INSTITUTE FOR ADVANCED COMPUTING

AND SOFTWARE DEVELOPMENT

AKURDI, PUNE – 411044

Documentation On

“Loan Default Prediction using ML and PySpark”

PG-DBDA FEB 2025

Submitted By:

Group No: 15

Aniket Wani (250241225008)
Shreyas Todkar (250241225052)

Dr. Shantanu Pathak
Project Guide

Mr. Prashant Deshpande
Centre Coordinator

DECLARATION

I, the undersigned hereby declare that the project report titled " Loan Default Prediction Using ML and PySpark " written and submitted by me to Institute For Advanced Computing And Software Development Akurdi Pune, in the fulfilment of requirement for the award of degree of Post Graduate Diploma in Big Data Analytics (PG DBDA) under the guidance of Dr. Shantanu Pathak. It is my original work I have not copied any code or content from any source without proper attribution, and I have not allowed anyone else to copy my work. The project was completed using Python and ML and libraries. The project was developed as part of my academic coursework. I also confirm that the project is original, and it has not been submitted previously for any other academic or professional purpose.

Place: Signature:

Date: Name: Aniket Wani / Shreyas Todkar

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to Dr. Shantanu Pathak, Project Guide, for providing me with the guidance and support to complete this academic project. His valuable insights, expertise and encouragement have been instrumental in the success of this project .

I would also like to thank my fellow classmates for their support and cooperation during the project. Their feedback and suggestions were helpful in improving the quality of the project.

I would like to extend my gratitude to Mr. Prashant Deshpande, Centre Coordinator, for providing me with the necessary resources and facilities to complete this project. Their support has been crucial in the timely completion of this project.

Finally, I would like to thank my family and friends for their constant encouragement and support throughout the project. Their belief in me has been a constant source of motivation and inspiration.

Thank you all for your support and guidance in completing this academic project.

ABSTRACT

Loan default prediction is crucial for financial institutions to minimize risks and optimize lending decisions. This project leverages Machine Learning (ML) and Apache Spark to develop a scalable and accurate predictive model for identifying potential loan defaulters.

The model utilizes historical loan data, including demographic details, financial behavior, and the form of transaction records. **Data preprocessing, feature engineering, and exploratory data analysis (EDA), ONNX conversion** are performed using Spark MLlib for efficient large-scale processing. Multiple machine learning algorithms, including **Logistic Regression, Random Forest, Support Vector Machine, XGBoost algorithms** are trained and evaluated using metrics like accuracy, precision, recall and classification report.

Table Of Contents

Table of Figure:

CHAPTER 1
1. INTRODUCTION

Loan default prediction is a critical task in the financial sector to assess credit risk and minimize losses. This project leverages **machine learning** and **Apache Spark** to analyze borrower attributes and loan an details for predicting defaults. The dataset includes financial indicators such as **age, income, loan amount, credit score, monthsemployed, interest rate, debt-to-income ratio and loanterm**. By utilizing scalable data processing with Spark and advanced predictive models, this project aims to enhance risk management and fraud detection, decision-making in lending institutions. So ultimately improving loan approval processes and reducing bad debt.

1.1Problem Statement

Financial institutions face significant losses due to borrowers failing to repay loans on time, commonly referred to as loan defaults. Accurately predicting the likelihood of default before loan approval can help minimize risk, improve decision-making, and maintain financial stability. This project aims to develop a machine learning–based predictive model that analyzes borrower information such as income, credit history, employment status, and loan amount to classify applicants as likely defaulters or non-defaulters. The outcome will enable lenders to make informed decisions, reduce bad debts, and enhance portfolio quality.

1.2 Scope

This project covers the **design, development, and deployment** of a machine learning-based loan defaulter prediction system. The scope includes:

- **Data Collection & Preprocessing**: Sourcing data related to demographics, financial history, and behavioural patterns of loan applicants.
- **Feature Engineering**: Identifying key attributes influencing default risk, including employment history, credit score, and debt-to-income ratio.
- **Model Selection & Training**: Implementing and comparing different classification algorithms, with **XGBoost** selected as the best-performing model.
- **Regulatory Compliance**: Ensuring model fairness, avoiding biases, and maintaining compliance with financial regulations.
- **Deployment & Integration**: Integrating the model into a **loan approval system** for real-time decision-making.

The solution is designed for **banks, NBFCs (Non-Banking Financial Companies), and fintech lenders** looking to optimize their risk management processes.

