A project report on

TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning

By

SmartInternz

Project Mentor :M.Ganesh

Project Id : LTVIP2025TMID59882

Team Members:

1.T.SAI RAGHAVENDRA (22HM1A05E4)

2.Y.NANDINI (22HM1A05B7)

3.S.AFRID (23HM5A0522)

4.S.RAJAK (22HM1A05B7)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,Annamacharya Institute of Technology & Sciences, Utukur (Post), Chinthakomma Dinne (V&M), Kadapa, YSR (Dist) Andhra Pradesh

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Institue**of **ANNAMACHARYA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. A. Sudhakar Reddy,** principal of Annamacharya Institute of technology and sciences and**Dr.C.V.Subbaiah, M.tech., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project mentor **MR.M.Ganesh,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project

ABSTRACT

The purpose of this project is to design and develop a traffic assessment system. Traffic estimate is determined by the amount of traffic congestion. Traffic jams cause people to lose valuable time, energy and frustration every day. Congestion is a global problem that affects all levels of society. The most common causes of traffic congestion are any driver getting stuck in a traffic jam on their journey. Accidents such as road accidents and road accidents often lead to unexpected unforeseen delays. There are also bad weather conditions due to low traffic flow speeds. It is difficult to accurately estimate traffic flow due to the very large data of the transportation system. This fact prompted us to work on a traffic prediction system to accurately and timely assess traffic flow information. We plan to use machine learning for prediction and regression based algorithm for image detection to analyze the bulk data of the transport system, we will use various graphical user fronts for interactive application.Machine learning provides better accuracy for Traffic volume flow prediction. It's addressed as a major element for the success of advanced traffic volume management systems, advanced public transportation systems, and traveler information systems. The rationale of this extension is to develop a prescient demonstration utilizing different machine learning calculations and to record the end-to-end steps. The Metro Interstate Activity Volume dataset could also be a relapse circumstance where we are trying to anticipate the esteem of a ceaseless variable. We'll be analyzing how the drift of month-to-month interstate activity volume changes over an extended time between 2012 and 2018.

Key Words: Traffic Volume, Random Forest, Machine Learning, RSME, Flask

CONTENTS

GitHub & Project Demo Link

1.Introduction

1.1 Project Overview

Traffic congestion in urban areas has become a major challenge, leading to increased pollution, fuel consumption, and loss of productivity. Traditional methods of traffic volume monitoring rely on fixed sensors and manual observation, which are often expensive and limited in scalability. **TrafficTelligence** is an intelligent system designed to estimate and analyze traffic volume using machine learning models. The system leverages historical and real-time traffic data to provide accurate traffic volume predictions, enabling authorities to make data-driven decisions for better traffic management.

1.2 Purpose

The purpose of this project is to develop a machine learning-based traffic volume estimation system that:

- Accurately predicts traffic volume using historical datasets
- Helps reduce congestion by enabling proactive measures
- Supports urban planners in traffic flow optimization
- Enhances the smart city infrastructure

2.Ideation Phase

2.1 The Problem Statements

Traffic congestion results in increased travel time, environmental degradation, and economic loss. Traditional systems lack intelligence and adaptability. There is a need for an automated, scalable, and intelligent system that can predict traffic volume based on various factors such as time, weather, and past data

Date	20june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	2 Marks

Customer Problem Statement Template:

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

I am	Describe customer with 3-4 key characteristics - *who are they?*	Describe the customer and their attributes here
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way – *what bothers them most?*	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists – *what needs to be solved?*	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view – *how does it impact them emotionally?*	Describe the emotions the result from experiencing the problems or barriers

Reference: https://miro.com/templates/customer-problem-statement/

Example:

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
PS-1	A daily commuter	Reach my destination on time	I often get stuck in traffic unexpectedly	I don't have access to real-time or predictive traffic data	Frustrated, anxious, and often late

PS-2	A traffic planner or city authority	Manage urban traffic flow efficiently	I can't predict future traffic surges accurately	The current systems don't use intelligent or predictive methods	Inefficient, reactive, and under pressure

2.1 Empathize & Discover

Date	31 January 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	4 Marks

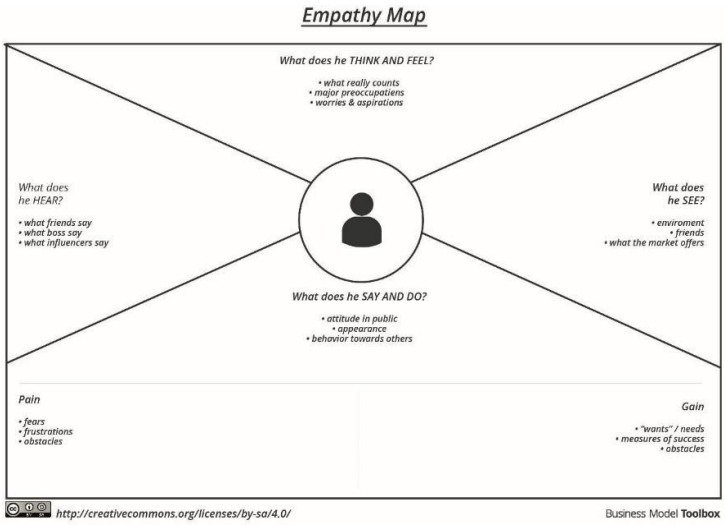
Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

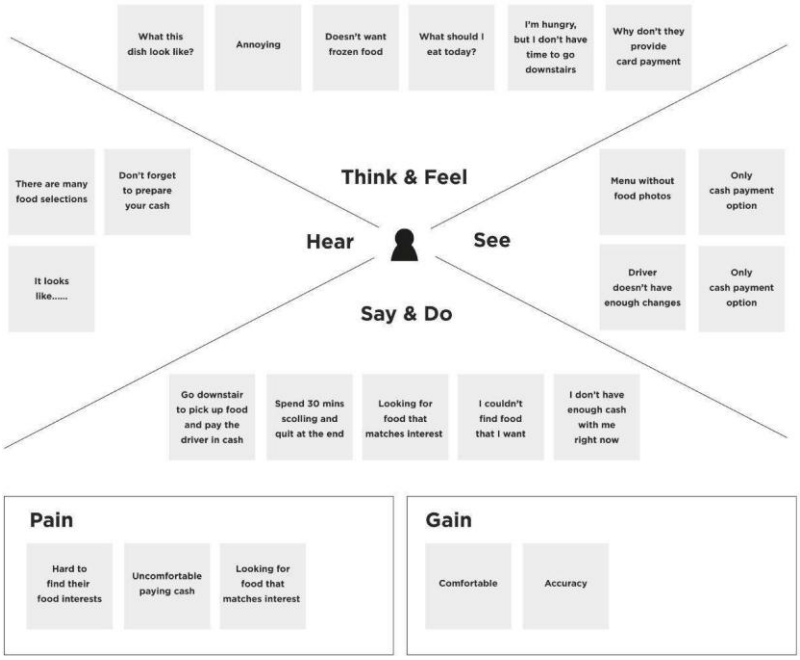
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Example:



Reference: https://www.mural.co/templates/empathy-map-canvas

Example: Food Ordering & Delivery Application



2.3. Brainstorm & Idea Prioritization Template

Date	21 june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum	4 Marks

Marks	

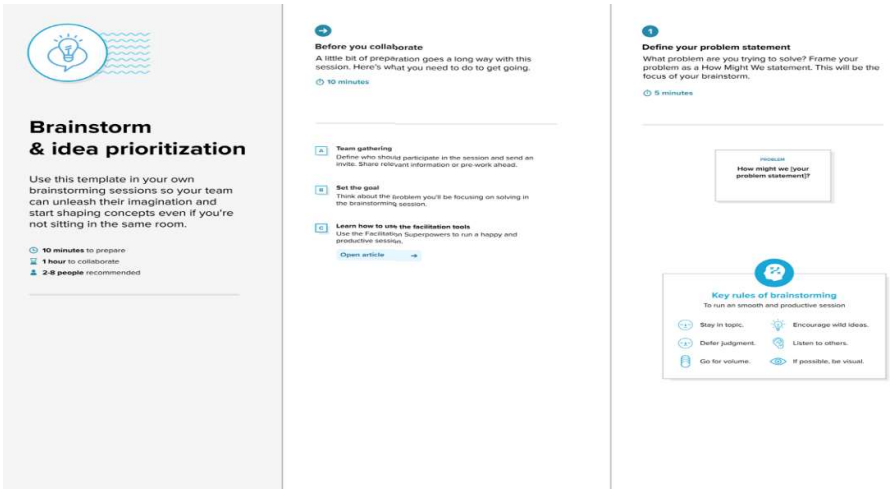
Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

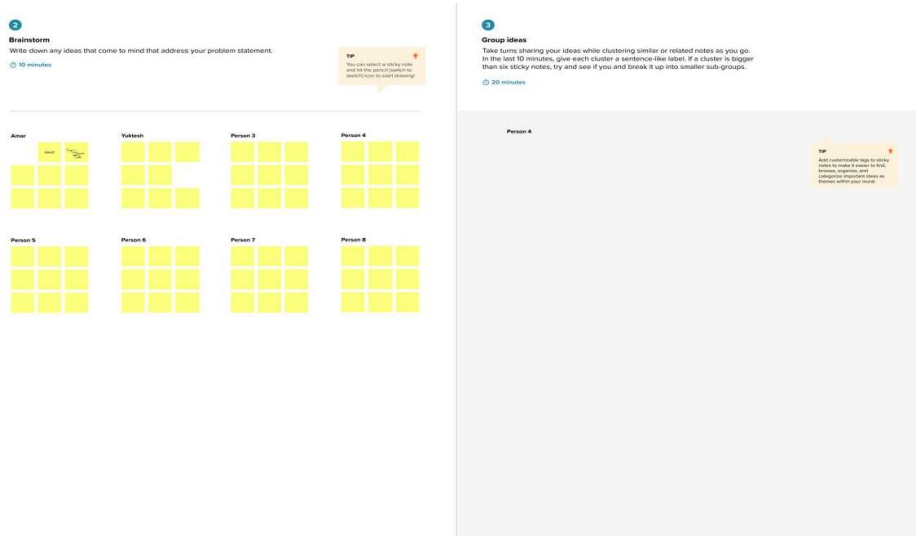
Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Reference: https://www.mural.co/templates/brainstorm-and-idea-prioritization

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Describe how users (e.g., traffic officers, city planners, drivers) interact with your solution.

