In [1]:
```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
mpl.style.use(['ggplot'])
%matplotlib inline
```

In [20]:
```
df = pd.read_csv('MY2022_Fuel_Consumption_Ratings[1].csv')
```

In [21]:
```
df.head()
```

Out[21]:

	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))	Emissions(g
0	2022	Acura	ILX	Compact	2.4	4	AM8	Z	9.9	7.0	8.6	33	
1	2022	Acura	MDX SH-AWD	SUV: Small	3.5	6	AS10	Z	12.6	9.4	11.2	25	
2	2022	Acura	RDX SH-AWD	SUV: Small	2.0	4	AS10	Z	11.0	8.6	9.9	29	
3	2022	Acura	RDX SH-AWD A-SPEC	SUV: Small	2.0	4	AS10	Z	11.3	9.1	10.3	27	
4	2022	Acura	TLX SH-AWD	Compact	2.0	4	AS10	Z	11.2	8.0	9.8	29	

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.shape
```

Out[23]:
```
(946, 15)
```

In [24]:
```
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 [26]:
```
df.isna().sum()
```

Out[26]:
```
Model Year                          0
Make                                0
Model                               0
Vehicle Class                       0
Engine Size(L)                      0
Cylinders                           0
Transmission                        0
Fuel Type                           0
Fuel Consumption (City (L/100 km)   0
Fuel Consumption(Hwy (L/100 km))    0
Fuel Consumption(Comb (L/100 km))   0
Fuel Consumption(Comb (mpg))        0
CO2 Emissions(g/km)                 0
CO2 Rating                          0
Smog Rating                         0
dtype: int64
```

In [27]:
```
df.duplicated()
```

Out[27]:
```
0      False
1      False
```

```
2      False
3      False
4      False
       ...
941    False
942    False
943    False
944    False
945    False
Length: 946, dtype: bool
```

In [29]:
```
df.describe()
```

Out[29]:

	Model Year	Engine Size(L)	Cylinders	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
count	946.0	946.000000	946.000000	946.000000	946.000000	946.000000	946.000000	946.000000	946.000000	946.000000
mean	2022.0	3.198732	5.668076	12.506448	9.363319	11.092072	27.247357	259.172304	4.539112	4.950317
std	0.0	1.374814	1.932670	3.452043	2.285125	2.876276	7.685217	64.443149	1.471799	1.679842
min	2022.0	1.200000	3.000000	4.000000	3.900000	4.000000	11.000000	94.000000	1.000000	1.000000
25%	2022.0	2.000000	4.000000	10.200000	7.700000	9.100000	22.000000	213.250000	3.000000	3.000000
50%	2022.0	3.000000	6.000000	12.200000	9.200000	10.800000	26.000000	257.000000	5.000000	5.000000
75%	2022.0	3.800000	6.000000	14.700000	10.700000	12.900000	31.000000	300.750000	5.000000	6.000000
max	2022.0	8.000000	16.000000	30.300000	20.900000	26.100000	71.000000	608.000000	10.000000	7.000000

In [31]:
```
cdf = df[['Engine Size(L)','Cylinders','Fuel Consumption(Comb (L/100 km))','CO2 Emissions(g/km)']]
cdf.head(9)
```

Out[31]:

	Engine Size(L)	Cylinders	Fuel Consumption(Comb (L/100 km))	CO2 Emissions(g/km)
0	2.4	4	8.6	200
1	3.5	6	11.2	263
2	2.0	4	9.9	232
3	2.0	4	10.3	242
4	2.0	4	9.8	230
5	2.0	4	9.8	231
6	3.0	6	11.0	256
7	3.0	6	11.2	261

8	2.0	4	8.7	205

In [32]:
```
cdf.rename(columns={'Engine Size(L)':'ENGINESIZE', 'Cylinders':'CYLINDERS','Fuel Consumption(Comb (L/100 km))':'FUEL_COM_COMB',
```

```
C:\Users\ahmed\AppData\Local\Temp\ipykernel_26568\1429791454.py:1: SettingWithCopyWarning:
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
  cdf.rename(columns={'Engine Size(L)':'ENGINESIZE', 'Cylinders':'CYLINDERS','Fuel Consumption(Comb (L/100 km))':'FUEL_COM_COMB','CO2 Emissions(g/km)':'CO2_EMISSIONS'}, inplace=True)
```

In [33]:
```
cdf.head()
```

Out[33]:

	ENGINESIZE	CYLINDERS	FUEL_COM_COMB	CO2_EMISSIONS
0	2.4	4	8.6	200
1	3.5	6	11.2	263
2	2.0	4	9.9	232
3	2.0	4	10.3	242
4	2.0	4	9.8	230

