Data Visualization (CSE3020)

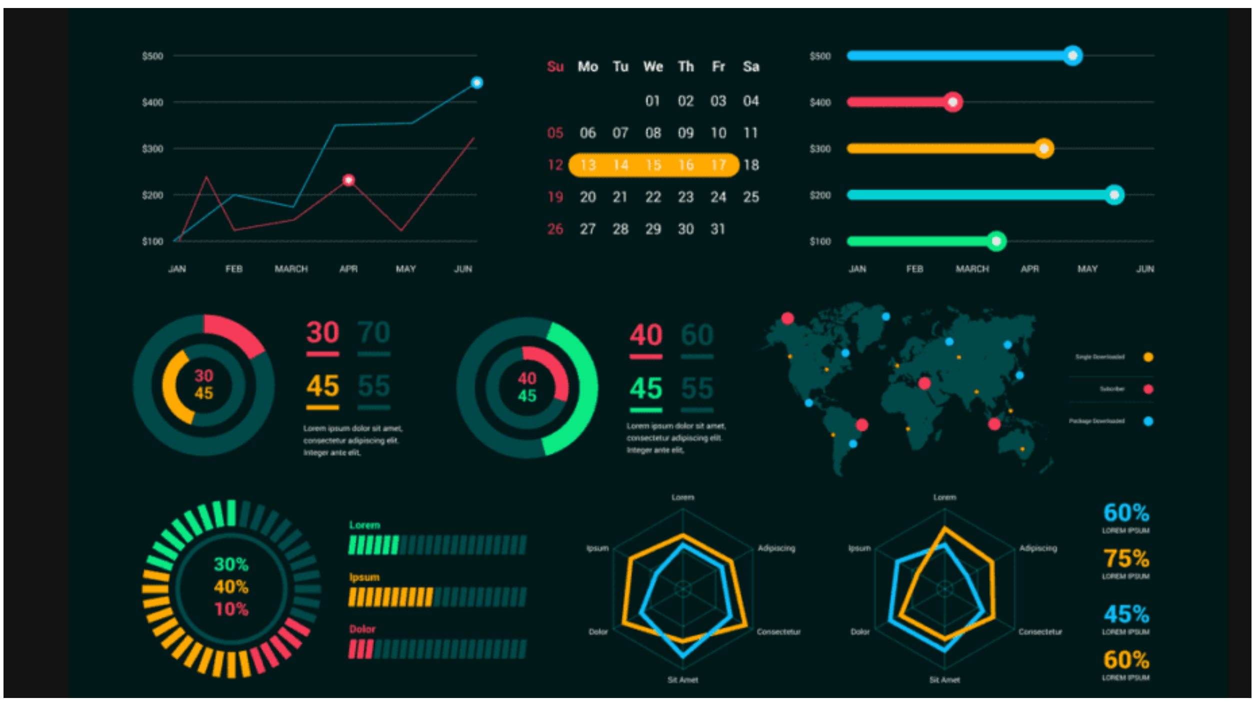
Slot: D1

Faculty: Lydia Jane G

Electric Vehicles

A thorough analysis into the world of electric vehicles

Review 2



Brought to you by

Shivam Raj, 20BCE0819

Ananya Singh, 20BCE0785

Data Abstraction :

The datatypes in the dataset are: Discrete Data, Nominal Data, Ratio Data.

Attribute definitions

Brand: The manufacturer of the vehicle, i.e. Hyundai

Model: Model of vehicle.

AccelSec: The amount of time in seconds taken for the acceleration to move from 0100km/h.

TopSpeed_KmH Top speed in km/h

Range_Km: Range in kilometers.

Efficiency_WhKm: Efficiency in watt-hour per kilometer.

FastCharge_KmH Charge km/h

RapidCharge: Is the feature available?

PowerTrain: Front, rear, or all-wheel drive

PlugType: Type of plug

BodyStyle: Basic size or style, i.e. SUV

Segment: Market segment

Seats: Number of seats

PriceEuro: Prices in Germany before tax incentives

Dataset details :

https://www.kaggle.com/tyrawls/electric-vehicle-analysis/data

https://www.kaggle.com/vijayakishoredusi/starter-evpopulation/data

Introduction :

As we know our environment is degrading day by day due to pollution and the major contribution to this is done by fuel vehicles. So to sustain our environment and to save our mother earth from depleting further, we need to do something to decrease the pollution levels.

Our Objective :

So one of the major initiatives we as youth can take is by reducing the use of fuel vehicles. People say to avoid cars and travel by either a public transport bus or use a cycle instead isn’t a permanent solution to this. We all have developed a kind of lifestyle that we can’t live without vehicles. And since COVID-19 came into existence, it is also not feasible to use public transports.

So we have analysed data in such a way that people go for electric cars instead of fuel vehicles. We have tried to cover each and every aspect that will lead the user in choosing a perfect car for himself !

General Overview :

Conventional cars use internal combustion engines (ICEs) that generally run on fossil fuels like gas or diesel. EVs use one or more electric motors powered by rechargeable lithium-ion batteries, the same kinds of batteries that power smartphones and laptops. And like electronic devices, EVs plug into external power sources for charging. Other types of batteries rely on regenerative braking for charging, or generating electricity from the vehicle's frictional energy. In addition to being less polluting than fuel engines, lithium-ion batteries often work more efficiently. Many have a guaranteed life span of 8-10 years.

Like stopping at a gas station, a network of charging stations gives EV batteries access to power on the road. An EV's driving range between stations is dependent on its battery life. Extreme driving conditions or weather can also affect an EV's range as they use more energy to compensate.

Because EVs do not rely on fossil fuels for power, they may not have certain components that ICE vehicles do. For instance, parts such as fuel lines, fuel tanks, and tailpipes. This means that most EVs do not emit carbon dioxide emissions (CO2), which helps reduce air pollution.

An electric car can be a great way for you, as a consumer, to save a lot of money on gas. However, there are so many different reasons why you should invest in an electric car in the modern-day of technology.

1. No Gas Required

2. More Convenient

3. Savings

4. No Emissions

5. Popularity

6. Safe to Drive

7. Cost-Effective

8. Low Maintenance

9. Reduced Noise Pollution

10. Battery Life & Cost

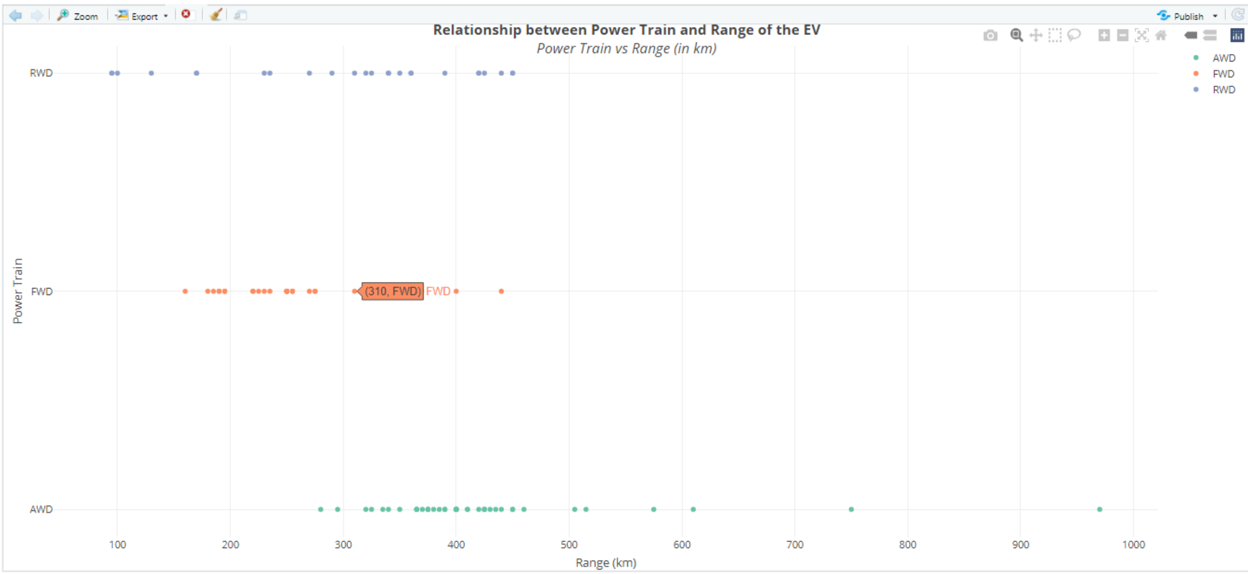
11. Easy Driving

The major benefit of electric cars is the contribution that they can make towards improving air quality in towns and cities. With no tailpipe, pure electric cars produce no carbon dioxide emissions when driving. This reduces air pollution considerably.

