

**Metro Interstate Traffic Volume Prediction**

Project Architecture

Domain: Machine Learning

Creator: Anurag Mukati

Dhar, India

+91 9425335205

[anuragmukati09@gmail.com](mailto:anuragmukati09@gmail.com)

**Architecture**

**Data Preparation**

**Model Development**

**Deployment**

**Deployment Process**

**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 but datatype of “data\_time” column was “object”; thus, it was converted to “datetime64”.

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

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. Lastly, unnecessary columns were dropped.

**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. The highest score is acquired from the RandomForest Model.

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 is 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.

**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.