Business Prombelm

Based on Features we are going to predict the fuel consumption

Data Mining

In [14]:
```
1 ls
```

```
 Volume in drive C has no label.
 Volume Serial Number is A007-E621

 Directory of C:\Users\tswar\Downloads\SocialTek\2022 Fuel Consumption

07/25/2022  04:06 PM    <DIR>          .
07/25/2022  04:06 PM    <DIR>          ..
07/25/2022  03:52 PM    <DIR>          .ipynb_checkpoints
06/21/2022  02:52 PM         8,061,682 2022FuelConsumption.pdf
07/25/2022  03:48 PM         3,533,694 A_fuel_Consumption.ipynb
04/06/2022  02:41 PM            73,008 MY2022 Fuel Consumption Ratings.csv
05/11/2022  09:26 PM           474,720 Untitled.ipynb
07/24/2022  04:54 PM           284,289 Untitled1.ipynb
07/24/2022  04:25 PM               588 Untitled2.ipynb
07/25/2022  04:06 PM             3,703 Untitled3.ipynb
               7 File(s)     12,431,684 bytes
               3 Dir(s)  103,347,896,320 bytes free
```

In [15]:
```
1 import pandas as pd
2 import numpy as np
3 import  seaborn as sns
4 import matplotlib.pyplot as plt
5 %matplotlib inline
```

In [16]:
```
1 import warnings
2 warnings.filterwarnings("ignore")
```

In [17]:
```
df = pd.read_csv("MY2022 Fuel Consumption Ratings.csv")
df.head(3)
```

Out[17]:

	Model Year	Make	Model	Vehicle Class	Engine Size(L)	Cylinders	Transmission	Fuel Type	Fuel Consumption (City (L/100 km)	Consump (L/
0	2022	Acura	ILX	Compact	2.4	4	AM8	Z	9.9	
1	2022	Acura	MDX SH-AWD	SUV: Small	3.5	6	AS10	Z	12.6	
2	2022	Acura	RDX SH-AWD	SUV: Small	2.0	4	AS10	Z	11.0	