Put simply, electric cars give us cleaner streets making our towns and cities a better place to be for pedestrians and cyclists. In over a year, just one electric car on the roads can save an average 1.5 million grams of CO_2. That's the equivalent of four return flights from London to Barcelona.

(1) Plot an Interactive graph to show the how Power Train affects Efficiency

```
library(ggplot2)
library(plotly)

evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarData_Clea
n.csv')
pwrtrn <- evs[, c(4:6, 9)]
avg_rng <- aggregate(pwrtrn$Range_Km ~ pwrtrn$PowerTrain, FUN
= mean)

plt <- plot_ly(data=evs, x= ~Range_Km, y= ~PowerTrain,
color=~PowerTrain ,type="scatter")

plt <- plt %>%  layout(title = "<b>Relationship between Power
Train and Range of the EV</b><br><i>Power Train vs Range (in
km)</i>",

                           yaxis = list(title = "Power
Train"),
                       xaxis = list(title = "Range (km)"))

plt
```



The above graph shows the relation between Range of the cars in Km vs the Power train. This graph is an interactive scatterplot where we can hover around our cursor to see a particular entry or we can zoom into the plot or select a particular region and also take snapshots.

(2) Deduce relation between Price and the Fast Charging Technology

```
library(tidyverse)
```

```
library(ggplot2)
library(plotly)
evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarDa
ta_Clean.csv')
pwrtrn <- evs[, c(4:6, 9)]
avg_rng <- aggregate(pwrtrn$Range_Km ~
pwrtrn$PowerTrain, FUN = mean)

Rapidcharge_price <- evs[, c(8, 14)]
Rapidcharge_price <-
Rapidcharge_price[order(Rapidcharge_price$PriceEuro),
]
avg_price <- aggregate(Rapidcharge_price$PriceEuro ~
Rapidcharge_price$RapidCharge, FUN = "mean")
avg_price <- avg_price %>% rename(RapidCharge =
`Rapidcharge_price$RapidCharge`, Avg_Price =
`Rapidcharge_price$PriceEuro`)
head(avg_price)

plt <- plot_ly(data=Rapidcharge_price, x= ~PriceEuro,
y= ~RapidCharge, color=~RapidCharge ,type="scatter")
plt <- plt %>%  layout(title = '<b>Relation between
price and rapid charging</b><br><i>Rapid Charging
Availabilty vs Price (Euro)</i>', plot_bgcolor =
"#e5ecf6", xaxis = list(title = 'Price Euro'),
                       yaxis = list(title = 'Rapid
Charge'), legend = list(title=list(text='<b> Rapid
Charging </b>')))


plt<- plt %>%

  add_annotations( x = avg_price$Avg_Price[2],
                   y = 1,
```

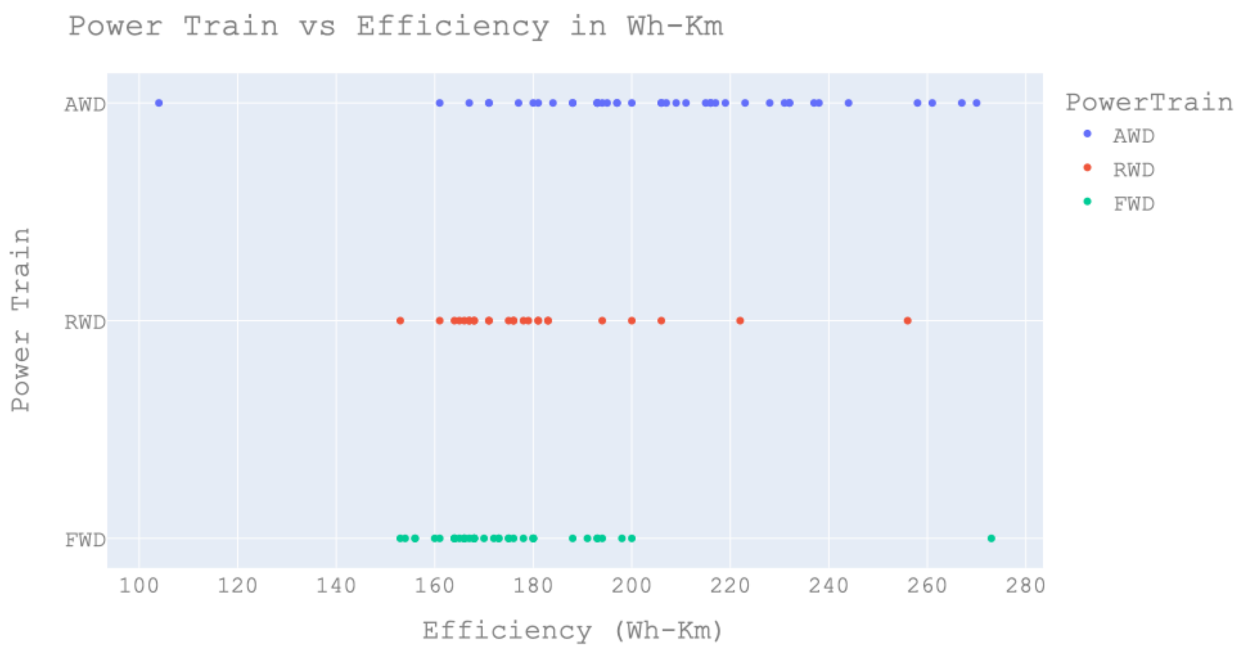
```
                        xref = "x", yref = "y",
                        axref = "x", ayref = "y",
                        text = "avg price for yes",
                        showarrow = T,
                        ax = avg_price$Avg_Price[2],
                        ay = 1.30)
plt<- plt %>%

  add_annotations( x = avg_price$Avg_Price[1],
                        y = 0,
                        xref = "x", yref = "y",
                        axref = "x", ayref = "y",
                        text = "Avg price for no",
                        showarrow = T,
                        ax = avg_price$Avg_Price[1],
                        ay = 0.30)

plt
```

This plot shows the relation between Price of the car and weather it has rapid charging or not. It also indicates the average for both the choices. This interactive plot helps user choose wisely as there is an arrow pointer which they can use to select particular price ranged cars. User can also zoom in to the graph for better clarity of number of options and also they can snip or capture the plot

(3) Can you show which power train has better efficiency

```
import warnings
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import scipy as sp
import seaborn as sns
from PIL import Image
from plotly.subplots import make_subplots
import plotly.express as px

ds = pd.read_csv("/Users/shivamraj/Desktop/VIT 3rd
Sem/DATA
VISUALIZATION/Project/ElectricCarData_Clean.csv.xls")

fig = px.scatter(ds, x = "Efficiency_WhKm", y =
"PowerTrain", color="PowerTrain")
fig.update_layout(
    title="Power Train vs Efficiency in Wh-Km",
    xaxis_title="Efficiency (Wh-Km)",
    yaxis_title="Power Train",
```

