Electric Vehicle

```
import numpy as np
import pandas as pd
import warnings
import matplotlib.pyplot as plt

df = pd.read_csv("F:/Innomatics/Electric Vehicles/dataset.csv")
df.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
```

```
df.tail()
```

```
      Make  Model_Year  Number_of_Vehicles
200  VOLVO        2019                 190
201  VOLVO        2020                 162
202  VOLVO        2021                 580
203  VOLVO        2022                 882
204  VOLVO        2023                  21
```

```
df.shape
```

```
(112634, 17)
```

```
df.describe()
```

```
        Model_Year  Number_of_Vehicles
count   205.000000          205.000000
mean   2017.834146          548.770732
std       4.361350         1511.343773
min    1997.000000            1.000000
25%    2016.000000           44.000000
50%    2019.000000          146.000000
75%    2021.000000          580.000000
max    2023.000000        14548.000000
```

```
df.columns
```

```
Index(['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'],
      dtype='object')
```

```
df.columns = df.columns.str.replace(" ","_")
```

```
df.columns
```

```
Index(['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'],
      dtype='object')
```

Missing Values

```
# Check for missing values in the dataset
missing_values = df.isnull().sum()
missing_values
```

```
Make                  0
Model_Year            0
Number_of_Vehicles    0
dtype: int64
```

```
# Check for duplicates in the dataset
duplicates = df.duplicated().sum()
duplicates
```

```
0
```

Renaming Columns

```
df.rename
(columns={"Clean_Alternative_Fuel_Vehicle_(CAFV)_Eligibility":
"CAFV_Eligibility"}, inplace = True)
df.columns
```

```
Index(['VIN_(1-10)', 'County', 'City', 'State', 'Postal_Code',
'Model_Year',
       'Make', 'Model', 'Electric_Vehicle_Type', 'CAFV_Eligibility',
       'Electric_Range', 'Base_MSRP', 'Legislative_District',
'DOL_Vehicle_ID',
       'Vehicle_Location', 'Electric_Utility', '2020_Census_Tract'],
      dtype='object')
```

```
df.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   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
```

```
df.isnull().sum().sort_values(ascending=False)
```

```
Electric_Utility         443
Legislative_District     286
Vehicle_Location          24
Model                     20
VIN_(1-10)                 0
CAFV_Eligibility           0
DOL_Vehicle_ID             0
Base_MSRP                  0
Electric_Range             0
Electric_Vehicle_Type      0
County                     0
Make                       0
Model_Year                 0
Postal_Code                0
State                      0
City                       0
2020_Census_Tract          0
dtype: int64
```

```
df_dropna = df.dropna()
```

```
df_dropna.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 112152 entries, 2 to 112633
Data columns (total 17 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   VIN_(1-10)              112152 non-null  object
 1   County                  112152 non-null  object
```

```
 2   City                    112152 non-null  object
 3   State                   112152 non-null  object
 4   Postal_Code             112152 non-null  int64
 5   Model_Year              112152 non-null  int64
 6   Make                    112152 non-null  object
 7   Model                   112152 non-null  object
 8   Electric_Vehicle_Type   112152 non-null  object
 9   CAFV_Eligibility        112152 non-null  object
 10  Electric_Range          112152 non-null  int64
 11  Base_MSRP               112152 non-null  int64
 12  Legislative_District    112152 non-null  float64
 13  DOL_Vehicle_ID          112152 non-null  int64
 14  Vehicle_Location        112152 non-null  object
 15  Electric_Utility        112152 non-null  object
 16  2020_Census_Tract       112152 non-null  int64
dtypes: float64(1), int64(6), object(10)
memory usage: 15.4+ MB
```

Values (Aggregate and Categorical)

```
discrete_df = df.select_dtypes(include=['object'])

numerical_df = df.select_dtypes(include=['int64', 'float64'])
```

Univariate Analysis

```
def discrete_univariate_analysis(discrete_data):
    for col_name in discrete_data:
        print("*"*10, col_name, "*"*10)
        print(discrete_data[col_name].agg(['count', 'nunique',
'unique']))
        print('Value Counts: \n',
discrete_data[col_name].value_counts())
        print()
discrete_univariate_analysis(discrete_df)
```

```
********** VIN_(1-10) **********
count                                                  112634
nunique                                                  7548
unique     [JTMEB3FV6N, 1G1RD6E45D, JN1AZ0CP8B, 1G1FW6S08...
Name: VIN_(1-10), dtype: object
Value Counts:
 VIN_(1-10)
5YJYGDEE9M    472
5YJYGDEE0M    465
5YJYGDEE8M    448
```

```
5YJYGDEE7M    448
5YJYGDEE2M    437
              ...
WA1LAAGE9M      1
5UXKT0C50H      1
5YJYGAED3M      1
WDC0G5DBXL      1
YV4ED3GM0P      1
Name: count, Length: 7548, dtype: int64

********** County **********
count                                                112634
nunique                                                 165
unique     [Monroe, Clark, Yakima, Skagit, Snohomish, Isl...
Name: County, dtype: object
Value Counts:
 County
King          59000
Snohomish     12434
Pierce         8535
Clark          6689
Thurston       4126
              ...
Pinal             1
Elmore            1
Portsmouth        1
Kings             1
Kootenai          1
Name: count, Length: 165, dtype: int64

********** City **********
count                                                112634
nunique                                                 629
unique     [Key West, Laughlin, Yakima, Concrete, Everett...
Name: City, dtype: object
Value Counts:
 City
Seattle         20305
Bellevue         5921
Redmond          4201
Vancouver        4013
Kirkland         3598
                ...
Hartline            1
Gaithersburg        1
El Paso             1
Klickitat           1
Worley              1
Name: count, Length: 629, dtype: int64
```

```
********** State **********
count                                                  112634
nunique                                                    45
unique     [FL, NV, WA, IL, NY, VA, OK, KS, CA, NE, MD, C...
Name: State, dtype: object
Value Counts:
 State
WA    112348
CA        76
VA        36
MD        26
TX        14
CO         9
NV         8
GA         7
NC         7
CT         6
DC         6
FL         6
AZ         6
IL         6
SC         5
OR         5
NE         5
HI         4
UT         4
AR         4
NY         4
TN         3
KS         3
MO         3
PA         3
MA         3
LA         3
NJ         3
NH         2
OH         2
WY         2
ID         2
KY         1
RI         1
ME         1
MN         1
SD         1
WI         1
NM         1
AK         1
MS         1
AL         1
```

```
DE          1
OK          1
ND          1
Name: count, dtype: int64

********** Make **********
count                                                  112634
nunique                                                    34
unique     [TOYOTA, CHEVROLET, NISSAN, FORD, TESLA, KIA, ...
Name: Make, dtype: object
Value Counts:
 Make
TESLA             52078
NISSAN            12880
CHEVROLET         10182
FORD               5819
BMW                4680
KIA                4483
TOYOTA             4405
VOLKSWAGEN         2514
AUDI               2332
VOLVO              2288
CHRYSLER           1794
HYUNDAI            1412
JEEP               1152
RIVIAN              885
FIAT                822
PORSCHE             818
HONDA               792
MINI                632
MITSUBISHI          588
POLESTAR            558
MERCEDES-BENZ       506
SMART               273
JAGUAR              219
LINCOLN             168
CADILLAC            108
LUCID MOTORS         65
SUBARU               59
LAND ROVER           38
LEXUS                33
FISKER               20
GENESIS              18
AZURE DYNAMICS        7
TH!NK                 3
BENTLEY               3
Name: count, dtype: int64

********** Model **********
```

```
count                                                  112614
nunique                                                   114
unique     [RAV4 PRIME, VOLT, LEAF, BOLT EV, FUSION, MODE...
Name: Model, dtype: object
Value Counts:
 Model
MODEL 3        23135
MODEL Y        17142
LEAF           12880
MODEL S         7377
BOLT EV         4910
               ...
745LE              2
S-10 PICKUP        1
SOLTERRA           1
918                1
FLYING SPUR        1
Name: count, Length: 114, dtype: int64

********** Electric_Vehicle_Type **********
count                                                  112634
nunique                                                     2
unique     [Plug-in Hybrid Electric Vehicle (PHEV), Batte...
Name: Electric_Vehicle_Type, dtype: object
Value Counts:
 Electric_Vehicle_Type
Battery Electric Vehicle (BEV)            86044
Plug-in Hybrid Electric Vehicle (PHEV)    26590
Name: count, dtype: int64

********** CAFV_Eligibility **********
count                                                  112634
nunique                                                     3
unique     [Clean Alternative Fuel Vehicle Eligible, Not ...
Name: CAFV_Eligibility, dtype: object
Value Counts:
 CAFV_Eligibility
Clean Alternative Fuel Vehicle Eligible                           58639
Eligibility unknown as battery range has not been researched    39236
Not eligible due to low battery range                             14759
Name: count, dtype: int64

********** Vehicle_Location **********
count                                                  112610
nunique                                                   758
unique     [POINT (-81.80023 24.5545), POINT (-114.57245 ...
Name: Vehicle_Location, dtype: object
Value Counts:
 Vehicle_Location
POINT (-122.13158 47.67858)    2916
```

```
POINT (-122.2066 47.67887)     2059
POINT (-122.1872 47.61001)     2001
POINT (-122.31765 47.70013)    1880
POINT (-122.12096 47.55584)    1852
                               ...
POINT (-124.33152 48.05431)       1
POINT (-77.41203 39.41574)        1
POINT (-123.61022 46.35588)       1
POINT (-112.04165 40.68741)       1
POINT (-116.91895 47.40077)       1
Name: count, Length: 758, dtype: int64

********** Electric_Utility **********
count                                                  112191
nunique                                                    73
unique     [nan, PACIFICORP, PUGET SOUND ENERGY INC, PUD ...
Name: Electric_Utility, dtype: object
Value Counts:
 Electric_Utility
PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)
40247
PUGET SOUND ENERGY INC
22172
CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)
21447
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF CLARK COUNTY - (WA)
6522
BONNEVILLE POWER ADMINISTRATION||CITY OF TACOMA - (WA)||PENINSULA
LIGHT COMPANY              5053

...
BONNEVILLE POWER ADMINISTRATION||PENINSULA LIGHT COMPANY
1
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF ASOTIN COUNTY
1
CITY OF SEATTLE - (WA)
1
BONNEVILLE POWER ADMINISTRATION||NESPELEM VALLEY ELEC COOP, INC
1
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF CLALLAM COUNTY|PUD NO 1
OF JEFFERSON COUNTY        1
Name: count, Length: 73, dtype: int64
```

