World Happiness

The World Happiness Report provides an annual analysis of what factors contribute to people's well-being and happiness. They examine many features that range from country economics to self-reported polls. For this project, we would like to examine factors that contribute to happiness, much like the World Happiness Report.

Obtaining the Data

We obtained the rankings found in global polls about how people ranked their happiness and various aspects of their lives from 2015 to 2019. The data points are separated by country and region. We start by concatenating all the data together for easier cleaning and managing. We would also like to add another column to label the year of each data point.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats

# Configure Pandas display settings.
pd.options.display.max_rows = 15

# Import the data sets.
happy_list = []
years = [2015, 2016, 2017, 2018, 2019]
for year in years:
  happy_list.append(pd.read_csv(str(year) + '.csv'))

# Add a column to each dataset for the year.
for happy, year in zip(happy_list, years):
  happy['Year'] = year

happy15, happy16, happy17, happy18, happy19 = happy_list

# Concatenate all the data together.
happy = pd.concat(happy_list)

happy
```

```
         Country          Region  Happiness Rank  Happiness Score  \
0    Switzerland  Western Europe             1.0            7.587
1        Iceland  Western Europe             2.0            7.561
2        Denmark  Western Europe             3.0            7.527
3         Norway  Western Europe             4.0            7.522
4         Canada   North America             5.0            7.427
..           ...             ...             ...              ...
151          NaN             NaN             NaN              NaN
```

```
152          NaN             NaN             NaN              NaN
153          NaN             NaN             NaN              NaN
154          NaN             NaN             NaN              NaN
155          NaN             NaN             NaN              NaN

     Standard Error  Economy (GDP per Capita)   Family  \
0           0.03411                   1.39651  1.34951
1           0.04884                   1.30232  1.40223
2           0.03328                   1.32548  1.36058
3           0.03880                   1.45900  1.33095
4           0.03553                   1.32629  1.32261
..              ...                       ...      ...
151             NaN                       NaN      NaN
152             NaN                       NaN      NaN
153             NaN                       NaN      NaN
154             NaN                       NaN      NaN
155             NaN                       NaN      NaN

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
...  \
0                     0.94143  0.66557                         0.41978
...
1                     0.94784  0.62877                         0.14145
...
2                     0.87464  0.64938                         0.48357
...
3                     0.88521  0.66973                         0.36503
...
4                     0.90563  0.63297                         0.32957
...
..                        ...      ...                             ...
...
151                       NaN      NaN                             NaN
...
152                       NaN      NaN                             NaN
...
153                       NaN      NaN                             NaN
...
154                       NaN      NaN                             NaN
...
155                       NaN      NaN                             NaN
...

     Trust..Government.Corruption.  Dystopia.Residual  Overall rank  \
0                              NaN                NaN           NaN
1                              NaN                NaN           NaN
2                              NaN                NaN           NaN
3                              NaN                NaN           NaN
4                              NaN                NaN           NaN
..                             ...                ...           ...
```

```
151                           NaN                NaN         152.0
152                           NaN                NaN         153.0
153                           NaN                NaN         154.0
154                           NaN                NaN         155.0
155                           NaN                NaN         156.0

            Country or region  Score  GDP per capita  Social
support  \
0                         NaN    NaN             NaN             NaN

1                         NaN    NaN             NaN             NaN

2                         NaN    NaN             NaN             NaN

3                         NaN    NaN             NaN             NaN

4                         NaN    NaN             NaN             NaN

..                        ...    ...             ...             ...

151                    Rwanda  3.334           0.359           0.711

152                  Tanzania  3.231           0.476           0.885

153               Afghanistan  3.203           0.350           0.517

154  Central African Republic  3.083           0.026           0.000

155               South Sudan  2.853           0.306           0.575


     Healthy life expectancy  Freedom to make life choices  \
0                        NaN                           NaN
1                        NaN                           NaN
2                        NaN                           NaN
3                        NaN                           NaN
4                        NaN                           NaN
..                       ...                           ...
151                    0.614                         0.555
152                    0.499                         0.417
153                    0.361                         0.000
154                    0.105                         0.225
155                    0.295                         0.010

     Perceptions of corruption
0                          NaN
1                          NaN
2                          NaN
3                          NaN
```

```
4                          NaN
..                         ...
151                      0.411
152                      0.147
153                      0.025
154                      0.035
155                      0.091

[782 rows x 31 columns]
```

Cleaning the Data

Before we can start working with the data, we must ensure that it is properly combined, cleaned, and free of errors.

Name Unification

The concatenation did not properly occur because it generated large columns with no values. With a quick review, it is apparent that the naming conventions of the columns have changed. To fix this, we must normalize all columns between the years.

Our first step is to ensure that the columns for each year correspond to each other. For instance, the data for 2017 are not separated by whitespace. Instead, it is separated by periods. We simply rename each column for 2017.

```
happy17.rename(columns={
    'Happiness.Score': 'Happiness Score',
    'Happiness.Rank': 'Happiness Rank',
    'Whisker.low': 'Lower Confidence Interval',
    'Whisker.high': 'Upper Confidence Interval',
    'Economy..GDP.per.Capita.': 'Economy (GDP per Capita)',
    'Health..Life.Expectancy.': 'Health (Life Expectancy)',
    'Trust..Government.Corruption.': 'Trust (Government Corruption)',
    'Dystopia.Residual': 'Dystopia Residual'
}, inplace=True)

happy17
```

```
                       Country  Happiness Rank  Happiness Score  \
0                       Norway               1            7.537
1                      Denmark               2            7.522
2                      Iceland               3            7.504
3                  Switzerland               4            7.494
4                      Finland               5            7.469
..                         ...             ...              ...
150                     Rwanda             151            3.471
151                      Syria             152            3.462
152                   Tanzania             153            3.349
153                    Burundi             154            2.905
154  Central African Republic             155            2.693
```

```
     Upper Confidence Interval  Lower Confidence Interval  \
0                     7.594445                   7.479556
1                     7.581728                   7.462272
2                     7.622030                   7.385970
3                     7.561772                   7.426227
4                     7.527542                   7.410458
..                         ...                        ...
150                   3.543030                   3.398970
151                   3.663669                   3.260331
152                   3.461430                   3.236570
153                   3.074690                   2.735310
154                   2.864884                   2.521116

     Economy (GDP per Capita)    Family  Health (Life Expectancy)
Freedom  \
0                    1.616463  1.533524                  0.796667
0.635423
1                    1.482383  1.551122                  0.792566
0.626007
2                    1.480633  1.610574                  0.833552
0.627163
3                    1.564980  1.516912                  0.858131
0.620071
4                    1.443572  1.540247                  0.809158
0.617951
..                        ...       ...                       ...
...
150                  0.368746  0.945707                  0.326425
0.581844
151                  0.777153  0.396103                  0.500533
0.081539
152                  0.511136  1.041990                  0.364509
0.390018
153                  0.091623  0.629794                  0.151611
0.059901
154                  0.000000  0.000000                  0.018773
0.270842

     Generosity  Trust (Government Corruption)  Dystopia Residual
Year
0      0.362012                       0.315964           2.277027
2017
1      0.355280                       0.400770           2.313707
2017
2      0.475540                       0.153527           2.322715
2017
3      0.290549                       0.367007           2.276716
2017
4      0.245483                       0.382612           2.430182
```

```
2017
..          ...                          ...                ...   ..
.
150    0.252756                     0.455220           0.540061
2017
151    0.493664                     0.151347           1.061574
2017
152    0.354256                     0.066035           0.621130
2017
153    0.204435                     0.084148           1.683024
2017
154    0.280876                     0.056565           2.066005
2017

[155 rows x 13 columns]
```

Furthermore, the data for 2018 and 2019 contain the same information but follow a different naming scheme than the previous year. We also need to rename each of these columns.

```
for happy in [happy18, happy19]:
    happy.rename(columns={
        'Overall rank': 'Happiness Rank',
        'Country or region': 'Country',
        'Score': 'Happiness Score',
        'GDP per capita': 'Economy (GDP per Capita)',
        'Social support': 'Family',
        'Healthy life expectancy': 'Health (Life Expectancy)',
        'Freedom to make life choices': 'Freedom',
        'Perceptions of corruption': 'Trust (Government Corruption)'
    }, inplace=True)
```

```
happy18
```

```
     Happiness Rank                  Country  Happiness Score  \
0                 1                  Finland            7.632
1                 2                   Norway            7.594
2                 3                  Denmark            7.555
3                 4                  Iceland            7.495
4                 5              Switzerland            7.487
..              ...                      ...              ...
151             152                    Yemen            3.355
152             153                 Tanzania            3.303
153             154              South Sudan            3.254
154             155  Central African Republic            3.083
155             156                  Burundi            2.905

     Economy (GDP per Capita)  Family  Health (Life Expectancy)
Freedom  \
0                       1.305   1.592                     0.874
```

```
0.681
1                      1.456   1.582                      0.861
0.686
2                      1.351   1.590                      0.868
0.683
3                      1.343   1.644                      0.914
0.677
4                      1.420   1.549                      0.927
0.660
..                       ...     ...                        ...      .
..
151                    0.442   1.073                      0.343
0.244
152                    0.455   0.991                      0.381
0.481
153                    0.337   0.608                      0.177
0.112
154                    0.024   0.000                      0.010
0.305
155                    0.091   0.627                      0.145
0.065

     Generosity  Trust (Government Corruption)  Year
0         0.202                          0.393  2018
1         0.286                          0.340  2018
2         0.284                          0.408  2018
3         0.353                          0.138  2018
4         0.256                          0.357  2018
..          ...                            ...   ...
151       0.083                          0.064  2018
152       0.270                          0.097  2018
153       0.224                          0.106  2018
154       0.218                          0.038  2018
155       0.149                          0.076  2018

[156 rows x 10 columns]
```

Next, between the data sets, some contain a column for standard error in the happiness score. Others, instead, have the upper and lower confidence intervals. We can combine the two columns by aggregating the upper and lower confidence levels. We create a new column for the standard error and calculate it as the difference between the upper confidence interval and the happiness score.

```
for happy in [happy16, happy17]:
    happy['Standard Error'] = happy['Upper Confidence Interval'] - \
                              happy['Happiness Score']
    happy.drop(columns=['Upper Confidence Interval',
                        'Lower Confidence Interval'], inplace=True)

happy16
```

```
        Country                          Region  Happiness Rank  \
0       Denmark                  Western Europe               1
1   Switzerland                  Western Europe               2
2       Iceland                  Western Europe               3
3        Norway                  Western Europe               4
4       Finland                  Western Europe               5
..          ...                             ...             ...
152       Benin              Sub-Saharan Africa             153
153 Afghanistan                   Southern Asia             154
154        Togo              Sub-Saharan Africa             155
155       Syria  Middle East and Northern Africa             156
156     Burundi              Sub-Saharan Africa             157

     Happiness Score  Economy (GDP per Capita)   Family  \
0              7.526                   1.44178  1.16374
1              7.509                   1.52733  1.14524
2              7.501                   1.42666  1.18326
3              7.498                   1.57744  1.12690
4              7.413                   1.40598  1.13464
..               ...                       ...      ...
152            3.484                   0.39499  0.10419
153            3.360                   0.38227  0.11037
154            3.303                   0.28123  0.00000
155            3.069                   0.74719  0.14866
156            2.905                   0.06831  0.23442

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.79504  0.57941                        0.44453

1                     0.86303  0.58557                        0.41203

2                     0.86733  0.56624                        0.14975

3                     0.79579  0.59609                        0.35776

4                     0.81091  0.57104                        0.41004

..                        ...      ...                            ...

152                   0.21028  0.39747                        0.06681

153                   0.17344  0.16430                        0.07112

154                   0.24811  0.34678                        0.11587

155                   0.62994  0.06912                        0.17233

156                   0.15747  0.04320                        0.09419
```

```
     Generosity  Dystopia Residual  Year  Standard Error
0       0.36171            2.73939  2016           0.066
1       0.28083            2.69463  2016           0.081
2       0.47678            2.83137  2016           0.168
3       0.37895            2.66465  2016           0.077
4       0.25492            2.82596  2016           0.062
..          ...                ...   ...             ...
152     0.20180            2.10812  2016           0.080
153     0.31268            2.14558  2016           0.072
154     0.17517            2.13540  2016           0.111
155     0.48397            0.81789  2016           0.133
156     0.20290            2.10404  2016           0.173

[157 rows x 13 columns]
```

Missing Values

After renaming the columns, we can check for any missing values in the combined data set. We start by recombining the edited data columns into one data frame.

