High Level Design (HLD)

Credit Card Default Prediction

Document Version Control

Build a solution that should be able to predict the probability of credit default based on the credit card owner's characteristics and payment history.

Date Issued	Version	Description	Author
23 March 2023	1	Initial HLD	Chetna Sharma

Contents

Abstract

The project follows a classical machine learning approach, encompassing essential tasks such as Data Exploration, Data Cleaning, Feature Engineering, Model Building, and Model Testing. Diverse machine learning algorithms will be explored and evaluated to identify the most suitable model for the problem at hand.

The significance of this project lies in its potential to enhance the risk assessment capabilities of commercial banks and financial institutions, leading to more prudent lending practices and increased financial stability. By accurately predicting credit default probabilities, banks can minimize potential losses and optimize their credit approval processes, ultimately benefiting both the financial institution and its customers.

1. Introduction

Why this High-Level Design Document?

The HLD will:

• present all of the design aspects and define them in detail

• describe the user interface being implemented

• describe the hardware and software interfaces

• describe the performance requirements

• include design features and the architecture of the project

• list and describe the non-functional attributes like:

o security

o reliability

o maintainability

o portability

o reusability

o application
compatibility

o resource utilization

o serviceability

Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture**,** application flow, and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators

Definitions

1. **Credit Risk:** The potential financial loss that may arise from a borrower's failure to repay a loan or meet its contractual obligations. In the context of this project, credit risk refers to the likelihood of a credit card owner defaulting on their payment obligations.
2. **Credit Default:** The failure of a borrower to make timely payments on a credit obligation, resulting in a breach of the loan agreement. In this project, credit default refers to the occurrence of a credit card owner failing to make required credit card payments.
3. **Feature Engineering:** The process of selecting, creating, or transforming features (variables) from raw data to enhance the performance of machine learning models.
4. **Model Building:** The creation and training of machine learning models using historical data to make predictions or classifications on new, unseen data.
5. **Model Testing/Evaluation:** The process of assessing the performance of machine learning models using evaluation metrics to measure how well they generalize to new, unseen data.
6. **Hyperparameter Tuning:** The process of finding the optimal values for the hyperparameters of a machine learning model to improve its performance.
7. **Scikit-learn:** An open-source machine learning library for Python that provides a wide range of algorithms and tools for data processing, model building, and evaluation.

2.General Description

Product Perspective

The credit risk prediction solution from a product perspective provides banks with valuable insights into their customers' creditworthiness. It facilitates risk mitigation, improves lending practices, enhances customer experience, and contributes to the overall financial stability and competitiveness of the financial institution.

Problem statement

Financial threats are displaying a trend in the credit risk of commercial banks as incredible improvement in the financial industry has arisen. In this way, one of the biggest threats faces by commercial banks is the risk prediction of credit clients. The goal is to predict the probability of credit default based on credit card owner’s characteristics and payment history.

Proposed Solution

The proposed solution is to develop a credit risk prediction system using machine learning algorithms. The system will analyze the credit card owner's characteristics and payment history to estimate the probability of credit default. The solution will follow a standard machine learning workflow, including data collection, preprocessing, feature engineering, model building, and model evaluation.

Technical Requirements

1. Scalability: The system should be able to handle a large volume of data and scale as the number of credit card owners and transactions increases over time. This ensures that the system remains efficient and responsive even with substantial growth in data.

2. Data Security: The system must adhere to strict data security measures to protect sensitive customer information and comply with

data protection regulations. Access controls, encryption, and secure data transmission are necessary components to safeguard data privacy.

3. Data Preprocessing Efficiency: Data preprocessing tasks, such as cleaning, normalization, and feature engineering, should be optimized to minimize processing time and enable real-time or near-real-time predictions.

4. Model Training Efficiency: The machine learning algorithms used for model training should be efficient and able to handle large datasets effectively. Training times should be reasonable to allow for quick iterations during model development.

5. Model Accuracy and Robustness: The system should achieve a high level of prediction accuracy while being robust enough to handle variations in data and generalize well to new, unseen data.

6. Model Explainability: The system should offer transparency in its predictions by providing explanations or feature importances for individual credit risk assessments. This is important for regulatory compliance and building trust with stakeholders.

7. Hyperparameter Tuning: The system should support automated hyperparameter tuning techniques to optimize model performance without manual intervention.

8. Real-Time Prediction Capability: The system should be able to make real-time credit risk predictions based on incoming credit card owner's data. This capability is essential for integration into loan approval processes and other real-time applications.

9. API or Web Service Integration: The system should be deployable as an API or web service, allowing seamless integration with other banking systems and applications.

10. Compatibility and Interoperability: The solution should be compatible with different operating systems and technologies used within the bank's infrastructure.

11. Monitoring and Logging: The system should have robust monitoring and logging mechanisms to track model performance, data inputs, and prediction outputs for debugging and maintenance purposes.

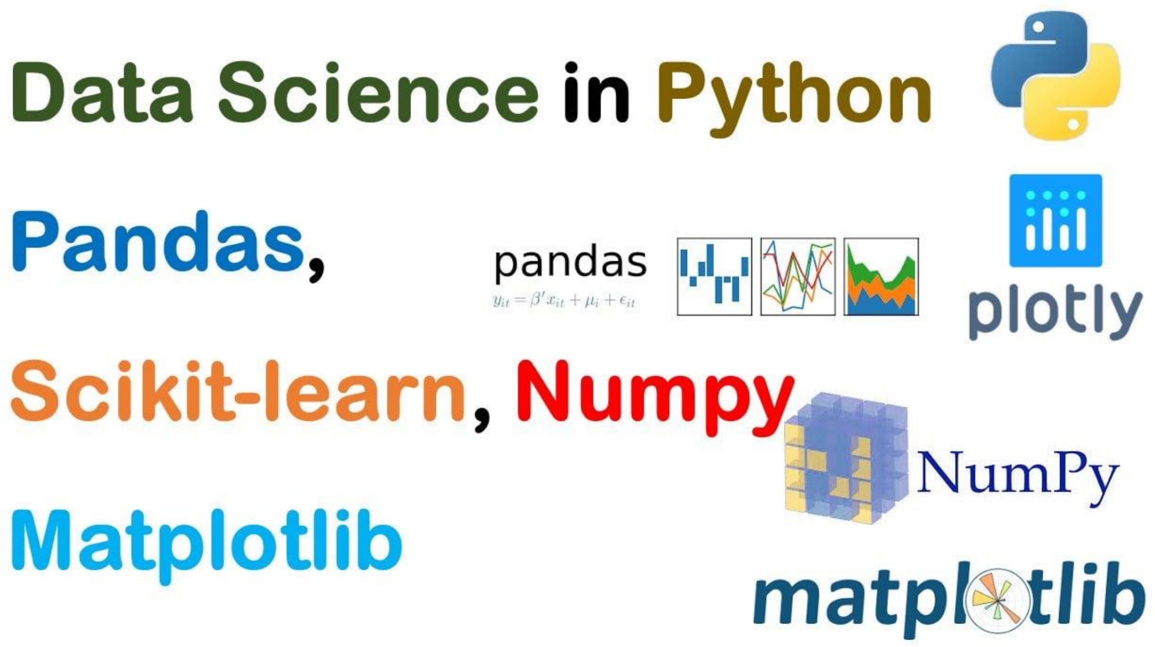
12. Version Control and Model Management: The system should support version control to manage different model versions, ensuring smooth transitions between updated models.

13. Documentation and Reporting: The project should have comprehensive documentation, including technical specifications, code explanations, and a report on the system's development, evaluation, and deployment.

14. Maintenance and Updates: The system should be designed for easy maintenance, allowing for regular updates and improvements as needed to keep the model accurate and up-to-date.

Tools used

1. **Python:** Python is the most popular programming language for data science and machine learning. Some of the essential Python libraries include:
   - NumPy and pandas for data manipulation and preprocessing.
   - scikit-learn for machine learning algorithms and evaluation.



2. **Data Visualization Libraries:** For visualizing data and model results, tools like Matplotlib, Seaborn, or Plotly are used.

3. **Data Storage:** A database (MongoDB) is used to store and manage large datasets.

4. **Version Control:** Git and platforms like GitHub or GitLab are used for version control, collaborative development, and tracking changes in the project.

5. **Web Frameworks:** framework like Flask is used for server-side development.

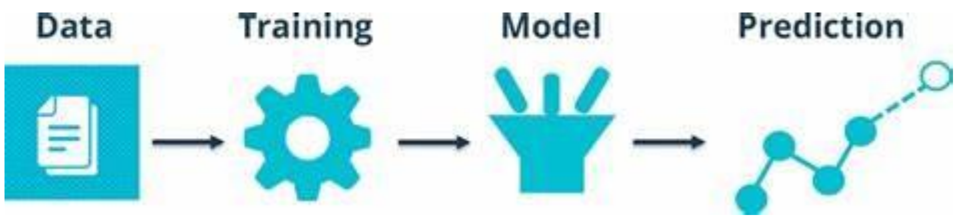
6. **Cloud Services:** The cloud platform (AWS)Amazon Web Services is used for scalability and hosting the web service.

7. **Text Editors/IDEs:** text editor or integrated development environment (IDE) used for coding is Visual Studio Code and JupyterNotebook.

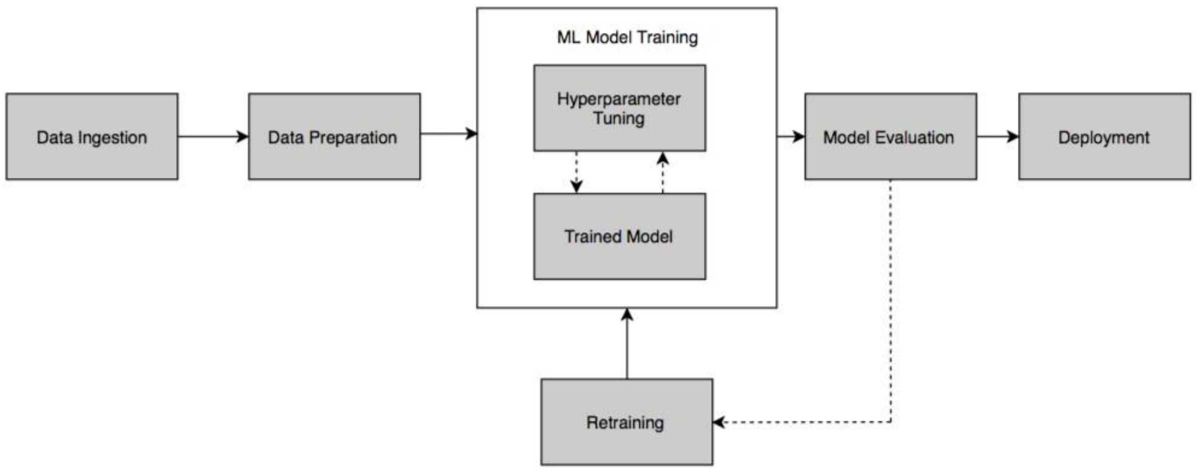
3.Design Details

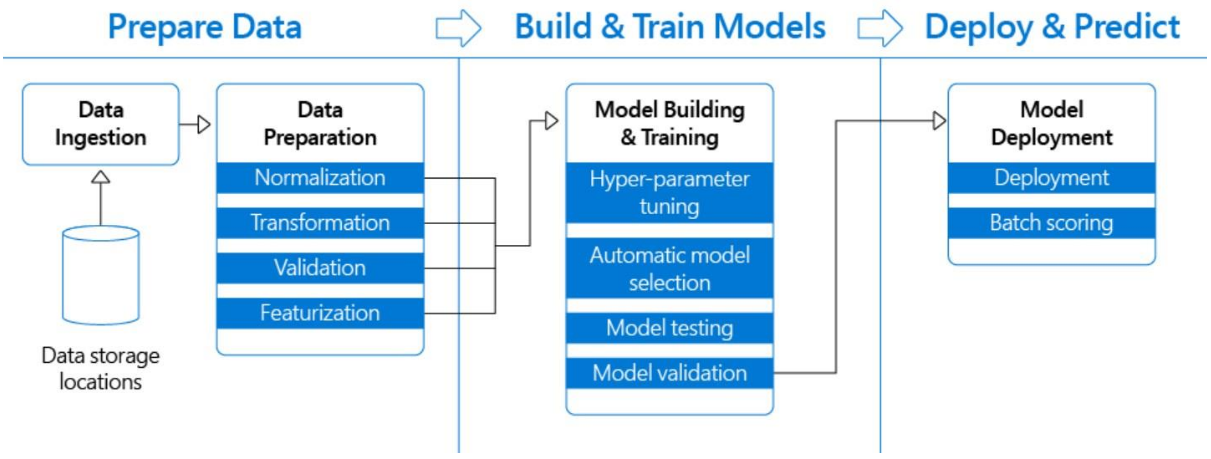
Process Flow

Proposed methodology

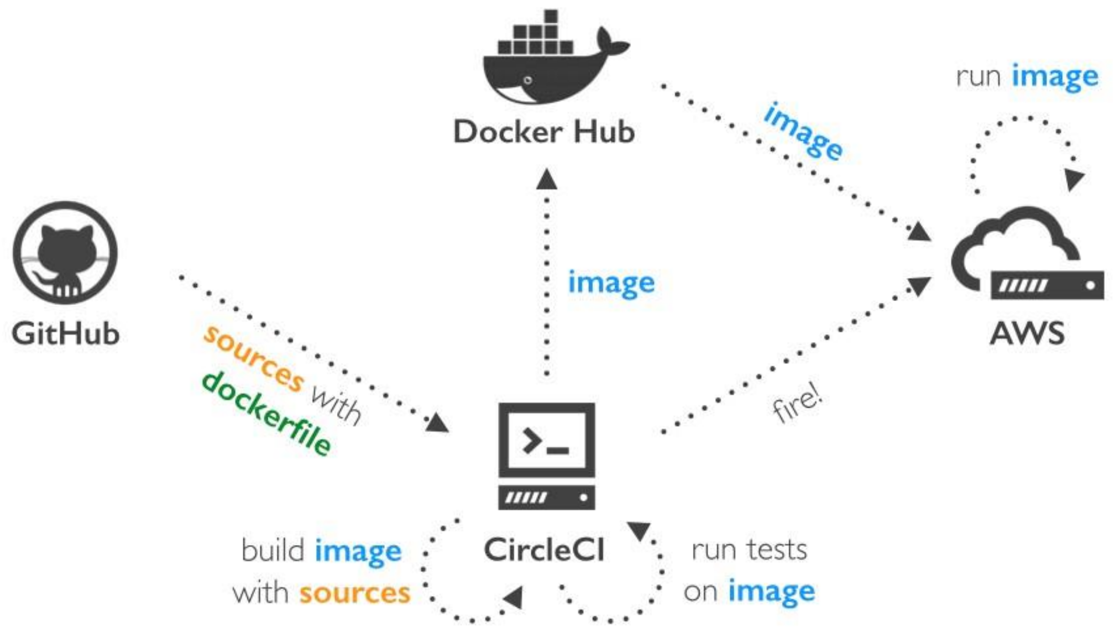


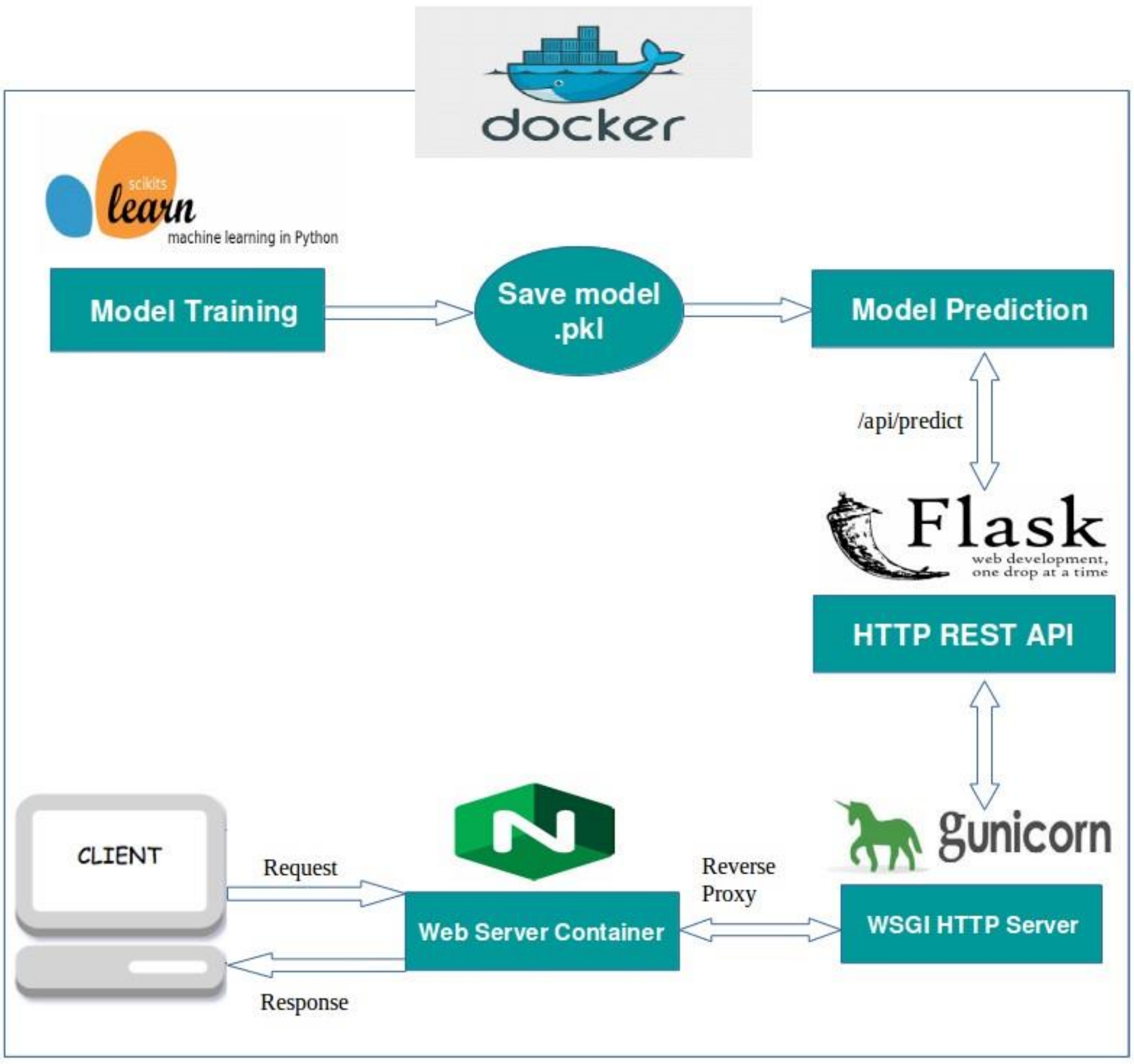
Model training and evaluation





Deployment
process





1. **Data Collection and Storage:** Used Mongodb for collecting and storing raw data. In the Training Pipeline Data Ingestion artifact is used for storing train and test data in the feature store folder which was used for further process.

2. **Data Validation**: During this process, we identify if the train data and test data are similar or if there are some changes that happened at the time of the train test split. The changes could be feature changes, data structure, data size, or there may be any kind of data drift.

3. **Data Transformation:** In the process of data transformation, various feature selection and extraction techniques are performed along with the normalization of data.

4. **Model Development and Training:** During Model training, first the appropriate model algorithm is selected for its training, and then hyperparameter tuning is performed on it to make it more effective.

5. **Model Evaluation:** At the time of model evaluation, the base model is compared to the current model with the help of a performance matrix to get the efficacy and reliability of the new model.

6. **Model pusher:** model pusher is used for saving artifact files and models used during training so, that they further can be used for monitoring and retraining.

7. **Prediction pipeline:** In the prediction pipeline process, we write code so that our code could be used for prediction purposes.

8. **Model Deployment and Serving:** In this process, we make an application and deploy it using the ci/cd pipeline with the help of AWS services.

Application compatibility and reusability

1. **Version Control:**

    - Used (Git) which is a version control system to track changes in the ML code, data preprocessing scripts, and model configuration files.

2. **Continuous Integration (CI):**
    - Set up CI to automatically build and test the ML code whenever changes are pushed to the version control repository.
    - Execute unit tests and data validation checks to ensure that the code and data are consistent and error-free.

3. **Model Training:**
    - Automated model training using predefined hyperparameters and configurations.

4. **Model Evaluation:**
    - Automatically evaluate model performance using validation datasets and predefined evaluation metrics.
    - Compare model performance against baseline models to ensure continuous improvement.

5. **Model Deployment:**
    - Containerize the trained ML model using Docker enabling consistency and portability across different environments.
    - Set up automated deployment to staging and production environments.

6. **Monitoring and Logging:**
    - Log important events and errors to facilitate troubleshooting and debugging.

7. **Model Versioning:**

   - Version models to keep track of different iterations and improvements.
   - Store model artifacts and saved models making it easy to reproduce specific versions.

8. **Continuous Deployment (CD):**

   - Automated the deployment of new model versions to production based on predefined criteria, such as model performance and stability.

Event log

The system logs every event so that the user will know what process is running internally. Proper formatting is done with date and time and the logging level is set to info.

Error Handling

When errors are encountered, an explanation will be displayed as to what went wrong. An error will be defined as anything that falls outside the normal and intended usage.

4.Conclusion

In conclusion, the credit risk prediction system is a significant and essential project for commercial banks, providing a solution to accurately assess the creditworthiness of credit card owners. By leveraging machine learning techniques and following a well-defined High-Level Design, the system aims to predict the probability of credit default based on credit card owner's characteristics and payment history.