1.3 Aim & Objective

The primary aim of this project is to develop an **accurate and interpretable machine learning model** for predicting loan defaults. This will help financial institutions make informed lending decisions, minimize risks, and improve operational efficiency.

To achieve the project aim, the following objectives are outlined:

1. **Data Acquisition & Preparation**: Gather loan application data, preprocess missing values, and perform exploratory data analysis.
2. **Feature Engineering**: Extract and select the most influential features impacting loan default predictions.
3. **Model Development & Evaluation**: Train multiple classification models and select the best-performing one (**XGBoost**).
4. **Performance Optimization**: Fine-tune the model for accuracy, precision, recall, and F1-score.
5. **Regulatory & Ethical Considerations**: Ensure fairness in model predictions, avoiding biases against demographic groups.
6. **Deployment & Implementation**: Integrate the trained model into a real-world loan processing system for automated decision-making.

CHAPTER 2
2.Project Description

2.1 Project Work Flow

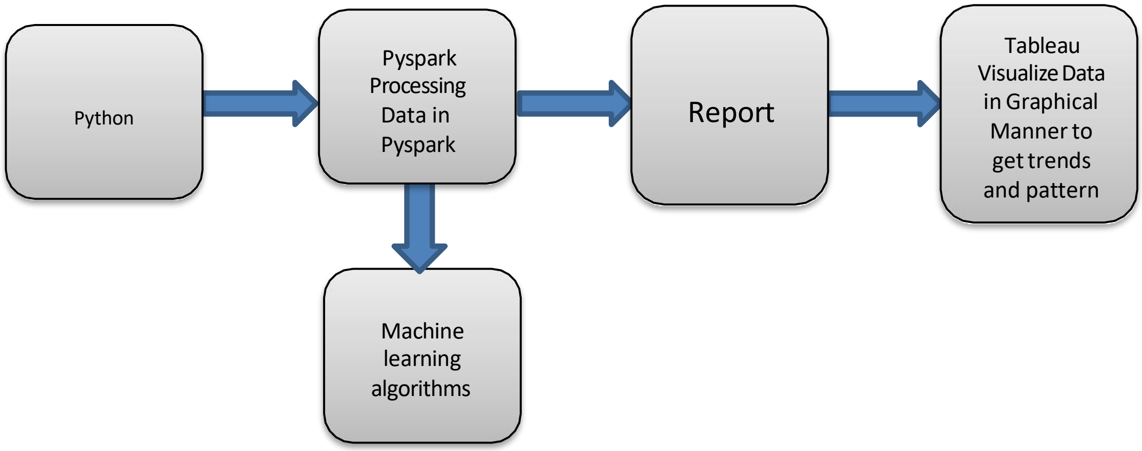

Python
Pyspark Processing Data in Pyspark
Report
Tableau Visualize Data in Graphical Manner to get trends and pattern
Machine learning algorithms

Fig 1: Project work flow diagram

2.2 Data Collection

The dataset was sourced from **Kaggle** and consists of **451388 rows and 29 columns**, covering demographic, financial, and behavioral attributes of loan applicants.

Data Characteristics

- **10 Numeric Columns**: Includes income, loan amount, interest rate, and credit score.
- **19 Categorical Columns**: Includes employment type, education level, loan purpose, and marital status.