3.2. Solution Requirements (Functional & Non-functional)

Date	22 june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	4 Marks

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN

FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Traffic Data Input	Upload CSV/Excel data Live data via API Manual data entry
FR-4	Data Preprocessing	Handle missing values Convert date/time formats Outlier detection
FR-5	Traffic Volume Prediction	Use trained ML model to predict Allow user input for prediction Return predictions in real-time
FR-6	Data Visualization	Line/bar chart of traffic volume Compare actual vs predicted values Interactive dashboard

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Simple and intuitive UI for both commuters and traffic planners
NFR-2	**Security**	Secure login with password encryption and OAuth 2.0 support
NFR-3	**Reliability**	Model should provide consistent predictions across similar inputs
NFR-4	**Performance**	System should respond to prediction requests in under 2 seconds
NFR-5	**Availability**	The system should be available 99.9% of the time (if deployed)
NFR-6	**Scalability**	Should be able to handle increasing amounts of data or users without performance degradation

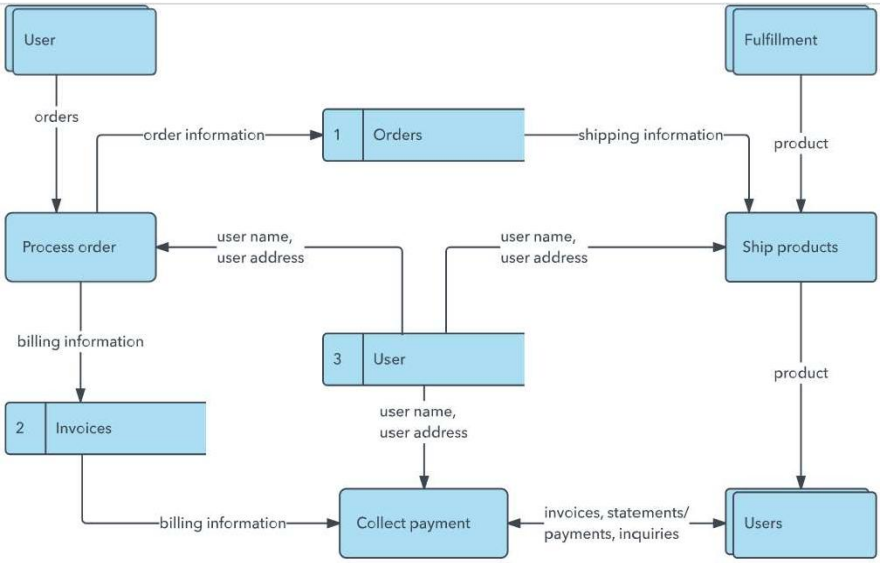
3.3 Data Flow Diagram & User Stories

Date	22 june2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	4 Marks

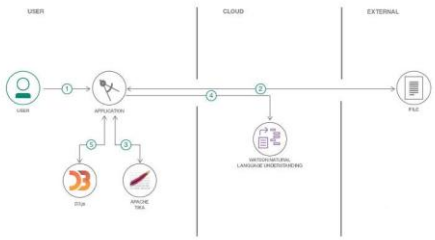
Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Example: [(Simplified)]



Flow



1. User configures credentials for the Watson Natural Language Understanding service and starts the app.
2. User selects data file to process and load.
3. Apache Tika extracts text from the data file.
4. Extracted text is passed to Watson NLU for enrichment.
5. Enriched data is visualized in the UI using the D3.js library.

User Stories

Use the below template to list all the user stories for the product.

User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Release**
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
USN-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
USN-2	Registration	USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
USN-3	Registration	USN-4	As a user, I can register for the application through Gmail	I can access my account / dashboard	Medium	Sprint-1
USN-4	Login	USN-5	As a user, I can log into the application by entering email & password	I can register & access the dashboard with Facebook Login	High	Sprint-1
	Dashboard	USN-5	As a user, I	I can		

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
			will receive a confirmation email once I have registered for the application	register & access the dashboard with Gmail		
Customer (Web user)	Registration	USN-8	As a user, I can register for the application through Gmail	I can access my dashboard securely		
Customer Care Executive	Registration	USN-7		The system accepts and processes my data		

3.4 Technology Stack (Architecture & Stack)

Date	31 January 3035
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Example: Order processing during pandemics for offline mode

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/

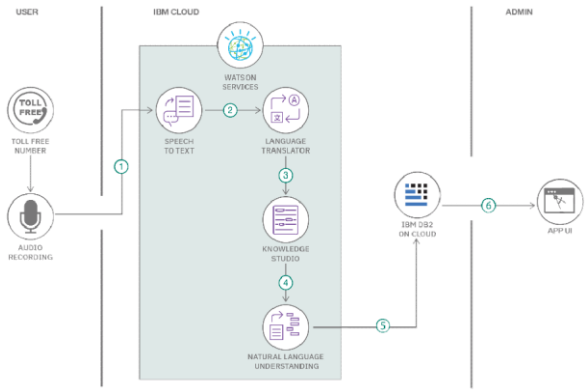


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.

9.	External API-2	Purpose of External API used in the application	Aadhar API, etc.
10.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-	Technology used

S.No	Characteristics	Description	Technology
		services)	
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Technology used
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Technology used

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d

4. Project Design Phase

4.1. Problem – Solution Fit

Date	23 june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum	2 Marks

Marks	

Problem – Solution Fit :

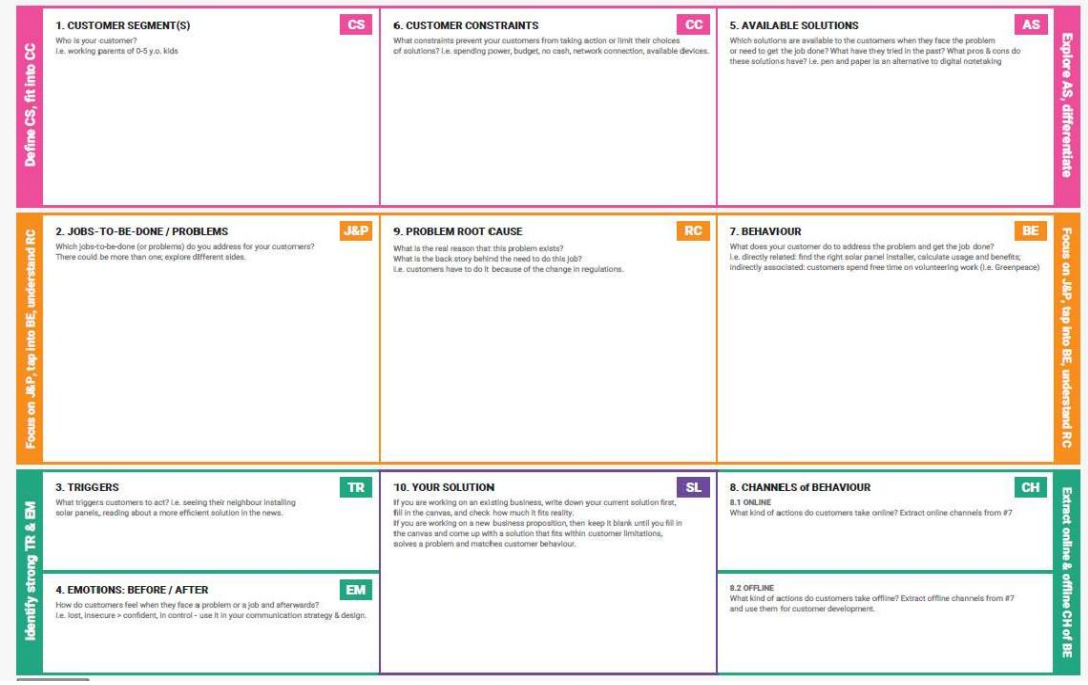
Problem Recap:

Urban traffic congestion is unpredictable and leads to increased travel time, fuel consumption, and stress. Traditional systems for monitoring and managing traffic lack predictive intelligence and require expensive infrastructure.

Solution Offered:

TrafficTelligence aims to provide an intelligent, low-cost, and scalable solution using machine learning to estimate and predict traffic volume. This empowers city authorities and commuters to make informed decisions in real time, helping to ease traffic flow and reduce congestion.

Template:



4.2 Proposed Solution

Date	23 june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	2 Marks

Proposed Solution :

Project team shall fill the following information in the proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Urban traffic congestion leads to inefficiencies in travel time, fuel usage, and pollution. Existing traffic monitoring systems lack predictive intelligence and require expensive infrastructure.
2.	Idea / Solution description	rafficTelligence is a machine learning-based system that predicts traffic volume using historical data (like time, date, and weather)
3.	Novelty / Uniqueness	Unlike static or sensor-based solutions, TrafficTelligence uses adaptive machine learning models that improve over time and do not require heavy infrastructure.
4.	Social Impact / Customer Satisfaction	his system improves commuter experiences by reducing travel time and stress.
5.	Business Model (Revenue Model)	The solution can be offered as a SaaS platform for city municipalities and enterprises. Revenue streams include subscription plans,
6.	Scalability of the Solution	TrafficTelligence is easily scalable to other cities or regions by training on local traffic datasets.

4.3. Solution Architecture

Date	23 june2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	4 Marks

Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:

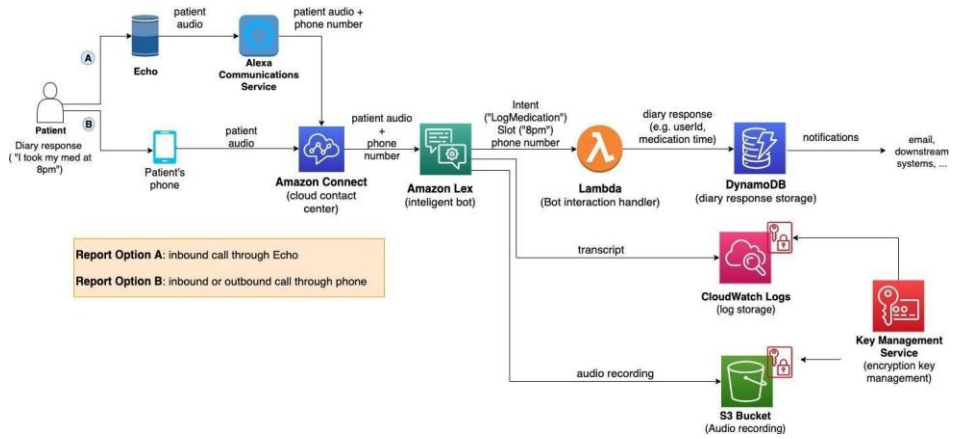


Figure 1: Architecture and data flow of the voice patient diary sample application

5.Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	25 june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	5 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Frontend Dev, Backend Dev
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Backend Dev
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	Backend Dev

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Raghavendra
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Nandini
Sprint-2	Dashboard	USN-6	As a user, I can upload a traffic dataset in CSV format	3	High	Rajak
Sprint-3	Reporting	USN-9	As a user, I can download traffic prediction results in CSV format	2	Medium	Afrid
Sprint-4	ML Model Management	USN-12	As an admin, I can update the ML model used for prediction	5	High	Raghavendra

6. Model Performance Test

Date	25 june 2025
Team ID	LTVIP2025TMID59882
Project Name	**TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**
Maximum Marks	4

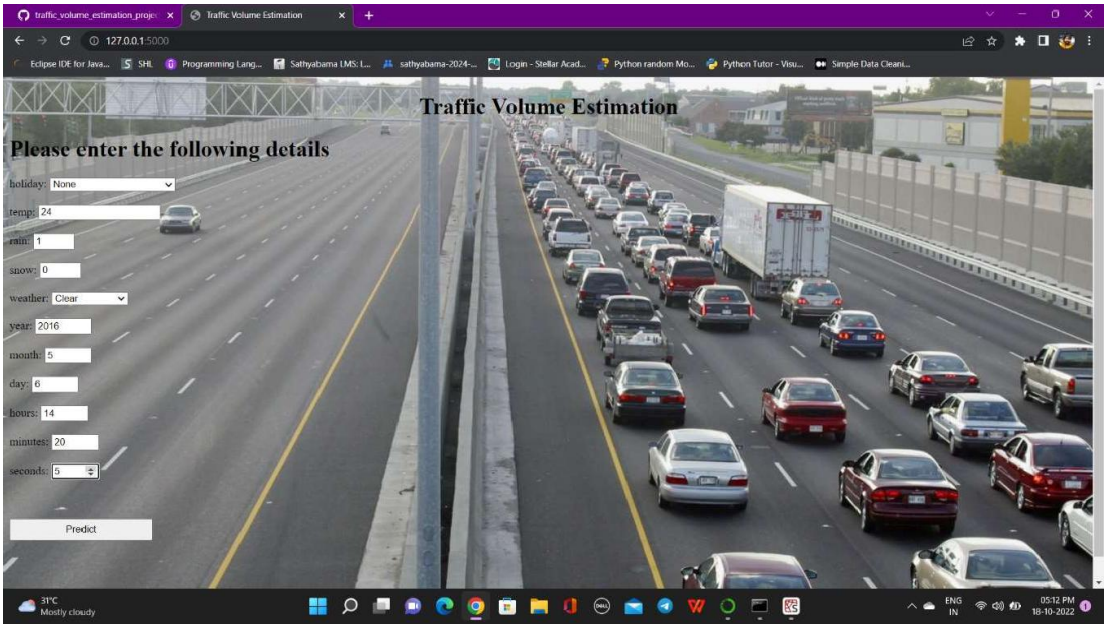
Model Performance Testing:

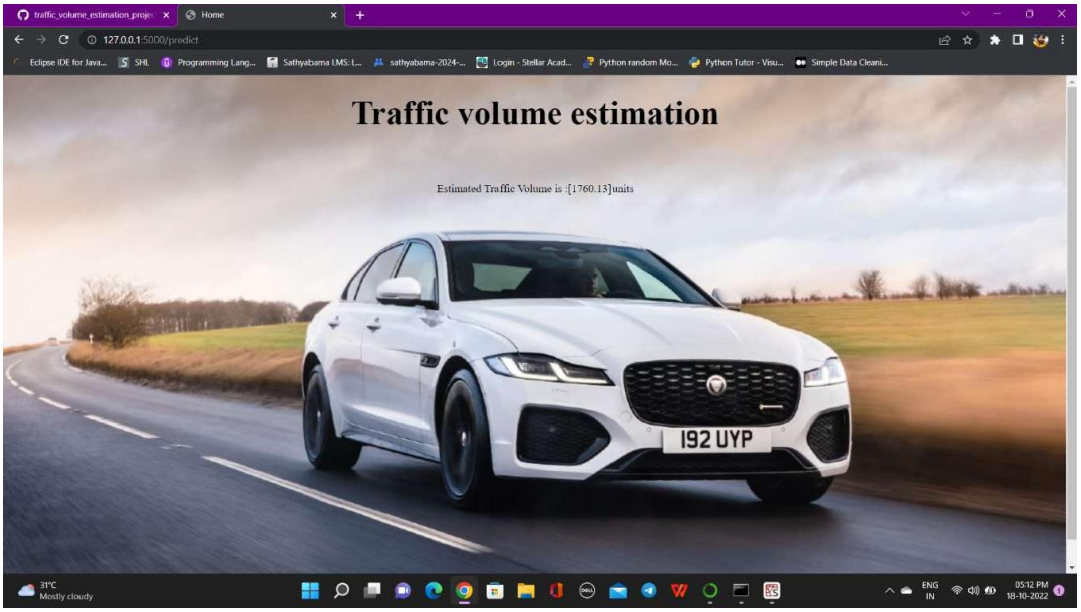
Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	**Model Summary**	**Model Used:** Random Forest Regressor **Features:** Date, Time, Temperature, Day of Week, Holiday Indicator **Target Variable:** Traffic Volume **Libraries Used:** Scikit-learn, Pandas, NumPy	
2.	**Training Accuracy (R^2 Score):** 0.94 **Validation Accuracy (R^2 Score):** 0.88	Training Accuracy - Validation Accuracy -	

7. Results

7.1. outputs:





8. Advantages and Disadvantages

8.1. Advantages

- Accurate traffic volume predictions using machine learning
- Cost-effective and easy to scale
- Helps reduce congestion and emissions
- Supports better planning for authorities
- User-friendly interface for all users

8.2. Disadvantages

- Accuracy depends on data quality
- Not real-time without live data integration
- Limited performance in areas with insufficient data
- Requires technical setup and maintenance

9.Conclusion

TrafficTelligence offers an intelligent, data-driven solution to address the growing problem of urban traffic congestion. By leveraging machine learning, it enables accurate traffic volume prediction using historical data, allowing for smarter planning and smoother traffic flow. The system is cost-effective, scalable, and easy to use, making it suitable for both public use and government authorities.
Overall, this project demonstrates how AI and data science can play a vital role in building smarter and more sustainable cities.

trafficTelligence successfully demonstrates the potential of applying machine learning to solve real-world urban problems like traffic congestion. The system leverages historical data and predictive analytics to forecast traffic volume, which can significantly improve how commuters plan their travel and how city authorities manage traffic flow.

The project focused on building a scalable, accurate, and easy-to-use platform that can work with available datasets and does not rely on expensive hardware or infrastructure. With a modular and extensible design, TrafficTelligence can be adapted for various cities, traffic conditions, or even integrated with IoT devices in the future.

10. FUTURE SCOPE

1. **Real-Time Traffic Prediction**
   Integration with real-time data sources such as traffic APIs, GPS devices, or IoT-based sensors to enable live traffic forecasting.

2. **Weather and Event Integration**
   Incorporate weather conditions, public holidays, and local events into the model to improve prediction accuracy.

3. **Mobile App Deployment**
   Launch a mobile application for commuters to get instant traffic forecasts, suggested alternate routes, and push notifications.

4. **Route Optimization**
   Enhance the system to recommend the best travel routes based on predicted traffic patterns and real-time conditions.

