Predicting and Analyzing

Traffic Levels

Project Report

Cameron Johnston

Bhavpreet Singh

Saneen Pervez

Jaskaran Sawhney

CSCI-436 W01 – Big Data Analytics

Professor Xueqing Huang

May 9th, 2024

Table of Contents

[**Abstract 3**](#_wq2o94vi96tn)

[**Introduction 3**](#_d3eydhh120ld)

[**Datasets 4**](#_d811fwgvlggz)

[Training Dataset 4](#_2fcfw2ps6rno)

[Testing Dataset 5](#_ohwt1mvmhhw0)

[**Technical Details 5**](#_du56i0jmpxa9)

[Visualization 6](#_fn0jjjmiulx9)

[Preprocessing 6](#_lm6g025jjgm7)

[Processing 7](#_vdct64qx8102)

[Logistic Regression 7](#_rptw2o718va2)

[Decision Tree 7](#_ggmxwpwz01rn)

[K Nearest Neighbors (KNN) 8](#_l8syawvq2vz8)

[Evaluation 8](#_2y3itonydwa)

[Accuracy 8](#_w8qdt8amesi4)

[Confusion Matrix 9](#_r35f4cw0r1po)

[Classification Report 9](#_pp48xj45ee42)

[**Results 9**](#_tqdi2o52qaj4)

[Initial Analysis 9](#_nofgvvok6cun)

[Time vs Traffic situation 10](#_ppb9w8k0oy9p)

[Day vs Traffic situation 11](#_d1uu95iu0uzw)

[Prediction Results 11](#_sxq9k1jyqcju)

[Logistic Regression 11](#_6v9vk2ajs6r7)

[Confusion Matrix: 12](#_3al1qvq9l7dd)

[Classification Reports: 13](#_rkqcq9a3rsu9)

[Decision Tree 13](#_kcqneevuw5jk)

[Confusion Matrix: 14](#_e8ltx6cm9guk)

[Classification Reports: 14](#_53et9vkwsfct)

[K Nearest Neighbors 15](#_x41eszh55vdk)

[Confusion Matrix: 15](#_cna9gl8yy1s)

[Classification Reports: 16](#_6kotzy3qhk39)

[Comparison of the scores across the Models 17](#_pvrfxlcccibv)

[Pair plot of the three models 17](#_4usy22tbjg39)

[Violin Plot of the three models 18](#_cfxrvlexsizl)

[KDE plot of the three models 18](#_u4ud4wuxqw49)

[**Discussions & Conclusions 19**](#_40p7pyehemim)

[**Individual member contribution to the group project 19**](#_5us438xngzol)

[Team Responsibilities for Traffic Prediction Project 19](#_7dkdvtyhn2ga)

# Abstract

In this project, we'll work with a dataset created using a special computer program to spot cars, bikes, buses, and trucks in city traffic. This data is saved in a CSV file, which has info like the time, date, and how many of each vehicle type were seen every 15 minutes. Our job is to study this data to understand how traffic behaves in cities. We'll use simple math and plots to find out trends and connections in the data. Then, we'll use different algorithms to guess how heavy traffic jams might happen. By knowing more about how traffic moves, we can help city planners make better decisions about building roads and managing traffic. This could mean finding ways to make traffic flow smoother or helping to avoid big jams. In the end. We hope to make city life easier for everyone by tackling traffic problems in a smart way. Our main task is to analyze this data to gain a deeper understanding of how traffic behaves in cities. Using different mathematical techniques, approaches and visualizations, we'll uncover trends and connections within the data, allowing us to predict when and where traffic jams might occur. By gaining insights into traffic movement, we aim to assist city planners in making informed decisions about infrastructure development and traffic management strategies. Ultimately, our objective is to contribute to the creation of more efficient and livable cities by addressing traffic issues in a smart and data-driven manner.