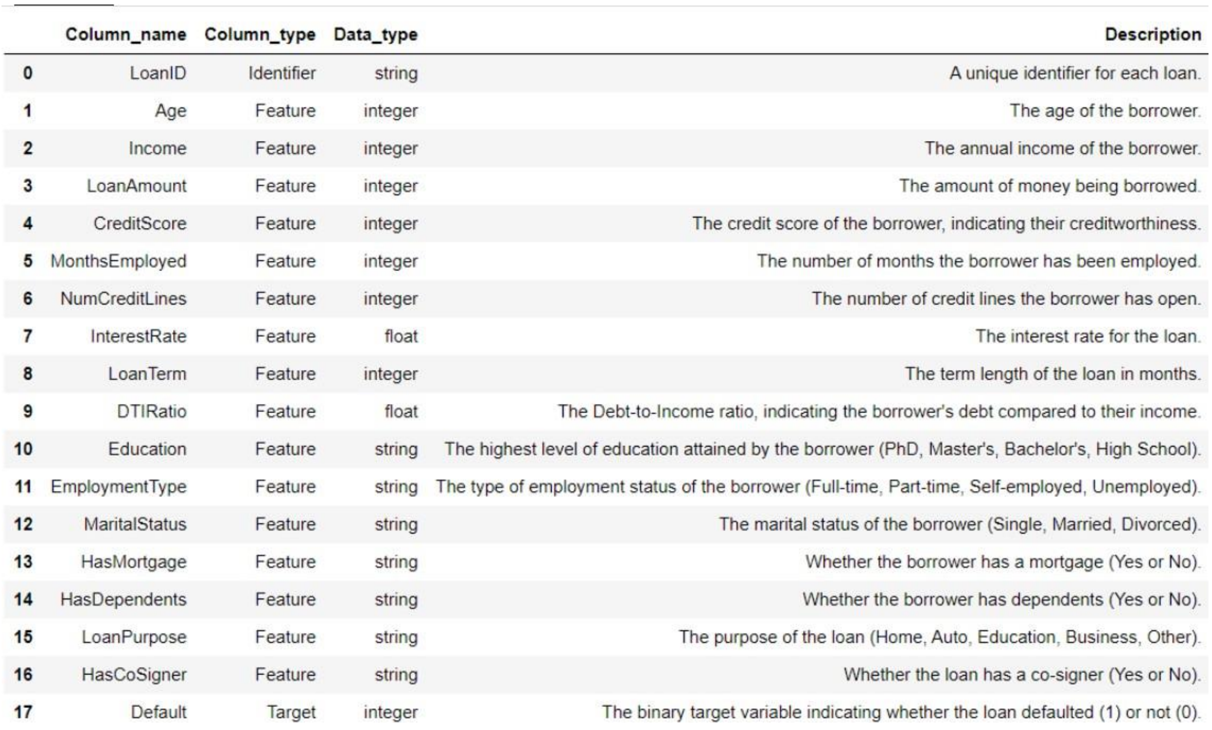
Handling Imbalanced Data

- **Synthetic Data Generation**: Minority class (defaulters) was balanced by duplicating rows.
- **Downsampling**: Majority class (non-defaulters) was reduced to prevent bias.

Preprocessing steps such as missing value handling, feature encoding, and normalization were applied to ensure data quality.

2.3 Studying the data

This dataset has been taken from Coursera's Loan Default Prediction Challenge and will provide you the opportunity to tackle one of the most industry-relevant machine learning problems with a unique dataset that will put your modeling skills to the test. The dataset contains 451,388 rows and 29 columns in total.



	Column_name	Column_type	Data_type	Description
0	LoanID	Identifier	string	A unique identifier for each loan.
1	Age	Feature	integer	The age of the borrower.
2	Income	Feature	integer	The annual income of the borrower.
3	LoanAmount	Feature	integer	The amount of money being borrowed.
4	CreditScore	Feature	integer	The credit score of the borrower, indicating their creditworthiness.
5	MonthsEmployed	Feature	integer	The number of months the borrower has been employed.
6	NumCreditLines	Feature	integer	The number of credit lines the borrower has open.
7	InterestRate	Feature	float	The interest rate for the loan.
8	LoanTerm	Feature	integer	The term length of the loan in months.
9	DTIRatio	Feature	float	The Debt-to-Income ratio, indicating the borrower's debt compared to their income.
10	Education	Feature	string	The highest level of education attained by the borrower (PhD, Master's, Bachelor's, High School).
11	EmploymentType	Feature	string	The type of employment status of the borrower (Full-time, Part-time, Self-employed, Unemployed).
12	MaritalStatus	Feature	string	The marital status of the borrower (Single, Married, Divorced).
13	HasMortgage	Feature	string	Whether the borrower has a mortgage (Yes or No).
14	HasDependents	Feature	string	Whether the borrower has dependents (Yes or No).
15	LoanPurpose	Feature	string	The purpose of the loan (Home, Auto, Education, Business, Other).
16	HasCoSigner	Feature	string	Whether the loan has a co-signer (Yes or No).
17	Default	Target	integer	The binary target variable indicating whether the loan defaulted (1) or not (0).

Fig 2 : Features Descriptions

Before model training, an in-depth analysis of the dataset was conducted to understand patterns, correlations, and potential challenges.

Key Steps in Data Exploration

- Descriptive Statistics: Summary statistics were generated to analyze distributions, central tendencies, and outliers.
- Class Distribution: The dataset was highly imbalanced, requiring balancing techniques.
- Feature Correlation: Heatmaps and correlation matrices helped identify relationships between numerical variables.
- Missing Data Analysis: Checked for null values and applied appropriate imputation techniques.

