**PUBLIC TRANSPORT EFFICIENCY ANALYSIS**

**DAC\_Phase3**

**INTRODUCTION:**

Transportation efficiency is a critical factor in urban planning and sustainability. This document initiates the process of analysing public transportation efficiency using IBM Cognos for visualization.

This project seeks to address fundamental questions surrounding public transportation efficiency: Are services running on schedule? Is ridership optimized for specific routes? What are the factors affecting customer satisfaction?

Beginning with an exploration of the concept of transportation efficiency, we aim to collect, process, and clean relevant data to facilitate in-depth analysis. This analysis will provide valuable insights for improving public transportation system.

**ANALYSIS OBJECTIVES:**

The primary objectives of this project are to assess and improve public transportation efficiency. This involves evaluating factors such as ridership trends, route optimization, on-time performance, and environmental impact. We seek to leverage IBM Cognos for data visualization to gain actionable insights, enhance decision-making for transportation authorities, and contribute to more sustainable and effective urban mobility systems.

**DATA CLEANING AND PREPROCESSING:**

In the initial phase of our project aimed at analyzing public transportation efficiency using IBM Cognos for visualization, we need to set clear analysis objectives. This involves determining the specific aspects of public transportation. It's crucial to ensure the data is comprehensive and up-to-date, as this forms the foundation of our analysis.

The collected data must then undergo a thorough process of preprocessing and cleaning. This step includes handling missing values, standardizing data formats, and addressing outliers or discrepancies. We also verify data accuracy to avoid errors in our analysis.

By meticulously defining our objectives and rigorously preprocessing and cleaning the data, we can guarantee that our analysis will be based on high-quality, accurate information. This will lead to more meaningful and insightful visualizations in IBM Cognos.

**DATASET:**

<https://www.kaggle.com/datasets/rednivrug/unisys?select=20140711.CSV>

**CODE IMPLEMENTATION:**

import os

for dirname,\_, filenames in os.walk('/kaggle/input'):

for filename in filenames:

print (os.path.join(dirname, filename))

The data fields in the given file are

* **Trip ID** Unique identity of trip
* **Route ID** Value representing public transport route
* **Stop ID** Unique identity of stop
* **Stop Name** Name of given stop
* **Week Beginning** Date representing first day of any week
* **Number of Boarding** Count of all boarding’s occurred at this stop for the named trip over the previous week

**# Step 1: Load the dataset**

**IN:**

print("Load the dataset")

import pandas as pd

data = pd.read\_csv('/kaggle/input/unisys/20140711.CSV', low\_memory=False)

data.shape

data.head(10)

| **TripID** | **RouteID** | **StopID** | **StopName** | **WeekBeginning** | **NumberOfBoardings** |
| --- | --- | --- | --- | --- | --- |
| 0 | 23631 | 100 | 14156 | 181 Cross Rd | 2013-06-30 00:00:00 | 1 |
| 1 | 23631 | 100 | 14144 | 177 Cross Rd | 2013-06-30 00:00:00 | 1 |
| 2 | 23632 | 100 | 14132 | 175 Cross Rd | 2013-06-30 00:00:00 | 1 |
| 3 | 23633 | 100 | 12266 | Zone A Arndale Interchange | 2013-06-30 00:00:00 | 2 |
| 4 | 23633 | 100 | 14147 | 178 Cross Rd | 2013-06-30 00:00:00 | 1 |
| 5 | 23634 | 100 | 13907 | 9A Marion Rd | 2013-06-30 00:00:00 | 1 |
| 6 | 23634 | 100 | 14132 | 175 Cross Rd | 2013-06-30 00:00:00 | 1 |
| 7 | 23634 | 100 | 13335 | 9A Holbrooks Rd | 2013-06-30 00:00:00 | 1 |
| 8 | 23634 | 100 | 13875 | 9 Marion Rd | 2013-06-30 00:00:00 | 1 |
| 9 | 23634 | 100 | 13045 | 206 Holbrooks Rd | 2013-06-30 00:00:00 | 1 |

**# Step 2: Drop duplicates and Check data types of columns**

**IN:**

data = data.drop\_duplicates()

import seaborn as sns

print(data.dtypes)

**OUT:**

TripID int64

RouteID object

StopID int64

StopName object

WeekBeginning object

NumberOfBoardings int64

dtype: object

**# Step 3: Handle mixed data types**

# 'RouteID' column has mixed types, convert it to numeric

**IN:**

data['RouteID'] = pd.to\_numeric(data['RouteID'], errors='coerce')

print("Handle mixed data types")

print(data.shape)

**OUT:**

Handle mixed data types

(10857234, 6)

**# Step 4: Handle missing values**

# Drop rows with missing values or fill them based on your project requirements

**IN:**

data = data.dropna()

print("\nHandle missing values")

print(data.shape)

**OUT:**

Handle missing values

(6414906, 6)

**# Step 5: Convert 'WeekBeginning' column to datetime format**

**IN:**

data['WeekBeginning'] = pd.to\_datetime(data['WeekBeginning'], errors='coerce')

print("\nConvert 'WeekBeginning' column to datetime format")

print(data['WeekBeginning'].head())

**OUT:**

Convert 'WeekBeginning' column to datetime format

0 2013-06-30

1 2013-06-30

2 2013-06-30

3 2013-06-30

4 2013-06-30

Name: WeekBeginning, dtype: datetime64[ns]

**# Step 6: Clean 'StopName' column**

# Remove leading and trailing whitespaces

**IN:**

data['StopName'] = data['StopName'].str.strip()

print("\nClean 'StopName' column")

print(data['StopName'].head())

**OUT:**

Clean 'StopName' column

0 181 Cross Rd

1 177 Cross Rd

2 175 Cross Rd

3 Zone A Arndale Interchange

4 178 Cross Rd

Name: StopName, dtype: object

**IN:**

data.head()

**OUT:**

| TripID | RouteID | StopID | StopName | WeekBeginning | NumberOfBoardings |
| --- | --- | --- | --- | --- | --- |
| 0 | 23631 | 100.0 | 14156 | 181 Cross Rd | 2013-06-30 | 1 |
| 1 | 23631 | 100.0 | 14144 | 177 Cross Rd | 2013-06-30 | 1 |
| 2 | 23632 | 100.0 | 14132 | 175 Cross Rd | 2013-06-30 | 1 |
| 3 | 23633 | 100.0 | 12266 | Zone A Arndale Interchange | 2013-06-30 | 2 |
| 4 | 23633 | 100.0 | 14147 | 178 Cross Rd | 2013-06-30 | 1 |

**#Step 8 : Unique values for each column in the DataFrame**

**IN:**

print(data.nunique())

**OUT:**

TripID 23926

RouteID 323

StopID 6718

StopName 3840

WeekBeginning 54

NumberOfBoardings 381

dtype: int64

**IN:**

data.shape

data.columns

data.head(3)

**OUT:**

| TripID | RouteID | StopID | StopName | WeekBeginning | NumberOfBoardings |
| --- | --- | --- | --- | --- | --- |
| 0 | 23631 | 100.0 | 14156 | 181 Cross Rd | 2013-06-30 | 1 |
| 1 | 23631 | 100.0 | 14144 | 177 Cross Rd | 2013-06-30 | 1 |
| 2 | 23632 | 100.0 | 14132 | 175 Cross Rd | 2013-06-30 | 1 |

**IN:**

#Count the number of missing value in each coloumn

data.isnull().sum()

**OUT:**

TripID 0

RouteID 0

StopID 0

StopName 0

WeekBeginning 0

NumberOfBoardings 0

dtype: int64

**IN:**

#different type of Unique Data in the dataset

data['WeekBeginning'].unique()

**OUT:**

**<DatetimeArray>**

['2013-06-30 00:00:00', '2013-07-07 00:00:00', '2013-07-14 00:00:00',

'2013-07-21 00:00:00', '2013-07-28 00:00:00', '2013-08-04 00:00:00',

'2013-08-11 00:00:00', '2013-08-18 00:00:00', '2013-08-25 00:00:00',

'2013-09-01 00:00:00', '2013-09-08 00:00:00', '2013-09-15 00:00:00',

'2013-09-22 00:00:00', '2013-09-29 00:00:00', '2013-10-06 00:00:00',

'2013-10-13 00:00:00', '2013-10-20 00:00:00', '2013-10-27 00:00:00',

'2013-11-03 00:00:00', '2013-11-10 00:00:00', '2013-11-17 00:00:00',

'2013-11-24 00:00:00', '2013-12-01 00:00:00', '2013-12-08 00:00:00',

'2013-12-15 00:00:00', '2013-12-22 00:00:00', '2013-12-29 00:00:00',

'2014-01-05 00:00:00', '2014-01-12 00:00:00', '2014-01-19 00:00:00',

'2014-01-26 00:00:00', '2014-02-02 00:00:00', '2014-02-09 00:00:00',

'2014-02-16 00:00:00', '2014-02-23 00:00:00', '2014-03-02 00:00:00',

'2014-03-09 00:00:00', '2014-03-16 00:00:00', '2014-03-23 00:00:00',

'2014-03-30 00:00:00', '2014-04-06 00:00:00', '2014-04-13 00:00:00',

'2014-04-20 00:00:00', '2014-04-27 00:00:00', '2014-05-04 00:00:00',

'2014-05-11 00:00:00', '2014-05-18 00:00:00', '2014-05-25 00:00:00',

'2014-06-01 00:00:00', '2014-06-08 00:00:00', '2014-06-15 00:00:00',

'2014-06-22 00:00:00', '2014-06-29 00:00:00', '2014-07-06 00:00:00']

Length: 54, dtype: datetime64[ns]

**IN:**

import matplotlib.pyplot as plt

fig,axrr=plt.subplots(3,2,figsize=(18,18))

data['NumberOfBoardings'].value\_counts().sort\_index().head(20).plot.bar(ax=axrr[0][0])

data['WeekBeginning'].value\_counts().plot.area(ax=axrr[0][1])

data['RouteID'].value\_counts().head(20).plot.bar(ax=axrr[1][0])

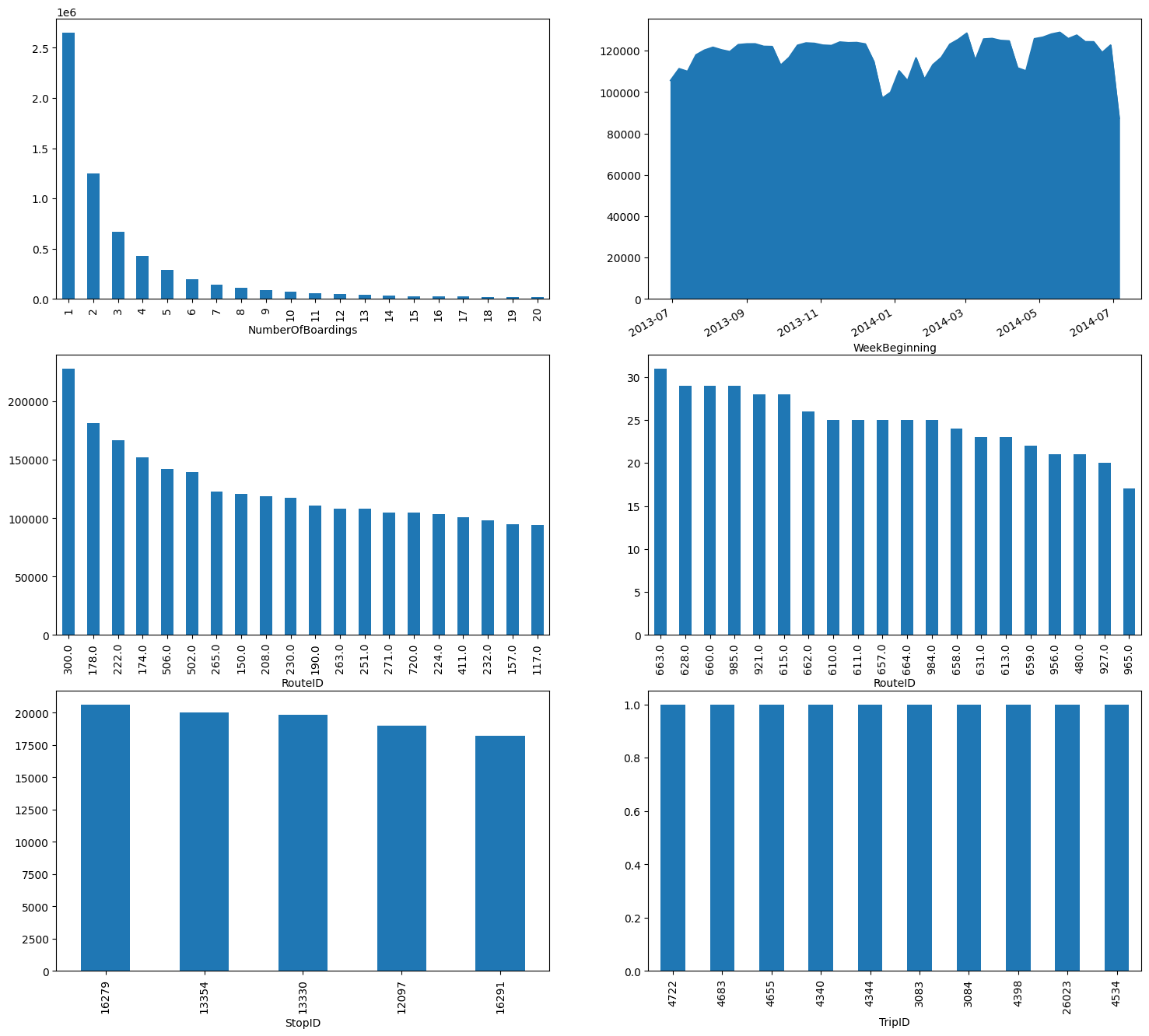
data['RouteID'].value\_counts().tail(20).plot.bar(ax=axrr[1][1])

data['StopID'].value\_counts().head(5).plot.bar(ax=axrr[2][0])

data['TripID'].value\_counts().tail(10).plot.bar(ax=axrr[2][1])

**OUT:**

<Axes: xlabel='TripID'>



**IN:**

# Save the cleaned dataset to a new CSV file

data.to\_csv('cleaned\_data.csv', index=False)

print("\nSave the cleaned dataset to a new CSV file")

print("Cleaned dataset saved successfully.")

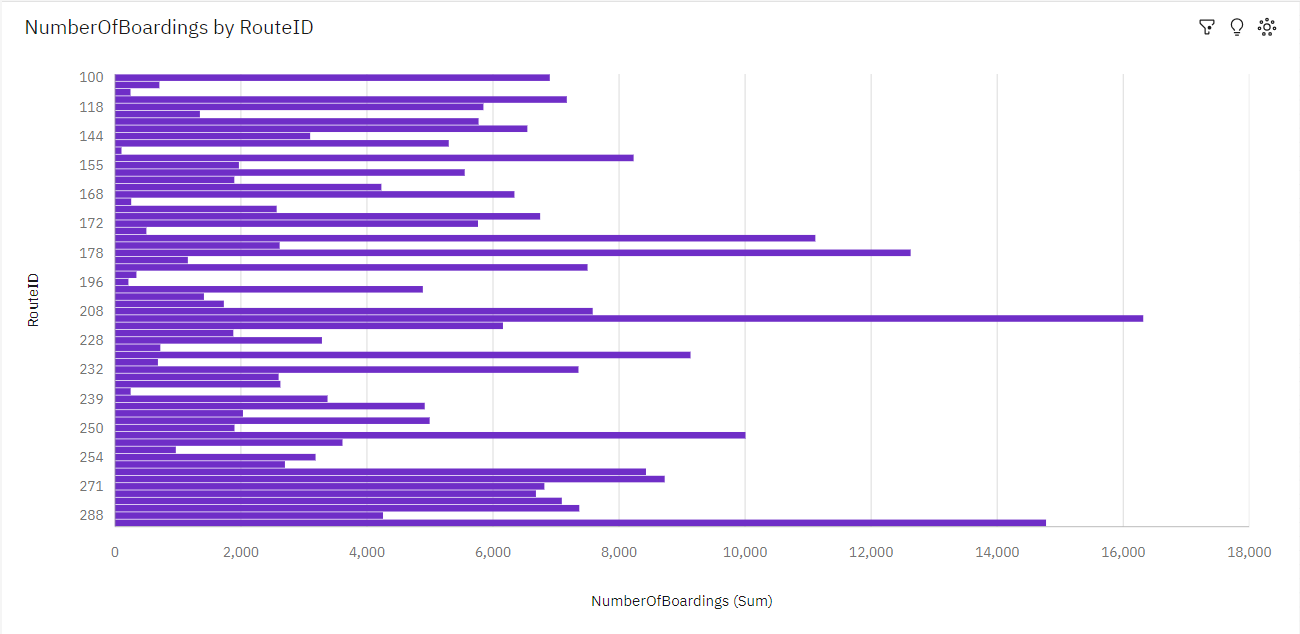
**OUT:**

Save the cleaned dataset to a new CSV file

Cleaned dataset saved successfully.

**VISUALISATION IN IBM COGNOS:**

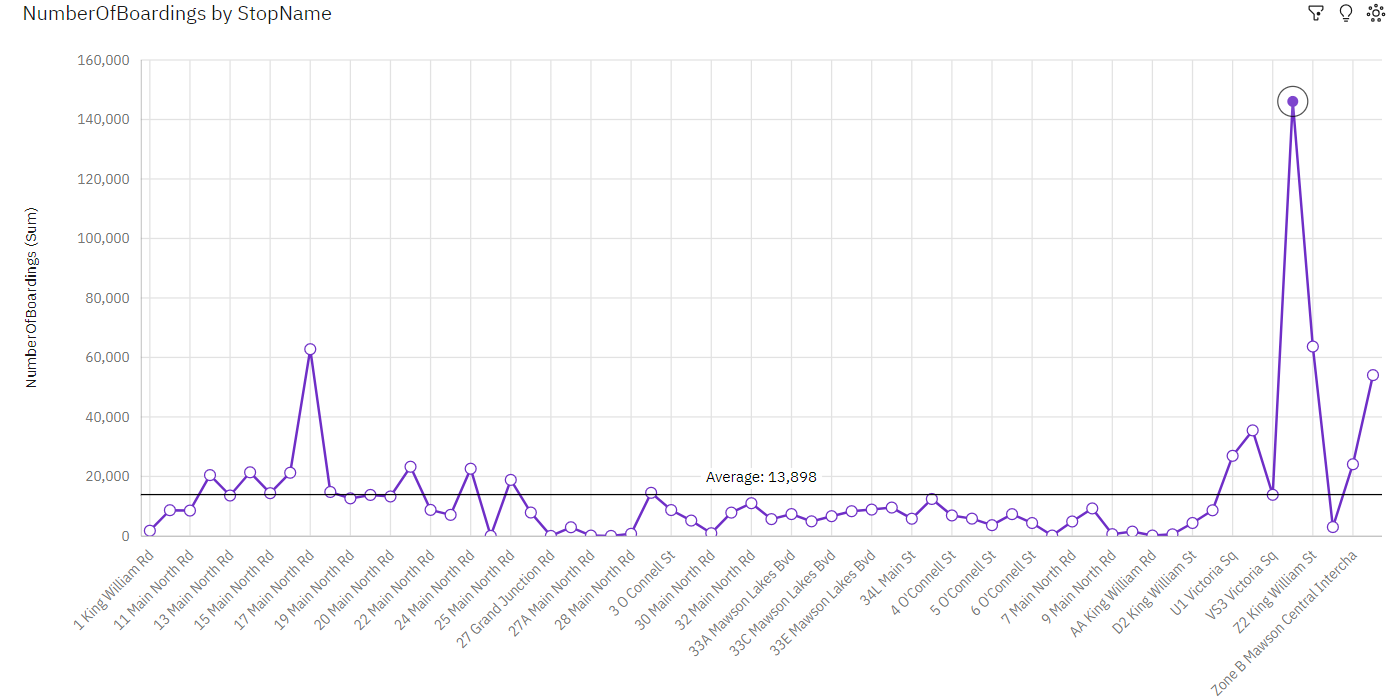
A bar chart visualizing the number of Boardings for each route for Route ID ranging from 100 to 288.



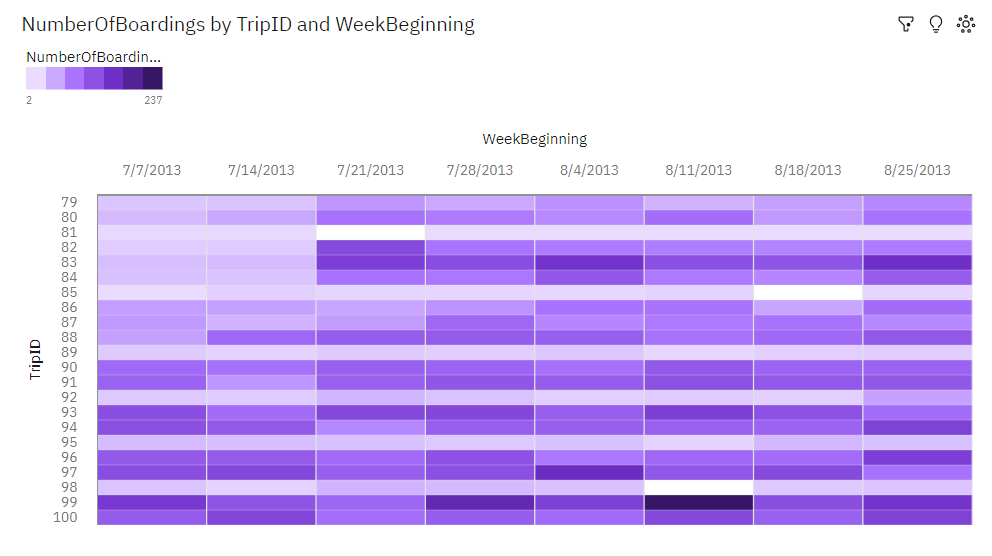
Insights:

Route ID 222.0 has the highest total Number of Boardings due to Week beginning 2013-07-21. Number of Boardings is unusually high when Route ID is 222 and 300.

Visualizing No of Boardings by Stop Name:



A heat map representing No of Boardings by Trip ID for the Week Beginning from 7/7/2013 to 8/25/2013



**CONCLUSION:**

In this initial phase of the project, the dataset was effectively processed and cleaned to ensure its accuracy and reliability. Subsequently, compelling visualizations were generated using IBM Cognos, setting the stage for a comprehensive analysis of public transportation efficiency. These preparatory steps are essential for facilitating informed decision-making and shaping the future of urban transportation systems.

**SUBMITTED BY:**

* + - **NAME: SARASWATHI G**
    - **REG.NO:412621243043**
    - **DEPT: B TECH AI&DS**
    - **YEAR: III**
    - **SEM: 5**
    - **COLLEGE NAME: 4126-SRI VENKATESWARAA COLLEGE OF TECHNOLOGY**