In [1]:
```
import matplotlib.pyplot as plt
import pandas as pd
import pylab as pl
import numpy as np
%matplotlib inline
```

In [2]:
```
df=pd.read_csv("FuelConsumption_0f271e04a27fdffc69a487f88705347c.csv")
df
```

Out[2]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5
1	2014	ACURA	ILX	COMPACT	2.4	4	M6
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6
...	...	...	...	...	...	...	...
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6

1067 rows × 13 columns

In [3]:
```
df.describe()
```

Out[3]:

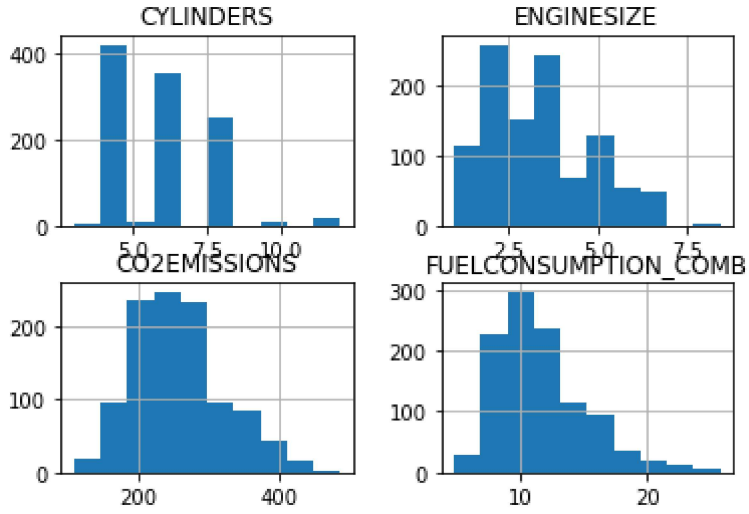
	MODELYEAR	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_CITY	FUELCONSUMPTION_HW
count	1067.0	1067.000000	1067.000000	1067.000000	1067.0000
mean	2014.0	3.346298	5.794752	13.296532	9.4746
std	0.0	1.415895	1.797447	4.101253	2.7945
min	2014.0	1.000000	3.000000	4.600000	4.9000
25%	2014.0	2.000000	4.000000	10.250000	7.5000
50%	2014.0	3.400000	6.000000	12.600000	8.8000
75%	2014.0	4.300000	8.000000	15.550000	10.8500
max	2014.0	8.400000	12.000000	30.200000	20.5000

In [4]:
```
cdf = df[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB','CO2EMISSIONS']]
cdf.head(9)
```

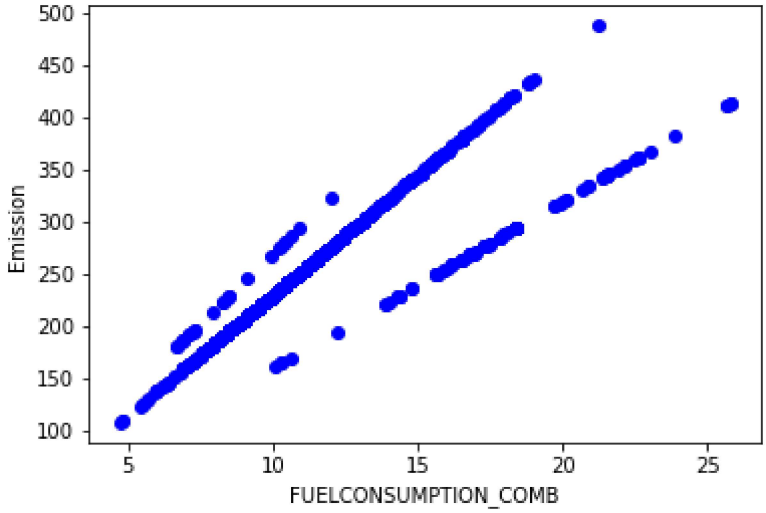
Out[4]:

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

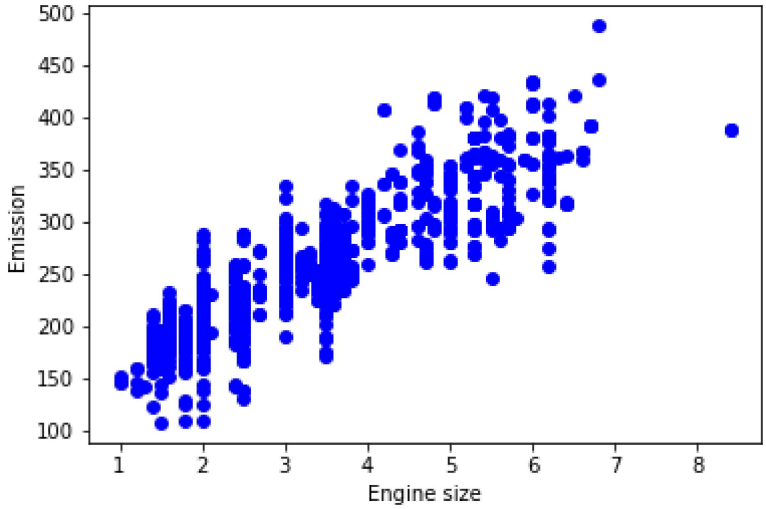
In [5]:
```
viz = cdf[['CYLINDERS','ENGINESIZE','CO2EMISSIONS','FUELCONSUMPTION_COMB']]
viz.hist()
plt.show()
```



In [6]:
```
plt.scatter(cdf.FUELCONSUMPTION_COMB, cdf.CO2EMISSIONS,  color='blue')
plt.xlabel("FUELCONSUMPTION_COMB")
plt.ylabel("Emission")
plt.show()
```

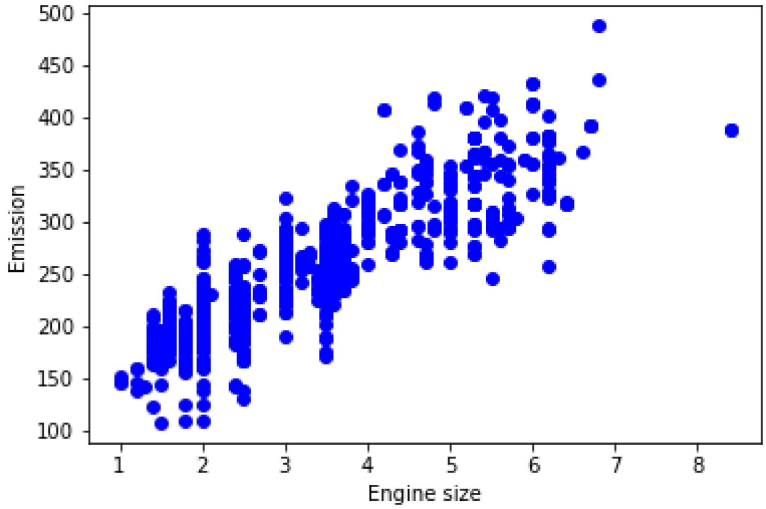


In [7]:
```
plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS,  color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```



In [8]:
```
msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]
```

In [9]:
```
plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS,  color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```

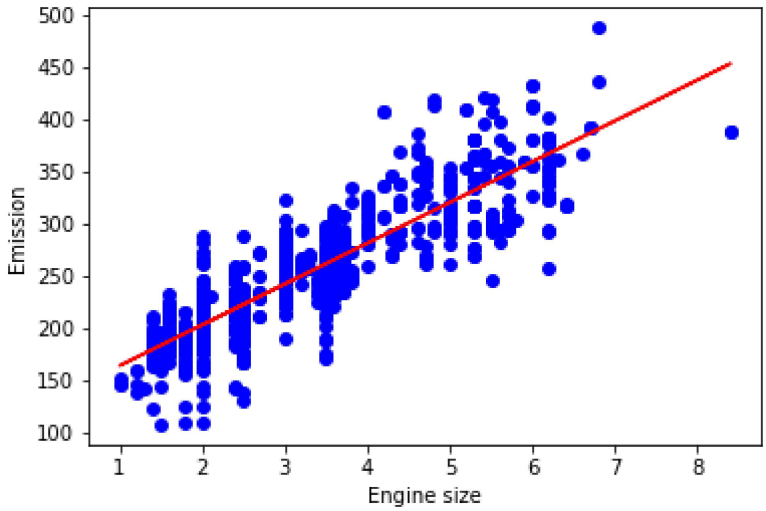


In [11]:
```
from sklearn import linear_model
regr = linear_model.LinearRegression()
train_x = np.asanyarray(train[['ENGINESIZE']])
train_y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit (train_x, train_y)
# The coefficients
print ('Coefficients: ', regr.coef_)
print ('Intercept: ',regr.intercept_)
```

```
Coefficients:  [[39.09074548]]
Intercept:  [125.18549703]
```

