1. INTRODUCTION

1.1 Project Overview

Machine Learning (ML) is one of the most important and popular emerging branches these days as it is a part of Artificial Intelligence (AI). In recent times, machine learning becomes an essential and upcoming research area for transportation engineering, especially in traffic prediction. Traffic congestion affects the country's economy directly or indirectly by its means. Traffic congestion also takes people's valuable time, cost of fuel every single day. As traffic congestion is a major problem for all classes in society, there has to be a small-scale traffic prediction for the people's sake of living their lives without frustration or tension. For ensuring the country's economic growth, the road user's ease is required in the first place. This is possible only when the traffic flow is smooth. To deal with this, Traffic prediction is needed so that we can estimate or predict the future traffic to some extent.

In addition to the country's economy, pollution can also be reduced. The government is also investing in the intelligent transportation system (ITS) to solve these issues. The plot of this research paper is to find different machine learning algorithms and speculating the models by utilizing python3.The goal of traffic flow prediction is to predict the traffic to the users as soon as possible. Nowadays the traffic becomes really hectic and this cannot be determined by the people when they are on roads.

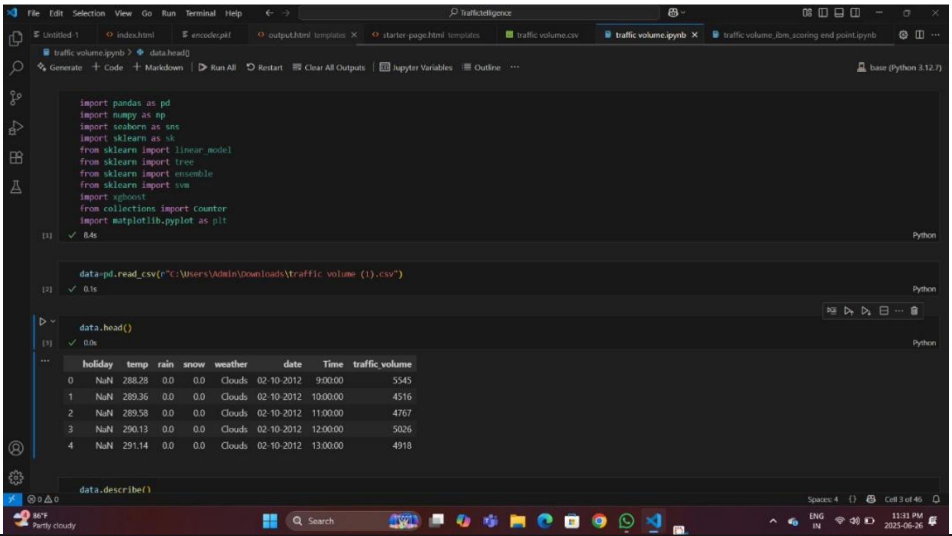
So, this research can be helpful to predict traffic. Machine learning is usually done using anaconda software but, in this paper, I have used the python program using command prompt window which is much easier than the usual way of predicting the data. In summary, the constructs of this paper consist of ten major sections. These are: Introduction, Purpose of Traffic Prediction, Problem Statement, Related Work, Overview, Methodology, Software Implementation and Conclusion with Future work.

