In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
!pip install bioinfokit
import plotly.express as px
```

```
Defaulting to user installation because normal site-packages is not write
able
Requirement already satisfied: bioinfokit in c:\users\hp\appdata\roaming
\python\python39\site-packages (2.1.0)
Requirement already satisfied: tabulate in c:\programdata\anaconda3\lib\s
ite-packages (from bioinfokit) (0.8.9)
Requirement already satisfied: matplotlib-venn in c:\users\hp\appdata\roa
ming\python\python39\site-packages (from bioinfokit) (0.11.9)
Requirement already satisfied: scikit-learn in c:\programdata\anaconda3\l
ib\site-packages (from bioinfokit) (1.0.2)
Requirement already satisfied: statsmodels in c:\programdata\anaconda3\li
b\site-packages (from bioinfokit) (0.13.2)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib
\site-packages (from bioinfokit) (3.5.1)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\sit
e-packages (from bioinfokit) (1.4.2)
Requirement already satisfied: textwrap3 in c:\users\hp\appdata\roaming\p
ython\python39\site-packages (from bioinfokit) (0.9.2)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site
-packages (from bioinfokit) (1.21.5)
Requirement already satisfied: adjustText in c:\users\hp\appdata\roaming
\python\python39\site-packages (from bioinfokit) (0.8)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site
-packages (from bioinfokit) (1.7.3)
Requirement already satisfied: seaborn in c:\programdata\anaconda3\lib\si
te-packages (from bioinfokit) (0.11.2)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\l
ib\site-packages (from matplotlib->bioinfokit) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anacon
da3\lib\site-packages (from matplotlib->bioinfokit) (4.25.0)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda
3\lib\site-packages (from matplotlib->bioinfokit) (21.3)
Requirement already satisfied: pyparsing>=2.2.1 in c:\programdata\anacond
a3\lib\site-packages (from matplotlib->bioinfokit) (3.0.4)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\ana
conda3\lib\site-packages (from matplotlib->bioinfokit) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3
\lib\site-packages (from matplotlib->bioinfokit) (9.0.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anacon
da3\lib\site-packages (from matplotlib->bioinfokit) (1.3.2)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\s
ite-packages (from python-dateutil>=2.7->matplotlib->bioinfokit) (1.16.0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\l
ib\site-packages (from pandas->bioinfokit) (2021.3)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\ana
conda3\lib\site-packages (from scikit-learn->bioinfokit) (2.2.0)
Requirement already satisfied: joblib>=0.11 in c:\users\hp\appdata\roamin
g\python\python39\site-packages (from scikit-learn->bioinfokit) (1.2.0)
Requirement already satisfied: patsy>=0.5.2 in c:\programdata\anaconda3\l
ib\site-packages (from statsmodels->bioinfokit) (0.5.2)
```

In [2]:

```
df = pd.read_csv("C:\\Users\\HP\\Downloads\\Electric_Vehicle_Population_Data.csv")
df.head()
```

Out[2]:

	VIN (1-10)	County	City	State	ZIP Code	Model Year	Make	Model	Electric Vehicle Type	Al
0	WA1AAAGE2M	Kitsap	POULSBO	WA	98370	2021	AUDI	E-TRON	Battery Electric Vehicle (BEV)	A
1	WBY8P2C00L	King	SEATTLE	WA	98122	2020	BMW	I3	Battery Electric Vehicle (BEV)	A
2	5YJXCBE21K	Cowlitz	SILVERLAKE	WA	98645	2019	TESLA	MODEL X	Battery Electric Vehicle (BEV)	A
3	1FTZR081XY	King	SEATTLE	WA	98117	2000	FORD	RANGER	Battery Electric Vehicle (BEV)	A
4	WBY1Z6C55H	King	SEATTLE	WA	98119	2017	BMW	I3	Battery Electric Vehicle (BEV)	A

In [3]:

```
df.shape
```

Out[3]:

```
(79767, 15)
```

In [4]:

```
df.isna().sum()
```

Out[4]:

```
VIN (1-10)                                             0
County                                                 5
City                                                   0
State                                                  0
ZIP Code                                               0
Model Year                                             0
Make                                                   0
Model                                                  0
Electric Vehicle Type                                  0
Clean Alternative Fuel Vehicle (CAFV) Eligibility      0
Electric Range                                         0
Base MSRP                                              0
Legislative District                                 146
DOL Vehicle ID                                         0
Vehicle Location                                       4
dtype: int64
```

In [5]:

```
df['Legislative District'].unique()
```

Out[5]:

```
array([23., 37., 20., 36., 45., 47., 11., 48.,  2., 18., 32., 41., 42.,
       46., 33., 34.,  6., 24., 35., 31., 28., 38.,  1.,  7.,  5., 19.,
       12., 40., 27., 25., 21.,  3., 49., 22., 17., 44., 10., 39., 13.,
       43.,  8., 14., 26.,  4., 16., 30., 29., 15.,  9., nan])
```

In [6]:

```
df['County'].unique()
```

Out[6]:

```
array(['Kitsap', 'King', 'Cowlitz', 'Thurston', 'Clark', 'Whatcom',
       'Spokane', 'Clallam', 'Pierce', 'Snohomish', 'Grays Harbor',
       'Chelan', 'San Juan', 'Mason', 'Island', 'Skagit', 'Lincoln',
       'Benton', 'Yakima', 'Grant', 'Lewis', 'Jefferson', 'Kittitas',
       'Okanogan', 'Wahkiakum', 'Franklin', 'Adams', 'Walla Walla',
       'Douglas', 'Skamania', 'Klickitat', 'Bell', 'Whitman', 'Arlingto
n',
       'Frederick', 'Stevens', 'Asotin', 'Nassau', 'Pacific', 'San Dieg
o',
       'Pend Oreille', 'Montgomery', 'Sonoma', 'Liberty', 'Garfield',
       'Rockingham', 'Otero', 'District Of Columbia', 'Santa Barbara',
       'Los Angeles', 'Camden', 'Multnomah', 'Bradley', 'Muscogee',
       'Columbia', 'Placer', 'Anne Arundel', 'Alexandria City',
       'Monterey', 'Chaves', 'Ferry', nan, 'Chesapeake City', 'Fairfax',
       'Norfolk City', 'New Castle', 'Wilson', 'Charles', 'Honolulu',
       'Saint Clair', 'Alameda', 'Bexar', 'Lake', 'Fresno', 'Riverside',
       'Suffolk City', 'Contra Costa', 'Santa Clara', 'El Paso',
       'New London', 'Glacier', 'Orange', 'Harrison', 'Okaloosa', 'Brya
n',
       'Tulare', 'Prince William', 'Dekalb', 'Dorchester', 'Saint Marys',
       'Saginaw', 'Newport', 'Klamath', 'Shelby', 'Ventura',
       'Leavenworth', 'Howard', 'Riley', 'Sacramento', 'Oldham',
       'Stafford', 'Goochland', 'Meade', 'San Francisco', 'Bartow',
       'Maricopa', 'Moore', 'Kent', 'Cumberland', 'Ozaukee', 'Passaic',
       'Middlesex', 'Saint Louis', 'Caddo'], dtype=object)
```

In [7]:

```
df['Vehicle Location'].unique()
```

Out[7]:

```
array(['POINT (-122.63339300000001 47.748427)',
       'POINT (-122.303413 47.61065)', 'POINT (-122.772699 46.32052
6)',
       'POINT (-122.379354 47.687571)',
       'POINT (-122.36772100000002 47.639264)',
       'POINT (-122.04272399999999 47.623594)',
       'POINT (-122.266685 47.308313)', 'POINT (-122.204248 47.71927
8)',
       'POINT (-122.276826 47.449726)', 'POINT (-122.188994 47.67840
6)',
       'POINT (-122.558621 46.888349)',
       'POINT (-122.40849800000001 45.620943)',
       'POINT (-122.37015900000002 47.743354)',
       'POINT (-122.168422 47.614824)', 'POINT (-122.151342 47.56019
2)',
       'POINT (-122.33024199999998 48.904379)',
       'POINT (-122.297534 47.685291)', 'POINT (-122.028168 47.58617
```

In [8]:

