A PROJECT REPORT

ON

FINANCIAL FRAUD DETECTION MODEL

SUBMITTED TO

UIET MDU, ROHTAK

IN

PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

Bachelor's of Technology

in

COMPUTER SCIENCE ENGINEERING

SUBMITTED TO:	**SUBMITTED BY:**
DR. HARKESH SEHRAWAT	MOHIT SHARMA
UIET MDU,ROHTAK	Exam Roll No: -2070031
	B. Tech(Final Year) CSE - A

UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY
MAHARSHI DAYANAND UNIVERSITY
ROHTAK-124001
May/June 2025

ABSTRACT

The purpose of this project report is to document the development and deployment of a predictive analytics tool titled **Financial Default Predictor**, created during the industrial training component of the Bachelor of Technology program in Computer Science and Engineering. The project aligns with the core educational objective of applying theoretical knowledge to real-world scenarios, particularly in the domains of machine learning and financial technology. This report reflects the comprehensive journey from conceptualization to implementation, showcasing how data-driven systems can aid in critical decision-making processes like creditworthiness assessment.

The project focuses on building a web-based system capable of predicting whether an individual is likely to default on a loan, along with estimating their probable credit score. This prediction is based on historical and financial data provided by the user, such as income levels, revolving utilization, debt ratios, and past delinquencies. Leveraging the **"Give Me Some Credit"** dataset from Kaggle, the system employs two core machine learning models: a **Logistic Regression** classifier for default prediction and a **Linear Regression** model for credit score estimation.

The web interface is built using **Streamlit**, a lightweight Python framework that enables rapid development of data apps with built-in support for charts, file uploads, and interactivity. The backend logic includes preprocessing, missing data imputation, data filtering, and ML model loading using joblib. Upon uploading a CSV file, the user receives predictions for each record along with insights such as risk categorization, visual distributions, and feature correlations.

The system is capable of categorizing users into high, moderate, and low-risk groups based on predicted credit scores and visualizing key features affecting default probability. It also provides interactive visual summaries like pie charts, box plots, and heatmaps to aid interpretation. The platform supports downloading prediction results as a CSV file, ensuring utility for both personal and institutional use.

By integrating machine learning models into a user-friendly web application, the Financial Default Predictor bridges the gap between advanced analytics and everyday financial risk assessment. It serves as a significant learning milestone, combining concepts from data science, web development, and financial analysis, and preparing the developer for real-world challenges in fintech and predictive modeling.

ACKNOWLEDGEMENT

I express my deepest gratitude to **Dr. Yudhvir Singh**, Director of the University Institute of Engineering and Technology (UIET), for granting me the opportunity to undertake this industrial training project as part of my B.Tech program. The opportunity to work on a real-world application of machine learning has significantly enriched my academic experience.

I am especially thankful to **Mrs. Amita Dhankar**, Coordinator of the Computer Science and Engineering Department, UIET, MDU Rohtak, for her continued encouragement and valuable suggestions throughout the duration of this project. Her support provided the necessary guidance and motivation to pursue a project that combines technical innovation with practical application.

I owe special thanks to **Dr. Harkesh Sehrawat**, Associate Professor, for his expert mentorship, constructive feedback, and insightful discussions that helped shape the direction and execution of this project. His encouragement played a vital role in the successful completion of this work.

I also wish to thank all faculty members of the Computer Science and Engineering Department for providing an intellectually stimulating environment that nurtured this project from conception to completion. Lastly, I am grateful to my friends and peers for their moral support and collaborative spirit throughout this journey.

Mohit Sharma

Exam Roll no.= 2070031

8th Semester

B. Tech (CSE), UIET

CANDIDATE DECLARATION CERTIFICATE

I hereby certify that the work presented in the project report titled **"Financial Default Predictor"** by **Mohit Sharma (2070031)** is a true and authentic record of my efforts carried out during the industrial training period from **January 2025 to May 2025**, under the guidance of **Dr. Harkesh Sehrawat**, Associate Professor at UIET, MDU Rohtak. This project is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering**.

This is to affirm that the content presented in this report is original and has not been submitted for the award of any other degree or diploma to any other institute or university.

Signature of the Student:	Signature of the SUPERVISOR:
Mohit Sharma	Dr. Harkesh Sehrawat
Exam Roll no.= 2070031	(UIET MDU, ROHTAK)

INDEX

Chapter 1: Introduction

The **Financial Default Predictor** is a smart, interactive, and data-driven web application developed to identify the likelihood of loan default and estimate an individual's credit score using machine learning techniques. Built during the final semester industrial training of the Bachelor of Technology program in Computer Science and Engineering, this project reflects the integration of predictive analytics with user-centric interface design to deliver valuable insights in the field of financial technology.

In today's credit-driven world, the ability to assess financial risk accurately is critical for both lenders and borrowers. Financial institutions rely heavily on credit scores and risk models to make decisions about loan approvals, credit limits, and interest rates. However, many such tools remain inaccessible to the general public or lack transparency in how predictions are made. The Financial Default Predictor was created to bridge this gap—offering users an open, interpretable, and accurate platform for evaluating default risk using real financial attributes.

The system uses a combination of supervised machine learning models: a **Logistic Regression** classifier to determine the probability of default, and a **Linear Regression** model to estimate the credit score based on input data. These models are trained on the well-known **"Give Me Some Credit"** dataset from Kaggle, which includes features like monthly income, debt ratio, age, late payments, and revolving utilization of unsecured credit lines.

To deliver this solution in a practical and user-friendly format, the application was developed using **Streamlit**, a Python-based web app framework that allows for rapid prototyping and real-time visualizations. Users can upload a CSV file containing financial records, and the system automatically preprocesses the data, runs predictions, and presents the output through interactive dashboards and charts.

Key features include:

- Visualization of income and age distribution
- Real-time predictions of default and credit score
- Risk categorization into High, Moderate, and Low groups
- Credit score distributions, heatmaps, and feature importance charts
- Option to download the prediction results for offline use

In summary, this project serves as a hands-on implementation of machine learning concepts within the financial domain. It offers a proof-of-concept for automated credit risk assessment systems and highlights the value of explainable AI in responsible financial decision-making.

Chapter 2 : Problem Statement

The increasing reliance on credit in both personal and commercial finance necessitates accurate, data-driven tools to assess the likelihood of loan default. Traditional credit scoring models, while widely adopted, often lack transparency and are usually limited to financial institutions. This creates a barrier for individuals and small lenders who wish to understand or anticipate default risk before making credit-related decisions. The primary goal of this project is to address these gaps by building an interpretable, web-based predictive system for default classification and credit score estimation.

Several problems in the current financial analysis landscape motivated the development of this project:

1. **Lack of accessible credit risk tools for the general public**
   Most risk assessment systems are proprietary and used by banks or financial entities. This restricts access for individuals who wish to proactively manage their credit health or assess their financial standing before applying for loans.

2. **Manual and error-prone risk evaluation**
   Many businesses still rely on rule-based assessments, which are vulnerable to human bias, inconsistent decisions, and outdated risk models. There is a strong need to automate this evaluation using machine learning for consistency and accuracy.

3. **Data complexity and feature interdependencies**
   Understanding how various features like revolving utilization, debt ratio, and late payments affect default risk is not straightforward. Without intelligent models, these interdependencies remain hidden, leading to oversimplified evaluations.

4. **Limited interpretability in AI-based financial models**
   Black-box AI models often lack explainability, which is critical in financial decision-making. This project addresses that by using interpretable models (like logistic regression) and visualizing top contributing features to improve transparency.

5. **Inability to visualize or quantify financial risk levels dynamically**
   Most existing systems provide scores without insight into what they mean for an individual's risk level. This project introduces a "Risk Category" system (Low, Moderate, High) to improve interpretability and aid in decision-making.

The **Financial Default Predictor** proposes a practical solution by:

- Enabling users to upload their financial data for analysis

- Predicting loan default and estimating credit scores instantly
- Categorizing users by risk levels with visual insights
- Offering model explainability through feature importance and correlation heatmaps
- Allowing easy integration with real-world datasets and business workflows

This project demonstrates how intelligent systems can democratize access to financial insights and enable smarter, data-informed decisions for individuals and institutions alike.

Chapter 3: How it works

The **Financial Default Predictor** is designed to function as an interactive, ML-powered web application that predicts a person's likelihood of defaulting on a loan and provides an estimate of their credit score. This is achieved through a blend of machine learning models and a visually rich, user-friendly front-end interface built using Streamlit. Below is a detailed explanation of how the system works, from data input to risk categorization and result visualization.

1. User Input and Interface

The application interface begins with a welcoming message and a clean layout that guides the user through the prediction workflow. Users are prompted to upload a CSV file containing financial information such as income, credit utilization, age, and delinquency history. For demonstration purposes, a sample dataset is also loaded and visualized when no file is provided. The interface allows users to control risk thresholds for categorizing credit scores into high, moderate, and low risk groups.

2. Data Loading and Preprocessing

Upon file upload, the application performs several preprocessing steps:

- Dropping irrelevant or empty columns (e.g., unnamed index columns)
- Handling missing values using median imputation
- Filtering invalid entries such as age less than or equal to zero or extremely high credit utilization ratios

These steps ensure that only clean, structured, and meaningful data is passed to the predictive models.

3. Machine Learning Models

Two distinct models are loaded using joblib:

- **Logistic Regression Classifier**: Trained to predict binary default outcomes (Yes/No)
- **Linear Regression Model**: Trained to estimate credit score values

Both models are pre-trained on the Kaggle "Give Me Some Credit" dataset, and the features used include:

- Monthly income
- Debt ratio
- Number of open credit lines
- Delinquency records
- Credit utilization of unsecured lines
- Age

The classifier outputs both the prediction (0 or 1) and the probability of default (0.00 to 1.00). The regression model provides a numerical score, which is further categorized into risk levels using the customizable thresholds.

4. Risk Categorization and Output Generation

Based on the predicted credit score:

- **Low Risk**: Score ≥ threshold_low (default: 750)
- **Moderate Risk**: threshold_moderate ≤ Score < threshold_low
- **High Risk**: Score < threshold_moderate

The final dataset is augmented with predicted default, probability, credit score, and risk category columns. These values are displayed in tabular format for user review.

5. Visual Insights and Analytics

The platform includes the following visual tools to assist in interpretation:

- Histograms for monthly income and age
- Distribution of predicted credit scores
- Box plots showing default probability by risk category
- Scatter plot for credit score vs. default probability
- Pie chart of overall default distribution
- Heatmap of feature correlation

- Bar chart for top contributing features based on model coefficients or importance scores

These charts offer deep insights into the data and model behavior, aiding transparency and decision support.

6. Download and Model Explanation

Users can download the prediction results as a CSV file. An expandable section explains the underlying models, their types, and the features used for training. This improves trust and understanding of the system.

In conclusion, the workflow is designed to be seamless—from uploading data to viewing detailed results, with clear explanations, flexible controls, and meaningful visuals that make the application both powerful and approachable.

Chapter 4: Feasibility Study

To determine the practicality and scalability of implementing the **Financial Default Predictor**, a feasibility study was conducted across three main domains: technical, operational, and financial. The goal was to ensure that the system could be realistically developed, deployed, and maintained using available tools, resources, and timelines.

1. Technical Feasibility

The technical requirements of the project are well-aligned with widely available open-source tools and platforms. The application uses:

- **Python** as the core programming language due to its simplicity and rich ecosystem
- **Scikit-learn** for model training, offering accessible APIs for regression and classification
- **Streamlit** for building the web interface without needing full-stack frontend/backend separation
- **Pandas and NumPy** for data handling
- **Matplotlib and Seaborn** for visualizations

The machine learning models are trained offline in a Jupyter notebook and exported using joblib for deployment in the Streamlit app. This separation of training and inference simplifies deployment and ensures faster load times during user interaction.

Since the model inference does not require GPU acceleration, the system can run on modest cloud-based compute instances or even local environments without specialized hardware. The risk thresholds are adjustable via sliders, allowing the system to be tailored to different use cases.

2. Operational Feasibility

From an operational standpoint, the application is easy to maintain and extend. The codebase is modular, with clear separation between data processing, model loading, and UI logic. This makes debugging and updates manageable.

The platform is intended for standalone use by individuals or small teams, and it does not require complex infrastructure or third-party integrations to function. Additionally, the use of Streamlit means no front-end development is needed, reducing maintenance overhead.

Security and privacy concerns are minimal since the system operates entirely on the client-side without storing user data in the cloud. Users upload their data, receive predictions, and download the results—all within the same session.

3. Financial Feasibility

The financial requirements for the project are minimal. The development was carried out using free, open-source tools, and the dataset was obtained from a public Kaggle repository. No licensing fees were incurred.

Deployment can be done using free or low-cost platforms like:

- **Streamlit Community Cloud**
- **Render** (basic tier)
- **Heroku** (free dynos)

No additional costs are associated with model inference since the models are lightweight and do not rely on external APIs or commercial machine learning services. Optional enhancements like adding login functionality or database storage can be added in the future if scaling is needed.

Conclusion

The **Financial Default Predictor** is technically sound, operationally maintainable, and financially viable. Its lightweight architecture, reliance on open-source tools, and ease of use make it an excellent candidate for real-world deployment, especially in academic or small-business environments seeking practical financial risk analysis.

Chapter 5: Objective and Scope

Objective

The main objective of the **Financial Default Predictor** project is to develop an intelligent, accessible, and explainable financial risk assessment platform that uses machine learning to predict default probability and estimate credit scores. This project aims to provide individuals and institutions with a tool to analyze creditworthiness efficiently, without relying on opaque third-party systems.