```
happy = pd.concat(happy_list)
happy.reset_index(drop=True, inplace=True)
happy
```

```
                     Country          Region  Happiness Rank  \
0                Switzerland  Western Europe               1
1                    Iceland  Western Europe               2
2                    Denmark  Western Europe               3
3                     Norway  Western Europe               4
4                     Canada   North America               5
..                       ...             ...             ...
777                  Rwanda             NaN             152
778                Tanzania             NaN             153
779             Afghanistan             NaN             154
780  Central African Republic             NaN             155
781             South Sudan             NaN             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.34951
1              7.561         0.04884                   1.30232
1.40223
2              7.527         0.03328                   1.32548
1.36058
3              7.522         0.03880                   1.45900
1.33095
4              7.427         0.03553                   1.32629
```

```
1.32261
..               ...             ...                        ...      ..
.
777            3.334             NaN                    0.35900
0.71100
778            3.231             NaN                    0.47600
0.88500
779            3.203             NaN                    0.35000
0.51700
780            3.083             NaN                    0.02600
0.00000
781            2.853             NaN                    0.30600
0.57500

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.41978

1                     0.94784  0.62877                        0.14145

2                     0.87464  0.64938                        0.48357

3                     0.88521  0.66973                        0.36503

4                     0.90563  0.63297                        0.32957

..                        ...      ...                            ...

777                   0.61400  0.55500                        0.41100

778                   0.49900  0.41700                        0.14700

779                   0.36100  0.00000                        0.02500

780                   0.10500  0.22500                        0.03500

781                   0.29500  0.01000                        0.09100


     Generosity  Dystopia Residual  Year
0       0.29678            2.51738  2015
1       0.43630            2.70201  2015
2       0.34139            2.49204  2015
3       0.34699            2.46531  2015
4       0.45811            2.45176  2015
..          ...                ...   ...
777     0.21700                NaN  2019
778     0.27600                NaN  2019
779     0.15800                NaN  2019
```

```
780     0.23500                NaN  2019
781     0.20200                NaN  2019

[782 rows x 13 columns]
```

We now summarize any missing values that are in the data frame.

```
happy.isnull().sum()
```

```
Country                             0
Region                            467
Happiness Rank                      0
Happiness Score                     0
Standard Error                    312
Economy (GDP per Capita)            0
Family                              0
Health (Life Expectancy)            0
Freedom                             0
Trust (Government Corruption)       1
Generosity                          0
Dystopia Residual                 312
Year                                0
dtype: int64
```

```
happy[happy['Region'].isnull() |
      happy['Standard Error'].isnull() |
      happy['Trust (Government Corruption)'].isnull() |
      happy['Dystopia Residual'].isnull()]
```

```
                     Country Region  Happiness Rank  Happiness Score
\
315                   Norway    NaN               1            7.537

316                  Denmark    NaN               2            7.522

317                  Iceland    NaN               3            7.504

318              Switzerland    NaN               4            7.494

319                  Finland    NaN               5            7.469

..                       ...    ...             ...              ...

777                   Rwanda    NaN             152            3.334

778                 Tanzania    NaN             153            3.231

779              Afghanistan    NaN             154            3.203

780  Central African Republic    NaN             155            3.083
```

```
781               South Sudan    NaN             156           2.853


     Standard Error  Economy (GDP per Capita)    Family  \
315        0.057445                  1.616463  1.533524
316        0.059728                  1.482383  1.551122
317        0.118030                  1.480633  1.610574
318        0.067772                  1.564980  1.516912
319        0.058542                  1.443572  1.540247
..              ...                       ...       ...
777             NaN                  0.359000  0.711000
778             NaN                  0.476000  0.885000
779             NaN                  0.350000  0.517000
780             NaN                  0.026000  0.000000
781             NaN                  0.306000  0.575000

     Health (Life Expectancy)   Freedom  Trust (Government Corruption)
\
315                  0.796667  0.635423                        0.315964

316                  0.792566  0.626007                        0.400770

317                  0.833552  0.627163                        0.153527

318                  0.858131  0.620071                        0.367007

319                  0.809158  0.617951                        0.382612

..                        ...       ...                             ...

777                  0.614000  0.555000                        0.411000

778                  0.499000  0.417000                        0.147000

779                  0.361000  0.000000                        0.025000

780                  0.105000  0.225000                        0.035000

781                  0.295000  0.010000                        0.091000


     Generosity  Dystopia Residual  Year
315    0.362012           2.277027  2017
316    0.355280           2.313707  2017
317    0.475540           2.322715  2017
318    0.290549           2.276716  2017
319    0.245483           2.430182  2017
..          ...                ...   ...
```

```
777    0.217000                NaN  2019
778    0.276000                NaN  2019
779    0.158000                NaN  2019
780    0.235000                NaN  2019
781    0.202000                NaN  2019

[467 rows x 13 columns]
```

Upon closer inspection, there are four columns with missing values: region, standard error, trust, and dystopia residual.

Missing Regions

The missing values for the region seem to be missing not at random (MNAR) because there are some years where the data was not included. Each country corresponds to a larger area from another year's data set. We can simply map the missing values because there are no missing values from the country field.

```
# Create a dictionary mapping country to region from 2015 and 2016.
regions = dict(zip(happy15['Country'], happy15['Region']))
regions.update(dict(zip(happy16['Country'], happy16['Region'])))

# Apply mapping to each missing region.
happy['Region'].fillna(happy['Country'].map(regions), inplace=True)
happy
```

```
                     Country             Region  Happiness Rank  \
0                Switzerland      Western Europe               1
1                    Iceland      Western Europe               2
2                    Denmark      Western Europe               3
3                     Norway      Western Europe               4
4                     Canada       North America               5
..                       ...                 ...             ...
777                  Rwanda  Sub-Saharan Africa             152
778                Tanzania  Sub-Saharan Africa             153
779             Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781             South Sudan  Sub-Saharan Africa             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.34951
1              7.561         0.04884                   1.30232
1.40223
2              7.527         0.03328                   1.32548
1.36058
3              7.522         0.03880                   1.45900
1.33095
4              7.427         0.03553                   1.32629
```

```
1.32261
..               ...             ...                        ...      ..
.
777            3.334             NaN                    0.35900
0.71100
778            3.231             NaN                    0.47600
0.88500
779            3.203             NaN                    0.35000
0.51700
780            3.083             NaN                    0.02600
0.00000
781            2.853             NaN                    0.30600
0.57500

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.41978

1                     0.94784  0.62877                        0.14145

2                     0.87464  0.64938                        0.48357

3                     0.88521  0.66973                        0.36503

4                     0.90563  0.63297                        0.32957

..                        ...      ...                            ...

777                   0.61400  0.55500                        0.41100

778                   0.49900  0.41700                        0.14700

779                   0.36100  0.00000                        0.02500

780                   0.10500  0.22500                        0.03500

781                   0.29500  0.01000                        0.09100


     Generosity  Dystopia Residual  Year
0       0.29678            2.51738  2015
1       0.43630            2.70201  2015
2       0.34139            2.49204  2015
3       0.34699            2.46531  2015
4       0.45811            2.45176  2015
..          ...                ...   ...
777     0.21700                NaN  2019
778     0.27600                NaN  2019
779     0.15800                NaN  2019
```

```
780     0.23500                NaN  2019
781     0.20200                NaN  2019

[782 rows x 13 columns]
```

```
happy[happy['Region'].isnull()]
```

```
                     Country Region  Happiness Rank  Happiness Score
\
347  Taiwan Province of China    NaN              33            6.422

385   Hong Kong S.A.R., China    NaN              71            5.472

507         Trinidad & Tobago    NaN              38            6.192

527           Northern Cyprus    NaN              58            5.835

664         Trinidad & Tobago    NaN              39            6.192

689           Northern Cyprus    NaN              64            5.718

709           North Macedonia    NaN              84            5.274

745                    Gambia    NaN             120            4.516


     Standard Error  Economy (GDP per Capita)    Family  \
347        0.072596                  1.433627  1.384565
385        0.077594                  1.551675  1.262791
507             NaN                  1.223000  1.492000
527             NaN                  1.229000  1.211000
664             NaN                  1.231000  1.477000
689             NaN                  1.263000  1.252000
709             NaN                  0.983000  1.294000
745             NaN                  0.308000  0.939000

     Health (Life Expectancy)   Freedom  Trust (Government Corruption)
\
347                  0.793984  0.361467                       0.063829

385                  0.943062  0.490969                       0.293934

507                  0.564000  0.575000                       0.019000

527                  0.909000  0.495000                       0.154000

664                  0.713000  0.489000                       0.016000

689                  1.042000  0.417000                       0.162000
```

```
709                  0.838000  0.345000                          0.034000

745                  0.428000  0.382000                          0.167000


     Generosity  Dystopia Residual  Year
347    0.258360           2.126607  2017
385    0.374466           0.554633  2017
507    0.171000                NaN  2018
527    0.179000                NaN  2018
664    0.185000                NaN  2019
689    0.191000                NaN  2019
709    0.185000                NaN  2019
745    0.269000                NaN  2019
```

When reexamining the count of missing region values, eight remain empty. These can be attributed to a break in naming conventions. We can simply unify the country names or fill each region in manually to complete the missing values.

```
# Unify country names.
happy.at[347, 'Country'] = 'Taiwan'
happy.at[385, 'Country'] = 'Hong Kong'
happy.at[507, 'Country'] = 'Trinidad and Tobago'
happy.at[527, 'Country'] = 'North Cyprus'
happy.at[664, 'Country'] = 'Trinidad and Tobago'
happy.at[689, 'Country'] = 'North Cyprus'
happy.at[709, 'Country'] = 'Macedonia'

# Fill region by mapping.
happy['Region'].fillna(happy['Country'].map(regions), inplace=True)

# Set the region for countries not in the mapping.
happy.at[745, 'Region'] = 'Sub-Saharan Africa'

happy
```

```
                     Country             Region  Happiness Rank  \
0                Switzerland     Western Europe               1
1                    Iceland     Western Europe               2
2                    Denmark     Western Europe               3
3                     Norway     Western Europe               4
4                     Canada      North America               5
..                       ...                ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156
```

```
     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.34951
1              7.561         0.04884                   1.30232
1.40223
2              7.527         0.03328                   1.32548
1.36058
3              7.522         0.03880                   1.45900
1.33095
4              7.427         0.03553                   1.32629
1.32261
..               ...             ...                       ...      ..
.
777            3.334             NaN                   0.35900
0.71100
778            3.231             NaN                   0.47600
0.88500
779            3.203             NaN                   0.35000
0.51700
780            3.083             NaN                   0.02600
0.00000
781            2.853             NaN                   0.30600
0.57500

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.41978

1                     0.94784  0.62877                        0.14145

2                     0.87464  0.64938                        0.48357

3                     0.88521  0.66973                        0.36503

4                     0.90563  0.63297                        0.32957

..                        ...      ...                            ...

777                   0.61400  0.55500                        0.41100

778                   0.49900  0.41700                        0.14700

779                   0.36100  0.00000                        0.02500

780                   0.10500  0.22500                        0.03500

781                   0.29500  0.01000                        0.09100
```

```
     Generosity  Dystopia Residual  Year
0       0.29678            2.51738  2015
1       0.43630            2.70201  2015
2       0.34139            2.49204  2015
3       0.34699            2.46531  2015
4       0.45811            2.45176  2015
..          ...                ...   ...
777     0.21700                NaN  2019
778     0.27600                NaN  2019
779     0.15800                NaN  2019
780     0.23500                NaN  2019
781     0.20200                NaN  2019

[782 rows x 13 columns]
```

Missing Standard Error

There are also missing values for the standard error. The types of missing values is also MNAR because some of the more recent years omitted a range of errors for the happiness scores. We can simply fill these in with zeros to indicate no error range. It will allow us to keep the standard error for the older years.

```
happy['Standard Error'].fillna(0.0, inplace=True)
happy
```

```
                      Country             Region  Happiness Rank  \
0                 Switzerland      Western Europe               1
1                     Iceland      Western Europe               2
2                     Denmark      Western Europe               3
3                      Norway      Western Europe               4
4                      Canada       North America               5
..                        ...                 ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.34951
1              7.561         0.04884                   1.30232
1.40223
2              7.527         0.03328                   1.32548
1.36058
3              7.522         0.03880                   1.45900
1.33095
4              7.427         0.03553                   1.32629
```

```
1.32261
..                ...             ...                         ...      ..
.
777            3.334         0.00000                     0.35900
0.71100
778            3.231         0.00000                     0.47600
0.88500
779            3.203         0.00000                     0.35000
0.51700
780            3.083         0.00000                     0.02600
0.00000
781            2.853         0.00000                     0.30600
0.57500

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                         0.41978

1                     0.94784  0.62877                         0.14145

2                     0.87464  0.64938                         0.48357

3                     0.88521  0.66973                         0.36503

4                     0.90563  0.63297                         0.32957

..                        ...      ...                             ...

777                   0.61400  0.55500                         0.41100

778                   0.49900  0.41700                         0.14700

779                   0.36100  0.00000                         0.02500

780                   0.10500  0.22500                         0.03500

781                   0.29500  0.01000                         0.09100


     Generosity  Dystopia Residual  Year
0       0.29678            2.51738  2015
1       0.43630            2.70201  2015
2       0.34139            2.49204  2015
3       0.34699            2.46531  2015
4       0.45811            2.45176  2015
..          ...                ...   ...
777     0.21700               NaN  2019
778     0.27600               NaN  2019
779     0.15800               NaN  2019
```

```
780     0.23500                NaN  2019
781     0.20200                NaN  2019

[782 rows x 13 columns]
```

Missing Trust Value

It should be noted that there is one value missing from the trust column. Seemingly, the value is missing completely at random (MCAR) because there is no indication of why it is not there. We would like to perform a mean imputation to fill in the value. The missing trust rank would be replaced with the average of the trust ranks of the United Arab Emirates from the other years.

```
happy_uae = happy[happy['Country'] == 'United Arab Emirates']
mean_trust_uae = np.mean(happy_uae['Trust (Government Corruption)'])

happy.loc[489, 'Trust (Government Corruption)'] = mean_trust_uae
mean_trust_uae
```

```
0.3119823909258842
```

Missing Dystopia Residuals

Lastly, many values are missing from the dystopia residual field. As with many of the missing parameters before, it is MNAR because recent years elected to omit the score. In this case, we would like to just drop the variable from our data set because it does not serve a great purpose.

```
happy.drop(columns=['Dystopia Residual'], inplace=True)

happy
```

```
                     Country             Region  Happiness Rank  \
0                Switzerland     Western Europe               1
1                    Iceland     Western Europe               2
2                    Denmark     Western Europe               3
3                     Norway     Western Europe               4
4                     Canada      North America               5
..                       ...                ...             ...
777                  Rwanda  Sub-Saharan Africa             152
778                Tanzania  Sub-Saharan Africa             153
779             Afghanistan      Southern Asia              154
780  Central African Republic  Sub-Saharan Africa             155
781             South Sudan  Sub-Saharan Africa              156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.34951
1              7.561         0.04884                   1.30232
1.40223
```

```
2              7.527         0.03328                   1.32548
1.36058
3              7.522         0.03880                   1.45900
1.33095
4              7.427         0.03553                   1.32629
1.32261
..               ...             ...                       ...      ..
.
777            3.334         0.00000                   0.35900
0.71100
778            3.231         0.00000                   0.47600
0.88500
779            3.203         0.00000                   0.35000
0.51700
780            3.083         0.00000                   0.02600
0.00000
781            2.853         0.00000                   0.30600
0.57500

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.41978

1                     0.94784  0.62877                        0.14145

2                     0.87464  0.64938                        0.48357

3                     0.88521  0.66973                        0.36503

4                     0.90563  0.63297                        0.32957

..                        ...      ...                            ...

777                   0.61400  0.55500                        0.41100

778                   0.49900  0.41700                        0.14700

779                   0.36100  0.00000                        0.02500

780                   0.10500  0.22500                        0.03500

781                   0.29500  0.01000                        0.09100


     Generosity  Year
0       0.29678  2015
1       0.43630  2015
2       0.34139  2015
3       0.34699  2015
```

```
4       0.45811  2015
..          ...   ...
777     0.21700  2019
778     0.27600  2019
779     0.15800  2019
780     0.23500  2019
781     0.20200  2019

[782 rows x 12 columns]
```

Outlying Values

After filling in the missing values, we must find outlying values in the data set. Outlying values should be marked and imputed to generate more accurate models. We should check each score: economy, family, health, freedom, trust, and generosity.

Our method for detecting outliers is to first visualize the frequency distribution of the scores. We then check if the distribution shape is similar to a "normal" curve. If so, we can confidently say that any point that falls outside of three standard deviations from the mean is an outlier.

To start, we define some variables and functions to help visualize these frequencies. Each visualization will be a histogram where the scores are binned in 0.0625-intervals.

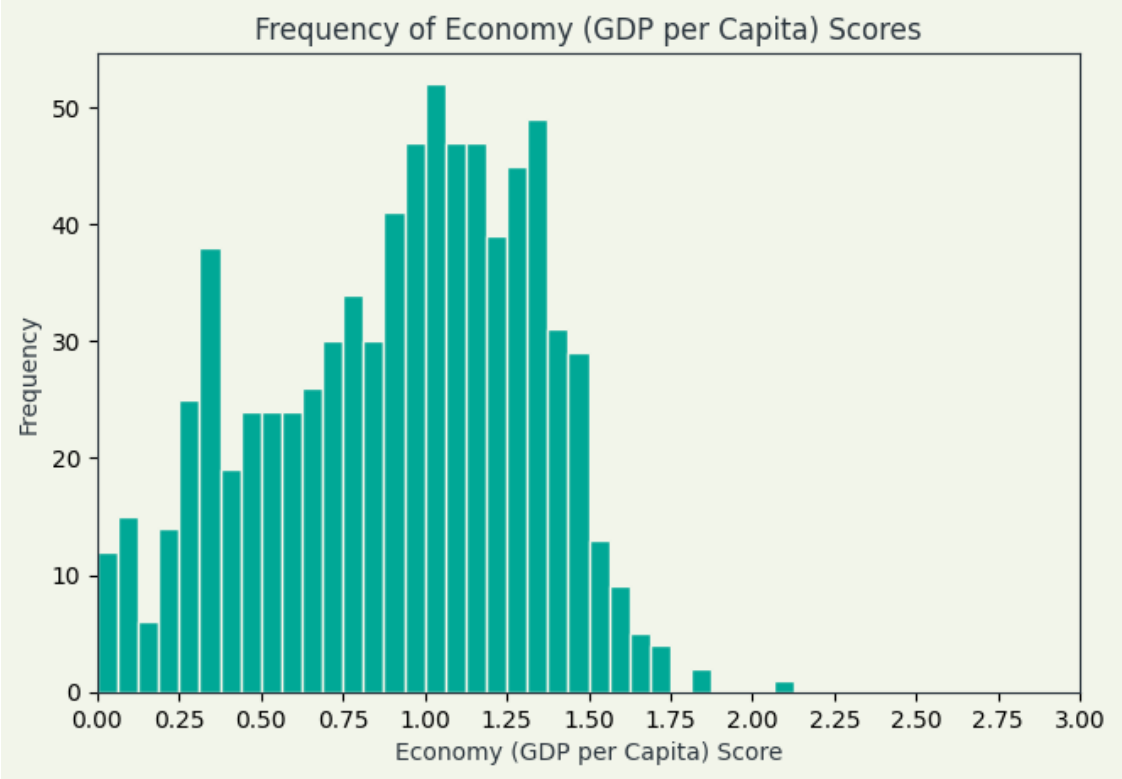
```
# Colors
ivory = '#F2F5EA'
persian_green = '#00A896'
slate_gray = '#748189'
blush = '#E75A7C'
gunmetal = '#2C363F'

# Plot Functions
def set_color(fig, ax):
    fig.patch.set_facecolor(ivory)
    ax.yaxis.label.set_color(gunmetal)
    ax.xaxis.label.set_color(gunmetal)
    ax.title.set_color(gunmetal)
    for spine in ax.spines.values():
        spine.set_edgecolor(gunmetal)


def show_frequency(score, x_min=0.0, x_max=3.0, group=0.0625,
                   color=persian_green):
    fig, ax = plt.subplots(figsize=[7.5, 5.0], dpi=100)

    set_color(fig, ax)

    ax.set(xlim=(x_min, x_max), xticks=np.arange(x_min, x_max + 0.25,
0.25),
           xlabel=f'{score} Score', ylabel='Frequency',
```

```
            title=f'Frequency of {score} Scores', facecolor=ivory)

    ax.hist(happy[score], bins=np.arange(x_min, x_max, group),
            color=color, edgecolor=ivory)

    return fig, ax
```

Outlying Economy Scores

```
show_frequency('Economy (GDP per Capita)')
```

```
(<Figure size 750x500 with 1 Axes>,
 <matplotlib.axes._subplots.AxesSubplot at 0x7f479df45cd0>)
```



We see that the frequency distribution of economic scores is relatively normally-distributed, with its bell shape. Thus, we can detect if there are any outliers by computing the Z-score for each point and checking if they are greater than 3 or less than -3.

```
happy[np.abs(stats.zscore(happy['Economy (GDP per Capita)'])) > 3.0]
```

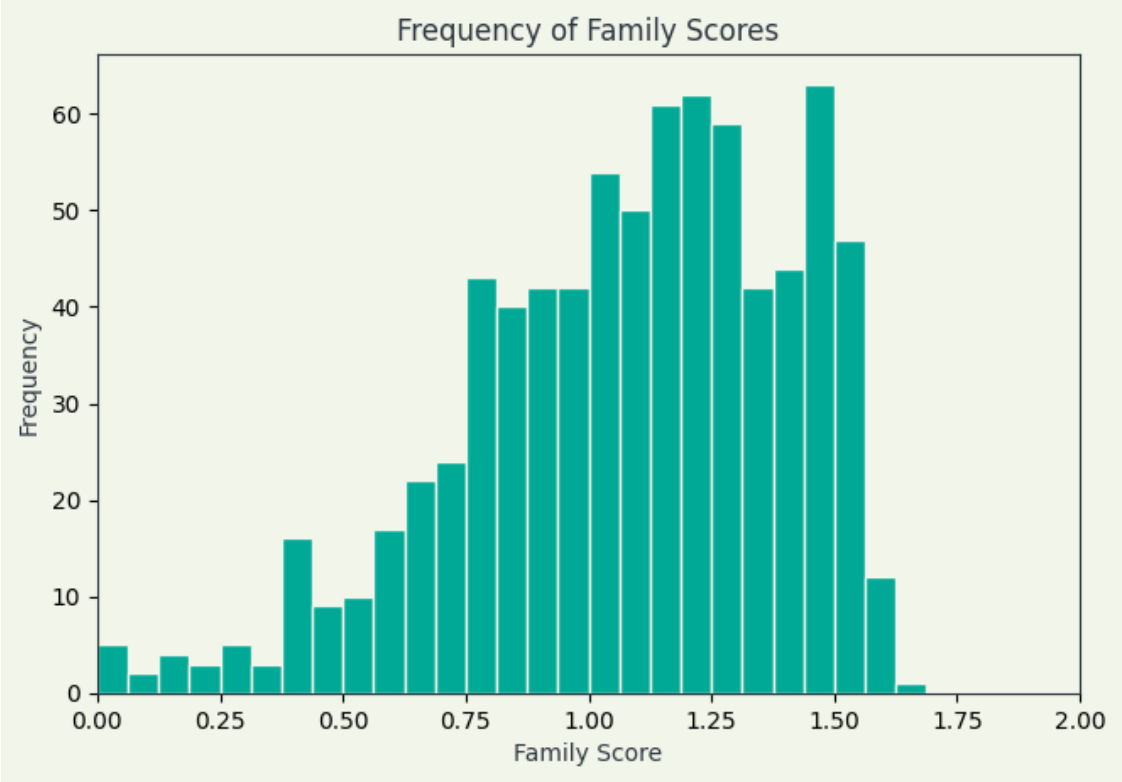
```
Empty DataFrame
Columns: [Country, Region, Happiness Rank, Happiness Score, Standard
Error, Economy (GDP per Capita), Family, Health (Life Expectancy),
Freedom, Trust (Government Corruption), Generosity, Year]
Index: []
```

As we can see, there are no outlying values for the economic scores.

Outlying Family Scores

```
show_frequency('Family', x_max=2.0)
```

```
(<Figure size 750x500 with 1 Axes>,
 <matplotlib.axes._subplots.AxesSubplot at 0x7f479a5ddb90>)
```




```
happy[np.abs(stats.zscore(happy['Family'])) > 3.0]
```

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family
147	Central African Republic	Sub-Saharan Africa	148	3.678	0.061120	0.07850	0.0
312	Togo	Sub-Saharan Africa	155	3.303	0.111000	0.28123	0.0
469	Central African Republic	Sub-Saharan Africa	155	2.693	0.171884	0.00000	0.0
624	Central African Republic	Sub-Saharan Africa	155	3.083	0.000000	0.02400	0.0
780	Central African Republic	Sub-Saharan Africa	155	3.083	0.000000	0.02600	0.0

```
     Health (Life Expectancy)   Freedom  Trust (Government Corruption)
\
147                   0.066990  0.488790                        0.082890

312                   0.248110  0.346780                        0.115870

469                   0.018773  0.270842                        0.056565

624                   0.010000  0.305000                        0.038000

780                   0.105000  0.225000                        0.035000


     Generosity  Year
147    0.238350  2015
312    0.175170  2016
469    0.280876  2017
624    0.218000  2018
780    0.235000  2019
```

Some outlying family scores fall out by three standard deviations from the mean. We simply substitute these values with the mean family score to address the issue. It should be noted that these values are correlated with the country. It indicates that although they are outlying values, they may not have occurred because of data entry errors.

