**PUBLIC TRANSPORTATION EFFICIENCY ANALYSIS**

**TEAM MEMBER** **732121104074 : SRIKANTH L**

**PROJECT : PUBLIC TRANSPORTATION EFFICIENCY**

**PHASE 3 : DEVELOPMENT PART 1**

**TOPIC :** Start building the public transportation efficiency and analysis by loading and pre-processing the dataset

****

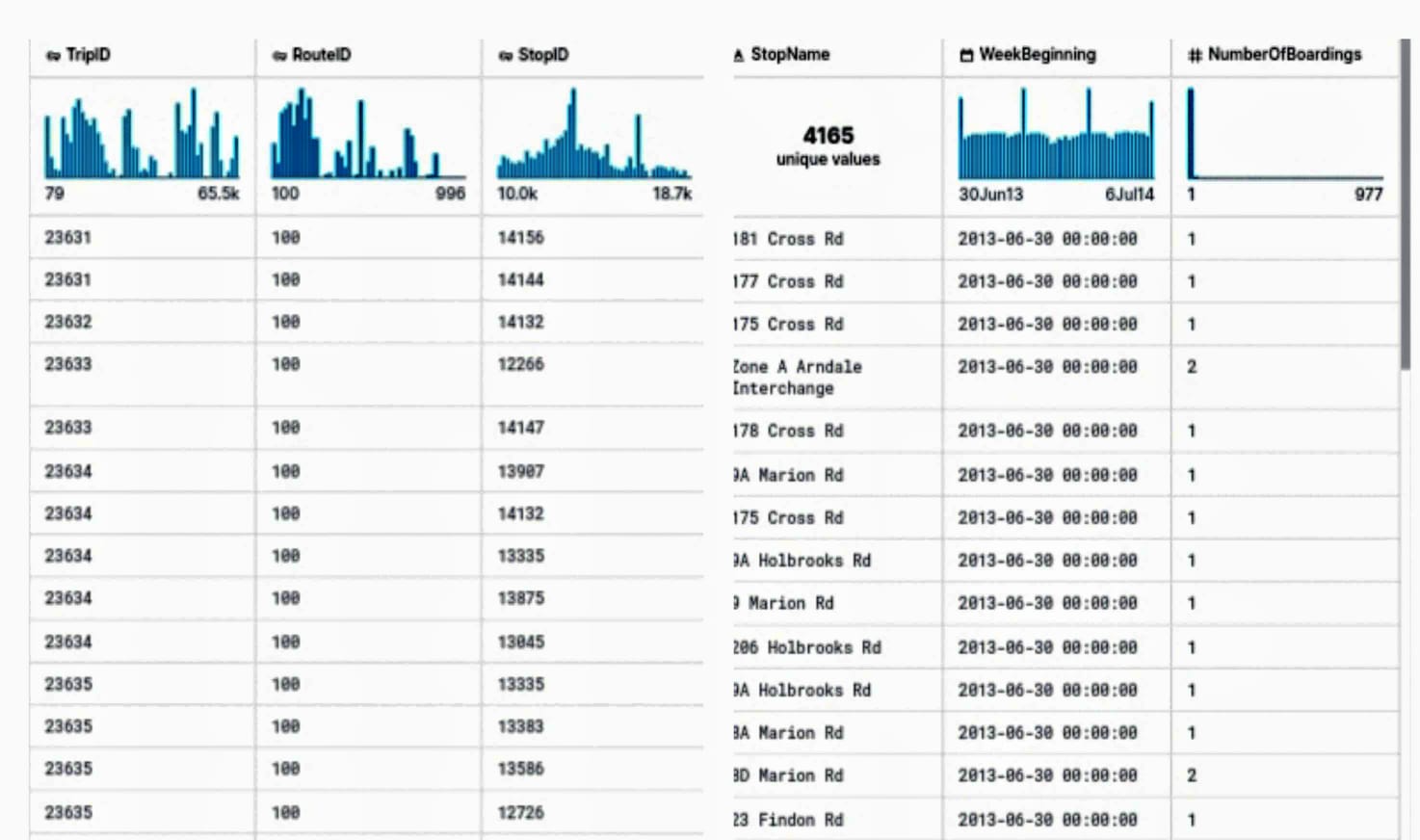
**INTRODUCTION:**

* **T**he exciting journey of enhancing public transportation efficiency through data analysis and advanced data preprocessing techniques. In our modern world, where urbanization and the need for sustainable transportation solutions are growing rapidly, harnessing the power of data is essential. By loading and pre-processing relevant datasets, we lay the foundation for optimizing public transportation systems, making them more accessible, reliable, and environmentally friendly.
* In this endeavor, we will explore the vast realm of public transportation data, gathering information from various sources to gain insights into passenger behavior, service utilization, and infrastructure performance. Our mission is to transform raw data into actionable insights that can drive decisions, improve operational efficiency, and ultimately enhance the overall quality of public transportation services.
* Through this journey, we'll delve into data cleansing, transformation, and the extraction of meaningful features to facilitate our analysis. By doing so, we aim to empower transit authorities, urban planners, and policymakers to make informed decisions that lead to a more efficient, convenient, and sustainable public transportation network.
* So, let's embark on this exploration together, where we'll not only navigate through the intricacies of public transportation data but also leverage the power of technology to pave the way for a more efficient and sustainable future in public transportation.

**DATA SOURCE:**

**Dataset Link:** https://www.kaggle.com/datasets/rednivrug/unisys?select=20140711.CSV

We have explored how people are travelling from different stops in Adelaide Metropolitan area and managing the buses on each route according to the no of passenger commuting through the buses.



To begin our journey towards enhancing public transportation efficiency and conducting insightful analysis, we must first tackle the critical task of loading and pre-processing the dataset. This foundational step will enable us to extract valuable information and pave the way for data-driven decision-making. Here's a step-by-step guide to get us started:

**Step 1: Data Collection**

Start by gathering the relevant dataset(s). Public transportation datasets can include information about routes, schedules, ridership, infrastructure, and more. Ensure you have the necessary permissions or access to these datasets, and consider the data sources, such as government agencies, transportation authorities, or private organizations.

**Step 2: Data Loading**

Depending on the format of the dataset, use appropriate tools or libraries to load the data into your analysis environment. Common data formats include CSV, JSON, Excel, and databases. Python libraries like pandas, R, or SQL can be used for this purpose.

python

import pandas as pd

# Example for loading a CSV dataset

dataset = pd.read\_csv('public\_transport\_data.csv')

**Step 3: Data Inspection**

Once the dataset is loaded, perform an initial inspection to understand its structure, size, and content. This will help you identify any missing values, outliers, or data quality issues that need to be addressed.

# Display the first few rows of the dataset

print(dataset.head())

# Check for missing values

print(dataset.isnull().sum())

**Step 4: Data Cleaning**

Clean the dataset by addressing missing values, removing duplicates, and handling outliers. This ensures the dataset is accurate and reliable for analysis.

# Handle missing values

dataset = dataset.dropna()

# Remove duplicates

dataset = dataset.drop\_duplicates()

# Outlier detection and treatment

**Step 5: Data Transformation**

Transform the data to make it suitable for analysis. This may involve converting data types, aggregating information, and creating new variables for meaningful insights.

# Example: Converting date strings to datetime objects

dataset['date'] = pd.to\_datetime(dataset['date'])

# Aggregating data (e.g., daily ridership)

daily\_ridership = dataset.groupby('date')['riders'].sum()

**Step 6: Feature Engineering**

Create new features or variables that can provide deeper insights into public transportation efficiency. This may involve calculating performance metrics, such as on-time performance, passenger-to-vehicle ratios, or route utilization.

**Step 7: Data Saving**

After pre-processing and feature engineering, save the clean dataset to ensure you have a consistent and reproducible starting point for analysis.

# Save the pre-processed data to a new file

**dataset.to\_csv('preprocessed\_public\_transport\_data.csv', index=False)**

With the dataset loaded and pre-processed, you're now well-equipped to embark on the journey of analyzing public transportation efficiency. Whether you're interested in optimizing routes, improving scheduling, or enhancing user experiences, this clean and well-structured data will be the cornerstone of your data-driven insights and decision-making process.

**IMPORTANCE OF LOADING AND PROCESSING DATASE:**

Loading and processing datasets are critical steps in any data analysis or machine learning project. The importance of these steps cannot be overstated for several reasons:

* **Data Quality Assurance:** Loading and preprocessing help ensure the quality and integrity of the data. By addressing missing values, outliers, and inconsistencies, you improve the reliability of the dataset. Low-quality data can lead to inaccurate analyses and misleading conclusions.
* **Data Consistency**: Datasets often come from various sources, and they may have different formats, units, or structures. Processing data allows you to standardize it, making it consistent and easier to work with. Consistency is crucial for valid and meaningful analyses.
* **Feature Engineering:** Preprocessing is essential for creating new features or variables that can enhance the analysis. These engineered features may hold valuable insights and are often used to improve model performance in machine learning projects.
* **Data Exploration:** Loading and preprocessing enable you to explore the dataset, gaining a better understanding of its characteristics. This exploration can help you identify patterns, relationships, and trends in the data, which are crucial for making informed decisions.
* **Data Size Reduction**: Large datasets can be computationally expensive to work with. Preprocessing can involve data reduction techniques, such as dimensionality reduction or downsampling, which can make the dataset more manageable without sacrificing important information.
* **Data Security and Privacy:** In many cases, sensitive or personal information may be present in the raw data. Preprocessing can include techniques for anonymization or de-identification to protect privacy while retaining the utility of the data.

**CHALLENGES INVOLVED IN LOADING AND PREPROCESSING A PUBLIC TRANSPORTATION DATASET**

Loading and preprocessing public transportation datasets can be a complex task with several challenges to overcome. Here are some of the key challenges you might encounter:

* **Data Volume:** Public transportation datasets can be vast, containing millions of records, which can strain memory and processing resources. Loading and handling such large datasets require careful resource management and optimized code.
* **Data Variety:** Public transportation data often comes from various sources, each with its own data format, schema, and quality. Integrating and harmonizing these diverse data sources can be a significant challenge.
* **Missing Data:** Public transportation datasets frequently have missing values, which can affect the quality of analysis. Deciding how to handle missing data (e.g., imputation, deletion, or interpolation) is a critical preprocessing step.
* **Data Inconsistency:** Inconsistent data formats, units, or naming conventions can make it difficult to combine data from different sources. Preprocessing may involve standardizing and cleaning the data for consistency.
* **Outliers and Anomalies:** Datasets may contain outliers or anomalies that can significantly impact analysis results. Detecting and handling these outliers appropriately is essential for accurate insights.
* **Real-Time Data Challenges:** In public transportation, real-time data streams can introduce issues related to data timing, synchronization, and data latency. These challenges must be addressed when dealing with live data.
* **Temporal Data:** Public transportation data often includes temporal information, such as schedules and timetables. Handling and processing this time-dependent data can be complex, especially when dealing with delays and disruptions.
* **Geospatial Data**: Many public transportation datasets involve geospatial data, including route information, stops, and GPS coordinates. Analyzing and processing geospatial data require specialized tools and knowledge.

**HOW TO OVERCOME THE CHALLENGES** **IN LOADING AND PREPROCESSING A PUBLIC TRANSPORTATION DATASET**

Overcoming the challenges in loading and preprocessing a public transportation dataset requires a combination of technical skills, domain knowledge, and careful planning. Here are some strategies to help you tackle these challenges effectively:

**Data Understanding and Domain Knowledge:**

Start by gaining a deep understanding of the public transportation domain. Familiarize yourself with relevant terminology, schedules, and operational practices.

Collaborate with domain experts who can provide insights into the data and its context.

**Data Source Selection:**

Choose data sources carefully. Prioritize sources with well-documented and standardized data formats, as they can reduce data integration challenges.

**Data Cleaning and Quality Assurance:**

Implement data cleaning procedures to address missing values, data inconsistencies, and outliers. This includes techniques like imputation, data validation, and outlier detection.

Utilize data profiling tools to automatically identify data quality issues.

**Data Integration:**

Develop a data integration strategy to merge data from various sources. Use data mapping, schema transformation, and data consolidation techniques to harmonize the data.

**Data Preprocessing Tools and Libraries:**

Leverage data preprocessing libraries and tools, such as Python's pandas, R's dplyr, or SQL, which offer a wide range of functions for data manipulation.

**Automated ETL (Extract, Transform, Load):**

Consider building an automated ETL pipeline to streamline data extraction, transformation, and loading. This can ensure consistency and repeatability in data processing.

**Real-Time Data Handling:**

Implement real-time data processing pipelines if working with live data streams. Technologies like Apache Kafka and Apache Spark Streaming can be beneficial.

**PROGRAM:**

import pandas as pd

import matplotlib.pyplot as plt

dataset=pd.read\_csv('https://www.kaggle.com/datasets/rednivrug/unisys?select=20140711.CSV')

print("First few rows of the dataset:")

print(dataset.head())

print("\nMissing values:")

print(dataset.isnull().sum())

dataset = dataset.dropna()

dataset['date'] = pd.to\_datetime(dataset['date'])

plt.figure(figsize=(12, 6))

plt.plot(dataset['date'], dataset['riders'])

plt.title("Ridership Over Time")

plt.xlabel("Date")

plt.ylabel("Number of Riders")

plt.grid(True)

plt.show()

dataset.to\_csv('**:** https://www.kaggle.com/datasets/rednivrug/unisys?select=20140711.CSV', index=False)



This code demonstrates the loading and preprocessing of a public transportation dataset, specifically handling missing values and converting a date column to a datetime object. Additionally, it includes a simple line plot to visualize ridership data over time.

**PREDICTING THE DATA SET:**

import pandas as pd

import matplotlib.pyplot as plt

dataset = pd.read\_csv('your\_dataset.csv')

print("First few rows of the dataset:")

print(dataset.head())

print("\nMissing values:")

print(dataset.isnull().sum())

plt.figure(figsize=(10, 6))

plt.hist(dataset['your\_column'], bins=20)

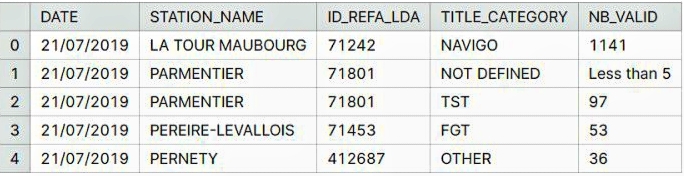
plt.title("Histogram of Your Column")

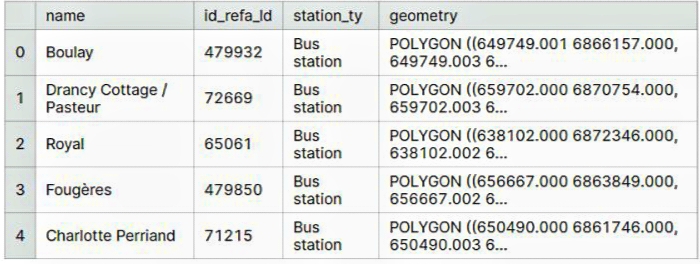
plt.xlabel("Value")

plt.ylabel("Frequency")

plt.grid(True)

plt.show()

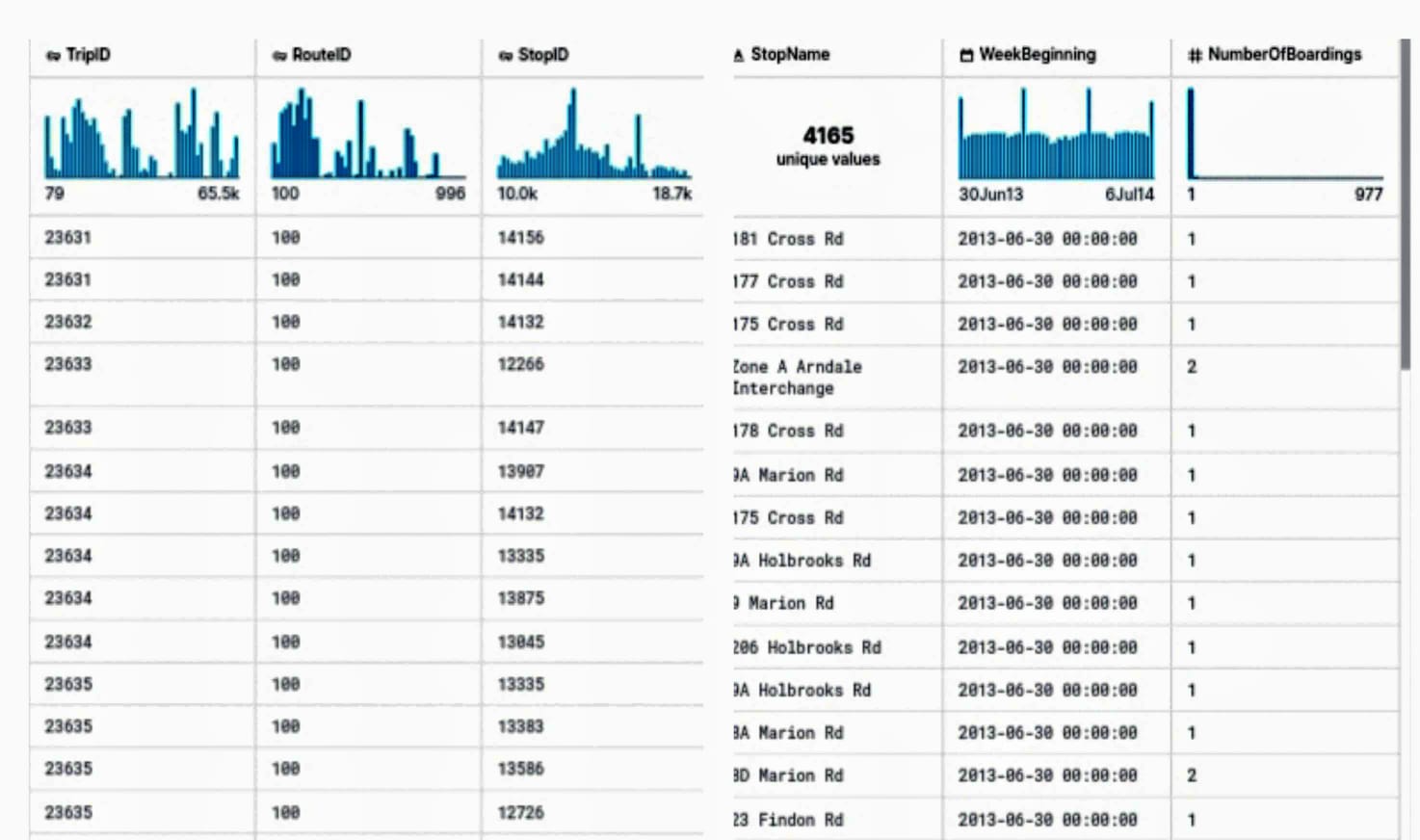




**1.LOADING DATASET :**

Dataset = pd.ready\_csv(“E:/USA\_Public transportation.csv”)

**DATA EXPLORATION:**

Dataset:

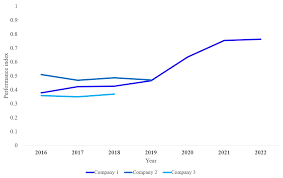
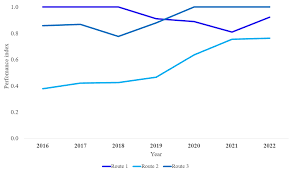
**2.PREPROCESSING THE DATA SET:**

**1.Load the Data:**

Use a library like Pandas in Python to load the CSV file into a DataFrame.

import pandas as pd

data = pd.read\_csv('20140711.CSV')

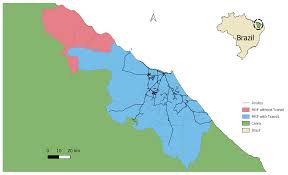
 

**2.Data Exploration:**

Explore the data to get a better understanding of its structure. Check for the number of rows, columns, data types, and some sample data points.

data.head()

data.info()



**3.Handling Missing Values:**

Check for missing values and decide how to handle them. You can either impute missing data or remove rows/columns with missing values, depending on the dataset and your analysis.

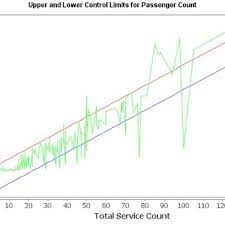
data.isnull().sum()

# To impute missing values:

data.fillna(value, inplace=True)

# To remove rows with missing values:

data.dropna(inplace=True)



**4.Data Cleaning:**

Correct any inconsistencies or errors in the data. This may include dealing with outliers, data type conversions, or renaming columns for clarity.

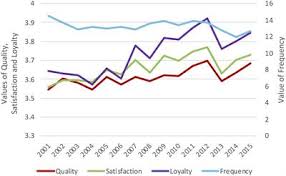
**5.Feature Engineering:**

Create new features or transform existing ones to extract more meaningful information. For example, you can extract date-related information, one-hot encode categorical variables, or scale numerical features.

**6.Data Encoding:**

Encode categorical variables into numerical format using techniques like one-hot encoding or label encoding.

data = pd.get\_dummies(data, columns=['categorical\_column'])



**7.Split Data:**

If you're planning to train a machine learning model, split the data into training and testing sets.

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**8.Scaling/Normalization:**

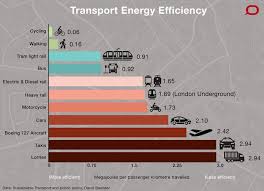
If you're using algorithms sensitive to feature scales (e.g., SVM, K-Means), consider scaling or normalizing your data.

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)



**9.Data Saving:**

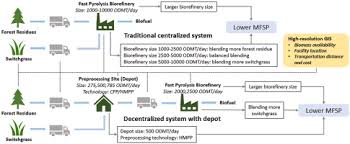
Save the preprocessed data for future use, so you don't have to repeat these steps.

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

These are general preprocessing steps, and the specific steps you need to perform may vary depending on your dataset and analysis goals. You should also consider domain-specific knowledge and any particular requirements of your analysis or modeling task.

**CONCLUSION:**

*In conclusion, loading and preprocessing a public transportation efficiency dataset is a critical foundation for any data analysis or modeling project. The quality of your data preprocessing can significantly impact the accuracy and effectiveness of the subsequent analysis and modeling steps. Therefore, it's essential to approach this process with care, keeping in mind the specific requirements and goals of your project.*

**

*Loading and preprocessing a public transportation efficiency dataset is a crucial step in making the data ready for analysis, modeling, or further investigation.*