INTELLIGENT TRAFFIC PREDICTION AND PATH OPTIMIZATION USING ML

A Machine Learning Approach For Efficient Traffic Management

Under Supervision of Dr. Rinky Sha | Assistant Professor | IIIT Kalyani

CONTENTS

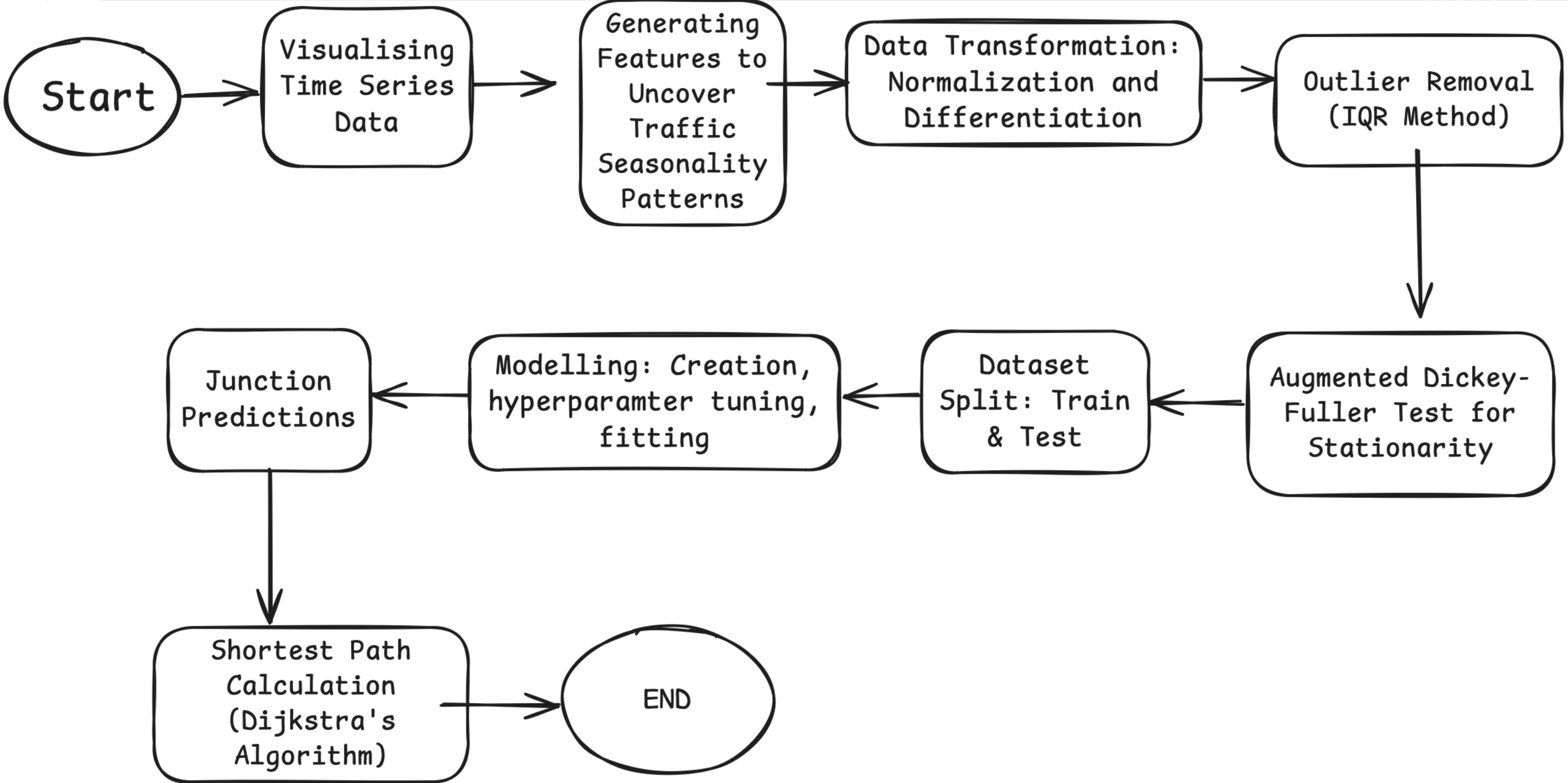
OBJECTIVE

Utilize ML to forecast congestion levels based on real-time and historical data, helping manage peak traffic periods effectively.

Develop a system to recommend the quickest routes based on current conditions, reducing travel time and fuel use.

Support smarter urban traffic management and planning, enhancing sustainability and easing congestion in cities.

WORKFLOW


Start
Visualising Time Series Data
Generating Features to Uncover Traffic Seasonality Patterns
Data Transformation: Normalization and Differentiation
Outlier Removal (IQR Method)
Augmented Dickey-Fuller Test for Stationarity
Dataset Split: Train & Test
Modelling: Creation, hyperparamter tuning, fitting
Junction Predictions
Shortest Path Calculation (Dijkstra's Algorithm)
END

METHODOLOGY

01

FEATURE ENGINEERING

Plotting of time series data and creation of new features for seasonality insights

02

DATA PREPROCESSING

Normalise features, Apply differentiation for seasonality removal, Remove Outliers (IQR Method)

03

STATIONARITY CHECK

Perform Augmented Dicky Fuller Test to check stationarity

04

MODELLING

Splitting the dataset: Train & Test; Creating the model; Hyperparameter Tuning; Fitting the Model

05

PATHFINDING

Perform predictions for each junction and finding the shortest path between them using Dijkstra's Algorithm

FEATURE ENGINEERING & STANDARDIZATION

- **Normalization:** Standardizes data by centering it at mean 0 and scaling to standard deviation 1, aiding pattern recognition across varied feature scales.