```
happy.loc[np.abs(stats.zscore(happy['Family'])) > 3.0, 'Family'] =
np.nan
happy.fillna(np.nanmean(happy['Family']), inplace=True)
happy
```

```
                     Country             Region  Happiness Rank  \
0                Switzerland      Western Europe               1
1                    Iceland      Western Europe               2
2                    Denmark      Western Europe               3
3                     Norway      Western Europe               4
4                     Canada       North America               5
..                       ...                 ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.349510
1              7.561         0.04884                   1.30232
```

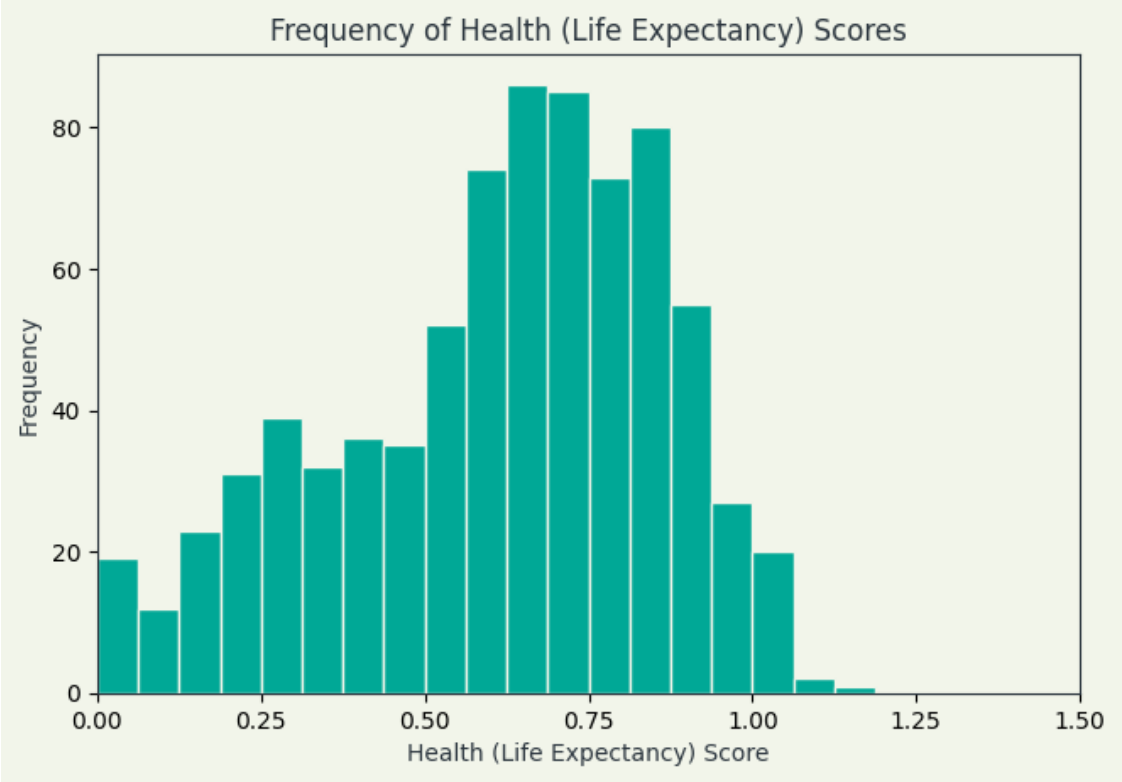
```
1.402230
2              7.527         0.03328                    1.32548
1.360580
3              7.522         0.03880                    1.45900
1.330950
4              7.427         0.03553                    1.32629
1.322610
..               ...            ...                        ...       .
..
777            3.334         0.00000                    0.35900
0.711000
778            3.231         0.00000                    0.47600
0.885000
779            3.203         0.00000                    0.35000
0.517000
780            3.083         0.00000                    0.02600
1.085332
781            2.853         0.00000                    0.30600
0.575000

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.41978

1                     0.94784  0.62877                        0.14145

2                     0.87464  0.64938                        0.48357

3                     0.88521  0.66973                        0.36503

4                     0.90563  0.63297                        0.32957

..                        ...      ...                            ...

777                   0.61400  0.55500                        0.41100

778                   0.49900  0.41700                        0.14700

779                   0.36100  0.00000                        0.02500

780                   0.10500  0.22500                        0.03500

781                   0.29500  0.01000                        0.09100


     Generosity  Year
0       0.29678  2015
1       0.43630  2015
2       0.34139  2015
```

```
3       0.34699  2015
4       0.45811  2015
..          ...   ...
777     0.21700  2019
778     0.27600  2019
779     0.15800  2019
780     0.23500  2019
781     0.20200  2019

[782 rows x 12 columns]
```

Outlying Health Scores

```
show_frequency('Health (Life Expectancy)', x_max=1.5)
```

```
(<Figure size 750x500 with 1 Axes>,
 <matplotlib.axes._subplots.AxesSubplot at 0x7f479a56de50>)
```




```
happy[np.abs(stats.zscore(happy['Health (Life Expectancy)'])) > 3.0]
```

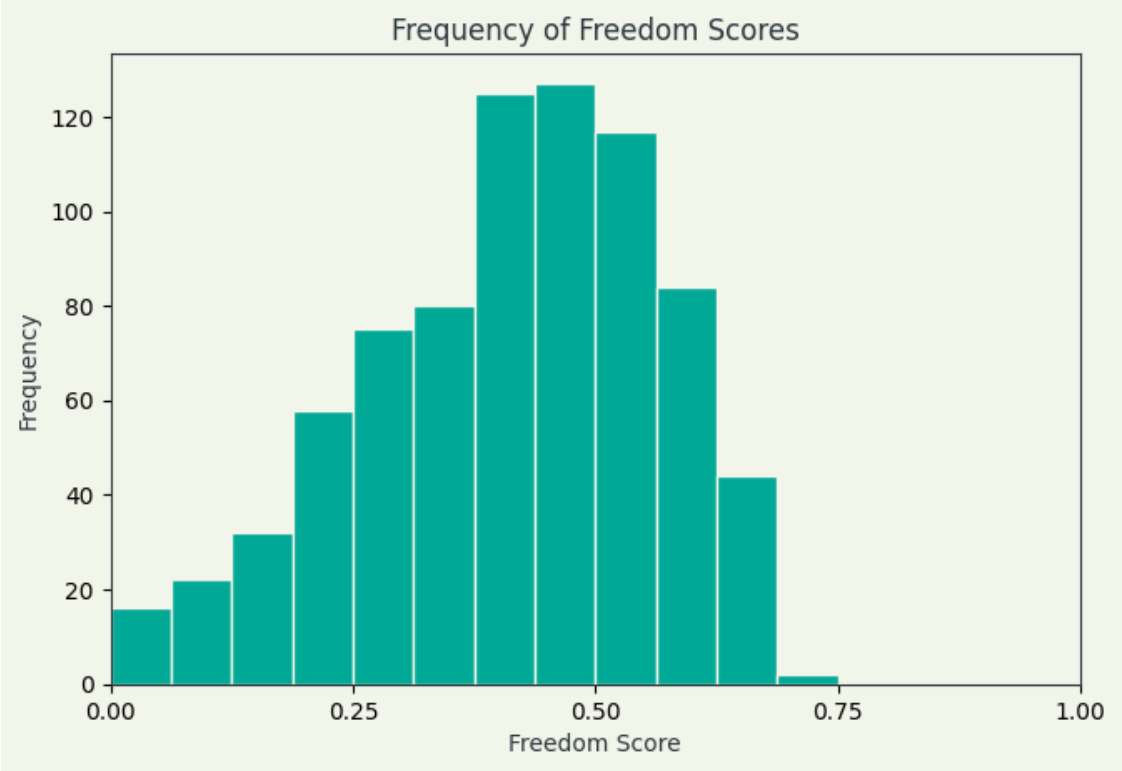
```
Empty DataFrame
Columns: [Country, Region, Happiness Rank, Happiness Score, Standard
Error, Economy (GDP per Capita), Family, Health (Life Expectancy),
Freedom, Trust (Government Corruption), Generosity, Year]
Index: []
```

Similar to the economic scores, the distribution of health scores visually fits a normal distribution. Furthermore, there are no outliers in the health scores.

Outlying Freedom Scores

```
show_frequency('Freedom', x_max=1.0, group=0.0625)
```

```
(<Figure size 750x500 with 1 Axes>,
 <matplotlib.axes._subplots.AxesSubplot at 0x7f479a483c90>)
```




```
happy[np.abs(stats.zscore(happy['Freedom'])) > 3.0]
```

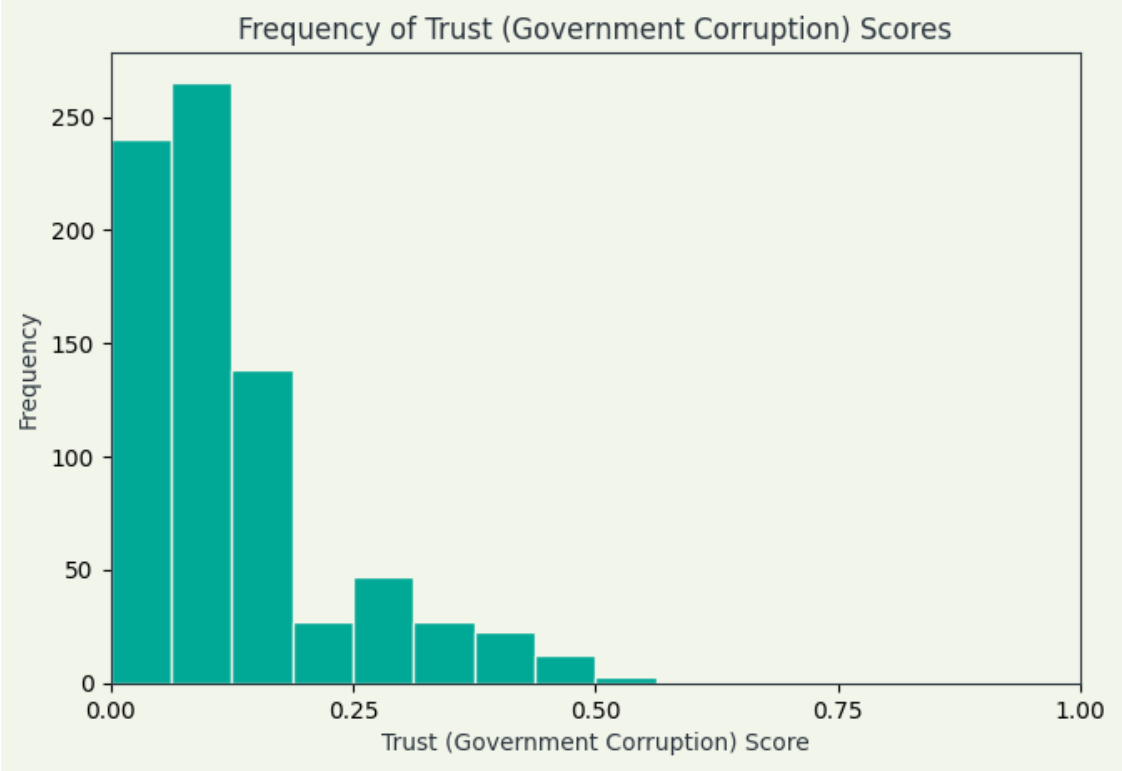
```
Empty DataFrame
Columns: [Country, Region, Happiness Rank, Happiness Score, Standard
Error, Economy (GDP per Capita), Family, Health (Life Expectancy),
Freedom, Trust (Government Corruption), Generosity, Year]
Index: []
```

As with the previous economic and health scores, there are no outlying values in the freedom scores either.

Outlying Trust Scores

```
show_frequency('Trust (Government Corruption)', x_max=1.0,
group=0.0625)
```

```
(<Figure size 750x500 with 1 Axes>,
 <matplotlib.axes._subplots.AxesSubplot at 0x7f479a40f1d0>)
```




```
happy[np.abs(stats.zscore(happy['Trust (Government Corruption)'])) >
3.0]
```

```
       Country                          Region  Happiness Rank  \
2      Denmark                  Western Europe               3
23   Singapore                Southeastern Asia              24
27       Qatar  Middle East and Northern Africa              28
153     Rwanda              Sub-Saharan Africa             154
158    Denmark                  Western Europe               1
179  Singapore                Southeastern Asia              22
193      Qatar  Middle East and Northern Africa              36
309     Rwanda              Sub-Saharan Africa             152
340  Singapore                Southeastern Asia              26
465     Rwanda              Sub-Saharan Africa             151
503  Singapore                Southeastern Asia              34
620     Rwanda              Sub-Saharan Africa             151
659  Singapore                Southeastern Asia              34

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
2              7.527        0.033280                  1.325480
1.360580
23             6.798        0.037800                  1.521860
1.020000
27             6.611        0.062570                  1.690420
1.078600
153            3.465        0.034640                  0.222080
```

0.773700
158 7.526 0.066000 1.441780
1.163740
179 6.739 0.065000 1.645550
0.867580
193 6.375 0.197000 1.824270
0.879640
309 3.515 0.071000 0.328460
0.615860
340 6.572 0.064723 1.692278
1.353814
465 3.471 0.072030 0.368746
0.945707
503 6.343 0.000000 1.529000
1.451000
620 3.408 0.000000 0.332000
0.896000
659 6.262 0.000000 1.572000
1.463000

	Health (Life Expectancy)	Freedom	Trust (Government Corruption)
2	0.874640	0.649380	0.483570
23	1.025250	0.542520	0.492100
27	0.797330	0.640400	0.522080
153	0.428640	0.592010	0.551910
158	0.795040	0.579410	0.444530
179	0.947190	0.487700	0.469870
193	0.717230	0.566790	0.480490
309	0.318650	0.543200	0.505210
340	0.949492	0.549841	0.464308
465	0.326425	0.581844	0.455220
503	1.008000	0.631000	0.457000
620	0.400000	0.636000	0.444000
659	1.141000	0.556000	0.453000

```
     Generosity  Year
2      0.341390  2015
23     0.311050  2015
27     0.325730  2015
153    0.226280  2015
158    0.361710  2016
179    0.327060  2016
193    0.323880  2016
309    0.235520  2016
340    0.345966  2017
465    0.252756  2017
503    0.261000  2018
620    0.200000  2018
659    0.271000  2019
```

The trust scores contain the most outliers within any of the categories. As with previous scores, the outlying points are correlated with specific countries. Thus, it should also be noted that the outlying trust scores are not due to data entry errors. Nonetheless, we would like to substitute these values with the mean.

```
happy.loc[np.abs(stats.zscore(happy['Trust (Government Corruption)']))
> 3.0,
          'Trust (Government Corruption)'] = np.nan

happy.fillna(np.nanmean(happy['Trust (Government Corruption)']),
inplace=True)
happy
```

```
                     Country             Region  Happiness Rank  \
0                Switzerland     Western Europe               1
1                    Iceland     Western Europe               2
2                    Denmark     Western Europe               3
3                     Norway     Western Europe               4
4                     Canada      North America               5
..                       ...                ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.349510
1              7.561         0.04884                   1.30232
1.402230
2              7.527         0.03328                   1.32548
1.360580
3              7.522         0.03880                   1.45900
```

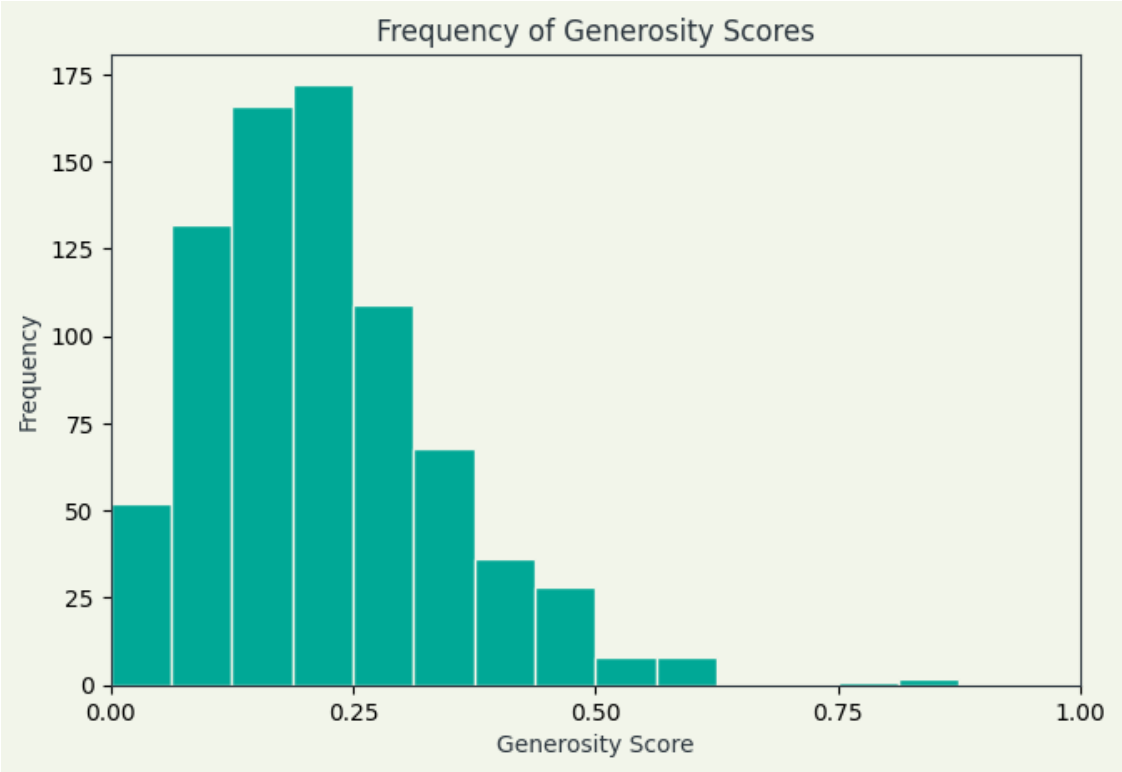
```
1.330950
4              7.427         0.03553                    1.32629
1.322610
..               ...             ...                        ...       .
..
777            3.334         0.00000                    0.35900
0.711000
778            3.231         0.00000                    0.47600
0.885000
779            3.203         0.00000                    0.35000
0.517000
780            3.083         0.00000                    0.02600
1.085332
781            2.853         0.00000                    0.30600
0.575000

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.419780

1                     0.94784  0.62877                        0.141450

2                     0.87464  0.64938                        0.119706

3                     0.88521  0.66973                        0.365030

4                     0.90563  0.63297                        0.329570

..                        ...      ...                             ...

777                   0.61400  0.55500                        0.411000

778                   0.49900  0.41700                        0.147000

779                   0.36100  0.00000                        0.025000

780                   0.10500  0.22500                        0.035000

781                   0.29500  0.01000                        0.091000


     Generosity  Year
0       0.29678  2015
1       0.43630  2015
2       0.34139  2015
3       0.34699  2015
4       0.45811  2015
..          ...   ...
777     0.21700  2019
```

```
778     0.27600  2019
779     0.15800  2019
780     0.23500  2019
781     0.20200  2019

[782 rows x 12 columns]
```

Outlying Generosity Scores

```
show_frequency('Generosity', x_max=1.0, group=0.0625)
```

```
(<Figure size 750x500 with 1 Axes>,
 <matplotlib.axes._subplots.AxesSubplot at 0x7f479a3a4cd0>)
```




```
happy[np.abs(stats.zscore(happy['Generosity'])) > 3.0]
```

```
       Country             Region  Happiness Rank  Happiness Score  \
128    Myanmar  Southeastern Asia             129            4.307
190   Thailand  Southeastern Asia              33            6.474
276    Myanmar  Southeastern Asia             119            4.395
395  Indonesia  Southeastern Asia              81            5.262
428    Myanmar  Southeastern Asia             114            4.545
599    Myanmar  Southeastern Asia             130            4.308

     Standard Error  Economy (GDP per Capita)    Family  \
128        0.043510                  0.271080  0.709050
190        0.078000                  1.089300  1.044770
276        0.068000                  0.341120  0.699810
```

```
395        0.090889                  0.995539  1.274445
428        0.069740                  0.367111  1.123236
599        0.000000                  0.682000  1.174000

     Health (Life Expectancy)   Freedom  Trust (Government Corruption)
\
128                  0.482460  0.440170                       0.190340

190                  0.649150  0.495530                       0.028330

276                  0.398800  0.426920                       0.202430

395                  0.492346  0.443323                       0.015317

428                  0.397523  0.514492                       0.188816

599                  0.429000  0.580000                       0.178000


     Generosity  Year
128    0.795880  2015
190    0.586960  2016
276    0.819710  2016
395    0.611705  2017
428    0.838075  2017
599    0.598000  2018
```

Lastly, there are several outliers for the generosity score. As with every other category, we provide the same treatment. We replace the outliers with the mean.

```
happy.loc[np.abs(stats.zscore(happy['Generosity'])) > 3.0,
'Generosity'] = \
    np.nan

happy.fillna(np.nanmean(happy['Generosity']), inplace=True)
happy
```

```
                     Country             Region  Happiness Rank  \
0                Switzerland      Western Europe               1
1                    Iceland      Western Europe               2
2                    Denmark      Western Europe               3
3                     Norway      Western Europe               4
4                     Canada       North America               5
..                       ...                 ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156
```

```
     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
0              7.587         0.03411                   1.39651
1.349510
1              7.561         0.04884                   1.30232
1.402230
2              7.527         0.03328                   1.32548
1.360580
3              7.522         0.03880                   1.45900
1.330950
4              7.427         0.03553                   1.32629
1.322610
..               ...             ...                       ...       .
..
777            3.334         0.00000                   0.35900
0.711000
778            3.231         0.00000                   0.47600
0.885000
779            3.203         0.00000                   0.35000
0.517000
780            3.083         0.00000                   0.02600
1.085332
781            2.853         0.00000                   0.30600
0.575000

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
0                     0.94143  0.66557                        0.419780

1                     0.94784  0.62877                        0.141450

2                     0.87464  0.64938                        0.119706

3                     0.88521  0.66973                        0.365030

4                     0.90563  0.63297                        0.329570

..                        ...      ...                             ...

777                   0.61400  0.55500                        0.411000

778                   0.49900  0.41700                        0.147000

779                   0.36100  0.00000                        0.025000

780                   0.10500  0.22500                        0.035000

781                   0.29500  0.01000                        0.091000
```

```
     Generosity  Year
0       0.29678  2015
1       0.43630  2015
2       0.34139  2015
3       0.34699  2015
4       0.45811  2015
..          ...   ...
777     0.21700  2019
778     0.27600  2019
779     0.15800  2019
780     0.23500  2019
781     0.20200  2019

[782 rows x 12 columns]
```

Exploratory Data Analysis

With the data cleaned, we would like to perform an overall analysis to capture trends and correlations in the data. Most notably, we would like to see how the happiness scores change over time and which features heavily contribute to a higher happiness score.

Happiness Score Central Tendencies

First, we would like to take the central tendencies of the happiness scores for each year. We do so by grouping the data by year and aggregating the happiness scores into the mean and median. Note that we do not consider the mode because the happiness scores are on a continuous interval. Thus, it will only determine which values are repeated because of a coincidence.

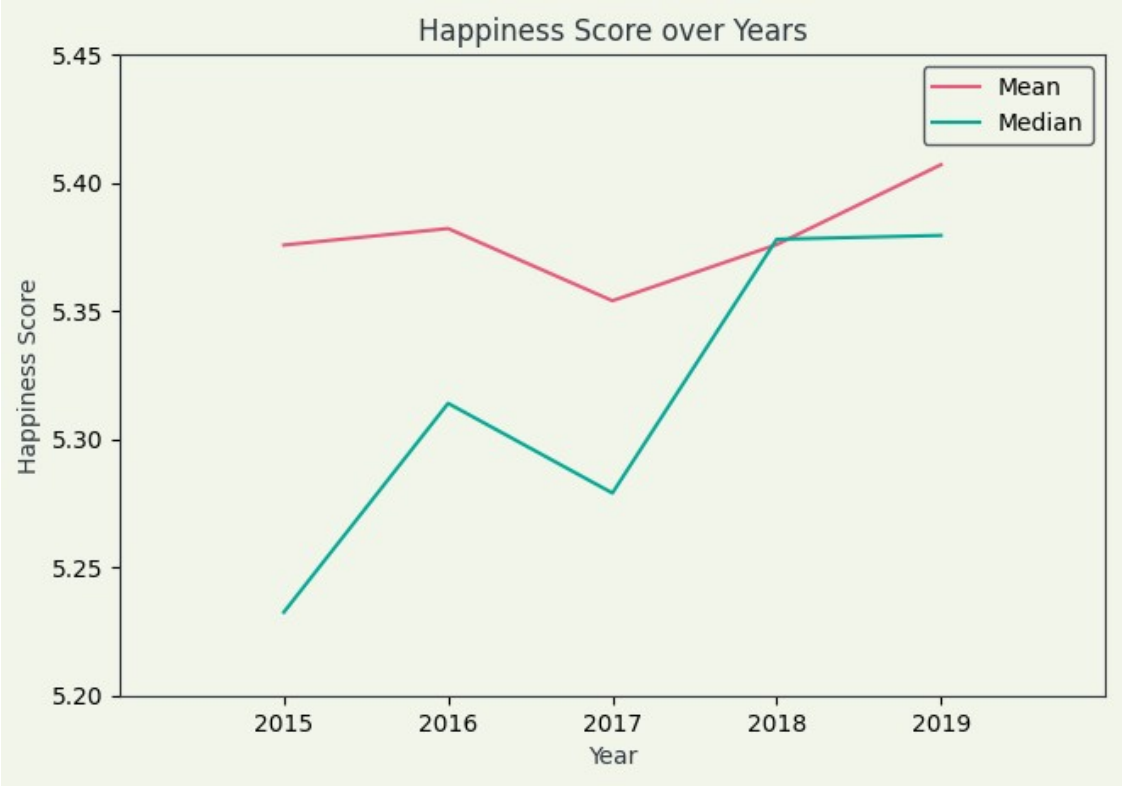
```
central_happy = happy.groupby('Year').agg(
    mean=('Happiness Score', np.mean),
    median=('Happiness Score', np.median)
).rename(columns={
    'mean': 'Mean',
    'median': 'Median'
})

central_happy
```

```
          Mean  Median
Year
2015  5.375734  5.2325
2016  5.382185  5.3140
2017  5.354019  5.2790
2018  5.375917  5.3780
2019  5.407096  5.3795
```

Now, we plot the central tendencies on a line graph to visualize the changes over the years.

```
cnt_hap_fig, cnt_hap_ax = plt.subplots(figsize=[7.5, 5], dpi=100)
set_color(cnt_hap_fig, cnt_hap_ax)

cnt_hap_ax.set(xlim=(2014, 2020), xticks=np.arange(2015, 2020, 1),
               xlabel='Year', ylim=(5.2, 5.45),
               yticks=np.arange(5.2, 5.50, 0.05), ylabel='Happiness
Score',
               title='Happiness Score over Years', facecolor=ivory)

cnt_hap_ax.plot(central_happy.index, central_happy['Mean'], c=blush)

cnt_hap_ax.plot(central_happy.index, central_happy['Median'],
c=persian_green)

cnt_hap_ax.legend(['Mean', 'Median'], facecolor=ivory,
edgecolor=gunmetal)
```

```
<matplotlib.legend.Legend at 0x7f479a28add0>
```



From the line graph, both the means and medians generally increase. It indicates that the happiness score tends to be raised throughout time. However, it is interesting that both central tendencies dipped in 2017. It can suggest that there may have been a global event that affected the well-being and happiness of individuals during that time.

Happiness Ranks

Next, we would like to check which countries improved or maintained their happiness rankings. We need to have a clear definition of improving a rank and what it means to have a steady score. For a country to have improved its ranking throughout five years, its rank must have increased every year. For a country to have a steady happiness level, its score must not change or incur the least amount of change throughout the years.

```
# Calculates total change in a series.
def get_total_change(series):
    total_change = 0
    last = 0
    for index, value in series.items():
        if last >= 0:
            total_change += np.abs(value - last)

        last = value
    return total_change

# Groups data by country and then calculates the total change in
happiness rank
# for each country.
steady_happy = happy.groupby('Country').agg(
    total_change=('Happiness Rank', get_total_change),
    minimum_rank=('Happiness Rank', np.min),
    maximum_rank=('Happiness Rank', np.max)
).rename(columns={
    'total_change': 'Total Happiness Rank Change',
    'minimum_rank': 'Minimum Happiness Rank',
    'maximum_rank': 'Maximum Happiness Rank'
}).reset_index().sort_values('Total Happiness Rank Change')

steady_happy
```

```
          Country  Total Happiness Rank Change  Minimum Happiness Rank
\
60        Iceland                            4                       2

142   Switzerland                            6                       1

38        Denmark                            8                       1

25         Canada                            9                       5

108        Norway                            9                       1

..            ...                          ...                     ...

55         Guinea                          184                     118
```

```
21   Burkina Faso                          189                       115

42          Egypt                          199                       104

68    Ivory Coast                          203                        99

14          Benin                          208                       102
```

```
     Maximum Happiness Rank
60                         4
142                        6
38                         3
25                         9
108                        4
..                       ...
55                       151
21                       152
42                       137
68                       151
14                       155

[165 rows x 4 columns]
```

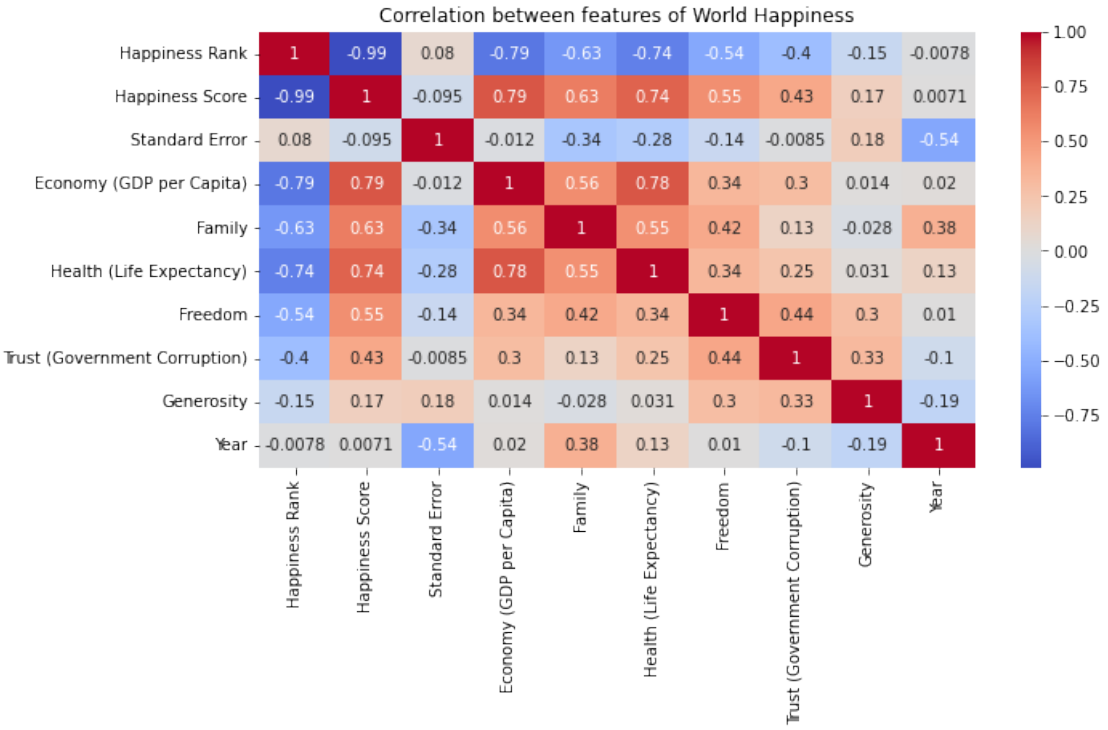
```
def is_decreasing(series):
    last = float('inf')
    for index, value in series.items():
        if value > last:
            return False
        last = value
    return True

increase_happy = happy.groupby('Country').filter(
    lambda df : is_decreasing(df['Happiness Rank'])
).groupby('Country').agg(
    total_change=('Happiness Rank', get_total_change)
).reset_index(
).rename(columns={'total_change': 'Total Change'}
).sort_values('Total Change', ascending=False)

increase_happy
```

```
         Country  Total Change
1          Benin           208
16   Ivory Coast           203
3   Burkina Faso           189
4       Cambodia           181
34          Togo           177
..           ...           ...
```

```
22          Oman            22
25   Puerto Rico            15
11       Finland            11
20   New Zealand            10
19   Netherlands             9

[35 rows x 2 columns]
```

Visualize the relationship between happiness score and other features such as GDP, social support, freedom, etc.

```
#create a heatmap so see the relationship between the happiness score
and other features

import seaborn as sns

plt.figure(figsize=(10, 5))
sns.heatmap(happy.corr(), cmap="coolwarm", annot = True)
plt.title('Correlation between features of World Happiness')
plt.show()
```



Find out what features contribute to happiness. If you are the president of a country, what would you do to make citizens happier?

From the heatmap, we can see that Happiness Score's correlation with other features from most correlated to least correlated is Economy, Health, Family, Freedom , Trust, and then Generosity. With a correlation of greater than 0.7, Happiness Score in a country and more tied in with their GDP and their life expectancy.

Therefore if we were the president of a country, we would first increase the GDP per capita of the country as there is a strong correlation between the GDP and Happiness score. To increase the GDP, we would improve the quality of education and increase job skills within the country. This will also increase the life expectancy of the country as it is highly correlated with GDP.

Modeling and Question Answering

Model 1: Linear Regression

Linear Regression is a model to determine the relationship between numerical features. It is very easy to understand and makes a good baseline model to understand the dataset.

```
# seperate into training and testing sets
testing = happy.loc[happy['Year'] == 2019]
training =happy.loc[happy['Year'].isin([2015, 2016, 2017, 2018])]

#create training data and testing data
#training data
train_lr_x = training[['Economy (GDP per Capita)', 'Freedom']]
train_lr_y= training['Happiness Score']
#testing data
test_lr_y = testing['Happiness Score']
test_lr_x = testing[['Economy (GDP per Capita)', 'Freedom']]
```

For our training data, we only use the 'Economy (GDP per Capita)' and 'Freedom' feature as those two features are not highly correlated with each other and they have high correlation with 'Happiness Score'.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

happinessLR = LinearRegression()
happinessLR.fit(train_lr_x, train_lr_y)
```

```
LinearRegression()
```

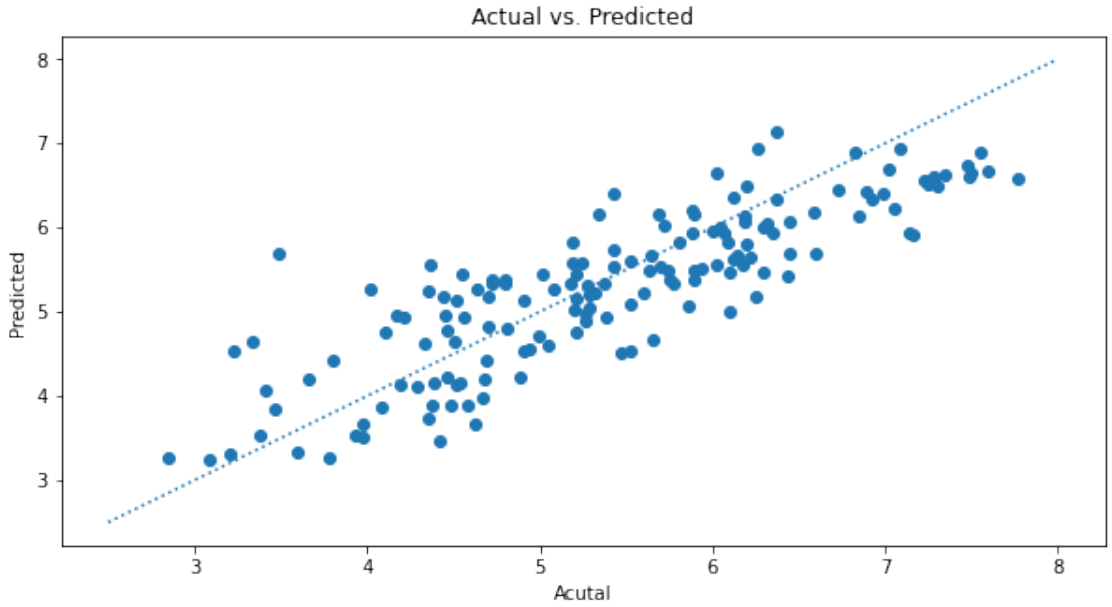
```
y_predict = happinessLR.predict(test_lr_x)
y_predict
```

```
array([6.59122894, 6.6626267 , 6.88615273, 6.65461629, 6.60418699,
       6.74503096, 6.62752092, 6.4955916 , 6.60982775, 6.50736622,
       6.5590498 , 5.92573599, 5.93800508, 6.93136353, 6.23229828,
       6.70080218, 6.41409853, 6.33002468, 6.42984292, 6.12851025,
       6.90252219, 6.44021604, 5.69740954, 6.18221622, 6.06366581,
       5.68480568, 5.43117848, 6.33789757, 7.14109672, 5.93549793,
       6.04255152, 5.47144719, 6.01211142, 6.93282477, 5.17122711,
       5.64030188, 6.49050922, 5.79395806, 6.13282728, 6.07159983,
       5.55509185, 5.67707724, 5.62517421, 6.36122596, 4.99499678,
       5.47645734, 5.8287791 , 5.93911491, 5.9964448 , 5.55419315,
```

```
       6.64821296, 5.95345919, 5.51721776, 5.48295232, 6.15832113,
       5.38290908, 5.94538211, 6.20917269, 5.06771411, 5.83665338,
       5.3292031 , 5.38961102, 5.48488445, 6.02249564, 5.54263244,
       6.15901702, 4.67409715, 5.67547294, 5.49881342, 5.21567534,
       4.52810134, 5.60825054, 5.10276708, 4.50803297, 5.53281909,
       6.40609089, 5.74553866, 4.93893504, 5.3462851 , 6.15901702,
       5.22285234, 5.03938666, 5.19324426, 5.32538054, 4.97744389,
       4.89893719, 5.5711946 , 4.74773537, 5.15597303, 5.45243306,
       5.02739815, 5.57334895, 5.82870687, 5.33916229, 5.27090368,
       4.59440701, 5.44985784, 4.71101142, 4.56334186, 4.5380522 ,
       5.14078978, 4.22523104, 4.8063015 , 5.34614065, 5.39189179,
       5.38632742, 5.34766856, 4.82978682, 5.18138771, 4.42807613,
       4.19744116, 3.98344374, 5.2624085 , 3.67223243, 3.8860104 ,
       4.92841331, 5.45069402, 4.16052135, 5.14531934, 4.14352268,
       4.64323341, 3.88790083, 4.21317308, 4.78722624, 4.96462197,
       5.1919247 , 3.47343628, 4.15808364, 3.89541121, 5.55746843,
       5.23823145, 3.72991512, 4.62332475, 4.10618893, 4.92611311,
       4.12708516, 4.9474316 , 4.76736068, 3.87068409, 5.25892071,
       3.67313528, 3.51119644, 3.52540599, 4.42946237, 3.26711727,
       4.2028708 , 3.32863362, 5.69497045, 3.85453825, 4.06794261,
       3.53801401, 4.64914781, 4.54237202, 3.31783825, 3.24134424,
       3.25876932])
```

```
import matplotlib.pyplot as plt
plt.figure(figsize=(10,5))
plt.scatter(test_lr_y, y_predict)
xpoints = np.array([2.5, 3, 4, 5, 6, 7, 8])
ypoints = np.array([2.5, 3, 4, 5, 6, 7, 8])
plt.plot(xpoints, ypoints, linestyle = 'dotted')
plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.title('Actual vs. Predicted')
```

```
Text(0.5, 1.0, 'Actual vs. Predicted')
```




```
np.column_stack((y_predict, test_lr_y))

array([[6.59122894, 7.769     ],
       [6.6626267 , 7.6       ],
       [6.88615273, 7.554     ],
       [6.65461629, 7.494     ],
       [6.60418699, 7.488     ],
       [6.74503096, 7.48      ],
       [6.62752092, 7.343     ],
       [6.4955916 , 7.307     ],
       [6.60982775, 7.278     ],
       [6.50736622, 7.246     ],
       [6.5590498 , 7.228     ],
       [5.92573599, 7.167     ],
       [5.93800508, 7.139     ],
       [6.93136353, 7.09      ],
       [6.23229828, 7.054     ],
       [6.70080218, 7.021     ],
       [6.41409853, 6.985     ],
       [6.33002468, 6.923     ],
       [6.42984292, 6.892     ],
       [6.12851025, 6.852     ],
       [6.90252219, 6.825     ],
       [6.44021604, 6.726     ],
       [5.69740954, 6.595     ],
       [6.18221622, 6.592     ],
       [6.06366581, 6.446     ],
       [5.68480568, 6.444     ],
       [5.43117848, 6.436     ],
       [6.33789757, 6.375     ],
       [7.14109672, 6.374     ],
```

```
[5.93549793, 6.354     ],
[6.04255152, 6.321     ],
[5.47144719, 6.3       ],
[6.01211142, 6.293     ],
[6.93282477, 6.262     ],
[5.17122711, 6.253     ],
[5.64030188, 6.223     ],
[6.49050922, 6.199     ],
[5.79395806, 6.198     ],
[6.13282728, 6.192     ],
[6.07159983, 6.182     ],
[5.55509185, 6.174     ],
[5.67707724, 6.149     ],
[5.62517421, 6.125     ],
[6.36122596, 6.118     ],
[4.99499678, 6.105     ],
[5.47645734, 6.1       ],
[5.8287791 , 6.086     ],
[5.93911491, 6.07      ],
[5.9964448 , 6.046     ],
[5.55419315, 6.028     ],
[6.64821296, 6.021     ],
[5.95345919, 6.008     ],
[5.51721776, 5.94      ],
[5.48295232, 5.895     ],
[6.15832113, 5.893     ],
[5.38290908, 5.89      ],
[5.94538211, 5.888     ],
[6.20917269, 5.886     ],
[5.06771411, 5.86      ],
[5.83665338, 5.809     ],
[5.3292031 , 5.779     ],
[5.38961102, 5.758     ],
[5.48488445, 5.743     ],
[6.02249564, 5.718     ],
[5.54263244, 5.697     ],
[6.15901702, 5.693     ],
[4.67409715, 5.653     ],
[5.67547294, 5.648     ],
[5.49881342, 5.631     ],
[5.21567534, 5.603     ],
[4.52810134, 5.529     ],
[5.60825054, 5.525     ],
[5.10276708, 5.523     ],
[4.50803297, 5.467     ],
[5.53281909, 5.432     ],
[6.40609089, 5.43      ],
[5.74553866, 5.425     ],
[4.93893504, 5.386     ],
[5.3462851 , 5.373     ],
```

```
[6.15901702, 5.339     ],
[5.22285234, 5.323     ],
[5.03938666, 5.287     ],
[5.19324426, 5.285     ],
[5.32538054, 5.274     ],
[4.97744389, 5.265     ],
[4.89893719, 5.261     ],
[5.5711946 , 5.247     ],
[4.74773537, 5.211     ],
[5.15597303, 5.208     ],
[5.45243306, 5.208     ],
[5.02739815, 5.197     ],
[5.57334895, 5.192     ],
[5.82870687, 5.191     ],
[5.33916229, 5.175     ],
[5.27090368, 5.082     ],
[4.59440701, 5.044     ],
[5.44985784, 5.011     ],
[4.71101142, 4.996     ],
[4.56334186, 4.944     ],
[4.5380522 , 4.913     ],
[5.14078978, 4.906     ],
[4.22523104, 4.883     ],
[4.8063015 , 4.812     ],
[5.34614065, 4.799     ],
[5.39189179, 4.796     ],
[5.38632742, 4.722     ],
[5.34766856, 4.719     ],
[4.82978682, 4.707     ],
[5.18138771, 4.7       ],
[4.42807613, 4.696     ],
[4.19744116, 4.681     ],
[3.98344374, 4.668     ],
[5.2624085 , 4.639     ],
[3.67223243, 4.628     ],
[3.8860104 , 4.587     ],
[4.92841331, 4.559     ],
[5.45069402, 4.548     ],
[4.16052135, 4.534     ],
[5.14531934, 4.519     ],
[4.14352268, 4.516     ],
[4.64323341, 4.509     ],
[3.88790083, 4.49      ],
[4.21317308, 4.466     ],
[4.78722624, 4.461     ],
[4.96462197, 4.456     ],
[5.1919247 , 4.437     ],
[3.47343628, 4.418     ],
[4.15808364, 4.39      ],
[3.89541121, 4.374     ],
```

```
       [5.55746843, 4.366     ],
       [5.23823145, 4.36      ],
       [3.72991512, 4.35      ],
       [4.62332475, 4.332     ],
       [4.10618893, 4.286     ],
       [4.92611311, 4.212     ],
       [4.12708516, 4.189     ],
       [4.9474316 , 4.166     ],
       [4.76736068, 4.107     ],
       [3.87068409, 4.085     ],
       [5.25892071, 4.015     ],
       [3.67313528, 3.975     ],
       [3.51119644, 3.973     ],
       [3.52540599, 3.933     ],
       [4.42946237, 3.802     ],
       [3.26711727, 3.775     ],
       [4.2028708 , 3.663     ],
       [3.32863362, 3.597     ],
       [5.69497045, 3.488     ],
       [3.85453825, 3.462     ],
       [4.06794261, 3.41      ],
       [3.53801401, 3.38      ],
       [4.64914781, 3.334     ],
       [4.54237202, 3.231     ],
       [3.31783825, 3.203     ],
       [3.24134424, 3.083     ],
       [3.25876932, 2.853     ]])
```

```
rms = mean_squared_error(test_lr_y, y_predict, squared=False)
print(rms)
```

```
0.6058271239843792
```

The root mean squared shows us that the standard deviation of the predicated values compared to the actual values is only 0.6.

```
#compare the
Predictions = pd.DataFrame({'Prediction Happiness': y_predict, 'Actual
Happiness': test_lr_y})
Predictions.insert(0, 'Actual Rankings', range(1, 1+len(Predictions)))
Predictions = Predictions.sort_values(by='Prediction Happiness',
ascending=False)
Predictions.insert(0, 'Predicted Rankings', range(1,
1+len(Predictions)))
print(Predictions.to_markdown())
```

```
|     |   Predicted Rankings |   Actual Rankings |   Prediction
Happiness |   Actual Happiness |
|----:|---------------------:|------------------:|--------------------
---:|-------------------:|
| 654 |                    1 |                29 |
```

7.1411	6.374	
659	2	34
6.93282	6.262	
639	3	14
6.93136	7.09	
646	4	21
6.90252	6.825	
628	5	3
6.88615	7.554	
631	6	6
6.74503	7.48	
641	7	16
6.7008	7.021	
627	8	2
6.66263	7.6	
629	9	4
6.65462	7.494	
676	10	51
6.64821	6.021	
632	11	7
6.62752	7.343	
634	12	9
6.60983	7.278	
630	13	5
6.60419	7.488	
626	14	1
6.59123	7.769	
636	15	11
6.55905	7.228	
635	16	10
6.50737	7.246	
633	17	8
6.49559	7.307	
662	18	37
6.49051	6.199	
647	19	22
6.44022	6.726	
644	20	19
6.42984	6.892	
642	21	17
6.4141	6.985	
701	22	76
6.40609	5.43	
669	23	44
6.36123	6.118	
653	24	28
6.3379	6.375	
643	25	18
6.33002	6.923	
640	26	15

6.2323	7.054	
683	27	58
6.20917	5.886	
649	28	24
6.18222	6.592	
691	29	66
6.15902	5.693	
705	30	80
6.15902	5.339	
680	31	55
6.15832	5.893	
664	32	39
6.13283	6.192	
645	33	20
6.12851	6.852	
665	34	40
6.0716	6.182	
650	35	25
6.06367	6.446	
656	36	31
6.04255	6.321	
689	37	64
6.0225	5.718	
658	38	33
6.01211	6.293	
674	39	49
5.99644	6.046	
677	40	52
5.95346	6.008	
682	41	57
5.94538	5.888	
673	42	48
5.93911	6.07	
638	43	13
5.93801	7.139	
655	44	30
5.9355	6.354	
637	45	12
5.92574	7.167	
685	46	60
5.83665	5.809	
672	47	47
5.82878	6.086	
718	48	93
5.82871	5.191	
663	49	38
5.79396	6.198	
702	50	77
5.74554	5.425	
648	51	23

5.69741	6.595	
773	52	148
5.69497	3.488	
651	53	26
5.68481	6.444	
667	54	42
5.67708	6.149	
693	55	68
5.67547	5.648	
661	56	36
5.6403	6.223	
668	57	43
5.62517	6.125	
697	58	72
5.60825	5.525	
717	59	92
5.57335	5.192	
712	60	87
5.57119	5.247	
755	61	130
5.55747	4.366	
666	62	41
5.55509	6.174	
675	63	50
5.55419	6.028	
690	64	65
5.54263	5.697	
700	65	75
5.53282	5.432	
678	66	53
5.51722	5.94	
694	67	69
5.49881	5.631	
688	68	63
5.48488	5.743	
679	69	54
5.48295	5.895	
671	70	46
5.47646	6.1	
657	71	32
5.47145	6.3	
715	72	90
5.45243	5.208	
742	73	117
5.45069	4.548	
722	74	97
5.44986	5.011	
652	75	27
5.43118	6.436	
730	76	105

5.39189	4.796	
687	77	62
5.38961	5.758	
731	78	106
5.38633	4.722	
681	79	56
5.38291	5.89	
732	80	107
5.34767	4.719	
704	81	79
5.34629	5.373	
729	82	104
5.34614	4.799	
719	83	94
5.33916	5.175	
686	84	61
5.3292	5.779	
709	85	84
5.32538	5.274	
720	86	95
5.2709	5.082	
738	87	113
5.26241	4.639	
765	88	140
5.25892	4.015	
756	89	131
5.23823	4.36	
706	90	81
5.22285	5.323	
695	91	70
5.21568	5.603	
708	92	83
5.19324	5.285	
751	93	126
5.19192	4.437	
734	94	109
5.18139	4.7	
660	95	35
5.17123	6.253	
714	96	89
5.15597	5.208	
744	97	119
5.14532	4.519	
726	98	101
5.14079	4.906	
698	99	73
5.10277	5.523	
684	100	59
5.06771	5.86	
707	101	82

5.03939	5.287	
716	102	91
5.0274	5.197	
670	103	45
4.995	6.105	
710	104	85
4.97744	5.265	
750	105	125
4.96462	4.456	
762	106	137
4.94743	4.166	
703	107	78
4.93894	5.386	
741	108	116
4.92841	4.559	
760	109	135
4.92611	4.212	
711	110	86
4.89894	5.261	
733	111	108
4.82979	4.707	
728	112	103
4.8063	4.812	
749	113	124
4.78723	4.461	
763	114	138
4.76736	4.107	
713	115	88
4.74774	5.211	
723	116	98
4.71101	4.996	
692	117	67
4.6741	5.653	
777	118	152
4.64915	3.334	
746	119	121
4.64323	4.509	
758	120	133
4.62332	4.332	
721	121	96
4.59441	5.044	
724	122	99
4.56334	4.944	
778	123	153
4.54237	3.231	
725	124	100
4.53805	4.913	
696	125	71
4.5281	5.529	
699	126	74

4.50803	5.467	
769	127	144
4.42946	3.802	
735	128	110
4.42808	4.696	
727	129	102
4.22523	4.883	
748	130	123
4.21317	4.466	
771	131	146
4.20287	3.663	
736	132	111
4.19744	4.681	
743	133	118
4.16052	4.534	
753	134	128
4.15808	4.39	
745	135	120
4.14352	4.516	
761	136	136
4.12709	4.189	
759	137	134
4.10619	4.286	
775	138	150
4.06794	3.41	
737	139	112
3.98344	4.668	
754	140	129
3.89541	4.374	
747	141	122
3.8879	4.49	
740	142	115
3.88601	4.587	
764	143	139
3.87068	4.085	
774	144	149
3.85454	3.462	
757	145	132
3.72992	4.35	
766	146	141
3.67314	3.975	
739	147	114
3.67223	4.628	
776	148	151
3.53801	3.38	
768	149	143
3.52541	3.933	
767	150	142
3.5112	3.973	
752	151	127

```
3.47344 |              4.418 |
| 772 |                152 |               147 |
3.32863 |              3.597 |
| 779 |                153 |               154 |
3.31784 |              3.203 |
| 770 |                154 |               145 |
3.26712 |              3.775 |
| 781 |                155 |               156 |
3.25877 |              2.853 |
| 780 |                156 |               155 |
3.24134 |              3.083 |
```

Model 2: Ridge Regression

Other options? Least Absolute Deviation Regression

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from math import sqrt

# seperate into training and testing sets
testing2 = happy.loc[happy['Year'] == 2019]
training2 = happy.loc[happy['Year'].isin([2015, 2016, 2017, 2018])]

target_column2 = ['Happiness Score']
not_used2 =  ['Country', 'Region', 'Happiness Rank', 'Year', 'Standard
Error']
excluded2 = target_column2 + not_used2
excluded2
```

```
['Happiness Score',
 'Country',
 'Region',
 'Happiness Rank',
 'Year',
 'Standard Error']
```

```
predictors2 = list(set(list(training2.columns))-set(excluded2))

X2 = training2[predictors2].values
Y2 = training2[target_column2].values

X2_train, X2_test, Y2_train, Y2_test = train_test_split(X2, Y2,
test_size=0.30, random_state=40)

print(X2_train.shape); print(X2_test.shape)
```

```
(438, 6)
(188, 6)
```

```
from sklearn.linear_model import Ridge

# define model
```

```
ridgeReg = Ridge(alpha=1.0)


ridgeReg.fit(X2_train, Y2_train)
pred_train_ridgeReg = ridgeReg.predict(testing2.drop(excluded2, axis =
1).values)

rms2 = mean_squared_error(testing2['Happiness Score'],
pred_train_ridgeReg, squared=False)
print(rms2)

0.5758837127385211

predictedRankings2 = testing2.copy()

predictedRankings2['Happiness Score'] = pred_train_ridgeReg

testing2

                     Country             Region  Happiness Rank  \
626                  Finland     Western Europe               1
627                  Denmark     Western Europe               2
628                   Norway     Western Europe               3
629                  Iceland     Western Europe               4
630              Netherlands     Western Europe               5
..                       ...                ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
626            7.769             0.0                     1.340
1.587000
627            7.600             0.0                     1.383
1.573000
628            7.554             0.0                     1.488
1.582000
629            7.494             0.0                     1.380
1.624000
630            7.488             0.0                     1.396
1.522000
..               ...             ...                       ...
..
777            3.334             0.0                     0.359
0.711000
778            3.231             0.0                     0.476
0.885000
779            3.203             0.0                     0.350
```

```
0.517000
780            3.083             0.0                      0.026
1.085332
781            2.853             0.0                      0.306
0.575000

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
626                     0.986    0.596                           0.393

627                     0.996    0.592                           0.410

628                     1.028    0.603                           0.341

629                     1.026    0.591                           0.118

630                     0.999    0.557                           0.298

..                        ...      ...                             ...

777                     0.614    0.555                           0.411

778                     0.499    0.417                           0.147

779                     0.361    0.000                           0.025

780                     0.105    0.225                           0.035

781                     0.295    0.010                           0.091


     Generosity  Year
626       0.153  2019
627       0.252  2019
628       0.271  2019
629       0.354  2019
630       0.322  2019
..          ...   ...
777       0.217  2019
778       0.276  2019
779       0.158  2019
780       0.235  2019
781       0.202  2019

[156 rows x 12 columns]

predictedRankings2
```

```
                     Country             Region  Happiness Rank  \
626                  Finland     Western Europe               1
627                  Denmark     Western Europe               2
628                   Norway     Western Europe               3
629                  Iceland     Western Europe               4
630              Netherlands     Western Europe               5
..                       ...                ...             ...
777                   Rwanda  Sub-Saharan Africa             152
778                 Tanzania  Sub-Saharan Africa             153
779              Afghanistan       Southern Asia             154
780  Central African Republic  Sub-Saharan Africa             155
781              South Sudan  Sub-Saharan Africa             156

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
626         6.666243             0.0                     1.340
1.587000
627         6.835956             0.0                     1.383
1.573000
628         6.888524             0.0                     1.488
1.582000
629         6.625989             0.0                     1.380
1.624000
630         6.710150             0.0                     1.396
1.522000
..               ...             ...                       ...       .
..
777         5.081384             0.0                     0.359
0.711000
778         4.679890             0.0                     0.476
0.885000
779         3.445801             0.0                     0.350
0.517000
780         3.610647             0.0                     0.026
1.085332
781         3.523566             0.0                     0.306
0.575000

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
626                     0.986    0.596                          0.393

627                     0.996    0.592                          0.410

628                     1.028    0.603                          0.341

629                     1.026    0.591                          0.118

630                     0.999    0.557                          0.298
```

```
..                      ...     ...                          ...

777                   0.614   0.555                        0.411

778                   0.499   0.417                        0.147

779                   0.361   0.000                        0.025

780                   0.105   0.225                        0.035

781                   0.295   0.010                        0.091


     Generosity  Year
626       0.153  2019
627       0.252  2019
628       0.271  2019
629       0.354  2019
630       0.322  2019
..          ...   ...
777       0.217  2019
778       0.276  2019
779       0.158  2019
780       0.235  2019
781       0.202  2019

[156 rows x 12 columns]
```

```
predictedRankings2['Happiness Rank'] = predictedRankings2['Happiness
Score'].rank(method='max', ascending=False)
```

```
predictedRankings2
```

```
                     Country             Region  Happiness Rank  \
626                  Finland      Western Europe            11.0
627                  Denmark      Western Europe             3.0
628                   Norway      Western Europe             1.0
629                  Iceland      Western Europe            13.0
630              Netherlands      Western Europe             9.0
..                       ...                 ...             ...
777                   Rwanda  Sub-Saharan Africa            98.0
778                 Tanzania  Sub-Saharan Africa           119.0
779              Afghanistan       Southern Asia           156.0
780  Central African Republic  Sub-Saharan Africa           154.0
781              South Sudan  Sub-Saharan Africa           155.0

     Happiness Score  Standard Error  Economy (GDP per Capita)
Family  \
626         6.666243             0.0                     1.340
```

```
1.587000
627         6.835956            0.0                     1.383
1.573000
628         6.888524            0.0                     1.488
1.582000
629         6.625989            0.0                     1.380
1.624000
630         6.710150            0.0                     1.396
1.522000
..               ...            ...                       ...       .
..
777         5.081384            0.0                     0.359
0.711000
778         4.679890            0.0                     0.476
0.885000
779         3.445801            0.0                     0.350
0.517000
780         3.610647            0.0                     0.026
1.085332
781         3.523566            0.0                     0.306
0.575000

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)
\
626                     0.986    0.596                          0.393

627                     0.996    0.592                          0.410

628                     1.028    0.603                          0.341

629                     1.026    0.591                          0.118

630                     0.999    0.557                          0.298

..                        ...      ...                            ...

777                     0.614    0.555                          0.411

778                     0.499    0.417                          0.147

779                     0.361    0.000                          0.025

780                     0.105    0.225                          0.035

781                     0.295    0.010                          0.091


     Generosity  Year
626       0.153  2019
```

```
627       0.252  2019
628       0.271  2019
629       0.354  2019
630       0.322  2019
..          ...   ...
777       0.217  2019
778       0.276  2019
779       0.158  2019
780       0.235  2019
781       0.202  2019

[156 rows x 12 columns]
```

Model 3: Robust Regresssion (RANdom SAmple Consensus (RANSAC))

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from math import sqrt

# seperate into training and testing sets
testing3 = happy.loc[happy['Year'] == 2019]
training3 = happy.loc[happy['Year'].isin([2015, 2016, 2017, 2018])]

target_column3 = ['Happiness Score']
not_used3 =  ['Country', 'Region', 'Happiness Rank', 'Year', 'Standard
Error']
excluded3 = target_column3 + not_used3
excluded3
```

```
['Happiness Score',
 'Country',
 'Region',
 'Happiness Rank',
 'Year',
 'Standard Error']
```

```
predictors3 = list(set(list(training3.columns))-set(excluded3))

X3 = training3[predictors3].values
Y3 = training3[target_column3].values

X3_train, X3_test, Y3_train, Y3_test = train_test_split(X3, Y3,
test_size=0.30, random_state=40)

print(X3_train.shape); print(X3_test.shape)
```

```
(438, 6)
(188, 6)
```

```
from sklearn.linear_model import RANSACRegressor

ransac = RANSACRegressor(LinearRegression(),
           max_trials=100,       # Number of Iterations
```

```
            min_samples=20,       # Minimum size of the sample
            loss='absolute_error',     # Metrics for loss
#absolute_loss was before
            residual_threshold=10      # Threshold
            )

# Train model
ransac.fit(X3_train, Y3_train)

pred_train_ransac = ransac.predict(testing3.drop(excluded3, axis =
1).values)

rms4 = mean_squared_error(testing3['Happiness Score'],
pred_train_ransac, squared=False)
print(rms4)
```

```
0.5811039449579765
```

Formula to calculate Happiness score

```
ridgeReg.coef_
```

```
array([[0.62788957, 0.62607593, 1.28876482, 1.04780362, 1.37226637,
        1.2066021 ]])
```

```
ridgeReg.intercept_
```

```
array([2.21216454])
```

Our formula for Happiness Score based on Ridge Regression is:

$$HappinessScore = (0.62788957 * Economy) + (0.62607593 * Family) + (1.28876482 * Health) + (1.04780$$