Data Intensive Computing – CSE 587A

Project Phase – 3

Problem Statement: To develop a machine learning model that can classify credit scores using information gathered from banks and credit-related datasets.

Code Documentation

Phase 1

- In the first phase of the project, the code contains Data Pre-processing and Exploratory Data Analysis.
- In the first half of the code of the Jupyter file we imported libraries, performed data loading, and used data pre-processing steps.
- Data cleaning involves 10 steps:
  1. Dropping Unwanted Columns: Columns like ID, Customer ID, Month, Age, Monthly Inhand Salary, Credit Mix, Credit History Age, Payment Behaviour, Name, and SSN are dropped using the drop () function.
  2. Datatype Conversion: Datatype conversion is performed using pandas functions to ensure compatibility and efficiency for analysis.
  3. Column Renaming: Columns are renamed for easy readability using the `rename()` function.
  4. Handling Null Values: Null values for Annual Income, Num of Loan, Type of Loan, Num of Delayed Payments, Changed Credit Limit, Num of Credit Inquiries, Outstanding Debt, Amount invested monthly, and Monthly Balance are handled using median tendency for numeric features and 'unknown' for categorical features.
  5. Cleaning Numeric Data: Numeric data is cleaned using numpy for Num of Loan, Changed Credit Limit, Delay from due date, and Amount invested monthly.
  6. Standardizing Text Data: Text data is standardized using pandas to remove punctuations and convert text into lowercase.
  7. Handling Outliers: Outliers for Annual Income and Amount Invested Monthly are handled by defining bounds and filtering out outliers. A box plot is plotted to visualize the effect of outlier removal.
  8. Encoding Categorical Data: Label encoding is initialized for the attribute Credit Score and One-hot Encoding is applied to the attribute occupation using pandas. The encoded columns are then joined back to the original data frame.
  9. Feature Engineering: Feature Engineering involves calculating the ratio of Annual Income to Amount Invested Monthly using a lambda function. Loan affordability of Annual Income is estimated using pandas. The updated data frame is displayed to verify the new features.
  10. Standardizing Numerical Features: Numerical features are standardized for applying PCA (Principal Component Analysis). After applying PCA, the data is reduced to two dimensions for illustration.
- Next, we performed Exploratory Data Analysis and plotted some graphs to understand the data. Visualization techniques used:

1. Exploratory data analysis is an excellent technique for deriving conclusions and understanding the data.
2. To get the statistics of data, we used data. describe ().
3. The histogram plotting code output is utilized to visualize the distribution of numerical data within a dataset.
4. The scatter plot output examines the relationship between two variables - Annual Income and Credit Score.
5. The count plot representation displays the frequency distribution of categorical variables - Occupation and Credit Score.
6. The scatter plot visualizes the relationship between numerical variables - Annual Income and Num of Loan, Annual Income and Monthly Balance.
7. The scatter plots visualize the relationship between pairs of variables - Num of Loan and Num of Delayed Payments.
8. The barplot compares the average annual income across different Credit Score categories.
9. The heatmap visualizes the relationship between categorical variables - Occupation and Credit Score.
10. The lower triangular heatmap represents the correlation matrix between different variables in a dataset.

Phase 2

- In the second phase, we performed model-building and evaluated the accuracy of the model using six different algorithms. They are:

Logistic Regression:

- Accuracy: 59.2%
- Precision: Ranges from 0.49 (Class 0) to 0.64 (Class 1)
- Recall: Ranges from 0.22 (Class 0) to 0.81 (Class 2)
- This model has the lowest accuracy among the algorithms compared. Its recall for Class 0 is notably low, indicating a significant number of false negatives. It performs best in identifying Class 2 instances.

K-Nearest Neighbors (KNN):

- Accuracy: 68.1%
- Precision: Ranges from 0.53 (Class 0) to 0.72 (Class 2)
- Recall: Ranges from 0.57 (Class 0) to 0.72 (Class 2)
- KNN shows a moderate improvement over Logistic Regression, with better overall accuracy and balanced precision and recall scores.