  Z = (Feature Value - Mean of Column)/Standard Deviation of Column
  df_normalized = (df[col] - average) / stdev

- **Differencing:** Removes trends in time series, enhancing stationarity by highlighting short-term fluctuations for improved prediction accuracy.

  Diffi=Xi−Xi−interval
  value = df[col][i] - df[col][i - interval]

FEATURE ENGINEERING & STANDARDIZATION

- **Time Series Feature Preparation (TnF):** Generates input-output pairs from data windows, enabling to capture temporal dependencies for forecasting or classification.

$X=[Xi-steps,Xi-steps+1,...,Xi-1], Y=Xi$

```
for i in range(steps, end_len):
    X.append(df[i - steps:i, 0])
    Y.append(df[i, 0])
```

- **Reshaping (FeatureFixShape):** This reshaping converts a 2D array into a 3D format (samples, timesteps, features), to process sequential data correctly.

$Reshape(X) \rightarrow (X.shape[0], X.shape[1], 1)$

```
train = np.reshape(train, (train.shape[0], train.shape[1], 1))
```

FEATURE ENGINEERING

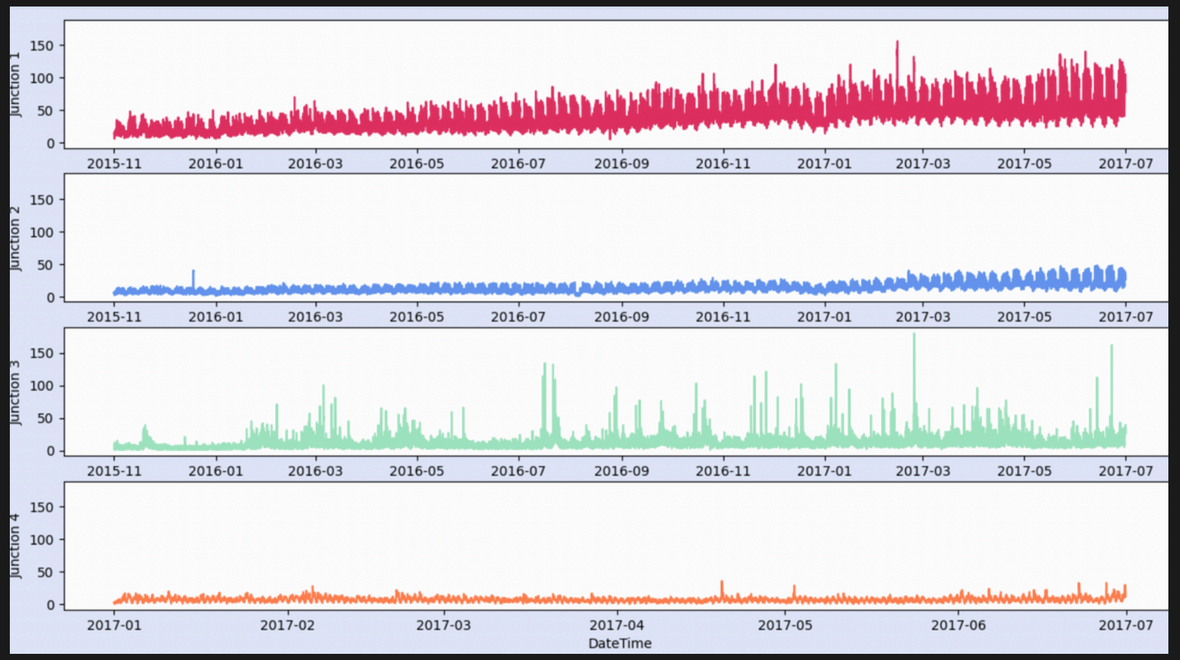
	DateTime	Junction	Vehicles	ID
0	2015-11-01 00:00:00	1	15	20151101001
1	2015-11-01 01:00:00	1	13	20151101011
2	2015-11-01 02:00:00	1	10	20151101021
3	2015-11-01 03:00:00	1	7	20151101031
4	2015-11-01 04:00:00	1	9	20151101041
...	...	...	...	...
48115	2017-06-30 19:00:00	4	11	20170630194
48116	2017-06-30 20:00:00	4	30	20170630204
48117	2017-06-30 21:00:00	4	16	20170630214
48118	2017-06-30 22:00:00	4	22	20170630224
48119	2017-06-30 23:00:00	4	12	20170630234

