Data Analysis on Electric Vehicles

- --------- TASK - 1 ---------- *

```
import pandas as pd

# Load the dataset

data=pd.read_csv("C:/Users/Admin/Downloads/dataset.csv")
data.head()

   VIN (1-10)     County      City State  Postal Code  Model Year
Make  \
0  JTMEB3FV6N     Monroe  Key West    FL        33040        2022
TOYOTA
1  1G1RD6E45D      Clark  Laughlin    NV        89029        2013
CHEVROLET
2  JN1AZ0CP8B     Yakima    Yakima    WA        98901        2011
NISSAN
3  1G1FW6S08H     Skagit  Concrete    WA        98237        2017
CHEVROLET
4  3FA6P0SU1K  Snohomish   Everett    WA        98201        2019
FORD

        Model                   Electric Vehicle Type  \
0  RAV4 PRIME  Plug-in Hybrid Electric Vehicle (PHEV)
1        VOLT  Plug-in Hybrid Electric Vehicle (PHEV)
2        LEAF          Battery Electric Vehicle (BEV)
3     BOLT EV          Battery Electric Vehicle (BEV)
4      FUSION  Plug-in Hybrid Electric Vehicle (PHEV)

  Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
0          Clean Alternative Fuel Vehicle Eligible              42
1          Clean Alternative Fuel Vehicle Eligible              38
2          Clean Alternative Fuel Vehicle Eligible              73
3          Clean Alternative Fuel Vehicle Eligible             238
4            Not eligible due to low battery range              26

   Base MSRP  Legislative District  DOL Vehicle ID  \
0          0                   NaN       198968248
1          0                   NaN         5204412
2          0                  15.0       218972519
3          0                  39.0       186750406
4          0                  38.0         2006714

              Vehicle Location        Electric Utility  2020 Census
Tract
0    POINT (-81.80023 24.5545)                     NaN
12087972100
1  POINT (-114.57245 35.16815)                     NaN
```

```
32003005702
2  POINT (-120.50721 46.60448)              PACIFICORP
53077001602
3   POINT (-121.7515 48.53892)  PUGET SOUND ENERGY INC
53057951101
4  POINT (-122.20596 47.97659)  PUGET SOUND ENERGY INC
