PROJECT REPORT

Traffic Intelligence: Advanced Traffic Volume Estimation

Name	Team ID
VISWA SAI V	NM2023TMID06574
SRI HARSHA P	
KOTHAPALLI MADHANMOHAN REDDY	
PERLA PRAVEEN KUMAR REDDY	

S.NO	CONTENTS
1.	**INTRODUCTION**
	1.1 Project Overview
	1.2 Purpose
2.	**IDEATION & PROPOSED SOLUTION**
	2.1 Problem Statement Definition
	2.2 Empathy Map Canvas
	2.3 Ideation & Brainstorming
	2.4 Proposed Solution
3.	**REQUIREMENT ANALYSIS**
	3.1 Functional requirement
	3.2 Non-Functional requirements
4.	**PROJECT DESIGN**
	4.1 Data Flow Diagrams
	4.2 Solution & Technical Architecture
	4.3 User Stories
5.	**CODING & SOLUTIONS**
	5.1 Feature 1
	5.2 Feature 2
	5.3 Database Schema
6.	**RESULTS**
	6.1 Performance Metrics
7.	**ADVANTAGES & DISADVANTAGES**
8.	**CONCLUSION**
9.	**FUTURE SCOPE**
10.	**APPENDIX**
	10.1 Source Code
	10.2GitHub & Project Video Demo Link

INTRODUCTION

1.1 Project overview:

Traffic problem is one of the major problem now a days, In the increase in no of vehicles and non –usage of public transport leading to traffic related issues, Making a eye on count of traffic at each level enables the government to take the further decisions such as building new roads, increasing infrastructure ,developing mutli-channel connectivity .To address such problems to tracking the vehicle count in each and every place AI-ML has given a solution to such kind of traffic related issues, which are able to measure the volume of traffic, identify the violations of traffic rules etc.ML models could give early alerts of severe traffic to help prevent issues related to traffic problems. Hence, there is needs to develop ML algorithms capable in predicting Traffic volume with acceptable level of precision and in reducing the error in the dataset of the projected Traffic volume from model with the expected observable Traffic volume**.**

1.2 Purpose:

The purpose of traffic volume estimation using machine learning is to accurately predict and analyse the flow of traffic in a given area or on a specific road segment. This information is valuable for various applications, including:

Transportation planning: Traffic volume estimation helps in designing and optimizing transportation systems by identifying congested areas, determining the need for additional infrastructure, and improving traffic flow efficiency.

Traffic management: Accurate traffic volume predictions enable better management of traffic congestion and facilitate the implementation of effective traffic control measures, such as signal timing adjustments and dynamic route guidance system

Infrastructure maintenance: By understanding traffic patterns and volume, authorities can plan maintenance activities on roads and bridges more effectively, reducing disruptions and ensuring the safety of commuters.

Intelligent transportation systems (ITS): Traffic volume estimation contributes to the development of ITS applications, such as real-time traffic monitoring, incident detection, and adaptive traffic signal control, enhancing overall transportation efficiency and safety.

Urban planning and development: Estimating traffic volume helps urban planners make informed decisions about land use, zoning, and the placement of new developments, ensuring proper integration with the existing transportation network.

Environmental impact assessment: Accurate traffic volume estimation supports environmental studies and assessments by evaluating the potential impact of traffic on air quality, noise pollution, and carbon emissions.

By leveraging machine learning algorithms, traffic volume estimation can harness historical traffic data, weather conditions, and other relevant factors to create predictive models that provide reliable and timely traffic volume information, benefiting transportation stakeholders and the general public alike.

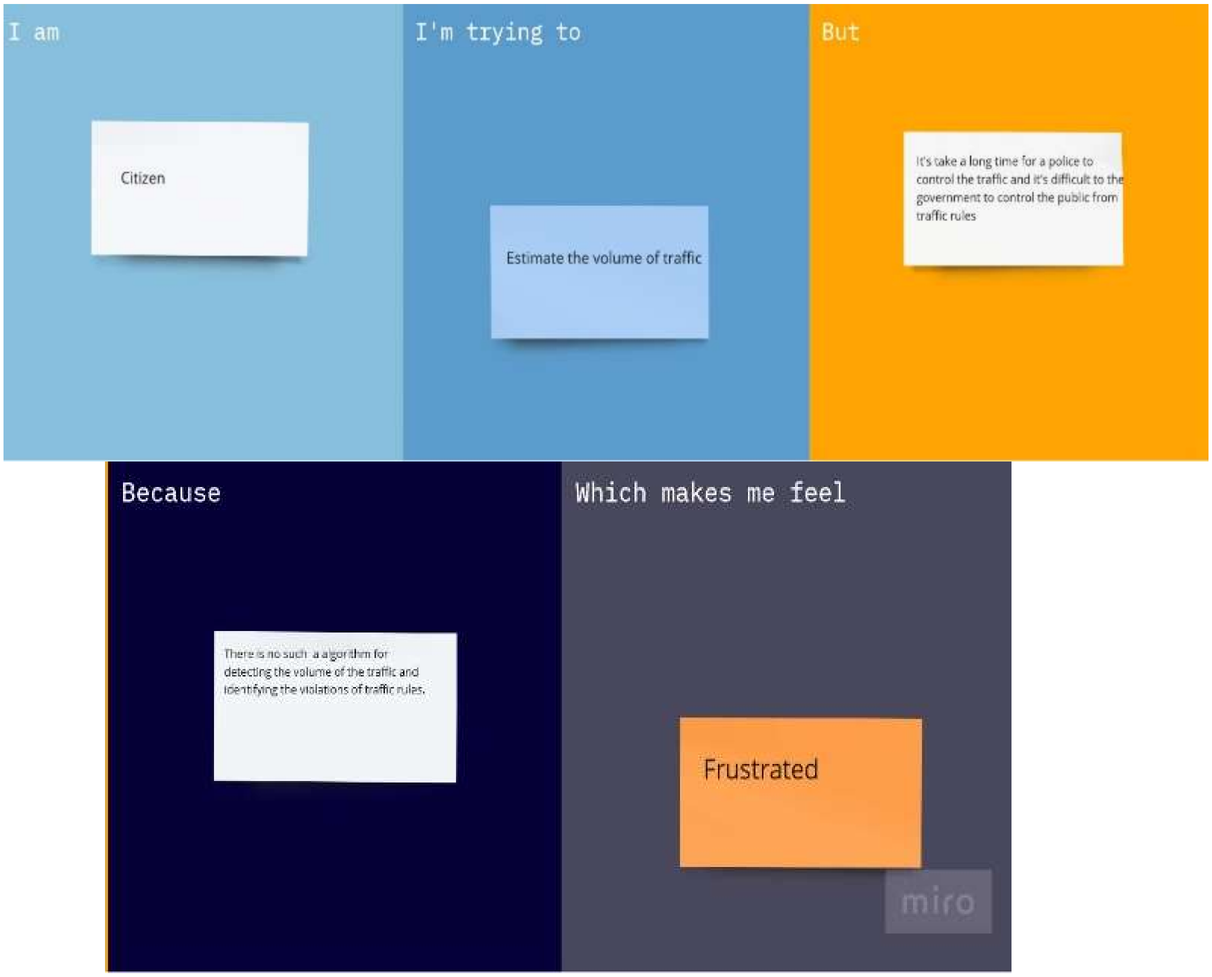
IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition:

Customer Problem Statement:

Traffic problem is one of the major problem now a days, In the increase in no of vehicles and non –usage of public transport leading to traffic related issues, Making an eye on count of traffic at each level enables the government to take the further decisions such as building new roads, increasing infrastructure, developing mutli-channel connectivity . Customer are getting frustrated due to heavy traffic and they getting late for their works and they getting angry due to violation off traffic rules by the other citizens.