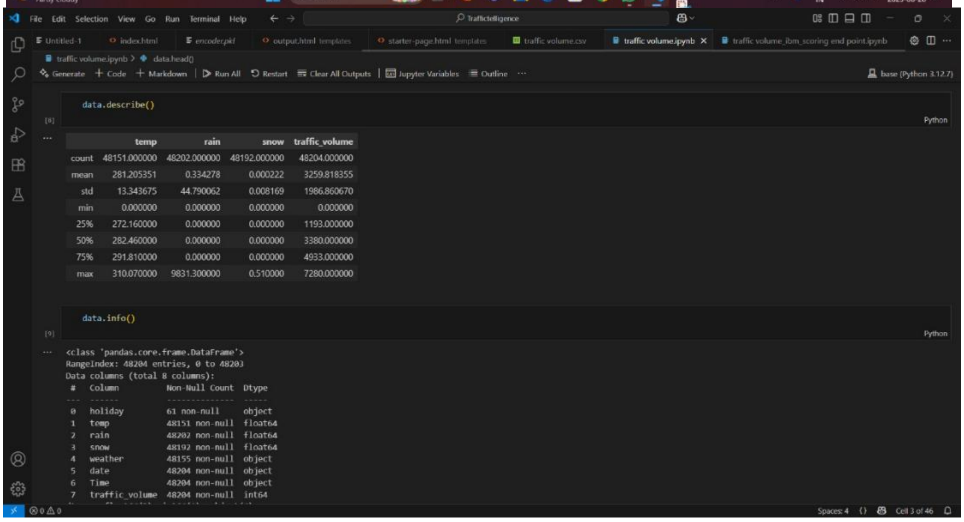
1.2Purpose of the Project

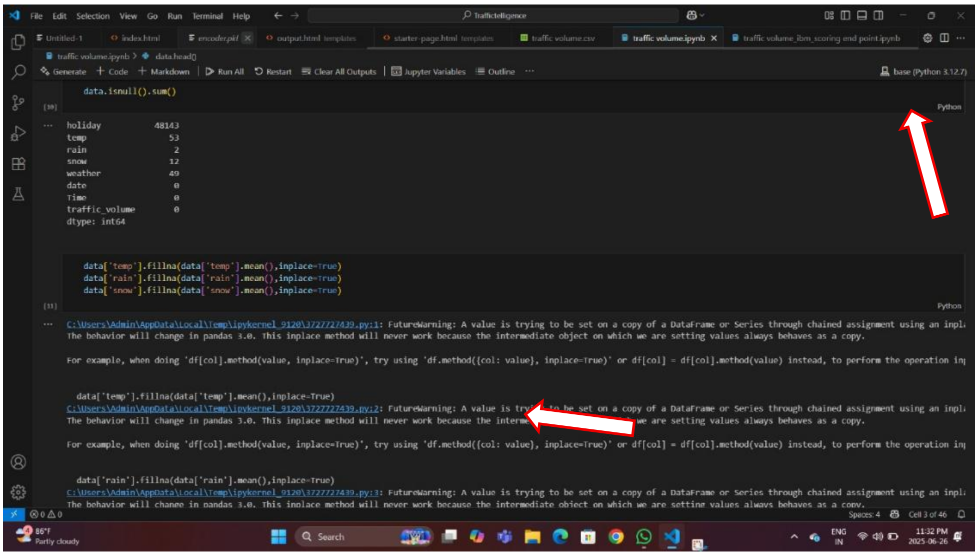
Many reports of the traffic data are of actual time but it is not favorable and ccessible to many users as we need to have prior decision in which route we need to travel. For example, During working days, we need to have daily traffic information or at times we need hourly traffic information but then the traffic congestion occurs; for solving this issue the user need to have actual time traffic prediction. Many factors are responsible forthe traffic congestion. This can be predicted by taking two datasets; one with the past year and one with the recent year's data set. If traffic is so heavy then the traffic can be predicted by referring the same time in the past year's data set and analyzing how congested the traffic would be. With the increasing cost of the fuel, the traffic congestion changes drastically. The goal of this prediction is to provide real-time gridlock and snarl up information. The traffic on the city becomes complex and are out of control these days, so such kind of systems are not sufficient for prediction. Therefore, research on traffic flow prediction plays a major role in ITS.

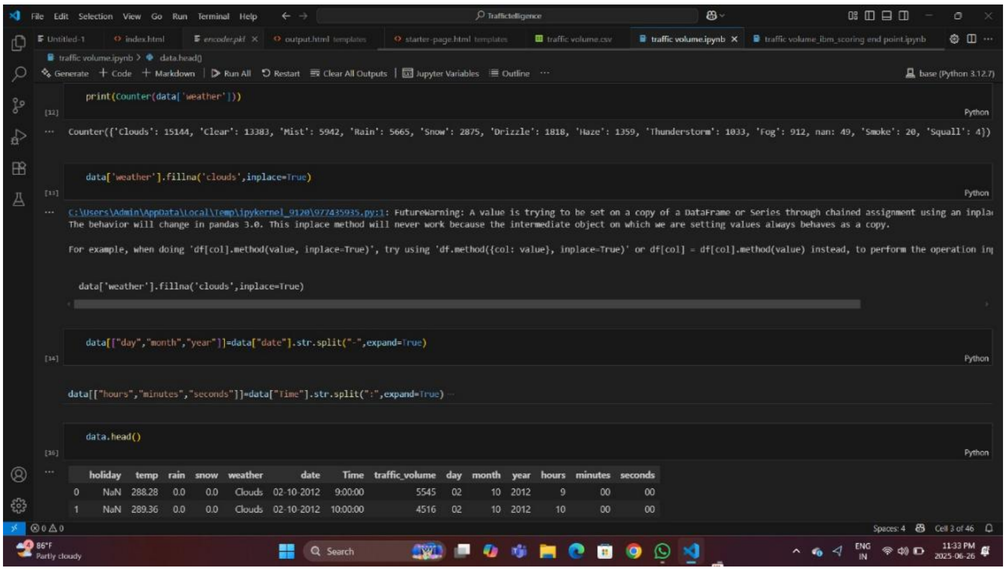
7. RESULTS

7.1 Output Screenshots

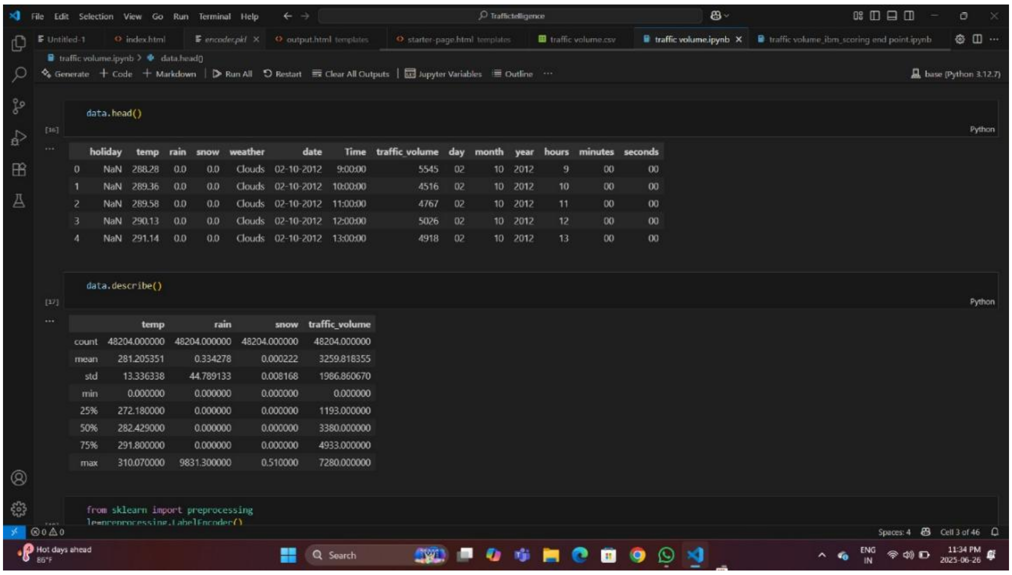




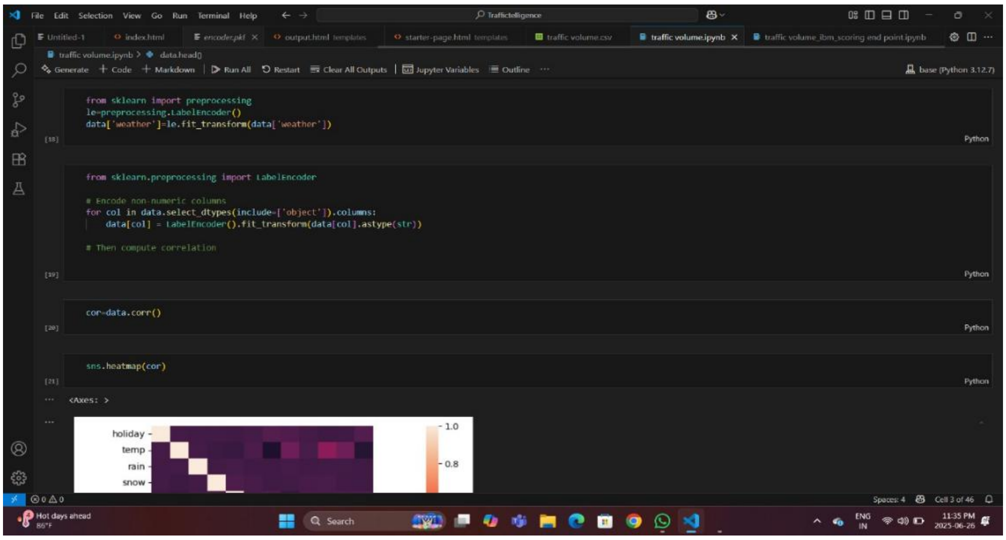




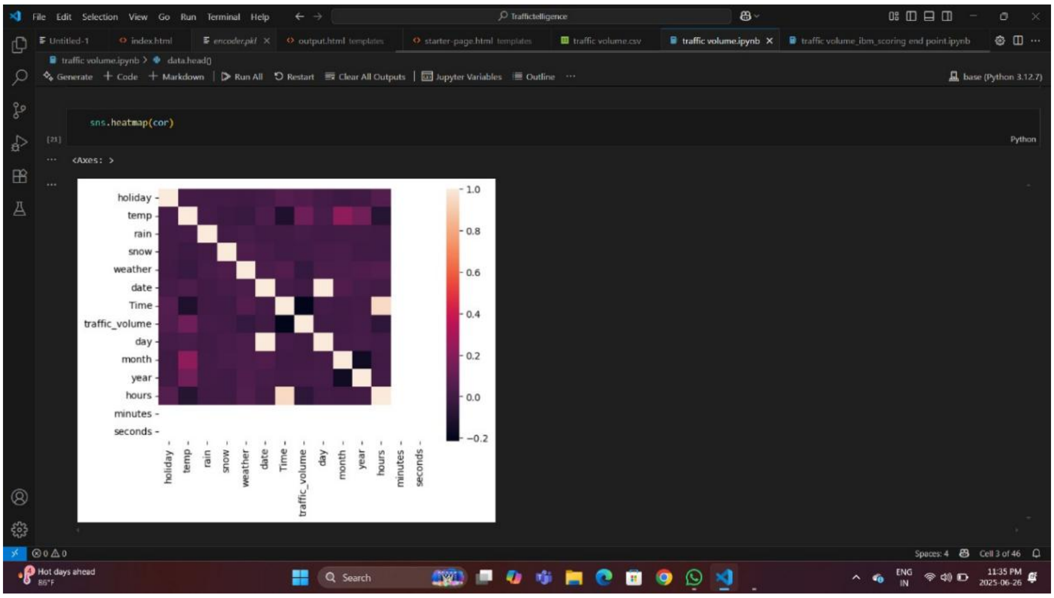
print(Counter(data['weather']))
Counter({'Clouds': 15144, 'Clear': 13383, 'Mist': 5942, 'Rain': 5665, 'Snow': 2875, 'Drizzle': 1818, 'Haze': 1359, 'Thunderstorm': 1033, 'Fog': 912, nan: 49, 'Smoke': 20, 'Squall': 4})
data['weather'].fillna('clouds',inplace=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_9120\977435935.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inpla
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation in
data['weather'].fillna('clouds',inplace=True)
data[["day","month","year"]]=data["date"].str.split("-",expand=True)
data[["hours","minutes","seconds"]]=data["Time"].str.split(":",expand=True)
data.head()
holiday temp rain snow weather date Time traffic_volume day month year hours minutes seconds
0 NaN 288.28 0.0 0.0 Clouds 02-10-2012 9:00:00 5545 02 10 2012 9 00 00
1 NaN 289.36 0.0 0.0 Clouds 02-10-2012 10:00:00 4516 02 10 2012 10 00 00



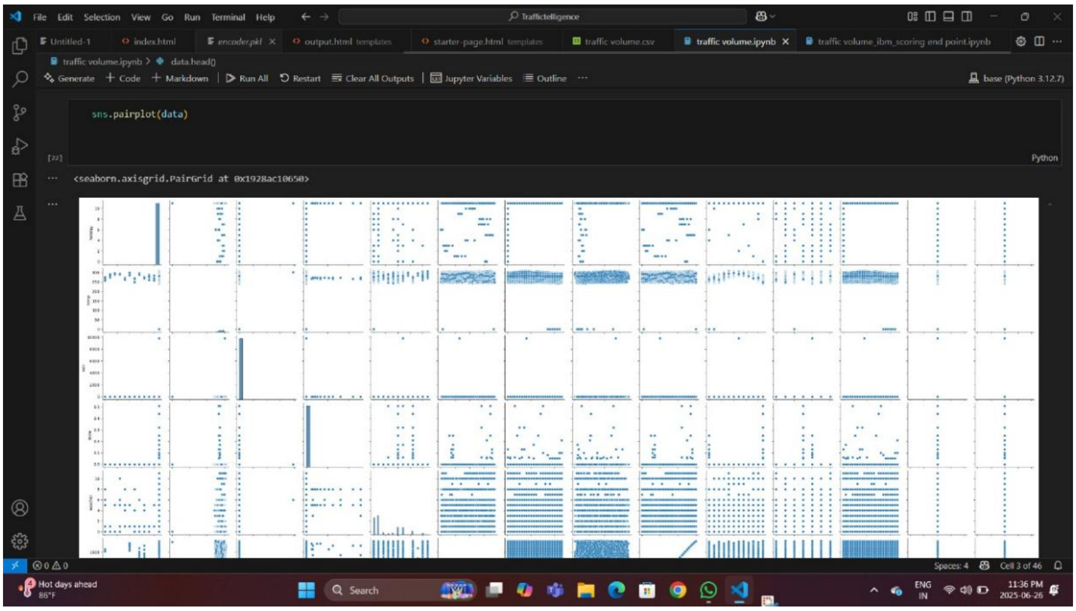
data.head()
holiday temp rain snow weather date Time traffic_volume day month year hours minutes seconds
0 NaN 288.28 0.0 0.0 Clouds 02-10-2012 9:00:00 5545 02 10 2012 9 00 00
1 NaN 289.36 0.0 0.0 Clouds 02-10-2012 10:00:00 4516 02 10 2012 10 00 00
2 NaN 289.58 0.0 0.0 Clouds 02-10-2012 11:00:00 4767 02 10 2012 11 00 00
3 NaN 290.13 0.0 0.0 Clouds 02-10-2012 12:00:00 5026 02 10 2012 12 00 00
4 NaN 291.14 0.0 0.0 Clouds 02-10-2012 13:00:00 4918 02 10 2012 13 00 00
data.describe()
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
from sklearn import preprocessing



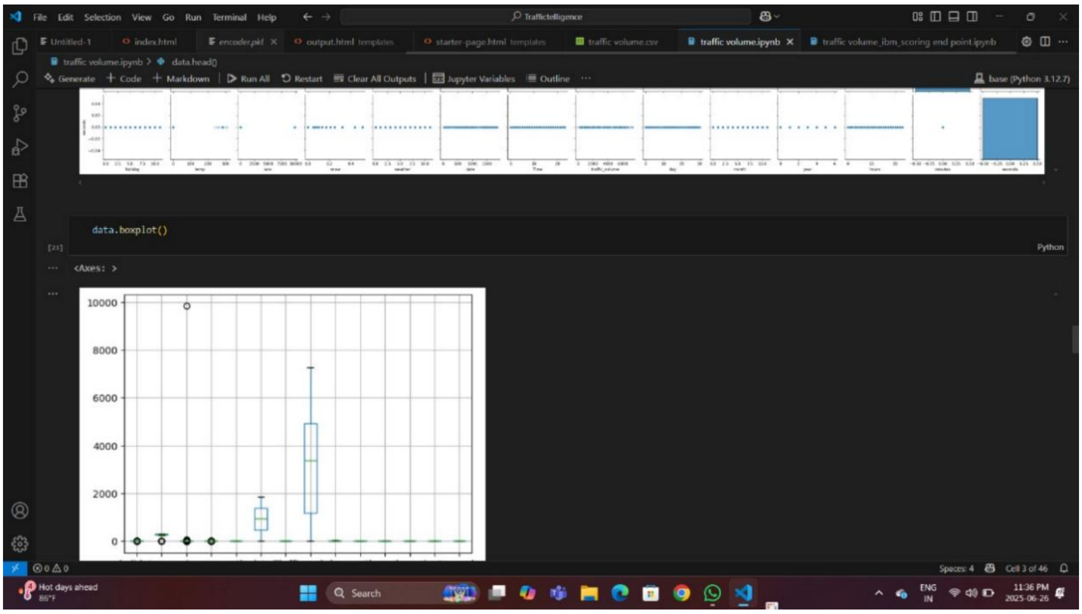
from sklearn import preprocessing
le=preprocessing.LabelEncoder()
data['weather']=le.fit_transform(data['weather'])
from sklearn.preprocessing import LabelEncoder
# Encode non-numeric columns
for col in data.select_dtypes(include=['object']).columns:
data[col] = LabelEncoder().fit_transform(data[col].astype(str))
# Then compute correlation
cor=data.corr()
sns.heatmap(cor)
<Axes: >
holiday
temp
rain
snow
1.0
0.8



