Traffic Flow Prediction and Management

Arpan Verma (2022105), Aditya Upadhyay(2022040)
Department of Computer Science Engineering
INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY
New Delhi, INDIA
Email: arpan22105@iiitd.ac.in aditya22040@iiitd.ac.in

***Abstract*—Traffic congestion is a persistent challenge to urban mobility. Ineffective traffic flow causes commuters to spend time, use more fuel, and become frustrated. This research explores state-of-the-art techniques and data-driven tactics in the field of traffic flow prediction and management. We may proactively put solutions into place to reduce congestion, improve the timing of traffic lights, and improve the general effectiveness of transportation systems by anticipating future patterns in traffic. This paper looks at real-time management strategies, how machine learning improves traffic prediction, and how it might help cities become greener and more efficient in the future.**

I. Introduction

Traffic congestion is a significant issue in urban areas, leading to wasted time, increased pollution, and economic losses. Effective traffic flow prediction and management are essential for alleviating congestion and improving transportation efficiency.

II. Methodology

A. Preprocessing

For preprocessing of data we have done Normalization by subtracting Mean and dividing by square root of variance. There we also plotted for each feature available the corresponding elements this Step is done to visualize thse data samples there corresponding Need in Dataset.

B. Model Used as First

First we simply used Linear Regression as it is most simple to implement it is very useful in data where our training set is not much complex. This was just start try built from Scratch the equatation following for gradient descendent is : -

$$\frac{1}{n_{samples}} \cdot np.dot(X.T`(y_{pred} - y))$$

To reduce Mean square error Normalized the Function
df_normalized $\leftarrow \frac{df\_numerical - mean(df\_numerical)}{std(df\_numerical)}$

Here $df\_numerical$ **are those columns that can be normalized.**

Preprocessing: Handling NULL Values and feature extraction

data.fillna(0)

Explanation: This step fills any missing values in the dataset with 0.

Feature Engineering: Extract DateTime features

data['DateTime'] $\leftarrow pd.to_datetime(data['DateTime'])$
data['Hour'] $\leftarrow data['DateTime'].dt.hour$
data['DayOfWeek'] $\leftarrow data['DateTime'].dt.dayofweek$
data['Month'] $\leftarrow data['DateTime'].dt.month$
data['Year'] $\leftarrow data['DateTime'].dt.year$
data["Date"] $\leftarrow data['DateTime'].dt.day$

Explanation: These steps extract various date and time components from the 'DateTime' column and create new features such as hour, day of week, month, year, and date.

Feature Engineering: Interaction Features

data['HourDayOfWeek'] $\leftarrow data['Hour'] \times data['DayOfWeek']$

Explanation: This step creates an interaction feature by multiplying the hour and day of week features.

Feature Engineering: Junction Features

data['Overall_Traffic'] $\leftarrow data[['Junction\_1',' Junction\_2',' Junctio$

Explanation: This step aggregates the traffic situation across all junctions to create a new feature representing overall traffic, which introduces some noise to prevent overfitting.

Feature Engineering: Temporal Features

data['Trend'] $\leftarrow data.index$

Explanation: This step creates a trend feature based on the index of the data.

Display the updated DataFrame with engineered features

C. Model 2 : Random Forest

The Random Forest Regressor model was trained and evaluated on two distinct datasets: one with normalized features and another with unnormalized features. This evaluation aimed to assess the impact of data normalization on model performance. The selected features for training and testing included 'Junction', 'Hour', 'DayOfWeek', and 'ID'. After fitting the regressor to the original training data, predictions were made on the corresponding test data for both normalized and unnormalized cases. Subsequently, mean squared error (MSE) was computed to quantify the prediction accuracy of the model.

Interestingly, the results revealed a notable improvement in model performance when utilizing normalized data. The MSE obtained from predictions made on the normalized dataset was considerably lower compared to the unnormalized counterpart. This observation underscores the significance of data preprocessing techniques, such as normalization, in enhancing the effectiveness of machine learning models.