Key objectives of the project are:

1. **Build an ML-Based Default Prediction System**
   Develop a machine learning classifier that can accurately predict whether an individual is likely to default on a loan using historical financial data.

2. **Estimate Credit Score Through Regression**
   Train a regression model that can predict a user's credit score based on a combination of features such as income, utilization ratio, and past payment history.

3. **Create a User-Friendly Web Interface**
   Design and deploy an interactive Streamlit web application that allows users to upload CSV files and receive instant predictions and visual insights.

4. **Visualize Financial Risk and Model Behavior**
   Integrate charts and heatmaps to show distribution of predicted scores, correlation between variables, and top contributing features in default prediction.

5. **Enable Risk Categorization Based on Score Thresholds**
   Allow dynamic classification of users into High, Moderate, and Low risk groups by setting custom score thresholds.

6. **Ensure Transparency and Model Explainability**
   Use interpretable models like Logistic and Linear Regression and display feature importance to help users understand the logic behind predictions.

7. **Promote Reusability and Modularity**
   Structure the code so that it can be reused, extended, or integrated into larger financial systems in the future.

8. **Support CSV Upload and Download Functionality**
   Provide end-to-end interaction from file upload to processed results, with an option to download final predictions for offline use.

Scope

The scope of the **Financial Default Predictor** encompasses building a full-fledged prototype that combines machine learning and web-based visualization to address key challenges in credit risk evaluation. The system is designed for scalability and modularity, enabling future enhancement and deployment at a larger scale.

The project's scope includes the following:

- **Functional Scope**:
  - User uploads financial data via CSV
  - Models run predictions on each record
  - Results are displayed along with visual charts and summary metrics
  - Users can modify risk thresholds and download outputs
- **Technical Scope**:
  - Uses Python for data processing and modeling
  - Implements Logistic and Linear Regression using scikit-learn
  - Frontend developed using Streamlit for interactivity and visualization
  - Uses Matplotlib and Seaborn for statistical plots
- **User Scope**:
  - Individuals who want to assess their credit status
  - Financial advisors evaluating client risk
  - Educators and students learning ML in finance
  - Institutions seeking proof-of-concept tools
- **Model Scope**:
  - Trained on real-world dataset from Kaggle
  - Can be extended to include more features (e.g., employment history, loan purpose)

  - Designed for batch prediction via uploaded datasets
- **Deployment Scope**:
  - Can be run locally or deployed on platforms like Streamlit Cloud, Render, or Heroku
  - Supports integration of updated models without UI changes
- **Future Scope**:
  - Add database support for user tracking and history
  - Expand to multi-class risk grading
  - Integrate authentication for secure, personalized dashboards
  - Connect with real-time financial APIs for live analysis

In conclusion, the scope of the project is broad enough to serve real use cases while being compact and well-structured for academic presentation and future enhancement.

Chapter 6: Technologies used in the project

The **Financial Default Predictor** combines various technologies from the domains of data science, machine learning, and web development. The stack was chosen based on simplicity, community support, and suitability for both rapid development and deployment.

1. Python

Python was the primary programming language used across all components. Its readable syntax and vast ecosystem make it ideal for both data analysis and web app development. Libraries like pandas, numpy, and joblib were used for data preprocessing, numerical computation, and model serialization, respectively.

2. Scikit-learn

Scikit-learn is a popular Python library for machine learning. It was used to:

- Train a **Logistic Regression** model for binary default classification
- Train a **Linear Regression** model to estimate credit score
- Evaluate model performance through metrics like accuracy and R^2 score

The models were exported as .pkl files using joblib for deployment.

3. Pandas and NumPy

These libraries were used for data manipulation and numerical operations. Key functionalities included handling missing values, filtering invalid entries, and preparing feature matrices for the models.

4. Streamlit

Streamlit was used to develop the web interface. It offers:

- Quick prototyping of dashboards and data apps
- Widgets for uploading files, adjusting thresholds, and displaying charts
- Seamless integration with Python ML scripts

5. Matplotlib and Seaborn

Both libraries were utilized for generating rich visualizations:

- **Histograms** for income and age distribution
- **Box plots** for default probability analysis
- **Scatter plots** to show relationships between score and risk
- **Pie charts** for class distribution
- **Heatmaps** for correlation analysis

These visuals help interpret model outputs and data patterns effectively.

6. Jupyter Notebook

The initial model training and data analysis were conducted in Jupyter Notebook. It allowed for iterative experimentation with models and easy visualization of performance metrics.

7. Joblib

Joblib was used to serialize the trained models so they could be efficiently loaded into the web app for real-time predictions without retraining.

8. HTML/CSS (via Streamlit injection)

Though Streamlit manages layout styling, custom CSS was embedded using st.markdown for UI enhancement. This was used to:

- Style buttons and headers
- Format risk category labels
- Improve color schemes for better UX

Chapter 7: Resource Requirements

The development of the **Financial Default Predictor** required a blend of software tools, libraries, and computing resources. As a data science project deployed via a web interface, the resource requirements were modest and focused primarily on model training, visualization, and user interaction.

1. Software Resources

- **Python (v3.8+)**

Used as the core language for data processing, model building, and app development. Python's simplicity and strong ecosystem made it the ideal choice for the entire project.

- **Jupyter Notebook**

Served as the environment for model training, data cleaning, feature exploration, and EDA (exploratory data analysis).

- **Scikit-learn**

Used to train and evaluate the machine learning models, specifically Logistic Regression for default classification and Linear Regression for credit score prediction.

- **Pandas & NumPy**

Essential libraries for data manipulation, feature engineering, and handling numerical arrays.

- **Matplotlib & Seaborn**

Used for creating static, animated, and interactive plots to visualize relationships between variables and summarize insights.

- **Streamlit**

The web application was built using Streamlit to allow users to upload data, run predictions, adjust risk thresholds, and interact with visual outputs.

- **Joblib**

Required to serialize and deserialize trained models for fast deployment.

- **IDE/Editor**

Visual Studio Code was used to write, manage, and test all Python and Streamlit scripts.

2. Hardware Resources

- **Development Machine**

A standard laptop or desktop with the following minimum configuration was sufficient for building and testing the app:

- ○ Processor: Intel i5 or equivalent
- ○ RAM: 8 GB
- ○ Storage: 256 GB SSD
- ○ OS: Windows/Linux/Mac

- **Cloud Hosting (Optional)**
The application can be deployed on free-tier services like:

- ○ **Streamlit Community Cloud**
- ○ **Render** (Basic Plan)
- ○ **Heroku** (Free Dynos)

- These platforms provide sufficient compute power for inference-only apps with low user concurrency.

3. Dataset

- **Source**: Kaggle – "Give Me Some Credit" Dataset
The dataset includes historical financial data for over 150,000 individuals, with features like:

- ○ Age
- ○ Monthly Income
- ○ Number of Open Credit Lines
- ○ Number of Delinquencies
- ○ Revolving Utilization
- ○ Debt Ratio

- This dataset served as the training foundation for both predictive models.

4. Human Resources

- **Developer (1)**: Responsible for end-to-end development—data analysis, model training, UI creation, and deployment.

- **Guide/Supervisor (1)**: Provided academic oversight and feedback on the project direction and outcomes.

- **Peer Reviewers (Optional)**: Classmates who provided usability testing and suggestions during the app's testing phase.

5. Optional Resources

- **Version Control**: Git & GitHub (for maintaining project versions)

- **Cloud Storage**: Google Drive or GitHub for storing training files and model weights

In summary, the project required minimal infrastructure and leveraged readily available tools and free datasets to deliver a fully functional, deployable, and educational application.

Chapter 8: Target Audience

The **Financial Default Predictor** is intended to assist a wide range of users who need accessible and reliable tools to assess financial risk. The application's versatility, ease of use, and interpretability make it beneficial across multiple domains:

1. Students and Academic Researchers

- **Purpose**: To learn about financial modeling, machine learning, and data visualization.
- **Benefits**:
  - Offers a real-world dataset for practice
  - Demonstrates ML deployment through Streamlit
  - Helps understand the impact of financial features on credit risk

2. Individual Users (Consumers)

- **Purpose**: To estimate personal creditworthiness before applying for a loan.
- **Benefits**:
  - Upload financial data securely
  - Receive risk categorization and score estimates
  - No need to register or provide personal identifiers

3. Financial Advisors and Consultants

- **Purpose**: To assess client risk and guide credit or loan applications.
- **Benefits**:
  - Run predictions on client portfolios
  - Visualize data trends and score distributions
  - Tailor advice based on categorized risk levels

4. Educational Institutions

- **Purpose**: To use as a teaching tool in finance, data science, or AI-related courses.
- **Benefits**:
  - Includes interpretable models
  - Encourages understanding of ethical AI usage in finance
  - Can be extended for coursework and lab assignments

5. Credit Analysts and Loan Officers (SMEs)

- **Purpose**: For initial pre-screening or analysis in microfinance or small institutions.
- **Benefits**:
  - Simple CSV-based workflow
  - Clear indicators of default risk and score range
  - No need for costly enterprise software

6. Data Science Enthusiasts

- **Purpose**: To learn end-to-end machine learning project structure.
- **Benefits**:
  - Explore model training and deployment flow
  - Practice building interactive apps using real data
  - Expand into more advanced algorithms or APIs

7. Fintech Startups (Proof of Concept)

- **Purpose**: To validate ideas around risk scoring systems or credit evaluation services.

- **Benefits**:
  - Lightweight and modular codebase
  - Can be integrated into larger ecosystems
  - Offers a starting point for compliance-based modeling

The application promotes inclusivity by avoiding login barriers and making risk modeling understandable for non-technical users. This wide reach ensures that the Financial Default Predictor is both academically valuable and practically useful in the evolving landscape of digital finance.

Chapter 9: Methodology

The development of the **Financial Default Predictor** followed a structured and iterative methodology rooted in the principles of the **Agile Software Development Life Cycle (SDLC)**. This approach was chosen to ensure modular development, regular testing, and continuous refinement of the system based on feedback and performance.

1. Requirement Analysis

The first step involved identifying the key objectives and requirements of the project. This included:

- The need to build a machine learning model to classify users into defaulters and non-defaulters
- A secondary goal of predicting estimated credit scores
- A simple, intuitive user interface for file upload, prediction, and output visualization
- Download functionality for processed results
- The ability to visualize insights from both the dataset and model predictions

The target audience was also identified—ranging from individuals to educators and financial professionals—which helped define the usability requirements and visualization preferences.

2. Dataset Understanding and Preprocessing

The **"Give Me Some Credit"** dataset from Kaggle was used as the training foundation. The dataset was cleaned and preprocessed by:

- Handling missing values (e.g., imputing MonthlyIncome)
- Removing outliers and zero/negative ages
- Normalizing certain features and checking for correlations
- Splitting data into features (X) and target labels (y)

Separate versions of the dataset were used for training the **Logistic Regression** model (binary classification) and the **Linear Regression** model (continuous credit score prediction).

3. Model Selection and Training

Two models were trained using **scikit-learn**:

- **Logistic Regression** for predicting default (output: 0 or 1)
- **Linear Regression** for estimating credit score (based on a numeric target generated synthetically for this project)

The models were evaluated using:

- Accuracy, Precision, and Recall for classification
- R^2 score and Mean Absolute Error (MAE) for regression

Once finalized, the trained models were serialized using joblib for deployment in the web app.

4. Frontend Development with Streamlit

The web interface was developed using **Streamlit**, allowing:

- File upload for financial data in CSV format
- Automatic execution of predictions upon upload
- Visualization of inputs (income, age, utilization)
- Risk threshold sliders to define High, Moderate, and Low risk zones
- Plots showing credit score distributions, default probabilities, and correlation matrices

Streamlit's widgets and layout tools enabled real-time interactivity and streamlined the user experience.

5. Testing and Debugging

Testing was done at multiple levels:

- **Unit Testing** for model inputs and outputs
- **Integration Testing** for pipeline steps (from upload to visualization)

- **Usability Testing** for the Streamlit UI

Edge cases (empty files, null values, invalid columns) were tested to ensure robust behavior under varied input conditions.

6. Deployment and Sharing

The app was prepared for deployment by:

- Hosting the code on GitHub
- Packaging dependencies into requirements.txt
- Testing with dummy data in a clean environment
- Optional deployment through Streamlit Cloud for public access

Future deployment options include Heroku, Render, or Docker-based containers for scalability.

7. Feedback and Iteration

User feedback from peers and mentors led to several improvements:

- Better error handling for corrupt uploads
- Clearer labels and instructions
- More detailed model explainability section
- Download button for predictions

The final version of the app provides an end-to-end experience from data entry to insights and output generation.

Chapter 10: Process Description

The creation of the **Financial Default Predictor** involved a phased, end-to-end development process, integrating machine learning with web deployment. Below is a detailed breakdown of each phase:

Phase 1: Problem Definition

The problem to solve was twofold:

1. Predict whether a user will default on a loan based on financial attributes

2. Estimate their credit score and categorize them into risk groups

These goals were defined through initial brainstorming and guided by the capabilities of the Kaggle dataset.

Phase 2: Data Collection and Cleaning

- The dataset was sourced from Kaggle’s open dataset repository.
- The raw data was loaded in Jupyter Notebook and examined.
- Irrelevant or duplicate columns were dropped.
- Missing values in MonthlyIncome were replaced using median imputation.
- Zero or negative values in Age and extreme outliers in credit utilization were filtered out.