53061041500

# shape of the data

data.shape

(112634, 17)

# data information

data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 112634 entries, 0 to 112633
Data columns (total 17 columns):
 #   Column                                              Non-Null Count
Dtype
---  ------                                              --------------
-----
 0   VIN (1-10)                                          112634 non-
null  object
 1   County                                              112634 non-
null  object
 2   City                                                112634 non-
null  object
 3   State                                               112634 non-
null  object
 4   Postal Code                                         112634 non-
null  int64
 5   Model Year                                          112634 non-
null  int64
 6   Make                                                112634 non-
null  object
 7   Model                                               112614 non-
null  object
 8   Electric Vehicle Type                               112634 non-
null  object
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  112634 non-
null  object
 10  Electric Range                                      112634 non-
null  int64
 11  Base MSRP                                           112634 non-
null  int64
 12  Legislative District                                112348 non-
```

```
null  float64
 13  DOL Vehicle ID                                     112634 non-
null  int64
 14  Vehicle Location                                   112610 non-
null  object
 15  Electric Utility                                   112191 non-
null  object
 16  2020 Census Tract                                  112634 non-
null  int64
dtypes: float64(1), int64(6), object(10)
memory usage: 14.6+ MB

# describing the data

data.describe()

         Postal Code     Model Year  Electric Range      Base MSRP  \
count  112634.000000  112634.000000   112634.000000  112634.000000
mean    98156.226850    2019.003365       87.812987    1793.439681
std      2648.733064       2.892364      102.334216   10783.753486
min      1730.000000    1997.000000        0.000000       0.000000
25%     98052.000000    2017.000000        0.000000       0.000000
50%     98119.000000    2020.000000       32.000000       0.000000
75%     98370.000000    2022.000000      208.000000       0.000000
max     99701.000000    2023.000000      337.000000  845000.000000

       Legislative District  DOL Vehicle ID  2020 Census Tract
count         112348.000000    1.126340e+05       1.126340e+05
mean              29.805604    1.994567e+08       5.296650e+10
std               14.700545    9.398427e+07       1.699104e+09
min                1.000000    4.777000e+03       1.101001e+09
25%               18.000000    1.484142e+08       5.303301e+10
50%               34.000000    1.923896e+08       5.303303e+10
75%               43.000000    2.191899e+08       5.305307e+10
max               49.000000    4.792548e+08       5.603300e+10

# column to list

data.columns.tolist()

['VIN (1-10)',
 'County',
 'City',
 'State',
 'Postal Code',
 'Model Year',
 'Make',
 'Model',
 'Electric Vehicle Type',
 'Clean Alternative Fuel Vehicle (CAFV) Eligibility',
 'Electric Range',
 'Base MSRP',
 'Legislative District',
 'DOL Vehicle ID',
 'Vehicle Location',
 'Electric Utility',
 '2020 Census Tract']
```

```python
# checking for missing values

data.isnull().sum()
```

```
VIN (1-10)                                             0
County                                                 0
City                                                   0
State                                                  0
Postal Code                                            0
Model Year                                             0
Make                                                   0
Model                                                 20
Electric Vehicle Type                                  0
Clean Alternative Fuel Vehicle (CAFV) Eligibility      0
Electric Range                                         0
Base MSRP                                              0