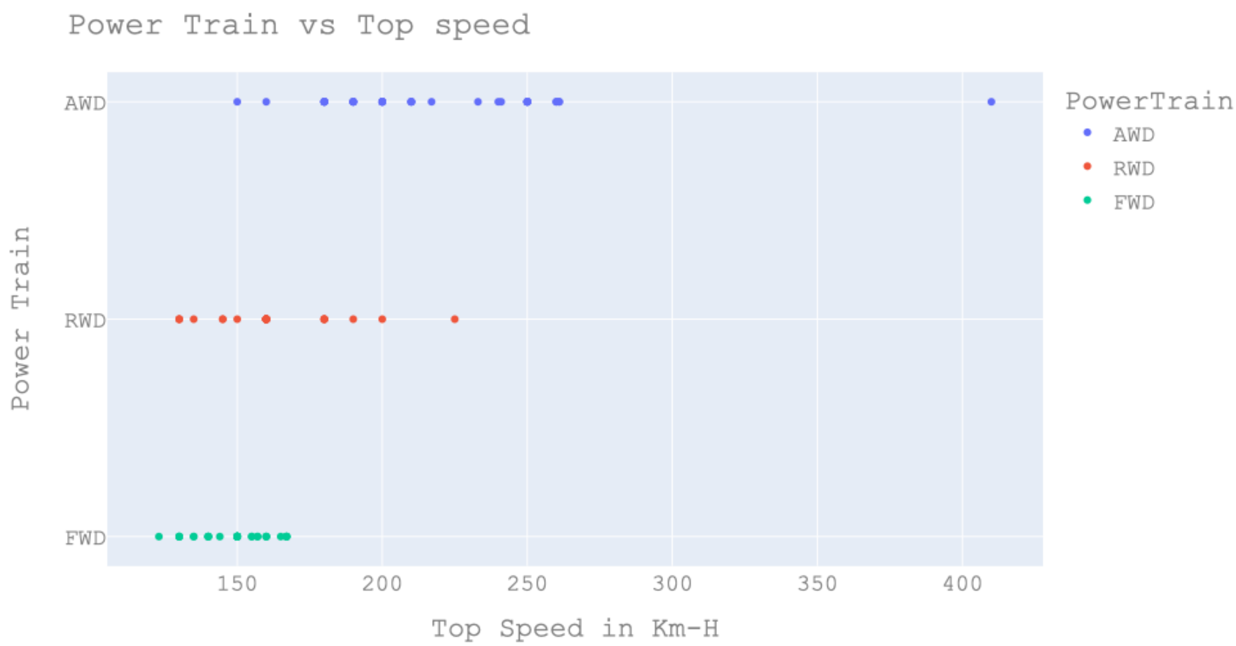
```
    font=dict(family="Courier New, monospace",
size=18,color="#7f7f7f"))
```



The above plot shows type of Power Trains that show efficiency in Wh-Km. This will help the user in choosing a car which can perform better in less power consumption. This interactive scatter plot also has different colours for different Power Trains by which user has a clear idea of what is more efficient for him.

(4) Show that how fast are Evs depending on their Power Train

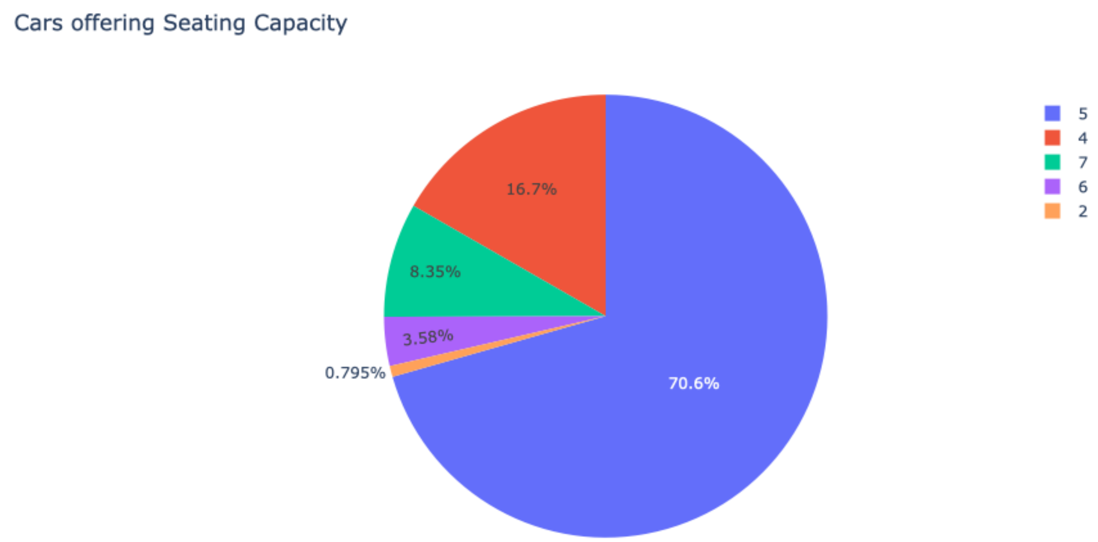
```
import warnings
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import scipy as sp
import seaborn as sns
from PIL import Image
from plotly.subplots import make_subplots
import plotly.express as px

ds = pd.read_csv("/Users/shivamraj/Desktop/VIT 3rd
Sem/DATA
VISUALIZATION/Project/ElectricCarData_Clean.csv.xls")

plt = px.scatter(ds, x = "TopSpeed_KmH", y =
"PowerTrain", color="PowerTrain")
plt.update_layout(
    title="Power Train vs Top speed",
    xaxis_title="Top Speed in Km-H",
    yaxis_title="Power Train",
    font=dict(family="Courier New, monospace",
size=18,color="#7f7f7f"))
```



The above plot shows the top speed of an EV in accordance to its Power Train Technology. We can see that AWD type have the max top speeds. Thus they are better choices. User can hover their cursor to the spots to get the clear idea of which power train will will go upto highest speeds.

(5) If user wants a car for his family, how can he select one?

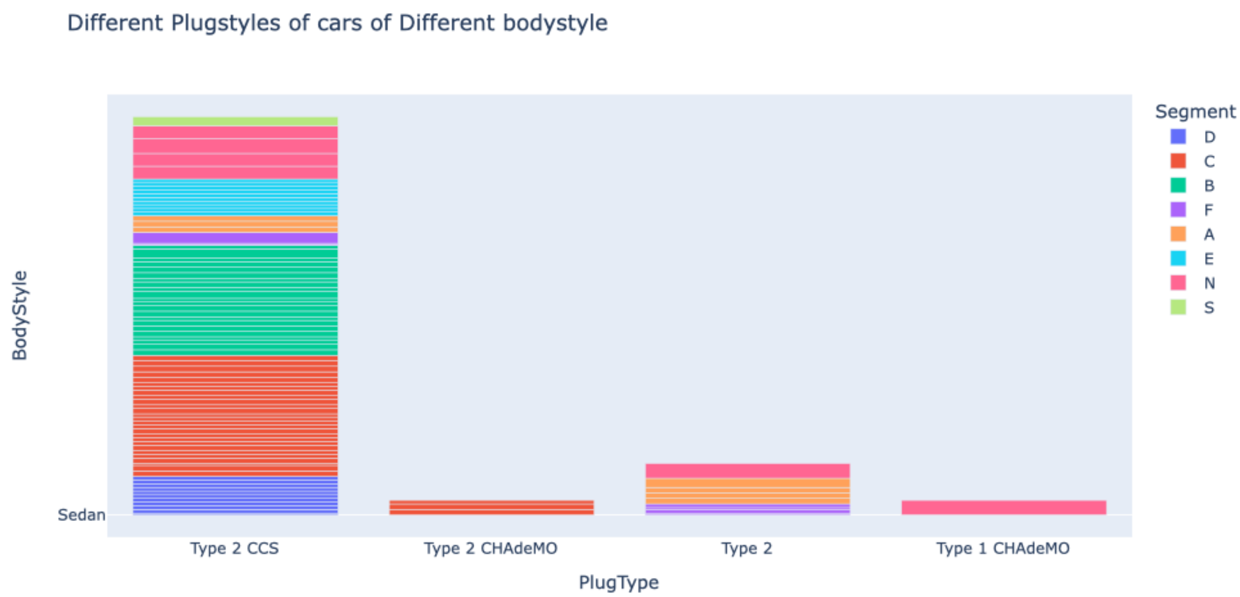
```
import warnings
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import scipy as sp
import seaborn as sns
from PIL import Image
from plotly.subplots import make_subplots
import plotly.express as px

ds = pd.read_csv("/Users/shivamraj/Desktop/VIT 3rd
Sem/DATA
VISUALIZATION/Project/ElectricCarData_Clean.csv.xls")

pie = px.pie(ds,values = "Seats", names = "Seats",
title = "Cars offering Seating Capacity")
pie
```