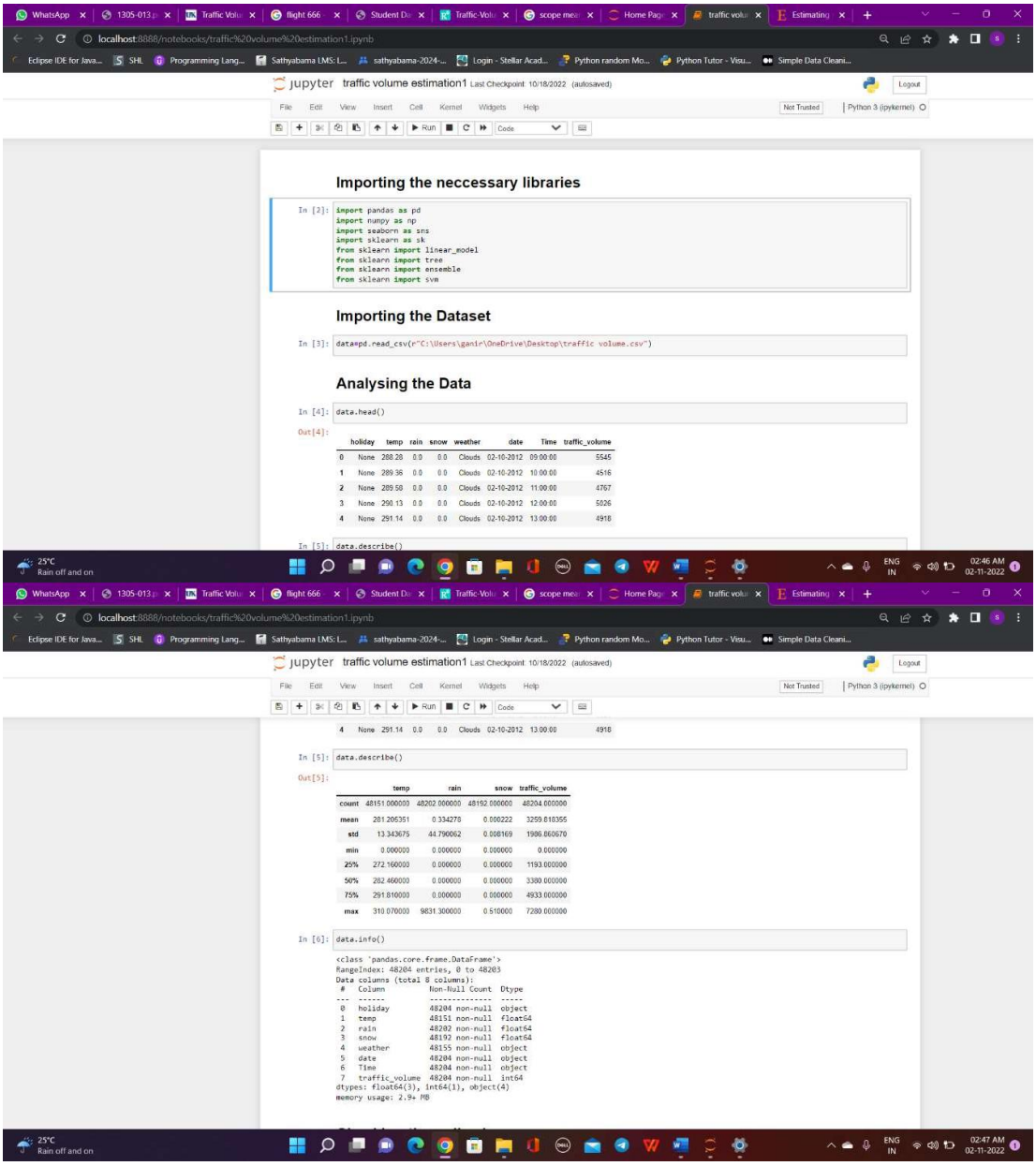
5. **City-Wide Implementation**
   Collaborate with traffic authorities to deploy the system across multiple regions or cities for public use and policy-making.

6. **AI Model Upgrades**
   Experiment with advanced AI techniques like deep learning or reinforcement learning to further improve forecasting accuracy.

7. **Crowdsourced Data**
   Enable users to report traffic incidents, which can be fed into the system for dynamic updates and better prediction learning.

11. Appendix

A.SCREENSHOTS



jupyter traffic volume estimation1 Last Checkpoint: 10/18/2022 (autosaved)

Importing the neccessary libraries

```
In [2]: 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
```

Importing the Dataset

```
In [3]: data=pd.read_csv(r"C:\Users\ganir\OneDrive\Desktop\traffic volume.csv")
```

Analysing the Data

```
In [4]: data.head()
```

Out[4]:

	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

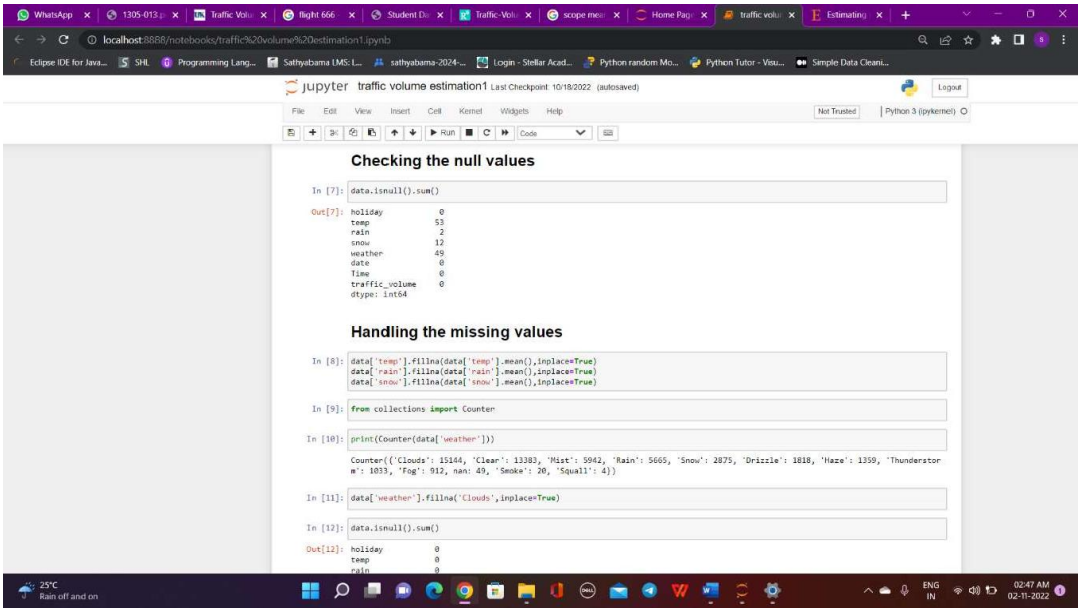
```
In [5]: data.describe()
```

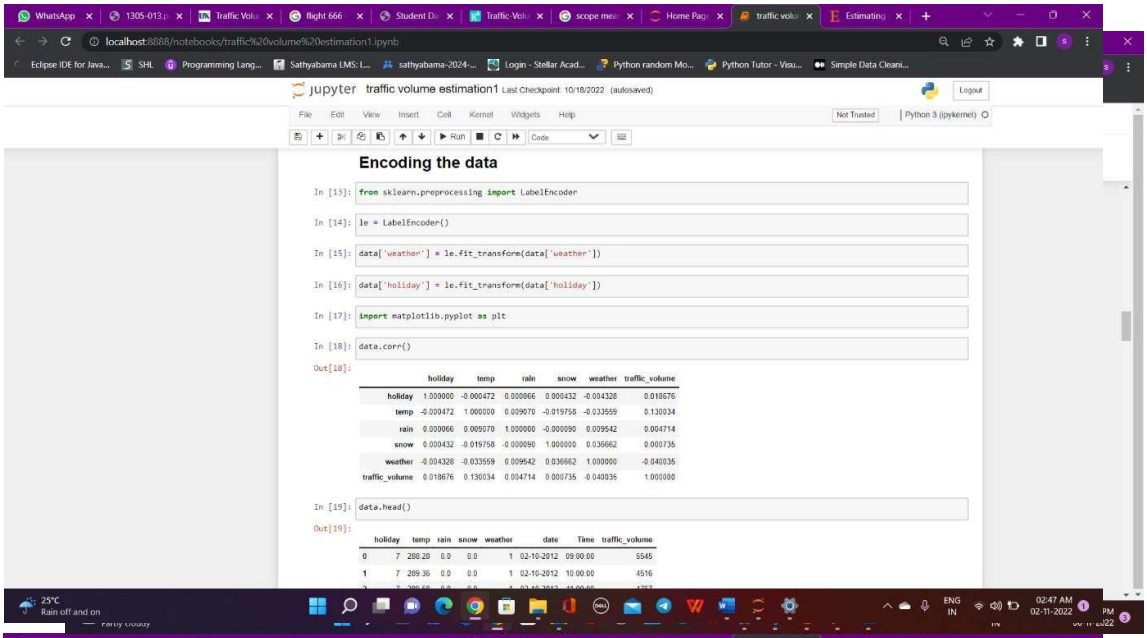
Out[5]:

	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

```
In [6]: 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
```


jupyter traffic volume estimation1 Last Checkpoint: 10/18/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Checking the null values
In [7]:
data.isnull().sum()
Out[7]:
holiday 0
temp 53
rain 2
snow 12
weather 49
date 0
Time 0
traffic_volume 0
dtype: int64
Handling the missing values
In [8]:
data['temp'].fillna(data['temp'].mean(),inplace=True)
data['rain'].fillna(data['rain'].mean(),inplace=True)
data['snow'].fillna(data['snow'].mean(),inplace=True)
In [9]:
from collections import Counter
In [10]:
print(Counter(data['weather']))
Counter({'Clouds': 15144, 'Clear': 13383, 'Mist': 5942, 'Rain': 5665, 'Snow': 2875, 'Drizzle': 1818, 'Haze': 1359, 'Thunderstorm': 1033, 'Fog': 912, nan: 49, 'Smoke': 20, 'Squall': 4})
In [11]:
data['weather'].fillna('Clouds',inplace=True)
In [12]:
data.isnull().sum()
Out[12]:
holiday 0
temp 0
rain 0
25°C
Rain off and on
ENG
IN
02:47 AM
02-11-2022