Legislative District                                 286
DOL Vehicle ID                                         0
Vehicle Location                                      24
Electric Utility                                     443
2020 Census Tract                                      0
dtype: int64
```

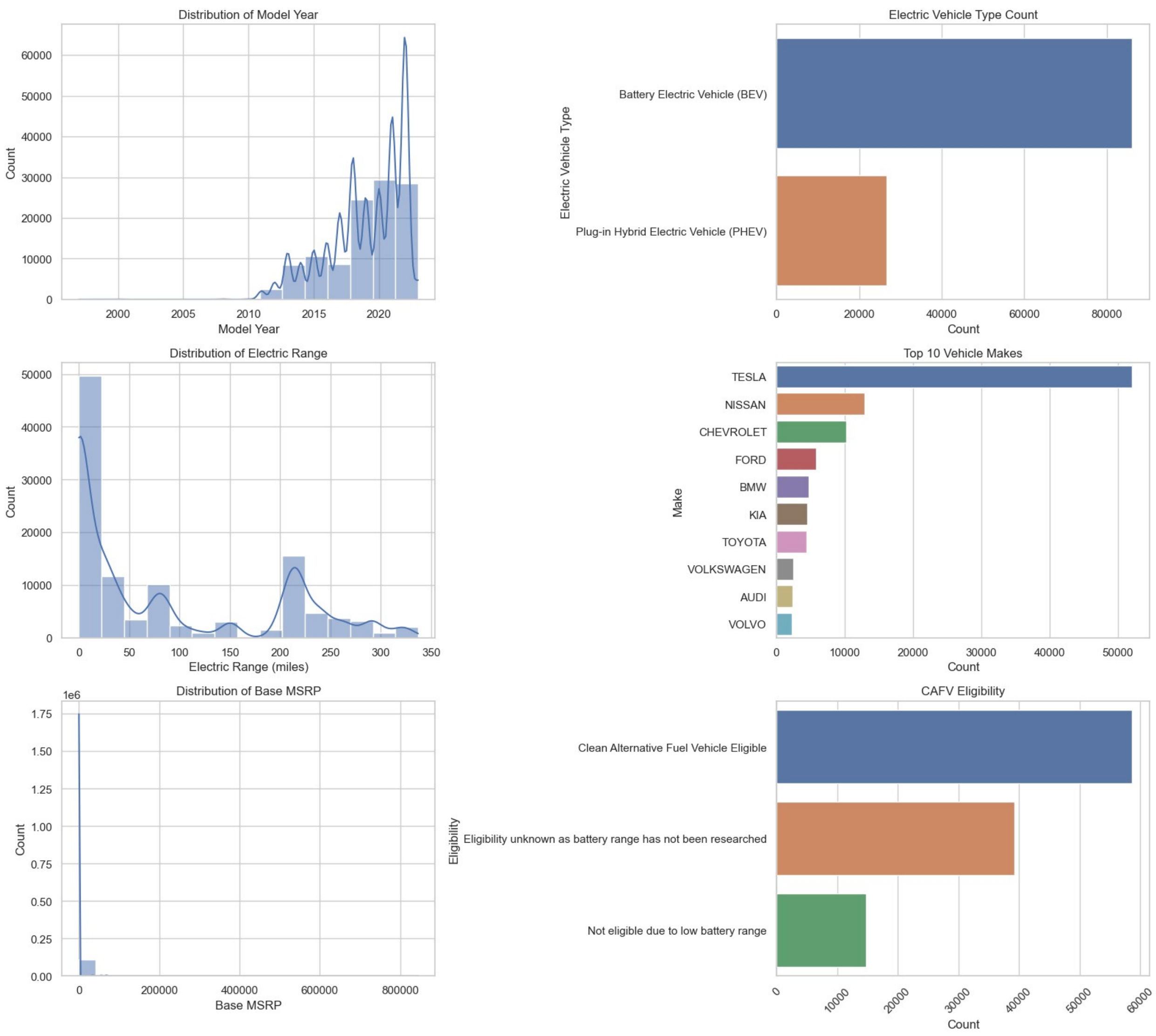
```python
# checking for duplicate values

data.nunique()
```

```
VIN (1-10)                                             7548
County                                                  165
City                                                    629
State                                                    45
Postal Code                                             773
Model Year                                               20
Make                                                     34
Model                                                   114
Electric Vehicle Type                                     2
Clean Alternative Fuel Vehicle (CAFV) Eligibility         3
Electric Range                                          101
Base MSRP                                                30
Legislative District                                     49
DOL Vehicle ID                                       112634
Vehicle Location                                        758
```

```
Electric Utility                                         73
2020 Census Tract                                      2026
dtype: int64
```

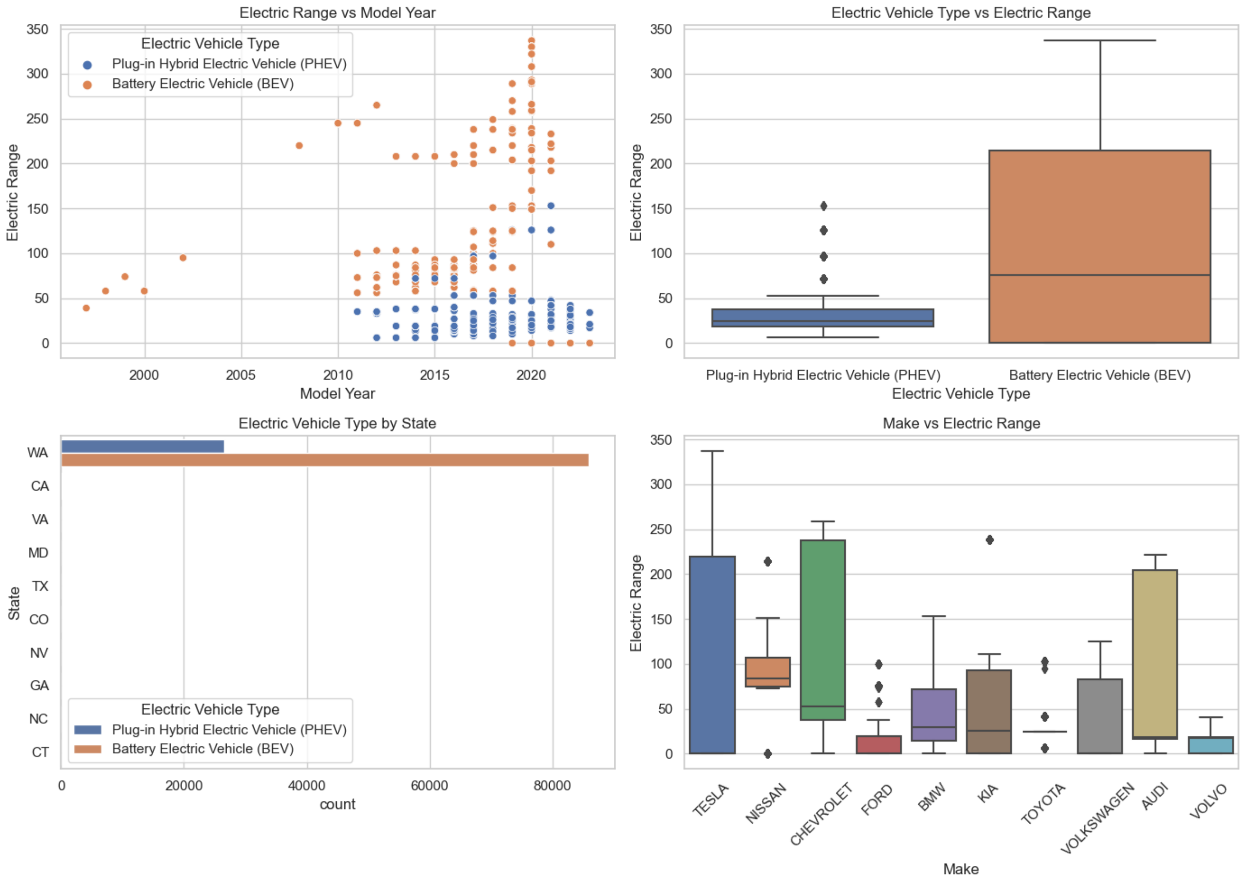
```python
# Univariate Analysis
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings as wr
wr.filterwarnings('ignore')

# Set the style for the plots
sns.set(style="whitegrid")

# Create subplots for multiple charts
fig, axes = plt.subplots(3,2,figsize=(16,14))  # 3 rows, 2 columns

# 1. Distribution of Model Year (Continuous Variable)
sns.histplot(data['Model Year'], bins=15, kde=True, ax=axes[0, 0])
axes[0, 0].set_title('Distribution of Model Year')
axes[0, 0].set_xlabel('Model Year')
axes[0, 0].set_ylabel('Count')

