Traffic Anomaly Detector + Self-Improving Model — Project Report

Step 1: Dataset Creation

We created a custom CSV dataset consisting of **1000 synthetic traffic records**. The dataset includes the following features:

- speed: Average speed of vehicles (in km/h)
- vehicle_count: Number of vehicles observed
- time: Time of the day (Morning, Afternoon, Evening, Night)
- day: Day of the week
- location: Location ID or name
- anomaly: Binary class label (1 = Anomaly, 0 = Normal)

This data mimics real-world traffic conditions, including both normal flow and anomalies such as congestion or unexpected slowdowns.

Step 2: Data Preprocessing & EDA (Exploratory Data Analysis)

- Used **Pandas** and **Seaborn** for data analysis and visualization.
- Verified class balance and feature distributions using count plots and heatmaps.
- Applied **Label Encoding** on categorical features such as time, day, and location.
- Data was split into training and testing sets using an 80-20 ratio.

Step 3: Model Training (Initial Model)

- Trained a **RandomForestClassifier** to detect traffic anomalies.
- The model was fitted on the preprocessed training data.
- Initial test accuracy was **1.0 (100%)**, indicating perfect separation of normal and anomalous data.

Step 4: Real-Time Prediction Simulation

- Simulated real-time inputs with values like speed, vehicle count, and time.
- The trained model made predictions on this live data.
- Two types of outputs were observed:
    - "**Anomaly Detected!"**
    - **"Normal Traffic"**

Step 5: Self-Improving Model (Dynamic Retraining)

- Appended additional real-time data (both normal and anomalies) into the dataset.
- Re-encoded and retrained the model with the new data.
- After retraining, the updated model achieved an **accuracy of 0.9933 (99.33%)**.
- This step demonstrates the model's ability to **self-learn and adapt** to evolving traffic patterns.

Step 6: Performance Visualization

- Plotted a **learning curve** to visualize model performance over varying training sizes.
- Observed no overfitting, indicating good generalization.
- Accuracy drift was tracked before and after retraining to measure improvement over time.

Future Scope

1. **Integration with Real-Time Traffic APIs**
   Connecting the model with real-world traffic data sources (e.g., Google Maps, Open Traffic APIs) can enhance the system's ability to detect and respond to live traffic anomalies effectively.
2. **Deployment on Edge Devices or IoT Systems**
   Implementing the model on edge devices such as smart cameras or Raspberry Pi units enables real-time, on-site anomaly detection, making it suitable for smart city infrastructure.
3. **Multi-Class Anomaly Detection**
   Expanding the current binary classification to detect specific types of traffic anomalies—such as accidents, roadblocks, or unusual congestion—can provide more detailed and actionable insights.

Conclusion

- Successfully built a machine learning-based **Traffic Anomaly Detection System** with adaptive learning capabilities.
- The model:
  - Detects anomalies in real-time traffic data.
  - Adapts over time using additional data (self-improvement).
  - Maintains high accuracy and reliability.
- This project can serve as a foundation for advanced real-world systems in **smart cities**, **traffic management**, and **road safety monitoring**.