A clean, standardized dataset was prepared for both training and demo usage.

Phase 3: Feature Engineering and Model Training

Key features were selected, including:

- Revolving utilization
- Debt ratio
- Number of delinquencies
- Number of open credit lines

- Monthly income
- Age

The dataset was split into train-test subsets. The following models were trained:

- Logistic Regression for classification
- Linear Regression for credit score estimation

Performance metrics were evaluated, and the models were fine-tuned for best results.

Phase 4: Model Serialization

Both models were saved using **joblib**:

python
CopyEdit

- joblib.dump(model, "model.pkl")

This allowed quick loading in the deployed app without retraining.

Phase 5: Streamlit App Development

- Streamlit's UI elements (file_uploader, slider, checkbox, button) were used to guide users through uploading, previewing, and analyzing their data.
- A default sample dataset was shown when no file was uploaded.
- Results were displayed with confidence scores, credit scores, and risk categories.
- Graphs were created using **Matplotlib** and **Seaborn**, embedded via st.pyplot.

Phase 6: Visualization and Insights

The following visual outputs were created:

- Histograms for income and age
- Pie chart of default classes
- Heatmap of correlation matrix

- Boxplot of default vs credit score
- Risk category pie chart and bar charts

These helped users interpret results visually.

Phase 7: Testing and Debugging

Test cases were added for:

- Incorrect file format
- Missing columns
- Empty datasets

Logging was used to monitor internal errors, and user-friendly messages were displayed when issues occurred.

Phase 8: Export and Download

A st.download_button() was added to let users export predictions with appended columns:

- Default_Prediction
- Default_Probability
- Estimated_Score
- Risk_Category

Phase 9: Final Review and Packaging

Final adjustments included:

- Polishing layout and labels
- Adding comments and documentation
- Preparing README for GitHub
- Verifying dependencies in requirements.txt

Chapter 11: Code Module

EDA &

```
import numpy as np
import pandas as pd

df = pd.read_csv("data/cs-training.csv")

df.info()


# In[2]:


df.head(50)


# In[3]:


df['SeriousDlqin2yrs'].value_counts()


# In[4]:


df.drop(columns=["Unnamed: 0"], inplace = True)
df.head()


# In[5]:


import seaborn as sns
import matplotlib.pyplot as plt

sns.countplot(x='SeriousDlqin2yrs', data=df)
plt.title("Loan Default Distribution")


# In[6]:
```

```
df.hist(bins=30, figsize=(15, 12))
plt.tight_layout()


# In[7]:


sns.boxplot(x=df['MonthlyIncome'])
sns.boxplot(x=df['DebtRatio'])


# In[8]:


corr = df.corr()
plt.figure(figsize=(9, 6))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title("Feature Correlation Heatmap")


# In[9]:


sns.boxplot(x='SeriousDlqin2yrs', y='MonthlyIncome', data=df)


# In[10]:


sns.histplot(data=df, x='age', hue='SeriousDlqin2yrs', bins=30, kde=True)


# In[11]:


#sns.histplot(df['RevolvingUtilizationOfUnsecuredLines'], bins=100)
#plt.xlim(0, 2)  # clip for readability


# In[12]:
```

```
late_cols = [
    'NumberOfTime30-59DaysPastDueNotWorse',
    'NumberOfTime60-89DaysPastDueNotWorse',
    'NumberOfTimes90DaysLate'
]

for col in late_cols:
    sns.countplot(x=col, hue='SeriousDlqin2yrs', data=df)
    plt.title(f'{col} vs Default')
    plt.show()


# In[13]:


from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score, mean_squared_error, r2_score
import joblib


# In[14]:


# 🎯 Step 5: Classification - Loan Default Prediction
X_cls = df.drop('SeriousDlqin2yrs', axis=1)
y_cls = df['SeriousDlqin2yrs']

X_train_cls, X_test_cls, y_train_cls, y_test_cls = train_test_split(X_cls, y_cls, test_size=0.3,
random_state=42)

rf_cls = RandomForestClassifier()
rf_cls.fit(X_train_cls, y_train_cls)
y_pred_cls = rf_cls.predict(X_test_cls)

print("Classification Report:\n", classification_report(y_test_cls, y_pred_cls))
print("ROC AUC:", roc_auc_score(y_test_cls, rf_cls.predict_proba(X_test_cls)[:, 1]))



# In[15]:
```

```
#joblib.dump(rf_cls, "default_model.pkl")


# In[16]:


# 📊 Step 6: Regression - Credit Score Forecasting
# Create a synthetic credit score (300-850 scale)
df['custom_credit_score'] = 850 \
    - df['RevolvingUtilizationOfUnsecuredLines'] * 100 \
    - df['NumberOfTime30-59DaysPastDueNotWorse'] * 20 \
    - df['NumberOfTime60-89DaysPastDueNotWorse'] * 25 \
    - df['NumberOfTimes90DaysLate'] * 30 \
    - df['DebtRatio'] * 50

df['custom_credit_score'] = df['custom_credit_score'].clip(lower=300, upper=850)

df.head()


# In[17]:


X_reg = df.drop(['SeriousDlqin2yrs', 'custom_credit_score'], axis=1)
y_reg = df['custom_credit_score']

X_train_reg, X_test_reg, y_train_reg, y_test_reg = train_test_split(X_reg, y_reg, test_size=0.2,
random_state=42)

rf_reg = RandomForestRegressor()
param_dist = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

search = RandomizedSearchCV(rf_reg, param_distributions=param_dist, n_iter=10, cv=3, n_jobs=-1,
random_state=42, scoring='neg_mean_squared_error')
search.fit(X_train_reg, y_train_reg)

best_rf_reg = search.best_estimator_
y_pred_reg = best_rf_reg.predict(X_test_reg)
```

```
print("R2 Score:", r2_score(y_test_reg, y_pred_reg))


# In[18]:


print("RMSE:", np.sqrt(mean_squared_error(y_test_reg, y_pred_reg)))


# In[19]:


#joblib.dump(best_rf_reg, "score_model.pkl")


# In[20]:


import matplotlib.pyplot as plt

feat_importances = pd.Series(best_rf_reg.feature_importances_, index=X_reg.columns)
feat_importances.nlargest(10).plot(kind='barh')
plt.title("Top 10 Feature Importances")
plt.show()


# In[21]:


test_df = pd.read_csv("data/cs-test.csv")
test_df.drop(columns=["Unnamed: 0","SeriousDlqin2yrs"], inplace=True)
test_df.head()


# In[22]:


test_df.fillna(test_df.median(), inplace=True)
test_df.head(10)


# In[23]:
```

```
# Load the trained model
import joblib
default_model = joblib.load("default_classifier.pkl")

# Predict
default_preds = default_model.predict(test_df)

# Optional: Probabilities
default_probs = default_model.predict_proba(test_df)[:, 1]

# Save results
submission_df = pd.DataFrame({
    "Id": test_df.index,
    "PredictedDefault": default_preds,
    "DefaultProbability": default_probs
})


# In[24]:


submission_df.head(50)


# In[25]:


score_model = joblib.load("score_model.pkl")

credit_score_preds = score_model.predict(test_df)

# Save results
credit_score_df = pd.DataFrame({
    "Id": test_df.index,
    "PredictedCreditScore": credit_score_preds
})


# In[26]:


credit_score_df.head(10)
```

```
# In[27]:


final_results = pd.DataFrame({
    "Id": test_df.index,
    "PredictedDefault": default_preds,
    "DefaultProbability": default_probs,
    "PredictedCreditScore": credit_score_preds
})


# In[28]:


final_results.head(10)


# In[29]:


final_results.to_csv("final_financial_advice.csv", index=False)


# In[30]:


sns.histplot(final_results["PredictedCreditScore"], bins=30)
plt.title("Predicted Credit Score Distribution")
plt.show()

sns.countplot(x="PredictedDefault", data=final_results)
plt.title("Predicted Default Counts")
plt.show()


# In[31]:


sns.scatterplot(
    x=final_results['PredictedCreditScore'],
    y=final_results['DefaultProbability'],
    hue=final_results['PredictedDefault'],
    palette='coolwarm',
    alpha=0.6
)
plt.title("Credit Score vs Default Probability")
plt.xlabel("Predicted Credit Score")
plt.ylabel("Default Probability")
plt.legend(title="Predicted Default")
plt.grid(True)
plt.show()


# In[32]:


# Categorize based on credit score
def risk_bucket(score):
    if score >= 750:
        return "Low Risk"
    elif score >= 600:
        return "Moderate Risk"
    else:
        return "High Risk"

final_results["RiskCategory"] = final_results["PredictedCreditScore"].apply(risk_bucket)

# Countplot
sns.countplot(x="RiskCategory", data=final_results, palette="Set2")
plt.title("Customer Distribution by Risk Category")
plt.show()


# In[33]:


default_counts = final_results['PredictedDefault'].value_counts()
labels = ['No Default', 'Default']
plt.pie(default_counts, labels=labels, autopct='%1.1f%%', startangle=140, colors=['#66b3ff',
'#ff6666'])
plt.axis('equal')
plt.title("Predicted Default Breakdown")
plt.show()


# In[34]:
```

```
sns.boxplot(x="RiskCategory", y="DefaultProbability", data=final_results, palette="pastel")
plt.title("Default Probability by Risk Category")
plt.show()


# In[35]:


sns.kdeplot(final_results["PredictedCreditScore"], fill=True, color='purple')
plt.title("Predicted Credit Score Density")
plt.xlabel("Score")
plt.ylabel("Density")
plt.show()
```

