LOW - LEVEL DESIGN(LLD)

Credit Card Default Prediction

PRATHMESH JAGTAP
iNeuron

Revision History

Version	Date	Author	Comments
0.1	31.11.2022	PRATHMESH JAGTAP	Draft Version

Contents

1.0 Introduction

1.1 What is Low – Level Design Document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Credit Card Default Prediction. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document. Low-level design is a detailed description of every module of software. It describes every module in detail by incorporating the logic behind every component in the system. It delves deep into every specification of every system, providing a micro-level design.

1.2 Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

2.0 Technical Specifications

2.1 Input Schema for Classification Problems

Feature Name	Datatype	Size	Null/Required	Description
Project Name	String	--	Required	Name/Title of the project
File Regex	String	--	Required	Regex for filename
Train Schema	JSON	--	Required	Schema Containing Columns names
Test Schema	JSON	--	Required	Schema Containing Columns names
Problem Type	String		Required	Classification
Target Column Name	String		Required	

2.2 Performing Model Training

- Preprocessing pipeline is applied based on the algorithm, and problem type.
- Using a bagging, boosting, stacking and blending approach to select the best model.

2.3 Logging

- We should be able to log every activity done by the user.
- The System identifies at what step logging is required.
- The System should be able to log each and every system flow.
- Developers can choose logging methods. You can choose database logging/File logging as well.
- System should not be hung even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

2.4 Deployment

- Heroku

3.0 Technology stack

Front End	HTML/CSS/JS
Backend	Python, Flask, Pandas
Database	--
Deployment	Heroku

4.0 Proposed Solution

A brief description of various stages in the machine learning pipeline is describes below:

1) **Extracting Data**
   Here the platform loads data form user input (though input predict form) into a dataframe format.

   Future Update; Platform will be able to load data form any cloud storage like AWS, GCP, Azure or warehouse such as BigQuery, Snowflake, Redshift, Oracle or ERP with various file extension such as .h5 or zip files.

2) **Validating Data**
   Data validation is done on the data extracted from the sources, in which data types of the column, encoding of the categorical columns, and feature engineering are performed.

3) **Transforming Data**
   Here the platforms transform data into numerical to categorical, and in standardized form using a python package for further processing.

4) **Model Evaluation**
   Here the evaluation of model is based on the given threshold which gives the sign of acceptance of the model.

5) **Model Training**
   Platform provides the ability to the end user to train models using our library and pipeline on the transformed and validated data based on request. The model training will add the best trained model to the system.
   Here the best trained model is returned after completion of complete machine learning pipeline.

6) **Performing Predictions**
   Platform provides the ability to the end user to make predictions models on the transformed and preprocessed data based on request by loading the best trained model form the storage.

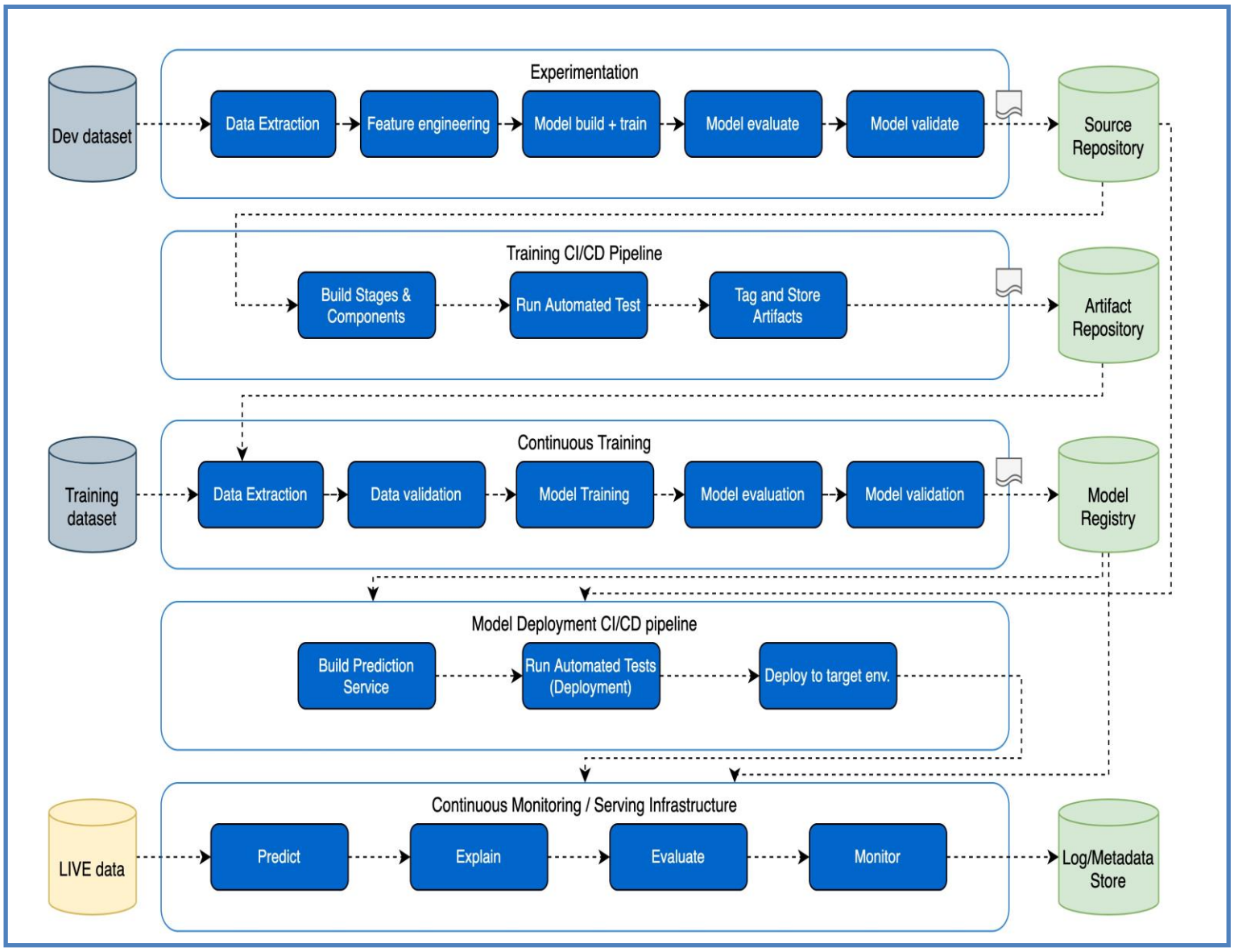
7) Dashboard

Users can see metrics of the models trained using the report management.

8) Logging

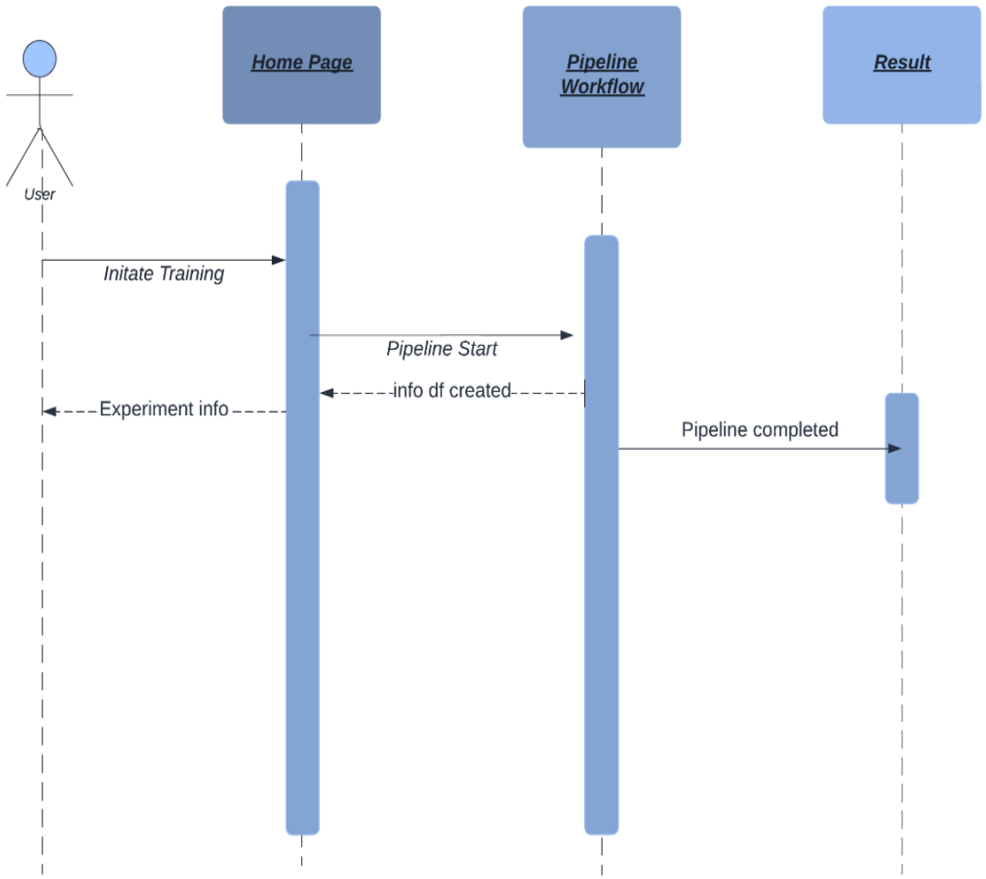
Users can check logs of the various internal processing steps of the project using log in application

5.0 Model Training / Validation Workflow



6.0 Low Level System Design

6.1 Sequence Diagram



7.0 Unit Testing

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. Application URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	1. Application URL is accessible 2. Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether user is able to see input fields on entering in.	1. Application is accessible.	User should be able to see input fields on entering in
Verify whether user is able to edit all input fields	1. Application is Accessible.	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	1. Application is Accessible.	User should get Submit button to submit the inputs
Verify whether user is presented with results on clicking submit	1. Application is Accessible.	User should be presented with recommended results on clicking submit
Verify whether the results are in accordance to the selections user made	1. Application is Accessible.	The recommended results should be in accordance to the selections user made