Encoding the data

```
In [13]: from sklearn.preprocessing import LabelEncoder

In [14]: le = LabelEncoder()

In [15]: data['weather'] = le.fit_transform(data['weather'])

In [16]: data['holiday'] = le.fit_transform(data['holiday'])

In [17]: import matplotlib.pyplot as plt

In [18]: data.corr()
```

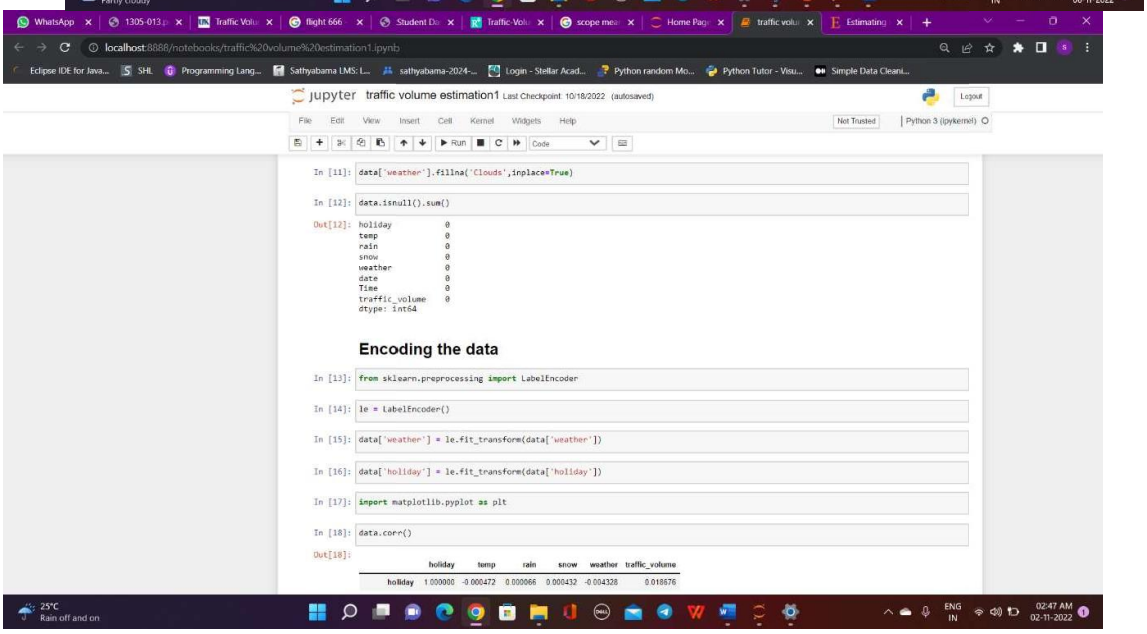
Out[18]:

	holiday	temp	rain	snow	weather	traffic_volume
holiday	1.000000	-0.000472	0.000066	0.000432	-0.004328	0.018676
temp	-0.000472	1.000000	0.009070	-0.019758	-0.033559	0.130034
rain	0.000066	0.009070	1.000000	-0.000090	0.009542	0.004714
snow	0.000432	-0.019758	-0.000090	1.000000	0.036662	0.000735
weather	-0.004328	-0.033559	0.009542	0.036662	1.000000	-0.040035
traffic_volume	0.018676	0.130034	0.004714	0.000735	-0.040035	1.000000

```
In [19]: data.head()
```

Out[19]:

	holiday	temp	rain	snow	weather	date	Time	traffic_volume
0	7	288.28	0.0	0.0	1	02-10-2012	09:00:00	5545
1	7	289.36	0.0	0.0	1	02-10-2012	10:00:00	4516



```
In [11]: data['weather'].fillna('Clouds',inplace=True)

In [12]: data.isnull().sum()
```

Out[12]:

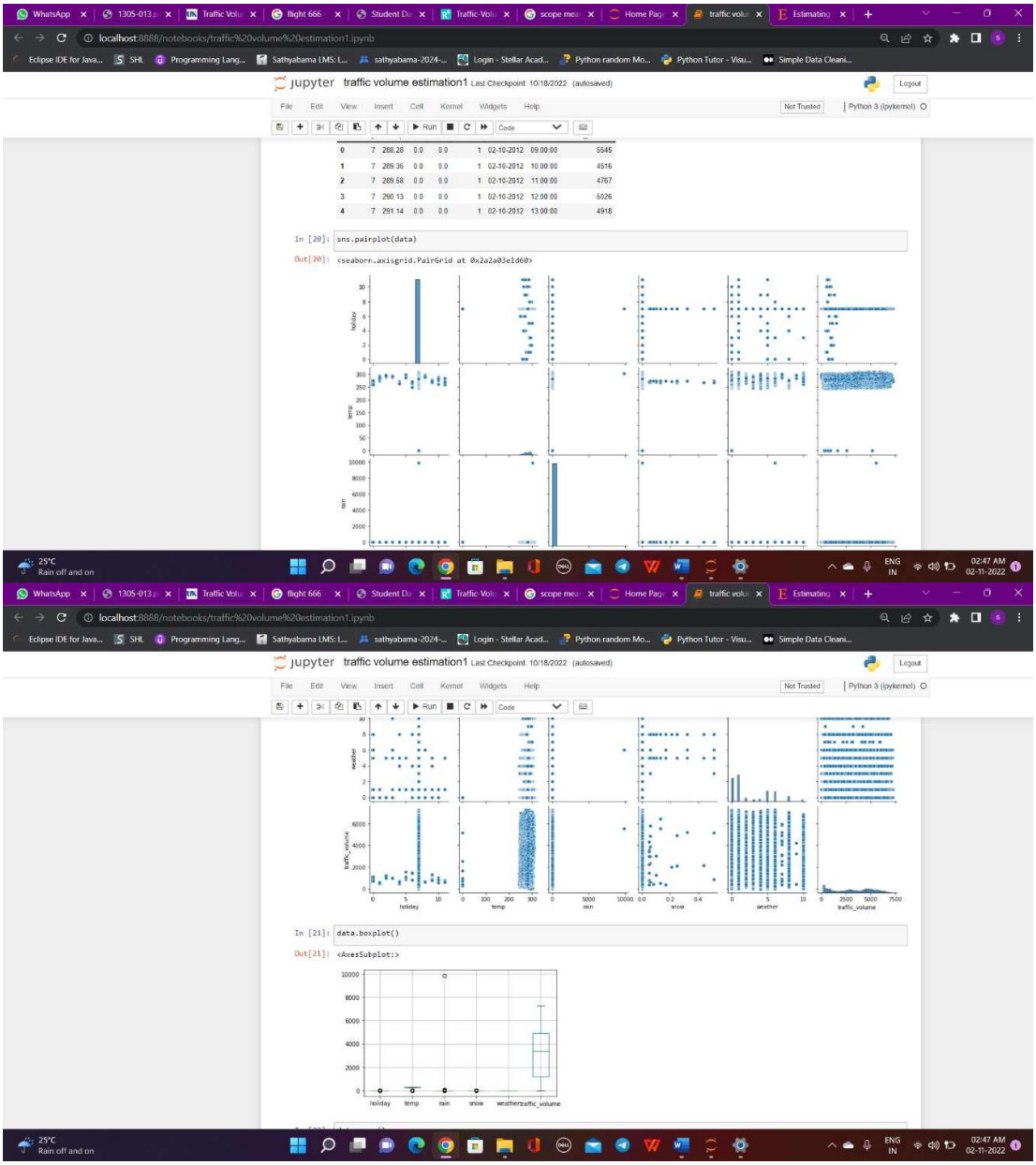
```
holiday           0
temp              0
rain              0
snow              0
weather           0
date              0
Time              0
traffic_volume    0
dtype: int64
```

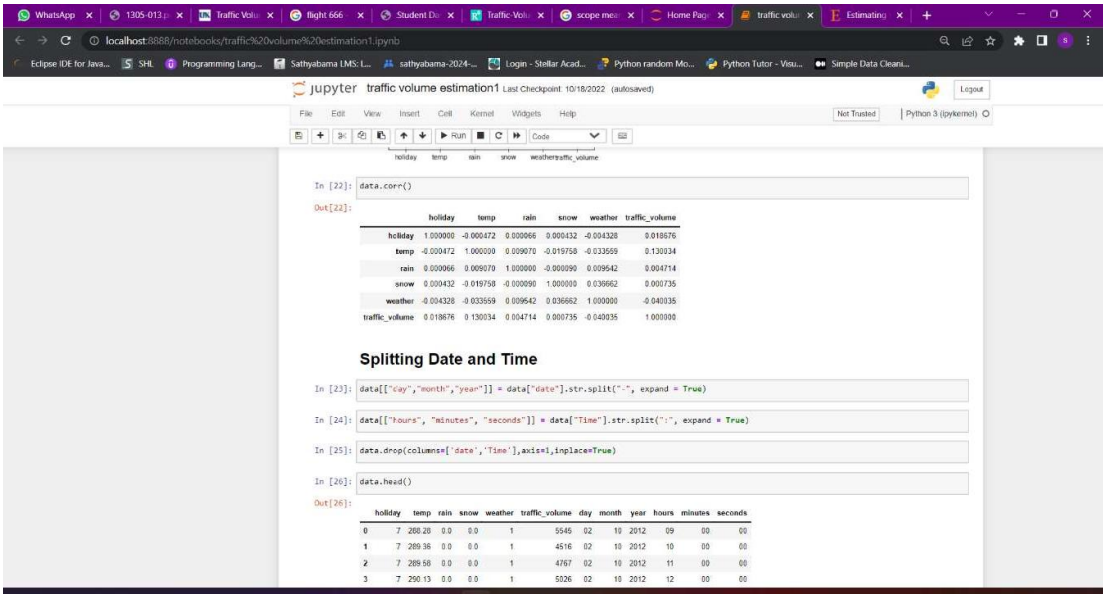
Encoding the data

```
In [13]: from sklearn.preprocessing import LabelEncoder

In [14]: le = LabelEncoder()

In [15]: data['weather'] = le.fit_transform(data['weather'])

In [16]: data['holiday'] = le.fit_transform(data['holiday'])

In [17]: import matplotlib.pyplot as plt

In [18]: data.corr()
```