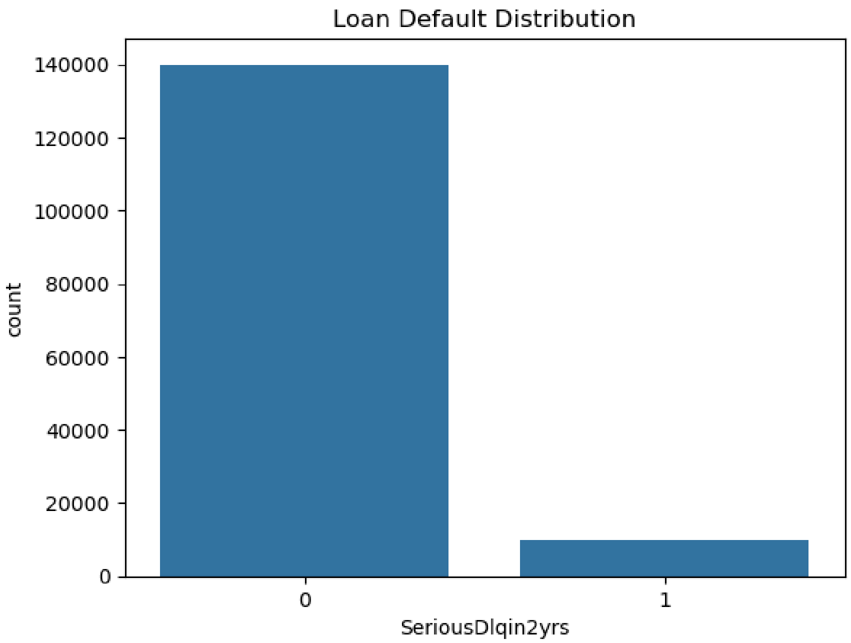
UI/UX FIle (app.py) -

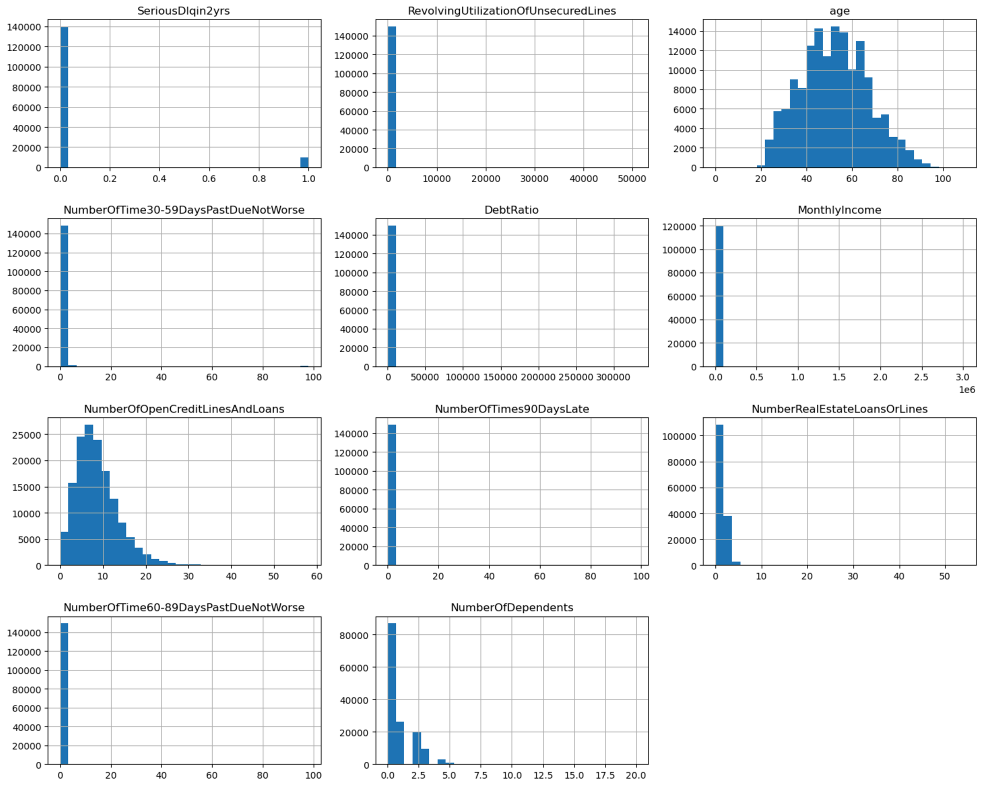
```
# app.py - Enhanced Streamlit UI for Financial Advisor ML Project

import streamlit as st
import pandas as pd
import joblib
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# ------------------- Page Configuration -------------------
st.set_page_config(page_title="💰 Financial Advisor App", layout="wide")

st.markdown("""
    <style>
        html, body, [class*="css"]  {
            font-family: 'Segoe UI', sans-serif;
        }
        .main {
            background-color: #f8f9fa;
        }
        h1, h2, h3 {
            color: #2c3e50;
        }
        .stButton > button {
            background-color: #0066cc;
            color: white;
            border-radius: 8px;
            padding: 8px 20px;
        }
        .stDownloadButton > button {
            background-color: #28a745;
            color: white;
            border-radius: 8px;
        }
        .risk-low {
            background-color: #d4edda;
            color: #155724;
            padding: 5px;
            border-radius: 5px;
        }
        .risk-moderate {
            background-color: #fff3cd;
            color: #856404;
            padding: 5px;
            border-radius: 5px;
        }
        .risk-high {
            background-color: #f8d7da;
            color: #721c24;
            padding: 5px;
            border-radius: 5px;
        }
    </style>
""", unsafe_allow_html=True)

# ------------------- Title and Description -------------------
st.title("💰 Financial Advisor - Default & Credit Score Prediction")

st.markdown("""
Welcome to the **Financial Advisor App**! This tool helps predict:
- Whether a person is likely to default on a loan
- An estimate of their credit score

👈 Upload your CSV in the sidebar to begin, or review the sample demo below to understand the model.
""")

# ------------------- Load and Show Demo Visualizations -------------------
@st.cache_data
def load_example():
    return pd.read_csv("data/sample_input.csv")
```

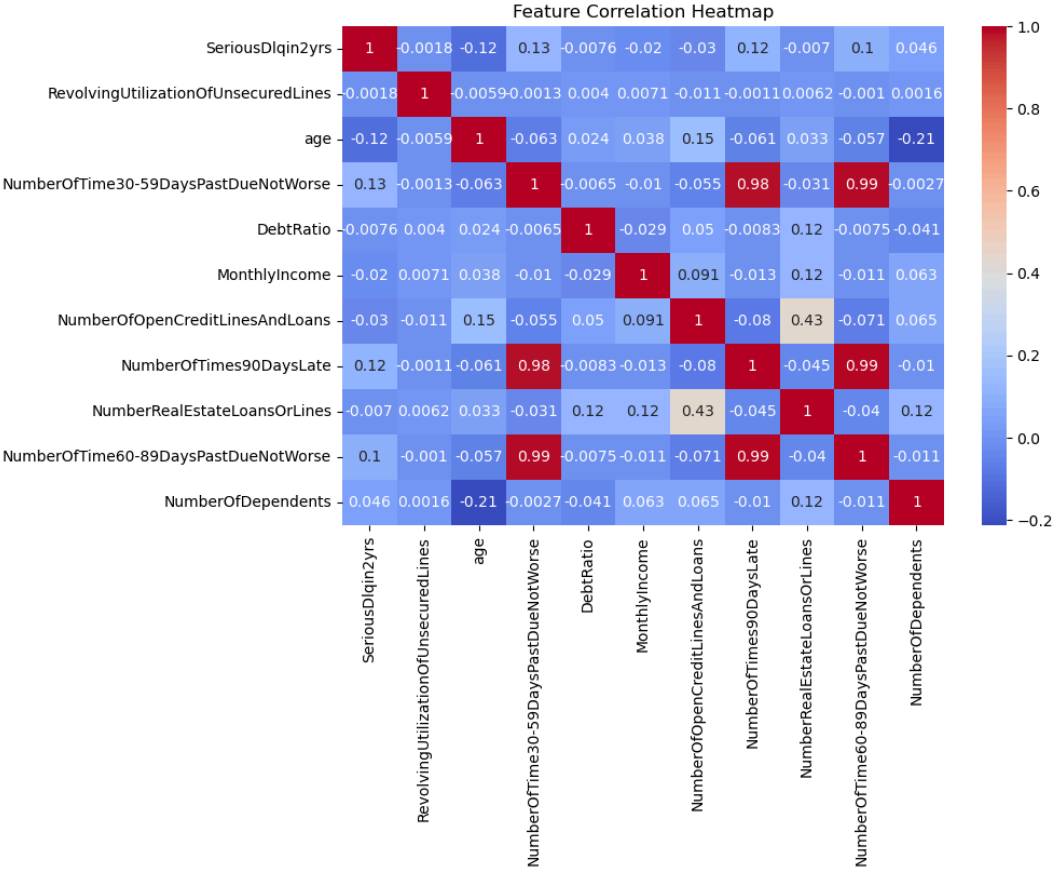
```
example_df = load_example()
example_df.fillna(example_df.median(numeric_only=True), inplace=True)
example_df = example_df[(example_df['age'] > 0) &
(example_df['RevolvingUtilizationOfUnsecuredLines'] <= 2)]

st.subheader("📊 Sample Insights")

col1, col2 = st.columns(2)

with col1:
    st.markdown("**Demo Monthly Income Distribution**")
    fig, ax = plt.subplots()
    sns.histplot(example_df['MonthlyIncome'], kde=True, bins=30, color='royalblue', ax=ax)
    st.pyplot(fig)

with col2:
    st.markdown("**Demo Age Distribution**")
    fig, ax = plt.subplots()
    sns.histplot(example_df['age'], kde=True, bins=30, color='darkcyan', ax=ax)
    st.pyplot(fig)

# ------------------- Upload CSV -------------------
st.sidebar.header("📂 Upload Your Data")
uploaded_file = st.sidebar.file_uploader("Upload test CSV file", type=["csv"])

threshold_low = st.sidebar.slider("Low Risk Threshold", min_value=700, max_value=800,
value=750)
threshold_moderate = st.sidebar.slider("Moderate Risk Threshold", min_value=500, max_value=699,
value=600)

if uploaded_file:
    test_df = pd.read_csv(uploaded_file)
    test_df.drop(columns=["Unnamed: 0"], errors='ignore', inplace=True)

    st.subheader("📄 Uploaded Data Preview")
    st.dataframe(test_df.head(), use_container_width=True)

    test_df.fillna(test_df.median(numeric_only=True), inplace=True)
    test_df = test_df[(test_df['age'] > 0) & (test_df['RevolvingUtilizationOfUnsecuredLines'] <= 2)]

    @st.cache_resource
    def load_models():
        return joblib.load("default_classifier.pkl"), joblib.load("score_model.pkl")
    clf_model, reg_model = load_models()

    st.subheader("🔮 Running Predictions")
    features_for_models = test_df.copy()
    test_df['PredictedDefault'] = clf_model.predict(features_for_models)
    test_df['DefaultProbability'] = clf_model.predict_proba(features_for_models)[:, 1]
    test_df['PredictedCreditScore'] = reg_model.predict(features_for_models)

    st.success("✅ Predictions complete!")

    st.subheader("📈 Predicted Results")
    st.dataframe(test_df[['PredictedDefault', 'DefaultProbability', 'PredictedCreditScore']].head(),
use_container_width=True)

    def risk_bucket(score):
        if score >= threshold_low:
            return "Low Risk"
        elif score >= threshold_moderate:
            return "Moderate Risk"
        else:
            return "High Risk"

    test_df["RiskCategory"] = test_df["PredictedCreditScore"].apply(risk_bucket)

    # ------------------- Visual Insights -------------------
    st.subheader("📊 Visual Insights")
    col1, col2 = st.columns(2)

    with col1:
        st.markdown("**📌 Credit Score Distribution**")
        fig, ax = plt.subplots()
        sns.histplot(test_df['PredictedCreditScore'], kde=True, bins=30, color='slateblue', ax=ax)
        st.pyplot(fig)

    with col2:
        st.markdown("**📌 Default Probability by Risk Category**")
        fig, ax = plt.subplots()
        sns.boxplot(x='RiskCategory', y='DefaultProbability', data=test_df, palette='Set2', ax=ax)
        st.pyplot(fig)

    st.markdown("**📌 Credit Score vs Default Probability**")
    fig, ax = plt.subplots()
    sns.scatterplot(x='PredictedCreditScore', y='DefaultProbability', hue='RiskCategory', data=test_df,
alpha=0.7, ax=ax)
    st.pyplot(fig)

    st.markdown("**📌 Default Prediction Breakdown**")
    pie_data = test_df['PredictedDefault'].value_counts()
    fig, ax = plt.subplots()
    ax.pie(pie_data, labels=['No Default', 'Default'], autopct='%1.1f%%', startangle=90,
colors=['#66b3ff','#ff6666'])
    ax.axis('equal')
    st.pyplot(fig)

    # ------------------ Feature Importance ------------------
    st.subheader("🧮 Top Contributing Features")

    if hasattr(clf_model, 'coef_'):
        importances = pd.Series(clf_model.coef_[0], index=features_for_models.columns)
    elif hasattr(clf_model, 'feature_importances_'):
        importances = pd.Series(clf_model.feature_importances_, index=features_for_models.columns)
    else:
        importances = pd.Series(dtype='float64')

    top_features = importances.abs().sort_values(ascending=False).head(10)
    fig, ax = plt.subplots()
    sns.barplot(x=top_features.values, y=top_features.index, palette="viridis", ax=ax)
    ax.set_title("Top Features Affecting Default Prediction")
    st.pyplot(fig)

    # ------------------ Correlation Heatmap ------------------
    st.subheader("📌 Feature Correlation Heatmap")
    corr = test_df[features_for_models.columns].corr()
    fig, ax = plt.subplots(figsize=(10, 6))
    sns.heatmap(corr, annot=True, fmt=".2f", cmap="coolwarm", ax=ax)
    st.pyplot(fig)

    # ------------------ Summary ------------------
    st.subheader("📌 Prediction Summary")
    st.markdown(f"**Total Records Analyzed:** {len(test_df)}")
    st.markdown(f"**Default Rate:** {test_df['PredictedDefault'].mean():.2%}")
    st.markdown(f"**Average Credit Score:** {test_df['PredictedCreditScore'].mean():.2f}")
    st.markdown(f"**High Risk Customers:** {(test_df['RiskCategory'] == 'High Risk').sum()}")

    # ------------------ High Risk Table ------------------
    st.subheader("🚨 Top High Risk Customers")
    st.dataframe(test_df.sort_values("DefaultProbability", ascending=False).head(5),
use_container_width=True)

    # ------------------- Download Section -------------------
    st.subheader("💾 Download Final Results")
    output_df = test_df[["PredictedDefault", "DefaultProbability", "PredictedCreditScore",
"RiskCategory"]]
    st.download_button(
        label="⬇️ Download Predictions as CSV",
        data=output_df.to_csv(index=False).encode('utf-8'),
        file_name="financial_predictions.csv",
        mime="text/csv"
    )

    # ------------------- Model Info -------------------
    with st.expander("ℹ️ About the Model"):
        st.markdown("""
        - **Default Prediction**: Logistic Regression
        - **Credit Score Estimation**: Linear Regression
        - **Features used**: Revolving utilization, age, late payments, debt ratio, monthly income, etc.
        - **Dataset Source**: Kaggle 'Give Me Some Credit'
        """)

else:
    st.info("☝️ Upload your CSV file from the sidebar to begin analysis.")
```

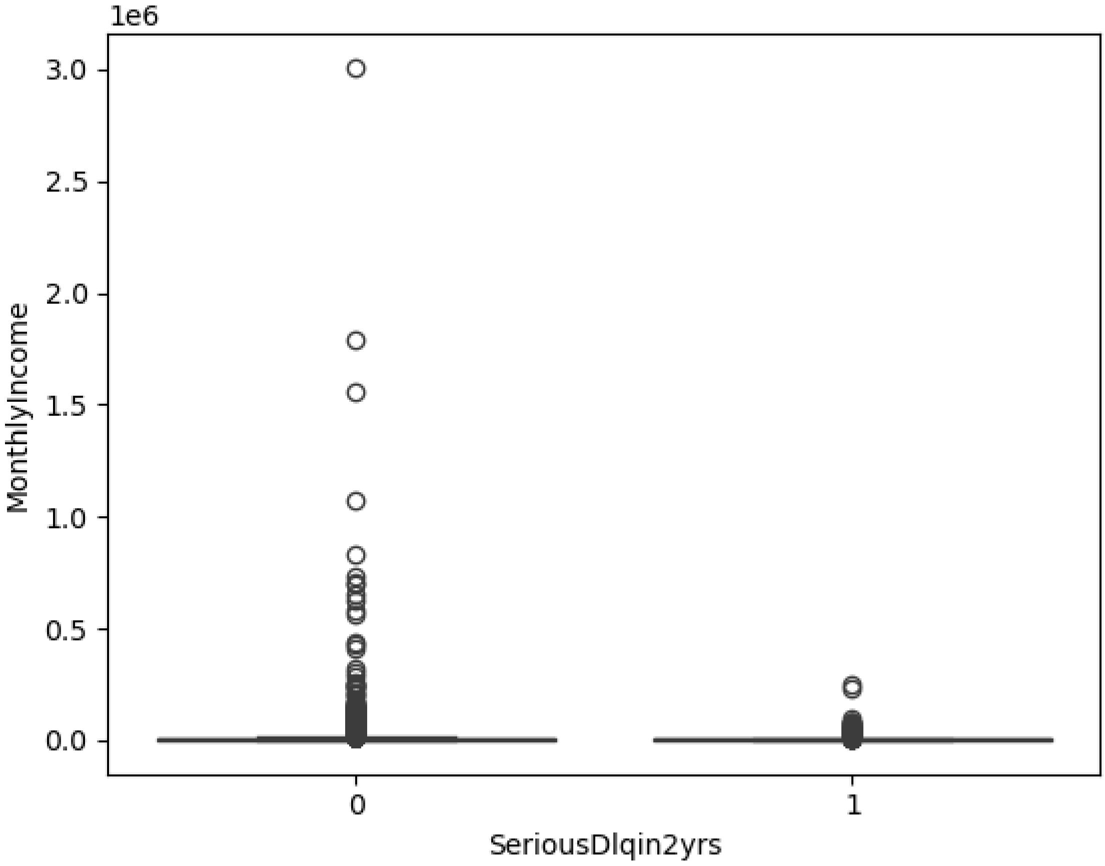
Output Snippets:

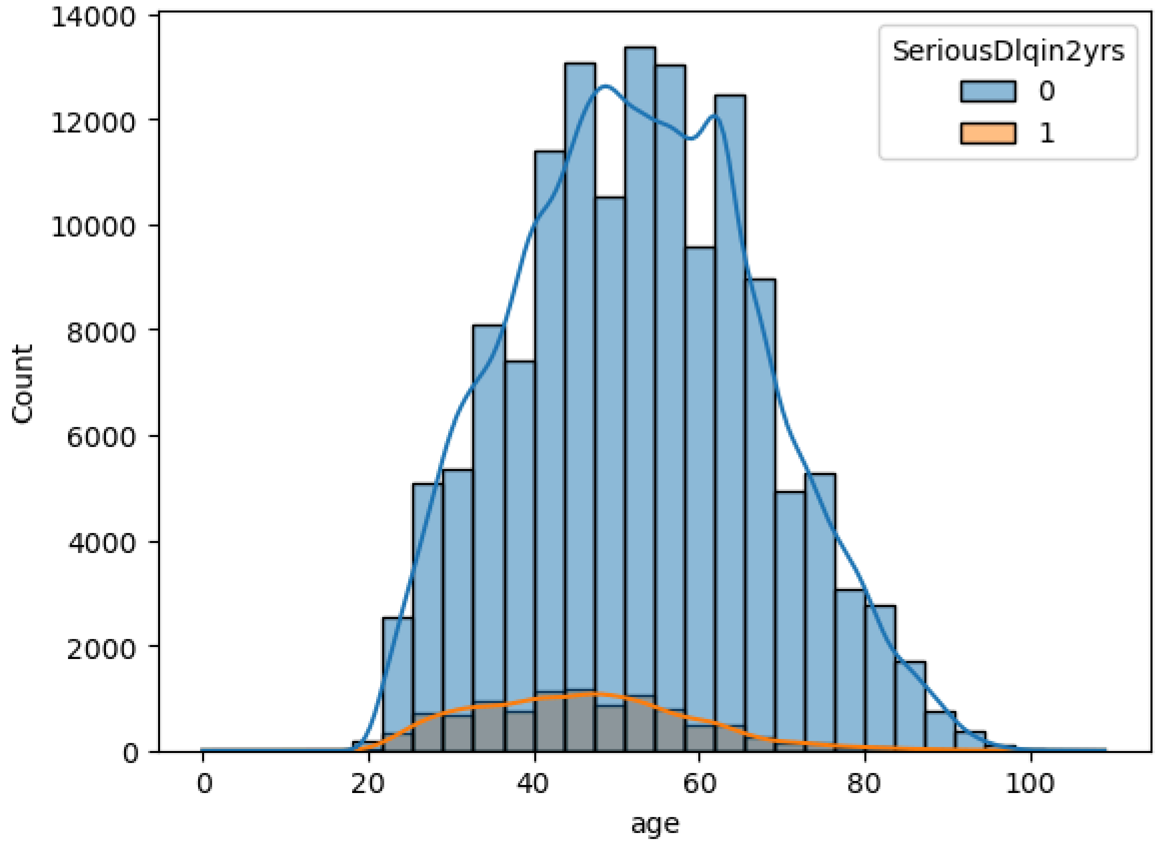
EDA & Elementary Analysis:-

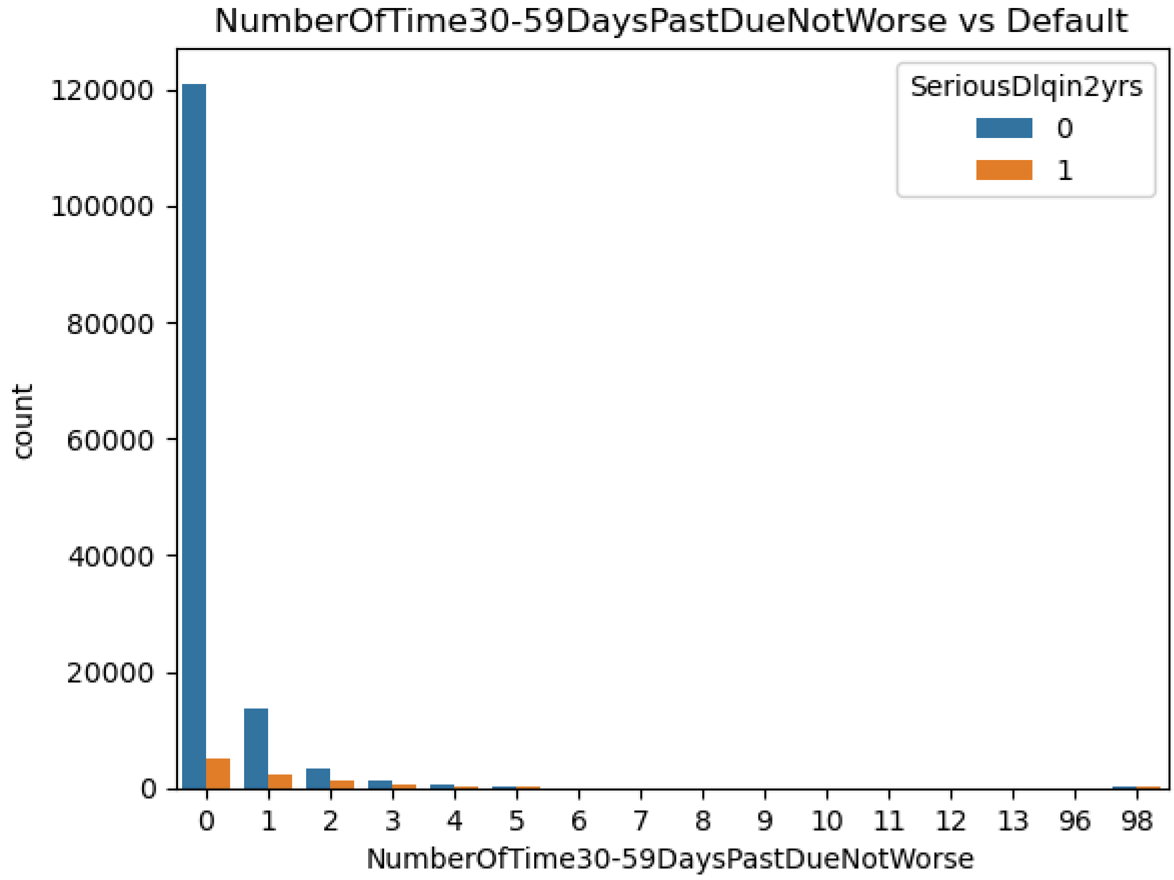


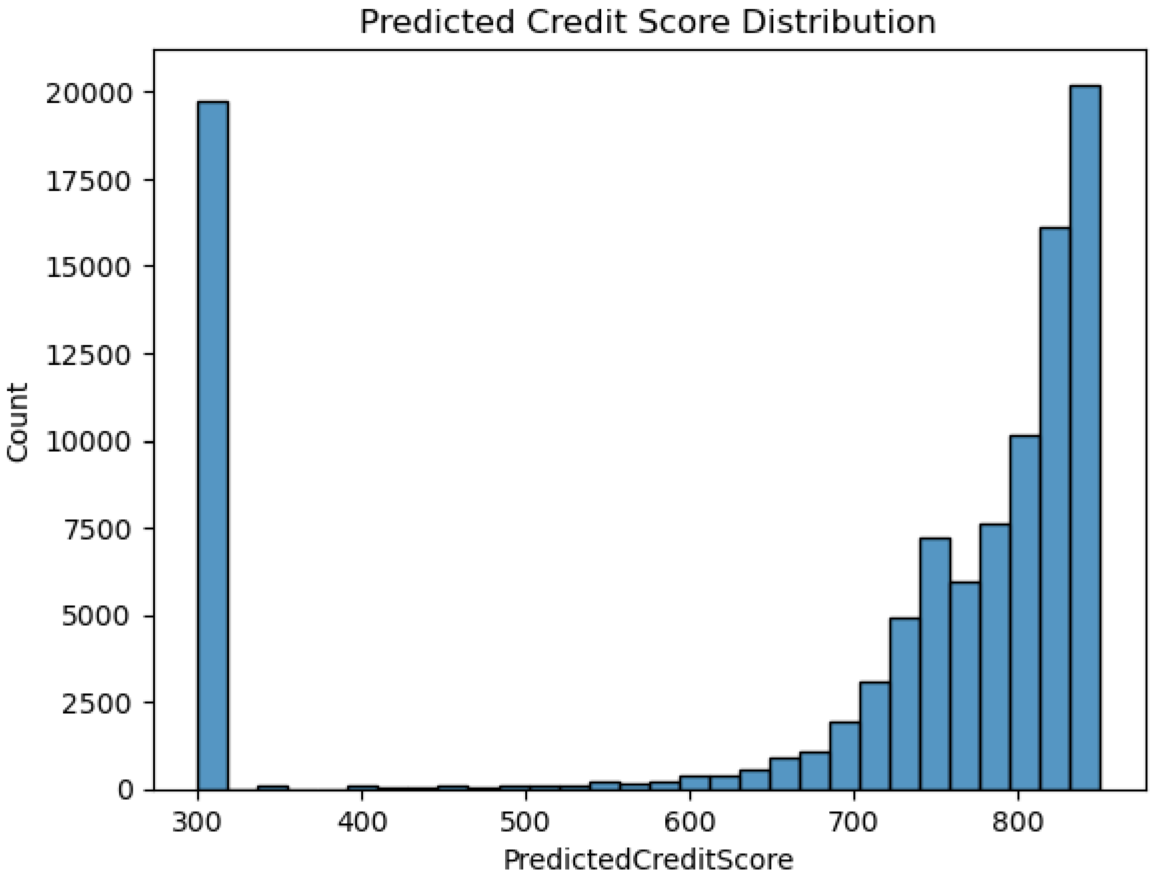


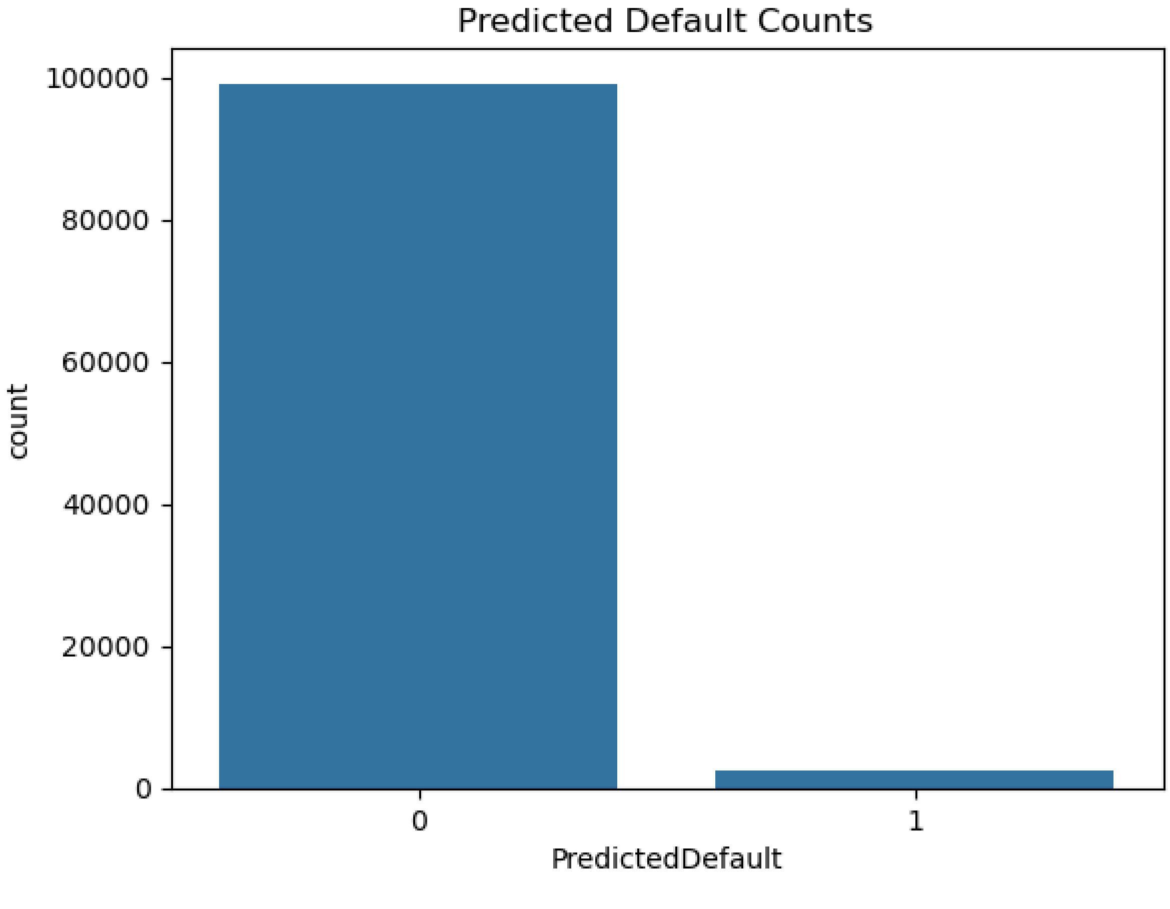

Feature Correlation Heatmap
SeriousDlqin2yrs
RevolvingUtilizationOfUnsecuredLines
age
NumberOfTime30-59DaysPastDueNotWorse
DebtRatio
MonthlyIncome
NumberOfOpenCreditLinesAndLoans
NumberOfTimes90DaysLate
NumberRealEstateLoansOrLines
NumberOfTime60-89DaysPastDueNotWorse
NumberOfDependents
1 -0.0018 -0.12 0.13 -0.0076 -0.02 -0.03 0.12 -0.007 0.1 0.046
-0.0018 1 -0.0059 -0.0013 0.004 0.0071 -0.011 -0.0011 0.0062 -0.001 0.0016
-0.12 -0.0059 1 -0.063 0.024 0.038 0.15 -0.061 0.033 -0.057 -0.21
0.13 -0.0013 -0.063 1 -0.0065 -0.01 -0.055 0.98 -0.031 0.99 -0.0027
-0.0076 0.004 0.024 -0.0065 1 -0.029 0.05 -0.0083 0.12 -0.0075 -0.041
-0.02 0.0071 0.038 -0.01 -0.029 1 0.091 -0.013 0.12 -0.011 0.063