```
df['Legislative District'].max()
```

Out[8]:

```
49.0
```

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 79767 entries, 0 to 79766
Data columns (total 15 columns):
 #   Column                                             Non-Null Count  D
type
---  ------                                             --------------  -
----
 0   VIN (1-10)                                         79767 non-null  o
bject
 1   County                                             79762 non-null  o
bject
 2   City                                               79767 non-null  o
bject
 3   State                                              79767 non-null  o
bject
 4   ZIP Code                                           79767 non-null  i
nt64
 5   Model Year                                         79767 non-null  i
nt64
 6   Make                                               79767 non-null  o
bject
 7   Model                                              79767 non-null  o
bject
 8   Electric Vehicle Type                              79767 non-null  o
bject
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  79767 non-null  o
bject
 10  Electric Range                                     79767 non-null  i
nt64
 11  Base MSRP                                          79767 non-null  i
nt64
 12  Legislative District                               79621 non-null  f
loat64
 13  DOL Vehicle ID                                     79767 non-null  i
nt64
 14  Vehicle Location                                   79763 non-null  o
bject
dtypes: float64(1), int64(5), object(9)
memory usage: 9.1+ MB
```

In [10]:

```
df.columns
```

Out[10]:

```
Index(['VIN (1-10)', 'County', 'City', 'State', 'ZIP Code', 'Model Year',
       'Make', 'Model', 'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Electric Ran
ge',
       'Base MSRP', 'Legislative District', 'DOL Vehicle ID',
       'Vehicle Location'],
      dtype='object')
```

In [11]:

```
df.dtypes
```

Out[11]:

```
VIN (1-10)                                            object
County                                                object
City                                                  object
State                                                 object
ZIP Code                                               int64
Model Year                                             int64
Make                                                  object
Model                                                 object
Electric Vehicle Type                                 object
Clean Alternative Fuel Vehicle (CAFV) Eligibility     object
Electric Range                                         int64
Base MSRP                                              int64
Legislative District                                 float64
DOL Vehicle ID                                         int64
Vehicle Location                                      object
dtype: object
```

In [12]:

```
df.isnull()
```

Out[12]:

	VIN (1-10)	County	City	State	ZIP Code	Model Year	Make	Model	Electric Vehicle Type	Clean Alternative Fuel Vehicle (CAFV) Eligibility	Electri Rang
0	False	False	False	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	...	.
79762	False	False	False	False	False	False	False	False	False	False	Fals
79763	False	False	False	False	False	False	False	False	False	False	Fals
79764	False	False	False	False	False	False	False	False	False	False	Fals
79765	False	False	False	False	False	False	False	False	False	False	Fals
79766	False	False	False	False	False	False	False	False	False	False	Fals

79767 rows × 15 columns

In [13]:

```
df.isnull().sum()
```

Out[13]:

```
VIN (1-10)                                             0
County                                                 5
City                                                   0
State                                                  0
ZIP Code                                               0
Model Year                                             0
Make                                                   0
Model                                                  0
Electric Vehicle Type                                  0
Clean Alternative Fuel Vehicle (CAFV) Eligibility      0
Electric Range                                         0
Base MSRP                                              0
Legislative District                                 146
DOL Vehicle ID                                         0
Vehicle Location                                       4
dtype: int64
```

In [14]:

```
df.dropna
```

Out[14]:

```
<bound method DataFrame.dropna of        VIN (1-10)   County        City 
State  ZIP Code  Model Year  \
0      WA1AAAGE2M   Kitsap     POULSBO    WA     98370        2021   
1      WBY8P2C00L     King     SEATTLE    WA     98122        2020   
2      5YJXCBE21K  Cowlitz  SILVERLAKE    WA     98645        2019   
3      1FTZR081XY     King     SEATTLE    WA     98117        2000   
4      WBY1Z6C55H     King     SEATTLE    WA     98119        2017   
...           ...      ...         ...   ...      ...         ...   
79762  JA4J24A5XJ     King     SEATTLE    WA     98115        2018   
79763  1G1FZ6S07L  Spokane   DEER PARK    WA     99006        2020   
79764  5YJYGDEE4L     King     SEATTLE    WA     98112        2020   
79765  1G1FZ6S06L   Pierce    LAKEWOOD    WA     98498        2020   
79766  5YJ3E1EB9J  Spokane     SPOKANE    WA     99223        2018   

             Make      Model           Electric Vehicle Type  \
0            AUDI     E-TRON  Battery Electric Vehicle (BEV)   
1             BMW         I3  Battery Electric Vehicle (BEV)   
2           TESLA    MODEL X  Battery Electric Vehicle (BEV)   
3            FORD     RANGER  Battery Electric Vehicle (BEV)   
4             BMW         I3  Battery Electric Vehicle (BEV)   
...           ...        ...                             ...   
79762  MITSUBISHI  OUTLANDER  Battery Electric Vehicle (BEV)   
79763   CHEVROLET    BOLT EV  Battery Electric Vehicle (BEV)   
79764       TESLA    MODEL Y  Battery Electric Vehicle (BEV)   
79765   CHEVROLET    BOLT EV  Battery Electric Vehicle (BEV)   
79766       TESLA    MODEL 3  Battery Electric Vehicle (BEV)   

      Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  
\
0               Clean Alternative Fuel Vehicle Eligible             222   
1               Clean Alternative Fuel Vehicle Eligible             153   
2               Clean Alternative Fuel Vehicle Eligible             289   
3               Clean Alternative Fuel Vehicle Eligible              58   
4               Clean Alternative Fuel Vehicle Eligible              81   
...                                                 ...             ...   
79762           Not eligible due to low battery range              22   
79763           Clean Alternative Fuel Vehicle Eligible             259   
79764           Clean Alternative Fuel Vehicle Eligible             291   
79765           Clean Alternative Fuel Vehicle Eligible             259   
79766           Clean Alternative Fuel Vehicle Eligible             215   

       Base MSRP  Legislative District  DOL Vehicle ID  \
0              0                  23.0       148815901   
1              0                  37.0       132197810   
2              0                  20.0       154341673   
3              0                  36.0       169378338   
4              0                  36.0       192605101   
...          ...                   ...             ...   
79762          0                  46.0       476074686   
79763          0                   7.0       127165531   
79764          0                  43.0       127108670   
79765          0                  28.0       141902162   
79766          0                   6.0       328614947   

                            Vehicle Location  
0      POINT (-122.63339300000001 47.748427)  
1               POINT (-122.303413 47.61065)  
2              POINT (-122.772699 46.320526)  
3              POINT (-122.379354 47.687571)  
4      POINT (-122.36772100000002 47.639264)  
...                                      ...  
```

In [15]:

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

In [16]:

```
df.isnull().sum()
```

Out[16]:

```
VIN (1-10)                                           0
County                                               0
City                                                 0
State                                                0
ZIP Code                                             0
Model Year                                           0
Make                                                 0
Model                                                0
Electric Vehicle Type                                0
Clean Alternative Fuel Vehicle (CAFV) Eligibility    0
Electric Range                                       0
Base MSRP                                            0
Legislative District                                 0
DOL Vehicle ID                                       0
Vehicle Location                                     0
dtype: int64
```