In [18]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 946 entries, 0 to 945
Data columns (total 15 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   Model Year                         946 non-null    int64
 1   Make                               946 non-null    object
 2   Model                              946 non-null    object
 3   Vehicle Class                      946 non-null    object
 4   Engine Size(L)                     946 non-null    float64
 5   Cylinders                          946 non-null    int64
 6   Transmission                       946 non-null    object
 7   Fuel Type                          946 non-null    object
 8   Fuel Consumption (City (L/100 km)  946 non-null    float64
 9   Fuel Consumption(Hwy (L/100 km))   946 non-null    float64
 10  Fuel Consumption(Comb (L/100 km))  946 non-null    float64
 11  Fuel Consumption(Comb (mpg))       946 non-null    int64
 12  CO2 Emissions(g/km)                946 non-null    int64
 13  CO2 Rating                         946 non-null    int64
 14  Smog Rating                        946 non-null    int64
dtypes: float64(4), int64(6), object(5)
memory usage: 111.0+ KB
```

In [19]:
```
1 df.describe()
```

Out[19]:

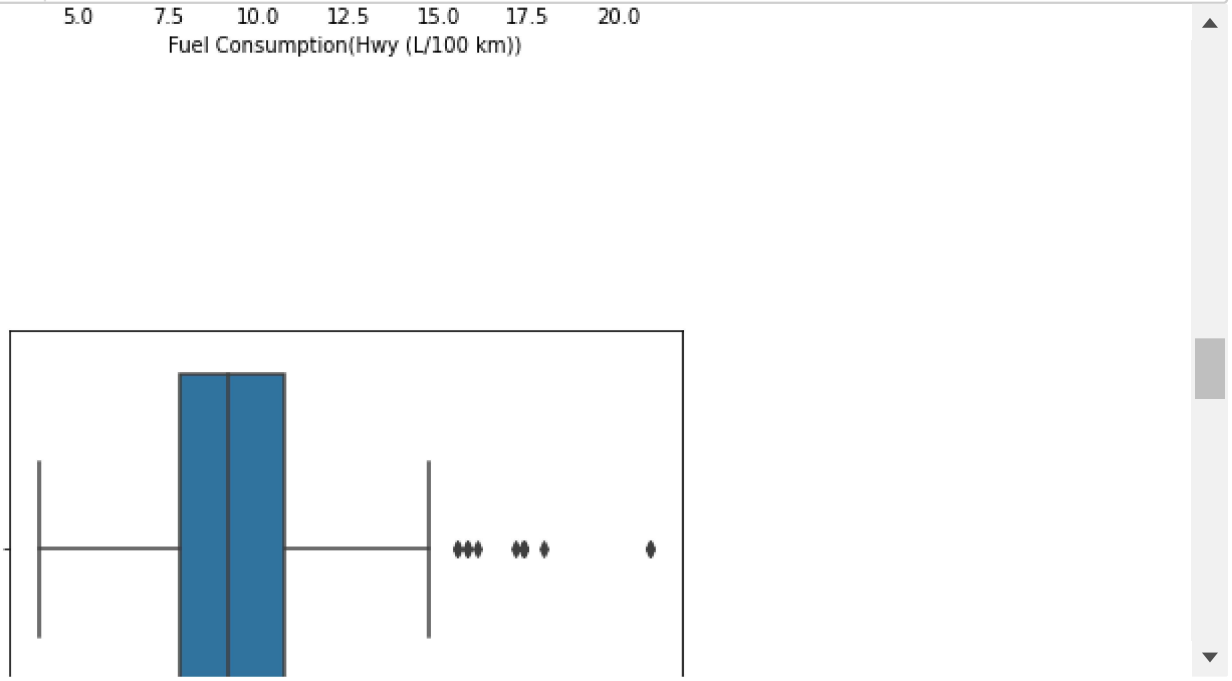
	Model Year	Engine Size(L)	Cylinders	Fuel Consumption (City (L/100 km)	Fuel Consumption(Hwy (L/100 km))	Fuel Consumption(Comb (L/100 km))	Cons
count	946.0	946.000000	946.000000	946.000000	946.000000	946.000000	
mean	2022.0	3.198732	5.668076	12.506448	9.363319	11.092072	
std	0.0	1.374814	1.932670	3.452043	2.285125	2.876276	
min	2022.0	1.200000	3.000000	4.000000	3.900000	4.000000	
25%	2022.0	2.000000	4.000000	10.200000	7.700000	9.100000	
50%	2022.0	3.000000	6.000000	12.200000	9.200000	10.800000	
75%	2022.0	3.800000	6.000000	14.700000	10.700000	12.900000	
max	2022.0	8.000000	16.000000	30.300000	20.900000	26.100000	

In [20]:
```
1 df.columns
```

Out[20]:
```
Index(['Model Year', 'Make', 'Model', 'Vehicle Class', 'Engine Size(L)',
       'Cylinders', 'Transmission', 'Fuel Type',
       'Fuel Consumption (City (L/100 km)', 'Fuel Consumption(Hwy (L/100 km))',
       'Fuel Consumption(Comb (L/100 km))', 'Fuel Consumption(Comb (mpg))',
       'CO2 Emissions(g/km)', 'CO2 Rating', 'Smog Rating'],
      dtype='object')
```

In [21]:
```
for i in df.columns:
    try:
        sns.boxplot(df[i])
        plt.show()
    except:
        print("This is Categorial",i)

```



In [22]:
```
df.columns
```

Out[22]:
```
Index(['Model Year', 'Make', 'Model', 'Vehicle Class', 'Engine Size(L)',
       'Cylinders', 'Transmission', 'Fuel Type',