48120 rows × 4 columns

	DateTime	Junction	Vehicles	Year	Month	Date_month	Hour	Day
0	2015-11-01 00:00:00	1	15	2015	11	1	0	Sunday
1	2015-11-01 01:00:00	1	13	2015	11	1	1	Sunday
2	2015-11-01 02:00:00	1	10	2015	11	1	2	Sunday
3	2015-11-01 03:00:00	1	7	2015	11	1	3	Sunday
4	2015-11-01 04:00:00	1	9	2015	11	1	4	Sunday
...	...	...	...	...	...	...	...	...
48115	2017-06-30 19:00:00	4	11	2017	6	30	19	Friday
48116	2017-06-30 20:00:00	4	30	2017	6	30	20	Friday
48117	2017-06-30 21:00:00	4	16	2017	6	30	21	Friday
48118	2017-06-30 22:00:00	4	22	2017	6	30	22	Friday
48119	2017-06-30 23:00:00	4	12	2017	6	30	23	Friday

48120 rows × 8 columns

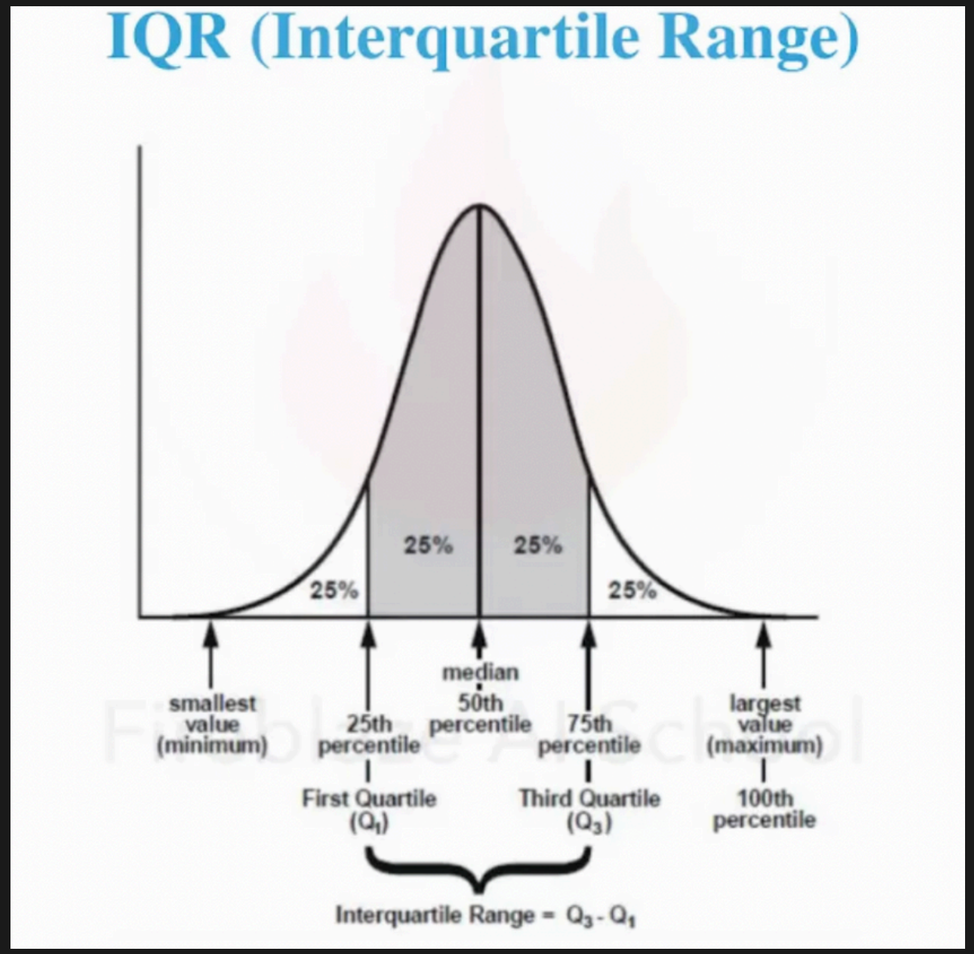
DATA TRANSFORMATION



before

after

REMOVING OUTLIERS: IQR METHOD



- **IQR Calculation:** Computes the interquartile range (IQR) to identify data spread in the middle 50%.
- **Bound Setting:** Defines upper and lower bounds based on a threshold (usually 1.5 times IQR).
- **Outlier Capping:** Replace outliers with boundary values to keep data within range.
- Repeated for Each Dataset: Apply this process across all datasets and intervals for consistency.

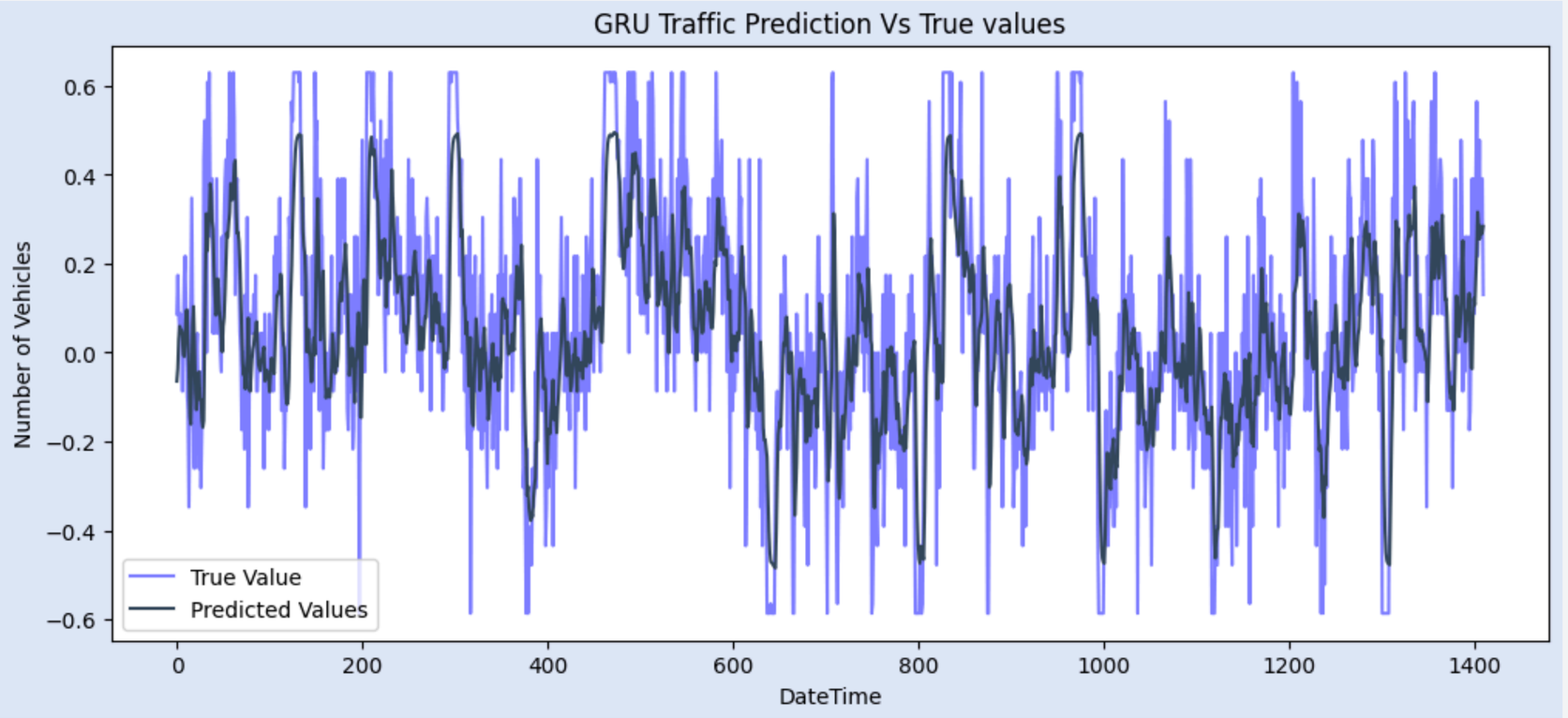
AUGMENTED DICKY-FULLER TEST

```
Checking the transformed series for stationarity:


ADF Statistic: -15.265303390415434
p-value: 4.798539876396819e-28
Critical values:
        1%: -3.431
        5%: -2.862
        10%: -2.567
Time series is stationary


ADF Statistic: -21.795891026940144
p-value: 0.0
Critical values:
        1%: -3.431
        5%: -2.862
        10%: -2.567
Time series is stationary


ADF Statistic: -28.001759908832977
p-value: 0.0
Critical values:
        1%: -3.431
        5%: -2.862
        10%: -2.567
Time series is stationary


ADF Statistic: -17.979092563052305
p-value: 2.7787875325953405e-30
Critical values:
        1%: -3.432
        5%: -2.862
        10%: -2.567
Time series is stationary
```

- **ADF Statistic:** Low ADF statistics (e.g., -15.27, -21.80) indicate strong stationarity.
- **p-value:** Near-zero p-values confirm rejection of the null hypothesis, supporting stationarity.
- **Critical Values:** ADF statistics are below the 1% critical level, reinforcing stationarity.
- **Conclusion:** All series are stationary after differencing, making the data suitable for time-series modeling.