Numerical Analysis

```
def numerical_univariate_analysis(numerical_data):
    for col_name in numerical_data:
```

```
        print("*"*10, col_name, "*"*10)
        print(numerical_data[col_name].agg(['min', 'max', 'mean',
'median', 'std']))
        print()
numerical_univariate_analysis(numerical_df)
```

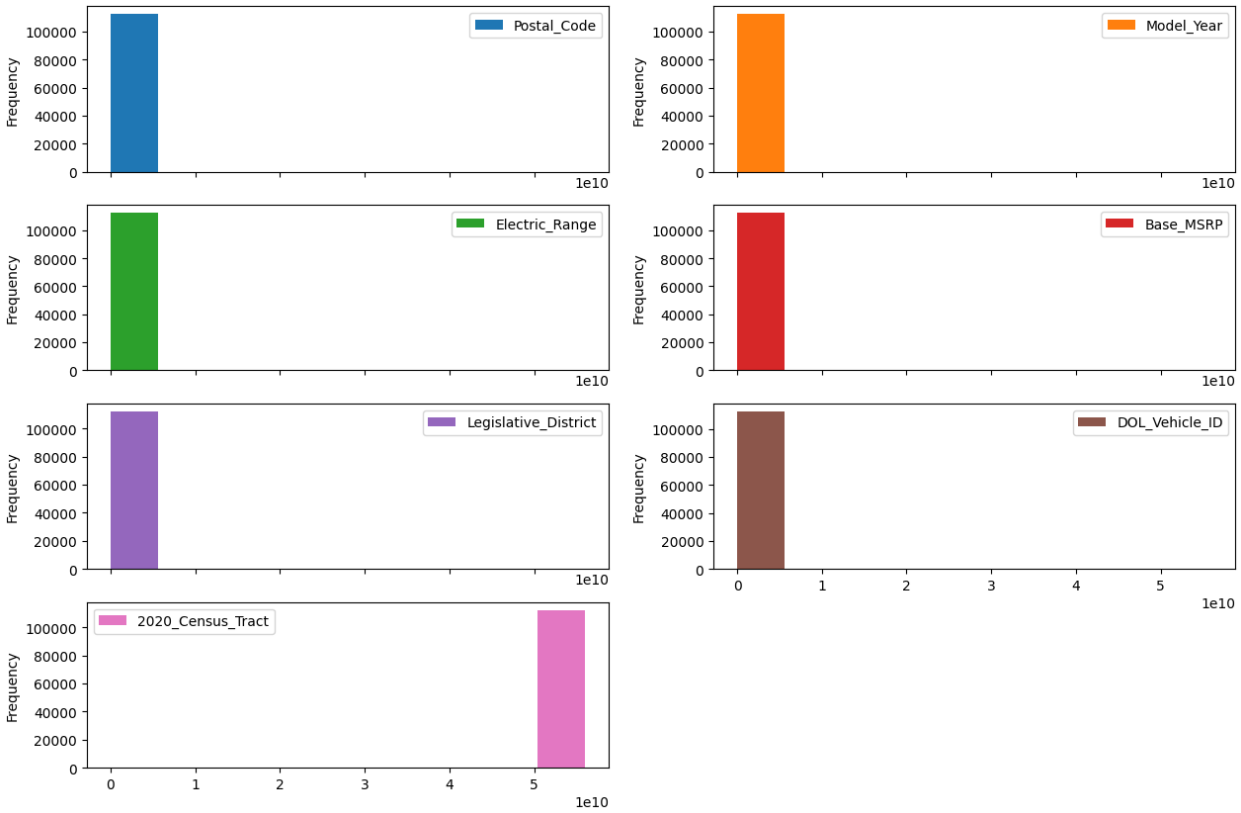
```
********** Postal_Code **********
min        1730.000000
max       99701.000000
mean      98156.226850
median    98119.000000
std        2648.733064
Name: Postal_Code, dtype: float64

********** Model_Year **********
min       1997.000000
max       2023.000000
mean      2019.003365
median    2020.000000
std          2.892364
Name: Model_Year, dtype: float64

********** Electric_Range **********
min         0.000000
max       337.000000
mean       87.812987
median     32.000000
std       102.334216
Name: Electric_Range, dtype: float64

********** Base_MSRP **********
min            0.000000
max       845000.000000
mean        1793.439681
median         0.000000
std        10783.753486
Name: Base_MSRP, dtype: float64

********** Legislative_District **********
min        1.000000
max       49.000000
mean      29.805604
median    34.000000
std       14.700545
Name: Legislative_District, dtype: float64

********** DOL_Vehicle_ID **********
min       4.777000e+03
max       4.792548e+08
mean      1.994567e+08
```

```
median    1.923896e+08
std       9.398427e+07
Name: DOL_Vehicle_ID, dtype: float64

********** 2020_Census_Tract **********
min       1.101001e+09
max       5.603300e+10
mean      5.296650e+10
median    5.303303e+10
std       1.699104e+09
Name: 2020_Census_Tract, dtype: float64
```

Visual Analysis (Univariate)

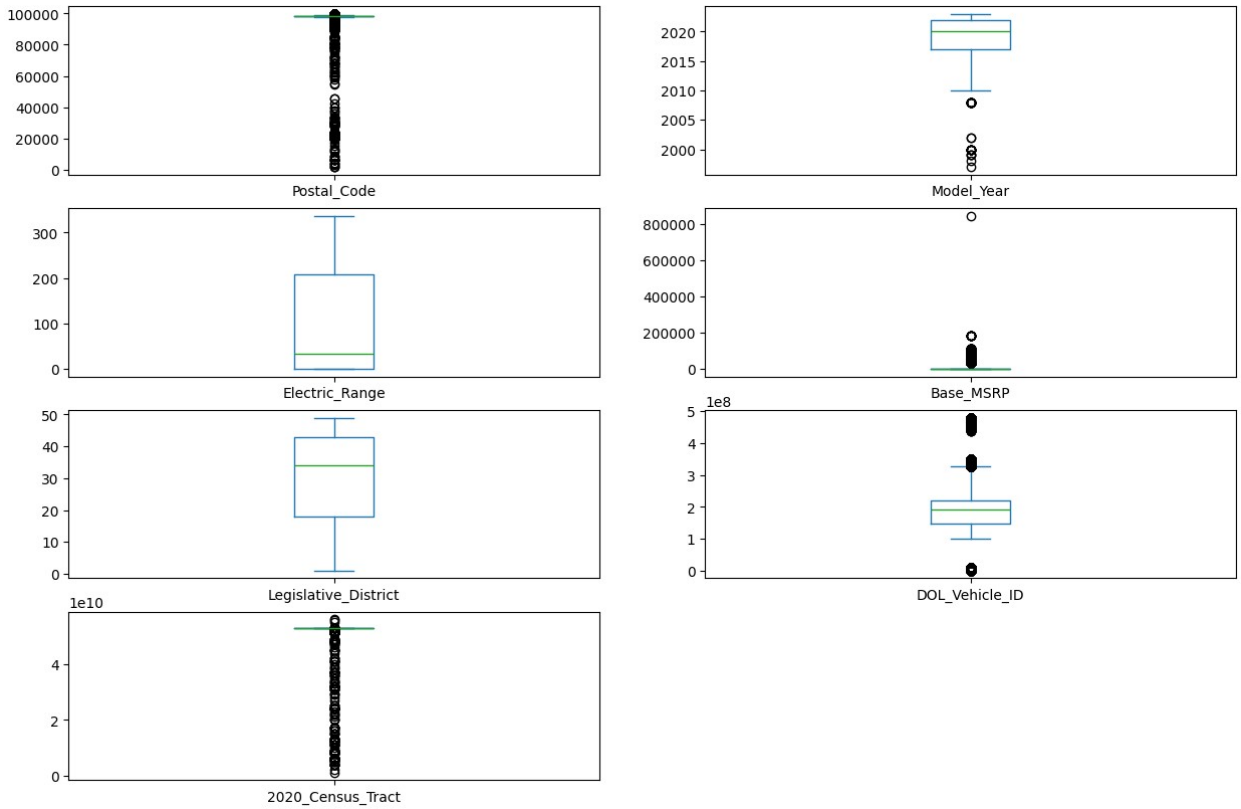
```
warnings.filterwarnings("ignore", category=UserWarning,
module="matplotlib.font_manager")

df.plot(kind='hist', subplots=True, layout=(4, 2), figsize=(15, 10))
```

```
array([[<Axes: ylabel='Frequency'>, <Axes: ylabel='Frequency'>],
       [<Axes: ylabel='Frequency'>, <Axes: ylabel='Frequency'>],
       [<Axes: ylabel='Frequency'>, <Axes: ylabel='Frequency'>],
       [<Axes: ylabel='Frequency'>, <Axes: ylabel='Frequency'>]],
      dtype=object)
```



