PROJECT REPORT

LTVIP2025TMID41765

1. INTRODUCTION

1.1 **Project Overview**

The Traffic Intelligence project aims to revolutionize traffic management through the implementation of advanced machine learning techniques for accurate and real-time traffic volume estimation. With the ever-increasing urbanization and the rise in the number of vehicles on the road, understanding and managing traffic patterns have become crucial for efficient urban planning and transportation systems.

1.2 Purpose

The "Traffic Intelligence - Advanced Traffic Volume Estimation using Machine Learning" project serves a crucial purpose in modern urban infrastructure by addressing the challenges inherent in traffic management. At its core, the project aims to optimize the efficiency of traffic control systems through the deployment of advanced machine learning techniques. By providing accurate and real-time traffic volume estimations, the project seeks to empower decision-makers, urban planners, and transportation authorities with invaluable data-driven insights. This, in turn, facilitates informed decision-making for resource allocation and infrastructure development. The project's ultimate goal is to enhance the overall efficiency of traffic management, contributing to optimized resource allocation, reduced congestion, and improved environmental sustainability. The development of a user-friendly interface ensures that stakeholders can easily access and interpret the traffic data, making it a valuable tool for both professionals and the wider community involved in urban planning and transportation.

2.IDEATION PHASE

2.1 Problem Statement
2.2 Empathy Map Canvas
2.3 Brainstorming

3.REQUIREMENT ANALYSIS

3.1 Customer Journey map
3.2 Solution Requirement
3.2 Data Flow Diagram
3.3 Technology Stack

4.PROJECT DESIGN

4.1 Problem Solution Fit
4.2 Proposed Solution
4.3 Solution Architecture

5.PROJECT PLANNING & SCHEDULING

5.1 Project Planning

6.FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

7. RESULTS

7.1 Output Screenshots

```
#Model Building
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost
```

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

```
#Testing the model
#1.using R-squared_score
from sklearn.metrics import r2_score
p1 = lin_reg.predict(x_test)
print(r2_score(p1,y_test))
```

```
-5.399396398322183
```

```
p2 = Dtree.predict(x_test)
print(r2_score(p2,y_test))
```

```
0.6932439744468677
```

```
p3 = Rand.predict(x_test)
print(r2_score(p3,y_test))
```

```
0.8058847456428343
```

```
p4 = svr.predict(x_test)
print(r2_score(p4,y_test))
```

```
-11.972215715232434
```

```
p5 = XGB.predict(x_test)
print(r2_score(p5,y_test))
```

```
0.8066516776309793
```

```
#2.Using Root mean squared error(RMSE)
from sklearn import metrics
```

```
MSE = metrics.mean_squared_error(p1,y_test)
np.sqrt(MSE)
```

```
1838.3976719006828
```

```
MSE = metrics.mean_squared_error(p2,y_test)
np.sqrt(MSE)
```

```
1096.4189738069322
```

```
MSE = metrics.mean_squared_error(p3,y_test)
np.sqrt(MSE) #Less compared to others
```

```
793.1434700361461
```

```
MSE = metrics.mean_squared_error(p4,y_test)
np.sqrt(MSE)
```

```
1715.2770939066922
```

```
MSE = metrics.mean_squared_error(p5,y_test)
np.sqrt(MSE)
```

```
794.8443863964126
```

In [71]:
```
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)
```

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

```
names = x.columns
```

```
from sklearn.preprocessing import scale
```

```
x = scale(x)
```

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

```
x.head()
```

	temp	rain	snow	day	month	year	hours	minutes	seconds	weather_v2	holiday_v2
0	0.530485	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	-0.345548	0.0	0.0	-0.566452	0.015856
1	0.611467	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	-0.201459	0.0	0.0	-0.566452	0.015856
2	0.627964	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	-0.057371	0.0	0.0	-0.566452	0.015856
3	0.669205	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	0.086718	0.0	0.0	-0.566452	0.015856
4	0.744939	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	0.230807	0.0	0.0	-0.566452	0.015856

```
#Model Deployment
#saving the model
import pickle
from sklearn.preprocessing import LabelEncoder
le = le = LabelEncoder()
pickle.dump(Rand, open("model.pkl",'wb'))
pickle.dump(le, open("encoder.pkl", "wb"))
```

```
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)
```