-0.03 -0.011 0.15 -0.055 0.05 0.091 1 -0.08 0.43 -0.071 0.065
0.12 -0.0011 -0.061 0.98 -0.0083 -0.013 -0.08 1 -0.045 0.99 -0.01
-0.007 0.0062 0.033 -0.031 0.12 0.12 0.43 -0.045 1 -0.04 0.12
0.1 -0.001 -0.057 0.99 -0.0075 -0.011 -0.071 0.99 -0.04 1 -0.011
0.046 0.0016 -0.21 -0.0027 -0.041 0.063 0.065 -0.01 0.12 -0.011 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

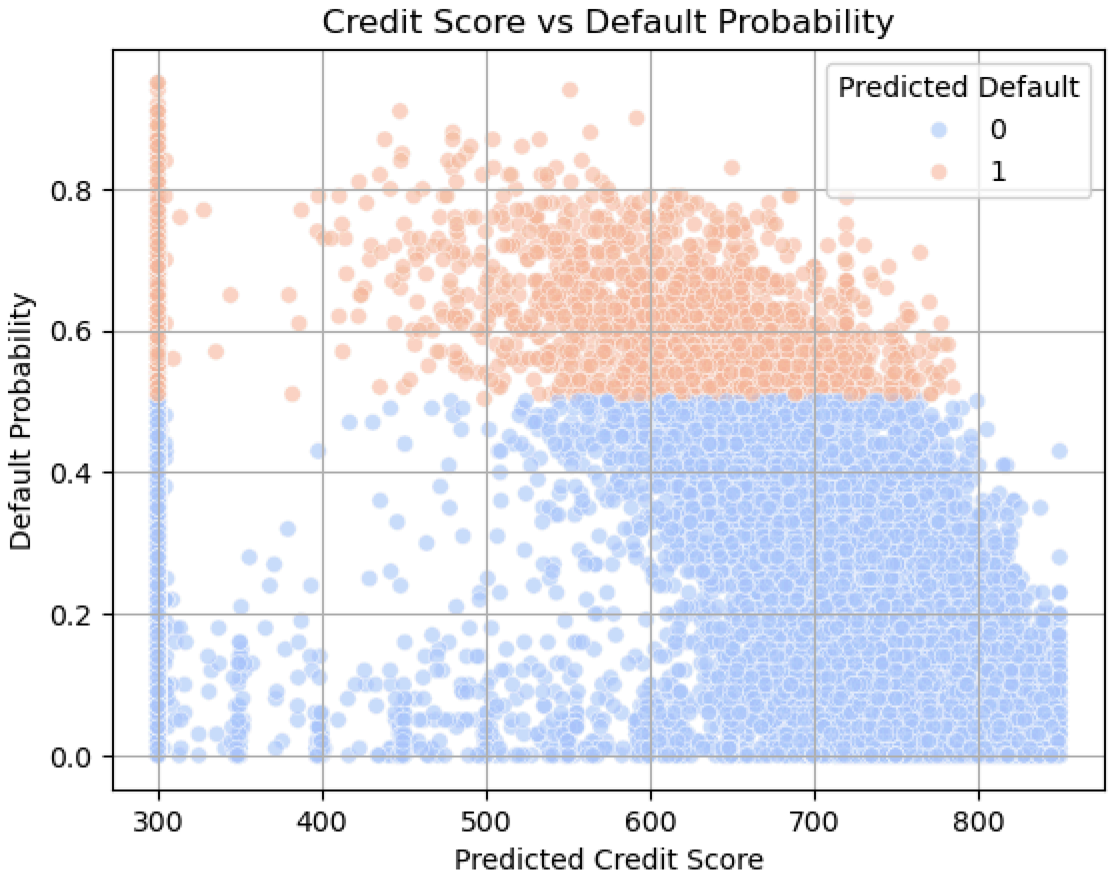

1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
MonthlyIncome
0
1
SeriousDlqin2yrs

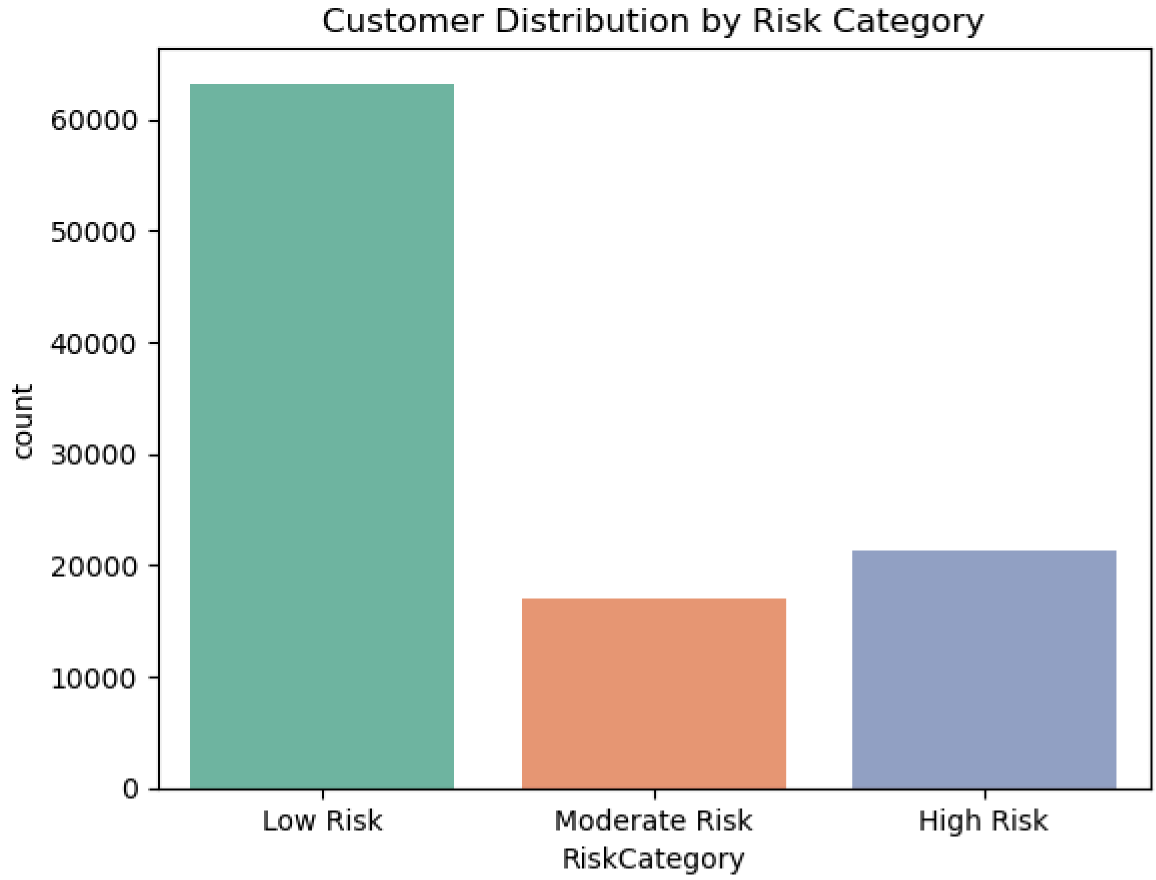

SeriousDlqin2yrs
0
1
Count
14000
12000
10000
8000
6000
4000
2000
0
0
20
40
60
80
100
age

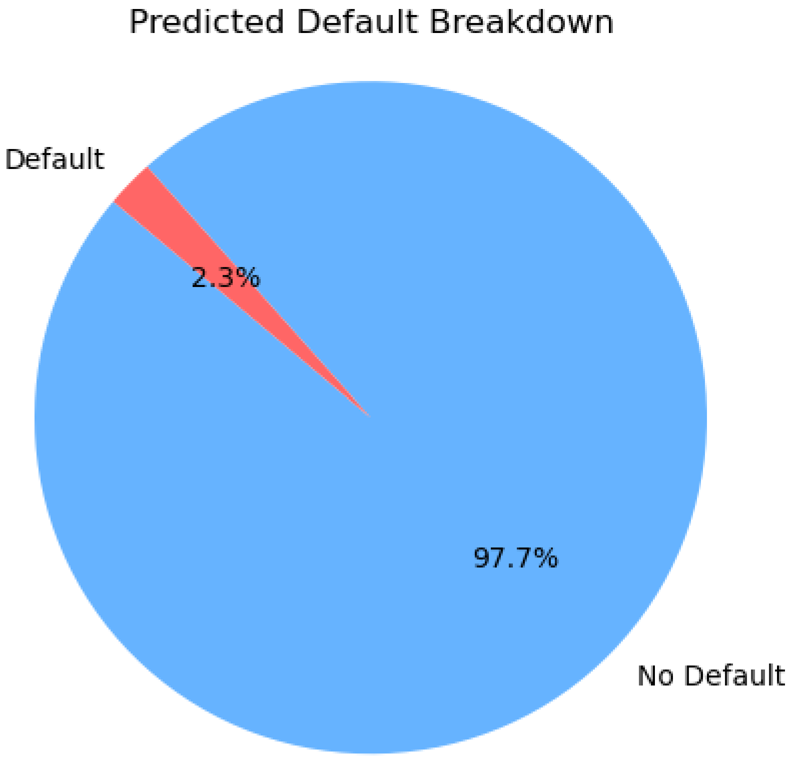

NumberOfTime30-59DaysPastDueNotWorse vs Default
SeriousDlqin2yrs
0
1
count
120000
100000
80000
60000
40000
20000
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
96
98
NumberOfTime30-59DaysPastDueNotWorse

