create risk-averse, well-informed decisions, with the goal of improving the current loan approval process. The project aims to improve the difficulties lenders encounter in assessing loan applications and lowering the risk of fraud and bad debt by utilizing data and sophisticated algorithms.

The project involves gathering and organizing pertinent data from multiple sources, applying feature engineering to derive significant attributes from the data, utilizing predictive modelling to build a strong and precise model, deploying the model to the lending platform, and continuously observing and enhancing the model to guarantee its efficacy in the long run.

If the Smart Lender project is implemented successfully, lenders, borrowers, and the financial ecosystem as a whole stand to gain significantly from lower fraud and bad debt risks, increased loan approval process efficiency and transparency, improved decision-making and risk assessment, faster and more efficient loan approval for borrowers, higher chances of loan approval, improved credit availability, improved financial inclusion and opportunities, increased financial system resilience and stability, lower overall credit risk, increased consumer confidence and participation, and promotion of financial innovation.

1.2 PURPOSE:

The banking industry's control over the credit system is one of the key elements influencing the financial and economic health of our nation. Banks all throughout the world use the same procedure for evaluating bank credit risk. There are numerous methods for calculating risk levels, and as we all know, credit risk rating is extremely important. Furthermore, a primary responsibility of the banking industry is credit risk.

Predicting credit defaulters is one of the most challenging jobs any bank has. However, by predicting the loan defaulters, the banks may be able to minimize their loss by cutting back on their non-profit assets, which would allow for the loss-free recovery of sanctioned loans and allow it to function as a contributing factor to the bank statement. This justifies the significance of researching this loan approval forecast. The prediction of this kind of data greatly benefits from the application of machine learning techniques.

By creating this kind of a thorough and reliable prediction model that can accurately determine the credibility of loan applicants and forecast the likelihood of loan acceptance, the Smart Lender project seeks to transform the loan approval procedure. By utilizing machine learning, this project seeks to reduce the risk of fraud and bad debt while addressing the difficulties lenders have in assessing loan applications. Smart Lender will help lenders make educated and risk-averse decisions by creating a predictive model that reliably evaluates creditworthiness and forecasts loan repayment behaviour, lowering the likelihood of defaults and raising approval rates. Borrowers will profit from quicker credit availability and more fair lending practices as a result.

LITERATURE SURVEY

2.1 EXISTING PROBLEM:

The traditional loan approval procedure is frequently criticized for being biased, inefficient, and slow. In order to evaluate risk, lenders mostly rely on manual assessments and conventional credit ratings, which can result in drawn-out application procedures, high rejection rates, and unethical lending practices. Traditional credit scores frequently fail of providing a complete picture of an applicant's creditworthiness because they ignore non-traditional elements that could affect loan repayment behaviour and ignore alternative data sources. Furthermore, borrowers are frequently left feeling dissatisfied and ignorant during the loan approval process, which makes it more difficult for them to raise their creditworthiness and gain access to funding.

The existing system also has a difficult time detecting and stopping fraud, which increases losses for lenders and erodes borrower confidence in the loan application process.

2.2 REFERENCES:

1. "Predicting Loan Defaults with Machine Learning" by John M. Henderson and Richard E. Krasinski (2014): This paper explores the use of machine learning techniques to predict loan defaults. The authors found that machine learning models can outperform traditional credit scoring models in predicting loan defaults.
2. "The Use of Machine Learning for Credit Risk Assessment" by Michael Baesens, Nathalie Van Gestel, and Bart Baesens (2010): This paper provides a comprehensive overview of the use of machine learning for credit risk assessment. The authors discuss various machine

learning algorithms that can be used for this task and provide examples of successful applications.

3. "A Machine Learning Approach to Fraud Detection in Credit Card Applications" by Silvia Chiappa and David Lozi (2007): This paper presents a machine learning approach to fraud detection in credit card applications. The authors found that their approach can effectively identify fraudulent applications and reduce fraud losses.

4. "Machine Learning for Loan Approval Prediction" by Ankur Srivastava (2023): This paper provides a tutorial on how to use machine learning to predict loan approval. The author covers the entire process, from data collection and preparation to model training and evaluation.

2.3 PROBLEM STATEMENT DEFINITION:

As the banking industry grows, more and more people are applying for bank loans. However, banks can only provide loans to a restricted number of individuals due to their limited assets, thus determining who is eligible for a loan is a routine procedure for them. By choosing a reliable individual, we attempted to lower this risk and preserve a great deal of bank resources. To do this, the computer was taught using a machine learning model, which produced the most accurate result, by mining the prior data of the borrowers who had previously received loans. This project's primary objective is to forecast the safety of a loan assignment to a certain individual.

A variety of loan types are handled by banks and finance organizations in our nation, including loans for personal, home, business, education, and retail purposes. In towns, cities, and villages are all the businesses and banks. Once the client has applied for a loan, these banks and businesses wish to confirm the applicant's customer information qualification. The primary goal of the system is to approve loan applications based on train models.

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

A problem statement often touches on the 5 w's (who, what, where, when and why) of the problem. We used mural empathy map canvas for analysing the 5 w's.

First, we divided the uses into 4 segments who will use the product.

1. Borrowers: People who are seeking loans for a variety of purposes, such as starting a business, buying a home, or financing their education.
2. Lenders: Banks, credit unions, and other financial institutions that provide loans.
3. Investors: People who invest in loans, either directly or through platforms like Lending Club and Prosper.
4. Regulators: Government agencies that oversee the lending industry.

Now, we tried to find answers for different questions with respect the above users and placed them in the empathy map template.
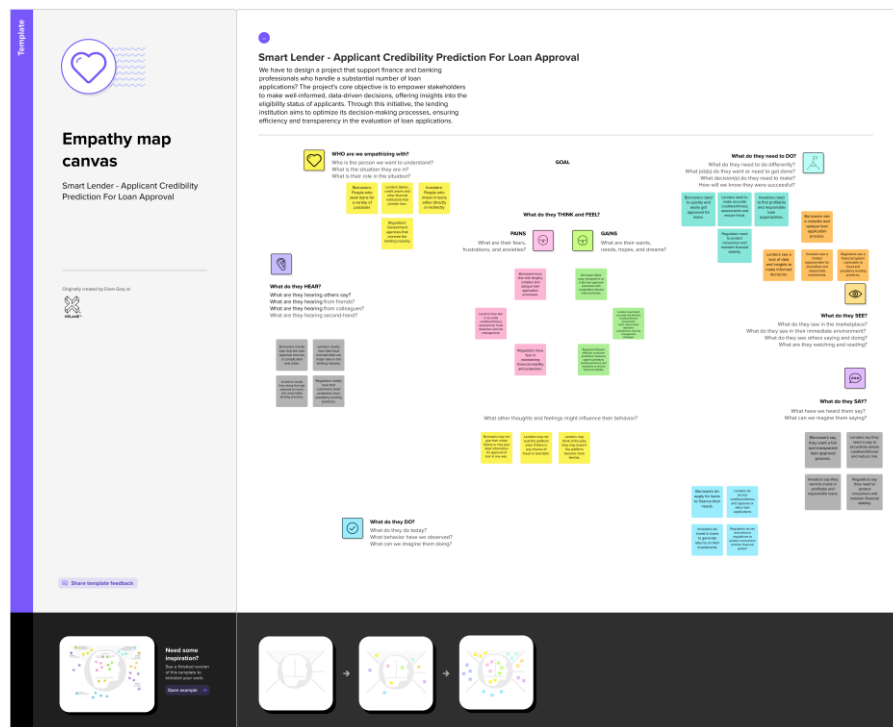
3.2 IDEATION AND BRAINSTORMING:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

We used mural brainstorm and idea prioritization template for this phase.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Step-2: Brainstorm, Idea Listing and Grouping

Step-3: Idea Prioritization

THE PROBLEM STATEMENT:

It is very difficult to reliably determine an applicant's credibility and forecast their loan payback behaviour because of the present loan approval procedure' inefficiency. This raises the risk of fraud and leads to high rejection rates and unfair lending practices. A reliable predictive model is required to deal with these issues. It must be able predict loan payback patterns, evaluate applicant credibility, streamline the loan approval process, and maintain fairness.

3.3 PROPOSED SOLUTION:

SOLUTION DESCRIPTION:

We propose this project, which uses advanced methods of machine learning to create a complete and reliable predictive model for loan repayment prediction and applicant credibility evaluation, in order to overcome the inefficiencies of the current loan approval process.

UNIQUENESS/NOVELTY:

The Smart Lender project utilizes advanced machine learning techniques to develop a comprehensive and robust predictive model for applicant credibility assessment and loan repayment prediction, unlike traditional methods that rely solely on credit scores or manual decision-making.

SOCIAL IMPACT/CUSTOMER SATISFACTION:

This project is expected to have a beneficial social impact through reducing the risk of predatory lending practices, increasing financial inclusion, and facilitating better credit access for marginalized communities. Borrowers, lenders, and investors will be more satisfied efficiency in the loan approval process.

BUSINESS MODEL:

By providing lenders, borrowers, and investors with a range of services, this project has the potential to be a successful business model. This platform can make money through efficient monthly or yearly subscriptions.

SCALABILITY OF THE SOLUTION:

Although it is now just a website, we can expand it to an API and integrate it with mobile banking apps to provide consumers with even more simple access to the results on their phones. On training with more data, the model can get more accuracy.

3.4 PROBLEM SOLUTION FIT:

CUSTOMER SEGMENT(S)

i. Buying a home   ii. Medical emergency   iii. Wedding purpose

iv. To start a business   v. To fund working capital

JOBS-TO-BE-DONE/PROBLEMS

i. Receive loan application   ii. Document collection

iii.Free verification process   iv. Finding perfective loan

TRIGGERS

• A personal loan is one option for financing the purchase of a person's dream
• One can avail a personal loan to fund any big-ticket purchase like electronic appliance and gadgets

EMOTIONS: BEFORE/AFTER

- I. Anxiety - this is a stress with the scab at the torn off
- II. Anger - Instead of panicking or denying problem, victim get mad
- III. Depression - hopelessness set in, as does no self-esteem it can lead

AVAILABLE SOLUTIONS

➢ If they applied for back loans
➢ Take insurance with big ticket
➢ Ensure time and regular prepayment

BEHAVIOUR

- check your credit score
- take steps to improve your score by
- checking for inaccuracy and paying

CHANNELS of BEHAVIOUR

1.1 ONLINE   1.2 OFFLINE
i. Make it easy to contact   i. First impression matters
ii. Empower your employ   ii. Educate the customer

(Left vertical label: PROBLEMS OLUTIONFIT)

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT:

The below described are the functional requirements of the proposed solution

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	Web-UI	Forms like UI to get the data from the user
FR-2	Predict the Credit Defaulters	Machine Learning Model is made to predict the Credit Defaulters
FR-3	Integration of Model and Web UI	Flask is used to integrate the Mode and Web UI

4.2 NON-FUNCTIONAL REQUIREMENT:

The below described are the non-functional requirements of the proposed solution

NFR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
NFR-1	Usability	The user interface that serves as a conduit between the user and the cloud-deployed DL Model
NFR-2	Security	The cloud-based paradigm should only be accessible to authorised users and should be inaccessible to attackers or terrorists
NFR-3	Reliability	The system would be dependable 98% of the time with >90% accuracy
NFR-4	Performance	Within seconds, the model predicts the credit defaulters with the provided data
NFR-5	Scalability	Can be scaled to an API and can be integrated into Mobile Banking Application

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 system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Data Flow Diagram
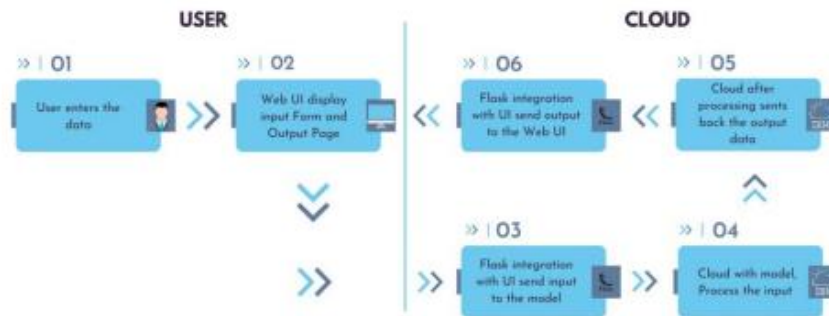
Data Flow Diagram - Level 0

Data Flow Diagram - Level 1
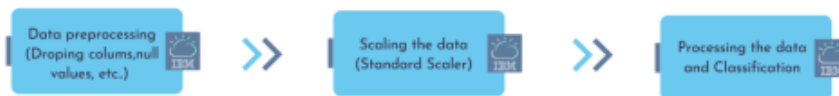
5.2 SOLUTION & TECHNICAL ARCHITECTURE

Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.
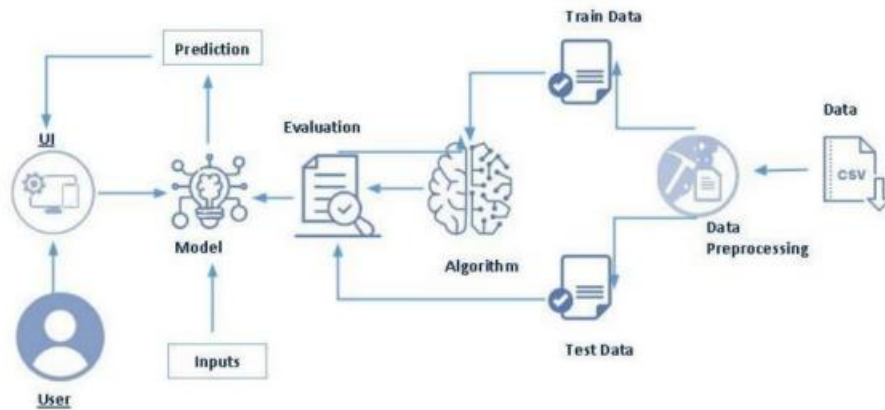
Explanation for the Architecture Diagram:

1. The acquired dataset is first used to train the model.
2. After pre-processing the dataset, the data will be divided into train and test sets.
3. A PKL file would then be saved for the model.
4. The model and website would be integrated using Flask, and a webpage would be made for the interaction.
5. The user would provide the input, which would then be processed to produce a prediction.
6. The result would be indicated as "Eligible" or "Not Eligible" when the forecast was made.
7. The consumer can find out eligibility even more easily by scaling this up as an API and integrating it with the mobile banking app

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table

2 **Example: Order processing during pandemics for offline mode**

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-orderprocessing-during-pandemics
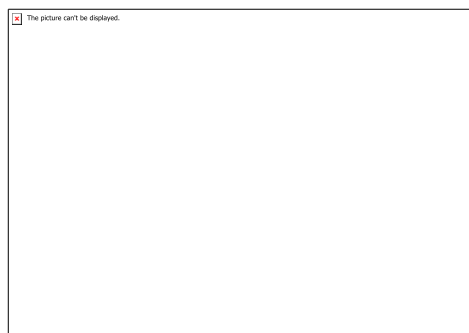
Guidelines:

1. Include all the processes (As an application logic / Technology Block)

2. Provide infrastructural demarcation (Local / Cloud)

3. Indicate external interfaces (third party API's etc.)

4. Indicate Data Storage components / services

5. Indicate interface to machine learning models (if applicable)

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web UI	HTML, CSS, JavaScript
2.	Application Logic-1 (Pre-Processing of Data Set)	Pre-processing is the important works to be done in the Dataset	Python, Pandas, NumPy, Scikit learn
3.	Application Logic-2 (Model Building)	Constructing a ML model to detect the Loan Defaulters.	Python, Pickle, Pandas, Scikit learn
4.	Application Logic-3 (Creating Web UI)	User for the User interaction	HTML, CSS, JavaScript
5.	Dataset	Dataset is collected	IBM
6.	Cloud Database	User for hosting the Web UI and also the exchange of Data	IBM Cloud
7.	File Storage	The dataset and source code are stored in files	IBM Block Storage or Other Storage Service or Local Filesystem
8.	Machine Learning Model	To find the Loan Defaulters	Python, Scikit learn

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Used for model building, package manager, code development, data analysis and evaluation	VS code, Python, Flask, Scikit – Learn, Pandas, NumPy, Matplotlib, Seaborn
2.	Buoyant	Can be trained for more accuracy	Python, Scikit - Learn

3.	Operability	Use a highly available server for deployment	IBM Cloud
4.	Execution	Web UI and Model	HTML, CSS, JS, Scikit - Learn
5.	Performance	Could yield results within seconds	Python, Flask and pickle

References:
https://c4model.com/
https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/
https://www.ibm.com/cloud/architecture
https://aws.amazon.com/architecture
https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d

5.3 USER STORIES

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web User)	Forms	USN-1	As a user, I can enter the data which the website asks me.	Submit the required data for prediction	High	Sprint-1
	Prediction	USN-2	As I have given the data into webpage, now the data can be predicted for the loan avail.	Pre-processing is done and data is scaled in Backend and sent to the model for prediction	High	Sprint-3
	Deployment of AI model in the cloud	USN-4	Model would be running on the cloud	I can access the model through the web address where I typed my data that's been set up on the IBM cloud.	Medium	Sprint-4
	Model Building	USN-5	I require an ML model that can categorise Credit defaulters	I can use the ML model to classify the Credit defaulters	High	Sprint-2
	User Interface building	USN-6	As a user, I need a medium to enter my data.	I can use the webpage which uses Flask at the backend to	Medium	Sprint-3

				integrate with the ML model created			

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Pre-processing	USN-1	Need for the data to be clean enough for Model Prediction	4	High	Moukthika Anvitha
Sprint-2	Web UI	USN-2	As a user, I would need a place to enter my data to predict my results	2	High	Moukthika
Sprint-2	Model Creation	USN-3	As the data is clean now, the data can be used to Train and Evaluate the results	3	Medium	Moukthika Varshitha
Sprint-3	Integration of Model and Web UI	USN-4	Using Flask, now we can integrate the Model with the input given by the user	1	Medium	Moukthika Anvitha Varshitha

6.2 SPRINT DELIVERY SCHEDULE

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	4	5 Days	5 Nov 2023	9 Nov 2023	6	9 Nov 2023
Sprint-2	2	3 Days	10 Nov 2023	12 Nov 2023		
Sprint-3	3	4 Days	13 Nov 2023	16 Nov 2023		
Sprint-4	1	2 Days	16 Nov 2023	17 Nov 2023		

Velocity:

Imagine we have a 14-day sprint duration, and the velocity of the team is 10 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)
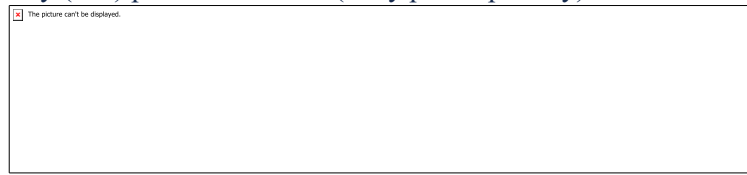
AV = sprint duration/velocity = 14/10 = 1.4

6.3 REPORTS FROM JIRA

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

Burndown Chart:
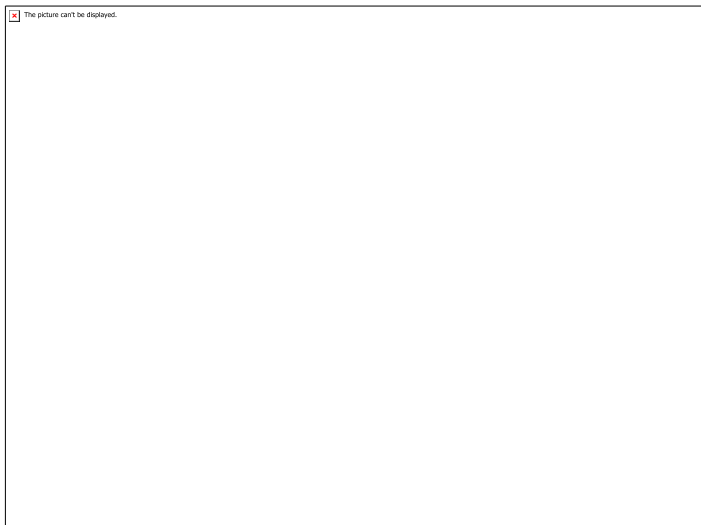
Backlog Chart:

Board Chart:

CODING & SOLUTIONING (Explain the features added in project along with code)
7.1 FEATURE 1:

```
#Importing the datset
df = pd.read_csv("train.csv")
df
df.head()
df.info()
df.shape
df=df.drop(columns=["Loan_ID"], axis=1)
```
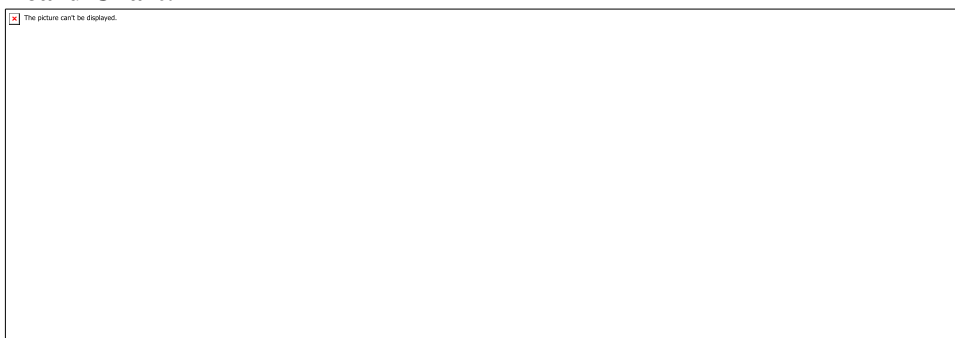
Checking for Null values

```
#Checking for Null values
df.isnull().sum()
```

```
#replacing null values
df['Self_Employed']=df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])
df['Married']=df['Married'].fillna(df['Married'].mode()[0])
df['Dependents']=df['Dependents'].fillna(df['Dependents'].mode()[0])
df['Self_Employed']=df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df['LoanAmount']=df['LoanAmount'].fillna(df['LoanAmount'].mode()[0])
df['Loan_Amount_Term']=df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0])
df['Credit_History']=df['Credit_History'].fillna(df['Credit_History'].mode()[0])
```

```
df.isnull().sum()
```

Data Visualization

```
#Data Visualization
sns.distplot(df.ApplicantIncome)
sns.countplot(x = 'Property_Area', data = df)
sns.countplot(x = 'Gender', data = df)
sns.countplot(x = 'Education', data = df)
sns.countplot(x = 'Self_Employed', data = df)
sns.countplot(x = 'Married', data = df)
sns.histplot(df.LoanAmount)
sns.displot(df.CoapplicantIncome)
plt.pie(df.Property_Area.value_counts(),[0,0,0],labels=['Semi urban','Urban','Rural'])
sns.countplot(x ='Married', hue = "Gender", data = df)
sns.countplot(x ='LoanAmount', hue = "Property_Area", data = df)
sns.countplot(x ='Education', hue = "Self_Employed", data = df)
sns.countplot(x ='LoanAmount', hue = "Loan_Amount_Term", data = df)
plt.scatter(df.ApplicantIncome,df.LoanAmount)
```

```
sns.countplot(x ='Dependents', hue = "Gender", data = df)
sns.heatmap(df.corr(),annot=True)
plt.plot(df.LoanAmount,df.Loan_Amount_Term,df.ApplicantIncome)
df.plot.line()
df.hist()
plt.plot(df.LoanAmount,df.ApplicantIncome,df.CoapplicantIncome)
plt.plot(df.Loan_Amount_Term,df.ApplicantIncome,df.CoapplicantIncome)
```

Perform Encoding
```
#Perform Encoding
le=LabelEncoder()
df.Gender=le.fit_transform(df.Gender)
df.Married=le.fit_transform(df.Married)
df.Education=le.fit_transform(df.Education)
df.Self_Employed=le.fit_transform(df.Self_Employed)
df.Property_Area=le.fit_transform(df.Property_Area)
df.Loan_Status=le.fit_transform(df.Loan_Status)
df.Dependents=le.fit_transform(df.Dependents)
```

Split Independent and dependent variables
```
df.head()
x=df.iloc[:,:-1]
y=df.Loan_Status
x.head()
y.head()
```

Feature Scaling
```
#Feature Scaling
scaler = MaxAbsScaler()
x_sc=scaler.fit_transform(x)
x_sc
# Balancing the dataset
sns.countplot(x = 'Loan_Status', data = df)
rus=RandomUnderSampler(sampling_strategy=1)
x_res,y_res=rus.fit_resample(x,y)
ax=y_res.value_counts().plot.pie(autopct='%.2f')
_=ax.set_title("under-sampling")
```

Split Data into Train and Test
```
#Splitting into train and test data
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.3, random_state=10)
xtrain.shape,xtest.shape,ytrain.shape,ytest.shape
```

7.2 FEATURE 2

Steps involved in model building

o Import the model building Libraries

o Initializing the model

o Training and testing the model

o Evaluation of Model

o Save the Model

Model Building using Decision Tree:

```
dmodel=DecisionTreeClassifier(random_state=100)
dmodel.fit(x_res,y_res)


ypredd=dmodel.predict(xtest)
ypred2d=dmodel.predict(xtrain)
```

Model Building using Random Forest:

```
Rmodel=RandomForestClassifier(n_estimators=100,max_depth=12,max_features=3)
Rmodel.fit(x_res,y_res)


ypredR=Rmodel.predict(xtest)
ypred2R=Rmodel.predict(xtrain)
```

Model Building using KNN:

```
kmodel=KNeighborsClassifier()
kmodel.fit(x_res,y_res)


ypredk=kmodel.predict(xtest)
ypred2k=kmodel.predict(xtrain)
```

Model Building using XgBoost:

```
xmodel=XGBClassifier(eval_metric='mlogloss',n_estimators=100,random_state=100)
xmodel.fit(x_res,y_res)


ypredx=xmodel.predict(xtest)
ypred2x=xmodel.predict(xtrain)
```

Comparing the Models:

```
print("Decision Tree Model Testing Accuracy")
print(accuracy_score(ytest,ypredd))
```

```
print("Decision Tree Model Training Accuracy")
print(accuracy_score(ytrain,ypred2d))


print("Random Forest Model Testing Accuracy")
print(accuracy_score(ytest,ypredR))
print("Random Forest Model Training Accuracy")
print(accuracy_score(ytrain,ypred2R))


print("KNN Model Testing Accuracy")
print(accuracy_score(ytest,ypredk))
print("KNN Model Training Accuracy")
print(accuracy_score(ytrain,ypred2k))


print("Xgboost Model Testing Accuracy")
print(accuracy_score(ytest,ypredx))
print("Xgboost Model Training Accuracy")
print(accuracy_score(ytrain,ypred2x))
```

Evaluating the Performance of Model and Saving the Model:
```
## Random Forest Model is Selected


print("Random Forest Model Testing Accuracy")
print(accuracy_score(ytest,ypredR))
print("Random Forest Model Training Accuracy")
print(accuracy_score(ytrain,ypred2R))


y=Rmodel.predict([[1,1,1,1,0,4583.0,1508.0,128.0,360.0,1.0,0]])
print(y)
y1=Rmodel.predict([[1,0,0,0,0,5849,0.0,146.412162,360.0,1.0,2]])
print(y1)
y2=Rmodel.predict([[1,0,0,1,0,678,987,90,24,1.0,2]])
print(y2)


f1_score(ypredR,ytest,average='weighted')


pd.crosstab(ytest,ypredR)


print(classification_report(ypredR,ytest))


##Saving the model by using pickle function
pickle.dump(Rmodel,open('model.pkl','wb'))
pickle.dump(scaler,open('scale.pkl','wb'))
```

TESTING

8.1 TEST CASES

The picture can't be displayed.

The picture can't be displayed.

The picture can't be displayed.

8.2 USER ACCEPTANCE TESTING

Defect Analysis:

The picture can't be displayed.
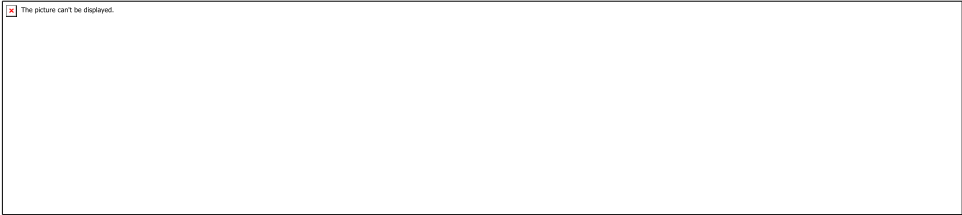
RESULTS

9.1 PERFORMANCE METRICS

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Classification Model:** Accuray Score - 93% & Classification Report - precision recall f1-score support 0 0.98 0.81 0.89 63	The picture can't be displayed.

| | | 1 0.91 0.99 0.95 122 accuracy 0.93 185 macro 0.95 0.90 0.92 185 avg
weighted
avg 0.93 0.93 0.93 185 | |

Screen shot
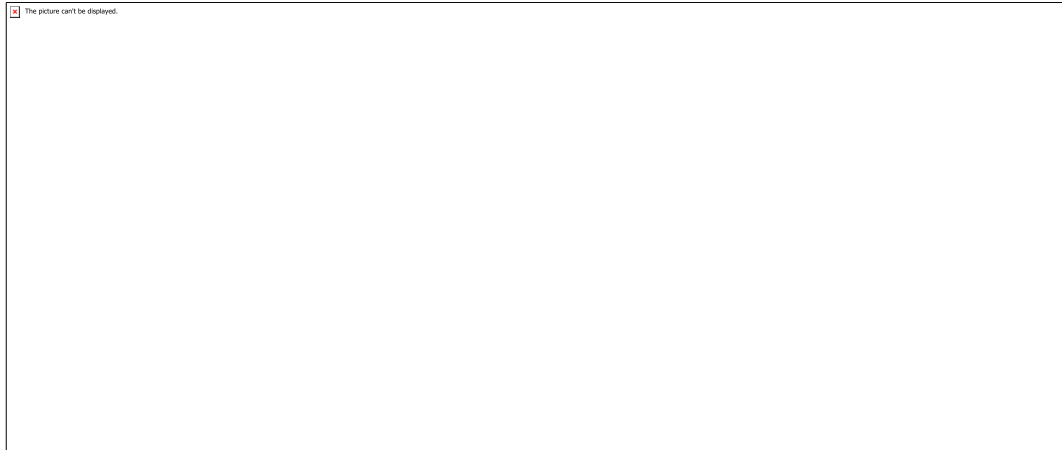
ADVANTAGES & DISADVANTAGES

Advantages:

1. Optimization of Loan Life Cycle
2. Digital Lending technology thrives on process speed
3. Easy capture of Applicant information
4. Quicker Decision Making
5. Consistency
6. Comfort across Devices
7. Perfect for first time borrowers
8. Compliance with Rules and Regulations
9. Power of Analysis

Disadvantages

1. Strict eligibility criteria
2. One of the major disadvantages of a bank loan is that banks can be cautious about lending to small businesses
3. Lengthy application process
4. Not suitable for ongoing expenses
5. Secured loans carry risk

CONCLUSION

There are several cases of computer glitches and errors in content, and the most significant weight of option is mounted in a machine-driven prediction system. Therefore, the so-called software system might be created in the future with more secure, reliable, and dynamic weight adjustment. In close to future, this module of prediction can be integrated with the module of machine-driven processing systems. The analysis starts with data cleaning and processing missing values, exploratory research, and finally, model building and evaluation of the model. The best accuracy on the public test set is when we get a higher accuracy score and other performance metrics which will be found out. This model can help to predict the approval of a bank loan or not for a candidate.

FUTURE SCOPE

The untapped loan industry, valued at over 20 lakh crores, presents a significant opportunity for smart lender platforms in the future. These lending platforms have a great deal of potential to offer cutting-edge innovations and disruptive technology for the lending and borrowing industry because they are technology-based. The younger generation of today is far more tech-savvy, well-informed, and believes that investing in a variety of life experiences is worthwhile. Platforms that offer instant approval and paperless loans with little time and personal interaction are predicted to grow significantly as consumerization becomes more commonplace. These platforms will enable more people to access financial services and will also generate higher returns for lenders, which will incentivize them.

APPENDIX

SOURCE CODE AND GITHUB

App.py

```
# save this as app.py
from flask import Flask, request, render_template
import pickle
import numpy as np
from markupsafe import escape


app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))
scale = pickle.load(open('scale.pkl','rb'))


@app.route('/')
def home():
    return render_template("index.html")
```

```python
@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method ==  'POST':
        gender = request.form['gender']
        married = request.form['married']
        dependents = request.form['dependents']
        education = request.form['education']
        employed = request.form['employed']
        credit = float(request.form['credit'])
        proparea = request.form['proparea']
        ApplicantIncome=float(request.form['ApplicantIncome'])
        CoapplicantIncome=float(request.form['CoapplicantIncome'])
        LoanAmount=float(request.form['LoanAmount'])
        Loan_Amount_Term=float(request.form['loan_Amount_Term'])

        # gender
        if (gender == "Male"):
            male=1
        else:
            male=0


        # married
        if(married=="Yes"):
            married_yes = 1
        else:
            married_yes=0


        # dependents
        if(dependents=='3+'):
            dependents = 3


        # education
        if (education=="Not Graduate"):
            not_graduate=1
        else:
            not_graduate=0


        # employed
        if (employed == "Yes"):
            employed_yes=1
        else:
            employed_yes=0
```

```python
        # property area


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



        features = [credit, male, married_yes, dependents, not_graduate,
employed_yes, proparea, ApplicantIncome,
CoapplicantIncome,LoanAmount,Loan_Amount_Term ]
        con_features = [np.array(features)]
        scale_features = scale.fit_transform(con_features)
        prediction = model.predict(scale_features)

        # print(prediction)
        print(prediction)
        if prediction==0:
            return render_template('approve.html',prediction_text ='Congratulations!
You are eligible for loan')
        else:
            return render_template('reject.html',prediction_text ='Sorry You are not
eligible for loan')
    else:
        return render_template("prediction.html")
if __name__ == "__main__":
    app.run(debug=True)
```

GIT PROFILE LINK: https://github.com/smartinternz02/SI-GuidedProject-613302-1698847818
PROJECT DEMO LINK:

https://drive.google.com/drive/folders/1gMaOYjWd2b0zCjOJjAoEi5PL6ewVMmpP?usp=drive_lin
k