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

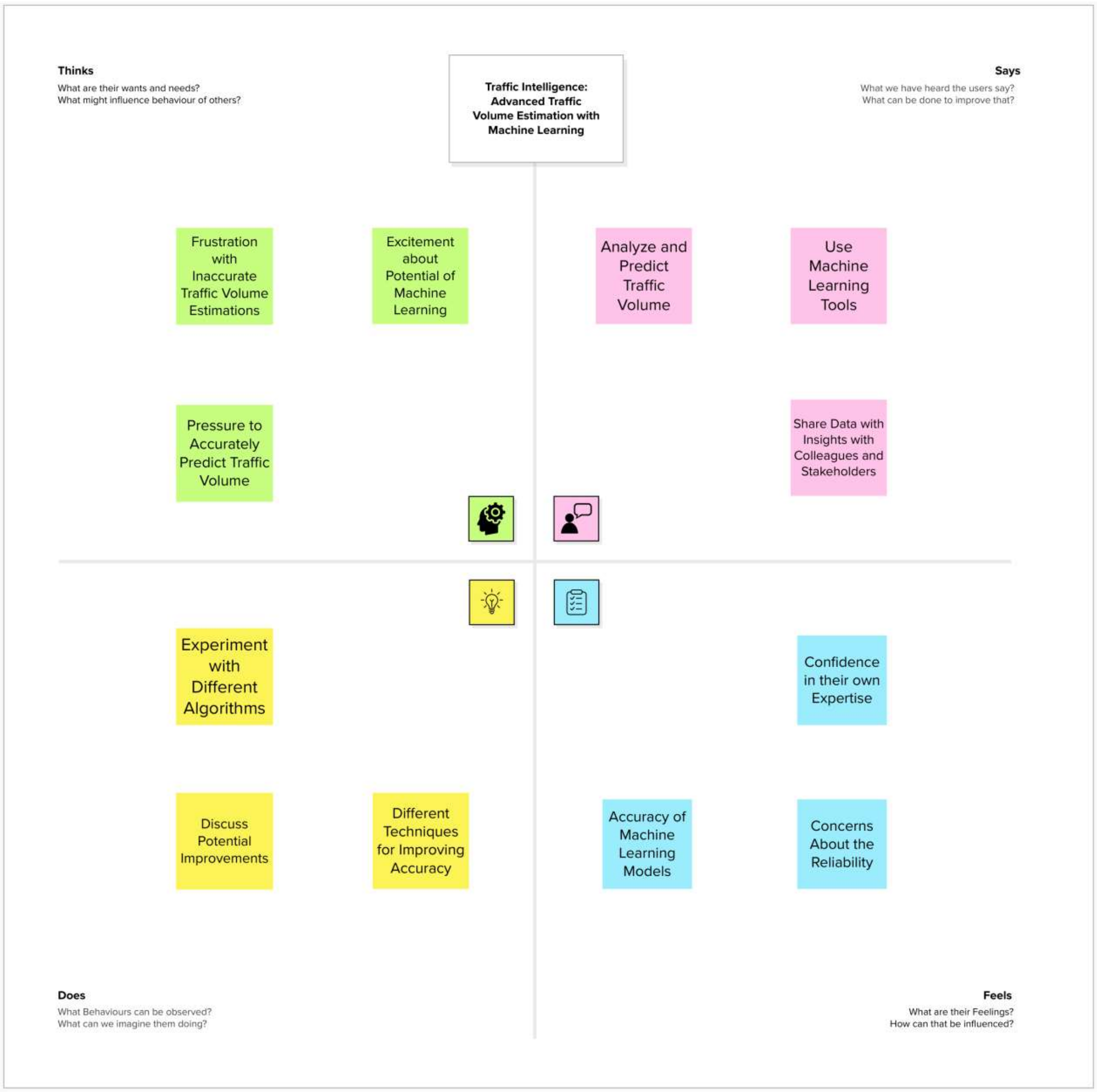

I am
Citizen
I'm trying to
Estimate the volume of traffic
But
It's take a long time for a police to control the traffic and it's difficult to the government to control the public from traffic rules
Because
There is no such a algorithm for detecting the volume of the traffic and identifying the violations of traffic rules.
Which makes me feel
Frustrated
miro

Problem Statement (PS)	I am (Citizen)	I'm trying to	But	Because	Which makes me feel
PS-1	Citizen	Estimate the volume of traffic	It's take a long time for a police to control the traffic and it's difficult to the government to control the public to follow traffic rules	There is no such a algorithm for detecting the volume of the traffic and identifying the violations of traffic rules.	Frustrated
PS-2	Citizen	Developing the Algorithm for traffic volume and identifying the violation of traffic rules using ML	Arranging the algorithm developed system in the heavy traffic places for better estimation of volume of a traffic.	This algorithm detect the traffic rules violations	Frustrated

2.2 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 challenge.

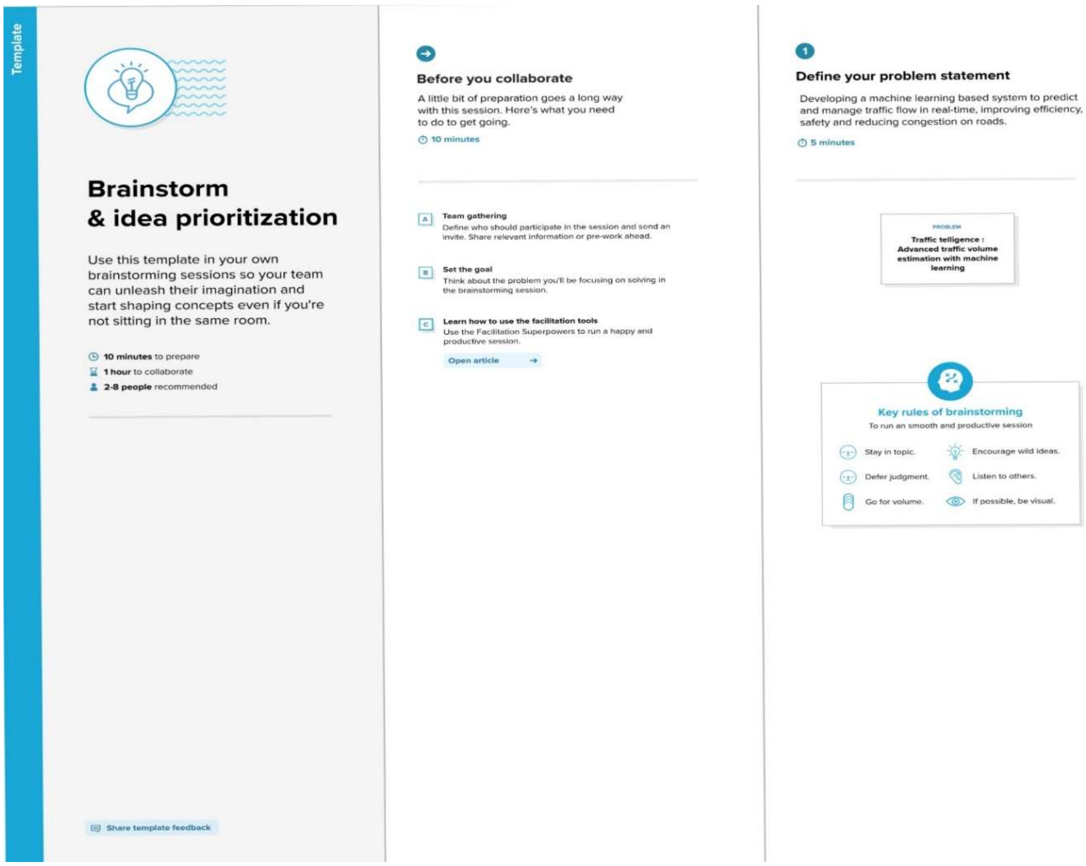
Traffic Intelligence: Advanced Traffic Volume Estimation:



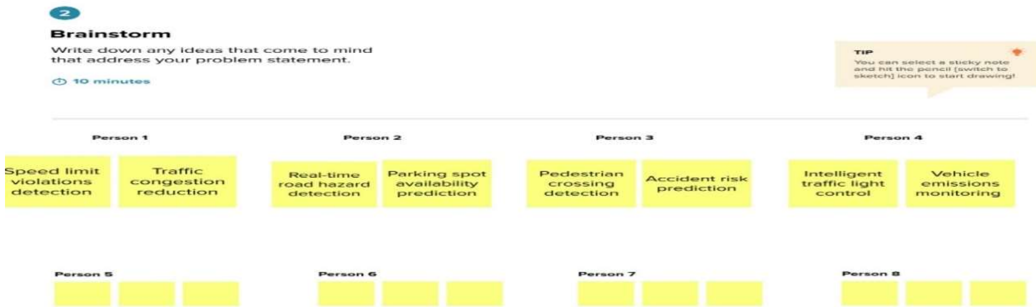
2.3 Ideation & Brainstorming:

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.

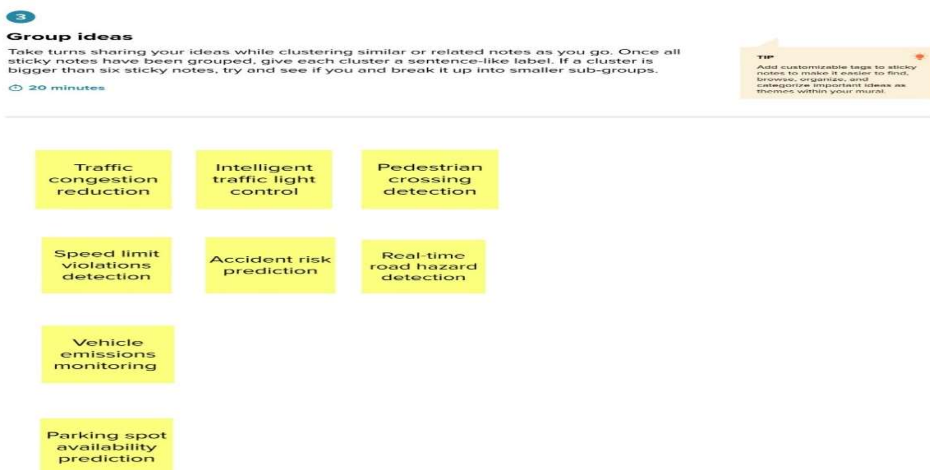
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



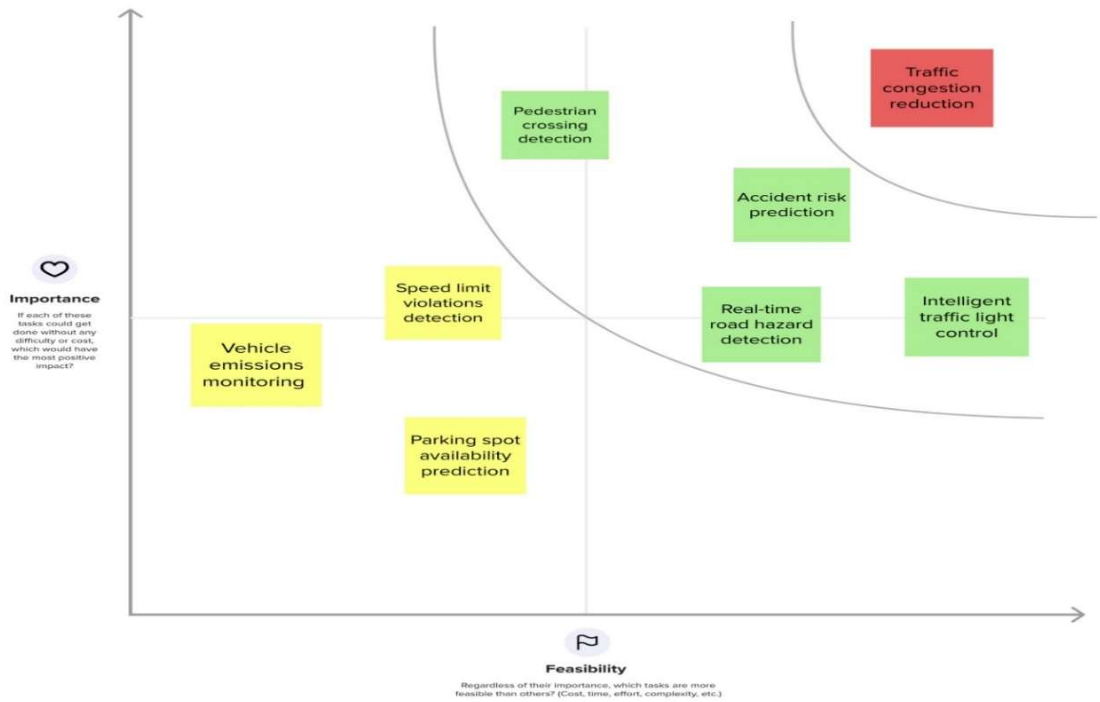
4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

> **TIP**
>
> Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.



2.4 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Traffic volume estimation is a crucial component of transportation planning and management. Accurate and reliable estimation of traffic volume is essential for optimizing traffic flow, reducing congestion, and improving safety on roadways.
2.	Idea / Solution description	With the help of Machine Learning Techniques, we'll be able improve the efficiency of traffic management on the road, as well as improve the safety on the road.
3.	Novelty / Uniqueness	With the use of Regression Algorithms, we can create a unique System that can be used to find out the relationship between a single dependent variable (target variable) dependent on several independent ones, giving us a better and more accurate Estimation of the Volume of Traffic.
4.	Social Impact / Customer Satisfaction	Our projects serve two major services. First one, improved safety of drivers on the road, which is the most important thing and the Second being the reduction of congestion on the road as the number of vehicles on the road is increasing every day.
5.	Business Model (Revenue Model)	Traffic volume estimation can help logistics companies optimize their delivery routes and schedules, reducing transportation costs and improving delivery times as well as it can help retailers determine the best location for their stores, as well as the best times to stock their inventory and offer promotions.
6.	Scalability of the Solution	As the number of vehicles on the road is increasing exponentially as compared the amount of road where they can be driven, the Scope and Scalability of Traffic Volume Estimation is immense as it can process a large amount of data and handle a growing number of users, making it a reliable and effective solution.