The above pie chart shows what percentage of car comes with how much seating capacity. If user wants a car for himself, he can go for a 4 or a 5 seater but if he has a family so he must think for a 6 or 7 seater car. This data in form of a pie interactive plot will give him a very clear idea about what’s right for him and also can look for a number of options.

(6) Show different types of plug-styles for different car segments.

```
import warnings
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import scipy as sp
import seaborn as sns
from PIL import Image
from plotly.subplots import make_subplots
import plotly.express as px

ds = pd.read_csv("/Users/shivamraj/Desktop/VIT 3rd
Sem/DATA
VISUALIZATION/Project/ElectricCarData_Clean.csv.xls")



fig = px.bar(ds,x="PlugType", y= "BodyStyle",
color="Segment", title="Different Plugstyles of cars
of Different bodystyle")
fig.show()
```

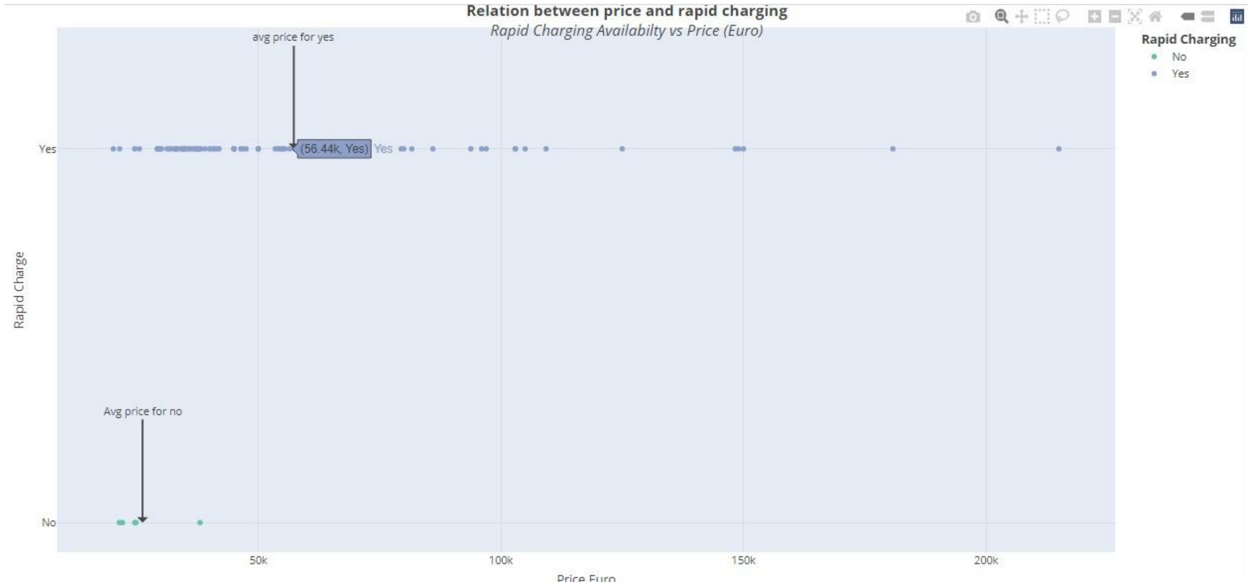


The above interactive bar plot shows segments of car which use different kinds of plug types. This will help the users to find what plug type is the most commonly available on highways or if they have any chargers from before so that they can reuse that in their new car. Colour wise interpretation makes that even more easier to find one for them. This plot also has all interactivity functions in it.

(7) Deduce relation between Price and the Fast Charging Technology

```
library(tidyverse)
library(ggplot2)
library(plotly)
evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarDa
ta_Clean.csv')
pwrtrn <- evs[, c(4:6, 9)]
avg_rng <- aggregate(pwrtrn$Range_Km ~
pwrtrn$PowerTrain, FUN = mean)

Rapidcharge_price <- evs[, c(8, 14)]
Rapidcharge_price <-
Rapidcharge_price[order(Rapidcharge_price$PriceEuro),
]
avg_price <- aggregate(Rapidcharge_price$PriceEuro ~
Rapidcharge_price$RapidCharge, FUN = "mean")
avg_price <- avg_price %>% rename(RapidCharge =
`Rapidcharge_price$RapidCharge`, Avg_Price =
`Rapidcharge_price$PriceEuro`)
head(avg_price)

plt <- plot_ly(data=Rapidcharge_price, x= ~PriceEuro,
y= ~RapidCharge, color=~RapidCharge ,type="scatter")
plt <- plt %>%  layout(title = '<b>Relation between
price and rapid charging</b><br><i>Rapid Charging
Availabilty vs Price (Euro)</i>', plot_bgcolor =
"#e5ecf6", xaxis = list(title = 'Price Euro'),
                       yaxis = list(title = 'Rapid
Charge'), legend = list(title=list(text='<b> Rapid
Charging </b>')))
```

```
plt<- plt %>%

  add_annotations( x = avg_price$Avg_Price[2],
                   y = 1,
                   xref = "x", yref = "y",
                   axref = "x", ayref = "y",
                   text = "avg price for yes",
                   showarrow = T,
                   ax = avg_price$Avg_Price[2],
                   ay = 1.30)
plt<- plt %>%

  add_annotations( x = avg_price$Avg_Price[1],
                   y = 0,
                   xref = "x", yref = "y",
                   axref = "x", ayref = "y",
                   text = "Avg price for no",
                   showarrow = T,
                   ax = avg_price$Avg_Price[1],
                   ay = 0.30)

plt
```

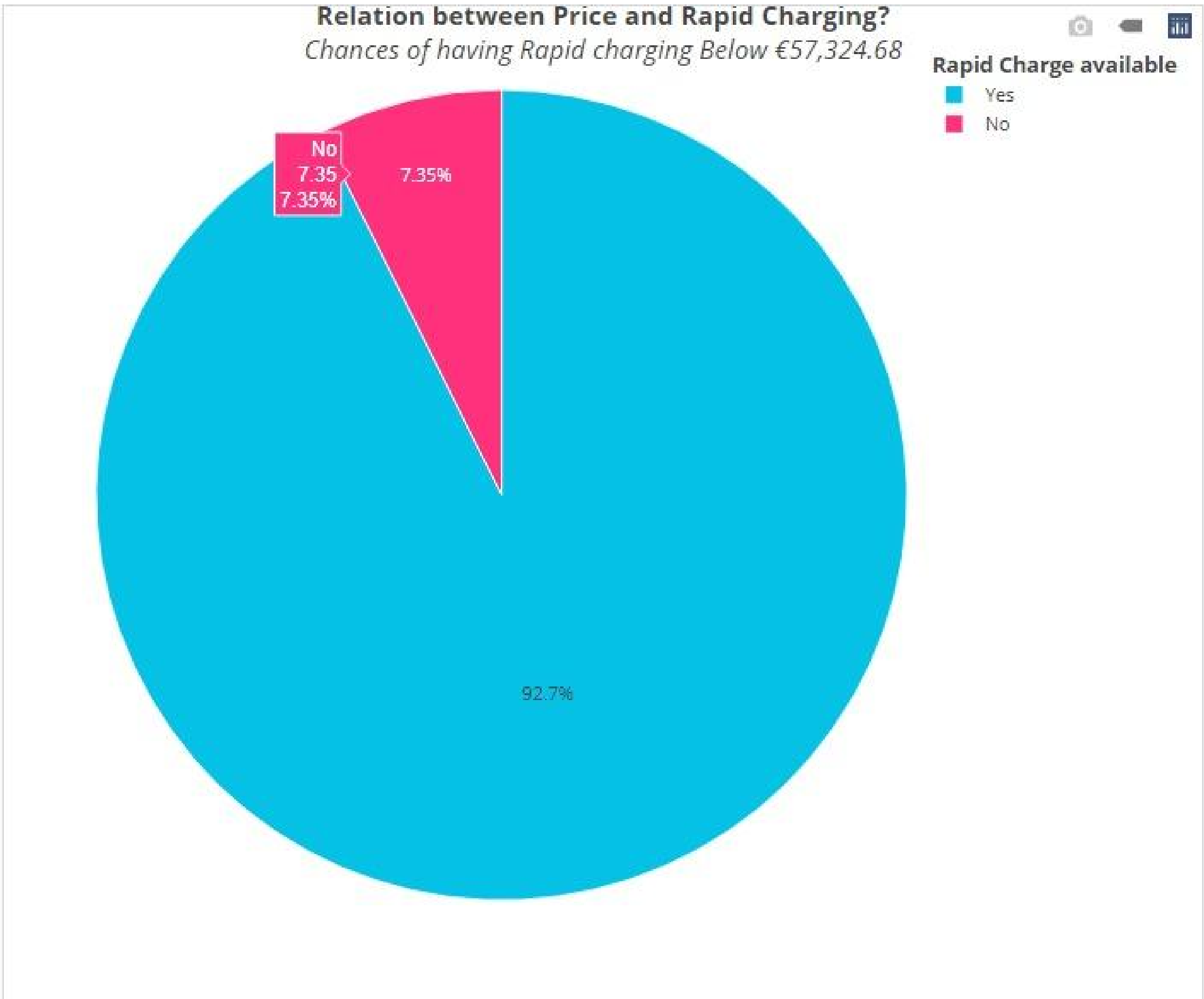


This plot shows the relation between Price of the car and weather it has rapid charging or not. This interactive plot helps user choose wisely as there is an arrow pointer which they can use to select particular price ranged cars. User can also zoom in to the graph for better clarity of number of options and also they can snip or capture the plot

(8) How Many cars have rapid charging whose price is below a particular range

```
library(tidyverse)
library(ggplot2)
library(plotly)
evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarDa
ta_Clean.csv')

Rapidcharge_price <- evs[, c(8, 14)]
Rapidcharge_price <-
Rapidcharge_price[order(Rapidcharge_price$PriceEuro),
]
avg_price <- aggregate(Rapidcharge_price$PriceEuro ~
Rapidcharge_price$RapidCharge, FUN = "mean")
avg_price <- avg_price %>% rename(RapidCharge =
`Rapidcharge_price$RapidCharge`, Avg_Price =
`Rapidcharge_price$PriceEuro`)
less_than_avg <- Rapidcharge_price %>%
filter(PriceEuro < avg_price$Avg_Price[2])
yes_percentage <- round((nrow(less_than_avg %>%
filter(RapidCharge == "Yes")) /nrow(less_than_avg)) *
100, 2)
no_percentage <- round(100 - yes_percentage, 2)
percentages <- data.frame(rapidcharge = c("Yes",
"No"), percentage = rbind("1" = yes_percentage,"2" =
no_percentage))
head(percentages)

colors <- c('rgb(5, 193, 230)', 'rgb(255, 48, 124)')
```

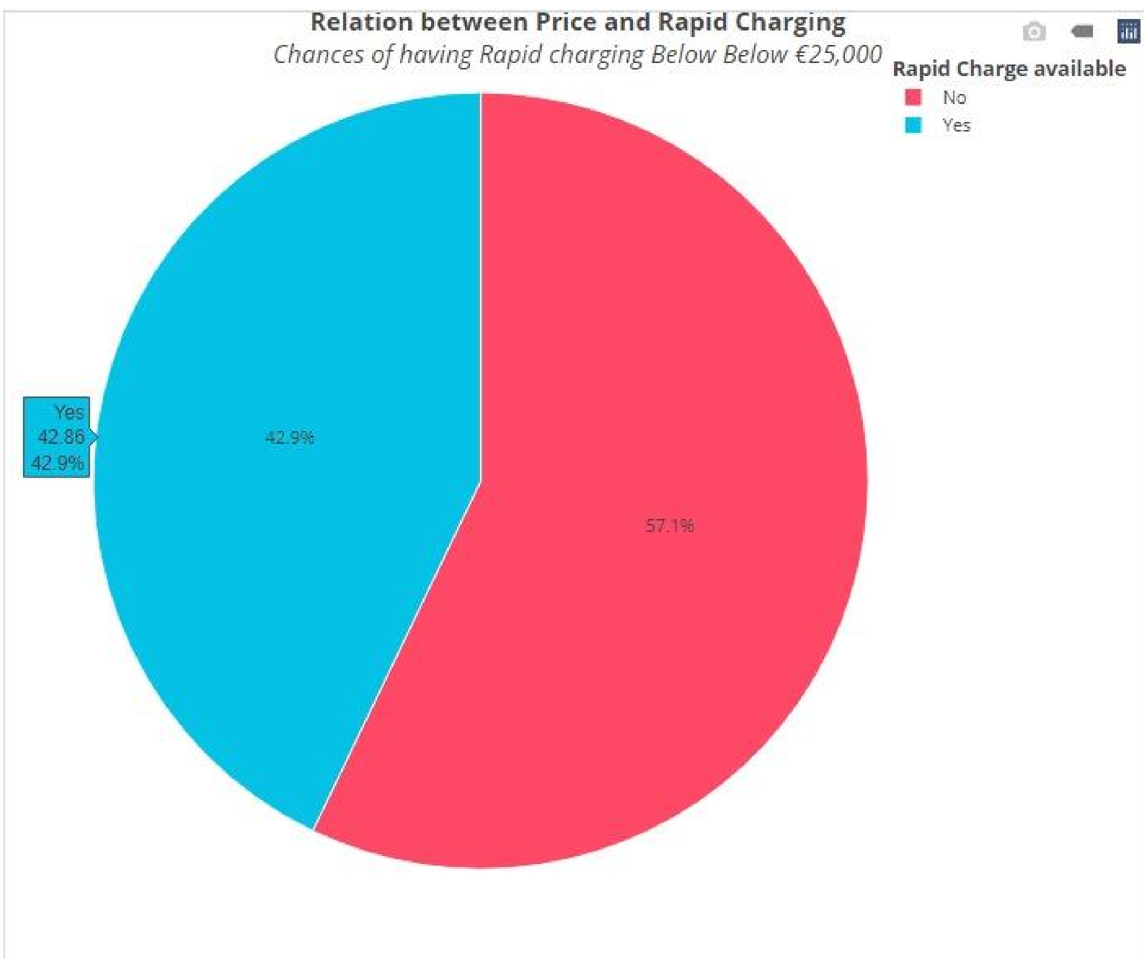
```
fig <- plot_ly(percentages, labels = ~rapidcharge,
values = ~percentage , type = 'pie',marker =
list(colors = colors,

line = list(color = '#FFFFFF', width = 1)))

fig <- fig %>% layout(title = '<b>Relation between
Price and Rapid Charging?</b><br><i>Chances of having
Rapid charging Below €57,324.68</i>',
                      xaxis = list(showgrid = FALSE,
zeroline = FALSE, showticklabels = FALSE),
                      yaxis = list(showgrid = FALSE,
zeroline = FALSE, showticklabels = FALSE),
                      legend =
list(title=list(text='<b> Rapid Charge available
</b>')))
fig
```



The above shown pie chart shows the percentage of cars supporting rapid charging below a price range so that users can make a smart choice by getting technologies at a cheaper price.

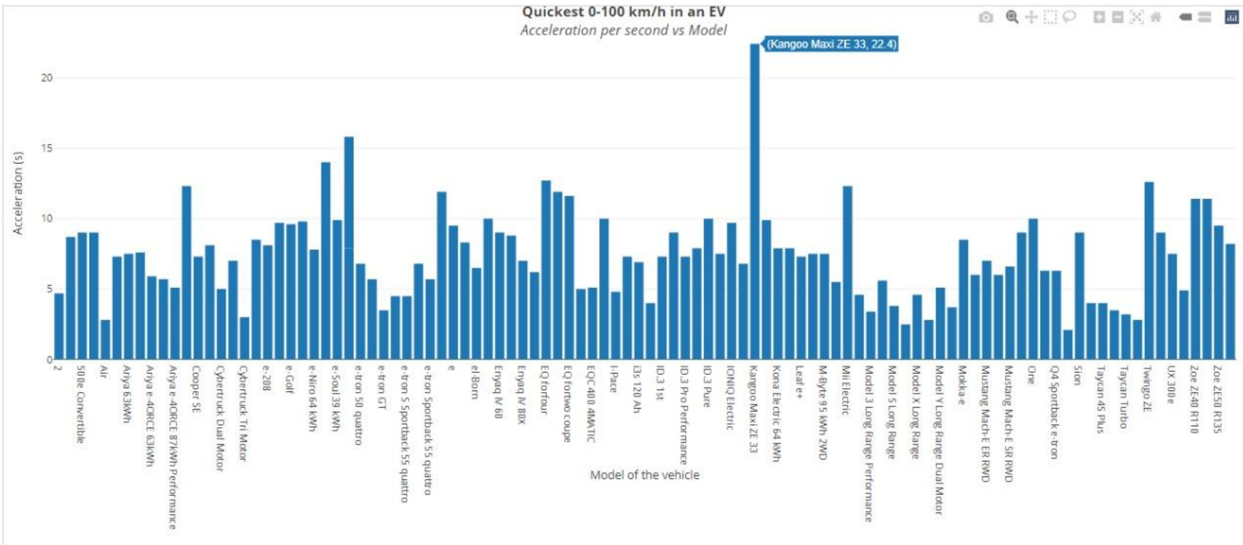
```
library(tidyverse)
library(ggplot2)
library(plotly)

evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarDa
ta_Clean.csv')

Rapidcharge_price <- evs[, c(8, 14)]
Rapidcharge_price <-
Rapidcharge_price[order(Rapidcharge_price$PriceEuro),
]
avg_price <- aggregate(Rapidcharge_price$PriceEuro ~
Rapidcharge_price$RapidCharge, FUN = "mean")
avg_price <- avg_price %>% rename(RapidCharge =
`Rapidcharge_price$RapidCharge`, Avg_Price =
`Rapidcharge_price$PriceEuro`)
less_than_25 <- Rapidcharge_price %>% filter(PriceEuro
< 25000)
yes_percentage <- round((nrow(less_than_25 %>%
filter(RapidCharge == "Yes")) /nrow(less_than_25)) *
100, 2)
no_percentage <- round(100 - yes_percentage, 2)
percentages_2 <- data.frame(rapidcharge = c("Yes",
"No"), percentage = rbind("1" = yes_percentage,"2" =
no_percentage))

colors <- c('rgb(5, 193, 230)', 'rgb(255, 71, 102)')

fig <- plot_ly(percentages_2, labels = ~rapidcharge,
values = ~percentage , type = 'pie',marker =
list(colors = colors,

line = list(color = '#FFFFFF', width = 1)))
```

```
fig <- fig %>% layout(title = '<b>Relation between
Price and Rapid Charging</b><br><i>Chances of having
Rapid charging Below Below €25,000</i>',
                      xaxis = list(showgrid = FALSE,
zeroline = FALSE, showticklabels = FALSE),
                      yaxis = list(showgrid = FALSE,
zeroline = FALSE, showticklabels = FALSE),
                      legend =
list(title=list(text='<b> Rapid Charge available
</b>')))
fig
```



(9) Which particular model of the car attains quickest 0-100 speed in least time

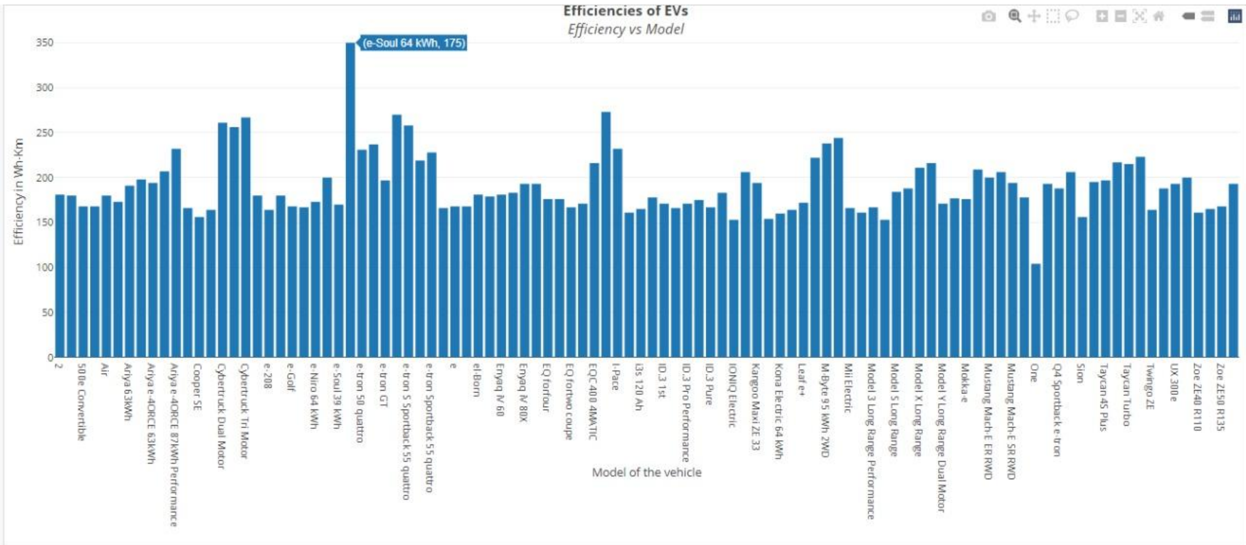
```
library(tidyverse)
library(ggplot2)
library(plotly)

evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarDa
ta_Clean.csv')

ev_acc<-evs[,1:3][order(evs$AccelSec),]

plt <- plot_ly(ev_acc, x = ~Model, y = ~AccelSec, type
='bar')
plt <- plt %>%  layout(title = "<b>Quickest 0-100 km/h
in an EV</b><br><i>Acceleration per second vs
Model</i>",

                        yaxis = list(title =
"Acceleration (s)"),
                        xaxis = list(title = "Model of
the vehicle") )
plt
```



The above interactive bar plot is mainly for speed enthusiasts as people have this misconception of weather EVs can deliver the power and speed that normal fuel vehicles do. So by this interactive graph users can easily go for cars which have faster accelerations with all interactive functions in it.

(10) Which particular model of the car is good in terms of efficiency

```
library(tidyverse)
library(ggplot2)
library(plotly)

evs <-
read.csv('C:/Users/Administrator/Desktop/ElectricCarDa
ta_Clean.csv')

ev_efficiency<-evs[,c(1:2,6)][order(evs$Efficiency_WhK
m),]

plt <- plot_ly(ev_efficiency, x = ~Model, y =
~Efficiency_WhKm, type ='bar')
plt <- plt %>%  layout(title = "<b>Efficiencies of
EVs</b><br><i>Efficiency vs Model</i>",

                       yaxis = list(title =
"Efficiency in Wh-Km"),
                       xaxis = list(title = "Model of
the vehicle") )

plt
```

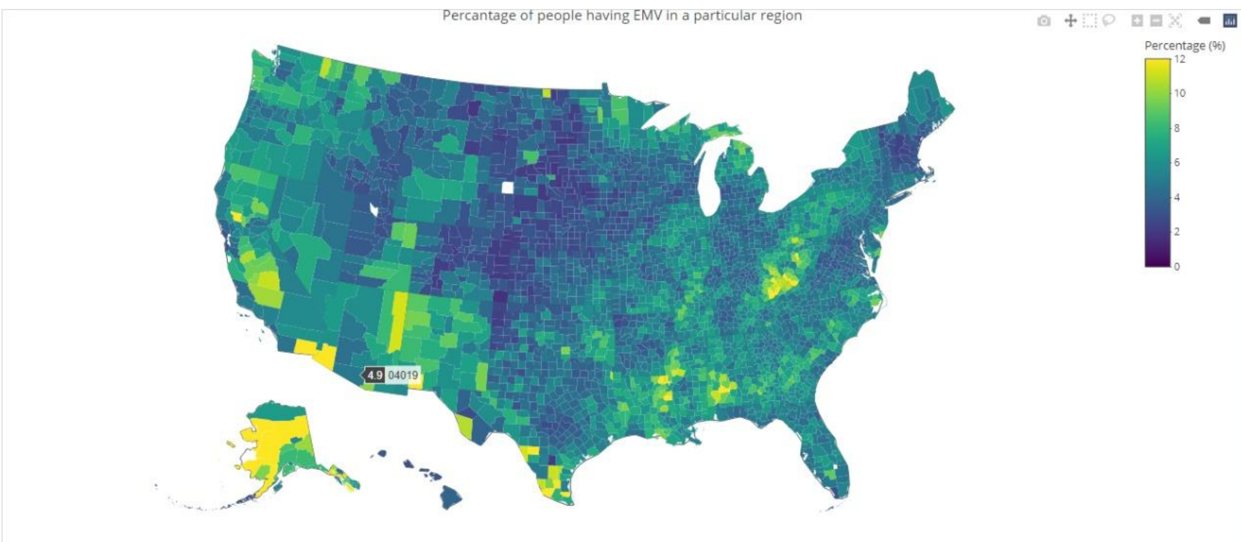


The above interactive bar plot is mainly for people who look for an economic drive as people have this misconception of whether EVs can deliver the same kind of fuel economy fuelled cars do. So by this interactive graph users can easily go for cars which have good economy with all interactive functions in it.

(11) Show that how much percentage of people own electric vehicles across the world through a geographic plot.

```
library(plotly)
library(rjson)

url <-
'https://raw.githubusercontent.com/plotly/datasets/mas
ter/geojson-counties-fips.json'
counties <- rjson::fromJSON(file=url)
url2<-
"https://raw.githubusercontent.com/plotly/datasets/mas
ter/fips-unemp-16.csv"
df <- read.csv(url2, colClasses=c(fips="character"))
g <- list(
  scope = 'usa',
  projection = list(type = 'albers usa'),
  showlakes = TRUE,
  lakecolor = toRGB('white')
)
fig <- plot_ly()
fig <- fig %>% add_trace(
  type="choropleth",
  geojson=counties,
  locations=df$fips,
  z=df$unemp,
  colorscale="Viridis",
  zmin=0,
  zmax=12,
  marker=list(line=list(
    width=0)
  )
)
```

```
fig <- fig %>% colorbar(title = " Percentage (%)")
fig <- fig %>% layout(
  title = "Percantage of people having EMV in a
particular region"
)

fig <- fig %>% layout(
  geo = g
)

fig
```



This interactive plot shows the popularity of EVs across USA. This data will lead new users to trust the technology and also see the availability across globe to make a wise decision.

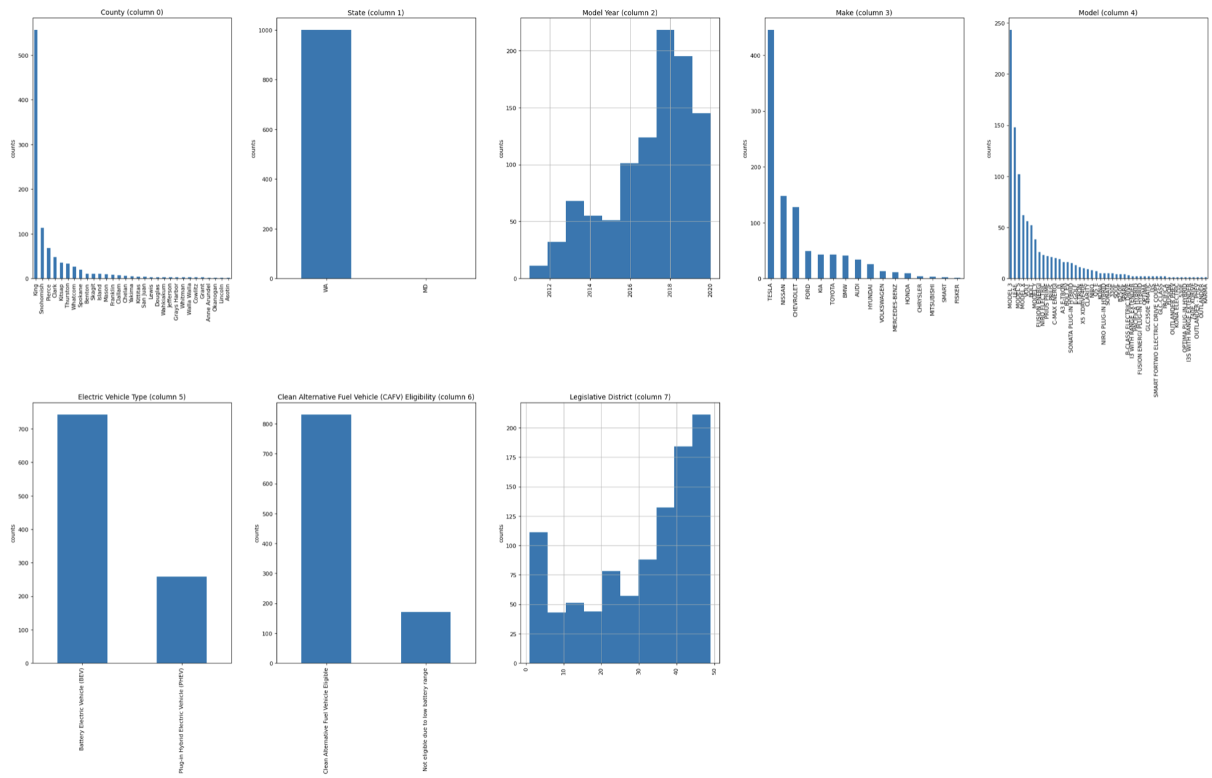
(12) Matrix Plotting

```
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd

nRowsRead = 1000
df1 = pd.read_csv('/Users/shivamraj/Desktop/VIT 3rd
Sem/DATA
VISUALIZATION/Project/Electric_Vehicle_Population_Data
(1).csv', delimiter=',', nrows = nRowsRead)


def plotPerColumnDistribution(df, nGraphShown,
nGraphPerRow):
    nunique = df.nunique()
    df = df[[col for col in df if nunique[col] > 1 and
nunique[col] < 50]]
    nRow, nCol = df.shape
    columnNames = list(df)
    nGraphRow = (nCol + nGraphPerRow - 1) /
nGraphPerRow
    plt.figure(num = None, figsize = (6 *
nGraphPerRow, 8 * nGraphRow), dpi = 80, facecolor =
'w', edgecolor = 'k')
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]),
np.number)):
```

```
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation = 90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad = 1.0, w_pad = 1.0, h_pad =
1.0)
    plt.show()



def plotCorrelationMatrix(df, graphWidth):
    filename = df.dataframeName
    df = df.dropna('columns') # drop columns with NaN
    df = df[[col for col in df if df[col].nunique() >
1]] # keep columns where there are more than 1 unique
values
    if df.shape[1] < 2:
        print(f'No correlation plots shown: The number
of non-NaN or constant columns ({df.shape[1]}) is less
than 2')
        return
    corr = df.corr()
    plt.figure(num=None, figsize=(graphWidth,
graphWidth), dpi=80, facecolor='w', edgecolor='k')
    corrMat = plt.matshow(corr, fignum = 1)
    plt.xticks(range(len(corr.columns)), corr.columns,
rotation=90)
    plt.yticks(range(len(corr.columns)), corr.columns)
    plt.gca().xaxis.tick_bottom()
    plt.colorbar(corrMat)
    plt.title(f'Correlation Matrix for {filename}',
fontsize=15)
```

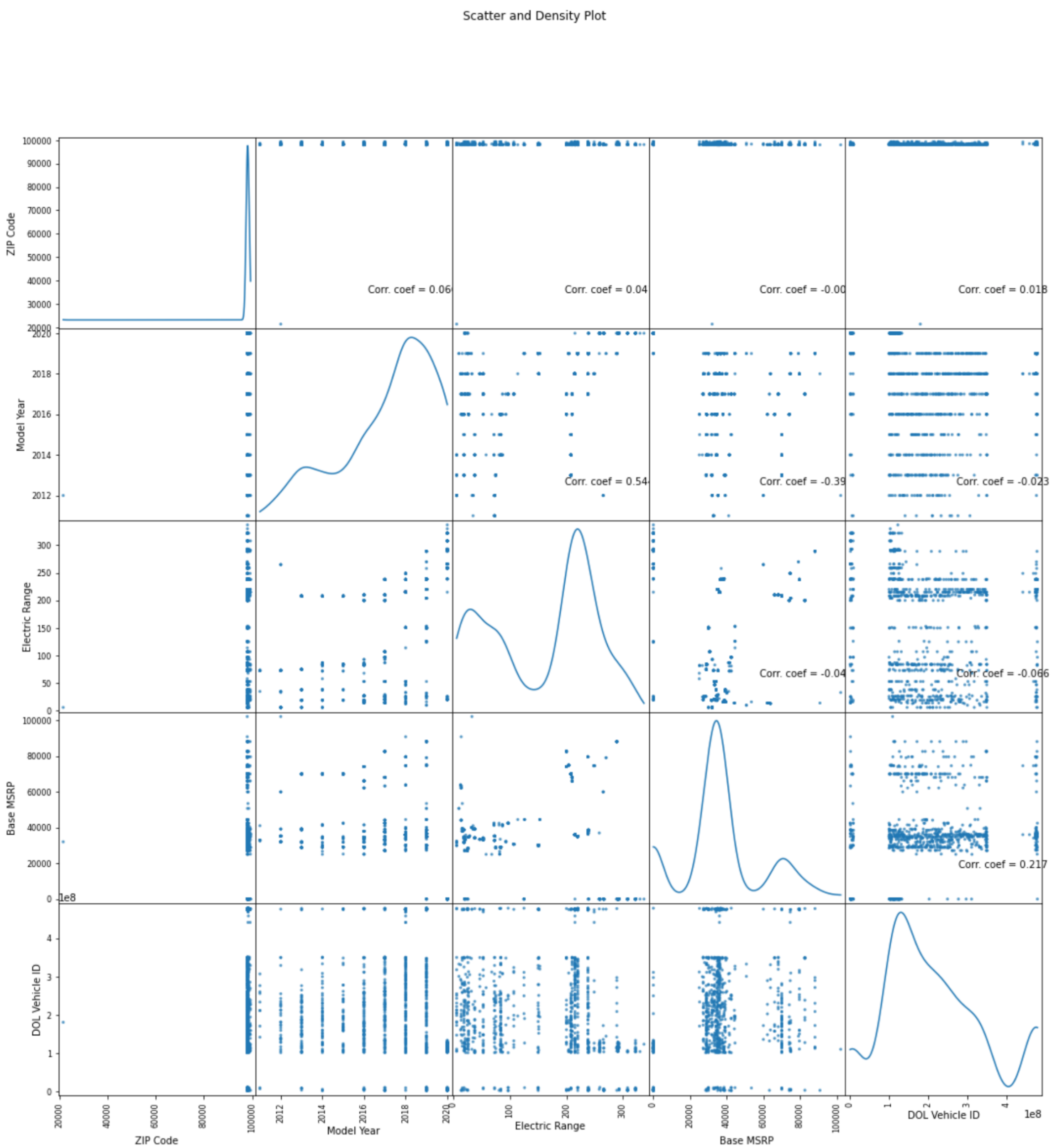
```
    plt.show()




def plotScatterMatrix(df, plotSize, textSize):
    df = df.select_dtypes(include =[np.number]) # keep only numerical columns
    # Remove rows and columns that would lead to df being singular
    df = df.dropna('columns')
    df = df[[col for col in df if df[col].nunique() > 1]] # keep columns where there are more than 1 unique values
    columnNames = list(df)
    if len(columnNames) > 10: # reduce the number of columns for matrix inversion of kernel density plots
        columnNames = columnNames[:10]
    df = df[columnNames]
    ax = pd.plotting.scatter_matrix(df, alpha=0.75, figsize=[plotSize, plotSize], diagonal='kde')
    corrs = df.corr().values
    for i, j in zip(*plt.np.triu_indices_from(ax, k = 1)):
        ax[i, j].annotate('Corr. coef = %.3f' % corrs[i, j], (0.8, 0.2), xycoords='axes fraction', ha='center', va='center', size=textSize)
    plt.suptitle('Scatter and Density Plot')
    plt.show()
```

```
plotPerColumnDistribution(df1, 10, 5)
```



The graph is bar plot and histograms with distribution over coloumns ie we have plotted all columns respectively.

```
plotScatterMatrix(df1, 18, 10)
```



We have plotted a scatter plot alongwith a density plot which has been portrayed as a square matrix form
Where all columns are compared to each other

```
plotCorrelationMatrix(df1, 8)
```

This is again similar as the last one. It is a Correlation Matrix where we have compared columns with each other and added colour for the interpretation