Furthermore, the process involved experimentation with different parameter combinations to optimize model performance. By systematically evaluating various parameter settings, including the number of trees in the random forest and the maximum depth of each tree, the configuration yielding the lowest MSE and the best fit in y_pred was identified. This rigorous parameter tuning process ensured that the model was fine-tuned to capture the underlying patterns in the data and produce accurate predictions.

To illustrate the impact of normalization and showcase the performance improvement achieved, a graphical representation of the MSE comparison between the normalized and unnormalized cases was generated. This visualization provided a clear depiction of the benefits conferred by data normalization, further reinforcing its importance in machine learning workflows.

In summary, the Random Forest Regressor exhibited promising performance in predicting the target variable. Through meticulous experimentation, parameter optimization, and data preprocessing, the model was effectively trained to accurately capture the relationships between the input features and the target variable, thereby demonstrating its suitability for the task at hand.

III. Comparison of Traffic Flow Prediction and Management Approaches

This section compares different approaches to traffic flow prediction and management, evaluating their effectiveness and applicability in various scenarios.

A. Model Comparison Linear

We experimented with several models to predict traffic flow and manage traffic effectively. One of the models we tested was the Linear Regressor, which we implemented ourselves. However, this model performed poorly, especially considering the complexity of the dataset.

B. Gradient Boosting

After observing the poor performance of the Linear Regressor, we explored Gradient Boosting as an alternative approach. While the findings were marginally better than those of the Linear Regressor, they still fell short of our expectations. We realized that the Linear Regressor struggled to separate the dataset into distinct parts by a straight line, leading us to shift our focus towards another regressor algorithm.

C. Random Forest

Random Forest emerged as a promising solution, providing satisfactory predictions for traffic flow. However, achieving these results required careful parameter tuning. We systematically tested various combinations of features to optimize model performance. Notably, the inclusion of 'Junction' and 'ID' features significantly improved prediction accuracy.

Upon closer examination, we discovered that the 'ID' feature played a crucial role in predicting traffic congestion. By analyzing the dataset, we observed that the 'ID' column contained dates and binary digits appended as numbers. Leveraging this insight, the model effectively predicted congestion patterns based on specific days. Disregarding the 'ID' feature resulted in inferior predictions, highlighting its importance in the modeling process.

While Random Forest successfully detected and predicted traffic congestion, there remains room for improvement. Despite its effectiveness, further enhancements are needed to address certain limitations and refine the model's performance.

D. Gradient Boosting

The Gradient Boosting Regressor algorithm, a powerful ensemble learning technique, is employed to build the regression model. This model is trained on the training data and subsequently used to make predictions on the testing data. The mean squared error (MSE) metric is calculated to assess the model's performance, representing the average squared difference between the predicted and actual number of vehicles. The goal of this code is to minimize the MSE, indicative of better predictive accuracy. By iteratively adjusting model hyperparameters, conducting feature engineering, and evaluating various algorithms, the aim is to refine the model and ultimately enhance its predictive capabilities for traffic flow prediction.

IV. Findings

A. Traffic Flow Prediction Results

B. Data Visualization and Junction Pie Chart

Despite efforts to gain insights into traffic patterns through data visualization, the results did not reveal significant trends or patterns. Figure 1 shows the visualization of traffic distribution across different junctions, which unfortunately did not yield actionable insights. Similarly, Figure 2 presents the percentage distribution of traffic among junctions using a pie chart, with no discernible patterns or trends observed.

C. Random Forest Regression Results

The evaluation of the Random Forest Regression model yielded mixed results. Figure 4 presents the prediction results obtained using the Random Forest Regression model.

D. Linear Regression Results

The evaluation of the Linear Regression model produced disappointing results. Figure 5 presents the prediction results obtained using the Linear Regression model, which exhibited poor performance in accurately predicting traffic flow. Despite the initial optimism, the model's predictions were inconsistent and unreliable.

E. Gradient Boosting Results

The evaluation of the Gradient Boosting model yielded mixed results. Figure 6 illustrates the prediction results obtained using the Gradient Boosting model.