Out[18]:

	holiday	temp	rain	snow	weather	traffic_volume
holiday	1.000000	-0.000472	0.000066	0.000432	-0.004328	0.018676


jupyter traffic volume estimation1 Last Checkpoint: 10/18/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
0 7 288.28 0.0 0.0 1 02-10-2012 09:00:00 5545
1 7 289.36 0.0 0.0 1 02-10-2012 10:00:00 4516
2 7 289.58 0.0 0.0 1 02-10-2012 11:00:00 4767
3 7 290.13 0.0 0.0 1 02-10-2012 12:00:00 5026
4 7 291.14 0.0 0.0 1 02-10-2012 13:00:00 4918
In [20]: sns.pairplot(data)
Out[20]: <seaborn.axisgrid.PairGrid at 0x2a2a03e1d60>
holiday
temp
rain
In [21]: data.boxplot()
Out[21]: <AxesSubplot:>
holiday
temp
rain
snow
weather
traffic_volume
25°C
Rain off and on
ENG
IN
02:47 AM
02-11-2022



jupyter traffic volume estimation1 Last Checkpoint: 10/18/2022 (autosaved)

holiday temp rain snow weathertraffic_volume

In [22]:
```
data.corr()
```

Out[22]:

	holiday	temp	rain	snow	weather	traffic_volume
holiday	1.000000	-0.000472	0.000066	0.000432	-0.004328	0.018676
temp	-0.000472	1.000000	0.009070	-0.019758	-0.033559	0.130034
rain	0.000066	0.009070	1.000000	-0.000090	0.009642	0.004714
snow	0.000432	-0.019758	-0.000090	1.000000	0.036662	0.000735
weather	-0.004328	-0.033559	0.009642	0.036662	1.000000	-0.040035
traffic_volume	0.018676	0.130034	0.004714	0.000735	-0.040035	1.000000

Splitting Date and Time

In [23]:
```
data[["day","month","year"]] = data["date"].str.split("-", expand = True)
```

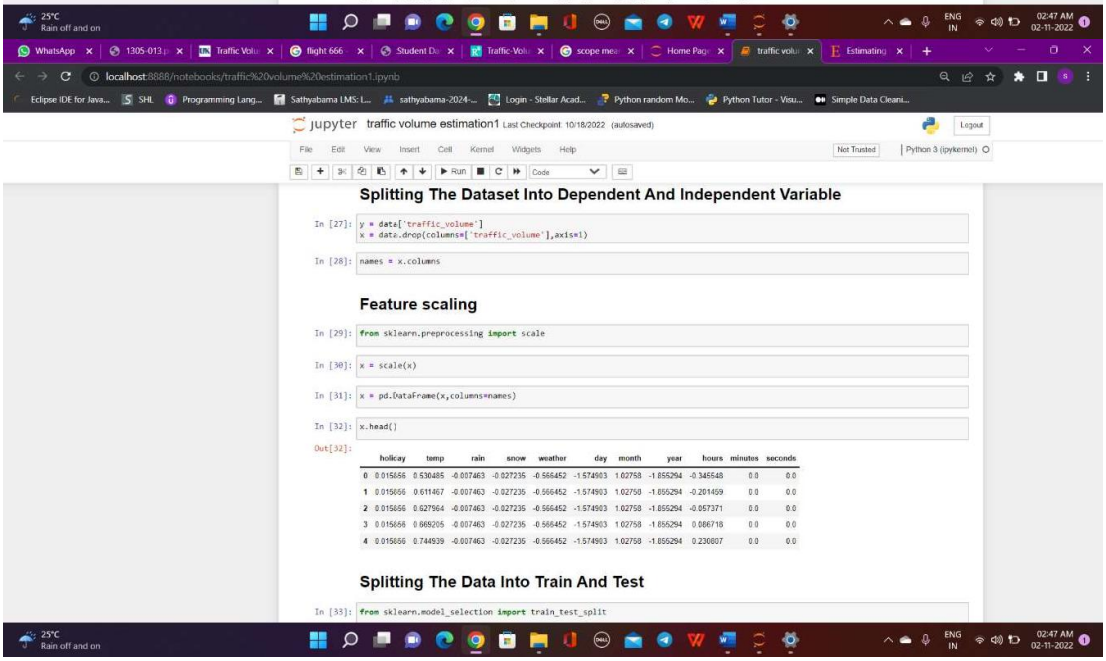
In [24]:
```
data[["hours", "minutes", "seconds"]] = data["Time"].str.split(":", expand = True)
```

In [25]:
```
data.drop(columns=['date','Time'],axis=1,inplace=True)
```

In [26]:
```
data.head()
```

Out[26]:

	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



Splitting The Dataset Into Dependent And Independent Variable

In [27]:
```
y = data['traffic_volume']
x = data.drop(columns=['traffic_volume'],axis=1)
```

In [28]:
```
names = x.columns
```

Feature scaling

In [29]:
```
from sklearn.preprocessing import scale
```

In [30]:
```
x = scale(x)
```

In [31]:
```
x = pd.DataFrame(x,columns=names)
```

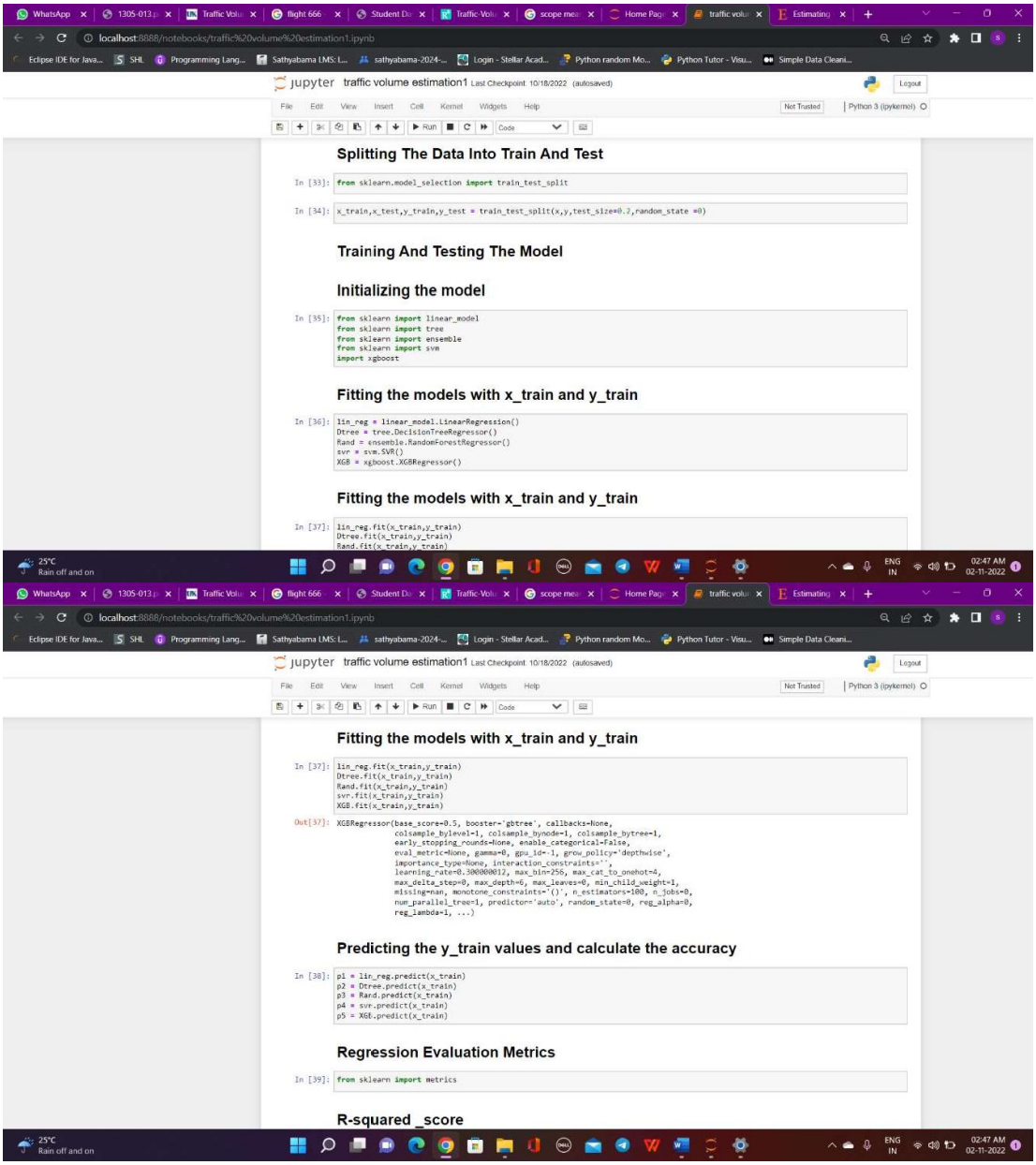
In [32]:
```
x.head()
```

Out[32]:

	holiday	temp	rain	snow	weather	day	month	year	hours	minutes	seconds
0	0.015656	0.530485	-0.007463	-0.027235	-0.566452	-1.574903	1.02758	-1.855294	-0.345548	0.0	0.0
1	0.015656	0.611467	-0.007463	-0.027235	-0.566452	-1.574903	1.02758	-1.855294	-0.201459	0.0	0.0
2	0.015656	0.627964	-0.007463	-0.027235	-0.566452	-1.574903	1.02758	-1.855294	-0.057371	0.0	0.0
3	0.015656	0.669205	-0.007463	-0.027235	-0.566452	-1.574903	1.02758	-1.855294	0.086718	0.0	0.0
4	0.015656	0.744939	-0.007463	-0.027235	-0.566452	-1.574903	1.02758	-1.855294	0.230807	0.0	0.0

