RESEARCH ARTICLE OPEN ACCESS

Credit Risk Prediction and Loan Suggestion System Using Financial Data

AJAY R, Dr. K. BANUROOPA MCA, M.Phil., Ph.D.
Associate Professor Department of Information Technology, Dr. N. G. P. Arts and Science College, Coimbatore - 48

ABSTRACT

Credit risk is one of the major challenges faced by banks and financial institutions when approving loans for customers. Financial organizations must carefully analyze whether a borrower has the ability to repay the loan without defaulting. Incorrect credit decisions may lead to financial losses and increase the risk for lending institutions. Therefore, an efficient and reliable credit risk evaluation system is necessary to support decision-making.

This project, **"Credit Risk Prediction System Using Machine Learning and Web-Based Interface"** aims to develop an automated system that predicts the credit risk level of loan applicants using financial and personal data. The system uses a **Random Forest Classifier** to analyze features such as age, income, employment status, loan amount, credit history, and debt ratio. These parameters help the model identify patterns and determine whether an applicant is likely to repay the loan or default.

A **Flask-based web application** is developed to allow administrators to upload datasets, train the machine learning model, and perform real-time predictions based on user inputs. The system processes the entered data and classifies applicants into categories such as **Low Risk, Medium Risk, and High Risk**. This helps financial institutions make faster and more accurate loan approval decisions.

The project demonstrates how **machine learning techniques can improve credit risk assessment by providing a data-driven and automated approach for financial decision-making.**

1.1 INTRODUCTION

In the modern financial industry, credit risk assessment plays a crucial role in the decision-making process of banks and financial institutions. Before approving a loan, organizations must evaluate whether the borrower has the ability to repay the loan within the specified period. Improper evaluation of credit risk can lead to loan defaults, financial losses, and instability in the banking system. Therefore, accurate prediction of credit risk is essential for maintaining financial stability and improving lending decisions.

Traditionally, credit risk assessment has been performed manually by financial experts who analyze various factors such as the applicant's income, employment status, credit history, loan amount, and repayment behavior. Although these traditional methods provide useful insights, they are often time-consuming and may involve human bias or errors. With the increasing volume of financial data, manual evaluation becomes inefficient and difficult to manage.

Advancements in **Machine Learning and Data Analytics** have provided new opportunities to improve the process of credit risk evaluation. Machine learning algorithms can analyze large datasets, identify hidden patterns, and make predictions based on historical data. By using these techniques, financial institutions can automate the credit evaluation process and improve the accuracy of their decisions.

The **Credit Risk Prediction System** is designed to predict the probability of loan default by analyzing different financial and personal attributes of loan applicants. The system uses machine learning algorithms such as **Random Forest Classifier** to build a predictive model using historical credit data. Based on the input provided by the user, the system predicts whether the applicant belongs to a **low-risk, medium-risk, or high-risk** category.To make the system accessible and user-friendly, a **web-based application is developed using Flask**. The application allows administrators to upload datasets, train the model, and perform real-time credit risk predictions. This system helps financial institutions make faster, more reliable, and data-driven decisions when approving loans.

1.2 OVERVIEW OF THE PROJECT

The main objective of the **Credit Risk Prediction System** is to develop an intelligent system that can predict the credit risk of loan applicants using machine learning techniques. The system aims to assist banks and financial institutions in making accurate and efficient decisions when approving loans.

The primary objectives of this project are:

• To develop a system that can analyze financial and personal information of loan applicants.

• To apply **machine learning algorithms** to predict whether a customer is likely to repay the loan or default.

• To build a predictive model using historical credit data for improving decision-making in financial institutions.

• To classify loan applicants into different risk categories such as **Low Risk, Medium Risk, and High Risk**.

• To design a **web-based application using Flask** that allows users to input applicant details and obtain credit risk predictions.

• To reduce manual effort and improve the efficiency of the credit evaluation process.

• To provide a reliable and automated system that supports financial institutions in minimizing credit risk.

By achieving these objectives, the proposed system helps organizations make faster and more accurate lending decisions, thereby reducing financial losses caused by loan defaults.

1.3 About the Organization

This project was carried out at **Nexus Global Solutions**, Coimbatore, as part of the internship and project training program for the final year B.Sc (Information Technology) students. Nexus Global Solutions is a technology-oriented organization that provides software development, IT solutions, and training services to students and professionals in various domains of modern computing.

The organization focuses on delivering innovative solutions using emerging technologies such as **Artificial Intelligence, Machine Learning, Data Science, Web Development, and Software Engineering**. It also supports students by offering practical training and project development opportunities that help them gain real-world technical experience.

During the project tenure from **December 2025 to February 2026**, the development of the project titled **“Credit Risk Prediction and Loan Suggestion System Using Financial Data”** was successfully completed under the guidance of professionals at Nexus Global Solutions. The organization provided the necessary support, technical guidance, and development environment required to design and implement the system.

Nexus Global Solutions encourages innovation, practical learning, and research-oriented development, helping students enhance their technical skills and gain industry exposure. The organization plays an important role in bridging the gap between academic knowledge and real-world software development practices.

SYSTEM ANALYSIS

System Analysis is an important phase in the development of any software system. It involves studying the current system, identifying its problems, and defining the requirements for the new system. The main purpose of system analysis is to understand how the system should function and what features are required to achieve the project objectives.

In the **Credit Risk Prediction System**, system analysis focuses on understanding the process of evaluating loan applicants and identifying factors that influence credit risk. Financial institutions must analyze several parameters such as income level, employment status, credit history, loan amount, and repayment capacity before approving loans. Manual analysis of these factors can be time-consuming and may not always provide accurate results.

The system analysis phase helps in identifying the limitations of the existing credit evaluation process and determining how machine learning techniques can improve the prediction accuracy. By analyzing historical credit data, machine learning models can identify patterns and relationships between borrower characteristics and loan repayment behavior.

The proposed system is designed to automate the credit risk prediction process by using machine learning algorithms and a web-based interface. The system allows administrators to upload datasets, train predictive models, and perform real-time predictions based on user inputs. This approach helps reduce human errors, improves efficiency, and supports financial institutions in making better lending decisions.

Through proper system analysis, the requirements of the **Credit Risk Prediction System** are clearly defined, ensuring that the system provides accurate predictions, user-friendly interaction, and efficient processing of loan applicant data.

2.1 EXISTING SYSTEM

In the existing system, credit risk assessment is mostly performed using traditional methods where financial experts manually evaluate loan applications. Banks and financial institutions analyze the financial records, credit history, income level, employment status, and other personal details of the applicants before making a decision. This process relies heavily on human judgment and experience.

Although the traditional credit evaluation process helps in identifying risky borrowers, it has several limitations. The process is time-consuming and requires a large amount of manual effort to verify and analyze applicant data. With the increasing number of loan applications, it becomes difficult for financial institutions to process and evaluate each application efficiently.

Another major drawback of the existing system is the possibility of human errors and bias during the decision-making process. Manual analysis may overlook certain patterns in the data that could indicate potential risk. In addition, traditional methods do not effectively utilize the large amount of historical financial data available to organizations.

Because of these limitations, the existing credit risk assessment methods may lead to inaccurate predictions and delayed decision-making. This highlights the need for an automated system that can analyze financial data efficiently and provide accurate credit risk predictions using advanced technologies such as machine learning.

2.2 PROPOSED SYSTEM

The proposed system aims to develop an automated **Credit Risk Prediction System** using machine learning techniques to improve the efficiency and accuracy of credit risk evaluation. The system analyzes various financial and personal attributes of loan applicants to predict their ability to repay the loan.

In this system, a **machine learning model** is trained using historical credit data that contains information about previous loan applicants and their repayment behavior. The model learns patterns and relationships between different attributes such as age, income, employment status, loan amount, credit history, and debt ratio. Based on this training, the system can predict whether a new applicant is likely to repay the loan or default.