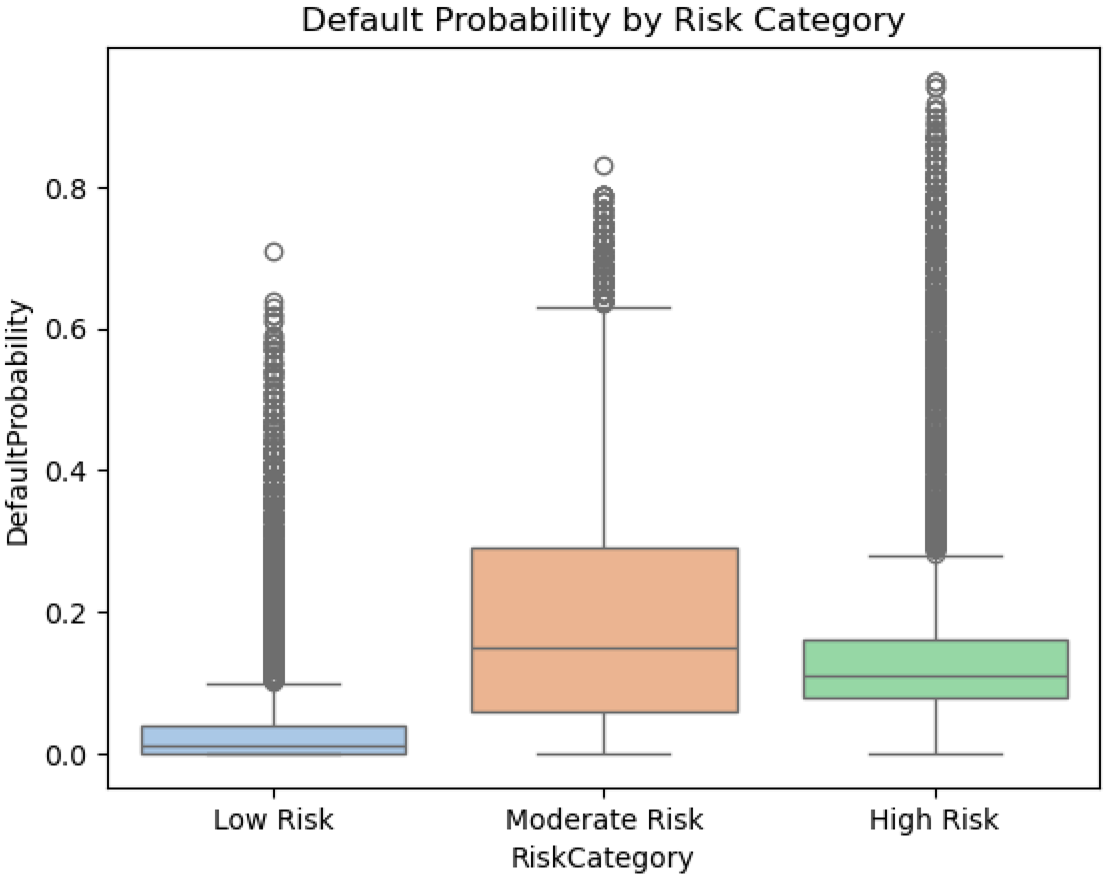

Predicted Credit Score Distribution
Count
20000
17500
15000
12500
10000
7500
5000
2500
0
300
400
500
600
700
800
PredictedCreditScore

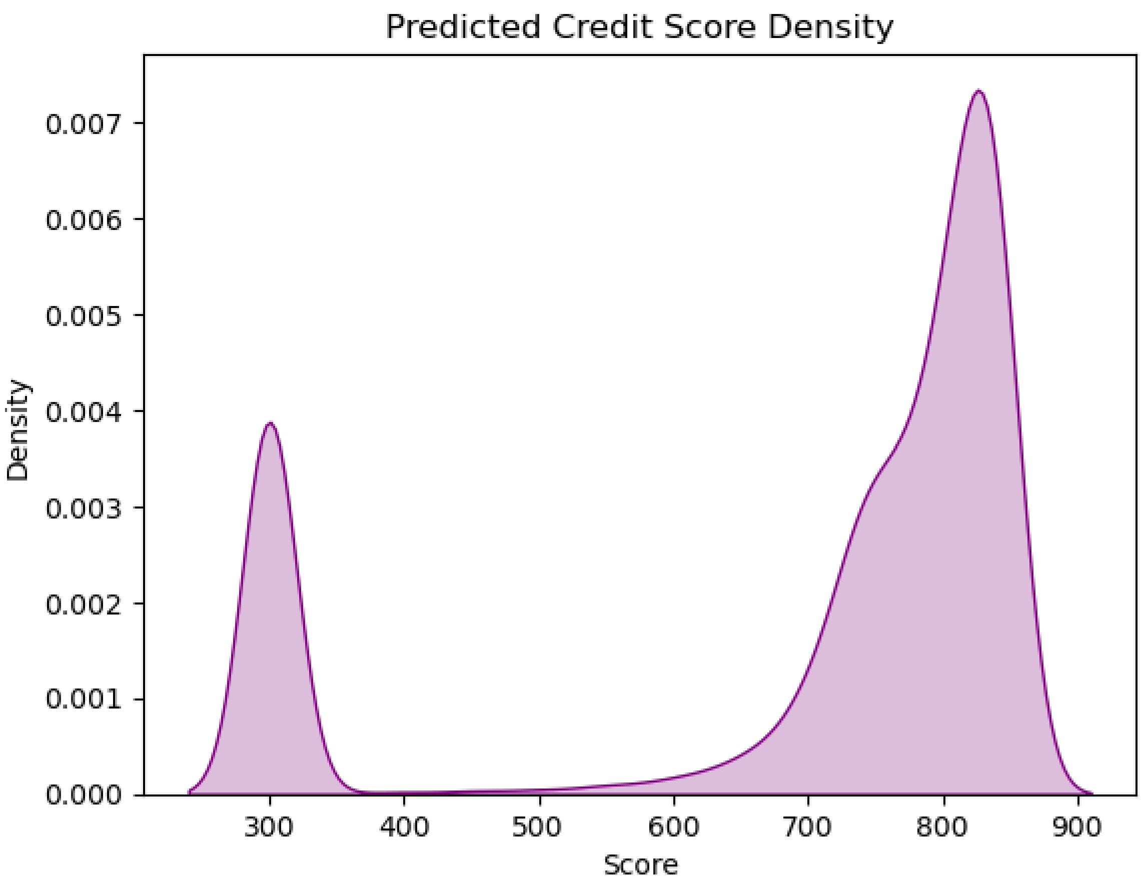

Predicted Default Counts
count
100000
80000
60000
40000
20000
0
0
1
PredictedDefault


Credit Score vs Default Probability
Predicted Default
0
1
Default Probability
0.0
0.2
0.4
0.6
0.8
Predicted Credit Score
300
400
500
600
700
800

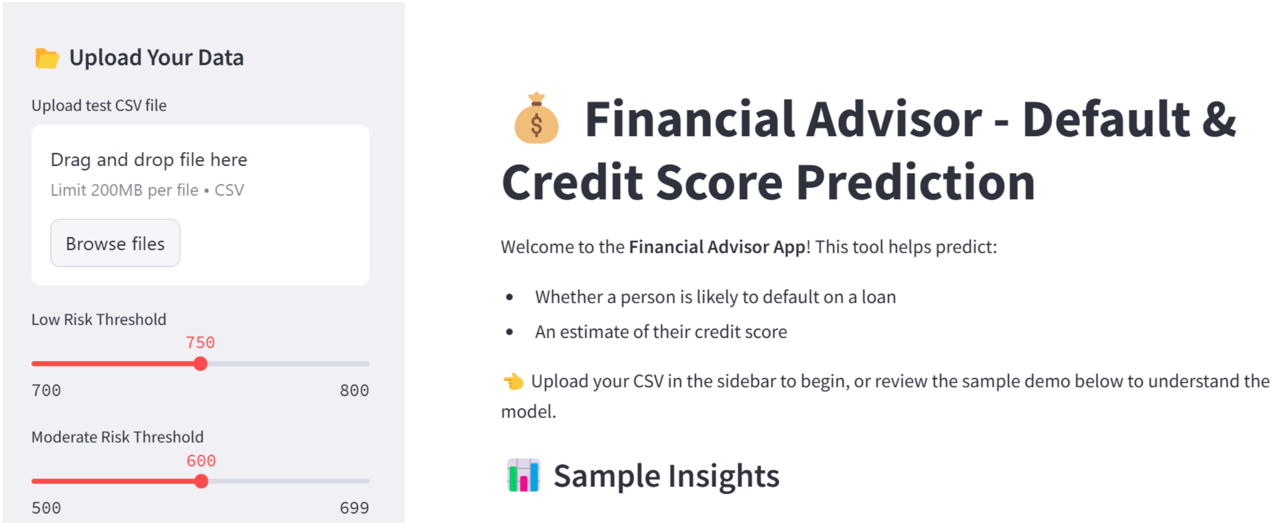

Customer Distribution by Risk Category
count
0
10000
20000
30000
40000
50000
60000
Low Risk
Moderate Risk
High Risk
RiskCategory


Predicted Default Breakdown
Default
2.3%
97.7%
No Default


Default Probability by Risk Category
DefaultProbability
0.8
0.6
0.4
0.2
0.0
Low Risk
Moderate Risk
High Risk
RiskCategory



Streamlit UI :-



📂 Upload Your Data

Upload test CSV file

Drag and drop file here

Limit 200MB per file • CSV

Browse files

Low Risk Threshold: 750 (700–800)

Moderate Risk Threshold: 600 (500–699)

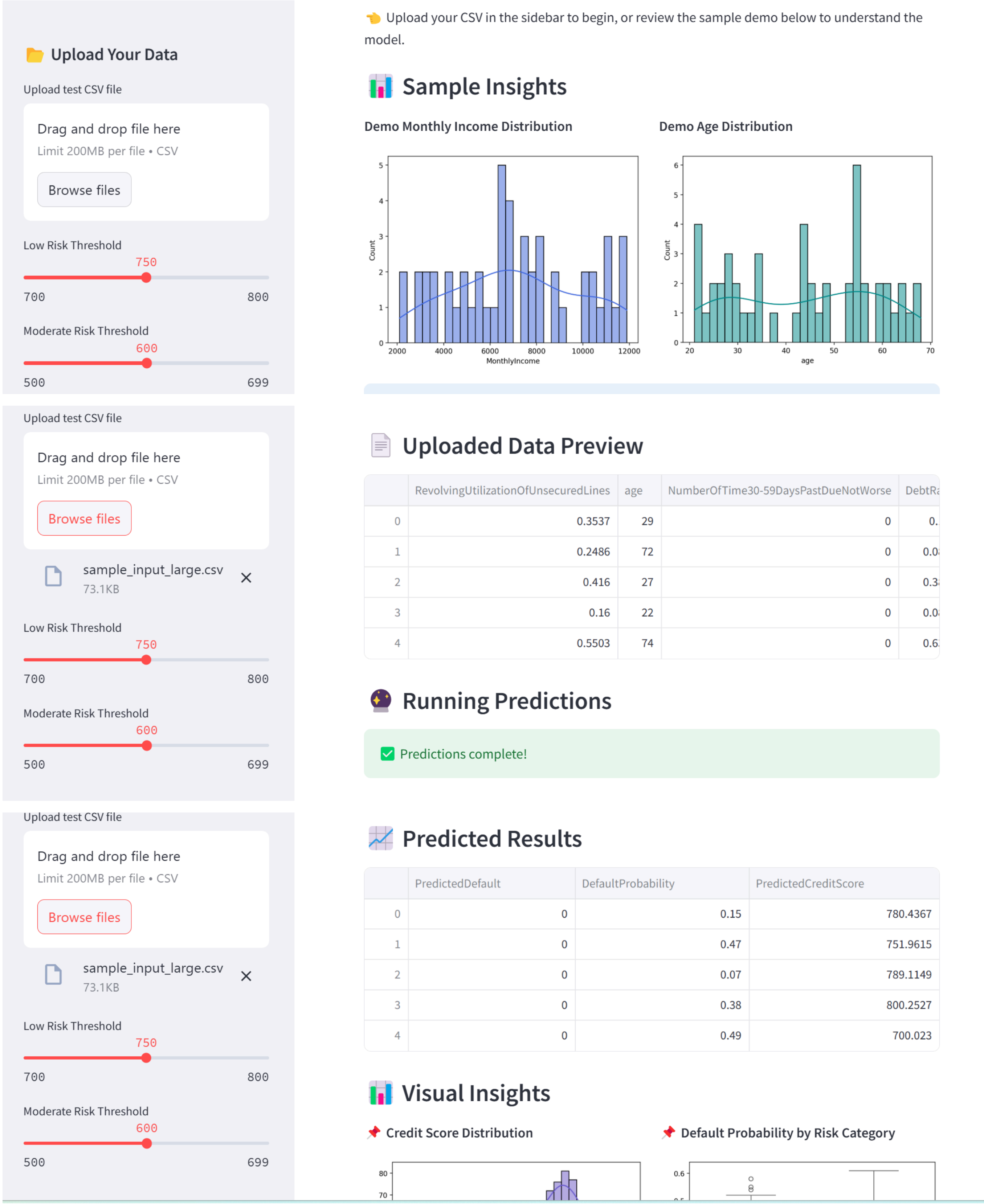
💰 Financial Advisor - Default & Credit Score Prediction

Welcome to the **Financial Advisor App**! This tool helps predict:

- Whether a person is likely to default on a loan
- An estimate of their credit score

👈 Upload your CSV in the sidebar to begin, or review the sample demo below to understand the model.

📊 Sample Insights



📂 Upload Your Data

Upload test CSV file

Drag and drop file here
Limit 200MB per file • CSV

Browse files

Low Risk Threshold
750
700 800

Moderate Risk Threshold
600
500 699

Upload test CSV file

Drag and drop file here
Limit 200MB per file • CSV

Browse files

sample_input_large.csv
73.1KB ×

Low Risk Threshold
750
700 800

Moderate Risk Threshold
600
500 699

Upload test CSV file

Drag and drop file here
Limit 200MB per file • CSV

Browse files

sample_input_large.csv
73.1KB ×

Low Risk Threshold
750
700 800

Moderate Risk Threshold
600
500 699

👈 Upload your CSV in the sidebar to begin, or review the sample demo below to understand the model.

📊 Sample Insights

Demo Monthly Income Distribution

Demo Age Distribution

📄 Uploaded Data Preview

	RevolvingUtilizationOfUnsecuredLines	age	NumberOfTime30-59DaysPastDueNotWorse	DebtRa
0	0.3537	29	0	0.
1	0.2486	72	0	0.0
2	0.416	27	0	0.3
3	0.16	22	0	0.0
4	0.5503	74	0	0.6

🔮 Running Predictions

✅ Predictions complete!