In [12]:
```
plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS,  color='blue')
plt.plot(train_x, regr.coef_[0][0]*train_x + regr.intercept_[0], '-r')
plt.xlabel("Engine size")
plt.ylabel("Emission")
```

Out[12]:
```
Text(0, 0.5, 'Emission')
```

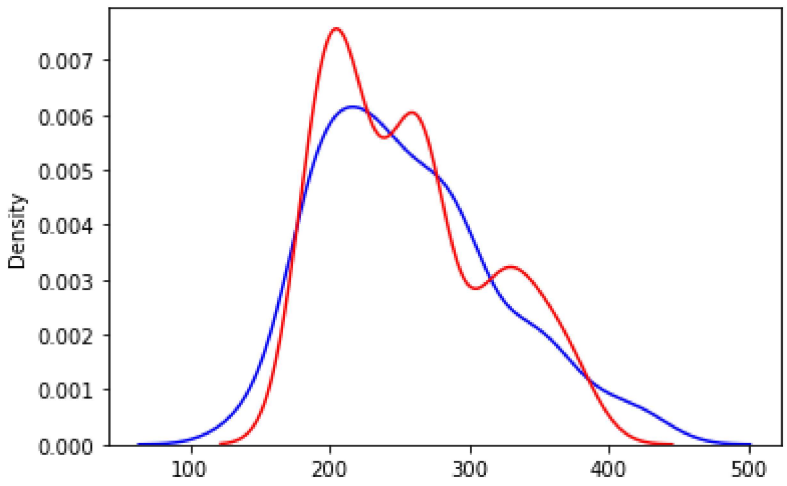


In [13]:
```
from sklearn.metrics import r2_score

test_x = np.asanyarray(test[['ENGINESIZE']])
test_y = np.asanyarray(test[['CO2EMISSIONS']])
test_y_hat = regr.predict(test_x)

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

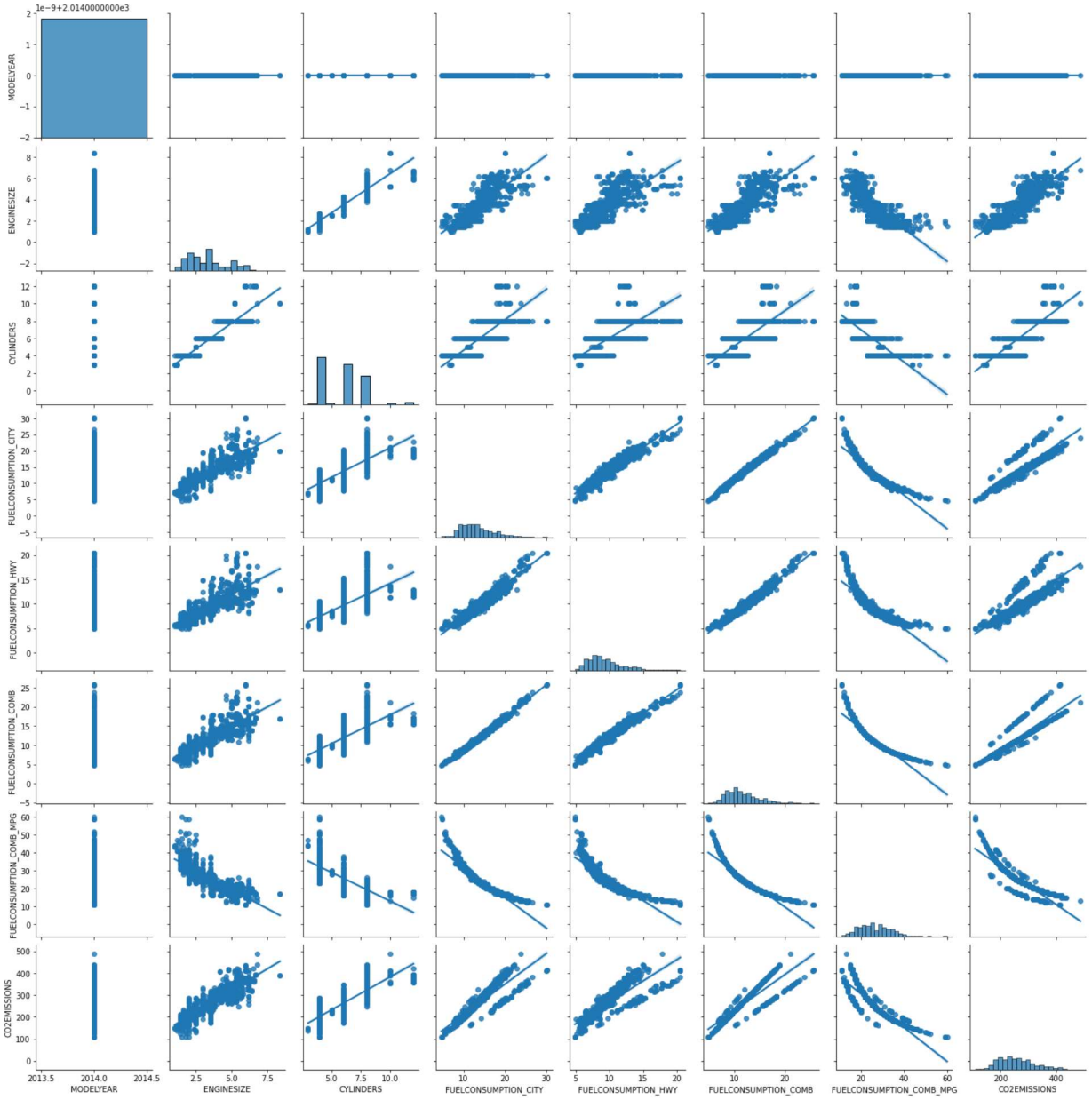
```
Mean absolute error: 22.47
Residual sum of squares (MSE): 903.44
R2-score: 0.72
```

In [14]:
```
import seaborn as sns
ax = sns.distplot(test_y, color = 'b', label="true-values", hist=False)
sns.distplot(test_y_hat, color='r', label="predicted values", hist= False , ax =
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future v
ersion. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `kdeplot` (an axes-level function for kernel densi
ty plots).
  warnings.warn(msg, FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future v
ersion. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `kdeplot` (an axes-level function for kernel densi
ty plots).
  warnings.warn(msg, FutureWarning)
```



In [19]:
```
sns.pairplot(df, kind="reg")
plt.show()
```



In [16]:
```
from sklearn import linear_model
regr = linear_model.LinearRegression()