# 2. Distribution of Electric Vehicle Type (Categorical Variable)
sns.countplot(y='Electric Vehicle Type', data=data, ax=axes[0, 1],
              order=data['Electric Vehicle
Type'].value_counts().index)
axes[0, 1].set_title('Electric Vehicle Type Count')
axes[0, 1].set_xlabel('Count')
axes[0, 1].set_ylabel('Electric Vehicle Type')

# 3. Distribution of Electric Range (Continuous Variable)
sns.histplot(data['Electric Range'], bins=15, kde=True, ax=axes[1, 0])
axes[1, 0].set_title('Distribution of Electric Range')
axes[1, 0].set_xlabel('Electric Range (miles)')
axes[1, 0].set_ylabel('Count')

# 4. Top 10 Vehicle Makes (Categorical Variable)
sns.countplot(y='Make', data=data, ax=axes[1, 1],
              order=data['Make'].value_counts().index[:10])  # Top 10
makes
axes[1, 1].set_title('Top 10 Vehicle Makes')
axes[1, 1].set_xlabel('Count')
axes[1, 1].set_ylabel('Make')

# 5. Distribution of Base MSRP (Continuous Variable)
sns.histplot(data['Base MSRP'], bins=20, kde=True, ax=axes[2, 0])
axes[2, 0].set_title('Distribution of Base MSRP')
axes[2, 0].set_xlabel('Base MSRP')
axes[2, 0].set_ylabel('Count')
```

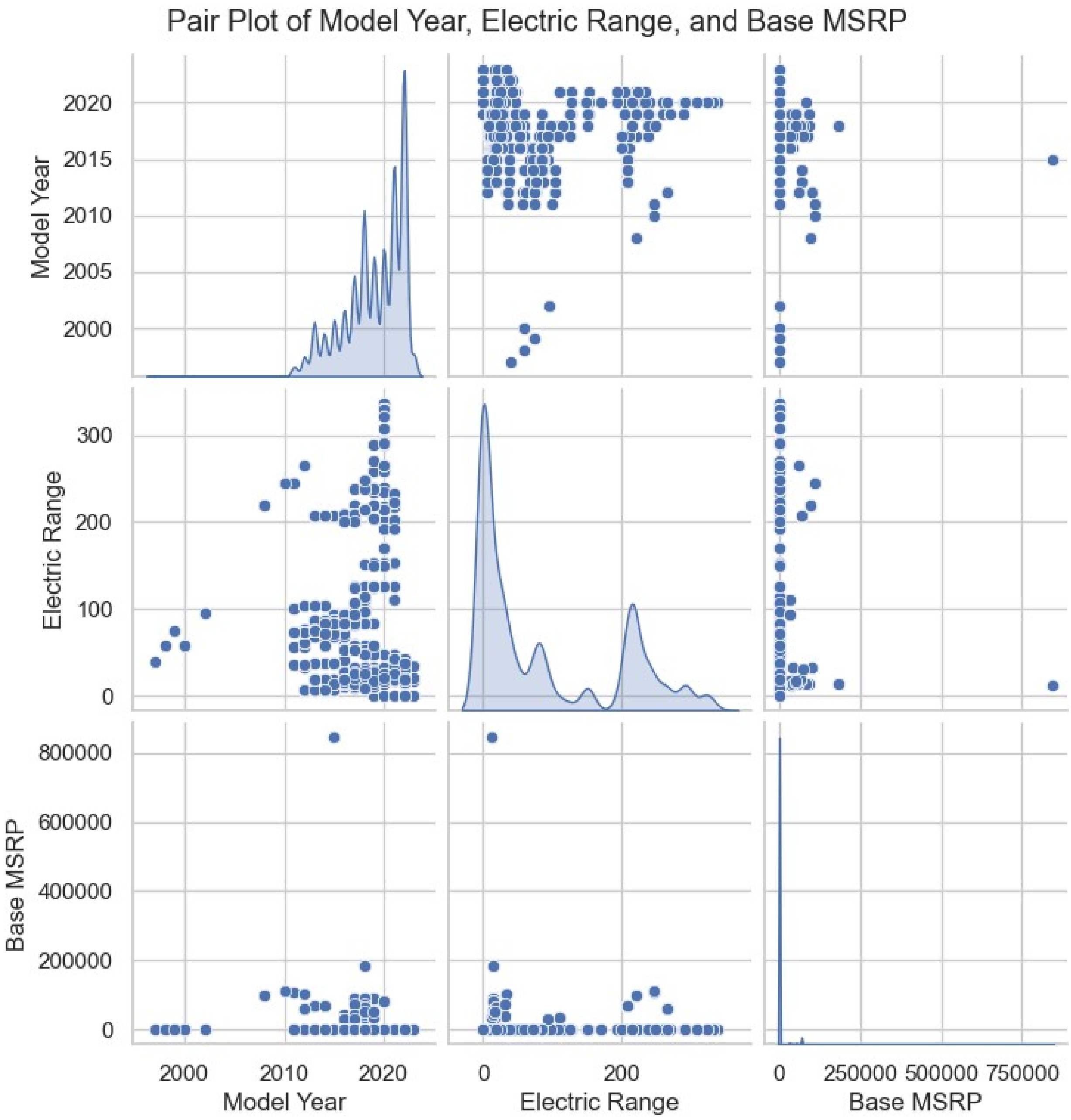
```python
# 6. Distribution of Clean Alternative Fuel Vehicle Eligibility
(Categorical)
sns.countplot(y='Clean Alternative Fuel Vehicle (CAFV) Eligibility',
data=data, ax=axes[2, 1],
              order=data['Clean Alternative Fuel Vehicle (CAFV)
Eligibility'].value_counts().index)
axes[2, 1].set_title('CAFV Eligibility')
axes[2, 1].set_xlabel('Count')
axes[2, 1].set_ylabel('Eligibility')

# Adjust the layout for better readability
plt.tight_layout()

# Rotate x-axis labels if necessary (for categorical variables with
long names)
plt.xticks(rotation=45)

plt.show()
```



```
# Set the style for the plots
sns.set(style="whitegrid")

# --- Bivariate Analysis ---

# Create subplots for bivariate charts
fig, axes = plt.subplots(2, 2, figsize=(14, 10))

# 1. Scatter Plot: Model Year vs Electric Range (Numerical vs
Numerical)
sns.scatterplot(x='Model Year', y='Electric Range', hue='Electric
Vehicle Type', data=data, ax=axes[0, 0])
axes[0, 0].set_title('Electric Range vs Model Year')

# 2. Box Plot: Electric Vehicle Type vs Electric Range (Categorical vs
Numerical)
```

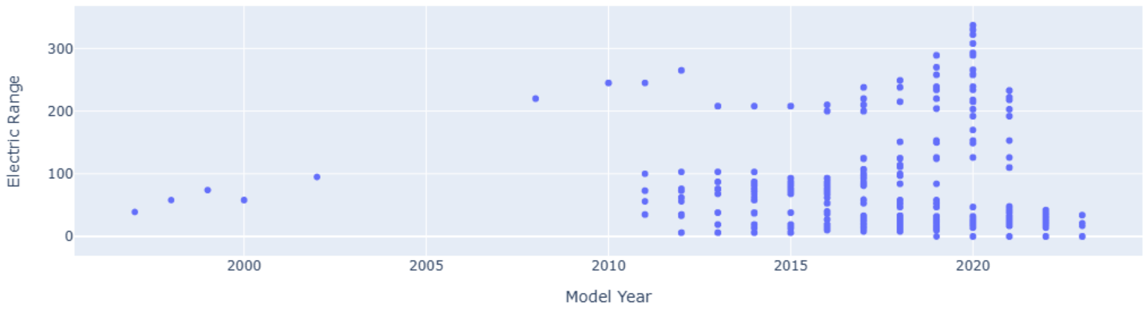
```
sns.boxplot(x='Electric Vehicle Type', y='Electric Range', data=data,
ax=axes[0, 1])
axes[0, 1].set_title('Electric Vehicle Type vs Electric Range')

# 3. Count Plot: State vs Electric Vehicle Type (Categorical vs
Categorical)
sns.countplot(y='State', hue='Electric Vehicle Type', data=data,
ax=axes[1, 0],
              order=data['State'].value_counts().index[:10])  # Top 10
states
axes[1, 0].set_title('Electric Vehicle Type by State')

# 4. Box Plot: Make vs Electric Range (Categorical vs Numerical)
sns.boxplot(x='Make', y='Electric Range', data=data, ax=axes[1, 1],
            order=data['Make'].value_counts().index[:10])  # Top 10
makes
axes[1, 1].set_title('Make vs Electric Range')
axes[1, 1].tick_params(axis='x', rotation=45)

plt.tight_layout()
plt.show()


# --- Multivariate Analysis ---
# Pair Plot for selected numerical variables
sns.pairplot(data[['Model Year', 'Electric Range', 'Base MSRP']],
diag_kind='kde')
plt.suptitle('Pair Plot of Model Year, Electric Range, and Base MSRP',
y=1.02)
plt.show()
```


Electric Range vs Model Year
Electric Vehicle Type
Plug-in Hybrid Electric Vehicle (PHEV)
Battery Electric Vehicle (BEV)
Electric Range
350
300
250
200
150
100
50
0
2000
2005
2010
2015
2020
Model Year
Electric Vehicle Type vs Electric Range
Electric Range
350
300
250
200
150
100
50
0
Plug-in Hybrid Electric Vehicle (PHEV)
Battery Electric Vehicle (BEV)
Electric Vehicle Type
Electric Vehicle Type by State
State
WA
CA
VA
MD
TX
CO
NV
GA
NC
CT
Electric Vehicle Type
Plug-in Hybrid Electric Vehicle (PHEV)
Battery Electric Vehicle (BEV)
0
20000
40000
60000
80000
count
Make vs Electric Range
Electric Range
350
300
250
200
150
100
50
0
TESLA
NISSAN
CHEVROLET
FORD
BMW
KIA
TOYOTA
VOLKSWAGEN
AUDI
VOLVO
Make



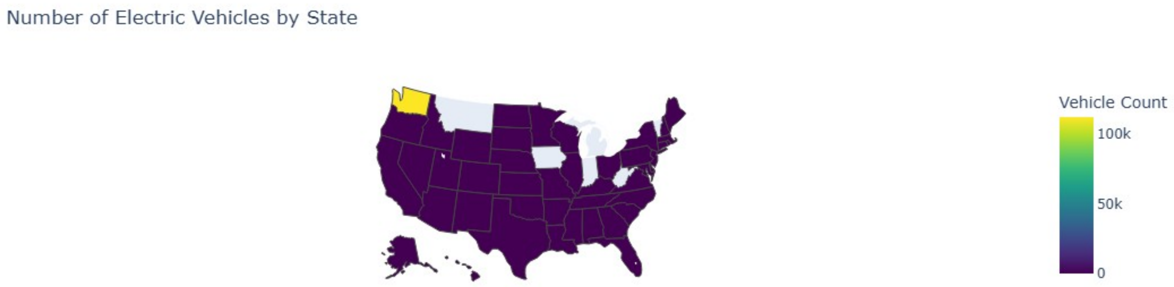
- --------- TASK - 2 ---------- *

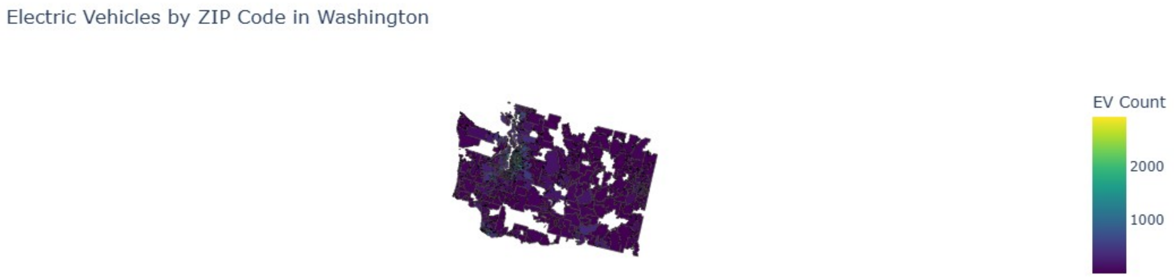
