TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning

# 1. Introduction

TrafficTelligence is a machine learning-based project designed to estimate traffic volume using features like temperature, rain, snow, cloudiness, and weather conditions. It includes a user-friendly web interface built with Flask and also demonstrates integration with IBM Watson for cloud-based model scoring.

# 2. Objectives

- Develop a machine learning model to estimate traffic volume.  
- Deploy the model using a Flask web application.  
- Allow user input via a web form for real-time prediction.  
- Integrate model scoring with IBM Watson services.  
- Provide proper documentation and a video demo for reference.

# 3. Tools & Technologies Used

- Programming Language: Python  
- Libraries: Pandas, Scikit-learn, NumPy, Flask  
- Deployment: Flask Web Framework  
- Cloud Integration: IBM Watson Machine Learning  
- IDE: Jupyter Notebook, VS Code

# 4. Dataset

The dataset used is 'traffic volume.csv' which includes various features such as date-time, temperature, weather conditions, and the target variable 'traffic\_volume'. It was preprocessed to handle null values, encode categorical variables, and scale numerical data.

# 5. Project Directory Structure

TrafficTelligence/  
├── Flask/  
│ ├── templates/  
│ ├── app.py  
│ ├── encoder.pkl  
│ └── model.pkl  
├── IBM/  
│ └── traffic\_volume\_lbm\_scoring\_end\_point.ipynb  
├── Requirements.txt  
├── Traffic volume estimation.docx  
├── traffic volume.csv  
└── traffic volume.ipynb

# 6. Model Development

The model was developed using Random Forest Regressor. The training involved splitting the dataset, fitting the model, and saving the trained model as 'model.pkl'. Label encoding was applied for categorical features like weather conditions, and the encoder was saved as 'encoder.pkl'.

# 7. Flask Application

The Flask application allows users to input parameters via an HTML form and view predicted traffic volume on the same page. The backend (`app.py`) loads the trained model and encoder to process user inputs and generate predictions.

# 8. IBM Watson Integration

A Jupyter Notebook demonstrates how to deploy the trained model on IBM Watson Machine Learning and make prediction requests using their API. This enables cloud-based scoring using the deployed model endpoint.

# 9. How to Run the Project

1. Install dependencies:  
 pip install -r Requirements.txt  
  
2. Run the Flask app:  
 cd Flask  
 python app.py  
  
3. Open your browser and go to http://127.0.0.1:5000/  
  
4. Enter values in the form to get traffic predictions.

# 10. Video Demonstration

A demonstration video is recorded to showcase how the application works, including web input, predictions, and IBM Watson integration. The video can be uploaded to YouTube or Google Drive for sharing.

# 11. Conclusion

This project demonstrates how machine learning models can be applied to real-world data for predictive analytics. By integrating Flask and IBM Watson, it becomes both accessible and scalable for broader use cases.