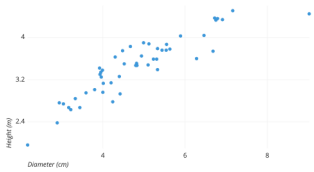
tiachunpun

November 15, 2020

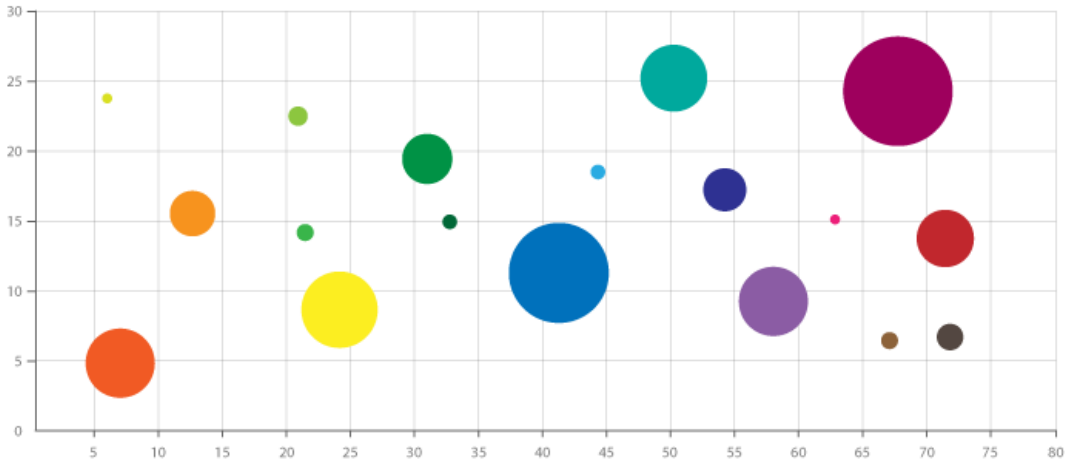
1 Visualization Technique

1.1 Bubble Plots - what are they?

Bubble plots are a more "advanced" version of the well-known scatter plot. Here is an example of a scatter plot.



As you can see, scatter plots visually tell us the correlation / relationship between two variables, the independent variable on the x-axis and the dependent one on the y-axis. Whenever we have a piece of data that belongs in those axis, we plot them as a dot as shown. Now let's see what a bubble plot is compared to the scatter plot.



Bubble plots show us not only the x and y dimension of data, but also the third dimension z, which is the weight of those data points, represented by the size of the circle as shown in the example above. Bubble plots utilize the Cartesian plane to show the relationships between the circles. Unlike scatter plots, the plotted points in bubble plots represent assigned labels or categories, and

sometimes different colors are also used to distinguish between them. The overall plot can be used to determine patterns and trends in the data.

1.2 When to use it

Bubble plots are designed to visually convey three or four dimensions of data. It is a good idea to use bubble plots when you have such data that you want to convey all information graphically all at once. Moreover, as it is fundamentally a scatter plot with different plot sizes, it is also good to use it to convey relationships and correlation between those data points as well.

1.3 When not to use it

The first mistake for choosing a bubble plot to display data is when such data can be explained better in terms of a pie chart. This means that if your data contains more dimensions than four, it is not advisable to crunch your data graphically into a bubble plot. The result of this can be confusion and incorrect analysis of data. Moreover, if many bubbles are included in the plot, this may overwhelm the audience, and so bubble plots usually have limited data size capacity.

2 Visualization Library

2.1 Seaborn

For this task, Seaborn is chosen as the library used to display our bubble plot. As quoted from their website:

`"Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics."`

It is built on top of matplotlib and integrates very well with the Pandas library, which will be used to store tabular data. Seaborn allows declarative API that allows the user to control the elements of the plots with minimal setup, producing beautiful plots that are usually enough for people to interpret data. The library is developed by Michael Waskom in 2014 and is open-source. It includes built-in themes for styling informative data plots.

The reason this library is chosen is because it provides easy-to-use APIs that integrate well with Pandas which produce results that are beautiful with not much effort to style the graph. It also integrates with Jupyter, showing graphs in-line.

2.2 Installation

To install, simply run the following command from your favorite package manager

If you use PyPI: `pip install seaborn`

If you use Anaconda: `conda install seaborn`

2.3 Dependencies

Python 3.6+

2.4 Required dependencies

If not already present, these libraries will be downloaded when you install seaborn.

- numpy
- scipy
- pandas
- matplotlib

2.5 Importing the library

[1]:
```
import seaborn as sns
```

3 Dataset

3.1 Selecting Dataset

Before we begin, a bubble plot needs at least 3 or 4 dimensions to display data visually. I have selected the following dataset for this task.

https://www.gapminder.org/data/

The above site contains dataset for populations of each country and many more information related to the population. For the sake of this demo, we would like to graphically display the following information, and analyze if there is a relationship between the data. We would like to explore if there is a relationship between income (gdp per capita) and time. In addition, we would also like to explore if there is a trend between income and life expectancy. Graphically, we will convert those data to a bubble plot to visually understand those data.

Here is what we need:

1. Current year (x-data, independent variable)
2. Income of that year, inflation-adjusted (y-data, dependent variable)
3. Number of years of life expectancy (size of bubbles, secondary dependent variable)

3.2 Retrieving the raw data

We'll load the csv raw data of the GDP per capita in US$, adjusted for inflation, into our Pandas dataframe.

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

income_df = pd.read_csv('income_per_person_gdppercapita_ppp_inflation_adjusted.
 ↪csv')
income_df.dropna(inplace=True)
income_df.head()
```

[2]:
```
       country  1800  1801  1802  1803  1804  1805  1806  1807  1808  …  \
0  Afghanistan   603   603   603   603   603   603   603   603   603  …
1      Albania   667   667   667   667   667   668   668   668   668  …
2      Algeria   715   716   717   718   719   720   721   722   723  …
3      Andorra  1200  1200  1200  1200  1210  1210  1210  1210  1220  …
4       Angola   618   620   623   626   628   631   634   637   640  …

    2031   2032   2033   2034   2035   2036   2037   2038   2039   2040
0   2550   2600   2660   2710   2770   2820   2880   2940   3000   3060
1  19400  19800  20200  20600  21000  21500  21900  22300  22800  23300
2  14300  14600  14900  15200  15500  15800  16100  16500  16800  17100
3  73600  75100  76700  78300  79900  81500  83100  84800  86500  88300
4   6110   6230   6350   6480   6610   6750   6880   7020   7170   7310

[5 rows x 242 columns]
```

Next, we will load the life expectancy for each countries into a separate dataframe.

[3]:
```
life_df = pd.read_csv('life_expectancy_years.csv')
life_df.dropna(inplace=True)
life_df.head()
```

[3]:
```
                country  1800  1801  1802  1803  1804  1805  1806  1807  1808  \
0           Afghanistan  28.2  28.2  28.2  28.2  28.2  28.2  28.1  28.1  28.1
1               Albania  35.4  35.4  35.4  35.4  35.4  35.4  35.4  35.4  35.4
2               Algeria  28.8  28.8  28.8  28.8  28.8  28.8  28.8  28.8  28.8
4                Angola  27.0  27.0  27.0  27.0  27.0  27.0  27.0  27.0  27.0
5  Antigua and Barbuda  33.5  33.5  33.5  33.5  33.5  33.5  33.5  33.5  33.5

   …  2091  2092  2093  2094  2095  2096  2097  2098  2099  2100
0  …  76.5  76.6  76.7  76.9  77.0  77.1  77.3  77.4  77.5  77.7
1  …  87.4  87.5  87.6  87.7  87.8  87.9  88.0  88.1  88.2  88.3
2  …  88.3  88.4  88.5  88.6  88.7  88.8  88.9  89.0  89.1  89.2
4  …  78.7  78.9  79.0  79.1  79.3  79.4  79.5  79.7  79.8  79.9
5  …  86.1  86.2  86.3  86.4  86.5  86.6  86.7  86.8  86.9  87.0

[5 rows x 302 columns]
```

3.3 Cleaning

We would like to clean the above dataframe so that we don't have an overwhelming number of GDP for too many years. We will simply select data from the year 1800 all the way up to the current year, 2020, by multiple of 10 (i.e. 1800, 1810, 1820, … 2010, 2020). We will also would like to select certain countries that can fit in the interquartile range of the GDPs (i.e. lowest, lower 25%, median, upper 25%, and maximum).

[4]:
```
# set the country column as index
income_df.set_index('country', inplace=True)
```

```
life_df.set_index('country', inplace=True)

# remove certain columns that are not in our interested decade years
income_df.columns = income_df.columns.astype(int)
life_df.columns = life_df.columns.astype(int)
cols = [col for col in income_df.columns if (col % 10 == 0) and (col < 2021)]
income_df = income_df[cols]
life_df = life_df[cols]
```

[5]:
```
def filter_countries_by_range(min, max, skip):
    df = income_df[income_df[2020].between(*income_df[2020].quantile([min,␣
 ↪max]).tolist())]
    df = df.sort_values(by=2020, ascending=True)
    df = df.iloc[::skip]
    return df


# select the countries that are in different ranges
df = pd.concat([filter_countries_by_range(0, 0.25, 6),
                filter_countries_by_range(0.25, 0.50, 6),
                filter_countries_by_range(0.50, 0.75, 6),
                filter_countries_by_range(0.75, 1.0, 6)])
df.drop_duplicates(inplace=True)
```

[6]:
```
# find the rows not in both df and drop it
income_countries = df.index.to_list()
life_countries = life_df[life_df.index.isin(income_countries)].index.to_list()
diff = np.setdiff1d(income_countries, life_countries)
df.drop(diff, inplace=True)
df
```

[6]:

country	1800	1810	1820	1830	1840	1850	1860	1870	1880	\
Burundi	418	422	426	430	434	438	442	446	450	
Malawi	350	350	351	351	352	353	353	354	354	
Haiti	633	634	635	703	779	862	954	1060	1170	
Burkina Faso	480	480	481	486	492	498	503	509	515	
Solomon Islands	363	364	364	377	391	405	419	434	450	
Comoros	696	703	710	718	725	732	740	747	755	
Sao Tome and Principe	850	866	882	899	916	933	950	968	986	
Senegal	497	498	499	523	549	576	605	635	666	
Mauritania	527	528	529	555	583	611	642	673	707	
Ghana	696	701	707	712	718	724	730	735	752	
Angola	618	645	674	704	736	769	804	840	878	
Timor-Leste	521	531	542	553	564	575	587	598	610	
India	863	825	786	788	788	790	751	710	726	
Cuba	902	980	1070	1150	1160	1290	1510	1670	1830	

Namibia	540	541	542	629	730	848	984	1140	1330
Georgia	543	544	545	571	599	628	658	690	724
Paraguay	835	836	838	949	1080	1220	1380	1570	1770
Peru	1120	1090	883	707	876	999	1390	1770	909
North Macedonia	690	691	692	751	815	884	919	956	1140
Iraq	3840	3840	3850	3920	3980	4050	4130	4200	4490
Dominican Republic	667	668	669	706	746	787	831	877	926
China	736	721	854	793	737	684	718	754	738
Uruguay	1810	1880	1950	2100	2260	2430	2950	3610	3380
Turkey	1190	1200	1210	1250	1300	1350	1400	1450	1530
Bahamas	1450	1450	1450	1550	1660	1770	1900	2030	2170
Hungary	1250	1250	1250	1350	1460	1810	2210	2660	3090
Slovenia	1410	1410	1410	1530	1660	1810	1880	1950	2320
New Zealand	658	659	660	661	662	1890	3640	5100	6170
Bahrain	1240	1290	1350	1410	1470	1530	1720	1920	2150
Austria	1850	1920	2110	2430	2630	2860	3080	3230	3600
United Arab Emirates	998	1040	1080	1120	1160	1200	1250	1300	1500
Qatar	1100	1140	1180	1230	1270	1320	1370	1430	1650

	1890	…	1930	1940	1950	1960	1970	1980	\
country		…							
Burundi	454	…	482	497	536	660	807	865	
Malawi	355	…	394	433	473	575	642	889	
Haiti	1300	…	1950	2250	2490	2510	2140	2770	
Burkina Faso	520	…	534	529	497	638	780	765	
Solomon Islands	466	…	672	796	941	1110	1240	1340	
Comoros	763	…	765	742	719	914	1620	2160	
Sao Tome and Principe	1010	…	1170	1280	1400	1480	2330	3120	
Senegal	699	…	1130	1400	1780	2040	2240	2170	
Mauritania	742	…	924	986	1050	1410	2550	2940	
Ghana	769	…	1760	1860	1680	2250	2340	1990	
Angola	917	…	1700	2330	3180	3860	5570	5110	
Timor-Leste	622	…	675	689	704	719	1060	1650	
India	777	…	966	913	824	1000	1190	1330	
Cuba	2390	…	3460	3010	3840	4360	4340	5980	
Namibia	1540	…	2820	3290	4130	5000	6580	6950	
Georgia	737	…	1190	1770	2620	4190	6850	7790	
Paraguay	2010	…	2910	2830	2560	2510	3440	7000	
Peru	894	…	2800	3220	3890	5000	6570	7360	
North Macedonia	1350	…	2140	2290	2540	3960	7100	10600	
Iraq	4800	…	7150	8260	9540	19100	21600	34900	
Dominican Republic	978	…	1540	1920	2390	3030	3580	5370	
China	769	…	807	770	637	843	851	1080	
Uruguay	3570	…	5850	4900	6890	7510	7820	10200	
Turkey	1620	…	1990	2670	2580	3780	5590	7870	
Bahamas	2320	…	5000	7190	11100	17000	26000	27700	
Hungary	3580	…	5850	6390	6030	8880	12600	16300	

```
Slovenia               2760  …  4050   3930   3940   6550   12300   21200
New Zealand            6180  …  8170  10400  13900  15600   18800   21200
Bahrain                2410  …  2890   5570  16000  22600   32800   37800
Austria                4240  …  6220   6870   6430  11300   17300   25000
United Arab Emirates   1730  …  1830   1230   1210   1770   41700  179000
Qatar                  1900  …  1730   1090   1710  17400  174000  138000

                         1990    2000    2010    2020
country
Burundi                  1030     718     734     628
Malawi                    747     884    1080    1210
Haiti                    2110    1760    1510    1640
Burkina Faso              844    1080    1420    1860
Solomon Islands          1880    1820    1870    2170
Comoros                  2580    2390    2430    2520
Sao Tome and Principe    2040    1960    2560    3090
Senegal                  2320    2420    2780    3590
Mauritania               2800    2830    3430    4030
Ghana                    1900    2220    3030    4590
Angola                   4760    3890    6360    5440
Timor-Leste              2590    3270    9010    7010
India                    1910    2710    4450    7630
Cuba                     6860    4880    6700    8370
Namibia                  5710    6540    8680    9660
Georgia                  7980    3530    6980   11300
Paraguay                 7930    7980    9740   12300
Peru                     5250    6430   10100   13300
North Macedonia          9630    8620   11400   14400
Iraq                    11600   12200   13200   16000
Dominican Republic       5520    8290   11400   17100
China                    1520    3690    9500   18100
Uruguay                  9840   12900   17200   21300
Turkey                  11400   13900   18000   25500
Bahamas                 31000   33000   29700   28300
Hungary                 17200   18000   22400   30600
Slovenia                18900   22700   28700   34600
New Zealand             24000   28100   32200   36700
Bahrain                 35100   44900   40600   42000
Austria                 31300   38800   43300   47300
United Arab Emirates   112000  103000   55400   65300
Qatar                   70600  108000  120000  116000

[32 rows x 23 columns]
```

[7]:
```
# filter the life-expectancy df to contain only the countries found in df
life_df = life_df[life_df.index.isin(df.index)]
life_df.sort_values(by=2020, ascending=True, inplace=True)
```

```
life_df
```

[7]:

country	1800	1810	1820	1830	1840	1850	1860	1870	1880	\
Burundi	31.5	31.5	31.5	31.5	31.5	31.5	31.5	31.5	31.6	
Burkina Faso	29.2	29.2	29.2	29.2	29.2	29.2	29.2	29.2	29.6	
Malawi	30.3	30.3	30.3	30.3	30.3	30.3	30.3	30.3	30.5	
Angola	27.0	27.0	27.0	27.0	27.0	27.0	27.0	27.0	27.7	
Haiti	29.0	29.0	29.0	29.0	29.0	29.0	29.0	29.0	28.9	
Ghana	28.0	28.0	28.0	28.0	28.0	28.0	28.0	28.0	28.4	
Sao Tome and Principe	31.0	31.0	31.0	31.0	31.0	31.0	31.0	31.0	31.1	
Namibia	32.4	32.4	32.4	32.4	32.4	32.4	32.4	32.4	32.8	
Senegal	25.2	25.2	25.2	25.2	25.2	25.2	25.2	25.2	25.8	
Solomon Islands	25.1	25.1	25.1	25.1	25.1	25.1	25.1	25.1	25.1	
Comoros	32.1	32.1	32.1	32.1	32.1	32.1	32.1	32.1	32.3	
India	25.4	25.4	25.4	25.4	25.4	25.4	23.0	25.4	25.5	
Mauritania	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.3	
Timor-Leste	28.9	28.9	28.9	28.9	28.9	28.9	28.9	28.9	29.4	
Georgia	31.9	31.9	31.9	31.9	31.9	31.9	31.9	29.3	30.8	
Dominican Republic	29.9	29.9	29.9	29.9	29.9	29.9	29.9	29.9	30.3	
United Arab Emirates	30.7	30.7	30.7	30.7	30.7	30.7	30.7	30.7	30.8	
Bahamas	35.2	35.2	35.2	35.2	35.2	35.2	35.2	35.2	35.1	
Paraguay	35.5	35.5	35.5	35.5	35.5	35.5	35.5	35.5	35.7	
North Macedonia	36.1	36.1	36.1	36.1	36.1	36.1	36.1	36.1	36.2	
Iraq	31.2	31.2	31.2	31.2	31.2	31.2	31.2	31.2	31.3	
Hungary	36.0	36.0	36.0	36.0	36.0	36.0	36.0	36.0	36.0	
Uruguay	32.9	32.9	32.9	32.9	32.9	32.9	32.9	32.9	32.9	
China	32.0	32.0	32.0	32.0	32.0	32.0	28.9	32.0	32.0	
Cuba	32.2	33.6	35.0	35.7	36.5	36.3	36.2	29.7	37.0	
Turkey	35.0	35.0	35.0	35.0	35.0	35.0	35.0	35.0	35.2	
Bahrain	30.3	30.3	30.3	30.3	30.3	30.3	30.3	30.3	30.4	
Qatar	30.8	30.8	30.8	30.8	30.8	30.8	30.8	30.8	30.6	
Peru	35.7	35.7	35.7	35.7	35.7	35.7	35.7	35.7	35.3	
Slovenia	36.6	36.6	36.6	36.6	36.6	36.6	36.6	36.6	36.5	
Austria	34.4	34.4	34.4	34.4	34.4	34.4	34.4	34.4	35.0	
New Zealand	34.0	34.0	34.0	34.0	34.0	34.0	34.0	34.0	38.5	

country	1890	…	1930	1940	1950	1960	1970	1980	1990	\
		…								
Burundi	31.7	…	32.2	32.3	39.5	42.6	45.4	47.0	48.7	
Burkina Faso	30.0	…	31.6	32.0	33.3	38.7	44.5	48.4	50.8	
Malawi	30.7	…	31.5	31.7	38.0	40.2	43.4	49.3	48.6	
Angola	28.4	…	31.1	31.8	35.2	40.6	46.5	47.6	47.9	
Haiti	28.9	…	28.5	28.5	35.5	41.2	46.0	50.7	54.5	
Ghana	28.7	…	33.0	39.1	45.3	51.1	55.6	58.0	59.2	
Sao Tome and Principe	31.3	…	31.9	37.0	47.3	52.5	58.5	63.0	63.3	
Namibia	33.1	…	34.6	35.0	43.9	51.6	58.5	59.5	61.6	

Senegal	26.4	…	28.1	31.7	40.9	46.2	48.3	53.1	58.4
Solomon Islands	25.1	…	25.2	32.5	44.6	49.5	54.6	58.9	60.2
Comoros	32.5	…	33.3	33.5	42.1	45.9	50.7	54.0	58.1
India	24.4	…	29.1	32.6	35.2	41.9	49.5	55.0	59.6
Mauritania	32.6	…	33.9	34.2	40.4	47.3	53.9	56.8	59.6
Timor-Leste	29.8	…	31.6	32.0	32.5	38.4	45.6	48.8	60.2
Georgia	32.6	…	40.5	46.3	57.9	61.3	64.7	69.4	70.0
Dominican Republic	30.7	…	32.2	37.8	48.9	57.8	66.0	70.3	72.0
United Arab Emirates	30.9	…	31.4	31.5	42.6	53.9	63.9	69.3	71.2
Bahamas	35.1	…	42.4	50.6	58.8	62.4	65.4	68.6	71.1
Paraguay	35.9	…	36.7	49.5	65.6	67.2	69.3	71.9	74.3
North Macedonia	36.4	…	42.3	47.9	54.6	62.6	68.8	70.7	71.9
Iraq	31.4	…	31.9	32.0	35.9	49.6	60.4	61.0	65.9
Hungary	36.0	…	50.1	56.9	62.0	68.0	69.1	69.2	69.5
Uruguay	32.9	…	39.5	52.7	65.9	67.9	68.8	70.4	73.0
China	32.0	…	33.8	34.3	41.5	31.6	62.5	65.8	68.7
Cuba	39.8	…	42.3	48.5	59.7	65.6	72.0	74.1	74.8
Turkey	35.5	…	36.4	34.5	42.3	48.1	55.8	62.8	68.6
Bahrain	30.5	…	30.9	34.7	42.2	53.3	65.4	71.1	70.5
Qatar	30.4	…	29.7	29.5	51.0	57.8	64.2	67.4	71.5
Peru	35.0	…	33.5	33.1	39.6	43.3	48.0	64.6	70.0
Slovenia	36.5	…	45.5	54.6	63.6	67.7	68.3	70.8	73.9
Austria	37.1	…	56.0	57.9	65.0	69.0	70.3	72.8	75.8
New Zealand	43.0	…	60.8	65.3	69.3	71.2	71.3	72.9	75.3

country	2000	2010	2020
Burundi	44.9	57.8	62.6
Burkina Faso	52.4	58.3	62.9
Malawi	44.6	54.2	64.6
Angola	51.7	59.9	65.4
Haiti	57.4	32.5	66.0
Ghana	59.4	61.6	66.4
Sao Tome and Principe	64.6	68.9	67.5
Namibia	54.5	59.0	67.6
Senegal	60.6	65.9	68.9
Solomon Islands	61.5	61.7	69.0
Comoros	61.6	66.5	69.3
India	62.9	66.7	69.7
Mauritania	63.5	68.2	71.3
Timor-Leste	65.3	70.0	71.5
Georgia	73.1	72.8	73.3
Dominican Republic	73.9	74.0	73.7
United Arab Emirates	71.8	72.9	73.8
Bahamas	71.5	73.4	74.2
Paraguay	74.5	74.6	76.5
North Macedonia	73.9	75.9	76.9

```
Iraq                  66.7  73.5  77.2
Hungary               71.8  74.6  77.3
Uruguay               74.6  76.4  77.5
China                 72.1  75.8  77.7
Cuba                  76.8  78.3  78.6
Turkey                74.1  77.6  79.7
Bahrain               72.6  78.2  79.9
Qatar                 74.1  78.0  80.6
Peru                  74.7  77.5  81.0
Slovenia              76.0  79.5  81.6
Austria               78.3  80.6  82.1
New Zealand           78.3  80.7  82.1

[32 rows x 23 columns]
```

[8]:
```
# rest columns as string
# Converts the data into something plottable. Here we have to convert the df␣
 ↪above into something the library understands
# in order to plot our bubble plot. From the columns of years, we need to␣
 ↪transform the data into a single column containing
# the year values and another column containing the gdp capita associated with␣
 ↪that year. This mapping will be stored as two
# columns that map to a country.
df.columns = df.columns.astype(str)
life_df.columns = life_df.columns.astype(str)
df = pd.melt(df.reset_index(), id_vars='country', value_vars=df.columns.
 ↪to_list(), var_name='year', value_name='gdp per capita')
df
```

[8]:
```
                  country  year  gdp per capita
0                 Burundi  1800             418
1                  Malawi  1800             350
2                   Haiti  1800             633
3            Burkina Faso  1800             480
4         Solomon Islands  1800             363
..                    …   …              …
731           New Zealand  2020           36700
732               Bahrain  2020           42000
733               Austria  2020           47300
734  United Arab Emirates  2020           65300
735                 Qatar  2020          116000

[736 rows x 3 columns]
```

[9]:
```
life_df = pd.melt(life_df.reset_index(), id_vars='country', value_vars=life_df.
 ↪columns.to_list(), var_name='year', value_name='life expectancy')
life_df
```

[9]:
```
           country  year  life expectancy
0          Burundi  1800             31.5
1     Burkina Faso  1800             29.2
2           Malawi  1800             30.3
3           Angola  1800             27.0
4            Haiti  1800             29.0
..             …   …               …
731         Qatar  2020             80.6
732          Peru  2020             81.0
733      Slovenia  2020             81.6
734       Austria  2020             82.1
735   New Zealand  2020             82.1

[736 rows x 3 columns]
```

[10]:
```
# Merge the two df together
df = df.merge(life_df, how='inner')
df
```

[10]:
```
                  country  year  gdp per capita  life expectancy
0                 Burundi  1800             418             31.5
1                  Malawi  1800             350             30.3
2                   Haiti  1800             633             29.0
3            Burkina Faso  1800             480             29.2
4         Solomon Islands  1800             363             25.1
..                    …   …               …               …
731          New Zealand  2020           36700             82.1
732              Bahrain  2020           42000             79.9
733              Austria  2020           47300             82.1
734  United Arab Emirates  2020           65300             73.8
735                 Qatar  2020          116000             80.6

[736 rows x 4 columns]
```

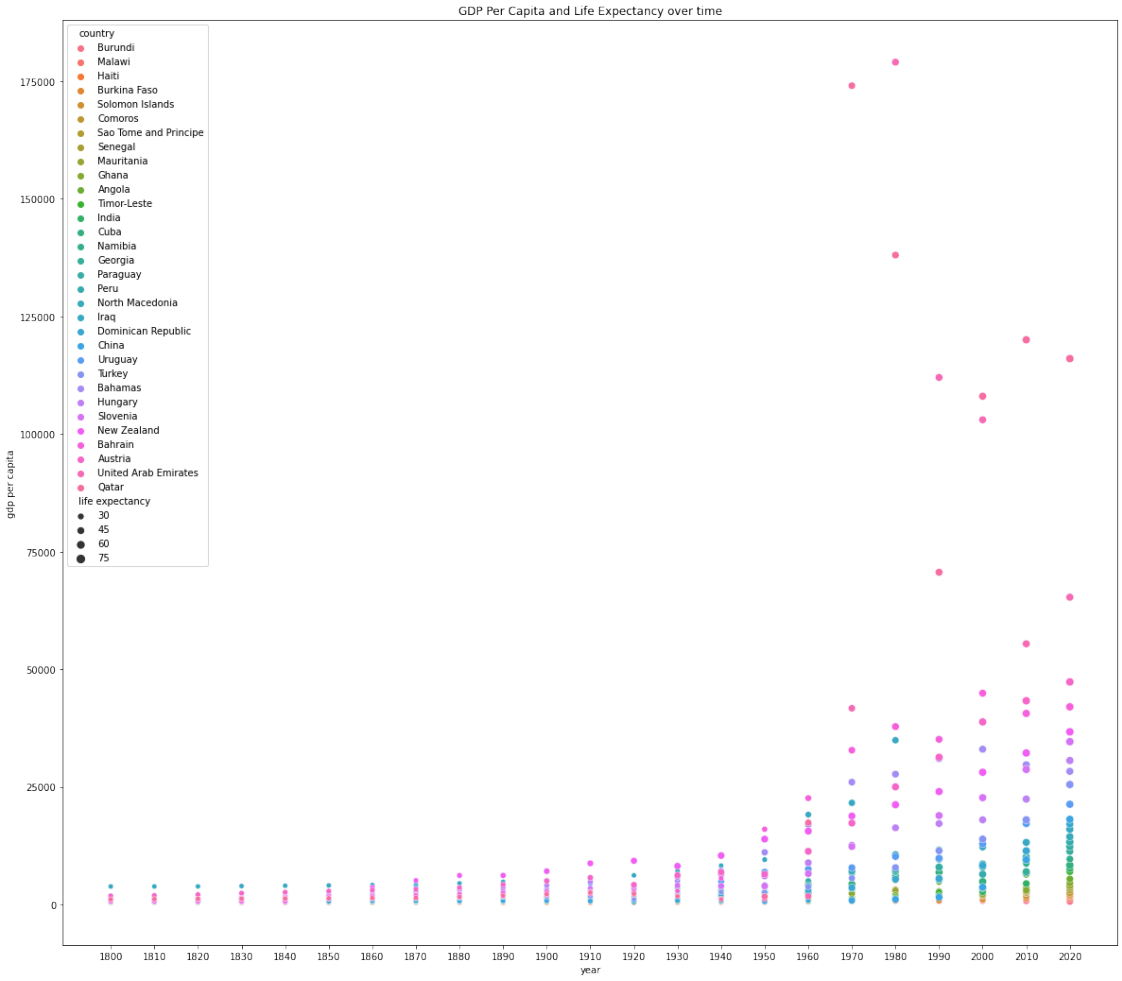
3.4 Plotting

Now that we have the appropriate dataframe, we can plug the variables in to the 'scatterplot' function to plot. From the documentation, the function takes in the x-axis data, the y-axis data, the optional data for controlling the size of the plots, the dataframe itself, and the optional data that controls all the categories of the plot. We simply input all the columns that correspond to those data into the function.

[11]:
```
plt.figure(figsize=(20,18))
sns.scatterplot(x="year",
                y="gdp per capita",
                size="life expectancy",
                data=df,
```

```
                hue='country').set_title('GDP Per Capita and Life Expectancy␣
↪over time')
```

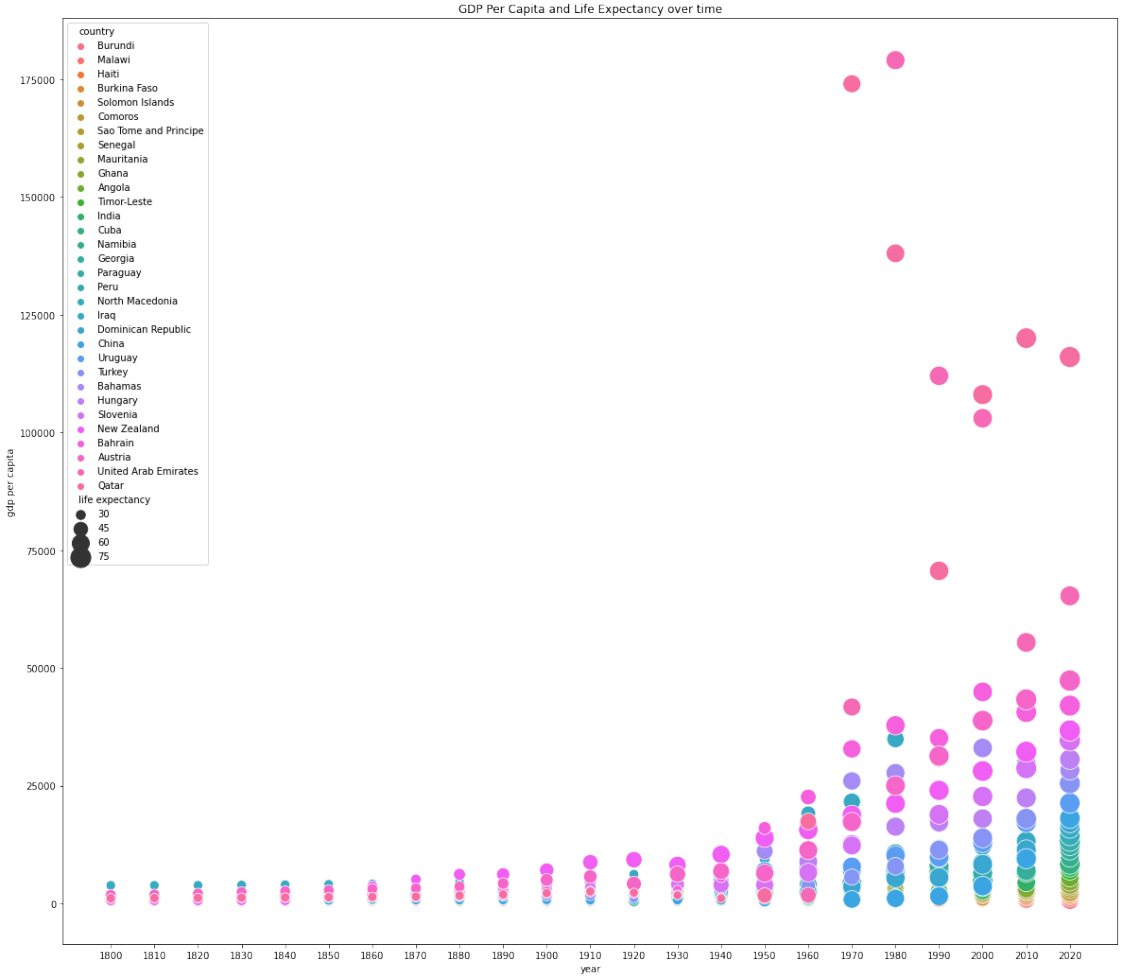
[11]:
```
Text(0.5, 1.0, 'GDP Per Capita and Life Expectancy over time')
```



Hmm, it looks like the bubble sizes are barely distinguishable. Luckily, the function takes an another parameter, sizes, that takes in a list of min and max size values. We'll pass in that list and see how it changes the graph.

[12]:
```
plt.figure(figsize=(20,18))
sns.scatterplot(x="year",
                y="gdp per capita",
                size="life expectancy",
                sizes=(0, 500),
                data=df,
                hue='country').set_title('GDP Per Capita and Life Expectancy␣
↪over time')
```

[12]: `Text(0.5, 1.0, 'GDP Per Capita and Life Expectancy over time')`



3.5 Conclusion

That is better. We can visually learn a number of things from this graph. First, the bubble plot allows us to see the trend between the year and the gdp per capita. We see that the changes seem exponential. Second, we see that the life expectancy improves as GDP per capita improves, regardless of how much it changes. Finally, we see the discrepancy between the 'rich' countries and the rest. Rich countries seem to be leading far ahead in terms of GDP per capita in 2020. Overall, the graph tells us many information about the relationship and the dimensions in the data. The limitation of this graph is also apparent here. You probably don't want to include too many bubble categories like in this graph if the data for it is not spread enough. This is a trial-an-error process. Ideally, to improve upon this graph, I would have to further reduce the number of countries down and be more selective in my criteria in choosing which countries to display.