EXPLORATORY DATA ANALYSIS

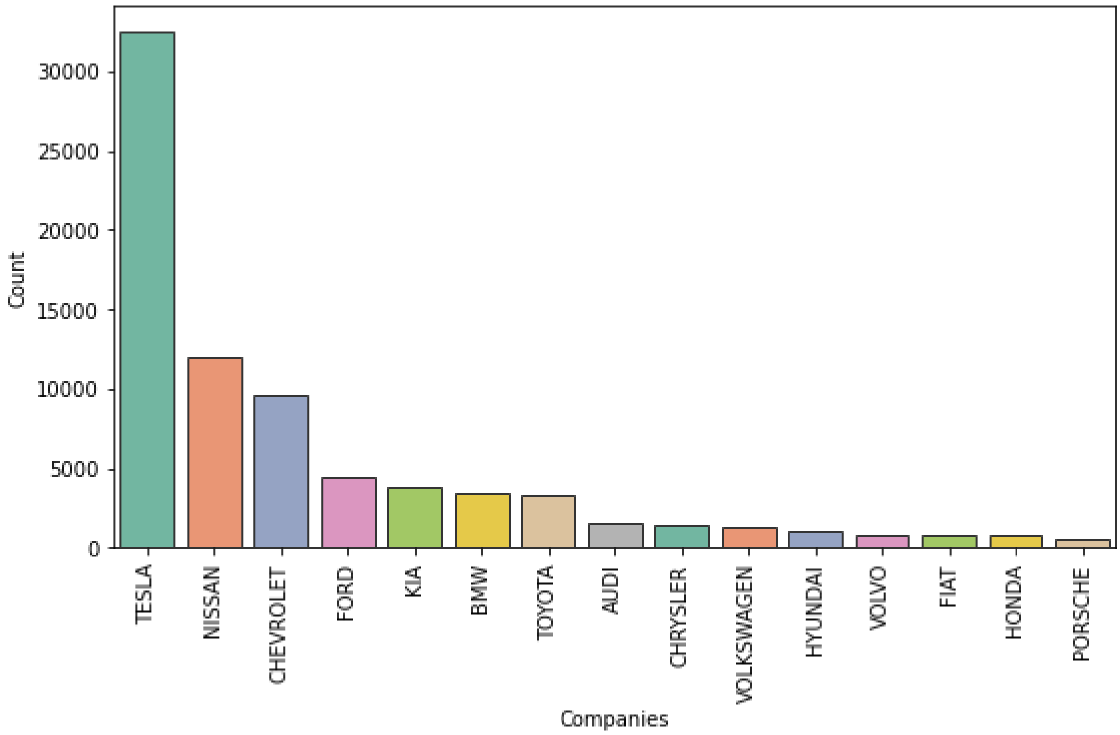
```
79762          POINT (-122.297534 47.685291)
79763          POINT (-117.481417 47.949511)
79764          POINT (-122.296466 47.631708)
79765           POINT (-122.55645 47.162344)
79766          POINT (-117.352803 47.604595)

[79767 rows x 15 columns]>
```

In [17]:

```
Counties = df.groupby('County').count().sort_values(by='City',ascending=False)['City'].i
values = df.groupby('County').count().sort_values(by='City',ascending=False)['City'].val

px.bar(x=list(Counties)[:15],y=values[:15],labels={'x':"County Name",'y':"Number of Cars
```

In [18]:

```
Companies = df.groupby('Make').count().sort_values(by='City',ascending=False)['City'].ir
values = df.groupby('Make').count().sort_values(by='City',ascending=False)['City'].value


plt.figure(figsize=(9,5))
sns.barplot(x=list(Companies)[:15],y=values[:15],edgecolor='.2',palette='Set2')
plt.xticks(rotation='90')
plt.xlabel('Companies')
plt.ylabel('Count')
plt.show()
```

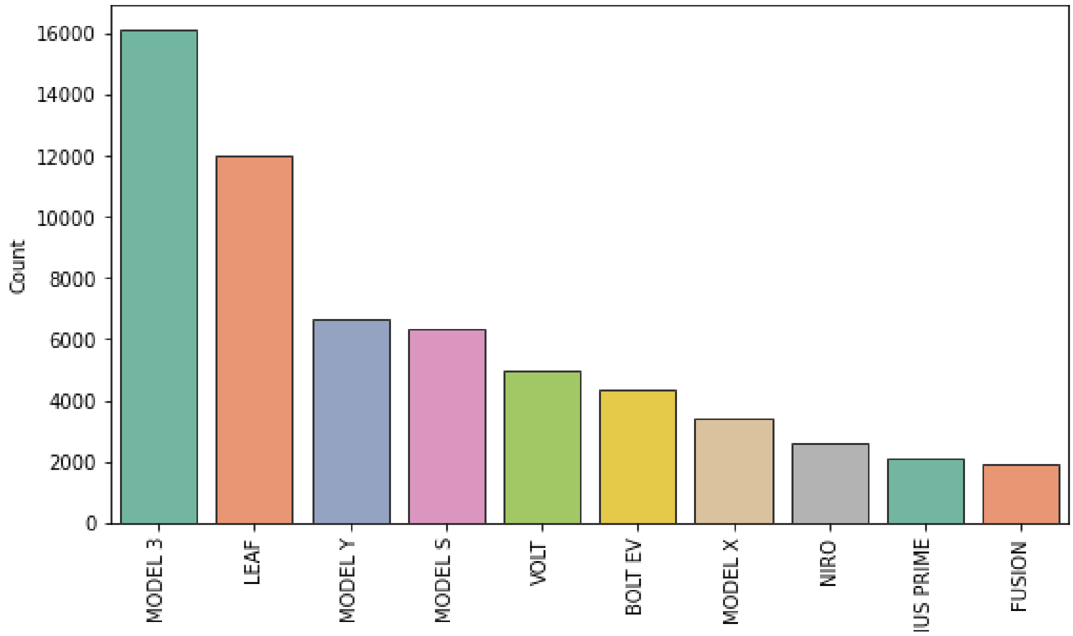


In [19]:

```
px.pie(names=list(Companies)[:15],values=values[:15],width=500,height=600)
```

In [20]:

```
Models = df.groupby('Model').count().sort_values(by='City',ascending=False)['City'].inde
values = df.groupby('Model').count().sort_values(by='City',ascending=False)['City'].valu


plt.figure(figsize=(9,5))
sns.barplot(x=list(Models)[:10],y=values[:10],edgecolor='.2',palette='Set2')
plt.xticks(rotation='90')
plt.xlabel('Models')
plt.ylabel('Count')
plt.show()
```



In [21]:

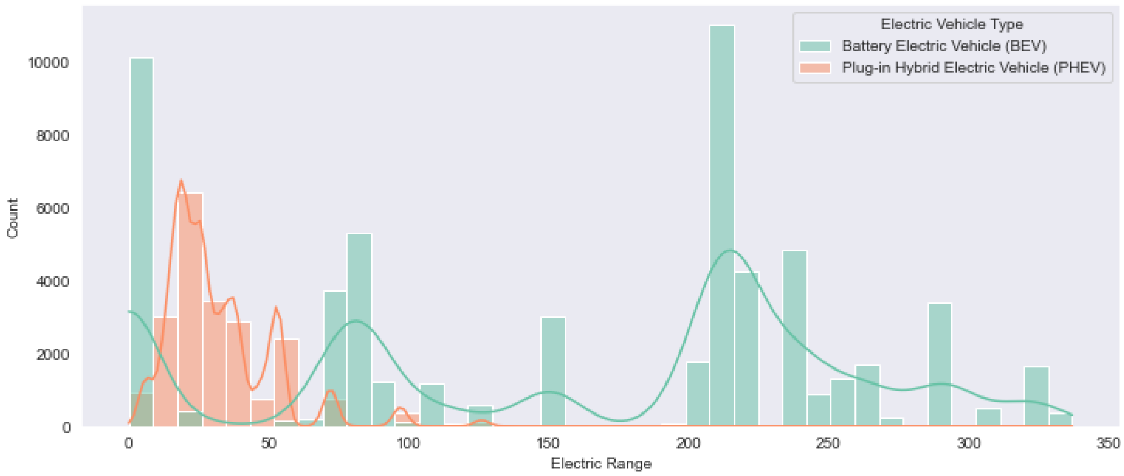
```
#Percentage of BEV vs PHEV

Vehicle_type = list(df.groupby('Electric Vehicle Type').count()['County'].index)
values = df.groupby('Electric Vehicle Type').count()['County'].values

px.pie(names=Vehicle_type,values=values,height=400)
```

In [22]:

```
#lets see the electric range difference between PHEV and BEV
plt.figure(figsize=(12,5))
sns.set_style(style='dark')
sns.histplot(x = 'Electric Range',data=df,kde=True,hue='Electric Vehicle Type',palette='
```