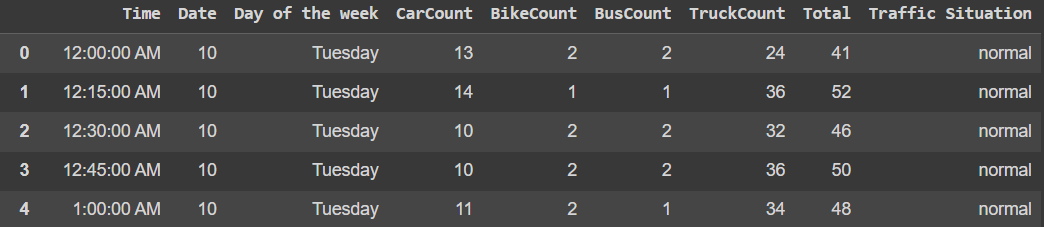
# Introduction

In busy cities, dealing with traffic jams is a big headache for everyone. Whether you're driving a car, riding a bike, or taking a bus, getting stuck in traffic can be a real pain. But what if we could understand how traffic moves and find ways to make it better? That's what this project is all about. We're diving into the world of city traffic using a special dataset created with high-tech cameras. Our mission is to dig into this data and uncover some useful insights. We want to figure out why traffic jams happen, where they're most likely to occur, and how we can make things smoother for everyone on the road. By using simple math and clever computer algorithms, we hope to come up with smart solutions to make city life a little easier for everyone. Our motivation is to improve urban life by tackling the widespread problem of traffic congestion. We aim to reduce stress and wasted time for commuters, minimize environmental impact, and boost economic efficiency by finding smart solutions to manage congestion effectively. Our task is to analyze traffic data to understand patterns and trends, predict congestion hotspots(like, what time and what day of the week), and develop strategies to manage traffic more efficiently. We'll use statistical analysis and machine learning techniques to extract insights from the data and provide actionable recommendations for improving urban mobility.

# Datasets

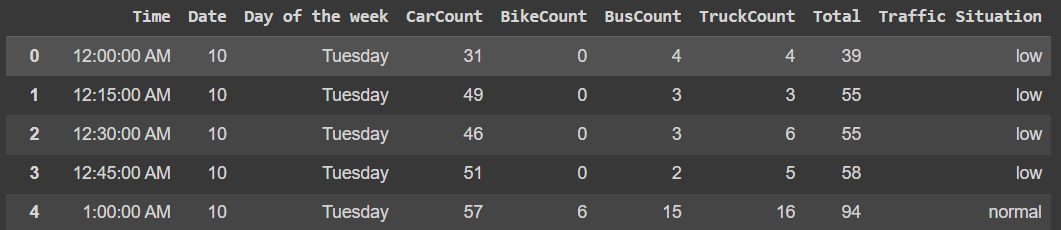
## Training Dataset

The training dataset used to train the models contained traffic data collected over a period of two months. It comprised 5952 instances, with each instance characterized by nine columns: Time, Date, Day of the week, CarCount, BikeCount, BusCount, TruckCount, Total, and Traffic Situation. The Time column recorded the time of day when the traffic data was collected, while the Date column indicated the specific date of data collection. The Day of the week column categorized the days into Monday through Sunday. The remaining columns, including CarCount, BikeCount, BusCount, and TruckCount, represented the count of different types of vehicles observed in the traffic, with the Total column denoting the total count of all vehicle types. The Traffic Situation column categorized the traffic conditions into different situations such as low, normal, high, or heavy traffic. This comprehensive dataset provided rich information about traffic patterns over the two-month period, enabling the machine learning models to learn and make predictions based on various traffic scenarios.



## Testing Dataset

The testing dataset, similar to the training dataset, contains traffic data but is utilized to evaluate the performance of the trained machine learning models. It comprises 2976 instances, each characterized by nine columns: Time, Date, Day of the week, CarCount, BikeCount, BusCount, TruckCount, Total, and Traffic Situation. The Time column records the time of day when the traffic data was collected, while the Date column indicates the specific date of data collection. The Day of the week column categorizes the days into Monday through Sunday. The remaining columns, including CarCount, BikeCount, BusCount, and TruckCount, represent the count of different types of vehicles observed in the traffic, with the Total column denoting the total count of all vehicle types. Similar to the training dataset, the Traffic Situation column categorizes the traffic conditions into different situations such as low, normal, high, or heavy traffic. This testing dataset serves as an independent set of observations to assess the generalization performance of the trained models on unseen data.



