ASSIGNMENT 1

Co2 Emission Prediction

By Debashis Garai

UID - 18BCS6054

The steps used are:

1. Exploration and Understanding of Data
2. Data Preparation
3. Model Building using Lasso

STEP 1: Exploration and Understanding of data

In this step, we usually import the necessary libraries and read the .csv files needed for the building the model

In [ ]:
```
# importing the libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import r2_score

# supress warnings

import warnings
warnings.filterwarnings('ignore')
```

In [ ]:
```
# reading csv 1

d1 = pd.read_csv('/content/cars_trucks_and_buses_per_1000_persons.csv')
d1
```

Out[308]:

	geo	2002	2003	2004	2005	2006	2007
0	Afghanistan	NaN	NaN	NaN	NaN	NaN	22.8
1	Albania	73.0	NaN	85.0	87.5	97.3	102.0
2	Algeria	NaN	88.0	89.0	91.0	NaN	NaN
3	Angola	NaN	NaN	NaN	NaN	NaN	39.6
4	Argentina	NaN	NaN	NaN	NaN	NaN	314.0
...	...	...	...	...	...	...	...
152	Venezuela	NaN	NaN	NaN	NaN	NaN	147.0
153	Vietnam	8.0	NaN	NaN	NaN	NaN	13.5
154	Yemen	NaN	NaN	NaN	NaN	NaN	34.8
155	Zambia	NaN	NaN	NaN	NaN	NaN	17.7
156	Zimbabwe	NaN	NaN	NaN	NaN	NaN	106.0

157 rows × 7 columns

In [ ]:
```
d1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 157 entries, 0 to 156
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     157 non-null    object 
 1   2002    80 non-null     float64
 2   2003    75 non-null     float64
 3   2004    62 non-null     float64
 4   2005    50 non-null     float64
 5   2006    37 non-null     float64
 6   2007    145 non-null    float64
dtypes: float64(6), object(1)
memory usage: 8.7+ KB
```

In [ ]:
```
# reading csv 2

d2 = pd.read_csv('/content/co2_emissions_tonnes_per_person.csv')
d2
```

Out[310]:

	geo	1800	1801	1802	1803	1804	1805	1806	1807	1808	1809	1810	1811	1812
0	Afghanistan	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Albania	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	Algeria	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	Andorra	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	Angola	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
187	Venezuela	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
188	Vietnam	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
189	Yemen	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
190	Zambia	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
191	Zimbabwe	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

192 rows × 216 columns

In [ ]:
```
# reading csv 3

d3 = pd.read_csv('/content/coal_consumption_per_cap.csv')
d3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65 entries, 0 to 64
Data columns (total 53 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     65 non-null     object 
 1   1965    57 non-null     float64
 2   1966    57 non-null     float64
 3   1967    57 non-null     float64
 4   1968    57 non-null     float64
 5   1969    57 non-null     float64
 6   1970    57 non-null     float64
 7   1971    57 non-null     float64
 8   1972    57 non-null     float64
 9   1973    57 non-null     float64
 10  1974    57 non-null     float64
 11  1975    57 non-null     float64
 12  1976    57 non-null     float64
 13  1977    57 non-null     float64
 14  1978    57 non-null     float64
 15  1979    57 non-null     float64
 16  1980    57 non-null     float64
 17  1981    57 non-null     float64
 18  1982    57 non-null     float64
 19  1983    57 non-null     float64
 20  1984    57 non-null     float64
 21  1985    65 non-null     float64
 22  1986    65 non-null     float64
 23  1987    65 non-null     float64
 24  1988    65 non-null     float64
 25  1989    65 non-null     float64
 26  1990    65 non-null     float64
 27  1991    65 non-null     float64
 28  1992    65 non-null     float64
 29  1993    65 non-null     float64
 30  1994    65 non-null     float64
 31  1995    65 non-null     float64
 32  1996    65 non-null     float64
 33  1997    65 non-null     float64
 34  1998    65 non-null     float64
 35  1999    65 non-null     float64
 36  2000    65 non-null     float64
 37  2001    65 non-null     float64
 38  2002    65 non-null     float64
 39  2003    65 non-null     float64
 40  2004    65 non-null     float64
 41  2005    65 non-null     float64
 42  2006    65 non-null     float64
 43  2007    65 non-null     float64
 44  2008    65 non-null     float64
 45  2009    65 non-null     float64
```

```
 46  2010    65 non-null     float64
 47  2011    65 non-null     float64
 48  2012    65 non-null     float64
 49  2013    65 non-null     float64
 50  2014    65 non-null     float64
 51  2015    65 non-null     float64
 52  2016    65 non-null     float64
dtypes: float64(52), object(1)
memory usage: 27.0+ KB
```