Gradient Boosting:

- Accuracy: 68.9%
- Precision: Ranges from 0.61 (Class 0) to 0.74 (Class 2)
- Recall: Ranges from 0.48 (Class 0) to 0.80 (Class 2)

- Gradient Boosting's performance is close to that of KNN but shows a slightly higher precision for Class 2 and better recall for Class 2, making it more balanced in terms of precision and recall.

Random Forest:

- Accuracy: 77.5%
- Precision: Ranges from 0.73 (Class 0) to 0.78 (Class 2)
- Recall: Ranges from 0.66 (Class 0) to 0.81 (Class 2)
- Random Forest outperforms the other models in accuracy and maintains relatively high precision and recall across all classes.

Decision Tree:

- Accuracy: 69.7%
- Precision: Ranges from 0.60 (Class 0) to 0.73 (Class 2)
- Recall: Ranges from 0.62 (Class 0) to 0.72 (Class 2)
- The Decision Tree has a comparable performance to KNN but is outperformed by the Random Forest, which is expected given that the Random Forest is an ensemble of Decision Trees.

XGBoost:

- Accuracy: 73.3%
- XGBoost shows higher accuracy than KNN, Decision Tree, and Gradient Boosting, but is slightly behind Random Forest.
- XGBoost has balanced precision and recall, indicating strong performance, particularly for Class 2, where it has a recall of 0.80 and a precision of 0.74.

Phase 3

- We have built a Spyder tool app.py flask file for credit risk prediction which is a web framework written in Python for server-side scripting.
- In this website the user will give 17 inputs to the model and the output whether credit is poor or standard or good is displayed on the result page.
- We have used Spyder, flask module, python and HTML to create this web application

User Interface Running Instructions

- app – our flask application name
- model – it will contain the model which we build
- @app.route() – it is a decorator that can redirect to different functions.
- one for routing to the home page (‘/’)
- route to the prediction page (‘/Prediction’)
- route to the same home page itself (‘/Home’)
- routing to the result page (‘/predict’)
- render_template () – it is used for render our html page from the templates folder
- predict() – is taking the values form the prediction page and storing it into a variable and then we are creating a DataFrame along with the values and 17 independent features and finally

we predict the values using or loaded model which we build and storing the output in a variable and returning it to the result page.

- app.run(debug=False) – for running our app
- We have created an HTML page which take the 16 input values from the user and pass it to the flask application. It will again take the result from the flask and display it to the user.
- The user needs to open the webpage and need to enter the credit risk-dependent features and the model will give the result on next page whether the credit risk is poor, standard or good.

Machine Learning Model Used to Create a Web Application

In phase 2 we used 6 different models to predict the accuracy of the credit risk level. They are Random Forest Classifier, Decision Tree Classifier, KNN Classifier, Linear Regression, XG Boost and Gradient Boosting. To create a web application, we have used the Random Forest Classifier Algorithm because out of all other models Random Forest gave more accuracy i.e., 77% accuracy.

Reasons for Choosing Random Forest Classifier

1. Robustness to Overfitting: The Random Forest algorithm is inherently resistant to overfitting, particularly in comparison to other algorithms. This is due to its method of averaging multiple decision trees, each trained on different subsets of the same data, which generally leads to more generalized results.

2. Capability to Handle Various Data Types: Our dataset includes a mix of both categorical and numerical inputs. Random Forest can handle this variety without the need for extensive preprocessing or transformations, simplifying the data preparation pipeline and reducing the potential for data leakage.

3. Feature Importance Analysis: A significant advantage of using Random Forest is its ability to provide insights into the importance of each feature in making predictions. This is critical for our application, as it allows us to explain which factors are most influential in predictions, enhancing transparency and trust in the model outputs.

