Metro Interstate Traffic Volume Prediction

Low Level Design

Domain: Machine Learning

Creator: Jaiprakash

Date: 10.07.2022

Document Version Control

Date issued	Version	Description	Author
July 12, 2022	1.1	First Draft	Muhammad Ojagzada
July 14, 2022	1.2	Added unit test cases	Muhammad Ojagzada

Contents

Introduction

What is Low-Level Design Document?

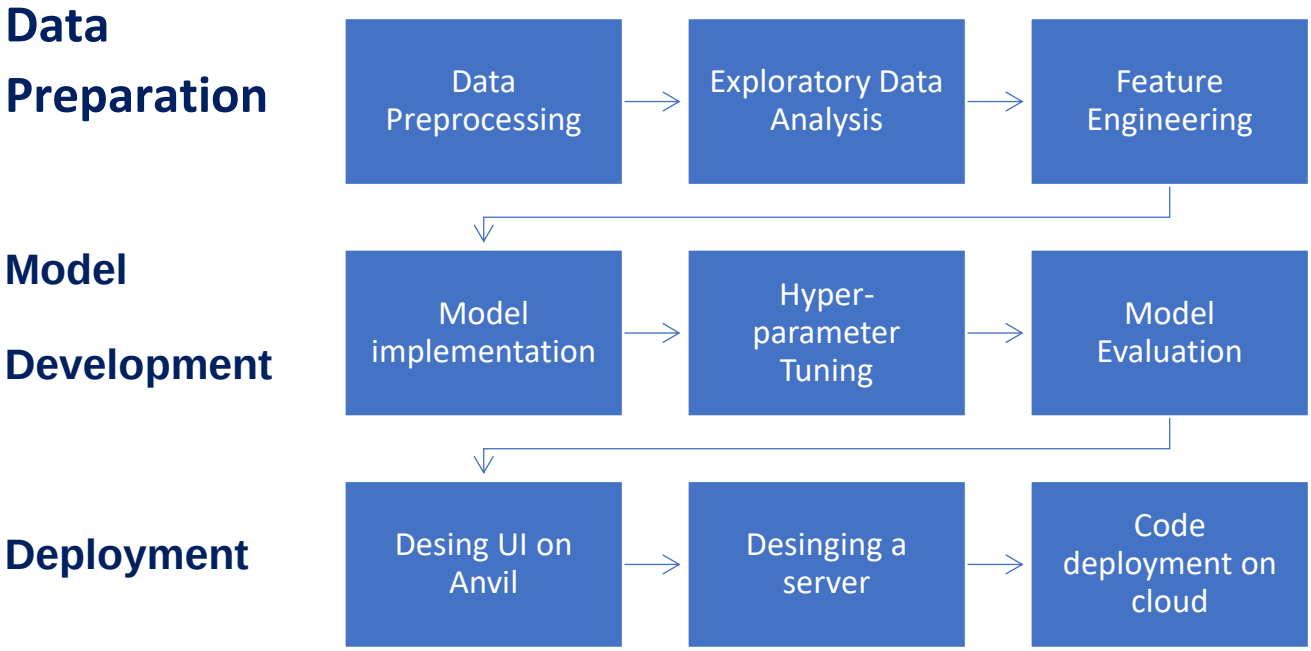
The goal of LLD or a low-level design document is to give the internal logical of the actual program code for Metro Interstate Traffic Volume Prediction. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli.

The main objective of the project is to predict if traffic volume is in high or low on particular date. Weather circumstance, special days like holidays, daytime (morning, afternoon, night and etc.), a temperature, a weekday, a numeric percentage of cloud cover are vital attributes for predicting traffic volume.

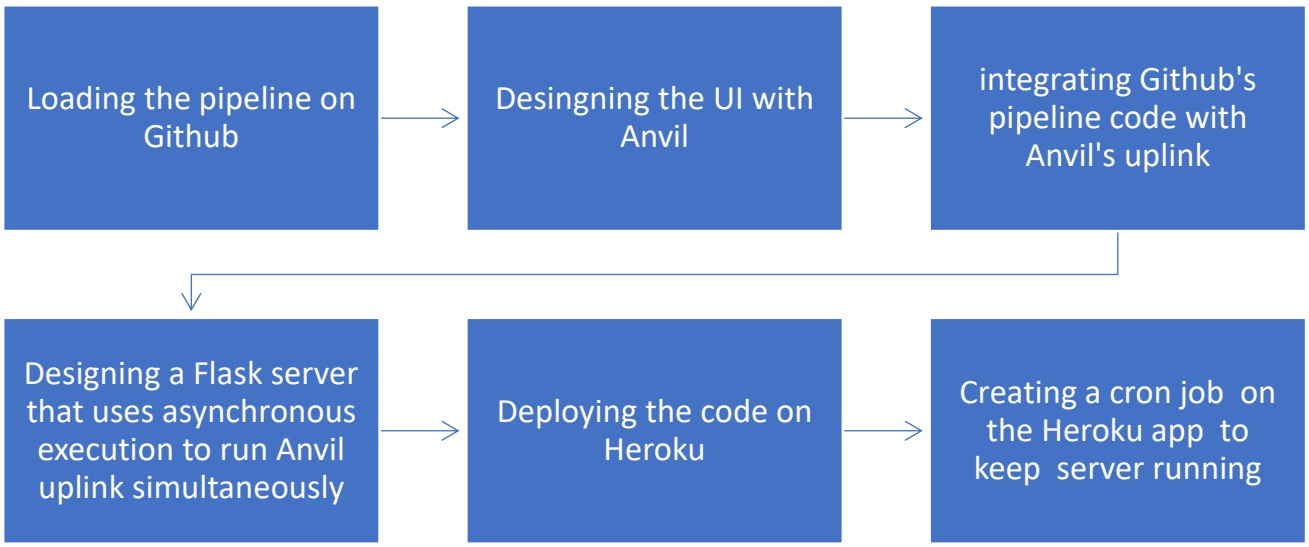
Scope

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.

Architecture



Deployment



Architecture Description

Data Preparation

Data Description

Hourly Interstate 94 Westbound traffic volume for MN DoT ATR station 301, roughly midway between Minneapolis and St Paul, MN. Hourly weather features and holidays included for impacts on traffic volume. The goal of this project is to build a prediction model using multiple machine learning techniques and to use a template to document the end-to-end stages. We're trying to forecast the value of a continuous variable with the Metro Interstate Traffic Volume dataset, which is a regression issue.

Data Preprocessing

In data preprocessing step, we check if there missing data, duplicate values, and datatypes of each feature. In our dataset, there was not any null and duplicate values

Exploratory Data Analysis

This step includes bivariate and univariate analysis of features. Checking outliers using boxplots, and outlier treatment is carried out as well. Distribution of numerical values is plotted to see to what extent our data is skewed.

Feature Engineering

In this part, datatypes of “date_time”, “rain_1h”, “snow_1h”, “temp” were corrected. Values is “temp” column are in Kelvin, they are converted to Celsius for convenience. Outliers were checked using boxplot and removed from the data. Lastly, some new columns (weekday, hour, month, year) were extracted from “date_time” column. Newly derived “hour” column is modified like “early morning”, “morning”, and etc. As “weather_main” column contains major data, “weather_description” feature was dropped. Moreover, most of values in “snow_1h” and, ”rain_1h” columns are almost 0, so they are also dropped from the data.

Model Development

Model implementation

After train and test splitting, pipeline containing Standard Scaler and Ordinal Encoder was fitted to several models such as AdaBoost Regressor, Gradient Boosting Regressor, RandomForest Regressor, CatBoost Regressor, XGB Regressor. Their R2 score were obtained. and it was determined that CatBoost performs better than other models.

Hyper-parameter Tuning

The best model is chosen, and Grid Search with Cross Validation is applied on that model to get the best parameters. Those parameters are then used on the model to get better result.

Model Evaluation

Test dataset is used to evaluate the model. 20% of dataset was separated for testing. Predicted results of the model are compared with the actual data to check the amount of error. As there was no considerable change after hyperparameter tuning, it helped us to overcome overfitting and perform better on new data.

Deployment

Designing UI with Anvil

For this project, a user interface is built on Anvil. It is a web application that helps us to create applications for projects. It is a free Python-based drag-and-drop web app builder.

Designing a server

A server should be created to run the UI application continuously. Flask server is built, and it is linked with Anvil uplink that connects Anvil UI with our server.

Code deployment on cloud

The codes for this machine learning model should be deployed to the cloud, so that when data is entered into the application, our code runs, and a user gets the result online.

Deployment Process

In this stage, we establish a server using Flask that runs the uplink code (server code) in parallel before developing the UI using Anvil and connecting with our code, where our model is executing, via an uplink. We will post the hole after execution or asynchronous execution. Git and GitHub are used to code in the Heroku cloud. Then, we'll configure a cron job to maintain the server and server code in operation indefinitely.

Unit cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	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.	Application is accessible	User should be able to see input fields
Verify whether user is able to edit all input fields	Application is accessible	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	Application is accessible	User should get Submit button to submit the inputs
Verify whether user is presented with results on clicking submit	Application is accessible	User should be presented with results on clicking submit
Verify whether the results are in accordance to the selections user made	Application is accessible	The results should be in accordance to the selections user made