Electric Vehicle Indian market Segmentation

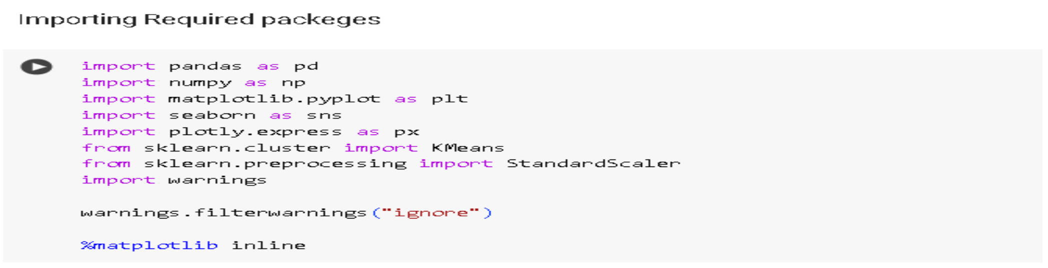
Submitted by: Kavita Lamani

Data Pre-processing:

Required libraries:

To perform data analysis, pre-processing, visualization, and clustering as discussed earlier, we'll need several Python libraries. Here's a list of the required libraries:

1. Pandas: For data manipulation and analysis, including reading data from various file formats, cleaning, and transforming data.
2. Matplotlib: For basic plotting and visualization.
3. Seaborn: For advanced statistical visualization, making plots more attractive and informative.
4. Scikit-learn: For machine learning algorithms, including clustering algorithms like K-means.
5. NumPy: For numerical computing and working with arrays.


Importing Required packeges

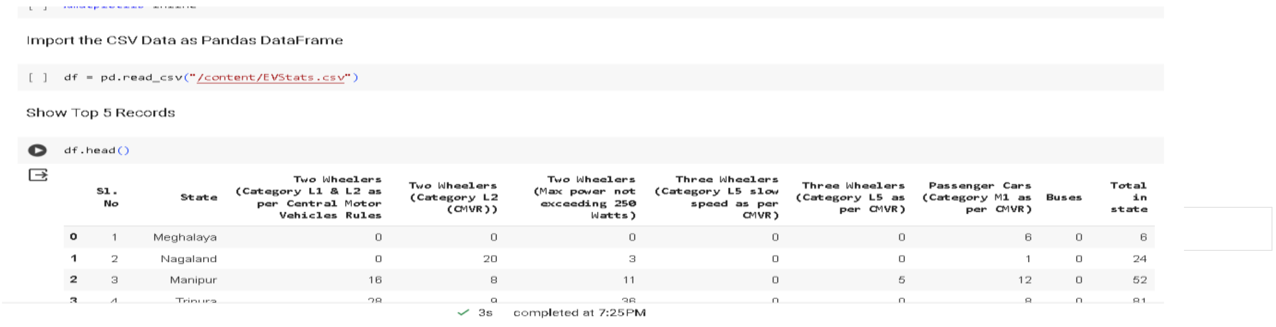
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import warnings

warnings.filterwarnings("ignore")

%matplotlib inline
```

Data Sources:

Since the data provided seems to be a sample dataset, it doesn't have a specific external source. However, for a real-world scenario, data on vehicle registrations or sales by state in India can be obtained from various sources:


Import the CSV Data as Pandas DataFrame

```
df = pd.read_csv("/content/EVStats.csv")
```

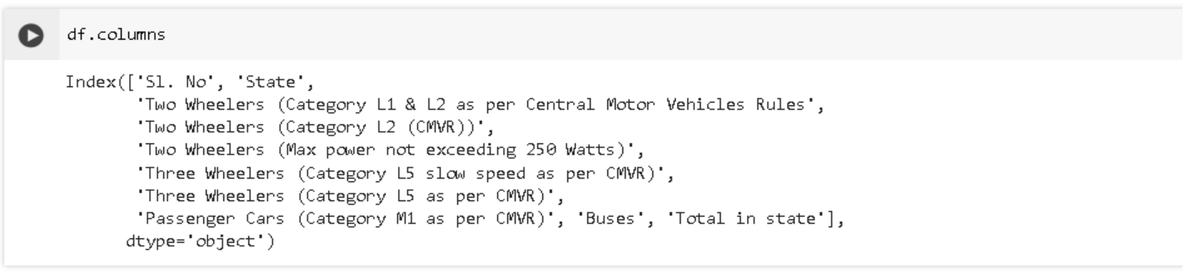
Show Top 5 Records

```
df.head()
```

	Sl. No	State	Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules	Two Wheelers (Category L2 (CMVR))	Two Wheelers (Max power not exceeding 250 Watts)	Three Wheelers (Category L5 slow speed as per CMVR)	Three Wheelers (Category L5 as per CMVR)	Passenger Cars (Category M1 as per CMVR)	Buses	Total in state
0	1	Meghalaya	0	0	0	0	0	6	0	6
1	2	Nagaland	0	20	3	0	0	1	0	24
2	3	Manipur	16	8	11	0	5	12	0	52
3	4	Tripura	28	9	36	0	0	8	0	81

✓ 3s completed at 7:25PM

We find that there are 10 attributes in the dataset (which includes categorical and numerical variables).

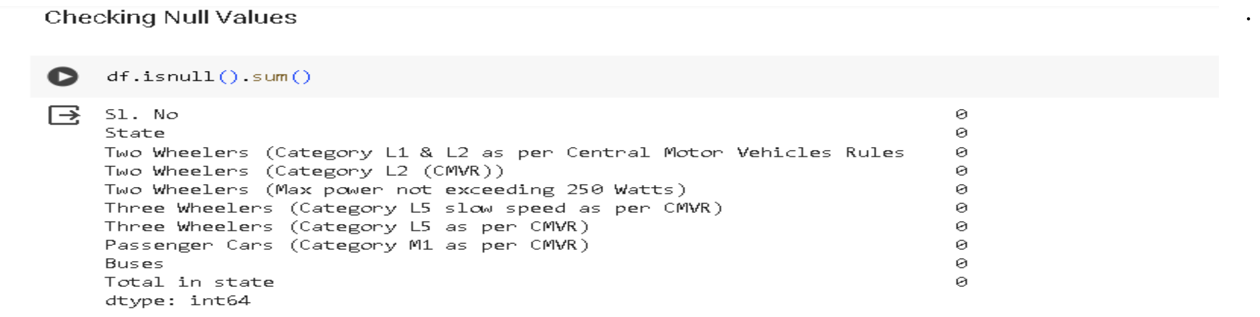


```
df.columns
```

```
Index(['Sl. No', 'State',
       'Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules',
       'Two Wheelers (Category L2 (CMVR))',
       'Two Wheelers (Max power not exceeding 250 Watts)',
       'Three Wheelers (Category L5 slow speed as per CMVR)',
       'Three Wheelers (Category L5 as per CMVR)',
       'Passenger Cars (Category M1 as per CMVR)', 'Buses', 'Total in state'],
      dtype='object')
```

Checking Null values:

Checking for null values is an essential step in data analysis to identify missing or incomplete data that may affect the quality and reliability of the analysis. Here's how you can check for null values in a dataset using Python's Pandas library:



Checking Null Values

```
df.isnull().sum()
```

```
Sl. No                                                              0
State                                                               0
Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules  0
Two Wheelers (Category L2 (CMVR))                                   0
Two Wheelers (Max power not exceeding 250 Watts)                    0
Three Wheelers (Category L5 slow speed as per CMVR)                 0
Three Wheelers (Category L5 as per CMVR)                            0
Passenger Cars (Category M1 as per CMVR)                            0
Buses                                                               0
Total in state                                                      0
dtype: int64
```

Data visualization:

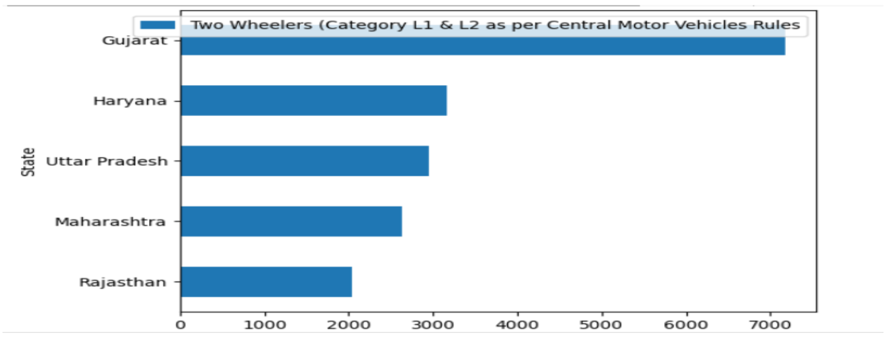
Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules) in Each State:

1. L1 & L2 Categories:

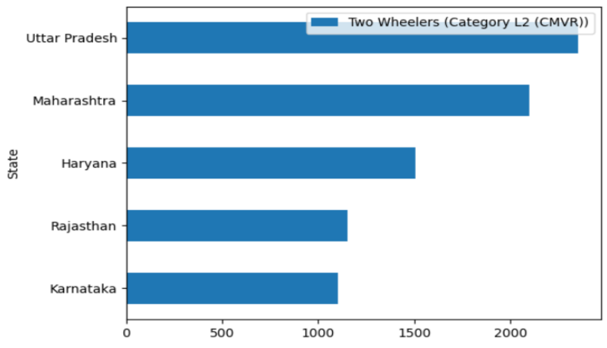
- These are specific categories defined under the Central Motor Vehicles Rules (CMVR) for two-wheeled vehicles.
- They likely represent different classes or types of two-wheelers based on characteristics such as engine displacement, vehicle weight, or design specifications.

2. Central Motor Vehicles Rules (CMVR):

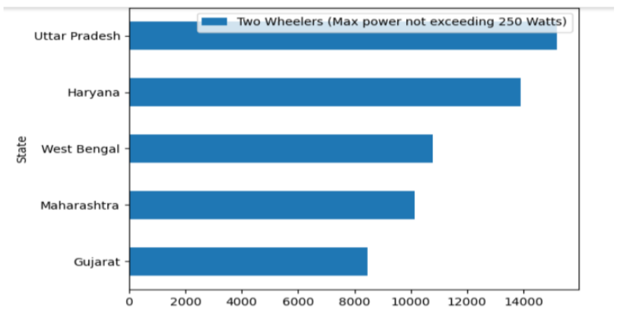
- CMVR is a set of regulations established by the Indian government to govern various aspects of motor vehicles, including their manufacture, registration, operation, and safety standards.
- Vehicles falling under the "Two Wheelers (Category L1 & L2)" classification must adhere to the specific regulations and standards outlined in the CMVR pertaining to their respective categories.



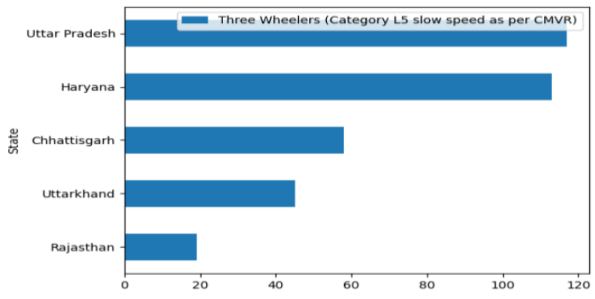
Two Wheelers (Category L2 (CMVR))



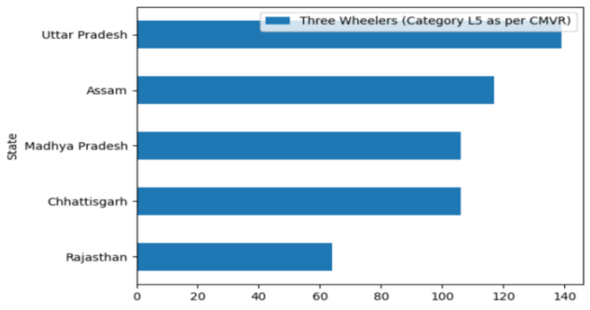
Two Wheelers (Max power not exceeding 250 Watts)



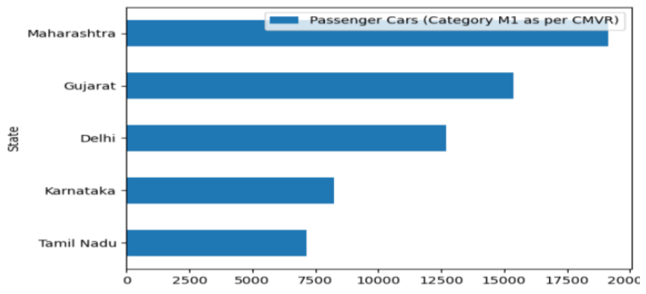
Three Wheelers (Category L5 slow speed as per CMVR)



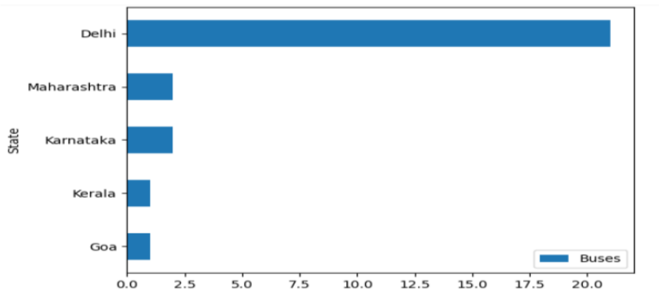
Three Wheelers (Category L5 as per CMVR)



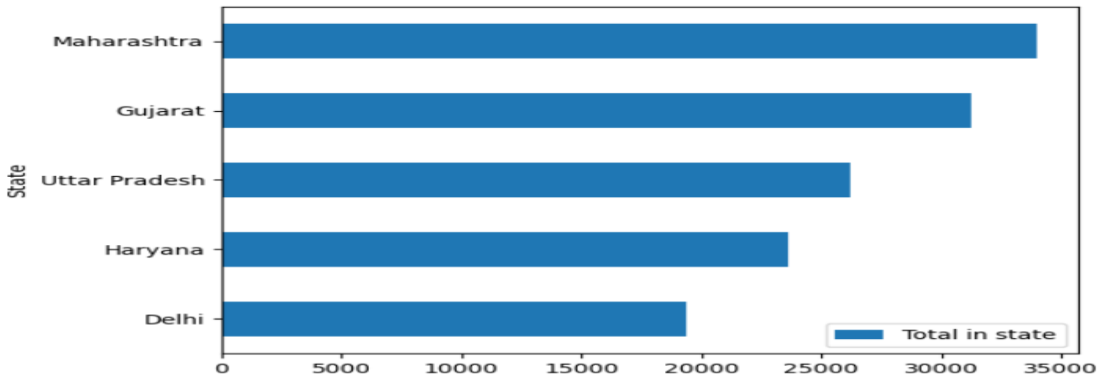
Passenger Cars (Category M1 as per CMVR)



Buses



Total Vehicle in Each State



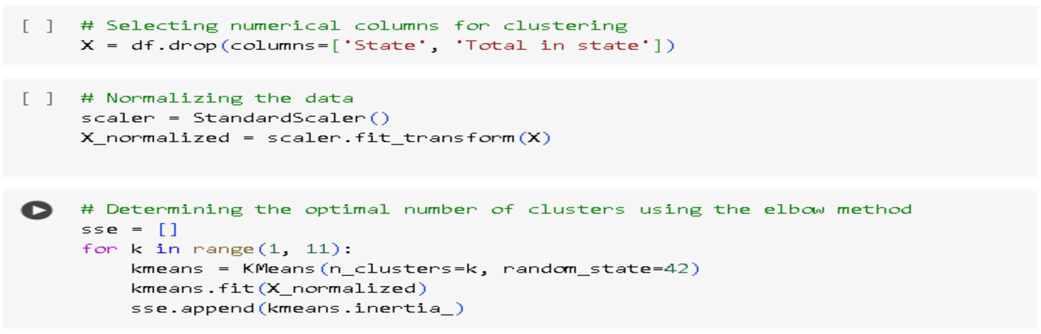
Segmentation Approach:

K-means clustering is an unsupervised machine learning algorithm used for partitioning a dataset into K distinct, non-overlapping clusters. It works by iteratively assigning data points to the nearest cluster centroid and updating the centroids based on the mean of the points assigned to each cluster. The algorithm aims to minimize the within-cluster variance, also known as inertia.

Step-by-step explanation of the K-means clustering algorithm:

1. Initialization: Choose the number of clusters K and randomly initialize K cluster centroids. These centroids can be randomly selected from the data points or initialized using other methods.
2. Assign Data Points to Clusters: For each data point, calculate the distance to each centroid and assign the point to the nearest centroid's cluster.
3. Update Cluster Centroids: After all data points have been assigned to clusters, calculate the mean of the points in each cluster and update the cluster centroids to the new mean.

4. Repeat: Iterate steps 2 and 3 until convergence criteria are met. Convergence occurs when the cluster assignments and centroids no longer change significantly between iterations or when a maximum number of iterations is reached.
5. Final Result: Once convergence is reached, the algorithm outputs K clusters, each represented by its centroid, and the data points assigned to each cluster.



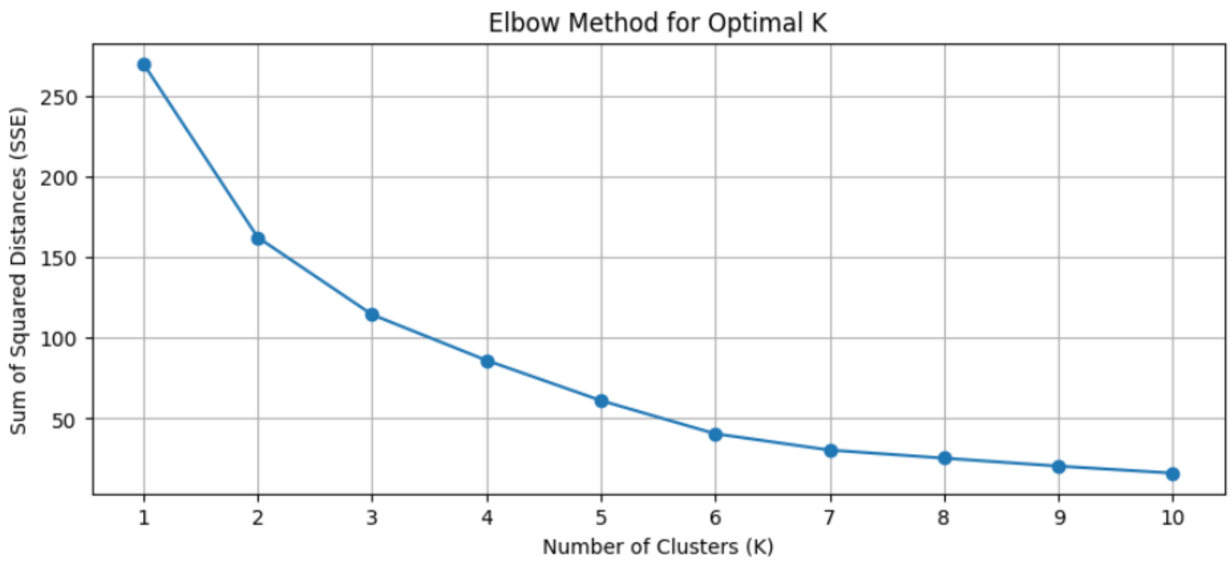
```
[ ]  # Selecting numerical columns for clustering
     X = df.drop(columns=['State', 'Total in state'])
```

```
[ ]  # Normalizing the data
     scaler = StandardScaler()
     X_normalized = scaler.fit_transform(X)
```

```
     # Determining the optimal number of clusters using the elbow method
     sse = []
     for k in range(1, 11):
         kmeans = KMeans(n_clusters=k, random_state=42)
         kmeans.fit(X_normalized)
         sse.append(kmeans.inertia_)
```

Elbow Method:

- Identification of Optimal K: The elbow method is used to determine the optimal number of clusters (K) in K-means clustering. It involves plotting the within-cluster sum of squares (inertia) as a function of the number of clusters.
- Elbow Point: The plot typically exhibits a downward trend, where the inertia decreases as the number of clusters increases. The "elbow point" is the point on the plot where the inertia starts to decrease at a slower rate.
- Selection of K: The optimal number of clusters is often chosen at the elbow point, as it represents a trade-off between maximizing the number of clusters (which reduces inertia) and avoiding over fitting by selecting too many clusters. However, sometimes the elbow point may not be clearly defined, and additional methods or domain knowledge may be required to determine the optimal K.


Elbow Method for Optimal K
Sum of Squared Distances (SSE)
Number of Clusters (K)