- Data Visualization: Plots such as histograms, box plots, and bar charts were used to identify trends and anomalies.

This analysis provided key insights, guiding feature engineering and model selection for optimal performance.

2.4 Studying the Model

To ensure accurate loan default predictions, various machine learning models were analyzed, focusing on performance, interpretability, and computational efficiency.

Model Selection & Justification

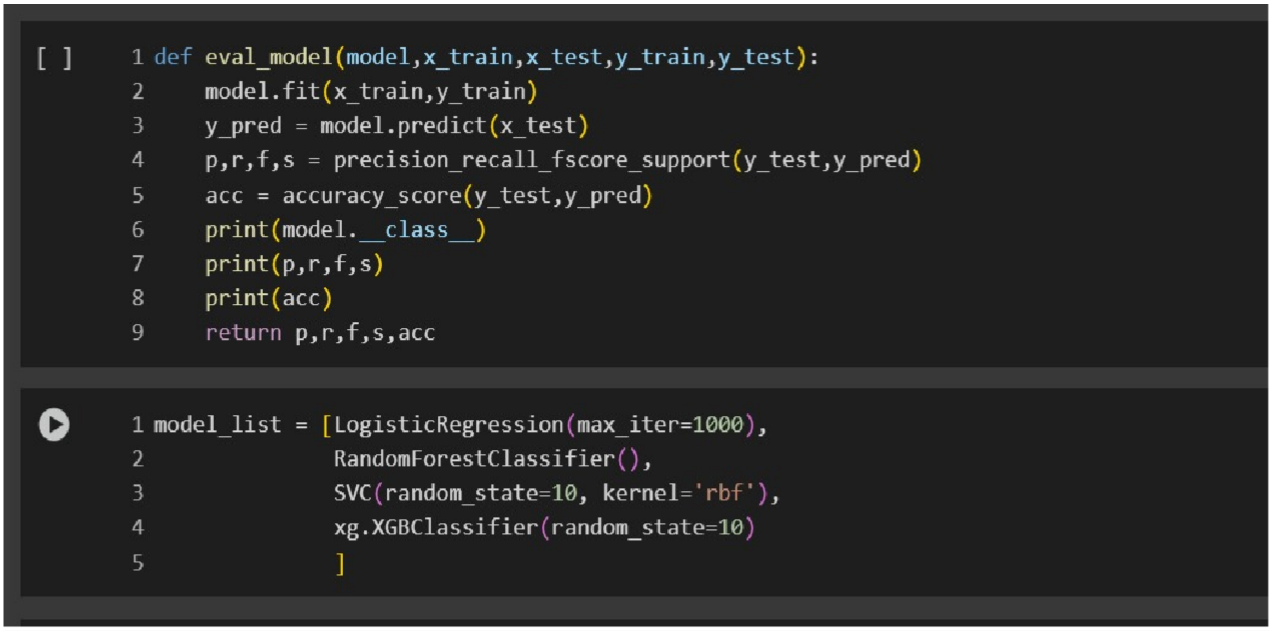
- **Random Forest:** Chosen for its ability to handle both numerical and categorical data, robustness against overfitting, and high accuracy in classification tasks.

Model Evaluation Metrics

- Accuracy & Precision: To measure correct classifications.
- Recall & F1-score: To balance performance on imbalanced data.
- ROC-AUC Curve: To assess model discrimination capability.

Hyperparameter tuning and cross-validation were performed to enhance performance, ensuring reliable and data-driven predictions.

2.5 Implementing the Model



```
def eval_model(model,x_train,x_test,y_train,y_test):
    model.fit(x_train,y_train)
    y_pred = model.predict(x_test)
    p,r,f,s = precision_recall_fscore_support(y_test,y_pred)
    acc = accuracy_score(y_test,y_pred)
    print(model.__class__)
    print(p,r,f,s)
    print(acc)
    return p,r,f,s,acc
```

```
model_list = [LogisticRegression(max_iter=1000),
              RandomForestClassifier(),
              SVC(random_state=10, kernel='rbf'),
              xg.XGBClassifier(random_state=10)
              ]
```

Fig 3: Code for Model Implementation

Multiple machine learning models were implemented to predict loan defaults, ensuring a comprehensive evaluation of different algorithms. The models used include XGBoost, Random Forest, Logistic Regression, Support Vector Machine.

