Electronic Vehicles Market Segmentation

By Nitya Ravi

Abstract: *This project leverages machine learning methodologies to segment the electric vehicle (EV) market for a burgeoning startup. Through the application of clustering algorithms and predictive modeling techniques on a rich array of data sources encompassing demographics, geographical information, and behavioural patterns, the project identifies discrete customer segments within the EV market landscape. By prioritizing interpretability, the methodology delivers actionable insights, empowering the startup to craft targeted marketing initiatives and refine product strategies tailored to the unique preferences and needs of each identified segment. This strategic approach equips the startup with the tools needed to efficiently target and engage customers, ultimately enhancing its competitive edge in the dynamic and rapidly evolving EV market.*

GitHub ID: #82902477

GitHub Link: NityaRavi12 (github.com)

GitHub Repository link: NityaRavi12/Feynn-Labs-Task-1- (github.com)

➢ Introduction:

- The electric vehicle (EV) market is currently experiencing unprecedented growth, fuelled by a confluence of factors including escalating environmental concerns, government incentives promoting sustainable transportation, and remarkable technological advancements. As a result, the landscape has become increasingly crowded with both established players and emerging startups vying for market share. In this fiercely competitive environment, differentiation is crucial for EV startups to carve out their niche and thrive. However, traditional demographic-based segmentation methods often fall short in capturing the intricacies of this dynamic and diverse market. As such, there is a growing recognition of the need for more sophisticated segmentation strategies, with machine learning emerging as a powerful tool to meet this demand.
- Machine learning offers a transformative approach to market segmentation by analyzing vast datasets to uncover subtle patterns and relationships that may not be apparent through traditional methods. By examining a multitude of variables, including demographics, psychographics, and behavioral data, machine learning algorithms can identify distinct market segments with a level of granularity and precision that was previously unattainable. This granular understanding enables startups to tailor their marketing campaigns and product offerings to the specific needs and preferences of each segment, thereby maximizing the effectiveness of their strategies.
- The interpretability of the models is one of the main benefits of using machine learning for market segmentation in the electric vehicle business. Machine learning algorithms, as opposed to black-box techniques, offer useful insights into the fundamental causes of segment membership. Startups are empowered by this interpretability to devise focused tactics that appeal to their intended audience and make well-informed judgments. The machine learning models can provide valuable insights into any part of an electric vehicle startup's business strategy, including messaging optimization to cater to diverse client segments and feature optimization.

➢ Problem Statement:

An essential choice for an Indian electric vehicle (EV) startup is who to target. The Indian market is vast and full of opportunities, but picking the correct market niche is crucial to success. This project uses market segmentation analysis to address this problem. Through comprehension of various consumer segments (geographic, demographic, etc.), we'll evaluate the competitors and their requirements. Finding the "sweet spot"—the market most likely to adopt the startup's electric vehicles—will be made easier with this aid. We will create a targeted entrance strategy that positions the firm for long-term success in the dynamic Indian market, taking into account data restrictions and market dynamics.

➢ Data Collection:

For our EV startup's Indian launch, I began with the data collecting activities in order to initiate the implementation of the market segmentation analysis. To obtain adequate and pertinent data for the project, I conducted thorough research and examined a variety of online data sources. This thorough data-gathering procedure establishes the setting of the stage for the next critical phase, which is to determine which market sector has the most chance of success for our startup's successful entry into the booming Indian EV market.

- https://www.kaggle.com/
- https://data.gov.in/
- https://datasetsearch.research.google.com/
- https://trends.google.com/trends/explore
- https://www.statista.com/statistics/1264923/india-electric-passenger-vehicle-sales-by-manufacturers/
- https://dataspace.mobi/dataset/electric-vehicle-charging-station-list

Analysing this data would help in making proper decisions about which states/cities to target that have already established/planned supporting infrastructure for electric vehicles.

➢ Code Implementation:

Importing the necessary libraries

```
[1] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    import statsmodels.api as sm
    import plotly.express as px
    import seaborn as sns
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler,PowerTransformer
    from sklearn.model_selection import cross_validate
    from sklearn.linear_model import LinearRegression,LogisticRegression
    from sklearn.decomposition import PCA
    from sklearn.cluster import KMeans
```

```
[2] %matplotlib inline
    from tqdm import tqdm
    from google.colab import files
    %pip install kaleido
    import kaleido
    from scipy.cluster.hierarchy import dendrogram, linkage
    from sklearn.cluster import KMeans, MeanShift, estimate_bandwidth
    from sklearn.datasets import make_blobs
    from yellowbrick.cluster import KElbowVisualizer, SilhouetteVisualizer, InterclusterDistance
    from collections import Counter
    from sklearn import metrics
    from sklearn.metrics import r2_score,silhouette_score,confusion_matrix,accuracy_score
    pd.set_option("display.precision",3)
    np.set_printoptions(precision=5, suppress=True)
    pd.options.display.float_format = '{:.4f}'.format
    import plotly.io as pio
```

```
Collecting kaleido
  Downloading kaleido-0.2.1-py2.py3-none-manylinux1_x86_64.whl (79.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.9/79.9 MB 8.8 MB/s eta 0:00:00
Installing collected packages: kaleido
Successfully installed kaleido-0.2.1
```

❖ EV Market India (Dataset1):

Data Loading and Preprocessing:

```
[3]  pio.renderers.default = "svg"
```

```
[4]  df = pd.read_csv('/content/data (1).csv')
     df.head()
```

	Unnamed: 0	Brand	Model	AccelSec	TopSpeed_KmH	Range_Km	Efficiency_WhKm	FastCharge_KmH	RapidCharge
0	0	Tesla	Model 3 Long Range Dual Motor	4.6000	233	450	161	940	Yes
1	1	Volkswagen	ID.3 Pure	10.0000	160	270	167	250	No
2	2	Polestar	2	4.7000	210	400	181	620	Yes
3	3	BMW	iX3	6.8000	180	360	206	560	Yes
4	4	Honda	e	9.5000	145	170	168	190	Yes

```
[11] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103 entries, 0 to 102
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Brand            103 non-null    object 
 1   Model            103 non-null    object 
 2   AccelSec         103 non-null    float64
 3   TopSpeed_KmH     103 non-null    int64  
 4   Range_Km         103 non-null    int64  
 5   Efficiency_WhKm  103 non-null    int64  
 6   FastCharge_KmH   103 non-null    int64  
 7   RapidCharge      103 non-null    int64  
 8   PowerTrain       103 non-null    object 
 9   PlugType         103 non-null    object 
 10  BodyStyle        103 non-null    object 
 11  Segment          103 non-null    object 
 12  Seats            103 non-null    int64  
 13  PriceEuro        103 non-null    int64  
 14  inr(10e3)        103 non-null    float64
dtypes: float64(2), int64(7), object(6)
memory usage: 12.2+ KB
```

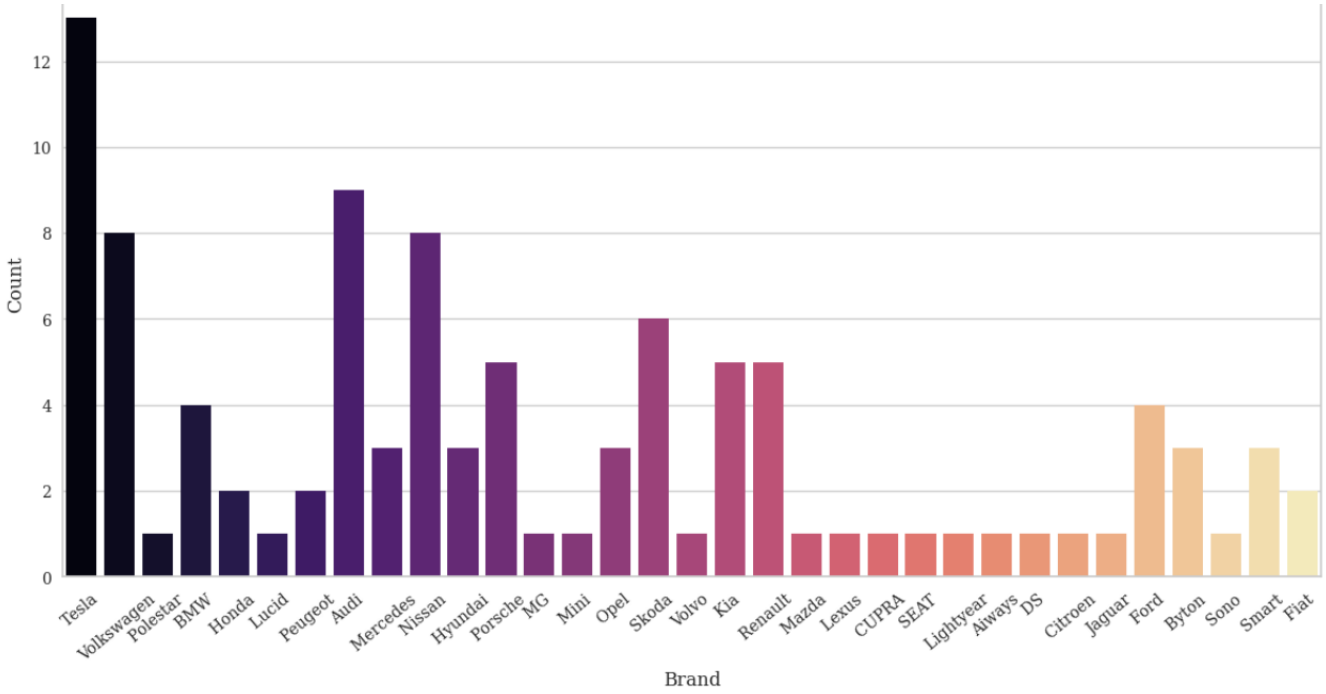
Exploring the Data

Number of EV Models produced by each brand

```
[14] sns.catplot(data=df, x='Brand', kind='count', palette='magma', height=6, aspect=2)
     sns.despine(right=False, top=False)
     plt.tick_params(axis='x', rotation=40)
     plt.xlabel('Brand',family='serif', size=12)
     plt.ylabel('Count', family='serif', size=12)
     plt.xticks(family='serif')
     plt.yticks(family='serif')
     plt.title('Number of EV Models Manufactured by Each Brand', family='serif', size=15)
     plt.show()
```

```
<ipython-input-14-1d4b3d8a8c8f>:1: FutureWarning:


Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variabl
```

