Project 1: Population Prediction

Authors:

- David Hoffman
- Kyle Kolodziej

Business Understanding

The history of demographic and population analysis stretches back to the 16th century where men such as Sir Francis Bacon and Sir Isaac Newton developed methods for gathering data and making predictions. As technology has advanced the methods for obtaining and manipulating data have inreased tenfold and with these technological advancements, the application of this data has also grown. Predictive population analysis allows us to see certain trends with various inputs giving us the opportunity to plan ahead for the years to come. For this project, we have been tasked with predicting the world population in the year 2122 and to do this we will make use of a proper machine-learning model. We hope that our prediction will help guide public policy so that our infrastructure, food resources, and per capita pollutant levels are ready for the continued growth of our global population.

Dataset URL: [https://www.kaggle.com/sansuthi/world-population-by-year (https://www.kaggle.com/sansuthi/world-population-by-year)](https://www.kaggle.com/sansuthi/world-population-by-year)

Methodology and Assumptions

To complete our prediction task, we will first be using a simple linear regression model. This model will be able to interpret historical data and predict out the next hundred years of the global population. After interpretting the results of this model, we have decided to incorporate a more sophisticated model as well. While our original model gives us a good baseline to work from, the prediction appears to be far too high given our own research and logical assessment. In order to create a more accurate model we will be assimilating a more complex dataset that tracks the population of each individual country rather than the earth as a whole. We will also be incorporating key factors such as fertility rate and life expectancy by country in order to generate a more informed prediction.

One of the key assuptions taken while creating our model is that we are not taking into account any kind of catestrophic event such as global war, natural disasters, or a pandemic. We are also assuming that the world will have the resources to sustain population growth and that this growth will not put any strains on global food or water resources or anything else that will effect the baseline survivability of the popualtion. Our final assumption is that this population growth will not incur extreme environmental strain reducing the livability of any areas of the world.

Data Preperation and Processing

In [1168]:
```
import numpy as np
import pandas as pd

df = pd.read_csv("WorldPopulation.csv")

df.head()
```

Out[1168]:

	Year	Population	ChangePerc	NetChange	Density	Urban	UrbanPerc
0	2020	7794798739	1.05	81330639	52	4378993944	56
1	2019	7713468100	1.08	82377060	52	4299438618	56
2	2018	7631091040	1.10	83232115	51	4219817318	55
3	2017	7547858925	1.12	83836876	51	4140188594	55
4	2016	7464022049	1.14	84224910	50	4060652683	54

In [1169]:
```
# confirm that there are no missing values
df.isnull().sum()
```

Out[1169]:
```
Year          0
Population    0
ChangePerc    0
NetChange     0
Density       0
Urban         0
UrbanPerc     0
dtype: int64
```

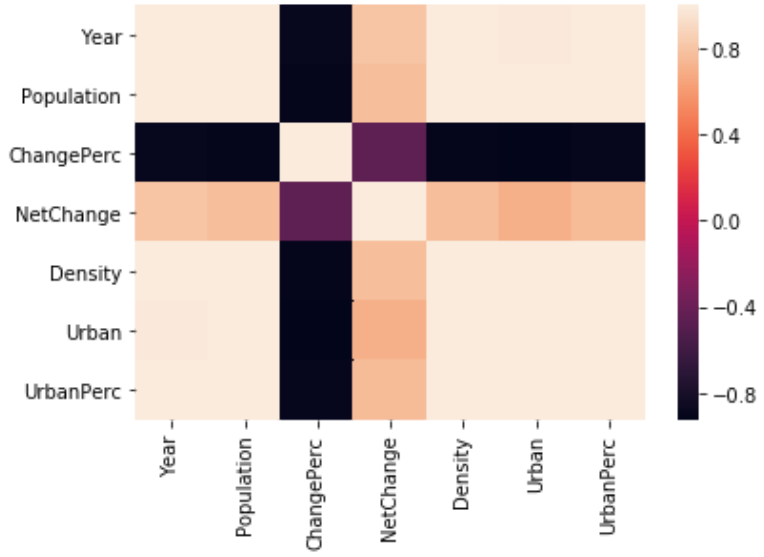
In [1170]:
```
# gives statistical information about the dataset
df.describe().T
```

Out[1170]:

	count	mean	std	min	25%	50%	7
Year	70.0	1.985500e+03	2.035109e+01	1.951000e+03	1.968250e+03	1.985500e+03	2.002750e
Population	70.0	5.002010e+09	1.594877e+09	2.584034e+09	3.570120e+09	4.915745e+09	6.361332e
ChangePerc	70.0	1.616429e+00	3.212883e-01	1.050000e+00	1.262500e+00	1.770000e+00	1.847500e
NetChange	70.0	7.511954e+07	1.256972e+07	4.674740e+07	7.314225e+07	7.942852e+07	8.356683e
Density	70.0	3.355714e+01	1.072399e+01	1.700000e+01	2.400000e+01	3.300000e+01	4.275000e
Urban	70.0	2.226815e+09	1.070191e+09	7.750677e+08	1.294408e+09	2.035272e+09	3.054260e
UrbanPerc	70.0	4.220000e+01	7.447595e+00	3.000000e+01	3.600000e+01	4.150000e+01	4.800000e

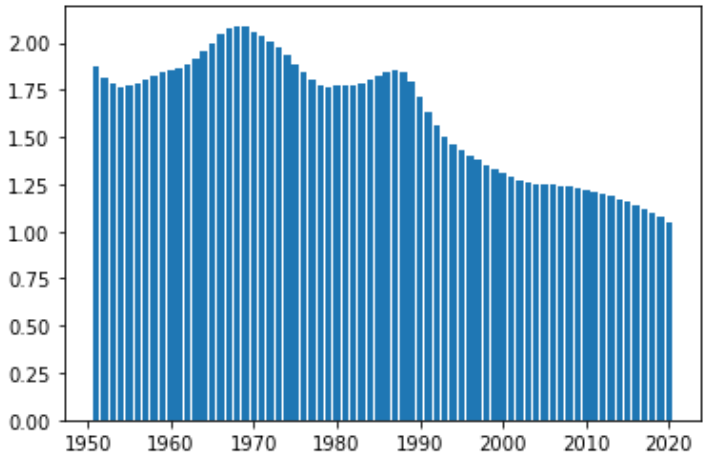
In [1171]:
```
import seaborn as sns

sns.heatmap(df.corr())
```

Out[1171]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c39db0f548>
```



In [1172]:
```
import matplotlib.pyplot as plt

plt.bar(df['Year'], df['ChangePerc'])
```

Out[1172]:
```
<BarContainer object of 70 artists>
```



In [1173]:
```
from sklearn.model_selection import train_test_split

# get values of everything except population because that is what is being predicte
d by y
x = df.iloc[:, :-6].values
y = df.iloc[:, 1].values

x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 1/3, random_sta
te=0)
```

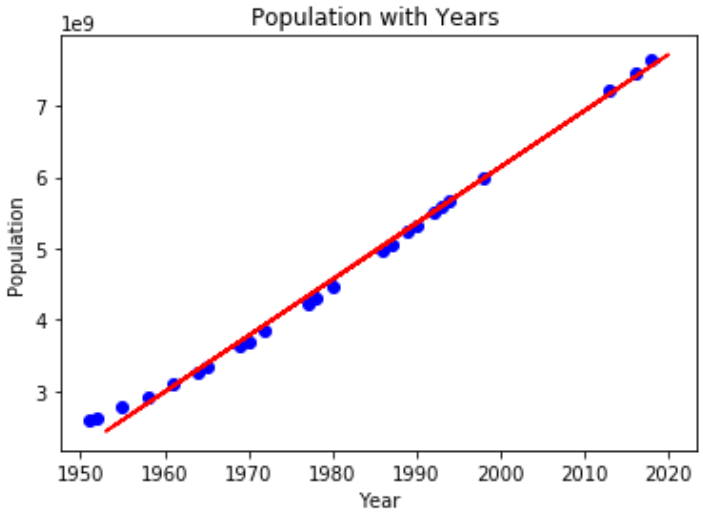
In the large coding section above we complete all of the preprocessing and data preperation required so that our dataset is ready to be sent through our model(s). The first step of this process requires us to grab the dataset from kaggle using our username and api key in order to download the dataset in the current runtime of our file. Next, we made use of the .head() function in order to print the first 5 rows of our dataset and see each of our features. After this, we used the .isnull() function paired with the .sum() function to check if there are any missing cells within our dataset, but luckily there aren't any. Next, we utilized the .describe function in order to conduct a statisical analysis on each of our features returning things such as the mean, standard deviation, min, and max. After returning our feature statistics, we made use of a few different graphs to gain a little more insight about our base dataset. The first graph is a heatmap which shows the stastical correlation between each of our features and the second graph compares the overal percentage change in population growth over each year. As you can see the total percentage change follows a pretty distinct trend of decline after the 1970s. Once all of this has been completed, we can see that we started with a very functional dataset that did not require much changing in order to be ready for our model. The final step of our preprocessing is to split our set into a train and test split and we do so using the simple sci-kit learn implementation while setting our testing set to contain about 33% of the total dataset.

Model Creation

In [1174]:
```
# Model implementation based on https://www.kaggle.com/avengerwang/population-analysis-forecast-with-linear-regress
# See item 2 in references
from sklearn.linear_model import LinearRegression

regressor = LinearRegression()
regressor.fit(x_train, y_train)

y_pred = regressor.predict(x_test)
x_pred = regressor.predict(x_train)

regressor.score(x_test, y_test)
```

Out[1174]:
```
0.9946059046903974
```

In [1175]:
```
plt.scatter(x_test, y_test, color="blue")
plt.plot(x_train, x_pred, color="red")
plt.title("Population with Years ")
plt.xlabel("Year")
plt.ylabel("Population")
plt.show()
```



In [1176]:
```
regressor.intercept_
```

Out[1176]:
```
-151460301652.88126
```

In [1177]:
```
regressor.coef_
```

Out[1177]:
```
array([78801117.7313113])
```

In [1178]:
```
year=[2122]
pop_year=[]

for i in year:
    y= -151460301652.88126+78801117.7313113*i
    pop_year.append(y)

print(pop_year)
```

```
[15755670172.961334]
```

The code above creates a simple linear regression model that we send our data through. The model is created using the sci-kit learn implementation which takes in our x and y testing sets as parameters. After passing our data through the linear regression function, we utilized the .score function to return the overall accuracy rating of our model when compared to our validation set. The .score method returned a very high accuracy rating of around 99% giving us confidence in this model. After the output of our regressor score, we built a graph which compares the results of the linear regression (shown with blue dots) to the training set which is shown with a red line. After the graph, we print the intercept and coefficient of our model which will be critical to our final population prediction. The intercept is the average expected value for the when all of the predictor variables are equal to zero and the coefficient represents the difference in the predicted value for each one-unit change in the predictor variable. In the last code chunk of this section, we finally print the output of this model for the world population by 2122 and get a result of ~15,755,670,172 people.

Model Analysis

Our first reaction to the result of our original model is that the prediction is a little high. The current population of the world is 7,794,798,739 people so expecting this number to double in only one hundred years is a bit of a stretch. Also, when conducting research prior to building our model, the general consensus amongst professionals was a prediction of around 10 billion people by 2122. The fact that our prediction is over 5 billion higher than the professional prediction does not give us very much confidence in our model; however, our first model does give us a good baseline to work from. As we work to create our second model we will try to include more limiting features as well as breaking up the population into segments by country rather than looking at the entire population as a whole to try to get a more accurate result. It also is worth noting that the farther out we try to extrapolate our prediction from the final year in our dataset (2019) generally the less accurate our linear regression will tend to be.

Strengths/Weaknesses of Model

Linear regression is a quick and easy algorithm with a short runtime that allows us to obtain a good baseline prediction upon which we can evaluate our second, more complex model. It is also a perfect algorithm for our dataset as it performs very well on linear data and can be used to determine the relationship between different features within the dataset. Linear regression also assumes that each row of data is independent which also works well for our dataset. One weakness of our model and a possible issue that could explain the really high population estimate could be overfitting which could be reduced using regularization, a technique that will be implemented in our second model. In addition to overfitting, another weakness of any linear regression model is the phenomena of underfitting where the hypothesis fails to fit the data well; however, we do not believe that this is effecting our model. Another general weakness of linear regression models is that they do not handle outliers very well, but our dataset is free of outliers allowing the model to operate successfully. The final weakness, and the most likely reason for our high prediction, is that the the model does not account for any limiting factors that would prevent the population from being able to grow uncontrollably. A more refined model would likely also try to predict things like food and water resources in order to understand how many people the earth can sustain before trying to predict the eventual population.

New Model

Now, we are going to build a model that breaks down population and other factors by each country and work on predicting each country's population in 2122 and summing that up to get a total world population prediction for 2122

Getting 2122 Values for Factors

Fertility Rate

In [1066]:
```
fertility = pd.read_csv("fertility-rate.csv") # Source: https://ourworldindata.org/fertility-rate
```

In [1067]:
```
fertility.head()
```

Out[1067]:

	Entity	Code	Year	Estimates, 1950 - 2020: Annually interpolated demographic indicators - Total fertility (live births per woman)
0	Afghanistan	AFG	1950	7.45
1	Afghanistan	AFG	1951	7.45
2	Afghanistan	AFG	1952	7.45
3	Afghanistan	AFG	1953	7.45
4	Afghanistan	AFG	1954	7.45

Let's change up the column names to have better labels and drop the Code column

In [1068]:
```
# Change the column name
# Estimates, 1950 - 2020: Annually interpolated demographic indicators - Total fertility (live births per woman)
# to
# Rate
fertility = fertility.rename(columns={'Estimates, 1950 - 2020: Annually interpolated demographic indicators - Total fertility (live births per woman)' : 'Rate'})
```

In [1069]:
```
fertility = fertility.drop(["Code"], axis = 1)
```

In [1070]:
```
fertility.head()
```

Out[1070]:

	Entity	Year	Rate
0	Afghanistan	1950	7.45
1	Afghanistan	1951	7.45
2	Afghanistan	1952	7.45
3	Afghanistan	1953	7.45
4	Afghanistan	1954	7.45

Now going to One Hot Encode the countries in the fertility data in order to deal with the categorical data of countries

In [1071]:
```
import category_encoders as ce
# Source: https://stackoverflow.com/questions/55609815/onehotencoder-change-name-columns
# Source: https://pypi.org/project/category-encoders/
fertDum = fertility
ohe = ce.OneHotEncoder(handle_unknown='ignore',
    use_cat_names=True)
label_fournisseur = ohe.fit_transform(list(fertDum['Entity']))
label_fournisseur.columns = [ x[2:] for x in label_fournisseur.columns ]
oheFert = fertDum.join(label_fournisseur)
oheFert
```

Out[1071]:

	Entity	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	...
0	Afghanistan	1950	7.450	1	0	0	0	0	0	0	...
1	Afghanistan	1951	7.450	1	0	0	0	0	0	0	...
2	Afghanistan	1952	7.450	1	0	0	0	0	0	0	...
3	Afghanistan	1953	7.450	1	0	0	0	0	0	0	...
4	Afghanistan	1954	7.450	1	0	0	0	0	0	0	...
5	Afghanistan	1955	7.450	1	0	0	0	0	0	0	...
6	Afghanistan	1956	7.450	1	0	0	0	0	0	0	...
7	Afghanistan	1957	7.450	1	0	0	0	0	0	0	...
8	Afghanistan	1958	7.450	1	0	0	0	0	0	0	...
9	Afghanistan	1959	7.450	1	0	0	0	0	0	0	...
10	Afghanistan	1960	7.450	1	0	0	0	0	0	0	...
11	Afghanistan	1961	7.450	1	0	0	0	0	0	0	...
12	Afghanistan	1962	7.450	1	0	0	0	0	0	0	...
13	Afghanistan	1963	7.450	1	0	0	0	0	0	0	...
14	Afghanistan	1964	7.450	1	0	0	0	0	0	0	...
15	Afghanistan	1965	7.450	1	0	0	0	0	0	0	...
16	Afghanistan	1966	7.450	1	0	0	0	0	0	0	...
17	Afghanistan	1967	7.450	1	0	0	0	0	0	0	...
18	Afghanistan	1968	7.450	1	0	0	0	0	0	0	...
19	Afghanistan	1969	7.450	1	0	0	0	0	0	0	...
20	Afghanistan	1970	7.450	1	0	0	0	0	0	0	...
21	Afghanistan	1971	7.450	1	0	0	0	0	0	0	...
22	Afghanistan	1972	7.450	1	0	0	0	0	0	0	...
23	Afghanistan	1973	7.450	1	0	0	0	0	0	0	...
24	Afghanistan	1974	7.450	1	0	0	0	0	0	0	...
25	Afghanistan	1975	7.450	1	0	0	0	0	0	0	...
26	Afghanistan	1976	7.450	1	0	0	0	0	0	0	...
27	Afghanistan	1977	7.449	1	0	0	0	0	0	0	...
28	Afghanistan	1978	7.449	1	0	0	0	0	0	0	...
29	Afghanistan	1979	7.449	1	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...
17436	Zimbabwe	1991	4.676	0	0	0	0	0	0	0	...
17437	Zimbabwe	1992	4.503	0	0	0	0	0	0	0	...
17438	Zimbabwe	1993	4.346	0	0	0	0	0	0	0	...
17439	Zimbabwe	1994	4.207	0	0	0	0	0	0	0	...
17440	Zimbabwe	1995	4.088	0	0	0	0	0	0	0	...

	Entity	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	...
17441	Zimbabwe	1996	3.989	0	0	0	0	0	0	0	...
17442	Zimbabwe	1997	3.907	0	0	0	0	0	0	0	...
17443	Zimbabwe	1998	3.839	0	0	0	0	0	0	0	...
17444	Zimbabwe	1999	3.786	0	0	0	0	0	0	0	...
17445	Zimbabwe	2000	3.748	0	0	0	0	0	0	0	...
17446	Zimbabwe	2001	3.725	0	0	0	0	0	0	0	...
17447	Zimbabwe	2002	3.718	0	0	0	0	0	0	0	...
17448	Zimbabwe	2003	3.725	0	0	0	0	0	0	0	...
17449	Zimbabwe	2004	3.744	0	0	0	0	0	0	0	...
17450	Zimbabwe	2005	3.775	0	0	0	0	0	0	0	...
17451	Zimbabwe	2006	3.819	0	0	0	0	0	0	0	...
17452	Zimbabwe	2007	3.873	0	0	0	0	0	0	0	...
17453	Zimbabwe	2008	3.931	0	0	0	0	0	0	0	...
17454	Zimbabwe	2009	3.988	0	0	0	0	0	0	0	...
17455	Zimbabwe	2010	4.034	0	0	0	0	0	0	0	...
17456	Zimbabwe	2011	4.059	0	0	0	0	0	0	0	...
17457	Zimbabwe	2012	4.058	0	0	0	0	0	0	0	...
17458	Zimbabwe	2013	4.030	0	0	0	0	0	0	0	...
17459	Zimbabwe	2014	3.974	0	0	0	0	0	0	0	...
17460	Zimbabwe	2015	3.896	0	0	0	0	0	0	0	...
17461	Zimbabwe	2016	3.804	0	0	0	0	0	0	0	...
17462	Zimbabwe	2017	3.707	0	0	0	0	0	0	0	...
17463	Zimbabwe	2018	3.615	0	0	0	0	0	0	0	...
17464	Zimbabwe	2019	3.531	0	0	0	0	0	0	0	...
17465	Zimbabwe	2020	3.460	0	0	0	0	0	0	0	...

17466 rows × 249 columns

Now that the countries are One Hot Encoded, we can drop the column Country in oheFert

In [1072]:
```
oheFert = oheFert.drop(["Entity"], axis = 1)
```

In [1073]:
```
oheFert.head()
```

Out[1073]:

	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	...	Vanua
0	1950	7.45	1	0	0	0	0	0	0	0	...	
1	1951	7.45	1	0	0	0	0	0	0	0	...	
2	1952	7.45	1	0	0	0	0	0	0	0	...	
3	1953	7.45	1	0	0	0	0	0	0	0	...	
4	1954	7.45	1	0	0	0	0	0	0	0	...	

5 rows × 248 columns

With oheFert containing the One Hot Encoding of the countries with properly labeled column titles and the unnecessary columns dropped, we can build a logistic regression model on this data. For the model, X will be the Year and all of the OHE country labels while Y will be the Fertility rate. Will just use a default 80/20 split with the large number of entries available

In [1074]:
```
from sklearn.model_selection import train_test_split
# Source: https://www.geeksforgeeks.org/how-to-split-a-dataset-into-train-and-test-sets-using-python/
X = oheFert.drop(["Rate"], axis = 1)
y = oheFert.iloc[:, 1:2]

trainX, testX, trainY, testY = train_test_split(X, y, test_size=.2)

print('X train shape:',trainX.shape)
print('X test shape:',testX.shape)
print('Y train shape:',trainY.shape)
print('Y test shape:',testY.shape)
```

```
X train shape: (13972, 247)
X test shape: (3494, 247)
Y train shape: (13972, 1)
Y test shape: (3494, 1)
```

In [1075]:
```
mlr = LinearRegression()
mlr.fit(trainX, trainY)
```

Out[1075]:
```
LinearRegression()
```

Now let's see what the 2122 fertility rate predictions look like for each country...

In [1076]:
```
class_mapping = {label: idx for idx, label in enumerate(np.unique(fertility['Entit
y']))}
class_mapping
```

Out[1076]:
```
{'Afghanistan': 0,
 'Africa': 1,
 'Albania': 2,
 'Algeria': 3,
 'Angola': 4,
 'Antigua and Barbuda': 5,
 'Argentina': 6,
 'Armenia': 7,
 'Aruba': 8,
 'Asia': 9,
 'Asia, Central': 10,
 'Australia': 11,
 'Australia & New Zealand': 12,
 'Austria': 13,
 'Azerbaijan': 14,
 'Bahamas': 15,
 'Bahrain': 16,
 'Bangladesh': 17,
 'Barbados': 18,
 'Belarus': 19,
 'Belgium': 20,
 'Belize': 21,
 'Benin': 22,
 'Bhutan': 23,
 'Bolivia': 24,
 'Bosnia and Herzegovina': 25,
 'Botswana': 26,
 'Brazil': 27,
 'Brunei': 28,
 'Bulgaria': 29,
 'Burkina Faso': 30,
 'Burundi': 31,
 'Cambodia': 32,
 'Cameroon': 33,
 'Canada': 34,
 'Cape Verde': 35,
 'Caribbean': 36,
 'Central African Republic': 37,
 'Central America': 38,
 'Central and Southern Asia': 39,
 'Chad': 40,
 'Channel Islands': 41,
 'Chile': 42,
 'China': 43,
 'Colombia': 44,
 'Comoros': 45,
 'Congo': 46,
 'Costa Rica': 47,
 "Cote d'Ivoire": 48,
 'Croatia': 49,
 'Cuba': 50,
 'Curacao': 51,
 'Cyprus': 52,
 'Czechia': 53,
 'Democratic Republic of Congo': 54,
 'Denmark': 55,
 'Djibouti': 56,
 'Dominican Republic': 57,
 'Eastern Africa': 58,
 'Eastern Asia': 59,
```

```
'Eastern Europe': 60,
'Eastern and South-Eastern Asia': 61,
'Ecuador': 62,
'Egypt': 63,
'El Salvador': 64,
'Equatorial Guinea': 65,
'Eritrea': 66,
'Estonia': 67,
'Eswatini': 68,
'Ethiopia': 69,
'Europe': 70,
'Europe and Northern America': 71,
'Europe, Western': 72,
'Fiji': 73,
'Finland': 74,
'France': 75,
'French Guiana': 76,
'French Polynesia': 77,
'Gabon': 78,
'Gambia': 79,
'Georgia': 80,
'Germany': 81,
'Ghana': 82,
'Greece': 83,
'Grenada': 84,
'Guadeloupe': 85,
'Guam': 86,
'Guatemala': 87,
'Guinea': 88,
'Guinea-Bissau': 89,
'Guyana': 90,
'Haiti': 91,
'High income countries': 92,
'Honduras': 93,
'Hong Kong': 94,
'Hungary': 95,
'Iceland': 96,
'India': 97,
'Indonesia': 98,
'Iran': 99,
'Iraq': 100,
'Ireland': 101,
'Israel': 102,
'Italy': 103,
'Jamaica': 104,
'Japan': 105,
'Jordan': 106,
'Kazakhstan': 107,
'Kenya': 108,
'Kiribati': 109,
'Kuwait': 110,
'Kyrgyzstan': 111,
'Land-locked Developing Countries (LLDC)': 112,
'Laos': 113,
'Latin America and the Caribbean': 114,
'Latvia': 115,
'Least Developed Countries': 116,
'Lebanon': 117,
'Lesotho': 118,
'Less Developed Regions': 119,
```

```
'Less Developed Regions, excluding China': 120,
'Less Developed Regions, excluding Least Developed Countries': 121,
'Liberia': 122,
'Libya': 123,
'Lithuania': 124,
'Low-income countries': 125,
'Lower-middle-income countries': 126,
'Luxembourg': 127,
'Macao': 128,
'Madagascar': 129,
'Malawi': 130,
'Malaysia': 131,
'Maldives': 132,
'Mali': 133,
'Malta': 134,
'Martinique': 135,
'Mauritania': 136,
'Mauritius': 137,
'Mayotte': 138,
'Melanesia': 139,
'Mexico': 140,
'Micronesia (country)': 141,
'Middle Africa': 142,
'Middle-income countries': 143,
'Moldova': 144,
'Mongolia': 145,
'Montenegro': 146,
'More Developed Regions': 147,
'Morocco': 148,
'Mozambique': 149,
'Myanmar': 150,
'Namibia': 151,
'Nepal': 152,
'Netherlands': 153,
'New Caledonia': 154,
'New Zealand': 155,
'Nicaragua': 156,
'Niger': 157,
'Nigeria': 158,
'No income group available': 159,
'North Korea': 160,
'North Macedonia': 161,
'Northern Africa': 162,
'Northern Africa and Western Asia': 163,
'Northern America': 164,
'Northern Europe': 165,
'Norway': 166,
'Oceania': 167,
'Oceania (excluding Australia and New Zealand)': 168,
'Oman': 169,
'Pakistan': 170,
'Palestine': 171,
'Panama': 172,
'Papua New Guinea': 173,
'Paraguay': 174,
'Peru': 175,
'Philippines': 176,
'Poland': 177,
'Portugal': 178,
'Puerto Rico': 179,
```

```
'Qatar': 180,
'Reunion': 181,
'Romania': 182,
'Russia': 183,
'Rwanda': 184,
'Saint Lucia': 185,
'Saint Vincent and the Grenadines': 186,
'Samoa': 187,
'Sao Tome and Principe': 188,
'Saudi Arabia': 189,
'Senegal': 190,
'Serbia': 191,
'Seychelles': 192,
'Sierra Leone': 193,
'Singapore': 194,
'Slovakia': 195,
'Slovenia': 196,
'Small Island Developing States (SIDS)': 197,
'Solomon Islands': 198,
'Somalia': 199,
'South Africa': 200,
'South America': 201,
'South Eastern Asia': 202,
'South Korea': 203,
'South Sudan': 204,
'Southern Africa': 205,
'Southern Asia': 206,
'Southern Europe': 207,
'Spain': 208,
'Sri Lanka': 209,
'Sub-Saharan Africa': 210,
'Sudan': 211,
'Suriname': 212,
'Sweden': 213,
'Switzerland': 214,
'Syria': 215,
'Taiwan': 216,
'Tajikistan': 217,
'Tanzania': 218,
'Thailand': 219,
'Timor': 220,
'Togo': 221,
'Tonga': 222,
'Trinidad and Tobago': 223,
'Tunisia': 224,
'Turkey': 225,
'Turkmenistan': 226,
'Uganda': 227,
'Ukraine': 228,
'United Arab Emirates': 229,
'United Kingdom': 230,
'United States': 231,
'United States Virgin Islands': 232,
'Upper-middle-income countries': 233,
'Uruguay': 234,
'Uzbekistan': 235,
'Vanuatu': 236,
'Venezuela': 237,
'Vietnam': 238,
'Western Africa': 239,
```

```
 'Western Asia': 240,
 'Western Sahara': 241,
 'World': 242,
 'Yemen': 243,
 'Zambia': 244,
 'Zimbabwe': 245}
```

In [1077]:
```
predictedFertilityRate = {}
fertCountries = np.unique(fertility['Entity'])
for key, value in class_mapping.items():
    temp = [0] * len(fertCountries)
    temp[value] = 1
    a = np.insert(temp, 0, 2122)
    predictedFertilityRate[key] = mlr.predict([a])

predictedFertilityRate
```

Out[1077]:
```
{'Afghanistan': array([[0.23898125]]),
 'Africa': array([[-0.85512543]]),
 'Albania': array([[-3.15453339]]),
 'Algeria': array([[-1.34526634]]),
 'Angola': array([[-0.04259491]]),
 'Antigua and Barbuda': array([[-3.92407227]]),
 'Argentina': array([[-3.81211472]]),
 'Armenia': array([[-3.99417496]]),
 'Aruba': array([[-3.90851974]]),
 'Asia': array([[-2.93966866]]),
 'Asia, Central': array([[-2.8360157]]),
 'Australia': array([[-4.43390274]]),
 'Australia & New Zealand': array([[-4.31470299]]),
 'Austria': array([[-5.00269318]]),
 'Azerbaijan': array([[-3.21020126]]),
 'Bahamas': array([[-3.84174538]]),
 'Bahrain': array([[-2.18481064]]),
 'Bangladesh': array([[-1.89444733]]),
 'Barbados': array([[-4.21033478]]),
 'Belarus': array([[-4.81193924]]),
 'Belgium': array([[-4.84762192]]),
 'Belize': array([[-1.96325302]]),
 'Benin': array([[-0.60416603]]),
 'Bhutan': array([[-1.67059326]]),
 'Bolivia': array([[-1.83572769]]),
 'Bosnia and Herzegovina': array([[-4.42735481]]),
 'Botswana': array([[-1.78780365]]),
 'Brazil': array([[-3.00029755]]),
 'Brunei': array([[-2.83661842]]),
 'Bulgaria': array([[-4.96969986]]),
 'Burkina Faso': array([[-0.33137894]]),
 'Burundi': array([[0.17616653]]),
 'Cambodia': array([[-1.63201332]]),
 'Cameroon': array([[-0.94472885]]),
 'Canada': array([[-4.59837341]]),
 'Cape Verde': array([[-1.63419151]]),
 'Caribbean': array([[-3.1633873]]),
 'Central African Republic': array([[-1.19147873]]),
 'Central America': array([[-2.22732925]]),
 'Central and Southern Asia': array([[-2.28072739]]),
 'Chad': array([[-0.09568787]]),
 'Channel Islands': array([[-4.98289299]]),
 'Chile': array([[-3.72587585]]),
 'China': array([[-3.45089531]]),
 'Colombia': array([[-2.85119057]]),
 'Comoros': array([[-0.73771095]]),
 'Congo': array([[-1.30426598]]),
 'Costa Rica': array([[-2.9371376]]),
 "Cote d'Ivoire": array([[-0.09073448]]),
 'Croatia': array([[-4.90274811]]),
 'Cuba': array([[-4.11509705]]),
 'Curacao': array([[-3.79893303]]),
 'Cyprus': array([[-4.34037018]]),
 'Czechia': array([[-4.86389923]]),
 'Democratic Republic of Congo': array([[-0.34677124]]),
 'Denmark': array([[-4.75460052]]),
 'Djibouti': array([[-1.23276329]]),
 'Dominican Republic': array([[-2.25603676]]),
 'Eastern Africa': array([[-0.32389259]]),
 'Eastern Asia': array([[-3.51036644]]),
```

```
'Eastern Europe': array([[-4.86502457]]),
'Eastern and South-Eastern Asia': array([[-3.35754585]]),
'Ecuador': array([[-2.20702744]]),
'Egypt': array([[-1.78464508]]),
'El Salvador': array([[-2.320261]]),
'Equatorial Guinea': array([[-1.1959095]]),
'Eritrea': array([[-0.73449707]]),
'Estonia': array([[-4.96247101]]),
'Eswatini': array([[-1.35643387]]),
'Ethiopia': array([[-0.2208252]]),
'Europe': array([[-4.79895592]]),
'Europe and Northern America': array([[-4.73627281]]),
'Europe, Western': array([[-4.88933182]]),
'Fiji': array([[-2.61088943]]),
'Finland': array([[-4.73484039]]),
'France': array([[-4.54165649]]),
'French Guiana': array([[-2.70277214]]),
'French Polynesia': array([[-2.91158295]]),
'Gabon': array([[-1.98927307]]),
'Gambia': array([[-0.7126236]]),
'Georgia': array([[-4.53037834]]),
'Germany': array([[-4.97987747]]),
'Ghana': array([[-1.081707]]),
'Greece': array([[-4.8157444]]),
'Grenada': array([[-2.72297096]]),
'Guadeloupe': array([[-3.25189972]]),
'Guam': array([[-3.13412285]]),
'Guatemala': array([[-1.23421288]]),
'Guinea': array([[-0.80144501]]),
'Guinea-Bissau': array([[-0.86274338]]),
'Guyana': array([[-2.65328217]]),
'Haiti': array([[-1.55969238]]),
'High income countries': array([[-4.51169968]]),
'Honduras': array([[-1.21498489]]),
'Hong Kong': array([[-4.42567444]]),
'Hungary': array([[-4.93996811]]),
'Iceland': array([[-4.08883858]]),
'India': array([[-2.44017601]]),
'Indonesia': array([[-2.83240509]]),
'Iran': array([[-2.13546181]]),
'Iraq': array([[-0.92645073]]),
'Ireland': array([[-4.04616547]]),
'Israel': array([[-3.37707138]]),
'Italy': array([[-4.95538139]]),
'Jamaica': array([[-3.15039444]]),
'Japan': array([[-4.92682457]]),
'Jordan': array([[-0.85914993]]),
'Kazakhstan': array([[-3.56219864]]),
'Kenya': array([[-0.34152412]]),
'Kiribati': array([[-1.8116684]]),
'Kuwait': array([[-2.11288452]]),
'Kyrgyzstan': array([[-2.76641846]]),
'Land-locked Developing Countries (LLDC)': array([[-1.17578506]]),
'Laos': array([[-1.64536476]]),
'Latin America and the Caribbean': array([[-2.90846634]]),
'Latvia': array([[-5.08732033]]),
'Least Developed Countries': array([[-0.83133698]]),
'Lebanon': array([[-3.05681992]]),
'Lesotho': array([[-1.97237778]]),
'Less Developed Regions': array([[-2.4934063]]),
```

```
 'Less Developed Regions, excluding China': array([[-2.13114548]]),
 'Less Developed Regions, excluding Least Developed Countries': array([[-2.7880344
4]]),
 'Liberia': array([[-0.56507874]]),
 'Libya': array([[-1.60856438]]),
 'Lithuania': array([[-4.73801994]]),
 'Low-income countries': array([[-0.7018261]]),
 'Lower-middle-income countries': array([[-2.1813755]]),
 'Luxembourg': array([[-5.0069046]]),
 'Macao': array([[-4.58517647]]),
 'Madagascar': array([[-0.57772255]]),
 'Malawi': array([[-0.32839012]]),
 'Malaysia': array([[-2.72172165]]),
 'Maldives': array([[-1.70655251]]),
 'Mali': array([[0.02939987]]),
 'Malta': array([[-4.52711868]]),
 'Martinique': array([[-3.54408836]]),
 'Mauritania': array([[-0.75640488]]),
 'Mauritius': array([[-3.64672852]]),
 'Mayotte': array([[-0.3981266]]),
 'Melanesia': array([[-1.72475052]]),
 'Mexico': array([[-2.29901314]]),
 'Micronesia (country)': array([[-1.34683037]]),
 'Middle Africa': array([[-0.50154495]]),
 'Middle-income countries': array([[-2.80260468]]),
 'Moldova': array([[-4.50569916]]),
 'Mongolia': array([[-1.91043091]]),
 'Montenegro': array([[-4.25136948]]),
 'More Developed Regions': array([[-4.72350693]]),
 'Morocco': array([[-1.96083069]]),
 'Mozambique': array([[-0.66298676]]),
 'Myanmar': array([[-2.5468502]]),
 'Namibia': array([[-1.57372093]]),
 'Nepal': array([[-2.0723114]]),
 'Netherlands': array([[-4.77436638]]),
 'New Caledonia': array([[-3.07114029]]),
 'New Zealand': array([[-4.1130619]]),
 'Nicaragua': array([[-1.75098419]]),
 'Niger': array([[0.7899704]]),
 'Nigeria': array([[-0.57030869]]),
 'No income group available': array([[-2.96942329]]),
 'North Korea': array([[-3.90403366]]),
 'North Macedonia': array([[-4.06114197]]),
 'Northern Africa': array([[-1.66374969]]),
 'Northern Africa and Western Asia': array([[-1.84728813]]),
 'Northern America': array([[-4.45305443]]),
 'Northern Europe': array([[-4.74157715]]),
 'Norway': array([[-4.63325691]]),
 'Oceania': array([[-3.81501198]]),
 'Oceania (excluding Australia and New Zealand)': array([[-1.69425011]]),
 'Oman': array([[-0.63714981]]),
 'Pakistan': array([[-1.03269196]]),
 'Palestine': array([[-0.29605484]]),
 'Panama': array([[-2.76881981]]),
 'Papua New Guinea': array([[-1.5855999]]),
 'Paraguay': array([[-2.09126091]]),
 'Peru': array([[-2.11268234]]),
 'Philippines': array([[-1.78727722]]),
 'Poland': array([[-4.61548424]]),
 'Portugal': array([[-4.5819931]]),
```

```
'Puerto Rico': array([[-4.00695038]]),
'Qatar': array([[-1.92725754]]),
'Reunion': array([[-2.97447205]]),
'Romania': array([[-4.63700867]]),
'Russia': array([[-4.75844955]]),
'Rwanda': array([[0.1340332]]),
'Saint Lucia': array([[-2.58043289]]),
'Saint Vincent and the Grenadines': array([[-2.44760513]]),
'Samoa': array([[-0.89452553]]),
'Sao Tome and Principe': array([[-1.05484772]]),
'Saudi Arabia': array([[-1.11575317]]),
'Senegal': array([[-0.40693283]]),
'Serbia': array([[-4.62240028]]),
'Seychelles': array([[-3.02716255]]),
'Sierra Leone': array([[-0.71220779]]),
'Singapore': array([[-3.97673607]]),
'Slovakia': array([[-4.63519287]]),
'Slovenia': array([[-4.87284279]]),
'Small Island Developing States (SIDS)': array([[-2.87634468]]),
'Solomon Islands': array([[-0.91756248]]),
'Somalia': array([[0.47727203]]),
'South Africa': array([[-2.39646721]]),
'South America': array([[-3.02489281]]),
'South Eastern Asia': array([[-2.66703224]]),
'South Korea': array([[-3.76076126]]),
'South Sudan': array([[-0.36468697]]),
'Southern Africa': array([[-2.30614853]]),
'Southern Asia': array([[-2.28136635]]),
'Southern Europe': array([[-4.81617355]]),
'Spain': array([[-4.76291847]]),
'Sri Lanka': array([[-3.25309563]]),
'Sub-Saharan Africa': array([[-0.60279274]]),
'Sudan': array([[-0.679142]]),
'Suriname': array([[-2.57509613]]),
'Sweden': array([[-4.77407074]]),
'Switzerland': array([[-4.95042419]]),
'Syria': array([[-1.12072945]]),
'Taiwan': array([[-3.75586891]]),
'Tajikistan': array([[-1.62481689]]),
'Tanzania': array([[-0.61344719]]),
'Thailand': array([[-3.28716278]]),
'Timor': array([[-1.20671844]]),
'Togo': array([[-0.62977982]]),
'Tonga': array([[-1.47232819]]),
'Trinidad and Tobago': array([[-3.71532059]]),
'Tunisia': array([[-2.38318825]]),
'Turkey': array([[-2.64260483]]),
'Turkmenistan': array([[-2.26392174]]),
'Uganda': array([[-0.08657837]]),
'Ukraine': array([[-4.89299965]]),
'United Arab Emirates': array([[-2.25522232]]),
'United Kingdom': array([[-4.76647568]]),
'United States': array([[-4.52511978]]),
'United States Virgin Islands': array([[-3.18656158]]),
'Upper-middle-income countries': array([[-3.33222961]]),
'Uruguay': array([[-4.24557114]]),
'Uzbekistan': array([[-2.31839752]]),
'Vanuatu': array([[-1.32264328]]),
'Venezuela': array([[-2.59537125]]),
'Vietnam': array([[-2.64289665]]),
```

```
 'Western Africa': array([[-0.49396133]]),
 'Western Asia': array([[-2.11762619]]),
 'Western Sahara': array([[-2.04400444]]),
 'World': array([[-3.08009911]]),
 'Yemen': array([[0.49050331]]),
 'Zambia': array([[-0.38931084]]),
 'Zimbabwe': array([[-1.12797546]])}
```

That doesn't look very realistic of the model predicting so many negative fertility rates in 2122. Below will look at some simple statistical tests to see how the model does on the testing data

In [1079]:
```
yhat = []
for r in range(len(testX)):
    toPredict = testX.iloc[r, :]
    val = mlr.predict([toPredict])
    yhat.append(val[0][0])
yhat[1:10]
```

Out[1079]:
```
[6.30897331237793,
 2.4797325134277344,
 1.9596443176269531,
 7.7154693603515625,
 0.7846279144287109,
 7.026912689208984,
 3.684581756591797,
 5.774026870727539,
 0.910614013671875]
```

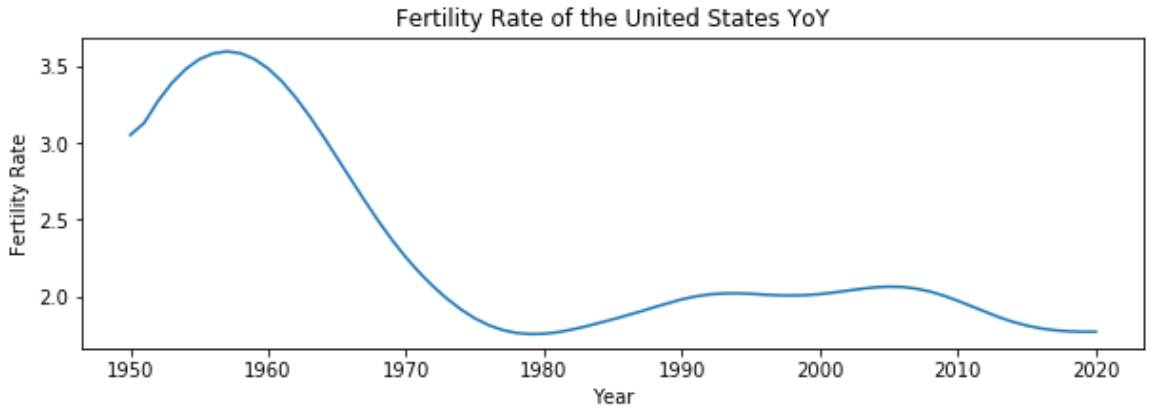
In [1080]:
```
# Source : https://www.datatechnotes.com/2019/10/accuracy-check-in-python-mae-mse-rmse-r.html
import sklearn.metrics as metrics
mae = metrics.mean_absolute_error(testY, yhat)
mse = metrics.mean_squared_error(testY, yhat)
rmse = np.sqrt(mse) #mse**(0.5)
r2 = metrics.r2_score(testY,yhat)

print("Results of sklearn.metrics:")
print("MAE:",mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R-Squared:", r2)
```

```
Results of sklearn.metrics:
MAE: 0.5708761918537946
MSE: 0.49901901069214716
RMSE: 0.7064127764219353
R-Squared: 0.8715810868374474
```

Errors don't seem that far off and a pretty good R^2 score for the model against the testing data. However, will need to adjust something in order to get more realistic results as all of the negative predictions for fertility rate in 2122 are physically impossible

Let's take a look at the United States' Fertility Rate Year over Year and see if something pops out that can applied to adjust the model to produce more realistic predictions of 2122 Fertility Rates

In [1081]:
```
import matplotlib.pyplot as plt
usFert = oheFert.where(oheFert['United States'] == 1)
plt.figure(figsize=(10, 3))
plt.plot(usFert['Year'],usFert['Rate'])
plt.ylabel('Fertility Rate')
plt.xlabel('Year')
plt.title('Fertility Rate of the United States YoY')
plt.show()
```



Interesting view of the fertility rate of the United States year over year. It seems to be oscilating in a the 1-2 range since around 1980. Going to make a visualization below that has a slider to select a specific country to have their fertility rate plotted YoY like above

In [1082]:
```
import plotly.express as px

# Neat little visualization of fertility rate for a selected country
# Source: https://plotly.com/python/sliders/
# Source: https://plotly.com/python/line-and-scatter/
fig = px.scatter(fertility, x="Year", y="Rate", animation_frame="Entity", animation
_group="Year",
                 range_x=[1950,2020], range_y=[0,10],color='Rate')

fig["layout"].pop("updatemenus") # optional, drop animation buttons
fig.show()
```

Each country seems to start off with a substantially higher fertility rate that drops pretty steeply from 1950 to 1980ish. Around 1980, each country's declining fertility rate seems to level off at range specific to that country (typically between 1-3)

Two new approaches to try after getting a better visualization of each country's fertility rate changing YoY:

- In order to take the more recent changes in technology and society, will just look at values of the 21st century to predict the fertility rate of each country for 2122 to take into account the leveling off trend shown above
- Change the model to take into account the logarithmic appearing curve of each country

In [1084]:
```
fertRecent = oheFert.where(oheFert.Year >=int(2000))
fertRecent = fertRecent.dropna()
fertRecent.head()
```

Out[1084]:

	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	...	Va
50	2000.0	7.485	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
51	2001.0	7.387	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
52	2002.0	7.272	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
53	2003.0	7.148	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
54	2004.0	7.016	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	

5 rows × 248 columns

In [1085]:
```
fertRecent['Year'] = fertRecent['Year'].astype(int)
fertRecent.head()
```

Out[1085]:

	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	...	Vani
50	2000	7.485	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
51	2001	7.387	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
52	2002	7.272	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
53	2003	7.148	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
54	2004	7.016	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	

5 rows × 248 columns

fertRecent is now ready to be fed into the regression model...

In [1086]:
```
# Source: https://www.geeksforgeeks.org/how-to-split-a-dataset-into-train-and-test-sets-using-python/
X = fertRecent.drop(["Rate"], axis = 1)
y = fertRecent.iloc[:, 1:2]

trainX, testX, trainY, testY = train_test_split(X, y, test_size=.2)

print('X train shape:',trainX.shape)
print('X test shape:',testX.shape)
print('Y train shape:',trainY.shape)
print('Y test shape:',testY.shape)
mlr = LinearRegression()
mlr.fit(trainX, trainY)
```

```
X train shape: (4132, 247)
X test shape: (1034, 247)
Y train shape: (4132, 1)
Y test shape: (1034, 1)
```

Out[1086]:
```
LinearRegression()
```

In [1087]:
```
recentPredictedFertilityRate = {}

for key, value in class_mapping.items():
    temp = [0] * len(fertCountries)
    temp[value] = 1
    a = []
    a = np.insert(temp, 0, 2122)
    recentPredictedFertilityRate[key] = mlr.predict([a])

recentPredictedFertilityRate
```

Out[1087]:

```
{'Afghanistan': array([[2.83790551]]),
 'Africa': array([[1.56566676]]),
 'Albania': array([[-1.47551718]]),
 'Algeria': array([[-0.45842863]]),
 'Angola': array([[2.92769]]),
 'Antigua and Barbuda': array([[-1.1732842]]),
 'Argentina': array([[-0.85373866]]),
 'Armenia': array([[-1.49095106]]),
 'Aruba': array([[-1.41957954]]),
 'Asia': array([[-0.93293427]]),
 'Asia, Central': array([[-0.54414509]]),
 'Australia': array([[-1.42698155]]),
 'Australia & New Zealand': array([[-1.32515594]]),
 'Austria': array([[-1.77345426]]),
 'Azerbaijan': array([[-1.25823518]]),
 'Bahamas': array([[-1.34669446]]),
 'Bahrain': array([[-0.97178182]]),
 'Bangladesh': array([[-0.80763227]]),
 'Barbados': array([[-1.53967443]]),
 'Belarus': array([[-1.72992212]]),
 'Belgium': array([[-1.51957291]]),
 'Belize': array([[-0.46145922]]),
 'Benin': array([[2.10983111]]),
 'Bhutan': array([[-0.77853375]]),
 'Bolivia': array([[0.06723109]]),
 'Bosnia and Herzegovina': array([[-1.91351076]]),
 'Botswana': array([[-0.18697282]]),
 'Brazil': array([[-1.32698854]]),
 'Brunei': array([[-1.22136628]]),
 'Bulgaria': array([[-1.78317109]]),
 'Burkina Faso': array([[2.61656079]]),
 'Burundi': array([[2.91722113]]),
 'Cambodia': array([[-0.31712083]]),
 'Cameroon': array([[1.8248174]]),
 'Canada': array([[-1.64388001]]),
 'Cape Verde': array([[-0.52661386]]),
 'Caribbean': array([[-0.88868607]]),
 'Central African Republic': array([[1.92185491]]),
 'Central America': array([[-0.72963065]]),
 'Central and Southern Asia': array([[-0.48070881]]),
 'Chad': array([[3.24265969]]),
 'Channel Islands': array([[-1.69826001]]),
 'Chile': array([[-1.38021933]]),
 'China': array([[-1.59939134]]),
 'Colombia': array([[-1.14560094]]),
 'Comoros': array([[1.54130086]]),
 'Congo': array([[1.48065828]]),
 'Costa Rica': array([[-1.28352859]]),
 "Cote d'Ivoire": array([[1.96653196]]),
 'Croatia': array([[-1.76214957]]),
 'Cuba': array([[-1.60038946]]),
 'Curacao': array([[-1.15612647]]),
 'Cyprus': array([[-1.76427471]]),
 'Czechia': array([[-1.79831832]]),
 'Democratic Republic of Congo': array([[3.21262234]]),
 'Denmark': array([[-1.44143866]]),
 'Djibouti': array([[0.14336289]]),
 'Dominican Republic': array([[-0.68494148]]),
 'Eastern Africa': array([[1.93031065]]),
 'Eastern Asia': array([[-1.62300967]]),
```

```
'Eastern Europe': array([[-1.7458875]]),
'Eastern and South-Eastern Asia': array([[-1.39179215]]),
'Ecuador': array([[-0.55008948]]),
'Egypt': array([[0.02222113]]),
'El Salvador': array([[-0.88610405]]),
'Equatorial Guinea': array([[1.94295325]]),
'Eritrea': array([[1.35775094]]),
'Estonia': array([[-1.67757123]]),
'Eswatini': array([[0.14956215]]),
'Ethiopia': array([[2.09270031]]),
'Europe': array([[-1.64866569]]),
'Europe and Northern America': array([[-1.56231977]]),
'Europe, Western': array([[-1.567602]]),
'Fiji': array([[-0.39302745]]),
'Finland': array([[-1.52503703]]),
'France': array([[-1.28238312]]),
'French Guiana': array([[0.29741032]]),
'French Polynesia': array([[-1.08887095]]),
'Gabon': array([[0.93676735]]),
'Gambia': array([[2.32421715]]),
'Georgia': array([[-1.40519527]]),
'Germany': array([[-1.80755276]]),
'Ghana': array([[1.04099855]]),
'Greece': array([[-1.88099735]]),
'Grenada': array([[-0.97178489]]),
'Guadeloupe': array([[-1.02581308]]),
'Guam': array([[-0.70383865]]),
'Guatemala': array([[0.28003138]]),
'Guinea': array([[2.12880474]]),
'Guinea-Bissau': array([[1.85771947]]),
'Guyana': array([[-0.5325583]]),
'Haiti': array([[0.29331065]]),
'High income countries': array([[-1.49378924]]),
'Honduras': array([[-0.16066714]]),
'Hong Kong': array([[-2.07257191]]),
'Hungary': array([[-1.84692811]]),
'Iceland': array([[-1.26550489]]),
'India': array([[-0.59123523]]),
'Indonesia': array([[-0.77767955]]),
'Iran': array([[-1.17752468]]),
'Iraq': array([[1.00651745]]),
'Ireland': array([[-1.27806977]]),
'Israel': array([[-0.22690223]]),
'Italy': array([[-1.86288929]]),
'Jamaica': array([[-1.00829087]]),
'Japan': array([[-1.8749459]]),
'Jordan': array([[0.26361715]]),
'Kazakhstan': array([[-0.53822235]]),
'Kenya': array([[1.02057165]]),
'Kiribati': array([[0.58630603]]),
'Kuwait': array([[-0.88911162]]),
'Kyrgyzstan': array([[-0.27335575]]),
'Land-locked Developing Countries (LLDC)': array([[1.23996753]]),
'Laos': array([[0.09095065]]),
'Latin America and the Caribbean': array([[-0.97570051]]),
'Latvia': array([[-1.67715648]]),
'Least Developed Countries': array([[1.29281895]]),
'Lebanon': array([[-1.1675694]]),
'Lesotho': array([[0.13669745]]),
'Less Developed Regions': array([[-0.50240714]]),
```

```
 'Less Developed Regions, excluding China': array([[-0.14196076]]),
 'Less Developed Regions, excluding Least Developed Countries': array([[-0.7890828
5]]),
 'Liberia': array([[1.80340261]]),
 'Libya': array([[-0.74462756]]),
 'Lithuania': array([[-1.71728924]]),
 'Low-income countries': array([[1.88787208]]),
 'Lower-middle-income countries': array([[-0.19898652]]),
 'Luxembourg': array([[-1.63744912]]),
 'Macao': array([[-2.11848572]]),
 'Madagascar': array([[1.4728896]]),
 'Malawi': array([[2.02184395]]),
 'Malaysia': array([[-1.00805057]]),
 'Maldives': array([[-1.04853756]]),
 'Mali': array([[3.22013372]]),
 'Malta': array([[-1.81046291]]),
 'Martinique': array([[-1.2422682]]),
 'Mauritania': array([[1.74602616]]),
 'Mauritius': array([[-1.59696453]]),
 'Mayotte': array([[1.13299642]]),
 'Melanesia': array([[0.62348175]]),
 'Mexico': array([[-0.85786602]]),
 'Micronesia (country)': array([[0.31103731]]),
 'Middle Africa': array([[2.80584259]]),
 'Middle-income countries': array([[-0.76910068]]),
 'Moldova': array([[-1.95808354]]),
 'Mongolia': array([[-0.77524087]]),
 'Montenegro': array([[-1.42501376]]),
 'More Developed Regions': array([[-1.58586466]]),
 'Morocco': array([[-0.66883141]]),
 'Mozambique': array([[2.12231045]]),
 'Myanmar': array([[-0.8126402]]),
 'Namibia': array([[0.36298754]]),
 'Nepal': array([[-0.59117655]]),
 'Netherlands': array([[-1.53444607]]),
 'New Caledonia': array([[-1.03405527]]),
 'New Zealand': array([[-1.20413662]]),
 'Nicaragua': array([[-0.57198367]]),
 'Niger': array([[4.15813944]]),
 'Nigeria': array([[2.56228373]]),
 'No income group available': array([[-0.65075339]]),
 'North Korea': array([[-1.26660243]]),
 'North Macedonia': array([[-1.69136628]]),
 'Northern Africa': array([[-0.0092875]]),
 'Northern Africa and Western Asia': array([[-0.14509884]]),
 'Northern America': array([[-1.3317566]]),
 'Northern Europe': array([[-1.43397819]]),
 'Norway': array([[-1.40681424]]),
 'Oceania': array([[-0.76922033]]),
 'Oceania (excluding Australia and New Zealand)': array([[0.53250405]]),
 'Oman': array([[-0.22849759]]),
 'Pakistan': array([[0.8125346]]),
 'Palestine': array([[1.08973548]]),
 'Panama': array([[-0.62446658]]),
 'Papua New Guinea': array([[0.7892714]]),
 'Paraguay': array([[-0.42124625]]),
 'Peru': array([[-0.70309859]]),
 'Philippines': array([[-0.04909836]]),
 'Poland': array([[-1.86149804]]),
 'Portugal': array([[-1.85402616]]),
```

```
'Puerto Rico': array([[-1.6432842]]),
'Qatar': array([[-1.01849708]]),
'Reunion': array([[-0.8730914]]),
'Romania': array([[-1.77016555]]),
'Russia': array([[-1.60718749]]),
'Rwanda': array([[1.35932414]]),
'Saint Lucia': array([[-1.57533324]]),
'Saint Vincent and the Grenadines': array([[-1.14462678]]),
'Samoa': array([[1.00296846]]),
'Sao Tome and Principe': array([[1.52818069]]),
'Saudi Arabia': array([[-0.28678257]]),
'Senegal': array([[1.78166339]]),
'Serbia': array([[-1.64608827]]),
'Seychelles': array([[-0.86035493]]),
'Sierra Leone': array([[1.9732954]]),
'Singapore': array([[-1.9733994]]),
'Slovakia': array([[-1.84071221]]),
'Slovenia': array([[-1.77697433]]),
'Small Island Developing States (SIDS)': array([[-0.61454491]]),
'Solomon Islands': array([[1.22690429]]),
'Somalia': array([[3.58582801]]),
'South Africa': array([[-0.66585174]]),
'South America': array([[-1.09231309]]),
'South Eastern Asia': array([[-0.83908721]]),
'South Korea': array([[-2.04796453]]),
'South Sudan': array([[2.13185779]]),
'Southern Africa': array([[-0.56846398]]),
'Southern Asia': array([[-0.47837354]]),
'Southern Europe': array([[-1.80544576]]),
'Spain': array([[-1.87783401]]),
'Sri Lanka': array([[-0.96462548]]),
'Sub-Saharan Africa': array([[1.99945386]]),
'Sudan': array([[1.67437473]]),
'Suriname': array([[-0.59381266]]),
'Sweden': array([[-1.42057954]]),
'Switzerland': array([[-1.73644332]]),
'Syria': array([[0.13722499]]),
'Taiwan': array([[-2.03754045]]),
'Tajikistan': array([[0.42936628]]),
'Tanzania': array([[2.15518603]]),
'Thailand': array([[-1.64310481]]),
'Timor': array([[1.82901821]]),
'Togo': array([[1.63877761]]),
'Tonga': array([[0.69971801]]),
'Trinidad and Tobago': array([[-1.46344087]]),
'Tunisia': array([[-1.08545827]]),
'Turkey': array([[-1.01420042]]),
'Turkmenistan': array([[-0.41585145]]),
'Uganda': array([[2.75748304]]),
'Ukraine': array([[-1.88821444]]),
'United Arab Emirates': array([[-1.36749418]]),
'United Kingdom': array([[-1.39331872]]),
'United States': array([[-1.26510775]]),
'United States Virgin Islands': array([[-0.98361568]]),
'Upper-middle-income countries': array([[-1.36223789]]),
'Uruguay': array([[-1.15825571]]),
'Uzbekistan': array([[-0.72854214]]),
'Vanuatu': array([[0.85901352]]),
'Venezuela': array([[-0.72733737]]),
'Vietnam': array([[-1.28659449]]),
```

```
'Western Africa': array([[2.39793264]]),
'Western Asia': array([[-0.27929693]]),
'Western Sahara': array([[-0.58422315]]),
'World': array([[-0.64944912]]),
'Yemen': array([[1.57041161]]),
'Zambia': array([[2.16750742]]),
'Zimbabwe': array([[0.5961784]])}
```

The model still predicts quite a few negative fertility rates even when it is just being trained on data of the 21st century. Time to try the second approach

In [1088]:
```
fertScaled = oheFert
```

In [1089]:
```
fertScaled.head()
```

Out[1089]:

	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	...	Vanua
0	1950	7.45	1	0	0	0	0	0	0	0	...	
1	1951	7.45	1	0	0	0	0	0	0	0	...	
2	1952	7.45	1	0	0	0	0	0	0	0	...	
3	1953	7.45	1	0	0	0	0	0	0	0	...	
4	1954	7.45	1	0	0	0	0	0	0	0	...	

5 rows × 248 columns

In [1090]:
```
# Function that will be used to scale each year
def func(x):
    return 1/np.log(x/1940.0)
```

In [1091]:
```
fertScaled["ScaledYear"] = func(fertScaled["Year"])
fertScaled.head()
```

Out[1091]:

	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	...	Venez
0	1950	7.45	1	0	0	0	0	0	0	0	...	
1	1951	7.45	1	0	0	0	0	0	0	0	...	
2	1952	7.45	1	0	0	0	0	0	0	0	...	
3	1953	7.45	1	0	0	0	0	0	0	0	...	
4	1954	7.45	1	0	0	0	0	0	0	0	...	

5 rows × 249 columns

In [1092]:
```
# Source: https://www.geeksforgeeks.org/how-to-split-a-dataset-into-train-and-test-sets-using-python/
X = fertScaled.drop(["Rate"], axis = 1)
X = X.drop(["Year"], axis = 1)
y = fertScaled.iloc[:, 1:2]

trainX, testX, trainY, testY = train_test_split(X, y, test_size=.2)

print('X train shape:',trainX.shape)
print('X test shape:',testX.shape)
print('Y train shape:',trainY.shape)
print('Y test shape:',testY.shape)
```

```
X train shape: (13972, 247)
X test shape: (3494, 247)
Y train shape: (13972, 1)
Y test shape: (3494, 1)
```

In [1093]:
```
scaledMLR = LinearRegression()
scaledMLR.fit(trainX, trainY)
```

Out[1093]:
```
LinearRegression()
```

In [1094]:
```
scaledPredictions = {}

for key, value in class_mapping.items():
    temp = [0] * len(fertCountries)
    temp[value] = 1
    a = []
    a = np.insert(temp, 0, 2122)
    scaledPredictions[key] = scaledMLR.predict([a])

scaledPredictions
```

Out[1094]:

```
{'Afghanistan': array([[-2.8917603e+12]]),
 'Africa': array([[-2.8917603e+12]]),
 'Albania': array([[-2.8917603e+12]]),
 'Algeria': array([[-2.8917603e+12]]),
 'Angola': array([[-2.8917603e+12]]),
 'Antigua and Barbuda': array([[-2.8917603e+12]]),
 'Argentina': array([[-2.8917603e+12]]),
 'Armenia': array([[-2.8917603e+12]]),
 'Aruba': array([[-2.8917603e+12]]),
 'Asia': array([[-2.8917603e+12]]),
 'Asia, Central': array([[-2.8917603e+12]]),
 'Australia': array([[-2.8917603e+12]]),
 'Australia & New Zealand': array([[-2.8917603e+12]]),
 'Austria': array([[-2.8917603e+12]]),
 'Azerbaijan': array([[-2.8917603e+12]]),
 'Bahamas': array([[-2.8917603e+12]]),
 'Bahrain': array([[-2.8917603e+12]]),
 'Bangladesh': array([[-2.8917603e+12]]),
 'Barbados': array([[-2.8917603e+12]]),
 'Belarus': array([[-2.8917603e+12]]),
 'Belgium': array([[-2.8917603e+12]]),
 'Belize': array([[-2.8917603e+12]]),
 'Benin': array([[-2.8917603e+12]]),
 'Bhutan': array([[-2.8917603e+12]]),
 'Bolivia': array([[-2.8917603e+12]]),
 'Bosnia and Herzegovina': array([[-2.8917603e+12]]),
 'Botswana': array([[-2.8917603e+12]]),
 'Brazil': array([[-2.8917603e+12]]),
 'Brunei': array([[-2.8917603e+12]]),
 'Bulgaria': array([[-2.8917603e+12]]),
 'Burkina Faso': array([[-2.8917603e+12]]),
 'Burundi': array([[-2.8917603e+12]]),
 'Cambodia': array([[-2.8917603e+12]]),
 'Cameroon': array([[-2.8917603e+12]]),
 'Canada': array([[-2.8917603e+12]]),
 'Cape Verde': array([[-2.8917603e+12]]),
 'Caribbean': array([[-2.8917603e+12]]),
 'Central African Republic': array([[-2.8917603e+12]]),
 'Central America': array([[-2.8917603e+12]]),
 'Central and Southern Asia': array([[-2.8917603e+12]]),
 'Chad': array([[-2.8917603e+12]]),
 'Channel Islands': array([[-2.8917603e+12]]),
 'Chile': array([[-2.8917603e+12]]),
 'China': array([[-2.8917603e+12]]),
 'Colombia': array([[-2.8917603e+12]]),
 'Comoros': array([[-2.8917603e+12]]),
 'Congo': array([[-2.8917603e+12]]),
 'Costa Rica': array([[-2.8917603e+12]]),
 "Cote d'Ivoire": array([[-2.8917603e+12]]),
 'Croatia': array([[-2.8917603e+12]]),
 'Cuba': array([[-2.8917603e+12]]),
 'Curacao': array([[-2.8917603e+12]]),
 'Cyprus': array([[-2.8917603e+12]]),
 'Czechia': array([[-2.8917603e+12]]),
 'Democratic Republic of Congo': array([[-2.8917603e+12]]),
 'Denmark': array([[-2.8917603e+12]]),
 'Djibouti': array([[-2.8917603e+12]]),
 'Dominican Republic': array([[-2.8917603e+12]]),
 'Eastern Africa': array([[-2.8917603e+12]]),
 'Eastern Asia': array([[-2.8917603e+12]]),
```

```
'Eastern Europe': array([[-2.8917603e+12]]),
'Eastern and South-Eastern Asia': array([[-2.8917603e+12]]),
'Ecuador': array([[-2.8917603e+12]]),
'Egypt': array([[-2.8917603e+12]]),
'El Salvador': array([[-2.8917603e+12]]),
'Equatorial Guinea': array([[-2.8917603e+12]]),
'Eritrea': array([[-2.8917603e+12]]),
'Estonia': array([[-2.8917603e+12]]),
'Eswatini': array([[-2.8917603e+12]]),
'Ethiopia': array([[-2.8917603e+12]]),
'Europe': array([[-2.8917603e+12]]),
'Europe and Northern America': array([[-2.8917603e+12]]),
'Europe, Western': array([[-2.8917603e+12]]),
'Fiji': array([[-2.8917603e+12]]),
'Finland': array([[-2.8917603e+12]]),
'France': array([[-2.8917603e+12]]),
'French Guiana': array([[-2.8917603e+12]]),
'French Polynesia': array([[-2.8917603e+12]]),
'Gabon': array([[-2.8917603e+12]]),
'Gambia': array([[-2.8917603e+12]]),
'Georgia': array([[-2.8917603e+12]]),
'Germany': array([[-2.8917603e+12]]),
'Ghana': array([[-2.8917603e+12]]),
'Greece': array([[-2.8917603e+12]]),
'Grenada': array([[-2.8917603e+12]]),
'Guadeloupe': array([[-2.8917603e+12]]),
'Guam': array([[-2.8917603e+12]]),
'Guatemala': array([[-2.8917603e+12]]),
'Guinea': array([[-2.8917603e+12]]),
'Guinea-Bissau': array([[-2.8917603e+12]]),
'Guyana': array([[-2.8917603e+12]]),
'Haiti': array([[-2.8917603e+12]]),
'High income countries': array([[-2.8917603e+12]]),
'Honduras': array([[-2.8917603e+12]]),
'Hong Kong': array([[-2.8917603e+12]]),
'Hungary': array([[-2.8917603e+12]]),
'Iceland': array([[-2.8917603e+12]]),
'India': array([[-2.8917603e+12]]),
'Indonesia': array([[-2.8917603e+12]]),
'Iran': array([[-2.8917603e+12]]),
'Iraq': array([[-2.8917603e+12]]),
'Ireland': array([[-2.8917603e+12]]),
'Israel': array([[-2.8917603e+12]]),
'Italy': array([[-2.8917603e+12]]),
'Jamaica': array([[-2.8917603e+12]]),
'Japan': array([[-2.8917603e+12]]),
'Jordan': array([[-2.8917603e+12]]),
'Kazakhstan': array([[-2.8917603e+12]]),
'Kenya': array([[-2.8917603e+12]]),
'Kiribati': array([[-2.8917603e+12]]),
'Kuwait': array([[-2.8917603e+12]]),
'Kyrgyzstan': array([[-2.8917603e+12]]),
'Land-locked Developing Countries (LLDC)': array([[-2.8917603e+12]]),
'Laos': array([[-2.8917603e+12]]),
'Latin America and the Caribbean': array([[-2.8917603e+12]]),
'Latvia': array([[-2.8917603e+12]]),
'Least Developed Countries': array([[-2.8917603e+12]]),
'Lebanon': array([[-2.8917603e+12]]),
'Lesotho': array([[-2.8917603e+12]]),
'Less Developed Regions': array([[-2.8917603e+12]]),
```

```
 'Less Developed Regions, excluding China': array([[-2.8917603e+12]]),
 'Less Developed Regions, excluding Least Developed Countries': array([[-2.8917603
e+12]]),
 'Liberia': array([[-2.8917603e+12]]),
 'Libya': array([[-2.8917603e+12]]),
 'Lithuania': array([[-2.8917603e+12]]),
 'Low-income countries': array([[-2.8917603e+12]]),
 'Lower-middle-income countries': array([[-2.8917603e+12]]),
 'Luxembourg': array([[-2.8917603e+12]]),
 'Macao': array([[-2.8917603e+12]]),
 'Madagascar': array([[-2.8917603e+12]]),
 'Malawi': array([[-2.8917603e+12]]),
 'Malaysia': array([[-2.8917603e+12]]),
 'Maldives': array([[-2.8917603e+12]]),
 'Mali': array([[-2.8917603e+12]]),
 'Malta': array([[-2.8917603e+12]]),
 'Martinique': array([[-2.8917603e+12]]),
 'Mauritania': array([[-2.8917603e+12]]),
 'Mauritius': array([[-2.8917603e+12]]),
 'Mayotte': array([[-2.8917603e+12]]),
 'Melanesia': array([[-2.8917603e+12]]),
 'Mexico': array([[-2.8917603e+12]]),
 'Micronesia (country)': array([[-2.8917603e+12]]),
 'Middle Africa': array([[-2.8917603e+12]]),
 'Middle-income countries': array([[-2.8917603e+12]]),
 'Moldova': array([[-2.8917603e+12]]),
 'Mongolia': array([[-2.8917603e+12]]),
 'Montenegro': array([[-2.8917603e+12]]),
 'More Developed Regions': array([[-2.8917603e+12]]),
 'Morocco': array([[-2.8917603e+12]]),
 'Mozambique': array([[-2.8917603e+12]]),
 'Myanmar': array([[-2.8917603e+12]]),
 'Namibia': array([[-2.8917603e+12]]),
 'Nepal': array([[-2.8917603e+12]]),
 'Netherlands': array([[-2.8917603e+12]]),
 'New Caledonia': array([[-2.8917603e+12]]),
 'New Zealand': array([[-2.8917603e+12]]),
 'Nicaragua': array([[-2.8917603e+12]]),
 'Niger': array([[-2.8917603e+12]]),
 'Nigeria': array([[-2.8917603e+12]]),
 'No income group available': array([[-2.8917603e+12]]),
 'North Korea': array([[-2.8917603e+12]]),
 'North Macedonia': array([[-2.8917603e+12]]),
 'Northern Africa': array([[-2.8917603e+12]]),
 'Northern Africa and Western Asia': array([[-2.8917603e+12]]),
 'Northern America': array([[-2.8917603e+12]]),
 'Northern Europe': array([[-2.8917603e+12]]),
 'Norway': array([[-2.8917603e+12]]),
 'Oceania': array([[-2.8917603e+12]]),
 'Oceania (excluding Australia and New Zealand)': array([[-2.8917603e+12]]),
 'Oman': array([[-2.8917603e+12]]),
 'Pakistan': array([[-2.8917603e+12]]),
 'Palestine': array([[-2.8917603e+12]]),
 'Panama': array([[-2.8917603e+12]]),
 'Papua New Guinea': array([[-2.8917603e+12]]),
 'Paraguay': array([[-2.8917603e+12]]),
 'Peru': array([[-2.8917603e+12]]),
 'Philippines': array([[-2.8917603e+12]]),
 'Poland': array([[-2.8917603e+12]]),
 'Portugal': array([[-2.8917603e+12]]),
```

```
'Puerto Rico': array([[-2.8917603e+12]]),
'Qatar': array([[-2.8917603e+12]]),
'Reunion': array([[-2.8917603e+12]]),
'Romania': array([[-2.8917603e+12]]),
'Russia': array([[-2.8917603e+12]]),
'Rwanda': array([[-2.8917603e+12]]),
'Saint Lucia': array([[-2.8917603e+12]]),
'Saint Vincent and the Grenadines': array([[-2.8917603e+12]]),
'Samoa': array([[-2.8917603e+12]]),
'Sao Tome and Principe': array([[-2.8917603e+12]]),
'Saudi Arabia': array([[-2.8917603e+12]]),
'Senegal': array([[-2.8917603e+12]]),
'Serbia': array([[-2.8917603e+12]]),
'Seychelles': array([[-2.8917603e+12]]),
'Sierra Leone': array([[-2.8917603e+12]]),
'Singapore': array([[-2.8917603e+12]]),
'Slovakia': array([[-2.8917603e+12]]),
'Slovenia': array([[-2.8917603e+12]]),
'Small Island Developing States (SIDS)': array([[-2.8917603e+12]]),
'Solomon Islands': array([[-2.8917603e+12]]),
'Somalia': array([[-2.8917603e+12]]),
'South Africa': array([[-2.8917603e+12]]),
'South America': array([[-2.8917603e+12]]),
'South Eastern Asia': array([[-2.8917603e+12]]),
'South Korea': array([[-2.8917603e+12]]),
'South Sudan': array([[-2.8917603e+12]]),
'Southern Africa': array([[-2.8917603e+12]]),
'Southern Asia': array([[-2.8917603e+12]]),
'Southern Europe': array([[-2.8917603e+12]]),
'Spain': array([[-2.8917603e+12]]),
'Sri Lanka': array([[-2.8917603e+12]]),
'Sub-Saharan Africa': array([[-2.8917603e+12]]),
'Sudan': array([[-2.8917603e+12]]),
'Suriname': array([[-2.8917603e+12]]),
'Sweden': array([[-2.8917603e+12]]),
'Switzerland': array([[-2.8917603e+12]]),
'Syria': array([[-2.8917603e+12]]),
'Taiwan': array([[-2.8917603e+12]]),
'Tajikistan': array([[-2.8917603e+12]]),
'Tanzania': array([[-2.8917603e+12]]),
'Thailand': array([[-2.8917603e+12]]),
'Timor': array([[-2.8917603e+12]]),
'Togo': array([[-2.8917603e+12]]),
'Tonga': array([[-2.8917603e+12]]),
'Trinidad and Tobago': array([[-2.8917603e+12]]),
'Tunisia': array([[-2.8917603e+12]]),
'Turkey': array([[-2.8917603e+12]]),
'Turkmenistan': array([[-2.8917603e+12]]),
'Uganda': array([[-2.8917603e+12]]),
'Ukraine': array([[-2.8917603e+12]]),
'United Arab Emirates': array([[-2.8917603e+12]]),
'United Kingdom': array([[-2.8917603e+12]]),
'United States': array([[-2.8917603e+12]]),
'United States Virgin Islands': array([[-2.8917603e+12]]),
'Upper-middle-income countries': array([[-2.8917603e+12]]),
'Uruguay': array([[-2.8917603e+12]]),
'Uzbekistan': array([[-2.8917603e+12]]),
'Vanuatu': array([[-2.8917603e+12]]),
'Venezuela': array([[-2.8917603e+12]]),
'Vietnam': array([[-2.8917603e+12]]),
```

```
'Western Africa': array([[-2.8917603e+12]]),
'Western Asia': array([[-2.8917603e+12]]),
'Western Sahara': array([[-2.8917603e+12]]),
'World': array([[-2.8917603e+12]]),
'Yemen': array([[-2.8917603e+12]]),
'Zambia': array([[-2.8917603e+12]]),
'Zimbabwe': array([[-2.89039755e+12]])}
```

This isn't much better with it just predicting every country to have (pretty much) the same fertility rate in 2122 which isn't right and they are negative

To get a more accurate prediction for each country, let's try this log scaling technique, but just build a seperate model for each country and aggregate their results. To start, let's try this approach just on the United States' fertility rate and see how the model looks compared to the actual data...

In [1096]:
```
usFert["ScaledYear"] = func(usFert["Year"])
usFert = usFert.dropna()
usFert.head()
```

Out[1096]:

	Year	Rate	Afghanistan	Africa	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	...
16401	1950.0	3.052	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
16402	1951.0	3.130	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
16403	1952.0	3.272	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
16404	1953.0	3.389	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
16405	1954.0	3.479	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...

5 rows × 249 columns

In [1097]:
```
xUS = usFert["ScaledYear"]
yUS = usFert["Rate"]
```

In [1098]:
```
xUS = xUS.ravel()
yUS = yUS.ravel()
xUS = xUS.reshape(-1, 1)
```

In [1099]:
```
usMLR = LinearRegression()
usMLR.fit(xUS, yUS)
```

Out[1099]:
```
LinearRegression()
```

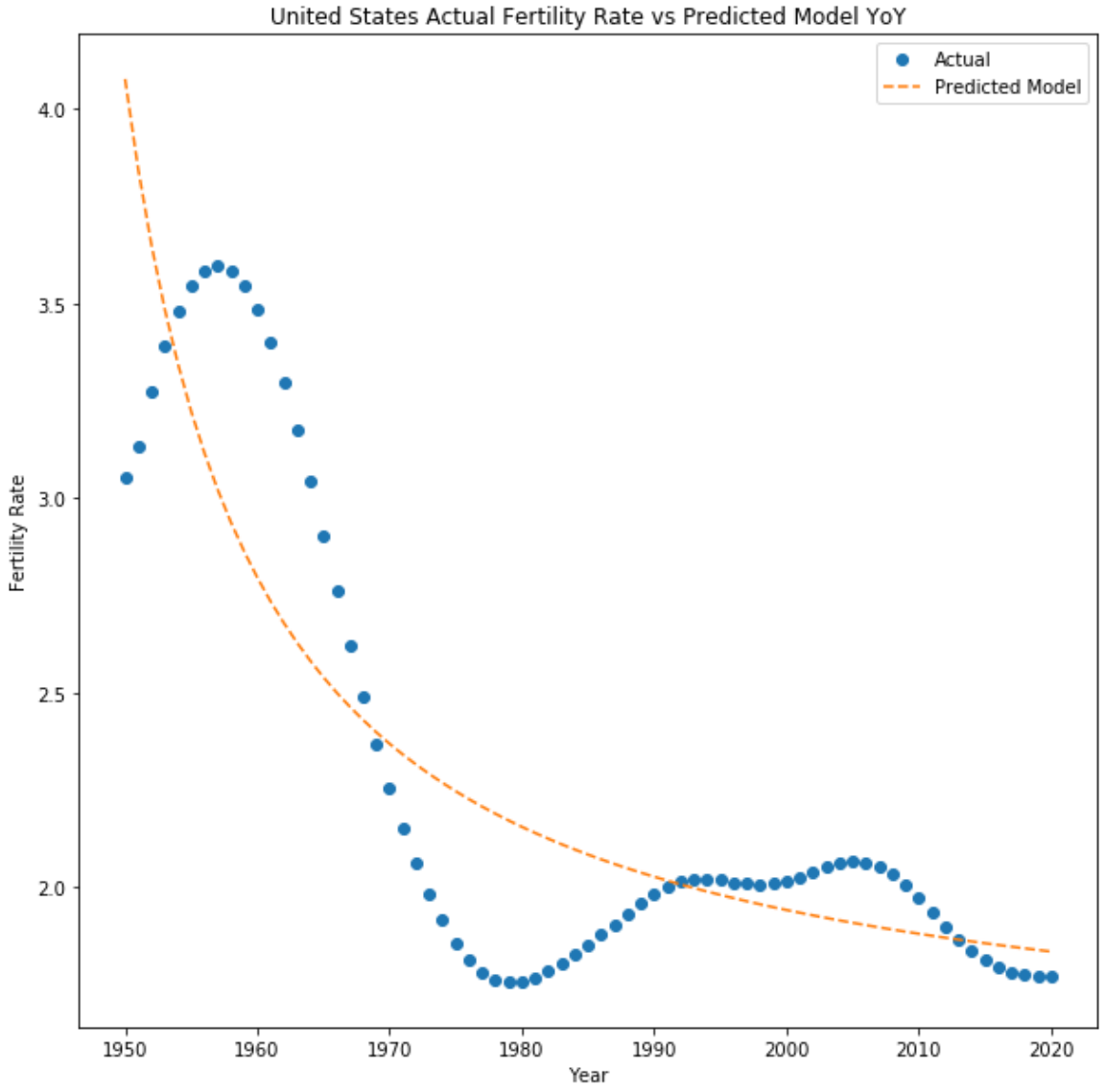
In [1100]:
```
years = list(range(1950, 2123))
```

In [1101]:
```
logYears = []
for i in range(len(years)):
    yearVal = func(years[i])
    logYears.append(yearVal)
```

In [1102]:
```
usPred = []
for year in logYears:
    temp = [year]
    predVal = usMLR.predict([temp])
    usPred.append(predVal)
```

In [1103]:
```
plt.figure(figsize=(10, 10))
plt.plot(usFert["Year"], usFert["Rate"],  linestyle='', marker='o', label='Actual')
plt.plot(usFert["Year"], yearPred[:len(usFert)],  linestyle='--', label='Predicted
 Model')
plt.ylabel("Fertility Rate")
plt.xlabel("Year")
plt.title("United States Actual Fertility Rate vs Predicted Model YoY")
plt.legend()
```

Out[1103]: <matplotlib.legend.Legend at 0x1c3a49a2748>



This model seems to be producing a much more accurate representation of the fertility rate and that it is trending in the right direction moving forward. Let's see what the predicted values of all the countries for 2122 are following this approach

In [1105]:
```
logPredictedCountriesFertilityRate = {}
toPredict = logYears[-1]
toPredict = toPredict.reshape(-1, 1)
for country in fertCountries:
    subsetData = fertility.where(fertility["Entity"] == country)
    subsetData = subsetData.dropna()
    subsetData["ScaledYear"] = func(subsetData["Year"])
    xSub = subsetData["ScaledYear"]
    ySub = subsetData["Rate"]
    xSub = xSub.ravel()
    ySub = ySub.ravel()
    xSub = xSub.reshape(-1, 1)
    subMLR = LinearRegression()
    subMLR.fit(xSub, ySub)
    predVal = subMLR.predict(toPredict)
    logPredictedCountriesFertilityRate[country] = predVal

logPredictedCountriesFertilityRate
```

Out[1105]:

```
{'Afghanistan': array([6.56259245]),
 'Africa': array([5.26141546]),
 'Albania': array([1.80824968]),
 'Algeria': array([3.51430728]),
 'Angola': array([6.8447852]),
 'Antigua and Barbuda': array([1.72832302]),
 'Argentina': array([2.61104168]),
 'Armenia': array([1.60719054]),
 'Aruba': array([1.3333467]),
 'Asia': array([2.38719256]),
 'Asia, Central': array([3.04513053]),
 'Australia': array([1.66855584]),
 'Australia & New Zealand': array([1.68468684]),
 'Austria': array([1.41602657]),
 'Azerbaijan': array([2.16277095]),
 'Bahamas': array([2.00068323]),
 'Bahrain': array([2.58672108]),
 'Bangladesh': array([3.36225414]),
 'Barbados': array([1.33295822]),
 'Belarus': array([1.52657093]),
 'Belgium': array([1.58549344]),
 'Belize': array([3.39302024]),
 'Benin': array([6.08931928]),
 'Bhutan': array([3.588953]),
 'Bolivia': array([3.74024835]),
 'Bosnia and Herzegovina': array([1.02970034]),
 'Botswana': array([3.63461661]),
 'Brazil': array([1.99467731]),
 'Brunei': array([2.0721582]),
 'Bulgaria': array([1.46063069]),
 'Burkina Faso': array([6.46072255]),
 'Burundi': array([6.72117469]),
 'Cambodia': array([3.74314598]),
 'Cameroon': array([5.75749229]),
 'Canada': array([1.22740828]),
 'Cape Verde': array([3.72738829]),
 'Caribbean': array([2.33675249]),
 'Central African Republic': array([5.47890774]),
 'Central America': array([2.8146029]),
 'Central and Southern Asia': array([3.25990978]),
 'Chad': array([6.97646292]),
 'Channel Islands': array([1.45908041]),
 'Chile': array([1.76827355]),
 'China': array([1.4343606]),
 'Colombia': array([2.07713821]),
 'Comoros': array([5.60643395]),
 'Congo': array([5.01655451]),
 'Costa Rica': array([1.98401405]),
 "Cote d'Ivoire": array([5.82318283]),
 'Croatia': array([1.38211457]),
 'Cuba': array([1.39723857]),
 'Curacao': array([1.65795164]),
 'Cyprus': array([1.52006893]),
 'Czechia': array([1.40912911]),
 'Democratic Republic of Congo': array([6.62001169]),
 'Denmark': array([1.56715379]),
 'Djibouti': array([4.47393791]),
 'Dominican Republic': array([2.41455712]),
 'Eastern Africa': array([5.72065722]),
 'Eastern Asia': array([1.42297508]),
```

```
'Eastern Europe': array([1.42133185]),
'Eastern and South-Eastern Asia': array([1.71802372]),
'Ecuador': array([2.8445397]),
'Egypt': array([3.58260314]),
'El Salvador': array([2.86953695]),
'Equatorial Guinea': array([5.5100367]),
'Eritrea': array([5.21505076]),
'Estonia': array([1.65791086]),
'Eswatini': array([4.11995791]),
'Ethiopia': array([5.99568521]),
'Europe': array([1.47316164]),
'Europe and Northern America': array([1.53315668]),
'Europe, Western': array([1.49494531]),
'Fiji': array([2.56233543]),
'Finland': array([1.45369129]),
'France': array([1.72791458]),
'French Guiana': array([3.41757676]),
'French Polynesia': array([2.29265656]),
'Gabon': array([4.92853179]),
'Gambia': array([5.75954463]),
'Georgia': array([1.89714379]),
'Germany': array([1.32401861]),
'Ghana': array([4.88720823]),
'Greece': array([1.41018591]),
'Grenada': array([2.52597176]),
'Guadeloupe': array([1.89363617]),
'Guam': array([2.34992844]),
'Guatemala': array([4.17197788]),
'Guinea': array([5.88253516]),
'Guinea-Bissau': array([5.70074666]),
'Guyana': array([2.58939594]),
'Haiti': array([4.15410705]),
'High income countries': array([1.60329883]),
'Honduras': array([3.80345208]),
'Hong Kong': array([0.8301302]),
'Hungary': array([1.37340587]),
'Iceland': array([1.78753516]),
'India': array([3.06729555]),
'Indonesia': array([2.64225393]),
'Iran': array([2.86086271]),
'Iraq': array([4.82699253]),
'Ireland': array([2.04704503]),
'Israel': array([2.84819903]),
'Italy': array([1.29763616]),
'Jamaica': array([2.61533345]),
'Japan': array([1.22408831]),
'Jordan': array([4.36002453]),
'Kazakhstan': array([2.33279919]),
'Kenya': array([5.11165432]),
'Kiribati': array([3.94586347]),
'Kuwait': array([2.56505528]),
'Kyrgyzstan': array([3.32166435]),
'Land-locked Developing Countries (LLDC)': array([4.9352959]),
'Laos': array([4.33283243]),
'Latin America and the Caribbean': array([2.42373228]),
'Latvia': array([1.57751486]),
'Least Developed Countries': array([5.1137948]),
'Lebanon': array([2.29687568]),
'Lesotho': array([3.87002312]),
'Less Developed Regions': array([2.85437291]),
```

```
 'Less Developed Regions, excluding China': array([3.42365039]),
 'Less Developed Regions, excluding Least Developed Countries': array([2.5361932
6]),
 'Liberia': array([5.71428228]),
 'Libya': array([3.5782292]),
 'Lithuania': array([1.50030983]),
 'Low-income countries': array([5.61920183]),
 'Lower-middle-income countries': array([3.40784717]),
 'Luxembourg': array([1.51177145]),
 'Macao': array([0.73994585]),
 'Madagascar': array([5.21615806]),
 'Malawi': array([6.0023085]),
 'Malaysia': array([2.36567931]),
 'Maldives': array([3.77777025]),
 'Mali': array([6.75202149]),
 'Malta': array([1.21155677]),
 'Martinique': array([1.50701107]),
 'Mauritania': array([5.39502533]),
 'Mauritius': array([1.33284903]),
 'Mayotte': array([4.98846416]),
 'Melanesia': array([4.06205076]),
 'Mexico': array([2.62745527]),
 'Micronesia (country)': array([4.03440279]),
 'Middle Africa': array([6.41879502]),
 'Middle-income countries': array([2.58034816]),
 'Moldova': array([1.45983981]),
 'Mongolia': array([3.42066335]),
 'Montenegro': array([1.5037]),
 'More Developed Regions': array([1.50903987]),
 'Morocco': array([3.07344987]),
 'Mozambique': array([5.73532188]),
 'Myanmar': array([2.77926509]),
 'Namibia': array([4.32126668]),
 'Nepal': array([3.61475605]),
 'Netherlands': array([1.42157623]),
 'New Caledonia': array([2.3571097]),
 'New Zealand': array([1.76681213]),
 'Nicaragua': array([3.27811545]),
 'Niger': array([7.61324951]),
 'Nigeria': array([6.11236489]),
 'No income group available': array([2.24028761]),
 'North Korea': array([2.11218693]),
 'North Macedonia': array([1.30724993]),
 'Northern Africa': array([3.68046457]),
 'Northern Africa and Western Asia': array([3.48437601]),
 'Northern America': array([1.61545888]),
 'Northern Europe': array([1.71110296]),
 'Norway': array([1.71450973]),
 'Oceania': array([2.30156814]),
 'Oceania (excluding Australia and New Zealand)': array([3.94989392]),
 'Oman': array([4.60788264]),
 'Pakistan': array([4.85388952]),
 'Palestine': array([5.31909774]),
 'Panama': array([2.55444962]),
 'Papua New Guinea': array([4.30392403]),
 'Paraguay': array([3.26358655]),
 'Peru': array([2.80385117]),
 'Philippines': array([3.28629193]),
 'Poland': array([1.31386743]),
 'Portugal': array([1.37059925]),
```

```
'Puerto Rico': array([1.39107843]),
'Qatar': array([2.93813842]),
'Reunion': array([1.7680624]),
'Romania': array([1.52397851]),
'Russia': array([1.44458293]),
'Rwanda': array([5.71696037]),
'Saint Lucia': array([2.22092811]),
'Saint Vincent and the Grenadines': array([1.96781711]),
'Samoa': array([4.4233818]),
'Sao Tome and Principe': array([5.20243673]),
'Saudi Arabia': array([4.13262205]),
'Senegal': array([5.67624269]),
'Serbia': array([1.58690542]),
'Seychelles': array([2.47287286]),
'Sierra Leone': array([5.91412074]),
'Singapore': array([0.54869133]),
'Slovakia': array([1.36451322]),
'Slovenia': array([1.37099653]),
'Small Island Developing States (SIDS)': array([2.59016979]),
'Solomon Islands': array([5.0958912]),
'Somalia': array([7.02501948]),
'South Africa': array([2.92959312]),
'South America': array([2.28033761]),
'South Eastern Asia': array([2.60849421]),
'South Korea': array([0.94305707]),
'South Sudan': array([5.95328402]),
'Southern Africa': array([3.0520728]),
'Southern Asia': array([3.27030342]),
'Southern Europe': array([1.38096919]),
'Spain': array([1.3956193]),
'Sri Lanka': array([1.96699572]),
'Sub-Saharan Africa': array([5.70679185]),
'Sudan': array([5.42977627]),
'Suriname': array([2.51698957]),
'Sweden': array([1.7244462]),
'Switzerland': array([1.3923904]),
'Syria': array([4.21920104]),
'Taiwan': array([0.6870098]),
'Tajikistan': array([4.42335649]),
'Tanzania': array([5.67091299]),
'Thailand': array([1.48740711]),
'Timor': array([4.96851168]),
'Togo': array([5.57404213]),
'Tonga': array([3.89413879]),
'Trinidad and Tobago': array([1.63994351]),
'Tunisia': array([2.54269339]),
'Turkey': array([2.26355376]),
'Turkmenistan': array([3.44034767]),
'Uganda': array([6.45056552]),
'Ukraine': array([1.33005185]),
'United Arab Emirates': array([2.65699043]),
'United Kingdom': array([1.73644611]),
'United States': array([1.65538765]),
'United States Virgin Islands': array([2.20115616]),
'Upper-middle-income countries': array([1.81805731]),
'Uruguay': array([2.25377162]),
'Uzbekistan': array([3.21442056]),
'Vanuatu': array([4.09112698]),
'Venezuela': array([2.51476801]),
'Vietnam': array([2.6605214]),
```

```
'Western Africa': array([6.04327416]),
'Western Asia': array([3.31575096]),
'Western Sahara': array([3.29042784]),
'World': array([2.65208038]),
'Yemen': array([6.31795708]),
'Zambia': array([5.91565124]),
'Zimbabwe': array([4.36001168])}
```

Nice! All positive and differing values that seem to be much more realistic...

Life Expectancy

In [1106]:
```
life = pd.read_csv("life-expectancy.csv") # Source: https://ourworldindata.org/life-expectancy
```

In [1107]:
```
life.head()
```

Out[1107]:

	Entity	Year	Life expectancy
0	Afghanistan	1950	27.638
1	Afghanistan	1951	27.878
2	Afghanistan	1952	28.361
3	Afghanistan	1953	28.852
4	Afghanistan	1954	29.350

Let's use the same visualization used for looking at each country's fertility rate for looking at a selected country's life expectancy and see if there are any trends in the visualization

In [1108]:
```
import plotly.express as px

# Neat little visualization of life expectancy for a selected country
# Source: https://plotly.com/python/sliders/
# Source: https://plotly.com/python/line-and-scatter/
fig = px.scatter(life, x="Year", y="Life expectancy", animation_frame="Entity", ani
mation_group="Year",
                 range_x=[1950,2020], range_y=[10,100],color='Life expectancy')

fig["layout"].pop("updatemenus") # optional, drop animation buttons
fig.show()
```

Each country's life expectancy appears to be following a slight gradual linear increase in life expectancy in the visualization above when toggling across various countries. Now let's build a linear regression model for each country to predict their life expectancy in 2122

First, will take the approach of just building the model for the United States' life expectancy and graphing the model's predicted values vs. their actual values to ensure this is producing reasonable results

In [1109]:
```
usLife = life.where(life['Entity'] == "United States")
usLife = usLife.dropna()
usLife.head()
```

Out[1109]:

	Entity	Year	Life expectancy
17980	United States	1880.0	39.410000
17981	United States	1890.0	45.209999
17982	United States	1901.0	49.299999
17983	United States	1902.0	50.500000
17984	United States	1903.0	50.599998

In [1110]:
```
# Let's just look at 1950 on
usLife = usLife.where(usLife['Year'] >= 1950)
usLife = usLife.dropna()
usLife.head()
```

Out[1110]:

	Entity	Year	Life expectancy
18031	United States	1950.0	68.202
18032	United States	1951.0	68.338
18033	United States	1952.0	68.601
18034	United States	1953.0	68.842
18035	United States	1954.0	69.063

In [1111]:
```
xLifeUS = usLife['Year']
yLifeUS = usLife['Life expectancy']
```

In [1112]:
```
xLifeUS = xLifeUS.ravel()
yLifeUS = yLifeUS.ravel()
xLifeUS = xLifeUS.reshape(-1, 1)
```

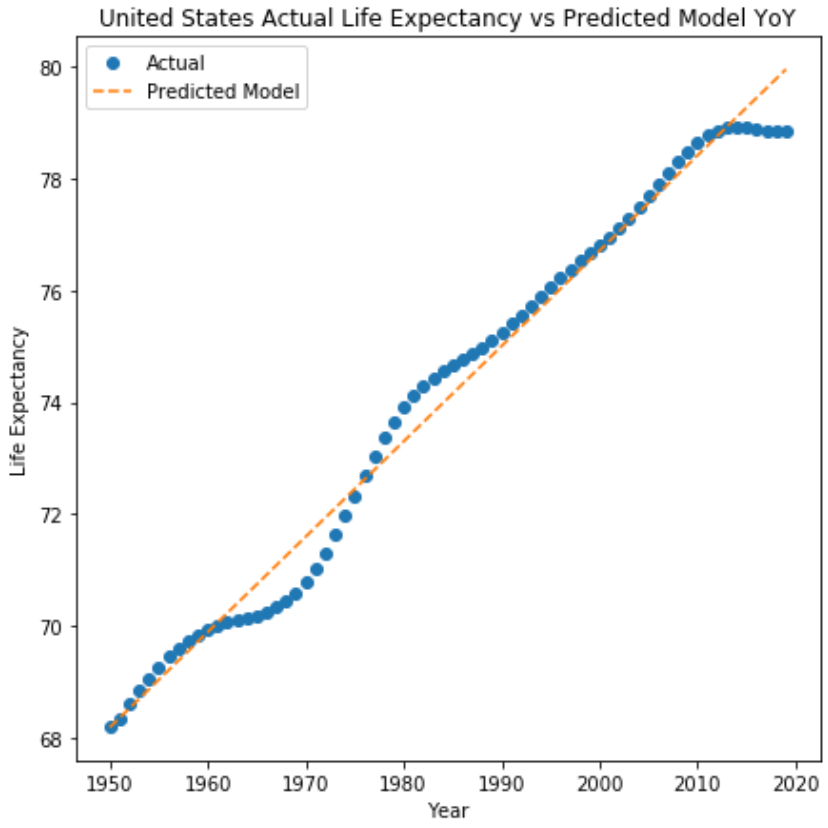
In [1113]:
```
usLifeMLR = LinearRegression()
usLifeMLR.fit(xLifeUS, yLifeUS)
```

Out[1113]:
```
LinearRegression()
```

In [1114]:
```
usLifePred = []
years = list(range(1950, 2123))
for year in years:
    temp = [year]
    predVal = usLifeMLR.predict([temp])
    usLifePred.append(predVal)
```

In [1115]:
```
plt.figure(figsize=(7, 7))
plt.plot(usLife["Year"], usLife["Life expectancy"],  linestyle='', marker='o', label='Actual')
plt.plot(usLife["Year"], usLifePred[:len(usLife)],  linestyle='--', label='Predicted Model')
plt.ylabel("Life Expectancy")
plt.xlabel("Year")
plt.title("United States Actual Life Expectancy vs Predicted Model YoY")
plt.legend()
```

Out[1115]: <matplotlib.legend.Legend at 0x1c3915383c8>



The model looks like a pretty good predictor of the United States' life expectancy so we can apply this to the other countries to get each countries' life expectancy for 2122 too

In [1116]:
```
# Just use data from 1950 on
life = life.where(life['Year'] >= 1950)
life = life.dropna()
life.head()
```

Out[1116]:

	Entity	Year	Life expectancy
0	Afghanistan	1950.0	27.638
1	Afghanistan	1951.0	27.878
2	Afghanistan	1952.0	28.361
3	Afghanistan	1953.0	28.852
4	Afghanistan	1954.0	29.350

In [1117]:
```
predictedCountriesLifeExpectancy = {}
yearToPredict = np.array(2122)
yearToPredict = yearToPredict.reshape(-1, 1)
lifeCountries = life['Entity'].unique()
for country in lifeCountries:
    subsetData = life.where(life["Entity"] == country)
    subsetData = subsetData.dropna()
    xSub = subsetData["Year"]
    ySub = subsetData["Life expectancy"]
    xSub = xSub.ravel()
    ySub = ySub.ravel()
    xSub = xSub.reshape(-1, 1)
    subMLR = LinearRegression()
    subMLR.fit(xSub, ySub)
    predVal = subMLR.predict(yearToPredict)
    predictedCountriesLifeExpectancy[country] = predVal

predictedCountriesLifeExpectancy
```

Out[1117]:

```
{'Afghanistan': array([124.96095183]),
 'Africa': array([96.99803714]),
 'Albania': array([111.93608237]),
 'Algeria': array([139.74015335]),
 'American Samoa': array([89.30258802]),
 'Americas': array([105.65551568]),
 'Andorra': array([111.67327474]),
 'Angola': array([90.72876244]),
 'Anguilla': array([126.51005492]),
 'Antigua and Barbuda': array([108.74602372]),
 'Argentina': array([99.5792971]),
 'Armenia': array([89.07052041]),
 'Aruba': array([99.49818226]),
 'Asia': array([126.39526246]),
 'Australia': array([107.19301484]),
 'Austria': array([106.82091215]),
 'Azerbaijan': array([91.46301529]),
 'Bahamas': array([91.32643597]),
 'Bahrain': array([133.56017601]),
 'Bangladesh': array([125.81712559]),
 'Barbados': array([115.58636585]),
 'Belarus': array([82.60048669]),
 'Belgium': array([103.45155678]),
 'Belize': array([100.40714385]),
 'Benin': array([109.21842051]),
 'Bermuda': array([108.53390463]),
 'Bhutan': array([139.52552905]),
 'Bolivia': array([124.45139193]),
 'Bonaire Sint Eustatius and Saba': array([102.86452375]),
 'Bosnia and Herzegovina': array([113.74538168]),
 'Botswana': array([86.15238623]),
 'Brazil': array([116.7198636]),
 'British Virgin Islands': array([113.76489768]),
 'Brunei': array([119.02131679]),
 'Bulgaria': array([88.19717617]),
 'Burkina Faso': array([106.79604637]),
 'Burundi': array([88.69013572]),
 'Cambodia': array([122.45232422]),
 'Cameroon': array([86.12545595]),
 'Canada': array([104.60040899]),
 'Cape Verde': array([120.28602308]),
 'Cayman Islands': array([130.89115168]),
 'Central African Republic': array([75.87402656]),
 'Chad': array([80.02322829]),
 'Channel Islands': array([104.936509]),
 'Chile': array([127.17246941]),
 'China': array([136.39481422]),
 'Colombia': array([118.32996146]),
 'Comoros': array([109.04322129]),
 'Congo': array([90.82305025]),
 'Cook Islands': array([120.6077111]),
 'Costa Rica': array([122.5548046]),
 "Cote d'Ivoire": array([94.9146935]),
 'Croatia': array([104.68223335]),
 'Cuba': array([111.13015203]),
 'Curacao': array([105.35173126]),
 'Cyprus': array([102.79865943]),
 'Czechia': array([95.48361687]),
 'Democratic Republic of Congo': array([89.31452424]),
 'Denmark': array([94.22978004]),
```

```
'Djibouti': array([100.02451656]),
'Dominica': array([110.75443679]),
'Dominican Republic': array([119.80476949]),
'Ecuador': array([126.46375941]),
'Egypt': array([125.47140353]),
'El Salvador': array([121.45494107]),
'Equatorial Guinea': array([99.91531644]),
'Eritrea': array([112.90697594]),
'Estonia': array([92.80257596]),
'Eswatini': array([71.2609059]),
'Ethiopia': array([108.5815568]),
'Europe': array([96.26465345]),
'Faeroe Islands': array([99.53679584]),
'Falkland Islands': array([101.2151968]),
'Fiji': array([83.81235493]),
'Finland': array([106.32887943]),
'France': array([107.36288309]),
'French Guiana': array([124.36006199]),
'French Polynesia': array([122.78083342]),
'Gabon': array([112.91695245]),
'Gambia': array([120.0961445]),
'Georgia': array([90.67230223]),
'Germany': array([104.00906608]),
'Ghana': array([96.14983688]),
'Gibraltar': array([94.45613687]),
'Greece': array([109.04099022]),
'Greenland': array([98.25583097]),
'Grenada': array([98.03696691]),
'Guadeloupe': array([129.01930998]),
'Guam': array([116.34712715]),
'Guatemala': array([126.87885177]),
'Guinea': array([106.61760934]),
'Guinea-Bissau': array([92.11977188]),
'Guyana': array([84.45946115]),
'Haiti': array([104.32974991]),
'Honduras': array([135.66023555]),
'Hong Kong': array([119.17966185]),
'Hungary': array([90.26619428]),
'Iceland': array([100.63863256]),
'India': array([123.23532081]),
'Indonesia': array([120.88067014]),
'Iran': array([139.30749086]),
'Iraq': array([122.09917204]),
'Ireland': array([103.98061125]),
'Isle of Man': array([100.54779678]),
'Israel': array([105.48460378]),
'Italy': array([111.05202061]),
'Jamaica': array([99.35150381]),
'Japan': array([119.47995197]),
'Jordan': array([122.60596296]),
'Kazakhstan': array([90.84035738]),
'Kenya': array([89.47826444]),
'Kiribati': array([108.29902063]),
'Kuwait': array([110.94590519]),
'Kyrgyzstan': array([98.80757997]),
'Laos': array([109.34330246]),
'Latin America and the Caribbean': array([116.26189873]),
'Latvia': array([84.78483961]),
'Lebanon': array([109.71937965]),
'Lesotho': array([54.10012996]),
```

```
'Liberia': array([110.2377139]),
'Libya': array([138.43344024]),
'Liechtenstein': array([106.88641034]),
'Lithuania': array([88.18476235]),
'Luxembourg': array([107.61937974]),
'Macao': array([123.76466909]),
'Madagascar': array([114.19211978]),
'Malawi': array([94.88622108]),
'Malaysia': array([109.60862925]),
'Maldives': array([159.59459592]),
'Mali': array([112.47527509]),
'Malta': array([108.88360241]),
'Marshall Islands': array([105.97203791]),
'Martinique': array([127.68436803]),
'Mauritania': array([106.18198389]),
'Mauritius': array([111.7121237]),
'Mayotte': array([134.0278748]),
'Mexico': array([119.55047947]),
'Micronesia (country)': array([93.89835927]),
'Moldova': array([87.67536537]),
'Monaco': array([119.55447135]),
'Mongolia': array([109.08280789]),
'Montenegro': array([101.9690182]),
'Montserrat': array([101.88874326]),
'Morocco': array([130.28706402]),
'Mozambique': array([87.27999073]),
'Myanmar': array([113.47189677]),
'Namibia': array([86.05491065]),
'Nauru': array([85.22825312]),
'Nepal': array([135.68630972]),
'Netherlands': array([97.22892685]),
'New Caledonia': array([121.53078044]),
'New Zealand': array([102.3976784]),
'Nicaragua': array([130.2955391]),
'Niger': array([103.48227824]),
'Nigeria': array([81.71103193]),
'Niue': array([95.71834099]),
'North Korea': array([120.15514627]),
'North Macedonia': array([108.63637348]),
'Northern America': array([98.29479279]),
'Northern Mariana Islands': array([112.62086855]),
'Norway': array([97.08897662]),
'Oceania': array([109.90231532]),
'Oman': array([152.12774466]),
'Pakistan': array([113.63631728]),
'Palau': array([100.32125695]),
'Palestine': array([125.66433126]),
'Panama': array([114.40905982]),
'Papua New Guinea': array([115.36026066]),
'Paraguay': array([91.87773556]),
'Peru': array([133.90898079]),
'Philippines': array([92.82994454]),
'Poland': array([99.19539811]),
'Portugal': array([117.10518648]),
'Puerto Rico': array([102.34449186]),
'Qatar': array([121.75020759]),
'Reunion': array([133.88671621]),
'Romania': array([93.55720852]),
'Russia': array([79.23737944]),
'Rwanda': array([91.95233019]),
```

```
'Saint Barthlemy': array([129.01330842]),
'Saint Helena': array([117.72162523]),
'Saint Kitts and Nevis': array([112.11569399]),
'Saint Lucia': array([118.29575645]),
'Saint Martin (French part)': array([128.41167212]),
'Saint Pierre and Miquelon': array([113.47320822]),
'Saint Vincent and the Grenadines': array([103.78486302]),
'Samoa': array([104.4351082]),
'San Marino': array([117.1092446]),
'Sao Tome and Principe': array([102.21410629]),
'Saudi Arabia': array([139.23457008]),
'Senegal': array([123.50786174]),
'Serbia': array([100.26570116]),
'Seychelles': array([93.34224057]),
'Sierra Leone': array([83.27943844]),
'Singapore': array([120.4004882]),
'Sint Maarten (Dutch part)': array([106.91269619]),
'Slovakia': array([91.7437588]),
'Slovenia': array([103.68274552]),
'Solomon Islands': array([122.38849406]),
'Somalia': array([91.65112725]),
'South Africa': array([86.51522754]),
'South Korea': array([144.89264946]),
'South Sudan': array([104.70294125]),
'Spain': array([112.51491771]),
'Sri Lanka': array([111.42774144]),
'Sudan': array([94.23285692]),
'Suriname': array([95.33991028]),
'Sweden': array([99.98274737]),
'Switzerland': array([106.76771436]),
'Syria': array([120.30795346]),
'Taiwan': array([114.18207978]),
'Tajikistan': array([104.51466008]),
'Tanzania': array([90.46947882]),
'Thailand': array([119.56929772]),
'Timor': array([136.78382336]),
'Togo': array([98.27038583]),
'Tokelau': array([117.91107254]),
'Tonga': array([94.654367]),
'Trinidad and Tobago': array([95.20886929]),
'Tunisia': array([149.53222175]),
'Turkey': array([139.34653379]),
'Turkmenistan': array([93.57328253]),
'Turks and Caicos Islands': array([121.31699687]),
'Tuvalu': array([100.79875664]),
'Uganda': array([79.62913073]),
'Ukraine': array([76.33598282]),
'United Arab Emirates': array([135.11954601]),
'United Kingdom': array([100.69296149]),
'United States': array([97.53223793]),
'United States Virgin Islands': array([107.94726579]),
'Uruguay': array([96.92948503]),
'Uzbekistan': array([94.3551614]),
'Vanuatu': array([115.89603223]),
'Vatican': array([114.91682742]),
'Venezuela': array([103.48309291]),
'Vietnam': array([114.09750123]),
'Wallis and Futuna': array([129.04525543]),
'Western Sahara': array([132.15507887]),
'World': array([113.18247583]),
```

```
'Yemen': array([141.31408743]),
'Zambia': array([71.87706418]),
'Zimbabwe': array([51.61246304])}
```

Some of the life expectancies are higher than I would've thought, but it looks like they are all pretty reasonable

Population Density

In [1118]:
```
density = pd.read_csv("population-density.csv") # Source: https://ourworldindata.org/grapher/population-density-3
density.head()
```

Out[1118]:

	Entity	Code	Year	Density
0	Afghanistan	AFG	-10000	0.023107
1	Afghanistan	AFG	-5000	0.117594
2	Afghanistan	AFG	0	3.135838
3	Afghanistan	AFG	500	3.919798
4	Afghanistan	AFG	1000	3.527818

In [1119]:
```
density = density.where(density['Year'] >= 1950)
density = density.dropna()
density.head()
```

Out[1119]:

	Entity	Code	Year	Density
11	Afghanistan	AFG	1950.0	12.780822
12	Afghanistan	AFG	1955.0	13.805061
13	Afghanistan	AFG	1960.0	15.077664
14	Afghanistan	AFG	1965.0	16.638278
15	Afghanistan	AFG	1970.0	18.563737

In [1124]:
```
import plotly.express as px

# Neat little visualization of population density for a selected country
# Source: https://plotly.com/python/sliders/
# Source: https://plotly.com/python/line-and-scatter/
fig = px.scatter(density, x="Year", y="Density", animation_frame="Entity", animatio
n_group="Year",
                 range_x=[1950,2020], range_y=[0,400],color='Density')

fig["layout"].pop("updatemenus") # optional, drop animation buttons
fig.show()
```

For this data set, we were given population density predictions for 2100. Instead of building my own model to predict the values of each countries' population density, we are just going to use the predicted values that were given to us

In [1125]:
```
temp = density.where(density['Year'] ==2100)
temp = temp.dropna()
temp.head()
```

Out[1125]:

	Entity	Code	Year	Density
41	Afghanistan	AFG	2100.0	160.302872
83	Albania	ALB	2100.0	77.114003
125	Algeria	DZA	2100.0	18.585888
167	American Samoa	ASM	2100.0	886.747850
209	Andorra	AND	2100.0	58.360026

In [1126]:
```
denseCountries = temp['Entity'].unique()
```

In [1127]:
```
densePred = {}
for c in denseCountries:
    val = temp['Density'].where(temp['Entity'] == c)
    val = val.dropna()
    densePred[c] = val.ravel()[0]
densePred
```

Out[1127]:

```
{'Afghanistan': 160.3028724,
 'Albania': 77.11400302,
 'Algeria': 18.58588765,
 'American Samoa': 886.7478501,
 'Andorra': 58.36002560000001,
 'Angola': 49.76746179999999,
 'Anguilla': 134.19534140000002,
 'Antigua and Barbuda': 212.51775899999998,
 'Argentina': 16.83346043,
 'Armenia': 46.212903600000004,
 'Aruba': 416.941708,
 'Australia': 4.259512421,
 'Austria': 92.13029806,
 'Azerbaijan': 77.82221034,
 'Bahamas': 37.24656837,
 'Bahrain': 1367.6529679999999,
 'Bangladesh': 1884.267525,
 'Barbados': 275.47535760000005,
 'Belarus': 17.47814076,
 'Belgium': 327.56994510000004,
 'Belize': 22.89166704,
 'Benin': 243.53127310000002,
 'Bermuda': 952.9321048,
 'Bhutan': 23.15763195,
 'Bolivia': 12.21001815,
 'Bosnia and Herzegovina': 32.1830898,
 'Botswana': 5.623990125,
 'Brazil': 21.21157304,
 'British Indian Ocean Territory': 16.66666667,
 'British Virgin Islands': 146.7904739,
 'Brunei': 120.79944509999999,
 'Bulgaria': 21.40873825,
 'Burkina Faso': 163.1634666,
 'Burundi': 946.2293115,
 'Cambodia': 132.91432360000002,
 'Cameroon': 86.39899240000001,
 'Canada': 5.7976001770000005,
 'Cape Verde': 143.9712279,
 'Cayman Islands': 248.1415199,
 'Central African Republic': 14.043427800000002,
 'Chad': 33.0164728,
 'Chile': 22.2566753,
 'China': 79.85774851,
 'Christmas Island': 6.701379814,
 'Cocos Islands': 0.0,
 'Colombia': 43.76379101,
 'Comoros': 757.4866194,
 'Congo': 23.95647605,
 'Cook Islands': 105.1351832,
 'Costa Rica': 102.7279887,
 "Cote d'Ivoire": 149.8262341,
 'Croatia': 47.34568063,
 'Cuba': 41.16782118,
 'Cyprus': 130.06780220000002,
 'Czechia': 96.10976608,
 'Democratic Republic of Congo': 86.16836402,
 'Denmark': 122.93435330000001,
 'Djibouti': 76.5728404,
 'Dominica': 64.59641193,
 'Dominican Republic': 219.011175,
```

```
'Ecuador': 58.12949665,
'Egypt': 117.2031983,
'El Salvador': 314.9756772,
'Equatorial Guinea': 76.09353014,
'Eritrea': 93.37295745,
'Estonia': 21.74923986,
'Eswatini': 116.91492659999999,
'Ethiopia': 195.7566966,
'Faeroe Islands': 49.55423664,
'Falkland Islands': 0.353865638,
'Fiji': 31.82937673,
'Finland': 18.0448068,
'France': 111.623391,
'French Guiana': 6.507478609,
'French Polynesia': 131.3778426,
'Gabon': 9.244454798,
'Gambia': 440.48725779999995,
'Georgia': 21.03085436,
'Germany': 163.10161010000002,
'Ghana': 198.2783259,
'Gibraltar': 4107.455061,
'Greece': 65.16928713,
'Greenland': 0.1909769,
'Grenada': 117.98728249999999,
'Guadeloupe': 188.5998916,
'Guam': 414.3624077,
'Guatemala': 242.72710519999995,
'Guinea': 130.6384844,
'Guinea-Bissau': 175.348514,
'Guyana': 0.40800436799999995,
'Haiti': 540.1932494,
'Honduras': 110.1779506,
'Hong Kong': 6665.583614,
'Hungary': 82.49359657,
'Iceland': 3.533152697,
'India': 574.3247331,
'Indonesia': 109.8518147,
'Iran': 41.71933965,
'Iraq': 173.07607040000002,
'Ireland': 84.89888399,
'Israel': 369.2287692,
'Italy': 141.9648877,
'Jamaica': 130.8461447,
'Japan': 148.9805956,
'Jordan': 104.4230512,
'Kazakhstan': 4.381933598,
'Kenya': 177.30168480000003,
'Kiribati': 13338.87895,
'Kuwait': 277.8059013,
'Kyrgyzstan': 23.36471607,
'Laos': 45.73426111,
'Latvia': 21.36971628,
'Lebanon': 386.2738539,
'Lesotho': 78.90504475,
'Liberia': 145.3951323,
'Libya': 5.137862913999999,
'Liechtenstein': 414.563805,
'Lithuania': 26.71534003,
'Luxembourg': 387.8990209,
'Macao': 12993.69228,
```

```
'Madagascar': 90.14778132,
'Malawi': 509.29062300000004,
'Malaysia': 104.8531775,
'Maldives': 1339.840733,
'Mali': 32.50239258,
'Malta': 1190.233828,
'Marshall Islands': 235.352486,
'Martinique': 128.95429620000002,
'Mauritania': 6.14554379,
'Mauritius': 602.7377352000001,
'Mexico': 42.14892276,
'Micronesia (country)': 144.429692,
'Moldova': 45.48033744,
'Monaco': 24656.05166,
'Mongolia': 1.5452379109999999,
'Montenegro': 41.52636341,
'Montserrat': 53.25400549,
'Morocco': 54.11640429,
'Mozambique': 72.90372341,
'Myanmar': 71.91994091,
'Namibia': 4.827093252,
'Nauru': 352.16456439999996,
'Nepal': 350.86472139999995,
'Netherlands': 448.74416619999994,
'Netherlands Antilles': 118.91919320000001,
'New Caledonia': 19.03460892,
'New Zealand': 18.55925873,
'Nicaragua': 48.07799968,
'Niger': 84.62986836,
'Nigeria': 368.2231618,
'Niue': 5.118964438,
'Norfolk Island': 37.07209528,
'North Korea': 156.78580549999998,
'North Macedonia': 43.15288011,
'Norway': 20.81645588,
'Oman': 15.46935382,
'Pakistan': 501.3326556,
'Palau': 54.13167088,
'Palestine': 21339.031469999998,
'Panama': 60.59998953,
'Papua New Guinea': 28.99124027,
'Paraguay': 23.9027083,
'Peru': 26.26159159,
'Philippines': 457.6392977,
'Pitcairn': 8.396616445,
'Poland': 56.58426608,
'Portugal': 76.07896139,
'Puerto Rico': 412.801393,
'Qatar': 158.38040880000003,
'Reunion': 376.6275436,
'Romania': 34.64942201,
'Russia': 4.595071408,
'Rwanda': 1040.559649,
'Saint Helena': 21.97050135,
'Saint Kitts and Nevis': 195.7942477,
'Saint Lucia': 275.4905773,
'Saint Pierre and Miquelon': 29.62458633,
'Saint Vincent and the Grenadines': 76.82107164,
'Samoa': 61.43029444,
'San Marino': 387.0012366,
```

```
'Sao Tome and Principe': 272.0017797,
'Saudi Arabia': 24.03759339,
'Senegal': 139.74826940000003,
'Serbia': 94.52433942,
'Seychelles': 501.4201834,
'Sierra Leone': 124.07529550000001,
'Singapore': 12542.789990000001,
'Slovakia': 68.95365641,
'Slovenia': 74.42926127,
'Solomon Islands': 34.50165159,
'Somalia': 22.43583413,
'South Africa': 63.51193292,
'South Korea': 177.00311269999997,
'Spain': 127.5651915,
'Sri Lanka': 192.40275269999998,
'Sudan': 28.23825617,
'Suriname': 4.583813062,
'Svalbard and Jan Mayen': 0.034231028999999996,
'Sweden': 30.21175112,
'Switzerland': 260.9622357,
'Syria': 178.19514880000003,
'Taiwan': 293.66571810000005,
'Tajikistan': 80.02949328,
'Tanzania': 94.69002956,
'Thailand': 100.82068100000001,
'Timor': 345.5217606,
'Togo': 220.93638109999998,
'Tokelau': 1054.9435349999999,
'Tonga': 222.3111806,
'Trinidad and Tobago': 143.25753480000003,
'Tunisia': 62.57454558,
'Turkey': 98.83857208,
'Turkmenistan': 10.69468179,
'Turks and Caicos Islands': 65.39495892,
'Tuvalu': 405.1227124,
'Uganda': 642.4284733,
'Ukraine': 29.06866176,
'United Arab Emirates': 107.2744295,
'United Kingdom': 307.4712733,
'United States': 36.31717384,
'United States Virgin Islands': 349.63341549999996,
'Uruguay': 8.561067527999999,
'Uzbekistan': 112.4489055,
'Vanuatu': 40.61398318,
'Vatican': 2390.9658670000003,
'Venezuela': 37.2085744,
'Vietnam': 206.8363846,
'Wallis and Futuna': 141.2920004,
'Western Sahara': 2.5441482630000003,
'Yemen': 92.29107737,
'Zambia': 51.06381205,
'Zimbabwe': 59.56232931}
```


Nice...got the predicted values of each countries' population density for 2100 in the densePred dictionary

For the fun of things, let's look at building our own model to get a prediction on our own. First, let's start with just building and visualizing the model's predicted values vs. the actual values for the United States to see if our model is looking realistic

In [1128]:
```
usDense = density.where(density['Entity'] == "United States")
usDense = usDense.dropna()
usDense.head()
```

Out[1128]:

	Entity	Code	Year	Density
9209	United States	USA	1950.0	17.382990
9210	United States	USA	1955.0	18.852299
9211	United States	USA	1960.0	20.523695
9212	United States	USA	1965.0	21.969547
9213	United States	USA	1970.0	23.072290

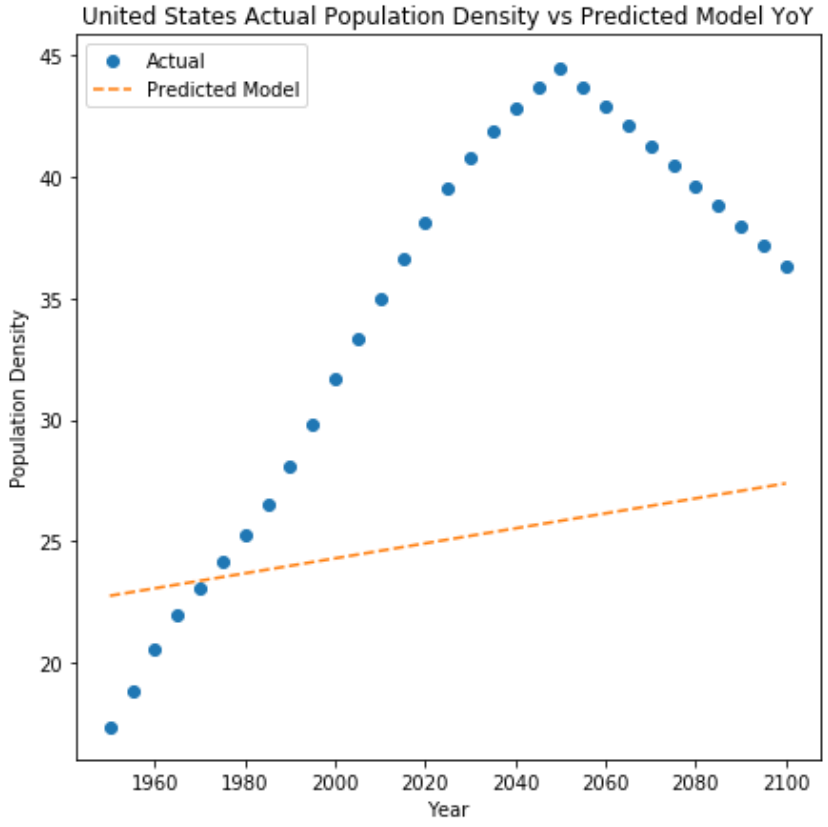
In [1129]:
```
xDenseUS = usDense['Year']
yDenseUS = usDense['Density']
xDenseUS = xDenseUS.ravel()
yDenseUS = yDenseUS.ravel()
xDenseUS = xDenseUS.reshape(-1, 1)
usDenseMLR = LinearRegression()
usDenseMLR.fit(xDenseUS, yDenseUS)
```

Out[1129]:
```
LinearRegression()
```

In [1131]:
```
usDensePred = []
years = list(range(1950, 2123))
for year in years:
    temp = [year]
    predVal = usDenseMLR.predict([temp])
    usDensePred.append(predVal)
```

In [1132]:
```
plt.figure(figsize=(7, 7))
plt.plot(usDense["Year"], usDense["Density"],  linestyle='', marker='o', label='Actual')
plt.plot(usDense["Year"], usDensePred[:len(usDense)],  linestyle='--', label='Predicted Model')
plt.ylabel("Population Density")
plt.xlabel("Year")
plt.title("United States Actual Population Density vs Predicted Model YoY")
plt.legend()
```

Out[1132]: <matplotlib.legend.Legend at 0x1c38bf526c8>



As shown above, our model doesn't look like it is doing the best job of predicting the United States' population density. Since we were given predicted values for each country, we will just stick with those rather than use this model which appears to be not very accurate

Median Age

In [1133]:
```
age = pd.read_csv("median-age.csv") # Source: https://ourworldindata.org/grapher/median-age
age.head()
```

Out[1133]:

	Entity	Code	Year	UN Population Division (Median Age) (2017)
0	Afghanistan	AFG	1950	19.400000
1	Afghanistan	AFG	1955	19.200001
2	Afghanistan	AFG	1960	18.799999
3	Afghanistan	AFG	1965	18.400000
4	Afghanistan	AFG	1970	17.900000

In [1134]:
```
# Change the column name
# UN Population Division (Median Age) (2017)
# to
# MedianAge
age = age.rename(columns={'UN Population Division (Median Age) (2017)' : 'MedianAge'})
```

In [1135]:
```
age.head()
```

Out[1135]:

	Entity	Code	Year	MedianAge
0	Afghanistan	AFG	1950	19.400000
1	Afghanistan	AFG	1955	19.200001
2	Afghanistan	AFG	1960	18.799999
3	Afghanistan	AFG	1965	18.400000
4	Afghanistan	AFG	1970	17.900000

In [1136]:
```
import plotly.express as px

# Neat little visualization of median age for a selected country
# Source: https://plotly.com/python/sliders/
# Source: https://plotly.com/python/line-and-scatter/
fig = px.scatter(age, x="Year", y="MedianAge", animation_frame="Entity", animation_
group="Year",
                 range_x=[1950,2020], range_y=[5,60],color='MedianAge')

fig["layout"].pop("updatemenus") # optional, drop animation buttons
fig.show()
```

For this data set, we were given median age predictions for 2100. Instead of building my own model to predict the values of each countries' median age, we can just use the predicted values that were given to us

In [1137]:
```
temp = age.where(age['Year'] ==2100)
temp = temp.dropna()
temp.head()
```

Out[1137]:

	Entity	Code	Year	MedianAge
30	Afghanistan	AFG	2100.0	43.200001
92	Albania	ALB	2100.0	53.200001
123	Algeria	DZA	2100.0	47.200001
154	Angola	AGO	2100.0	31.000000
185	Antigua and Barbuda	ATG	2100.0	47.299999

In [1138]:
```
ageCountries = temp['Entity'].unique()
```

In [1139]:
```
agePred = {}
for c in ageCountries:
    val = temp['MedianAge'].where(temp['Entity'] == c)
    val = val.dropna()
    agePred[c] = val.ravel()[0]
agePred
```

Out[1139]:

```
{'Afghanistan': 43.200001,
 'Albania': 53.200001,
 'Algeria': 47.200001,
 'Angola': 31.0,
 'Antigua and Barbuda': 47.299999,
 'Argentina': 46.900002,
 'Armenia': 50.599998,
 'Aruba': 46.900002,
 'Australia': 46.700001,
 'Austria': 49.400002,
 'Azerbaijan': 46.099998,
 'Bahamas': 47.5,
 'Bahrain': 49.0,
 'Bangladesh': 50.599998,
 'Barbados': 46.299999,
 'Belarus': 45.299999,
 'Belgium': 47.099998,
 'Belize': 44.200001,
 'Benin': 32.599998,
 'Bhutan': 48.799999,
 'Bolivia': 45.099998,
 'Bosnia and Herzegovina': 49.799999,
 'Botswana': 44.099998,
 'Brazil': 50.799999,
 'Brunei': 49.0,
 'Bulgaria': 47.799999,
 'Burkina Faso': 33.900002,
 'Burundi': 31.9,
 'Cambodia': 45.700001,
 'Cameroon': 35.099998,
 'Canada': 47.700001,
 'Cape Verde': 47.900002,
 'Central African Republic': 37.700001,
 'Chad': 33.400002,
 'Channel Islands': 49.599998,
 'Chile': 50.099998,
 'China': 49.599998,
 'Colombia': 49.400002,
 'Comoros': 36.0,
 'Congo': 33.599998,
 'Costa Rica': 51.099998,
 "Cote d'Ivoire": 30.700001,
 'Croatia': 50.5,
 'Cuba': 50.799999,
 'Curacao': 48.299999,
 'Cyprus': 50.400002,
 'Czechia': 47.799999,
 'Democratic Republic of Congo': 35.799999,
 'Denmark': 46.700001,
 'Djibouti': 42.200001,
 'Dominican Republic': 49.0,
 'Ecuador': 48.400002,
 'Egypt': 42.200001,
 'El Salvador': 51.700001,
 'Equatorial Guinea': 38.400002,
 'Eritrea': 39.299999,
 'Estonia': 49.0,
 'Eswatini': 41.799999,
 'Ethiopia': 43.0,
 'Fiji': 45.099998,
```

```
'Finland': 47.900002,
'France': 47.299999,
'French Guiana': 42.200001,
'French Polynesia': 49.900002,
'Gabon': 40.200001,
'Gambia': 35.900002,
'Georgia': 47.299999,
'Germany': 50.0,
'Ghana': 36.400002,
'Greece': 51.900002,
'Grenada': 51.900002,
'Guadeloupe': 51.5,
'Guam': 49.900002,
'Guatemala': 46.400002,
'Guinea': 35.900002,
'Guinea-Bissau': 35.200001,
'Guyana': 44.299999,
'Haiti': 42.599998,
'Honduras': 48.099998,
'Hong Kong': 48.700001,
'Hungary': 48.599998,
'Iceland': 49.900002,
'India': 45.900002,
'Indonesia': 43.900002,
'Iran': 51.200001,
'Iraq': 34.5,
'Ireland': 47.5,
'Israel': 45.0,
'Italy': 50.5,
'Jamaica': 56.0,
'Japan': 51.799999,
'Jordan': 44.799999,
'Kazakhstan': 43.400002,
'Kenya': 40.0,
'Kiribati': 36.799999,
'Kuwait': 43.5,
'Kyrgyzstan': 43.299999,
'Laos': 47.400002,
'Latvia': 47.599998,
'Lebanon': 51.099998,
'Lesotho': 40.700001,
'Liberia': 35.799999,
'Libya': 46.299999,
'Lithuania': 46.400002,
'Luxembourg': 46.099998,
'Macao': 47.400002,
'Madagascar': 35.900002,
'Malawi': 37.099998,
'Malaysia': 47.599998,
'Maldives': 51.200001,
'Mali': 34.299999,
'Malta': 50.0,
'Martinique': 51.400002,
'Mauritania': 33.299999,
'Mauritius': 48.799999,
'Mayotte': 44.700001,
'Melanesia': 38.0,
'Mexico': 50.400002,
'Micronesia (country)': 42.299999,
'Moldova': 49.299999,
```

```
'Mongolia': 42.599998,
'Montenegro': 50.299999,
'Morocco': 49.5,
'Mozambique': 34.299999,
'Myanmar': 42.599998,
'Namibia': 41.099998,
'Nepal': 51.099998,
'Netherlands': 48.700001,
'New Caledonia': 46.799999,
'New Zealand': 48.599998,
'Nicaragua': 51.400002,
'Niger': 29.4,
'Nigeria': 33.099998,
'North Korea': 46.900002,
'North Macedonia': 48.700001,
'Norway': 47.099998,
'Oman': 49.5,
'Pakistan': 41.5,
'Palestine': 40.400002,
'Panama': 47.599998,
'Papua New Guinea': 37.200001,
'Paraguay': 45.900002,
'Peru': 48.900002,
'Philippines': 42.099998,
'Poland': 51.5,
'Polynesia': 45.299999,
'Portugal': 52.5,
'Puerto Rico': 53.599998,
'Qatar': 47.0,
'Reunion': 51.200001,
'Romania': 48.599998,
'Russia': 43.900002,
'Rwanda': 43.099998,
'Saint Lucia': 52.5,
'Saint Vincent and the Grenadines': 51.099998,
'Samoa': 42.700001,
'Sao Tome and Principe': 34.799999,
'Saudi Arabia': 47.5,
'Senegal': 34.799999,
'Serbia': 48.700001,
'Seychelles': 47.400002,
'Sierra Leone': 37.299999,
'Singapore': 56.799999,
'Slovakia': 48.599998,
'Slovenia': 49.0,
'Solomon Islands': 40.400002,
'Somalia': 31.1,
'South Africa': 43.299999,
'South Korea': 51.599998,
'South Sudan': 35.299999,
'Spain': 51.700001,
'Sri Lanka': 51.400002,
'Sudan': 34.599998,
'Suriname': 45.700001,
'Sweden': 46.5,
'Switzerland': 49.400002,
'Syria': 47.099998,
'Taiwan': 50.299999,
'Tajikistan': 41.700001,
'Tanzania': 32.799999,
```

```
'Thailand': 50.5,
'Timor': 38.099998,
'Togo': 33.799999,
'Tonga': 40.099998,
'Trinidad and Tobago': 45.900002,
'Tunisia': 47.700001,
'Turkey': 50.200001,
'Turkmenistan': 41.799999,
'Uganda': 34.5,
'Ukraine': 46.5,
'United Arab Emirates': 45.700001,
'United Kingdom': 47.700001,
'United States': 45.299999,
'United States Virgin Islands': 56.0,
'Uruguay': 48.900002,
'Uzbekistan': 45.900002,
'Vanuatu': 41.200001,
'Venezuela': 47.700001,
'Vietnam': 47.400002,
'Western Sahara': 44.099998,
'World': 41.599998,
'Yemen': 43.099998,
'Zambia': 31.299999,
'Zimbabwe': 41.099998}
```

Nice, got the predictions pulled out into the dictionary agePred to be used for the aggregated data model

For the fun of things let's see building a simple linear regression model and visualizing how this model's predictions compare to the actual values and those of the predictions given

In [1140]:
```
usAge = age.where(age['Entity'] == "United States")
usAge = usAge.dropna()
usAge.head()
```

Out[1140]:

	Entity	Code	Year	MedianAge
6975	United States	USA	1950.0	30.200001
6976	United States	USA	1955.0	30.299999
6977	United States	USA	1960.0	29.799999
6978	United States	USA	1965.0	28.600000
6979	United States	USA	1970.0	28.400000

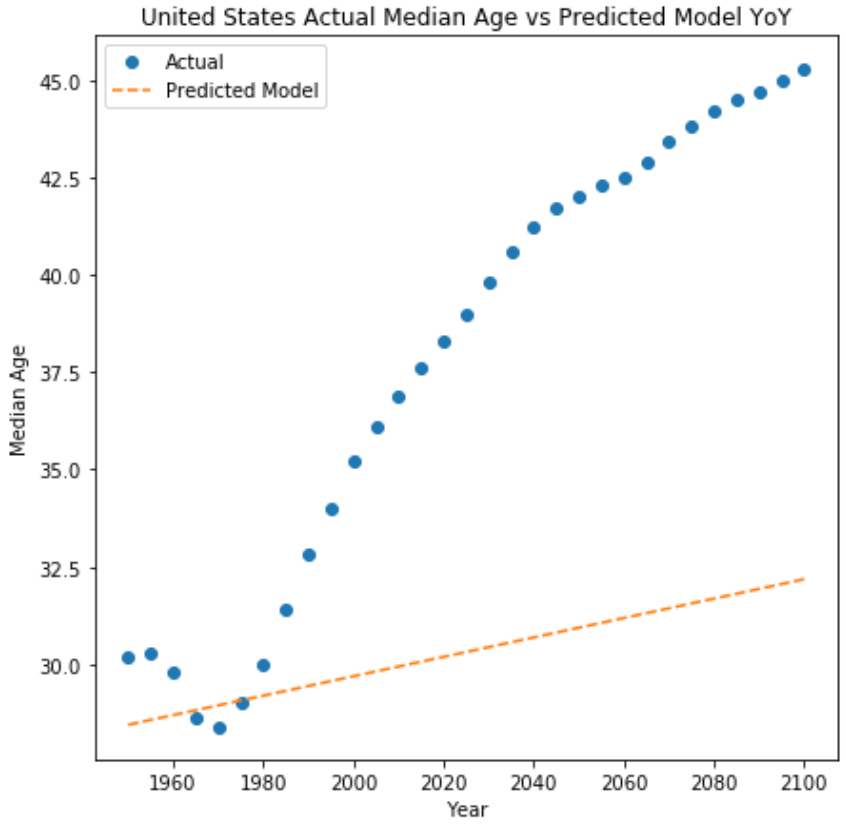
In [1141]:
```
xAgeUS = usAge['Year']
yAgeUS = usAge['MedianAge']
xAgeUS = xAgeUS.ravel()
yAgeUS = yAgeUS.ravel()
xAgeUS = xAgeUS.reshape(-1, 1)
usAgeMLR = LinearRegression()
usAgeMLR.fit(xAgeUS, yAgeUS)
```

Out[1141]:
```
LinearRegression()
```

In [1142]:
```
usAgePred = []
years = list(range(1950, 2123))
for year in years:
    temp = [year]
    predVal = usAgeMLR.predict([temp])
    usAgePred.append(predVal)
```

In [1143]:
```
plt.figure(figsize=(7, 7))
plt.plot(usAge["Year"], usAge["MedianAge"],  linestyle='', marker='o', label='Actual')
plt.plot(usAge["Year"], usAgePred[:len(usAge)],  linestyle='--', label='Predicted Model')
plt.ylabel("Median Age")
plt.xlabel("Year")
plt.title("United States Actual Median Age vs Predicted Model YoY")
plt.legend()
```

Out[1143]: <matplotlib.legend.Legend at 0x1c3920d6c88>



Similar to the population density, the simple linear regression model doesn't appear to be doing the best job so we will just stick with the predicted and actual values that were given to us in building our population prediction model

Building the Model

Population Data Set

In [1145]:
```
population = pd.read_csv("population.csv") # Source: https://ourworldindata.org/grapher/population-since-1800?country=CHN~IND~ZAF~NGA~USA~IRN~BRA
population.head()
```

Out[1145]:

	Entity	Code	Year	Population (historical estimates)
0	Afghanistan	AFG	1800	3280000
1	Afghanistan	AFG	1801	3280000
2	Afghanistan	AFG	1802	3280000
3	Afghanistan	AFG	1803	3280000
4	Afghanistan	AFG	1804	3280000

In [1147]:
```
population = population.rename(columns={'Population (historical estimates)' : 'Population'})
```

In [1148]:
```
# Let's use that visualization from above to look at each country's population YoY
fig = px.scatter(population, x="Year", y="Population", animation_frame="Entity", animation_group="Year",
                 range_x=[1950,2019], range_y=[10000,50000000],color='Population')

fig["layout"].pop("updatemenus") # optional, drop animation buttons
fig.show()
```

Aggregate the Data Sets

In [1149]:
```
aggData = pd.merge(population, life,
                   on=['Entity', 'Year'],
                   how='inner')
aggData.head()
```

Out[1149]:

	Entity	Code	Year	Population	Life expectancy
0	Afghanistan	AFG	1950	7752117	27.638
1	Afghanistan	AFG	1951	7840151	27.878
2	Afghanistan	AFG	1952	7935996	28.361
3	Afghanistan	AFG	1953	8039684	28.852
4	Afghanistan	AFG	1954	8151316	29.350

In [1150]:
```
aggData = pd.merge(aggData, fertility,
                   on=['Entity', 'Year'],
                   how='inner')
aggData.head()
```

Out[1150]:

	Entity	Code	Year	Population	Life expectancy	Rate
0	Afghanistan	AFG	1950	7752117	27.638	7.45
1	Afghanistan	AFG	1951	7840151	27.878	7.45
2	Afghanistan	AFG	1952	7935996	28.361	7.45
3	Afghanistan	AFG	1953	8039684	28.852	7.45
4	Afghanistan	AFG	1954	8151316	29.350	7.45

In [1151]:
```
aggData = pd.merge(aggData, age,
                   on=['Entity', 'Year'],
                   how='inner')
aggData.head()
```

Out[1151]:

	Entity	Code_x	Year	Population	Life expectancy	Rate	Code_y	MedianAge
0	Afghanistan	AFG	1950	7752117	27.638	7.45	AFG	19.400000
1	Afghanistan	AFG	1955	8270992	29.854	7.45	AFG	19.200001
2	Afghanistan	AFG	1960	8996967	32.446	7.45	AFG	18.799999
3	Afghanistan	AFG	1965	9956318	34.948	7.45	AFG	18.400000
4	Afghanistan	AFG	1970	11173654	37.409	7.45	AFG	17.900000

In [1152]:
```
aggData = aggData.drop(["Code_x"], axis = 1)
aggData = aggData.drop(["Code_y"], axis = 1)
```

In [1153]:
```
aggData = pd.merge(aggData, density,
                   on=['Entity', 'Year'],
                   how='inner')
aggData.head()
```

Out[1153]:

	Entity	Year	Population	Life expectancy	Rate	MedianAge	Code	Density
0	Afghanistan	1950	7752117	27.638	7.45	19.400000	AFG	12.780822
1	Afghanistan	1955	8270992	29.854	7.45	19.200001	AFG	13.805061
2	Afghanistan	1960	8996967	32.446	7.45	18.799999	AFG	15.077664
3	Afghanistan	1965	9956318	34.948	7.45	18.400000	AFG	16.638278
4	Afghanistan	1970	11173654	37.409	7.45	17.900000	AFG	18.563737

In [1154]:
```
aggData = aggData.drop(["Code"], axis = 1)
```

In [1155]:
```
aggData.head()
```

Out[1155]:

	Entity	Year	Population	Life expectancy	Rate	MedianAge	Density
0	Afghanistan	1950	7752117	27.638	7.45	19.400000	12.780822
1	Afghanistan	1955	8270992	29.854	7.45	19.200001	13.805061
2	Afghanistan	1960	8996967	32.446	7.45	18.799999	15.077664
3	Afghanistan	1965	9956318	34.948	7.45	18.400000	16.638278
4	Afghanistan	1970	11173654	37.409	7.45	17.900000	18.563737

Get rid of the continents to make sure no duplicate data being counted

In [1162]:
```
continents = ['Asia', 'Africa', 'Europe', 'North America', 'South America', 'Oceani
a', 'Australia', 'Antarctica', 'World']
for c in continents:
    aggData = aggData.where(mergedData['Entity'] != c)
aggData = aggData.dropna()
```

Let's check the current population total in the merged data set and compare it to what we know to be true. If there is missing data for the total population, we can multiply our predicted value for 2122 by this ratio to adjust for the missing data

In [1163]:
```
currPop = aggData.where(aggData['Year'] == 2015)
currPop = currPop.dropna()
currPop.head()
```

Out[1163]:

	Entity	Year	Population	Life expectancy	Rate	MedianAge	Density
13	Afghanistan	2015.0	34413603.0	63.377	4.976	17.299999	53.694768
27	Albania	2015.0	2890524.0	78.025	1.677	36.200001	114.907872
41	Algeria	2015.0	39728020.0	76.090	3.043	27.500000	16.418910
55	Angola	2015.0	27884380.0	59.398	5.774	16.400000	17.334907
69	Antigua and Barbuda	2015.0	93571.0	76.483	2.003	30.700001	222.489788

In [1164]:
```
currPop['Population'].sum()
```

Out[1164]:
```
7074757788.0
```

In [1165]:
```
# Ratio used to multiply the model's predicted value by to account for the missing
 population data in 2015
# Source: https://www.worldometers.info/world-population/world-population-by-year/
popRatio = 7379797139 / currPop['Population'].sum()
popRatio
```

Out[1165]:
```
1.0431165787071042
```

In [1166]:
```
predictedCountriesPopulation = {}
countries = aggData['Entity'].unique()
for country in countries:
    subsetData = aggData.where(aggData["Entity"] == country)
    subsetData = subsetData.dropna()
    xSub = subsetData.iloc[:, 1:]
    xSub = xSub.drop(["Population"], axis = 1)
    ySub = subsetData["Population"]
    toPredict = [2122, predictedCountriesLifeExpectancy[country][0], logPredictedCo
untriesFertilityRate[country][0],
                 agePred[country], densePred[country]]
    popMLR = LinearRegression()
    popMLR.fit(xSub, ySub)
    predVal = popMLR.predict([toPredict])
    predictedCountriesPopulation[country] = predVal

predictedCountriesPopulation
```

Out[1166]:

```
{'Afghanistan': array([1.24068689e+08]),
 'Albania': array([2031912.32261943]),
 'Algeria': array([29614529.32416539]),
 'Angola': array([56223556.35307954]),
 'Antigua and Barbuda': array([162904.1759164]),
 'Azerbaijan': array([6771691.94265185]),
 'Bahamas': array([497233.38658507]),
 'Bahrain': array([4217350.24545725]),
 'Bangladesh': array([3.0571896e+08]),
 'Barbados': array([346648.56691185]),
 'Belarus': array([8079770.83255269]),
 'Belgium': array([6664300.8566906]),
 'Belize': array([391173.19276532]),
 'Benin': array([28074934.31437854]),
 'Bhutan': array([920277.88910008]),
 'Bolivia': array([17237505.00855285]),
 'Bosnia and Herzegovina': array([3042896.43073131]),
 'Botswana': array([4395102.01932091]),
 'Brazil': array([1.81466739e+08]),
 'Brunei': array([658031.21434567]),
 'Bulgaria': array([2686895.22455844]),
 'Burkina Faso': array([33886379.79330134]),
 'Burundi': array([20697783.53267787]),
 'Cambodia': array([19554449.38370461]),
 'Cameroon': array([42648270.09548377]),
 'Canada': array([74338682.40365541]),
 'Cape Verde': array([424536.87859565]),
 'Central African Republic': array([17958135.49310499]),
 'Chad': array([52331401.35180725]),
 'Chile': array([25610695.00210503]),
 'China': array([3.11823604e+08]),
 'Colombia': array([-41475181.10310507]),
 'Comoros': array([818927.80395977]),
 'Congo': array([11343679.19066169]),
 'Costa Rica': array([4089658.7186426]),
"Cote d'Ivoire": array([48137711.48672973]),
 'Croatia': array([4705697.06251221]),
 'Cuba': array([6143064.14510037]),
 'Cyprus': array([1866272.98027206]),
 'Czechia': array([7640799.79748396]),
 'Eswatini': array([1026589.15937309]),
 'Ethiopia': array([2.55767263e+08]),
 'Fiji': array([500331.59693943]),
 'Finland': array([6176452.56616589]),
 'France': array([74414829.22084284]),
 'French Guiana': array([530161.76852024]),
 'French Polynesia': array([-20633.47689047]),
 'Gabon': array([4340811.33093913]),
 'Gambia': array([4430221.20632134]),
 'Georgia': array([-1337998.74519287]),
 'Germany': array([67380470.67175621]),
 'Ghana': array([33093277.12508774]),
 'Greece': array([5767253.80947297]),
 'Grenada': array([47566.57552662]),
 'Guadeloupe': array([-330830.4005511]),
 'Guam': array([261812.79403712]),
 'Guatemala': array([26057815.84045747]),
 'Guinea': array([28040280.44358639]),
 'Guinea-Bissau': array([2808841.21547525]),
 'Guyana': array([259853.82202163]),
```

```
'Haiti': array([13853412.89220133]),
'Honduras': array([8447741.24573806]),
'Hong Kong': array([8477266.45347668]),
'Hungary': array([7130844.83360475]),
'Iceland': array([529816.85342335]),
'India': array([1.74122614e+09]),
'Indonesia': array([4.43413072e+08]),
'Iran': array([64127344.73462099]),
'Iraq': array([63466186.66810614]),
'Ireland': array([5689368.16071051]),
'Israel': array([12293268.54354271]),
'Italy': array([30035467.26057038]),
'Jamaica': array([2189440.60531382]),
'Japan': array([81401316.51503742]),
'Jordan': array([23314927.56031704]),
'Kazakhstan': array([19079292.79002595]),
'Kenya': array([1.07339349e+08]),
'Kiribati': array([188588.01604947]),
'Kuwait': array([4132651.88411851]),
'Kyrgyzstan': array([2921434.32763186]),
'Laos': array([9281535.14896404]),
'Latvia': array([599647.47692959]),
'Lebanon': array([11436325.23329645]),
'Lesotho': array([-783707.82562946]),
'Liberia': array([12513814.13920106]),
'Libya': array([7627446.14003145]),
'Lithuania': array([1042576.66001534]),
'Luxembourg': array([1123372.56132454]),
'Macao': array([752127.60701783]),
'Madagascar': array([52382472.73559324]),
'Malawi': array([52485380.23362564]),
'Malaysia': array([46528818.56089592]),
'Maldives': array([414864.21390901]),
'Mali': array([52137529.58297632]),
'Malta': array([634485.64914129]),
'Martinique': array([-205281.03477585]),
'Mauritania': array([4965281.50988148]),
'Mauritius': array([737772.18562855]),
'Mexico': array([2.26415939e+08]),
'Micronesia (country)': array([61256.01152514]),
'Moldova': array([5144557.83622319]),
'Mongolia': array([3214016.49451182]),
'Montenegro': array([147389.26808492]),
'Morocco': array([42190298.58436613]),
'Mozambique': array([54251477.41638117]),
'Myanmar': array([64720283.22546434]),
'Namibia': array([3595864.75981168]),
'Nepal': array([3612100.66023174]),
'Netherlands': array([15058400.18442093]),
'New Caledonia': array([299958.91946669]),
'New Zealand': array([4104257.07072143]),
'Nicaragua': array([6038120.17677256]),
'Niger': array([1.0542929e+08]),
'Nigeria': array([3.03210403e+08]),
'North Korea': array([30780661.45505625]),
'North Macedonia': array([1400742.84628328]),
'Norway': array([5636136.06597872]),
'Oman': array([7713530.29567084]),
'Pakistan': array([4.17405058e+08]),
'Palestine': array([7330684.3673626]),
```

```
'Panama': array([4205572.72995596]),
'Papua New Guinea': array([14050898.02601928]),
'Paraguay': array([7126721.13122096]),
'Peru': array([5787907.86779249]),
'Philippines': array([1.13684623e+08]),
'Poland': array([23393237.43415244]),
'Portugal': array([11528729.6588428]),
'Puerto Rico': array([-2395852.04821363]),
'Qatar': array([3156245.35094223]),
'Reunion': array([672643.40127154]),
'Romania': array([1434664.47562025]),
'Russia': array([1.29656556e+08]),
'Rwanda': array([24150357.08037157]),
'Saint Lucia': array([135952.49757289]),
'Saint Vincent and the Grenadines': array([21760.29152924]),
'Samoa': array([236446.87722761]),
'Sao Tome and Principe': array([292536.60138974]),
'Saudi Arabia': array([57591536.55857597]),
'Senegal': array([20088029.10157718]),
'Serbia': array([7673081.6975341]),
'Seychelles': array([151979.42425194]),
'Sierra Leone': array([9083611.66287121]),
'Singapore': array([14287584.54447648]),
'Slovakia': array([2714732.50747455]),
'Slovenia': array([1746154.34198481]),
'Solomon Islands': array([1034993.29582611]),
'Somalia': array([-17415329.34779669]),
'South Africa': array([1.10690733e+08]),
'South Korea': array([49686673.10272133]),
'Spain': array([73644543.29985154]),
'Sri Lanka': array([16734050.54024832]),
'Sudan': array([88346859.34321296]),
'Suriname': array([757715.18518685]),
'Sweden': array([13923942.06883558]),
'Switzerland': array([11725623.21077676]),
'Syria': array([75093322.80259931]),
'Taiwan': array([15677812.53541484]),
'Tajikistan': array([11484119.18390276]),
'Tanzania': array([79753339.96978685]),
'Thailand': array([53967852.3872062]),
'Timor': array([3923445.26300583]),
'Togo': array([11744351.242622]),
'Tonga': array([141958.49889427]),
'Trinidad and Tobago': array([2227769.97162728]),
'Tunisia': array([10951502.93582083]),
'Turkey': array([1.75714548e+08]),
'Turkmenistan': array([4779158.95641577]),
'Uganda': array([1.23120051e+08]),
'Ukraine': array([22754111.06350398]),
'United Arab Emirates': array([37481711.91410422]),
'United Kingdom': array([74248153.92167413]),
'United States': array([3.45053029e+08]),
'United States Virgin Islands': array([52278.05106032]),
'Uruguay': array([1409630.97993345]),
'Uzbekistan': array([55965356.54309781]),
'Vanuatu': array([511160.10650416]),
'Venezuela': array([4659072.55464578]),
'Vietnam': array([38654252.21754986]),
'Western Sahara': array([654069.4845412]),
'Yemen': array([28737383.06824432]),
```

```
 'Zambia': array([43020781.04388222]),
 'Zimbabwe': array([22494926.29644157])}
```

In [1167]:
```
totalPopulationPredicted = sum(predictedCountriesPopulation2.values()) * popRatio
totalPopulationPredicted
```

Out[1167]:
```
array([8.91201894e+09])
```

Nice! Our model predicted a population of ~8.9 billion for 2122 which brings our initial model's prediction down like we wanted (although maybe by a little too much)