Each model was trained on the dataset, with XGBoost utilizing an evaluation set for early stopping to prevent overfitting. Performance was assessed using classification reports and accuracy scores, allowing a comparative analysis of model effectiveness.

Hyperparameters such as iterations, estimators, max depth, and solvers were optimized to enhance model accuracy. The results guided the selection of the best-performing model for loan default prediction.

2.6 Validating the Model

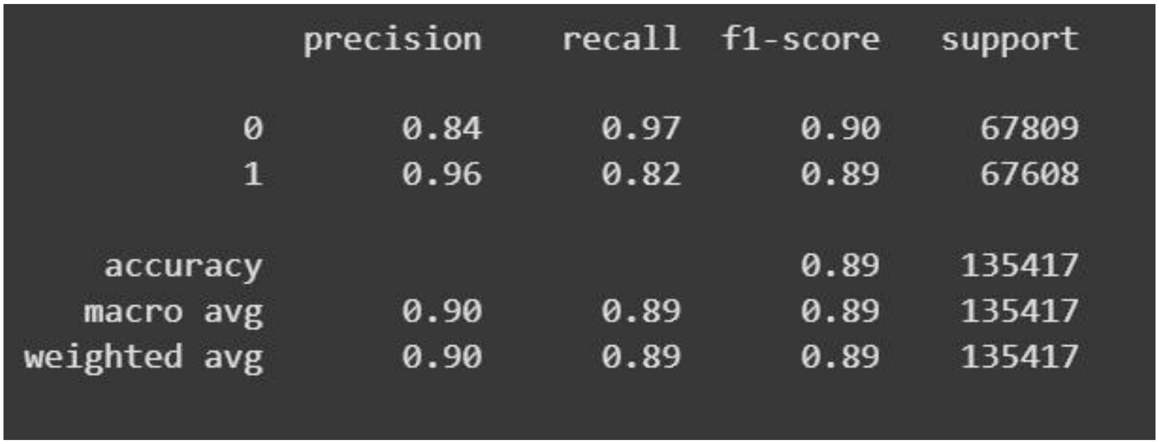
Model Validation

The selected model, XGBoost, was validated using multiple performance metrics to ensure reliability and accuracy.

Validation Techniques Used:

- Train-Test Split: The dataset was divided into training and testing sets to evaluate generalization.
- Classification Report: Precision, Recall, and F1-score were analysed to assess performance across different classes.
- Accuracy Score: The model's overall correctness in predictions was measured.
- Cross-Validation: Ensured stability and consistency of model performance across different data splits.

These validation steps confirmed that the Random Forest model effectively predicts loan defaults with high accuracy and robustness.



```
              precision    recall  f1-score   support

           0       0.84      0.97      0.90     67809
           1       0.96      0.82      0.89     67608

    accuracy                           0.89    135417
   macro avg       0.90      0.89      0.89    135417
weighted avg       0.90      0.89      0.89    135417
```

Fig 4 : Valid

CHAPTER 3
3. MODEL DESCRIPTION

3.1 ML

For this project, we implemented a **XGBoost** to predict loan defaulters based on various financial and demographic features. XGBoost was chosen due to its ability to handle both numerical and categorical data efficiently while reducing the risk of overfitting by aggregating multiple decision trees.

The model was trained and ensuring a balance between accuracy and generalization. It leverages multiple weak learners to create a robust predictive model that effectively captures complex relationships in the dataset.

3.2 WHY PySpark

Implementing the project in **PySpark with XGBoost** enables efficient processing of large-scale loan applicant data by leveraging Spark's distributed computing. PySpark ensures scalability, allowing the model to handle millions of records without performance bottlenecks.

XGBoost provides high accuracy and robustness through gradient boosting, making it ideal for imbalanced classification like loan defaults.

The combination reduces training time significantly compared to single-machine setups. It also supports easy integration with cloud platforms for deployment.

This approach ensures both **speed** and **predictive performance**, making it suitable for real-world financial risk analysis.

Key Reasons for Using PySpark:

1. **Scalability –** PySpark's distributed computing can process large datasets (millions of loan records) efficiently across multiple nodes.Automated Report Generation
2. **Speed -**

   Parallel data processing in PySpark combined with XGBoost's optimized gradient boosting drastically reduces training time.
3. **Accuracy -**

   XGBoost provides state-of-the-art performance for classification tasks, handling imbalanced datasets like loan defaults effectively.
4. **Fault Tolerance -**

   Spark automatically recovers from node failures, ensuring reliable training on big data.