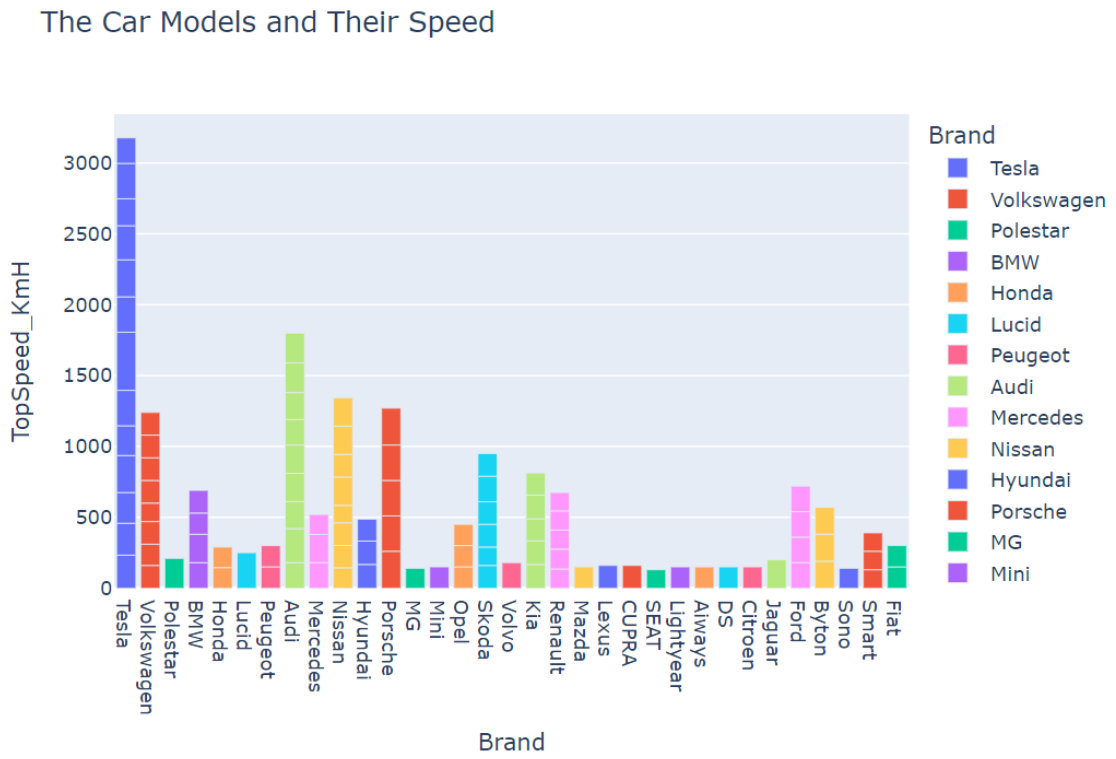
[14]



Which car has high speed?

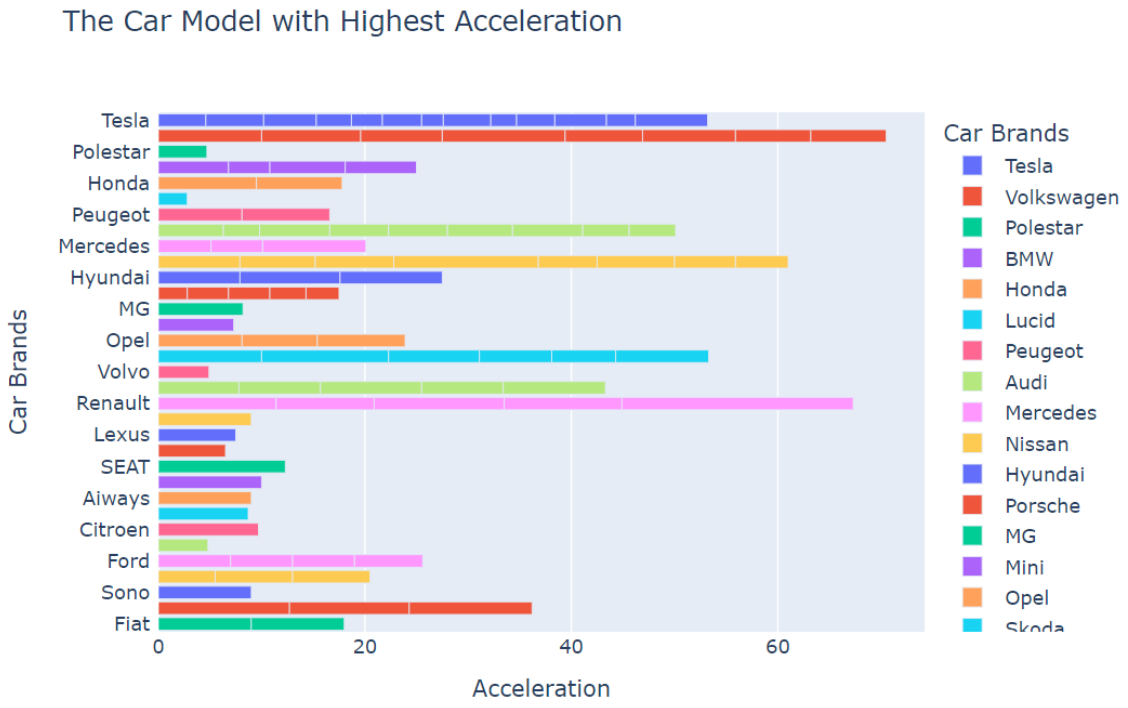
```
[15] import plotly.io as pio
     fig = px.bar(df,x='Brand',y = 'TopSpeed_KmH',color = 'Brand',title = 'The Car Models and Their Speed',labels
     pio.show(fig)
```

[15]



Which Car has the fastest acceleration?

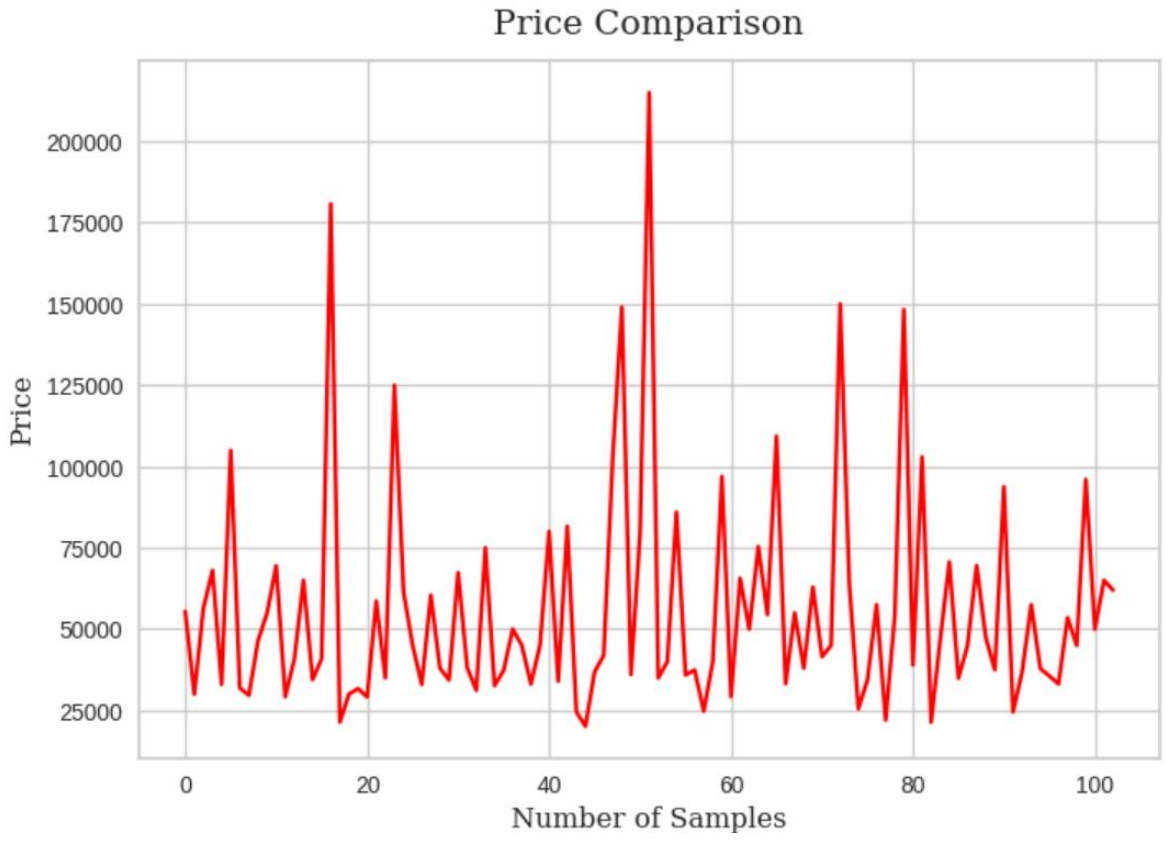
```
[16] import plotly.express as px
     import plotly.io as pio

     # Create bar graph using Plotly Express
     fig = px.bar(df, x='AccelSec', y='Brand', color='Brand',
                  orientation='h',
                  title='The Car Model with Highest Acceleration',
                  labels={'AccelSec': 'Acceleration', 'Brand': 'Car Brands'})

     # Show the plot
     pio.show(fig)
```



