Importing the libraries for data cleaning, preprocessing and feature engineering

In [2]:

```
import pandas as pd
import io

from google.colab import files
uploaded = files.upload()

##Importing the dataset##

data = pd.read_csv(io.BytesIO(uploaded['data.csv']), header = 0)
```

Choose File **No file selected**

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving data.csv to data.csv
```

In [3]:

```
data
```

Out[3]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964
0	Arab World	ARB	% of females ages 15-49 having comprehensive c...	SH.HIV.KNOW.FE.ZS	NaN	NaN	NaN	NaN	NaN
1	Arab World	ARB	% of males ages 15-49 having comprehensive cor...	SH.HIV.KNOW.MA.ZS	NaN	NaN	NaN	NaN	NaN
2	Arab World	ARB	Adolescent fertility rate (births per 1,000 wo...	SP.ADO.TFRT	133.555013	134.159119	134.857912	134.504576	134.105211
3	Arab World	ARB	Adults (ages 15+) and children (0-14 years) li...	SH.HIV.TOTL	NaN	NaN	NaN	NaN	NaN
4	Arab World	ARB	Adults (ages 15+) and children (ages 0-14) new...	SH.HIV.INCD.TL	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
74837	Somalia	SOM	School enrollment, secondary, male (% net)	SE.SEC.NENR.MA	NaN	NaN	NaN	NaN	NaN
74838	Somalia	SOM	School enrollment, tertiary (% gross)	SE.TER.ENRR	NaN	NaN	NaN	NaN	NaN
74839	Somalia	SOM	School enrollment, tertiary, female (% gross)	SE.TER.ENRR.FE	NaN	NaN	NaN	NaN	NaN

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964
74840	Somalia	SOM	Sex ratio at birth (male births per female bir...	SP.POP.BRTH.MF	NaN	NaN	1.030000	NaN	NaN
74841	Somalia	SOM	Share of women employed in the nonagricultural...	NaN	NaN	NaN	NaN	NaN	NaN

74842 rows × 60 columns

In [4]:

```
##We have 74842 rows and 60 columns##
```

In [5]:

```
##Dropping the rows where all elements are missing##

data.dropna(how='all')
```

Out[5]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964
0	Arab World	ARB	% of females ages 15-49 having comprehensive c...	SH.HIV.KNOW.FE.ZS	NaN	NaN	NaN	NaN	NaN
1	Arab World	ARB	% of males ages 15-49 having comprehensive cor...	SH.HIV.KNOW.MA.ZS	NaN	NaN	NaN	NaN	NaN
2	Arab World	ARB	Adolescent fertility rate (births per 1,000 wo...	SP.ADO.TFRT	133.555013	134.159119	134.857912	134.504576	134.105211
3	Arab World	ARB	Adults (ages 15+) and children (0-14 years) li...	SH.HIV.TOTL	NaN	NaN	NaN	NaN	NaN
4	Arab World	ARB	Adults (ages 15+) and children (ages 0-14) new...	SH.HIV.INCD.TL	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
74837	Somalia	SOM	School enrollment, secondary, male (% net)	SE.SEC.NENR.MA	NaN	NaN	NaN	NaN	NaN
74838	Somalia	SOM	School enrollment, tertiary (% gross)	SE.TER.ENRR	NaN	NaN	NaN	NaN	NaN
74839	Somalia	SOM	School enrollment, tertiary, female (% gross)	SE.TER.ENRR.FE	NaN	NaN	NaN	NaN	NaN
74840	Somalia	SOM	Sex ratio at birth (male births per female bir...	SP.POP.BRTH.MF	NaN	NaN	1.030000	NaN	NaN
74841	Somalia	SOM	Share of women employed in the nonagricultural...	NaN	NaN	NaN	NaN	NaN	NaN

74842 rows × 60 columns

In [6]:

```
from google.colab import files
uploaded = files.upload()

##Importing files where countries population from 1995 to 2020##

data2 = pd.read_csv(io.BytesIO(uploaded['Countries Population from 1995 to 2020.csv']), h
eader = 0)
```

Choose File **No file selected**

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Countries Population from 1995 to 2020.csv to Countries Population from 1995 to 20
20.csv
```

In [7]:

```
data2
```

Out[7]:

	Year	Country	Population	Yearly % Change	Yearly Change	Migrants (net)	Median Age	Fertility Rate	Density (P/Km²)	Urban Pop %	Urban Population	Country's Share of World Pop %	Pop
0	2020	China	1439323776	0.39	5540090	-348399.0	38.4	1.69	153	60.8	875075919.0	18.47	7794
1	2019	China	1433783686	0.43	6135900	-348399.0	37.0	1.65	153	59.7	856409297.0	18.59	7713
2	2018	China	1427647786	0.47	6625995	-348399.0	37.0	1.65	152	58.6	837022095.0	18.71	7631
3	2017	China	1421021791	0.49	6972440	-348399.0	37.0	1.65	151	57.5	816957613.0	18.83	7547
4	2016	China	1414049351	0.51	7201481	-348399.0	37.0	1.65	151	56.3	796289491.0	18.94	7464
...	...	...	...	...	...	...	...	...	...	...	...	...	
4190	1975	Holy See	728	2.48	17	NaN	NaN	NaN	1,820	NaN	NaN	0.00	4079
4191	1970	Holy See	644	-5.49	-42	NaN	NaN	NaN	1,610	NaN	NaN	0.00	3700
4192	1965	Holy See	854	-1.18	-10	NaN	NaN	NaN	2,135	NaN	NaN	0.00	3339
4193	1960	Holy See	906	-0.04	0	NaN	NaN	NaN	2,265	NaN	NaN	0.00	3034
4194	1955	Holy See	908	0.00	0	NaN	NaN	NaN	2,270	NaN	NaN	0.00	2773

4195 rows × 14 columns

In [8]:

```
##Opening a database connection and building a new database to operate with sqlite3##

import sqlite3

##Creating an access to the database local.db by using the sqlite3.connect() method##

conn = sqlite3.connect('local.db')
```

```
data.to_sql("countries", conn, if_exists="replace", index=False)

pd.read_sql_query('select * from countries', conn)
```

```
/usr/local/lib/python3.8/dist-packages/pandas/core/generic.py:2872: UserWarning: The spac
es in these column names will not be changed. In pandas versions < 0.14, spaces were conv
erted to underscores.
  sql.to_sql(
```

Out[8]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964
0	Arab World	ARB	% of females ages 15-49 having comprehensive c...	SH.HIV.KNOW.FE.ZS	NaN	NaN	NaN	NaN	NaN
1	Arab World	ARB	% of males ages 15-49 having comprehensive cor...	SH.HIV.KNOW.MA.ZS	NaN	NaN	NaN	NaN	NaN
2	Arab World	ARB	Adolescent fertility rate (births per 1,000 wo...	SP.ADO.TFRT	133.555013	134.159119	134.857912	134.504576	134.105211
3	Arab World	ARB	Adults (ages 15+) and children (0-14 years) li...	SH.HIV.TOTL	NaN	NaN	NaN	NaN	NaN
4	Arab World	ARB	Adults (ages 15+) and children (ages 0-14) new...	SH.HIV.INCD.TL	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
74837	Somalia	SOM	School enrollment, secondary, male (% net)	SE.SEC.NENR.MA	NaN	NaN	NaN	NaN	NaN
74838	Somalia	SOM	School enrollment, tertiary (% gross)	SE.TER.ENRR	NaN	NaN	NaN	NaN	NaN
74839	Somalia	SOM	School enrollment, tertiary, female (% gross)	SE.TER.ENRR.FE	NaN	NaN	NaN	NaN	NaN
74840	Somalia	SOM	Sex ratio at birth (male births per female bir...	SP.POP.BRTH.MF	NaN	NaN	1.030000	NaN	NaN
74841	Somalia	SOM	Share of women employed in the nonagricultural...	None	NaN	NaN	NaN	NaN	NaN

74842 rows × 60 columns

In [9]:

```
##The act of reading a SQL query or database table into a DataFrame##

data = pd.read_sql_query('select * from countries where "Indicator Name" = "Birth rate,
crude (per 1,000 people)" group by "Country Name" ', conn)
```

In [10]:

```
##Returning the column labels of the given Dataframe##
data.columns
```

Out[10]:

```
Index(['Country Name', 'Country Code', 'Indicator Name', 'Indicator Code',
       '1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968',
       '1969', '1970', '1971', '1972', '1973', '1974', '1975', '1976', '1977',
       '1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
       '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
       '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004',
       '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013',
       '2014', '2015'],
      dtype='object')
```

In [11]:

```
##Adding data from a DataFrame to a SQL database##

data2.to_sql("population", conn, if_exists="replace", index=False)

pd.read_sql_query('select * from population', conn)
```

```
/usr/local/lib/python3.8/dist-packages/pandas/core/generic.py:2872: UserWarning: The spac
es in these column names will not be changed. In pandas versions < 0.14, spaces were conv
erted to underscores.
  sql.to_sql(
```

Out[11]:

	Year	Country	Population	Yearly % Change	Yearly Change	Migrants (net)	Median Age	Fertility Rate	Density (P/Km²)	Urban Pop %	Urban Population	Country's Share of World Pop %	Pop
0	2020	China	1439323776	0.39	5540090	-348399.0	38.4	1.69	153	60.8	875075919.0	18.47	7794
1	2019	China	1433783686	0.43	6135900	-348399.0	37.0	1.65	153	59.7	856409297.0	18.59	7713
2	2018	China	1427647786	0.47	6625995	-348399.0	37.0	1.65	152	58.6	837022095.0	18.71	7631
3	2017	China	1421021791	0.49	6972440	-348399.0	37.0	1.65	151	57.5	816957613.0	18.83	7547
4	2016	China	1414049351	0.51	7201481	-348399.0	37.0	1.65	151	56.3	796289491.0	18.94	7464
...	...	...	...	...	...	...	...	...	...	...	...	...	
4190	1975	Holy See	728	2.48	17	NaN	NaN	NaN	1,820	NaN	NaN	0.00	4079
4191	1970	Holy See	644	-5.49	-42	NaN	NaN	NaN	1,610	NaN	NaN	0.00	3700
4192	1965	Holy See	854	-1.18	-10	NaN	NaN	NaN	2,135	NaN	NaN	0.00	3339
4193	1960	Holy See	906	-0.04	0	NaN	NaN	NaN	2,265	NaN	NaN	0.00	3034
4194	1955	Holy See	908	0.00	0	NaN	NaN	NaN	2,270	NaN	NaN	0.00	2773

4195 rows × 14 columns

In [12]:

```
##Delivering a DataFrame that corresponds to the query's return set##

data2 = pd.read_sql_query('select * from population where "Year" = 2000 group by "Countr
y" ', conn)
```

In [13]:

```
##Displaying columns##

data2.columns
```

Out[13]:

```
Index(['Year', 'Country', 'Population', 'Yearly % Change', 'Yearly Change',
       'Migrants (net)', 'Median Age', 'Fertility Rate', 'Density (P/Km²)',
       'Urban Pop %', 'Urban Population', 'Country's Share of World Pop %',
       'World Population', 'Country Global Rank'],
      dtype='object')
```

In [14]:

```
##Merging the process of combining similar records##

new_data=data[['Country Name', 'Indicator Name', '2000']].merge(data2[['Country', 'Popul
ation']], left_on=['Country Name']
                                                               , right_on=['Country'])
```

In [15]:

```
##Collecting the functions in visualization elements of figure format includes figure, pl
otting area,
#plotting lines, and adding plot labels##

%matplotlib inline
import matplotlib.pyplot as plt
```

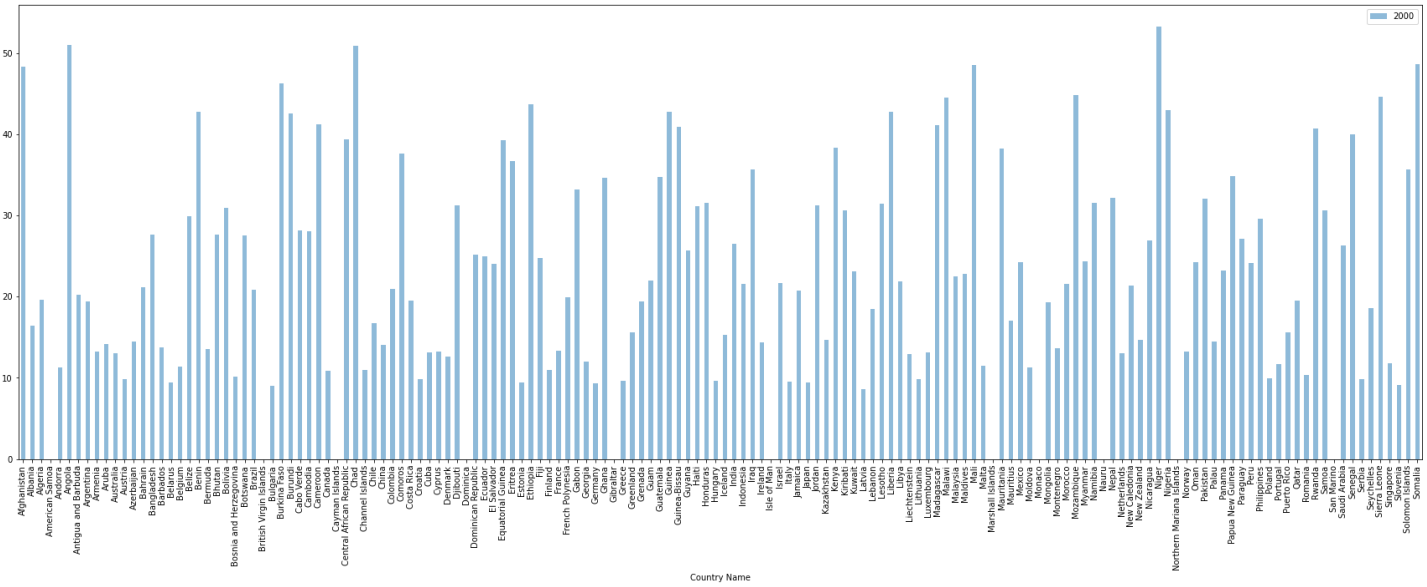
In [16]:

```
##Showing the data with country name and year (2000) with plot bar##

new_data[['Country Name', '2000']].groupby(by=['Country Name']).mean().plot.bar(alpha=0.
5,figsize=(30,10))
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb3e2459df0>
```



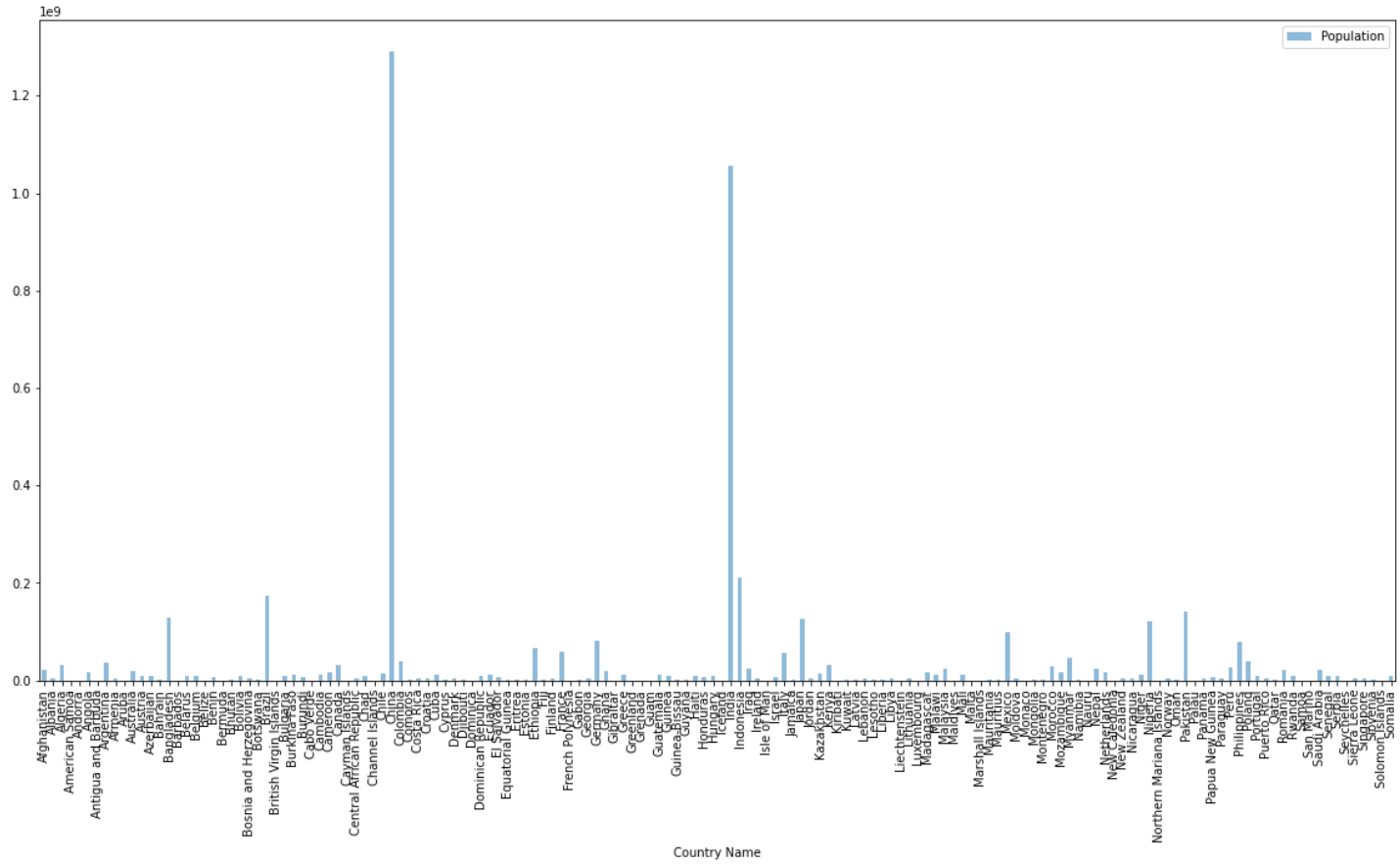
In [17]:

```
##Showing the data with coiuntry name and year (2000) with plot bar with different figure
size##

new_data[['Country Name', 'Population']].groupby(by=['Country Name']).mean().plot.bar(al
pha=0.5,figsize=(20,10))
```

Out[17]:

<matplotlib.axes._subplots.AxesSubplot at 0x7fb3e0e663d0>



In [18]:

```
##Here is the data with country name, indicator name, year, country and population##

new_data
```

Out[18]:

	Country Name	Indicator Name	2000	Country	Population
0	Afghanistan	Birth rate, crude (per 1,000 people)	48.332	Afghanistan	20779953
1	Albania	Birth rate, crude (per 1,000 people)	16.401	Albania	3129243
2	Algeria	Birth rate, crude (per 1,000 people)	19.570	Algeria	31042235
3	American Samoa	Birth rate, crude (per 1,000 people)	NaN	American Samoa	57821
4	Andorra	Birth rate, crude (per 1,000 people)	11.300	Andorra	65390
...	...	...	...	...	...
147	Sierra Leone	Birth rate, crude (per 1,000 people)	44.634	Sierra Leone	4584571
148	Singapore	Birth rate, crude (per 1,000 people)	11.800	Singapore	4028871
149	Slovenia	Birth rate, crude (per 1,000 people)	9.100	Slovenia	1987717
150	Solomon Islands	Birth rate, crude (per 1,000 people)	35.615	Solomon Islands	412660
151	Somalia	Birth rate, crude (per 1,000 people)	48.668	Somalia	8872254

152 rows × 5 columns

In [19]:

```
##Displaying the new columns after eliminating unwanted columns##

new_data.columns
```

Out[19]:

```
Index(['Country Name', 'Indicator Name', '2000', 'Country', 'Population'], dtype='object'
)
```

In [20]:

```
new_data = new_data[['Country Name', 'Indicator Name', '2000', 'Population']]
new_data
```

Out[20]:

	Country Name	Indicator Name	2000	Population
0	Afghanistan	Birth rate, crude (per 1,000 people)	48.332	20779953
1	Albania	Birth rate, crude (per 1,000 people)	16.401	3129243
2	Algeria	Birth rate, crude (per 1,000 people)	19.570	31042235
3	American Samoa	Birth rate, crude (per 1,000 people)	NaN	57821
4	Andorra	Birth rate, crude (per 1,000 people)	11.300	65390
...	...	...	...	...
147	Sierra Leone	Birth rate, crude (per 1,000 people)	44.634	4584571
148	Singapore	Birth rate, crude (per 1,000 people)	11.800	4028871
149	Slovenia	Birth rate, crude (per 1,000 people)	9.100	1987717
150	Solomon Islands	Birth rate, crude (per 1,000 people)	35.615	412660
151	Somalia	Birth rate, crude (per 1,000 people)	48.668	8872254

152 rows × 4 columns

In [21]:

```
##Importing DecisionTree classifier##

from sklearn.tree import DecisionTreeClassifier

##Using Labelencoder to transform non-numerical labels to numerical labels##

from sklearn.preprocessing import LabelEncoder
```

In [22]:

```
type(new_data)
```

Out[22]:

```
pandas.core.frame.DataFrame
```

In [23]:

```
##Dislaying all the columns till last##

new_data = new_data.iloc[:,:].values
```

In [24]:

```
new_data
```

Out[24]:

```
array([['Afghanistan', 'Birth rate, crude (per 1,000 people)', 48.332,
        20779953],
       ['Albania', 'Birth rate, crude (per 1,000 people)', 16.401,
        3129243],
       ['Algeria', 'Birth rate, crude (per 1,000 people)', 19.57,
        31042235],
       ['American Samoa', 'Birth rate, crude (per 1,000 people)', nan,
        57821],
       ['Andorra', 'Birth rate, crude (per 1,000 people)', 11.3, 65390],
       ['Angola', 'Birth rate, crude (per 1,000 people)', 51.009,
        16395473],
       ['Antigua and Barbuda', 'Birth rate, crude (per 1,000 people)',
        20.24, 76016],
       ['Argentina', 'Birth rate, crude (per 1,000 people)', 19.413,
        36870787],
       ['Armenia', 'Birth rate, crude (per 1,000 people)', 13.203,
        3069591],
       ['Aruba', 'Birth rate, crude (per 1,000 people)', 14.187, 90853],
       ['Australia', 'Birth rate, crude (per 1,000 people)', 13.0,
        18991431],
       ['Austria', 'Birth rate, crude (per 1,000 people)', 9.8, 8069276],
       ['Azerbaijan', 'Birth rate, crude (per 1,000 people)', 14.5,
        8122741],
       ['Bahrain', 'Birth rate, crude (per 1,000 people)', 21.165,
        664611],
       ['Bangladesh', 'Birth rate, crude (per 1,000 people)', 27.626,
        127657854],
       ['Barbados', 'Birth rate, crude (per 1,000 people)', 13.79,
        271515],
       ['Belarus', 'Birth rate, crude (per 1,000 people)', 9.4, 9871632],
       ['Belgium', 'Birth rate, crude (per 1,000 people)', 11.4,
        10282033],
       ['Belize', 'Birth rate, crude (per 1,000 people)', 29.876, 247315],
       ['Benin', 'Birth rate, crude (per 1,000 people)', 42.707, 6865951],
       ['Bermuda', 'Birth rate, crude (per 1,000 people)', 13.5, 65012],
       ['Bhutan', 'Birth rate, crude (per 1,000 people)', 27.601, 591021],
       ['Bolivia', 'Birth rate, crude (per 1,000 people)', 30.905,
        8418264],
       ['Bosnia and Herzegovina', 'Birth rate, crude (per 1,000 people)',
        10.168, 3751176],
       ['Botswana', 'Birth rate, crude (per 1,000 people)', 27.502,
        1643334],
       ['Brazil', 'Birth rate, crude (per 1,000 people)', 20.894,
        174790340],
       ['British Virgin Islands', 'Birth rate, crude (per 1,000 people)',
        nan, 20311],
       ['Bulgaria', 'Birth rate, crude (per 1,000 people)', 9.0, 7997957],
       ['Burkina Faso', 'Birth rate, crude (per 1,000 people)', 46.256,
        11607942],
       ['Burundi', 'Birth rate, crude (per 1,000 people)', 42.571,
        6378871],
       ['Cabo Verde', 'Birth rate, crude (per 1,000 people)', 28.198,
        428188],
       ['Cambodia', 'Birth rate, crude (per 1,000 people)', 28.06,
        12155239],
       ['Cameroon', 'Birth rate, crude (per 1,000 people)', 41.203,
        15513945],
       ['Canada', 'Birth rate, crude (per 1,000 people)', 10.9, 30588383],
       ['Cayman Islands', 'Birth rate, crude (per 1,000 people)', nan,
        42303],
       ['Central African Republic',
        'Birth rate, crude (per 1,000 people)', 39.325, 3640427],
       ['Chad', 'Birth rate, crude (per 1,000 people)', 50.855, 8355654],
       ['Channel Islands', 'Birth rate, crude (per 1,000 people)',
        10.992, 148443],
       ['Chile', 'Birth rate, crude (per 1,000 people)', 16.741,
```

['Chile', 'Birth rate, crude (per 1,000 people)', 16.741,
15342353],
['China', 'Birth rate, crude (per 1,000 people)', 14.03,
1290550765],
['Colombia', 'Birth rate, crude (per 1,000 people)', 20.959,
39629968],
['Comoros', 'Birth rate, crude (per 1,000 people)', 37.661,
542357],
['Costa Rica', 'Birth rate, crude (per 1,000 people)', 19.499,
3962372],
['Croatia', 'Birth rate, crude (per 1,000 people)', 9.8, 4428075],
['Cuba', 'Birth rate, crude (per 1,000 people)', 13.125, 11126430],
['Cyprus', 'Birth rate, crude (per 1,000 people)', 13.244, 943290],
['Denmark', 'Birth rate, crude (per 1,000 people)', 12.6, 5341194],
['Djibouti', 'Birth rate, crude (per 1,000 people)', 31.186,
717584],
['Dominica', 'Birth rate, crude (per 1,000 people)', nan, 69650],
['Dominican Republic', 'Birth rate, crude (per 1,000 people)',
25.167, 8471321],
['Ecuador', 'Birth rate, crude (per 1,000 people)', 24.922,
12681123],
['El Salvador', 'Birth rate, crude (per 1,000 people)', 24.083,
5887936],
['Equatorial Guinea', 'Birth rate, crude (per 1,000 people)',
39.303, 606181],
['Eritrea', 'Birth rate, crude (per 1,000 people)', 36.64,
2292416],
['Estonia', 'Birth rate, crude (per 1,000 people)', 9.4, 1399112],
['Ethiopia', 'Birth rate, crude (per 1,000 people)', 43.69,
66224804],
['Fiji', 'Birth rate, crude (per 1,000 people)', 24.739, 811006],
['Finland', 'Birth rate, crude (per 1,000 people)', 11.0, 5187954],
['France', 'Birth rate, crude (per 1,000 people)', 13.3, 59015096],
['French Polynesia', 'Birth rate, crude (per 1,000 people)',
19.961, 240686],
['Gabon', 'Birth rate, crude (per 1,000 people)', 33.22, 1228360],
['Georgia', 'Birth rate, crude (per 1,000 people)', 11.955,
4362187],
['Germany', 'Birth rate, crude (per 1,000 people)', 9.3, 81400882],
['Ghana', 'Birth rate, crude (per 1,000 people)', 34.589,
19278856],
['Gibraltar', 'Birth rate, crude (per 1,000 people)', nan, 31077],
['Greece', 'Birth rate, crude (per 1,000 people)', 9.6, 11082104],
['Greenland', 'Birth rate, crude (per 1,000 people)', 15.6, 56175],
['Grenada', 'Birth rate, crude (per 1,000 people)', 19.399,
102833],
['Guam', 'Birth rate, crude (per 1,000 people)', 21.936, 155329],
['Guatemala', 'Birth rate, crude (per 1,000 people)', 34.748,
11650743],
['Guinea', 'Birth rate, crude (per 1,000 people)', 42.777,
8240730],
['Guinea-Bissau', 'Birth rate, crude (per 1,000 people)', 40.876,
1201301],
['Guyana', 'Birth rate, crude (per 1,000 people)', 25.64, 746715],
['Haiti', 'Birth rate, crude (per 1,000 people)', 31.125, 8463806],
['Honduras', 'Birth rate, crude (per 1,000 people)', 31.52,
6574509],
['Hungary', 'Birth rate, crude (per 1,000 people)', 9.6, 10220507],
['Iceland', 'Birth rate, crude (per 1,000 people)', 15.3, 280435],
['India', 'Birth rate, crude (per 1,000 people)', 26.459,
1056575549],
['Indonesia', 'Birth rate, crude (per 1,000 people)', 21.518,
211513823],
['Iraq', 'Birth rate, crude (per 1,000 people)', 35.626, 23497585],
['Ireland', 'Birth rate, crude (per 1,000 people)', 14.4, 3783103],
['Isle of Man', 'Birth rate, crude (per 1,000 people)', nan,
76941],
['Israel', 'Birth rate, crude (per 1,000 people)', 21.7, 5945950],
['Italy', 'Birth rate, crude (per 1,000 people)', 9.5, 56692178],
['Jamaica', 'Birth rate, crude (per 1,000 people)', 20.7, 2654701],
['Japan', 'Birth rate, crude (per 1,000 people)', 9.4, 127524174],
['Jordan', 'Birth rate, crude (per 1,000 people)', 31.218,
5122493],

```
['Kazakhstan', 'Birth rate, crude (per 1,000 people)', 14.7,
 14922719],
['Kenya', 'Birth rate, crude (per 1,000 people)', 38.35, 31964557],
['Kiribati', 'Birth rate, crude (per 1,000 people)', 30.583,
 84396],
['Kuwait', 'Birth rate, crude (per 1,000 people)', 23.09, 2045123],
['Latvia', 'Birth rate, crude (per 1,000 people)', 8.6, 2384164],
['Lebanon', 'Birth rate, crude (per 1,000 people)', 18.53,
 3842778],
['Lesotho', 'Birth rate, crude (per 1,000 people)', 31.443,
 2032804],
['Liberia', 'Birth rate, crude (per 1,000 people)', 42.765,
 2848456],
['Libya', 'Birth rate, crude (per 1,000 people)', 21.859, 5357891],
['Liechtenstein', 'Birth rate, crude (per 1,000 people)', 12.9,
 33184],
['Lithuania', 'Birth rate, crude (per 1,000 people)', 9.8,
 3501839],
['Luxembourg', 'Birth rate, crude (per 1,000 people)', 13.1,
 436103],
['Madagascar', 'Birth rate, crude (per 1,000 people)', 41.123,
 15766806],
['Malawi', 'Birth rate, crude (per 1,000 people)', 44.475,
 11148758],
['Malaysia', 'Birth rate, crude (per 1,000 people)', 22.543,
 23194257],
['Maldives', 'Birth rate, crude (per 1,000 people)', 22.761,
 279398],
['Mali', 'Birth rate, crude (per 1,000 people)', 48.466, 10946445],
['Malta', 'Birth rate, crude (per 1,000 people)', 11.5, 393645],
['Marshall Islands', 'Birth rate, crude (per 1,000 people)', nan,
 50753],
['Mauritania', 'Birth rate, crude (per 1,000 people)', 38.229,
 2630219],
['Mauritius', 'Birth rate, crude (per 1,000 people)', 17.0,
 1185145],
['Mexico', 'Birth rate, crude (per 1,000 people)', 24.245,
 98899845],
['Moldova', 'Birth rate, crude (per 1,000 people)', 11.298,
 4202658],
['Monaco', 'Birth rate, crude (per 1,000 people)', nan, 32147],
['Mongolia', 'Birth rate, crude (per 1,000 people)', 19.327,
 2397418],
['Montenegro', 'Birth rate, crude (per 1,000 people)', 13.692,
 613559],
['Morocco', 'Birth rate, crude (per 1,000 people)', 21.573,
 28793679],
['Mozambique', 'Birth rate, crude (per 1,000 people)', 44.784,
 17711927],
['Myanmar', 'Birth rate, crude (per 1,000 people)', 24.318,
 46719701],
['Namibia', 'Birth rate, crude (per 1,000 people)', 31.546,
 1794571],
['Nauru', 'Birth rate, crude (per 1,000 people)', nan, 10337],
['Nepal', 'Birth rate, crude (per 1,000 people)', 32.137,
 23941110],
['Netherlands', 'Birth rate, crude (per 1,000 people)', 13.0,
 15926188],
['New Caledonia', 'Birth rate, crude (per 1,000 people)', 21.4,
 217269],
['New Zealand', 'Birth rate, crude (per 1,000 people)', 14.66,
 3858999],
['Nicaragua', 'Birth rate, crude (per 1,000 people)', 26.879,
 5069302],
['Niger', 'Birth rate, crude (per 1,000 people)', 53.251,
 11331557],
['Nigeria', 'Birth rate, crude (per 1,000 people)', 42.965,
 122283850],
['Northern Mariana Islands',
 'Birth rate, crude (per 1,000 people)', nan, 57453],
['Norway', 'Birth rate, crude (per 1,000 people)', 13.2, 4499367],
['Oman', 'Birth rate, crude (per 1,000 people)', 24.238, 2267973],
```

```
       ['Pakistan', 'Birth rate, crude (per 1,000 people)', 32.018,
        142343578],
       ['Palau', 'Birth rate, crude (per 1,000 people)', 14.5, 19105],
       ['Panama', 'Birth rate, crude (per 1,000 people)', 23.182,
        3030328],
       ['Papua New Guinea', 'Birth rate, crude (per 1,000 people)',
        34.802, 5847586],
       ['Paraguay', 'Birth rate, crude (per 1,000 people)', 27.079,
        5323201],
       ['Peru', 'Birth rate, crude (per 1,000 people)', 24.1, 26459944],
       ['Philippines', 'Birth rate, crude (per 1,000 people)', 29.635,
        77991755],
       ['Poland', 'Birth rate, crude (per 1,000 people)', 9.9, 38556693],
       ['Portugal', 'Birth rate, crude (per 1,000 people)', 11.7,
        10297112],
       ['Puerto Rico', 'Birth rate, crude (per 1,000 people)', 15.59,
        3668960],
       ['Qatar', 'Birth rate, crude (per 1,000 people)', 19.543, 592468],
       ['Romania', 'Birth rate, crude (per 1,000 people)', 10.4,
        22137419],
       ['Rwanda', 'Birth rate, crude (per 1,000 people)', 40.697,
        7933681],
       ['Samoa', 'Birth rate, crude (per 1,000 people)', 30.595, 174454],
       ['San Marino', 'Birth rate, crude (per 1,000 people)', nan, 27462],
       ['Saudi Arabia', 'Birth rate, crude (per 1,000 people)', 26.275,
        20663843],
       ['Senegal', 'Birth rate, crude (per 1,000 people)', 39.942,
        9797734],
       ['Serbia', 'Birth rate, crude (per 1,000 people)', 9.8, 9487616],
       ['Seychelles', 'Birth rate, crude (per 1,000 people)', 18.6,
        80996],
       ['Sierra Leone', 'Birth rate, crude (per 1,000 people)', 44.634,
        4584571],
       ['Singapore', 'Birth rate, crude (per 1,000 people)', 11.8,
        4028871],
       ['Slovenia', 'Birth rate, crude (per 1,000 people)', 9.1, 1987717],
       ['Solomon Islands', 'Birth rate, crude (per 1,000 people)',
        35.615, 412660],
       ['Somalia', 'Birth rate, crude (per 1,000 people)', 48.668,
        8872254]], dtype=object)
```

In [25]:

```
##Replacing the categorical value with a numeric value##

country = LabelEncoder()
```

In [26]:

```
##Performing fit and transform on the input data at a single time and convert data points
##

new_data[:,0]= country.fit_transform(new_data[:,0])
```

In [27]:

```
new_data
```

Out[27]:

```
array([[0, 'Birth rate, crude (per 1,000 people)', 48.332, 20779953],
       [1, 'Birth rate, crude (per 1,000 people)', 16.401, 3129243],
       [2, 'Birth rate, crude (per 1,000 people)', 19.57, 31042235],
       [3, 'Birth rate, crude (per 1,000 people)', nan, 57821],
       [4, 'Birth rate, crude (per 1,000 people)', 11.3, 65390],
       [5, 'Birth rate, crude (per 1,000 people)', 51.009, 16395473],
       [6, 'Birth rate, crude (per 1,000 people)', 20.24, 76016],
       [7, 'Birth rate, crude (per 1,000 people)', 19.413, 36870787],
       [8, 'Birth rate, crude (per 1,000 people)', 13.203, 3069591],
       [9, 'Birth rate, crude (per 1,000 people)', 14.187, 90853],
       [10, 'Birth rate, crude (per 1,000 people)', 13.0, 18991431],
       [11, 'Birth rate, crude (per 1,000 people)', 9.8, 80692761
```

[11, 'Birth rate, crude (per 1,000 people)', 9.6, 6009270],
[12, 'Birth rate, crude (per 1,000 people)', 14.5, 8122741],
[13, 'Birth rate, crude (per 1,000 people)', 21.165, 664611],
[14, 'Birth rate, crude (per 1,000 people)', 27.626, 127657854],
[15, 'Birth rate, crude (per 1,000 people)', 13.79, 271515],
[16, 'Birth rate, crude (per 1,000 people)', 9.4, 9871632],
[17, 'Birth rate, crude (per 1,000 people)', 11.4, 10282033],
[18, 'Birth rate, crude (per 1,000 people)', 29.876, 247315],
[19, 'Birth rate, crude (per 1,000 people)', 42.707, 6865951],
[20, 'Birth rate, crude (per 1,000 people)', 13.5, 65012],
[21, 'Birth rate, crude (per 1,000 people)', 27.601, 591021],
[22, 'Birth rate, crude (per 1,000 people)', 30.905, 8418264],
[23, 'Birth rate, crude (per 1,000 people)', 10.168, 3751176],
[24, 'Birth rate, crude (per 1,000 people)', 27.502, 1643334],
[25, 'Birth rate, crude (per 1,000 people)', 20.894, 174790340],
[26, 'Birth rate, crude (per 1,000 people)', nan, 20311],
[27, 'Birth rate, crude (per 1,000 people)', 9.0, 7997957],
[28, 'Birth rate, crude (per 1,000 people)', 46.256, 11607942],
[29, 'Birth rate, crude (per 1,000 people)', 42.571, 6378871],
[30, 'Birth rate, crude (per 1,000 people)', 28.198, 428188],
[31, 'Birth rate, crude (per 1,000 people)', 28.06, 12155239],
[32, 'Birth rate, crude (per 1,000 people)', 41.203, 15513945],
[33, 'Birth rate, crude (per 1,000 people)', 10.9, 30588383],
[34, 'Birth rate, crude (per 1,000 people)', nan, 42303],
[35, 'Birth rate, crude (per 1,000 people)', 39.325, 3640427],
[36, 'Birth rate, crude (per 1,000 people)', 50.855, 8355654],
[37, 'Birth rate, crude (per 1,000 people)', 10.992, 148443],
[38, 'Birth rate, crude (per 1,000 people)', 16.741, 15342353],
[39, 'Birth rate, crude (per 1,000 people)', 14.03, 1290550765],
[40, 'Birth rate, crude (per 1,000 people)', 20.959, 39629968],
[41, 'Birth rate, crude (per 1,000 people)', 37.661, 542357],
[42, 'Birth rate, crude (per 1,000 people)', 19.499, 3962372],
[43, 'Birth rate, crude (per 1,000 people)', 9.8, 4428075],
[44, 'Birth rate, crude (per 1,000 people)', 13.125, 11126430],
[45, 'Birth rate, crude (per 1,000 people)', 13.244, 943290],
[46, 'Birth rate, crude (per 1,000 people)', 12.6, 5341194],
[47, 'Birth rate, crude (per 1,000 people)', 31.186, 717584],
[48, 'Birth rate, crude (per 1,000 people)', nan, 69650],
[49, 'Birth rate, crude (per 1,000 people)', 25.167, 8471321],
[50, 'Birth rate, crude (per 1,000 people)', 24.922, 12681123],
[51, 'Birth rate, crude (per 1,000 people)', 24.083, 5887936],
[52, 'Birth rate, crude (per 1,000 people)', 39.303, 606181],
[53, 'Birth rate, crude (per 1,000 people)', 36.64, 2292416],
[54, 'Birth rate, crude (per 1,000 people)', 9.4, 1399112],
[55, 'Birth rate, crude (per 1,000 people)', 43.69, 66224804],
[56, 'Birth rate, crude (per 1,000 people)', 24.739, 811006],
[57, 'Birth rate, crude (per 1,000 people)', 11.0, 5187954],
[58, 'Birth rate, crude (per 1,000 people)', 13.3, 59015096],
[59, 'Birth rate, crude (per 1,000 people)', 19.961, 240686],
[60, 'Birth rate, crude (per 1,000 people)', 33.22, 1228360],
[61, 'Birth rate, crude (per 1,000 people)', 11.955, 4362187],
[62, 'Birth rate, crude (per 1,000 people)', 9.3, 81400882],
[63, 'Birth rate, crude (per 1,000 people)', 34.589, 19278856],
[64, 'Birth rate, crude (per 1,000 people)', nan, 31077],
[65, 'Birth rate, crude (per 1,000 people)', 9.6, 11082104],
[66, 'Birth rate, crude (per 1,000 people)', 15.6, 56175],
[67, 'Birth rate, crude (per 1,000 people)', 19.399, 102833],
[68, 'Birth rate, crude (per 1,000 people)', 21.936, 155329],
[69, 'Birth rate, crude (per 1,000 people)', 34.748, 11650743],
[70, 'Birth rate, crude (per 1,000 people)', 42.777, 8240730],
[71, 'Birth rate, crude (per 1,000 people)', 40.876, 1201301],
[72, 'Birth rate, crude (per 1,000 people)', 25.64, 746715],
[73, 'Birth rate, crude (per 1,000 people)', 31.125, 8463806],
[74, 'Birth rate, crude (per 1,000 people)', 31.52, 6574509],
[75, 'Birth rate, crude (per 1,000 people)', 9.6, 10220507],
[76, 'Birth rate, crude (per 1,000 people)', 15.3, 280435],
[77, 'Birth rate, crude (per 1,000 people)', 26.459, 1056575549],
[78, 'Birth rate, crude (per 1,000 people)', 21.518, 211513823],
[79, 'Birth rate, crude (per 1,000 people)', 35.626, 23497585],
[80, 'Birth rate, crude (per 1,000 people)', 14.4, 3783103],
[81, 'Birth rate, crude (per 1,000 people)', nan, 76941],
[82, 'Birth rate, crude (per 1,000 people)', 21.7, 5945950],
[83, 'Birth rate, crude (per 1,000 people)', 9.5, 56692178]

```
 [84, 'Birth rate, crude (per 1,000 people)', 20.7, 2654701],
 [85, 'Birth rate, crude (per 1,000 people)', 9.4, 127524174],
 [86, 'Birth rate, crude (per 1,000 people)', 31.218, 5122493],
 [87, 'Birth rate, crude (per 1,000 people)', 14.7, 14922719],
 [88, 'Birth rate, crude (per 1,000 people)', 38.35, 31964557],
 [89, 'Birth rate, crude (per 1,000 people)', 30.583, 84396],
 [90, 'Birth rate, crude (per 1,000 people)', 23.09, 2045123],
 [91, 'Birth rate, crude (per 1,000 people)', 8.6, 2384164],
 [92, 'Birth rate, crude (per 1,000 people)', 18.53, 3842778],
 [93, 'Birth rate, crude (per 1,000 people)', 31.443, 2032804],
 [94, 'Birth rate, crude (per 1,000 people)', 42.765, 2848456],
 [95, 'Birth rate, crude (per 1,000 people)', 21.859, 5357891],
 [96, 'Birth rate, crude (per 1,000 people)', 12.9, 33184],
 [97, 'Birth rate, crude (per 1,000 people)', 9.8, 3501839],
 [98, 'Birth rate, crude (per 1,000 people)', 13.1, 436103],
 [99, 'Birth rate, crude (per 1,000 people)', 41.123, 15766806],
 [100, 'Birth rate, crude (per 1,000 people)', 44.475, 11148758],
 [101, 'Birth rate, crude (per 1,000 people)', 22.543, 23194257],
 [102, 'Birth rate, crude (per 1,000 people)', 22.761, 279398],
 [103, 'Birth rate, crude (per 1,000 people)', 48.466, 10946445],
 [104, 'Birth rate, crude (per 1,000 people)', 11.5, 393645],
 [105, 'Birth rate, crude (per 1,000 people)', nan, 50753],
 [106, 'Birth rate, crude (per 1,000 people)', 38.229, 2630219],
 [107, 'Birth rate, crude (per 1,000 people)', 17.0, 1185145],
 [108, 'Birth rate, crude (per 1,000 people)', 24.245, 98899845],
 [109, 'Birth rate, crude (per 1,000 people)', 11.298, 4202658],
 [110, 'Birth rate, crude (per 1,000 people)', nan, 32147],
 [111, 'Birth rate, crude (per 1,000 people)', 19.327, 2397418],
 [112, 'Birth rate, crude (per 1,000 people)', 13.692, 613559],
 [113, 'Birth rate, crude (per 1,000 people)', 21.573, 28793679],
 [114, 'Birth rate, crude (per 1,000 people)', 44.784, 17711927],
 [115, 'Birth rate, crude (per 1,000 people)', 24.318, 46719701],
 [116, 'Birth rate, crude (per 1,000 people)', 31.546, 1794571],
 [117, 'Birth rate, crude (per 1,000 people)', nan, 10337],
 [118, 'Birth rate, crude (per 1,000 people)', 32.137, 23941110],
 [119, 'Birth rate, crude (per 1,000 people)', 13.0, 15926188],
 [120, 'Birth rate, crude (per 1,000 people)', 21.4, 217269],
 [121, 'Birth rate, crude (per 1,000 people)', 14.66, 3858999],
 [122, 'Birth rate, crude (per 1,000 people)', 26.879, 5069302],
 [123, 'Birth rate, crude (per 1,000 people)', 53.251, 11331557],
 [124, 'Birth rate, crude (per 1,000 people)', 42.965, 122283850],
 [125, 'Birth rate, crude (per 1,000 people)', nan, 57453],
 [126, 'Birth rate, crude (per 1,000 people)', 13.2, 4499367],
 [127, 'Birth rate, crude (per 1,000 people)', 24.238, 2267973],
 [128, 'Birth rate, crude (per 1,000 people)', 32.018, 142343578],
 [129, 'Birth rate, crude (per 1,000 people)', 14.5, 19105],
 [130, 'Birth rate, crude (per 1,000 people)', 23.182, 3030328],
 [131, 'Birth rate, crude (per 1,000 people)', 34.802, 5847586],
 [132, 'Birth rate, crude (per 1,000 people)', 27.079, 5323201],
 [133, 'Birth rate, crude (per 1,000 people)', 24.1, 26459944],
 [134, 'Birth rate, crude (per 1,000 people)', 29.635, 77991755],
 [135, 'Birth rate, crude (per 1,000 people)', 9.9, 38556693],
 [136, 'Birth rate, crude (per 1,000 people)', 11.7, 10297112],
 [137, 'Birth rate, crude (per 1,000 people)', 15.59, 3668960],
 [138, 'Birth rate, crude (per 1,000 people)', 19.543, 592468],
 [139, 'Birth rate, crude (per 1,000 people)', 10.4, 22137419],
 [140, 'Birth rate, crude (per 1,000 people)', 40.697, 7933681],
 [141, 'Birth rate, crude (per 1,000 people)', 30.595, 174454],
 [142, 'Birth rate, crude (per 1,000 people)', nan, 27462],
 [143, 'Birth rate, crude (per 1,000 people)', 26.275, 20663843],
 [144, 'Birth rate, crude (per 1,000 people)', 39.942, 9797734],
 [145, 'Birth rate, crude (per 1,000 people)', 9.8, 9487616],
 [146, 'Birth rate, crude (per 1,000 people)', 18.6, 80996],
 [147, 'Birth rate, crude (per 1,000 people)', 44.634, 4584571],
 [148, 'Birth rate, crude (per 1,000 people)', 11.8, 4028871],
 [149, 'Birth rate, crude (per 1,000 people)', 9.1, 1987717],
 [150, 'Birth rate, crude (per 1,000 people)', 35.615, 412660],
 [151, 'Birth rate, crude (per 1,000 people)', 48.668, 8872254]],
dtype=object)
```

In [28]:

```
##Displaying data without columns##

new_data = pd.DataFrame(new_data)
new_data
```

Out[28]:

	0	1	2	3
0	0	Birth rate, crude (per 1,000 people)	48.332	20779953
1	1	Birth rate, crude (per 1,000 people)	16.401	3129243
2	2	Birth rate, crude (per 1,000 people)	19.57	31042235
3	3	Birth rate, crude (per 1,000 people)	NaN	57821
4	4	Birth rate, crude (per 1,000 people)	11.3	65390
...	...	...	...	...
147	147	Birth rate, crude (per 1,000 people)	44.634	4584571
148	148	Birth rate, crude (per 1,000 people)	11.8	4028871
149	149	Birth rate, crude (per 1,000 people)	9.1	1987717
150	150	Birth rate, crude (per 1,000 people)	35.615	412660
151	151	Birth rate, crude (per 1,000 people)	48.668	8872254

152 rows × 4 columns

In [29]:

```
##Adding columns to the final data##

new_data.columns = ['Country Name', 'Indicator Name', '2000', 'Population']
```

In [30]:

```
new_data
```

Out[30]:

	Country Name	Indicator Name	2000	Population
0	0	Birth rate, crude (per 1,000 people)	48.332	20779953
1	1	Birth rate, crude (per 1,000 people)	16.401	3129243
2	2	Birth rate, crude (per 1,000 people)	19.57	31042235
3	3	Birth rate, crude (per 1,000 people)	NaN	57821
4	4	Birth rate, crude (per 1,000 people)	11.3	65390
...	...	...	...	...
147	147	Birth rate, crude (per 1,000 people)	44.634	4584571
148	148	Birth rate, crude (per 1,000 people)	11.8	4028871

	Country Name	Indicator Name	2000	Population
149	149	Birth rate, crude (per 1,000 people)	9.1	1987717
150	150	Birth rate, crude (per 1,000 people)	35.615	412660
151	151	Birth rate, crude (per 1,000 people)	48.668	8872254

152 rows × 4 columns

In [31]:

```
##Dropping the rows with a null by checking all the columns in each row##

new_data = new_data.dropna(axis=0)
```

In [32]:

```
new_data
```

Out[32]:

	Country Name	Indicator Name	2000	Population
0	0	Birth rate, crude (per 1,000 people)	48.332	20779953
1	1	Birth rate, crude (per 1,000 people)	16.401	3129243
2	2	Birth rate, crude (per 1,000 people)	19.57	31042235
4	4	Birth rate, crude (per 1,000 people)	11.3	65390
5	5	Birth rate, crude (per 1,000 people)	51.009	16395473
...	...	...	...	...
147	147	Birth rate, crude (per 1,000 people)	44.634	4584571
148	148	Birth rate, crude (per 1,000 people)	11.8	4028871
149	149	Birth rate, crude (per 1,000 people)	9.1	1987717
150	150	Birth rate, crude (per 1,000 people)	35.615	412660
151	151	Birth rate, crude (per 1,000 people)	48.668	8872254

141 rows × 4 columns

In [33]:

```
##Showing the final data with country name, year and population##

new_data = new_data[['Country Name', '2000', 'Population']]
```

In [34]:

```
##Splitting the data arrays into two subsets by importing train_test_split##

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

##Importing Support Vector Classifier##

from sklearn.svm import SVC
```

In [35]:

```
X = new_data.drop(columns=['Population'])
Y = new_data['Population'].astype('int')
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2)

our_model= DecisionTreeClassifier()
our_model.fit(X_train,Y_train)
predictions = our_model.predict(X_test)


score = accuracy_score(Y_test, predictions)
score
```

Out[35]:

```
0.0
```

In [36]:

```
##Showing the predictions after using DecisionTree Classifier model##

X = new_data.drop(columns=['Population'])
Y = new_data['Population'].astype('int')


our_model= DecisionTreeClassifier()
our_model.fit(X,Y)
predictions = our_model.predict([[27,9],[46,12.6]])
predictions
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/base.py:450: UserWarning: X does not have
valid feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

Out[36]:

```
array([7997957, 5341194])
```

In [37]:

```
##Showing the predictions after using Support Vector Classifier model##

X = new_data.drop(columns=['Population'])
Y = new_data['Population'].astype('int')
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2)


second_model = SVC(kernel='rbf', random_state=1)
second_model.fit(X,Y)


new_predictions = second_model.predict([[27,9],[46,12.6]])
new_predictions
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/base.py:450: UserWarning: X does not have
valid feature names, but SVC was fitted with feature names
  warnings.warn(
```

Out[37]:

```
array([7997957, 5341194])
```

Finally, the predictions are same in both models where I showed.

In [ ]:

```
new_data
```

Out[ ]:

Country Name	2000	Population

	Country Name	2000	Population
0	0	48.332	20779953
1	1	16.401	3129243
2	2	19.57	31042235
4	4	11.3	65390
5	5	51.009	16395473
...	...	...	...
147	147	44.634	4584571
148	148	11.8	4028871
149	149	9.1	1987717
150	150	35.615	412660
151	151	48.668	8872254

141 rows × 3 columns