Credit Card Default Prediction Model

Low Level Design (LLD)

Mehul Rampratap Nayak

Revision Number 3.0
Revised Date 25/10/2022

Document Version Control

Date Issued	Version	Description
23/10/2022	1.0	Initial LLD
24/10/2022	2.0	Flowchart
25/10/2022	3.0	Final

Contents

1. Introduction

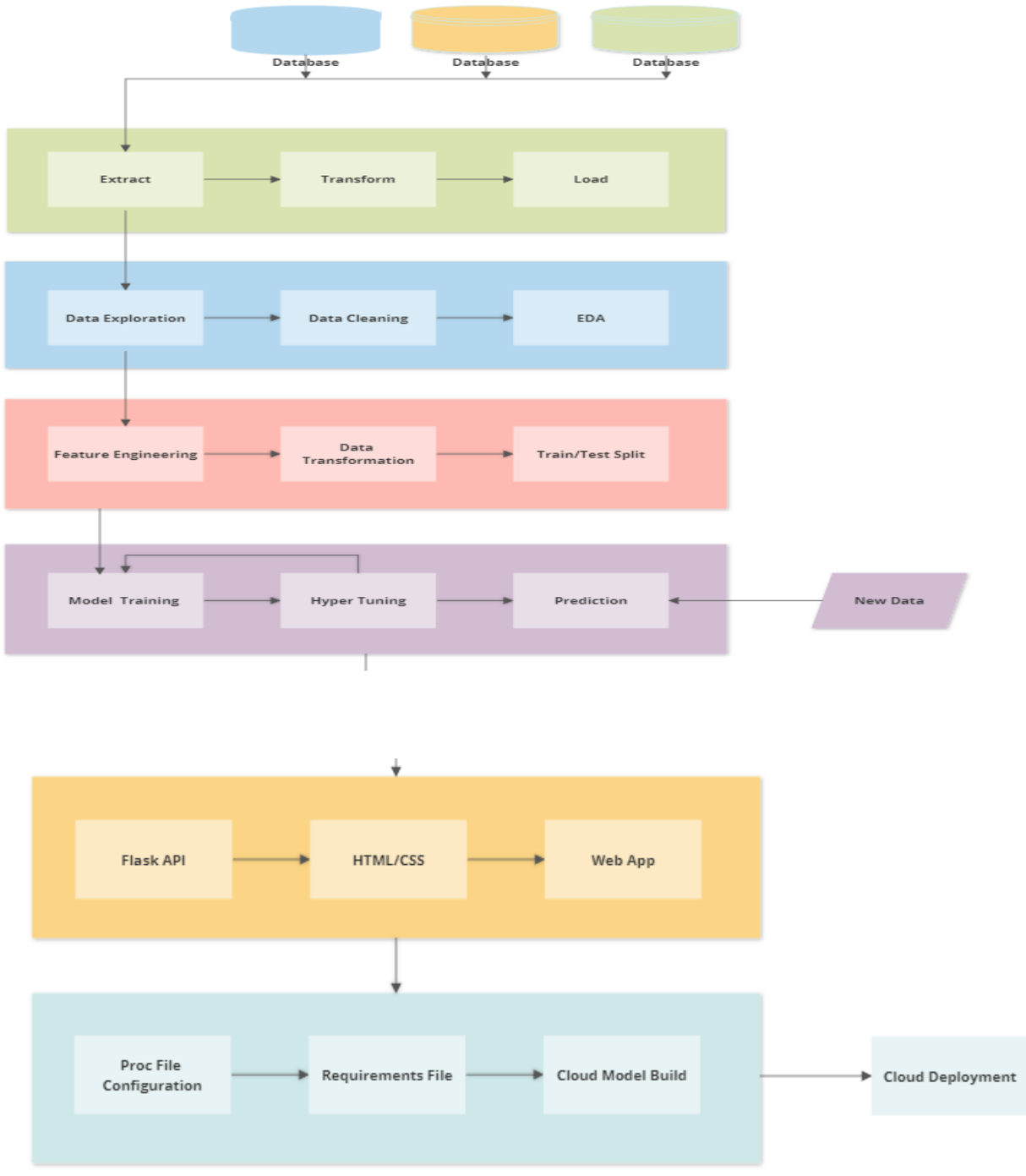
1.1 Why Low-Level Design?

Low-Level design is a component-level design process with detailed description and logic for every module. It is also called micro level design designed by administrators and developers converting high level solution into detailed solution.

1.2 Scope

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code. Low-level design is created based on the high level design. LLD describes the class diagrams with the methods and relations between classes and program specs.

2.1 Architecture



2.2 Extract, Transform and Load

Here the data is retrieved from multiple databases ir entities into a pool and then transformation is applied to extract the required data in useable format. Before this process the data will be a clutter and will not be viable to for data analysis.

Later the transformed data is sent to different entities based on the request.

2.3 Data Exploration

Data received will be in tabular format but it may contain data in unstructured manner like string data type might be used for columns having numbers. So our first step is to explore all rows and columns for an eagle eye overview.

2.4 Data Cleaning

After being explored we should have an idea about the quality of the data to step into data cleaning process. This step is very crucial for the model performance as it may affect the model prediction if wrong data is used to train. Many columns may have missing values or garbage values which can be handled by replacing it with mean or median in-case of numerical data or we can drop the corresponding rows after thorough examination making sure that it wont affect the prediction.

2.5 Data Transformation

Data should be normally distributed for maximum model performance, so we transform the data using various techniques like log transformation, polynomial transformation and then train the model for premium performance.

2.6 EDA (Exploratory Data Analysis)

It is an approach of analysing the data to summarize the main characteristics using statistical graphics and data visualization methods. It shows what the data can tell us beyond the formal modelling and thereby contrasts traditional hypothesis.

2.6 Feature Engineering

It is a technique that leverages data to create new variables that aren't in the training set. It can produce new features for both supervised and unsupervised learning with the goal of simplifying and speeding up data transformation while enhancing model accuracy

2.7 Train/Test Split

We now split the dataset into two set, one for training with 80% data and the remaining 20% data is used for testing the model for accuracy. You can split the dataset in any ratio with the majority for training and minority for testing respectively. We use the random state value to get the same training and test dataset for training all models repeatedly after hyper tuning.

2.8 Model Building

Performance of the model not only depends on the training data but also with choice of algorithm. Depending on the problem statement the algorithm is chosen and trained with tuning to yield best results. In our case we chose to build the model using multiple classification algorithms like Logistic Regression, Decision Tree, SVM Classifier, Random Forest and Naive Bayes. We train all models with same training dataset and predict the accuracy with same test data.

2.9 Hyper Tuning

Tuning is performed to improve predicting accuracy based on problem statement. Some problem concentrates more false negative, in that case we tune our model for better recall accuracy. Incase of true positive we tune for better precision accuracy. We compare all models and perform tuning for best to improve recall accuracy because our model is concerned more on False Negative.

2.10 Prediction

After the training and tuning we feed test data to predict the result. We compare the predicted result with actual result to calculate the accuracy. If the model performance is not up-to the mark we will fine-tune the parameters till we achieve the best accuracy.

2.11 Flask API

We need a front-end interface to receive input from the user. We receive account number, age, credit limit, outstanding balance as integer values and others in categorical options. All categorical options are converted into numerical values in flask API and fed into model. Also it make use of HTML and CSS for designing the web interface.

2.12 HTML/CSS

Front-end app needs presentation to receive input in fancier way. We make use of HTML/CSS to design the template and use it for flask app to execute the APIs in web interface. CSS style-sheets are bootstrapped from online for better visualization.

2.13 Proc file

This file is necessary for the whole model to deploy into Heroku cloud. It has the configuration how the application has to run in the cloud environment. In our case we specify to run it as web application. Once the model is build the application will run as Web App.

2.14 Requirements file

This file helps to build the cloud container with necessary packages to run the application. Without this we cannot deploy as it may not know the environment for the app to run.

2.15 Cloud deployment

Once we provide proc and requirement file the cloud container starts to build the environment, installing necessary packages and runs the application as Web App.

3.0 Unit Test Case

Test Case Description	Expected Result
Verify whether Application URL is accessible to the user	URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	The Application should load completely for the user when the URL is accessed
Verify whether user is able to see input fields	User should be able to see input fields
Verify whether user gets Submit button to submit the inputs	User should get Submit button to submit the inputs
Verify whether user is presented with recommended results on clicking submit	User should be presented with prediction results on clicking submit
Verify whether the recommended results are in accordance to the selections user made	The recommended results should be in accordance to the selections user made