

TrafficTelligence:Advanced Traffic Volume Estimation with Machine Learning

(KRISHANMURTI)

(RAJANI MUDIMADUGU)

(SHASHI KUMAR)

(VIKASH KUMAR UPADHAYA)

Team ID : 738291

Guided by

Prof.Ms Rasha

A Dissertation Submitted to SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY, An Autonomous Institution affiliated to 'JNTU Ananthapur' in Partial Fulfilment of the Requirements for the Bachelor of Technology (Computer Engineering) with Specialization in Artificial Intelligence and Machine Learning.

MAY 2024



***SRI VENKATESHWARA COLLEGE OF ENGINEERING AND
TECHNOLOGY***

R.V.S Nagar Tirupati road, Andhra Pradesh-517127

Compliance certificate

This is to certify that the research work embodied in this dissertation entitled “TrafficTelligence:Advanced Traffic Volume Estimation with Machine Learning” was carried out by (Krishanmurti),(Rajanimudimadugu),(Shashi kumar),(Vikash kumar upadhaya) at Sri Venkateswara College of Engineering and Technology for partial fulfilment of Bachelor of Technology (Computer Engineering) with Specialization in Artificial Intelligence and Machine Learning degree to be awarded by JNTU Anantapur. We have complied to the comments given by the Dissertation phase – We as well as Mid Semester Dissertation Reviewer to our satisfaction.

Date : 14/05/2024

Place : Chittoor

(Prof .Ms Rasha)

Head ,(CSE(AI&ML))(M Dr Lavanya)

Principal (Dr M Mohan Babu)



SRI VENKATESHWARA COLLEGE OF ENGINEERING AND TECHNOLOGY

R.V.S Nagar Tirupati road, Andhra Pradesh-517127

Declaration of originality

We hereby certify that We were the sole author of this dissertation and that neither any part of this dissertation nor the whole of the dissertation has been submitted for a degree to any other University or Institution.

We certify that, to the best of my knowledge, my dissertation does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations or any other material from the work of other people included in our dissertation, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that we have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Indian Copyright Act, We certify that we have obtained a written permission from the copyright owner(s) to include such material(s) in our dissertation and have included copies of such copyright clearances to our appendix.

We declare that this is a true copy of dissertation, including any final revisions, as approved by our dissertation review committee.

Date: 05/05/24

Place: Chittoor

(Krishanmurti Kumar)

(Rajani Mudimadugu)

(Shashi Kumar)

(Vikash Kumar Upadhaya)



SRI VENKATESHWARA COLLEGE OF ENGINEERING AND TECHNOLOGY

R.V.S Nagar Tirupati Road, Andhra Pradesh-517127

Abstract

The objective of this project is to address the critical need for advanced traffic volume estimation using machine learning techniques. With increasing urbanization and the consequent rise in vehicular traffic, accurate prediction of traffic volume is essential for optimizing traffic volume. It is essential for optimising traffic flow, planning infrastructure, and

enhancing overall urban mobility. Traditional methods of traffic volume estimation often rely on manual data collection or simplistic models, leading to limited accuracy and scalability. To tackle this challenge, Traffic Intelligence seeks to leverage advanced machine learning algorithms to develop a robust and scalable solution for traffic volume estimation. By harnessing data from various sources such as traffic cameras, sensors, GPS data, and historical traffic patterns, the project aims to create predictive models capable of accurately forecasting traffic volume in real-time.

Acknowledgment

This thesis is based on research work conducted for “Stock Price Prediction Using Machine

Learning”. This work would not be possible without many people whose contributions can’t be ignored.

We would like to pay our special regards to Sri Venkateswara College of Engineering and Technology for providing required resources for this work. We wish to express my sincere appreciation to our mentor Prof. Miss Rasha, whose assistance was a milestone in completion of this project and Prof. Sowmya intelligently solved my queries. We owe our gratitude to the HOD OF CSE(AI&ML) Dr. M Lavanya who helped in understanding different machine learning techniques. Without their support, a significant portion of project work was not possible in constrained time.

We must also appreciate our family and friends for helping us survive all the stress throughout the year. To our parents, for supporting me both on and off the water. We would like to thank you all mentioned and other people who have helped us directly or indirectly for pushing farther than we thought we could go.

LIST OF FIGURES

TOPICS

1.overview of Traffic Volume Estimation:

- 1.1.Importance of Traffic Data in Urban Planning
- 1.2.Traditional Traffic Volume estimation methods and their limitations
- 1.3.Role of Machine Learning
- 1.4.applications
- 1.5.objectives
- 1.6.Brief overview of Machine Learning can transform traffic estimation

2.TrafficTelligence:Advanced Traffic Volume Estimation With Machine Learning

- 2.1.project flow
- 2.2.prior knowledge
- 2.3.project objectives
- 2.4.project structure

3.data collection&preparation

- 3.1.collect the dataset
- 3.2.importing the necessary libraries
- 3.3.Analyse The Data
- 3.4.Handling The Missing Values
- 3.5.Data Visualization
- 3.6.Splitting The Dataset Into Dependent And Independent Variable
- 3.7.Feature Scaling
- 3.8.Splitting The Data into Train and Test

4.Model Building

4.1.Training And Testing The Model

4.2.Model Evaluation

4.3.Save The Model

5.application building

5.1.Build HTML CODE

5.2.Main Python Script

5.3.Run The APP

5.4.Output

1.OVERVIEW OF TRAFFIC VOLUME ESTIMATION:

Traffic volume estimation is a critical component in the management and optimization of transportation systems. Accurate estimation of traffic volumes aids in urban planning, infrastructure development, traffic control, and congestion management. This overview will cover the importance, methods, challenges, and applications of traffic volume estimation.

Importance of Traffic Volume Estimation

1. **Urban Planning and Development:** Accurate traffic volume data is essential for urban planners to design and develop infrastructure that meets the demands of growing urban populations. It helps in making informed decisions about road expansions, public transportation systems, and traffic management solutions.
2. **Traffic Management:** Estimating traffic volumes enables traffic authorities to implement effective traffic control measures, such as signal timings, lane usage, and congestion pricing. It helps in reducing congestion, improving safety, and enhancing the overall efficiency of the transportation network.
3. **Environmental Impact:** Understanding traffic patterns and volumes helps in assessing the environmental impact of vehicular emissions. This information is vital

for developing strategies to reduce pollution and promoting sustainable transportation options.

4. **Economic Benefits:** Efficient traffic management and infrastructure development based on accurate traffic volume estimation can lead to significant economic benefits. Reduced travel times, lower fuel consumption, and decreased vehicle wear and tear contribute to cost savings for both individuals and businesses.

1.1 Importance of traffic data in urban planning:

Traffic data plays a pivotal role in urban planning, providing essential insights that guide the development and management of transportation infrastructure. It helps urban planners, policymakers, and engineers make informed decisions to create efficient, sustainable, and safe urban environments. Here, we will discuss the various aspects of how traffic data impacts urban planning.

1. Infrastructure Development

1.1 Road Network Design: Traffic data helps in designing road networks that can handle current and future traffic volumes. It informs the placement of new roads, expansions, and the need for bypasses and alternative routes to alleviate congestion.

1.2 Public Transportation Systems: Understanding traffic patterns allows planners to design effective public transportation routes. This includes optimizing bus and train schedules, planning for new transit lines, and ensuring connectivity between different modes of transportation.

1.3 Intersection and Interchange Design: Traffic data aids in the design of intersections and interchanges to minimize congestion and maximize safety. It helps determine the appropriate number of lanes, signal timings, and the need for traffic control devices such as roundabouts and traffic lights.

2. Traffic Management

2.1 Congestion Mitigation: By analyzing traffic data, urban planners can identify congestion hotspots and develop strategies to mitigate them. This includes implementing traffic signal synchronization, congestion pricing, and dynamic lane usage.

2.2 Real-time Traffic Monitoring: Continuous traffic data collection enables real-time monitoring and management of traffic flow. This helps in responding promptly to traffic incidents, managing special event traffic, and optimizing traffic signal timings.

2.3 Intelligent Transportation Systems (ITS): Traffic data is a cornerstone of ITS, which uses technology to improve transportation efficiency and safety. ITS applications include

adaptive traffic control systems, incident detection and management, and providing real-time traffic information to the public..