CHAPTER 4
4. DATA FLOW

4.1 Data Flow in Loan Defaulter Prediction Project

The data flow in our project follows a structured pipeline, ensuring seamless processing from raw data collection to final decision-making. The key steps involved are:

1. **Data Collection & Preprocessing**
   - The dataset, sourced from **Kaggle**, contains **451,388 rows and 29 columns**.
   - Imbalanced target classes were handled using **synthetic data generation (duplicating rows) and downsampling**.
   - Features were categorized into **8 numerical** and **18 categorical columns**.
   - Missing values and outliers were handled to improve model performance.
2. **Feature Engineering & Selection**
   - Categorical variables were encoded using **one-hot encoding** and **label encoding**.
   - Numerical features were standardized to improve model training.
   - Feature selection was performed to retain only relevant attributes.
3. **Model Training & Evaluation**
   - Multiple models, including **Random Forest, XGBoost, Logistic Regression, SVM** were tested.
   - **Random Forest** was selected as the final model due to its high accuracy and robustness.
   - The model was trained using **70-30 train-test split** and evaluated

using **accuracy, precision, recall, and F1-score**.

4. **Prediction & Risk Assessment**
    - The trained model predicts whether an applicant will **default or not** based on financial and demographic factors.
    - Probability scores help assess the confidence of each prediction.

5. **Deployment & Decision Support**
    - The final model is integrated into a decision-support system for **banks, NBFCs, and financial institutions**.
    - Stakeholders can access **automated reports and risk analysis dashboards** to make informed lending decisions.

This structured data flow ensures efficient handling of loan default predictions while maintaining transparency and compliance.

CHAPTER 5

5. PROJECT REQUIREMENT

Project Requirements

The loan defaulter prediction project was developed using various tools and libraries that played a crucial role in building, training, and deploying the model. Below is a detailed description of the tools and technologies used:

Libraries Used:

1. **Python**: Python was the primary programming language used for developing the entire project. It is widely used in data science and machine learning due to its vast ecosystem of libraries and its simplicity. Python enabled us to implement various machine learning algorithms, data preprocessing, and model deployment effectively.
2. **scikit-learn**: This Python library was used for implementing machine learning algorithms. It provides a comprehensive collection of tools for data mining and data analysis, including various classification algorithms such as Random Forest, Logistic Regression. We also used it for data splitting, evaluation metrics, and feature engineering.
3. **SMOTE**: It is a method to handle class imbalance by creating synthetic examples of the minority class instead of simply duplicating existing ones. It generates new samples by interpolating between a minority class sample and its nearest neighbors in feature space. This helps improve model performance on imbalanced datasets by giving the minority class more representation during training.
4. **Flask**: Flask, a lightweight Python web framework, was used to deploy the machine learning model as a web application. Flask enabled us to create a user-friendly interface where users can input customer data and receive

predictions about loan default, along with the reasoning behind the prediction.

Deployment:

The project was deployed on an **AWS EC2 instance** running **Ubuntu** with **16 GB of storage** and **1 GB of RAM**. The EC2 instance provided the necessary compute power and storage capacity to handle the machine learning workload and serve the model to end-users.

AWS EC2 allowed for the creation of a scalable and reliable environment for hosting the Flask web application, ensuring high availability and seamless access to the loan default prediction system. The EC2 instance facilitated real-time interaction between the users and the deployed model, making it accessible from anywhere with internet access.

This combination of libraries and deployment tools ensures that the loan defaulter prediction system is robust, scalable, and efficient for practical use in a production environment.

CHAPTER 6
6. FUTURE SCOPE

1. **Incorporating More Data**: Adding additional data sources, such as transaction history and customer behavior, could enhance model accuracy.
2. **Real-Time Predictions**: Upgrading the system to provide real-time loan default predictions for instant decision-making.
3. **Model Improvement**: Exploring advanced algorithms like XGBoost and CatBoost to improve predictive performance.
4. **Integration with Credit Scoring**: Combining the model with existing credit scoring systems to offer a more comprehensive risk analysis.
5. **Explainable AI**: Implementing techniques like LIME to improve model transparency and offer clearer explanations for loan default predictions.