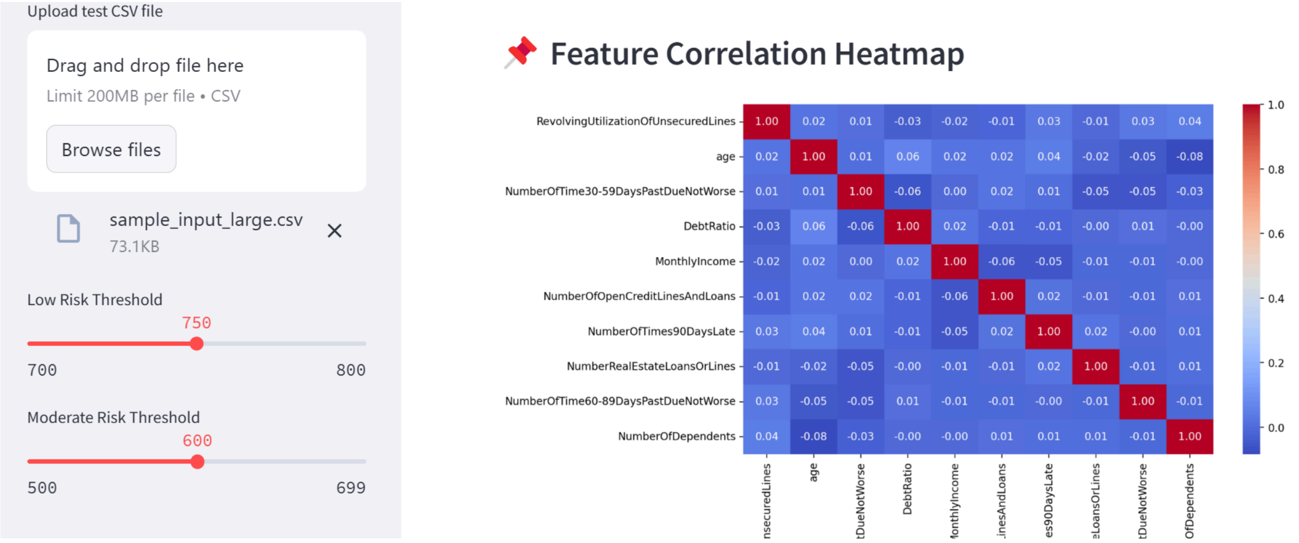
📈 Predicted Results

	PredictedDefault	DefaultProbability	PredictedCreditScore
0	0	0.15	780.4367
1	0	0.47	751.9615
2	0	0.07	789.1149
3	0	0.38	800.2527
4	0	0.49	700.023

📊 Visual Insights

📌 **Credit Score Distribution**

📌 **Default Probability by Risk Category**



Upload test CSV file

Drag and drop file here

Limit 200MB per file • CSV

Browse files

sample_input_large.csv
73.1KB

Low Risk Threshold

750

700 800

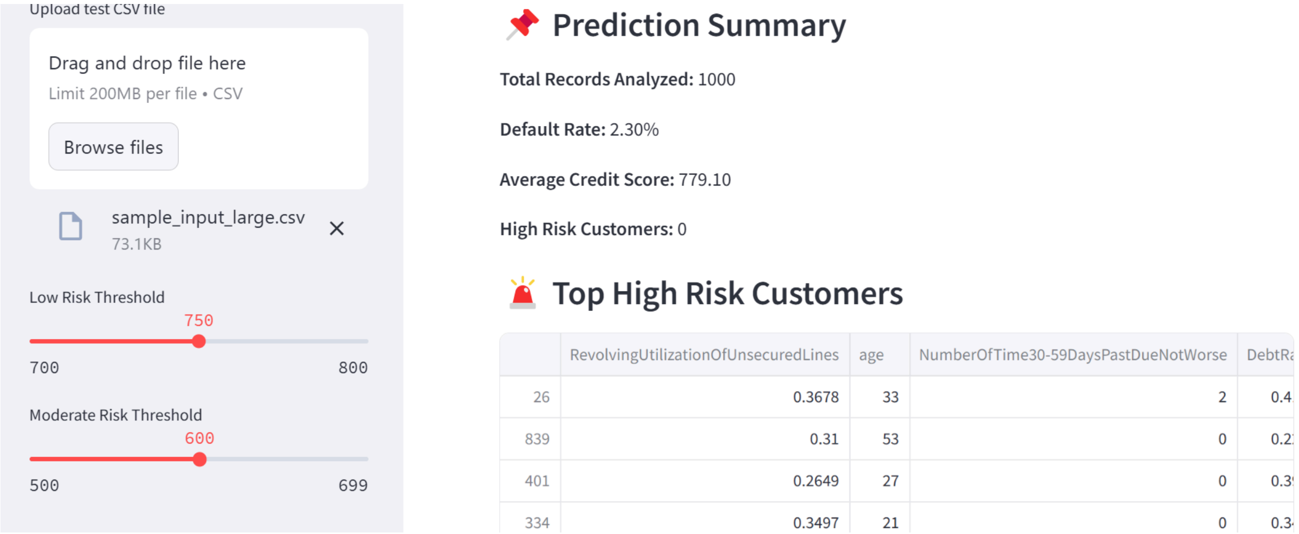
Moderate Risk Threshold

600

500 699

📌 Feature Correlation Heatmap

	nsecuredLines	age	tDueNotWorse	DebtRatio	MonthlyIncome	inesAndLoans	es90DaysLate	eLoansOrLines	tDueNotWorse	OfDependents
RevolvingUtilizationOfUnsecuredLines	1.00	0.02	0.01	-0.03	-0.02	-0.01	0.03	-0.01	0.03	0.04
age	0.02	1.00	0.01	0.06	0.02	0.02	0.04	-0.02	-0.05	-0.08
NumberOfTime30-59DaysPastDueNotWorse	0.01	0.01	1.00	-0.06	0.00	0.02	0.01	-0.05	-0.05	-0.03
DebtRatio	-0.03	0.06	-0.06	1.00	0.02	-0.01	-0.01	-0.00	0.01	-0.00
MonthlyIncome	-0.02	0.02	0.00	0.02	1.00	-0.06	-0.05	-0.01	-0.01	-0.00
NumberOfOpenCreditLinesAndLoans	-0.01	0.02	0.02	-0.01	-0.06	1.00	0.02	-0.01	-0.01	0.01
NumberOfTimes90DaysLate	0.03	0.04	0.01	-0.01	-0.05	0.02	1.00	0.02	-0.00	0.01
NumberRealEstateLoansOrLines	-0.01	-0.02	-0.05	-0.00	-0.01	-0.01	0.02	1.00	-0.01	0.01
NumberOfTime60-89DaysPastDueNotWorse	0.03	-0.05	-0.05	0.01	-0.01	-0.01	-0.00	-0.01	1.00	-0.01
NumberOfDependents	0.04	-0.08	-0.03	-0.00	-0.00	0.01	0.01	0.01	-0.01	1.00



Upload test CSV file

Drag and drop file here

Limit 200MB per file • CSV

Browse files

sample_input_large.csv
73.1KB

Low Risk Threshold

750

700 800

Moderate Risk Threshold

600

500 699

📌 Prediction Summary

Total Records Analyzed: 1000

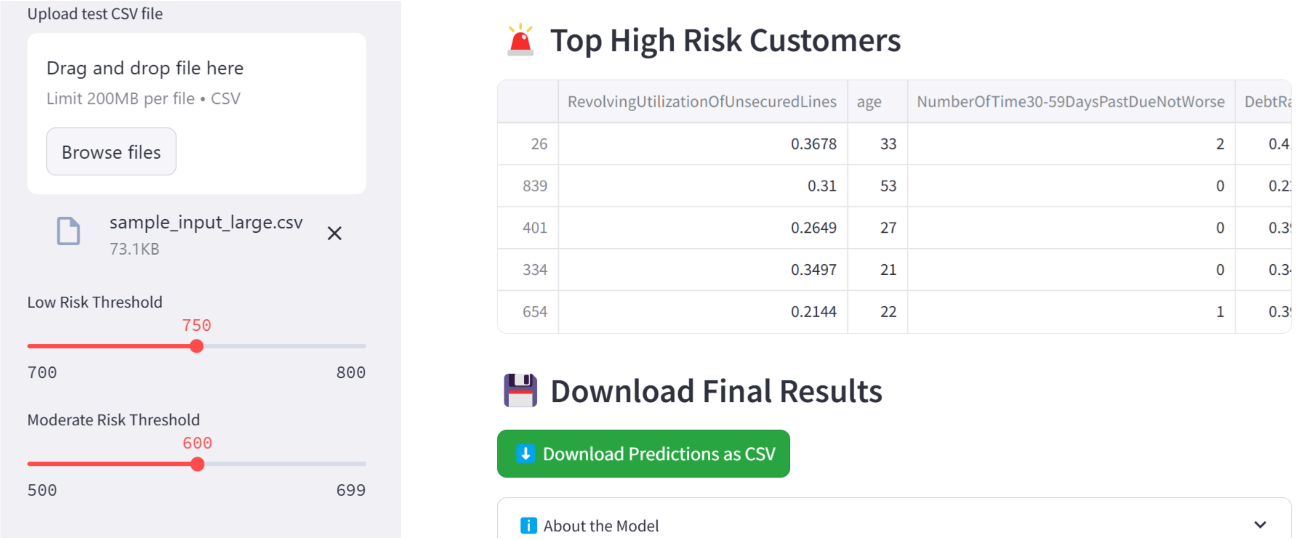
Default Rate: 2.30%

Average Credit Score: 779.10

High Risk Customers: 0

🚨 Top High Risk Customers

	RevolvingUtilizationOfUnsecuredLines	age	NumberOfTime30-59DaysPastDueNotWorse	DebtRa
26	0.3678	33	2	0.4
839	0.31	53	0	0.2
401	0.2649	27	0	0.3
334	0.3497	21	0	0.3



Upload test CSV file

Drag and drop file here

Limit 200MB per file • CSV

Browse files

sample_input_large.csv
73.1KB

Low Risk Threshold

750

700 800

Moderate Risk Threshold

600

500 699

🚨 Top High Risk Customers

	RevolvingUtilizationOfUnsecuredLines	age	NumberOfTime30-59DaysPastDueNotWorse	DebtRa
26	0.3678	33	2	0.4
839	0.31	53	0	0.2
401	0.2649	27	0	0.3
334	0.3497	21	0	0.3
654	0.2144	22	1	0.3

💾 Download Final Results

Chapter 12: Outcomes

The **Financial Default Predictor** project successfully delivered a machine learning-powered web application capable of predicting loan default probability and estimating credit scores based on user-uploaded financial data. The system offers clear, interpretable, and interactive outputs, demonstrating how predictive modeling can be applied in the financial domain to aid in decision-making.

The key outcomes are as follows:

1. **Functioning ML Models**

○ A Logistic Regression model that accurately classifies users as defaulters or non-defaulters

○ A Linear Regression model that predicts a synthetic credit score with reasonable accuracy

2. **Streamlit Web Application**

○ Fully functional UI to accept, process, and analyze financial records

○ Risk categorization and detailed prediction reports

3. **Interactive Visualizations**

○ Real-time analytics for financial attributes, correlation plots, and risk distributions

4. **Modular Codebase**

○ Easily extendable architecture with clear separation between training and deployment logic

5. **Practical Utility**

○ Provides a valuable learning tool for students, a decision-support system for advisors, and a risk estimator for individuals

6. **Minimal Resource Requirement**

○ Can be deployed on free hosting platforms or run locally with low compute power

7. **Academic Achievement**

○ Demonstrates a full cycle of machine learning application—from raw data to deployed interface

This project not only met its technical objectives but also prepared the developer for real-world work in domains such as fintech, data analytics, and intelligent systems development.

Chapter 13: Conclusion

The **Financial Default Predictor** is a significant demonstration of how machine learning can be used to derive actionable insights in the finance industry. This project brings together data analysis, statistical modeling, and user-focused application development in a cohesive workflow.

The core achievement lies in translating a complex process—credit risk assessment—into a system that is both interpretable and accessible. By training predictive models on real financial data and integrating them into a Streamlit interface, the tool serves as a proof-of-concept for intelligent financial advisors or automated loan screening systems.

From a technical standpoint, the project strengthened understanding of:

- Data preprocessing techniques
- Feature engineering for financial datasets
- Supervised learning models (logistic and linear regression)
- Web app deployment with Streamlit
- Presenting analytical results in a digestible visual form

From a practical perspective, the system empowers users to upload data, explore predictions, and make informed decisions—all without needing prior data science expertise. Its modularity ensures that the tool can be extended with more sophisticated models, additional features, or security/authentication layers as needed.

Overall, the **Financial Default Predictor** stands as a robust academic project that bridges the gap between theoretical machine learning knowledge and practical financial problem-solving. It lays a strong foundation for future work in AI-powered financial applications and provides a meaningful contribution to the growing intersection of technology and finance.

Chapter 14: References

1. Kaggle Dataset: “Give Me Some Credit”
https://www.kaggle.com/datasets/c/GiveMeSomeCredit

2. Scikit-learn Documentation
https://scikit-learn.org/stable/documentation.html

3. Streamlit Documentation
https://docs.streamlit.io/

4. Seaborn Documentation
https://seaborn.pydata.org/

5. Matplotlib Documentation
https://matplotlib.org/stable/contents.html

6. Pandas Documentation
https://pandas.pydata.org/

7. NumPy Documentation
https://numpy.org/

8. Joblib Documentation
https://joblib.readthedocs.io/

9. Python Official Documentation
https://docs.python.org/3/

10. Visual Studio Code
https://code.visualstudio.com/