Price Comparison

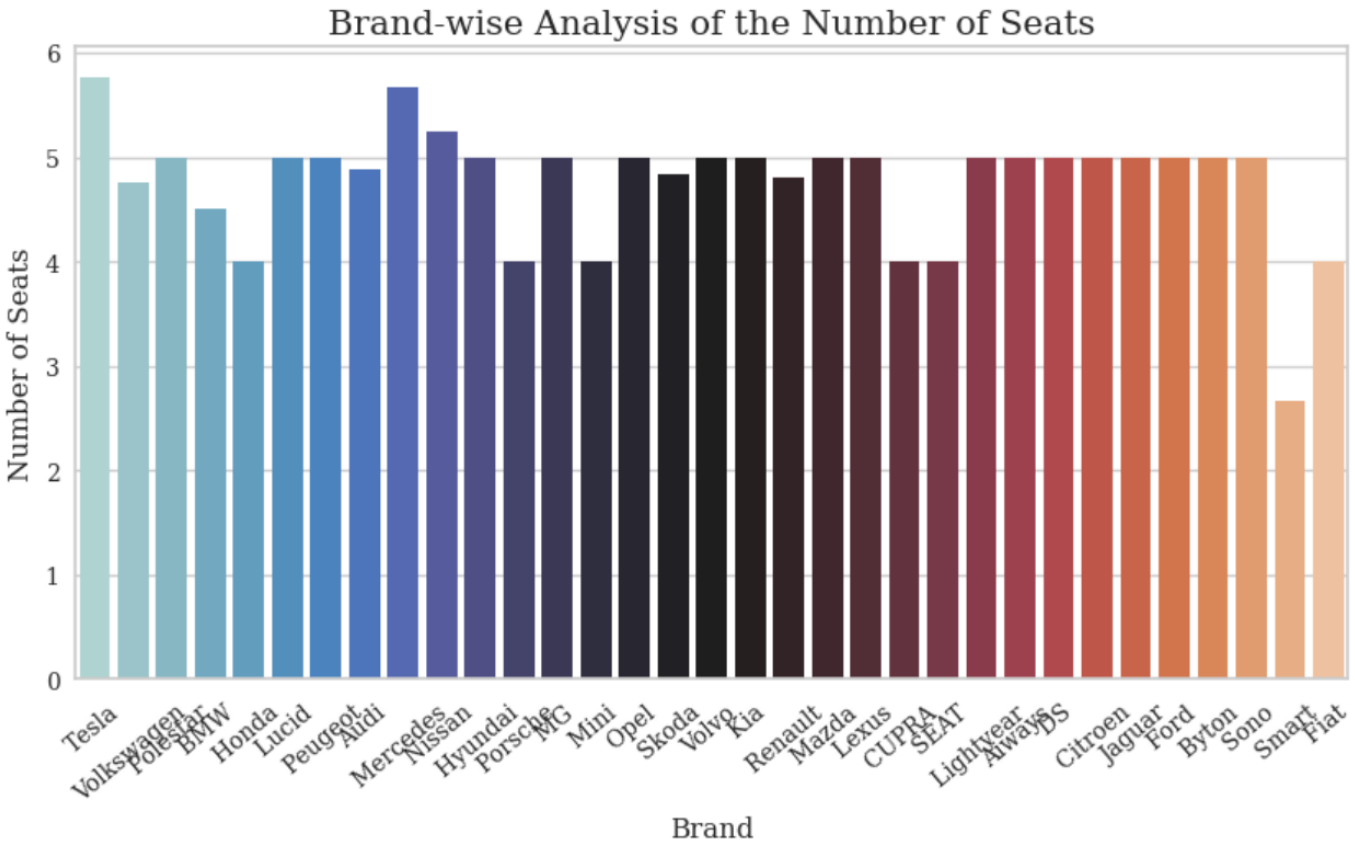
```
[17] plt.subplot(111)
     plt.plot(df['PriceEuro'], color='red')
     plt.xlabel('Number of Samples', family='serif', size=12)
     plt.ylabel('Price', family='serif', size=12)
     plt.title('Price Comparison', family='serif', size=15, pad=12)
```

```
Text(0.5, 1.0, 'Price Comparison')
```



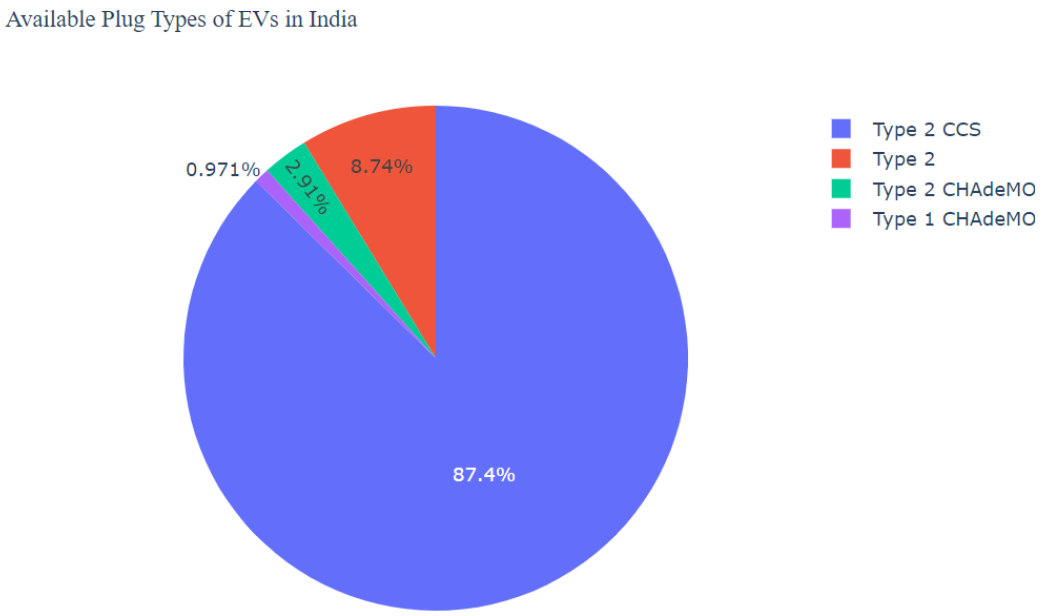
Brand-wise analysis of the number of seats

```
[18] fig, ax = plt.subplots(figsize=(10, 5))
     sns.barplot(x='Brand', y='Seats', data=df, palette='icefire', ci=None, ax=ax)
     sns.despine(right=False, top=False, ax=ax)
     plt.sca(ax)
     plt.tick_params(axis='x', rotation=40)
     plt.xlabel('Brand',family='serif', size=12)
     plt.ylabel('Number of Seats', family='serif', size=12)
     plt.xticks(rotation=40, family='serif')
     plt.yticks(family='serif')
     plt.title('Brand-wise Analysis of the Number of Seats', family='serif', size=15)
```

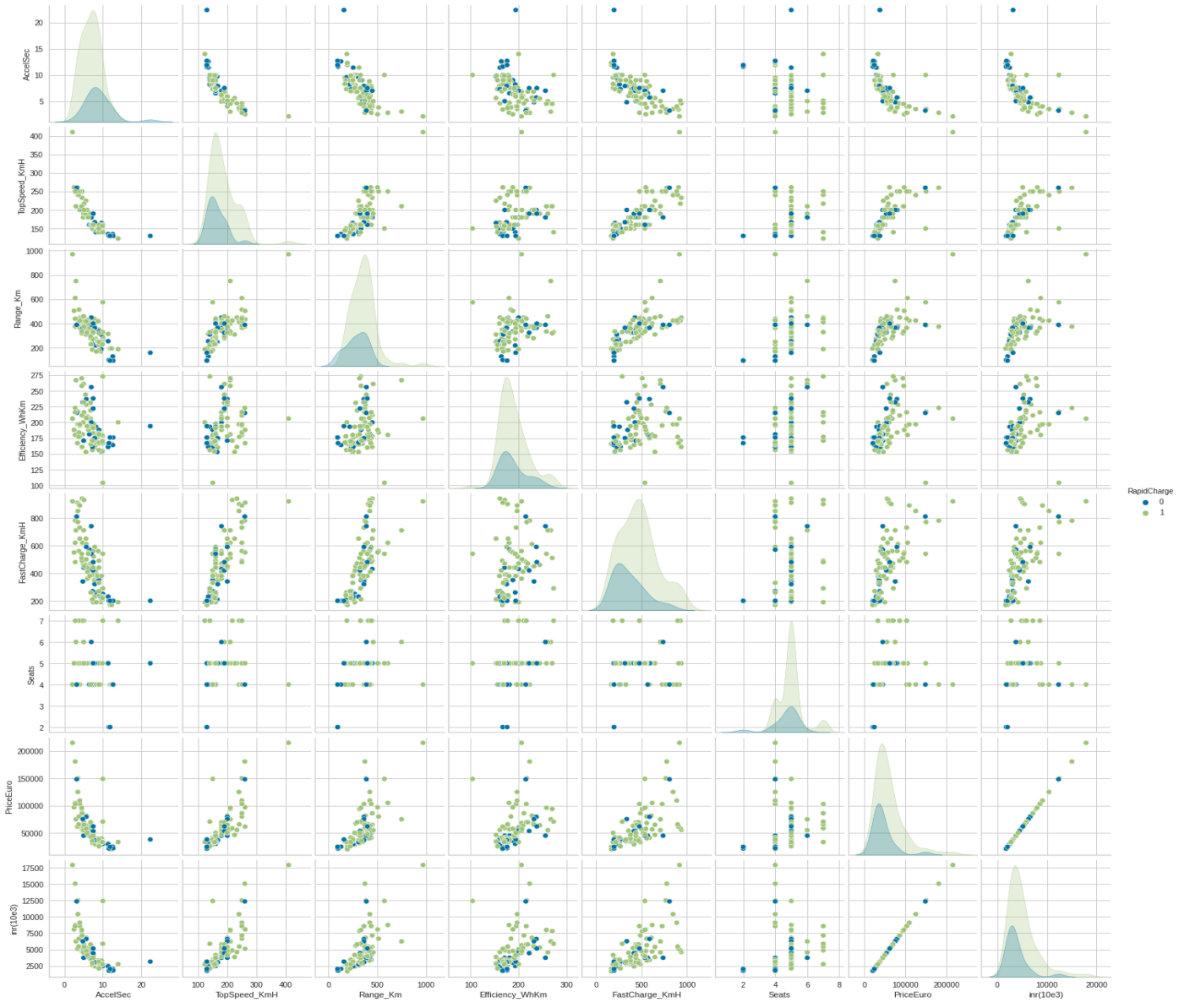


Available plug types of EVs in India

```
[21] import plotly.express as px

     # Get the value counts of 'PlugType'
     plugtype_counts = df['PlugType'].value_counts()

     # Convert to DataFrame for easier manipulation
     plugtype_counts_df = plugtype_counts.reset_index()
     plugtype_counts_df.columns = ['Plug Type', 'Count']

     # Create pie chart using Plotly Express
     fig = px.pie(plugtype_counts_df, names='Plug Type', values='Count',
                  title='Available Plug Types of EVs in India')

     # Customize labels and title
     fig.update_layout(
         title=dict(font=dict(family='serif', size=15))
     )
```