The proposed system uses the **Random Forest Classifier**, which is a powerful machine learning algorithm known for its accuracy and ability to handle complex datasets. This algorithm analyzes multiple decision trees and combines their outputs to generate reliable predictions.

A **web-based application developed using Flask** is used as the interface for interacting with the system. The application allows administrators to upload datasets, train the machine learning model, and perform credit risk predictions by entering applicant details. Once the user provides the necessary information, the system processes the data and classifies the applicant into categories such as **Low Risk, Medium Risk, or High Risk**.

The proposed system helps financial institutions make faster and more reliable loan approval decisions. It reduces manual effort, minimizes human errors, and provides a data-driven approach to credit risk evaluation.

REQUIREMENTS

To ensure high performance, scalability, and maintainability, the project adopts a modern and efficient technology stack.

3.1 SOFTWARE REQUIREMENTS:

The software requirements refer to the programs, tools, and frameworks used to develop and run the **Credit Risk Prediction System**. These software components provide the necessary environment for implementing machine learning algorithms, processing data, and building the web-based interface of the application.

The following software tools are used in the development of the system:

- **Operating System** : Windows 10 or later
- **Programming Language** : Python
- **Framework** : Flask
- **Libraries** : NumPy, Pandas, Scikit-learn, Matplotlib
- **Development Environment** : Visual Studio Code / PyCharm
- **Database** : CSV Dataset

Python is used as the primary programming language because it provides powerful libraries for data analysis and machine learning. Libraries such as **NumPy** and **Pandas** are used for data processing and analysis, while **Scikit-learn** is used for implementing the machine learning algorithm used in the credit risk prediction model.

The **Flask framework** is used to develop the web-based interface that allows users to input applicant details and receive predictions. The system uses a dataset stored in CSV format to train the machine learning model and perform credit risk classification.

These software tools together provide a suitable platform for developing and deploying the Credit Risk Prediction System efficiently.

3.2 HARDWARE REQUIREMENTS:

The hardware requirements refer to the physical components needed to develop and run the **Credit Risk Prediction System**. Since the system is implemented as a machine learning–based web application, it does not require high-end hardware and can operate efficiently on a standard computer system.

The following are the minimum hardware requirements for the system:

Processor: Intel Core i3 or higher
RAM: 4 GB or higher
Hard Disk: 500 GB or higher
Monitor: Standard display monitor
Input Devices: Keyboard and Mouse

4.1 ARCHITECTURE OVERVIEW

The architecture of the **Credit Risk Prediction System** describes the overall workflow of the system and how different components interact to produce the credit risk prediction result. The system is designed to process loan applicant data, apply machine learning techniques, and generate predictions that help financial institutions evaluate credit risk.The process begins with the **user uploading the loan dataset** into the system. This dataset contains various attributes related to loan applicants such as income, age, employment status, loan amount, and credit history. The uploaded dataset serves as the primary input for building and training the prediction model.After the dataset is uploaded, the **data preprocessing stage** is performed. In this stage, the raw data is cleaned and prepared for analysis. Data preprocessing includes tasks such as handling missing values, removing unnecessary information, and converting categorical values into a suitable format that can be processed by the machine learning algorithm.

The next step is **feature extraction**, where the most relevant attributes from the dataset are selected for model training. Feature extraction helps improve the performance of the machine learning model by focusing on the important variables that influence credit risk prediction.

Once the important features are extracted, the processed data is passed to the **Support Vector Machine (SVM) algorithm**. The SVM algorithm is used to train the model and identify patterns in the historical loan data. The trained model is then capable of analyzing new loan applications and determining the level of credit risk associated with them.A **test dataset** is also provided to evaluate the performance of the trained model. The test data is used to validate the accuracy of the prediction model and ensure that it performs well when predicting unseen data.

Finally, the system generates the **credit risk prediction result**, which is displayed to the user. The result indicates the risk level associated with the loan applicant, helping financial institutions make better decisions regarding loan approvals.This architecture ensures a systematic flow of data from dataset upload to prediction generation, enabling efficient and accurate credit risk assessment using machine learning techniques.

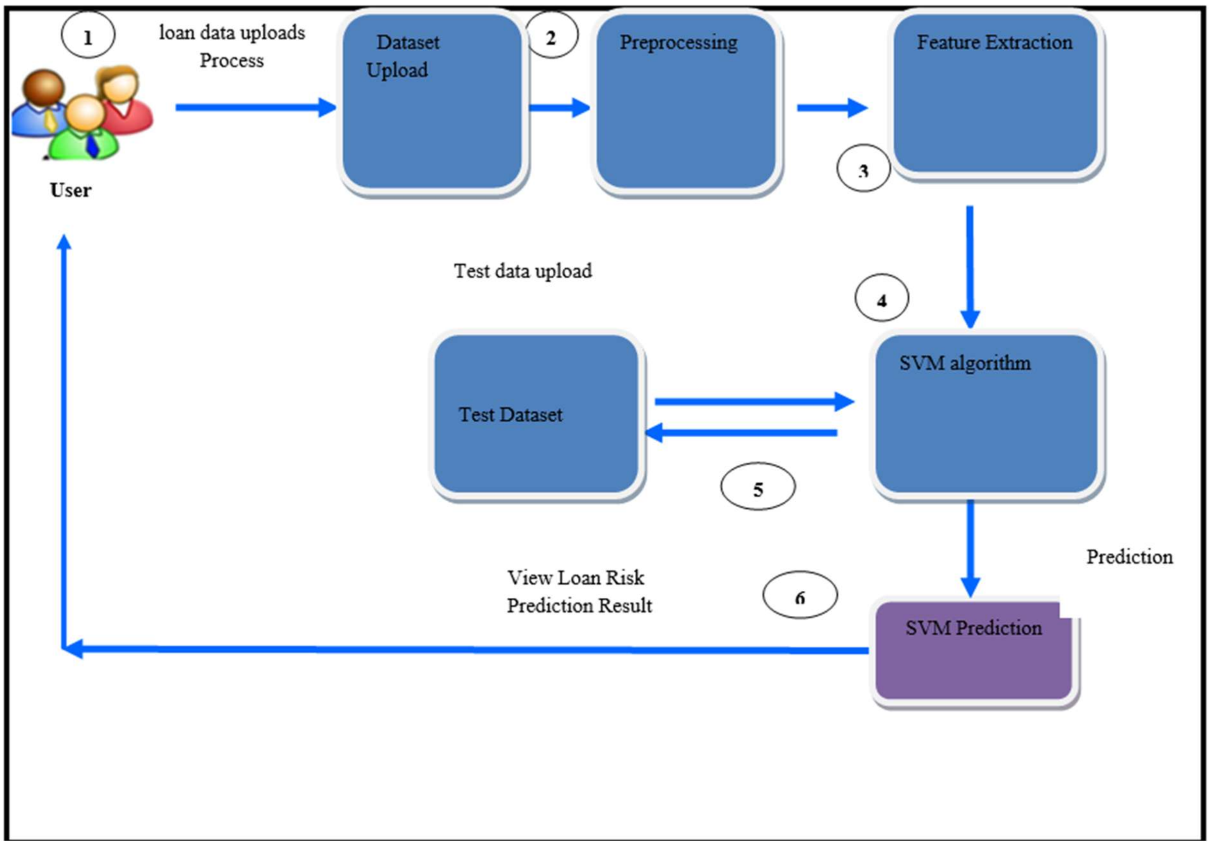


Fig: system architecture

4.2 DATA FLOW

The **Data Flow** of the Credit Risk Prediction System describes how data moves through different stages of the system to generate the final prediction. It shows the sequence of operations performed on the dataset from the moment it is uploaded until the credit risk prediction result is displayed to the user.