In [ ]:
```
# reading csv 4

d4 = pd.read_csv('/content/electricity_generation_per_person.csv')
d4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65 entries, 0 to 64
Data columns (total 33 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   geo     65 non-null     object
 1   1985    65 non-null     float64
 2   1986    65 non-null     float64
 3   1987    65 non-null     float64
 4   1988    65 non-null     float64
 5   1989    65 non-null     float64
 6   1990    65 non-null     float64
 7   1991    65 non-null     float64
 8   1992    65 non-null     int64
 9   1993    65 non-null     int64
 10  1994    65 non-null     float64
 11  1995    65 non-null     int64
 12  1996    65 non-null     float64
 13  1997    65 non-null     float64
 14  1998    65 non-null     int64
 15  1999    65 non-null     int64
 16  2000    65 non-null     int64
 17  2001    65 non-null     int64
 18  2002    65 non-null     int64
 19  2003    65 non-null     int64
 20  2004    65 non-null     int64
 21  2005    65 non-null     int64
 22  2006    65 non-null     int64
 23  2007    65 non-null     int64
 24  2008    65 non-null     int64
 25  2009    65 non-null     int64
 26  2010    65 non-null     int64
 27  2011    65 non-null     int64
 28  2012    65 non-null     int64
 29  2013    65 non-null     int64
 30  2014    65 non-null     int64
 31  2015    65 non-null     int64
 32  2016    65 non-null     int64
dtypes: float64(10), int64(22), object(1)
memory usage: 16.9+ KB
```

In [ ]:
```
# reading csv 5

d5 = pd.read_csv('/content/electricity_use_per_person.csv')
d5.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 138 entries, 0 to 137
Data columns (total 56 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     138 non-null    object 
 1   1960    25 non-null     float64
 2   1961    25 non-null     float64
 3   1962    25 non-null     float64
 4   1963    25 non-null     float64
 5   1964    25 non-null     float64
 6   1965    26 non-null     float64
 7   1966    26 non-null     float64
 8   1967    26 non-null     float64
 9   1968    26 non-null     float64
 10  1969    26 non-null     float64
 11  1970    26 non-null     float64
 12  1971    109 non-null    float64
 13  1972    109 non-null    float64
 14  1973    109 non-null    float64
 15  1974    109 non-null    float64
 16  1975    109 non-null    float64
 17  1976    109 non-null    float64
 18  1977    109 non-null    float64
 19  1978    109 non-null    float64
 20  1979    109 non-null    float64
 21  1980    109 non-null    float64
 22  1981    110 non-null    float64
 23  1982    110 non-null    float64
 24  1983    110 non-null    float64
 25  1984    110 non-null    float64
 26  1985    111 non-null    float64
 27  1986    111 non-null    float64
 28  1987    111 non-null    float64
 29  1988    111 non-null    float64
 30  1989    111 non-null    float64
 31  1990    131 non-null    float64
 32  1991    132 non-null    float64
 33  1992    132 non-null    float64
 34  1993    132 non-null    float64
 35  1994    132 non-null    float64
 36  1995    134 non-null    float64
 37  1996    134 non-null    float64
 38  1997    134 non-null    float64
 39  1998    134 non-null    float64
 40  1999    134 non-null    float64
 41  2000    136 non-null    float64
 42  2001    136 non-null    float64
 43  2002    136 non-null    float64
 44  2003    136 non-null    float64
 45  2004    136 non-null    float64
```

```
 46  2005    137 non-null    float64
 47  2006    137 non-null    float64
 48  2007    137 non-null    float64
 49  2008    137 non-null    float64
 50  2009    137 non-null    float64
 51  2010    137 non-null    float64
 52  2011    137 non-null    float64
 53  2012    137 non-null    float64
 54  2013    137 non-null    float64
 55  2014    137 non-null    float64
dtypes: float64(55), object(1)
memory usage: 60.5+ KB
```

In [ ]:
```
# reading csv 6

d6 = pd.read_csv('/content/forest_coverage_percent.csv')
d6.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 192 entries, 0 to 191
Data columns (total 27 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     192 non-null    object 
 1   1990    162 non-null    float64
 2   1991    165 non-null    float64
 3   1992    184 non-null    float64
 4   1993    188 non-null    float64
 5   1994    188 non-null    float64
 6   1995    188 non-null    float64
 7   1996    188 non-null    float64
 8   1997    188 non-null    float64
 9   1998    188 non-null    float64
 10  1999    188 non-null    float64
 11  2000    190 non-null    float64
 12  2001    190 non-null    float64
 13  2002    190 non-null    float64
 14  2003    190 non-null    float64
 15  2004    190 non-null    float64
 16  2005    190 non-null    float64
 17  2006    192 non-null    float64
 18  2007    192 non-null    float64
 19  2008    192 non-null    float64
 20  2009    192 non-null    float64
 21  2010    192 non-null    float64
 22  2011    191 non-null    float64
 23  2012    191 non-null    float64
 24  2013    191 non-null    float64
 25  2014    191 non-null    float64
 26  2015    191 non-null    float64
dtypes: float64(26), object(1)
memory usage: 40.6+ KB
```