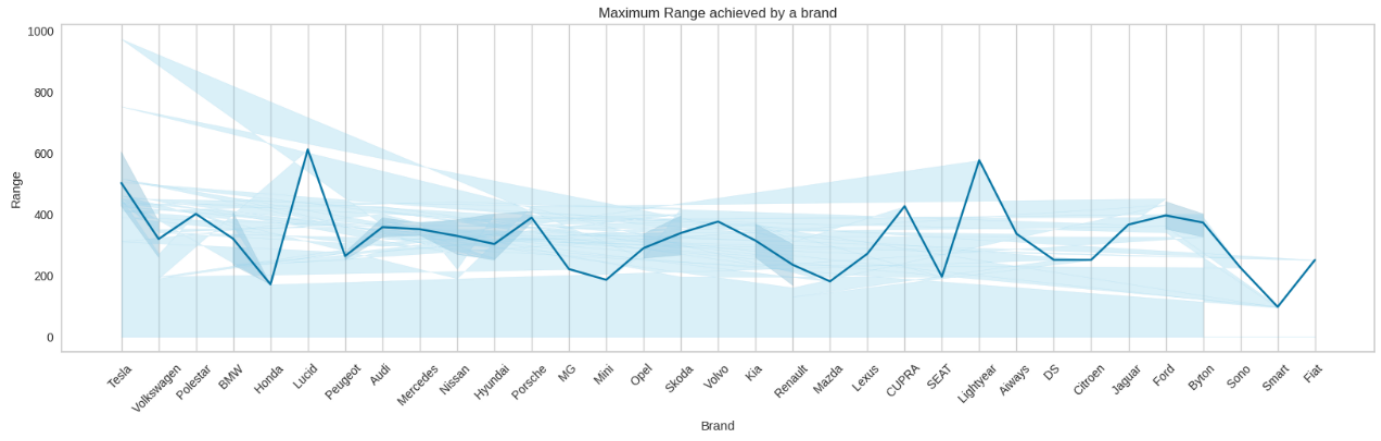
<u>One of the most crucial factors in the market for electric vehicles is charging:</u>



Maximum Range Achieved by each brand

```
[23] import seaborn as sns
     import matplotlib.pyplot as plt

     plt.figure(figsize=(20, 5))
     sns.lineplot(x='Brand', y='Range_Km', data=df, palette='tab10', sort=False)
     plt.fill_between(df['Brand'], df['Range_Km'], color='skyblue', alpha=0.3)
     plt.grid(axis='y')
     plt.title('Maximum Range achieved by a brand')
     plt.xlabel('Brand')
     plt.ylabel('Range')
     plt.xticks(rotation=45)
     plt.show()
```

```
Ignoring `palette` because no `hue` variable has been assigned.
```



Correlation Matrix:

A correlation matrix is essentially a tabular representation showcasing the correlation between variables. It's most effective when applied to variables demonstrating a linear relationship. The coefficients in the matrix illustrate the correlations among different variables. Typically, the correlation coefficient falls between -1 and 1.

Strong negative correlations, shown by values near -1, imply that when one variable rises, the other tends to fall.

Strong positive correlations, denoting values near 1, suggest that a rise in one variable also tends to increase the other.

If there is no linear relationship between the variables, the correlation coefficient is 0.

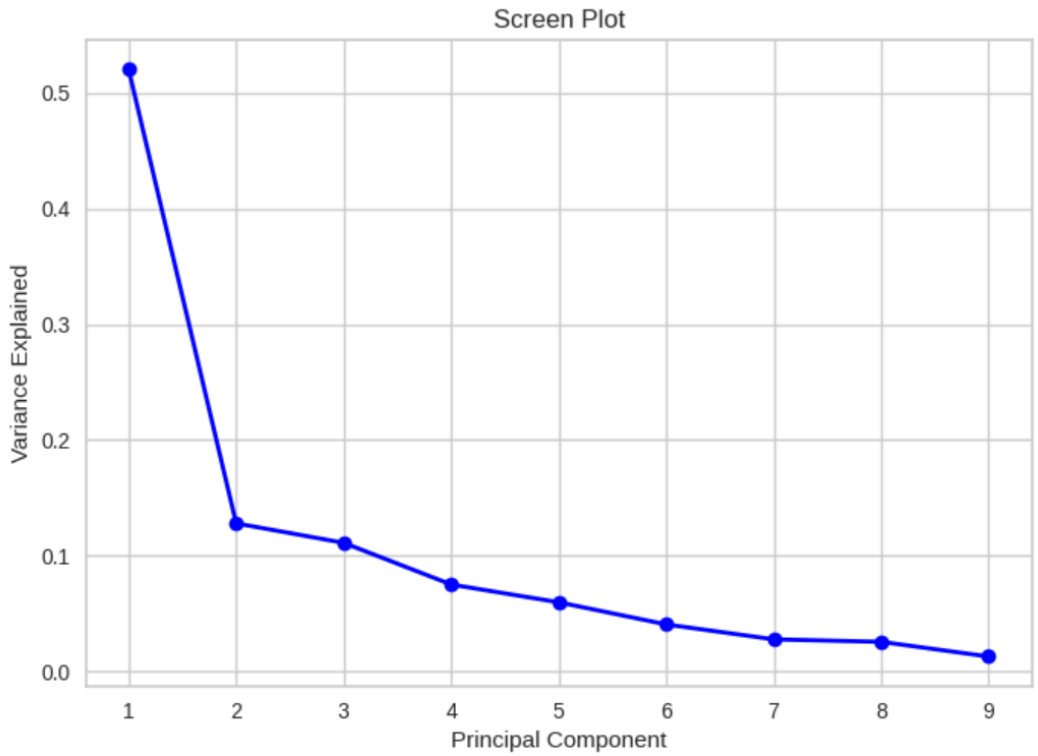
This correlation matrix is typically visualized using a heatmap, as depicted in the figure below.

Screen Plot:

The Scree Plot is a useful visual tool for figuring out how many Principal Components (PCs) to keep. The eigenvalues of each PC are plotted on a simple line graph in this plot, with the number of factors on the x-axis and the eigenvalues on the y-axis. Plots typically show a downward curve, peaking at a high point on the left, swiftly falling, and eventually plateauing.

Finding the "elbow" in the curve—the point at which the decrease levels off—is necessary to meet the Scree Plot requirement. This figure is the ideal number of PCs to hold onto. Furthermore, the Proportion of variation Plot is employed to guarantee that the chosen PCs account for a minimum of 80% of the overall variation in the data.

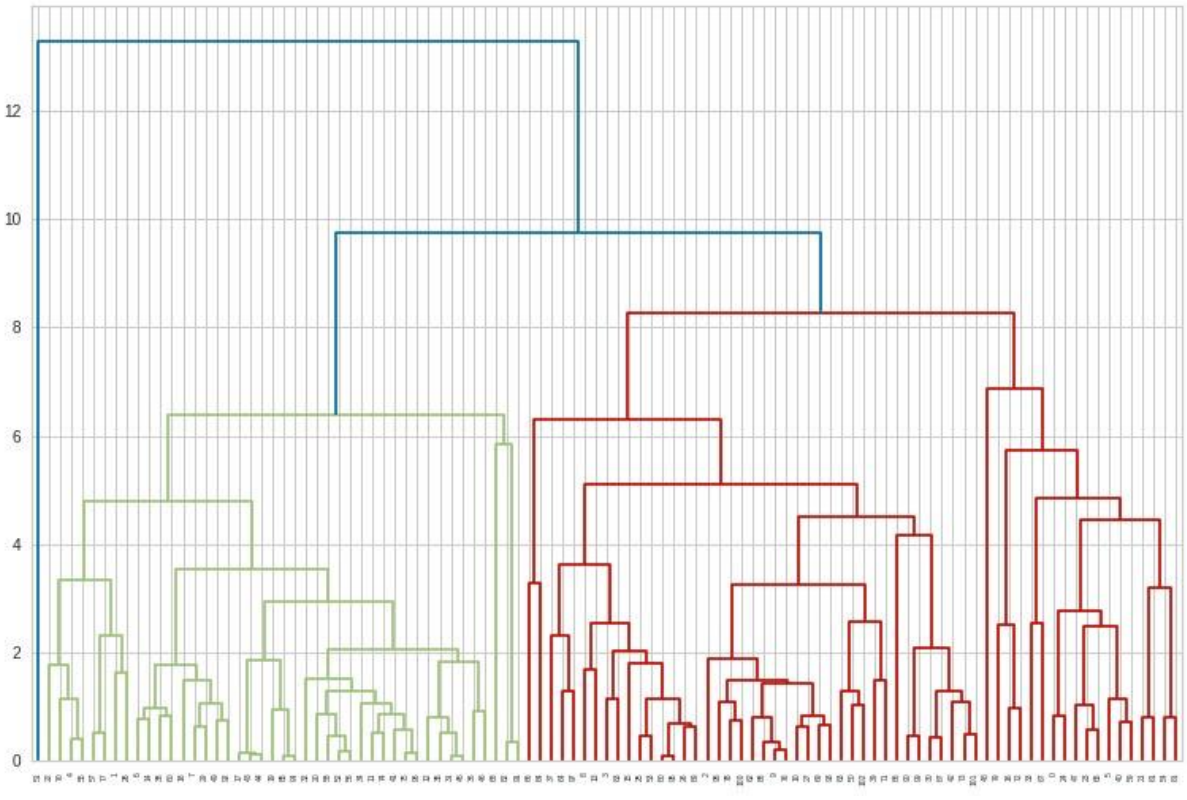
[43]



➢ Extracting Segments:

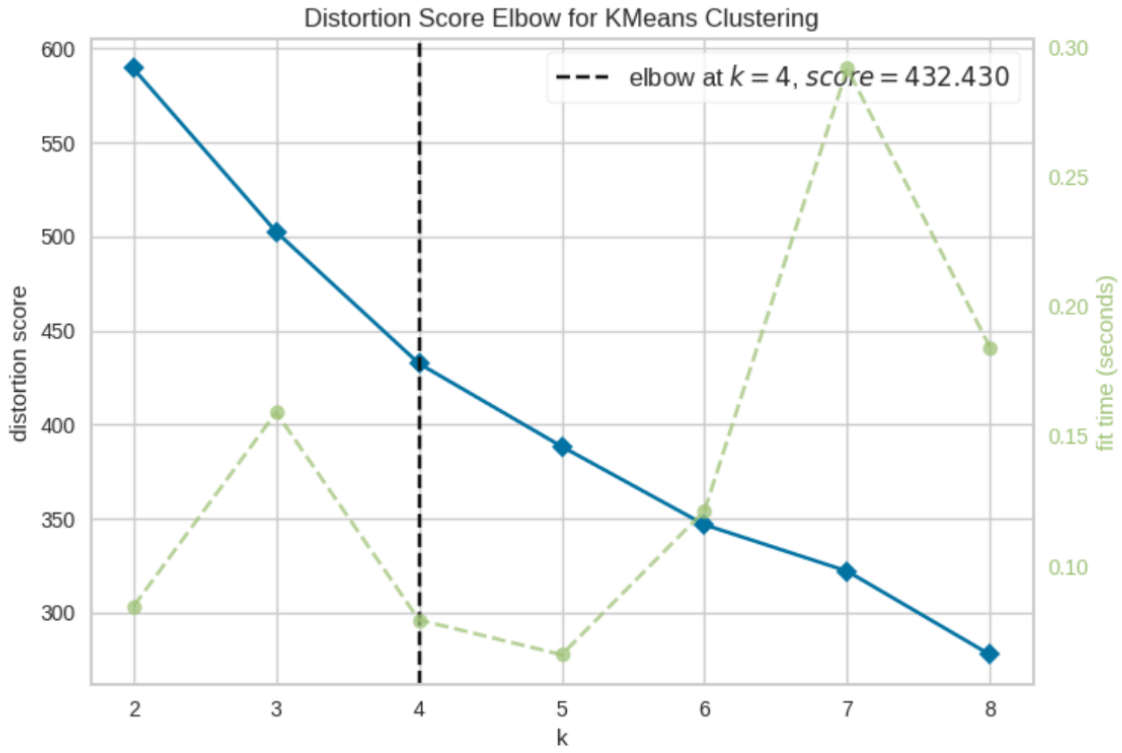
Dendrogram:

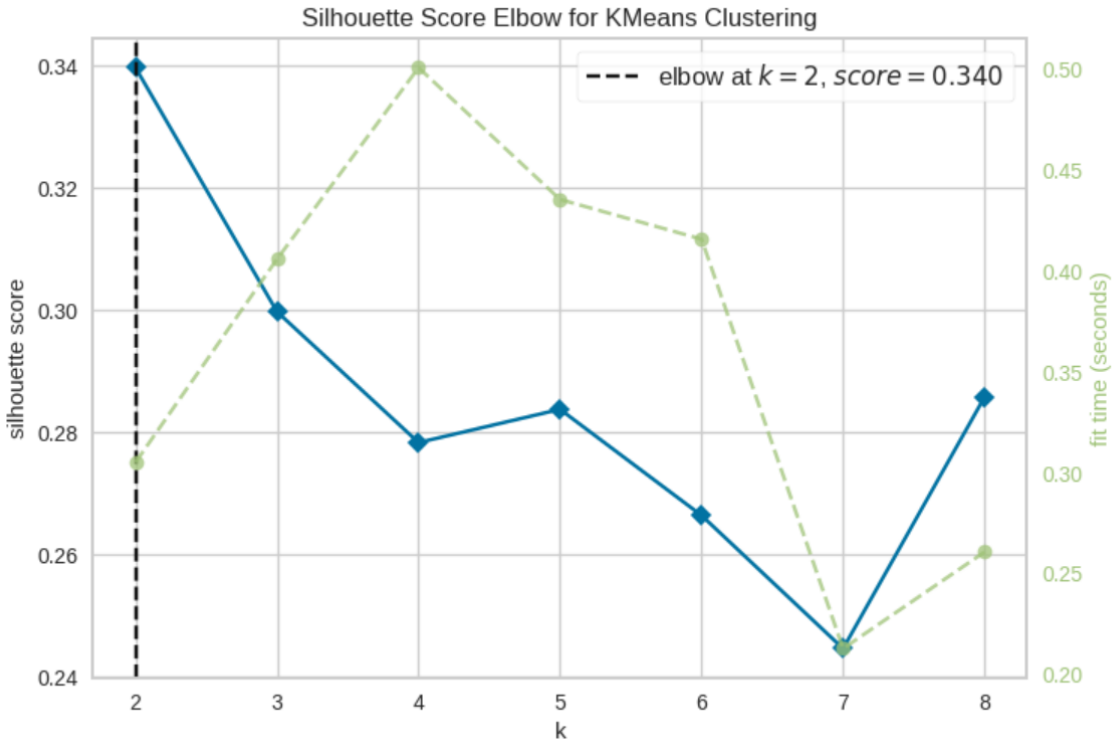
This method is designed especially for the agglomerative hierarchical clustering approach. With this approach, every data point begins as a separate cluster, and clusters are gradually combined hierarchically according to their distances from one another. A dendrogram, which is a tree-like graphic that shows the order of cluster breaks or merges, is used to calculate the ideal number of clusters for hierarchical clustering. The agglomerative hierarchical approach often recommends taking into account four to five clusters for effective clustering, in line with other cluster validation measures.

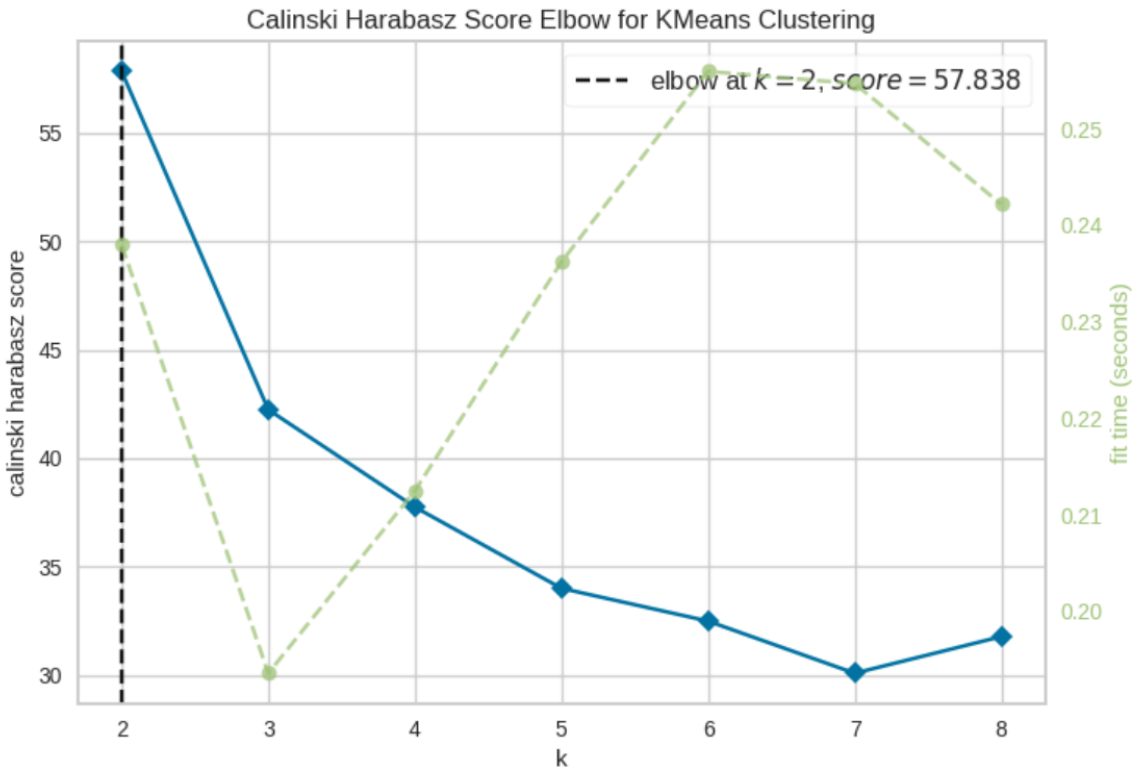


Elbow Method:

One popular method for figuring out how many clusters is the ideal number in a dataset is the Elbow Method. It works by figuring out the point at which the change in WSS considerably diminishes for a certain number of clusters (k) by computing the Within-Cluster-Sum of Squared Errors (WSS). Furthermore, the function gives information on how long it takes to create models for different cluster numbers, as shown by the green line.







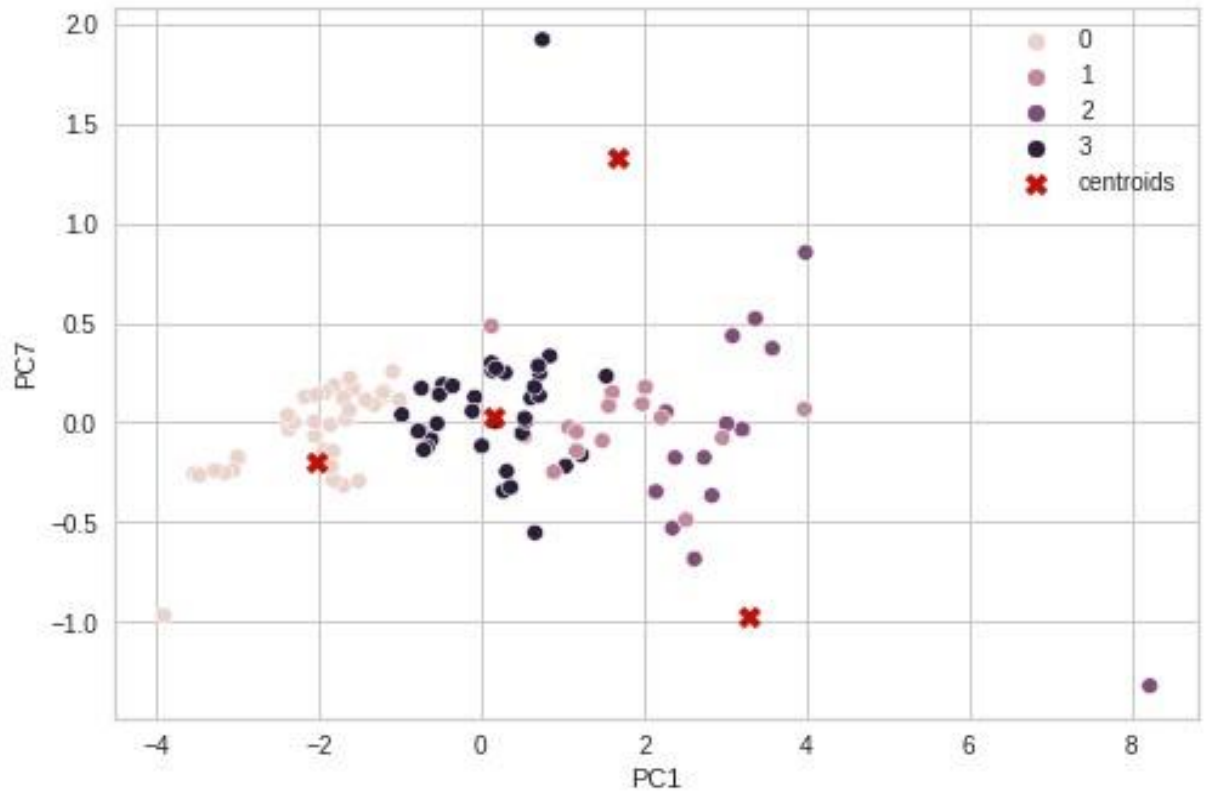
Methodology:

K-Means Algorithm

The K Means algorithm is an iterative process that seeks to divide the dataset into unique, non-overlapping subgroups that are predefined, with each data point belonging to a single group. It attempts to maintain as much difference between the clusters as well as as much similarity between the intra-cluster data points. It groups data points into clusters so that the total squared distance between the cluster's centroid and the data points is as small as possible. The homogeneity of data points within a cluster increases with decreasing variation within the cluster.