```
!pip install plotly

Requirement already satisfied: plotly in c:\users\admin\anaconda3\lib\
site-packages (5.9.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\admin\
anaconda3\lib\site-packages (from plotly) (8.2.2)

import plotly.express as px

px.scatter(data,x='Model Year',y='Electric Range')
```



```
# Prepare data for choropleth
location_data = data.groupby(['State'])['VIN (1-
10)'].count().reset_index()
location_data.columns = ['State', 'Vehicle Count']

# Create choropleth map using plotly
fig = px.choropleth(location_data,
                    locations='State',
                    locationmode="USA-states",
                    color='Vehicle Count',
                    scope="usa",
                    color_continuous_scale="Viridis",
                    title="Number of Electric Vehicles by State")

# Show the choropleth
fig.show()

# Filter the data to include only Washington (WA)
wa_data = data[data['State'] == 'WA']

# Group by Postal Code (ZIP) to count the number of vehicles in each
ZIP code
zipcode_data = wa_data.groupby('Postal Code')['VIN (1-
10)'].count().reset_index()
zipcode_data.columns = ['ZIP Code', 'Vehicle Count']

# Convert ZIP codes to string for consistency
zipcode_data['ZIP Code'] = zipcode_data['ZIP Code'].astype(str)

# Create a choropleth map using plotly
fig = px.choropleth(zipcode_data,

geojson='https://raw.githubusercontent.com/OpenDataDE/State-zip-code-
GeoJSON/master/wa_washington_zip_codes_geo.min.json',
                    locations='ZIP Code',
                    featureidkey="properties.ZCTA5CE10",  # This
matches ZIP codes in the geojson file
```

```
                    color='Vehicle Count',
                    color_continuous_scale="Viridis",
                    scope="usa",
                    labels={'Vehicle Count':'EV Count'},
                    title="Electric Vehicles by ZIP Code in
Washington")

# Update the map layout
fig.update_geos(fitbounds="locations", visible=False)

# Show the choropleth map
fig.show()
```



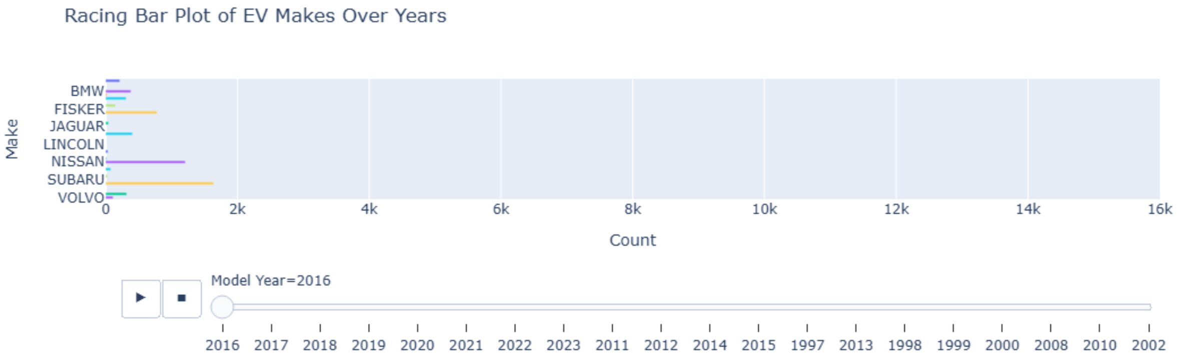


- --------- TASK - 3 --------- *

```
!pip install bar-chart-race

Requirement already satisfied: bar-chart-race in c:\users\admin\
anaconda3\lib\site-packages (0.1.0)
Requirement already satisfied: pandas>=0.24 in c:\users\admin\
anaconda3\lib\site-packages (from bar-chart-race) (2.1.4)
Requirement already satisfied: matplotlib>=3.1 in c:\users\admin\
anaconda3\lib\site-packages (from bar-chart-race) (3.8.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
```

```
(1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(1.4.4)
Requirement already satisfied: numpy<2,>=1.21 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(10.2.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\admin\
anaconda3\lib\site-packages (from matplotlib>=3.1->bar-chart-race)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\
anaconda3\lib\site-packages (from pandas>=0.24->bar-chart-race)
(2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\admin\
anaconda3\lib\site-packages (from pandas>=0.24->bar-chart-race)
(2023.3)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\
lib\site-packages (from python-dateutil>=2.7->matplotlib>=3.1->bar-
chart-race) (1.16.0)
```

```
import pandas as pd

# Group by 'Make' and 'Model Year', then count the number of vehicles
make_counts = data.groupby(['Make', 'Model Year'])['VIN (1-
10)'].count().reset_index()
make_counts.columns = ['Make', 'Model Year', 'Count']
import plotly.express as px

# Create the racing bar plot
fig = px.bar(
    make_counts,
    x='Count',
    y='Make',
    color='Make',
    animation_frame='Model Year',
    range_x=[0, make_counts['Count'].max() * 1.1],  # Adjust x-axis range
    title='Racing Bar Plot of EV Makes Over Years',
    orientation='h'  # Horizontal bar chart
)

# Update layout for better visibility
fig.update_layout(
    yaxis=dict(title='Make'),
    xaxis=dict(title='Count'),
    showlegend=False
)

fig.show()
```


Racing Bar Plot of EV Makes Over Years
Make
BMW
FISKER
JAGUAR
LINCOLN
NISSAN
SUBARU
VOLVO
2k
4k
6k
8k
10k
12k
14k
16k
Count
▶
■
Model Year=2016