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

```
#importing necessary libraries

import pandas as pd
import numpy as np
import seaborn as sns
import sklearn as sk
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost
```

```
#importing the data

data = pd.read_csv('DataSets/traffic volume.csv')
```

In [4]:
```
#displaying first 5 rows of the data
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 [6]:
```
# used to display the null values of the data

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

Out[6]:
```
holiday           0
temp             53
rain              2
snow             12
weather          49
date              0
Time              0
traffic_volume    0
dtype: int64
```

In [14]:
```
data['temp'].fillna(data['temp'].mean(),inplace=True)
data['rain'].fillna(data['rain'].mean(),inplace=True)
data['snow'].fillna(data['snow'].mean(),inplace=True)

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

```
Counter({'Clouds': 15144, 'Clear': 13383, 'Mist': 5942, 'Rain': 5665, 'Snow': 2875, 'Drizzle': 1818, 'Haze': 1359, 'Thunderstor
m': 1033, 'Fog': 912, nan: 49, 'Smoke': 20, 'Squall': 4})
```

In [15]:
```
data['weather'].fillna('Clouds',inplace=True)
```

In [15]:
```
data['weather'].fillna('Clouds',inplace=True)
```

In [17]:
```
#splitting the date column into year,month,day
data[["day", "month", "year"]] = data["date"].str.split("-", expand = True)
```

In [18]:
```
#splitting the Time column into hour,minute,second
data[["hours", "minutes", "seconds"]] = data["Time"].str.split(":", expand = True)
```

In [19]:
```
data.drop(columns=['date','Time'],axis=1,inplace=True)
```

In [20]:
```
data.head()
```

Out[20]:

	holiday	temp	rain	snow	weather	traffic_volume	day	month	year	hours	minutes	seconds
0	None	288.28	0.0	0.0	Clouds	5545	02	10	2012	09	00	00
1	None	289.36	0.0	0.0	Clouds	4516	02	10	2012	10	00	00
2	None	289.58	0.0	0.0	Clouds	4767	02	10	2012	11	00	00
3	None	290.13	0.0	0.0	Clouds	5026	02	10	2012	12	00	00
4	None	291.14	0.0	0.0	Clouds	4918	02	10	2012	13	00	00

In [21]:
```
#used to understand the descriptive analysis of the data
data.describe()
```

Out[21]:

	temp	rain	snow	traffic_volume
count	48204.000000	48204.000000	48204.000000	48204.000000
mean	281.205351	0.334278	0.000222	3259.818355
std	13.336338	44.789133	0.008168	1986.860670
min	0.000000	0.000000	0.000000	0.000000
25%	272.180000	0.000000	0.000000	1193.000000
50%	282.429000	0.000000	0.000000	3380.000000
75%	291.800000	0.000000	0.000000	4933.000000
max	310.070000	9831.300000	0.510000	7280.000000

In [27]:
```
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows
# how to understand word labels.
label_encoder = preprocessing.LabelEncoder()
```

In [30]:
```
 data['weather']= label_encoder.fit_transform(data['weather'])
data['holiday'] = label_encoder.fit_transform(data['holiday'])
```

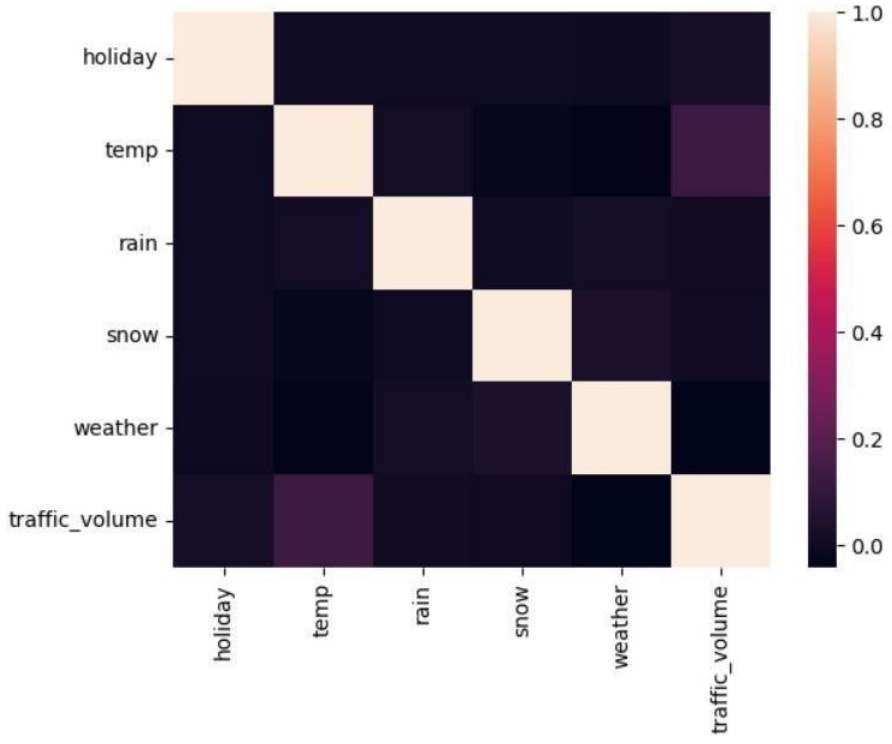
In [32]:
```
cor = data.corr()
cor
```

Out[32]:

	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 [33]:
```
sns.heatmap(cor)
```

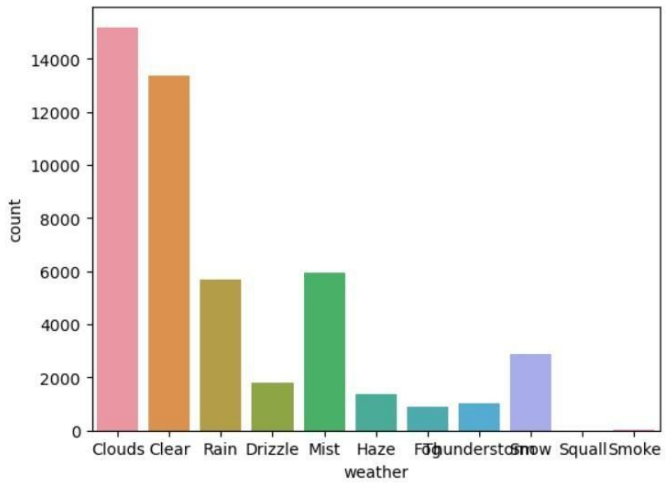
Out[33]: <AxesSubplot:>



In [52]:
```
sns.countplot(data['weather'])
```

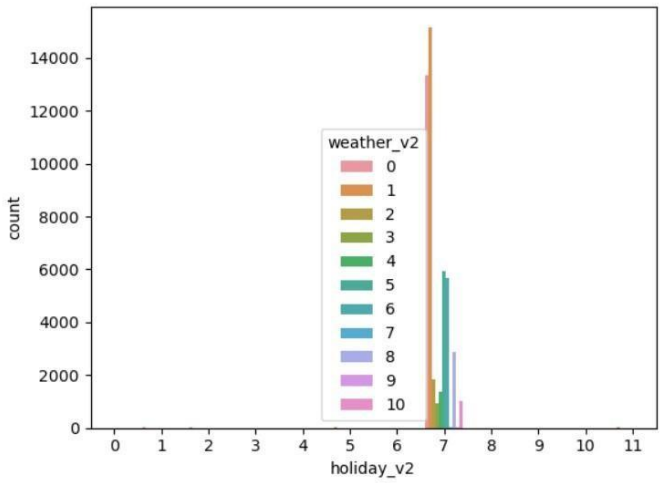
```
D:\Anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
ersion 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will re
sult in an error or misinterpretation.
  warnings.warn(
```

Out[52]: <AxesSubplot:xlabel='weather', ylabel='count'>

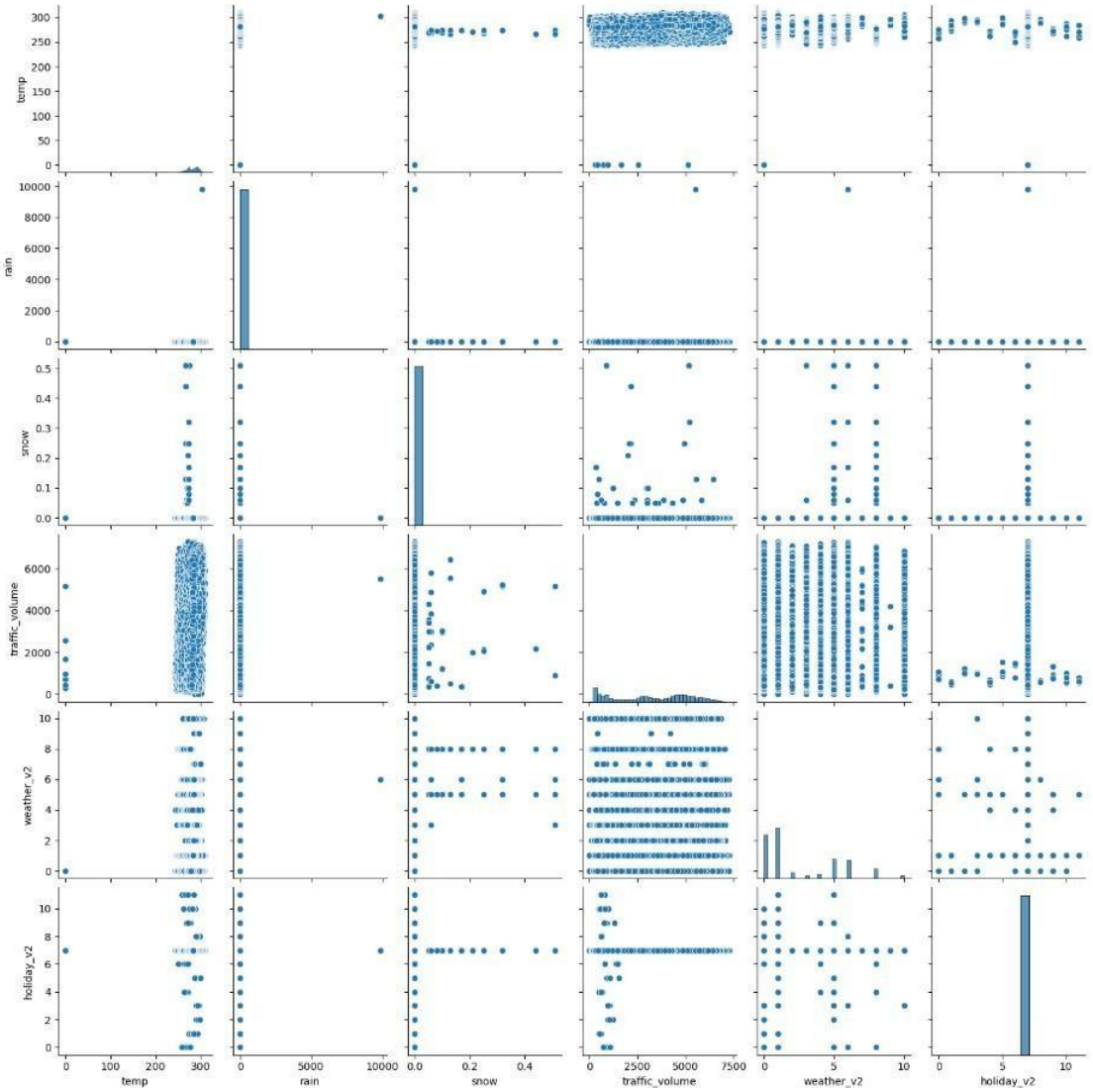


In [57]:
```
sns.countplot(data['holiday_v2'],hue=data['weather_v2'])
plt.show()
```

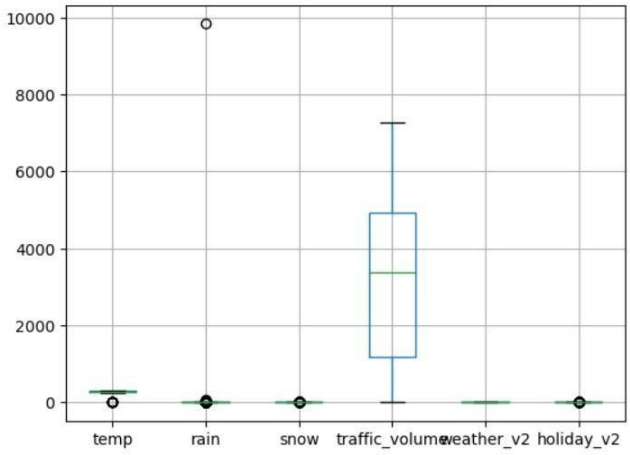
```
D:\Anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
ersion 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will re
sult in an error or misinterpretation.
  warnings.warn(
```



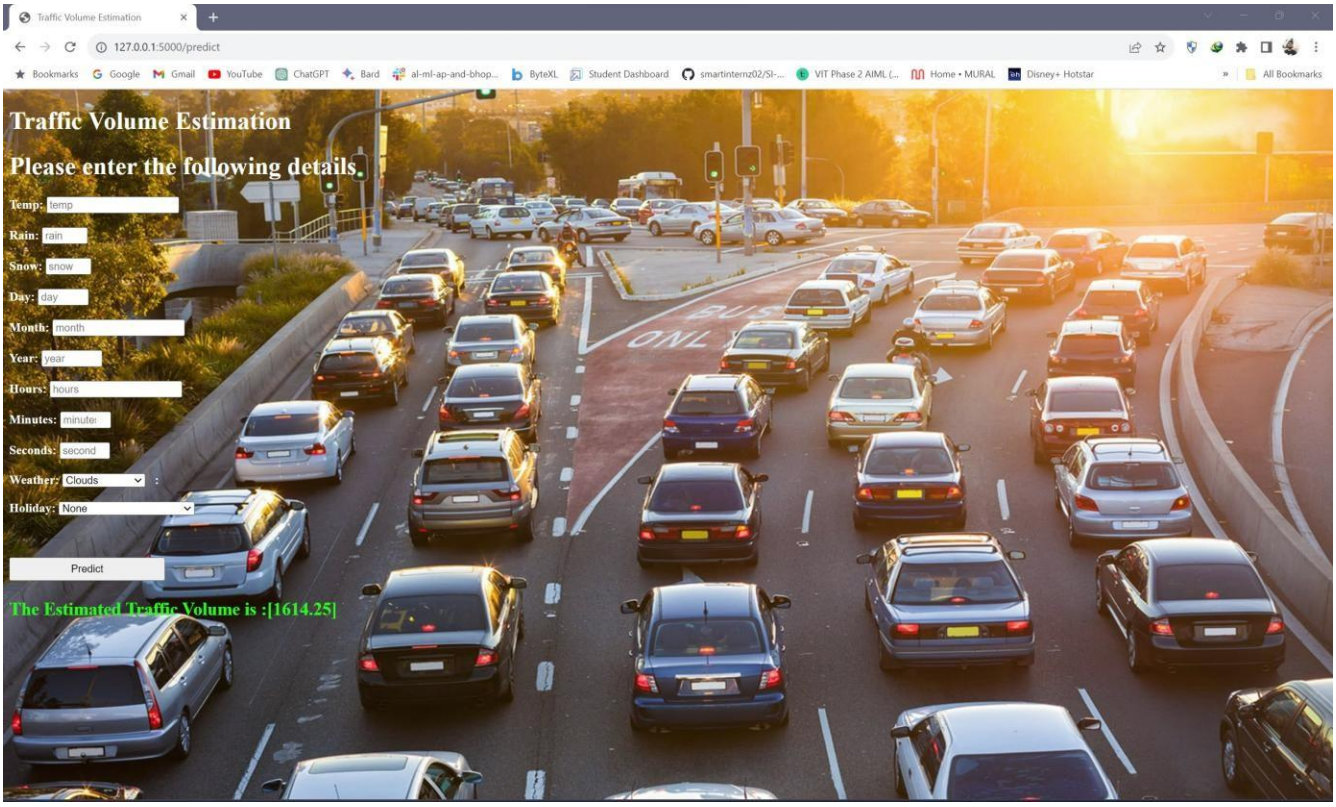
Out[58]: <seaborn.axisgrid.PairGrid at 0x239d5d007f0>



```
In [59]: data.boxplot()
Out[59]: <AxesSubplot:>
```



Our Final Website will be looking like this:



8. ADVANTAGES & DISADVANTAGES

Advantages:

1. **Improved Accuracy:**

- Machine learning models can analyze large datasets and identify complex patterns that may be challenging for traditional methods. This leads to more accurate traffic volume predictions.

2. **Integration with Sensor Data:**
   - Machine learning models can effectively integrate data from various sources, such as traffic cameras, sensors, and GPS devices, providing a comprehensive view of the traffic situation.
3. **Scalability:**
   - Machine learning algorithms can scale to handle large and complex datasets, making them suitable for cities with extensive traffic networks.
4. **Predictive Capabilities:**
   - Machine learning models can be used to predict future traffic conditions based on historical data, helping authorities proactively manage traffic flow and prevent congestion.

Disadvantages:

1. **Data Dependency:**
   - Machine learning models heavily rely on high-quality and representative data. If the training data is biased or incomplete, the model's predictions may be inaccurate or skewed.
2. **Complexity:**
   - Building and maintaining machine learning models can be complex and require specialized knowledge. This complexity can hinder the adoption of these systems, especially for smaller municipalities with limited resources.
3. **Dynamic Nature of Traffic:**
   - Traffic patterns are influenced by a wide range of factors, and they can change rapidly. Machine learning models may struggle to keep up with these dynamic changes, especially if not continuously updated and retrained.

9. CONCLUSION

In conclusion, the application of machine learning for advanced traffic volume estimation in the realm of traffic intelligence brings forth a set of notable advantages and challenges. The accuracy and adaptability offered by machine learning models

present a promising avenue for enhancing traffic management. Real-time analysis capabilities, integration with diverse data sources, scalability, and predictive capabilities contribute to more efficient and proactive traffic control.

However, the successful implementation of machine learning in this context requires addressing several challenges. The dependency on high-quality and unbiased data, the inherent complexity of building and maintaining these models, and the interpretability issues associated with certain algorithms pose significant hurdles. Additionally, the dynamic nature of traffic patterns and the computational resources required for training and running sophisticated models underscore the need for careful consideration and resource allocation.

10. FUTURE SCOPE

In the future, the application of advanced traffic volume estimation using machine learning holds tremendous promise in reshaping urban mobility and transportation systems. Ongoing research efforts are likely to focus on enhancing prediction accuracy through the exploration of sophisticated algorithms, feature engineering techniques, and ensemble methods. A significant avenue for development lies in the integration of traffic intelligence with broader smart city initiatives, facilitating interconnected urban transportation systems that optimize traffic flow and minimize environmental impact.
The adoption of edge computing is poised to enable real-time analysis at the source, reducing latency and enhancing responsiveness. Overcoming the interpretability challenge by incorporating explainable AI techniques will be crucial for building trust among city planners and the public. Future systems may extend beyond road traffic to encompass multi-modal transportation, incorporating pedestrians, cyclists, and public transit. The dynamic adaptation of machine learning models to unforeseen events and continuous improvement mechanisms through online learning and feedback loops are vital considerations. Collaborative efforts between municipalities, transportation agencies, and technology providers can lead to more comprehensive and effective traffic management solutions, fostering a connected and efficient transportation network. Ultimately, the future of machine learning in traffic intelligence lies in its ability to create sustainable, adaptive, and energy-efficient urban mobility solutions.

11. APPENDIX

Our Complete Source Code

1. app.py

2. traffic volume estimation1.ipynb
3. index.html
4. traffic_volume.csv
5. DEMO VIDEO.mp4