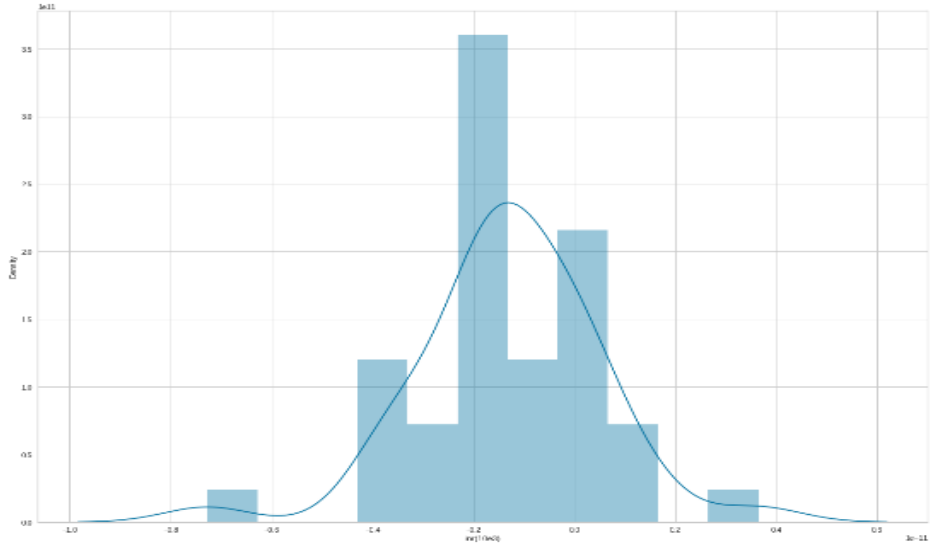
```
[41] kmeans = KMeans(n_clusters=4, init='k-means++', random_state=0).fit(t)
     df['cluster_num'] = kmeans.labels_ #adding to df
     print (kmeans.labels_) #Label assigned for each data point
     print (kmeans.inertia_) #gives within-cluster sum of squares.
     print(kmeans.n_iter_) #number of iterations that k-means algorithm runs to get a minimum within-cluster sum o
     print(kmeans.cluster_centers_) #Location of the centroids on each cluster.
```

```
[0 3 2 1 1 0 3 3 1 2 2 1 1 2 3 1 0 1 3 1 1 2 1 0 0 1 1 2 3 3 2 1 1 2 1 1 1
 3 3 2 0 1 2 1 1 1 1 0 0 3 2 0 1 1 2 1 1 3 1 0 3 2 2 2 3 0 1 2 3 2 1 2 0 2
 1 1 2 3 2 0 1 2 3 1 2 1 2 2 2 1 2 3 3 2 1 1 1 3 1 2 2 2 2]
427.887468451736
5
[[ 3.38081 -1.38223 -0.36489  0.10477  0.40601  0.27185  0.242   -0.10662
   0.04313]
 [-1.28035  0.15751 -0.8038   0.03883 -0.26171  0.05765 -0.02518 -0.04843
  -0.00967]
 [ 1.4734   0.75533  0.44439  0.22305  0.00588 -0.20901 -0.0464   0.13708
   0.00747]
 [-2.16662 -0.64972  1.15112 -0.52704  0.2495   0.04766 -0.03575 -0.0585
  -0.02224]]
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to su
```



Prediction of Prices most used cars

After completion of training the model process, we test the remaining 60% of data on the model. The obtained results are checked using a scatter plot between predicted values and the original test data set for the dependent variable and acquired similar to a straight line as shown in the figure and the density function is also normally distributed.



Model Development:

Linear Regression

A basic statistical method for simulating the relationship between a dependent variable and one or more independent variables is called linear regression. The goal of linear regression is to find the best-fitting linear equation that predicts the value of the dependent variable from the values of the independent variables.

```
[23] X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3,random_state=365)
```

```
[24] from sklearn.pipeline import make_pipeline

     lr_pipe = make_pipeline(LinearRegression())
     lr_pipe.fit(X_train, y_train)
     pred = lr_pipe.predict(X_test)
```

```
[25] r2 = r2_score(y_test, pred)
     print(f'{r2 * 100:.2f}')

     78.77
```

```
[26] y1=df[['RapidCharge']]
     x1=df[['PriceEuro']]
```

```
[27] X1_train, X1_test, y1_train, y1_test = train_test_split(x1, y1, test_size=0.2,random_state=365)
```

```
[28] log= LogisticRegression()
     log.fit(X1_train, y1_train)
     pred1 = log.predict(X1_test)
     pred1

    r/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning:

    olumn-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for exam

    ay([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

➢ Conclusion:

So, from the analysis we can see that the optimum targeted segment should belong to the following categories:

Behavioural: Mostly from our analysis there are cars with 5 seats.

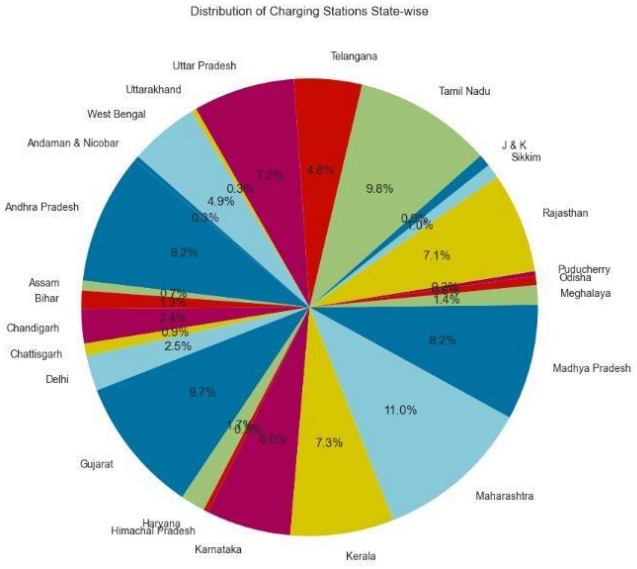
Demographic:

• Top Speed & Range: In a vast market, the price of cars is mostly determined by their top speeds and maximum range.

• Efficiency: The majority of the segments exhibit maximum efficiency.

Price: From the above analysis, the price range is between 16,00,000 to 1,80,00,000

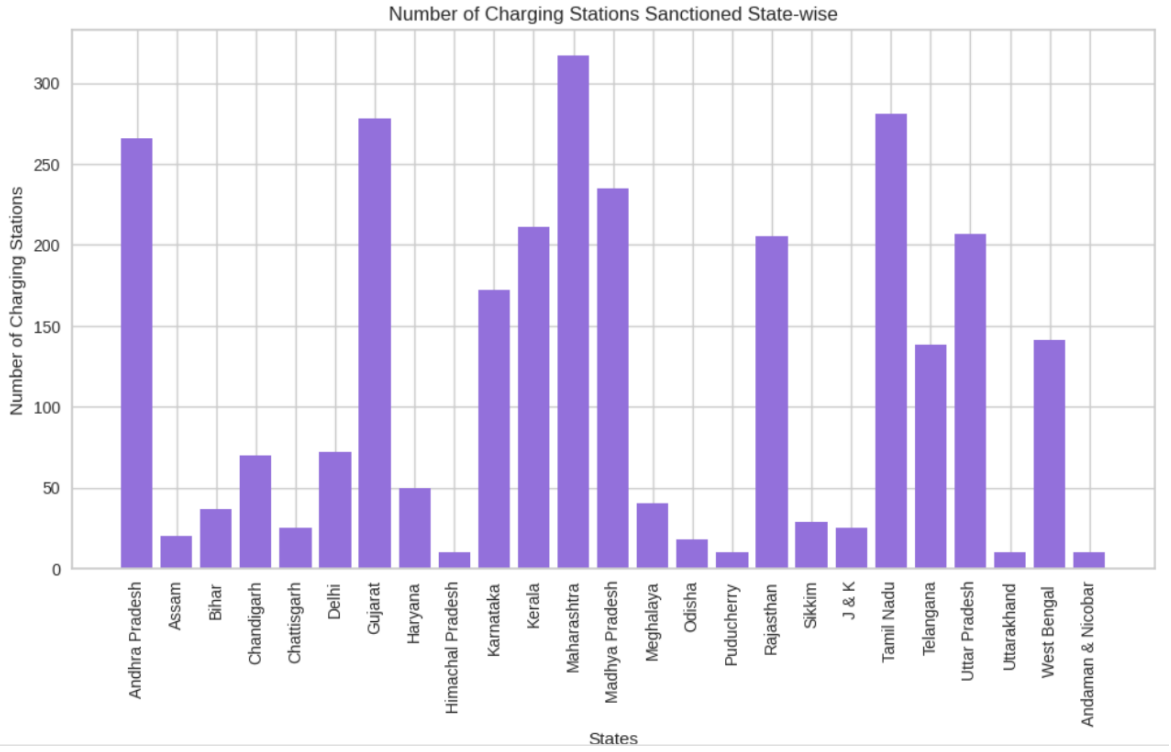
❖ State-wise Charging Station Sanctioned (Dataset 2)



The following graph shows the states with the greatest number of authorized charging stations:

1. Maharashtra
2. Tamil Nadu
3. Gujarat
4. Andhra Pradesh
5. Madhya Pradesh

<u>City-wise Charging Station Sanctioned</u>



Based on the following graph we can see that the cities with the maximum number of sanctioned charging stations are:

1. Mumbai
2. Ahmedabad
3. Bengaluru
4. Kolkata

❖ Fuel Type wise vehicle registration in India (Dataset 3)

Data Loading and Preprocessing

```
[53] FuelType = pd.read_csv("/content/Fuel type Registration of Vehicles.csv")
```

```
[54] FuelType.head()
```

	Month	CNG ONLY	DIESEL	DIESEL/HYBRID	DUAL DIESEL/CNG	ELECTRIC(BOV)	ETHANOL	LPG ONLY	NOT APPLICABLE	PETROL	PETROL/CN
0	Jan-14	2103	270915	3	0	232	0	188	10278	1347016	2062
1	Feb-14	1607	219601	3	1	171	1	116	8884	1176669	1526
2	Mar-14	2026	258723	3	1	220	1	106	11115	1329273	1880
3	Apr-14	1718	222632	3	1	252	0	121	8522	1296500	1971
4	May-14	1727	237336	6	0	186	2	103	9656	1408836	2062

```
[56] print(pd.isnull(FuelType).sum())
```

```
Month                0
CNG ONLY             0
DIESEL               0
DIESEL/HYBRID        0
DUAL DIESEL/CNG      0
ELECTRIC(BOV)        0
ETHANOL              0
LPG ONLY             0
NOT APPLICABLE       0
PETROL               0
PETROL/CNG           0
PETROL/ETHANOL       0
PETROL/HYBRID        0
PETROL/LPG           0
SOLAR                0
FUEL CELL HYDROGEN   0
LNG                  0
METHANOL             0
DUAL DIESEL/LNG      0
dtype: int64
```

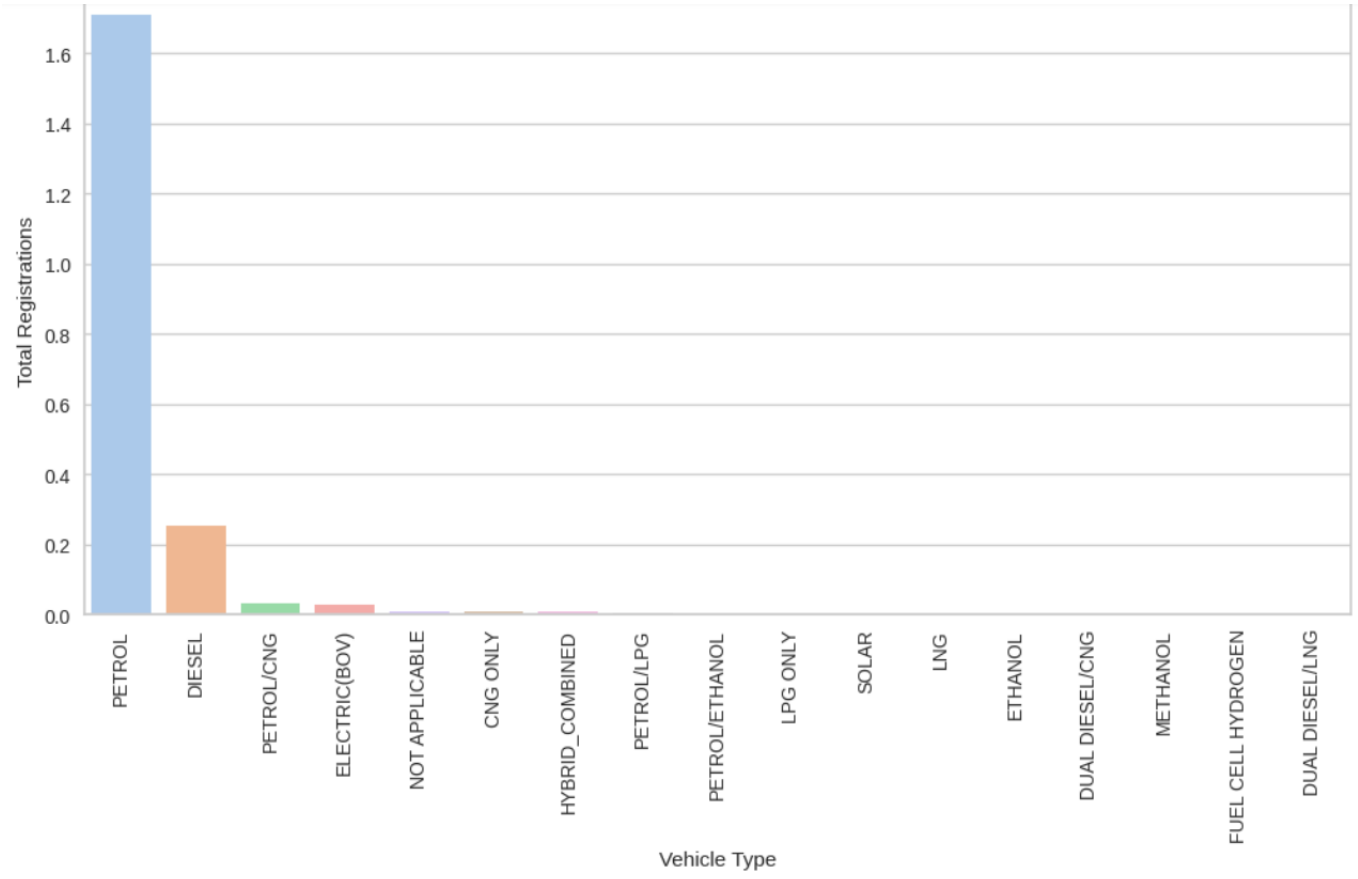
```
[60] FuelType["Month"]=pd.to_datetime(FuelType["Month"], format='%b-%y')
     FuelType["Month"].head()
```

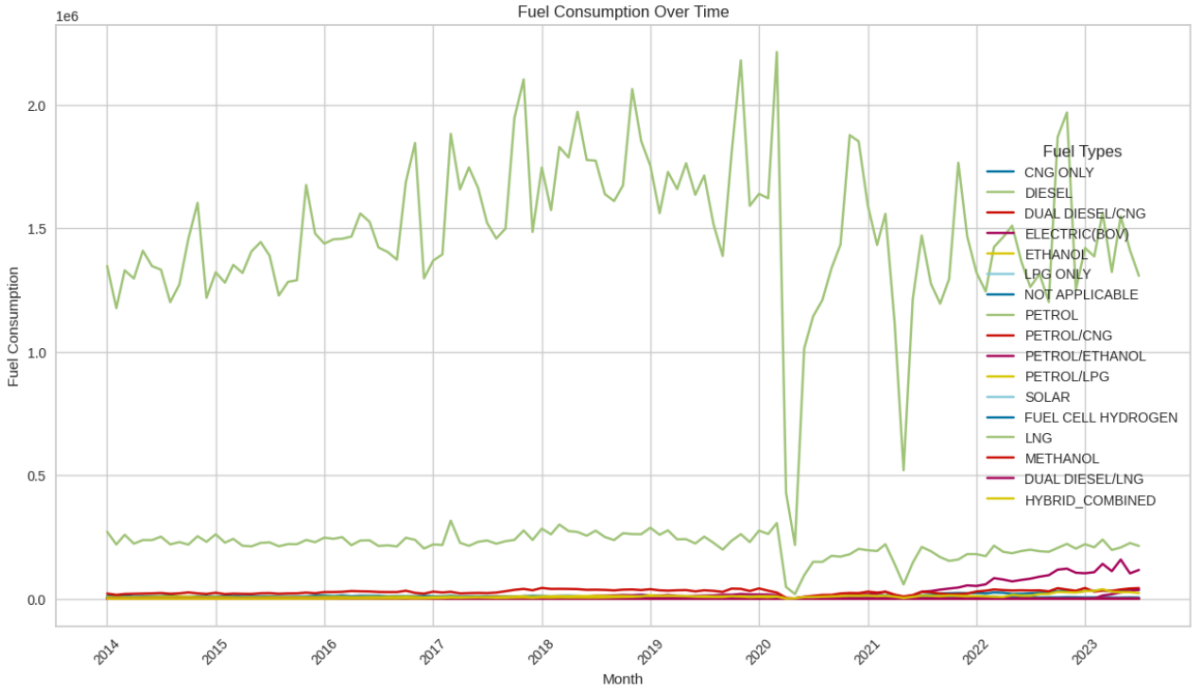
```
0   2014-01-01
1   2014-02-01
2   2014-03-01
3   2014-04-01
4   2014-05-01
Name: Month, dtype: datetime64[ns]
```

Exploring the Data

Total Vehicles registered by type:

```
[63] columnsInterest = FuelType.columns

     # Create a subset of the DataFrame with the specified columns
     selectedColumns = FuelType[columnsInterest]
     selectedColumns = selectedColumns.drop('Month', axis=1, errors='ignore')
     totalRegistration = selectedColumns.sum()

     # Sorting the columns based on total registration in descending order
     sortedColumns = totalRegistration.sort_values(ascending=False)

     plt.figure(figsize=(12, 6))
     sns.barplot(x=sortedColumns.index, y=sortedColumns.values, palette='pastel')
     plt.xticks(rotation=90)
     plt.xlabel('Vehicle Type')
     plt.ylabel('Total Registrations')
     plt.title('Total Vehicle Registrations by Type')
     plt.show()
```



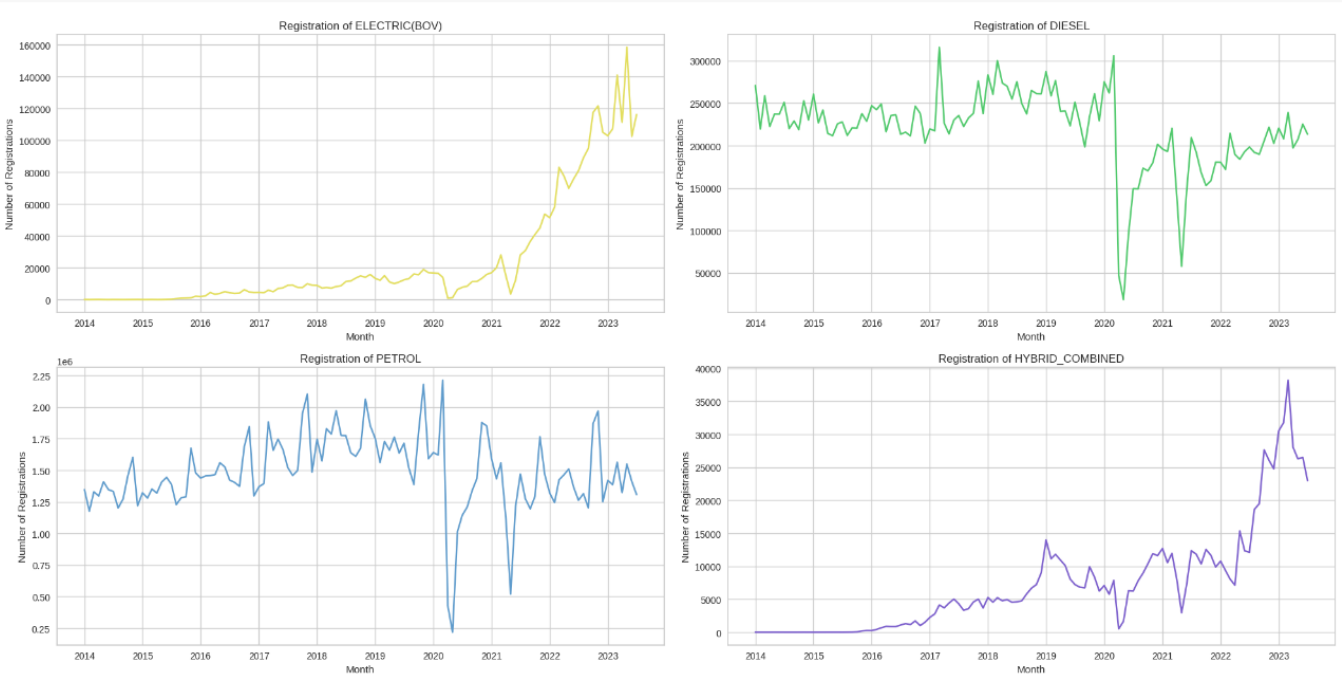


The graph shows us that:

• Registrations for gasoline vehicles have been consistently the highest over time, indicating their dominance in the market;

• Registrations for diesel vehicles have been relatively stable over time, indicating a consistent demand for diesel-powered vehicles.

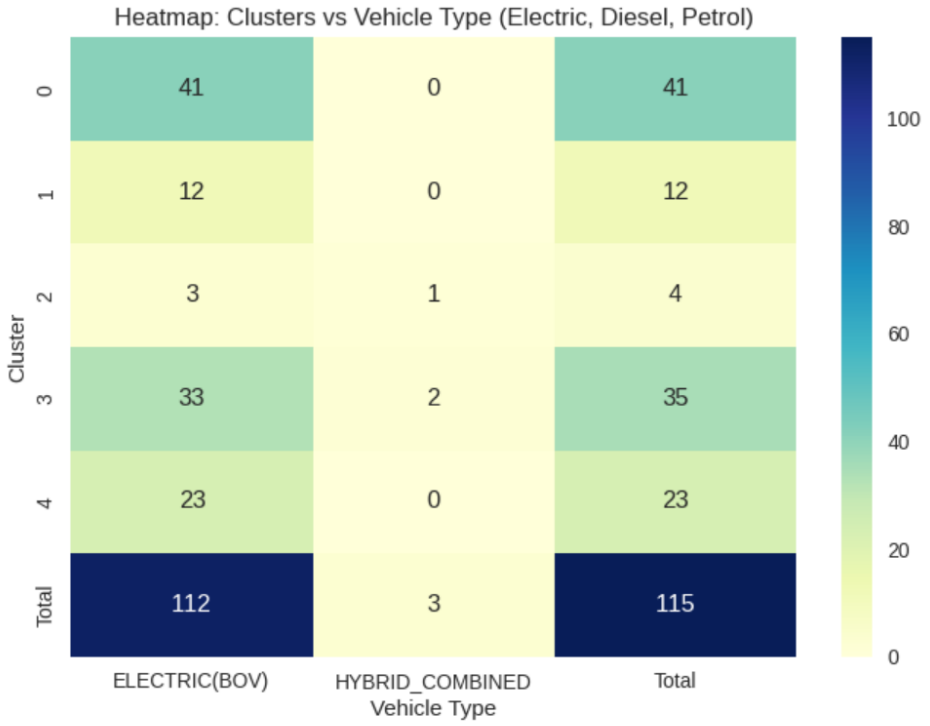
• From late 2021 onward, the number of electric car registrations shows an upward trend, suggesting a growing interest in or acceptance of electric mobility. There may be a movement in the market toward greener and more sustainable modes of transportation, as seen by the rising trend in EV registrations.



- Among the four fuel type types, the registrations of hybrid and electric vehicles exhibit a discernible upward tendency over time, which appears to point to a potential expanding market shortly.
- The steady demand for the remaining cars indicates a stable market presence, perhaps fuelled by particular industry requirements or consumer preferences.
- The increasing trend may shift consumer preferences towards more environmentally friendly and sustainable transportation options.

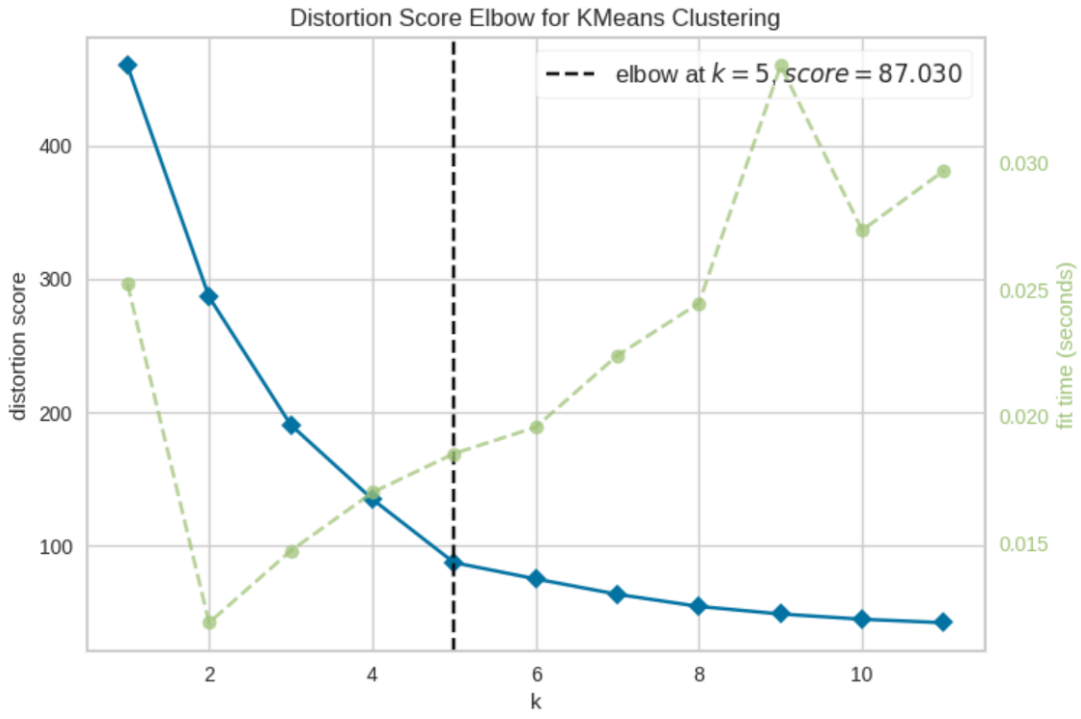
Correlation Matrix:

- "DIESEL" and "PETROL" have strong positive correlations (0.87), indicating that the values of both fuel kinds tend to move in tandem.
- There is a moderating positive connection showing some degree of co-occurrence between "HYBRID COMBINED" and "ELECTRIC(BOV)" (0.44) and "HYBRID COMBINED" and "PETROL" (0.54).
- "Year" and "ELECTRIC(BOV)" have a weakly positive association (0.23), indicating that the number of electric vehicles may rise over time.
- "ELECTRIC(BOV)" and "DIESEL" have a substantial negative correlation (-0.84), suggesting that a rise in electric cars is correlated with a fall in the use of diesel.



- Cluster 0: Compared to diesel and gasoline vehicles, the number of electric vehicles is noticeably higher in this cluster.
- Cluster 1: The distribution of vehicle types in this cluster is more balanced, with the biggest number of petrol, electric, and diesel vehicles.
- Cluster 2: The majority of the cars in this cluster are diesel, with very few electric and gasoline-powered vehicles.
- Cluster 3: There are a fair number of electric cars in this cluster, followed by diesel and gasoline cars.

<u>Elbow Method:</u>



- There is a noticeable slowdown in the distortion score reduction about k=5, indicated by a sharp elbow. This implies that the clustering quality is not significantly improved by adding clusters beyond k=5.
- The elbow technique suggests that k=5 is the ideal number of clusters for the provided data. This indicates that the data points may be efficiently grouped into five clusters using the KMeans algorithm with the least distortion.

➢ Conclusion:

The Indian mobility landscape is undergoing a remarkable shift, spurred by a growing consciousness of environmental concerns and evolving consumer preferences. Despite the prevailing dominance of petrol vehicles, there has been a remarkable upsurge in the adoption of electric vehicles (EVs). This transition is especially conspicuous in states like Maharashtra, Tamil Nadu, and Gujarat, along with major urban centres such as Mumbai, Ahmedabad, and Bengaluru. Against this backdrop of change, there emerges a highly promising opportunity for our EV startup to strategically target two key segments:

By delineating two distinct consumer segments within the Indian EV market, our startup can strategically position itself for success in this rapidly evolving landscape. Firstly, targeting Environmentally Conscious Early Adopters represented by Cluster 0, primarily located in key regions like Maharashtra, Tamil Nadu, and Gujarat, presents an opportunity to capitalize on the growing demand for sustainable transportation solutions. To effectively engage this segment, our messaging should emphasize the environmental benefits of EVs, address concerns regarding charging infrastructure availability, and highlight the innovative features of our products.

Secondly, focusing on Price-Conscious Hybrid-Open Consumers, as identified in Segment 4, offers a chance to tap into a group open to both EVs and hybrids but driven by considerations of affordability and fuel efficiency. By tailoring our marketing approach to this segment, we can underscore the competitive pricing of our offerings, highlight their flexibility in accommodating different preferences, and emphasize their superior fuel efficiency compared to traditional vehicles.

By aligning our marketing strategies with the distinct needs and preferences of these consumer segments, our startup can effectively penetrate and capitalize on the burgeoning Indian EV market. This targeted approach ensures that we not only meet the demands of diverse consumer groups but also establish a strong foothold for sustained success and growth in the electrifying landscape of the Indian mobility sector.