1.2.Traditional traffic volume estimation methods and their limitation:

Traffic volume estimation is crucial for urban planning, traffic management, and infrastructure development. Traditional methods have long been used to collect traffic data, but they come with various limitations. Here, we will discuss the common traditional methods of traffic volume estimation and their associated limitations.

Traditional Traffic Volume Estimation Methods

1. **Manual Counts**
2. **Pneumatic Road Tubes**
3. **Inductive Loop Sensors**
4. **Infrared Sensors**
5. **Magnetic Sensors**

1. Manual Counts

Description: Manual counts involve human observers standing at a specific location and counting the number of vehicles passing through that point. These counts can be done for different types of vehicles and can be categorized by time of day.

Limitations:

- **Labor-Intensive:** Requires significant human resources, especially for long durations or multiple locations.
- **Time-Consuming:** Counting and categorizing vehicles manually takes a considerable amount of time.
- **Prone to Human Error:** Fatigue, distractions, and inconsistencies among counters can lead to inaccuracies.
- **Limited Coverage:** Manual counts are typically feasible only for specific points and short durations, providing limited data coverage.

2. Pneumatic Road Tubes

Description: Pneumatic road tubes are placed across the road surface. When a vehicle passes over the tube, air pressure changes, which is recorded by a counter device. These counts can provide data on vehicle volume, speed, and classification.

Limitations:

- **Installation and Maintenance:** Requires physical installation and regular maintenance to ensure accuracy.
- **Interference:** Debris, weather conditions, and vandalism can affect the accuracy of the data.
- **Limited Durability:** Tubes can wear out quickly under heavy traffic or extreme weather conditions.
- **Data Gaps:** They can only measure traffic at the specific locations where they are installed.

3. Inductive Loop Sensors

Description: Inductive loop sensors are embedded in the road surface and detect vehicles by measuring changes in inductance as a vehicle passes over the loop. They are often used at intersections to control traffic signals.

Limitations:

- **Costly Installation:** Installation involves significant costs and disruption to traffic, as the road surface must be cut to embed the loops.
- **Maintenance Challenges:** Repairing and maintaining these sensors can be difficult and costly, requiring road closures.
- **Sensitivity Issues:** May not detect motorcycles or bicycles accurately, leading to incomplete data.
- **Fixed Locations:** They provide data only at the specific points where they are installed, limiting spatial coverage.

4. Infrared Sensors

Description: Infrared sensors use infrared light to detect vehicles as they pass through the sensor's field of view. These sensors can be mounted on poles or overpasses and can count vehicles, measure speed, and classify vehicle types.

Limitations:

- **Weather Sensitivity:** Performance can be affected by adverse weather conditions such as fog, rain, and snow.
- **Line-of-Sight Requirement:** Requires a clear line of sight to the roadway, which can be obstructed by tall vehicles or other obstacles.
- **Installation Costs:** Installing and calibrating these sensors can be expensive and complex.
- **Limited Range:** Effective only within a certain range and field of view, necessitating multiple installations for broader coverage.

5. Magnetic Sensors

Description: Magnetic sensors detect the presence of vehicles by measuring changes in the Earth's magnetic field caused by the metal in the vehicle. These sensors can be installed on or in the road surface.

Limitations:

- **Accuracy Variability:** The accuracy can be affected by the speed and size of the vehicles, as well as the sensor's sensitivity.
- **Installation and Maintenance:** Like inductive loops, they require embedding in the road, leading to installation and maintenance challenges.
- **Environmental Factors:** Nearby magnetic fields or electronic devices can cause interference, affecting the accuracy of the sensors.
- **Fixed Data Points:** They provide data only at their installed locations, limiting overall coverage.

Summary of Limitations

Traditional traffic volume estimation methods, while foundational, have several limitations:

- **High Labor and Maintenance Costs:** Manual counts and sensor installations require significant human and financial resources.
- **Data Accuracy Issues:** Human error, environmental factors, and sensor limitations can affect the accuracy of the collected data.
- **Limited Spatial and Temporal Coverage:** Most traditional methods provide data for specific locations and times, which may not be representative of overall traffic patterns.
- **Installation Disruption:** Methods requiring road installations (inductive loops, magnetic sensors) cause traffic disruptions and incur high costs.

1.3 Role of Machine Learning:

Machine learning (ML) plays a transformative role in traffic intelligence, offering advanced methods for estimating traffic volume that overcome the limitations of traditional approaches. By leveraging large datasets, real-time analytics, and sophisticated algorithms, machine learning enhances the accuracy, efficiency, and predictive capabilities of traffic volume estimation. Here, we explore the various roles and benefits of machine learning in traffic intelligence.

1. Enhanced Data Collection and Integration

1.1 Real-time Data Collection: Machine learning models can process data from a variety of real-time sources, such as GPS devices, mobile phone signals, traffic cameras, and sensors embedded in roadways. This continuous data stream provides a comprehensive view of traffic conditions.

1.2 Data Integration: Machine learning algorithms can integrate data from multiple sources, including historical traffic data, weather conditions, roadworks, and special events. This holistic approach leads to more accurate and context-aware traffic volume estimations.

2. Improved Accuracy and Efficiency

2.1 Automated Analysis: Machine learning models can automatically analyze vast amounts of traffic data, identifying patterns and anomalies that would be difficult for humans to detect. This leads to more precise traffic volume estimates.

2.2 High-resolution Predictions: Unlike traditional methods that may provide point-specific or periodic data, machine learning can offer high-resolution, continuous predictions of traffic volume across entire networks.

3. Predictive Capabilities

3.1 Short-term Forecasting: Machine learning models can predict traffic volumes in the short term, helping in real-time traffic management. For instance, they can forecast congestion build-ups and suggest proactive measures to mitigate traffic jams.

3.2 Long-term Planning: By analyzing historical data and identifying long-term trends, machine learning can assist in urban planning and infrastructure development. Planners can use these predictions to design roads, public transport systems, and traffic control measures that accommodate future traffic volumes.

4. Dynamic Traffic Management

4.1 Adaptive Traffic Control: Machine learning enables adaptive traffic signal control systems that adjust signal timings based on real-time traffic conditions. This reduces wait times, optimizes traffic flow, and minimizes congestion.

4.2 Incident Detection and Management: ML algorithms can quickly detect traffic incidents such as accidents or road blockages from traffic camera feeds or sensor data. Rapid detection allows for quicker response and clearance, reducing the impact on traffic flow.

5. Personalized and Contextual Traffic Information

5.1 Personalized Routing: Machine learning can provide drivers with personalized route recommendations based on current traffic conditions and individual driving habits. This reduces travel time and improves overall traffic distribution.

5.2 Context-aware Alerts: Drivers can receive real-time alerts about traffic conditions, road closures, and alternative routes, tailored to their specific location and destination.

6. Advanced Analytical Tools

6.1 Anomaly Detection: Machine learning models can identify unusual traffic patterns, such as those caused by sudden weather changes, accidents, or events. This capability helps in timely intervention and management.

6.2 Traffic Pattern Analysis: By continuously learning from traffic data, ML models can reveal underlying traffic patterns and trends. This insight helps in better understanding traffic behavior and planning accordingly.

Case Studies and Applications

1. Smart City Initiatives: Cities like Singapore and Los Angeles use machine learning to manage traffic in smart city initiatives. These systems analyze traffic data in real-time to optimize traffic light timings, predict congestion, and provide travelers with up-to-date traffic information.

2. Ride-sharing Services: Companies like Uber and Lyft employ machine learning to predict demand and optimize driver routes, reducing wait times and improving service efficiency.

3. Autonomous Vehicles: Autonomous vehicles rely heavily on machine learning for navigation and traffic management. These vehicles use ML algorithms to interpret sensor data, predict traffic movements, and make real-time driving decisions.

Challenges and Future Directions

1. Data Privacy and Security: The collection and use of traffic data must be managed carefully to protect individual privacy and ensure data security.

2. Model Robustness and Reliability: Machine learning models must be robust and reliable, capable of handling diverse traffic scenarios and unexpected events.

3. Infrastructure Investment: Implementing ML-driven traffic intelligence systems requires significant investment in infrastructure, including sensors, communication networks, and computational resources.

4. Continuous Learning and Adaptation: Traffic patterns evolve over time, so ML models need continuous updates and training to maintain accuracy and relevance.

1.4 APPLICATIONS:

1. Real-Time Traffic Monitoring and Management

1.1 Adaptive Traffic Signal Control

- **Dynamic Signal Timing:** Machine learning algorithms analyze real-time traffic data to adjust traffic signal timings dynamically. This optimization reduces wait times at intersections, leading to smoother traffic flow.
- **Incident Detection:** Real-time monitoring systems detect incidents such as accidents or roadblocks. Immediate alerts to traffic management centers enable quick response, minimizing disruptions.

1.2 Intelligent Transportation Systems (ITS)

- **Smart Traffic Lights:** Integration of IoT sensors with traffic lights to monitor traffic conditions and adjust signals in real time.
- **Connected Vehicles:** Vehicles equipped with communication systems share traffic data with central systems, enhancing overall traffic management.

2. Predictive Traffic Analysis

2.1 Traffic Congestion Prediction

- **Short-term Forecasting:** Machine learning models predict traffic congestion minutes or hours ahead, allowing for proactive measures to manage traffic flow.
- **Long-term Traffic Trends:** Analysis of historical data helps in understanding long-term trends and patterns, aiding infrastructure planning.

2.2 Demand Prediction for Public Transport

- **Ridership Forecasting:** Estimating future public transport demand helps optimize routes and schedules, improving efficiency and reducing congestion.

3. Smart City Integration

3.1 Urban Mobility Management

- **Integrated Transport Systems:** Combining data from various transport modes (buses, trains, bicycles) to create a unified, efficient urban mobility system.
- **Smart Parking Solutions:** Real-time data guides drivers to available parking spots, reducing search time and traffic congestion.

3.2 Environmental Monitoring

- **Air Quality Management:** Traffic data helps monitor and predict air quality, leading to measures that reduce pollution.
- **Noise Pollution Control:** Traffic flow data aids in designing noise mitigation strategies.

4. Enhanced User Experience

4.1 Personalized Navigation

- **Real-time Route Suggestions:** Personalized routing based on current traffic conditions and user preferences.
- **Multimodal Navigation:** Integration of different transport modes for efficient travel planning.

4.2 Travel Time Estimation

- **Accurate Arrival Predictions:** Machine learning models provide precise travel time predictions, aiding in better journey planning.

5. Traffic Safety Enhancements

5.1 Accident Prevention

- **Hazard Detection:** Real-time data analytics detect potential road hazards, alerting drivers and reducing accident risks.
- **Driver Assistance Systems:** Advanced systems provide real-time guidance to drivers, improving safety.

5.2 Pedestrian and Cyclist Safety

- **Smart Crosswalks:** Sensors and signals ensure safe passage for pedestrians and cyclists.
- **Protected Bike Lanes:** Data-driven design of safe bike lanes.

6. Urban Planning and Development

6.1 Infrastructure Planning

- **Data-driven Decisions:** Traffic data informs urban planners about current and future needs, guiding infrastructure development.
- **Simulation and Modeling:** Traffic simulation models evaluate the impact of new infrastructure on traffic flow.

6.2 Policy and Regulation

- **Congestion Pricing:** Traffic data supports the implementation of congestion pricing, reducing peak-time traffic volumes.
- **Traffic Calming Measures:** Data analysis identifies areas needing speed bumps or reduced speed limits to enhance safety.

7. Commercial Applications

7.1 Logistics and Fleet Management

- **Route Optimization:** Optimized delivery routes reduce travel time and fuel consumption.
- **Predictive Maintenance:** Traffic data combined with vehicle telemetry helps predict maintenance needs.

7.2 Ride-sharing and Mobility Services

- **Demand Forecasting:** Predicting demand ensures availability of ride-sharing services where needed.
- **Dynamic Pricing:** Adjusting fares based on current traffic conditions and demand levels.

1.5 Brief Overview of Machine Learning can Transform the Traffic Estimation:

Introduction

Traffic estimation involves predicting the flow of vehicles on road networks, which is critical for urban planning, traffic management, and improving transportation systems. Traditional methods rely on manual counts, historical data, and simplistic models that often fail to capture the complexity of modern traffic systems. Machine learning, with its ability to handle vast amounts of data and uncover intricate patterns, offers a transformative approach to traffic estimation.

Advantages of Machine Learning in Traffic Estimation

1. Enhanced Accuracy

- **Complex Pattern Recognition:** Machine learning algorithms can identify and learn from complex traffic patterns, improving the accuracy of traffic predictions.
- **Real-time Data Processing:** These models can process real-time traffic data from various sources such as sensors, cameras, and GPS devices, providing up-to-the-minute traffic estimates.

2. Scalability

- **Handling Large Datasets:** Machine learning techniques are well-suited to handle and analyze large datasets, which is crucial for modern traffic systems with data from numerous sources.
- **Adapting to Changes:** These models can be retrained and updated with new data, allowing them to adapt to changing traffic conditions and trends.

3. Predictive Capabilities

- **Short-term and Long-term Forecasting:** Machine learning models can be designed to predict traffic conditions in both the short term (minutes to hours) and the long term (days to weeks).
- **Incident Prediction:** Advanced models can also predict traffic incidents or congestion before they occur, enabling proactive management measures.

Key Machine Learning Techniques in Traffic Estimation

1. Supervised Learning

- **Regression Models:** Linear regression, decision trees, and neural networks can predict traffic volume based on historical data.
- **Classification Models:** These models classify traffic conditions (e.g., free-flowing, congested) based on input features.

2. Unsupervised Learning

- **Clustering:** Techniques like k-means clustering can group similar traffic patterns, which helps in understanding different traffic behaviors and conditions.
- **Anomaly Detection:** Identifying unusual traffic patterns that may indicate accidents or other disruptions.

3. Reinforcement Learning

- **Adaptive Traffic Control:** Reinforcement learning models can optimize traffic signal timings dynamically based on current traffic conditions, improving traffic flow and reducing congestion.

Applications of Machine Learning in Traffic Estimation

1. Real-time Traffic Monitoring

- **Live Traffic Updates:** Machine learning models provide real-time traffic updates and predict future traffic conditions, helping drivers choose optimal routes.
- **Incident Detection and Management:** Real-time analysis helps in quickly detecting and managing traffic incidents.

2. Smart Traffic Systems

- **Adaptive Traffic Signals:** Traffic lights adjust in real time based on current traffic flow, reducing wait times and improving overall traffic movement.
- **Integrated Urban Mobility:** Machine learning helps in coordinating different modes of transport, enhancing urban mobility.

3. Infrastructure Planning

- **Data-driven Planning:** Analysis of traffic data helps urban planners design and implement infrastructure projects more effectively.
- **Predictive Maintenance:** Predicting when and where road maintenance is needed based on traffic patterns and wear-and-tear data.

4. Environmental Impact

- **Emission Reduction:** Optimizing traffic flow reduces idle times and stop-and-go traffic, leading to lower vehicle emissions.
- **Noise Pollution Management:** Understanding traffic patterns helps in designing measures to control noise pollution in urban areas.

Challenges and Considerations

1. Data Quality and Privacy

- Ensuring the accuracy and reliability of traffic data is crucial for effective machine learning models. Additionally, protecting the privacy of individuals whose data is being collected is essential.

2. Model Interpretability

- Machine learning models, especially complex ones like deep neural networks, can be difficult to interpret. Ensuring that traffic management authorities can understand and trust the models' predictions is important.

3. Integration with Existing Systems

- Integrating machine learning models with existing traffic management systems can be challenging. It requires careful planning and coordination.

2. Traffic Intelligence: Advanced Traffic Volume Estimation With Machine Learning:

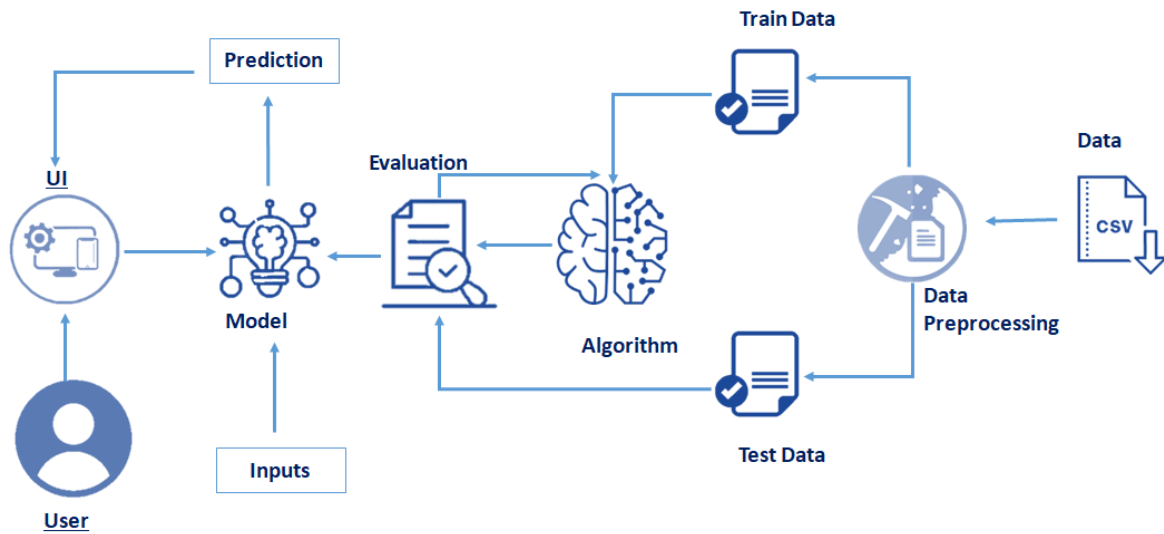
Introduction:

Growth in the number of vehicles and degree of urbanization means that the annual cost of traffic jams is increasing in cities. This leads to a decrease in the quality of life among citizens through a considerable waste of time and excessive fuel consumption and air pollution in congested areas

Traffic congestion has been one of the major issues that most metropolises are facing despite measures being taken to mitigate and reduce it. The safe and time-efficient movement of the people and goods is dependent on Traffic flow, which is directly connected to the traffic characteristics. Early analysis of congestion events and prediction of traffic volumes is a crucial step to identify traffic bottlenecks, which can be utilized to assist traffic management centres.

We will be using Regression algorithms such as Linear Regression, Decision tree, Random forest, and xgboost to predict the count of traffic volume. We will train and test the data with these algorithms. From this best model is selected and saved in .pkl (Pickle) format. Once the model is saved, we integrate it with flask application and also deploy the model in IBM.

Technical Architecture:



2.1 project flow:

- User interacts with the UI (User Interface) to enter the input values.
- Entered input values are analyzed by the model which is integrated.
- Once the model analyses the input the prediction is showcased on the UI.

To accomplish this, we have to complete all the activities and tasks listed below

- Data Collection.
 - Collect the dataset or Create the dataset
- Data Pre-processing.
 - Import the Libraries.
 - Importing the dataset.
 - Checking for Null Values.
 - Data Visualization.
 - Taking care of Missing Data.
 - Feature Scaling.
 - Splitting Data into Train and Test.
- Model Building
 - Import the model building Libraries
 - Initializing the model
 - Training and testing the model
 - Evaluation of Model

- Save the Model
- Application Building
 - Create an HTML file
 - Build a Python Code
 - Run the App

2.2 Prior Knowledge:

To complete this project, you must require the following software's, concepts, and packages

Anaconda navigator:

Refer to the link below to download anaconda navigator

Link:

□

<https://www.youtube.com/watch?v=5mDYijMfSzs>

Python packages:

Open anaconda prompt as administrator.

- Type “pip install numpy” and click enter.
- Type “pip install pandas” and click enter.
- Type “pip install matplotlib” and click enter.
- Type “pip install scikit-learn” and click enter.
- Type “pip install Flask” and click enter.
- Type “pip install xgboost” and click enter.

Machine Learning concepts:

- Supervised learning

<https://youtu.be/QeKshry8pWQ?si=iL0ws8eEVDsoSNSo>

- Unsupervised learning

<https://youtu.be/D6gtZrsYi6c>

- Metrics

<https://youtu.be/aWAnNHXIKww>

- Flask Basics

https://youtu.be/lj4l_CvBnt0

2.3 Project Objectives:

By the end of this project:

- You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
- You will be able to know how to pre-process/clean the data using different data pre-processing techniques.
- You will be able to analyze or get insights into data through visualization.
- Applying different algorithms according to a dataset and based on visualization.
- You will be able to know how to find the accuracy of the model.
- You will be able to know how to build a web application using the Flask framework.

2.4 Project Structure: Create a Project folder that contains files as shown below

| Name | Type | Date Modified |
|--|-------------|------------------|
| > .ipynb_checkpoints | File Folder | 11-12-2021 13:07 |
| ✓ Flask | File Folder | 10-02-2022 11:35 |
| > templates | File Folder | 10-02-2022 11:35 |
| app.py | py File | 10-02-2022 11:35 |
| 01 encoder.pkl | pkl File | 11-12-2021 13:01 |
| 01 model.pkl | pkl File | 11-12-2021 13:01 |
| ✓ IBM | File Folder | 15-02-2022 14:58 |
| > Flask | File Folder | 31-01-2022 10:30 |
| traffic volume_ibm_scoring end point.ipynb | ipynb File | 31-01-2022 10:27 |
| Requirements.txt | txt File | 15-12-2021 10:57 |
| Traffic volume estimation.docx | docx File | 15-02-2022 15:04 |
| traffic volume.csv | csv File | 10-12-2021 11:08 |
| traffic volume.ipynb | ipynb File | 11-12-2021 13:03 |

- Flask files consist of template folder which has HTML pages, app.py file and .pkl files which are used for application building
- IBM folder has flask files and scoring endpoint.ipynb- model training code file.
- We need the model which is saved and the saved model in this content is Traffic volume. Pkl
- Templates folder which contains index.HTML file, chance.HTML file, noChance.HTML file.
- Scale.pkl for scaling, encoder.pkl file for encoding the categorical data, imputer.pkl file for filling out the missing values

3.Data Collection And Preparation:

ML depends heavily on data, without data, it is impossible for an “AI” model to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training **data set**. It is the actual **data set** used to train the model for performing various actions.

You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository etc.

3.1 Collect The Dataset:

Please refer to the [link](#) given below to download the data set and to know about the dataset.

https://drive.google.com/file/d/1iV5PfYAmI6YP0_0S4KYy1ZahHOqMgDbM/view

3.2 Importing The Necessary Libraries:

It is important to import all the necessary libraries such as pandas, NumPy, matplotlib.

- **Numpy**- It is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform

mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.

- **Pandas**- It is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.
- **Seaborn**- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Matplotlib**- Visualisation with python. It is a comprehensive library for creating static, animated, and interactive visualizations in Python
- **Sklearn** – which contains all the modules required for model building.

```
# importing the necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import sklearn as sk
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost
```

3.3 Import The dataset:

- You might have your data in .csv files, .excel files
- Let's load a .csv data file into pandas using read_csv() function. We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program).
- If your dataset is in some other location, Then
- **Data=pd.read_csv(r"File_location/datasetname.csv")**

```
# importing the data
data = pd.read_csv(r"G:\AI&ML\ML projects\Traffic_volume\traffic volume.csv")
```

Note: r stands for "raw" and will cause backslashes in the string to be interpreted as

actual backslashes rather than special characters.

- If the dataset is in the same directory of your program, you can directly read it, without giving raw as r.
- Our Dataset weatherAus.csv contains the following Columns
- Holiday - working day or holiday
- Temp- temperature of the day
- Rain and snow – whether it is raining or snowing on that day or not
- Weather = describes the weather conditions of the day
- Date and time = represents the exact date and time of the day
- Traffic volume – output column

The output column to be predicted is Traffic volume. Based on the input variables we predict the volume of the traffic. The predicted output gives them a fair idea of the count of traffic

3.4 Analyse The Data:

head() method is used to return top n (5 by default) rows of a DataFrame or series.

```
# displaying first 5 columns of the data
data.head()
```

| | holiday | temp | rain | snow | weather | date | Time | traffic_volume |
|---|---------|--------|------|------|---------|------------|----------|----------------|
| 0 | None | 288.28 | 0.0 | 0.0 | Clouds | 02-10-2012 | 09:00:00 | 5545 |
| 1 | None | 289.36 | 0.0 | 0.0 | Clouds | 02-10-2012 | 10:00:00 | 4516 |
| 2 | None | 289.58 | 0.0 | 0.0 | Clouds | 02-10-2012 | 11:00:00 | 4767 |
| 3 | None | 290.13 | 0.0 | 0.0 | Clouds | 02-10-2012 | 12:00:00 | 5026 |
| 4 | None | 291.14 | 0.0 | 0.0 | Clouds | 02-10-2012 | 13:00:00 | 4918 |

describe() method computes a summary of statistics like count, mean, standard deviation, min, max, and quartile values.

```
data.describe()
```

The output is as shown below


```
# used to understand the descriptive analysis of the data
data.describe()
```

| | temp | rain | snow | traffic_volume |
|-------|--------------|--------------|--------------|----------------|
| count | 48151.000000 | 48202.000000 | 48192.000000 | 48204.000000 |
| mean | 281.205351 | 0.334278 | 0.000222 | 3259.818355 |
| std | 13.343675 | 44.790062 | 0.008169 | 1986.860670 |
| min | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 272.160000 | 0.000000 | 0.000000 | 1193.000000 |
| 50% | 282.460000 | 0.000000 | 0.000000 | 3380.000000 |
| 75% | 291.810000 | 0.000000 | 0.000000 | 4933.000000 |
| max | 310.070000 | 9831.300000 | 0.510000 | 7280.000000 |

From the data, we infer that there are only decimal values and no categorical values. **info()** gives information about the data - paste the image here.

```
# used to display the basic information of the data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48204 entries, 0 to 48203
Data columns (total 8 columns):
#   Column      Non-Null Count  Dtype
---  -
0   holiday     48204 non-null  object
1   temp        48151 non-null  float64
2   rain        48202 non-null  float64
3   snow        48192 non-null  float64
4   weather     48155 non-null  object
5   date        48204 non-null  object
6   Time        48204 non-null  object
7   traffic_volume 48204 non-null  int64
dtypes: float64(3), int64(1), object(4)
memory usage: 2.9+ MB
```

3.5 Handling The Missing Values:

1. The Most important step in data pre-processing is dealing with missing data, the presence of missing data in the dataset can lead to low accuracy.
2. Check whether any null values are there or not. if it is present then the following can be done.

```
# used to display the null values of the data  
data.isnull().sum()
```

```
holiday      0  
temp        53  
rain         2  
snow        12  
weather      49  
date         0  
Time         0  
traffic_volume 0  
dtype: int64
```

There are missing values in the dataset, we will fill the missing values in the columns.

3. We are using mean and mode methods for filling the missing values

- Columns such as temp, rain, and snow are the numeric columns, when there is a numeric column you should fill the missing values with the mean/median method. so here we are using the mean method to fill the missing values.
- Weather column has a categorical data type, in such case missing data needs to be filled with the most repeated/ frequent value. Clouds are the most repeated value in the column, so imputing with clouds value.

```
data['temp'].fillna(data['temp'].mean(),inplace=True)  
data['rain'].fillna(data['rain'].mean(),inplace=True)  
data['snow'].fillna(data['snow'].mean(),inplace=True)  
  
print(Counter(data['weather']))  
  
Counter({'Clouds': 15144, 'Clear': 13383, 'Mist': 5942, 'Rain': 5665, 'Snow': 2875, 'Drizzle': 1818, 'H  
'Fog': 912, nan: 49, 'Smoke': 20, 'Squall': 4})  
  
data['weather'].fillna('Clouds',inplace=True)
```

3.6 Data Visualization:

Data visualization is where a given data set is presented in a graphical format. It helps the detection of patterns, trends and correlations that might go undetected in text-based data.

Understanding your data and the relationship present within it is just as important as any algorithm used to train your machine learning model. In fact, even the most sophisticated

machine learning models will perform poorly on data that wasn't visualized and understood properly.

- To visualize the dataset we need libraries called Matplotlib and Seaborn.
- The Matplotlib library is a Python 2D plotting library that allows you to generate plots, scatter plots, histograms, bar charts etc.

Let's visualize our data using Matplotlib and seaborn library.

Before diving into the code, let's look at some of the basic properties we will be using when plotting.

xlabel: Set the label for the x-axis.

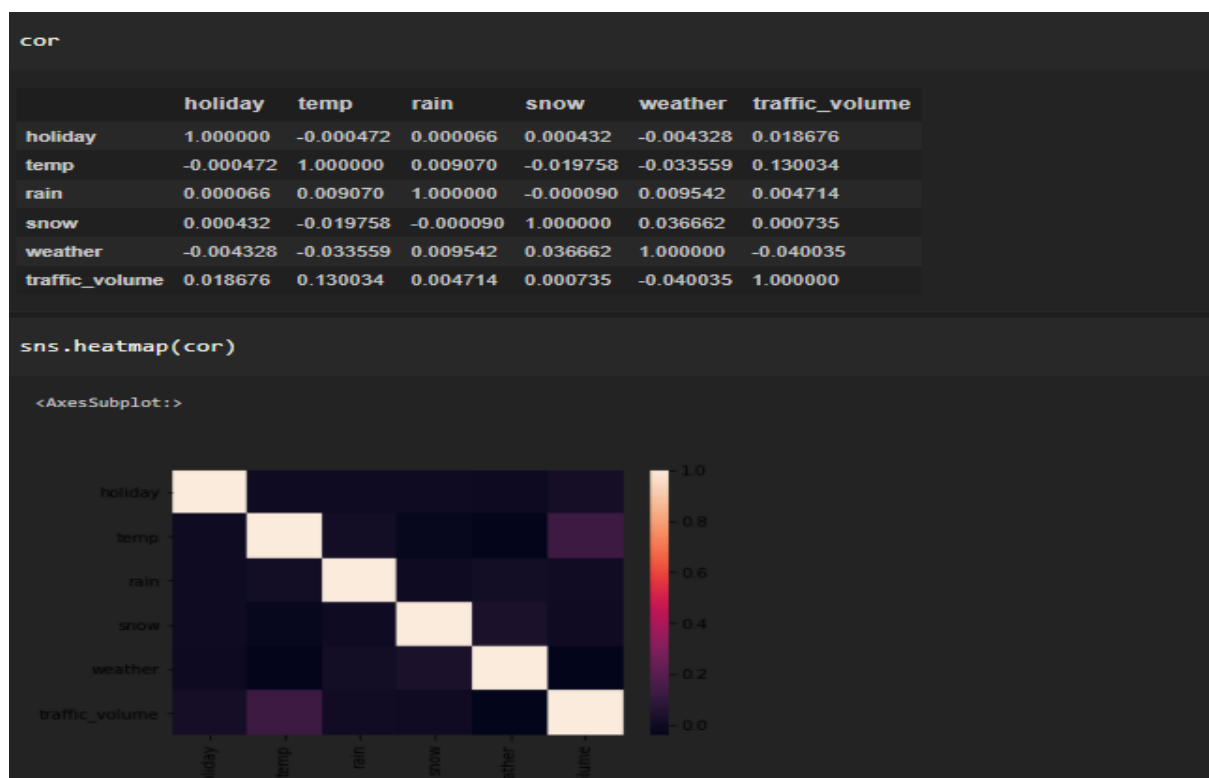
ylabel: Set the label for the y-axis.

title: Set a title for the axes.

Legend: Place a legend on the axes.

1. data.corr() gives the correlation between the columns

Correlation is a statistical term describing the degree to which two variables move in coordination with one another. If the two variables move in the same direction, then those variables are said to have a positive correlation. If they move in opposite directions, then they have a negative correlation. -



- Correlation strength varies based on colour, lighter the colour between two variables, more the strength between the variables, darker the colour displays the weaker correlation
- We can see the correlation scale values on the left side of the above image

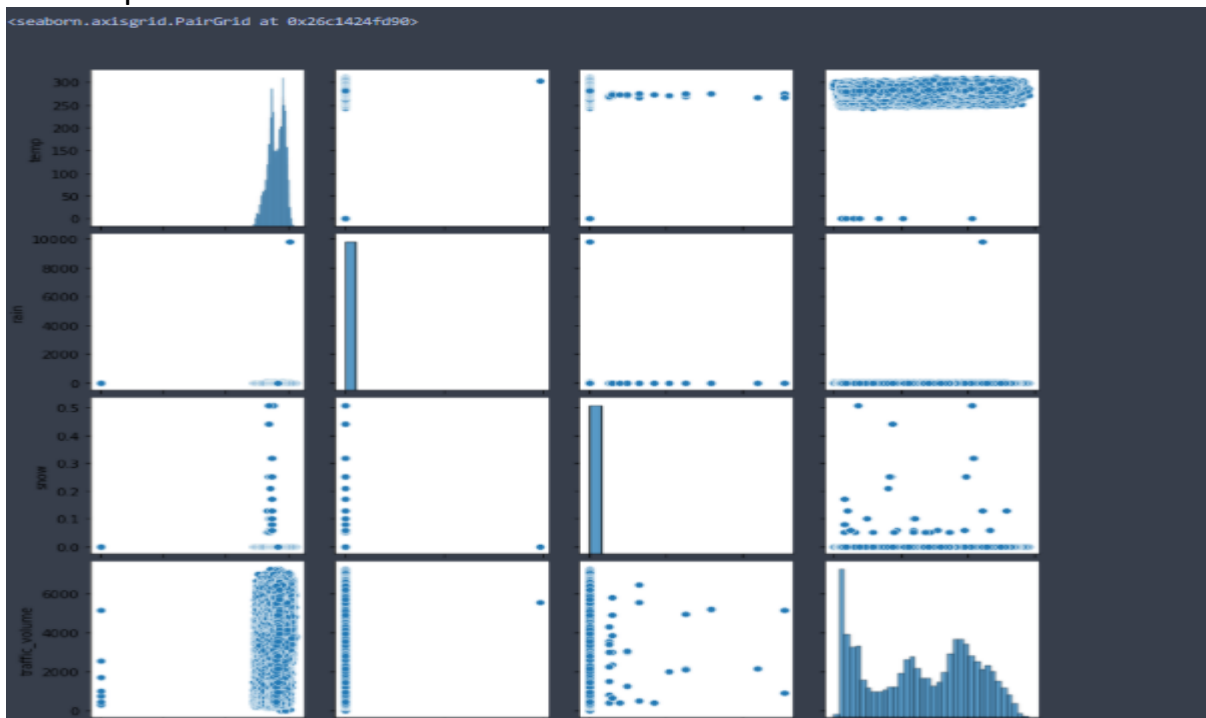
2. Pair Plot: Plot pairwise relationships in a dataset.

A pair plot is used to understand the best set of features to explain a relationship between two variables or to form the most separated clusters. It also helps to form some simple classification models by drawing some simple lines or making a linear separation in our data-set.

- By default, this function will create a grid of Axes such that each numeric variable in data will be shared across the y-axes across a single row and the x-axes across a single column. The diagonal plots are treated differently: a univariate distribution plot is drawn to show the marginal distribution of the data in each column.
- We implement this using the below code.

Code:- sns.pairplot(data)

The output is as shown below-



Pair plot usually gives pairwise relationships of the columns in the dataset

From the above pair plot, we infer that

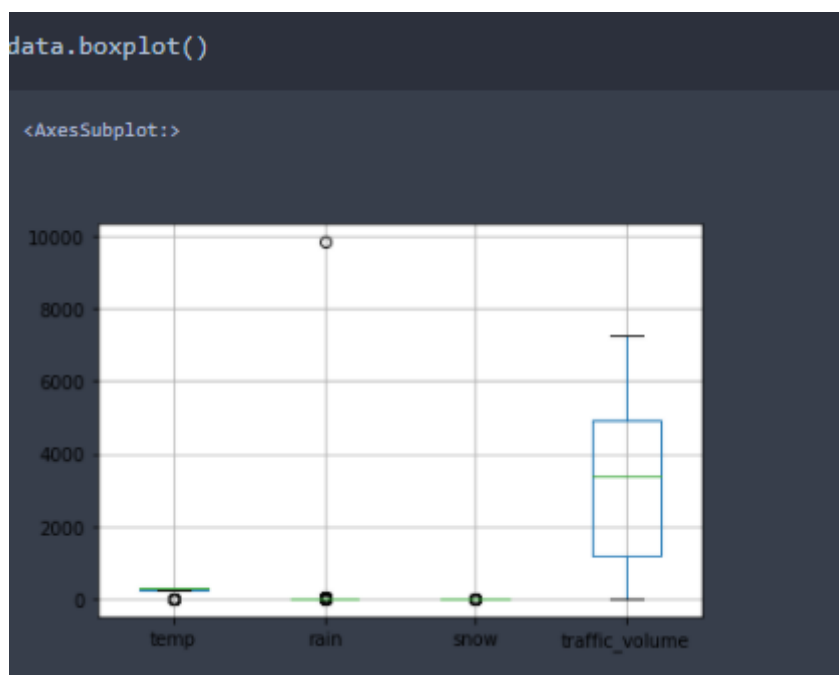
1. From the above plot we can draw inferences such as linearity and strength between the variables. how features are correlated (positive, neutral and negative)

3. Box Plot:

Box-plot is a type of chart often used in explanatory data analysis. Box plots visually show the distribution of numerical data and skewness through displaying the data quartiles (or percentiles) and averages.

Box plots are useful as they show the average score of a data set. The median is the average value from a set of data and is shown by the line that divides the box into two parts. Half the scores are greater than or equal to this value and half are less.

jupyter has a built-in function to create a boxplot called `boxplot()`. A boxplot plot is a type of plot that shows the spread of data in all the quartiles.



From the above box plot, we infer how the data points are spread and the existence of the outliers

4. Data and time columns need to be split into columns so that analysis and training of the model can be done in an easy way, so we use the split function to convert date into the year, month and day. time column into hours, minutes and seconds.

```
# splitting the date column into year,month,day
data[["day", "month", "year"]] = data["date"].str.split("-", expand = True)

# splitting the date column into year,month,day
data[["hours", "minutes", "seconds"]] = data["Time"].str.split(":", expand = True)

data.drop(columns=['date', 'Time'],axis=1,inplace=True)

data.head()
```

| | holiday | temp | rain | snow | weather | traffic_volume | day | month | year | hours | minutes | seconds |
|---|---------|--------|------|------|---------|----------------|-----|-------|------|-------|---------|---------|
| 0 | 7 | 288.28 | 0.0 | 0.0 | 1 | 5545 | 02 | 10 | 2012 | 09 | 00 | 00 |
| 1 | 7 | 289.36 | 0.0 | 0.0 | 1 | 4516 | 02 | 10 | 2012 | 10 | 00 | 00 |
| 2 | 7 | 289.58 | 0.0 | 0.0 | 1 | 4767 | 02 | 10 | 2012 | 11 | 00 | 00 |
| 3 | 7 | 290.13 | 0.0 | 0.0 | 1 | 5026 | 02 | 10 | 2012 | 12 | 00 | 00 |
| 4 | 7 | 291.14 | 0.0 | 0.0 | 1 | 4918 | 02 | 10 | 2012 | 13 | 00 | 00 |

3.7 Splitting The Data Into Dependent And Independent Variables:

In machine learning, the concept of the dependent variable (y) and independent variables(x) is important to understand. Here, the Dependent variable is nothing but output in dataset and the independent variable is all inputs in the dataset.

- With this in mind, we need to split our dataset into the matrix of independent variables and the vector or dependent variable. Mathematically, Vector is defined as a matrix that has just one column.

To read the columns, we will use iloc of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

Let's split our dataset into independent and dependent variables.

y = data[traffic_volume] - independent

x = data.drop(traffic_volume,axis=1)

```
y = data['traffic_volume']
x = data.drop(columns=['traffic_volume'],axis=1)
```

3.8 Feature Scaling:

There is a huge disparity between the x values so let us use feature scaling.

Feature scaling is a method used to normalize the range of independent variables or features of data.

```
y = data['traffic_volume']
x = data.drop(columns=['traffic_volume'],axis=1)

names = x.columns

from sklearn.preprocessing import scale

x = scale(x)

x = pd.DataFrame(x,columns=names)

x.head()
```

| | holiday | temp | rain | snow | weather | day | month | year | hours | minutes | seconds |
|---|----------|----------|-----------|-----------|-----------|-----------|---------|-----------|-----------|---------|---------|
| 0 | 0.015856 | 0.530485 | -0.007463 | -0.027235 | -0.566452 | -1.574903 | 1.02758 | -1.855294 | -0.345548 | 0.0 | 0.0 |
| 1 | 0.015856 | 0.611467 | -0.007463 | -0.027235 | -0.566452 | -1.574903 | 1.02758 | -1.855294 | -0.201459 | 0.0 | 0.0 |
| 2 | 0.015856 | 0.627964 | -0.007463 | -0.027235 | -0.566452 | -1.574903 | 1.02758 | -1.855294 | -0.057371 | 0.0 | 0.0 |
| 3 | 0.015856 | 0.669205 | -0.007463 | -0.027235 | -0.566452 | -1.574903 | 1.02758 | -1.855294 | 0.086718 | 0.0 | 0.0 |
| 4 | 0.015856 | 0.744939 | -0.007463 | -0.027235 | -0.566452 | -1.574903 | 1.02758 | -1.855294 | 0.230807 | 0.0 | 0.0 |

- After scaling the data will be converted into an array form
- Loading the feature names before scaling and converting them back to data frame after standard scaling is applied

3.9 Splitting The Data Into Train And Test:

When you are working on a model and you want to train it, you obviously have a dataset. But after training, we have to test the model on some test datasets. For this, you will a dataset which is different from the training set you used earlier. But it might not always be possible to have so much data during the development phase. In such cases, the solution is to split the dataset into two sets, one for training and the other for testing.

- The train-test split is a technique for evaluating the performance of a machine learning algorithm.
- Train Dataset: Used to fit the machine learning model.
- Test Dataset: Used to evaluate the fit machine learning model.
- In general you can allocate 80% of the dataset to the training set and the remaining 20% to test.
- Now split our dataset into train set and test using train_test_split class from sci-kit learn library.

```
from sklearn import model_selection
```

```
x_train,x_test,y_train,y_test=model_selection.train_test_split(x,y,test_size=0.2,random_state=0)
```

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

4.Model Building:

The model building includes the following main tasks

- o Training and testing the model
- o Evaluation of Model
- o Save the Model

4.1: Training And Testing The Model:

- Once after splitting the data into train and test, the data should be fed to an algorithm to build a model.

- There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have it may be Classification algorithms are Regression algorithms.

1. Linear Regression
2. Decision Tree Regressor
3. Random Forest Regressor
4. KNN
5. svm
5. xgboost

Steps in Building the model:-

Initialize the model -

```
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost

lin_reg = linear_model.LinearRegression()
Dtree = tree.DecisionTreeRegressor()
Rand = ensemble.RandomForestRegressor()
svr = svm.SVR()
XGB = xgboost.XGBRegressor()
```

Fit the models with x_train and y_train -

```
lin_reg.fit(x_train,y_train)
Dtree.fit(x_train,y_train)
Rand.fit(x_train,y_train)
svr.fit(x_train,y_train)
XGB.fit(x_train,y_train)
```

Predict the y_train values and calculate the accuracy -

```
p1 = lin_reg.predict(x_train)
p2 = Dtree.predict(x_train)
p3 = Rand.predict(x_train)
p4 = svr.predict(x_train)
p5 = XGB.predict(x_train)
```

We're going to use the x-train and y-train obtained above in the train_test_split section to train our Random forest regression model. We're using the fit method and passing the parameters as shown below.

We are using the algorithm from Scikit learn library to build the model as shown below, Once the model is trained, it's ready to make predictions. We can use the `predict` method on the model and pass `x_test` as a parameter to get the output as `y_pred`.

Notice that the prediction output is an array of real numbers corresponding to the input array.

4.2 Evaluation Of Model:

After training the model, the model should be tested by using the test data which is been separated while splitting the data for checking the functionality of the model.

Regression Evaluation Metrics:

These model evaluation techniques are used to find out the accuracy of models built in the Regression type of machine learning models. We have three types of evaluation methods.

- `R-square_score`
- `RMSE` – root mean squared error

1. R-squared _score -

Formula

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2 = coefficient of determination

RSS = sum of squares of residuals

TSS = total sum of squares

It is the ratio of the number of correct predictions to the total number of input samples.

Calculating the r2 score value using for all the models.

```
from sklearn import metrics
```

```
print(metrics.r2_score(p1,y_train))
print(metrics.r2_score(p2,y_train))
print(metrics.r2_score(p3,y_train))
print(metrics.r2_score(p4,y_train))
print(metrics.r2_score(p5,y_train))
```

```
-5.517285423636865
1.0
0.9747969952887571
-12.188104231382285
0.8349874938269883
```

```
p2 = Dtree.predict(x_test)
p3 = Rand.predict(x_test)
p4 = svr.predict(x_test)
p5 = XGB.predict(x_test)
```

```
print(metrics.r2_score(p1,y_test))
print(metrics.r2_score(p2,y_test))
print(metrics.r2_score(p3,y_test))
print(metrics.r2_score(p4,y_test))
print(metrics.r2_score(p5,y_test))
```

```
-5.399396398322181
0.6920677009517378
0.8031828166614183
-11.972215715232434
0.7922184852381723
```

- After considering both r squared values of test and train we concluded that random forest regressor is giving the better value, it is able to explain the 97% of the data in train values.
- Random forest gives the best r2-score, so we can select this model.

RMSE –Root Mean Square Error

Formula

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^N (x_i - \hat{x}_i)^2}{N}}$$

RMSE = root-mean-square deviation

i = variable i

N = number of non-missing data points

x_i = actual observations time series

\hat{x}_i = estimated time series

Randomforest gives the best r-score value

```
#RMSE values
```

```
MSE = metrics.mean_squared_error(p3,y_test)
```

```
np.sqrt(MSE)
```

```
798.4970439382182
```

RMSE value for Random forest is very less when compared with other models, so saving the Random forest model and deploying using the following process

4.3 Save The Model:

After building the model we have to save the model.

Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions or transport data over the network. wb indicates write method and rd indicates read method.

This is done by the below code

```
import pickle

pickle.dump(Rand,open("model.pkl",'wb'))
pickle.dump(le,open("encoder.pkl",'wb'))
```

5.Application Building:

In this section, we will be building a web application that is integrated into the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

- Building HTML Pages
- Building server-side script

5.1 Build HTML Code:

In this HTML page, we will create the front-end part of the web page. On this page, we will accept input from the user and Predict the values.

For more information regarding HTML, click on the [link](#).

In our project we have HTML files, they are

1.index.html - paste the image

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Traffic Volume Estimation</title>
</head>

<body background= "https://cdn.vox-cdn.com/thumbor/fv0AbZferUT6SfU5n9DxRmJcTMwE7zBj2kQKbu7Gcf/1600:900/cdn.vox-cdn.com/uploads/thorus_image/image/4421936f/7149d096_0_0.jpg" ext="black">

<div class="login">
<center><h1>Traffic Volume Estimation</h1></center>

<!-- Main Input For Receiving Query to our ML -->
<form action="{ {{ url_for('predict')}} }" method="post">
<h1>Please enter the following details</h1>

</style></head>


<label for="holiday">holiday:</label>
<select id="holiday" name="holiday">
<option value=7>None</option>
<option value=1>Columbus Day</option>
<option value=10>Veterans Day</option>
<option value=9>Thanksgiving Day</option>
<option value=0>Christmas Day</option>
<option value=6>New Years Day</option>
<option value=11>Washington's Birthday</option>
<option value=5>Memorial Day</option>
<option value=2>Independence Day</option>
<option value=8>State Fair</option>
<option value=3>Labor Day</option>
<option value=4>Martin Luther King Jr Day</option>

</select> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&br>

<br>
<label>temp:</label>
<input type="number" name="temp" placeholder="temp" required="required" /><br>

<br>
<label>rain:</label>
<input type="number" min="0" max="1" name="rain" placeholder="rain" required="required" /><br>

<br>
<label>snow:</label>
<input type="number" min="0" max="1" name="snow" placeholder="snow" required="required" /><br>

<br>

<label for="weather">weather:</label>
<select id="weather" name="weather">
<option value=1>Clouds</option>
<option value=0>Clear</option>
<option value=6>Rain</option>
<option value=2>Drizzle</option>
<option value=5>Mist</option>
<option value=4>Haze</option>
<option value=3>Fog</option>
<option value=10>Thunderstorm</option>
<option value=8>Snow</option>
<option value=9>Squall</option>
<option value=7>Smoke</option>

</select> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&br>

<br>
<label>year:</label>
<input type="number" min="2012" max="2022" name="year" placeholder="year" required="required" /><br>

<br>
<label>month:</label>
<input type="number" min="1" max="12" name="month" placeholder="month" required="required" /><br>

<br>
<label>day:</label>
<input type="number" min="1" max="31" name="day" placeholder="day" required="required" /><br>

<br>
<label>hours:</label>
<input type="number" min="0" max="24" name="hours" placeholder="hours" required="required" /><br>

<br>
<label>minutes:</label>
<input type="number" min="0" max="60" name="minutes" placeholder="minutes" required="required" /><br>

<br>
<label>seconds:</label>
<input type="number" min="0" max="60" name="seconds" placeholder="seconds" required="required" /><br>

<br>
<br><br>

<button type="submit" class="btn btn-primary btn-block btn-large" style="height:30px;width:200px">Predict</button>

</form>
<br>

{{ prediction_text }}

<br>
<br>
<script src="data:image/jpg;base64,{url}" alt="Submit Form" height="100" width="222" onError="this.style.display='none'"/>
```

2. The HTML page looks like this-

Traffic Volume Estimation

Please enter the following details

holiday:

temp:

rain:

snow:

weather:

year:

month:

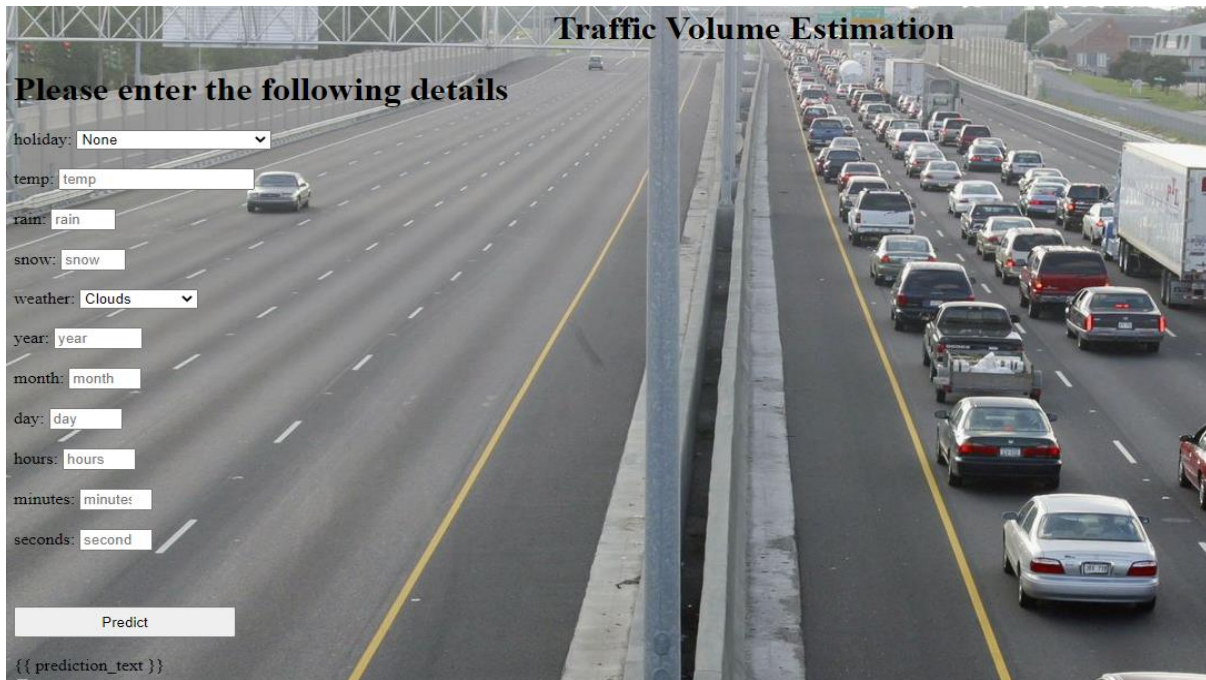
day:

hours:

minutes:

seconds:

{{ prediction_text }}



3. It will display all the input parameters and the prediction text will display the output value of the data given by the user.



5.2 Main Python Script:

Let us build an app.py flask file which is a web framework written in python for server-side scripting. Let's see step by step procedure for building the backend application.

In order to develop web API with respect to our model, we basically use the Flask framework which is written in python.

Line 1-9 We are importing necessary libraries like Flask to host our model request

Line 12 Initialise the Flask application

Line 13 Loading the model using pickle

Line 16 Routes the API URL

Line 18 Rendering the template. This helps to redirect to the home page. In this home page, we give our input and ask the model to predict

In line 23 we are taking the inputs from the form

Line 28 Feature Scaling the inputs

Line 31 Predicting the values given by the user

Line 32-35 if the output is false render no chance template If the output is True render chance template

Line 36 The value of `__name__` is set to `__main__` when the module run as the main program otherwise it is set to the name of the module.

```
1 import numpy as np
2 import pickle
3 import joblib
4 import matplotlib
5 import matplotlib.pyplot as plt
6 import time
7 import pandas
8 import os
9 from flask import Flask, request, jsonify, render_template
10
11 app = Flask(__name__)
12 model = pickle.load(open('G:/AI&ML/ML projects/Traffic_volume/model.pkl', 'rb'))
13 scale = pickle.load(open('C:/Users/SmartbridgePC/Desktop/AI&ML/Guided projects/scale.pkl', 'rb'))
14
15 @app.route('/')# route to display the home page
16 def home():
17     return render_template('index.html') #rendering the home page
18
19 @app.route('/predict',methods=["POST","GET"])# route to show the predictions in a web UI
20 def predict():
21     # reading the inputs given by the user
22     input_feature=[float(x) for x in request.form.values() ]
23     features_values=[np.array(input_feature)]
24     names = [['holiday', 'temp', 'rain', 'snow', 'weather', 'year', 'month', 'day',]
25             'hours', 'minutes', 'seconds']]
26     data = pandas.DataFrame(features_values,columns=names)
27     data = scale.fit_transform(data)
28     data = pandas.DataFrame(data,columns = names)
29     # predictions using the loaded model file
30     prediction=model.predict(data)
31     print(prediction)
32     text = "Estimated Traffic Volume is : "
33     return render_template("index.html",prediction_text = text + str(prediction))
34     # showing the prediction results in a UI
35 if __name__=="__main__":
36     # app.run(host='0.0.0.0', port=8080,debug=True) # running the app
37     port=int(os.environ.get('PORT',5000))
38     app.run(port=port,debug=True,use_reloader=False)
```


5.3 Run The App:

Open anaconda prompt from the start menu

- Navigate to the folder where your python script is.
- Now type the “python app.py” command

Navigate to the localhost where you can view your web page, Then it will run on **localhost:5000**

```
In [1]: runfile('G:/AI&ML/ML projects/Traffic_volume/app.py', wdir='G:/AI&ML/ML projects/Traffic_volume')
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
C:\Users\SmartbridgePC\anaconda3\lib\site-packages\sklearn\base.py:324:
UserWarning: Trying to unpickle estimator StandardScaler from version 0.23.2 when
loading version 1.0.1. This might lead to breaking code or invalid results. Use at
your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-
sustainability-limitations
  warnings.warn(
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

5.4 OUTPUT:

- Copy the HTTP link and paste it in google link tab, it will display the form page
- Enter the values as per the form and click on predict button
- It will redirect to the page based on prediction output
- The output will be displayed in the prediction text as Estimated Traffic volume is in units.

Traffic volume estimation

Estimated Traffic Volume is :[4495.41]



□

□