0.48

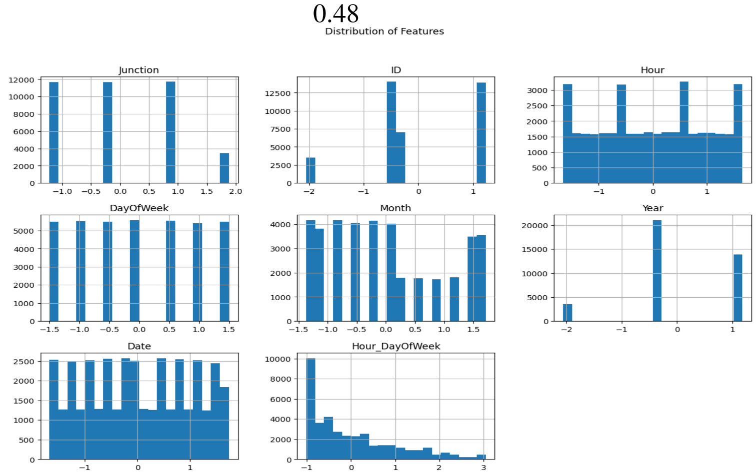

Fig. 1. Traffic distribution across junctions.

0.48

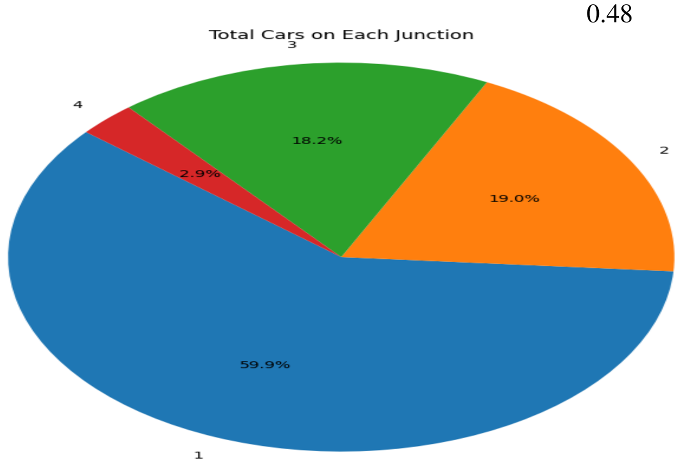

Fig. 2. Percentage distribution of traffic across junctions.

Fig. 3. Inconclusive findings from data visualization and junction pie chart.

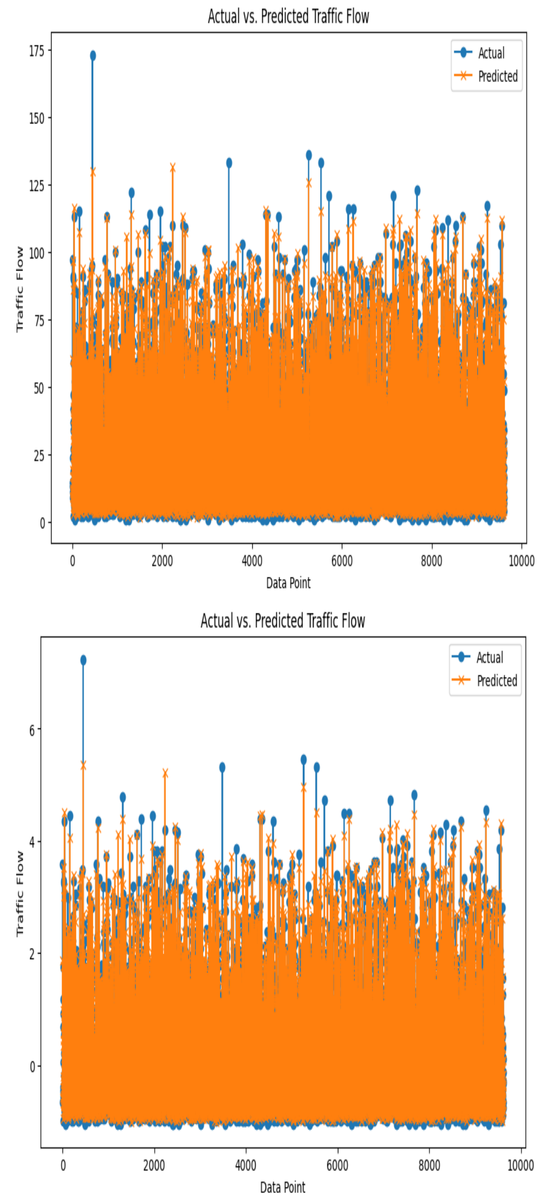

Fig. 4. Traffic flow prediction results using Random Forest Regression model.

V. Conclusion

In conclusion, this report underscores the critical role of traffic flow prediction and management in enhancing transportation efficiency. Through our analysis, we have approached the development of effective models, yet recognize the complexity inherent in these tasks. While our efforts have brought us close to achieving a satisfactory model, it's evident that algorithms such as Multilayer Perceptron (MLP) and Deep Learning (DL) hold promise for greater accuracy and predictive power.

Moving forward, it's imperative to embrace these advanced techniques and continue refining our approaches. Research and implementation efforts should focus on harnessing the capabilities of MLP and DL algorithms to further enhance traffic flow prediction and management systems. By doing so, we can contribute to the ongoing improvement of transportation systems, leading to safer, more efficient, and sustainable urban mobility solutions.

Also note we will extending this project to predict the value of vechiles in application to manipulate traffic lights timings to minimize the congestion at any junction at any instant of time.

References

https://www.kaggle.com/code/puneetgupta24/traffic-prediction
https://www.geeksforgeeks.org/linear-regression-implementation-from-scratch-using-python/
https://www.overleaf.com

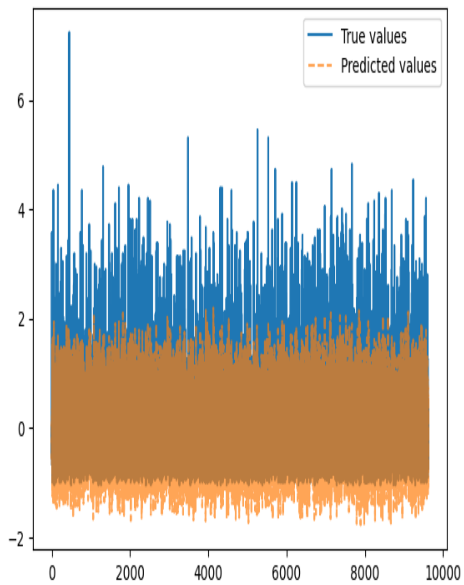

Fig. 5. Ineffective traffic flow prediction results using Linear Regression model.

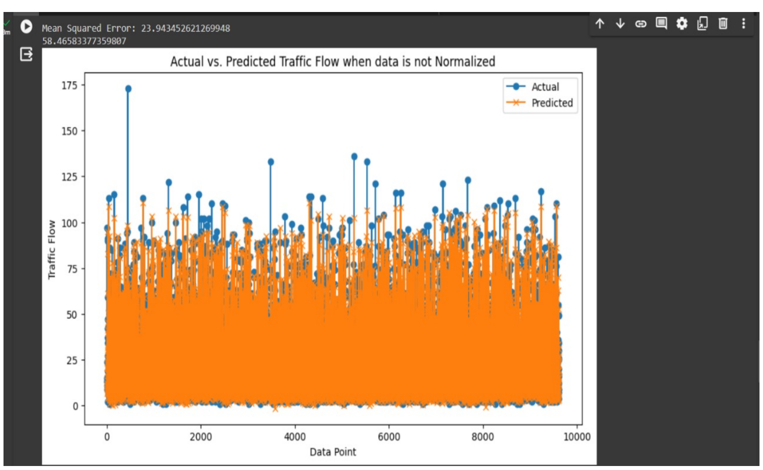

Fig. 6. Traffic flow prediction results using Gradient Boosting model.