CHAPTER 7
7. CONCLUSION

This project successfully demonstrates how machine learning can be leveraged to predict loan defaults based on various customer attributes, such as demographic and financial factors. By analysing the dataset through correlation checks and visualizations, we gained valuable insights into the relationships between different features and loan default risk. Among the various machine learning models tested, XGBoost was chosen as the most effective algorithm, achieving an accuracy of 89%. This approach not only improves the decision-making process for lenders but also provides transparency, which is crucial in financial risk assessments. The project has shown promising results and offers a foundation for further advancements in loan prediction systems.

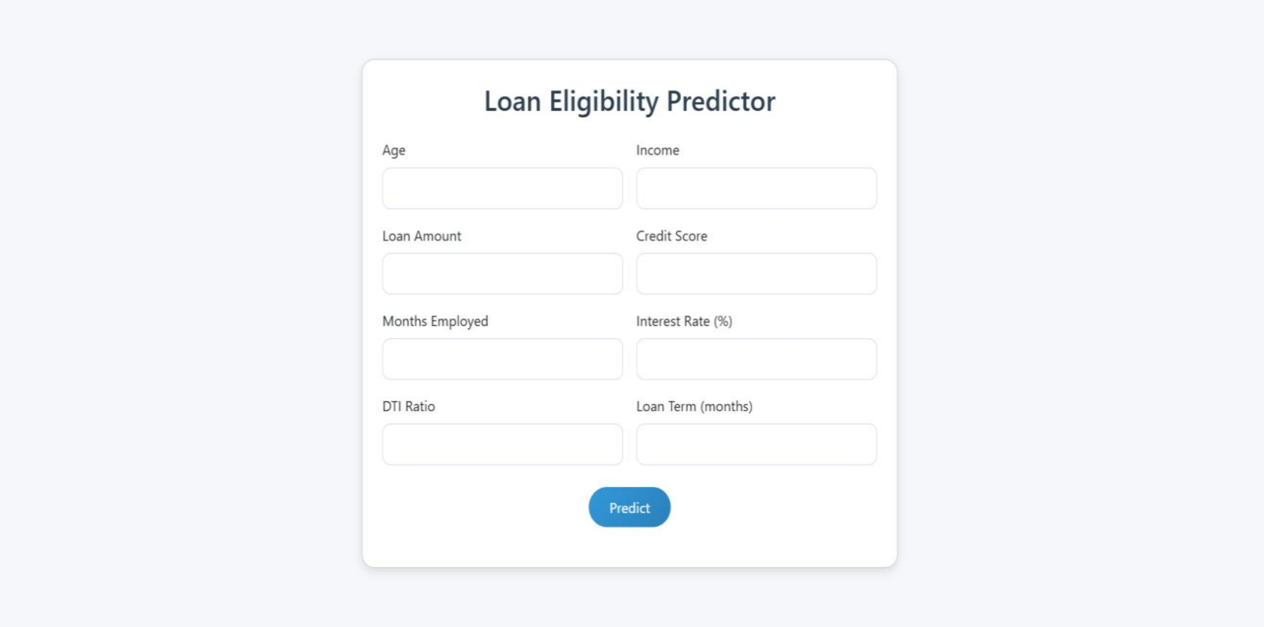
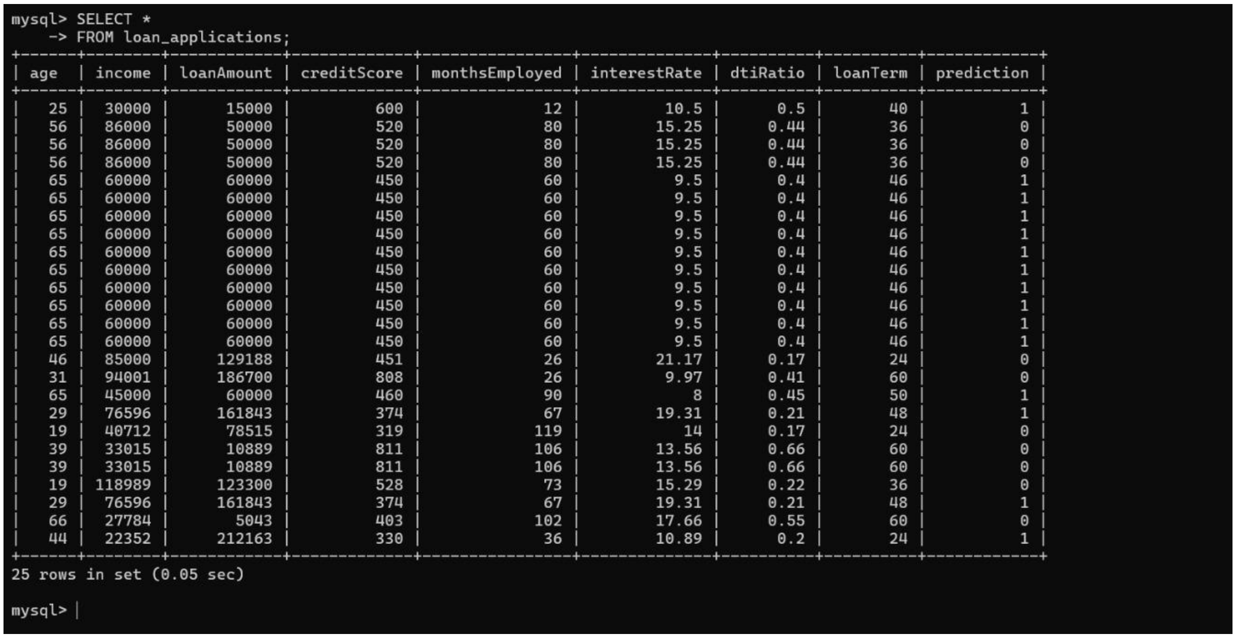