Out[22]:

```
<AxesSubplot:xlabel='Electric Range', ylabel='Count'>
```



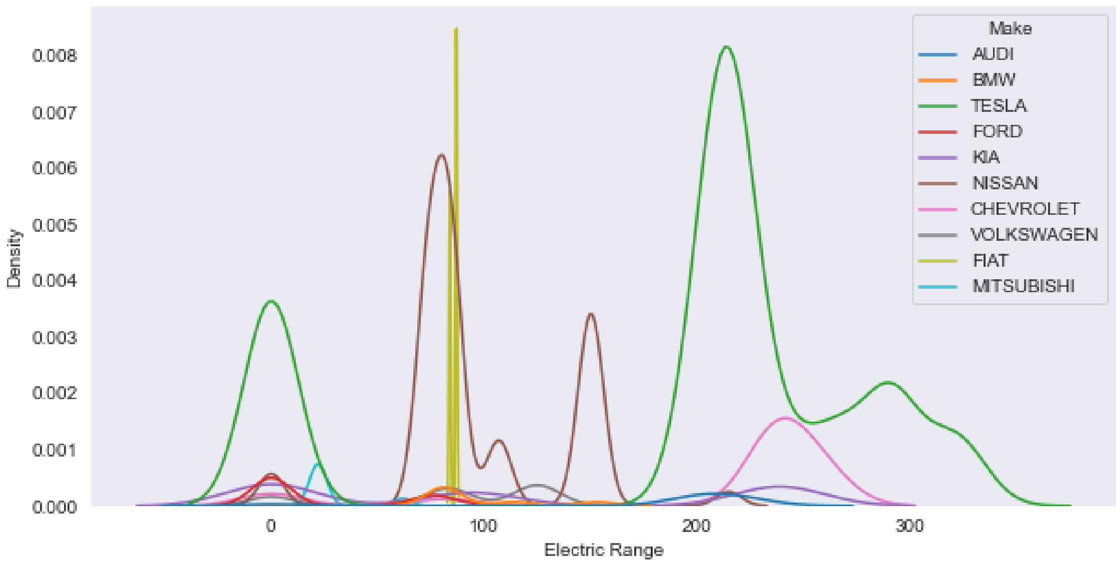
In [23]:

```
data_bev = df[df['Electric Vehicle Type']!='Plug-in Hybrid Electric Vehicle (PHEV)']
companies=list(data_bev.groupby('Make').count().sort_values(by='City',ascending=False)['
data_bev['bev'] = data_bev['Make'].apply(lambda x:1 if x in companies else 0 )
data_bev = data_bev[data_bev['bev']==1]

plt.figure(figsize=(10,5))
sns.kdeplot(x='Electric Range',hue='Make',data=data_bev)
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_1532\535499912.py:3: SettingWith
CopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
```

Out[23]:

```
<AxesSubplot:xlabel='Electric Range', ylabel='Density'>
```



In [24]:

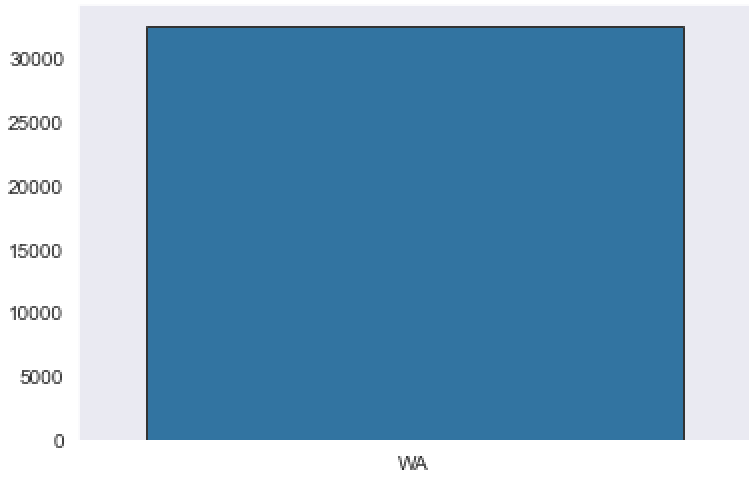
```
data_TESLA = df[df['Make']=='TESLA']
top_10_states_TESLA = list(data_TESLA.groupby('State').count().sort_values(by='City',asc
values = list(data_TESLA.groupby('State').count().sort_values(by='City',ascending=False)
```

In [25]:

```
sns.barplot(x = top_10_states_TESLA,y=values,edgecolor='.2')
```

Out[25]:

<AxesSubplot:>

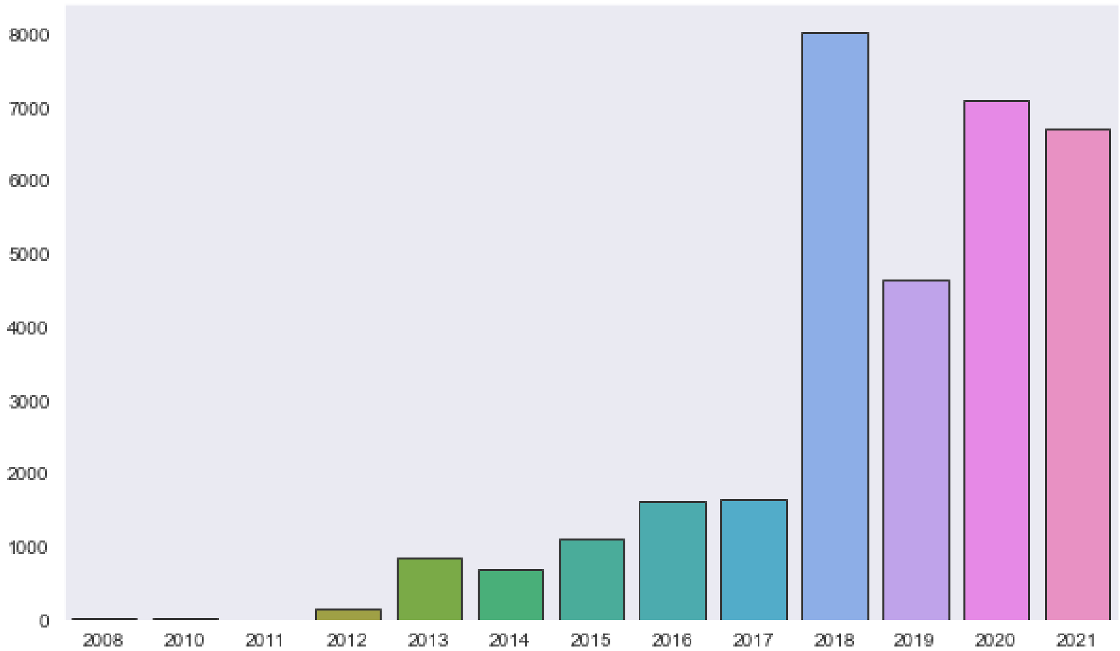


In [26]:

```
plt.figure(figsize=(10,6))
top_10_year_TESLA = list(data_TESLA.groupby('Model Year').count().sort_values(by='City',
values = list(data_TESLA.groupby('Model Year').count().sort_values(by='City',ascending=F
sns.barplot(x = top_10_year_TESLA,y=values,edgecolor='.2')
```

Out[26]:

<AxesSubplot:>



In [27]:

```
locations = list(df.groupby('Vehicle Location').count()['County'].index)
values = list(df.groupby('Vehicle Location').count()['County'].values)
Location_data = pd.DataFrame({'Locations':locations,'Count':values})
Location_data['Lattitude'] = Location_data['Locations'].apply(lambda x:float(x.split(' '
Location_data['Longitude'] = Location_data['Locations'].apply(lambda x:float(x.split(' '

plt.figure(figsize=(10,5))
plt.scatter(x = Location_data['Lattitude'],y = Location_data['Longitude'],s=Location_dat
plt.xlim(-130,-60)
plt.ylim(20,60)
```

Out[27]:

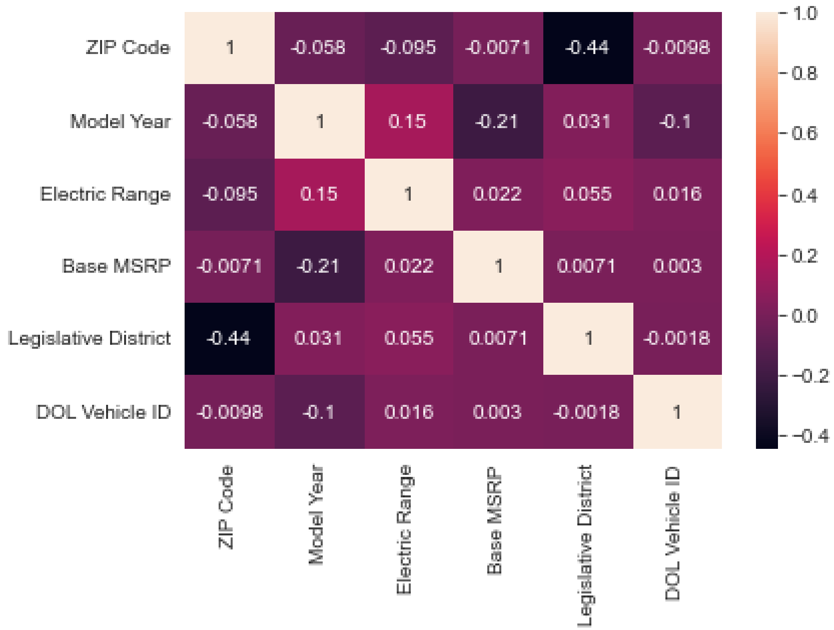
```
(20.0, 60.0)
```

In [28]:

```
cor_matrix=df.corr()
```

In [29]:

```
sns.heatmap(cor_matrix,annot=True)
plt.show()
```



In [30]:

```
df['Vehicle Location'].value_counts()
df['County'].value_counts()
df['State'].value_counts()
```

Out[30]:

```
WA    79619
Name: State, dtype: int64
```

In [31]:

```
df.describe()
```

Out[31]:

	ZIP Code	Model Year	Electric Range	Base MSRP	Legislative District	DOL Vehicle ID
count	79619.000000	79619.000000	79619.000000	79619.000000	79619.000000	7.961900e+0
mean	98258.343586	2017.630126	124.569412	2941.137794	30.156144	2.000480e+0
std	299.454690	2.575760	102.883121	13191.904432	14.581814	1.186409e+0
min	98001.000000	1993.000000	0.000000	0.000000	1.000000	4.385000e+0
25%	98052.000000	2016.000000	25.000000	0.000000	20.000000	1.284651e+0
50%	98121.000000	2018.000000	84.000000	0.000000	34.000000	1.611223e+0
75%	98370.000000	2020.000000	215.000000	0.000000	43.000000	2.556195e+0
max	99403.000000	2022.000000	337.000000	845000.000000	49.000000	4.792548e+0

In [32]:

```
df.describe(include="all")
```

Out[32]:

	VIN (1-10)	County	City	State	ZIP Code	Model Year	Make	Model
count	79619	79619	79619	79619	79619.000000	79619.000000	79619	79619
unique	5196	39	425	1	NaN	NaN	32	93
top	5YJ3E1EB6J	King	SEATTLE	WA	NaN	NaN	TESLA	MODEL 3
freq	339	41740	15179	79619	NaN	NaN	32537	16139
mean	NaN	NaN	NaN	NaN	98258.343586	2017.630126	NaN	NaN
std	NaN	NaN	NaN	NaN	299.454690	2.575760	NaN	NaN
min	NaN	NaN	NaN	NaN	98001.000000	1993.000000	NaN	NaN
25%	NaN	NaN	NaN	NaN	98052.000000	2016.000000	NaN	NaN
50%	NaN	NaN	NaN	NaN	98121.000000	2018.000000	NaN	NaN
75%	NaN	NaN	NaN	NaN	98370.000000	2020.000000	NaN	NaN
max	NaN	NaN	NaN	NaN	99403.000000	2022.000000	NaN	NaN

In [33]:

```
df.fillna(df.mode(),inplace=True)
df
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_1532\593388003.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
(https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy)
```

Out[33]:

	VIN (1-10)	County	City	State	ZIP Code	Model Year	Make	Model	Electric Vehicle

In [34]:

```
df['VIN (1-10)'].value_counts()
```

Out[34]:

```
5YJ3E1EB6J    339
5YJ3E1EBXJ    336
5YJ3E1EB0J    332
5YJ3E1EB1J    330
5YJ3E1EB5J    327
             ... 
5YJXCBE43K      1
WBY8P8C5XK      1
YV4BR0PM6J      1
KMHC75LD5J      1
KL8CK6S09E      1
Name: VIN (1-10), Length: 5196, dtype: int64
```

In [35]:

```
df['County'].value_counts()
```

Out[35]:

```
King               41740
Snohomish           8463
Pierce              6038
Clark               4562
Thurston            3095
Kitsap              2871
Whatcom             2174
Spokane             1880
Island               980
Benton               970
Skagit               935
Clallam              579
San Juan             578
Jefferson            531
Chelan               458
Yakima               417
Cowlitz              406
Mason                391
Grays Harbor         340
Lewis                324
Kittitas             249
Franklin             239
Grant                227
Walla Walla          195
Douglas              154
Klickitat            129
Whitman              121
Pacific              111
Okanogan             108
Stevens               99
Skamania              92
Asotin                40
Adams                 30
Wahkiakum             23
Lincoln               23
Pend Oreille          23
Ferry                 12
Columbia               9
Garfield               3
Name: County, dtype: int64
```

In [36]:

```
df['City'].value_counts()
```

Out[36]:

```
SEATTLE          15179
BELLEVUE          4115
REDMOND           3127
VANCOUVER         2835
KIRKLAND          2537
                 ...  
BRIDGEPORT           1
LEWIS MCCHORD        1
KITTITAS             1
KLICKITAT            1
LATAH                1
Name: City, Length: 425, dtype: int64
```

In [37]:

```
df['State'].value_counts()
```

Out[37]:

```
WA    79619
Name: State, dtype: int64
```

In [38]:

```
df['ZIP Code'].value_counts()
```

Out[38]:

```
98052    2185
98033    1473
98115    1416
98004    1366
98006    1289
         ... 
98647       1
99116       1
98819       1
98227       1
99018       1
Name: ZIP Code, Length: 526, dtype: int64
```

In [39]:

```
df['Model Year'].value_counts()
```

Out[39]:

```
2018    13910
2021    11104
2020    11014
2019    10595
2017     9853
2016     6441
2015     5149
2013     4944
2014     3800
2012     1787
2011      871
2022       80
2010       29
2008       27
2000        7
1999        3
2002        2
1997        1
1998        1
1993        1
Name: Model Year, dtype: int64
```

In [40]:

```
df['Make'].value_counts()
```

Out[40]:

```
TESLA                    32537
NISSAN                   12037
CHEVROLET                 9569
FORD                      4470
KIA                       3834
BMW                       3464
TOYOTA                    3327
AUDI                      1622
CHRYSLER                  1381
VOLKSWAGEN                1293
HYUNDAI                   1087
VOLVO                      820
FIAT                       779
HONDA                      753
PORSCHE                    518
MITSUBISHI                 514
MINI                       325
MERCEDES-BENZ              293
JEEP                       284
SMART                      269
JAGUAR                     192
CADILLAC                    91
SUBARU                      44
LAND ROVER                  39
LINCOLN                     30
POLESTAR                    16
FISKER                      14
AZURE DYNAMICS               9
TH!NK                        3
BENTLEY                      2
WHEEGO ELECTRIC CARS         2
DODGE                        1
Name: Make, dtype: int64
```

In [41]:

```
df['Model'].value_counts()
```

Out[41]:

```
MODEL 3                  16139
LEAF                     12037
MODEL Y                   6627
MODEL S                   6328
VOLT                      4977
                         ...
918 SPYDER                   1
S-10 PICKUP                  1
XC90 AWD PHEV                1
CARAVAN                      1
PRIUS PLUG-IN HYBRID         1
Name: Model, Length: 93, dtype: int64
```