Splitting The Data Into Train And Test

In [33]:
```
from sklearn.model_selection import train_test_split
```



Splitting The Data Into Train And Test

```
In [33]: from sklearn.model_selection import train_test_split
```

```
In [34]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state =0)
```

Training And Testing The Model

Initializing the model

```
In [35]: from sklearn import linear_model
         from sklearn import tree
         from sklearn import ensemble
         from sklearn import svm
         import xgboost
```

Fitting the models with x_train and y_train

```
In [36]: lin_reg = linear_model.LinearRegression()
         Dtree = tree.DecisionTreeRegressor()
         Rand = ensemble.RandomForestRegressor()
         svr = svm.SVR()
         XGB = xgboost.XGBRegressor()
```

Fitting the models with x_train and y_train

```
In [37]: lin_reg.fit(x_train,y_train)
         Dtree.fit(x_train,y_train)
         Rand.fit(x_train,y_train)
```


Fitting the models with x_train and y_train

```
In [37]: 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)
```

```
Out[37]: XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
                      colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
                      early_stopping_rounds=None, enable_categorical=False,
                      eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
                      importance_type=None, interaction_constraints='',
                      learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
                      max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
                      missing=nan, monotone_constraints='()', n_estimators=100, n_jobs=0,
                      num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
                      reg_lambda=1, ...)
```

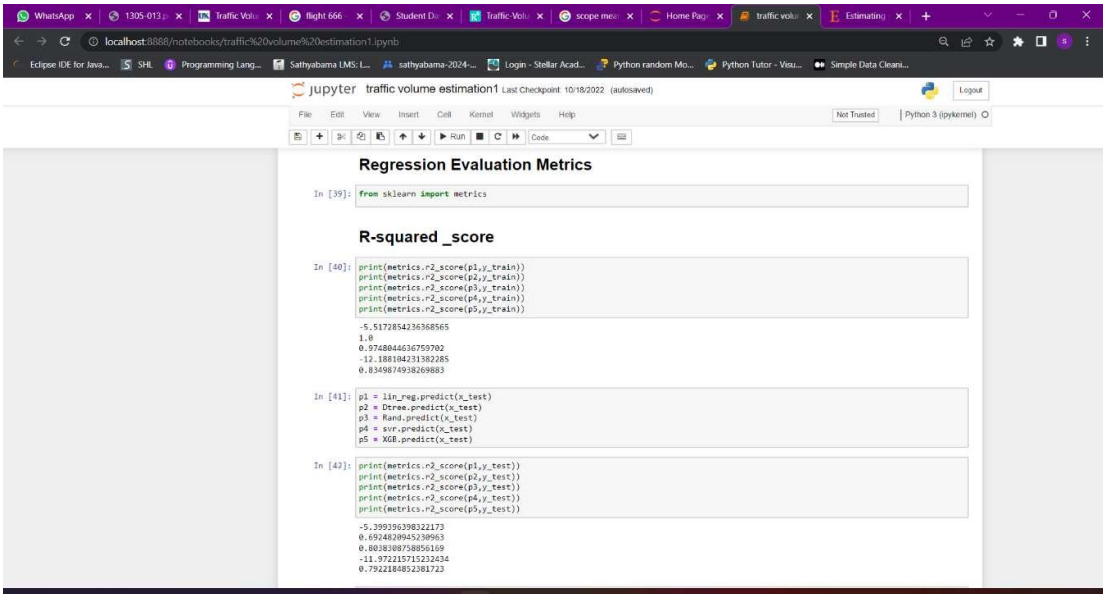
Predicting the y_train values and calculate the accuracy

```
In [38]: 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)
```

Regression Evaluation Metrics

```
In [39]: from sklearn import metrics
```

R-squared _score



jupyter traffic volume estimation1 Last Checkpoint: 10/18/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

Regression Evaluation Metrics

In [39]:
```
from sklearn import metrics
```

R-squared _score

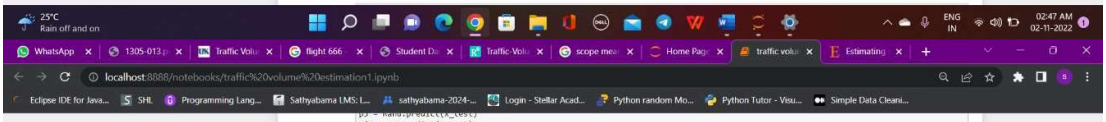
In [40]:
```
print(metrics.r2_score(p1,y_train))
print(metrics.r2_score(p2,y_train))
print(metrics.r2_score(p3,y_train))
print(metrics.r2_score(p4,y_train))
print(metrics.r2_score(p5,y_train))
```

```
-5.5172854236368565
1.0
0.9748044636759702
-12.188104231382285
0.8349874938269883
```

In [41]:
```
p1 = lin_reg.predict(x_test)
p2 = Dtree.predict(x_test)
p3 = Rand.predict(x_test)
p4 = svr.predict(x_test)
p5 = XGB.predict(x_test)
```

In [42]:
```
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))
```

```
-5.399396398322173
0.6924820945230963
0.8038308758856169
-11.972215715232434
0.7922184852381723
```



In [42]:
```
p4 = svr.predict(x_test)
p5 = XGB.predict(x_test)
```

In [42]:
```
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))
```

```
-5.399396398322173
0.6924820945230963
0.8038308758856169
-11.972215715232434
0.7922184852381723
```


RMSE –Root Mean Square Error

In [42]:
```
MSE = metrics.mean_squared_error(p3,y_test)
```

In [43]:
```
np.sqrt(MSE)
```

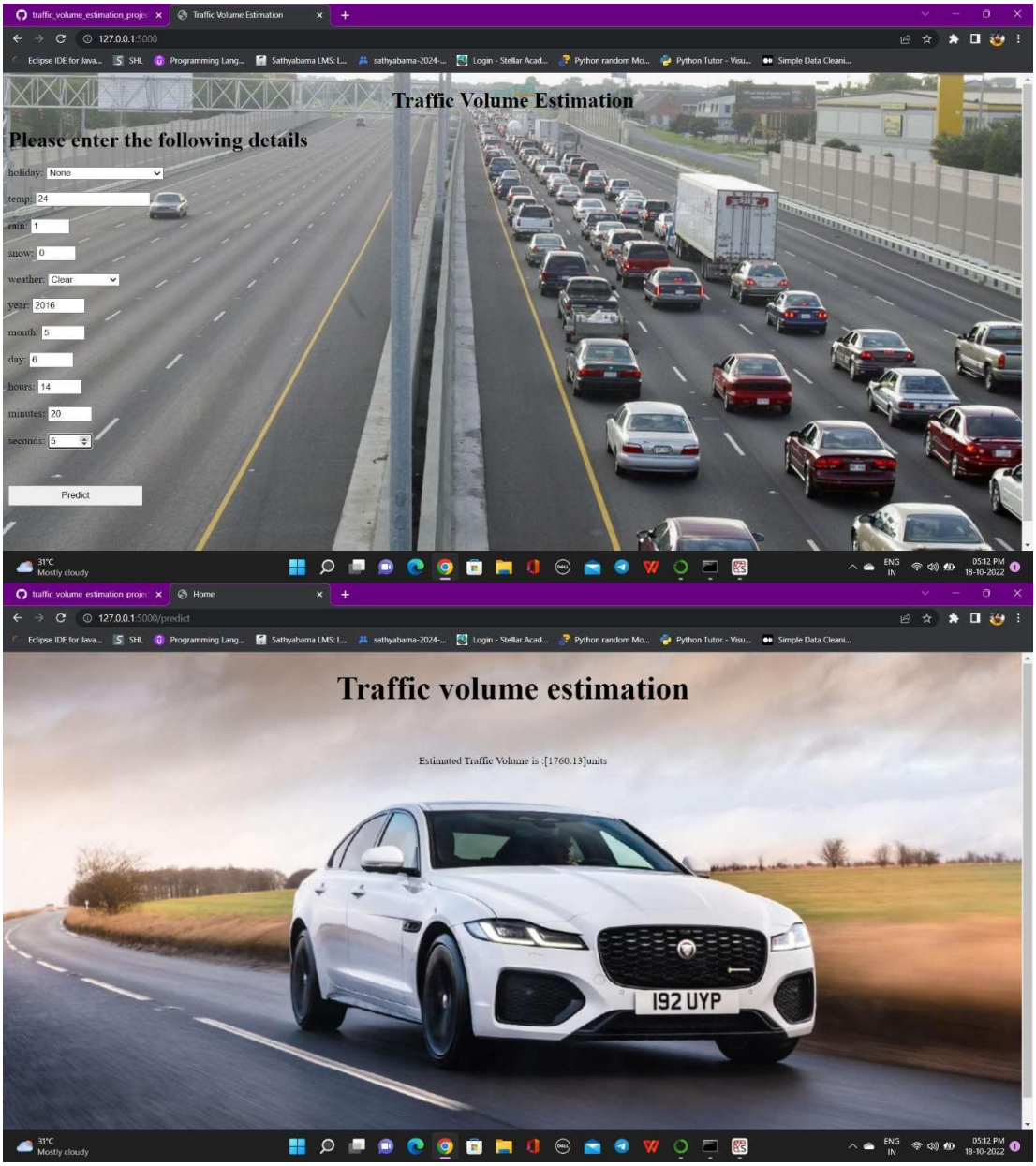
Out[43]:
```
794.1526330414208
```

Saving the Model

In [46]:
```
import pickle
```

In [47]:
```
pickle.dump(Rand,open("model.pkl",'wb'))
pickle.dump(le,open("encoder.pkl",'wb'))
```