In [ ]:
```
# reading csv 7

d7 = pd.read_csv('/content/hydro_power_generation_per_person.csv')
d7.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 118 entries, 0 to 117
Data columns (total 53 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     118 non-null    object 
 1   1960    24 non-null     float64
 2   1961    24 non-null     float64
 3   1962    24 non-null     float64
 4   1963    24 non-null     float64
 5   1964    24 non-null     float64
 6   1965    25 non-null     float64
 7   1966    25 non-null     float64
 8   1967    25 non-null     float64
 9   1968    25 non-null     float64
 10  1969    25 non-null     float64
 11  1970    25 non-null     float64
 12  1971    88 non-null     float64
 13  1972    89 non-null     float64
 14  1973    90 non-null     float64
 15  1974    90 non-null     float64
 16  1975    90 non-null     float64
 17  1976    90 non-null     float64
 18  1977    90 non-null     float64
 19  1978    90 non-null     float64
 20  1979    90 non-null     float64
 21  1980    90 non-null     float64
 22  1981    91 non-null     float64
 23  1982    91 non-null     float64
 24  1983    91 non-null     float64
 25  1984    91 non-null     float64
 26  1985    92 non-null     float64
 27  1986    93 non-null     float64
 28  1987    93 non-null     float64
 29  1988    93 non-null     float64
 30  1989    93 non-null     float64
 31  1990    112 non-null    float64
 32  1991    113 non-null    float64
 33  1992    114 non-null    float64
 34  1993    114 non-null    float64
 35  1994    114 non-null    float64
 36  1995    114 non-null    float64
 37  1996    114 non-null    float64
 38  1997    115 non-null    float64
 39  1998    115 non-null    float64
 40  1999    115 non-null    float64
 41  2000    115 non-null    float64
 42  2001    115 non-null    float64
 43  2002    117 non-null    float64
 44  2003    117 non-null    float64
 45  2004    117 non-null    float64
```

```
 46  2005    118 non-null    float64
 47  2006    117 non-null    float64
 48  2007    117 non-null    float64
 49  2008    117 non-null    float64
 50  2009    117 non-null    float64
 51  2010    117 non-null    float64
 52  2011    34 non-null     float64
dtypes: float64(52), object(1)
memory usage: 49.0+ KB
```

In [ ]:

```
# reading csv 8

d8 = pd.read_csv('/content/income_per_person_gdppercapita_ppp_inflation_adjusted
d8.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 193 entries, 0 to 192
Columns: 220 entries, geo to 2018
dtypes: int64(219), object(1)
memory usage: 331.8+ KB
```

In [ ]:
```
#  reading csv 9

d9 = pd.read_csv('/content/industry_percent_of_gdp.csv')
d9.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 189 entries, 0 to 188
Data columns (total 59 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     189 non-null    object 
 1   1960    40 non-null     float64
 2   1961    42 non-null     float64
 3   1962    42 non-null     float64
 4   1963    44 non-null     float64
 5   1964    46 non-null     float64
 6   1965    64 non-null     float64
 7   1966    67 non-null     float64
 8   1967    68 non-null     float64
 9   1968    71 non-null     float64
 10  1969    72 non-null     float64
 11  1970    80 non-null     float64
 12  1971    81 non-null     float64
 13  1972    81 non-null     float64
 14  1973    81 non-null     float64
 15  1974    82 non-null     float64
 16  1975    86 non-null     float64
 17  1976    89 non-null     float64
 18  1977    95 non-null     float64
 19  1978    97 non-null     float64
 20  1979    97 non-null     float64
 21  1980    106 non-null    float64
 22  1981    109 non-null    float64
 23  1982    110 non-null    float64
 24  1983    112 non-null    float64
 25  1984    112 non-null    float64
 26  1985    116 non-null    float64
 27  1986    117 non-null    float64
 28  1987    119 non-null    float64
 29  1988    119 non-null    float64
 30  1989    122 non-null    float64
 31  1990    136 non-null    float64
 32  1991    140 non-null    float64
 33  1992    142 non-null    float64
 34  1993    145 non-null    float64
 35  1994    151 non-null    float64
 36  1995    170 non-null    float64
 37  1996    169 non-null    float64
 38  1997    171 non-null    float64
 39  1998    172 non-null    float64
 40  1999    172 non-null    float64
 41  2000    177 non-null    float64
 42  2001    178 non-null    float64
 43  2002    179 non-null    float64
 44  2003    179 non-null    float64
 45  2004    181 non-null    float64
```

```
 46  2005    180 non-null    float64
 47  2006    182 non-null    float64
 48  2007    184 non-null    float64
 49  2008    183 non-null    float64
 50  2009    182 non-null    float64
 51  2010    182 non-null    float64
 52  2011    182 non-null    float64
 53  2012    182 non-null    float64
 54  2013    184 non-null    float64
 55  2014    183 non-null    float64
 56  2015    180 non-null    float64
 57  2016    172 non-null    float64
 58  2017    142 non-null    float64
dtypes: float64(58), object(1)
memory usage: 87.2+ KB
```

In [ ]:
```
# reading csv 10

d10 = pd.read_csv('/content/natural_gas_production_per_person.csv')
d10.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49 entries, 0 to 48
Data columns (total 48 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     49 non-null     object 
 1   1970    43 non-null     float64
 2   1971    43 non-null     float64
 3   1972    43 non-null     float64
 4   1973    43 non-null     float64
 5   1974    43 non-null     float64
 6   1975    43 non-null     float64
 7   1976    43 non-null     float64
 8   1977    43 non-null     float64
 9   1978    43 non-null     float64
 10  1979    43 non-null     float64
 11  1980    43 non-null     float64
 12  1981    43 non-null     float64
 13  1982    43 non-null     float64
 14  1983    43 non-null     float64
 15  1984    43 non-null     float64
 16  1985    49 non-null     float64
 17  1986    49 non-null     float64
 18  1987    49 non-null     float64
 19  1988    49 non-null     float64
 20  1989    49 non-null     float64
 21  1990    49 non-null     float64
 22  1991    49 non-null     float64
 23  1992    49 non-null     float64
 24  1993    49 non-null     float64
 25  1994    49 non-null     float64
 26  1995    49 non-null     float64
 27  1996    49 non-null     float64
 28  1997    49 non-null     float64
 29  1998    49 non-null     float64
 30  1999    49 non-null     float64
 31  2000    49 non-null     float64
 32  2001    49 non-null     float64
 33  2002    49 non-null     float64
 34  2003    49 non-null     float64
 35  2004    49 non-null     float64
 36  2005    49 non-null     float64
 37  2006    49 non-null     float64
 38  2007    49 non-null     float64
 39  2008    49 non-null     float64
 40  2009    49 non-null     float64
 41  2010    49 non-null     float64
 42  2011    49 non-null     float64
 43  2012    49 non-null     float64
 44  2013    49 non-null     float64
 45  2014    49 non-null     float64
```

```
 46  2015    49 non-null     float64
 47  2016    49 non-null     float64
dtypes: float64(47), object(1)
memory usage: 18.5+ KB
```

In [ ]:
```
# reading csv 11

d11 = pd.read_csv('/content/oil_consumption_per_cap.csv')
d11.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65 entries, 0 to 64
Data columns (total 53 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     65 non-null     object 
 1   1965    56 non-null     float64
 2   1966    56 non-null     float64
 3   1967    56 non-null     float64
 4   1968    56 non-null     float64
 5   1969    56 non-null     float64
 6   1970    56 non-null     float64
 7   1971    57 non-null     float64
 8   1972    57 non-null     float64
 9   1973    57 non-null     float64
 10  1974    57 non-null     float64
 11  1975    57 non-null     float64
 12  1976    57 non-null     float64
 13  1977    57 non-null     float64
 14  1978    57 non-null     float64
 15  1979    57 non-null     float64
 16  1980    57 non-null     float64
 17  1981    57 non-null     float64
 18  1982    57 non-null     float64
 19  1983    57 non-null     float64
 20  1984    57 non-null     float64
 21  1985    65 non-null     float64
 22  1986    65 non-null     float64
 23  1987    65 non-null     float64
 24  1988    65 non-null     float64
 25  1989    65 non-null     float64
 26  1990    65 non-null     float64
 27  1991    65 non-null     float64
 28  1992    65 non-null     float64
 29  1993    65 non-null     float64
 30  1994    65 non-null     float64
 31  1995    65 non-null     float64
 32  1996    65 non-null     float64
 33  1997    65 non-null     float64
 34  1998    65 non-null     float64
 35  1999    65 non-null     float64
 36  2000    65 non-null     float64
 37  2001    65 non-null     float64
 38  2002    65 non-null     float64
 39  2003    65 non-null     float64
 40  2004    65 non-null     float64
 41  2005    65 non-null     float64
 42  2006    65 non-null     float64
 43  2007    65 non-null     float64
 44  2008    65 non-null     float64
 45  2009    65 non-null     float64
```

```
 46  2010    65 non-null     float64
 47  2011    65 non-null     float64
 48  2012    65 non-null     float64
 49  2013    65 non-null     float64
 50  2014    65 non-null     float64
 51  2015    65 non-null     float64
 52  2016    65 non-null     float64
dtypes: float64(52), object(1)
memory usage: 27.0+ KB
```

In [ ]:
```
#reading csv 12

d12 = pd.read_csv('/content/oil_production_per_person.csv')
d12.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49 entries, 0 to 48
Data columns (total 53 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   geo     49 non-null     object 
 1   1965    32 non-null     float64
 2   1966    33 non-null     float64
 3   1967    34 non-null     float64
 4   1968    35 non-null     float64
 5   1969    35 non-null     float64
 6   1970    35 non-null     float64
 7   1971    36 non-null     float64
 8   1972    37 non-null     float64
 9   1973    37 non-null     float64
 10  1974    37 non-null     float64
 11  1975    37 non-null     float64
 12  1976    37 non-null     float64
 13  1977    37 non-null     float64
 14  1978    37 non-null     float64
 15  1979    37 non-null     float64
 16  1980    37 non-null     float64
 17  1981    38 non-null     float64
 18  1982    38 non-null     float64
 19  1983    38 non-null     float64
 20  1984    38 non-null     float64
 21  1985    43 non-null     float64
 22  1986    44 non-null     float64
 23  1987    45 non-null     float64
 24  1988    45 non-null     float64
 25  1989    45 non-null     float64
 26  1990    45 non-null     float64
 27  1991    45 non-null     float64
 28  1992    46 non-null     float64
 29  1993    47 non-null     float64
 30  1994    47 non-null     float64
 31  1995    47 non-null     float64
 32  1996    47 non-null     float64
 33  1997    47 non-null     float64
 34  1998    47 non-null     float64
 35  1999    47 non-null     float64
 36  2000    47 non-null     float64
 37  2001    47 non-null     float64
 38  2002    47 non-null     float64
 39  2003    48 non-null     float64
 40  2004    48 non-null     float64
 41  2005    48 non-null     float64
 42  2006    48 non-null     float64
 43  2007    48 non-null     float64
 44  2008    48 non-null     float64
 45  2009    48 non-null     float64
```

```
 46  2010    48 non-null     float64
 47  2011    48 non-null     float64
 48  2012    49 non-null     float64
 49  2013    49 non-null     float64
 50  2014    49 non-null     float64
 51  2015    49 non-null     float64
 52  2016    49 non-null     float64
dtypes: float64(52), object(1)
memory usage: 20.4+ KB
```