RESULTS

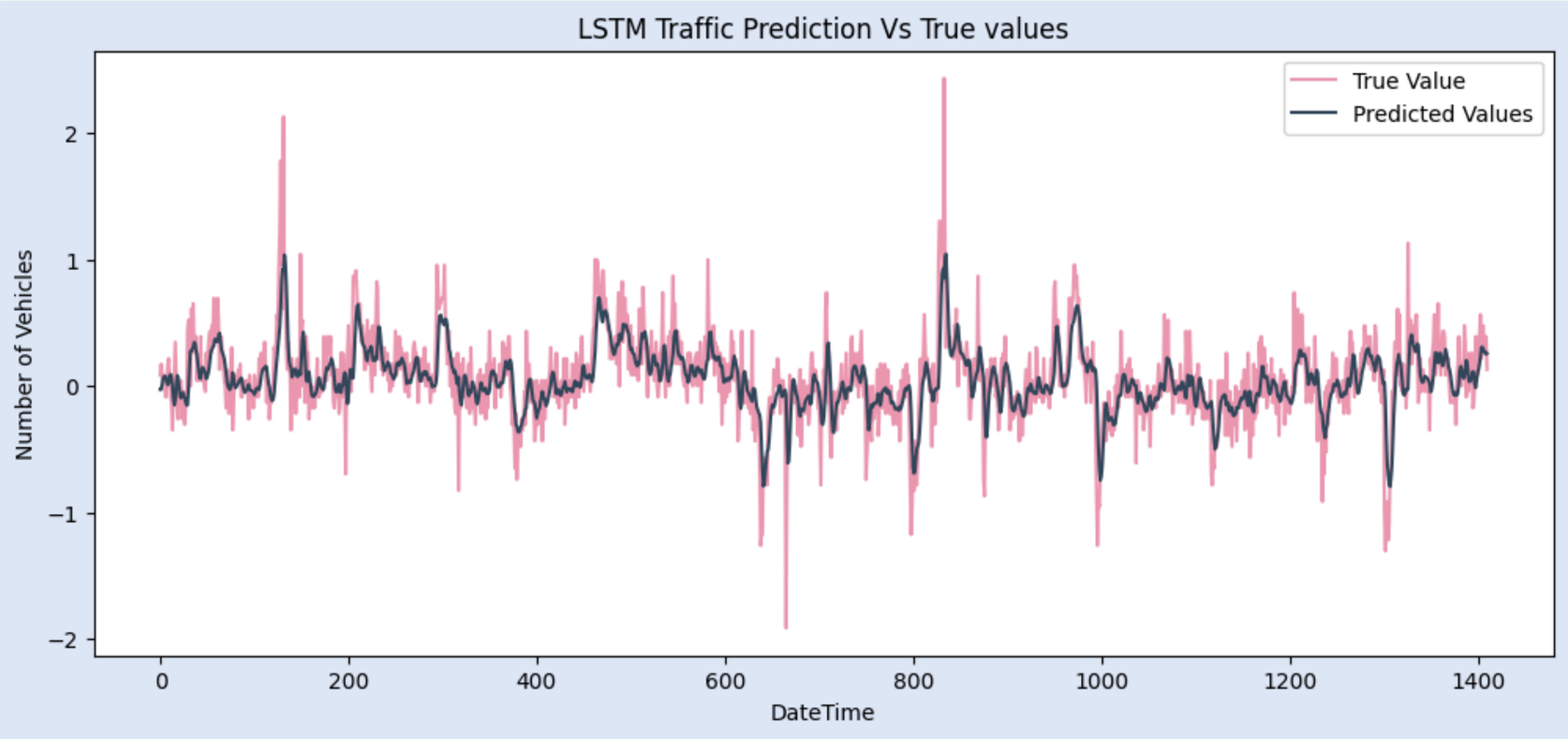
GRU MODEL



- **GRU Model Setup:** Builds a multi-layer GRU model with dropout for traffic prediction.
- **Training and Prediction:** Trains the model with SGD and early stopping, then predicts test values.
- **Evaluation:** Assesses accuracy with RMSE and visualizes true vs. predicted values.