REQUIREMENT ANALYSIS

3.1 Functional requirement:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	As a user, I can register for the application by entering my email, password, and confirming my password.
FR-2	User Login	As a user, I will receive confirmation email once I have registered for the application
FR-3	User Dashboard	As a user, I can register for the web using mail id

3.2 Non Functional requirements:

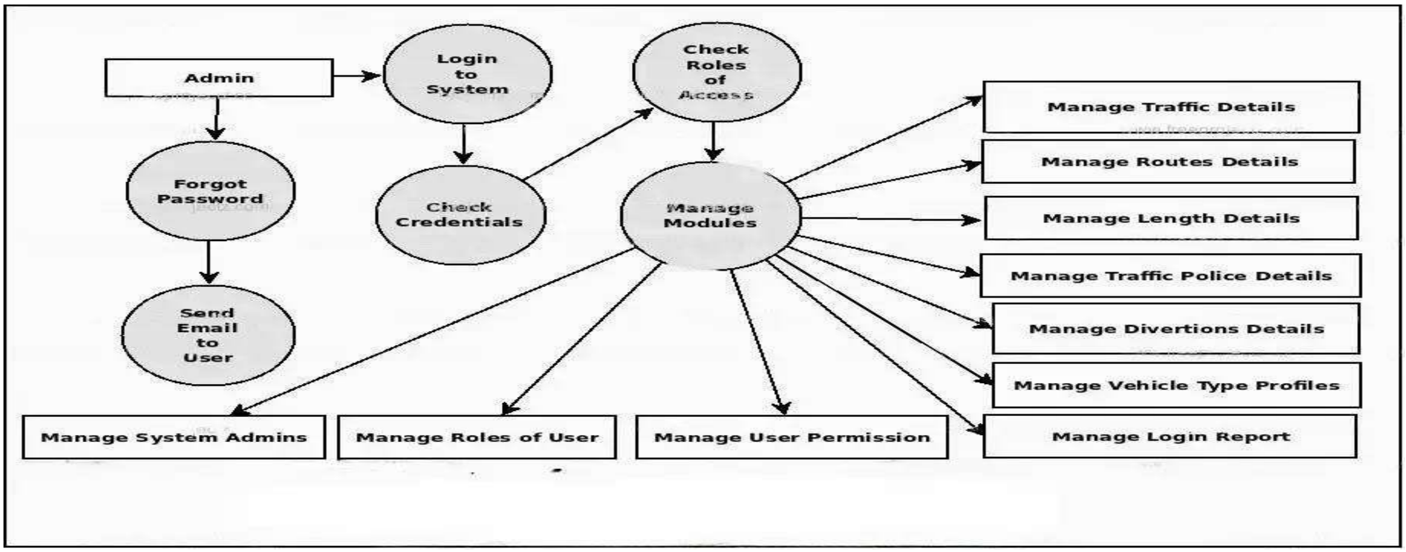
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The system should be easy to use, with an intuitive user interface and clear visualizations that enable users to quickly understand and act on the information provided.
NFR-2	**Security**	The system should be secure, with appropriate measures to protect against unauthorized access, data breaches, or cyber-attacks.
NFR-3	**Reliability**	The system should be highly reliable, with minimal downtime or system failures, to ensure that it can operate 24/7.
NFR-4	**Performance**	The system should be able to process large amounts of data quickly and efficiently to ensure real-time monitoring and response.
NFR-5	**Availability**	The system should be highly available, with multiple redundancy and failover mechanisms to ensure uninterrupted operation.
NFR-6	**Scalability**	The system should be scalable to handle increasing amounts of traffic data as the number of vehicles and population grows.

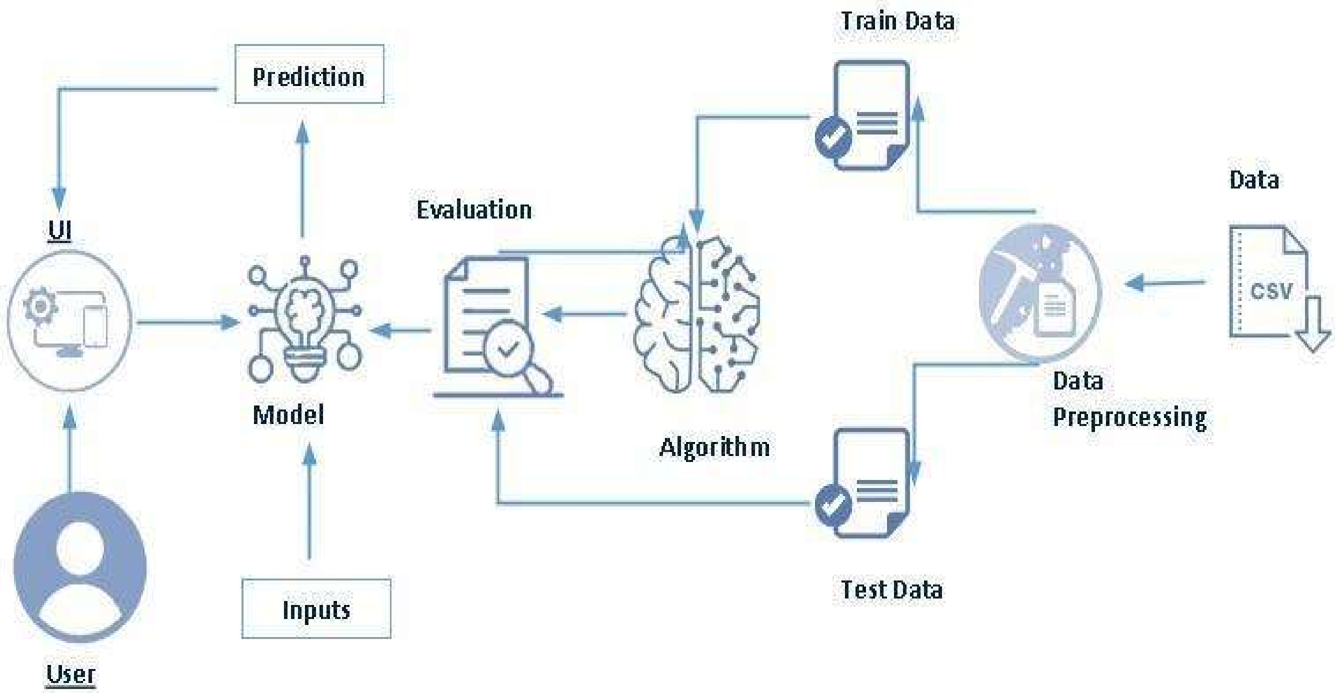
PROJECT DESIGN

4.1 Data Flow Diagram

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 rightamount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



4.2 Solution Architecture Diagram:



Solution Architecture:

To address the problems with tracking the vehicle count in each and every place, AI-ML has given a solution to such kind of traffic related issues, which are able to measure the volume of traffic, identify the violations of traffic rules etc. ML models could give early alerts of severe traffic to help prevent issues related to traffic problems. Here is a possible solution architecture for this problem:

1. Data Collection and Preprocessing: The first step is to collect the Traffic Volume data and preprocess it to remove any irrelevant information such as URLs, HTML tags, and special characters. This can be done using Python libraries like BeautifulSoup, pandas, and NLTK.

2. Feature Extraction: The next step is to extract the relevant features from the preprocessed data. This involves converting the text data into a numerical format that can be used by machine learning algorithms. Common feature extraction techniques include Bag of Words, TF-IDF, and Word Embedding.

3. Machine Learning Model: Once the features are extracted, a machine learning model needs to be trained on the data to predict the estimated traffic in any area. A multi-label classification Model can be used here, as each frame could have multiple number of vehicles. Popular machine learning algorithms for multi-label classification include Random Forest, Naive Bayes, and Neural Networks.

4. Evaluation: Once the model is trained, it can be used to estimate the Traffic Volume. The accuracy of the model can be evaluated using metrics like F1 score, precision, and recall. The model can be retrained and refined based on the evaluation results.

5. Deployment: Finally, the model can be deployed as a web service, allowing users to access relevant information about the Traffic Volume. This can be done using Python libraries like Flask or Djasngo.

Overall, this solution architecture can provide an effective way to autonomously Estimate the Traffic Volume using machine learning and natural language processing techniques.

Technical Architecture:

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

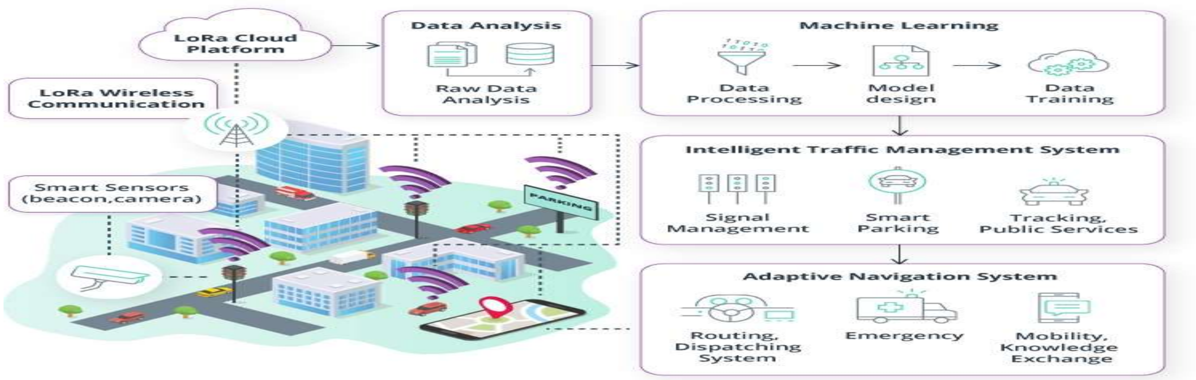


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	Traffic models	Traffic models use statistical and mathematical algorithms to predict trafficpatterns and congestion levels.	Machine learning
2.	Sensors	Sensors are an essential component of traffic intelligence systems. They collect data on traffic flow, volume, and speed	IOT sensors , RFID, AIDC
3.	Data collection and processing	Data collection and processing systems are used to collect and analyze data from sensors. This includes software that can aggregate, clean, and organize data from multiple sources.	Data collection Technology
4.	Geographic Information Systems(GIS)	GIS technology is used to visualize trafficdata and display it on maps. This allows planners to see traffic patterns and congestion levels in real-time and make better-informed decisions.	GIS Technology
5.	Artificial intelligence and machinelearning	AI and machine learning techniques can beused to analyze traffic data and make predictions about traffic patterns	AI and ML Technology
6.	Real-time traffic information	Real-time traffic information systems use data from sensors and other sources to provide up-to-date information about trafficconditions.	Sensors, GPS, Smart cameras

S.No	Characteristics	Description	Technology
1.	Real-time data	Traffic intelligence applications require real-time data to be effective. so that transportation planners,drivers, and other stakeholders can make timely decisions.	Big data
2.	Accuracy	Traffic intelligence applications must be accurate.Any inaccuracies in the data can lead to incorrect decisions, which can have negative impacts on traffic flow and safety.	ICR and OCR
3.	Scalability	Traffic intelligence applications must be scalable tohandle large amounts of data. Traffic intelligence applications must be able to handle this data growth and continue to provide reliable information.	Technology used
4.	Flexibility	raffic intelligence applications must be flexible to adapt to changes in traffic patterns, infrastructure, and other factors	Technology used
5.	Integration	Traffic intelligence applications must be integrated with other transportation systems and technologies.This includes traffic management systems, GPS devices, traffic signal control systems, and othertechnologies.	Technology used
6.	User-friendly	Traffic intelligence applications must be user-friendly for transportation planners, drivers, andother stakeholders.	Technology used

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	Team Member
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by Entering my email, password, and confirmingmy password.	I can access my account / dashboard	High	Surya
		USN-2	As a user, I will receive confirmation emailonce I have registered for the application	I can receive confirmation email & click confirm	High	Nitin
		USN-3	As a user, I can register for the web usingmail id	I can register & access the dashboard using Gmail	High	Sriakash
		USN-4	As a user, I can register for the application through password if it id forgot then thee codewill be send to their registered mail id	I can register by recreatingthe login password	Medium	Santosh
	Login	USN-5	As a user, I can log into the application by entering email & password	I can register and accessthe dashboard	High	Pramoth
	Dashboard		After login we can check the our credentials And in the dashboard we can have access tomanage the number of modules like we can manage the route, traffic and length of the traffic etc..			

CODING & SOLUTIONING

5.1 Feature 1:

Data Collection: Gather a dataset that includes historical traffic volume along with relevant features such as time of day, day of week, weather conditions, road characteristics, and any other factors that might impact traffic.

Data Preprocessing: Clean the dataset by handling missing values, outliers, and inconsistencies. Perform feature engineering to extract useful information from the raw data, such as creating additional features like hour of the day or day of the week from the timestamp.

Feature Selection: Select the most relevant features that have a significant impact on traffic volume. You can use techniques like correlation analysis or feature importance from a machine learning algorithm.

Model Selection: Choose a suitable machine learning algorithm for traffic volume estimation. Some commonly used algorithms include linear regression, decision trees, random forests, and gradient boosting algorithms.

Code:

```
import os, types

import pandas as pd

from botocore.client import Config

import ibm_boto3

def __iter__(self): return 0

cos_client = ibm_boto3.client(service_name='s3',

    ibm_api_key_id='yA8fM2HPHxIVdnnGAEaRSWjpRFsYnrLctqmJQKo3',

    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",

    config=Config(signature_version='oauth'),

    endpoint_url='https://s3.private.us-south.cloud-object-storage.appdomain.cloud')

bucket = 'traffictelligence-donotdelete-pr-sad9xevdpo31uu'

object_key = 'traffic volume.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']

if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )


dataset = pd.read_csv(body)

dataset.head()
```

5.2 Feature 2

Model Training: Split the dataset into training and testing sets. Train the selected model using the training set, adjusting the model parameters if needed. Ensure you use appropriate evaluation metrics, such as mean squared error (MSE) or mean absolute error (MAE), to assess the model's performance.

Model Evaluation: Evaluate the trained model using the testing set. Compare the predicted traffic volume with the actual values using the chosen evaluation metrics to determine the accuracy and performance of the model.

Model Optimization: Fine-tune the model by adjusting hyperparameters or trying different algorithms to improve its performance. You can use techniques like grid search or random search to find the best combination of hyperparameters.

Model Deployment: Once you are satisfied with the model's performance, deploy it into a production environment where it can be used to estimate traffic volume in real-time. Ensure you have a mechanism to update the model periodically with new data to maintain its accuracy.

Remember that the specific coding and implementation details will depend on the programming language and machine learning framework you choose to work.

Code:

```
//Model Training

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)

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)

//Deploying

wml_credentials={

   "url":"https://us-south.ml.cloud.ibm.com",

   "apikey":"mNWuonLOy5PwWpuz2o1bFZZtfZFdcPne2eE"

}

client=APIClient(wml_credentials)
```

5.3 Database Scheme

A typical database schema for traffic volume estimation in machine learning might include the following tables:

Traffic Data: This table stores the raw traffic data collected from various sources. It can include fields such as timestamp, location, vehicle count, vehicle type, weather conditions, and any other relevant data.

Preprocessed Data: This table holds the preprocessed and cleaned data ready for model training. It may include additional features derived from the raw data, such as time of day, day of the week, holidays, and any other relevant variables.

Model Parameters: This table stores the parameters and configurations of the machine learning models used for traffic volume estimation. It can include information about the model type, hyperparameters, and any other relevant settings.

Model Performance: This table tracks the performance metrics of the trained models, such as accuracy, precision, recall, and mean squared error. It can also include additional information like training duration and dataset used.

Predictions: This table stores the predictions generated by the trained models for traffic volume estimation. It typically includes fields such as timestamp, predicted volume, and any other relevant output variables.

Actual Volume: This table holds the actual traffic volume data for comparison and evaluation purposes. It can include fields such as timestamp and observed volume.

Evaluation Metrics: This table stores the evaluation metrics calculated based on the comparison between predicted and actual traffic volume. It can include metrics such as mean absolute error, root mean squared error, and any other relevant measures.

Code:

```
//Accuracy

from sklearn.metrics import accuracy_score

accuracy = accuracy_score(y_test, predictions)

print("Accuracy:", accuracy)
```

```
//Metrics
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5, 10]
}
grid_search = GridSearchCV(estimator=randomforest, param_grid=param_grid, cv=5)
grid_search.fit(x_train, y_train)
best_params = grid_search.best_params_
best_model = grid_search.best_estimator_
predictions = best_model.predict(x_test)
print(grid_search.best_score_)
```

RESULTS

6.1Performance Metrics

Performance metrics for a model, you typically need a set of labelled data (with known ground truth) to evaluate the model's predictions. The choice of performance metrics depends on the type of problem you are solving, such as classification, regression, or clustering. Here are some common performance metrics for different types of models:

Metrics:

- **MAE:**

  The "MAE" metric stands for Mean Absolute Error. It is a commonly used evaluation metric in machine learning and statistics to measure the average magnitude of errors in a set of predictions or estimates, without considering their direction.

  The Mean Absolute Error is calculated by taking the absolute difference between each predicted value and its corresponding true value, summing these differences, and then dividing by the total number of observations.

```
In [91]: MAE = mean_absolute_error(p3, y_test)

In [92]: MAE
Out[92]: 506.2704397884037
```

- **MSE:**

  The "MSE" metric stands for Mean Squared Error. It is another commonly used evaluation metric in machine learning and statistics to measure the average squared difference between the predicted and true values.

  The Mean Squared Error is calculated by taking the squared difference between each predicted value and its corresponding true value, summing these squared differences, and then dividing by the total number of observations.

```
In [89]: MSE = metrics.mean_squared_error(p3, y_test)

In [90]: MSE
Out[90]: 637498.9541861528
```

- **RMSE:**

  The "RMSE" metric stands for Root Mean Squared Error. It is a commonly used evaluation metric in machine learning and statistics to measure the average magnitude of the residuals (i.e., differences) between the predicted and true values.

  The Root Mean Squared Error is calculated by taking the square root of the mean of the squared differences between the predicted and true values.

```
In [95]: np.sqrt(MSE)

Out[95]: 798.435316219262
```

- **R2 Score:**

  The "R-squared" or "R2 score" metric, also known as the coefficient of determination, is a statistical measure used to evaluate the goodness of fit of a regression model. It indicates the proportion of the variance in the dependent variable that can be explained by the independent variables in the model.

  The R2 score ranges from 0 to 1, where:

  0 indicates that the model does not explain any of the variance in the dependent variable.

  1 indicates that the model perfectly explains all the variance in the dependent variable.

```
In [118]: grid_search.fit(x_train, y_train)

          # Get the best hyperparameters and the best model
          best_params = grid_search.best_params_
          best_model = grid_search.best_estimator_

          # Use the best model for predictions
          predictions = best_model.predict(x_test)

In [119]: print(grid_search.best_score_)

          0.8325488098210168
```

ADVANTAGES AND DISADVANTAGES

Advantages:

- Accurate Traffic Volume Prediction: By utilizing machine learning algorithms and a large historical dataset, the system can learn patterns, trends, and correlations in traffic volume over time. This can result in accurate predictions of traffic volume for future time periods, enabling better planning and decision-making for various applications.
- Real-Time Traffic Monitoring: The ML system can be deployed in real-time to monitor traffic volume and make up-to-date predictions. This can be useful for traffic management, route optimization, and identifying congestion patterns or traffic anomalies.
- Adaptability to Seasonal and Long-Term Trends: The extended dataset covering multiple years allows the system to capture and adapt to seasonal variations and long-term trends in traffic volume. It can identify patterns associated with weekdays, weekends, holidays, special events, and other factors that impact traffic patterns over time.
- Improved Traffic Management and Infrastructure Planning: Accurate traffic volume estimation helps transportation authorities and urban planners make informed decisions regarding road infrastructure planning, traffic signal optimization, and resource allocation. It allows for better management of traffic flow, reduces congestion, and improves overall transportation efficiency.
- Enhanced Safety and Incident Management: The ML system can help identify abnormal traffic conditions and predict potential incidents or congestion hotspots. This information can be used to proactively manage traffic, reroute vehicles, and dispatch emergency services to ensure safer and more efficient traffic operations.
- Data-Driven Insights: By analysing the historical traffic data, the ML system can provide valuable insights into traffic patterns, peak hours, and other factors influencing traffic volume. These insights can be used for long-term planning, policy-making, and optimizing transportation systems.

Disadvantages:

- Data Bias and Generalization: The historical dataset may be subject to biases, such as data collection methods, sensor placement, or changes in infrastructure over time. These biases can impact the model's ability to generalize accurately to new locations or future time periods, leading to potential prediction errors.
- Lack of Real-Time Accuracy: ML models trained on historical data might not capture sudden changes or unforeseen events that can affect traffic volume in real-time. Examples include accidents, road construction, weather conditions, or major public events. Therefore, the system's accuracy might decrease when faced with dynamic and unpredictable situations.
- Data Quality and Missing Values: The dataset may have missing or incomplete data, which can affect the model's performance. Missing data can lead to biased predictions or require imputation techniques that may introduce additional uncertainties or inaccuracies.
- Sensitivity to External Factors: ML models used for traffic volume estimation often rely on various external factors, such as weather data, holidays, or social events. Changes or anomalies in these factors that were not present or accounted for in the historical dataset can impact the model's accuracy and reliability.
- Model Complexity and Interpretability: ML models used for traffic volume estimation, such as deep learning models, can be highly complex and difficult to interpret. This can limit the ability to explain the reasoning behind predictions and make it challenging to identify and troubleshoot potential issues or biases.
- Computational Requirements: Developing and deploying ML models for traffic volume estimation may require significant computational resources, including processing power and memory. This can pose challenges in terms of infrastructure requirements and operational costs.
- Model Maintenance and Adaptability: Traffic patterns and infrastructure evolve over time. The ML model would need to be regularly retrained and updated to adapt to these changes. Ensuring the availability of up-to-date and representative data for retraining can be a resource-intensive and time-consuming process.
- Privacy and Security Concerns: Handling traffic data, which may include sensitive information such as location and travel patterns, raises privacy and security concerns. Proper data anonymization and security measures must be in place to protect the privacy of individuals and prevent unauthorized access or misuse of data.

CONCLUSION

In conclusion, an ML-based Traffic Volume Estimation system utilizing a dataset spanning from 2012 to 2022 offers numerous advantages, including accurate traffic volume prediction, real-time monitoring, adaptability to seasonal and long-term trends, improved traffic management and infrastructure planning, enhanced safety and incident management, data-driven insights, scalability, and automation.

However, it is crucial to consider the potential disadvantages associated with such a system. These include data bias and generalization, lack of real-time accuracy, data quality issues and missing values, sensitivity to external factors, model complexity and interpretability challenges, computational requirements, model maintenance and adaptability, as well as privacy and security concerns.

Addressing these disadvantages requires continuous monitoring, validation, and improvement of the ML model, considering potential biases and updating it to accommodate real-time changes and emerging factors. Additionally, combining ML-based predictions with human expertise and judgment can enhance the system's reliability and effectiveness.

Moreover, it is important to recognize that an ML-based Traffic Volume Estimation system is not a standalone solution for traffic management and planning. While it can provide valuable predictions and insights, it should be complemented by other data sources and methodologies. Integrating data from traffic sensors, GPS devices, and other sources can help validate and refine the ML model's predictions, ensuring a more comprehensive and accurate understanding of traffic patterns.

In conclusion, an ML-based Traffic Volume Estimation system utilizing a historical data offers significant advantages in terms of accurate predictions, real-time monitoring, adaptability, and data-driven insights. However, it is essential to address the potential disadvantages, including data bias, real-time accuracy challenges, and model complexity. By incorporating validation processes, maintaining data quality, and involving stakeholders, these limitations can be mitigated. Ultimately, the successful implementation of an ML-based Traffic Volume Estimation system requires a combination of technical expertise, domain knowledge, on-going monitoring, and collaboration to ensure its effectiveness in improving traffic management and infrastructure planning.

FUTURE SCOPE

The future scope of the Traffic Volume Estimation system is promising, with several potential advancements and opportunities on the horizon. Here are a few areas where the system can evolve and expand:

1. Integration of Real-Time Data: Enhancing the system's capability to incorporate real-time data from various sources, such as connected vehicles, mobile applications, and social media, can significantly improve its accuracy and responsiveness. By leveraging real-time data streams, the system can better adapt to dynamic traffic conditions and provide more accurate predictions and insights.

2. Multi-Modal Traffic Estimation: Expanding the system's capabilities beyond road traffic to include other modes of transportation, such as pedestrian and bicycle traffic, public transportation, and shared mobility services, can provide a comprehensive view of the transportation network. This integration can enable more holistic planning and management strategies, considering the interactions and interdependencies between different modes of transportation.

3. Improved Predictive Modelling: Advancements in machine learning algorithms, such as deep learning and reinforcement learning, can further enhance the accuracy and predictive capabilities of the system. These techniques can help capture complex patterns and non-linear relationships in traffic volume data, leading to more precise predictions and better performance under diverse scenarios.

4. Enhanced Spatial Resolution: Fine-grained spatial resolution is essential for localized traffic management and planning. Future developments may focus on increasing the granularity of the system by utilizing high-resolution traffic data, such as data from smart intersections, traffic cameras, or sensors embedded in road infrastructure. This can provide more detailed and localized insights, enabling targeted interventions for specific areas or intersections.

5. Integration with Intelligent Transportation Systems: The ML-based Traffic Volume Estimation system can be integrated with other intelligent transportation systems, such as adaptive traffic signal control, dynamic routing algorithms, or incident detection systems. This integration can create a synergistic effect, allowing the system to leverage additional data sources and collaborate with existing traffic management technologies to optimize overall transportation efficiency.

6. Sustainability and Environmental Considerations: Future iterations of the system can incorporate environmental factors and promote sustainable transportation solutions. By integrating data on emissions, energy consumption, and alternative modes of transport, the system can support decision-making that prioritizes environmentally friendly transportation options and helps reduce traffic-related pollution and carbon footprint.

7. Policy and Decision Support: ML-based Traffic Volume Estimation systems can serve as valuable decision support tools for policymakers, urban planners, and transportation authorities. By providing accurate predictions and simulations, the system can assist in evaluating the impact of policy interventions, infrastructure changes, or new mobility solutions, enabling evidence-based decision-making for transportation planning and management.

In summary, the future scope of an ML-based Traffic Volume Estimation system is expansive and holds great potential for advancements in accuracy, real-time capabilities, multi-modal integration, spatial resolution, and integration with other intelligent transportation systems, sustainability considerations, and decision support. As technology continues to evolve and new data sources become available, the system's ability to improve traffic management, infrastructure planning, and sustainable transportation will continue to grow, ultimately enhancing the efficiency and effectiveness of transportation systems in the future.