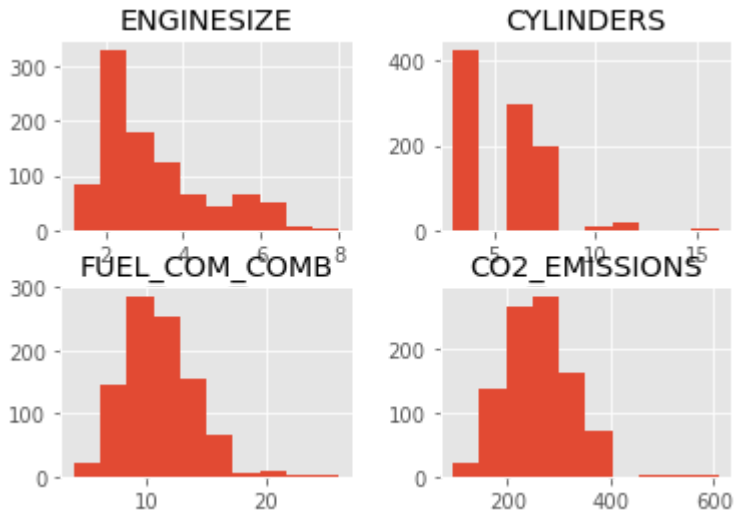
In [43]:
```
cdf.corr()
```

Out[43]:

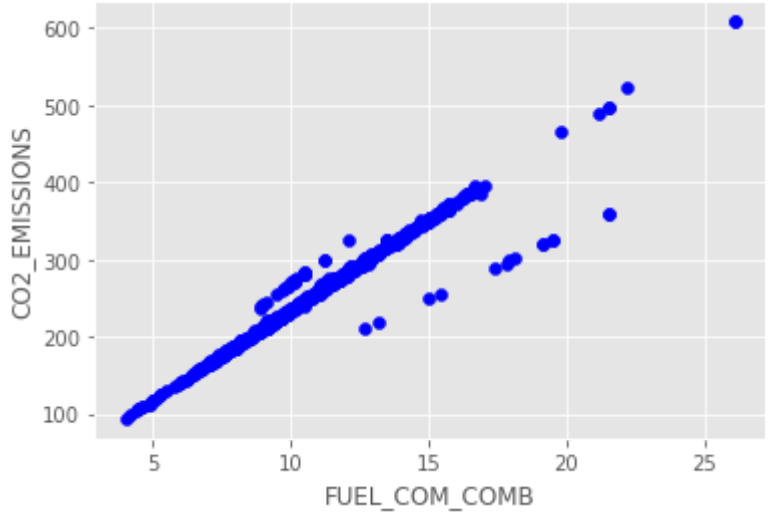
	ENGINESIZE	CYLINDERS	FUEL_COM_COMB	CO2_EMISSIONS
ENGINESIZE	1.000000	0.920698	0.818694	0.824188
CYLINDERS	0.920698	1.000000	0.821718	0.833241
FUEL_COM_COMB	0.818694	0.821718	1.000000	0.971671
CO2_EMISSIONS	0.824188	0.833241	0.971671	1.000000

In [39]:
```
cdf.hist()
```

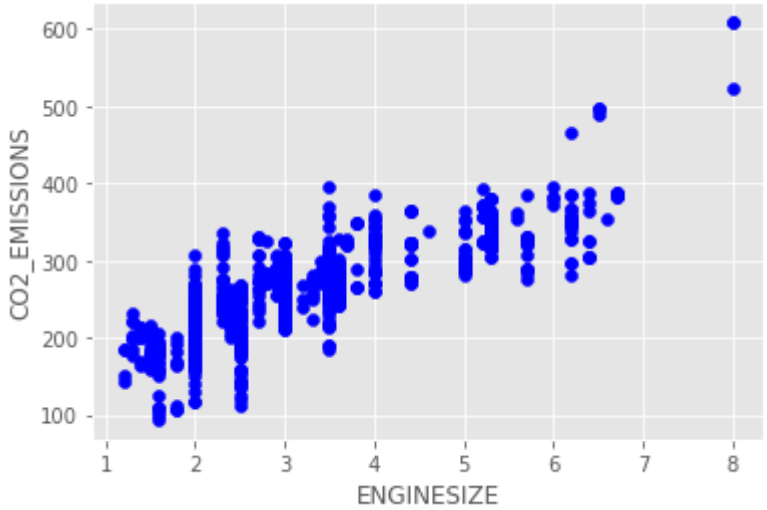
Out[39]:
```
array([[<AxesSubplot:title={'center':'ENGINESIZE'}>,
        <AxesSubplot:title={'center':'CYLINDERS'}>],
       [<AxesSubplot:title={'center':'FUEL_COM_COMB'}>,
        <AxesSubplot:title={'center':'CO2_EMISSIONS'}>]], dtype=object)
```



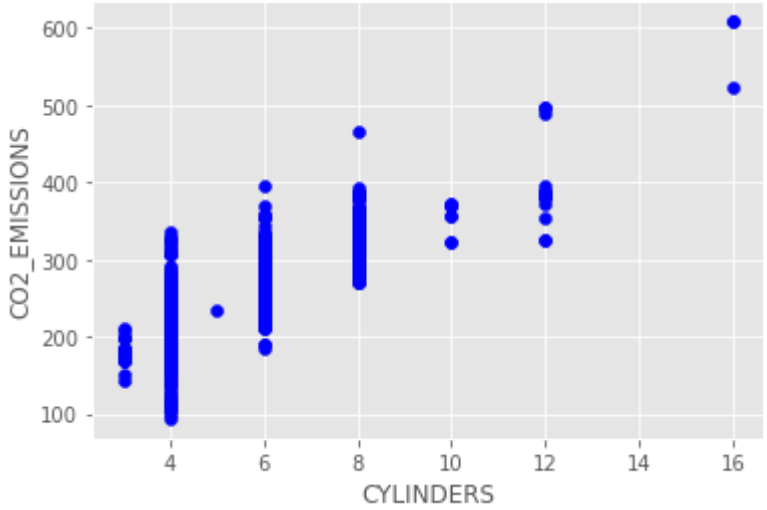
In [40]:
```
plt.scatter(cdf.FUEL_COM_COMB, cdf.CO2_EMISSIONS ,color = 'blue')
plt.xlabel('FUEL_COM_COMB')
plt.ylabel('CO2_EMISSIONS')
plt.show()
```



In [41]:
```
plt.scatter(cdf.ENGINESIZE, cdf.CO2_EMISSIONS ,color = 'blue')
plt.xlabel('ENGINESIZE')
plt.ylabel('CO2_EMISSIONS')
plt.show()
```

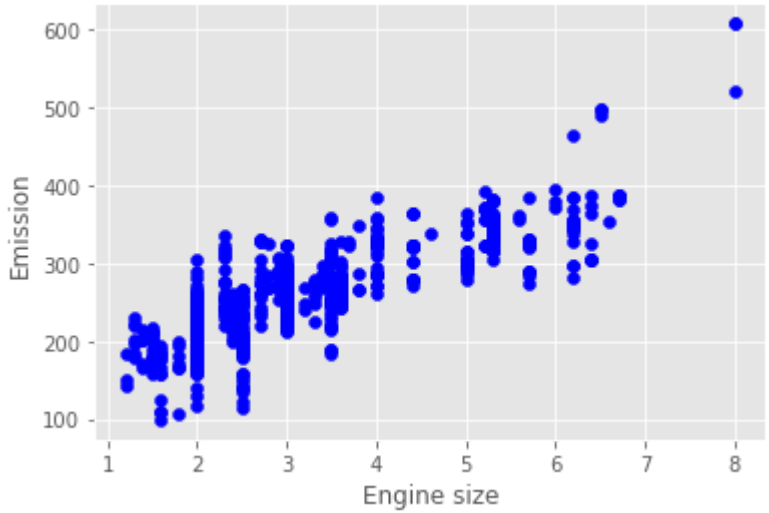


In [42]:
```
plt.scatter(cdf.CYLINDERS, cdf.CO2_EMISSIONS ,color = 'blue')
plt.xlabel('CYLINDERS')
plt.ylabel('CO2_EMISSIONS')
plt.show()
```



In [44]:
```
msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]
```

In [45]:
```
plt.scatter(train.ENGINESIZE, train.CO2_EMISSIONS,  color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```

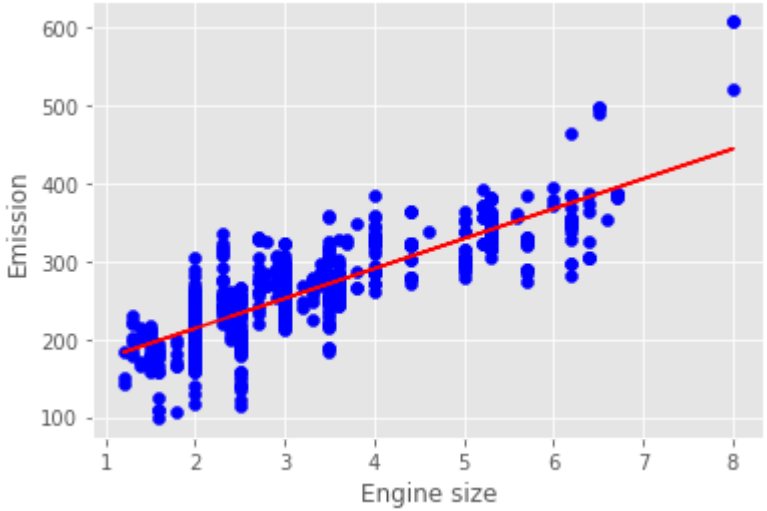


In [48]:
```
from sklearn import linear_model
reg = linear_model.LinearRegression()
train_x = np.asanyarray(train[['ENGINESIZE']])
train_y = np.asanyarray(train[['CO2_EMISSIONS']])
reg.fit(train_x,train_y)
print('Coefficients:', reg.coef_)
print('Intercept:', reg.intercept_)
```

```
Coefficients: [[38.44579138]]
Intercept: [136.80469491]
```

In [50]:
```
plt.scatter(train.ENGINESIZE, train.CO2_EMISSIONS,  color='blue')
plt.plot(train_x, reg.coef_[0][0]*train_x + reg.intercept_[0], '-r')
plt.xlabel("Engine size")
plt.ylabel("Emission")
```

Out[50]:
```
Text(0, 0.5, 'Emission')
```



In [54]:
```
from sklearn.metrics import r2_score

test_x = np.asanyarray(test[['ENGINESIZE']])
test_y = np.asanyarray(test[['CO2_EMISSIONS']])
test_y_ = reg.predict(test_x)

print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_ - test_y)))
print("Residual sum of squares (MSE): %.2f" % np.mean((test_y_ - test_y) ** 2))
print("R2-score: %.2f" % r2_score(test_y , test_y_))
```

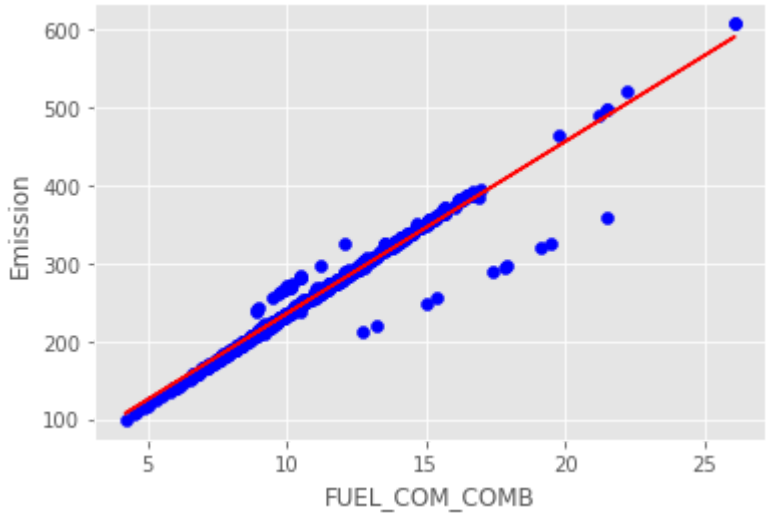
```
Mean absolute error: 27.83
Residual sum of squares (MSE): 1307.99
R2-score: 0.66
```

In [55]:
```
train_x = np.asanyarray(train[['FUEL_COM_COMB']])
train_y = np.asanyarray(train[['CO2_EMISSIONS']])
reg.fit(train_x,train_y)
print('Coefficients:', reg.coef_)
print('Intercept:', reg.intercept_)
```

```
Coefficients: [[22.09494781]]
Intercept: [14.3423615]
```

In [56]:
```
plt.scatter(train.FUEL_COM_COMB, train.CO2_EMISSIONS,  color='blue')
plt.plot(train_x, reg.coef_[0][0]*train_x + reg.intercept_[0], '-r')
plt.xlabel("FUEL_COM_COMB")
plt.ylabel("Emission")
```

Out[56]:
```
Text(0, 0.5, 'Emission')
```



In [58]:
```
from sklearn.metrics import r2_score

test_x = np.asanyarray(test[['FUEL_COM_COMB']])
test_y = np.asanyarray(test[['CO2_EMISSIONS']])
test_y_ = reg.predict(test_x)

print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_ - test_y)))
print("Residual sum of squares (MSE): %.2f" % np.mean((test_y_ - test_y) ** 2))
print("R2-score: %.2f" % r2_score(test_y , test_y_))
```

```
Mean absolute error: 6.64
Residual sum of squares (MSE): 351.86
R2-score: 0.91
```

In [ ]: