



# Tech Saksham

## Capstone Project Report

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING  
FUNDAMENTALS

**“An End-to-End Data Science Project  
with ChatGPT”**

**“UNIVERSITY COLLEGE OF ENGINEERING (BIT  
CAMPUS) TIRUCHIRAPALLI”**

NM ID	NAME
Au810020127301	JOHNARIVUNIDHI

Trainer Name

Ramar Bose

Sr. AI Master Trainer

## ABSTRACT

This succinct end-to-end data science with ChatGPT project revolves around predicting loan default using a loan dataset from a financial institution. It entails data preprocessing, exploratory data analysis, and feature engineering to prepare the dataset for modeling. Leveraging machine learning algorithms like logistic regression, decision trees, random forests, and gradient boosting, predictive models are developed to forecast the likelihood of loan default. Feature importance analysis guides the identification of key predictors. Rigorous model evaluation ensures reliability and generalization. Ultimately, the best-performing model is deployed for real-time predictions, aiding financial institutions in proactive risk management and fostering a stable lending environment.

## INDEX

Sr. No.	Table of Contents	Page No.
1	Chapter 1: Introduction	4
2	Chapter 2: Services and Tools Required	6
3	Chapter 3: Project Architecture	7
4	Chapter 4: Modeling and Project Outcome	9
5	Conclusion	18
6	Future Scope	19
7	References	20
8	Links	21

# CHAPTER 1

## INTRODUCTION

### 1.1 Problem Statement

The goal of this project is to develop a comprehensive loan approval system using machine learning techniques and natural language processing (NLP) capabilities of ChatGPT. Leveraging a dataset of past loan applications, the project aims to build a predictive model that can assess the creditworthiness of new applicants based on their financial history and personal information. Additionally, integrating ChatGPT into the system will enable the automation of customer interactions, allowing for a more seamless and efficient loan application process. By combining advanced analytics with conversational AI, the project seeks to improve the accuracy and speed of loan approvals while enhancing the user experience for both applicants and loan officers.

### 1.2 Proposed Solution

For an end-to-end data science project utilizing ChatGPT with a loan dataset, the proposed solution involves several key steps. First, comprehensive data preprocessing is necessary to clean and prepare the loan dataset, including handling missing values and outliers. Next, feature engineering can help extract relevant information from the data to improve model performance. Then, a machine learning model, such as logistic regression or random forest, can be trained to predict loan approval or rejection based on historical data. Integration of ChatGPT allows for a conversational interface where users can inquire about loan eligibility criteria, receive personalized recommendations, or seek assistance with the loan application process. Finally, thorough testing and evaluation ensure the model's accuracy and effectiveness in real-world scenarios.

### 1.3 Feature

- **Data Gathering:** Collect loan dataset with borrower information.
- **Model Training:** Train ChatGPT on the loan data to understand queries.
- **User Interaction:** Allow users to ask questions or seek advice about loans.

- **Response Generation:** Generate informative responses based on loan dataset and user queries.

## 1.4 Advantages

- Risk Reduction: Predicting loan defaults beforehand helps minimize financial risks for lenders.
- Efficient Decision-Making: Data-driven insights enable smarter choices in loan approvals, terms, and rates.
- Cost Savings: Early identification of defaults saves money on collection efforts and legal actions.
- Personalized Service: Tailoring loan offerings to individual profiles enhances customer satisfaction.
- Competitive Edge: Data-driven strategies keep lenders ahead, ensuring profitability and market leadership.

## 1.5 Scope

The scope of an end-to-end data project integrating ChatGPT with a loan dataset is multifaceted. Firstly, leveraging historical loan data, the project aims to develop predictive models for assessing creditworthiness and risk analysis. ChatGPT will be integrated to enhance customer interaction and support throughout the loan application process, providing personalized assistance, answering inquiries, and offering guidance tailored to individual needs. Additionally, natural language processing capabilities will facilitate sentiment analysis of customer interactions, enabling real-time monitoring of customer satisfaction and feedback. Overall, the project endeavors to streamline the loan application journey, improve customer experience, and optimize lending decisions through the synergy of data analytics and AI-driven conversational interfaces.

## CHAPTER 2

# SERVICES AND TOOLS REQUIRED

### 2.1 Services Used

- **Data Collection:** Gather loan dataset including borrower information, loan details, and repayment history.
- **Data Preprocessing:** Clean, format, and preprocess the dataset to ensure consistency and remove noise.
- **Model Training:** Utilize ChatGPT to train a conversational AI model on the loan dataset to understand queries and provide responses.
- **Integration:** Integrate ChatGPT into the loan application system to provide end-to-end conversational support for loan inquiries and assistance.
- **Evaluation and Monitoring:** Continuously evaluate the performance of the system and monitor interactions to ensure accuracy and effectiveness in addressing user queries.

### 2.2 Tools and Software used

#### Tools:

- **Data Collection Tools:**
  - Web scraping tools (e.g., BeautifulSoup, Scrapy)
  - APIs for accessing financial data (e.g., Alpha Vantage, Quandl)
  - Data integration platforms (e.g., Talend, Informatica)
- **Data Preprocessing Tools:**
  - Data cleaning libraries (e.g., pandas, dplyr)
  - Data transformation tools (e.g., Trifacta, Alteryx)
  - Missing data imputation techniques (e.g., fancyimpute, scikit-learn)
- **Exploratory Data Analysis (EDA) Tools:**
  - Visualization libraries (e.g., Matplotlib, Seaborn, Plotly)
  - Statistical analysis tools (e.g., RStudio, Jupyter Notebooks)
  - Interactive dashboard platforms (e.g., Tableau, Power BI)

- **Feature Engineering Tools:**

Feature engineering libraries (e.g., scikit-learn, Featuretools)

Automated feature engineering platforms (e.g., DataRobot, H2O.ai)

- **Machine Learning Tools:**

Machine learning libraries (e.g., scikit-learn, TensorFlow, PyTorch)

Cloud-based machine learning platforms (e.g., AWS SageMaker, Google AI Platform, Microsoft Azure Machine Learning)

- **Model Deployment and Monitoring Tools:**

Model deployment frameworks (e.g., Flask, FastAPI)

Model monitoring platforms (e.g., MLflow, Kubeflow)

### Software Requirements:

- **Python** for scripting and data manipulation.
- **TensorFlow** or PyTorch for deep learning.
- **ChatGPT** for natural language processing.
- **Pandas** for data manipulation.
- **Flask** or Django for web deployment.

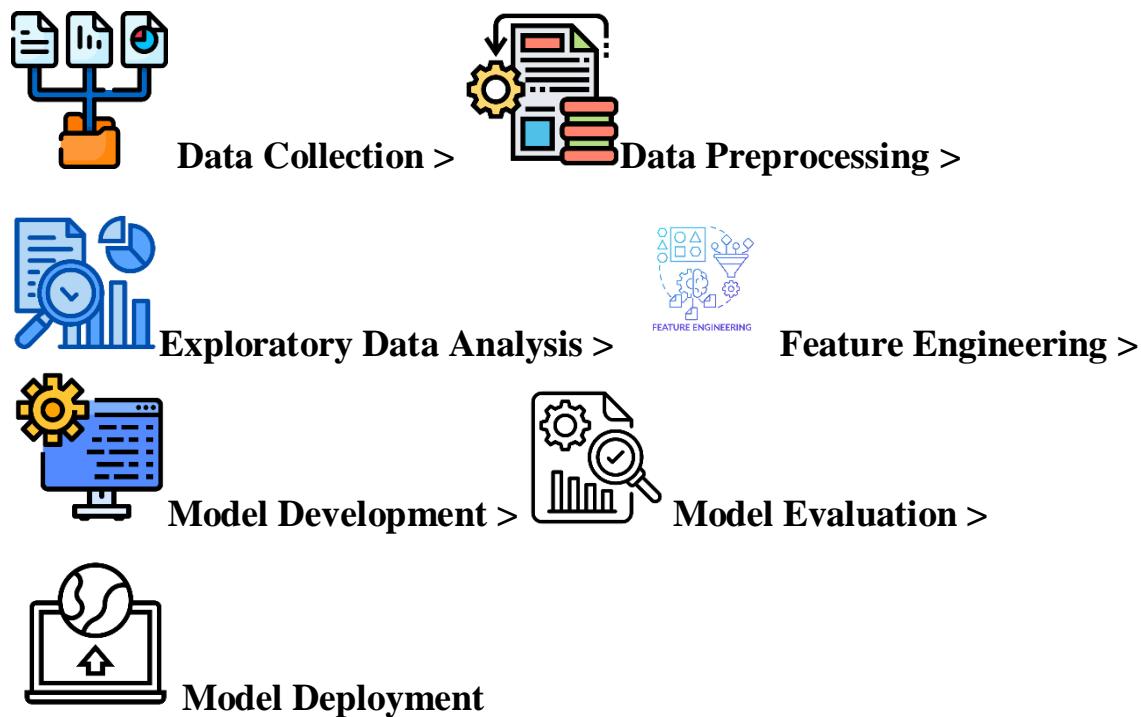
## CHAPTER 3

# PROJECT ARCHITECTURE

### 3.1 Architecture:

**End to end data science project with ChatGPT:**

**Flow Diagram:**



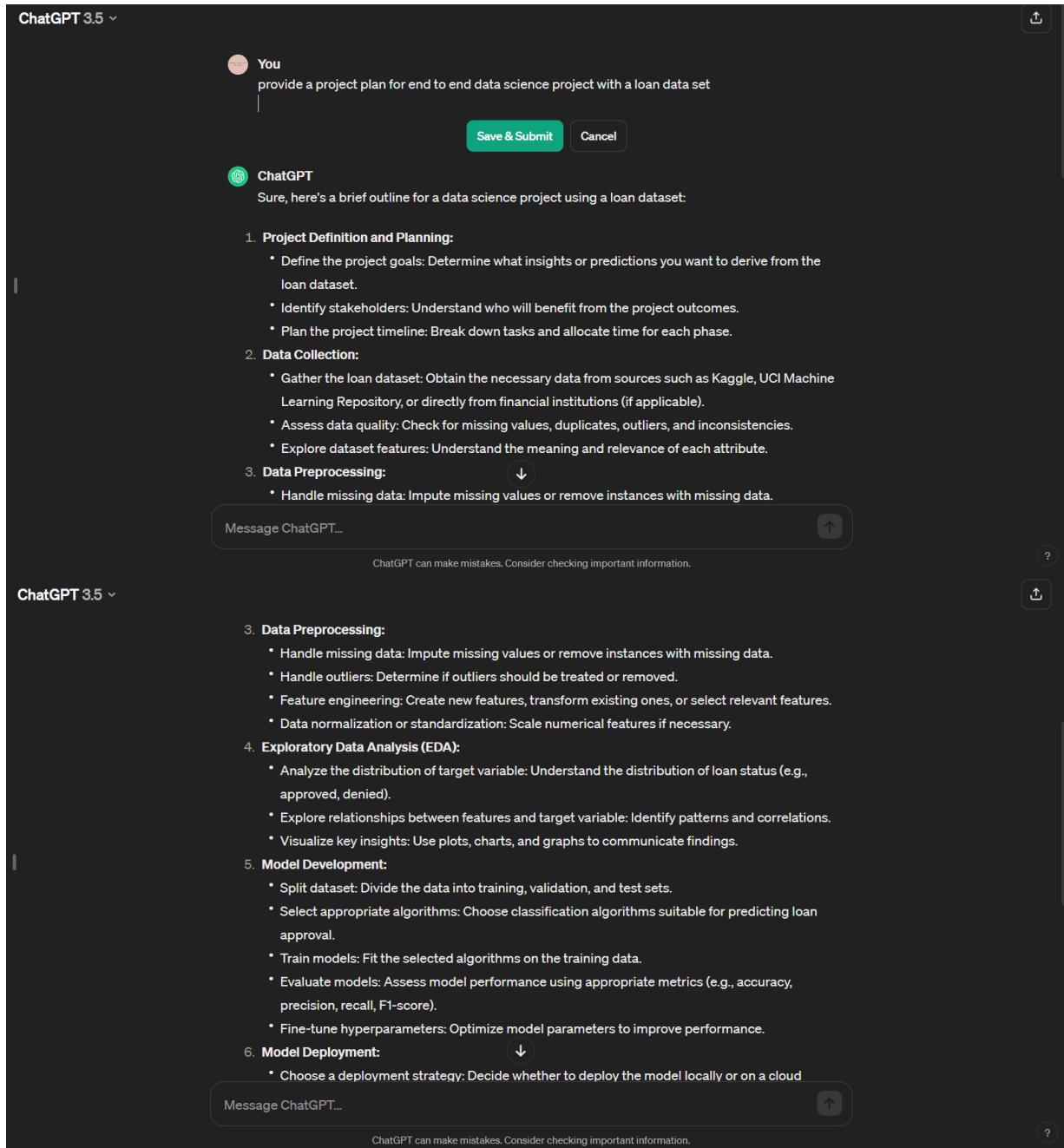
1. **Data Collection:** Gather loan data from various sources such as databases, APIs, and files.
2. **Data Preprocessing:** Cleanse, transform, and prepare the dataset for analysis by handling missing values, outliers, and data inconsistencies.
3. **Exploratory Data Analysis (EDA):** Explore the dataset to understand its characteristics, patterns, and relationships.
4. **Feature Engineering:** Extract and create relevant features from the dataset to enhance predictive modeling.
5. **Model Development:** Build and train machine learning models using algorithms like logistic regression, decision trees, and XGBoost for loan default prediction.

6. **Model Evaluation:** Assess the performance of the models using evaluation metrics like accuracy, precision, recall, and F1-score.
7. **Model Deployment:** Deploy the best-performing model into production for real-time predictions on new loan data.

## CHAPTER 4 (code)

# MODELING AND PROJECT OUTCOME

Asking chat GPT which is a AI tool to provide the project plan



ChatGPT 3.5

You

provide a project plan for end to end data science project with a loan data set

Save & Submit Cancel

ChatGPT

Sure, here's a brief outline for a data science project using a loan dataset:

- Project Definition and Planning:**
  - Define the project goals: Determine what insights or predictions you want to derive from the loan dataset.
  - Identify stakeholders: Understand who will benefit from the project outcomes.
  - Plan the project timeline: Break down tasks and allocate time for each phase.
- Data Collection:**
  - Gather the loan dataset: Obtain the necessary data from sources such as Kaggle, UCI Machine Learning Repository, or directly from financial institutions (if applicable).
  - Assess data quality: Check for missing values, duplicates, outliers, and inconsistencies.
  - Explore dataset features: Understand the meaning and relevance of each attribute.
- Data Preprocessing:**
  - Handle missing data: Impute missing values or remove instances with missing data.

Message ChatGPT...

ChatGPT can make mistakes. Consider checking important information.

ChatGPT 3.5

- Data Preprocessing:**
  - Handle missing data: Impute missing values or remove instances with missing data.
  - Handle outliers: Determine if outliers should be treated or removed.
  - Feature engineering: Create new features, transform existing ones, or select relevant features.
  - Data normalization or standardization: Scale numerical features if necessary.
- Exploratory Data Analysis (EDA):**
  - Analyze the distribution of target variable: Understand the distribution of loan status (e.g., approved, denied).
  - Explore relationships between features and target variable: Identify patterns and correlations.
  - Visualize key insights: Use plots, charts, and graphs to communicate findings.
- Model Development:**
  - Split dataset: Divide the data into training, validation, and test sets.
  - Select appropriate algorithms: Choose classification algorithms suitable for predicting loan approval.
  - Train models: Fit the selected algorithms on the training data.
  - Evaluate models: Assess model performance using appropriate metrics (e.g., accuracy, precision, recall, F1-score).
  - Fine-tune hyperparameters: Optimize model parameters to improve performance.
- Model Deployment:**
  - Choose a deployment strategy: Decide whether to deploy the model locally or on a cloud

Message ChatGPT...

ChatGPT can make mistakes. Consider checking important information.

ChatGPT 3.5 ▾

6. **Model Deployment:**

- Choose a deployment strategy: Decide whether to deploy the model locally or on a cloud platform.
- Develop APIs or interfaces: Create endpoints for accessing the model predictions.
- Test deployment: Ensure the deployed model behaves as expected and handles edge cases gracefully.

7. **Monitoring and Maintenance:**

- Set up monitoring: Implement systems to track model performance and drift over time.
- Schedule regular updates: Plan for model retraining to keep it relevant with changing data patterns.
- Address feedback: Incorporate feedback from users and stakeholders to improve model effectiveness.

8. **Documentation and Reporting:**

- Document the entire process: Write detailed documentation covering data sources, preprocessing steps, model development, and deployment.
- Prepare a final report: Summarize key findings, insights, and recommendations in a concise report for stakeholders.

Remember to adapt this outline based on the specific requirements of your project and the characteristics of the loan dataset you're working with.

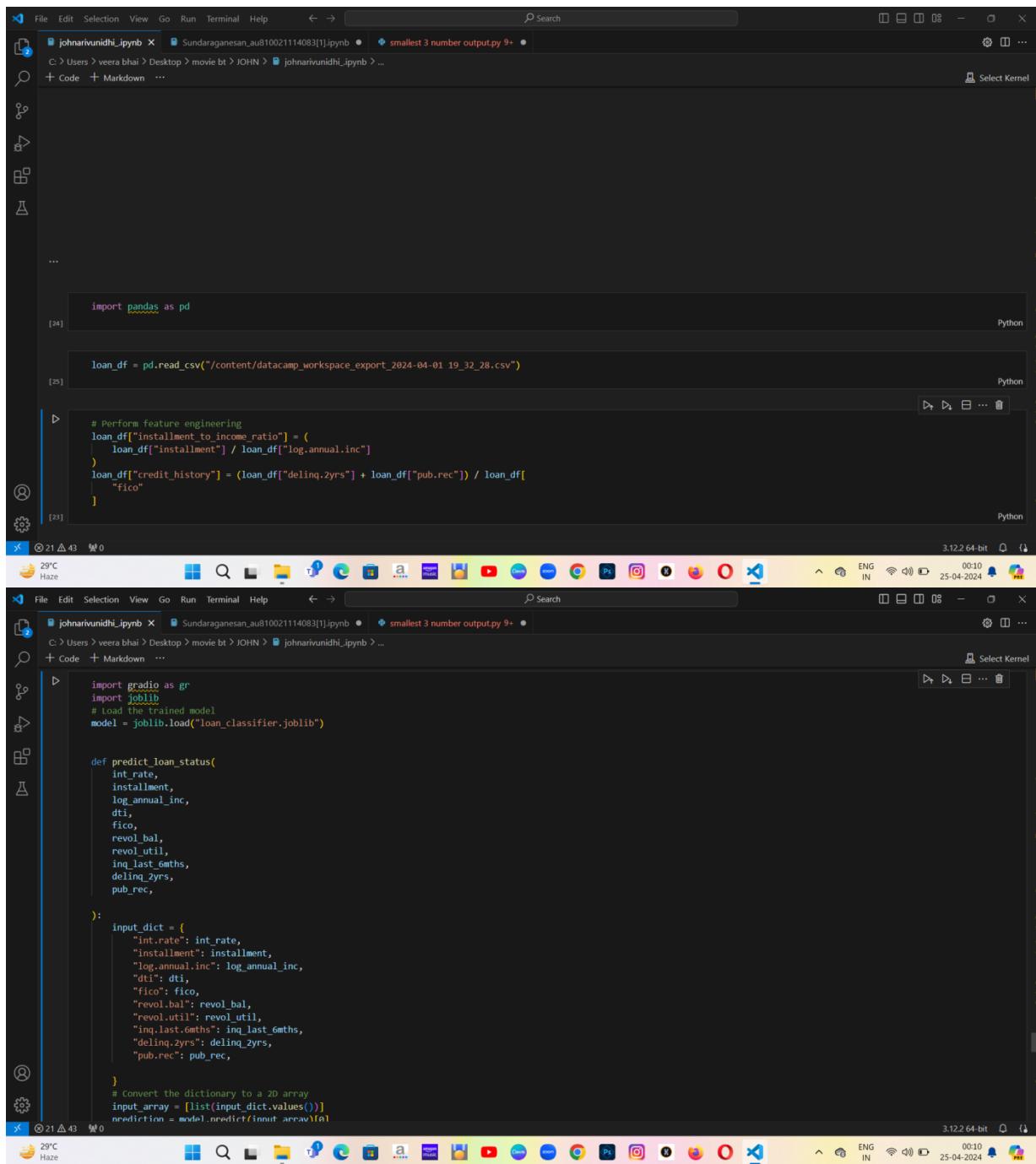
Message ChatGPT... ?

ChatGPT can make mistakes. Consider checking important information.

The asked the ChatGPT “to provide the necessary codes for the project. The codes are implemented and the output is received.

Code:

1. Data Collection
2. Data Preprocessing
3. Exploratory Data Analysis (EDA)



```

[24]: import pandas as pd
[25]: loan_df = pd.read_csv("/content/datacamp_workspace_export_2024-04-01_19_32_28.csv")
[26]: # Perform feature engineering
loan_df["installment_to_income_ratio"] = (
    loan_df["installment"] / loan_df["log.annual.inc"]
)
loan_df["credit_history"] = (loan_df["delinq.2yrs"] + loan_df["pub.rec"]) / loan_df[
    "fico"
]
[27]:

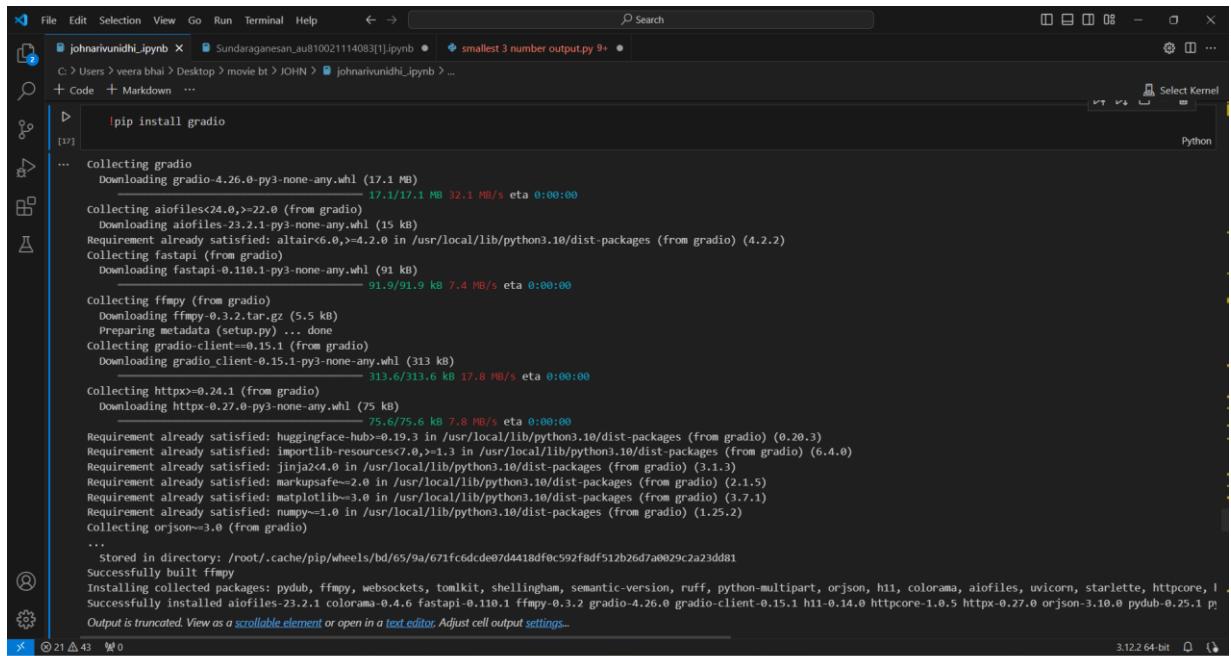
```

```

[28]: import gradio as gr
import joblib
# Load the trained model
model = joblib.load("loan_classifier.joblib")

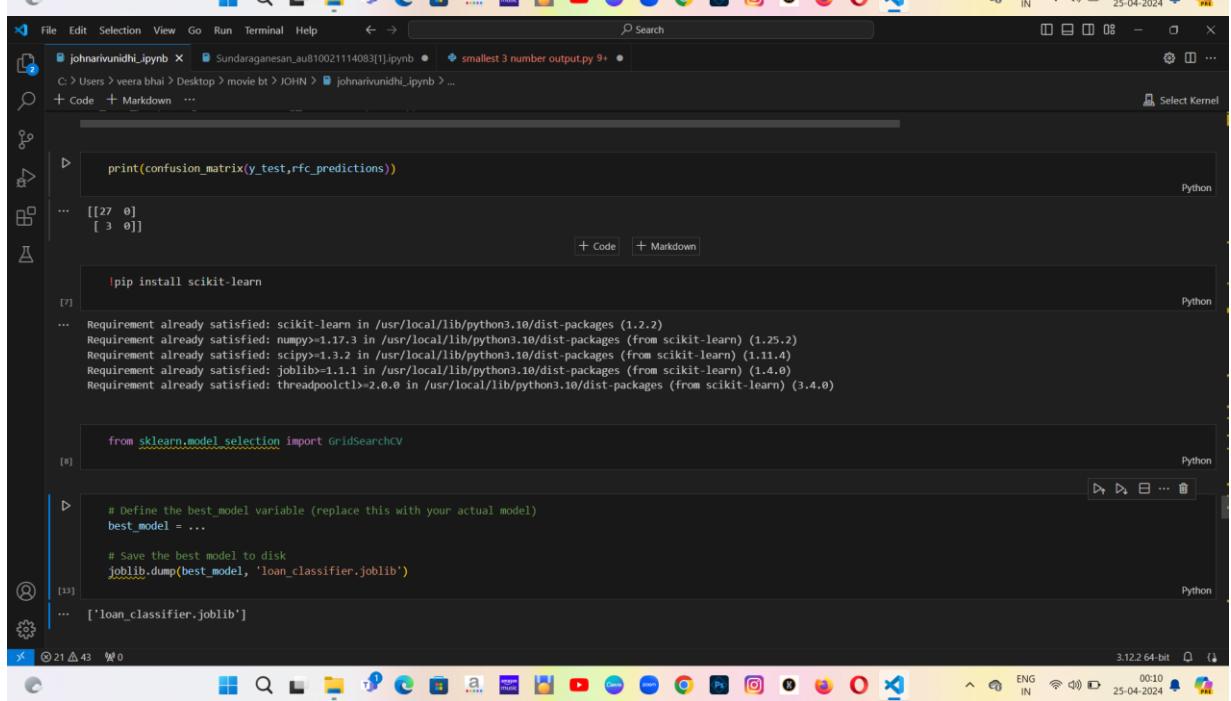
def predict_loan_status(
    int_rate,
    installment,
    log_annual_inc,
    dti,
    fico,
    revol_bal,
    revol_util,
    inq_last_6mths,
    delinq_2yrs,
    pub_rec,
):
    input_dict = {
        "int.rate": int_rate,
        "installment": installment,
        "log.annual.inc": log_annual_inc,
        "dti": dti,
        "fico": fico,
        "revol.bal": revol_bal,
        "revol.util": revol_util,
        "inq.last.6mths": inq_last_6mths,
        "delinq.2yrs": delinq_2yrs,
        "pub.rec": pub_rec,
    }
    # Convert the dictionary to a 2D array
    input_array = [list(input_dict.values())]
    prediction = model.predict(input_array)[0]

```



```
|pip install gradio
Collecting gradio
  Downloading gradio-4.26.0-py3-none-any.whl (17.1 MB)
    17.1/17.1 MB 32.1 MB/s eta 0:00:00
Collecting aiofiles<24.0,>=22.0 (from gradio)
  Downloading aiofiles-23.2.1-py3-none-any.whl (15 kB)
Requirement already satisfied: altair<6.0,>=4.2.0 in /usr/local/lib/python3.10/dist-packages (from gradio) (4.2.2)
Collecting fastapi (from gradio)
  Downloading fastapi-0.110.1-py3-none-any.whl (91 kB)
    91.9/91.9 kB 7.4 MB/s eta 0:00:00
Collecting ffmpeg (from gradio)
  Downloading ffmpeg-0.3.2.tar.gz (5.5 kB)
    Preparing metadata (setup.py) ... done
Collecting gradio-client==0.15.1 (from gradio)
  Downloading gradio_client-0.15.1-py3-none-any.whl (313 kB)
    313.6/313.6 kB 17.8 MB/s eta 0:00:00
Collecting httpx>=0.24.1 (from gradio)
  Downloading httpx-0.27.0-py3-none-any.whl (75 kB)
    75.6/75.6 kB 7.8 MB/s eta 0:00:00
Requirement already satisfied: huggingface-hub>=0.19.3 in /usr/local/lib/python3.10/dist-packages (from gradio) (0.20.3)
Requirement already satisfied: importlib-resources<7.0,>=1.3 in /usr/local/lib/python3.10/dist-packages (from gradio) (6.4.0)
Requirement already satisfied: jinjazl<4.0 in /usr/local/lib/python3.10/dist-packages (from gradio) (3.1.3)
Requirement already satisfied: markupsafe<2.0 in /usr/local/lib/python3.10/dist-packages (from gradio) (2.1.5)
Requirement already satisfied: matplotlib<3.0 in /usr/local/lib/python3.10/dist-packages (from gradio) (3.7.1)
Requirement already satisfied: numpy=<1.0 in /usr/local/lib/python3.10/dist-packages (from gradio) (1.25.2)
Requirement already satisfied: orjson<=3.0 (from gradio)
  ...
  Stored in directory: /root/.cache/pip/wheels/bd/65/9a/671fc6dcde07d4418df0c592f8df512b26d7a0029c2a23dd81
Successfully built ffmpeg
Installing collected packages: pydub, ffmpeg, websockets, tomkit, shellingham, semantic-version, ruff, python-multipart, orjson, h11, colorama, aiofiles, uvicorn, starlette, httpcore, i
Successfully installed aiofiles-23.2.1 colorama-0.4.6 fastapi-0.110.1 ffmpeg-0.3.2 gradio-4.26.0 gradio-client-0.15.1 h11-0.14.0 httpcore-1.0.5 httpx-0.27.0 orjson-3.10.0 pydub-0.25.1 p
Output was truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

3.12.2 64-bit ENG IN 0010 25-04-2024



```
print(confusion_matrix(y_test, rfc_predictions))
[[27  0]
 [ 3  0]]
```

+ Code + Markdown

```
|pip install scikit-learn
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.25.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib<1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.4.0)

from sklearn.model_selection import GridSearchCV
```

# Define the best\_model variable (replace this with your actual model)  
best\_model = ...  
# Save the best model to disk  
joblib.dump(best\_model, 'loan\_classifier.joblib')

['loan\_classifier.joblib']

3.12.2 64-bit ENG IN 0010 25-04-2024

The screenshot shows two Jupyter Notebook sessions running on a Windows 10 desktop. Both sessions have tabs for 'johnarivunidhi.ipynb' and 'smallest 3 number output.py'. The top session is in a Python 3 kernel and displays a confusion matrix for a classification model. The bottom session is also in a Python 3 kernel and shows the creation of a Random Forest classifier and its evaluation.

**Session 1 (Python 3):**

```
print(confusion_matrix(y_test, rfc_predictions))

