**TRAFFIC SIMULATION AND PREDICTION**

**INTRODUCTION:**

In many big cities, traffic jams are a big problem, especially during busy times of the day. Lots of cars on the roads cause long delays, more fuel is used up, the air gets dirtier, and people get frustrated. The usual ways of managing traffic aren't enough to solve this issue, so we need new and creative ideas to make traffic better and help people move around the city more easily.

In the modern era of rapid urbanization and increasing reliance on automobiles, the issue of traffic congestion has emerged as a pressing concern for major cities. The bustling metropolises that once symbolized progress and prosperity now find themselves entrapped in a web of gridlock during peak hours, impeding economic productivity and diminishing the overall quality of life for their residents.

By continuously monitoring popular and alternative routes, advanced traffic simulation systems can provide valuable insights into traffic patterns, bottlenecks, and areas of high congestion. Armed with this information, traffic controllers can implement adaptive measures to regulate traffic flow, while navigation apps can offer drivers alternative routes to circumvent congested areas, reducing the strain on key roadways.

In this project we are taking data from the below mentioned datasets and we have built a model which estimates the traffic congestion. We have chosen to implement machine learning technique - time series forecasting.

**Dataset1:** <https://data.cityofchicago.org/Transportation/Chicago-Traffic-Tracker-Historical-Congestion-Esti/kf7e-cur8>

The data set "Chicago Traffic Tracker - Congestion Estimates by Region" provides valuable information about traffic congestion in the city of Chicago, Illinois. The dataset is maintained by the City of Chicago and is available on the data portal.

This dataset contains the historical estimated congestion for the 29 traffic regions, from January 2018 to current. The Chicago Traffic Tracker estimates traffic congestion on Chicago’s arterial streets (non-freeway streets) in real-time by continuously monitoring and analyzing GPS traces received from Chicago Transit Authority (CTA) buses.

**Dataset2** : <https://data.cityofchicago.org/Transportation/Chicago-Traffic-Tracker-Congestion-Estimates-by-Re/t2qc-9pjd>

This dataset contains the traffic congestion estimates Chicago for 29 regions in Chicago, where this dataset contains column “LAST\_UPDATE” which holds the data of the traffic congestion updated every 10 mints.

In this dataset we have a column “CURRENT\_SPEED” which gives us the reflection on the condition level in the region.

**METHODOLOGY:**

**Data Acquisition:**

We have got this data from the Chicago open data portal where it provides the history of traffic recorded and stores for every 5 years. We have downloaded the dataset from the option of Export where the portal provides in CSV format and we can extract the data directly from the API. We have imported the dataset into the program using pandas library in python.

A screenshot of a computer

Description automatically generated

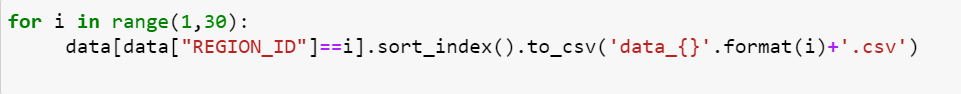
**Data Cleaning and Transformation:**

The data we downloaded is the raw data and it needs to be cleaned. We have seen that the data is not in the sorted order and data is not uploaded every 10 minutes. We also observed that due to some technical errors for some period the data was not uploaded. We have extracted only the time from the TIME column. We dropped some of the columns which are not useful.

We have calculated the latitude and longitude of each region and added it to the dataset to represent each of the regions in the map using a library called folium.



We have 29 regions in a data set, we have extracted each individual region into a CSV file.



We have written a function that set the time intervals to 10 minutes since the time was inconsistent which will be difficult to train a model with data containing irregular time intervals.

A screenshot of a computer program

Description automatically generated

We have sorted the data according to the time and we have used the backfill and forward fill to fill in the continuous missing data.

A screenshot of a computer code

Description automatically generated

**Exploratory Data Analysis:**

For region one speed, we have plotted a graph using MATLAB function, observe the frequency of region one's speed.

A screenshot of a computer

Description automatically generated



From the above graph we couldn’t figure out the frequency of this particular region speed, so we have narrowed down the graph 2 period to January 2023- April 2023 to get more insights about the region 1 speed.

A screen shot of a computer

Description automatically generated



From the above figure we can understand the patterns of how the data is on weekends and on weekdays. We observe that the congestion levels are high on weekends and congestion levels are low on weekdays.

A screenshot of a computer

Description automatically generated

In the above figure we used stat’s function in Python to analyze the particular period of data for a week, with the trend and the season.

**RESULTS:**

Model Estimation:- As the trend is horizontal, we need to get an appropriate model for forecasting future values. We studied different stats models. Simple Exponential Smoothing, Double and Triple Exponential Smoothing. After enough trial and error, we inspected that Triple Exponential Smoothing can recognize the pattern of the traffic flow.