APPENDIX

10.1 Source Code

Code to Train Model:

```
# TrafficTelligence

## Importing Dependencies

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

from sklearn import metrics

import xgboost

import matplotlib.pyplot as plt

from sklearn.preprocessing import scale

from sklearn.preprocessing import LabelEncoder

from sklearn.model_selection import train_test_split

import pickle

from sklearn.metrics import mean_squared_error

from sklearn.metrics import mean_absolute_error

from sklearn.metrics import r2_score

## Importing the CSV File

import os, types

import pandas as pd

from botocore.client import Config

import ibm_boto3

def __iter__(self): return 0

# @hidden_cell

# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.

# You might want to remove those credentials before you share the notebook.

cos_client = ibm_boto3.client(service_name='s3',

    ibm_api_key_id='yA8fM2HPHxIVdnnGAEaRSWjpRFspupwYnrLctqmJQKo3',
```

```
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us-south.cloud-object-storage.appdomain.cloud')
bucket = 'traffictelligence-donotdelete-pr-sad9xevdpo31uu'
object_key = 'traffic volume.csv'
body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )
dataset = pd.read_csv(body)
dataset.head()
dataset
dataset.head()
dataset.info()
## Checking and Handling NULL Values
dataset.isnull().sum()
dataset['temp'].fillna(dataset['temp'].mean(), inplace=True)
dataset['rain'].fillna(dataset['rain'].mean(), inplace=True)
dataset['snow'].fillna(dataset['snow'].mean(), inplace=True)
from collections import Counter
print(Counter(dataset['weather']))
dataset['weather'].fillna('Clouds', inplace =True)
(dataset['weather'])
dataset[["day","month","year"]] = dataset["date"].str.split("-", expand = True)
dataset[["hours","minutes","seconds"]] = dataset["Time"].str.split(":", expand = True)
dataset.drop(columns=['date','Time'], axis=1, inplace = True)
dataset.isnull().sum()
dataset.head()
dataset.corr()
sns.heatmap(dataset.corr())
dataset.describe()
sns.countplot(x='weather', data=dataset)
sns.countplot(x='holiday', hue='weather', data=dataset)
plt.show()
 sns.pairplot(dataset)
## Data Encoding
```

```
le = LabelEncoder()
dataset['weather'] = le.fit_transform(dataset['weather'])
dataset['holiday'] = le.fit_transform(dataset['holiday'])
y = dataset['traffic_volume']
x = dataset.drop(columns=['traffic_volume'], axis=1)
names = x.columns
x = scale(x)
x = pd.DataFrame(x, columns = names)
x.head()
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
## Initializing and Analyzing Algorithms
lin_reg = linear_model.LinearRegression()
Dtree = tree.DecisionTreeRegressor()
Rand = ensemble.RandomForestRegressor()
svr = svm.SVR()
XGB = xgboost.XGBRegressor()
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)
## Checking Algorithm Metrics
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)
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))
MSE = metrics.mean_squared_error(p3, y_test)
MSE
MAE = mean_absolute_error(p3, y_test)
MAE
R2 = r2_score(p3, y_test)
R2
np.sqrt(MSE)
## Creating Model using Pickle
pickle.dump(Rand, open("model.pk1","wb"))
pickle.dump(le, open("encoder.pk1","wb"))
## Using IBM Watson Studio to create an API for Web Application to Fetch Data From
from ibm_watson_machine_learning import APIClient
wml_credentials={
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"mNWuonLOy5PwWpuKEXHH0Lm9z2o1bFZZtfZFdcPne2eE"
}
client=APIClient(wml_credentials)
def guid_from_space_name(client, space_new):
    space = client.spaces.get_details()
    return (next(item for item in space['resources'] if item['entity']['name'] == space_new)['metadata']['id'])
space_uid = guid_from_space_name(client, 'TrafficTelligenceDeploymentSpace')
print("Space_UID: "+ space_uid)
client.set.default_space(space_uid)
client.software_specifications.list()
software_spec_uid = client.software_specifications.get_uid_by_name('runtime-22.2-py3.10')
software_spec_uid
from sklearn.ensemble import RandomForestRegressor
c = RandomForestRegressor()
```

```
model_details = client.repository.store_model(model= randomforest, meta_props={
    client.repository.ModelMetaNames.NAME:"RandomForest",
    client.repository.ModelMetaNames.TYPE:'scikit-learn_1.1',
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
})
model_id = client.repository.get_model_id(model_details)
model_id
from sklearn.metrics import accuracy_score
# Assuming you have the actual target values (y_test) and predictions (predictions)
accuracy = accuracy_score(y_test, predictions)
print("Accuracy:", accuracy)
# from sklearn.metrics import accuracy_score
# # Assuming you have predicted labels (y_pred) and actual labels (y_true)
# accuracy = accuracy_score(y_true, y_pred)
# print("Accuracy:", accuracy)
from sklearn.model_selection import GridSearchCV
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5, 10]
}
grid_search = GridSearchCV(estimator=randomforest, param_grid=param_grid, cv=5)
grid_search.fit(x_train, y_train)
# Get the best hyperparameters and the best model
best_params = grid_search.best_params_
best_model = grid_search.best_estimator_
# Use the best model for predictions
predictions = best_model.predict(x_test)
print(grid_search.best_score_)
```

Flask Code:

```
import numpy as np

import os

import pandas

from flask import Flask, request, jsonify, render_template

import requests

API_KEY = "mNWuonLOy5PwWpuKEXHH0LmfZFdcPne2eE"

token_response = requests.post(

    "https://iam.cloud.ibm.com/identity/token",

    data={"apikey": API_KEY, "grant_type": "urn:ibm:params:oauth:grant-type:apikey"},

)

mltoken = token_response.json()["access_token"]

header = {"Content-Type": "application/json", "Authorization": "Bearer " + mltoken}

app = Flask(__name__, template_folder="templates")

@app.route("/")

def home():

    return render_template("home.html")

@app.route("/index")

def index():

    return render_template("index.html")

@app.route("/predict", methods=["POST", "GET"])

def predict():

    if request.method == "POST":

        data1 = request.form["holiday"]

        data2 = request.form["temp"]

        data3 = request.form["rain"]

        data4 = request.form["snow"]

        data5 = request.form["weather"]

        data6 = request.form["year"]

        data7 = request.form["month"]

        data8 = request.form["day"]

        data9 = request.form["hours"]

        data10 = request.form["minutes"]

        data11 = request.form["seconds"]
```

```
 payload_scoring = {
      "input_data": [
          {
              "field": [
                  [
                      "holiday", "temp", "rain", "snow", "weather", "year", "month", "day", "hours", "minutes", "seconds",
                  ]
              ],
              "values": [
                  [
                      data1, data2, data3, data4, data5, data6, data7, data8, data9, data10, data11,
                  ]
              ],
          }
      ]
   }
   response_scoring = requests.post(
      "https://us-south.ml.cloud.ibm.com/ml/v4/deployments/d2d55c1d-b69e-43e5-988b-
6f84d6b164fb/predictions?version=2023-05-19",
      json=payload_scoring,
      headers={"Authorization": "Bearer " + mltoken},
   )
   print("Scoring response")
   print(response_scoring.json())
   predictions = response_scoring.json()
   prediction = predictions["predictions"][0]["values"][0][0]
   text = "Estimated Traffic is: "
   return render_template("output.html", result=str(text + str(prediction)) + "units")
if __name__ == "__main__":
   port = int(os.environ.get("PORT", 5000))
   app.run(port=port, debug=True, use_reloader=False)
```

10.2 GITHUB & Project Demo Video Link

GITHUB : **https://github.com/naanmudhalvan-SI/IBM--18091-1682506636**

Demo Video Link :
https://drive.google.com/file/d/1f_Ub7zSx3UZYN0e8yKGgilU7F4G7o2JA/view?usp=drivesdk