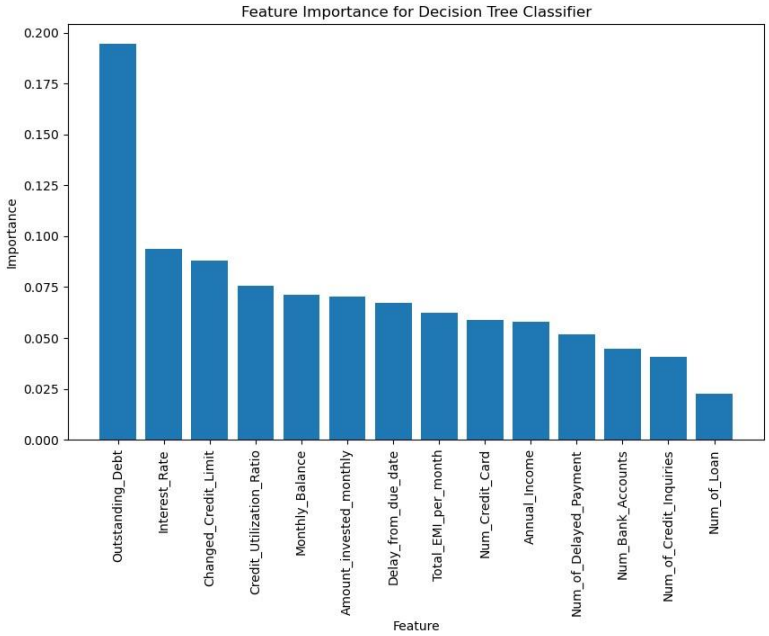
Model Evaluation

We employed several metrics to evaluate the model's performance, including accuracy, precision, recall, and the F1 score. Additionally, we used cross-validation techniques to ensure that our model performs consistently across different data subsets.

- Accuracy: 77.5%
- Precision: Ranges from 0.73 (Class 0) to 0.78 (Class 2)
- Recall: Ranges from 0.66 (Class 0) to 0.81 (Class 2)
- Random Forest outperforms the other models in accuracy and maintains relatively high precision and recall across all classes.

Overall, the Decision Tree Model demonstrates a solid baseline performance across the different classes. Notably, its effectiveness is most pronounced for Class 2, where both precision and recall metrics are highest compared to other courses. This suggests that the model is particularly great at correctly identifying Class 2 instances with a high level of precision and recall.

Feature Importance:



This visualization offers insights into which features the model found most influential in predicting the target variable. The most important feature is Outstanding Debt, followed by Interest Rate, Credit Limit, Credit Utilization Ratio, Monthly Balance, Amount Invested Monthly, Delay from due date, Total EMI per month, Num of Credit Cards, Annual Income, Num of Delayed Payments, Num of Bank Accounts, Num of Credit Inquiries, and Num of Loan.

This intelligence is critical for understanding the driving factors behind the model's predictions. For example, the high importance of Outstanding Debt suggests that this feature significantly influences the likelihood of a credit event, such as a default, which is key information for credit risk analysis. Features with lower importance might have a lesser impact on the model's decision-making process or may providing redundant information that is already captured by more important features. This insight into feature importance can guide further data collection, feature engineering, and the refinement of the model to improve performance and interpretability.

These are the reasons why we chose the Random Forest Classifier Algorithm in our Credit Risk Prediction Web Application.

Future Recommendations on our problem statement and analysis

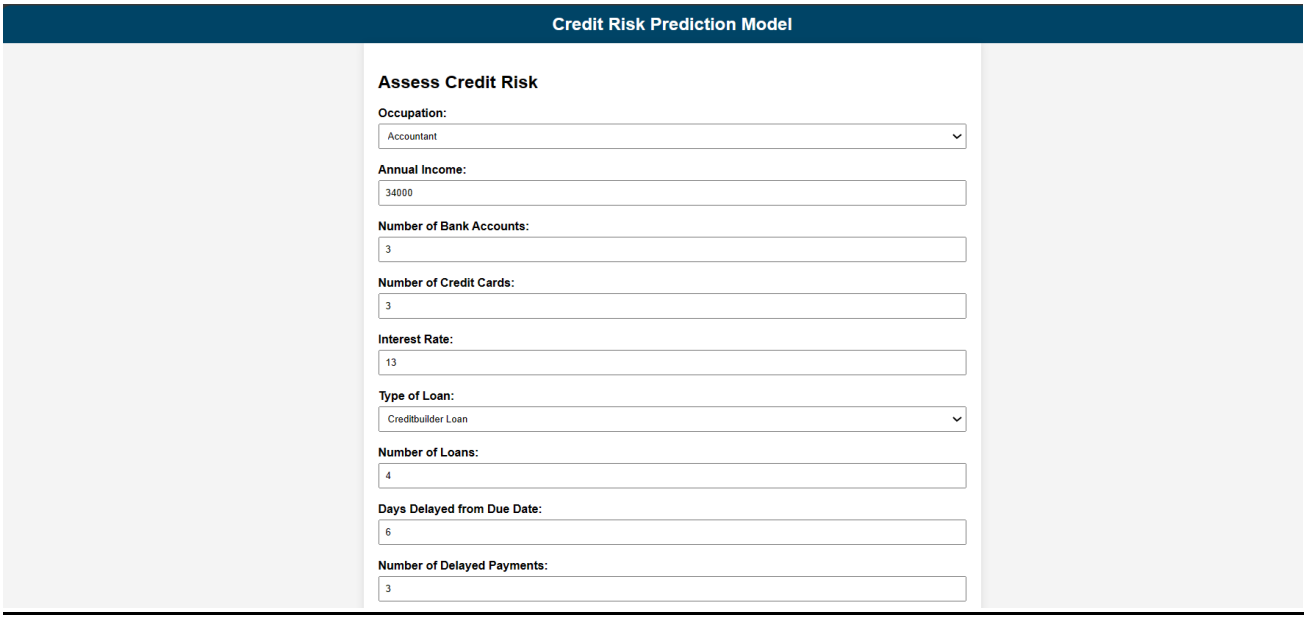
- In our project, we aim to develop a machine-learning model capable of accurately classifying credit scores based on various financial data. By utilizing techniques such as data preprocessing, feature engineering, and model training, we strive to create a predictive model that can categorize individuals into three risk categories: good, standard, or poor. This model will enable financial institutions to make informed decisions regarding credit restrictions, loan approvals, and interest rates, ultimately reducing the risk associated with lending.

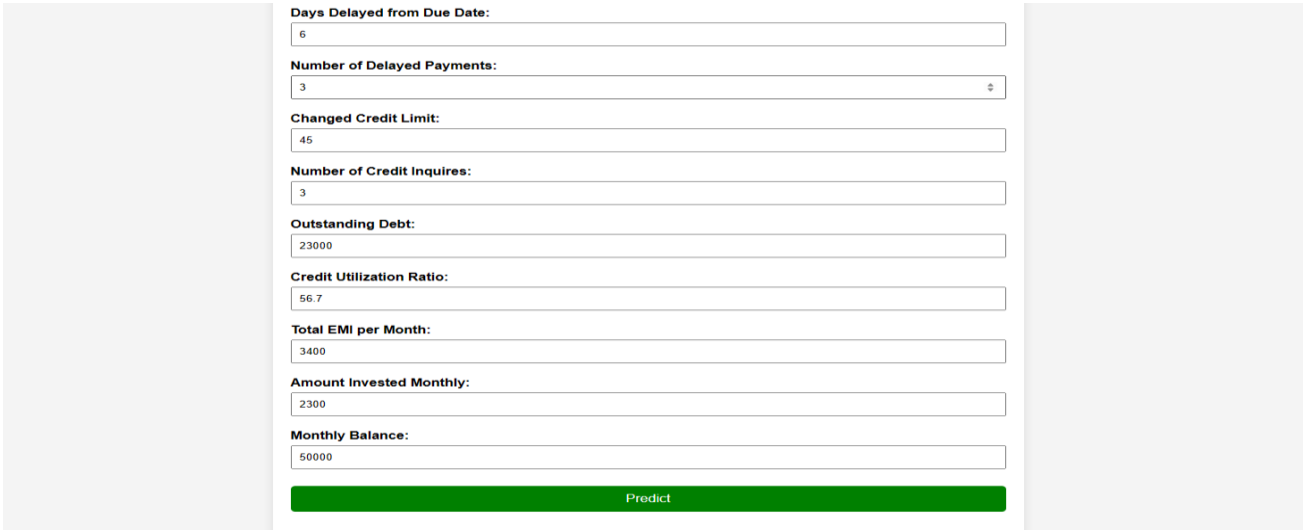
- Our project addresses the critical need for accurate credit scoring in the financial industry. By leveraging machine learning algorithms, we can analyze vast amounts of data, including income, payment history, number of loans, and credit scores, to develop a comprehensive

understanding of a customer's creditworthiness. This predictive model will empower lenders to tailor their financial offerings to meet the needs of different consumer segments, mitigate risks associated with loan defaults, and optimize their lending practices.

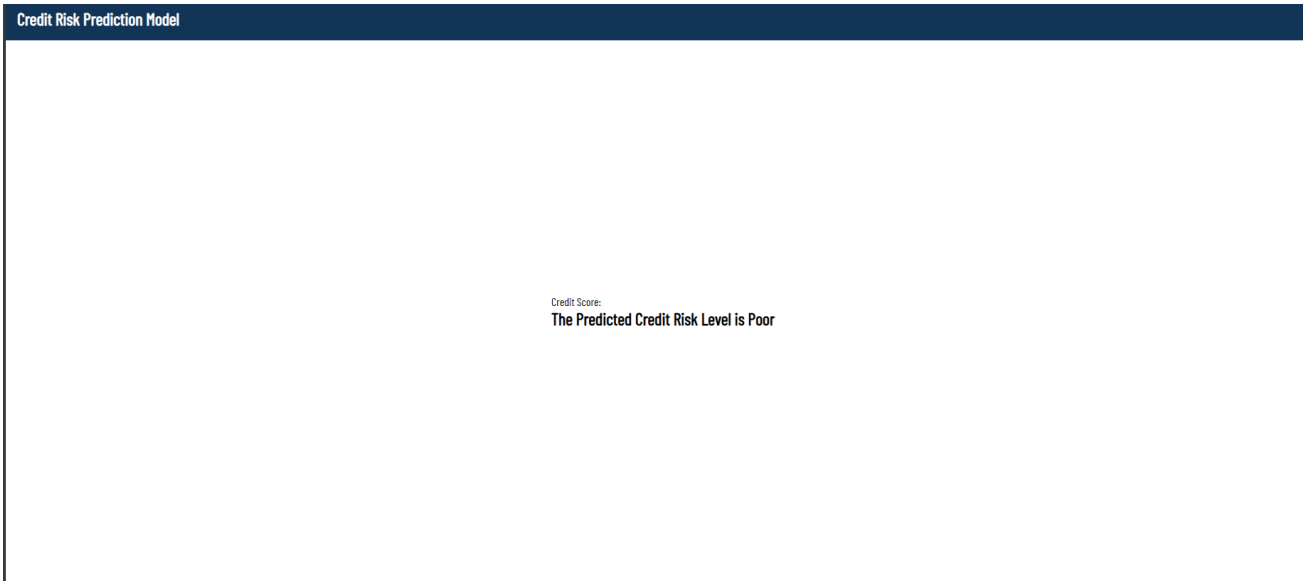
- Our analysis will provide valuable insights and recommendations for financial institutions. Users will learn how machine learning models can enhance credit risk assessment processes, leading to more informed lending decisions. The model's predictions can guide lenders in setting appropriate credit limits, determining interest rates, and approving loans with confidence. Additionally, our project opens avenues for further research, such as exploring the incorporation of alternative data sources or refining the model to improve its accuracy and robustness. Furthermore, we can extend our project by developing a user-friendly interface or integrating the model into existing banking systems to streamline the credit evaluation process for financial institutions.

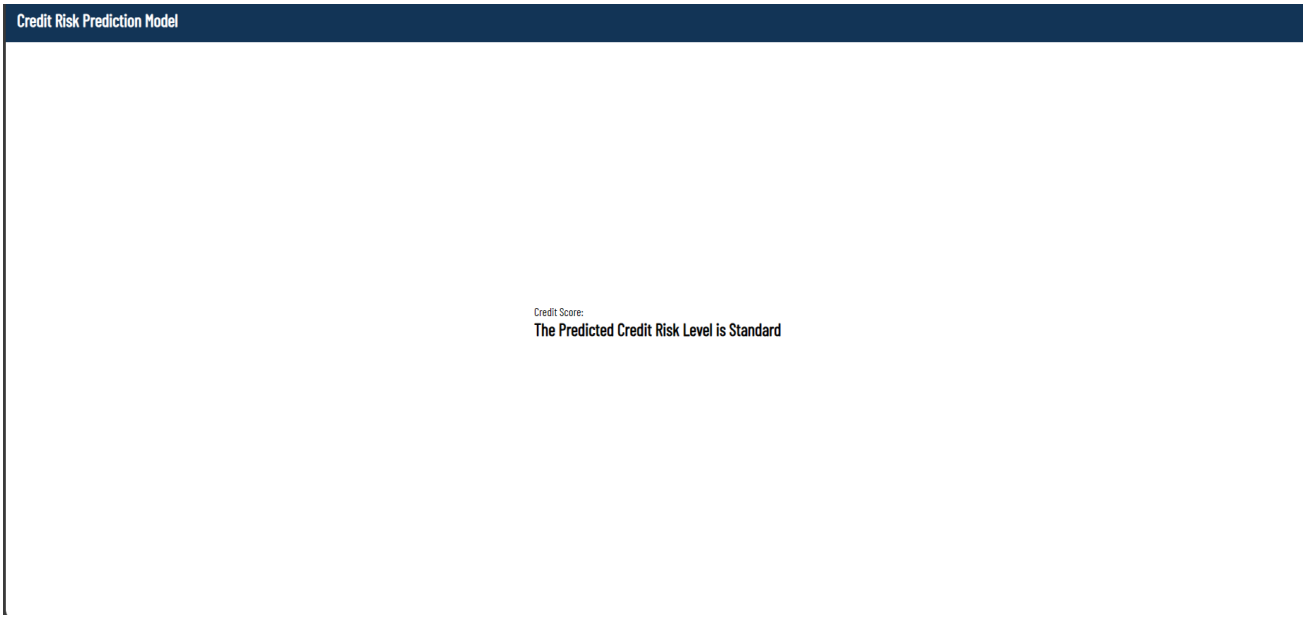
Deployment of Machine Learning Model using Flask - Credit Risk Prediction Web Application





The above is the index page of my credit risk prediction web application. The page features a detailed form where users can input various financial metrics, such as their occupation, annual income, number of bank accounts and credit cards, and more. The form also captures specifics about their borrowing history, including types of loans, payment delays, and credit inquiries. Once all the required fields are filled, a user can hit the 'Predict' button to receive an assessment of their credit risk based on the information they've provided. This tool is designed to help users understand their financial standing and is particularly useful for financial institutions assessing potential borrowers.





These are the output pages of my credit risk prediction web application. After users input their financial information and click the "Predict" button on the index page, they are directed to one of these output pages, which display the assessment of their credit risk. Here's how the outputs appear:

- Standard Credit Risk Level: This output is shown if the analysis determines that the user has a typical or average credit risk. It indicates a balanced financial profile that may not present a significant risk to lenders.
- Poor Credit Risk Level: This output is provided if the user's financial data indicates a higher risk of defaulting on loans. It suggests there might be significant issues with the user's financial habits or current financial status, such as high levels of debt, poor payment history, or other negative factors.
- Good Credit Risk Level: This result is displayed when the user demonstrates a strong financial position, characterized by high levels of financial responsibility, low debt levels, and a consistent payment history. This indicates that the user is a low-risk borrower, potentially qualifying for better borrowing terms.

Each result page is designed to communicate the credit risk level based on the analysis of the detailed financial information provided. This helps users understand their financial health in the context of credit risk and could assist them in making informed decisions about their credit and loans.