In [42]:

```
df['Electric Vehicle Type'].value_counts()
```

Out[42]:

```
Battery Electric Vehicle (BEV)            58383
Plug-in Hybrid Electric Vehicle (PHEV)    21236
Name: Electric Vehicle Type, dtype: int64
```

In [43]:

```
df['Vehicle Location'].value_counts()
```

Out[43]:

```
POINT (-122.122018 47.678465)           2185
POINT (-122.188994 47.678406)           1473
POINT (-122.297534 47.685291)           1416
POINT (-122.20316899999999 47.619011)   1366
POINT (-122.151342 47.560192)           1289
                                        ...
POINT (-123.45692199999999 46.303845)      1
POINT (-119.09443399999999 48.280155)      1
POINT (-122.478122 48.754899)              1
POINT (-118.33470700000001 47.424407)      1
POINT (-117.17122200000001 47.28842)       1
Name: Vehicle Location, Length: 524, dtype: int64
```

In [44]:

```
df['DOL Vehicle ID'].value_counts()
```

Out[44]:

```
148815901    1
107129088    1
263776334    1
247585222    1
141108565    1
            ..
2565741      1
101569513    1
243016693    1
341041018    1
328614947    1
Name: DOL Vehicle ID, Length: 79619, dtype: int64
```

In [45]:

```
df['Legislative District'].value_counts()
```

Out[45]:

```
41.0    5290
45.0    5152
48.0    4663
36.0    3892
46.0    3572
43.0    3557
1.0     3191
5.0     3136
34.0    2625
37.0    2620
22.0    2130
40.0    2029
23.0    2010
18.0    1971
32.0    1937
21.0    1789
44.0    1767
11.0    1594
26.0    1581
10.0    1532
24.0    1302
17.0    1298
47.0    1251
42.0    1233
31.0    1232
27.0    1204
49.0    1169
35.0    1132
33.0    1036
39.0    1030
28.0    1021
2.0      871
30.0     863
8.0      834
38.0     831
25.0     775
12.0     704
20.0     694
6.0      684
29.0     571
19.0     536
4.0      536
14.0     506
13.0     489
9.0      422
3.0      419
16.0     389
7.0      369
15.0     180
Name: Legislative District, dtype: int64
```

In [46]:

```
df['Base MSRP'].value_counts()
```

Out[46]:

```
0         75154
69900      1533
34600       528
31950       460
28500       232
52900       203
32250       176
38500       165
59900       153
54950       132
39995       118
44100        98
36900        87
64950        83
33950        68
45600        54
36800        52
55700        47
52650        45
34995        44
110950       29
98950        27
81100        17
53400        16
90700        16
102000       14
75095        14
35390        11
43700        11
184400       10
89100         9
109000        5
91250         3
32995         2
845000        1
66300         1
32000         1
Name: Base MSRP, dtype: int64
```

In [47]:

```
df['Electric Range'].value_counts()
```

Out[47]:

```
0      10140
215     6509
84      4220
220     4068
238     3630
       ...
95         2
57         1
59         1
80         1
39         1
Name: Electric Range, Length: 99, dtype: int64
```

Lable Encoding

In [48]:

```
from sklearn.preprocessing import LabelEncoder
def labelling(x):
    df[x] = LabelEncoder().fit_transform(df[x])
    return df
cat = ['VIN (1-10)', 'County', 'City', 'State', 'ZIP Code', 'Model Year', 'Make','Base M
       'Model', 'Electric Vehicle Type', 'Clean Alternative Fuel Vehicle (CAFV) Eligibil
for i in cat:
    labelling(i)
df
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_1532\852114733.py:3: SettingWith
CopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
```

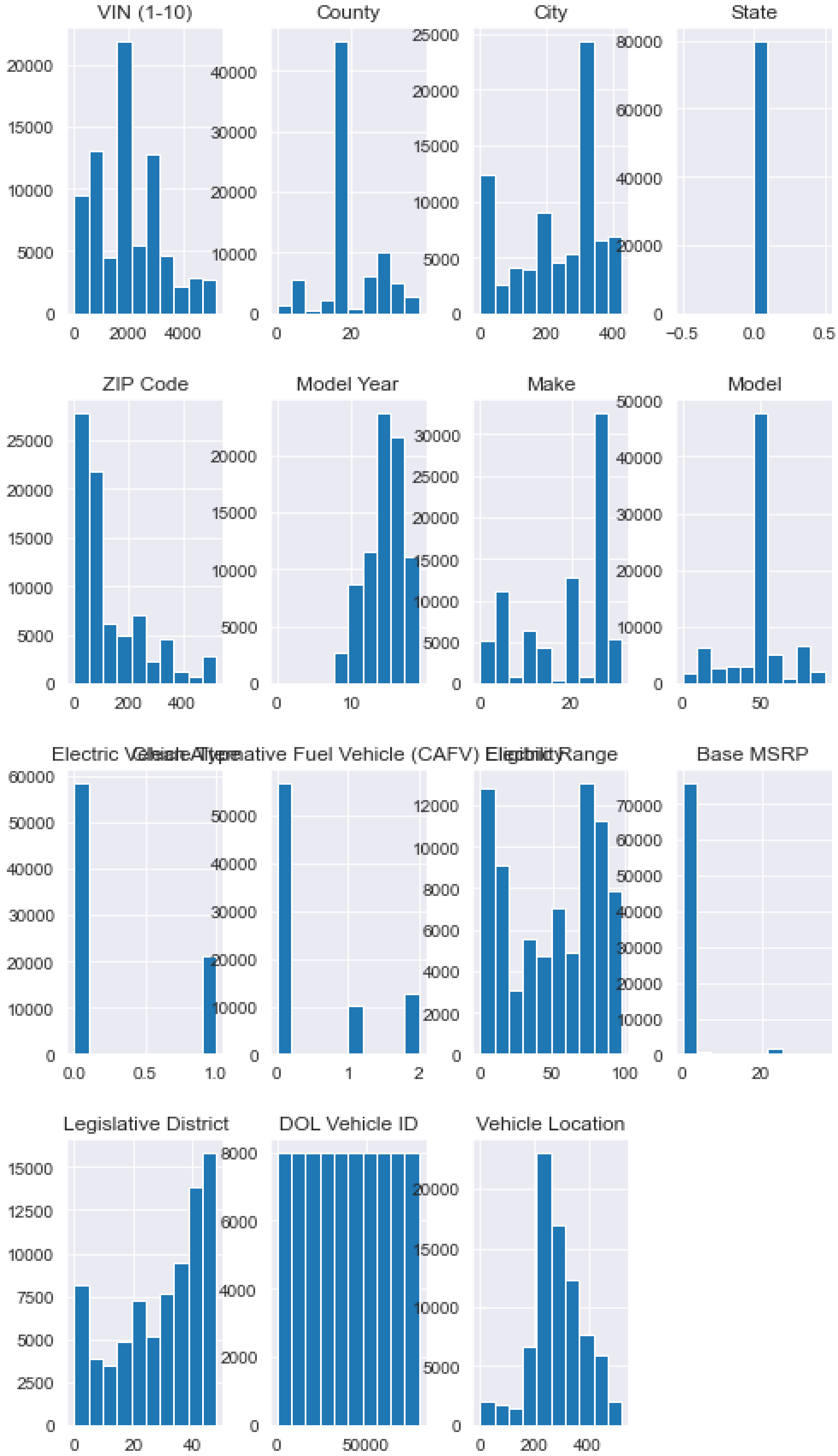
Out[48]:

	VIN (1-10)	County	City	State	ZIP Code	Model Year	Make	Model	Electric Vehicle Type	Clean Alternative Fuel Vehicle (CAFV) Eligibility	Electric Range
0	3857	17	291	0	202	18	0	25	0	0	80
1	4432	16	331	0	73	17	3	39	0	0	69
2	2365	7	343	0	340	16	26	51	0	0	92
3	170	16	331	0	69	4	10	68	0	0	39
4	4319	16	331	0	71	14	3	39	0	0	49
...	...	...	...	...	...	...	...	...	...	...	...
79762	2734	16	331	0	67	15	20	58	0	2	16
79763	331	31	99	0	423	17	5	14	0	0	88
79764	2620	16	331	0	66	17	26	52	0	0	93
79765	328	26	188	0	246	17	5	14	0	0	88
79766	1730	31	357	0	495	15	26	49	0	0	77

