Importing packages

In [298]:

```
import numpy as np
import pandas as pd

import sklearn
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score

%matplotlib inline
```

Loading dataset

In [299]:

```
data = pd.read_csv(r"C:\Data\data_worldbank.csv")
```

In [300]:

```
data.head()
```

Out[300]:

	Time	Country Code	AG.LND.FRST.K2	AG.PRD.LVSK.XD	EG.ELC.RNWX.KH	EG.FEC.RNEW.ZS	EG.
0	1960	The BHS	NaN	NaN	NaN	NaN	
1	1960	ABW	NaN	NaN	NaN	NaN	
2	1960	AFG	NaN	NaN	NaN	NaN	
3	1960	AGO	NaN	NaN	NaN	NaN	
4	1960	ALB	NaN	NaN	NaN	NaN	

In [301]:

```
data_copy=data.copy()
```

In [302]:

```
data_copy=data_copy[data_copy['EN.ATM.CO2E.PC'].notnull()]
```

In [303]:

```
data_copy.head()
```

Out[303]:

	Time	Country Code	AG.LND.FRST.K2	AG.PRD.LVSK.XD	EG.ELC.RNWX.KH	EG.FEC.RNEW.ZS	I
6570	1990	The BHS	5098.6	57.42	NaN	0.00	
6572	1990	AFG	12084.4	67.11	NaN	23.00	
6573	1990	AGO	792627.8	47.33	0.0	72.26	
6574	1990	ALB	7888.0	50.94	0.0	25.50	
6575	1990	AND	160.0	NaN	NaN	14.05	

In [304]:

```
data_copy=data_copy.dropna()
```

In [305]:

```
data_copy.head()
```

Out[305]:

	Time	Country Code	AG.LND.FRST.K2	AG.PRD.LVSK.XD	EG.ELC.RNWX.KH	EG.FEC.RNEW.ZS	I
6573	1990	AGO	792627.8	47.33	0.0	72.26	
6574	1990	ALB	7888.0	50.94	0.0	25.50	
6576	1990	ARE	2450.0	23.84	0.0	0.00	
6577	1990	ARG	352040.0	79.52	107000000.0	8.57	
6581	1990	AUS	1338822.0	83.78	750000000.0	8.01	

Preparing y and x variables

In [306]:

```
y=data_copy['EN.ATM.CO2E.PC']
```

In [307]:

```
x=data_copy.drop(columns=['EN.ATM.CO2E.PC'])
```

In [308]:

```
x2=x.drop(columns=['SP.URB.TOTL'])
x2=x2.drop(columns=['SP.URB.TOTL.IN.ZS'])
x2=x2.drop(columns=['Country Code'])
x2=x2.drop(columns=['Time'])
```

In [309]:
```
x2.head(10)
```

Out[309]:

	AG.LND.FRST.K2	AG.PRD.LVSK.XD	EG.ELC.RNWX.KH	EG.FEC.RNEW.ZS	EG.USE.ELEC.KH
6573	792627.8	47.33	0.000000e+00	72.26	53.260
6574	7888.0	50.94	0.000000e+00	25.50	552.252
6576	2450.0	23.84	0.000000e+00	0.00	8179.876
6577	352040.0	79.52	1.070000e+08	8.57	1303.218
6581	1338822.0	83.78	7.500000e+08	8.01	8527.214
6582	37756.7	97.58	1.126000e+09	25.17	6111.476
6586	48351.5	45.57	0.000000e+00	93.70	33.895
6588	19203.3	41.50	0.000000e+00	71.67	47.933
6589	33270.0	302.51	0.000000e+00	1.81	4758.731
6590	2.2	15.40	0.000000e+00	0.00	14972.420

In [310]:
```
x2.rename(columns={'AG.LND.FRST.K2': 'ForestArea', 'AG.PRD.LVSK.XD': 'Livestoc
                   'EG.ELC.RNWX.KH': 'ElectricityProduction', 'EG.FEC.RNEW.ZS':
                   'EG.USE.ELEC.KH.PC': 'ElectricPowerConsumption', 'EG.USE.PCA
                   'EN.ATM.CO2E.PC': 'CO2Emissions', 'NY.GDP.PCAP.KD': 'GDPperc
                   'NY.GDP.PCAP.KD.ZG': 'GDPpercapitagrowth', 'SP.POP.GROW': 'P
                   'SP.POP.TOTL': 'Populationtotal', 'SP.URB.GROW': 'Urbanpopul
```

In [311]:
```
x2.head()
```

Out[311]:

	ForestArea	Livestock	ElectricityProduction	RenewableEnergyConsumption	ElectricPowerC
6573	792627.8	47.33	0.0	72.26	
6574	7888.0	50.94	0.0	25.50	
6576	2450.0	23.84	0.0	0.00	
6577	352040.0	79.52	107000000.0	8.57	
6581	1338822.0	83.78	750000000.0	8.01	

In [312]:

```
x2.describe()
```

Out[312]:

	ForestArea	Livestock	ElectricityProduction	RenewableEnergyConsumption	ElectricP
count	3.151000e+03	3151.000000	3.151000e+03	3151.000000	
mean	3.006112e+05	80.845519	3.389366e+09	30.042793	
std	9.475031e+05	27.826392	1.554241e+10	29.062494	
min	0.000000e+00	9.510000	0.000000e+00	0.000000	
25%	9.599980e+03	62.105000	0.000000e+00	4.755000	
50%	4.127000e+04	84.090000	2.500000e+07	20.130000	
75%	1.819695e+05	96.800000	9.300000e+08	50.285000	
max	8.151356e+06	447.470000	2.980230e+11	98.340000	

In [313]:

```
y.describe()
```

Out[313]:

```
count    3151.000000
mean        5.152218
std         5.847699
min         0.025247
25%         0.964062
50%         3.295402
75%         7.597563
max        47.656962
Name: EN.ATM.CO2E.PC, dtype: float64
```

In [314]:

```
y.head(10)
```

Out[314]:

```
6573     0.554941
6574     1.844035
6576    29.055796
6577     3.073563
6581    15.437183
6582     7.590107
6586     0.064306
6588     0.107550
6589     8.442677
6590    20.752003
Name: EN.ATM.CO2E.PC, dtype: float64
```

Try out different subsets of x-variables

In [ ]:

```
# use x2
```

In [315]:

```
x_Model1=x2.drop(columns=['ElectricityProduction'])
x_Model1=x_Model1.drop(columns=['EnergyUse'])
x_Model1=x_Model1.drop(columns=['GDPpercapitagrowth'])
x_Model1=x_Model1.drop(columns=['Populationgrowth'])
x_Model1=x_Model1.drop(columns=['Populationtotal'])
```

Fit the models

In [316]:

```
X_train, X_test, y_train, y_test=train_test_split(x2, y, test_size=0.2, random
```

In [317]:

```
dtree = tree.DecisionTreeRegressor(max_depth=3)
```

In [318]:

```
dtree.fit(X_train, y_train)
```

Out[318]:

```
▾       DecisionTreeRegressor

DecisionTreeRegressor(max_depth=3)
```

In [319]:

```
y_pred = dtree.predict(X_test)

y_pred
```

Out[319]:

```
array([ 0.70770919,  0.70770919,  0.70770919, 17.24661065,  9.04282305,
        6.14528579,  0.70770919,  9.04282305,  9.04282305,  2.20973085,
        9.04282305,  3.94629937,  6.14528579,  0.70770919,  0.70770919,
       17.24661065,  6.14528579,  3.94629937,  0.70770919,  0.70770919,
        9.04282305,  9.04282305,  9.04282305,  0.70770919,  6.14528579,
        0.70770919,  3.94629937,  0.70770919,  9.04282305,  9.04282305,
        6.14528579,  6.14528579,  0.70770919,  0.70770919,  9.04282305,
        3.94629937,  0.70770919,  0.70770919,  0.70770919,  3.94629937,
        2.20973085,  0.70770919,  3.94629937,  0.70770919,  0.70770919,
        0.70770919,  6.14528579,  6.14528579,  0.70770919, 17.24661065,
       14.51671968,  9.04282305,  9.04282305, 17.24661065,  3.94629937,
        2.20973085,  2.20973085,  9.04282305,  0.70770919,  0.70770919,
        0.70770919,  0.70770919,  3.94629937,  0.70770919,  0.70770919,
       17.24661065,  9.04282305,  9.04282305,  9.04282305,  0.70770919,
        0.70770919,  9.04282305,  0.70770919,  0.70770919,  2.20973085,
        3.94629937, 14.51671968,  0.70770919,  0.70770919, 14.51671968,
        0.70770919,  3.94629937,  0.70770919,  0.70770919, 17.24661065,
        0.70770919,  0.70770919, 17.24661065,  0.70770919, 17.24661065,
        0.70770919,  0.70770919,  6.14528579, 17.24661065,  0.70770919,
        9.04282305, 17.24661065,  2.20973085,  0.70770919,  0.70770919,
```

In [320]:
```
y_test
```

Out[320]:
```
6845      1.472945
7800      0.075194
9223      0.936961
8877     19.611964
10245     8.445471
            ...
6676      5.777465
7481      2.521119
7291      5.913505
7905      1.071724
12018     0.550493
Name: EN.ATM.CO2E.PC, Length: 631, dtype: float64
```

In [321]:
```
r2_score(y_test, y_pred)
```

Out[321]:
```
0.8775090474487776
```

In [322]:
```
regr=RandomForestRegressor(max_depth=2, random_state=0, n_estimators=100)
```

In [323]:
```
regr.fit(X_train, y_train)
```

Out[323]:
```
▾               RandomForestRegressor

RandomForestRegressor(max_depth=2, random_state=0)
```

In [324]:
```
y_pred = regr.predict(X_test)
r2_score(y_test, y_pred)
```

Out[324]:
```
0.8010932489155408
```

In [325]:
```
reg=LinearRegression().fit(X_train,y_train)
y_pred = reg.predict(X_test)
r2_score(y_test, y_pred)
```

Out[325]:
```
0.8966108524255035
```

In [326]:
```
print("Coefficients: \n", reg.coef_)
```

```
Coefficients:
 [ 2.86596448e-07 -1.55728419e-03  6.63904936e-12 -4.01387334e-02
 -3.19441602e-04  1.97138161e-03  5.95099567e-05 -5.63591116e-03
  2.87521049e-01 -2.10848991e-10 -5.43830010e-02]
```

In [ ]:
```
### --> Linear regression has highest R²
```

In [ ]:
```

```

In [327]:
```
X_train, X_test, y_train, y_test=train_test_split(x_Model1, y, test_size=0.2,
```

In [328]:
```
dtree = tree.DecisionTreeRegressor(max_depth=3)
```

In [329]:
```
dtree.fit(X_train, y_train)
```

Out[329]:
```
▾       DecisionTreeRegressor

DecisionTreeRegressor(max_depth=3)
```

In [330]:
```
y_pred = dtree.predict(X_test)

y_pred
```

Out[330]:
```
array([ 3.04217241,  0.87043781,  0.87043781,  8.90788524,  8.90788524,
        8.90788524,  0.87043781,  8.90788524,  8.90788524,  3.04217241,
        8.90788524,  4.571998  ,  4.571998  ,  3.04217241,  0.87043781,
       17.67891463,  3.04217241,  4.571998  ,  0.87043781,  0.87043781,
        8.90788524,  8.90788524,  8.90788524,  0.87043781,  4.571998  ,
        0.87043781,  4.571998  ,  0.87043781,  8.90788524,  8.90788524,
        8.90788524,  4.571998  ,  0.87043781,  0.87043781,  8.90788524,
        4.571998  ,  3.04217241,  3.04217241,  0.87043781,  4.571998  ,
        3.04217241,  0.87043781,  3.04217241,  3.04217241,  0.87043781,
        0.87043781,  0.87043781,  3.04217241,  0.87043781,  8.90788524,
       15.86764957,  8.90788524,  8.90788524, 17.67891463,  8.90788524,
        3.04217241,  3.04217241,  8.90788524,  0.87043781,  0.87043781,
        0.87043781,  0.87043781,  3.04217241,  0.87043781,  0.87043781,
       15.86764957,  8.90788524,  7.78163861,  8.90788524,  3.04217241,
        0.87043781,  8.90788524,  0.87043781,  0.87043781,  0.87043781,
       17.67891463, 15.86764957,  0.87043781,  0.87043781, 15.86764957,
        0.87043781,  3.04217241,  0.87043781,  0.87043781,  8.90788524,
        0.87043781,  3.04217241, 15.86764957,  0.87043781, 17.67891463,
        3.04217241,  0.87043781,  4.571998  ,  8.90788524,  0.87043781,
        7.78163861, 17.67891463,  0.87043781,  0.87043781,  0.87043781,
```

In [331]:
```
y_test
```

Out[331]:
```
6845      1.472945
7800      0.075194
9223      0.936961
8877     19.611964
10245     8.445471
           ...
6676      5.777465
7481      2.521119
7291      5.913505
7905      1.071724
12018     0.550493
Name: EN.ATM.CO2E.PC, Length: 631, dtype: float64
```

In [332]:
```
r2_score(y_test, y_pred)
```

Out[332]:
```
0.8673567619846418
```

In [333]:
```
regr=RandomForestRegressor(max_depth=3, random_state=0, n_estimators=100)
```

```
In [334]: regr.fit(X_train, y_train)
```

```
Out[334]: ▾               RandomForestRegressor
          RandomForestRegressor(max_depth=3, random_state=0)
```

```
In [335]: y_pred = regr.predict(X_test)
          r2_score(y_test, y_pred)
```

```
Out[335]: 0.8823676766967923
```

```
In [336]: reg=LinearRegression().fit(X_train,y_train)
          y_pred = reg.predict(X_test)
          r2_score(y_test, y_pred)
```

```
Out[336]: 0.6735705888265976
```

```
In [337]: print("Coefficients: \n", reg.coef_)
```

```
Coefficients: 
 [ 5.15551295e-07 -7.00922145e-03 -8.73209191e-02  4.27836301e-04
  9.45539951e-05  5.71043096e-01]
```

```
In [ ]: ### --> Random Forest has highest R²
```