       'Fuel Consumption (City (L/100 km)', 'Fuel Consumption(Hwy (L/100 km))',
       'Fuel Consumption(Comb (L/100 km))', 'Fuel Consumption(Comb (mpg))',
       'CO2 Emissions(g/km)', 'CO2 Rating', 'Smog Rating'],
      dtype='object')
```

In [23]:
```
df.rename(columns={'Fuel Consumption(Comb (L/100 km))':"Fuel_consuption",'CO2
```

In [24]:
```
df.columns
```

Out[24]:
```
Index(['Model Year', 'Make', 'Model', 'Vehicle Class', 'ES', 'Cylinders',
       'Transmission', 'Fuel Type', 'Fuel Consumption (City (L/100 km)',
       'Fuel Consumption(Hwy (L/100 km))', 'Fuel_consuption',
       'Fuel Consumption(Comb (mpg))', 'co2', 'CO2 Rating', 'Smog Rating'],
      dtype='object')
```

In [25]:
```
df.drop(columns=['Fuel Consumption(Hwy (L/100 km))','Fuel Consumption(Comb (
```

In [26]:
```
1 df.columns
```

Out[26]:
```
Index(['Model Year', 'Make', 'Model', 'Vehicle Class', 'ES', 'Cylinders',
       'Transmission', 'Fuel Type', 'Fuel_consuption', 'co2', 'CO2 Rating',
       'Smog Rating'],
      dtype='object')
```

In [27]:
```
1 from feature_engine.outliers import Winsorizer
```

In [28]:
```
1 win_E = Winsorizer(capping_method="iqr",tail='both',fold=1.5,variables=["ES"
```

In [29]:
```
1 win_C = Winsorizer(capping_method="iqr",tail='both',fold=1.5,variables=["Cyl
```

In [30]:
```
1 win_CO = Winsorizer(capping_method="iqr",tail='both',fold=1.5,variables=["co
```

In [31]:
```
1 win_CO2 = Winsorizer(capping_method="iqr",tail='both',fold=1.5,variables=["C
```

In [32]:
```
1 df["ES"] = win_E.fit_transform(df[["ES"]])
```

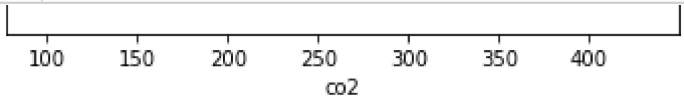
In [33]:
```
1 df["Cylinders"] = win_C.fit_transform(df[["Cylinders"]])
```

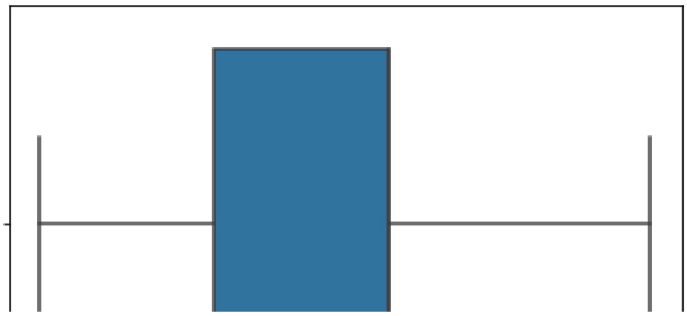
In [34]:
```
1 df["co2"]=win_CO.fit_transform(df[["co2"]])
```

In [35]:
```
1 df["CO2 Rating"] =win_CO2.fit_transform(df[["CO2 Rating"]])
```

In [36]:
```
for i in df.columns:
    try:
        sns.boxplot(df[i])
        plt.show()
    except:
        print("This is Categorial",i)

```





In [37]:
```
df.head(3)
```

Out[37]:

	Model Year	Make	Model	Vehicle Class	ES	Cylinders	Transmission	Fuel Type	Fuel_consuption	co2	C Rat
0	2022	Acura	ILX	Compact	2.4	4.0	AM8	Z	8.6	200.0	
1	2022	Acura	MDX SH-AWD	SUV: Small	3.5	6.0	AS10	Z	11.2	263.0	
2	2022	Acura	RDX SH-AWD	SUV: Small	2.0	4.0	AS10	Z	9.9	232.0	

Encodeing

In [38]:
```
from sklearn.preprocessing import LabelEncoder
```

In [39]:
```
le = LabelEncoder()
```

In [40]:
```
df["Vehicle Class"] = le.fit_transform(df[["Vehicle Class"]])
```

In [41]:
```
df["Transmission"] = le.fit_transform(df[["Transmission"]])
```

In [43]:
```
df["Fuel Type"] = le.fit_transform(df[["Fuel Type"]])
```

In [44]:
```
df.head(3)
```

Out[44]:

	Model Year	Make	Model	Vehicle Class	ES	Cylinders	Transmission	Fuel Type	Fuel_consuption	co2	CC Ratii
0	2022	Acura	ILX	0	2.4	4.0	7	3	8.6	200.0	6
1	2022	Acura	MDX SH-AWD	7	3.5	6.0	8	3	11.2	263.0	4
2	2022	Acura	RDX SH-AWD	7	2.0	4.0	8	3	9.9	232.0	5

In [45]:
```
from sklearn.preprocessing import OneHotEncoder
```

In [48]:
```
socialprachar = OneHotEncoder(sparse=False)
```

In [55]:
```
x = socialprachar.fit_transform(df[["Make"]])
```

In [53]:
```
y = socialprachar.get_feature_names_out()
```

In [56]:
```
x1 = pd.DataFrame(x,columns=y)
```

In [57]:
```
x1
```

Out[57]:

	Make_Acura	Make_Alfa Romeo	Make_Aston Martin	Make_Audi	Make_BMW	Make_Bentley	Make_Bugatti	M
0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	
941	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
942	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
943	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
944	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
945	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

946 rows × 39 columns

In [58]:
```
a =socialprachar.fit_transform(df[['Model']])
a
```

Out[58]:
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

In [60]:
```
b=socialprachar.get_feature_names_out()
```

In [61]:
```
x2 = pd.DataFrame(a,columns=b)
x2
```

Out[61]:

	Model_1500	Model_1500 4X4	Model_1500 4X4 EcoDiesel	Model_1500 4X4 TRX	Model_1500 4X4 eTorque	Model_1500 Classic	Model_1500 Classic 4X4	Mo
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	
941	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
942	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
943	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
944	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
945	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

946 rows × 715 columns

In [62]:
```
final = pd.concat([df,x1,x2],axis=1)
final
```

Out[62]:

	Model Year	Make	Model	Vehicle Class	ES	Cylinders	Transmission	Fuel Type	Fuel_consuption	co2	...
0	2022	Acura	ILX	0	2.4	4.0	7	3	8.6	200.0	...
1	2022	Acura	MDX SH-AWD	7	3.5	6.0	8	3	11.2	263.0	...
2	2022	Acura	RDX SH-AWD	7	2.0	4.0	8	3	9.9	232.0	...
3	2022	Acura	RDX SH-AWD A-SPEC	7	2.0	4.0	8	3	10.3	242.0	...
4	2022	Acura	TLX SH-AWD	0	2.0	4.0	8	3	9.8	230.0	...
...	...	...	...	...	...	...	...	...	...	...	...
941	2022	Volvo	XC40 T5 AWD	7	2.0	4.0	12	3	9.4	219.0	...
942	2022	Volvo	XC60 B5 AWD	7	2.0	4.0	12	3	9.4	219.0	...
943	2022	Volvo	XC60 B6 AWD	7	2.0	4.0	12	3	9.9	232.0	...
944	2022	Volvo	XC90 T5 AWD	8	2.0	4.0	12	3	10.1	236.0	...
945	2022	Volvo	XC90 T6 AWD	8	2.0	4.0	12	3	10.8	252.0	...