In [ ]:
```
# reading csv 13

d13 = pd.read_csv('/content/yearly_co2_emissions_1000_tonnes.csv')
d13.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 192 entries, 0 to 191
Columns: 265 entries, geo to 2014
dtypes: float64(264), object(1)
memory usage: 397.6+ KB
```

Merging the dataframes to form a single dataframe

In [ ]:
```
merge_df = pd.DataFrame(d2['geo'])
merge_df
```

Out[322]:

	geo
0	Afghanistan
1	Albania
2	Algeria
3	Andorra
4	Angola
...	...
187	Venezuela
188	Vietnam
189	Yemen
190	Zambia
191	Zimbabwe

192 rows × 1 columns

In [ ]:
```
d = pd.DataFrame(d2[['geo','2014']])

# merging using outer join

merge_df = merge_df.merge(d,how="outer")
```

In [ ]:
```
# renaming the columns

merge_df = merge_df.rename(columns={"2014":"co2_emmison_per_tonne"})
merge_df
```

Out[324]:

	geo	co2_emmison_per_tonne
0	Afghanistan	0.299
1	Albania	1.960
2	Algeria	3.720
3	Andorra	5.830
4	Angola	1.290
...	...	...
187	Venezuela	6.030
188	Vietnam	1.800
189	Yemen	0.865
190	Zambia	0.288
191	Zimbabwe	0.780

192 rows × 2 columns

In [ ]:
```
d = pd.DataFrame(d3[['geo','2014']])
d
```

Out[325]:

	geo	2014
0	Algeria	0.00458
1	Argentina	0.03460
2	Australia	1.82000
3	Austria	0.34700
4	Azerbaijan	0.00017
...	...	...
60	United Kingdom	0.45700
61	United States	1.43000
62	Uzbekistan	0.04000
63	Venezuela	0.00641
64	Vietnam	0.20500

65 rows × 2 columns

In [ ]:
```
# merging using outer join

merge_df = pd.merge(merge_df, d, how="outer", on='geo')
```

In [ ]:
```
merge_df = merge_df.rename(columns={"2014":"coal_consumption_per_cap"})
```

In [ ]:
```
d = d4[['geo','2014']]
d
```

Out[328]:

	geo	2014
0	Algeria	1640
1	Argentina	3290
2	Australia	10500
3	Austria	7540
4	Azerbaijan	2600
...	...	...
60	United Kingdom	5200
61	United States	13700
62	Uzbekistan	1820
63	Venezuela	3590
64	Vietnam	1540

65 rows × 2 columns

In [ ]:
```
# merging using outer join

merge_df = pd.merge(merge_df,d,how="outer",on='geo')
merge_df
```

Out[329]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	2014
0	Afghanistan	0.299	NaN	NaN
1	Albania	1.960	NaN	NaN
2	Algeria	3.720	0.00458	1640.0
3	Andorra	5.830	NaN	NaN
4	Angola	1.290	NaN	NaN
...	...	...	...	...
187	Venezuela	6.030	0.00641	3590.0
188	Vietnam	1.800	0.20500	1540.0
189	Yemen	0.865	NaN	NaN
190	Zambia	0.288	NaN	NaN
191	Zimbabwe	0.780	NaN	NaN

192 rows × 4 columns

In [ ]:
```
merge_df = merge_df.rename(columns={"2014":"electicity_generation_per_person"}
```

In [ ]:
```
d = d5[['geo','2014']]
d
```

Out[331]:

	geo	2014
0	Albania	2310.0
1	Algeria	1360.0
2	Angola	312.0
3	Argentina	3050.0
4	Armenia	1970.0
...	...	...
133	Venezuela	2660.0
134	Vietnam	1410.0
135	Yemen	216.0
136	Zambia	707.0
137	Zimbabwe	537.0

138 rows × 2 columns

In [ ]:
```
# merging using outer join

merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
```

In [ ]:
```
d = d6[['geo','2014']]
d
```

Out[333]:

	geo	2014
0	Afghanistan	2.07
1	Albania	28.20
2	Algeria	0.82
3	Andorra	34.00
4	Angola	46.50
...	...	...
187	Venezuela	53.10
188	Vietnam	47.20
189	Yemen	1.04
190	Zambia	65.70
191	Zimbabwe	37.20

192 rows × 2 columns

In [ ]:
```
# merging using outer join

merge_df = pd.merge(merge_df , d , how="outer" , on='geo')

# renaming the column

merge_df = merge_df.rename(columns={"2014":"forest_coverage_precent"})
merge_df
```

Out[334]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
188	Vietnam	1.800	0.20500	15
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	

193 rows × 6 columns

In [ ]:
```
d = d8[['geo','2014']]
merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
merge_df = merge_df.rename(columns={"2014":"income_per_person_gdp"})
merge_df
```

Out[335]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	
193	Monaco	NaN	NaN	

194 rows × 7 columns

In [ ]:
```
d = d8[['geo','2014']]
merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
merge_df = merge_df.rename(columns={"2014":"industry_percent_of_gdp"})
merge_df
```

Out[336]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	
193	Monaco	NaN	NaN	

194 rows × 8 columns

In [ ]:
```
d = d9[['geo','2014']]
merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
merge_df = merge_df.rename(columns={"2014":"natursl_gas_prod_per_person"})
merge_df
```

Out[337]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	
193	Monaco	NaN	NaN	

194 rows × 9 columns

In [ ]:
```
d = d11[['geo','2014']]
merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
merge_df = merge_df.rename(columns={"2014":"oil_consumption_per_person"})
merge_df
```

Out[338]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	
193	Monaco	NaN	NaN	

194 rows × 10 columns

In [ ]:
```
d = d12[['geo','2014']]
merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
merge_df = merge_df.rename(columns={"2014":"oil_production_per_cap"})
merge_df
```

Out[339]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	
193	Monaco	NaN	NaN	

194 rows × 11 columns

In [ ]:
```
d = d13[['geo','2014']]
merge_df = pd.merge(merge_df , d , how="outer" , on='geo')
merge_df = merge_df.rename(columns={"2014":" co2_emmisiion"})
merge_df
```

Out[340]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299	NaN	
1	Albania	1.960	NaN	
2	Algeria	3.720	0.00458	16
3	Andorra	5.830	NaN	
4	Angola	1.290	NaN	
...	...	...	...	
189	Yemen	0.865	NaN	
190	Zambia	0.288	NaN	
191	Zimbabwe	0.780	NaN	
192	San Marino	NaN	NaN	
193	Monaco	NaN	NaN	

194 rows × 12 columns

In [ ]:
```
# for the statistical elements of the data frama

merge_df.describe()
```

Out[341]:

	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_person	
count	192.000000	65.00000	65.000000	13
mean	4.440085	0.44212	6188.215385	425
std	6.065368	0.53450	5046.927099	602
min	0.044500	0.00000	350.000000	3
25%	0.659000	0.04000	2890.000000	81
50%	2.265000	0.25000	4750.000000	258
75%	5.695000	0.59400	8110.000000	536
max	45.400000	2.34000	27600.000000	5380

STEP 2: Data Preparation

In [ ]:
```
# to find the null value sum

merge_df.isnull().sum()
```

Out[342]:
```
geo                              0
co2_emmison_per_tonne            2
coal_consumption_per_cap       129
electicity_generation_per_person   129
2014_x                          57
2014_y                           3
income_per_person_gdp            1
industry_percent_of_gdp          1
natursl_gas_prod_per_person     11
oil_consumption_per_person     129
oil_production_per_cap         145
 co2_emmisiion                   2
dtype: int64
```

As we can see there are many null values in the data, so we need to treat them. There are many methods to treat the null values. The method used here is by filling with the mean values

In [ ]:
```
# filling with the mean

merge_df.fillna(merge_df.mean(),inplace=True)
```

In [ ]:
```
merge_df
```

Out[344]:

	geo	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_pe
0	Afghanistan	0.299000	0.44212	6188.21
1	Albania	1.960000	0.44212	6188.21
2	Algeria	3.720000	0.00458	1640.00
3	Andorra	5.830000	0.44212	6188.21
4	Angola	1.290000	0.44212	6188.21
...	...	...	...	
189	Yemen	0.865000	0.44212	6188.21
190	Zambia	0.288000	0.44212	6188.21
191	Zimbabwe	0.780000	0.44212	6188.21
192	San Marino	4.440085	0.44212	6188.21
193	Monaco	4.440085	0.44212	6188.21

194 rows × 12 columns

In [ ]:
```
merge_df.isnull().sum()
```

Out[345]:
```
geo                              0
co2_emmison_per_tonne            0
coal_consumption_per_cap         0
electicity_generation_per_person 0
2014_x                           0
2014_y                           0
income_per_person_gdp            0
industry_percent_of_gdp          0
natursl_gas_prod_per_person      0
oil_consumption_per_person       0
oil_production_per_cap           0
 co2_emmisiion                   0
dtype: int64
```

In [ ]:
```
merge_df.describe()
```

Out[346]:

	co2_emmison_per_tonne	coal_consumption_per_cap	electicity_generation_per_person	
count	194.000000	194.000000	194.000000	19
mean	4.440085	0.442120	6188.215385	425
std	6.033859	0.307793	2906.286095	505
min	0.044500	0.000000	350.000000	3
25%	0.695250	0.442120	6188.215385	145
50%	2.295000	0.442120	6188.215385	425
75%	5.615000	0.442120	6188.215385	425
max	45.400000	2.340000	27600.000000	5380

In [ ]:
```
merge_df.columns
```

Out[347]:
```
Index(['geo', 'co2_emmison_per_tonne', 'coal_consumption_per_cap',
       'electicity_generation_per_person', '2014_x', '2014_y',
       'income_per_person_gdp', 'industry_percent_of_gdp',
       'natursl_gas_prod_per_person', 'oil_consumption_per_person',
       'oil_production_per_cap', ' co2_emmisiion'],
      dtype='object')
```

In [ ]:
```
# finding the correlations

merge_df.corr
```

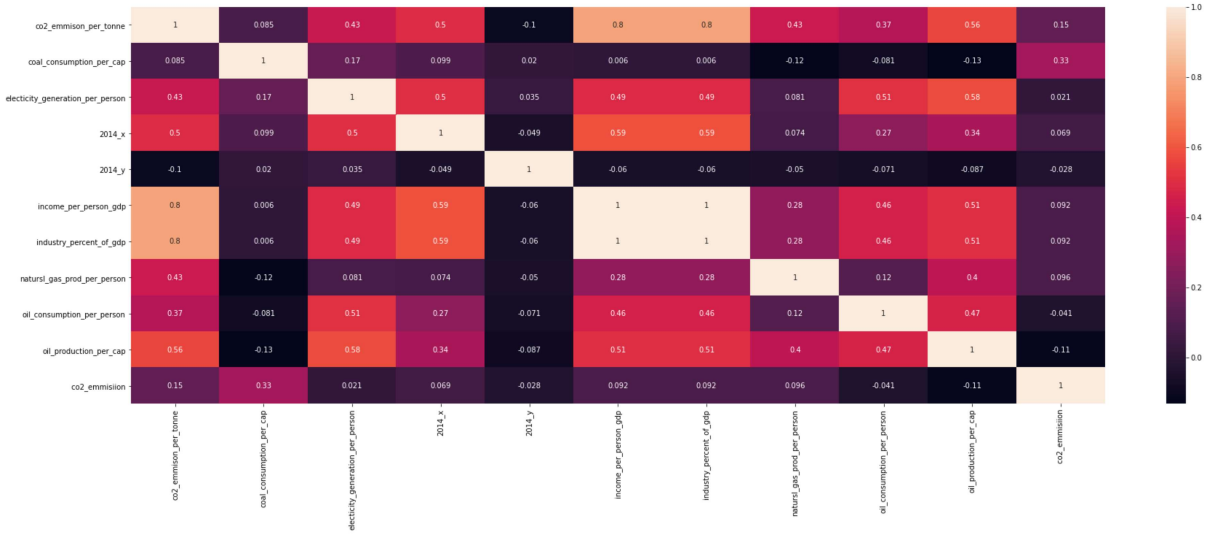
Out[348]:
```
<bound method DataFrame.corr of            geo  co2_emmison_per_tonne  ...  oil_p
roduction_per_cap   co2_emmisiion
0    Afghanistan                0.299000  ...              4.747659     9810.000000
1        Albania                1.960000  ...              4.747659     5720.000000
2        Algeria                3.720000  ...              1.760000   145000.000000
3        Andorra                5.830000  ...              4.747659      462.000000
4         Angola                1.290000  ...              3.080000    34800.000000
..           ...                     ...  ...                   ...             ...
189       Yemen                0.865000  ...              0.256000    22700.000000
190      Zambia                0.288000  ...              4.747659     4500.000000
191    Zimbabwe                0.780000  ...              4.747659    12000.000000
192   San Marino                4.440085  ...              4.747659   175992.541667
193      Monaco                4.440085  ...              4.747659   175992.541667

[194 rows x 12 columns]>
```

In [ ]:
```
%matplotlib inline

# to draw the heatmap for the correlations

plt.figure(figsize = (30,10))
sns.heatmap(merge_df.corr(), annot = True)
plt.show()
```



Train and Test Split

In [ ]:
```
x_split = merge_df.loc[:,['coal_consumption_per_cap',
       'electicity_generation_per_person', '2014_x', '2014_y',
       'income_per_person_gdp', 'industry_percent_of_gdp',
       'natursl_gas_prod_per_person', 'oil_consumption_per_person',
       'oil_production_per_cap',' co2_emmisiion']]

y_split = merge_df['co2_emmison_per_tonne']
```

In [ ]:
```
# importing libraries for train and test split

from sklearn.model_selection import train_test_split
x_train,x_test , y_train , y_test = train_test_split(x_split,y_split,test_size = 0.25)
x_train
```

Out[351]:

	coal_consumption_per_cap	electicity_generation_per_person	2014_x	2014_y	income_p
184	0.44212	6188.215385	3070.000000	10.40	
106	0.44212	6188.215385	4253.621898	70.20	
54	0.44212	6188.215385	4253.621898	15.00	
157	0.25000	5990.000000	5360.000000	36.80	
91	0.44212	6188.215385	3510.000000	54.00	
...	...	...	...	...	
144	0.00458	10100.000000	9440.000000	0.45	
187	0.00641	3590.000000	2660.000000	53.10	
128	0.02550	578.000000	471.000000	1.97	
181	0.16100	12800.000000	11300.000000	3.85	
59	0.13500	8750.000000	6940.000000	30.80	

145 rows × 10 columns

In [ ]:
```
# scaling the values for both train and test datasets

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

STEP 3: Model Building using LASSO

In [ ]:
```
params = {'alpha': [0.001, 0.01, 0.05, 0.1,
 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]}
```

In [ ]:
```
lasso = Lasso()
folds = 5
# cross validation
model_cv = GridSearchCV(estimator = lasso,
                        param_grid = params,
                        scoring= 'neg_mean_absolute_error',
                        cv = folds,
                        return_train_score=True,
                        verbose = 1)

model_cv.fit(x_train, y_train)
```

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done  60 out of  60 | elapsed:    0.1s finished
```

Out[354]:
```
GridSearchCV(cv=5, error_score=nan,
             estimator=Lasso(alpha=1.0, copy_X=True, fit_intercept=True,
                             max_iter=1000, normalize=False, positive=False,
                             precompute=False, random_state=None,
                             selection='cyclic', tol=0.0001, warm_start=False),
             iid='deprecated', n_jobs=None,
             param_grid={'alpha': [0.001, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5,
                                   0.6, 0.7, 0.8, 0.9]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=True,
             scoring='neg_mean_absolute_error', verbose=1)
```

In [ ]:
```
cv_results = pd.DataFrame(model_cv.cv_results_)
cv_results.head()
```

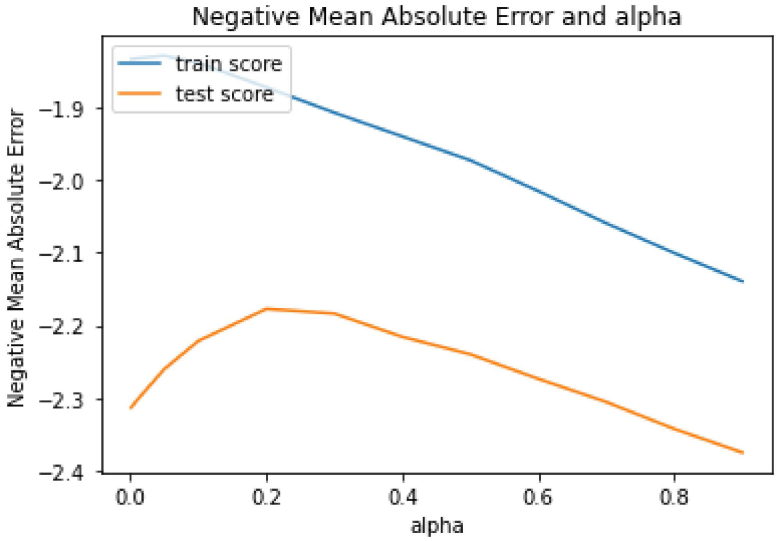
Out[355]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_alpha	params	split0_tes
0	0.001855	0.000275	0.000642	0.000150	0.001	{'alpha': 0.001}	-1
1	0.000991	0.000390	0.000345	0.000025	0.01	{'alpha': 0.01}	-1
2	0.000749	0.000011	0.000326	0.000010	0.05	{'alpha': 0.05}	-1
3	0.000777	0.000017	0.000334	0.000010	0.1	{'alpha': 0.1}	-1
4	0.000749	0.000021	0.000335	0.000018	0.2	{'alpha': 0.2}	-1

In [ ]:
```
# plotting mean test and train scoes with alpha
cv_results['param_alpha'] = cv_results['param_alpha'].astype('float32')

# plotting
plt.plot(cv_results['param_alpha'], cv_results['mean_train_score'])
plt.plot(cv_results['param_alpha'], cv_results['mean_test_score'])
plt.xlabel('alpha')
plt.ylabel('Negative Mean Absolute Error')

plt.title("Negative Mean Absolute Error and alpha")
plt.legend(['train score', 'test score'], loc='upper left')
plt.show()
```



In [ ]:
```
alpha =0.001

lasso = Lasso(alpha=alpha)

lasso.fit(x_train, y_train)
```

Out[357]:
```
Lasso(alpha=0.001, copy_X=True, fit_intercept=True, max_iter=1000,
      normalize=False, positive=False, precompute=False, random_state=None,
      selection='cyclic', tol=0.0001, warm_start=False)
```

In [ ]:
```
lasso.coef_
```

Out[358]:
```
array([ 0.88707198, -0.64543195,  0.23160598, -0.07564214,  4.07568537,
        0.        ,  1.21736902, -0.29544899,  1.45110259,  0.31401513])
```

In [ ]:
```
y_pred = model_cv.predict(x_test)
```

In [ ]:
```
# to calculate r2 score

print(r2_score(y_pred, y_test))
```

```
0.7046529908449242
```

In [ ]:
```

```