Investigate_a_Dataset

October 15, 2021

1 Project: Investigate a Dataset - Gapminder World

1.1 Table of Contents

1.1.1 Dataset Description

Gapminder collects all sorts of data about populations around the word, from taxes rates, to CO2 emissions, to the proportion of women contributing to a country's workforce. Their data is for various countries across different years, from 1800s, to predictions up till 2100s. Their site provides all this information for free in .csv or .xlsx format.

1.1.2 Questions for Analysis

For this investigation, I was curious about the factors that affect life expectancy around the world. How does it vary from a country to another? How does it vary from one time period to another? Is average income a great factor in determining the average life expectancy of a country? Does the government's contribution to the health sector greatly affect them? Which of these two variables have the bigger weight? And lastly, as a bonus, is there any correlation between a country's life expectancy and the happiness of its citizens?

To answer these questions, I've collected various datasets from Gapminder's website. I provide brief descriptions from their site below:

- Life Expectancy: The average number of years a newborn baby would live if mortality patterns were to remain the same
- Income per Person: Gross domestic product per person adjusted for differences in purchasing power.
- Govt. Health Spending of Total Gov. Spending: Proportion of total government expenditure that has been expended in health.
- Happiness Score: This is the national average response to the question of life evaluations asking the following "Please imagine a ladder, with steps numbered from 0 at the bottom to 10 at the top. The top of the ladder represents the best possible life for you and the bottom of the ladder represents the worst possible life for you. On which step of the ladder would you

say you personally feel you stand at this time?" This measure is also referred to as Cantril life ladder. Gapminder has converted this indicator's scale from 0 to 100 to easly communicate it in terms of percentage.

```
In [1]: # imports
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns

        % matplotlib inline

        # set style using seaborn for nicer visuals
        sns.set_style('darkgrid')
```

```
In [2]: # Upgrade pandas to use dataframe.explode() function.
        !pip install --upgrade pandas==0.25.0
```

```
Requirement already up-to-date: pandas==0.25.0 in /opt/conda/lib/python3.6/site-packages (0.25.0
Requirement already satisfied, skipping upgrade: pytz>=2017.2 in /opt/conda/lib/python3.6/site-p
Requirement already satisfied, skipping upgrade: python-dateutil>=2.6.1 in /opt/conda/lib/python
Requirement already satisfied, skipping upgrade: numpy>=1.13.3 in /opt/conda/lib/python3.6/site-
Requirement already satisfied, skipping upgrade: six>=1.5 in /opt/conda/lib/python3.6/site-packa
```

Data Wrangling ### General Properties

```
In [3]: # Reading CSV files
        life_exp_df = pd.read_csv('life_expectancy_years.csv')
        income_df = pd.read_csv('income_per_person_gdppercapita_ppp_inflation_adjusted.csv')
        govt_health_spending_df = pd.read_csv('government_health_spending_of_total_gov_spending_
        happiness_score_df = pd.read_csv('hapiscore_whr.csv')
```

```
In [4]: def print_df(df):
            """
            Print some relevant information about the dataframe passed
            """
            print('Dataset Sample:')
            print(df.head())
            print()
            print('Dataset information:')
            print(df.info())
            print()
            print(f'Number of rows: {df.shape[0]}\nNumber of columns: {df.shape[1]}')
            print('-------------------------------------------------------------')
```

```
In [5]: print('Life Expectancy in Years')
        print_df(life_exp_df)
```

```
Life Expectancy in Years
Dataset Sample:
                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                Angola  27.0  27.0  27.0  27.0  27.0  27.0  27.0  27.0  27.0
2               Albania  35.4  35.4  35.4  35.4  35.4  35.4  35.4  35.4  35.4
3               Andorra   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
4  United Arab Emirates  30.7  30.7  30.7  30.7  30.7  30.7  30.7  30.7  30.7

   ...  2091  2092  2093  2094  2095  2096  2097  2098  2099  2100
0  ...  75.5  75.7  75.8  76.0  76.1  76.2  76.4  76.5  76.6  76.8
1  ...  78.8  79.0  79.1  79.2  79.3  79.5  79.6  79.7  79.9  80.0
2  ...  87.4  87.5  87.6  87.7  87.8  87.9  88.0  88.2  88.3  88.4
3  ...   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
4  ...  82.4  82.5  82.6  82.7  82.8  82.9  83.0  83.1  83.2  83.3

[5 rows x 302 columns]

Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Columns: 302 entries, country to 2100
dtypes: float64(301), object(1)
memory usage: 460.2+ KB
None

Number of rows: 195
Number of columns: 302
----------------------------------------------------------------
```

For the life expectancy dataset, we can observe we have values for 195 countries, across 302 years. Some years seem to be mising, but at least the datatype of the life expectancy in years is correct: float.

```
In [6]: print('Income Per Person')
        print_df(income_df)

Income Per Person
Dataset Sample:
                country  1800  1801  1802  1803  1804  1805  1806  1807  1808  \
0           Afghanistan   674   674   674   674   674   674   674   674   674
1                Angola   691   693   697   700   702   705   709   712   716
2               Albania   746   746   746   746   746   747   747   747   747
3               Andorra  1340  1340  1340  1350  1350  1350  1350  1360  1360
4  United Arab Emirates  1120  1120  1120  1130  1130  1140  1140  1150  1150

   ...   2041   2042   2043   2044   2045   2046   2047   2048   2049   2050
```

```
0  ...   2880   2940   3000   3070   3130   3200   3270   3340   3410   3480
1  ...   8040   8220   8390   8570   8750   8940   9120   9320   9520   9720
2  ...  24.5k    25k  25.5k  26.1k  26.6k  27.2k  27.8k  28.3k  28.9k  29.6k
3  ...   108k   111k   113k   116k   118k   121k   123k   126k   128k   131k
4  ...  74.5k  76.1k  77.7k  79.3k    81k  82.7k  84.5k  86.3k  88.1k    90k

[5 rows x 252 columns]

Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Columns: 252 entries, country to 2050
dtypes: int64(103), object(149)
memory usage: 384.0+ KB
None

Number of rows: 195
Number of columns: 252
------------------------------------------------------------
```

The number of countries represented in the income dataset is the same, 195, but only across 252 years. Some numbers appear to have the letter 'k' in them denoting thousands, which results in some columns being integers and others strings. We will have to fix that later to have consistent data.

```
In [7]: print('Government Health Spending Percentage of Total Spending')
        print_df(govt_health_spending_df)

Government Health Spending Percentage of Total Spending
Dataset Sample:
                country   1995   1996   1997   1998   1999   2000   2001  \
0           Afghanistan    NaN    NaN    NaN    NaN    NaN    NaN    NaN
1                Angola   5.00   2.68   3.57   3.15   1.76   3.26   6.06
2               Albania   5.26   6.34   6.47   6.10   7.18   7.03   7.24
3               Andorra  23.60  23.80  23.20  28.70  20.80  19.10  19.20
4  United Arab Emirates   8.09   7.13   8.76   8.00   8.01   7.64   7.73

    2002   2003   2004   2005   2006   2007   2008   2009   2010
0   1.48   1.48   1.48   1.48   1.48   1.48   1.48   1.58   1.59
1   3.74   4.83   4.12   4.38   6.06   5.75   6.40  10.10   7.18
2   7.32   7.64   9.23   9.79   9.05   8.46   8.21   8.42   8.42
3  20.00  22.00  22.70  22.00  22.80  21.30  21.30  21.30  21.30
4   7.98   8.35   8.21   8.70   8.95   8.93   8.85   8.76   8.79

Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 192 entries, 0 to 191
```

```
Data columns (total 17 columns):
country    192 non-null object
1995       189 non-null float64
1996       190 non-null float64
1997       190 non-null float64
1998       191 non-null float64
1999       191 non-null float64
2000       191 non-null float64
2001       191 non-null float64
2002       190 non-null float64
2003       190 non-null float64
2004       190 non-null float64
2005       190 non-null float64
2006       190 non-null float64
2007       190 non-null float64
2008       190 non-null float64
2009       190 non-null float64
2010       187 non-null float64
dtypes: float64(16), object(1)
memory usage: 25.6+ KB
None

Number of rows: 192
Number of columns: 17
------------------------------------------------------------
```

The number of countries in the govt. health spending data is less, and is only across 17 years starting from 1995 till 2010. Some data also appear to be missing, but all datatypes are correct.

```
In [8]: print('Happiness Score')
        print_df(happiness_score_df)

Happiness Score
Dataset Sample:
                country  2005  2006  2007  2008  2009  2010  2011  2012  2013  \
0           Afghanistan   NaN   NaN   NaN  37.2  44.0  47.6  38.3  37.8  35.7
1                Angola   NaN   NaN   NaN   NaN   NaN   NaN  55.9  43.6  39.4
2               Albania   NaN   NaN  46.3   NaN  54.9  52.7  58.7  55.1  45.5
3  United Arab Emirates   NaN  67.3   NaN   NaN  68.7  71.0  71.2  72.2  66.2
4             Argentina   NaN  63.1  60.7  59.6  64.2  64.4  67.8  64.7  65.8

   2014  2015  2016  2017  2018  2019
0  31.3  39.8  42.2  26.6  26.9  25.7
1  37.9   NaN   NaN   NaN   NaN   NaN
2  48.1  46.1  45.1  46.4  50.0  48.8
3  65.4  65.7  68.3  70.4  66.0  67.9
4  66.7  67.0  64.3  60.4  57.9  59.7
```

```
Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 163 entries, 0 to 162
Data columns (total 16 columns):
country    163 non-null object
2005       27 non-null float64
2006       89 non-null float64
2007       101 non-null float64
2008       109 non-null float64
2009       113 non-null float64
2010       123 non-null float64
2011       145 non-null float64
2012       140 non-null float64
2013       135 non-null float64
2014       143 non-null float64
2015       141 non-null float64
2016       140 non-null float64
2017       146 non-null float64
2018       134 non-null float64
2019       151 non-null float64
dtypes: float64(15), object(1)
memory usage: 20.5+ KB
None

Number of rows: 163
Number of columns: 16
------------------------------------------------------------
```

The happiness score is available for only 163 countries for the years 2005-2019. Some data is missing for some countries as well.

1.1.3 Data Cleaning

Now we will start cleaning our data according to the observations we made: 1. We will check for duplicates for all datasets and drop any duplicates we find 2. We will check the rows with missing values for each dataset and if insignificant, drop the countries with missing values, and fill them somehow otherwise 3. For the income dataset, we need to change the datatype of the columns that are string to int by dealing with the k problem 4. Finally we will start merging some datasets to explore them more efficiently later on

1. Checking for duplicates

```
In [9]: life_exp_df.duplicated().sum()

Out[9]: 0

In [10]: income_df.duplicated().sum()
```

```
Out[10]: 0

In [11]: govt_health_spending_df.duplicated().sum()

Out[11]: 0

In [12]: happiness_score_df.duplicated().sum()

Out[12]: 0
```

Nothing seems to be duplicated. Superb!

2. Checking for missing values

```
In [13]: life_exp_df.isna().sum()

Out[13]: country    0
         1800       9
         1801       9
         1802       9
         1803       9
                   ..
         2096       9
         2097       9
         2098       9
         2099       9
         2100       9
         Length: 302, dtype: int64
```

It seems like the same 9 countries don't have values in this dataset. Let's drop them.

```
In [14]: life_exp_df.dropna(inplace=True)
         life_exp_df.isna().sum().sum()

Out[14]: 0

In [15]: income_df.isna().sum().sum()

Out[15]: 0

In [16]: govt_health_spending_df.isna().sum()

Out[16]: country    0
         1995       3
         1996       2
         1997       2
         1998       1
         1999       1
         2000       1
         2001       1
```

```
2002       2
2003       2
2004       2
2005       2
2006       2
2007       2
2008       2
2009       2
2010       5
dtype: int64
```

Number of missing values seem to be insignificant. Let's drop those as well.

```
In [17]: govt_health_spending_df.dropna(inplace=True)
         govt_health_spending_df.isna().sum().sum()

Out[17]: 0

In [18]: happiness_score_df.isna().sum()

Out[18]: country      0
         2005       136
         2006        74
         2007        62
         2008        54
         2009        50
         2010        40
         2011        18
         2012        23
         2013        28
         2014        20
         2015        22
         2016        23
         2017        17
         2018        29
         2019        12
         dtype: int64
```

For the year 2005, it seems data is missing for 136 countries out of 163. It would be simpler to drop the column itself. As for the rest of the years, it seems a bit excessive to drop all of these rows, so we will try filling them with the year before them instead. Let's try filling them before deleting the 2005 column to retain some of the data as well.

```
In [19]: happiness_score_df.ffill(inplace=True)
         happiness_score_df.drop('2005', axis = 1, inplace=True)
         happiness_score_df.isna().sum().sum()

Out[19]: 5

In [20]: happiness_score_df.isna().sum()
```

```
Out[20]: country    0
         2006       3
         2007       2
         2008       0
         2009       0
         2010       0
         2011       0
         2012       0
         2013       0
         2014       0
         2015       0
         2016       0
         2017       0
         2018       0
         2019       0
         dtype: int64
```

Now we've narrowed down the null values immensely to just 5 values! Let's drop the remaining rows now.

```
In [21]: happiness_score_df.dropna(inplace=True)
```

Now let's check for the shape of our datasets after trimming them down.

```
In [22]: life_exp_df.shape

Out[22]: (186, 302)

In [23]: income_df.shape

Out[23]: (195, 252)

In [24]: govt_health_spending_df.shape

Out[24]: (184, 17)

In [25]: happiness_score_df.shape

Out[25]: (160, 15)
```

3. Normalizing datatypes in income dataset How will we go about this? We could simply query for all values that have k in them, extract the number from it, then multiply it by 1000. Let's get to work.

```
In [26]: for c in income_df.columns[1:]:  # This is to exclude the country column
             if income_df.dtypes[c] == object:
                 income_df[c] = (income_df[c].str.extract('(\d+)').astype(float) * 1000).astype(
                 # Here we iterated over the columns and used a Regular Expression to extract th
                 # multiplied it by 1000, then converted it to int
```

```
In [27]: print_df(income_df)

Dataset Sample:
                country  1800  1801  1802  1803  1804  1805  1806  1807  1808  \
0           Afghanistan   674   674   674   674   674   674   674   674   674
1                Angola   691   693   697   700   702   705   709   712   716
2               Albania   746   746   746   746   746   747   747   747   747
3               Andorra  1340  1340  1340  1350  1350  1350  1350  1360  1360
4  United Arab Emirates  1120  1120  1120  1130  1130  1140  1140  1150  1150

   ...     2041     2042     2043     2044     2045     2046     2047  \
0  ...  2880000  2940000  3000000  3070000  3130000  3200000  3270000
1  ...  8040000  8220000  8390000  8570000  8750000  8940000  9120000
2  ...    24000    25000    25000    26000    26000    27000    27000
3  ...   108000   111000   113000   116000   118000   121000   123000
4  ...    74000    76000    77000    79000    81000    82000    84000

      2048     2049     2050
0  3340000  3410000  3480000
1  9320000  9520000  9720000
2    28000    28000    29000
3   126000   128000   131000
4    86000    88000    90000

[5 rows x 252 columns]

Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Columns: 252 entries, country to 2050
dtypes: int64(251), object(1)
memory usage: 384.0+ KB
None

Number of rows: 195
Number of columns: 252
------------------------------------------------------------
```

Now all of our year columns are of type int. Perfect!
Let's take a look at our cleaned datasets before moving on to the next step

```
In [28]: life_exp_df.head()

Out[28]:                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                Angola  27.0  27.0  27.0  27.0  27.0  27.0  27.0  27.0  27.0
         2               Albania  35.4  35.4  35.4  35.4  35.4  35.4  35.4  35.4  35.4
         4  United Arab Emirates  30.7  30.7  30.7  30.7  30.7  30.7  30.7  30.7  30.7
```

```
5             Argentina  33.2  33.2  33.2  33.2  33.2  33.2  33.2  33.2  33.2

   ...  2091  2092  2093  2094  2095  2096  2097  2098  2099  2100
0  ...  75.5  75.7  75.8  76.0  76.1  76.2  76.4  76.5  76.6  76.8
1  ...  78.8  79.0  79.1  79.2  79.3  79.5  79.6  79.7  79.9  80.0
2  ...  87.4  87.5  87.6  87.7  87.8  87.9  88.0  88.2  88.3  88.4
4  ...  82.4  82.5  82.6  82.7  82.8  82.9  83.0  83.1  83.2  83.3
5  ...  86.2  86.3  86.5  86.5  86.7  86.8  86.9  87.0  87.1  87.2

[5 rows x 302 columns]
```

In [29]:
```
income_df.head()
```

Out[29]:
```
                country  1800  1801  1802  1803  1804  1805  1806  1807  1808  \
0           Afghanistan   674   674   674   674   674   674   674   674   674
1                Angola   691   693   697   700   702   705   709   712   716
2               Albania   746   746   746   746   746   747   747   747   747
3               Andorra  1340  1340  1340  1350  1350  1350  1350  1360  1360
4  United Arab Emirates  1120  1120  1120  1130  1130  1140  1140  1150  1150

   ...     2041     2042     2043     2044     2045     2046     2047  \
0  ...  2880000  2940000  3000000  3070000  3130000  3200000  3270000
1  ...  8040000  8220000  8390000  8570000  8750000  8940000  9120000
2  ...    24000    25000    25000    26000    26000    27000    27000
3  ...   108000   111000   113000   116000   118000   121000   123000
4  ...    74000    76000    77000    79000    81000    82000    84000

      2048     2049     2050
0  3340000  3410000  3480000
1  9320000  9520000  9720000
2    28000    28000    29000
3   126000   128000   131000
4    86000    88000    90000

[5 rows x 252 columns]
```

In [30]:
```
govt_health_spending_df.head()
```

Out[30]:
```
                country   1995   1996   1997   1998   1999   2000   2001  \
1                Angola   5.00   2.68   3.57   3.15   1.76   3.26   6.06
2               Albania   5.26   6.34   6.47   6.10   7.18   7.03   7.24
3               Andorra  23.60  23.80  23.20  28.70  20.80  19.10  19.20
4  United Arab Emirates   8.09   7.13   8.76   8.00   8.01   7.64   7.73
5             Argentina  15.30  15.20  15.00  14.90  15.10  14.70  14.30

    2002   2003   2004   2005   2006   2007   2008   2009   2010
1   3.74   4.83   4.12   4.38   6.06   5.75   6.40  10.10   7.18
2   7.32   7.64   9.23   9.79   9.05   8.46   8.21   8.42   8.42
3  20.00  22.00  22.70  22.00  22.80  21.30  21.30  21.30  21.30
```

```
4   7.98   8.35   8.21   8.70   8.95   8.93   8.85   8.76   8.79
5  15.30  14.80  15.20  14.30  14.40  13.90  13.80  14.70  14.70
```

```
In [31]: happiness_score_df.head()
```

Out[31]:

	country	2006	2007	2008	2009	2010	2011	2012	2013	2014	\
3	United Arab Emirates	67.3	46.3	37.2	68.7	71.0	71.2	72.2	66.2	65.4	
4	Argentina	63.1	60.7	59.6	64.2	64.4	67.8	64.7	65.8	66.7	
5	Armenia	42.9	48.8	46.5	41.8	43.7	42.6	43.2	42.8	44.5	
6	Australia	42.9	72.9	72.5	41.8	74.5	74.1	72.0	73.6	72.9	
7	Austria	71.2	72.9	71.8	41.8	73.0	74.7	74.0	75.0	69.5	

	2015	2016	2017	2018	2019
3	65.7	68.3	70.4	66.0	67.9
4	67.0	64.3	60.4	57.9	59.7
5	43.5	43.3	42.9	50.6	46.8
6	73.1	72.5	72.6	71.8	72.2
7	70.8	70.5	72.9	74.0	72.9

4. Merging datasets Now for the last step of our data wrangling process, we want to transform our datasets to be able to extract information to analyze using visuals and statistics. For now, we can first melt our dataframes so that years is a column, and then merge on country and year. Additional slicing can be done later on.

```
In [32]: life_exp_melt = life_exp_df.melt(id_vars=['country'], var_name='year', value_name='life
         income_melt = income_df.melt(id_vars=['country'], var_name='year', value_name='income')
         govt_health_melt = govt_health_spending_df.melt(id_vars=['country'], var_name='year', v
         happiness_melt = happiness_score_df.melt(id_vars=['country'], var_name='year', value_na
```

Now that we've "melted" our dataframes, we now have new dataframes that have the columns country and year, as well as the variable for the dataframe: life expectancy, income, govt. spending, or happiness score.