79619 rows × 15 columns

In [49]:

```
plt.rcParams['figure.figsize'] = (8,15)
df.hist()
plt.show()
```



In [50]:

```
x = df.loc[:,cat].values
x
```

Out[50]:

```
array([[ 3857,    17,   291, ...,      0,     80, 32412],
       [ 4432,    16,   331, ...,      0,     69, 21673],
       [ 2365,     7,   343, ...,      0,     92, 36963],
       ...,
       [ 2620,    16,   331, ...,      0,     93, 19021],
       [  328,    26,   188, ...,      0,     88, 27596],
       [ 1730,    31,   357, ...,      0,     77, 67709]], dtype=int64)
```

In [51]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x = sc.fit_transform(x)
```

In [52]:

```
from sklearn.decomposition import PCA
pca_model = PCA(n_components=15)
x_PCA = pca_model.fit_transform(x)
```

In [53]:

```
pca_model.explained_variance_ratio_
```

Out[53]:

```
array([1.78444593e-01, 1.13900292e-01, 9.97819324e-02, 9.01390547e-02,
       8.82027945e-02, 7.43801855e-02, 7.04183261e-02, 6.85851572e-02,
       5.69727295e-02, 4.76542824e-02, 4.41496947e-02, 3.31887554e-02,
       2.44178576e-02, 9.76434472e-03, 1.17700961e-33])
```

In [54]:

```
X = df.iloc[:,[3,4]].values
```

K-means clustering

In [55]:

```
from sklearn.cluster import KMeans
```

In [56]:

```
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
```

In [57]:

```
wcss
```

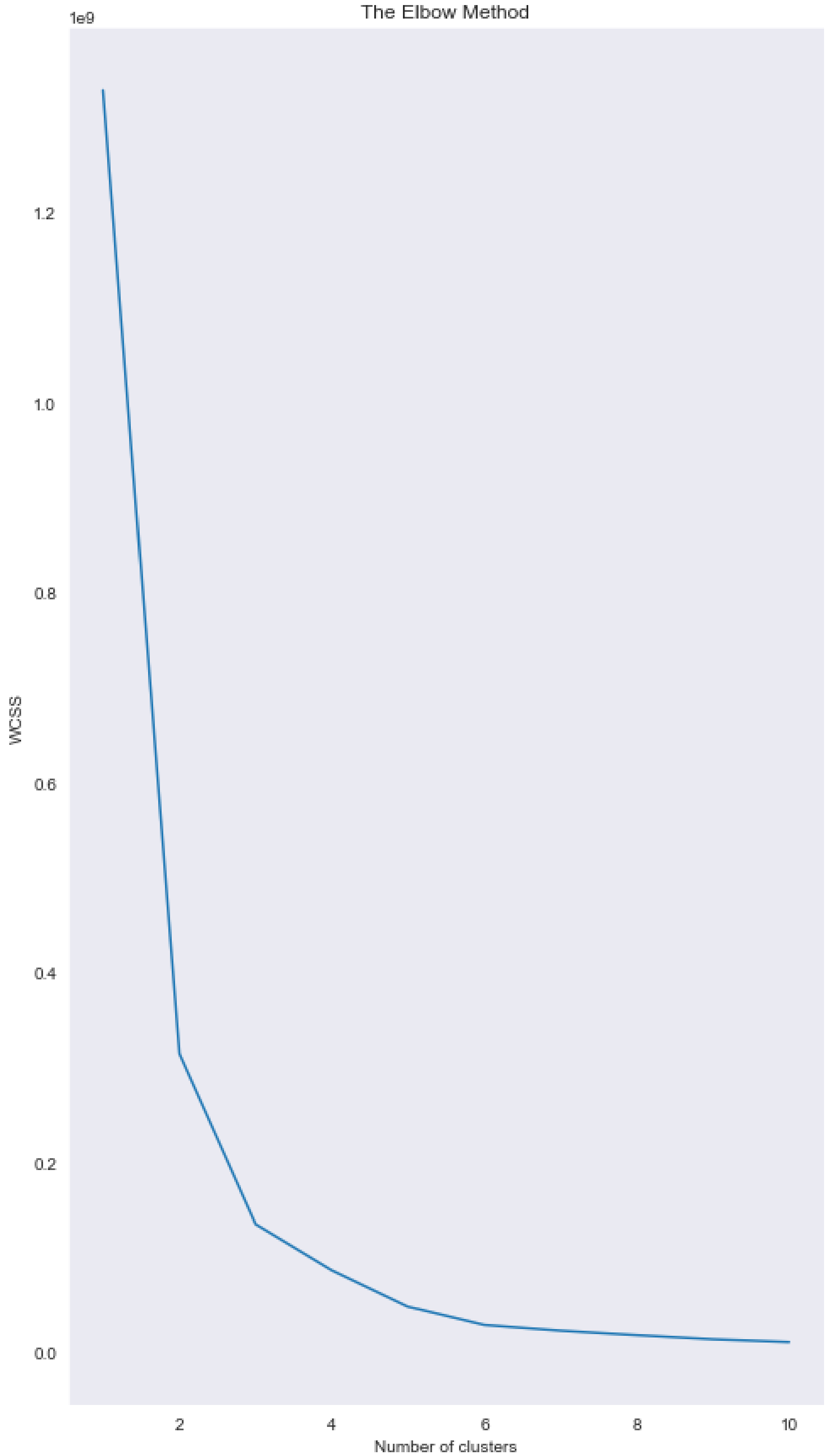
Out[57]:

```
[1328957249.9840012,
 314294678.4677549,
 135019505.17092726,
 86574080.20231578,
 48266749.99067938,
 28996394.16967954,
 22933193.055935137,
 18223294.409017555,
 13888757.80920052,
 10900144.928630024]
```

Elbow Curve

In [58]:

```
plt.plot(range(1,11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



In [59]:

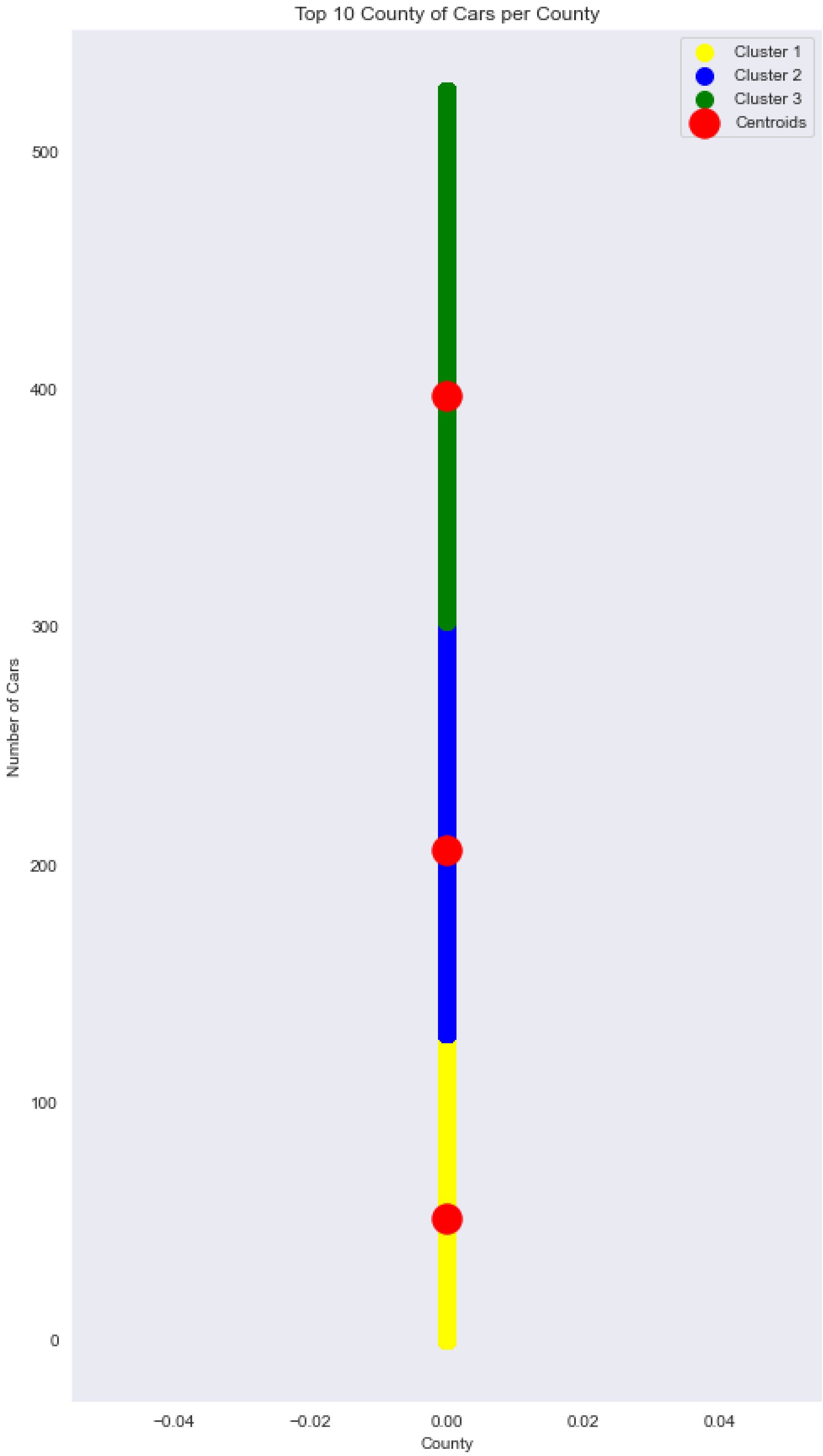
```
kmeans = KMeans(n_clusters = 3,init = 'k-means++', random_state = 42)
```

In [60]:

```
y_kmeans = kmeans.fit_predict(X)
```

In [68]:

```
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1],
            s = 100, c = 'yellow', label = 'Cluster 1')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1],
            s = 100, c = 'blue', label = 'Cluster 2')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1],
            s = 100, c = 'green', label = 'Cluster 3')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
           s = 300, c = 'red', label = 'Centroids')
plt.title('Top 10 County of Cars per County')
plt.xlabel('County')
plt.ylabel('Number of Cars')
plt.legend()
plt.show()
```



In [ ]:

DBSCAN CLUSTERING

In [62]:

```
from sklearn.cluster import DBSCAN
```

In [63]:

```
dbs = DBSCAN(eps=5, min_samples=5)
```

In [64]:

```
y_dbs = dbs.fit_predict(X)
```

In [65]:

```
y_dbs
```

Out[65]:

```
array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

In [66]:

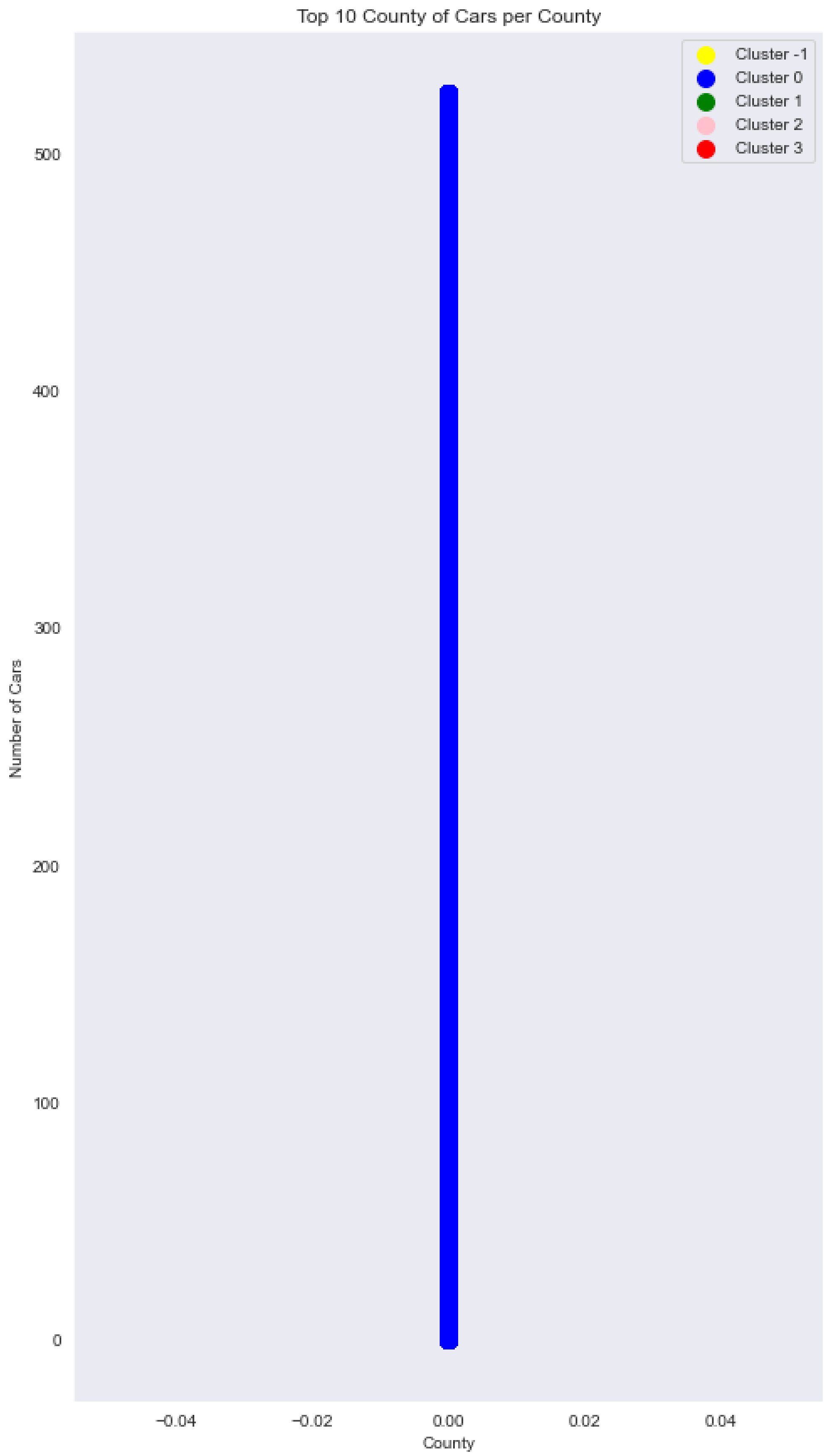
```
np.unique(y_dbs)
```

Out[66]:

```
array([0], dtype=int64)
```

In [67]:

```
plt.scatter(X[y_dbs == -1, 0], X[y_dbs == -1, 1],s = 100, c = 'yellow', label = 'Cluster
plt.scatter(X[y_dbs == 0, 0], X[y_dbs == 0, 1],s = 100, c = 'blue', label = 'Cluster 0')
plt.scatter(X[y_dbs== 1, 0], X[y_dbs == 1, 1], s = 100, c = 'green', label = 'Cluster 1'
plt.scatter(X[y_dbs== 2, 0], X[y_dbs == 2, 1],s = 100, c = 'pink', label = 'Cluster 2')
plt.scatter(X[y_dbs== 3, 0], X[y_dbs == 3, 1],s = 100, c = 'red', label = 'Cluster 3')
plt.title('Top 10 County of Cars per County')
plt.xlabel('County')
plt.ylabel('Number of Cars')
plt.legend()
plt.show()
```


Top 10 County of Cars per County
Cluster -1
Cluster 0
Cluster 1
Cluster 2
Cluster 3
500
400
300
200
100
0
Number of Cars
-0.04
-0.02
0.00
0.02
0.04
County

In [ ]: