Low Level Design

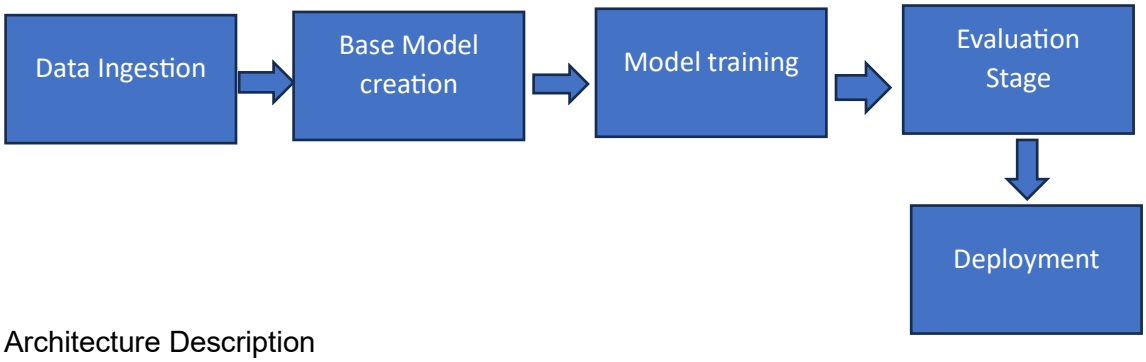
Inter-state traffic Volume Prediction

Revision Number: 1.0
Last date of Revision: 06/01/2024

Document Version Control

Date Issued	Version	Description	Author
06/01/2024	1	Initial LLD- V1.0	Ajith Kumar V HariHara Sudan R

1. 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 Food Recommendation System. 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.
   1.2. Scope
   Low-level design (LLD) is a component-level design process that follows a step-bystep 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. Architecture



3. Architecture Description
   3.1. Data Ingestion

   Data Description
   Given is the variable name, variable type, the measurement unit and a brief description. The order of the listing corresponds to the order of numerals along the rows of the database.

Name	Data Type	Measurement
clouds_all	int	Percentage of cloud cover
day	int	dayof the data collected
holiday	int	US National holidays plus regional holiday
hour	int	Hour of the data collected in local CST time
month	int	Month of the particular data collected
rain_1h	float	Amount in mm of rain that occurred in the hour

snow_1h	float	Amount in mm of snow that occurred in the hour
temp	float	Average temp in kelvin
traffic_volume	int	Hourly I-94 ATR 301 reported westbound traffic volume
weather_description	int	Longer textual description of the current weather
weather_main	int	Short textual description of the current weather
year	int	Year of the particular data collected

Data Gathering
Data Source: https://archive.ics.uci.edu/dataset/492/metro+interstate+traffic+volume
Train and Test data are split in ratio of 80:20 using random seed during training and validating.

Data Transformation
Before sending data into database, data transformation is required so that data are converted into such form which can be easily interpreted by the model. Here, weather description, weather main were in textual form so they are modified into numeric form using one hot encoding and then datetime were in string format so they are splitted into year,month,date and hour so that it can be interpreted easily by the model.

Database insertion
Dataset is inserted into cloud database and here Cassandra DB is used which was specified as a requirement

Export as 'CSV' from database
From database we are making a CQL query using necessary drivers and tokens and the data is stored into CSV file. Now this CSV file is used for further processing.

Feature Selection
Necessary features like holiday data,time, rain fall amount, snow fall amount and other weather parameters are selected for processing. Inclusion of these key features aims to improve the accuracy and reliability of our predictive models ensuring a comprehensive consideration of factors influencing traffic volume.

3.2. Base model creation

After doing all kinds of operations mentioned above our focus turns to the pivotal stage of model building.Following a thorough exploration and refinement of feature selection and parameter tuning we have opted for LSTM model. This choice is based on the fact that LSTM is good in time series prediction. The constructed model comprises of two layers of LSTM which can accept a feature of (11,1) and two dense layers enhancing models capacity for capturing relationships within the data.

3.3. Model Training

Parameter Tuning
Parameters like learning rate and epoch are chosen so that the loss converges. This ensures that our predictive models achieve optimal performance striking a balance between computational efficiency and the precision required for accurate traffic volume predictions.

Model Training
In the process of training our chosen LSTM model, we discovered that running it for 200 cycles (epochs) with groups of 500 data points (batch size) resulted in the validation loss settling down. This means the model became good at making predictions, providing us with reliable forecasts for traffic volume.

Model Saving
Model is saved using .h5 format using tf library

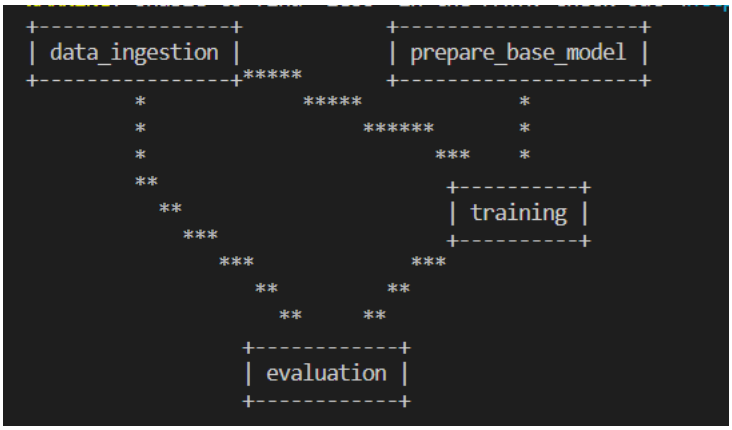
3.4. Evaluation stage

Model will be evaluated with the validation set and then it will proceed to deployment.

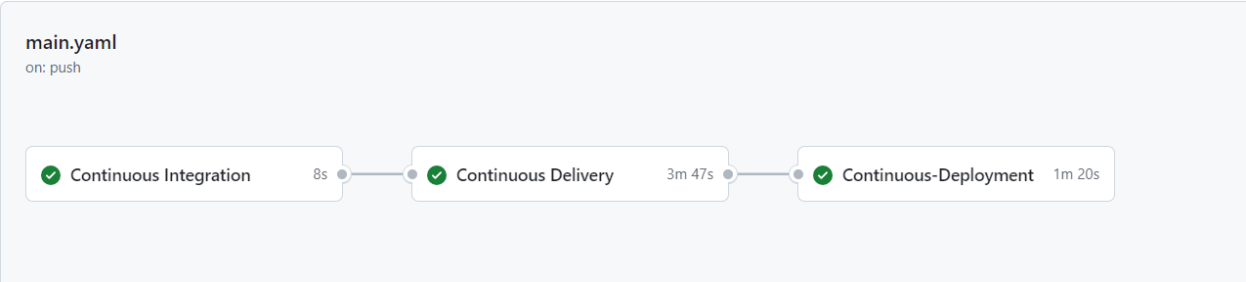
3.5. Deployment

Flask Setup for user interaction
Setting up Flask for user interaction involves a few steps. First, after saving the model, we initiate the process of building an API using Flask. This creates a web application. Now, whenever a user enters data, it gets checked to make sure it's valid, and then it's sent to the model for predicting traffic volume. This way, users can easily interact with and get predictions from the model through the web application.

DVC Check
To make sure we're not doing the same thing over and over, we've included a Data Version Check (DVC check). This helps us avoid unnecessary repetition of a specific stage in our process. If we've already done something once, the check ensures we don't do it again, saving time and resources.



Github
The whole project is pushed into github repository and CI/CD pipeline is being used with AWS as host.



Deployment
We arranged things so our project could run on the internet using Amazon Web Services (AWS). Specifically, we used AWS ECR to store our software and AWS EC2 to run it. This setup allows our system to be accessible and functional in the cloud.

4. Unit Test Cases

Test Case Description	Prerequisite	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 URLis accessed	1.Application URLis accessible 2.Application is deployed	The Application should load completely for the user when theURL is accessed
Verify whether a user is able to see inputfields while opening the application	1. Application is accessible 2.The user is able to see the input fields	Users should be able to see inputfields on logging in
Verify whether a user is able to enter the input values.	1.Application is accessible 2. The user is able to see the input fields	The user should be able to fill the input field
Verify whether a user gets predict button to submit the inputs	1.Application is accessible 2.The user is able to see the input fields	Users should get Submit button to submit the inputs
Verify whether a user is presented with recommended results on clicking submit	1. Application is accessible 2. The user is able to see the input fields. 3. The user is able to see the submit button	Users should be presented with recommended results on clicking submit

Verify whether a result is in accordance with the input that the user has entered	1.Application is accessible 2.The user is able to see the input fields. 3.The user is able to see the submit button	The result should be in accordance with the input that the user has entered