Let's explore these new dataframes a little bit

```
In [33]: life_exp_melt.nunique()
```

```
Out[33]: country     186
         year        301
         life_exp    843
         dtype: int64
```

```
In [34]: income_melt.nunique()
```

```
Out[34]: country     195
         year        251
         income     2903
         dtype: int64
```

```
In [35]: govt_health_melt.nunique()
```

```
Out[35]: country                 184
         year                     16
         govt_health_spending    709
         dtype: int64

In [36]: happiness_melt.nunique()

Out[36]: country            160
         year                14
         happiness_score    443
         dtype: int64
```

Now let's finally merge all 4 dataframes into one neat dataframe

```
In [37]: merged_df = life_exp_melt.merge(income_melt, on=['country', 'year']) \
         .merge(govt_health_melt, on=['country', 'year']) \
         .merge(happiness_melt, on=['country', 'year'])

In [38]: merged_df.head()

Out[38]:                 country  year  life_exp   income  govt_health_spending  \
         0  United Arab Emirates  2006      69.5    86000                  8.95
         1             Argentina  2006      75.4    20000                 14.40
         2               Armenia  2006      73.1  8460000                  7.42
         3             Australia  2006      81.5    42000                 16.80
         4               Austria  2006      80.1    50000                 15.70

            happiness_score
         0             67.3
         1             63.1
         2             42.9
         3             42.9
         4             71.2

In [39]: merged_df.nunique()

Out[39]: country                149
         year                     5
         life_exp               282
         income                 352
         govt_health_spending   329
         happiness_score        299
         dtype: int64

In [40]: merged_df.shape

Out[40]: (745, 6)
```

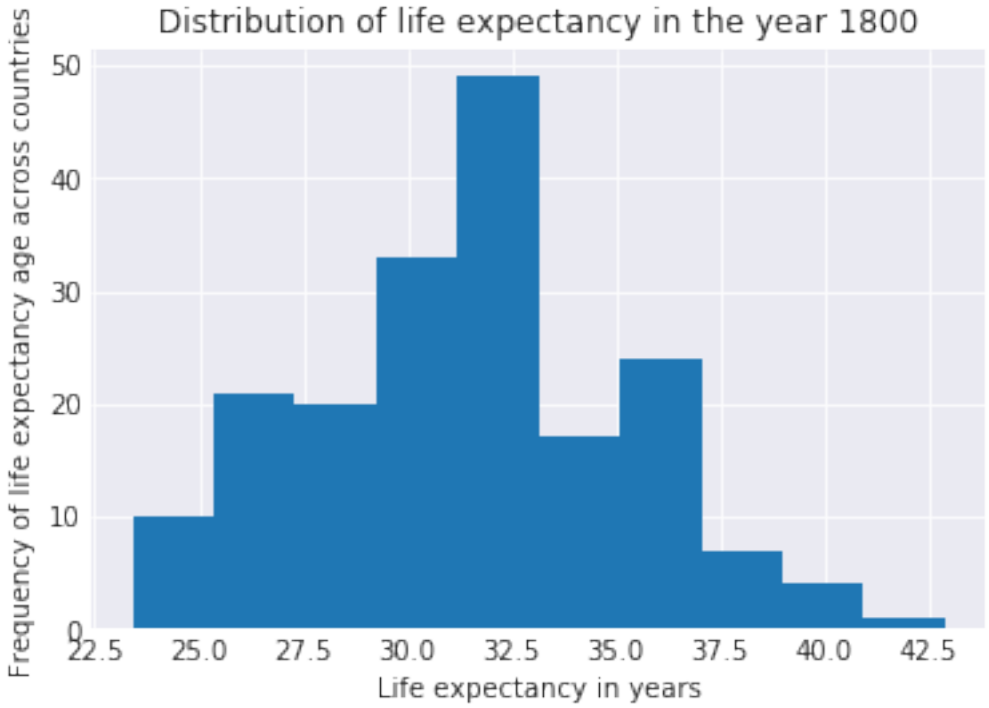
It seems that since we used inner join, only 149 countries and 5 years were common among all dataframes. The rest were dropped. That's fine, we now have more than enough data to compare different variables and how they relate to our questions.

Exploratory Data Analysis Let's summarize our investigation into 3 questions: 1. Is life expectancy increasing over time? 2. Which factors from our datasets affect life expectancy? 3. Are life expectancy and happiness correlated?

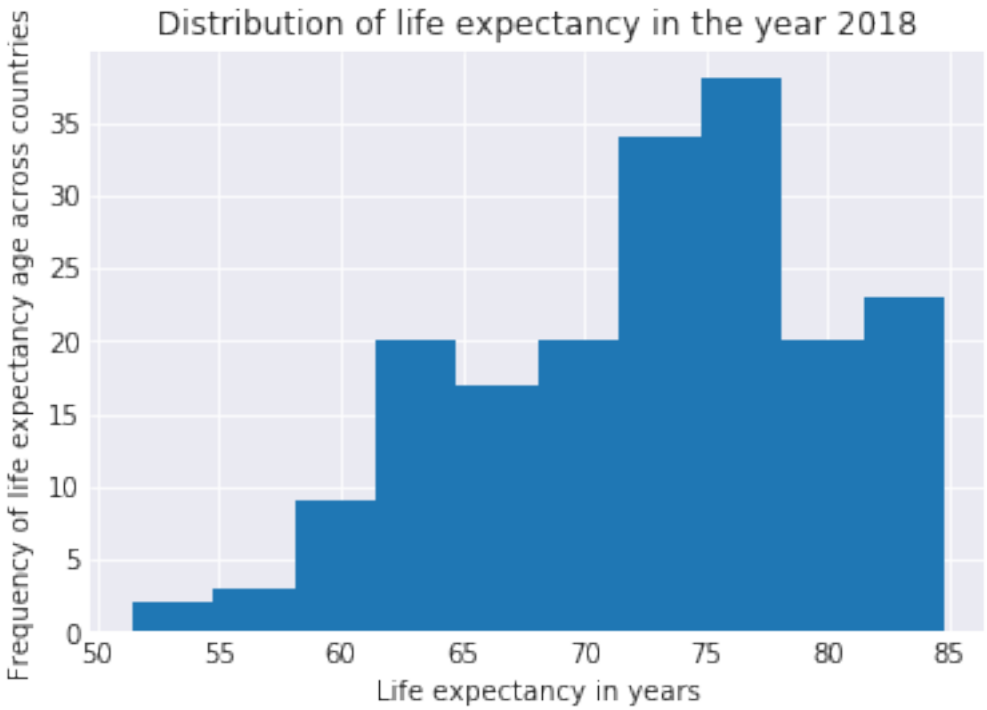
1.1.4 1. Is life expectancy increasing over time?

First lets examine a histogram of the distribution of life expectancy in the year 1800 and the year 2018.

```
In [41]: plt.figure();
         life_exp_df['1800'].hist();
         plt.xlabel('Life expectancy in years');
         plt.ylabel('Frequency of life expectancy age across countries');
         plt.title('Distribution of life expectancy in the year 1800');
```




```
In [42]: plt.figure();
         life_exp_df['2018'].hist();
         plt.xlabel('Life expectancy in years');
         plt.ylabel('Frequency of life expectancy age across countries');
         plt.title('Distribution of life expectancy in the year 2018');
```



We can notice that the data has become more left skewed over time. Good news for everyone alive in this era I guess.

Let's observe how the life expectancy has changed over time in more detail for each country using a line plot

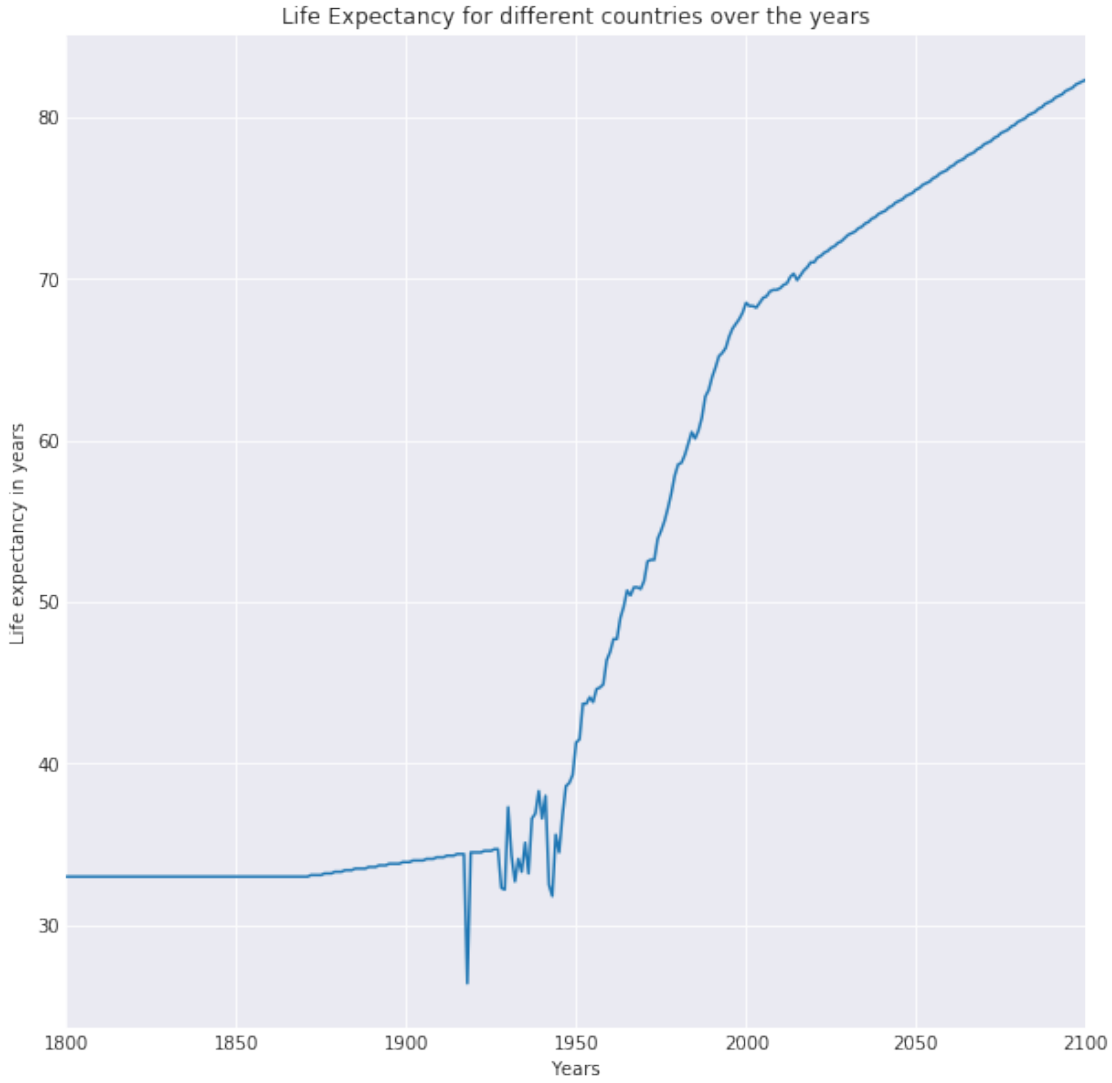
```
In [43]: transposed_df = life_exp_df.set_index('country').transpose()
         plt.figure()
         transposed_df.plot(figsize=(10,10));
         plt.xlabel('Years');
         plt.ylabel('Life expectancy in years');
         plt.title('Life Expectancy for different countries over the years')
         plt.legend(loc='upper right');
```

```
<matplotlib.figure.Figure at 0x7f5e8ca68cf8>
```


Life Expectancy for different countries over the years
Life expectancy in years
Years
0
20
40
60
80
1800
1850
1900
1950
2000
2050
2100
Afghanistan
Angola
Albania
United Arab Emirates
Argentina
Armenia
Antigua and Barbuda
Australia
Austria
Azerbaijan
Burundi
Belgium
Benin
Burkina Faso
Bangladesh
Bulgaria
Bahrain
Bahamas
Bosnia and Herzegovina
Belarus
Belize
Bolivia
Brazil
Barbados
Brunei
Bhutan
Botswana
Central African Republic
Canada
Switzerland
Chile
China
Cote d'Ivoire
Cameroon
Congo, Dem. Rep.
Congo, Rep.
Colombia
Comoros
Cape Verde
Costa Rica
Cuba
Cyprus
Czech Republic
Germany
Djibouti
Denmark
Dominican Republic
Algeria
Ecuador
Egypt
Eritrea
Spain
Estonia
Ethiopia
Finland
Fiji
France
Micronesia, Fed. Sts.
Gabon
United Kingdom
Georgia
Ghana
Guinea
Gambia
Guinea-Bissau
Equatorial Guinea
Greece
Grenada
Guatemala
Guyana
Hong Kong, China
Honduras
Croatia
Haiti
Hungary
Indonesia
India
Ireland
Iran
Iraq
Iceland
Israel
Italy
Jamaica
Jordan
Japan
Kazakhstan
Kenya
Kyrgyz Republic
Cambodia
Kiribati
South Korea
Kuwait
Lao
Lebanon
Liberia
Libya
St. Lucia
Sri Lanka
Lesotho
Lithuania
Luxembourg
Latvia
Morocco
Moldova
Madagascar
Maldives
Mexico
North Macedonia
Mali
Malta
Myanmar
Montenegro
Mongolia
Mozambique
Mauritania
Mauritius
Malawi
Malaysia
Namibia
Niger
Nigeria
Nicaragua
Netherlands
Norway
Nepal
New Zealand
Oman
Pakistan
Panama
Peru
Philippines
Papua New Guinea
Poland
North Korea
Portugal
Paraguay
Palestine
Qatar
Romania
Russia
Rwanda
Saudi Arabia
Sudan
Senegal
Singapore
Solomon Islands
Sierra Leone
El Salvador
Somalia
Serbia
South Sudan
Sao Tome and Principe
Suriname
Slovak Republic
Slovenia
Sweden
Eswatini
Seychelles
Syria
Chad
Togo
Thailand
Tajikistan
Turkmenistan
Timor-Leste
Tonga
Trinidad and Tobago
Tunisia
Turkey
Taiwan
Tanzania
Uganda
Ukraine
Uruguay
United States
Uzbekistan
St. Vincent and the Grenadines
Venezuela
Vietnam
Vanuatu
Samoa
Yemen
South Africa
Zambia
Zimbabwe