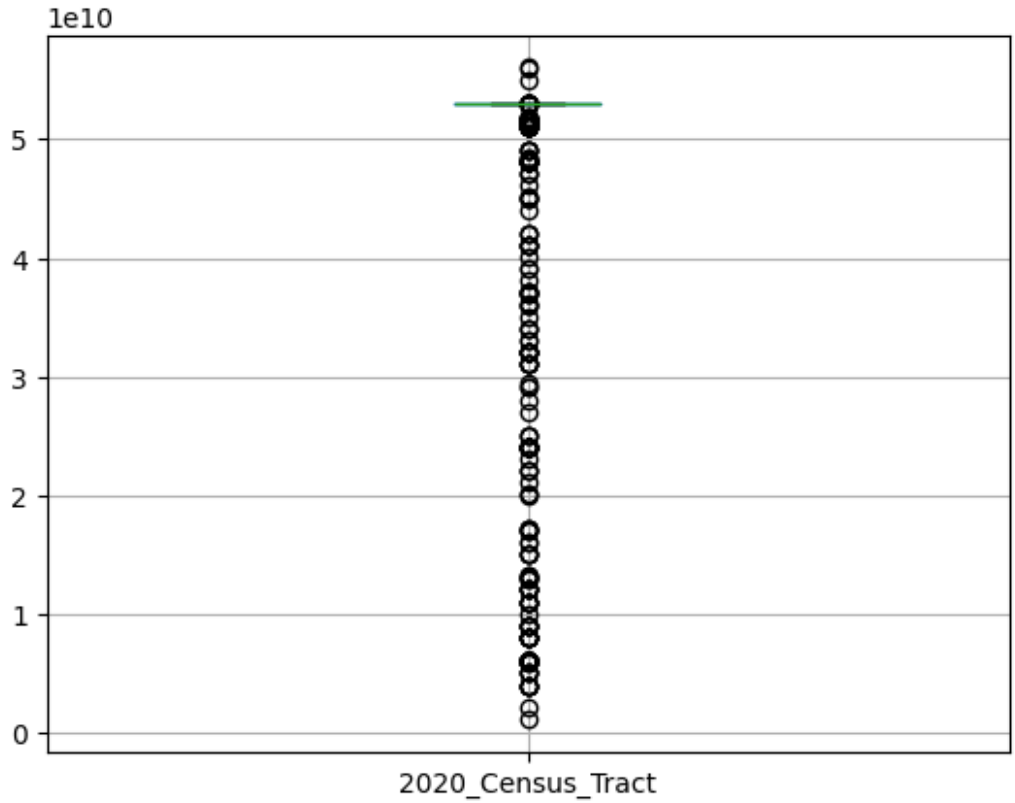

```
df.plot(kind='box', subplots=True, layout=(4, 2), figsize=(15, 10))
```

```
Postal_Code                Axes(0.125,0.712609;0.352273x0.167391)
Model_Year               Axes(0.547727,0.712609;0.352273x0.167391)
Electric_Range             Axes(0.125,0.511739;0.352273x0.167391)
Base_MSRP                Axes(0.547727,0.511739;0.352273x0.167391)
Legislative_District         Axes(0.125,0.31087;0.352273x0.167391)
DOL_Vehicle_ID           Axes(0.547727,0.31087;0.352273x0.167391)
2020_Census_Tract              Axes(0.125,0.11;0.352273x0.167391)
dtype: object
```




```
df.boxplot(column = "2020_Census_Tract")
```

```
<Axes: >
```



Bivariate Analysis

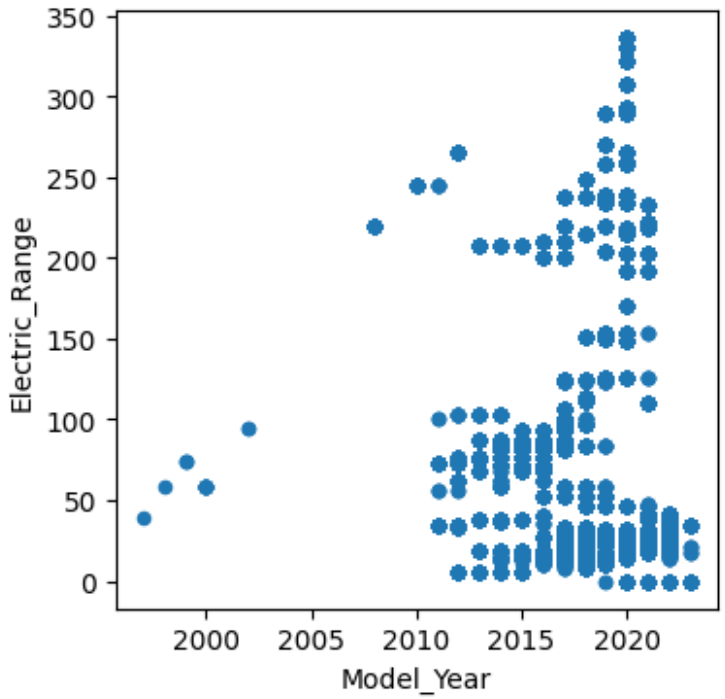
```
numerical_df.corr()
```

```
                      Postal_Code  Model_Year  Electric_Range
Base_MSRP  \
Postal_Code              1.000000   -0.004485        0.000385
0.001151
Model_Year              -0.004485    1.000000       -0.288433  -
0.229130
Electric_Range           0.000385   -0.288433        1.000000
0.085025
Base_MSRP                0.001151   -0.229130        0.085025
1.000000
Legislative_District    -0.433405    0.010439        0.024387
0.012426
DOL_Vehicle_ID           0.003365   -0.068295        0.009682
0.000504
2020_Census_Tract        0.501170    0.000714        0.000722
0.000979

                      Legislative_District  DOL_Vehicle_ID
2020_Census_Tract
```

```
Postal_Code                       -0.433405        0.003365
0.501170
Model_Year                         0.010439       -0.068295
0.000714
Electric_Range                     0.024387        0.009682
0.000722
Base_MSRP                          0.012426        0.000504
0.000979
Legislative_District               1.000000       -0.001671          -
0.111991
DOL_Vehicle_ID                    -0.001671        1.000000
0.002754
2020_Census_Tract                 -0.111991        0.002754
1.000000

numerical_df.columns

Index(['Postal_Code', 'Model_Year', 'Electric_Range', 'Base_MSRP',
       'Legislative_District', 'DOL_Vehicle_ID', '2020_Census_Tract'],
      dtype='object')
```

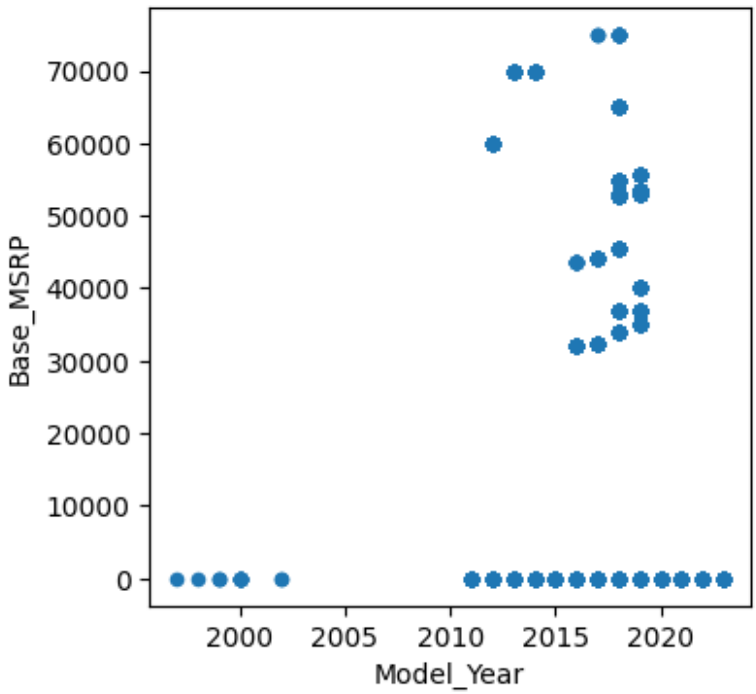
Continuous vs Continuous

```
df.plot(kind='scatter', x='Model_Year', y='Electric_Range',
figsize=(4, 4))

<Axes: xlabel='Model_Year', ylabel='Electric_Range'>
```




```
df = df.loc[df['Base_MSRP'] < 80000]

df.plot(kind='scatter', x='Model_Year', y='Base_MSRP', figsize=(4, 4))
```

```
<Axes: xlabel='Model_Year', ylabel='Base_MSRP'>
```



Discrete vs Discrete

```
discrete_df.columns
```

```
Index(['VIN_(1-10)', 'County', 'City', 'State', 'Make', 'Model',
       'Electric_Vehicle_Type', 'CAFV_Eligibility',
'Vehicle_Location',
       'Electric_Utility'],
      dtype='object')
```

```
pd.crosstab(df['Model_Year'], df['CAFV_Eligibility'], normalize =
True)
```

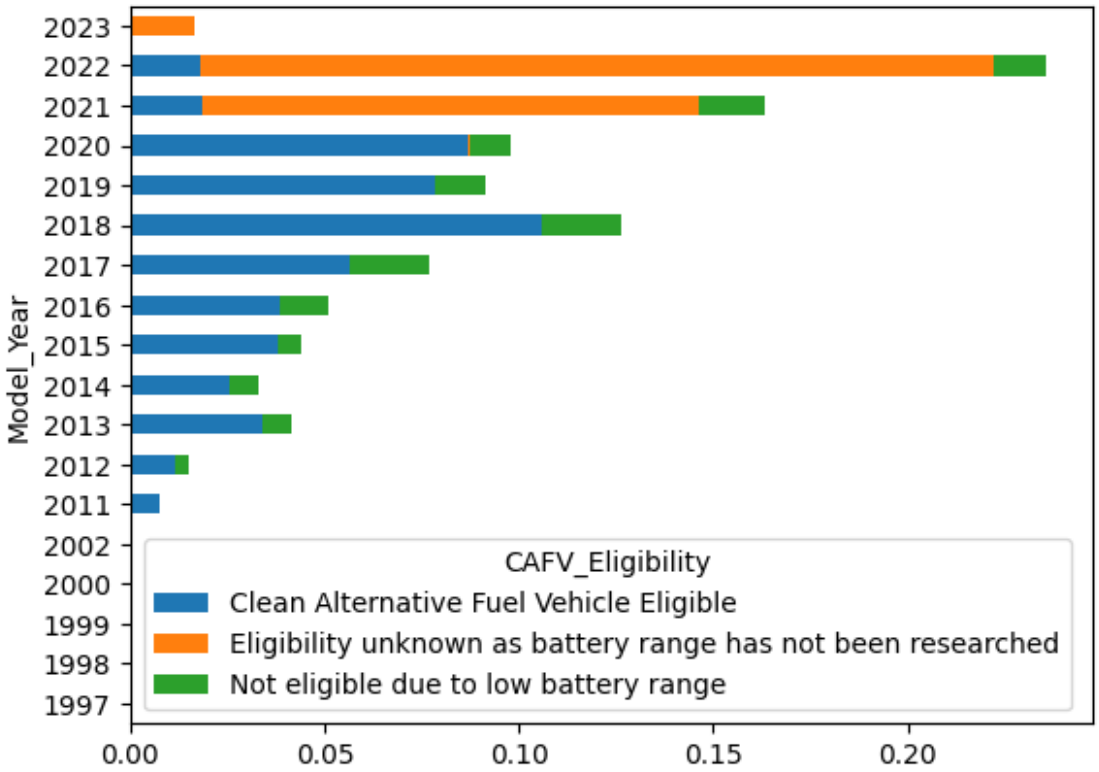
```
CAFV_Eligibility  Clean Alternative Fuel Vehicle Eligible  \
Model_Year
1997                                                0.000009
1998                                                0.000009
1999                                                0.000027
2000                                                0.000089
2002                                                0.000018
2011                                                0.007405
2012                                                0.011645
2013                                                0.034098
2014                                                0.025743
2015                                                0.037885
2016                                                0.038490
2017                                                0.056428
2018                                                0.105966
2019                                                0.078615
2020                                                0.086970
2021                                                0.018658
2022                                                0.018151
2023                                                0.000382

CAFV_Eligibility  Eligibility unknown as battery range has not been
researched  \
Model_Year

1997                                                          0.000000

1998                                                          0.000000

1999                                                          0.000000

2000                                                          0.000000

2002                                                          0.000000

2011                                                          0.000000

2012                                                          0.000000
```

```
2013                                                    0.000000
2014                                                    0.000000
2015                                                    0.000000
2016                                                    0.000000
2017                                                    0.000000
2018                                                    0.000000
2019                                                    0.000018
2020                                                    0.000471
2021                                                    0.127878
2022                                                    0.204093
2023                                                    0.016311
```

CAFV_Eligibility	Not eligible due to low battery range
Model_Year	
1997	0.000000
1998	0.000000
1999	0.000000
2000	0.000000
2002	0.000000
2011	0.000000
2012	0.003333
2013	0.007600
2014	0.007013
2015	0.006018
2016	0.012489
2017	0.020347
2018	0.020391
2019	0.012587
2020	0.010507
2021	0.016703
2022	0.013582
2023	0.000071

```
tab = pd.crosstab(df['Model_Year'], df['CAFV_Eligibility'], normalize
= True)
tab.plot(kind='barh', stacked=True)
```

```
<Axes: ylabel='Model_Year'>
```

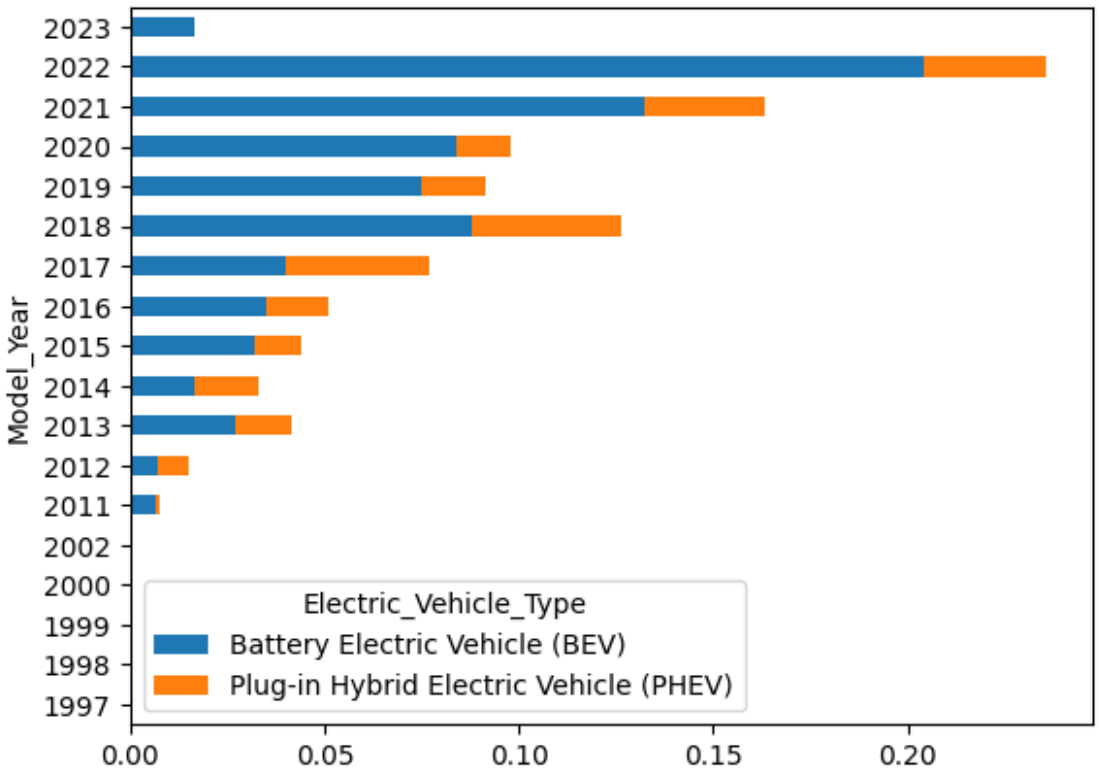



```
pd.crosstab(df['Model_Year'],df['Electric_Vehicle_Type'])

Electric_Vehicle_Type  Battery Electric Vehicle (BEV)  \
Model_Year
1997                                                1
1998                                                1
1999                                                3
2000                                               10
2002                                                2
2011                                              762
2012                                              814
2013                                             3018
2014                                             1864
2015                                             3625
2016                                             3938
2017                                             4498
2018                                             9902
2019                                             8440
2020                                             9444
2021                                            14873
2022                                            22960
2023                                             1835

Electric_Vehicle_Type  Plug-in Hybrid Electric Vehicle (PHEV)
Model_Year
```

```
1997     0
1998     0
1999     0
2000     0
2002     0
2011    71
2012   871
2013  1673
2014  1821
2015  1314
2016  1797
2017  4139
2018  4313
2019  1822
2020  1575
2021  3491
2022  3570
2023    51
```

```
tab = pd.crosstab(df['Model_Year'], df['Electric_Vehicle_Type'],
normalize = True)
tab.plot(kind='barh', stacked=True)
```

```
<Axes: ylabel='Model_Year'>
```



Discrete vs Continuous

```
group = df.groupby('Model')
group['Model_Year'].agg(['min', 'max', 'mean', 'median'])
```

```
        min   max         mean  median
Model
330E   2016  2022  2018.983498  2018.0
500    2013  2019  2015.447689  2015.0
530E   2018  2022  2018.727554  2018.0
745E   2020  2020  2020.000000  2020.0
745LE  2022  2022  2022.000000  2022.0
...     ...   ...          ...     ...
X3     2020  2021  2020.726027  2021.0
X5     2016  2022  2019.914712  2021.0
XC40   2021  2023  2021.507071  2022.0
XC60   2018  2022  2020.368946  2021.0
XC90   2016  2022  2019.485366  2020.0

[110 rows x 4 columns]
```

```
group = df.groupby('Electric_Vehicle_Type')
group['Model_Year'].agg(['min', 'max', 'mean', 'median'])
```

```
                                          min   max         mean
median
Electric_Vehicle_Type

Battery Electric Vehicle (BEV)           1997  2023  2019.356425
2020.0
Plug-in Hybrid Electric Vehicle (PHEV)   2011  2023  2017.884299
2018.0
```

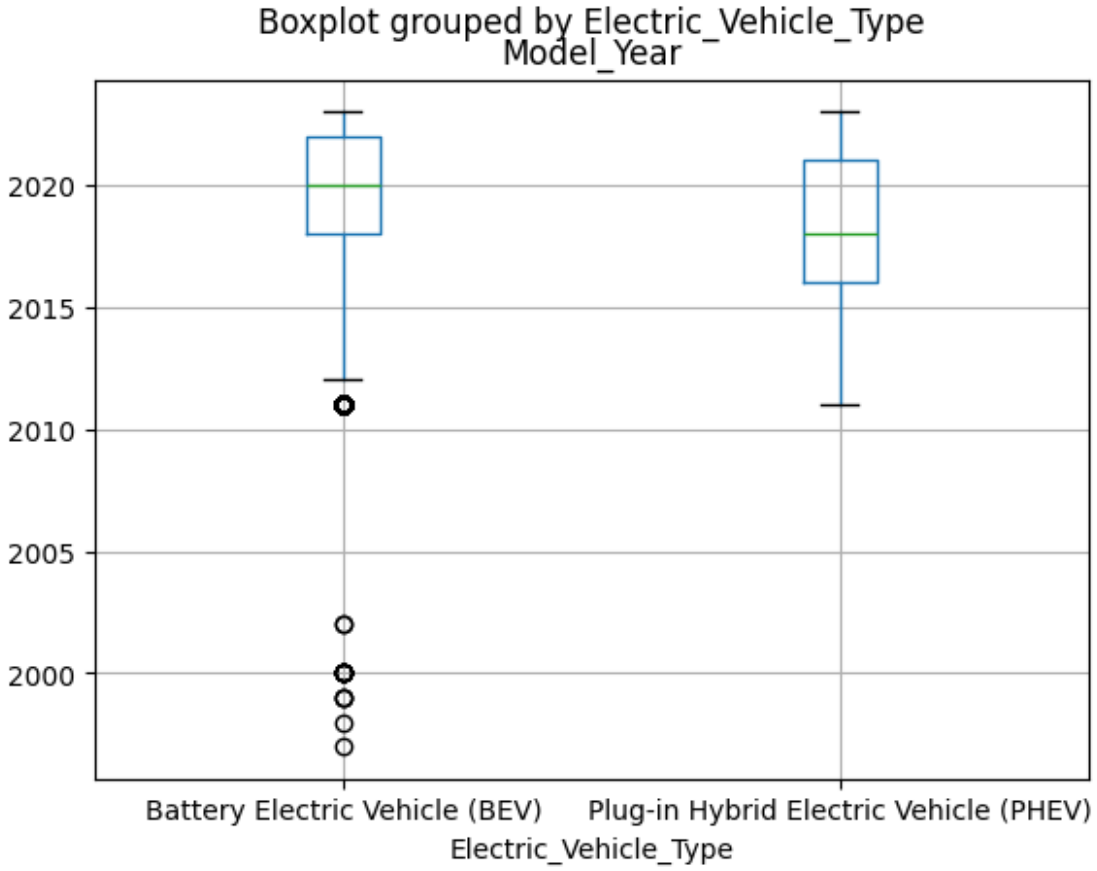
```
group = df.groupby('CAFV_Eligibility')
group['Model_Year'].agg(['min', 'max', 'mean', 'median'])
```

```
                                                    min   max
mean  \
CAFV_Eligibility

Clean Alternative Fuel Vehicle Eligible             1997  2023
2017.496850
Eligibility unknown as battery range has not be...  2019  2023
2021.677261
Not eligible due to low battery range               2012  2023
2017.915493

                                                    median
CAFV_Eligibility
Clean Alternative Fuel Vehicle Eligible             2018.0
```

```
Eligibility unknown as battery range has not be...  2022.0
Not eligible due to low battery range               2018.0
```

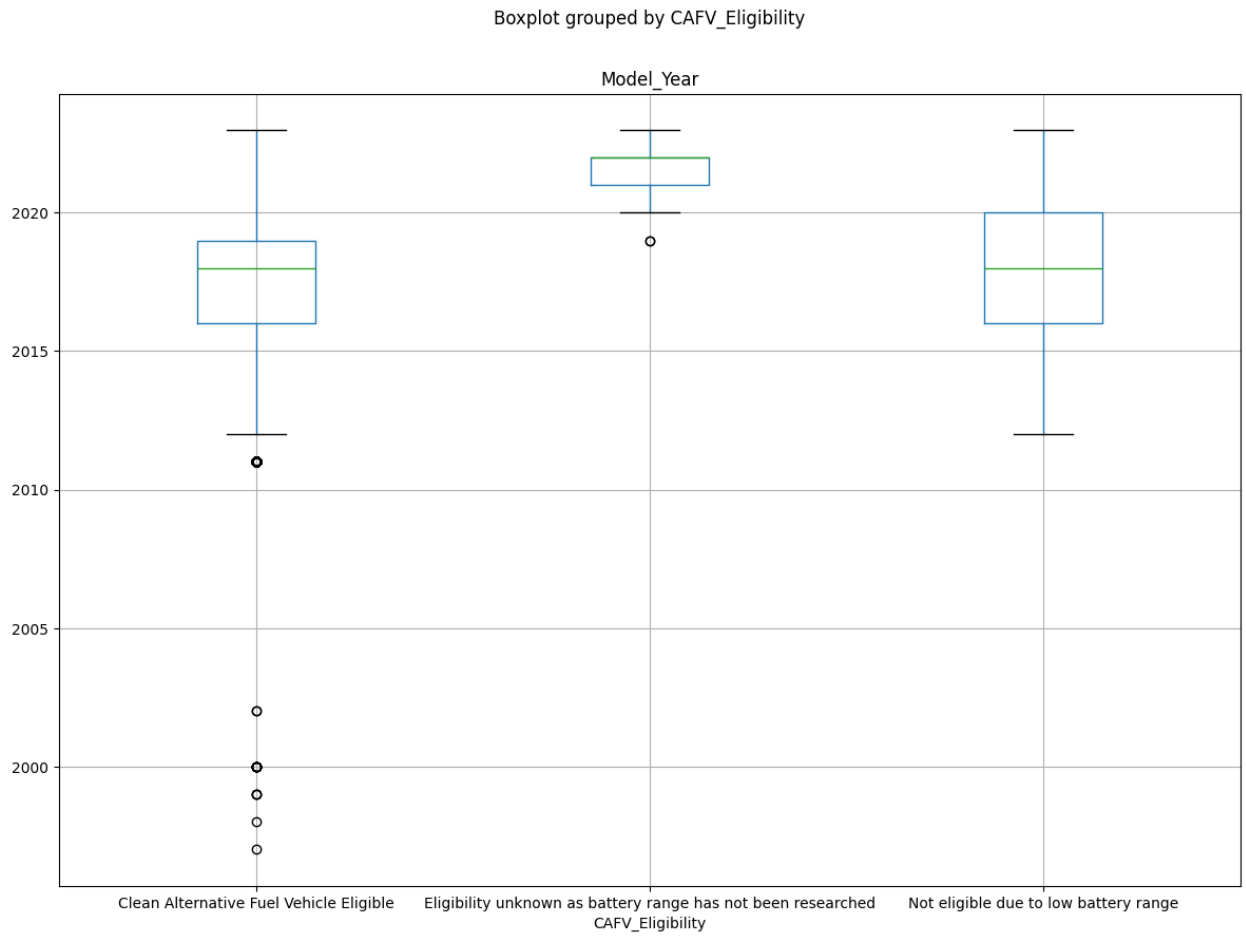
```
df.boxplot(by="Electric_Vehicle_Type",column=['Model_Year'])
```

```
<Axes: title={'center': 'Model_Year'}, xlabel='Electric_Vehicle_Type'>
```




```
df.boxplot(by="CAFV_Eligibility",column=['Model_Year'], figsize=(14,
10))
```

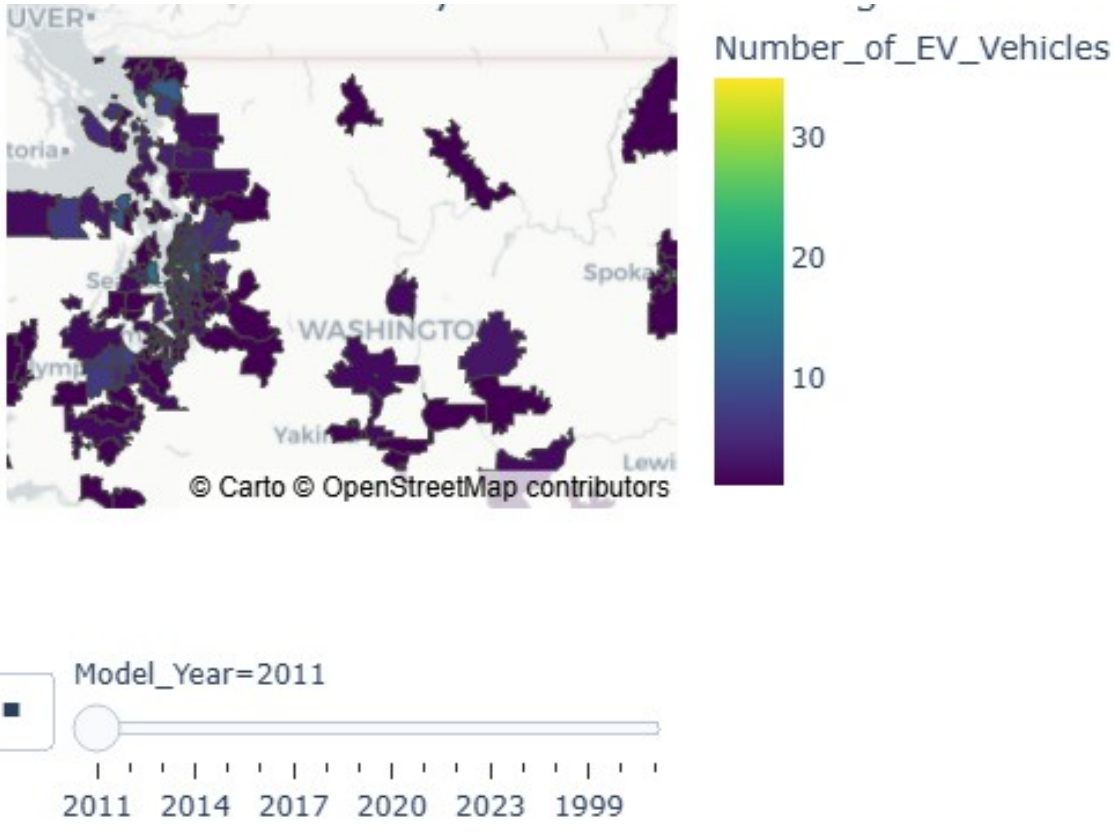
```
<Axes: title={'center': 'Model_Year'}, xlabel='CAFV_Eligibility'>
```



Task 2 : Choropleth Using Plotly.Express

```
import plotly.express as px

ev_count_by_pincode = df.groupby(['Postal_Code', 'Model_Year',
'State']).size().reset_index(name='Number_of_EV_Vehicles')

state_data = ev_count_by_pincode[ev_count_by_pincode['State'] == 'WA']

geojson_url = "https://raw.githubusercontent.com/OpenDataDE/State-zip-
code-GeoJSON/master/wa_washington_zip_codes_geo.min.json"

fig = px.choropleth_mapbox(state_data,
                           geojson=geojson_url,
                           locations='Postal_Code',
                           color='Number_of_EV_Vehicles',
                           featureidkey="properties.ZCTA5CE10",  #
Make sure this matches the ZIP code field in the geojson
                           mapbox_style="carto-positron",
                           zoom=5,  # Adjust zoom level
                           center={"lat": 47.7511, "lon": -120.7401},
```

```
# Centering on Washington
                            title="Number of EV Vehicles by ZIP Code in
Washington Over Time",
                            animation_frame="Model_Year",  # Animated
by year
                            color_continuous_scale="Viridis",
                            hover_data=['Number_of_EV_Vehicles']  #
Hover info
                           )

# Update layout for better fit and aesthetics
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})

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



Task 3 : Racing Bar

```
!pip install bar_chart_race

Requirement already satisfied: bar_chart_race in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (0.1.0)
Requirement already satisfied: pandas>=0.24 in c:\users\payal\appdata\
```

```
local\programs\python\python311\lib\site-packages (from
bar_chart_race) (2.2.2)
Requirement already satisfied: matplotlib>=3.1 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
bar_chart_race) (3.9.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\payal\appdata\
local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (4.53.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (1.4.5)
Requirement already satisfied: numpy>=1.23 in c:\users\payal\appdata\
local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (1.26.3)
Requirement already satisfied: packaging>=20.0 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (23.1)
Requirement already satisfied: pillow>=8 in c:\users\payal\appdata\
local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (10.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
matplotlib>=3.1->bar_chart_race) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\payal\appdata\
local\programs\python\python311\lib\site-packages (from pandas>=0.24-
>bar_chart_race) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\payal\
appdata\local\programs\python\python311\lib\site-packages (from
pandas>=0.24->bar_chart_race) (2024.1)
Requirement already satisfied: six>=1.5 in c:\users\payal\appdata\
local\programs\python\python311\lib\site-packages (from python-
dateutil>=2.7->matplotlib>=3.1->bar_chart_race) (1.16.0)


[notice] A new release of pip is available: 23.2.1 -> 24.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
import bar_chart_race as bcr
```

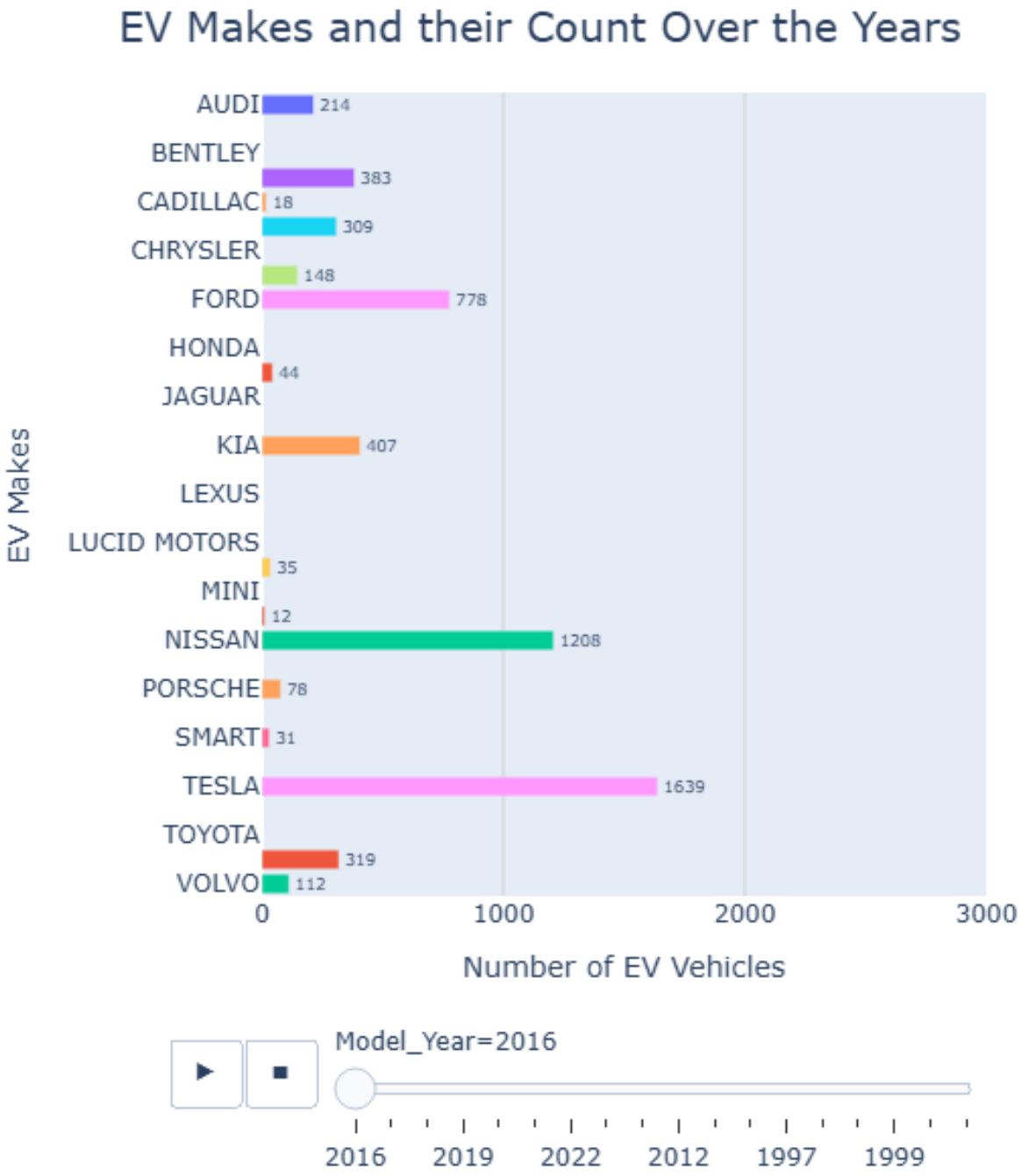
```
df = df.groupby(['Make',
'Model_Year']).size().reset_index(name='Number_of_Vehicles')
```

```python
# Display the resulting DataFrame for verification
print(df)
```

```
      Make  Model_Year  Number_of_Vehicles
0     AUDI        2016                 214
1     AUDI        2017                 187
2     AUDI        2018                 174
3     AUDI        2019                 392
4     AUDI        2020                 224
..     ...         ...                 ...
200  VOLVO        2019                 190
201  VOLVO        2020                 162
202  VOLVO        2021                 580
203  VOLVO        2022                 882
204  VOLVO        2023                  21

[205 rows x 3 columns]
```

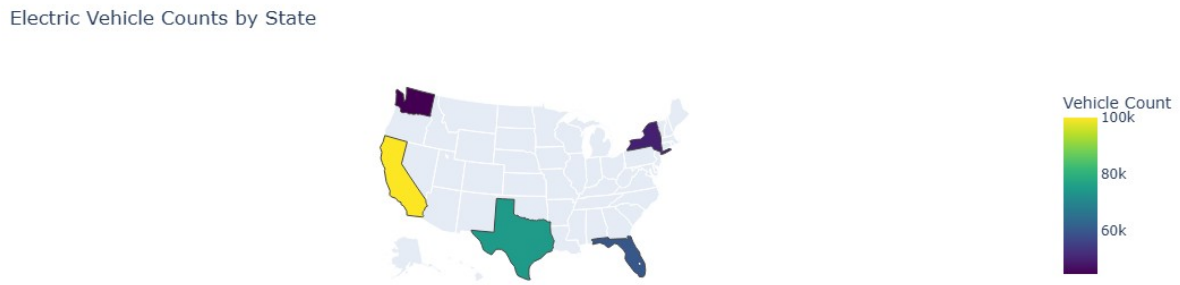
```python
# Create the animated racing bar plot with annotations
fig = px.bar(df,
             y='Make',  # Place Make on y-axis
             x='Number_of_Vehicles',  # Place the count of EV vehicles
on the x-axis
             color='Make',  # Color each make differently
             animation_frame='Model_Year',  # Create animation by year
             orientation='h',  # Horizontal bar chart
             title='EV Makes and their Count Over the Years',
             labels={'Number_of_Vehicles': 'Number of EV Vehicles'},
             range_x=[0, 3000]
            )

# Update traces for displaying values
fig.update_traces(texttemplate='%{x}',  # Display the actual x-axis
values (Number_of_Vehicles)
                  textposition='outside',  # Place the text outside
the bars
                  textfont_size=16)  # Adjust the font size for better
readability

# Adjust the layout for improved visibility and emphasis on movement
fig.update_layout(
    xaxis=dict(showgrid=True, gridcolor='LightGray'),  # Show grid for
better visibility
    yaxis_title='EV Makes',
    xaxis_title='Number of EV Vehicles',
    showlegend=False,  # Hide legend as it's not necessary for this
chart
    title_x=0.5,  # Center title
    title_font=dict(size=20),  # Increase title font size
```

```
    margin=dict(l=50, r=50, t=50, b=50),  # Adjust margins
    width=800,  # Set a fixed width
    height=600  # Set a fixed height
)

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



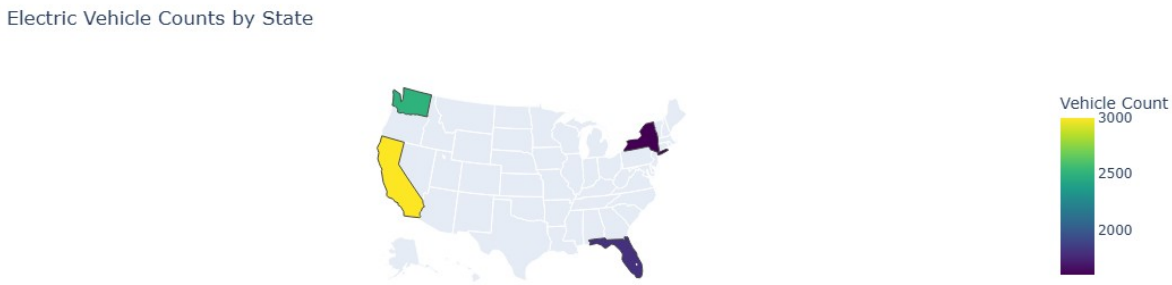
Different Visualization

```
# Example data for state counts
data = {
    'State': ['CA', 'TX', 'NY', 'FL', 'WA'],
    'Vehicle Count': [100000, 75000, 50000, 60000, 45000]
}

# Create a DataFrame from the data
state_counts = pd.DataFrame(data)

import plotly.express as px

# Creating the choropleth map for Task 2 using Plotly
fig2 = px.choropleth(state_counts,
                     locations='State',
                     locationmode="USA-states",
                     color='Vehicle Count',
                     color_continuous_scale="Viridis",
                     scope="usa",
                     title="Electric Vehicle Counts by State")

fig2.show()
```




```
import plotly.express as px

# Replace with your actual dataset
state_counts = pd.DataFrame({
    'State': ['CA', 'WA', 'FL', 'NY'],  # Example states
    'Vehicle Count': [3000, 2500, 1800, 1600]  # Example counts
})

fig = px.choropleth(state_counts,
                    locations='State',
                    locationmode="USA-states",
                    color='Vehicle Count',
                    color_continuous_scale="Viridis",
```

```
                    scope="usa",
                    title="Electric Vehicle Counts by State")
fig.show()
```

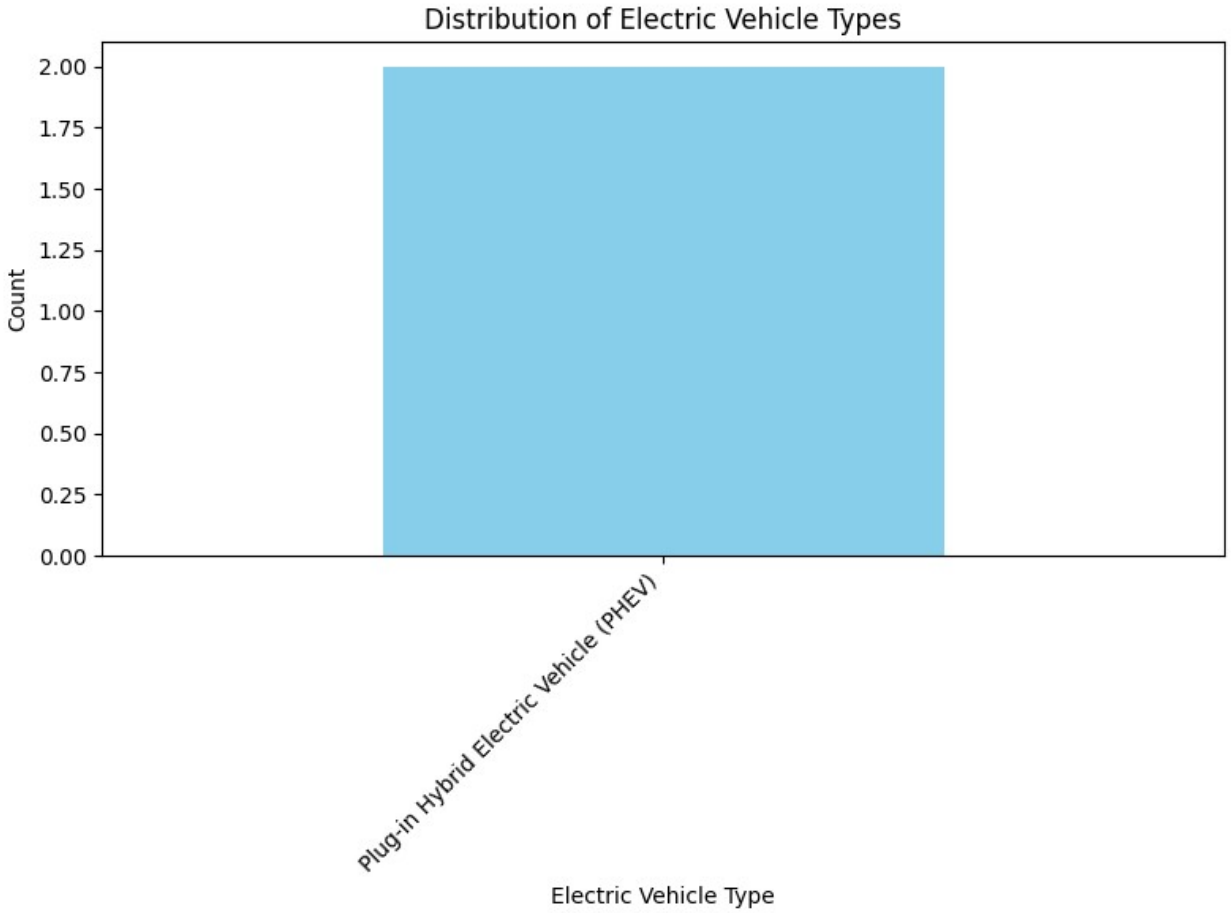



```
df.columns
```

```
Index(['Make', 'Model_Year', 'Number_of_Vehicles'], dtype='object')
```

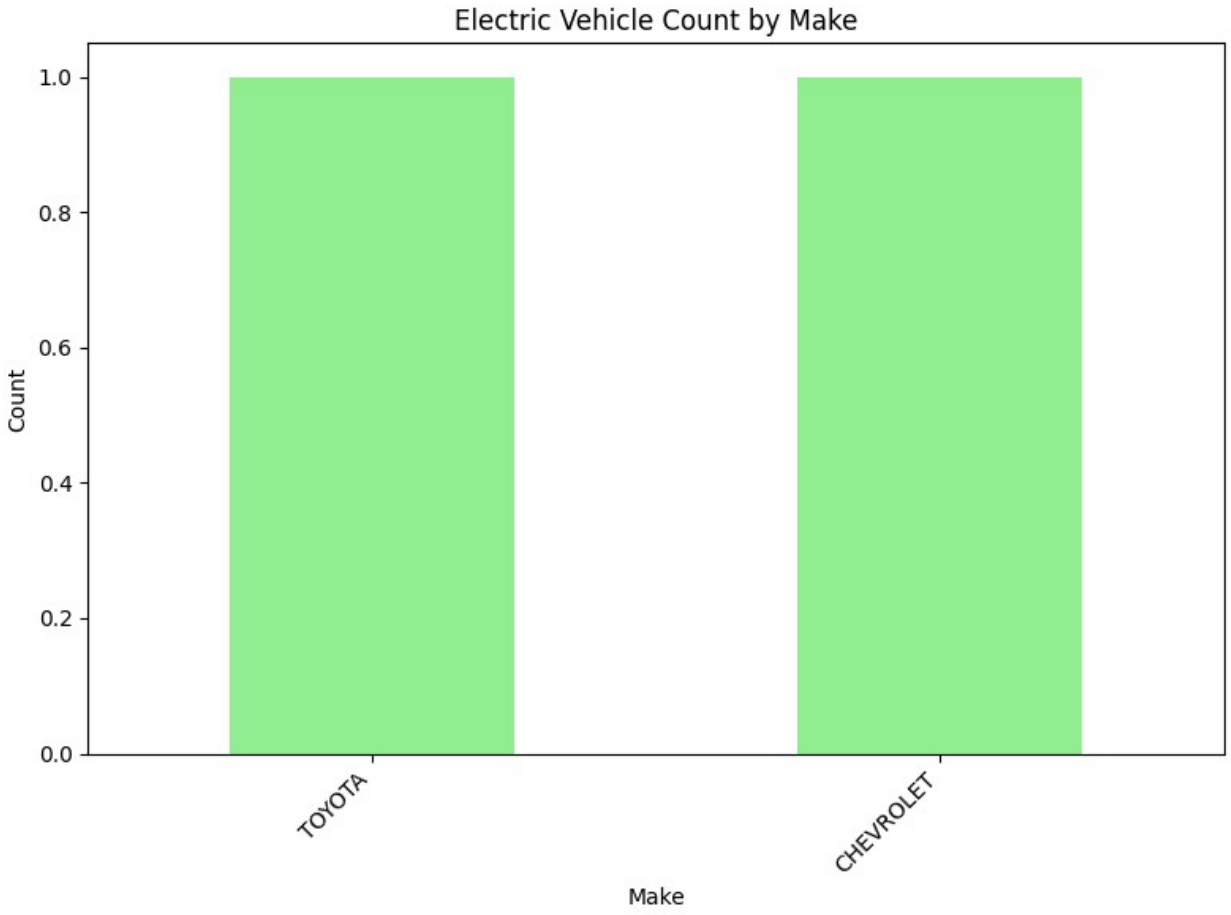
```
# Creating a sample DataFrame based on the structure identified from
the notebook cells
data = {
    "VIN (1-10)": ["JTMEB3FV6N", "1G1RD6E45D"],
    "County": ["Monroe", "Clark"],
    "City": ["Key West", "Laughlin"],
    "State": ["FL", "NV"],
    "Postal Code": [33040, 89029],
    "Model Year": [2022, 2013],
    "Make": ["TOYOTA", "CHEVROLET"],
    "Model": ["RAV4 PRIME", "VOLT"],
    "Electric Vehicle Type": ["Plug-in Hybrid Electric Vehicle
(PHEV)", "Plug-in Hybrid Electric Vehicle (PHEV)"],
    "Clean Alternative Fuel Vehicle (CAFV) Eligibility": ["Eligible",
"Eligible"],
    "Electric Range": [42, 38],
    "Base MSRP": [0, 0],
    "Legislative District": [None, None],
    "DOL Vehicle ID": [198968248, None],
    "Vehicle Location": ["POINT (-81.80023 24.5545)", None],
    "Electric Utility": [None, None],
    "2020 Census Tract": [12087972100, None]
}

# Create a DataFrame using the above data for demonstration purposes
df = pd.DataFrame(data)

# Visualizing the Electric Vehicle Types Distribution
vehicle_type_counts = df['Electric Vehicle Type'].value_counts()
```

```
# Plotting the distribution of Electric Vehicle Types
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
vehicle_type_counts.plot(kind='bar', color='skyblue')
plt.title('Distribution of Electric Vehicle Types')
plt.xlabel('Electric Vehicle Type')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

plt.show()
```