# Technical Details

In this project, our objective was to visualize the distribution of traffic situations over time using data plotting and visualization techniques with the Matplotlib and Seaborn libraries, and train and test different models to correctly predict traffic.

## Visualization

The initial step in any data analysis process is gaining an understanding of the dataset. In this code, the training data is loaded from a CSV file named 'TrafficTwoMonth.csv' and visualized using various plots. The time distribution of traffic situations is illustrated through a histogram, allowing us to observe patterns in traffic at different times of the day. Additionally, a histogram based on the day of the week provides insights into traffic variations across weekdays and weekends which helped us understand the trends of the data.

## Preprocessing

To prepare our data:

The time data in the dataset is initially in string format. To make it more usable for analysis, it is converted into a numeric format representing the number of seconds elapsed since midnight. This conversion simplifies time-based calculations and allows the models to learn patterns effectively. The time data is first converted from string format to datetime format using pd.to\_datetime(). Then, it's broken down into hour, minute, and second components, and the time is converted into seconds elapsed since midnight.

Standardization is applied to numerical features in the dataset using StandardScaler. This process transforms the numerical features to have a mean of 0 and a standard deviation of 1. Standardization is essential for many machine learning algorithms, ensuring that features are on a similar scale and preventing any single feature from dominating the learning process. This involved applying the StandardScaler transformation to the feature dataset, X\_train. Categorical variables like 'Day of the week' and 'Traffic Situation' are encoded into numerical values. We encoded the 'Day of week' variable into numerical values ranging from 1 to 7 and the 'Traffic Situation' variable into numerical values from 0 to 3. Subsequently, we split the dataset into training and testing sets, with X\_test representing the features used for testing the machine learning model, excluding the target variable 'Traffic Situation'.

## Processing

Once the data is pre-processed, it's ready for model training. In this project, three different classification algorithms are trained and used:

### Logistic Regression

Logistic regression is a popular classification algorithm used for binary classification tasks. It models the probability of a binary outcome based on one or more predictor variables. In this code, logistic regression is trained on the pre-processed training data to predict traffic situations. Logistic regression models the probability of a categorical outcome (in this case, traffic situations) given the input features. By learning the relationships between the features (e.g., time, day of the week, number of vehicles, etc) and the probability of each traffic situation, logistic regression can make predictions about the likelihood of encountering different traffic scenarios. Its simplicity and efficiency make it a good starting point for classification tasks, providing a baseline performance that other, more complex models can be compared against.

### Decision Tree

Decision trees are a non-parametric supervised learning method used for classification and regression tasks. Decision trees partition the feature space into regions based on simple decision rules inferred from the training data. By recursively splitting the data based on the most informative features, decision trees create a tree-like structure that can predict the target variable (traffic situations) for new instances. Decision trees are interpretable, allowing us to understand the decision-making process behind the model's predictions. Additionally, they can capture nonlinear relationships between the features and the target variable, making them suitable for complex classification tasks like predicting traffic situations..

### K Nearest Neighbors (KNN)

KNN is a simple, instance-based learning algorithm used for classification and regression tasks. KNN operates on the principle of similarity, where the class of a new instance is determined by the class of its nearest neighbors in the feature space. By calculating distances between the new instance and all other instances in the training data, KNN identifies the k nearest neighbors and assigns the most common class among them as the predicted class for the new instance. KNN is non-parametric and instance-based, meaning it does not make any assumptions about the underlying data distribution and learns directly from the training instances. This makes KNN robust to noisy data and suitable for datasets with complex decision boundaries, such as those encountered in traffic prediction tasks.

## Evaluation

Evaluation of model performance is essential to assess the effectiveness of the trained models. In this code, several evaluation metrics are computed:

### Accuracy

Accuracy measures the proportion of correctly classified instances out of the total instances. It provides a general overview of model performance but may not be sufficient for imbalanced datasets.

### Confusion Matrix

A confusion matrix provides a detailed breakdown of model predictions, showing the number of true positive, true negative, false positive, and false negative predictions. It helps in understanding the types of errors made by the model and the overall performance across different classes.

### Classification Report

The classification report provides a summary of important classification metrics such as precision, recall, and F1-score for each class. Precision measures the proportion of true positive predictions out of all positive predictions, while recall measures the proportion of true positive predictions out of all actual positive instances. F1-score is the harmonic mean of precision and recall, providing a balanced measure of a classifier's performance.

# Results

## Initial Analysis

Our initial analysis revealed several notable trends:

### Time vs Traffic situation

A graph with different colored lines

Description automatically generated

Traffic levels were consistently normal between 10:00 PM and 3:45 AM, as well as from 11:00 AM to 1:00 PM. During the early hours of the morning, from 3:45 AM to 5:45 AM, traffic was observed to be low. Conversely, heavy traffic congestion was observed during peak commuting hours, from 6:00 AM to 9:00 AM and from 4:00 PM to 6:15 PM.

### Day vs Traffic situation

A graph of different colored lines

Description automatically generated

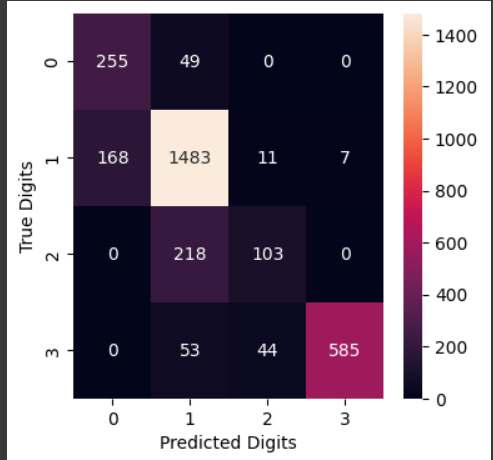
Traffic levels were consistently normal throughout the week. During the start of the weekend on Friday, traffic was observed to be low. Conversely, heavy traffic congestion was observed during peak commuting hours, from 6:00 AM to 9:00 AM and from 4:00 PM to 6:15 PM on the week.

## Prediction Results

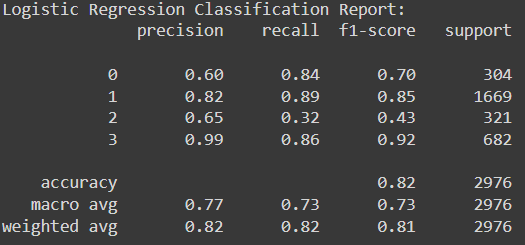
### Logistic Regression

The logistic regression model achieved an accuracy of 81.519%, indicating that it correctly classified approximately 81.52% of the instances in the test dataset. The classification report provides further insights into the model's performance across different classes. For class 0 (representing low traffic situations), the precision and recall were 0.60 and 0.84 respectively, resulting in an F1-score of 0.70. Class 1 (normal traffic situations) exhibited higher precision (0.82) and recall (0.89), leading to an F1-score of 0.85. Class 2 (high traffic situations) had lower precision (0.65) and recall (0.32), resulting in an F1-score of 0.43. Class 3 (heavy traffic situations) showed excellent precision (0.99) and recall (0.86), yielding an impressive F1-score of 0.92. The weighted average F1-score of the model was 0.81, indicating its overall effectiveness in classifying traffic situations. The confusion matrix provides a detailed breakdown of the model's predictions, showing the number of true positive, true negative, false positive, and false negative predictions for each class. Overall, the logistic regression model demonstrated strong performance, particularly in accurately predicting heavy traffic situations, while also providing reasonable performance for other traffic scenarios.

#### Confusion Matrix:



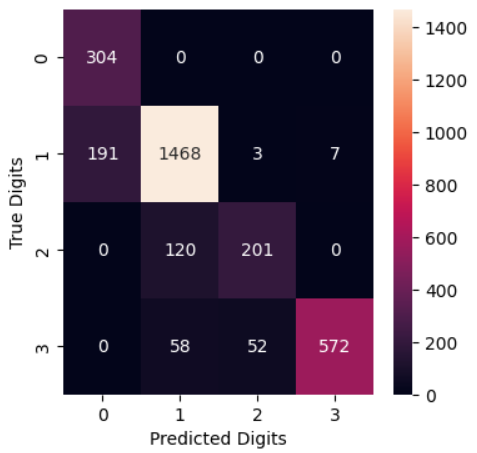
#### Classification Reports:



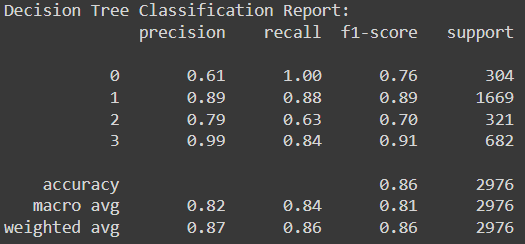
### Decision Tree

The decision tree classifier exhibited strong performance with an accuracy of 85.517%, indicating that it accurately classified approximately 85.52% of the instances in the test dataset. The classification report provides detailed insights into the model's performance across different classes. For class 0 (representing low traffic situations), the precision was 0.61 and recall was 1.00, resulting in an F1-score of 0.76. Class 1 (normal traffic situations) showed high precision (0.89) and recall (0.88), leading to an impressive F1-score of 0.89. Class 2 (high traffic situations) had reasonable precision (0.79) and recall (0.63), resulting in an F1-score of 0.70. Class 3 (heavy traffic situations) exhibited excellent precision (0.99) and recall (0.84), yielding a high F1-score of 0.91. The confusion matrix provides a detailed breakdown of the model's predictions, showing the number of true positive, true negative, false positive, and false negative predictions for each class. Overall, the decision tree classifier demonstrated robust performance across all traffic situations, particularly excelling in accurately identifying low traffic situations while maintaining high precision and recall for other traffic scenarios.

#### Confusion Matrix:



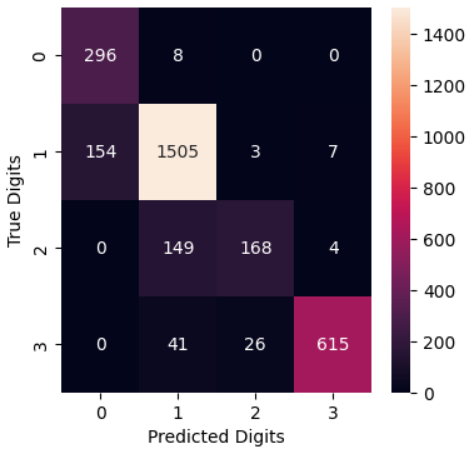
#### Classification Reports:



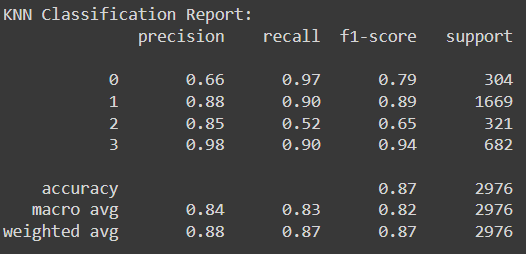
### K Nearest Neighbors

The K Nearest Neighbors (KNN) classifier demonstrated strong performance with an accuracy of 86.828%, indicating that it correctly classified approximately 86.83% of the instances in the test dataset. The classification report provides detailed insights into the model's performance across different classes. For class 0 (representing low traffic situations), the precision was 0.66 and recall was 0.97, resulting in an F1-score of 0.79. Class 1 (normal traffic situations) exhibited high precision (0.88) and recall (0.90), leading to an impressive F1-score of 0.89. Class 2 (high traffic situations) had reasonable precision (0.85) and recall (0.52), resulting in an F1-score of 0.65. Class 3 (heavy traffic situations) showed excellent precision (0.98) and recall (0.90), yielding a high F1-score of 0.94. The confusion matrix provides a detailed breakdown of the model's predictions, showing the number of true positive, true negative, false positive, and false negative predictions for each class. Overall, the KNN classifier demonstrated robust performance across all traffic situations, with particularly high accuracy and precision for normal and heavy traffic scenarios.

#### Confusion Matrix:

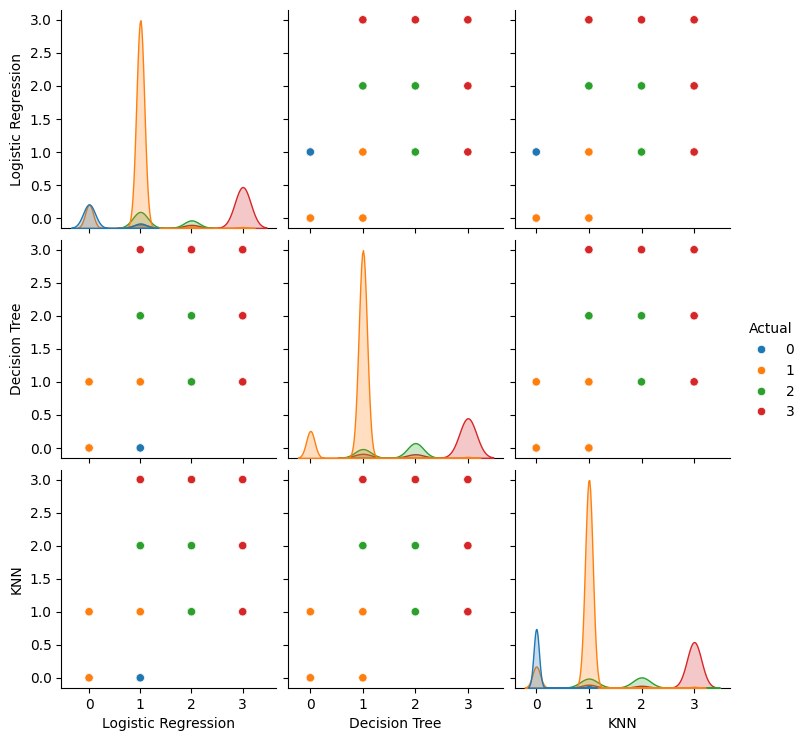


#### Classification Reports:

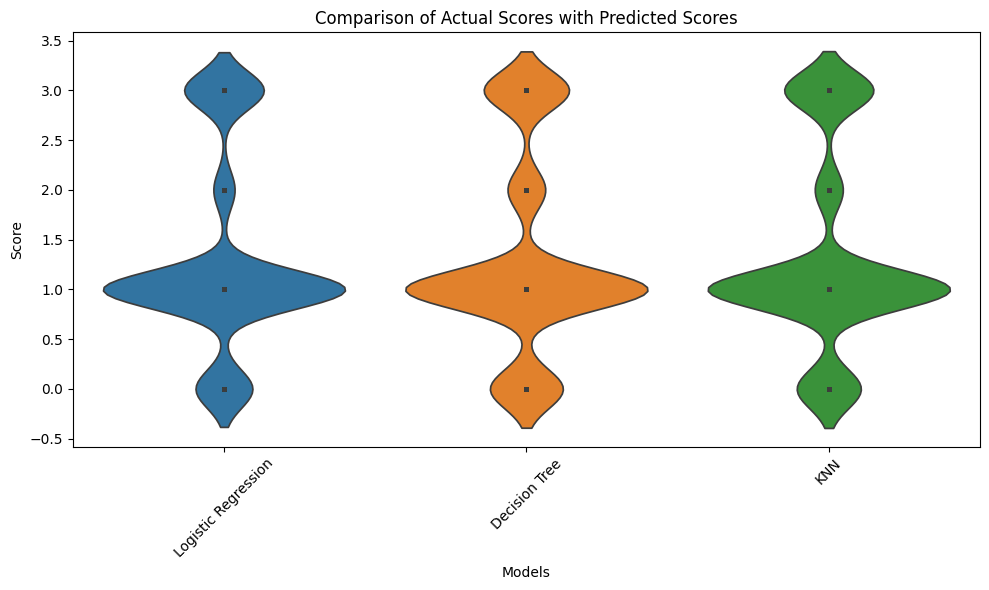


## Comparison of the scores across the Models

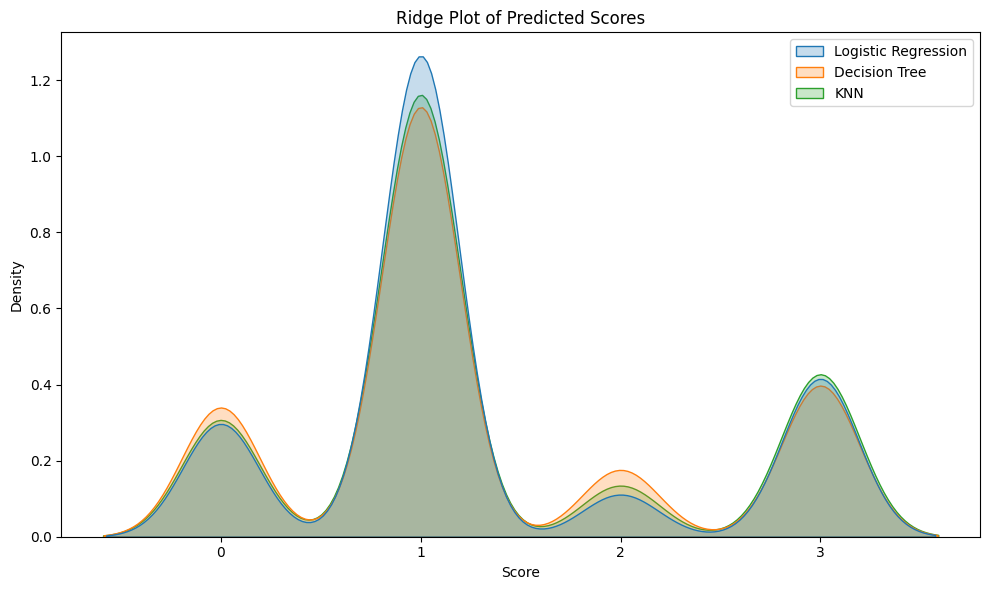
### Pair plot of the three models



### Violin Plot of the three models



### KDE plot of the three models



# Discussions & Conclusions

Throughout the project, we talked a lot about choosing which details to use in our model and how to understand the predictions it makes. We tried out different combinations of details, like what time of day it is and what day of the week it is, to see which ones might help the model make better guesses. We found that focusing on these time-related details really made a difference in how well our model worked. Also, we spent time figuring out what the model was telling us about why traffic gets bad sometimes. We used special tricks to look at which details were most important and how they affected the model's predictions. These discussions showed us how important it is to pick the right details and understand what our model is trying to tell us when it makes predictions about traffic. In wrapping up, this project showed us how working together from different fields and trying things out over and over are key to finding good ways to predict traffic and manage congestion. By using fancy tools to look at the data and smart computer programs to make guesses, We learned a lot about how traffic works. These guesses could help people who plan Things in cities make better choices, like when to fix roads or how to plan public transportation. In the end, our work could help make getting around in cities easier and better for everyone. making cities more eco-friendly and efficient.

# Individual member contribution to the group project

## Team Responsibilities for Traffic Prediction Project

In our traffic prediction project, each team member has distinct roles and responsibilities to ensure the project progresses smoothly and achieves its objectives.

**Jaskaran** takes the lead in data collection and preprocessing tasks, cleaning and preparing the dataset for analysis by handling missing values, outliers, and feature engineering. They also conduct thorough data analysis to uncover insights and patterns within the dataset.

**Saneen** facilitates communication among team members, ensuring everyone stays on track and informed about project developments.

**Cameron** leads the development and evaluation of predictive models, selecting suitable machine learning or statistical models and fine-tuning them using prepared datasets.

**Bhavpreet** evaluates the performance of predictive models and optimizes their performance, assisting in feature selection and ensuring the smooth integration of models into a usable system.

Through collaborative efforts, the team aims to develop accurate and effective models for predicting traffic congestion across various forms of road traffic.