traffic volume.ipynb
sns.heatmap(cor)
<Axes: >
holiday
temp
rain
snow
weather
date
Time
traffic_volume
day
month
year
hours
minutes
seconds
1.0
0.8
0.6
0.4
0.2
0.0
−0.2



traffic volume.ipynb
sns.pairplot(data)
<seaborn.axisgrid.PairGrid at 0x1928ac10650>



traffic volume.ipynb
data.boxplot()
<Axes: >
10000
8000
6000
4000
2000
0

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

```python
data.head()
```

	holiday	temp	rain	snow	weather	traffic_volume	day	month	year	hours	minutes	seconds
0	11	288.28	0.0	0.0	1	5545	1	9	0	23	0	0
1	11	289.36	0.0	0.0	1	4516	1	9	0	2	0	0
2	11	289.58	0.0	0.0	1	4767	1	9	0	3	0	0
3	11	290.13	0.0	0.0	1	5026	1	9	0	4	0	0
4	11	291.14	0.0	0.0	1	4918	1	9	0	5	0	0

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

```python
x.shape
y.shape
```

```
(48204,)
```

```python
names=x.columns
```

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

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

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

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

```python
x.head()
```

	holiday	temp	rain	snow	weather	day	month	year	hours	minutes	seconds
0	0.031687	0.530485	-0.007463	-0.027235	-0.567564	-1.574903	1.02758	-1.855294	1.638872	0.0	0.0
1	0.031687	0.611467	-0.007463	-0.027235	-0.567564	-1.574903	1.02758	-1.855294	-1.376863	0.0	0.0
2	0.031687	0.627964	-0.007463	-0.027235	-0.567564	-1.574903	1.02758	-1.855294	-1.233257	0.0	0.0
3	0.031687	0.669205	-0.007463	-0.027235	-0.567564	-1.574903	1.02758	-1.855294	-1.089650	0.0	0.0
4	0.031687	0.744939	-0.007463	-0.027235	-0.567564	-1.574903	1.02758	-1.855294	-0.946044	0.0	0.0

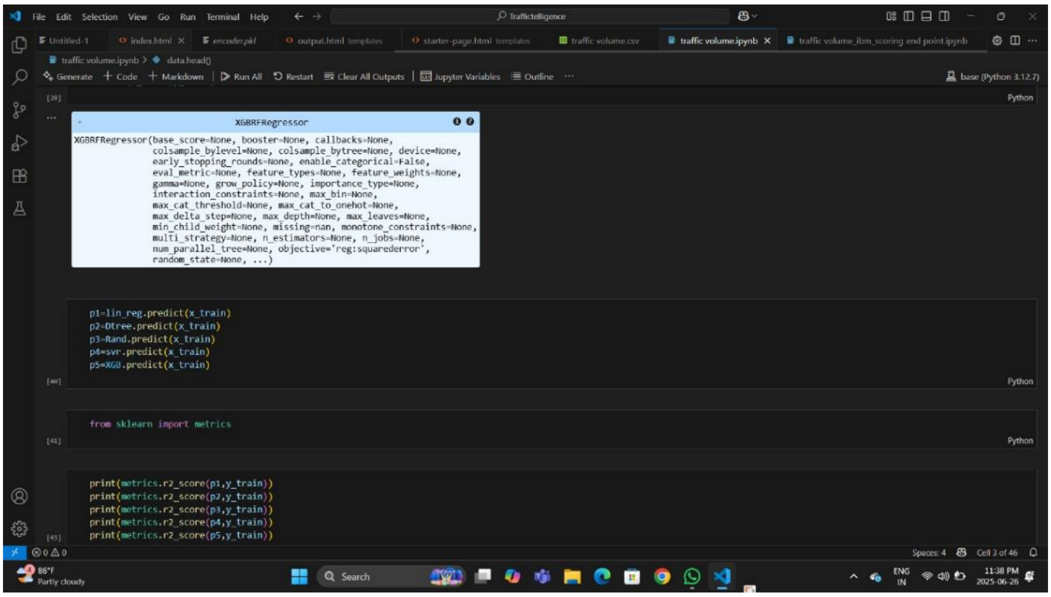
```python
from sklearn.model_selection import train_test_split
```

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

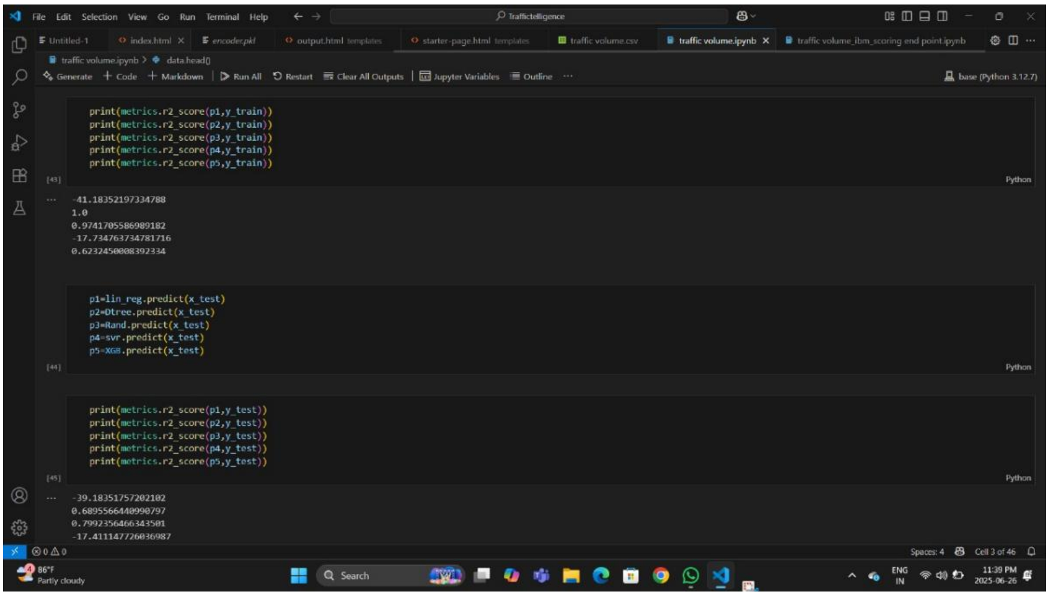
```python
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost
```

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

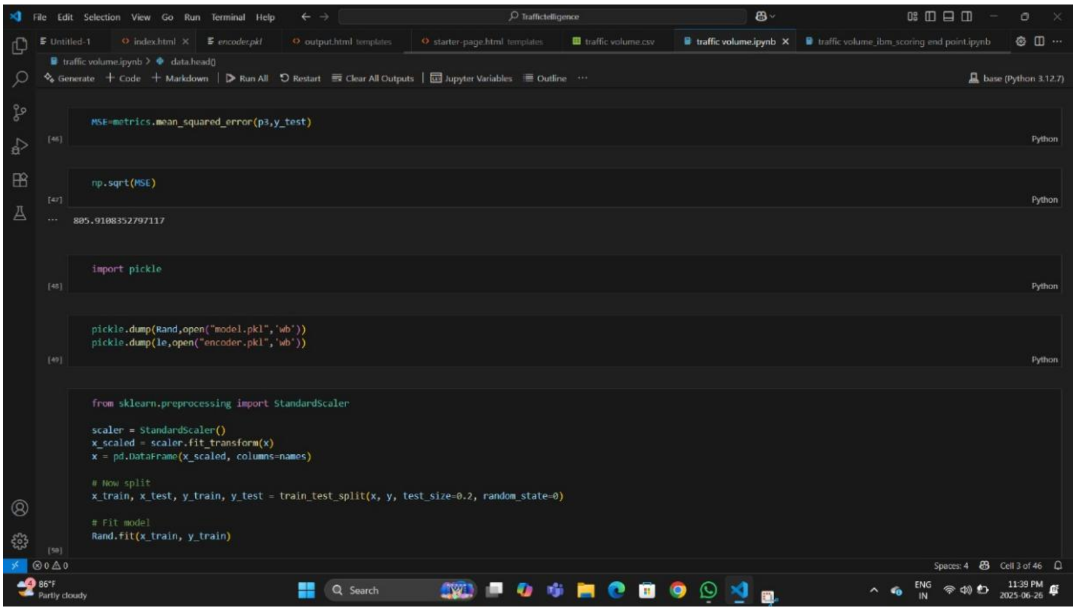
```python
lin_reg.fit(x_train,y_train)
Dtree.fit(x_train,y_train)
Rand.fit(x_train,y_train)
svr.fit(x_train,y_train)
XGB.fit(x_train,y_train)
```



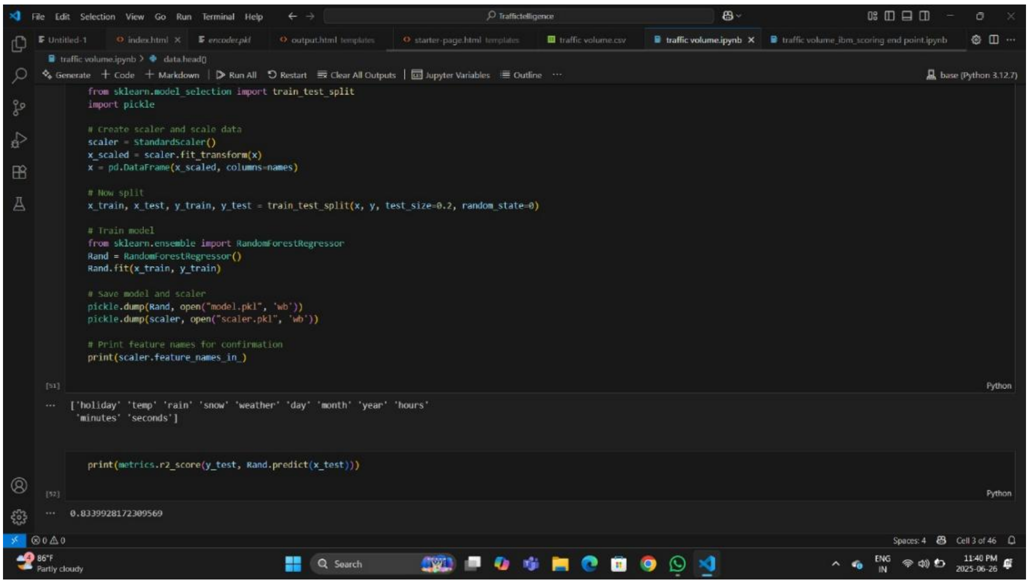
traffic volume.ipynb > data.head()
[39]
XGBRFRegressor
XGBRFRegressor(base_score=None, booster=None, callbacks=None,
colsample_bylevel=None, colsample_bytree=None, device=None,
early_stopping_rounds=None, enable_categorical=False,
eval_metric=None, feature_types=None, feature_weights=None,
gamma=None, grow_policy=None, importance_type=None,
interaction_constraints=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, objective='reg:squarederror',
random_state=None, ...)
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)
[40]
from sklearn import metrics
[41]
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))
[43]

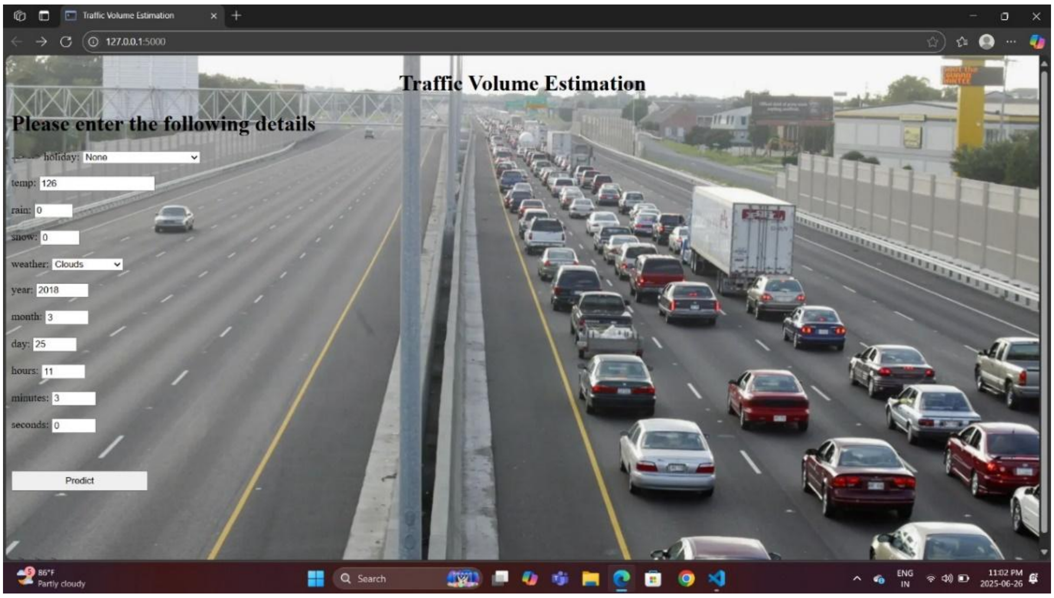


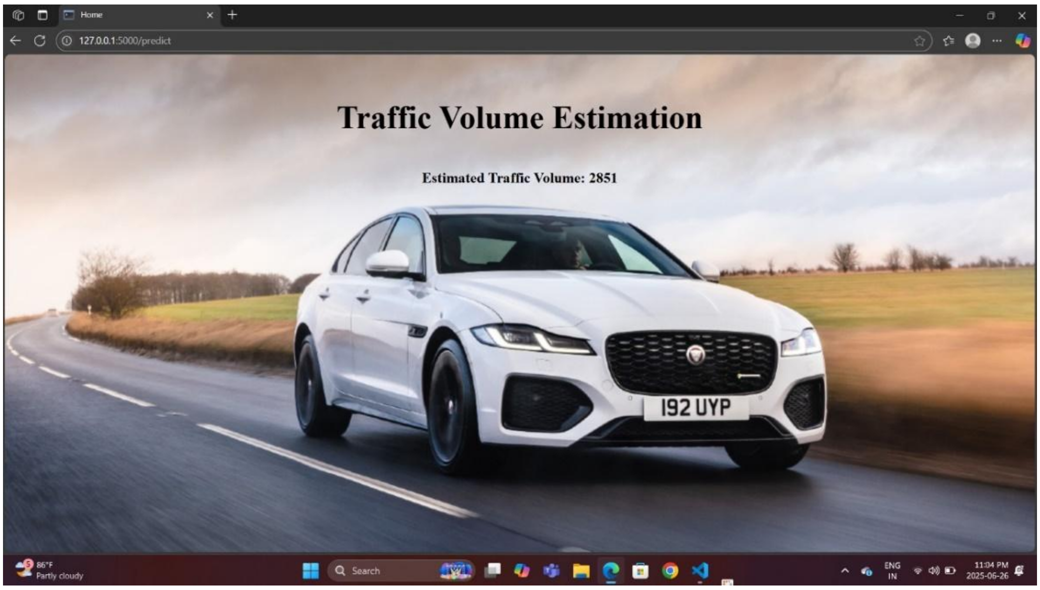
print(metrics.r2_score(p1,y_train))
print(metrics.r2_score(p2,y_train))
print(metrics.r2_score(p3,y_train))
print(metrics.r2_score(p4,y_train))
print(metrics.r2_score(p5,y_train))
[43]
-41.18352197334788
1.0
0.9741705586989182
-17.734763734781716
0.6232450008392334
p1=lin_reg.predict(x_test)
p2=Dtree.predict(x_test)
p3=Rand.predict(x_test)
p4=svr.predict(x_test)
p5=XGB.predict(x_test)
[44]
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))
[45]
-39.18351757202102
0.6895566440990797
0.7992356466343501
-17.411147726036987



MSE=metrics.mean_squared_error(p3,y_test)
[46]
np.sqrt(MSE)
[47]
805.9108352797117
import pickle
[48]
pickle.dump(Rand,open("model.pkl",'wb'))
pickle.dump(le,open("encoder.pkl",'wb'))
[49]
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)
x = pd.DataFrame(x_scaled, columns=names)
# Now split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
# Fit model
Rand.fit(x_train, y_train)
[50]


from sklearn.model_selection import train_test_split
import pickle
# Create scaler and scale data
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)
x = pd.DataFrame(x_scaled, columns=names)
# Now split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
# Train model
from sklearn.ensemble import RandomForestRegressor
Rand = RandomForestRegressor()
Rand.fit(x_train, y_train)
# Save model and scaler
pickle.dump(Rand, open("model.pkl", 'wb'))
pickle.dump(scaler, open("scaler.pkl", 'wb'))
# Print feature names for confirmation
print(scaler.feature_names_in_)
['holiday' 'temp' 'rain' 'snow' 'weather' 'day' 'month' 'year' 'hours'
'minutes' 'seconds']
print(metrics.r2_score(y_test, Rand.predict(x_test)))
0.8339928172309569


Traffic Volume Estimation
127.0.0.1:5000
Traffic Volume Estimation
Please enter the following details
holiday: None
temp: 126
rain: 0
snow: 0
weather: Clouds
year: 2018
month: 3
day: 25
hours: 11
minutes: 3
seconds: 0
Predict


Home
127.0.0.1:5000/predict
Traffic Volume Estimation
Estimated Traffic Volume: 2851

8. ADVANTAGES & DISADVANTAGES

8.1 Advantages

Advantage	Description
High Accuracy	ML models can learn complex patterns and deliver more accurate traffic volume predictions than traditional methods.
Real-Time Capability	With proper integration of GPS, IoT, and streaming data, predictions can be updated in real-time.
Scalability	The system can be easily scaled to cover different cities or road networks by retraining or fine-tuning models.
Cost-Effective Over Time	Reduces the need for expensive physical infrastructure (e.g., loop detectors, manual counting).
Data-Driven Decision Making	Authorities can use insights from the model to improve traffic signal timings, road planning, and congestion control.
Adaptability	Can be retrained or updated based on changing traffic conditions, events, or new data sources.

Table 8.1: Advantages

8.2 Disadvantages / Limitations

Disadvantage	Description
Data Dependency	Requires large volumes of quality data (e.g., GPS traces, historical traffic) to train accurate models.
Complexity in Implementation	Integrating various data sources and building models requires technical expertise.
Real-Time Infrastructure	Real-time predictions need high-speed data processing pipelines and infrastructure, which can be costly initially.
Model Interpretability	Deep learning models (e.g., LSTM, GNN) may act as black boxes, making it hard to explain predictions.
Generalization Issues	A model trained for one city or area may not perform well in another without retraining.

Table 8.2: Disadvantages

9.CONCLUSION

In the system, it has been concluded that we develop the traffic flow prediction system by using a machine learning algorithm. By using regression model, the prediction is done. The public gets the benefits such as the current situation of the traffic flow, they can also check what will be the flow of traffic on the right after one hour of the situation and they can also know how the roads are as they can know mean of the vehicles passing though a particular junction that is 4 here. The weather conditions have been changing from years to years. The cost of fuel is also playing a major role in the transportation system. Many people are not able to afford the vehicle because of the fuel cost. So, there can be many variations in the traffic data. There is one more scenario where people prefer going on their own vehicle without car-pooling, this also matters in the traffic congestion. So, this prediction can help judging the traffic flow by comparing them with these 2 years data sets. The forecasting or the prediction can help people or the users in judging the road traffic easier beforehand and even they can decide which way to go using their navigator and also this will prediction will be also helpful.

10. FUTURE SCOPE

In the future, the system are often further improved using more factors that affect traffic management using other methods like deep learning, artificial neural network, and even big data. The users can then use this technique to seek out which route would be easiest to achieve on destination. The system can help in suggesting the users with their choice of search and also it can help to find the simplest choice where traffic isn't in any crowded environment. Many forecasting methods have already been applied in road traffic jam forecasting. While there's more scope to create the congestion prediction more precise, there are more methods that give precise and accurate results from the prediction. Also, during this period, the employment of the increased available traffic data by applying the newly developed forecasting models can improve the prediction accuracy. These days, traffic prediction is extremely necessary for pretty much a part of the state and also worldwide. So, this method of prediction would be helpful in predicting the traffic before and beforehand. For better congestion prediction, the grade and accuracy are prominent in traffic prediction. within the future, the expectation are going to be the estimation of established order accuracy prediction with much easier and user-friendly methods so people would find the prediction model useful and that they won’t be wasting their time and energy to predict the information. There will be some more accessibility like weather outlook, GPS that's the road and accident-prone areas will be highlighted in order that people wouldn't prefer using the paths which aren't safe and simultaneously they'll predict the traffic. This will be done by deep learning, big data, and artificial neural networks.

11. APPENDIX

Source Code:

All codes are submitted in Git-Hub Repository

Git-Hub Repository Link:

https://github.com/Siva-Priyanka25/TrafficTelligence-Advanced-Traffic-Volume-Estimation-with-Machine-Learning

Dataset Link:

https://drive.google.com/file/d/1AAvFlqZfhti5GhotjYfoXJ6beOEbckpM/view?usp=drivesdk

Project Demo Link:

https://drive.google.com/file/d/19xI-ej1W4tuvOWGD6MYk3hHvjFtnxmjP/view?usp=drivesdk