Fig 5 : Input Form



```
mysql> SELECT *
    -> FROM loan_applications;
+------+--------+------------+-------------+----------------+--------------+----------+----------+------------+
| age  | income | loanAmount | creditScore | monthsEmployed | interestRate | dtiRatio | loanTerm | prediction |
+------+--------+------------+-------------+----------------+--------------+----------+----------+------------+
|   25 |  30000 |      15000 |         600 |             12 |         10.5 |      0.5 |       40 |          1 |
|   56 |  86000 |      50000 |         520 |             80 |        15.25 |     0.44 |       36 |          0 |
|   56 |  86000 |      50000 |         520 |             80 |        15.25 |     0.44 |       36 |          0 |
|   56 |  86000 |      50000 |         520 |             80 |        15.25 |     0.44 |       36 |          0 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   65 |  60000 |      60000 |         450 |             60 |          9.5 |      0.4 |       46 |          1 |
|   46 |  85000 |     129188 |         451 |             26 |        21.17 |     0.17 |       24 |          0 |
|   31 |  94001 |     186700 |         808 |             26 |         9.97 |     0.41 |       60 |          0 |
|   65 |  45000 |      60000 |         460 |             90 |            8 |     0.45 |       50 |          1 |
|   29 |  76596 |     161843 |         374 |             67 |        19.31 |     0.21 |       48 |          1 |
|   19 |  40712 |      78515 |         319 |            119 |           14 |     0.17 |       24 |          0 |
|   39 |  33015 |      10889 |         811 |            106 |        13.56 |     0.66 |       60 |          0 |
|   39 |  33015 |      10889 |         811 |            106 |        13.56 |     0.66 |       60 |          0 |
|   19 | 118989 |     123300 |         528 |             73 |        15.29 |     0.22 |       36 |          0 |
|   29 |  76596 |     161843 |         374 |             67 |        19.31 |     0.21 |       48 |          1 |
|   66 |  27784 |       5043 |         403 |            102 |        17.66 |     0.55 |       60 |          0 |
|   44 |  22352 |     212163 |         330 |             36 |        10.89 |      0.2 |       24 |          1 |
+------+--------+------------+-------------+----------------+--------------+----------+----------+------------+
25 rows in set (0.05 sec)

mysql> |
```

Fig 6 : SQL Database

CHAPTER 8
8. REFERENCES

Here are some references you can add to the report:

1. **Online Documentation and Articles:**
   - "Random Forest Classifier - Scikit-learn Documentation." https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
   - "Flask Documentation." https://flask.palletsprojects.com/
   - SMOTE Documentation - https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html
   - Pandas Documentation - https://pandas.pydata.org/docs/
   - Numpy Documentation - https://numpy.org/doc/2.3/reference/index.html
   - Scikit-learn Documentation - https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
   - Onnx documentation - https://onnx.ai/onnx/
   - Distributed XGBoost with PySpark - https://xgboost.readthedocs.io/en/stable/tutorials/spark_estimator.html#sparkxgbclassifier