CREDIT CARD DEFAULT PREDICTION

High Level Design Document

Prepared By:
Binaya Kumar Pradhan

Abstract

Credit risk plays a significant role in the banking industry's operations, which encompass services like providing loans, credit cards, investments, mortgages, and various financial offerings. Among these, credit card services have witnessed substantial growth in recent years. However, the increasing number of credit card users has presented a challenge for banks in the form of a rising default rate. In response to this issue, data analytics can offer viable solutions for managing credit risks effectively.
This project focuses on the development of a predictive model designed to assess whether a given credit card holder is likely to default in the upcoming month. The model utilizes demographic and behavioral data from the preceding six months to make these predictions.

Table of Contents

1. Introduction

1.1. What Is HLD?

The High-Level Design (HLD) Document serves several important purposes for the project. Its primary objective is to provide additional detail to the current project description, making it suitable for coding. By doing so, the document helps identify any contradictions or inconsistencies before the actual coding phase begins. Moreover, it serves as a valuable reference manual, illustrating how different modules interact with each other at a high level within the project. Overall, the HLD Document ensures a clear and well-defined structure for the coding process and enhances the project's efficiency and accuracy.

1.2. Scope

Scope of the HLD Documentation:

- Presents the system structure, including:
    - Database architecture.
    - Application architecture (layers).
    - Application flow (Navigation).
    - Technology architecture.
- Uses non-technical to mildly-technical terms for better understanding by system administrators.

2. General Description

2.1. Problem Statement

The problem at hand is to develop a machine learning solution that can predict the probability of credit default based on credit card owner's characteristics and payment history. With the increasing financial threats and the need for commercial banks to assess credit risk accurately, there is a demand for a reliable predictive model that can assist in identifying clients at a higher risk of defaulting on credit.

2.2. Proposed Solution

- The challenge lies in analyzing the various factors and patterns in credit card owner's characteristics and payment history to determine the likelihood of default. The model should consider features such as credit limit, age, bill amounts, payment amounts, and payment history indicators like the status of payments in previous months.
- The solution should go beyond traditional statistical analysis by leveraging machine learning algorithms to uncover complex relationships and patterns that contribute to credit default. The model should be capable of handling large volumes of data, processing it efficiently, and providing accurate predictions in real-time.
- The goal is to build a robust and scalable solution that can be easily integrated into existing banking systems. The model should be capable of accurately predicting credit default probability, enabling banks to make informed decisions and take proactive measures to mitigate credit risk.
- By addressing this problem, the project aims to enhance the credit assessment process for commercial banks, leading to more effective risk management, reduced financial losses, and improved overall stability in the banking industry.

2.3. Data Requirements

Dataset: https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset

This dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005. There are 25 variables:

- **ID**: ID of each client
- **LIMIT_BAL**: Amount of given credit in NT dollars (includes individual and family/supplementary credit
- **SEX**: Gender (1=male, 2=female)
- **EDUCATION**: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown)
- **MARRIAGE**: Marital status (1=married, 2=single, 3=others)
- **AGE**: Age in years
- **PAY_1**: Repayment status in September, 2005 (-1=pay duly, 1=payment delay for one month, 2=payment delay for two months, … 8=payment delay for eight months, 9=payment delay for nine months and above)

- **PAY_2**: Repayment status in August, 2005 (scale same as above)
- **PAY_3**: Repayment status in July, 2005 (scale same as above)
- **PAY_4**: Repayment status in June, 2005 (scale same as above)
- **PAY_5**: Repayment status in May, 2005 (scale same as above)
- **PAY_6**: Repayment status in April, 2005 (scale same as above)
- **BILL_AMT1**: Amount of bill statement in September, 2005 (NT dollar)
- **BILL_AMT2**: Amount of bill statement in August, 2005 (NT dollar)
- **BILL_AMT3**: Amount of bill statement in July, 2005 (NT dollar)
- **BILL_AMT4**: Amount of bill statement in June, 2005 (NT dollar)
- **BILL_AMT5**: Amount of bill statement in May, 2005 (NT dollar)
- **BILL_AMT6**: Amount of bill statement in April, 2005 (NT dollar)
- **PAY_AMT1**: Amount of previous payment in September, 2005 (NT dollar)
- **PAY_AMT2**: Amount of previous payment in August, 2005 (NT dollar)
- **PAY_AMT3**: Amount of previous payment in July, 2005 (NT dollar)
- **PAY_AMT4**: Amount of previous payment in June, 2005 (NT dollar)
- **PAY_AMT5**: Amount of previous payment in May, 2005 (NT dollar)
- **PAY_AMT6**: Amount of previous payment in April, 2005 (NT dollar)
- **default. payment. next. Month**: Default payment (1=yes, 0=no)

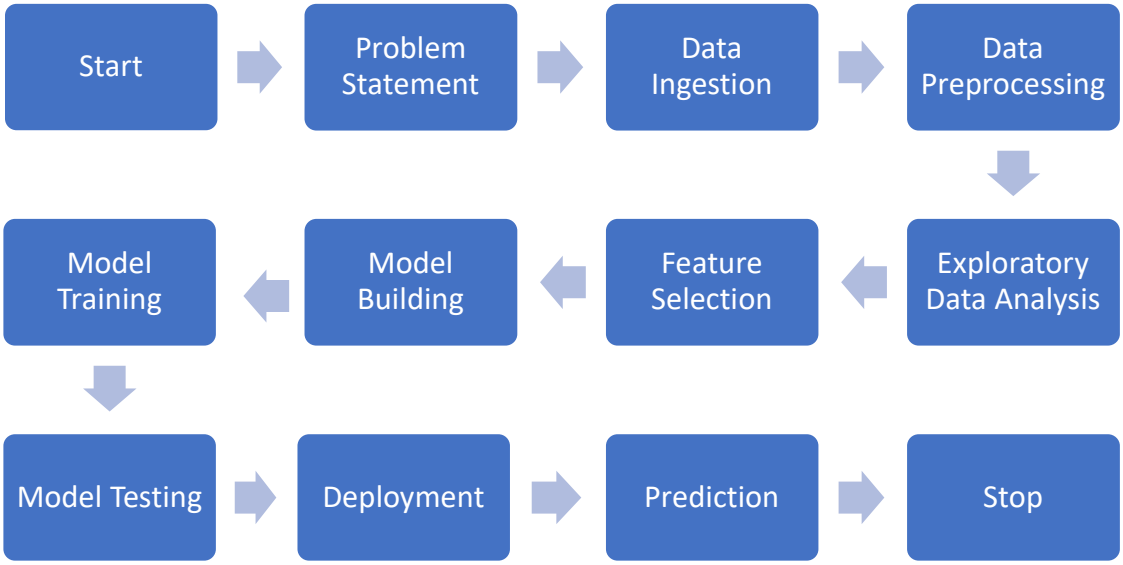
2.4. Tools Used

Python programming language and various frameworks such as NumPy, Pandas, Scikit-learn are used to build the whole model.

- Jupyter Notebook is used as IDE.
- For visualization of the plots, Matplotlib and Seaborn are used.
- AWS Elastic Beanstalk is used for deployment of the model.
- Front end development is done using HTML/CSS
- Python is used for backend development.
- GitHub is used as version control system.

3. Design Details

3.1. Process Flow



3.2. Application Compatibility & Resource Utilization

- **Application Compatibility**: In this project, all the various components will communicate with each other using Python as their interface. Each component has a specific role to play, and Python is responsible for ensuring seamless information exchange among them.
- **Resource Utilization:** Whenever a task is executed, it tends to utilize the available processing power to complete that task efficiently. This approach optimizes the utilization of system resources during task execution.

3.3. Data Ingestion

- Raw credit card data is collected from various sources in Taiwan, including information about credit card clients, their demographics, repayment history, bill statements, and payment amounts.

3.4. Preprocessing

- Data preprocessing is performed to handle missing values, resolve inconsistencies, and transform the data into a suitable format for analysis and modelling. Feature engineering is applied to create new relevant features.

3.5. Data Analysis

- Exploratory Data Analysis (EDA) is conducted to gain insights into the data distribution, detect outliers, and identify relationships between variables.
- Data visualizations, statistical summaries, and data profiling are used to understand the data characteristics.

3.6. Feature Selection

- Based on the data analysis, relevant features (variables) are selected for building the predictive models. This step aims to choose the most informative attributes for predicting credit card defaults.

3.7. Logging & Exception Handling

- **Logging:** Think of logging as a digital diary for our application. It diligently records significant events and actions, enabling us to monitor the system's behaviour, identify issues, and gain insights into its performance. Logging is a valuable tool for debugging, real-time monitoring, and enhancing the application's dependability.
- **Exception Handling:** Exception handling ensures that our application gracefully responds to unexpected problems or errors, preventing complete breakdowns. It offers informative error messages for users, safeguards their data, and enhances the overall user experience. This feature adds a layer of resilience to the application, making it more user-friendly and reliable.

3.8. Model Building

- Machine learning algorithms (logistic regression, decision trees, SVM) are applied to build individual base models using the selected features and the historical data.
- Ensemble techniques (bagging, boosting, stacking) are utilized to construct ensemble models, combining multiple base models to enhance predictive accuracy and robustness.

3.9. Model Training

- The individual base models and ensemble models are evaluated using the training dataset.
- Evaluation metrics, such as accuracy, precision, recall, F1-score, and ROC-AUC, are calculated to assess the models' effectiveness in predicting credit card defaults.
- The predictions of the base models and ensemble models are combined using the specific rules of the ensemble method (e.g., averaging, voting, stacking).
- Ensemble combination creates a more accurate and robust prediction compared to individual base models.

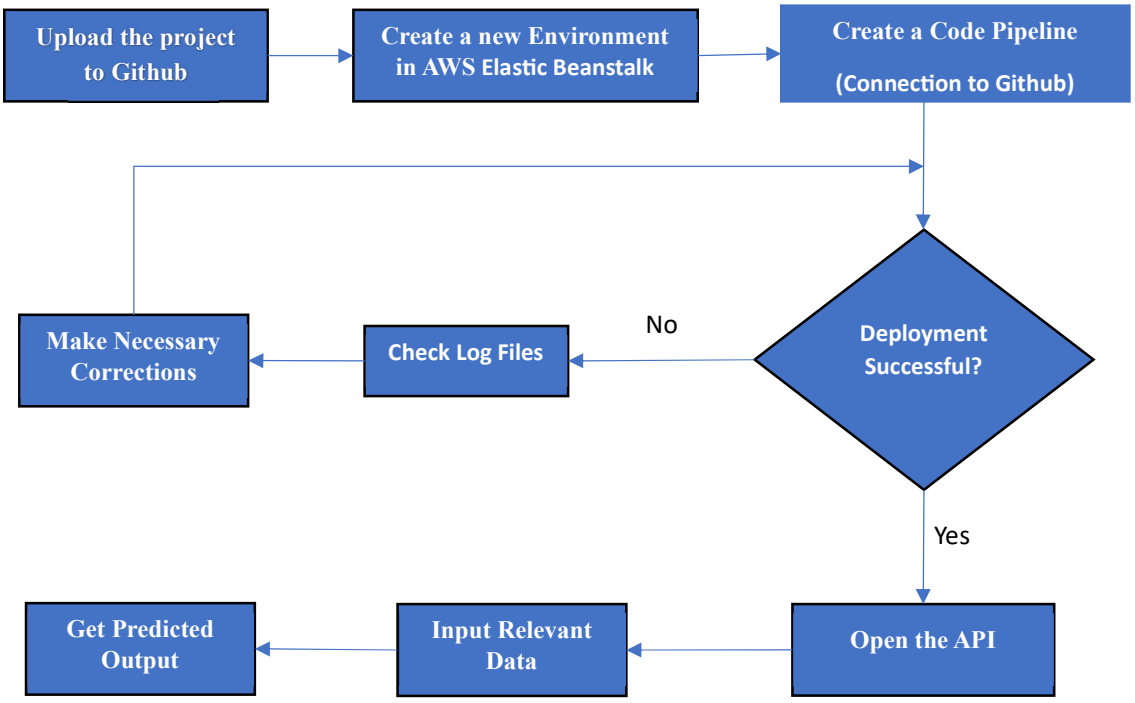
3.10. Model Testing

- The best-performing ensemble model is selected as the final predictive model for credit card default prediction.
- The Test dataset is evaluated using the best performing model.
- Evaluation metrics, such as accuracy, precision, recall, F1-score are calculated to assess the prediction credit card defaults.

3.11. Performance & Reusability

- **Performance:** The Credit Card Default Prediction App is designed to help anticipate whether a specific customer might have trouble making their payments in the next month. This forecast is based on the customer's background information and how they've been using their credit over the past six months in this project. This tool enables the business to take proactive measures and make plans to assist individual customers accordingly.
- **Reusability:** The code and components employed in this project are designed with reusability in mind. They are structured to be easily adaptable and reusable without encountering any complications or issues.

3.12. Deployment



- The project is uploaded to a GitHub repository for version control and collaboration.
- It is connected to AWS Elastic Beanstalk through a CodePipeline, streamlining the deployment process.
- The model is then deployed on AWS Elastic Beanstalk, making it accessible for use.

3.13. Prediction

- To interact with the deployed Flask API, users can access it via the generated IP address.
- After accessing the API, user needs to fill the fields to get the predicted result.

4. Conclusion

This application serves as a valuable tool for financial institutions to foresee whether a specific customer is at risk of defaulting on payments in the upcoming month. It achieves this by analyzing a range of customer demographic and behavioural data. By making these predictions, the application empowers financial institutions to take proactive measures and make informed decisions to prevent potential defaults before they occur. This not only helps in safeguarding the financial interests of the institution but also contributes to maintaining healthy financial relationships with customers.