PROJECT REPORT ON

"ELECTRIC CAR DATA"

Submitted in partial fulfilment of the requirement for the award of Internship Bachelor's Degree of Computer Application of the year 2023-24.

By

Aishwarya Mahesh Bhandurge – (U15AQ21S006)

Vijayalaxmi Sahadev Fadake – (U15AQ21S0121)

Under the Guidance of

Prof.Abhisheka Janhavi.N.

CERTIFICATE

This is to certify that, Aishwarya Mahesh Bhandurge and Vijayalaxmi Sahadev Fadake has satisfactorily completed the project work entitled “ELECTRIC CAR DATA” for the partial fulfilment of the internship for the year 2023-2024 under the Guide Head prof. Abhisheka Janhavi.N.

ACKNOWLEDGEMENT I take this opportunity to acknowledge the contribution of each individual who has in so.

Guide

Head of the department

ACKNOWLEDGEMENT

I take this opportunity to acknowledge the contribution of each individual who has in some way or the other helped me in completing this project successfully.

I express my gratitude to our institute.

We feel it a fit occasion to open our heart to express a tender feeling and sense of deep gratitude to our guide Prof. Abhisheka Janhavi for her inspiration guidance and constant encouragement throughout project completion with valuable suggestion helped us in getting expertise in all aspects of our project.

Aishwarya Mahesh Bhandurge

Vijayalaxmi Sahadev Fadake

INTODUCTION

Data Science

Data science is the interdisciplinary field that utilizes scientific methods, processes, algorithms, and systems to extract knowledge and insights from structured and unstructured data. It employs techniques from mathematics, statistics, computer science, and domain-specific fields to uncover patterns, trends, and correlations, enabling informed decision-making and predictive analytics.

Importance of Data Science

1**. Informed Decision-Making**: Data science enables organizations to make data-driven decisions rather than relying solely on intuition or experience. By analyzing large volumes of data, businesses can identify trends, patterns, and correlations that may not be apparent through traditional methods.

2**. Competitive Advantage**: In today's highly competitive landscape, organizations that harness the power of data science gain a significant competitive edge. By leveraging data to understand customer behavior, optimize processes, and innovate products and services, companies can stay ahead of the curve.

3**. Improved Efficiency and Productivity**: Data science helps streamline operations and improve efficiency by identifying bottlenecks, optimizing resource allocation, and automating repetitive tasks. This leads to cost savings and increased productivity across various functions within an organization.

4.**Risk Management**: Data science plays a crucial role in identifying and mitigating risks across different industries, including finance, insurance, and healthcare. By analyzing historical data and predicting future outcomes, organizations can make informed decisions to minimize potential risks and losses.

Data Analysis

Data analytics is the process of analyzing raw data to uncover insights and drive decision-making. It involves techniques like statistical analysis, machine learning, and data visualization to transform data into actionable intelligence. By leveraging data analytics, organizations can optimize processes, improve efficiency, and gain a competitive edge.

Importance of data analysis

1. **Data-Driven Decision-Making**: Data analytics allows organizations to make decisions based on evidence rather than intuition. By analyzing data, businesses can gain insights into customer behavior, market trends, and operational performance, enabling them to make informed decisions that align with their strategic objectives.

2. **Improved Efficiency and Effectiveness**: Data analytics helps organizations optimize their processes and operations. By identifying inefficiencies and bottlenecks, businesses can streamline workflows, allocate resources more effectively, and improve overall efficiency.

3**. Risk Management**: Data analytics enables organizations to identify and mitigate risks effectively. By analyzing historical data and identifying patterns, businesses can anticipate potential risks and take proactive measures to mitigate them, reducing the likelihood of negative outcomes.

4. Optimized Marketing and Sales: Data analytics helps businesses optimize their marketing and sales efforts. By analyzing customer data and market trends, organizations can target the right audience with the right message at the right time, improving marketing ROI and sales effectiveness.7. Continuous Improvement: Data analytics facilitates continuous improvement by providing organizations with actionable insights. By monitoring key metrics and performance indicators, businesses can identify areas for improvement, implement changes, and measure the impact of those changes over time.

What is exploratory data analysis (EDA)?

Exploratory data analysis (EDA) is a process of analyzing and visualizing data to understand its main characteristics, patterns, and relationships. It involves examining the data from different angles, summarizing its main features, and identifying any potential outliers or missing values.

EDA helps in gaining insights into the data, formulating hypotheses, and guiding further analysis or modeling. It typically includes techniques such as data visualization, summary statistics, correlation analysis, and distribution analysis. EDA is an important step in the data analysis process as it helps in understanding the data before applying more complex statistical or machine learning techniques.

Why is exploratory data analysis important in data science?

Exploratory data analysis (EDA) is crucial in data science for several reasons. Firstly, EDA helps in understanding the underlying structure of the data, identifying patterns, trends, and relationships that may not be apparent initially. This understanding is essential for making informed decisions and formulating hypotheses before diving into more advanced analysis techniques. Secondly, EDA helps in detecting outliers, missing values, and other data quality issues that can significantly impact the results of any subsequent analysis or model.

By addressing these issues early on, data scientists can ensure the reliability and accuracy of their findings. Additionally, EDA can provide valuable insights into the data distribution, variability, and potential biases, which are essential for selecting appropriate model techniques and interpreting the results effectively. Overall, EDA plays a critical role in the data science process by guiding subsequent analysis, ensuring data quality, and uncovering valuable insights that can drive business decisions and inform further research.

Types of exploratory data analysis

There are four primary types of EDA:

1.**Univariate Non-graphical**: this is the simplest form of data analysis as during this we use just one variable to research the info. The standard goal of univariate non-graphical EDA is to know the underlying sample distribution/ data and make observations about the population. Outlier detection is additionally part of the analysis.

2.**Univariate graphical**: Non-graphical methods are quantitative and objective, they are not able to give the complete picture of the data; therefore, graphical methods are used more as they involve a degree of subjective analysis, also are required. Common sorts of univariate graphics are:

- **Histogram**: The foremost basic graph is a histogram, which may be a barplot during which each bar represents the frequency (count) or proportion (count/total count) of cases for a variety of values
- **Stem-and-leaf plots**: An easy substitute for a histogram may be stem-and-leaf plots. It shows all data values and therefore the shape of the distribution
- **Boxplots**: Boxplots are excellent at presenting information about central tendency and show robust measures of location and spread also as providing information about symmetry and outliers, although they will be misleading about aspects like multimodality.

3. **Multivariate Non-graphical**: Multivariate non-graphical EDA technique is usually used to show the connection between two or more variables within the sort of either cross-tabulation or statistics.

4.**Multivariate graphical**: Multivariate graphical data uses graphics to display relationships between two or more sets of knowledge. The sole one used commonly may be a grouped barplot with each group representing one level of 1 of the variables and every bar within a gaggle representing the amount of the opposite variable.

Exploratory Data Analysis in Python

Data Pre-processing

Data pre-processing is a crucial step in the data analysis pipeline that involves cleaning, transforming, and preparing raw data for analysis. This process aims to ensure the data is accurate, consistent, and suitable for modeling or visualization.

Data pre-processing tasks typically include handling missing values, removing duplicates, dealing with outliers, encoding categorical variables, scaling numerical features, and performing feature engineering. By addressing these issues, data pre-processing enhances the quality and reliability of the dataset, leading to more accurate insights and better model performance. Additionally, pre-processing plays a vital role in improving the efficiency and effectiveness of machine learning algorithms by providing them with clean and structured data to learn from. Overall, data pre-processing is essential for maximizing the value and utility of the data in any analytical or modeling project.

EDA using Python

Data Collection

Data collection is like gathering ingredients for a recipe. You need the right information to answer your questions or understand a situation. This information can be anything from numbers to text, and you can either collect it yourself (like conducting a survey) or use existing data from others. The important thing is to be clear about what you're looking for and how you'll get it, so you end up with high-quality information that helps you make the best decisions.

The data depicted below represents the used cars that is available on Kaggle. It contains information on the car brand, location, year, km driven, fuel type, transmission, Owner_type, mileage Engine, Power, seats, new price, price.

Step 1: Import Python Libraries

Import all libraries which are required for our analysis, such as Data Loading, Statistical analysis, Visualizations, Data Transformations, Merge and Joins, etc.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Step 2: Reading Dataset

The Pandas library offers a wide range of possibilities for loading data into the pandas DataFrame from files like JSON, .csv, .xlsx, .sql, pickle, .html, .txt, images etc.

Most of the data are available in a tabular format of CSV files. It is trendy and easy to access. Using the **read_csv()** function, data can be converted to a pandas DataFrame.

In this article, the data to predict **Used car price** is being used as an example. In this dataset, we are trying to analysis the used car's price and how EDA focuses on identifying the factors influencing the car price. We have stored the data in the DataFrame **data.**

```
elc_car=pd.read_csv("D:\intership project
fn\ElectricCarData_Clean (1).csv")
```

Step 3. Data Cleaning

Data Cleaning refers to the process of removing unwanted variables and values from your dataset and getting rid of any irregularities in it. Such anomalies can disproportionately skew the data and hence adversely affect the results. Some steps that can be done to clean data are:

- Removing missing values, outliers, and unnecessary rows/ columns.
- Re-indexing and reformatting our data.

- Before we make any inferences, we listen to our data by examining all variables in the data.
- The main goal of data understanding is to gain general insights about the data, which covers the number of rows and columns, values in the data, data types, and Missing values in the dataset.
- **shape** – **shape** will display the number of observations(rows) and features(columns) in the dataset

  There are 103 observations and 14 variables in our dataset

  head() will display the top 5 observations of the dataset

```
head() will display the top 5 observations of the dataset
```

```
[70]: elc_car.head()
```

[70]:

	Brand	Model	AccelSec	TopSpeed_KmH	Range_Km	Efficiency_WhKm	FastCharge_KmH	RapidCharge	PowerTrain	PlugType	BodyStyle	Segment	Seats	Price
0	Tesla	Model 3 Long Range Dual Motor	4.6	233.0	450.0	161.0	940	Yes	AWD	Type 2 CCS	Sedan	D	5	5
1	Volkswagen	ID.3 Pure	10.0	160.0	270.0	167.0	250	Yes	RWD	Type 2 CCS	Hatchback	C	5	3
2	Polestar	2	4.7	210.0	400.0	181.0	620	Yes	AWD	Type 2 CCS	Liftback	D	5	5
3	BMW	iX3	6.8	180.0	360.0	206.0	560	Yes	RWD	Type 2 CCS	SUV	D	5	6
4	Honda	e	9.5	145.0	170.0	168.0	190	Yes	RWD	Type 2 CCS	Hatchback	B	4	3

```
[ ]: tail() will display the top 5 observations of the dataset
```

```
[71]: elc_car.tail()
```

[71]:

	Brand	Model	AccelSec	TopSpeed_KmH	Range_Km	Efficiency_WhKm	FastCharge_KmH	RapidCharge	PowerTrain	PlugType	BodyStyle	Segment	Seats	Pr
98	Nissan	Ariya 63kWh	7.5	160.0	330.0	191.0	440	Yes	FWD	Type 2 CCS	Hatchback	C	5	
99	Audi	e-tron S Sportback 55 quattro	4.5	210.0	335.0	258.0	540	Yes	AWD	Type 2 CCS	SUV	E	5	
100	Nissan	Ariya e-4ORCE 63kWh	5.9	200.0	325.0	194.0	440	Yes	AWD	Type 2 CCS	Hatchback	C	5	
101	Nissan	Ariya e-4ORCE 87kWh Performance	5.1	200.0	375.0	232.0	450	Yes	AWD	Type 2 CCS	Hatchback	C	5	
102	Byton	M-Byte 95 kWh 2WD	7.5	190.0	400.0	238.0	480	Yes	AWD	Type 2 CCS	SUV	E	5	

info() helps to understand the data type and information about data, including the number of records in each column, data having null or not null, Data type, the memory usage of the dataset

```
[74]: elc_car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103 entries, 0 to 102
Data columns (total 14 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     101 non-null    float64
 4   Range_Km         102 non-null    float64
 5   Efficiency_WhKm  102 non-null    float64
 6   FastCharge_KmH   102 non-null    object
 7   PowerTrain       102 non-null    object
 8   PlugType         103 non-null    object
 9   BodyStyle        103 non-null    object
 10  Segment          103 non-null    object
 11  Seats            103 non-null    int64
 12  PriceEuro        103 non-null    int64
 13  PriceIND         103 non-null    float64
dtypes: float64(5), int64(2), object(7)
memory usage: 11.4+ KB
```

```
[75]: mean=elc_car["PriceIND"].mean()
      mean
```

```
[75]: 5027505.604660193
```

elc_car.info()shows the variables TopSpeed_KmH, Range_Km, Efficiency_WhKm ,FasterCharge_Kmh,PowerTrain have missing values. Numeric variables like TopSpeed_KmH, Range_Km, Efficiency_WhKm are of datatype as float64 and objective Categorical variables like FasterCharge_Kmh,PowerTrain Type are of object data type

Missing Values Calculation

Isnull() is widely been in all pre-processing steps to identify null values in the dataIn our example, **data.isnull().sum()** is used to get the number of missing records in each column

```
elc_car.isnull().sum()
```

```
memory usage: 11.4+ KB

[59]: #Finding out the number of null values

[70]: elc_car.isnull().sum()

[70]: Brand              0
      Model              0
      AccelSec           0
      TopSpeed_KmH       2
      Range_Km           1
      Efficiency_WhKm    1
      FastCharge_KmH     1
      PowerTrain         1
      PlugType           0
      BodyStyle          0
      Segment            0
      Seats              0
      PriceEuro          0
      PriceIND           0
      dtype: int64
```

Some columns or variables can be dropped if they do not add value to our analysis. In our dataset, the column rapid charge have only character values, assuming they don't have any predictive power to predict the dependent variable.

#removing Rapid charge column from elc_data

```
[69]: #drop
print("delete the specific column")
elc_car=elc_car.drop(['RapidCharge'],axis=1)
elc_car.info()
```

```
delete the specific column
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103 entries, 0 to 102
Data columns (total 14 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      101 non-null    float64
 4   Range_Km          102 non-null    float64
 5   Efficiency_WhKm   102 non-null    float64
 6   FastCharge_KmH    102 non-null    object
 7   PowerTrain        102 non-null    object
 8   PlugType          103 non-null    object
 9   BodyStyle         103 non-null    object
 10  Segment           103 non-null    object
 11  Seats             103 non-null    int64
 12  PriceEuro         103 non-null    int64
 13  PriceIND          103 non-null    float64
dtypes: float64(5), int64(2), object(7)
memory usage: 11.4+ KB
```

Duplicated values

```
[6]: elc_car.duplicated()
```

```
[6]: 0      False
     1      False
     2      False
     3      False
     4      False
            ...
     98     False
     99     False
     100    False
     101    False
     102    False
     Length: 103, dtype: bool
```

Step 4: EDA Exploratory Data Analysis

Exploratory Data Analysis refers to the crucial process of performing initial investigations on data to discover patterns to check assumptions with the help of

summary statistics and graphical representations.

- EDA can be leveraged to check for outliers, patterns, and trends in the given data.
- EDA helps to find meaningful patterns in data.

- EDA provides in-depth insights into the data sets to solve our business problems.
- EDA gives a clue to impute missing values in the dataset

Step 5: Statistics Summary

The information gives a quick and simple description of the data.

Can include Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation, etc.Statistics summary gives a high-level idea to identify whether the data has any outliers, data entry error, distribution of data such as the data is normally distributed or left/right skewed

In python, this can be achieved using describe()describe() function gives all

statistics summary of data

describe()– Provide a statistics summary of data belonging to numerical datatype such as int, float

elc_car.describe().

```
[72]: #summary
      elc_car.describe()
```

[72]:

	AccelSec	TopSpeed_KmH	Range_Km	Efficiency_WhKm	Seats	PriceEuro	PriceIND
count	103.000000	101.000000	102.000000	102.000000	103.000000	103.000000	1.030000e+02
mean	7.396117	179.702970	339.803922	189.264706	4.883495	55811.563107	5.027506e+06
std	3.017430	43.850552	126.210812	29.695460	0.795834	34134.665280	3.074851e+06
min	2.100000	123.000000	95.000000	104.000000	2.000000	20129.000000	1.813220e+06
25%	5.100000	150.000000	251.250000	168.000000	5.000000	34429.500000	3.101409e+06
50%	7.300000	165.000000	340.000000	180.500000	5.000000	45000.000000	4.053600e+06
75%	9.000000	200.000000	400.000000	204.500000	5.000000	65000.000000	5.855200e+06
max	22.400000	410.000000	970.000000	273.000000	7.000000	215000.000000	1.936720e+07

From the statistics summary, we can infer the below findings :

- Years range from 1996- 2019 and has a high in a range which shows used cars contain both latest models and old model cars.
- On average of Kilometers-driven in Used cars are ~58k KM. The range shows a huge difference between min and max as max values show 650000 KM shows the evidence of an outlier. This record can be removed.
- Min value of Mileage shows 0 cars won't be sold with 0 mileage. This sounds like a data entry issue.
- It looks like Engine and Power have outliers, and the data is right-skewed.
- The average number of seats in a car is 5. car seat is an important feature in price contribution.
- The max price of a used car is 160k which is quite weird, such a high price for used cars. There may be an outlier or data entry issue.

describe(include='all') provides a statistics summary of all data, include object, categories etc

```
elc_car.describe(include='all')
```

```
[72]: elc_car.describe(include='all')
```

[72]:

	Brand	Model	AccelSec	TopSpeed_KmH	Range_Km	Efficiency_WhKm	FastChar
count	103	103	103.000000	101.000000	102.000000	102.000000	
unique	33	102	NaN	NaN	NaN	NaN	
top	Tesla	e-Soul 64 kWh	NaN	NaN	NaN	NaN	
freq	13	2	NaN	NaN	NaN	NaN	
mean	NaN	NaN	7.396117	179.702970	339.803922	189.264706	
std	NaN	NaN	3.017430	43.850552	126.210812	29.695460	
min	NaN	NaN	2.100000	123.000000	95.000000	104.000000	
25%	NaN	NaN	5.100000	150.000000	251.250000	168.000000	
50%	NaN	NaN	7.300000	165.000000	340.000000	180.500000	
75%	NaN	NaN	9.000000	200.000000	400.000000	204.500000	
max	NaN	NaN	22.400000	410.000000	970.000000	273.000000	

Mean

Mean is a fundamental concept in data science used to summarize a set of data by finding its average value. It essentially represents the central tendency of the data.

```
#Calculate Mean
mean=elc_car["PriceIND"].mean()
mean
```

```
5027505.604660193
```

Here we see Cars data the mean of the Price in Indian Rs. Is 5,02,750.605

Median

The median is a fundamental statistic used to understand the "center" of a data set. It represents the middle value when the data is arranged in ascending or descending order.

```
#Calculate Median
mid=elc_car["TopSpeed_KmH"].median()
mid
```

```
165.0
```

Here we see Cars data the median of the Top Speed Kilomiters hours is 165

Kurtoses

Kurtosis is a statistical measure that dives into the shape of a data distribution, particularly focusing on its tails. It tells you how "peaked" or "tailed" your data is compared to a normal distribution (the bell curve).

```
#Calculate Fisher's Kurtosis
print("Fisher's Kurtosis:",kurtosis(cars[['AccelSec','Range_Km']],fisher=True))
```

```
Fisher's Kurtosis: [-1.39466693 -0.55715206]
```

Here we see Cars data the Kurtosis using Fishers kurtosis of the Acces second of cars is -1.39466693 and Range Km of car is -0.55715206

```
from scipy.stats import kurtosis
#Calculate Person Kurtosis
print("Person Kurtosis:",kurtosis(cars[['AccelSec','Range_Km']],fisher=False))
```

```
Person Kurtosis: [1.60533307 2.44284794]
```

Here we see Cars data the Kurtosis using Person kurtosis of the Acces second of cars is 1.60533307and Range Km of car is 2.44284794

Skew

Skew is a concept that describes the asymmetry of a data distribution. It tells you whether your data is clustered more on one side of the mean (average) compared to the other. Imagine a bell curve, which is a symmetrical or normal distribution. Skew helps you understand how much your data deviates from that symmetrical ideal.

```
#calculate skew
from scipy.stats import skew
skew(elc_car[['AccelSec','PriceIND']],axis=0)
```

```
array([1.22709988, 2.18944118])
```

Here we see Cars data the skew of the Access Second is 1.22709988
And Price in Indian Rs. Is 2.18944118

Step 6: EDA Univariate Analysis

Analysing /visualizing the dataset by taking one variable at a time:

Data visualization is essential; we must decide what charts to plot to better

understand the data. In this article, we visualize our data using Matplotlib and Seaborn libraries.

Matplotlib is a Python 2D plotting library used to draw basic charts we use Matplotlib.

Seaborn is also a python library built on top of Matplotlib that uses short lines of code to create and style statistical plots from Pandas and Numpy

Univariate analysis can be done for both Categorical and Numerical variables.

Categorical variables can be visualized using a Count plot, Bar Chart, Pie Plot, etc.

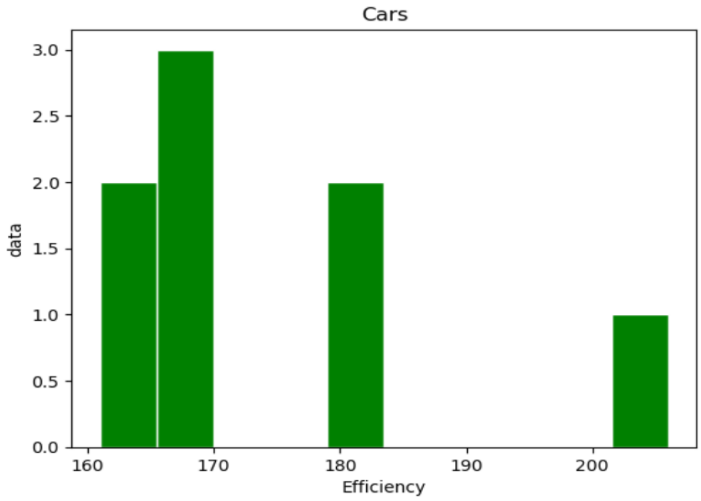
Numerical Variables can be visualized using Histogram, Box Plot, Density Plot, etc.

In our example, we have done a Univariate analysis using Histogram and Box Plot for continuous Variables.

In the below fig , histogram used to show the pattern of the variables

categorical variables are being visualized using a count plot. Categorical variables provide the pattern of factors influencing car price

```
b=cars['Efficiency_WhKm']
plt.hist(b, color='g',edgecolor='white')
plt.xlabel('Speed')
plt.ylabel('data')
plt.title('Cars')
plt.show()
```



Observation [Histogram Efficiency]

- The graph shows the distribution of fuel efficiency of cars.
- The x-axis represents the fuel efficiency of the cars and the y-axis represents the number of cars with that fuel efficiency.
- The graph shows that most cars have a fuel efficiency of around 170, with a smaller number of cars having fuel efficiency of 160, 180 and 200

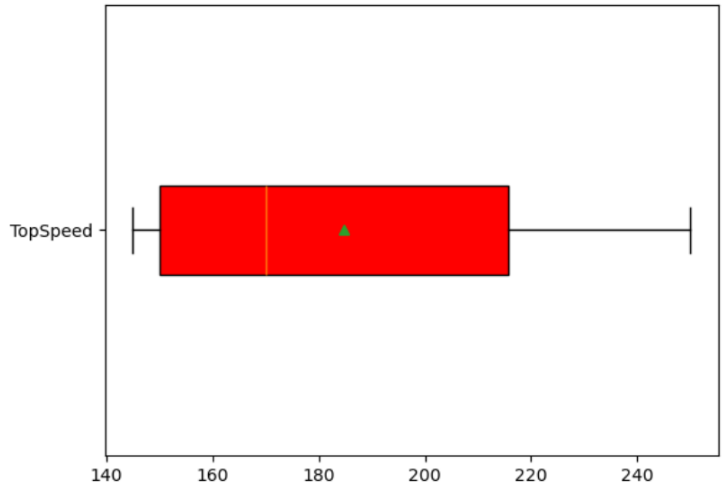
Conclusion

The histogram shows the distribution of fuel efficiency for a sample of cars. There are three distinct groups of cars with fuel efficiency around 165, 170 and 205. The most common fuel efficiency is around 170. There is no car with fuel efficiency between 175 and 200.

Box Plot

A box plot, also known as a box-and-whisker plot, is a graphical representation of the distribution of a dataset. It provides a visual summary of key statistical measures such as the median, quartiles, and potential outliers within the data.

```
[12]: x=cars['TopSpeed_KmH']
      plt.boxplot(x, widths=0.2,labels=['TopSpeed'] ,showmeans= True, patch_artist= True,boxprops=dict(facecolor="red"),vert=False)
      plt.show()
```



Observation [Box plot]

This box plot defines the observations of the Top speed Kmh here, here media n speed is 184.750 ,Maximum speed is 250.00 , min speed is 184.75, lower quartile is 150.00, upper quartile is 215.7500

The boxplot shows that the distribution of TopSpeed_KmH is fairly symmetrical . The data is centered around a value of 200 KmH and the vast majority of cars have a top speed between 150 KmH and 250 KmH.

Conclusion:

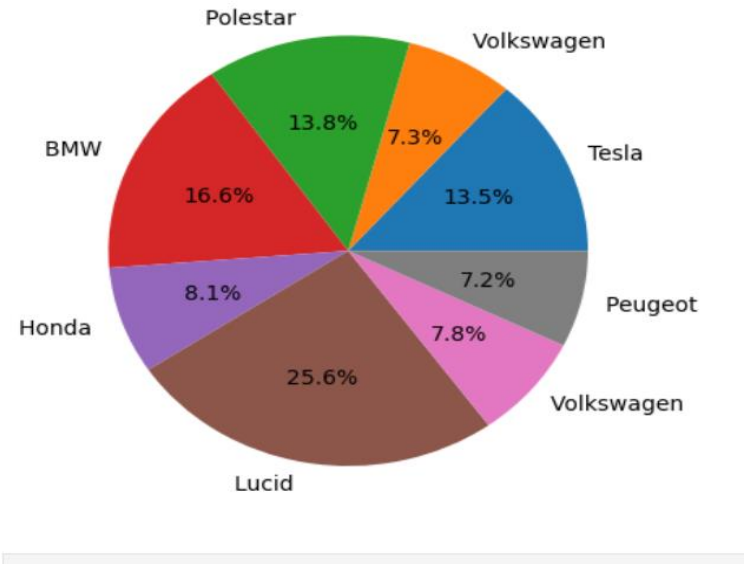
- Most cars have a moderate fuel efficiency: The graph shows that the maj ority of cars have a fuel efficiency between 1.5 and 2.5.
- Few cars have high fuel efficiency: Only a small number of cars have a fu el efficiency above 2.5, with a maximum of 3.0.
- Some cars have low fuel efficiency: A small number of cars have a fuel ef ficiency below 1.5, with a minimum of 0.0.

- The average fuel efficiency is around 2.0: The graph suggests that the av erage fuel efficiency of the cars is around 2.0, which is a moderate value.
- There is a wide range of fuel efficiencies: The graph shows that there is a wide range of fuel efficiencies among the cars, from 0.0 to 3.0.

Step 7: MatPlotlib

Pie chart

```
b=cars['Brand']
p=cars['PriceIND']
plt.pie(p,labels=b,autopct='%0.1f%%')
plt.show()
```



Observations [Pie chart]

- The graph shows that the prices of cars range from around 160 to 200, with most cars clustering between 170 and 190.

Conclusion:

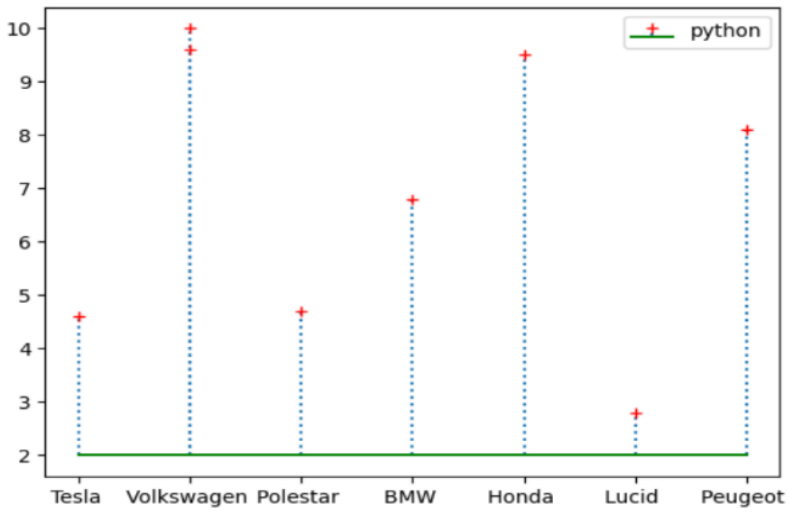
Cars have varying prices and efficiency ratings: The graph shows that cars have a wide range of prices and efficiency ratings, indicating that there are many options available in the market.

Cars can be categorized into groups: The groupings of cars with similar prices and efficiency ratings suggest that cars can be categorized into different segments or groups based on these factors.

This Pie chart defines the observations of the Price Indian Rs.here, here height price of car is Lucida 25.6%, and lowest price of car is Peugeot 7.2%.

Stem plot

```
x=cars['Brand']
y=cars['AccelSec']
plt.stem(x,y, linefmt=":", markerfmt='r+', bottom=2, basefmt="g",label ="python")
plt.legend()
plt.show()
```



Observations [Stem plot]

- The graph shows the relationship between the efficiency rating of cars (x-axis) and the access time in seconds (y-axis).

- Efficiency ratings: The efficiency ratings range from 0.0 to 3.0, with most cars having ratings between 1.0 and 2.5.
- Access time: The access time in seconds ranges from around 160 to 200, with most cars having access times between 170 and 190.

Conclusion:

Cars with higher efficiency ratings do not necessarily have faster access times: The graph shows that cars with higher efficiency ratings do not always have faster access times. In fact, some cars with lower efficiency ratings have similar or even faster access times.

Access time is not a direct indicator of a car's efficiency: The lack of correlation between efficiency rating and access time suggests that access time is not a reliable indicator of a car's overall efficiency.

Step 9: EDA Bivariate Analysis

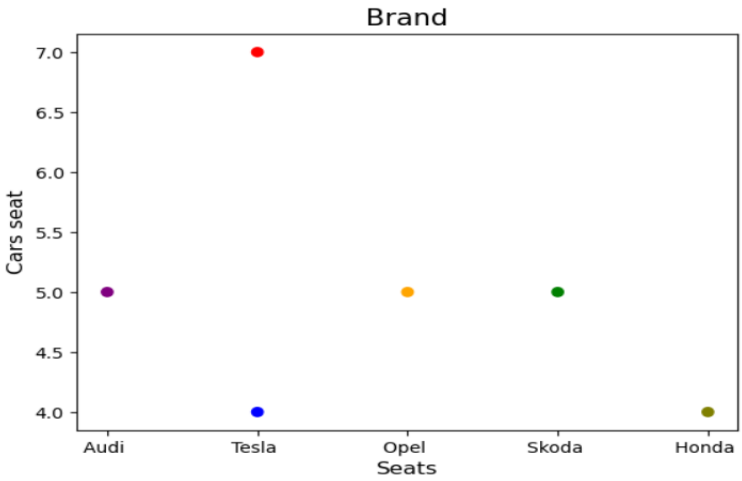
Bivariate Analysis helps to understand how variables are related to each other and the relationship between dependent and independent variables present in the dataset.

For Numerical variables, Pair plots and Scatter plots are widely been used to do Bivariate Analysis.

A Stacked bar chart can be used for categorical variables if the output variable is a classifier. Bar plots can be used if the output variable is continuous

In our example, a Scatter plot has been used to show the relationship between two Numerical variables.

```
x=c['Brand']
y=c['Seats']
colors=["purple",'b','orange','g','r','olive']
plt.scatter(x,y, c=colors)
plt.title("Brand",fontsize=15)
plt.xlabel("Seats", fontsize=12)
plt.ylabel("Cars seat",fontsize=12)
plt.show()
```



Observations [Scatter Plot]

The graph shows the number of seats in cars from different brands. The x-axis lists the brands, and the y-axis shows the number of seats.

Audi has 5 seats.

Tesla has 4 seats.

Opel has 5 seats.

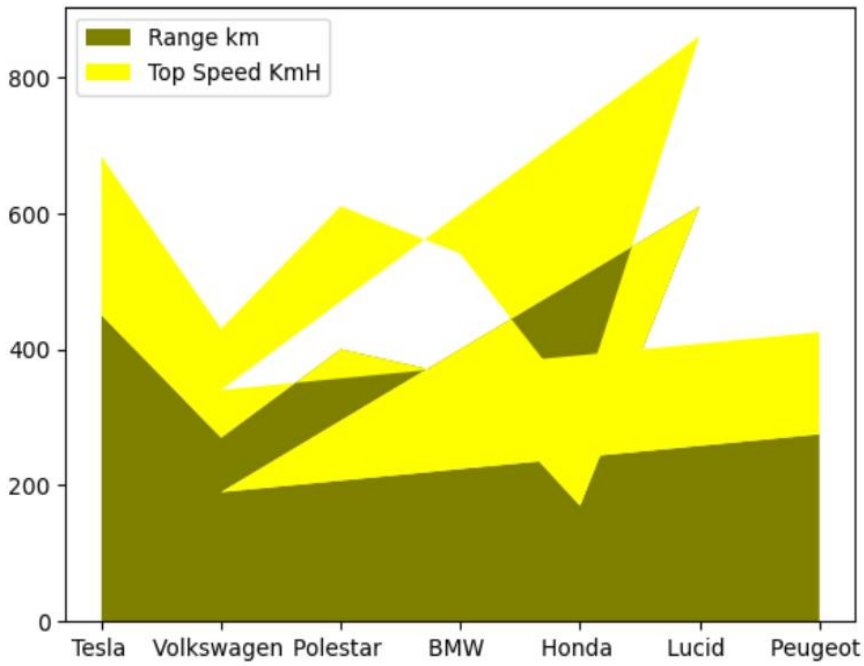
Skoda has 5 seats.

Honda has 4 seats.

Conclusion:

Therefore, we can conclude that most of the cars in this graph have 5 seats, except for Tesla and Honda, which have 4 seats. This information could be useful for people who are looking for cars with a specific number of seats

```
x=cars['Brand']
area=cars['Range_Km']
area1=cars['TopSpeed_KmH']
label=['Range km','Top Speed KmH']
plt.stackplot(x,area,area1,labels=label,colors=['olive','yellow'])
plt.legend(loc=2)
plt.show()
```



Observations [Stack Plot]

- Tesla: Has the highest range but a slightly lower top speed compared to other cars.
- Volkswagen: Has a lower range but a higher top speed compared to Tesla.
- Polestar: Has a moderate range and a moderate top speed.
- BMW: Has a lower range compared to Tesla but a higher top speed.
- Honda: Has a lower range compared to Tesla but a higher top speed.
- Lucid: Has the lowest range but the highest top speed.
- Peugeot: Has a moderate range and a moderate top speed.

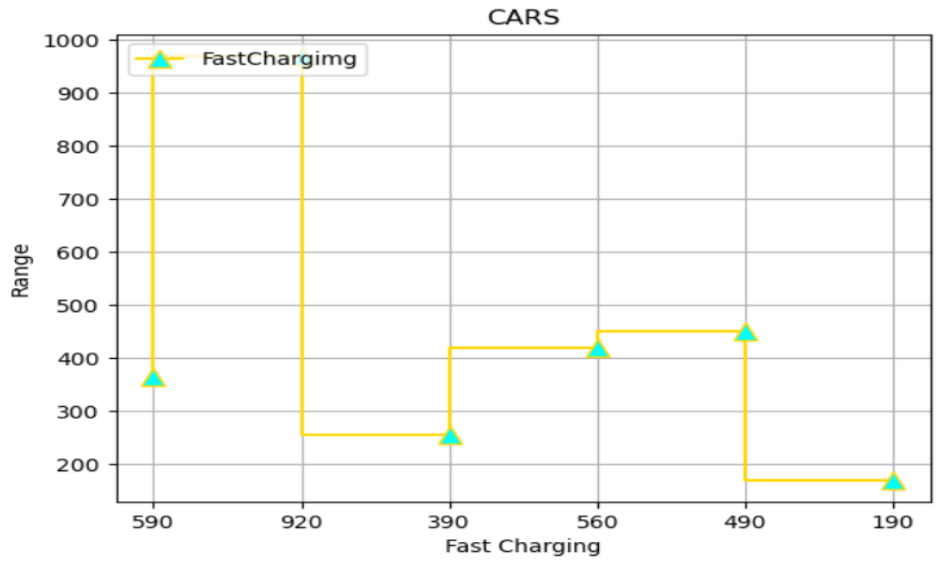
Conclusion:

Based on this graph, it can be concluded that there is no clear relationship between the range of an electric car and its top speed.

The graph showcases different cars prioritize either range or top speed depending on their targeted audience and design goals. For instance, Tesla prioritizes range, while Lucid prioritizes top speed.

Ultimately, the best choice for a particular individual depends on their individual needs and priorities.

```
y=c['Range_Km']

x=c['FastCharge_KmH']

plt.step(x,y, color='gold', marker='^',ms=10, mfc='cyan', label='FastChargimg')

plt.title("CARS")

plt.xlabel('Fast Charging')

plt.ylabel('Range')

plt.grid()

plt.legend(loc=2)
plt.show()
```



Observations [Stack Plot]

- The graph shows the range of cars based on their fastcharging capabilities.
- It can be observed that the cars with higher fast charging capability generally have a lower range.
- This could be because a higher fast charging capability may require a larger battery pack, which could lead to a decrease in range.
- Range and Fast charging of car here, here highest Fast charging is 1000 and lowest Fast charging is 150 .

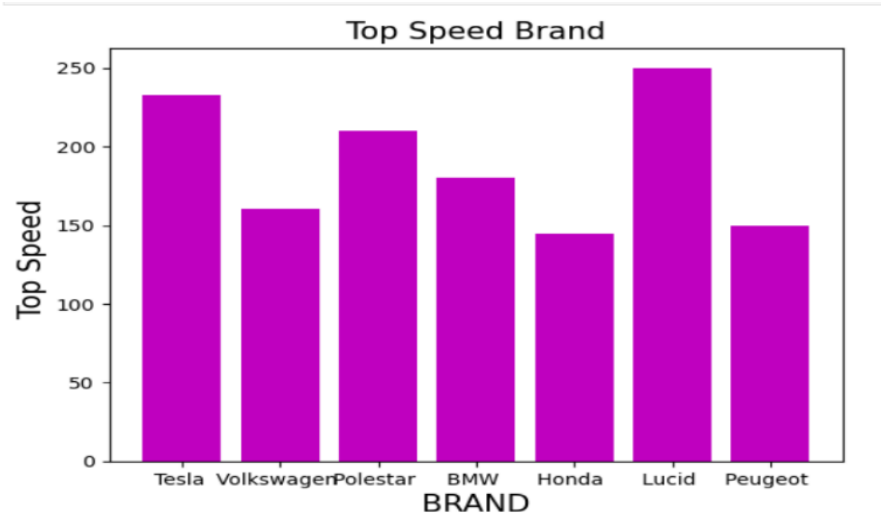
Conclusion:

the car with the highest fast charging capability of around 950 has a range of around 200. On the other hand, the car with the lowest fast charging capability of around 170 has a range of around 450.

There seems to be a trend of decreasing range as the fastcharging capability increases, although it is not a perfectly linear relationship. It is important to note that other factors may also contribute to the range of a car, such as the type of engine, the weight of the car, and the efficiency of the drivetrain.

A bar plot can be used to **show the relationship between Categorical variables and continuous variables**

```
X=cars[‘Brand]
Y=cars[‘TopSpeed_KmH’]
plt.xlabel[“BRAND”,fontsize=15]
plt.ylabel[“Top Speed”,fontsize=15]
plt.title('Top Speed Brand',fontsize=15)
plt.bar(x,y,color='m')
plt.show()
```



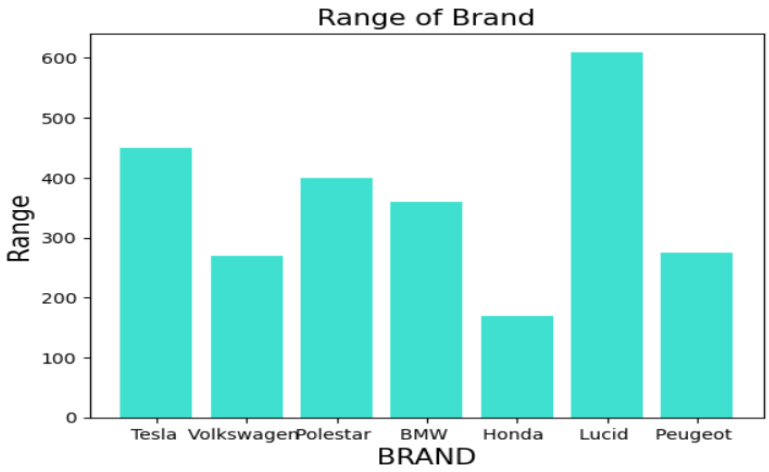
Observations [Bar plot]

- The graph shows a general trend of increasing top speed with increasing efficiency. This suggests that cars with higher efficiency tend to have higher top speeds.
- Tesla has the highest top speed, with a value of around 200 km/h, and an efficiency of around 3.0. This indicates that Tesla's cars are not only fast but also highly efficient.
- Lucid, Polestar, and BMW have similar top speeds, ranging from 180 to 190 km/h, and efficiencies ranging from 2.0 to 2.5. These cars are also fast and efficient, but not as much as Tesla.
- Honda and Volkswagen have lower top speeds, ranging from 160 to 170 km/h, and efficiencies ranging from 1.5 to 2.0. These cars are slower and less efficient than the previous group.
- Peugeot has the lowest top speed, with a value of around 150 km/h, and an efficiency of around 1.0. This indicates that Peugeot's cars are the slowest and least efficient in the group.

Conclusion:

In conclusion, the graph shows that there is a positive correlation between the efficiency of a car and its top speed. Cars with higher efficiency tend to have higher top speeds. Tesla stands out as the leader in both efficiency and top speed, while Peugeot lags behind in both categories. The other car brands fall somewhere in between, with varying degrees of efficiency and top speed.

```
x=cars['Brand']
y=cars['Range_Km']
plt.xlabel("BRAND", fontsize=15)
plt.ylabel("Range", fontsize=15)
plt.title('Range of Brand',fontsize=15)
plt.bar(x,y,color='turquoise')
plt.show()
```



Observations [Bar plot]

- Starting from the left side of the x-axis, Lucid has the highest range of 600 Km, making it the brand with the longest-range EV. Next, Tesla and Polestar have ranges between 400-500 Km

- Moving towards the right side of the x-axis, BMW and Peugeot have ranges between 200-300 Km. These ranges are still decent.

- Finally, Honda and Volkswagen have the lowest ranges of under 200 Km. These ranges are relatively low compared to the other brands.

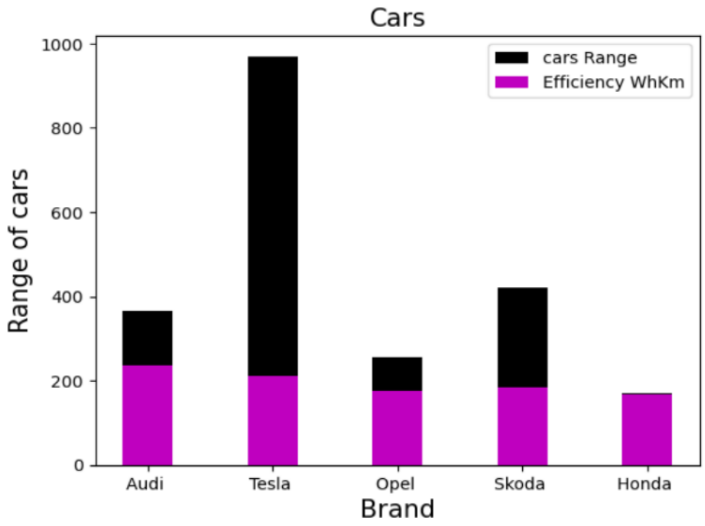
Conclusion:

The graph shows that Lucid has the highest range, followed by Tesla and Polestar. BMW and Peugeot have decent ranges, while Honda and Volkswagen have the lowest ranges.

This information can be useful for customers who are looking for an EV with a long range. However, it is important to note that the range of an EV can depend on various factors such as driving conditions, temperature, and driving style.

```
x=c['Brand']
y=c['Range_Km']
z=c['Efficiency_WhKm']
plt.xlabel('Brand', fontsize=15)
```

```
plt.ylabel("Range of cars", fontsize=15)
plt.title('Cars',fontsize=15)
plt.bar(x,y,width =0.4, color="k",  label="cars Range")
plt.bar(x,z,width =0.4, color="m",  label="Efficiency WhKm")
plt.legend()
plt.show()
```



Observations [Bar plot]

- The graph shows the range and efficiency of different car brands. The range is represented by black bars and the efficiency is represented by magenta bars.

- The brand with the highest range is Tesla, with a range of over 900 km. Audi has a range of 350 km, Opel has a range of 250 km, Skoda has a range of 400 km, and Honda has a range of 175 km.

Conclusion:

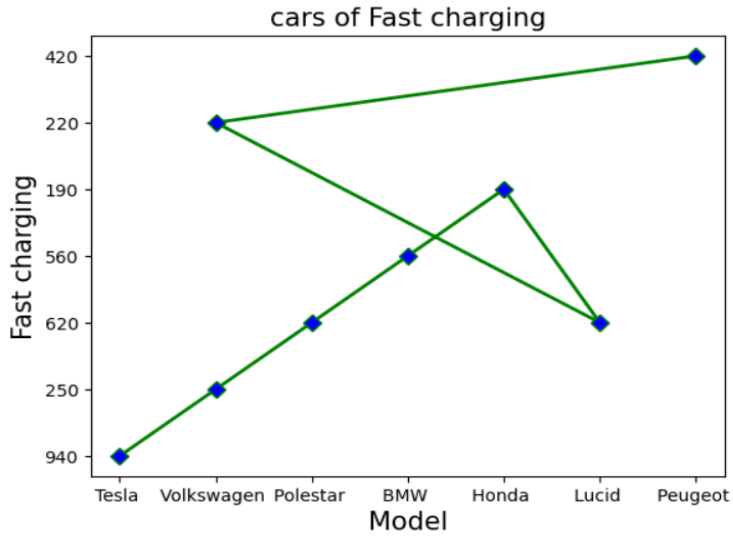
The graph shows that Tesla has the highest range of any of the car brands listed, but it also has the lowest efficiency. This suggests that Tesla cars are able to travel the furthest on a single charge, but they use more energy to do so than other brands.

Honda, on the other hand, has the lowest range, but it also has the highest efficiency, indicating that Honda cars are able to travel the furthest on a given amount of energy.

Line plot

Line plot are a useful tool for visualizing time-series data or showing the relationship between variables. To create a line chart using Matplotlib, use the plot() function, which accepts x and y coordinates.

```
x=cars['Brand']
y=cars['FastCharge_KmH']
plt.plot(x,y, color ="g", linewidth=2,marker="D",markerfacecolor="b",markersize=7)
plt.xlabel("Model", fontsize=15)
plt.ylabel("Fast charging", fontsize=15)
plt.title("cars of Fast charging",fontsize=15 )
plt.show()
```



Observations [Line plot]

- The graph shows the fast charging speed of different electric car models. The x-axis shows the car model and the y-axis shows the fast charging speed in kilometers per hour (km/h).

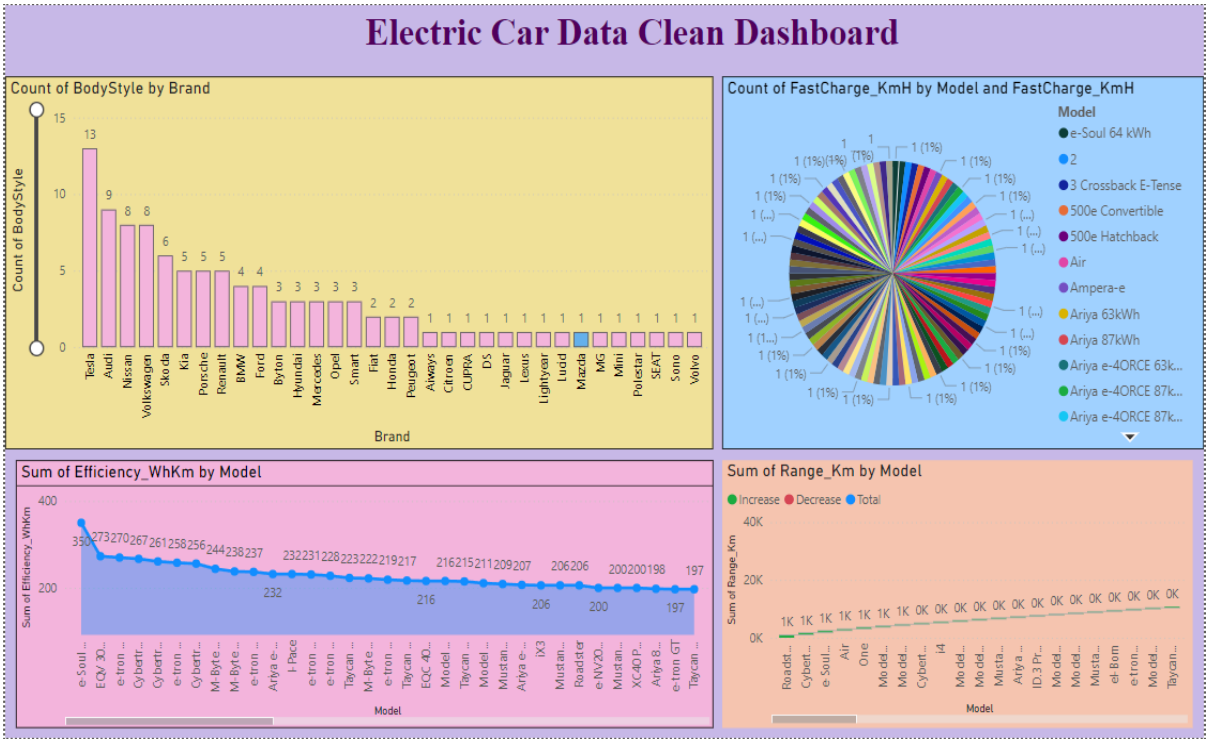
- The graph shows that the Peugeot model has the fastest charging speed, at 420 km/h. The Lucid model comes in second at 620 km/h.
- Then comes Honda at 590km/h. BMW follows at 560 km/h. The Polestar, Volkswagen, and Tesla models have slower charging speeds of 620, 250, and 940 km/h, respectively.

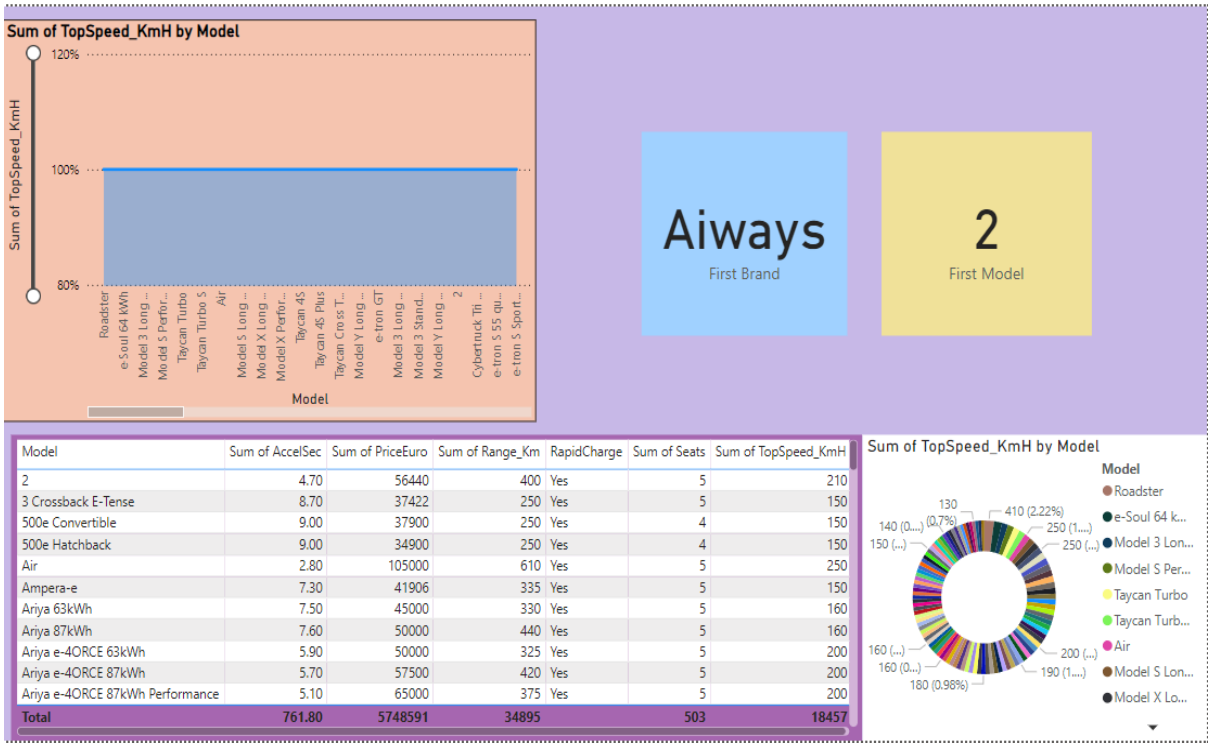
Conclusion:

Based on the data, we can conclude that Peugeot and Lucid electric cars are the fastest charging cars in the market, with Peugeot leading by a significant margin.

Other models, such as Polestar, Volkswagen, and Tesla, have relatively slow charging speeds.

POWER-BI ON Electric Cars Data Clean Dashboard





Model	Sum of AccelSec	Sum of PriceEuro	Sum of Range_Km	RapidCharge	Sum of Seats	Sum of TopSpeed_KmH
2	4.70	56440	400	Yes	5	210
3 Crossback E-Tense	8.70	37422	250	Yes	5	150
500e Convertible	9.00	37900	250	Yes	4	150
500e Hatchback	9.00	34900	250	Yes	4	150
Air	2.80	105000	610	Yes	5	250
Ampera-e	7.30	41906	335	Yes	5	150
Ariya 63kWh	7.50	45000	330	Yes	5	160
Ariya 87kWh	7.60	50000	440	Yes	5	160
Ariya e-4ORCE 63kWh	5.90	50000	325	Yes	5	200
Ariya e-4ORCE 87kWh	5.70	57500	420	Yes	5	200
Ariya e-4ORCE 87kWh Performance	5.10	65000	375	Yes	5	200
Total	**761.80**	**5748591**	**34895**		**503**	**18457**