Team 6: Peter Muller, Ariana Rocha, Goutam Mukku, Katie Burgess, Ayush Tripathi

API Documentation for Deployed Traffic Prediction Model

Overview

This document provides details on how to interact with the RESTful API for the Holistic Traffic Prediction for Smart Cities model. The model, deployed using Kubernetes and Docker, serves predictions on real-time traffic data using the METR-LA dataset. The API allows users to send requests and receive traffic flow predictions. This documentation will cover the available endpoints, input/output formats, and example requests and responses.

Base URL

The API is hosted at:

```
Unset
http://localhost:5000
```

Endpoints

Predict Traffic Flow

Endpoint:

```
Unset
 /predict
```

Method:

```
Unset
GET
```

Description:
This endpoint generates traffic flow predictions using pre-trained LSTM and GRU models for both MAE and RMSE metrics. The predictions are based on pre-loaded test data sequences and are saved into a CSV file (predictions.csv).

Example Request & Response:

Example Request	Expected Response
`curl http://localhost:5000/predict`	Status code: `200 OK` Response Body: Unset `Predictions generated and saved successfully!` *File with prediction is saved to predictions.csv*

Process:

1. Load test data: Loads input sequences for testing from input_sequences_test.npy
2. Two sets of models are loaded:
    a. LSTM and GRU models optimized for MAE
    b. LSTM and GRU models optimized for RMSE
3. Prediction Generation: Predictions are made for both MAE and RMSE using the respective models.
4. Saving Predictions: The predictions are saved in a CSV file (predictions.csv) with columns LSTM_MAE, GRU_MAE, LSTM_RMSE, and GRU_RMSE.

Model Monitoring Report

Endpoint:

```
Unset
 /report
```

Method:

```
Unset
GET
```

Description:
This endpoint generates a data drift report using the Evidently library. The report compares the predictions from the model (stored in predictions.csv) with the ground truth data (stored in ground_truth.csv) and checks for data drift. If significant drift is detected, a warning is logged. The report is saved as an HTML file (report.html), and the endpoint renders this file for viewing.

Example Request & Response:

Example Request	**Expected Response**
`curl http://localhost:5000/report`	Status code: `200 OK` Response Body: *The HTML report is rendered in the browser.*

Process:

1. Load Data: The endpoint loads the predictions.csv (generated from the model) and ground_truth.csv files.
2. Create Report: The Evidently Report class is used with the DataDriftPreset to compare the two datasets.
3. Check for Data Drift:
    a. The endpoint calculates a dataset_drift score.
    b. If the drift exceeds a defined threshold (0.5), a warning is logged, indicating significant data drift.
4. Save and Render Report: The generated report is saved as report.html and displayed to the user.