Project Development Phase
Model Performance Test

Project Name: TrafficTelligence - Advanced Traffic Volume Estimation with Machine Learning

Team ID: 591643

Model Performance Testing:

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Accuracy Score& Classification Report -	**Linear Regression Model:**  **Decision Tree Classifier:**

```
In [46]: from sklearn.linear_model import LinearRegression

In [47]: lin=LinearRegression()

In [48]: lin.fit(x_train,y_train)

Out[48]: LinearRegression()

In [49]: pred=lin.predict(x_test)
         print(pred)

         [2772.97158299 3836.68109019 2411.26619623 ... 4344.47049723 2612.11958056
          2003.58651767]

In [50]: from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
```

```
In [53]: pd.crosstab(y_test,pred)

Out[53]:
col_0          -1446.225124  -1119.971242  1099.295312  1115.166248  1398.875233  1570.274856  1608.679140  1637.577527  1648.869377  1659.021304  ...  4661.46
traffic_volume
1                         0             0            0            0            0            0            0            0            0            0  ...
2                         0             0            0            0            0            0            0            0            0            0  ...
3                         0             0            0            0            0            0            0            0            0            0  ...
5                         0             0            0            0            0            0            0            0            0            0  ...
6                         0             0            0            0            0            0            0            0            0            0  ...
...                     ...           ...          ...          ...          ...          ...          ...          ...          ...          ...  ...   ...
7094                      0             0            0            0            0            0            0            0            0            0  ...
7107                      0             0            0            0            0            0            0            0            0            0  ...
7110                      0             0            0            0            0            0            0            0            0            0  ...
7126                      0             0            0            0            0            0            0            0            0            0  ...
7150                      0             0            0            0            0            0            0            0            0            0  ...
```

```
In [62]: from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
         mae=mean_absolute_error(y_test,pred)
         mse=mean_squared_error(y_test,pred)
         rmse=mean_squared_error(y_test,pred,squared=False)
         r2=r2_score(y_test,pred)
         print(f"Mean Absolute Error (MAE):{mae}")
         print(f"Mean Squared Error (MSE):{mse}")
         print(f"Root Mean Squared Error (RMSE): {rmse}")
         print(f"R-squared (R2):{r2}")

         Mean Absolute Error (MAE):1625.4983207588698
         Mean Squared Error (MSE):3379706.0000498504
         Root Mean Squared Error (RMSE): 1838.3976719006828
         R-squared (R2):0.13579375538526695
```

```
In [100]: from sklearn.tree import DecisionTreeClassifier

In [101]: dtc=DecisionTreeClassifier()

In [102]: dtc.fit(x_train,y_train)

Out[102]: DecisionTreeClassifier
          DecisionTreeClassifier()

In [103]: Pred=dtc.predict(x_test)
          print(Pred)

          [5721 2746  557 ... 2996  668  624]

In [104]: from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,roc_auc_score,roc_curve

In [105]: accuracy_score(y_test,Pred)

Out[105]: 0.188569650451198

In [106]: confusion_matrix(y_test,Pred)

Out[106]: array([[0, 0, 0, ..., 0, 0, 0],
                 [0, 1, 0, ..., 0, 0, 0],
                 [0, 0, 0, ..., 0, 0, 0],
                 ...,
                 [0, 0, 0, ..., 0, 0, 0],
                 [0, 0, 0, ..., 0, 0, 0],
                 [0, 0, 0, ..., 0, 0, 0]], dtype=int64)

In [107]: print(classification_report(y_test,Pred))
```

	precision	recall	f1-score	support
1	0.00	0.00	0.00	1
2	0.50	1.00	0.67	1
3	0.00	0.00	0.00	1
5	0.00	0.00	0.00	1
6	1.00	1.00	1.00	1
7	1.00	0.50	0.67	2
8	0.00	0.00	0.00	1
10	0.50	0.50	0.50	2
15	0.00	0.00	0.00	1
24	0.00	0.00	0.00	1
64	0.00	0.00	0.00	1
108	0.00	0.00	0.00	1
125	0.00	0.00	0.00	0
151	1.00	1.00	1.00	1
163	0.00	0.00	0.00	1
171	0.00	0.00	0.00	1
175	0.67	1.00	0.80	2

```
In [108]: from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
          mae=mean_absolute_error(y_test,Pred)
          mse=mean_squared_error(y_test,Pred)
          rmse=mean_squared_error(y_test,Pred,squared=False)
          r2=r2_score(y_test,Pred)
          print(f"Mean Absolute Error (MAE):{mae}")
          print(f"Mean Squared Error (MSE):{mse}")
          print(f"Root Mean Squared Error (RMSE): {rmse}")
          print(f"R-squared (R2):{r2}")

          Mean Absolute Error (MAE):592.5111502956125
          Mean Squared Error (MSE):1243077.9226221347
          Root Mean Squared Error (RMSE): 1114.934044068139
          R-squared (R2):0.6821393034610368
```

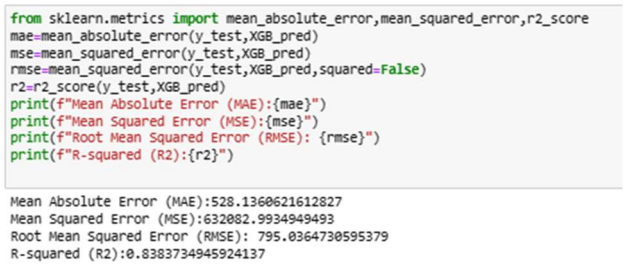
XGBoost classifier:

```
import xgboost
XGB=xgboost.XGBRegressor()
```

```
XGB.fit(x_train,y_train)
```

```
XGBRegressor(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, ...)
```

```
XGB_pred=XGB.predict(x_test)
```

			
2.	Tune the Model	Hyperparameter Tuning Validation Method -	Since the accuracy has reached up to 97.6% for randomforest model we ignored the tuning.