Initially, the **user uploads the loan dataset** into the system. This dataset contains important information related to loan applicants such as income, employment status, credit history, loan amount, and other financial attributes. This data serves as the input for the system.

After the dataset is uploaded, the system performs **data preprocessing**. In this stage, the raw dataset is cleaned and prepared for analysis. This includes handling missing values, removing duplicate records, and converting categorical data into a suitable format that can be processed by the machine learning model.

Once preprocessing is completed, the system performs **feature extraction**. Feature extraction involves selecting the most relevant attributes from the dataset that influence credit risk prediction. By selecting important features, the system improves the accuracy and efficiency of the prediction model.

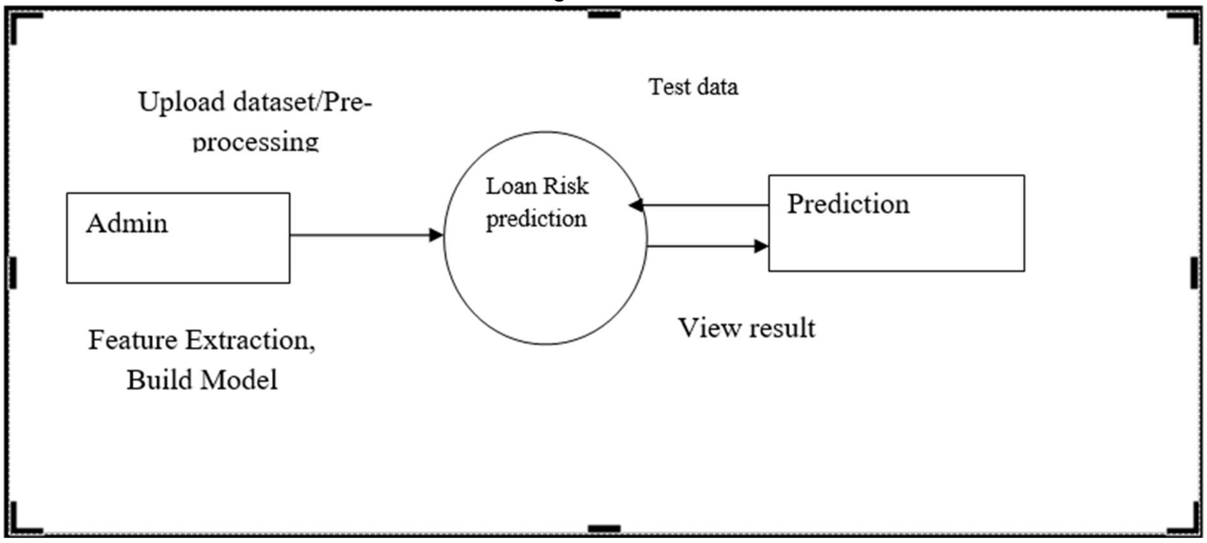
The processed data is then passed to the **Support Vector Machine (SVM) algorithm**, which is used to train the prediction model. The algorithm analyzes the relationships between different features and learns patterns from the historical loan data.

Next, the **test dataset** is used to evaluate the performance of the trained model. The test data helps verify whether the model can accurately predict credit risk for new loan applicants.

Finally, the trained model produces the **credit risk prediction**, which is displayed to the user. The prediction result indicates whether the loan applicant falls into a particular risk category, helping financial institutions make informed lending decisions.

Thus, the data flows through several stages including dataset upload, preprocessing, feature extraction, model training, testing, and prediction to generate accurate credit risk evaluation results.

Fig 1: level 0


Upload dataset/Pre-processing
Test data
Admin
Loan Risk prediction
Prediction
Feature Extraction, Build Model
View result

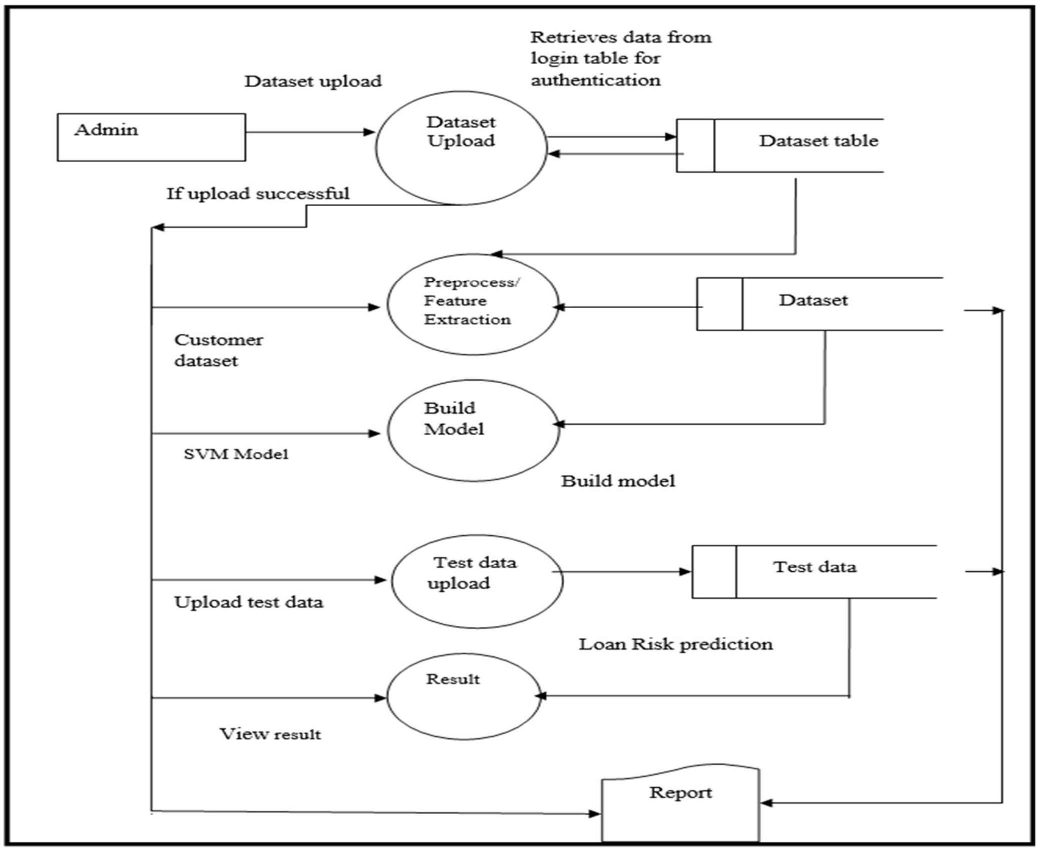


Fig 2: level 1

4.3 MODULE DESCRIPTION

The **Credit Risk Prediction System** is divided into several modules to ensure smooth functioning and efficient processing of data. Each module performs a specific task in the system, starting from data input to generating the final credit risk prediction.

1. Dataset Upload Module

This module allows the user or administrator to upload the loan dataset into the system. The dataset contains information about loan applicants such as income, age, employment status, loan amount, and credit history. This dataset is used to train and evaluate the machine learning model.

2. Data Preprocessing Module

The preprocessing module prepares the raw dataset for analysis. In this stage, unnecessary or inconsistent data is removed and missing values are handled. The module also converts categorical data into numerical format so that it can be processed by the machine learning algorithm.

3. Feature Extraction Module

The feature extraction module selects the most relevant attributes from the dataset that influence credit risk prediction. By focusing on important features, the system improves prediction accuracy and reduces unnecessary data processing.

4. Model Training Module

In this module, the processed dataset is used to train the **Support Vector Machine (SVM) algorithm**. The algorithm analyzes the historical loan data and learns patterns that help identify whether a loan applicant is likely to repay the loan or default.

5. Testing Module

The testing module uses a test dataset to evaluate the performance of the trained machine learning model. This step ensures that the model produces reliable and accurate predictions when applied to new data.

6. Prediction Module

This module generates the final **credit risk prediction** based on the input data. The trained model analyzes the provided details of a loan applicant and predicts the level of credit risk. The result is then displayed to the user through the system interface.

By dividing the system into these modules, the Credit Risk Prediction System becomes easier to manage, maintain, and improve in the future.

4.4 SOFTWARE DESCRIPTION

The **Credit Risk Prediction System** is developed using various software tools and technologies that support data processing, machine learning model development, and web application deployment. These software components provide an efficient environment for building and running the system.

The primary programming language used in the development of the system is **Python**. Python is widely used in machine learning and data science because of its simplicity, flexibility, and availability of powerful libraries for data analysis and model development.

The system uses the **Flask framework** to develop the web-based interface. Flask is a lightweight web framework that allows easy integration between the user interface and the backend machine learning model. It enables users to input loan applicant details and obtain credit risk predictions through a web application.

Several Python libraries are used to support data processing and model development. **Pandas** is used for handling and manipulating datasets, while **NumPy** helps in performing numerical computations. The **Scikit-learn** library is used to implement the machine learning algorithm for credit risk prediction.

For development and coding, an integrated development environment such as **Visual Studio Code** or **PyCharm** can be used. These tools provide features such as code editing, debugging, and project management, which make the development process easier and more efficient.

The dataset used in the system is stored in **CSV (Comma Separated Values) format**, which allows easy loading and processing of data during model training and prediction.

By using these software tools and technologies, the Credit Risk Prediction System can efficiently process loan data, train machine learning models, and provide accurate credit risk predictions through a user-friendly web interface.

5.1 SYSTEM DESIGN

System design is an important phase in the development of the **Credit Risk Prediction System**. It defines the structure of the system and explains how different components interact with each other to perform the required tasks. The main objective of system design is to create an organized framework that ensures the efficient functioning of the application.

The system is designed as a **web-based application** that allows users to interact with the credit risk prediction model through a simple interface. The design includes both the frontend and backend components. The frontend provides a user-friendly interface where users can upload datasets and enter loan applicant details. The backend processes the input data, performs machine learning operations, and generates prediction results.

The system design includes several stages such as dataset upload, data preprocessing, feature extraction, model training, and prediction generation. When the user uploads the dataset, the system first performs preprocessing to clean and prepare the data. After preprocessing, important features are extracted to improve the efficiency and accuracy of the prediction model.

The processed data is then used to train the machine learning model. The trained model analyzes the relationships between different attributes of loan applicants and predicts the credit risk level. Finally, the system displays the prediction result to the user through the web interface.

This structured system design ensures that the **Credit Risk Prediction System** operates efficiently and provides accurate results for evaluating the creditworthiness of loan applicants.

5.2 System Flow Diagram

A system flow diagram (SFD) is a visual representation of the flow of information, processes, and data within a system. It helps clarify how different components interact, making it easier to understand complex systems.

- Workflow Visualization: Mapping user interactions, system processes, and data flow to ensure clarity during development.
- Requirement Analysis: Helping stakeholders understand functional requirements and how different components interact.

By using SFDs in these contexts, organizations can improve clarity, efficiency, and communication, leading to better outcomes in system design and implementation.

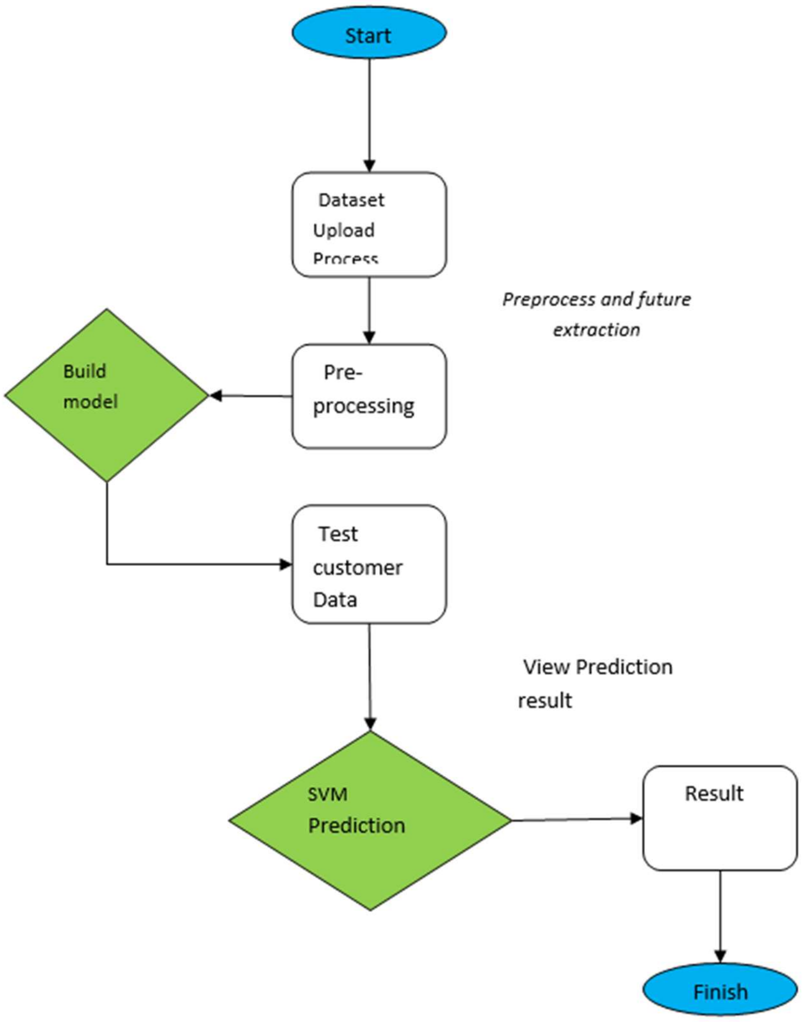


Figure 3: System flow design

5.3 Dataset Design

Dataset design plays an important role in the development of the **Credit Risk Prediction System**. The dataset contains information related to loan applicants, which is used to train and test the machine learning model. The quality and structure of the dataset directly influence the accuracy of the credit risk prediction.

The dataset used in this project consists of several attributes that describe the financial and personal details of loan applicants. These attributes help the machine learning model analyze patterns and determine whether a borrower is likely to repay the loan or default.

The dataset is stored in **CSV (Comma Separated Values) format**, which allows easy processing using Python libraries such as Pandas and NumPy.

Dataset Attributes

The dataset contains the following fields:

• **Applicant_ID** – Unique identification number of the loan applicant.
• **Age** – Age of the applicant applying for the loan.
• **Gender** – Gender of the applicant.
• **Marital_Status** – Indicates whether the applicant is married or single.
• **Education** – Educational qualification of the applicant.
• **Employment_Status** – Employment status of the applicant (employed / self-employed).

• **Annual_Income** – Total annual income of the applicant.
• **Loan_Amount** – Amount of loan requested by the applicant.
• **Loan_Term** – Duration of the loan repayment period.
• **Credit_History** – Previous credit repayment history of the applicant.
• **Existing_Loans** – Number of loans already taken by the applicant.
• **Debt_to_Income_Ratio** – Ratio between total debt and income of the applicant.
• **Property_Area** – Area where the applicant owns property (urban / semi-urban / rural).
• **Risk_Level (Target Variable)** – Final output indicating the credit risk level such as **Low Risk, Medium Risk, or High Risk**.

The dataset is divided into **training data and testing data**. The training data is used to build the machine learning model, while the testing data is used to evaluate the performance and accuracy of the prediction model. By designing the dataset with relevant attributes, the system can effectively analyze financial patterns and provide accurate credit risk predictions.

5.4 ER Diagram

The **Entity Relationship (ER) Diagram** represents the logical structure of the database used in the **Credit Risk Prediction System**. It illustrates how different entities in the system are related to each other and how data is organized and stored.

In this system, the main entity is the **Loan Applicant**, which contains important information about individuals applying for loans. The applicant details include personal and financial attributes such as age, income, employment status, loan amount, and credit history.

Another important entity is the **Loan Details**, which stores information related to the loan requested by the applicant. This includes the loan amount, loan term, and repayment details. These attributes help the system analyze the financial condition of the applicant.

The **Credit History** entity stores previous credit-related information of the applicant, such as past loan records and repayment behavior. This information plays a key role in determining the creditworthiness of the applicant.

The **Prediction Result** entity stores the final output generated by the machine learning model. It records whether the applicant is categorized as **Low Risk, Medium Risk, or High Risk** based on the analysis performed by the system.

The ER diagram shows the relationships between these entities, demonstrating how applicant information, loan details, and credit history are connected to generate the final credit risk prediction.

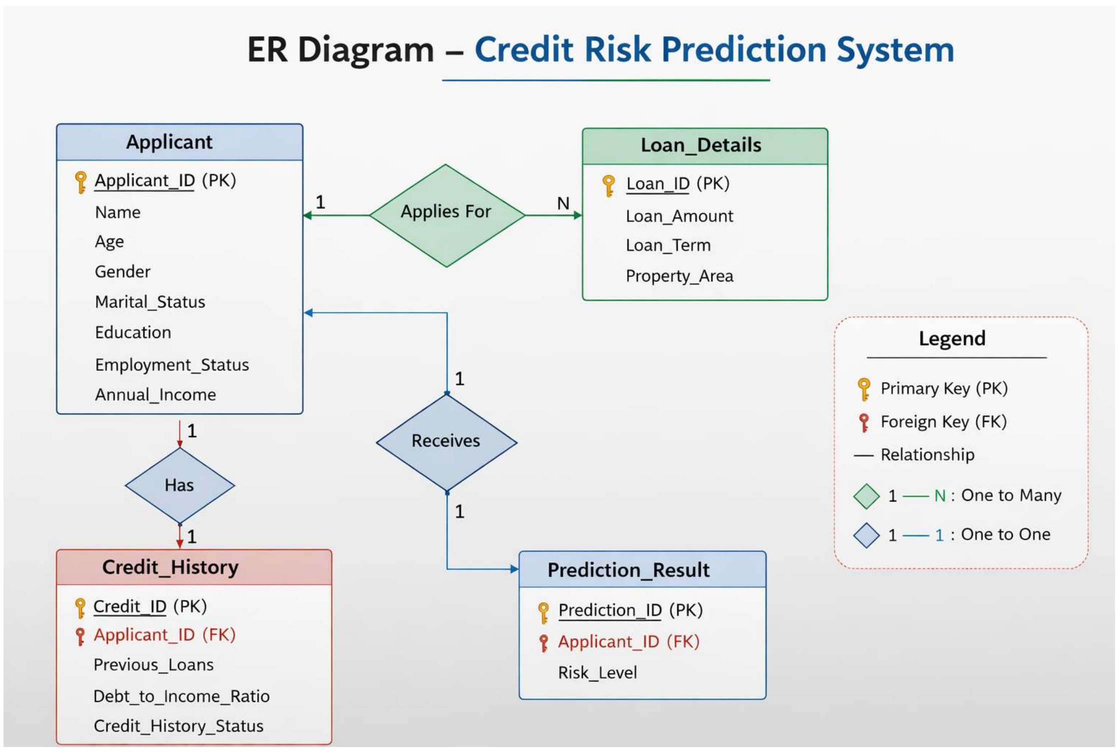


Figure 5: ER Diagram

5.5 FORM DESIGN

Form design plays an important role in the Credit Risk Prediction System as it provides the interface through which users can interact with the application. Forms are used to collect necessary information from the user and pass it to the system for processing and prediction.

The system includes a loan applicant input form where users enter the required details related to the loan applicant. These details include personal and financial information such as age, gender, marital status, education level, employment status, annual income, loan amount, loan term, credit history, and property area. The form is designed in a simple and user-friendly manner so that users can easily enter the required data without confusion.

Once the user fills in the required fields and submits the form, the data is sent to the backend system for processing. The backend performs data preprocessing and passes the information to the trained machine learning model. The model analyzes the provided data and predicts the credit risk level of the loan applicant.

The form also displays the prediction result after processing the input data. The result indicates whether the applicant belongs to a Low Risk, Medium Risk, or High Risk category. This information helps financial institutions make informed decisions about loan approvals.

The form design ensures that all necessary input fields are clearly labeled and validated before submission. Proper form validation helps prevent incorrect or incomplete data entry, thereby improving the accuracy of the prediction results.

By providing a structured and user-friendly form interface, the system enables efficient interaction between the user and the Credit Risk Prediction System.

6.1 SYSTEM TESTING

Software testing is a critical element of software quality assurance that represents the ultimate review of specifications, design and coding. The user tests the developed system and changes are made according to their needs. The testing phase involves the testing of developed system using various kinds of data. It involves user training, system testing and successful running of the developed system.

The changes are made according to their needs. The testing phase involves the testing of the developed system using various kinds of data. While testing, errors are noted and corrections are made system testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. The candidate system is subject to a variety of tests: stress recovery, security and usability tests.

Test Plan

Testing is the process of executing a program with the intent of finding any errors. A good test of course has the high probability of finding an undiscovered error. A successful testing is the one that uncovers a yet undiscovered error.

A test is vital to the success of the system; system test makes a logical assumption that if all parts of the system are correct, then goal will be successfully achieved. The candidate system is subjected to a variety of tests online like responsiveness, its value, stress and security. A series of tests are performed before the system is ready for user acceptance testing.

Testing Methods

The different types of testing are: -

- **Unit Testing**
- **Integration Testing**
- **Validation Testing**

UNIT TESTING

Unit testing focuses verification efforts on the smallest unit of software design, the module. This is also known as "Module Testing" The modules are tested separately this testing is carried out during programming stage itself. In this step each module is found to be working satisfaction as regard to the expected output from the module.

INTEGRATION TESTING

Integration testing focuses on the design and construction of software architecture. Data can be lost across an interface, one module can have adverse effect on another sub functions and show on. Thus integration testing is a systematic technique for constructing test to uncover errors associated with in the interface. In this project, all the modules are companied and then the entire program is tested.

VALIDATION TESTING

Validation testing is the requirement established as a part of software requirement analysis is validated against the software that has been constructed. This test provides the final assurance whether the software needs all functional, behavioral and performance requirements

Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactory.

6.2 SYSTEM IMPLEMENTATION

System implementation is the stage where the **Credit Risk Prediction System** is developed and executed based on the designed architecture and system requirements. In this phase, the theoretical design of the system is converted into a working application using appropriate programming tools and technologies.

The system is implemented using the **Python programming language**, which provides strong support for data analysis and machine learning. The development process includes loading the dataset, performing data preprocessing, extracting important features, and training the machine learning model for credit risk prediction.

In this project, the **Support Vector Machine (SVM) algorithm** is used to train the model. The algorithm analyzes historical loan data and learns patterns related to borrower behavior and repayment capability. Once the model is trained, it can be used to predict the credit risk level of new loan applicants.

The implementation also includes the development of a **web-based interface using the Flask framework**. This interface allows users to interact with the system by entering loan applicant details through a form. The input data is sent to the backend where it is processed and passed to the trained machine learning model for prediction.

The system uses Python libraries such as **Pandas** for dataset handling, **NumPy** for numerical operations, and **Scikit-learn** for implementing the machine learning algorithm. These libraries help in efficiently processing the data and generating accurate prediction results.

After processing the input data, the system generates the **credit risk prediction**, which is displayed to the user through the web interface. The result indicates whether the applicant belongs to a low-risk, medium-risk, or high-risk category.

Thus, the implementation phase ensures that the designed system functions properly and provides reliable credit risk predictions for loan applicants.

7.1 CONCLUSION

The **Credit Risk Prediction System** was developed to assist financial institutions in evaluating the creditworthiness of loan applicants using machine learning techniques. The system analyzes various financial and personal attributes of applicants, such as income, employment status, credit history, and loan amount, to predict the level of credit risk associated with each loan application.

By implementing machine learning algorithms, the system can identify patterns in historical loan data and provide accurate predictions regarding whether an applicant is likely to repay the loan or default. This helps financial institutions make informed and faster decisions during the loan approval process.

The system also provides a user-friendly web interface developed using Flask, which allows users to upload datasets, input applicant details, and obtain prediction results efficiently. The integration of data preprocessing, feature extraction, model training, and prediction ensures that the system performs credit risk evaluation effectively.

Overall, the Credit Risk Prediction System helps reduce manual effort, minimize human errors, and improve the efficiency of credit assessment. It provides a reliable and automated approach for predicting loan risk, which can support banks and financial organizations in reducing financial losses and improving their decision-making process.

7.2 FUTURE ENHANCEMENT

Although the **Mental Health State Prediction System** performs effectively, there are several improvements that can be implemented in the future to enhance the system's performance and usability. One of the major enhancements would be the use of a **larger and more diverse dataset**, which can help the machine learning model learn more complex behavioral patterns and improve prediction accuracy.

Another possible improvement is the implementation of **additional machine learning algorithms** such as Support Vector Machines, Decision Trees, or Neural Networks. Comparing different algorithms can help identify the most suitable model for predicting mental health conditions more accurately.

The system can also be improved by developing a **mobile application** so that users can easily access the prediction system through smartphones. Features such as **real-time monitoring, graphical reports, and personalized mental health suggestions** can also be added to provide more useful insights to users.

In the future, the system could also be integrated with **wearable devices or health tracking applications** to automatically collect data such as sleep patterns and physical activity. This would make the prediction process more automated and provide more accurate mental health assessments.

8. BIBLIOGRAPHY

[1] Tom M. Mitchell, *Machine Learning*, McGraw-Hill Education, 1997.
[2] Ian H. Witten, Eibe Frank, Mark A. Hall, and Christopher J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Publishers, 2016.
[3] Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, 2019.
[4] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *The Elements of Statistical Learning*, Springer, 2009.
[5] Wes McKinney, *Python for Data Analysis*, O'Reilly Media, 2017.
[6] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, *An Introduction to Statistical Learning*, Springer, 2013.
[7] Python Software Foundation, *Python Programming Language Documentation*, Available at: https://www.python.org
[8] Scikit-learn Documentation, Available at: https://scikit-learn.org
[9] NumPy Documentation, Available at: https://numpy.org
[10] Pandas Documentation, Available at: https://pandas.pydata.org
[11] Flask Documentation, Available at: https://flask.palletsprojects.com
[12] Research papers and articles related to **Credit Risk Prediction using Machine Learning Techniques** from online academic journals.
[13] Research articles on **loan default prediction and financial risk analysis** from IEEE and other academic databases.
[14] Online tutorials and documentation related to **Machine Learning Algorithms and Credit Risk Analysis**.
[15] Various online resources and technical blogs related to **Python-based machine learning and financial data analysis**.

9.1 APPENDIX
SCREENSHOTS

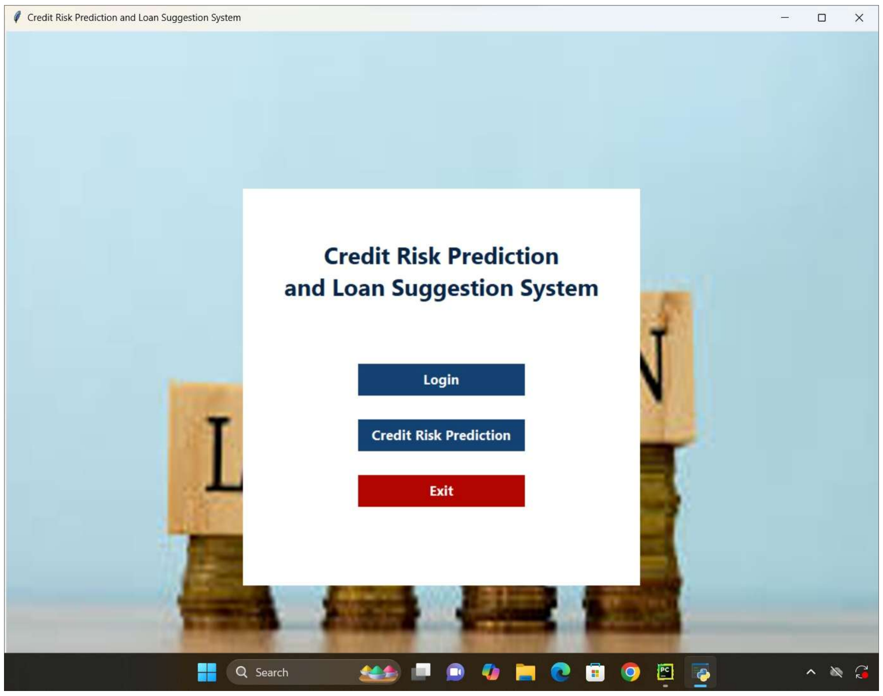

Credit Risk Prediction and Loan Suggestion System
Credit Risk Prediction
and Loan Suggestion System
Login
Credit Risk Prediction
Exit
Search

Figure 5: home page

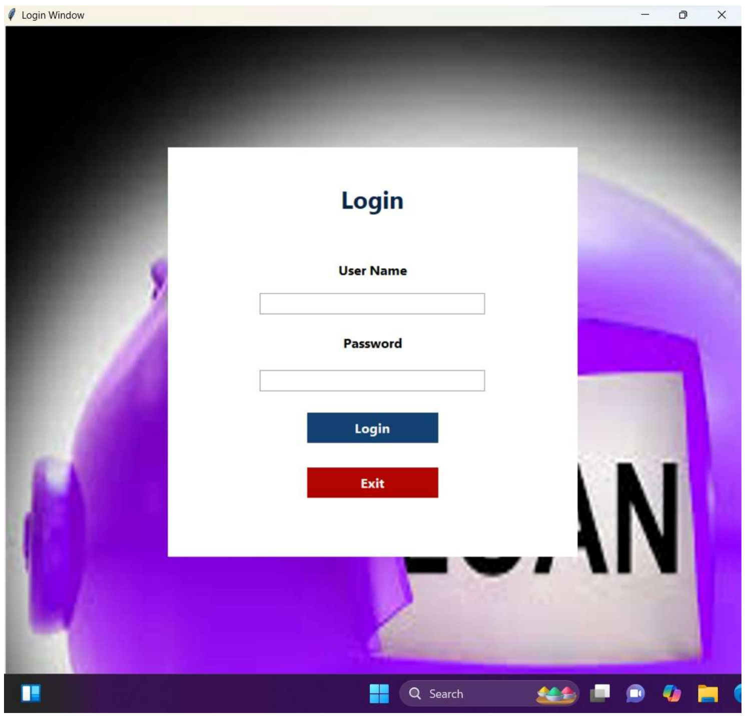


Figure 6: login Page

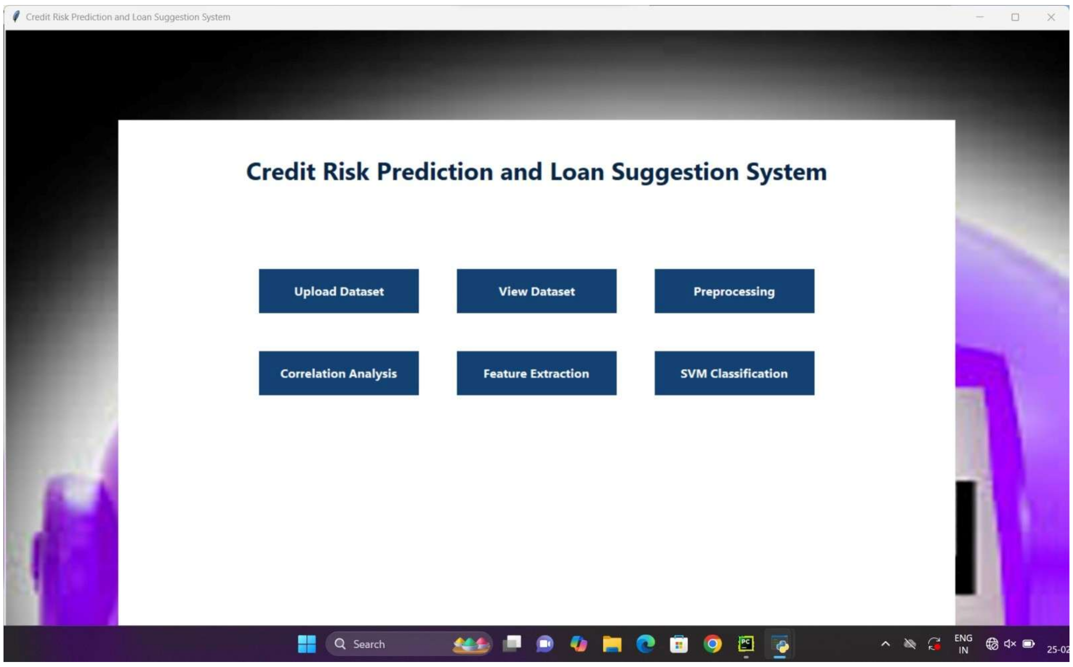
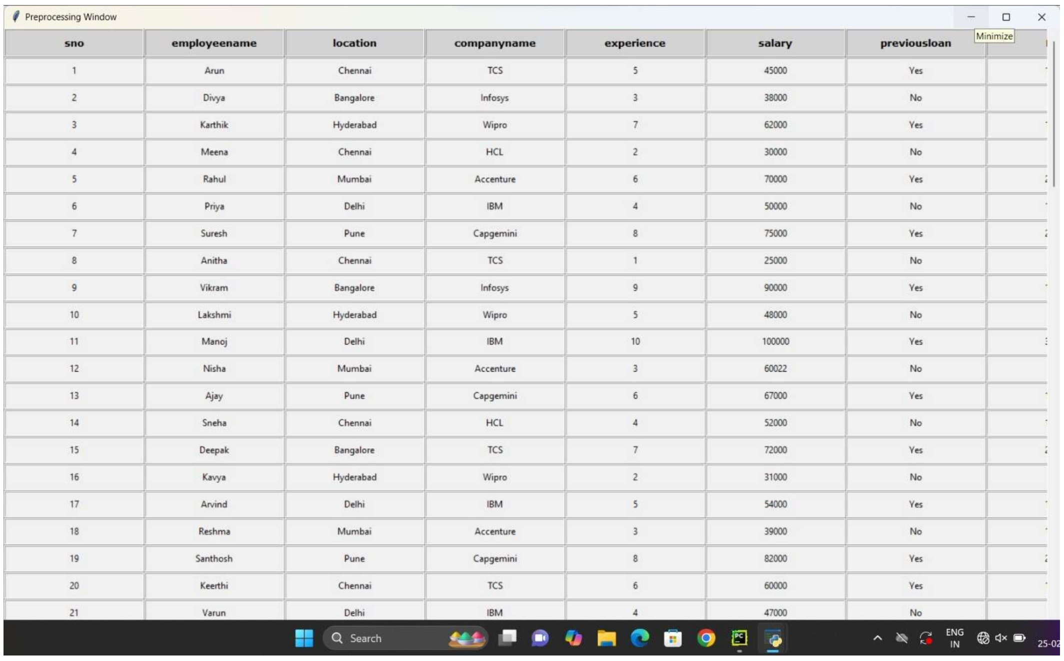

Credit Risk Prediction and Loan Suggestion System
Credit Risk Prediction and Loan Suggestion System
Upload Dataset
View Dataset
Preprocessing
Correlation Analysis
Feature Extraction
SVM Classification

Figure 7: upload page



sno	employeename	location	companyname	experience	salary	previousloan
1	Arun	Chennai	TCS	5	45000	Yes
2	Divya	Bangalore	Infosys	3	38000	No
3	Karthik	Hyderabad	Wipro	7	62000	Yes
4	Meena	Chennai	HCL	2	30000	No
5	Rahul	Mumbai	Accenture	6	70000	Yes
6	Priya	Delhi	IBM	4	50000	No
7	Suresh	Pune	Capgemini	8	75000	Yes
8	Anitha	Chennai	TCS	1	25000	No
9	Vikram	Bangalore	Infosys	9	90000	Yes
10	Lakshmi	Hyderabad	Wipro	5	48000	No
11	Manoj	Delhi	IBM	10	100000	Yes
12	Nisha	Mumbai	Accenture	3	60022	No
13	Ajay	Pune	Capgemini	6	67000	Yes
14	Sneha	Chennai	HCL	4	52000	No
15	Deepak	Bangalore	TCS	7	72000	Yes
16	Kavya	Hyderabad	Wipro	2	31000	No
17	Arvind	Delhi	IBM	5	54000	Yes
18	Reshma	Mumbai	Accenture	3	39000	No
19	Santhosh	Pune	Capgemini	8	82000	Yes
20	Keerthi	Chennai	TCS	6	60000	Yes
21	Varun	Delhi	IBM	4	47000	No