RESULTS

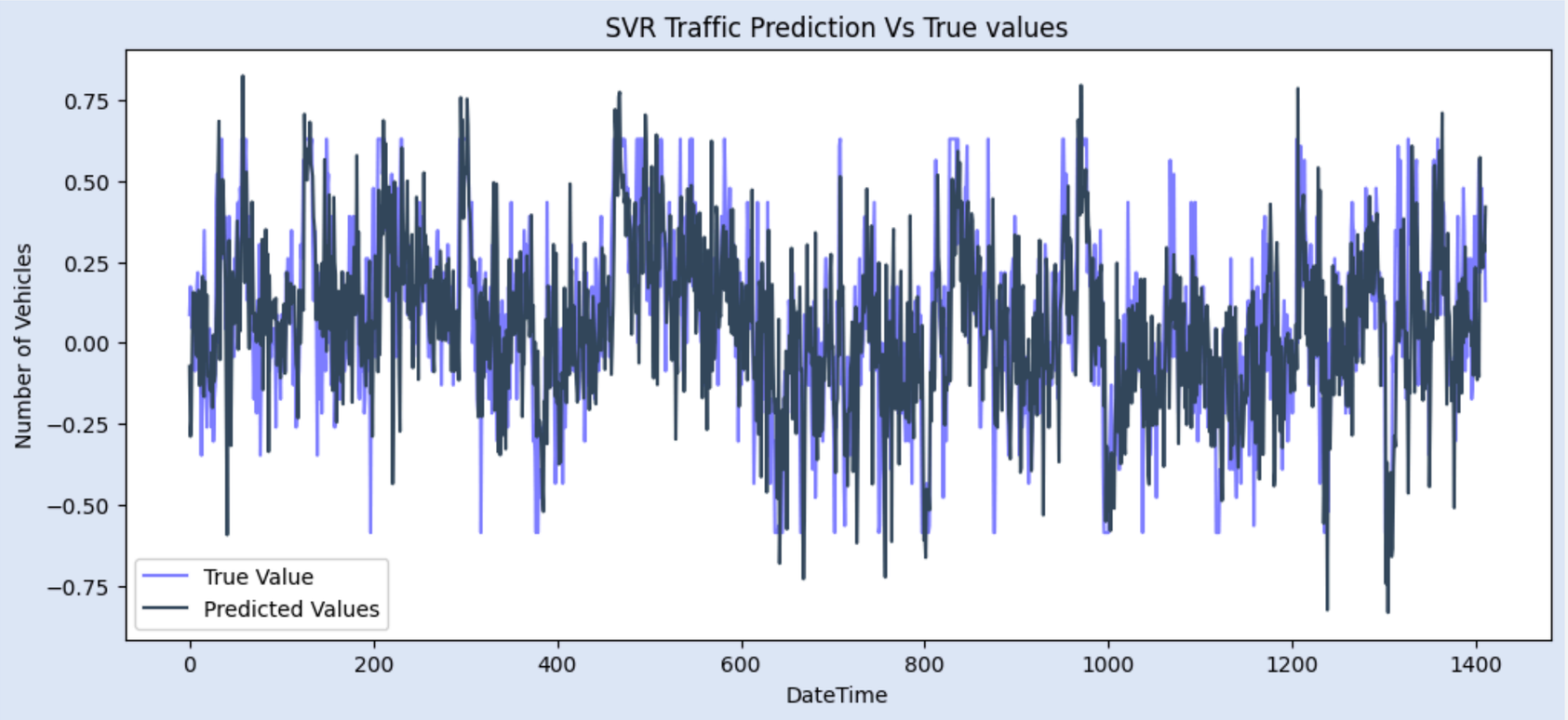
LSTM MODEL



- **LSTM Model Structure:** Creates an LSTM model with three layers and dropout for traffic prediction, optimized with early stopping.
- **Training and Prediction:** The model, using SGD, is trained on data and then used to predict test traffic values.
- **Evaluation and Visualization:** The model's accuracy is assessed with RMSE, and results are visualized by comparing actual vs. predicted traffic.

RESULTS

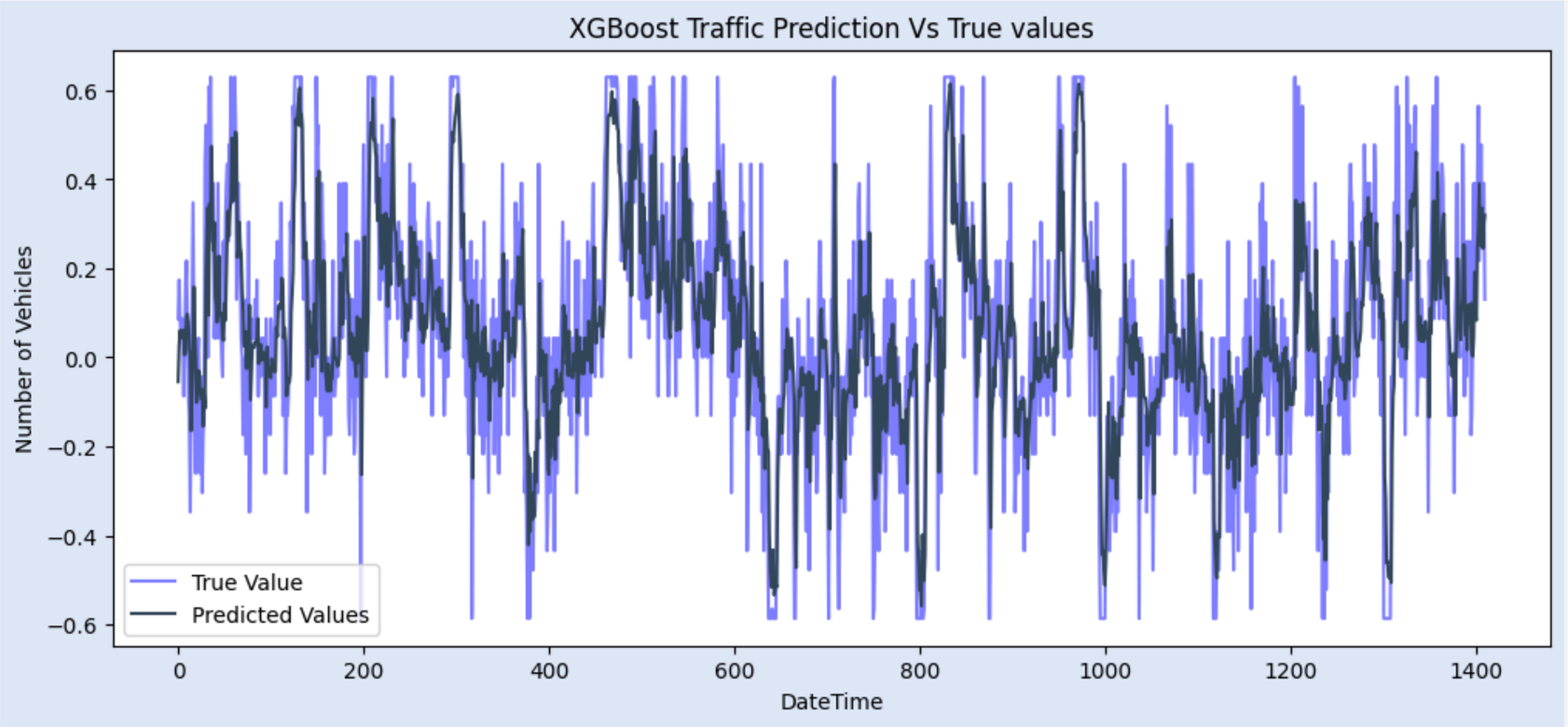
SVR MODEL



- **SVR Model:** Defines an SVR model with an RBF kernel for predicting traffic, flattening the input data as required.
- **Training and Prediction:** Trains the model on the training set and makes predictions on the test set.
- **Evaluation and Visualization:** Calculates the RMSE for accuracy assessment and plots actual vs. predicted values to visually compare the results.