In [ ]:
```

```


Traffic Volume Estimation
Please enter the following details
holiday: None
temp: 24
rain: 1
snow: 0
weather: Clear
year: 2016
month: 5
day: 6
hours: 14
minutes: 20
seconds: 5
Predict
Traffic volume estimation
Estimated Traffic Volume is :[1760.13]units
192 UYP

B. SOURCE CODE

- **PYTHON CODE USED IN JUPYTER NOTEBOOK**

```
# Importing the neccessary 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


# Importing the Dataset

data=pd.read_csv(r"C:\Users\ganir\OneDrive\Desktop\traffic volume.csv")


# Analysing the Data

data.head()


data.describe()


data.info()


# Checking the null values
```

```
data.isnull().sum()

# Handling the missing values
data['temp'].fillna(data['temp'].mean(),inplace=True)
data['rain'].fillna(data['rain'].mean(),inplace=True)
data['snow'].fillna(data['snow'].mean(),inplace=True)

from collections import Counter

print(Counter(data['weather']))

data['weather'].fillna('Clouds',inplace=True)

data.isnull().sum()

# Encoding the data
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

data['weather'] = le.fit_transform(data['weather'])

data['holiday'] = le.fit_transform(data['holiday'])
```

```
import matplotlib.pyplot as plt


data.corr()

sns.heatmap(data.corr())


data.head()


sns.pairplot(data)


data.boxplot()


data.corr()


# Splitting Date and Time

data[["day","month","year"]] = data["date"].str.split("-", expand = True)


data[["hours", "minutes", "seconds"]] = data["Time"].str.split(":", expand = True)


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


data.head()
```

```
# Splitting The Dataset Into Dependent And Independent Variable

y = data['traffic_volume']

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


names = x.columns


# Feature scaling

from sklearn.preprocessing import scale


x = scale(x)


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


x.head()


# Splitting The Data Into Train And Test

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)


# Training And Testing The Model

# Initializing the model
```

```
from sklearn import linear_model

from sklearn import tree

from sklearn import ensemble

from sklearn import svm

import xgboost


# Fitting the models with x_train and y_train

lin_reg = linear_model.LinearRegression()

Dtree = tree.DecisionTreeRegressor()

Rand = ensemble.RandomForestRegressor()

svr = svm.SVR()

XGB = xgboost.XGBRegressor()


# Fitting 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)


# Predicting the y_train values and calculate the accuracy

p1 = lin_reg.predict(x_train)

p2 = Dtree.predict(x_train)
```

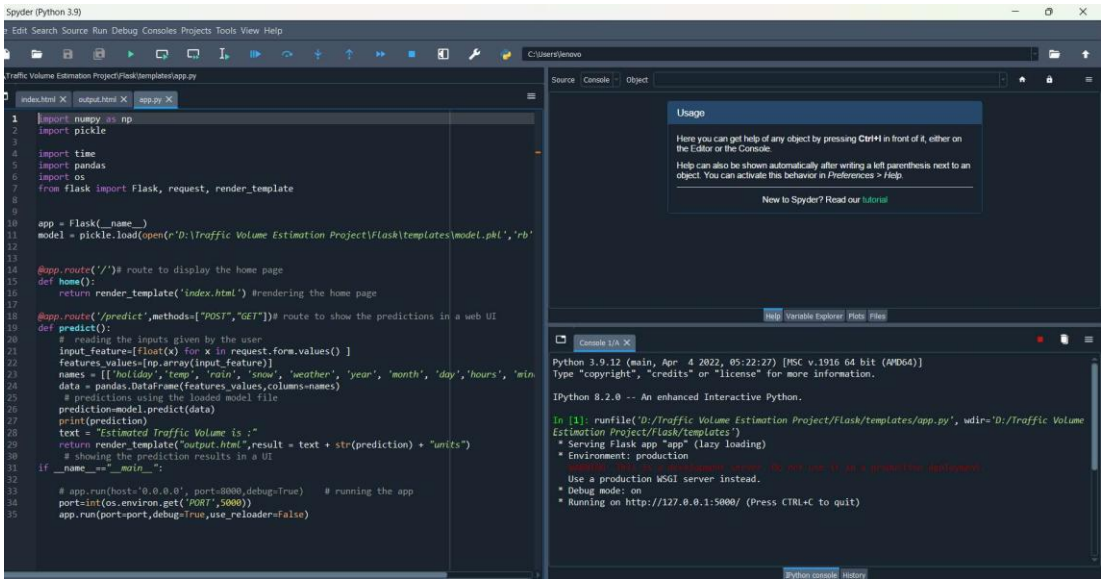
```
p3 = Rand.predict(x_train)

p4 = svr.predict(x_train)

p5 = XGB.predict(x_train)


# Regression Evaluation Metrics

from sklearn import metrics


#  R-squared _score

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))


p1 = lin_reg.predict(x_test)

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))


# RMSE –Root Mean Square Error

MSE = metrics.mean_squared_error(p3,y_test)


np.sqrt(MSE)


# Saving the Model

import pickle


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

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

- **PYTHON CODE USED FOR APP BUILDING**

```
import numpy as np
import pickle

import time
import pandas
import os
from flask import Flask, request, render_template


app = Flask(__name__,template_folder='Template')
model = pickle.load(open(r"D:\Traffic volume estimation
project\flask\Template\model.pkl",'rb'))
```

```
@app.route('/')# route to display the home page
def index():
    return render_template('index.html') #rendering the home page

@app.route('/predict',methods=["POST","GET"])# route to show the
predictions in a web UI
def predict():
    #  reading the inputs given by the user
input_feature=[float(x) for x in request.form.values() ]
features_values=[np.array(input_feature)]
    names = [['holiday','temp', 'rain', 'snow', 'weather', 'year', 'month',
'day','hours', 'minutes', 'seconds']]
    data = pandas.DataFrame(features_values,columns=names)
    # predictions using the loaded model file
    prediction=model.predict(data)
    print(prediction)
    text = "Estimated Traffic Volume is :"
    return render_template("output.html",result = text + str(prediction) + "units")
     # showing the prediction results in a UI
if __name__=="__main__":

    # app.run(host='0.0.0.0', port=8000,debug=True)      # running the app
    port=int(os.environ.get('PORT',5000))
app.run(port=port,debug=True,use_reloader=False)
```



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 .

- **HTML CODES USED**
  - **Index.html**

```
<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<title>Traffic Volume Estimation</title>

</head>


<body background="https://cdn.vox-
cdn.com/thumbor/voARJfEKvTp6iMSzW3ExPn06TDM=/0x78:3000x1766/
1600x900/cdn.vox-
cdn.com/uploads/chorus_image/image/44219366/72499026.0.0.jpg"
text="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>Washingtons 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>  <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>  <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>
```


<label>seconds:</label>

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

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

</form>

{{ prediction_text }}

<imgsrc="{{url_3}}" alt="Submit Form" height="180" width="233" onerror="this.style.display='none'"/>

<imgsrc="{{url_1}}" alt="Submit Form" height="180" width="233" onerror="this.style.display='none'"/>

<imgsrc="{{url_4}}" alt="Submit Form" height="180" width="233" onerror="this.style.display='none'"/>

<imgsrc="{{url_2}}" alt="Submit Form" height="150" width="711" onerror="this.style.display='none'"/>

</body>

</html>

- **Output.html**

```
<!DOCTYPE html>
<html>
<head>
<title>Home</title>
<style>
body
{
    background-image: url("https://stat.overdrive.in/wp-content/uploads/2021/10/2021-jaguar-xf-facelift-india-01.jpg");
    background-size: cover;
}
.pd{
padding-bottom:45%;}
}
</style>
</head>
<body>

<br>
```

```
<center><b class="pd"><font color="black" size="15" font-
family="Comic Sans MS" >Traffic volume
estimation</font></b></center><br><br>
<div>
<br>
<center>
<p><font color="black"> {{result}} </p>
</center>
</div>
</body>
</html>
```

C.DATASET LINK
:https://drive.google.com/file/d/1PNgGMYpF7wSSIzJAiyAuEkhN9lx6vlpf/view?usp=drivesdk

D.GITHUB LINK :

https://github.com/nandini-yarabham/TrafficTelligence-Advanced-Traffic-Volume-Estimation-with-Machine-Learning

E.PROJECT DEMO LINK
:https://drive.google.com/file/d/1P7mf_FMcM5d5jAuKlahP4H3t6zG0S6M3/view?usp=drivesdk