Figure 8: dataset Page

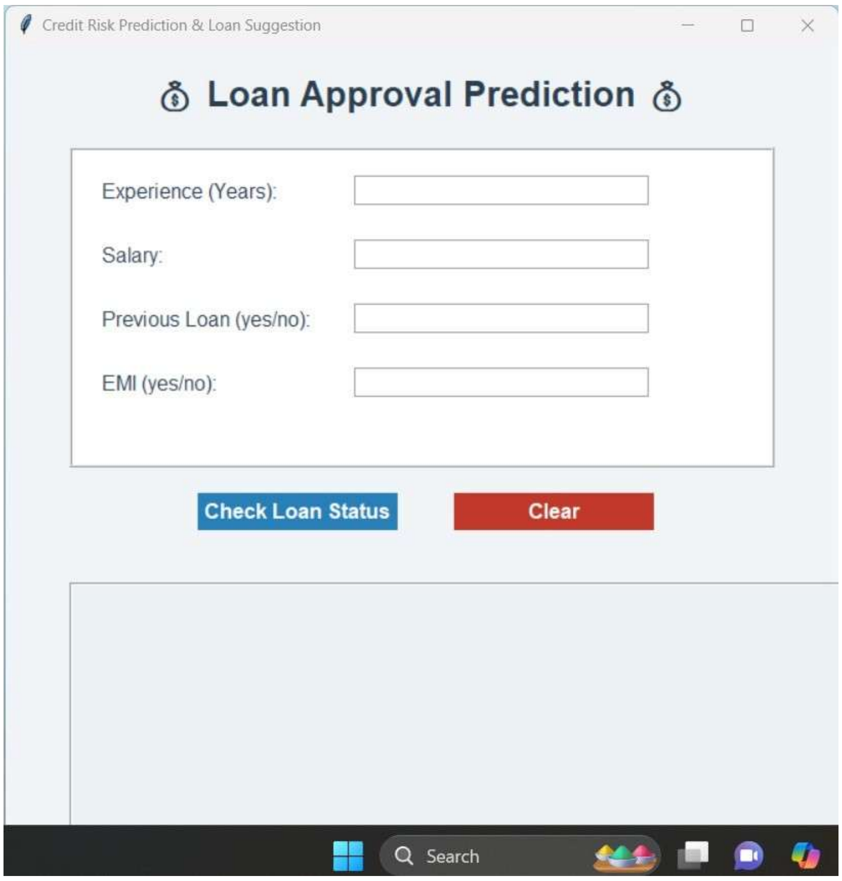


Figure 9: result Page

SAMPLE CODING

```
from tkinter import *
import random
import mysql.connector as mysql
win = Tk()
win.title("Credit Loan Risk ")
win.geometry("600x500")
win.configure(bg='#8df2cd')
Label(win ,text="Loan Approval Prediction ",font='Verdana 20 bold').place(x=100,y=80)



str1 = random.randint(80, 95)
str2 = random.randint(60, 80)
str3 = random.randint(1, 80)

if int(str3)>=50:

    btn = Button(win, text="   ", font='Verdana 10 bold', width="10")
    btn.place(x=150, y=300)
    btn.config(background="green")
    btn = Button(win, text="Normal User ", font='Verdana 10 bold', width="10")
    btn.place(x=150, y=200)
    print(str3)


else:
   print(str3)
   btn2 = Button(win, text="", font='Verdana 10 bold', width="10")
   btn2.place(x=310, y=300)
   btn2.config(background="Red")
   btn = Button(win, text="Risk User" , font='Verdana 10 bold', width="10")
   btn.place(x=150, y=300)


win.mainloop()
from fileinput import filename
from tkinter import *
import tkinter as tk
import tkinter
from tkinter import ttk, messagebox
from PIL import Image, ImageTk
import random
import pymysql
import pandas as pd
import csv
from csv import writer
from tkinter import simpledialog
from tkinter.filedialog import askopenfilename
import mysql.connector as mysql

class ViewData:
    def __init__(self):
        def regression():
            wingrid1 = Tk()
            wingrid1.title("View Dataset  Window")
            wingrid1.configure(bg='#6495ED')
            wingrid1.maxsize(width=1400, height=900)
            wingrid1.minsize(width=1400, height=900)
            userinput = simpledialog.askstring(title="Regression Values", prompt="Enter Area Name :")
            print(userinput)
            con = mysql.connect(host="localhost", user="root", password="root", database="credit")

            cur = con.cursor()

            cur.execute("select sum(price),avg(price),VARIANCE((price) from Land where Area=%s ",
(userinput))
            row = cur.fetchone()
            sumarea, avgarea, vararea = row
            print(sumarea)
            print(avgarea)
            print(vararea)
            con = mysql.connect(host="localhost", user="root", password="root", database="credit")
            cur = con.cursor()

            cur.execute("insert into  Regre(area, total,mean,vari) values (%s,%s,%s,)", (userinput, sumarea,
avgarea))
            str1 = "Total Value : " + str(sumarea) + "\nMean Value : " + str(avgarea) + "\nVariance Value : " + str(
                vararea) + " saved successfully"

            messagebox.showinfo("Record Saved", str1, parent=wingrid1)

        def feature():
            wingrid1 = Tk()
            wingrid1.title("View Dataset  Window")
            wingrid1.configure(bg='#6495ED')
            wingrid1.maxsize(width=1400, height=900)
            wingrid1.minsize(width=1400, height=900)

            with open(filename) as file:
                reader = csv.reader(file)
                r = 1
                for row in reader:
                    c = 0
                    for col in row:

                        if (c == 1 or c == 4 or c == 6):
                            label = Label(wingrid1, width=10, height=2, text=col, relief=tkinter.RIDGE)
                            label.grid(row=r, column=c)
                        c = c + 1

                    r += 1

        def upload(winland):
            f_types = [('CSV Files', '*.csv'), ('Xlsx Files', '*.xlsx')]
            filename = askopenfilename(filetypes=f_types)

            if filename.endswith('.xlsx'):
                file = pd.read_excel(filename)
                file.to_csv(filename.rstrip('.xlsx') + ".csv")
                filename = filename.rstrip('.xlsx') + ".csv"

            con = mysql.connect(host="localhost", user="root", password="root", database="credit")
            cur = con.cursor()
            cur.execute("delete from loan")

            con.commit()
            count = 0
            with open(filename, newline="") as file:
                reader = csv.reader(file)
                r = 1
                for row in reader:
                    count += 1

                    sno, employeename, location, companyname, experience, salary, previousloan, EMI=row
                    cur.execute(
                        "insert into
loan(sno,employeename,location,companyname,experience,salary,previousloan,EMI) values
(%s,%s,%s,%s,%s,%s,%s,%s)",
                        (sno, employeename, location, companyname, experience, salary, previousloan, EMI))
                con.commit()

                con.close()
                s1 = "Record uploaded Successfully"

            messagebox.showinfo("Record Uploaded Successfully", filename)

        def viewdataset():
            wingrid = Tk()
            wingrid.title("View Dataset Window")
            wingrid.configure(bg='#6495ED')
            wingrid.geometry("1400x900")

            # -------------------------
            # Main frame to hold canvas
            # -------------------------
            main_frame = Frame(wingrid)
            main_frame.pack(fill=BOTH, expand=1)

            # -------------------------
            # Canvas to allow scrolling
            # -------------------------
            my_canvas = Canvas(main_frame)
            my_canvas.pack(side=LEFT, fill=BOTH, expand=1)

            # -------------------------
            # Scrollbars
            # -------------------------
            v_scroll = ttk.Scrollbar(main_frame, orient=VERTICAL, command=my_canvas.yview)
            v_scroll.pack(side=RIGHT, fill=Y)

            h_scroll = ttk.Scrollbar(wingrid, orient=HORIZONTAL, command=my_canvas.xview)
            h_scroll.pack(side=BOTTOM, fill=X)

            my_canvas.configure(yscrollcommand=v_scroll.set, xscrollcommand=h_scroll.set)
            my_canvas.bind('<Configure>', lambda e: my_canvas.configure(scrollregion=my_canvas.bbox("all")))

            # -------------------------
            # Frame inside canvas to hold table
            # -------------------------
            display_frame = Frame(my_canvas)
            my_canvas.create_window((0, 0), window=display_frame, anchor="nw")
```

```
# -------------------------
# Database connection
# -------------------------
con = mysql.connect(host="localhost", user="root", password="root", database="credit")
cur = con.cursor()

# Fetch all data as-is (no preprocessing)
cur.execute("SELECT * FROM loan")
data = cur.fetchall()

# -------------------------
# Column headers
# -------------------------
columns = ["sno", "employeename", "location", "companyname", "experience", "salary",
"previousloan", "EMI"]
for col_index, col_name in enumerate(columns):
    header_label = Label(display_frame, text=col_name, relief=tkinter.RIDGE, width=23, height=2,
                         bg='lightgrey', font=('Verdana', 10, 'bold'))
    header_label.grid(row=0, column=col_index, sticky="nsew")

# -------------------------
# Display data starting from row=1
# -------------------------
r = 1
for row_data in data:
    c = 0
    for cell in row_data:
        label = Label(display_frame, width=23, height=2, text=cell, relief=tkinter.RIDGE)
        label.grid(row=r, column=c, sticky="nsew")
        c += 1
    r += 1

con.close()
```