**Tools and Frameworks Used in Your Model**

**1. Programming Language**

* **Python**: The entire notebook is written in Python, a versatile programming language widely used for data science, machine learning, and time series analysis.

**2. Data Manipulation and Processing**

* **Pandas**:
  + Used for data manipulation and analysis.
  + In your notebook, Pandas is used to create and manipulate DataFrames (e.g., prepare\_traffic\_df converts raw data into a DataFrame with columns like timestamp, total\_count, day\_of\_week, etc.).
  + Example: df = pd.DataFrame(data\_list), df['timestamp'] = pd.to\_datetime(df['ts'], unit='ms').
* **NumPy**:
  + Used for numerical computations and array operations.
  + In your model, NumPy arrays are used to prepare time series data for the LSTM model (e.g., X and y in prepare\_time\_series\_data) and to perform calculations like inverse scaling.
  + Example: X = np.array(X), y = np.array(y).
* **JSON** (Standard Library):
  + Used to load raw data from JSON files.
  + Example: with open(file\_path, 'r') as f: data = json.load(f) in read\_json\_file.

**3. Machine Learning and Deep Learning**

* **TensorFlow/Keras**:
  + **TensorFlow**: An open-source machine learning framework developed by Google, used as the backend for Keras.
  + **Keras**: A high-level API for building and training deep learning models, now integrated into TensorFlow (as tf.keras).
  + In your notebook, Keras is used to build and train the LSTM model for time series prediction.
  + Example: model = Sequential(), model.add(LSTM(units=units, return\_sequences=True)), model.fit(X\_train, y\_train, epochs=epochs).
* **LSTM (Long Short-Term Memory)**:
  + A type of recurrent neural network (RNN) architecture designed for time series data, capable of learning long-term dependencies.
  + In your model, LSTM layers are used to predict future traffic counts based on past data.
  + Example: model.add(LSTM(units=units, return\_sequences=True)).

**4. Hyperparameter Optimization**

* **Optuna**:
  + An open-source hyperparameter optimization framework.
  + In your notebook, Optuna is used to tune the LSTM model’s hyperparameters (e.g., number of units, learning rate, batch size) to minimize the validation loss.
  + Example: study = optuna.create\_study(direction='minimize'), study.optimize(objective, n\_trials=20).

**5. Data Preprocessing**

* **Scikit-learn**:
  + **MinMaxScaler**: Used to scale the data (e.g., total\_count, day\_of\_week, hour) to a range of [0, 1] before feeding it into the LSTM model.
    - Example: scaler = MinMaxScaler(), scaled\_data = scaler.fit\_transform(df[['total\_count', 'day\_of\_week', 'hour']]).
  + **mean\_absolute\_error**, **mean\_squared\_error**: Used to compute evaluation metrics (MAE, MSE, RMSE) for model performance.
    - Example: mae = mean\_absolute\_error(y\_true\_inv, y\_pred\_inv), mse = mean\_squared\_error(all\_y\_true, all\_y\_pred).

**6. Visualization**

* **Plotly**:
  + An interactive plotting library used for creating visualizations.
  + In your notebook, Plotly Express (px) and Plotly Graph Objects (go) are used to create graphs:
    - Section #2: A line graph showing actual vs predicted traffic for all aliases.
      * Example: fig11 = go.Figure(), fig11.add\_trace(go.Scatter(...)).
    - Section #3: A bar chart showing combined MSE and RMSE for the overall dataset.
      * Example: fig12 = px.bar(perf\_df, x='Metric', y='Value', ...).

**7. Time and Date Handling**

* **Datetime** (Standard Library):
  + Used to handle timestamps and date ranges.
  + Example: start\_date = datetime(2024, 12, 1).date(), df['timestamp'] = pd.to\_datetime(df['ts'], unit='ms').

**8. File System Operations**

* **os** (Standard Library):
  + Used to interact with the file system, such as listing files in a directory.
  + Example: os.listdir(data\_dir) in the data loading loop.
* **glob** (Standard Library):
  + Used to find files matching a pattern (e.g., all .json files in a directory).
  + Example: glob.glob(os.path.join(data\_dir, '\*.json')).