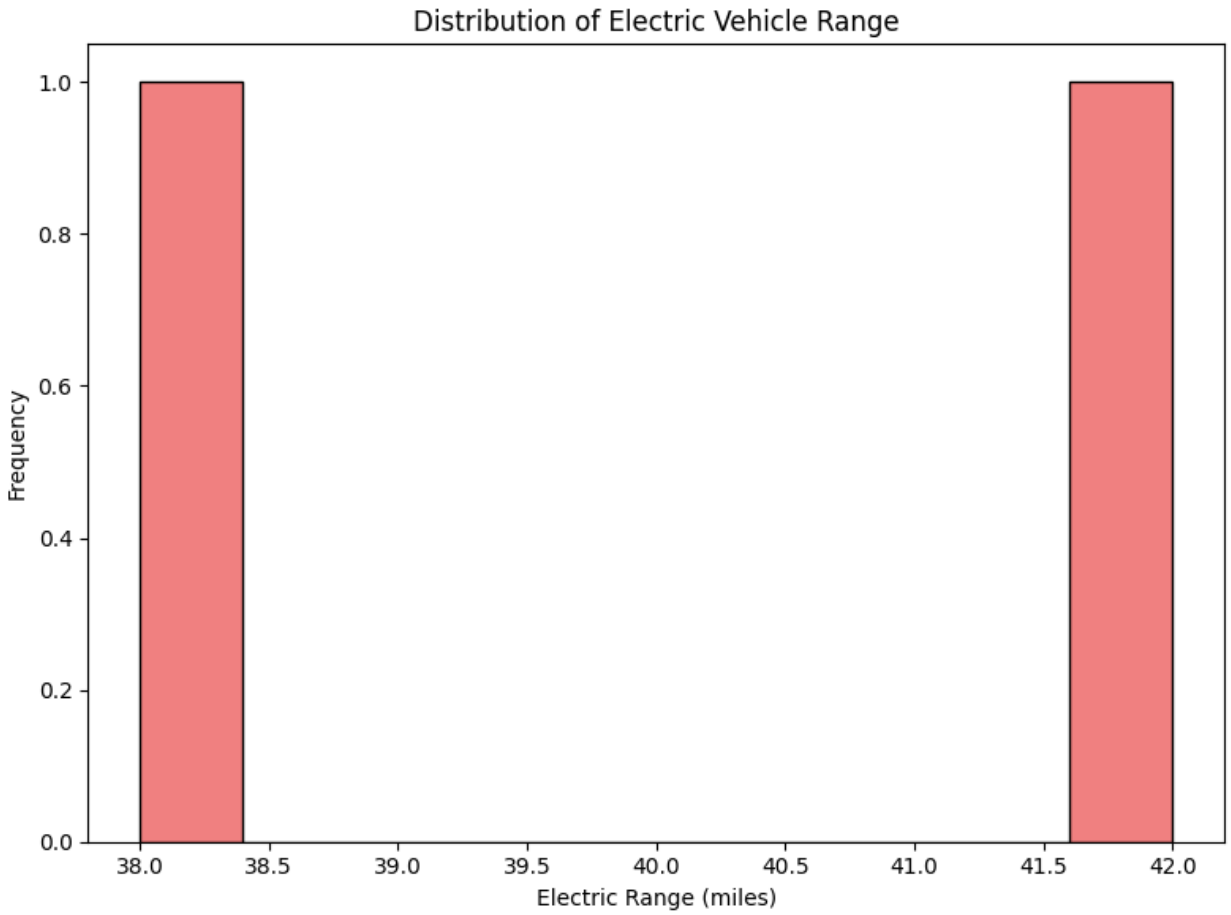



```
# Visualizing the Electric Vehicle Count by Make
make_counts = df['Make'].value_counts()

# Plotting the Electric Vehicle Count by Make
plt.figure(figsize=(8, 6))
make_counts.plot(kind='bar', color='lightgreen')
plt.title('Electric Vehicle Count by Make')
```

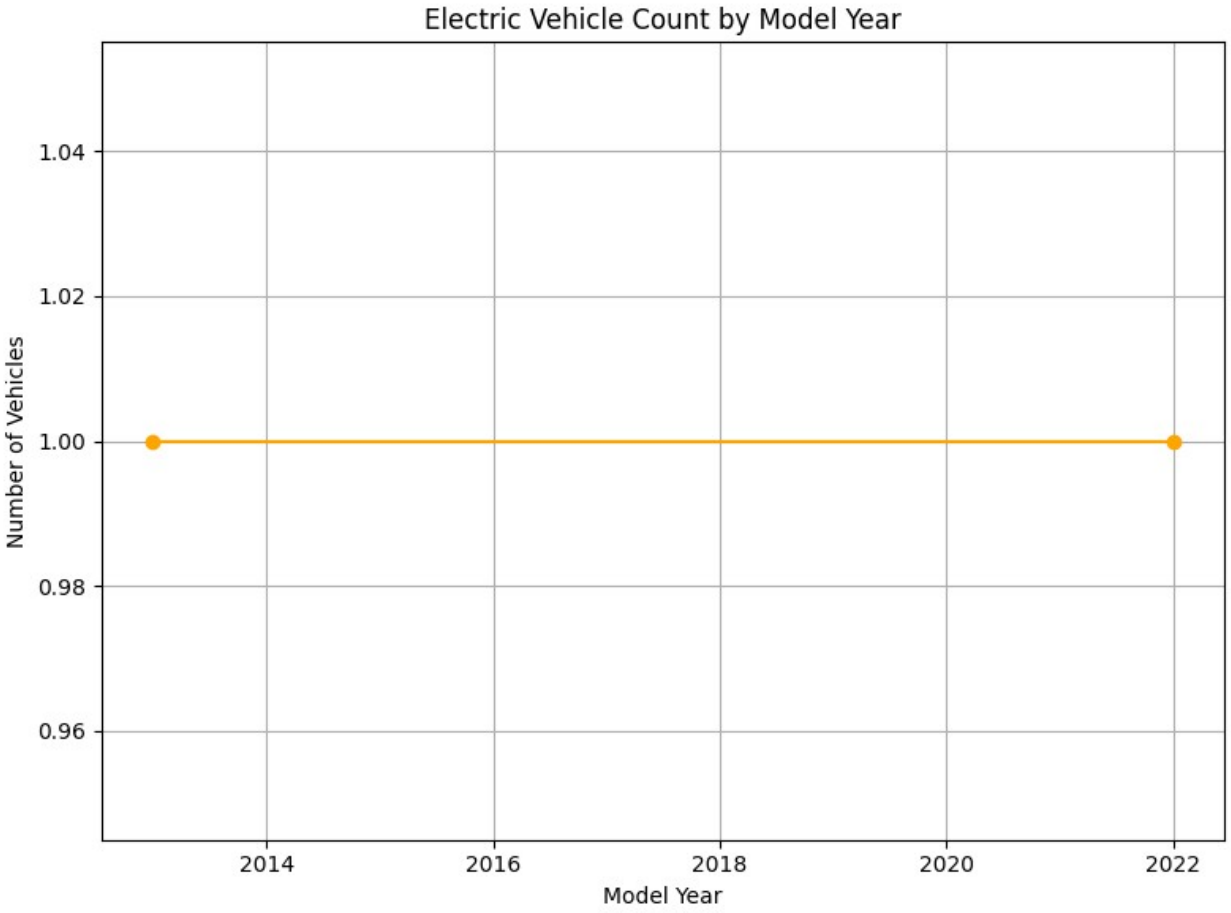
```
plt.xlabel('Make')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

plt.show()
```




```
# Visualizing the Electric Range Distribution
plt.figure(figsize=(8, 6))
plt.hist(df['Electric Range'], bins=10, color='lightcoral',
edgecolor='black')
plt.title('Distribution of Electric Vehicle Range')
plt.xlabel('Electric Range (miles)')
plt.ylabel('Frequency')
plt.tight_layout()

plt.show()
```




```
# Visualizing the trend of Electric Vehicles by Model Year
model_year_counts = df['Model Year'].value_counts().sort_index()

# Plotting the Model Year trends
plt.figure(figsize=(8, 6))
model_year_counts.plot(kind='line', marker='o', color='orange')
plt.title('Electric Vehicle Count by Model Year')
plt.xlabel('Model Year')
plt.ylabel('Number of Vehicles')
plt.grid(True)
plt.tight_layout()

plt.show()
```


Electric Vehicle Count by Model Year
1.04
1.02
1.00
0.98
0.96
Number of Vehicles
2014
2016
2018
2020
2022
Model Year