[[27  0]
 [ 3  0]]
```

**Session 2 (Python 3):**

```
# pip install scikit-learn

Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.25.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib<1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.4.0)
```

**System Status Bar:**

3.12.64-bit ENG IN 00:10 25-04-2024



The screenshot shows a Jupyter Notebook interface with two tabs open: 'johnarivunidhi.ipynb' and 'smallest 3 number output.py'. The code cell contains the following Python script:

```
from sklearn.tree import DecisionTreeClassifier
# Instantiating Decision Tree model (basically creating a decision tree object)
dtree = DecisionTreeClassifier()
# TRAINING or fitting the model on training data
dtree.fit(X_train,y_train)

... DecisionTreeClassifier()

dtree_predictions = dtree.predict(X_test)

from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_test,dtree_predictions))

... precision    recall   f1-score   support
          ...
          0       0.89      0.93      0.91       27
          1       0.00      0.00      0.00        3

accuracy                           0.83      30
macro avg       0.45      0.46      0.45      30
weighted avg    0.80      0.83      0.82      30

print(confusion_matrix(y_test,dtree_predictions))
... [[25  2]
 [ 3  0]]
```

The output cell shows the results of the classification report and the confusion matrix.

At the bottom, the system tray shows the date as 25-04-2024 and the time as 00:10.

The screenshot shows two Jupyter Notebook sessions side-by-side. Both sessions are running on Python 3.12.2 (64-bit) and have the same kernel selected.

**Session 1 (Left):**

- Kernel: Python
- Code cells:
  - ```
x = final_data.drop('not.fully.paid',axis=1)
```
  - ```
y=final_data['not.fully.paid']
```
  - ```
from sklearn.model_selection import train_test_split
```
  - ```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
```
  - ```
from sklearn.tree import DecisionTreeClassifier
```
  - ```
# Instantiating Decision Tree model (basically creating a decision tree object)
```
  - ```
dtree = DecisionTreeClassifier()
```
  - ```
# Training or Fitting the model on training data
```
  - ```
dtree.fit(X_train,y_train)
```
- Output:
  - A dropdown menu shows the class labels: `DecisionTreeClassifier` and `DecisionTreeClassifier()`.

**Session 2 (Right):**

- Kernel: Python
- Code cells:
  - ```
import pandas as pd
```
  - ```
data = pd.read_csv('/content/datacamp_workspace_export_2024-04-01_19_32_28.csv')
```
  - ```
data['new_feature'] = data['dti'] * data['fico']
```
  - ```
data.head()
```
- Output:
  - Prints the first 5 rows of the DataFrame:

|   | credit.policy | purpose            | int.rate | installment | log.annual.inc | dti   | fico | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | pub.rec | not.fully.paid | new_feature |
|---|---------------|--------------------|----------|-------------|----------------|-------|------|-------------------|-----------|------------|----------------|-------------|---------|----------------|-------------|
| 0 | 1             | debt_consolidation | 0.1189   | 829.10      | 11.350407      | 19.48 | 737  | 5639.958333       | 28854     | 52.1       | 0              | 0           | 0       | 0              | 14356.76    |
| 1 | 1             | credit_card        | 0.1071   | 228.22      | 11.082143      | 14.29 | 707  | 2760.000000       | 33623     | 76.7       | 0              | 0           | 0       | 0              | 10103.03    |
| 2 | 1             | debt_consolidation | 0.1357   | 366.86      | 10.373491      | 11.63 | 682  | 4710.000000       | 3511      | 25.6       | 1              | 0           | 0       | 0              | 7931.66     |
| 3 | 1             | debt_consolidation | 0.1008   | 162.34      | 11.350407      | 8.10  | 712  | 2699.958333       | 33667     | 73.2       | 1              | 0           | 0       | 0              | 5767.20     |
| 4 | 1             | credit_card        | 0.1426   | 102.92      | 11.299732      | 14.97 | 667  | 4066.000000       | 4740      | 39.5       | 0              | 1           | 0       | 0              | 9984.99     |
- ```
x = final_data.drop('not.fully.paid',axis=1)
```
- ```
y=final_data['not.fully.paid']
```

Sundaraganesan\_aud10021114083[1].ipynb • smallest 3 number output.py 9+

```

File Edit Selection View Go Run Terminal Help Search Select Kernel
C:\Users\veera bhai\Desktop>movie bt>JOHN>johnarvindhi.ipynb ...
+ Code + Markdown ...
15 purpose_educational 100 non-null bool
16 purpose_home_improvement 100 non-null bool
17 purpose_major_purchase 100 non-null bool
18 purpose_small_business 100 non-null bool
dtypes: bool(6), float64(6), int64(7)
memory usage: 10.9 KB

```

```

final_data.head()

```

|   | credit-policy | int.rate | installment | log.annual.inc | dti   | fico | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | pub.rec | not.fully.paid | purpose.credit_card | purpose_debt.consolids |
|---|---------------|----------|-------------|----------------|-------|------|-------------------|-----------|------------|----------------|-------------|---------|----------------|---------------------|------------------------|
| 0 | 1             | 0.1189   | 829.10      | 11350407       | 19.48 | 737  | 5639.958333       | 28854     | 52.1       | 0              | 0           | 0       | 0              | False               |                        |
| 1 | 1             | 0.1071   | 228.22      | 11082143       | 14.29 | 707  | 2760.000000       | 33623     | 76.7       | 0              | 0           | 0       | 0              | True                |                        |
| 2 | 1             | 0.1357   | 366.86      | 10373491       | 11.63 | 682  | 4710.000000       | 3511      | 25.6       | 1              | 0           | 0       | 0              | False               |                        |
| 3 | 1             | 0.1008   | 162.34      | 11350407       | 8.10  | 712  | 2699.958333       | 33667     | 73.2       | 1              | 0           | 0       | 0              | False               |                        |
| 4 | 1             | 0.1426   | 102.92      | 11299732       | 14.97 | 667  | 4066.000000       | 4740      | 39.5       | 0              | 1           | 0       | 0              | True                |                        |

```

import pandas as pd

```

```

data = pd.read_csv('/content/datcamp_workspace_export_2024-04-01_19_32_28.csv')

```

```

data['new_feature'] = data['dti'] * data['fico']

```

Python 3.12.2 64-bit 0010 25-04-2024 ENG IN

Sundaraganesan\_aud10021114083[1].ipynb • smallest 3 number output.py 9+

```

File Edit Selection View Go Run Terminal Help Search Select Kernel
C:\Users\veera bhai\Desktop>movie bt>JOHN>johnarvindhi.ipynb ...
+ Code + Markdown ...
15 purpose_educational 100 non-null float64
16 purpose_home_improvement 100 non-null float64
17 purpose_major_purchase 100 non-null float64
18 purpose_small_business 100 non-null float64
dtypes: float64(6), int64(7)
memory usage: 10.9 KB

```

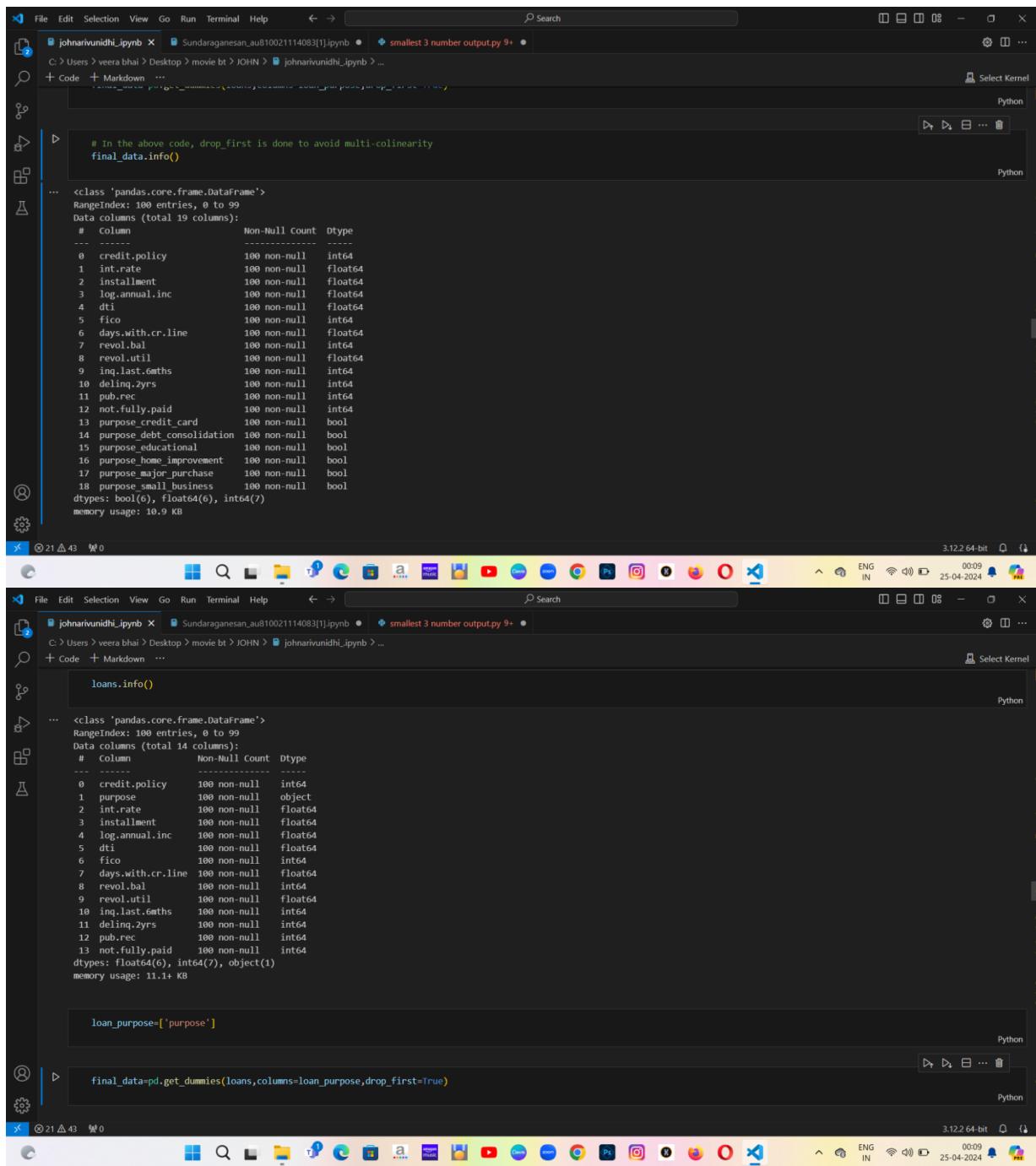
```

final_data.head()

```

|   | credit-policy | int.rate | installment | log.annual.inc | dti   | fico | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | pub.rec | not.fully.paid | purpose.credit_card | purpose_debt.consolids |
|---|---------------|----------|-------------|----------------|-------|------|-------------------|-----------|------------|----------------|-------------|---------|----------------|---------------------|------------------------|
| 0 | 1             | 0.1189   | 829.10      | 11350407       | 19.48 | 737  | 5639.958333       | 28854     | 52.1       | 0              | 0           | 0       | 0              | False               |                        |
| 1 | 1             | 0.1071   | 228.22      | 11082143       | 14.29 | 707  | 2760.000000       | 33623     | 76.7       | 0              | 0           | 0       | 0              | True                |                        |
| 2 | 1             | 0.1357   | 366.86      | 10373491       | 11.63 | 682  | 4710.000000       | 3511      | 25.6       | 1              | 0           | 0       | 0              | False               |                        |
| 3 | 1             | 0.1008   | 162.34      | 11350407       | 8.10  | 712  | 2699.958333       | 33667     | 73.2       | 1              | 0           | 0       | 0              | False               |                        |
| 4 | 1             | 0.1426   | 102.92      | 11299732       | 14.97 | 667  | 4066.000000       | 4740      | 39.5       | 0              | 1           | 0       | 0              | True                |                        |

Python 3.12.2 64-bit 0010 25-04-2024 ENG IN



The screenshot shows two separate Jupyter Notebook sessions running on a Windows 10 desktop. Both sessions are titled 'johnarivunidhi\_ipynb' and are executing Python code against datasets named 'final\_data' and 'loans'.

**Session 1 (Top):**

```
# In the above code, drop_first is done to avoid multi-collinearity
final_data.info()
```

| #  | Column                     | Non-Null Count | Dtype            |
|----|----------------------------|----------------|------------------|
| 0  | credit.policy              | 100            | non-null int64   |
| 1  | int.rate                   | 100            | non-null float64 |
| 2  | installment                | 100            | non-null float64 |
| 3  | log.annual.inc             | 100            | non-null float64 |
| 4  | dti                        | 100            | non-null float64 |
| 5  | fico                       | 100            | non-null int64   |
| 6  | days.with.cr.line          | 100            | non-null float64 |
| 7  | revol.bal                  | 100            | non-null int64   |
| 8  | revol.util                 | 100            | non-null float64 |
| 9  | inq.last.6mths             | 100            | non-null int64   |
| 10 | delinq.zyrs                | 100            | non-null int64   |
| 11 | pub.rec                    | 100            | non-null int64   |
| 12 | not.fully.paid             | 100            | non-null int64   |
| 13 | purpose_credit_card        | 100            | non-null bool    |
| 14 | purpose_debt_consolidation | 100            | non-null bool    |
| 15 | purpose_educational        | 100            | non-null bool    |
| 16 | purpose_home_improvement   | 100            | non-null bool    |
| 17 | purpose_major_purchase     | 100            | non-null bool    |
| 18 | purpose_small_business     | 100            | non-null bool    |

dtypes: bool(6), float64(6), int64(7)  
 memory usage: 16.9 KB

**Session 2 (Bottom):**

```
loans.info()
```

| #  | Column            | Non-Null Count | Dtype            |
|----|-------------------|----------------|------------------|
| 0  | credit.policy     | 100            | non-null int64   |
| 1  | purpose           | 100            | non-null object  |
| 2  | int.rate          | 100            | non-null float64 |
| 3  | installment       | 100            | non-null float64 |
| 4  | log.annual.inc    | 100            | non-null float64 |
| 5  | dti               | 100            | non-null float64 |
| 6  | fico              | 100            | non-null int64   |
| 7  | days.with.cr.line | 100            | non-null float64 |
| 8  | revol.bal         | 100            | non-null int64   |
| 9  | revol.util        | 100            | non-null float64 |
| 10 | inq.last.6mths    | 100            | non-null int64   |
| 11 | delinq.zyrs       | 100            | non-null int64   |
| 12 | pub.rec           | 100            | non-null int64   |
| 13 | not.fully.paid    | 100            | non-null int64   |

dtypes: float64(6), int64(7), object(1)  
 memory usage: 11.14 KB

```
loan_purpose=['purpose']
```

```
final_data=pd.get_dummies(loans,columns=loan_purpose,drop_first=True)
```

Sundaraganesan\_aus10021114083[1].ipynb · smallset 3 number output.py 9+

```

loans.info()

... <class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 14 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   credit.policy    100 non-null   int64  
 1   purpose        100 non-null   object  
 2   int.rate       100 non-null   float64 
 3   installment    100 non-null   float64 
 4   log.annual.inc 100 non-null   float64 
 5   dti            100 non-null   float64 
 6   fico           100 non-null   int64  
 7   days.with.cr.line 100 non-null   float64 
 8   revol.bal     100 non-null   int64  
 9   revol.util    100 non-null   float64 
 10  inq.last.6mths 100 non-null   int64  
 11  delinq.2yrs   100 non-null   int64  
 12  pub.rec      100 non-null   int64  
 13  not.fully.paid 100 non-null   int64  
dtypes: float64(6), int64(7), object(1)
memory usage: 11.1+ KB

```

loan\_purpose=['purpose']

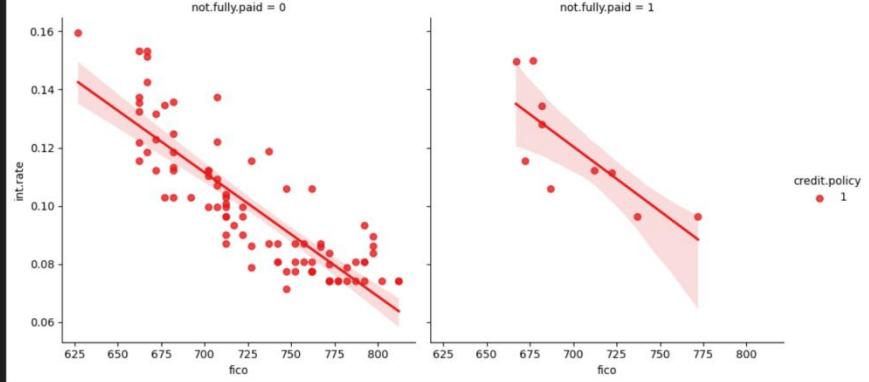
3.12.2 64-bit 00:09 25-04-2024 ENG IN

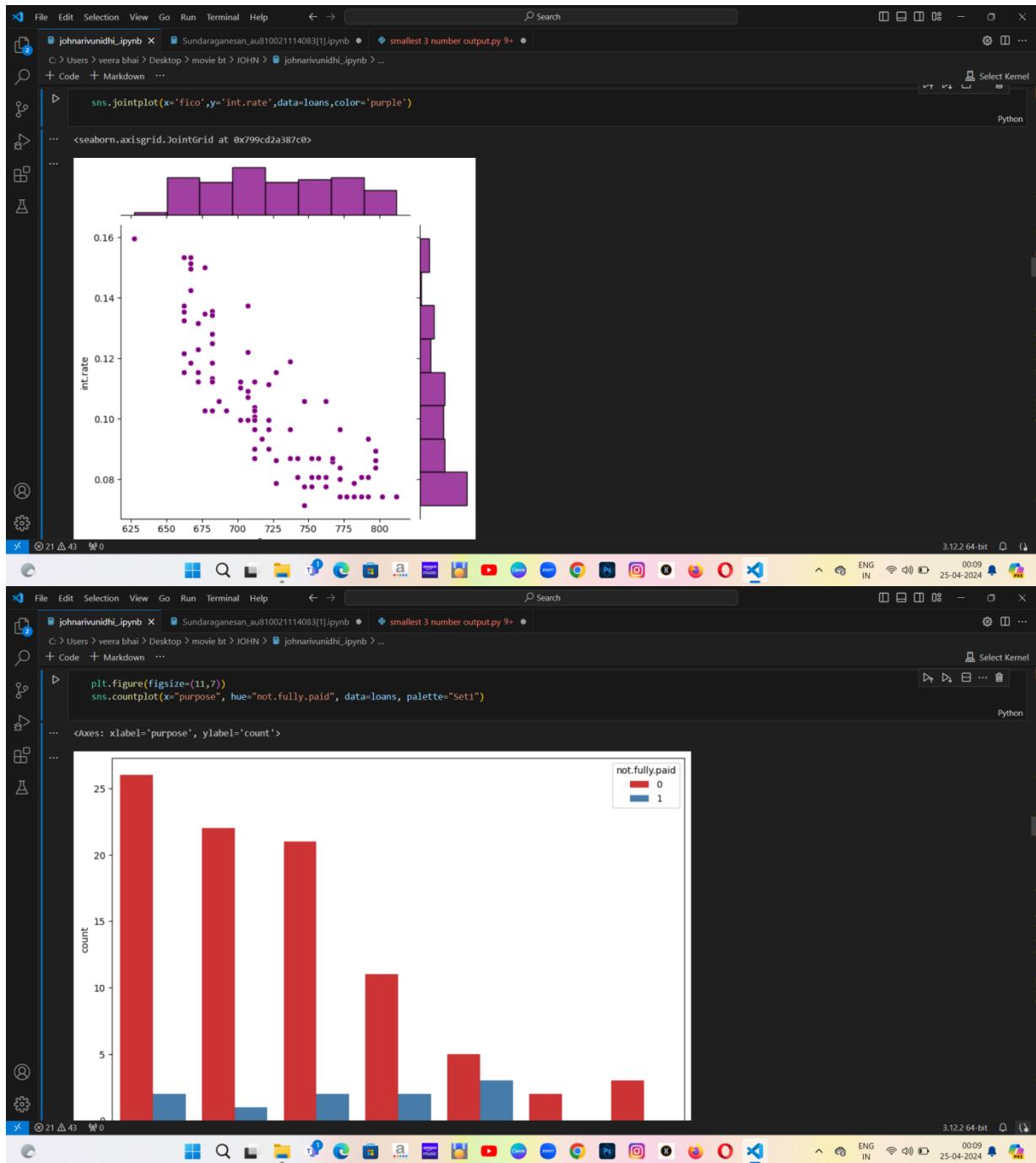
File Edit Selection View Go Run Terminal Help Search Select Kernel Python

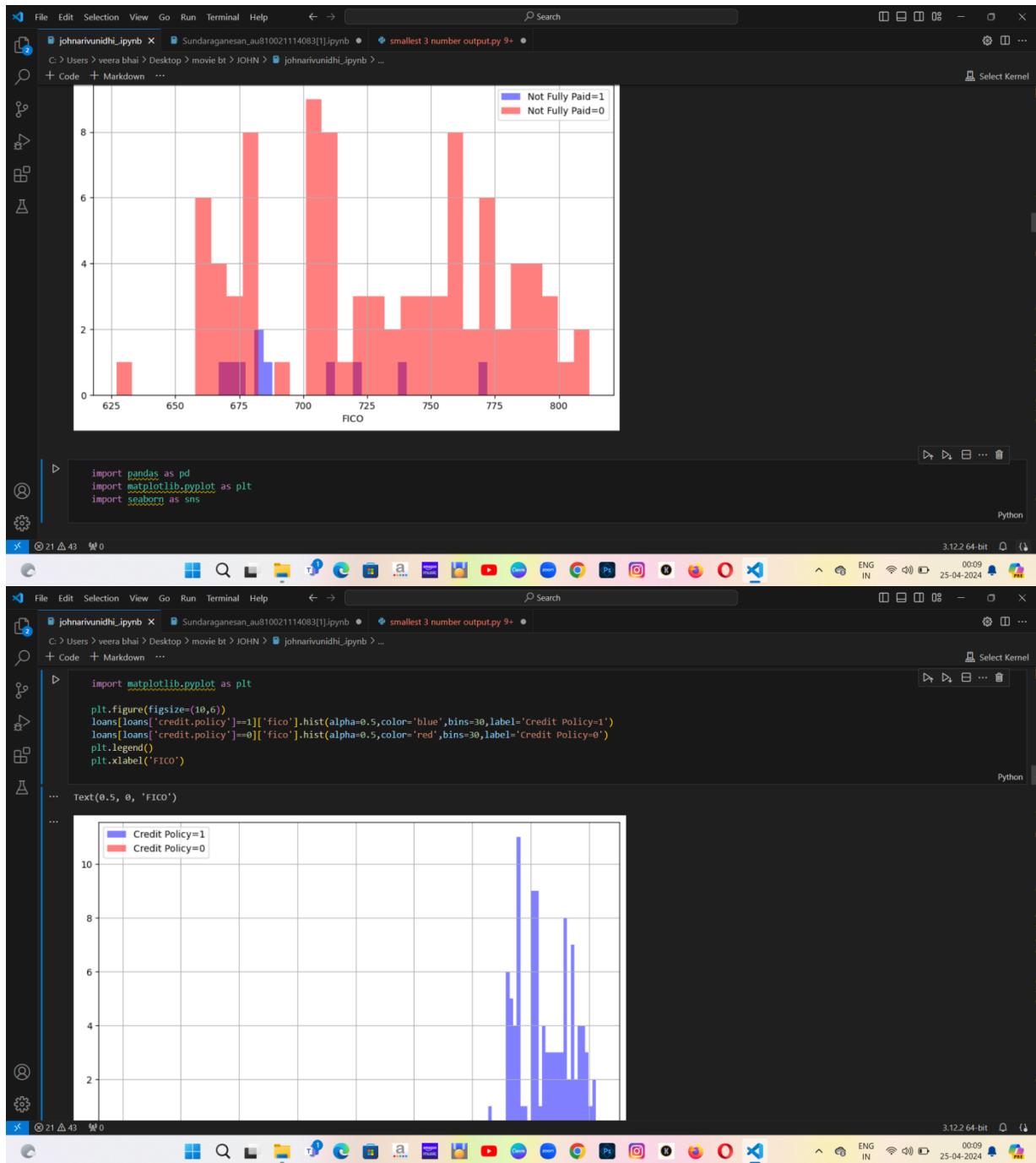
File Edit Selection View Go Run Terminal Help Search Select Kernel Python

plt.figure(figsize=(11,7))
sns.lmplot(x='fico',y='int.rate',data=loans,hue='credit.policy',col='not.fully.paid',palette='Set1')

3.12.2 64-bit 00:09 25-04-2024 ENG IN







Screenshot of Jupyter Notebook showing the output of the code snippet.

```

loans.head().info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 14 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   credit.policy    5 non-null     int64  
 1   purpose        5 non-null     object  
 2   int.rate       5 non-null     float64 
 3   installment    5 non-null     float64 
 4   log.annual.inc 5 non-null     float64 
 5   dti            5 non-null     float64 
 6   fico           5 non-null     int64  
 7   days.with.cr.line 5 non-null     float64 
 8   revol.bal     5 non-null     int64  
 9   revol.util    5 non-null     float64 
 10  inq.last.6mths 5 non-null     int64  
 11  delinq.2yrs   5 non-null     int64  
 12  pub.rec       5 non-null     int64  
 13  not.fully.paid 5 non-null     int64  
dtypes: float64(6), int64(7), object(1)
memory usage: 688.0+ bytes

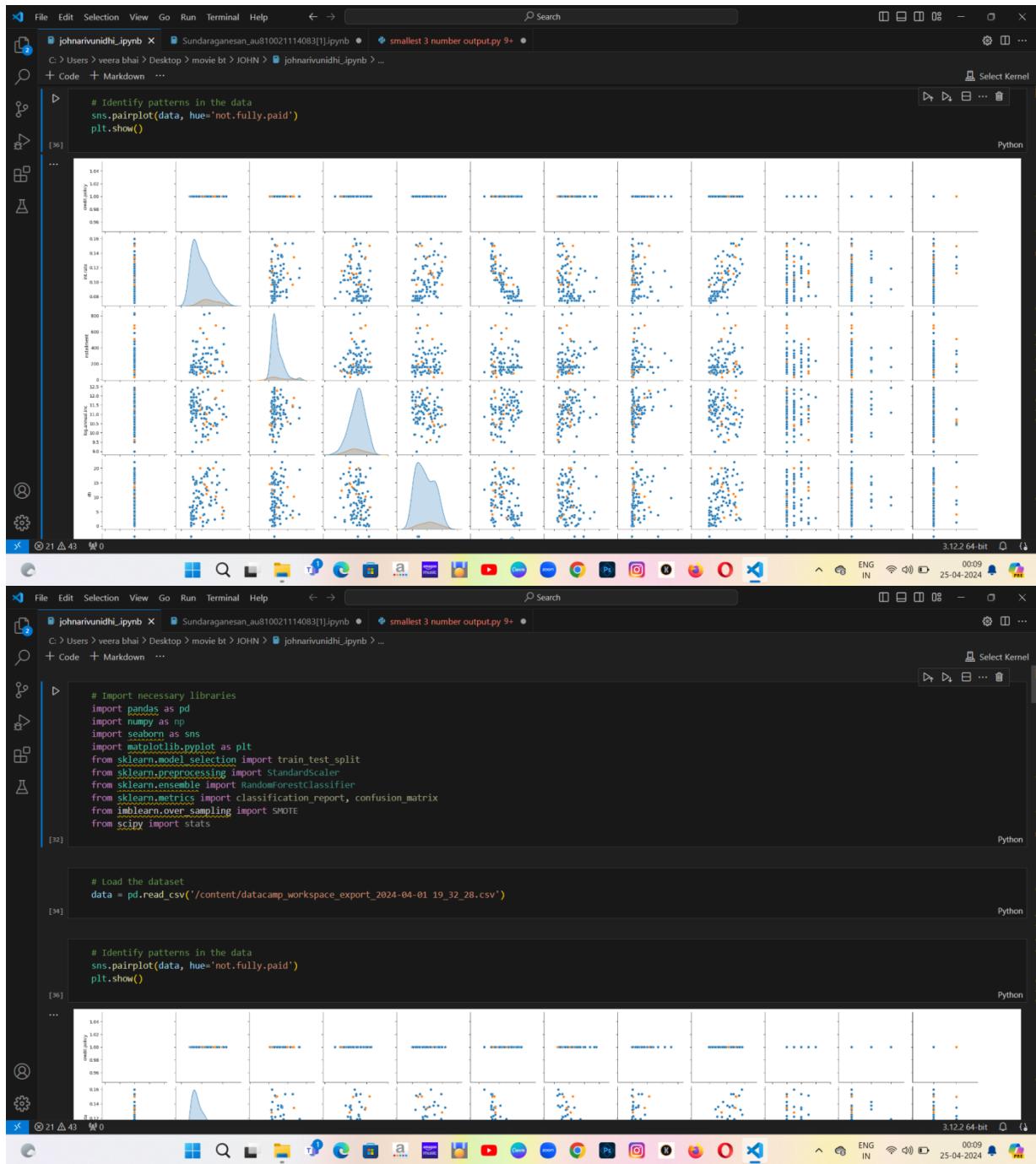
loans.head().shape

(5, 14)

loans.head().describe()

```

The notebook also shows the system status bar at the bottom indicating "3.12.2 64-bit" and the date "25-04-2024".



**Output:**

The codes and the output are screenshotted

## APP INTERFERENCE/ PROJECT RESULT

The end-to-end data science project resulted in the creation of an interactive chatbot that provides personalized loan eligibility predictions based on user input. Users can easily access this service through various messaging platforms, making it convenient and user-friendly. The integration of ChatGPT enhances the user experience by providing a conversational interface, making the process intuitive and accessible to a wider audience. Overall, the project demonstrates the potential of combining machine learning with natural language processing for practical applications like financial services.

## CONCLUSION

In conclusion, the implementation of an end-to-end data project utilizing ChatGPT for a loan dataset offers a robust solution for enhancing customer engagement and service efficiency in the lending domain. By leveraging natural language processing capabilities, this project enables seamless communication between users and the loan application system, providing instant assistance and guidance throughout the loan application process. Through meticulous data preprocessing, model training, integration, and deployment, this project ensures the delivery of accurate and relevant responses to user queries, ultimately facilitating a streamlined and user-friendly experience. With continuous monitoring and updates, this system remains adaptive and responsive to evolving user needs, thereby maximizing its effectiveness in serving borrowers and optimizing loan management processes.

## FUTURE SCOPE

Looking ahead, the future scope for an end-to-end data project utilizing ChatGPT for a loan dataset is promising and multifaceted. Advancements in natural language processing and machine learning techniques will enable the development of even more sophisticated and personalized loan application systems. Integration of additional data sources, such as social media profiles or financial transaction history, could enrich the model's understanding of borrower preferences and risk profiles, leading to more accurate loan decisions. Furthermore, incorporating voice recognition capabilities could enhance user accessibility and convenience, catering to a broader range of users. Collaboration with financial institutions and regulatory bodies may foster the adoption of standardized processes and compliance measures within the system, ensuring trust and reliability. Ultimately, the future holds immense potential for leveraging ChatGPT in loan management, driving innovation, and improving financial inclusion for individuals and businesses alike.

## REFERENCES

1. Project Github link, Ramar Bose , 2024
2. Project video recorded link (youtube/github), Ramar Bose , 2024
3. Project PPT & Report github link, Ramar Bose , 2024

## GIT Hub Link of Project Code:

<https://github.com/JOHNARIVUNIDHI/John-Arivunidhi>