946 rows × 766 columns

In [67]:
```
final.drop(columns=["Model Year","Make","Model"],inplace=True)
```

In [68]:
```
1 final
```

Out[68]:

	Vehicle Class	ES	Cylinders	Transmission	Fuel Type	Fuel_consuption	co2	CO2 Rating	Smog Rating	Make_Ac
0	0	2.4	4.0	7	3	8.6	200.0	6.0	3	
1	7	3.5	6.0	8	3	11.2	263.0	4.0	5	
2	7	2.0	4.0	8	3	9.9	232.0	5.0	6	
3	7	2.0	4.0	8	3	10.3	242.0	5.0	6	
4	0	2.0	4.0	8	3	9.8	230.0	5.0	7	
...	...	...	...	...	...	...	...	...	...	
941	7	2.0	4.0	12	3	9.4	219.0	5.0	5	
942	7	2.0	4.0	12	3	9.4	219.0	5.0	5	
943	7	2.0	4.0	12	3	9.9	232.0	5.0	7	
944	8	2.0	4.0	12	3	10.1	236.0	5.0	5	
945	8	2.0	4.0	12	3	10.8	252.0	5.0	7	

946 rows × 763 columns

Data Prepartion

In [69]:
```
1 x = final.drop("Fuel_consuption",axis=1)
```

In [70]:
```
1 x
```

Out[70]:

	Vehicle Class	ES	Cylinders	Transmission	Fuel Type	co2	CO2 Rating	Smog Rating	Make_Acura	Make_Alfa Romeo
0	0	2.4	4.0	7	3	200.0	6.0	3	1.0	0.0
1	7	3.5	6.0	8	3	263.0	4.0	5	1.0	0.0
2	7	2.0	4.0	8	3	232.0	5.0	6	1.0	0.0
3	7	2.0	4.0	8	3	242.0	5.0	6	1.0	0.0
4	0	2.0	4.0	8	3	230.0	5.0	7	1.0	0.0
...	...	...	...	...	...	...	...	...	...	...
941	7	2.0	4.0	12	3	219.0	5.0	5	0.0	0.0
942	7	2.0	4.0	12	3	219.0	5.0	5	0.0	0.0
943	7	2.0	4.0	12	3	232.0	5.0	7	0.0	0.0
944	8	2.0	4.0	12	3	236.0	5.0	5	0.0	0.0
945	8	2.0	4.0	12	3	252.0	5.0	7	0.0	0.0

946 rows × 762 columns

In [71]:
```
1 y = final["Fuel_consuption"]
2 y
```

Out[71]:
```
0       8.6
1      11.2
2       9.9
3      10.3
4       9.8
       ...
941     9.4
942     9.4
943     9.9
944    10.1
945    10.8
Name: Fuel_consuption, Length: 946, dtype: float64
```

In [72]:
```
1 from sklearn.model_selection import train_test_split
```

In [73]:
```
1 x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_st
```

In [74]:
```
1 x_train.shape,y_train.shape
```

Out[74]:
```
((662, 762), (662,))
```

In [75]:
```
1 x_test.shape,y_test.shape
```

Out[75]: ((284, 762), (284,))

In [76]:
```
1 from sklearn.linear_model import LinearRegression
```

In [77]:
```
1 model = LinearRegression()
```

In [80]:
```
1 model.fit(x_train,y_train)
```

Out[80]: LinearRegression()

In [83]:
```
1 model.score(x_train,y_train)
```

Out[83]: 0.9832169319786306

In [84]:
```
1 model.score(x_test,y_test)
```

Out[84]: -2.484890262268565e+18

In [87]:
```
1 from sklearn.ensemble import RandomForestRegressor
```

In [88]:
```
1 Model = RandomForestRegressor()
```

In [89]:
```
1 Model.fit(x_train,y_train)
```

Out[89]: RandomForestRegressor()

In [90]:
```
1 Model.score(x_train,y_train)
```

Out[90]: 0.992881839533678

In [91]:
```
1 Model.score(x_test,y_test)
```

Out[91]: 0.9672957225369877

In [ ]:
```
1
```

In [ ]:
```
1
```