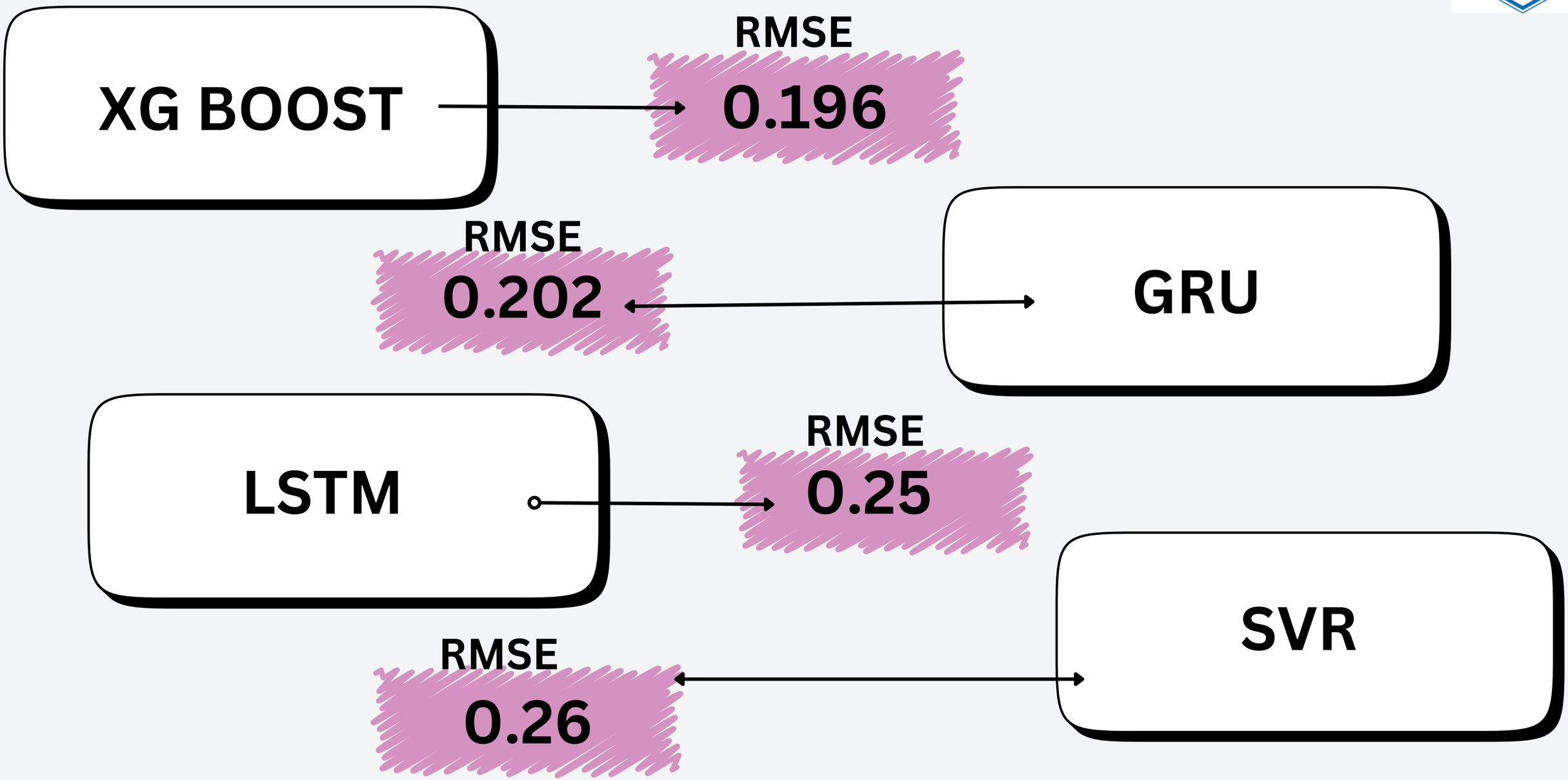
RESULTS

XGBoost MODEL

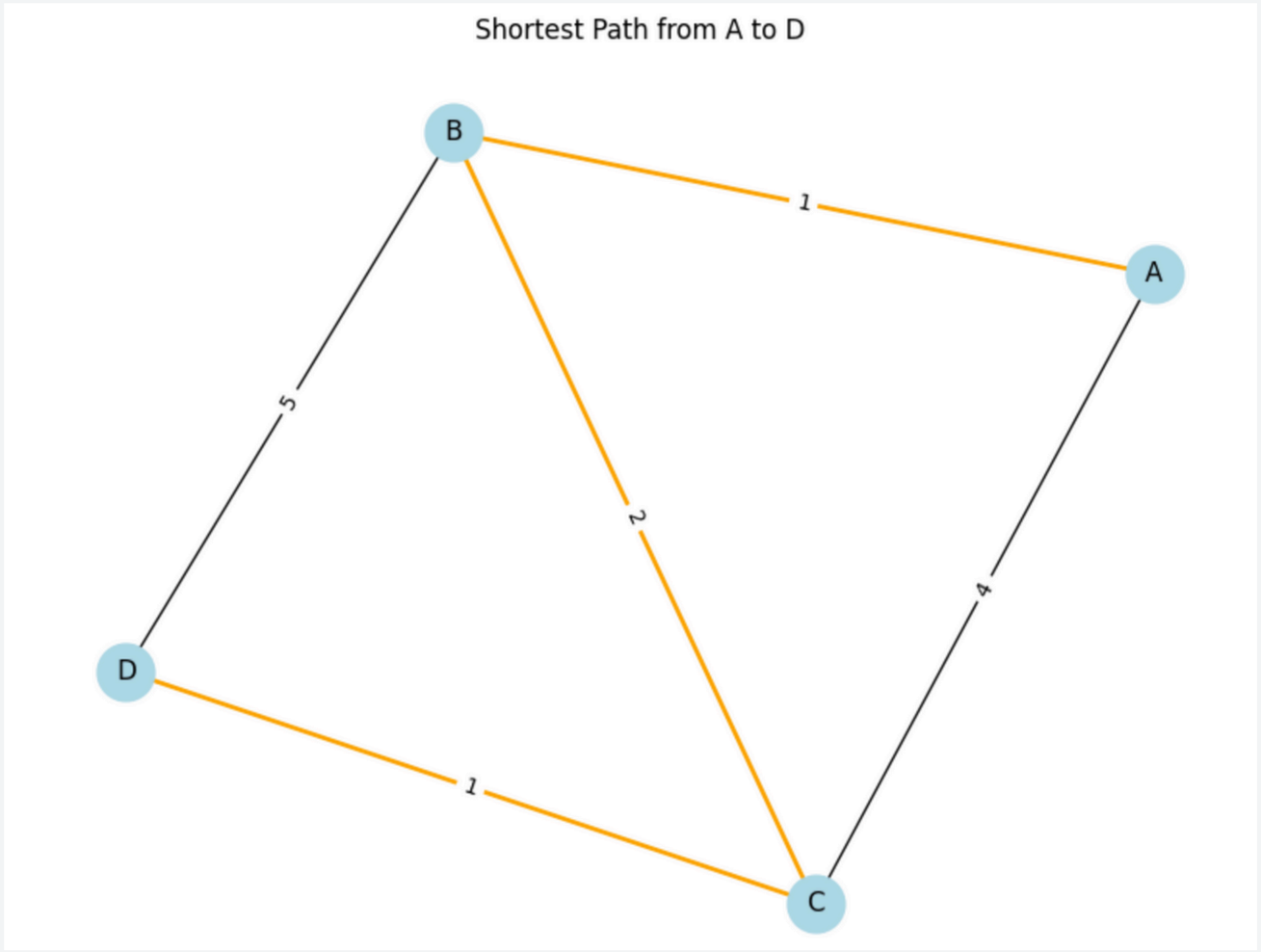


- **Model Training and Optimization:** The XGBoost regressor is used to predict traffic, with optional hyperparameter tuning via GridSearchCV for improved performance.
- **Performance Measurement:** RMSE is computed to evaluate the accuracy of predictions against true values.
- **Visualization of Predictions:** A comparative plot illustrates the relationship between true traffic data and model predictions for trend analysis.

MODELS COMPARISON



FINDING OPTIMAL PATH



By Dijkstra's Algorithm: -

INPUT:

```
Enter the start node: A
Enter the destination node: D
```

OUTPUT:

```
Shortest distance from A to D: 5.6
Path: A -> B -> C -> D
```

CONCLUSION

- ***Model Comparison:*** *GRU, LSTM, SVR, and XGBoost were used for traffic prediction, with XGBoost and GRU showing higher accuracy due to their temporal modeling capabilities.*

- ***Traffic Optimization:*** *These models support dynamic traffic management by predicting congestion and suggesting alternative routes for smoother traffic flow.*

- ***Real-Time Integration:*** *Incorporating live traffic data from IoT sensors could improve adaptability and accuracy in real-world applications.*

- ***Ensemble Modeling:*** *Using ensemble techniques with GRU, LSTM, SVR, and XGBoost could enhance prediction accuracy and scalability for complex traffic patterns.*

REFERENCES

- *Zhu, X., Xu, X., He, H., & Li, J. (2016). "Traffic Flow Prediction with Improved Machine Learning Algorithms in a Large-Scale Traffic Network." IEEE Transactions on Intelligent Transportation Systems, 17(4), 1-8.*

- *Sun, S., & Zhang, C. (2018). "Application of a Gradient Boosting Decision Tree Algorithm in Traffic Speed Prediction and Optimization." Journal of Intelligent & Fuzzy Systems, 34(3), 1735-1744.*

- *Van Lint, J. W. C., & Van Hinsbergen, C. P. I. (2012). "Short-Term Traffic and Travel Time Prediction Models." IEEE Transactions on Intelligent Transportation Systems, 13(1), 60-69.*

OUR TEAM

Nalla Vamsi

ECE/21135/795

Sonam Atta

CSE/21085/745

Nilotpal Sarkar

ECE/21138/798