Oh this is a mess! Let's trim down the number of countries we're plotting. First, let's investigate one country, Egypt for example.

```
In [44]: transposed_df.Egypt.plot(figsize=(10,10));
         plt.xlabel('Years');
         plt.ylabel('Life expectancy in years');
         plt.title('Life Expectancy for different countries over the years');
```



It seems there were is a bit of noise in the data, but the general trend is that life expectancy does increase as time goes on. However, it is not a simple straight line. It seems the increase was a bit slow at first, then experienced a drastic rise, until it starts slowing down and plateaus in the future. Interstingly, this is also the predicted model for any fast growing population according to Wikipedia.

Now let's add more countries into the fray

```
In [45]: transposed_df[['China', 'United States', 'United Kingdom', 'Russia', 'Egypt', 'Australi
         plt.xlabel('Years');
         plt.ylabel('Life expectancy in years');
         plt.title('Life Expectancy for different countries over the years');
```

Despite the fluctuations, a clear trend can be observed. Neat!

1.1.5 2. Which factors from our datasets affect life expectancy?

Let's examine some statistics regarding our fully merged dataset

```
In [46]: merged_df.describe()
```

```
Out[46]:          life_exp        income  govt_health_spending  happiness_score
         count  745.000000  7.450000e+02            745.000000       745.000000
         mean    70.273154  2.060851e+06             11.243569        54.286443
         std      8.923396  2.889960e+06              4.441394        11.030859
         min     32.500000  1.000000e+04              0.911000        28.100000
         25%     64.400000  2.300000e+04              7.820000        45.900000
         50%     72.800000  6.300000e+04             11.300000        52.600000
         75%     77.200000  3.140000e+06             14.200000        62.400000
         max     83.300000  9.890000e+06             30.600000        79.700000
```

Then let's determine some statistics regarding the countries and life expectancy specifically.

```
In [47]: merged_df.groupby('country').describe()['life_exp']
```

```
Out[47]:             count   mean       std   min   25%   50%   75%   max
         country
         Algeria       5.0  73.90  0.474342  73.3  73.6  73.9  74.2  74.5
         Argentina     5.0  75.62  0.258844  75.3  75.4  75.7  75.8  75.9
         Armenia       5.0  73.52  0.286356  73.1  73.5  73.5  73.6  73.9
         Australia     5.0  81.74  0.260768  81.5  81.5  81.7  81.9  82.1
         Austria       5.0  80.44  0.260768  80.1  80.3  80.5  80.5  80.8
         ...           ...    ...       ...   ...   ...   ...   ...   ...
         Uzbekistan    5.0  65.74  0.618870  65.0  65.3  65.7  66.2  66.5
         Venezuela     5.0  74.74  0.343511  74.4  74.6  74.6  74.8  75.3
         Vietnam       5.0  73.02  0.130384  72.9  72.9  73.0  73.1  73.2
         Yemen         5.0  66.88  0.554076  66.2  66.5  66.9  67.2  67.6
         Zambia        5.0  53.64  2.688494  50.0  51.9  54.1  55.7  56.5

         [149 rows x 8 columns]
```

```
In [48]: max_exp = merged_df.life_exp.max()
         merged_df.query('life_exp == @max_exp')
```

```
Out[48]:     country  year  life_exp  income  govt_health_spending  happiness_score
         515   Japan  2009      83.3   36000                  18.4             58.4
         664   Japan  2010      83.3   37000                  18.4             60.6
```

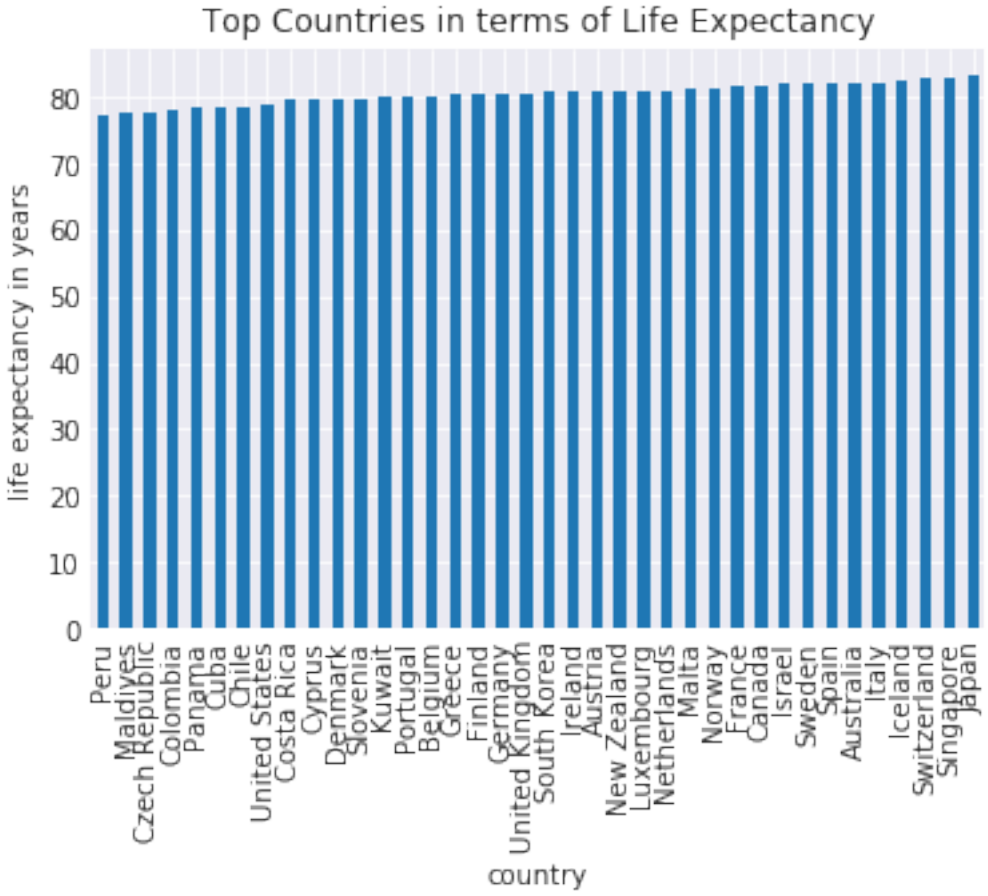
It seems Japan has the highest life expectancy out of all countries as of 2010! What about the lowest life expectancy?

```
In [49]: min_exp = merged_df.life_exp.min()
         merged_df.query('life_exp == @min_exp')
```

```
Out[49]:     country  year  life_exp   income  govt_health_spending  happiness_score
         653   Haiti  2010      32.5  2740000                  4.51             37.7
```

The lowest life expectancy goes to Haiti, with 32.5 years as of year 2010.
Next, let's plot a bar chart with some countries that have the highest life expectancies.

```
In [50]: # Filtering our data to find the countries with life expectancy that exceeded the 3rd q
         quantile_third_exp = merged_df.life_exp.quantile(0.75)
         piv_df = pd.pivot_table(merged_df, values='life_exp', index='country', columns='year')
         plt.figure();
         piv_df[piv_df['2010'] >= quantile_third_exp]['2010'].sort_values().plot.bar();  # We so
         plt.ylabel('life expectancy in years');
         plt.title('Top Countries in terms of Life Expectancy');
```



Let's see which country had the most percentage increase since the year 1800 as of the year 2018

```
In [51]: life_exp_df['percentage_increase'] = ((life_exp_df['2018'] - life_exp_df['1800'])/life_
         life_exp_df.head()
```

Out[51]:

	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	Angola	27.0	27.0	27.0	27.0	27.0	27.0	27.0	27.0	27.0	
2	Albania	35.4	35.4	35.4	35.4	35.4	35.4	35.4	35.4	35.4	
4	United Arab Emirates	30.7	30.7	30.7	30.7	30.7	30.7	30.7	30.7	30.7	
5	Argentina	33.2	33.2	33.2	33.2	33.2	33.2	33.2	33.2	33.2	

```
   ...  2092  2093  2094  2095  2096  2097  2098  2099  2100  \
0  ...  75.7  75.8  76.0  76.1  76.2  76.4  76.5  76.6  76.8
1  ...  79.0  79.1  79.2  79.3  79.5  79.6  79.7  79.9  80.0
2  ...  87.5  87.6  87.7  87.8  87.9  88.0  88.2  88.3  88.4
4  ...  82.5  82.6  82.7  82.8  82.9  83.0  83.1  83.2  83.3
5  ...  86.3  86.5  86.5  86.7  86.8  86.9  87.0  87.1  87.2

   percentage_increase
0           122.340426
1           139.259259
2           121.468927
4           140.065147
5           130.421687

[5 rows x 303 columns]
```

```
In [52]: max_inc = life_exp_df.percentage_increase.max()
         life_exp_df[life_exp_df['percentage_increase'] == max_inc].loc[:,['country','1800', '20
```

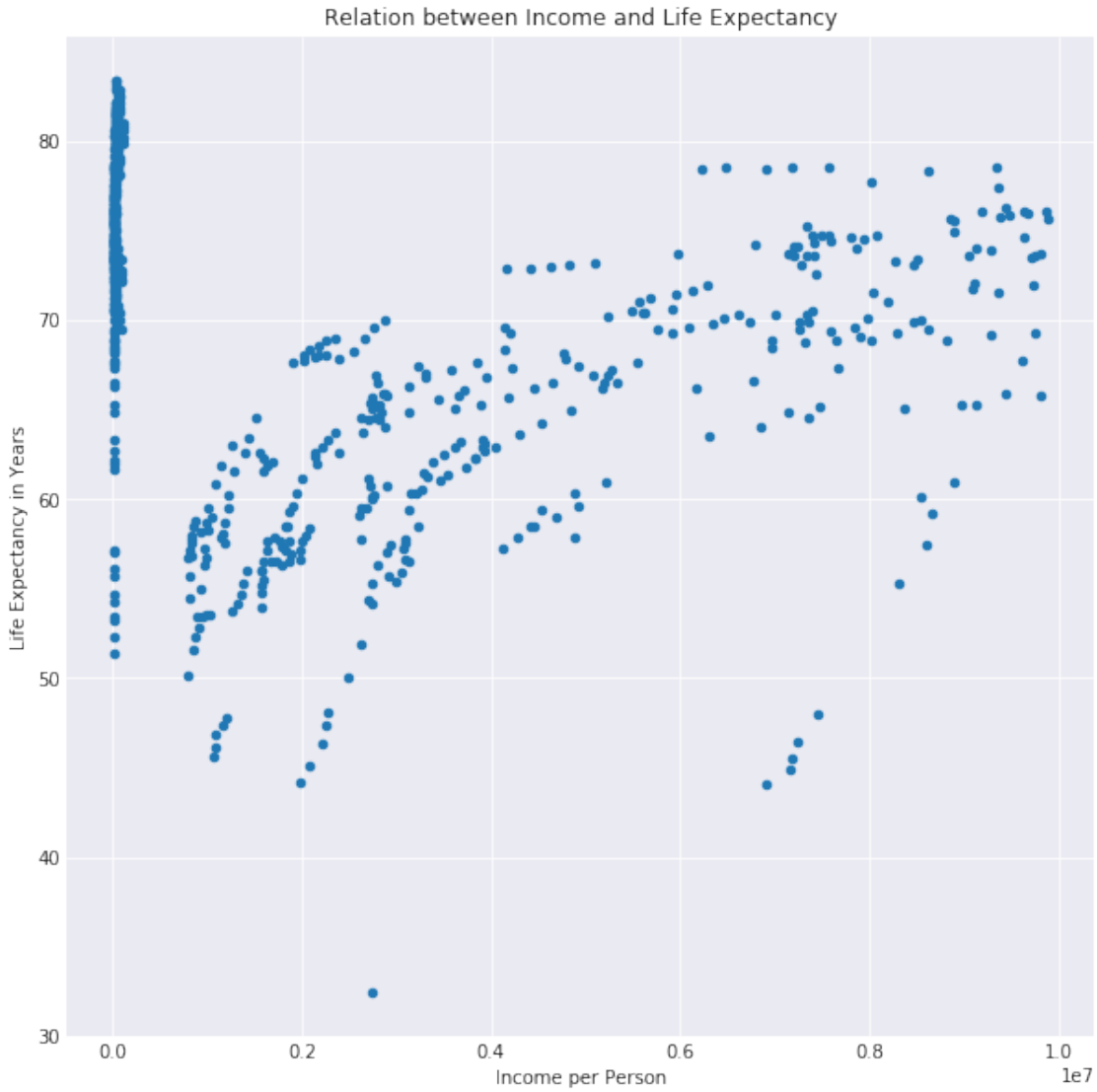
```
Out[52]:         country  1800  2018  percentage_increase
         94  South Korea  25.8  82.9           221.317829
```

South Korea has had the greatest percentage increase in life expectancy from the year 1800 by a whopping 221.3%!

Next, let's explore the relations between life expectancy and other variables. Let's start with income. Does income affect the life expectancy of a country?

```
In [53]: plt.figure();
         merged_df.plot(kind='scatter', x='income', y='life_exp', figsize=(10,10));
         plt.xlabel('Income per Person');
         plt.ylabel('Life Expectancy in Years');
         plt.title('Relation between Income and Life Expectancy');
```

```
<matplotlib.figure.Figure at 0x7f5e823047f0>
```



There does seem to be a postive correlation between the two variables! Let's explore the relationship between life expectancy and how much the government is spending on the health of its citizens (in proportion to its total expenditure).

```
In [54]: plt.figure();
         merged_df.plot(kind='scatter', x='govt_health_spending', y='life_exp', figsize=(10,10))
         plt.xlabel('Government Health Spending per Total Government Spending Percentage');
         plt.ylabel('Life Expectancy in Years');
         plt.title('Relation between Govt. Spending on Health and Life Expectancy');
```

```
<matplotlib.figure.Figure at 0x7f5e82304ef0>
```

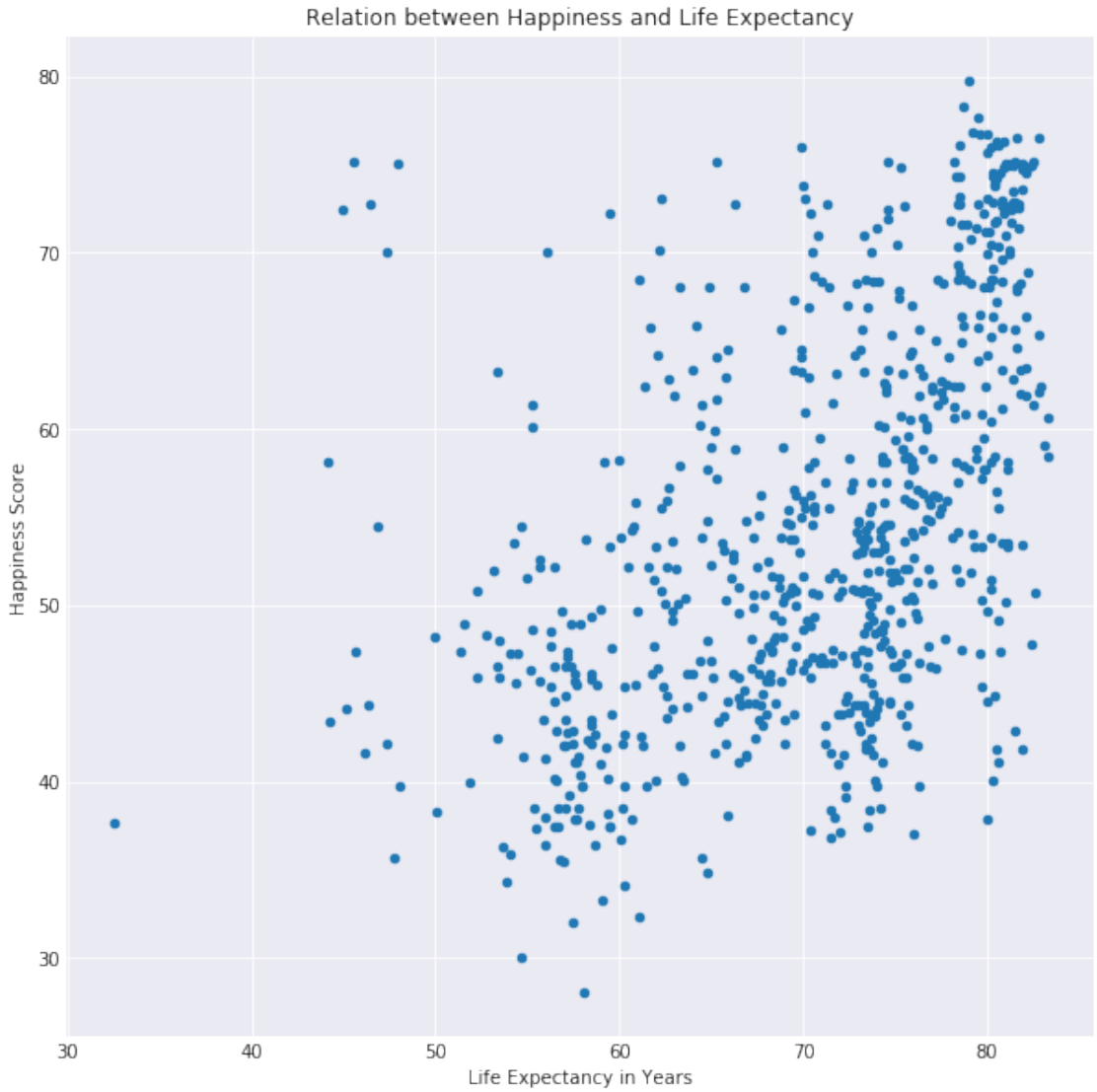
Hmm, it is difficult to say in this case. There seems to be a correlation, but it is very weak. We can probably overlook this and not consider this as a factor.

1.1.6 3. Are life expectancy and happiness correlated?

Do happier countries experience longevity? This is what we'll explore next

```
In [55]: plt.figure();
         merged_df.plot(kind='scatter', y='happiness_score', x='life_exp', figsize=(10,10));
         plt.ylabel('Happiness Score');
         plt.xlabel('Life Expectancy in Years');
         plt.title('Relation between Happiness and Life Expectancy');
```

```
<matplotlib.figure.Figure at 0x7f5e8440a668>
```



There seems to be a positive correlation, though it is not as strong as we had hoped. Do you think the secret to a long life could be being more happy with life?

Conclusions

We have explored the data from multiple angles and have determined the following: - Life expectancy seems to be increasing over time according to our data - Japan has the highest life expectancy as of the year 2010 and Haiti has the lowest - South Korea has had the greatest percentage increase in life expectancy between the years 1800 and 2018 - Income and life expectancy are positively correlated - Happiness and life expectancy are positively correlated as well

To investigate further, we can gather census data for each continent, or for each economic bracket, and use this grouping to examine life expectancy among different social levels as well. There are also many other variables that could be used to determine what affects life expectancy that could be gathered from Gapminder website, such as information on diseases such as cancer, number of children per mother, and infant mortality rates.

1.1.7 Limitations

The government expenditure on health per total expenditure data was probably not enough to give a solid overview on the state of the health sector in each country. More data could have been gathered regarding number of doctors in each country, number of hospitals, percentage of population with chronic diseases, and many more. Gapminder provides extensive data on health that could be used to further pursue this line of investigation.

Moreover, the happiness metric seemed to be almost qualitative in nature due to it relying on surveys. More information about the habits of each country could help gather more information about whether or not life expectancy is really caused by happiness, such as smoking and drinking habits, suicide rates, and so on.

For this assignment I used primarily the official documentation for the pandas, numpy, matplotlib, and seaborn libraries, and stackoverflow to search for solutions and suggestions whenever I was stuck.

```
In [56]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])
```

```
Out[56]: 0
```