A screenshot of a computer

Description automatically generated

A graph of orange and blue lines

Description automatically generated

The graph shows how the model estimates on the given data. The blue color plot represents the actual data and the orange color is the estimates of model. We can observe that the model predicts are some accurate at the peaks and almost same in other cases.



A graph of blue and orange lines

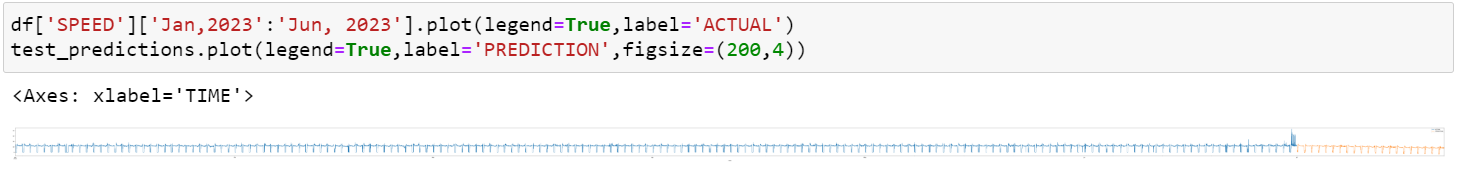
Description automatically generated

Since we can’t see the detailed insights into how the model behaves, we have plotted a graph for a particular period of samples i.e from 80000 to 80500.

A screenshot of a computer

Description automatically generated

Now we have trained and tested the model with the 2023 data i.e from January to June. We have calculated that in a week, for one region we get 1008 samples, so are using the model to predict the next 3000 samples (July 1 to July 21).



A graph showing a number of blue and orange lines

Description automatically generated

We observe that the model predicts the traffic congestion of the three weeks which is represented in orange. From the prediction we can see it’s almost same as the other months.

A screenshot of a data

Description automatically generated

A map with blue pins

Description automatically generated

We have calculated the latitude and longitude of each region using the columns east, west, north and south and plotted in the Chicago map using the folium library.

A screenshot of a computer

Description automatically generated

We have used ipywidgets to make interactive widgets and gives the user to choose the source and destination from the dropdown menu. A screenshot of a computer

Description automatically generated

With the given source, destination, date and time, the model is predicting the speed.

A map with blue pins on it

Description automatically generated

Using the folium library we are first pointing the source and destination of the user.

A screenshot of a computer program

Description automatically generated

In this we have used the OSMnx and networkx to find the path that the user can reach in minimum time. OSMnx stands for Open Streep Map where it is used to visualize and analyze real-world street model. Netowrkx is used for complex structures such as maps, graphs.

A screen shot of a computer code

Description automatically generated

A map of a city

Description automatically generated

The above figure shows us the shortest path between the source and destination. The source is represented with the green pointer and destination with the red pointer. We have calculated the three alternative routes with the traffic congestion and time is used as the sorting parameter.

A screenshot of a computer

Description automatically generated

A map of a city

Description automatically generated

A map with a blue line

Description automatically generated

A map of a city

Description automatically generated

**CHALLENGES FACED:**

* Data is not at regular intervals of time i.e for every 10 mins of hour as mentioned in the data description. Samples are running around 9,10,11 minutes interval. So we need to address this issue.
* Finding missing dates and inputting them with forward and backward fill.
* Google Routes API provides the first 40000 API calls for free and then they charge for every API call, so we must find an alternative to this.
* While building the model using Exponential Smoothing, we have faced some accuracy issues with simple and double exponential smoothing.
* For the three alternative routes, we have got some routes the same as the original one but it is a rare case.
* Large Data volume and Processing
* Outlier detections

**FUTURE WORK:**

**Scalability and Generalization**: - Evaluate model performance in different cities and regions for scalability and generalization.

**Incorporating External Factors**

Weather Impact: Integrate weather data into the model to understand how weather conditions affect traffic congestion.

**Collaborative Data Sharing**

Data Collaboration: Collaborate with other cities and transportation agencies to share traffic data and insights for cross-city traffic management.

**Seasonal Variation Analysis**

Seasonal Factors: Analyze seasonal variations in traffic congestion to adapt the model accordingly. Holiday and Event Impact: Investigate the impact of holidays and special events on traffic flow for more accurate predictions.

**User Feedback and Engagement**

User Surveys: Collect user feedback to understand the impact of predictions on commuters.

**TUTORIAL:**

**GitHub Link:** [**https://github.com/DJ2803/Traffic-Simulation-and-Prediction.git**](https://github.com/DJ2803/Traffic-Simulation-and-Prediction.git)