x = np.asanyarray(train[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit (x, y)
# The coefficients
print ('Coefficients: ', regr.coef_)
```

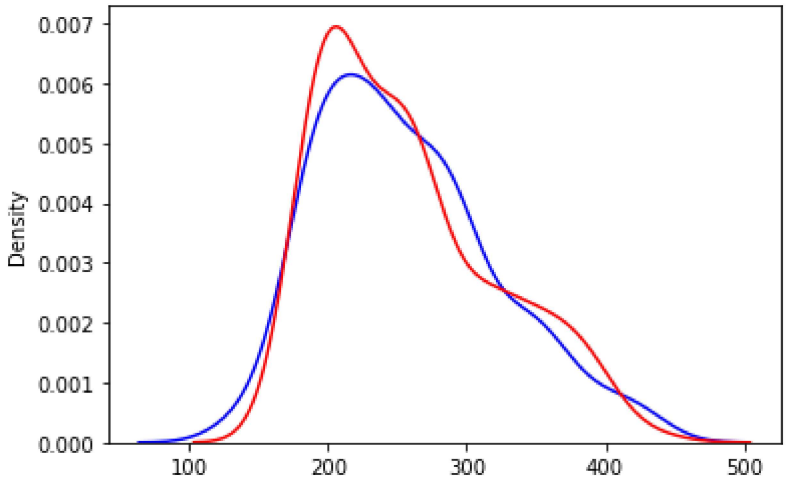
```
Coefficients:  [[10.66988927  7.62366514  9.82780484]]
```

In [20]:
```
y_hat= regr.predict(test[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
x = np.asanyarray(test[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
y = np.asanyarray(test[['CO2EMISSIONS']])
print("Residual sum of squares: %.2f"
      % np.mean((y_hat - y) ** 2))

# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % regr.score(x, y))
```

```
Residual sum of squares: 591.91
Variance score: 0.86

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X
has feature names, but LinearRegression was fitted without feature names
  warnings.warn(
```

In [21]:
```
import seaborn as sns
ax = sns.distplot(test_y, color = 'b', label="true-values", hist=False)
sns.distplot(y_hat, color='r', label="predicted values", hist= False , ax = ax)
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future v
ersion. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `kdeplot` (an axes-level function for kernel densi
ty plots).
  warnings.warn(msg, FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future v
ersion. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `kdeplot` (an axes-level function for kernel densi
ty plots).
  warnings.warn(msg, FutureWarning)
```



In [ ]: