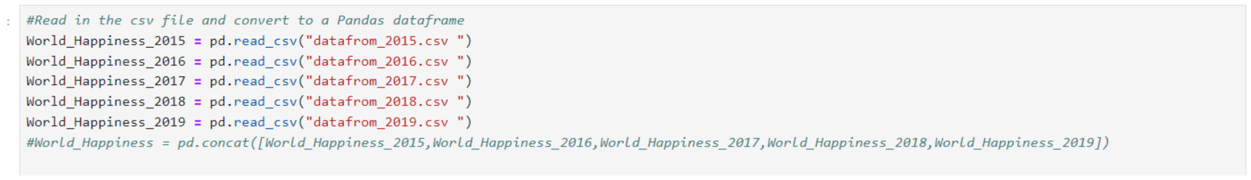
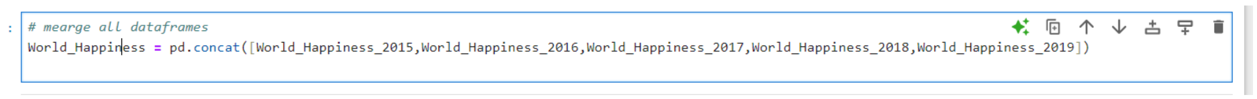
Usecase :

Loading the data :



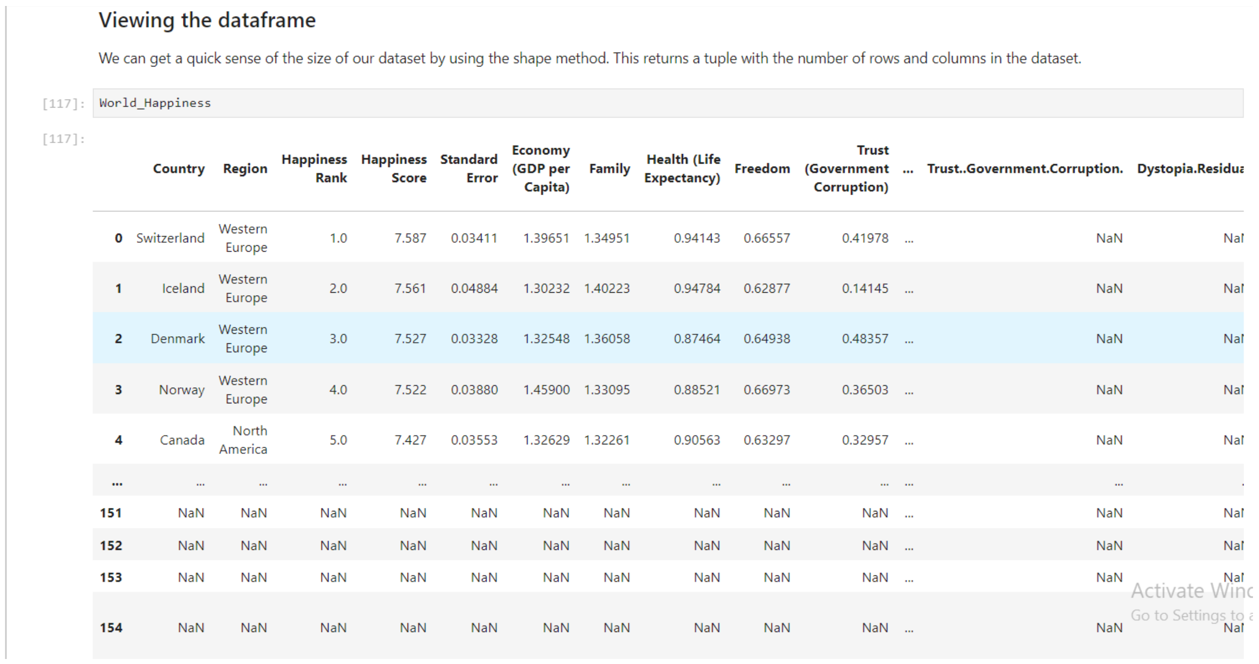
```
#Read in the csv file and convert to a Pandas dataframe
World_Happiness_2015 = pd.read_csv("datafrom_2015.csv ")
World_Happiness_2016 = pd.read_csv("datafrom_2016.csv ")
World_Happiness_2017 = pd.read_csv("datafrom_2017.csv ")
World_Happiness_2018 = pd.read_csv("datafrom_2018.csv ")
World_Happiness_2019 = pd.read_csv("datafrom_2019.csv ")
#World_Happiness = pd.concat([World_Happiness_2015,World_Happiness_2016,World_Happiness_2017,World_Happiness_2018,World_Happiness_2019])
```

merge all datagrams :



```
# mearge all dataframes
World_Happiness = pd.concat([World_Happiness_2015,World_Happiness_2016,World_Happiness_2017,World_Happiness_2018,World_Happiness_2019])
```

Viewing the dataframe :



Viewing the dataframe

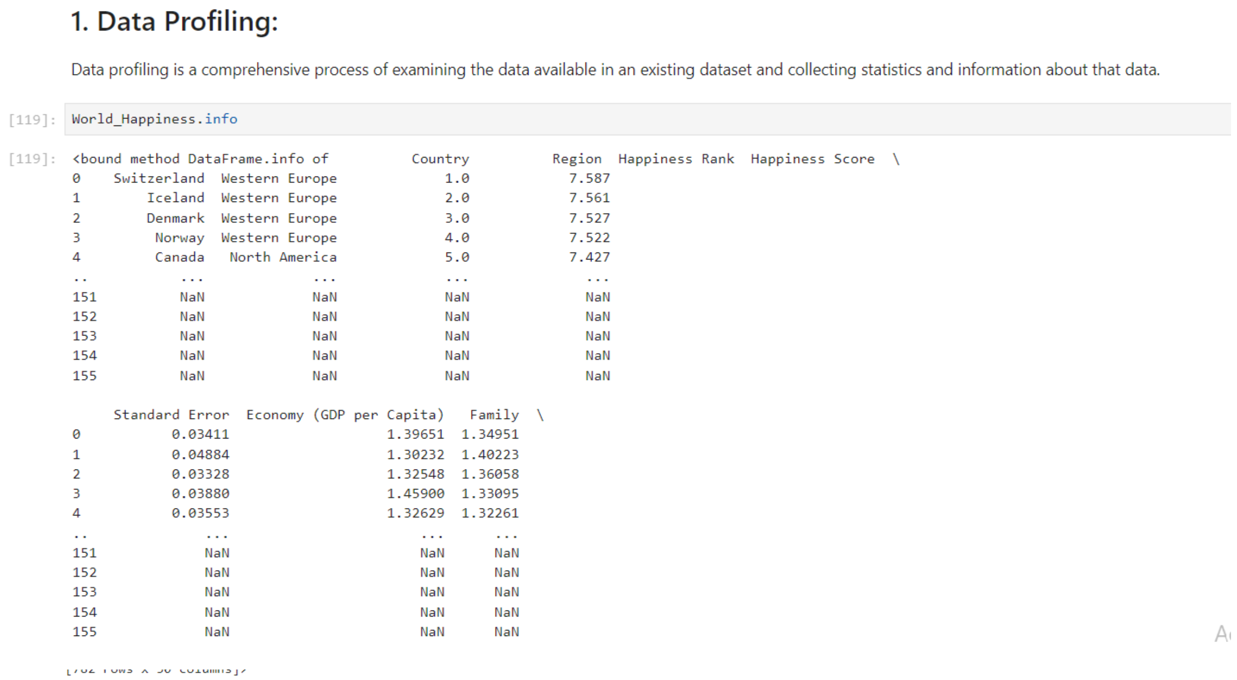
We can get a quick sense of the size of our dataset by using the shape method. This returns a tuple with the number of rows and columns in the dataset.

```
[117]: World_Happiness
```

[117]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	...	Trust..Government.Corruption.	Dystopia.Residua
0	Switzerland	Western Europe	1.0	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	...	NaN	Nal
1	Iceland	Western Europe	2.0	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	...	NaN	Nal
2	Denmark	Western Europe	3.0	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	...	NaN	Nal
3	Norway	Western Europe	4.0	7.522	0.03880	1.45900	1.33095	0.88521	0.66973	0.36503	...	NaN	Nal
4	Canada	North America	5.0	7.427	0.03553	1.32629	1.32261	0.90563	0.63297	0.32957	...	NaN	Nal
...	...	...	...	...	...	...	...	...	...	...	...	...	.
151	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	Nal
152	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	Nal
153	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	Nal
154	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	Nal

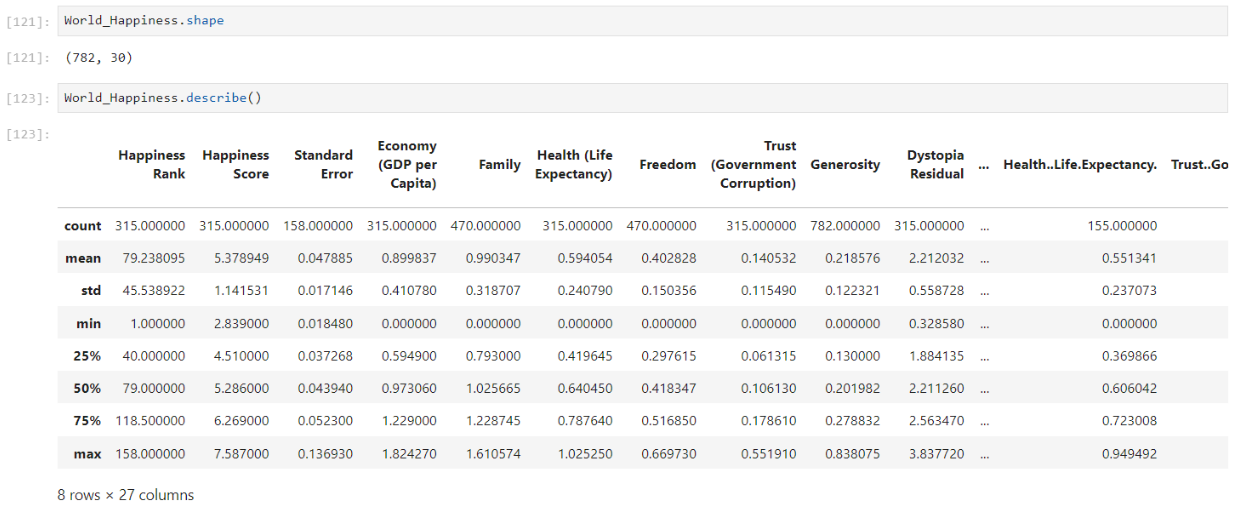
Data Profiling before do consistency processes:



1. Data Profiling:

Data profiling is a comprehensive process of examining the data available in an existing dataset and collecting statistics and information about that data.

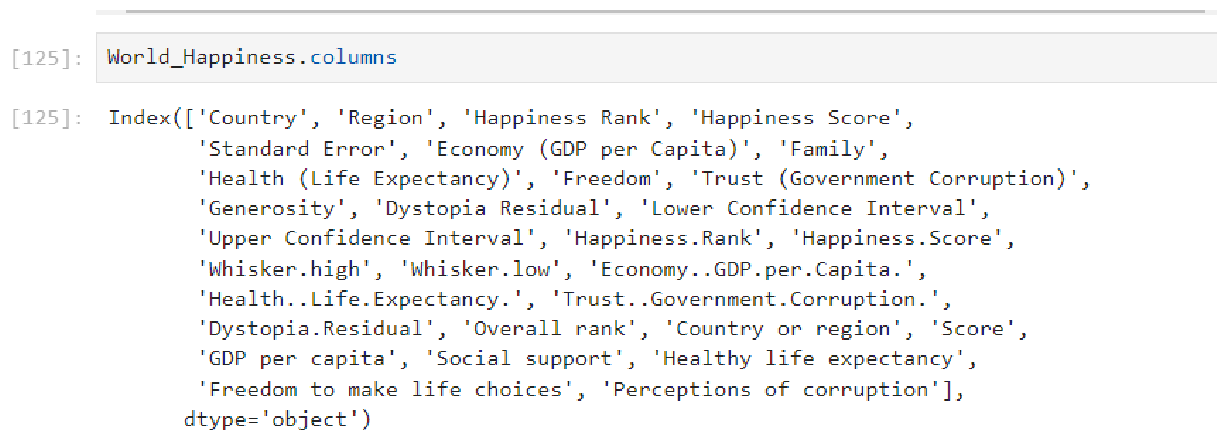
```
[119]: World_Happiness.info

[119]: <bound method DataFrame.info of          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

       [782 rows x 30 columns]>
```



```
[121]: World_Happiness.shape

[121]: (782, 30)

[123]: World_Happiness.describe()

[123]:
```

	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual	...	Health..Life.Expectancy.	Trust..Go
count	315.000000	315.000000	158.000000	315.000000	470.000000	315.000000	470.000000	315.000000	782.000000	315.000000	...	155.000000	
mean	79.238095	5.378949	0.047885	0.899837	0.990347	0.594054	0.402828	0.140532	0.218576	2.212032	...	0.551341	
std	45.538922	1.141531	0.017146	0.410780	0.318707	0.240790	0.150356	0.115490	0.122321	0.558728	...	0.237073	
min	1.000000	2.839000	0.018480	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.328580	...	0.000000	
25%	40.000000	4.510000	0.037268	0.594900	0.793000	0.419645	0.297615	0.061315	0.130000	1.884135	...	0.369866	
50%	79.000000	5.286000	0.043940	0.973060	1.025665	0.640450	0.418347	0.106130	0.201982	2.211260	...	0.606042	
75%	118.500000	6.269000	0.052300	1.229000	1.228745	0.787640	0.516850	0.178610	0.278832	2.563470	...	0.723008	
max	158.000000	7.587000	0.136930	1.824270	1.610574	1.025250	0.669730	0.551910	0.838075	3.837720	...	0.949492	

8 rows × 27 columns

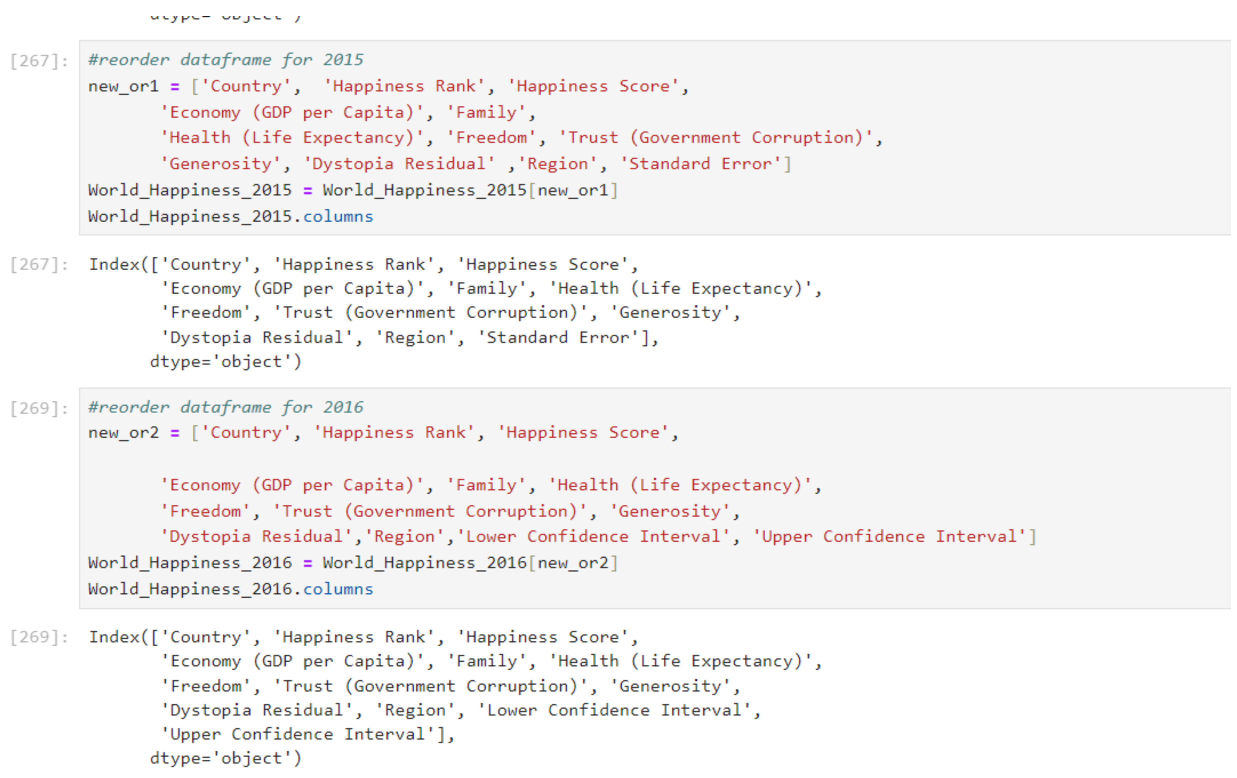


```
[125]: World_Happiness.columns

[125]: Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
              'Standard Error', 'Economy (GDP per Capita)', 'Family',
              'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
              'Generosity', 'Dystopia Residual', 'Lower Confidence Interval',
              'Upper Confidence Interval', 'Happiness.Rank', 'Happiness.Score',
              'Whisker.high', 'Whisker.low', 'Economy..GDP.per.Capita.',
              'Health..Life.Expectancy.', 'Trust..Government.Corruption.',
              'Dystopia.Residual', 'Overall rank', 'Country or region', 'Score',
              'GDP per capita', 'Social support', 'Healthy life expectancy',
              'Freedom to make life choices', 'Perceptions of corruption'],
             dtype='object')
```

When I merged the data frames, I found that the order of the columns did not match, and their names did not match. Now I will perform some operations to correct these problems and make the data consistent.

1.Reorder the columns :

Dataframs 2015 & 2016:



```
        dtype='object')
```

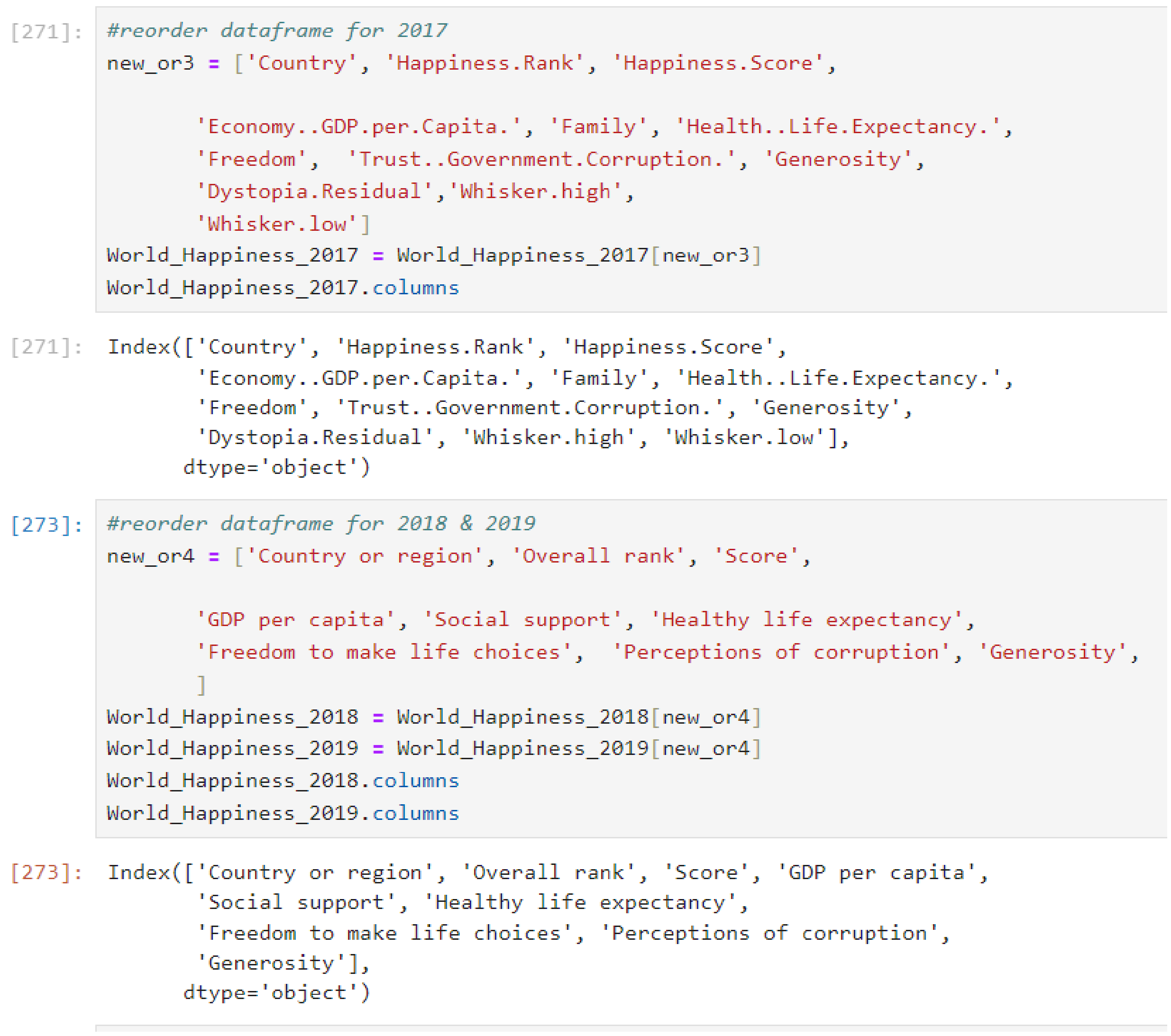
```
[267]: #reorder dataframe for 2015
       new_or1 = ['Country',  'Happiness Rank', 'Happiness Score',
              'Economy (GDP per Capita)', 'Family',
              'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
              'Generosity', 'Dystopia Residual' ,'Region', 'Standard Error']
       World_Happiness_2015 = World_Happiness_2015[new_or1]
       World_Happiness_2015.columns
```

```
[267]: Index(['Country', 'Happiness Rank', 'Happiness Score',
              'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)',
              'Freedom', 'Trust (Government Corruption)', 'Generosity',
              'Dystopia Residual', 'Region', 'Standard Error'],
             dtype='object')
```

```
[269]: #reorder dataframe for 2016
       new_or2 = ['Country', 'Happiness Rank', 'Happiness Score',

              'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)',
              'Freedom', 'Trust (Government Corruption)', 'Generosity',
              'Dystopia Residual','Region','Lower Confidence Interval', 'Upper Confidence Interval']
       World_Happiness_2016 = World_Happiness_2016[new_or2]
       World_Happiness_2016.columns
```

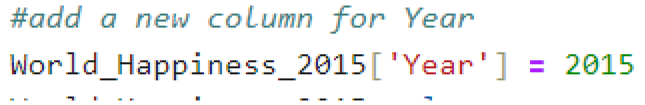
```
[269]: Index(['Country', 'Happiness Rank', 'Happiness Score',
              'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)',
              'Freedom', 'Trust (Government Corruption)', 'Generosity',
              'Dystopia Residual', 'Region', 'Lower Confidence Interval',
              'Upper Confidence Interval'],
             dtype='object')
```

Dataframes 2017&2018&2019 :

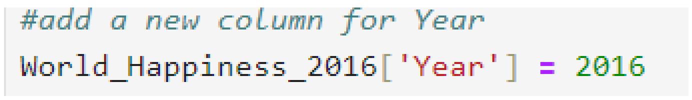


```
[271]: #reorder dataframe for 2017
       new_or3 = ['Country', 'Happiness.Rank', 'Happiness.Score',

              'Economy..GDP.per.Capita.', 'Family', 'Health..Life.Expectancy.',
              'Freedom',  'Trust..Government.Corruption.', 'Generosity',
              'Dystopia.Residual','Whisker.high',
              'Whisker.low']
       World_Happiness_2017 = World_Happiness_2017[new_or3]
       World_Happiness_2017.columns

[271]: Index(['Country', 'Happiness.Rank', 'Happiness.Score',
              'Economy..GDP.per.Capita.', 'Family', 'Health..Life.Expectancy.',
              'Freedom', 'Trust..Government.Corruption.', 'Generosity',
              'Dystopia.Residual', 'Whisker.high', 'Whisker.low'],
             dtype='object')

[273]: #reorder dataframe for 2018 & 2019
       new_or4 = ['Country or region', 'Overall rank', 'Score',

              'GDP per capita', 'Social support', 'Healthy life expectancy',
              'Freedom to make life choices',  'Perceptions of corruption', 'Generosity',
              ]
       World_Happiness_2018 = World_Happiness_2018[new_or4]
       World_Happiness_2019 = World_Happiness_2019[new_or4]
       World_Happiness_2018.columns
       World_Happiness_2019.columns

[273]: Index(['Country or region', 'Overall rank', 'Score', 'GDP per capita',
              'Social support', 'Healthy life expectancy',
              'Freedom to make life choices', 'Perceptions of corruption',
              'Generosity'],
             dtype='object')
```

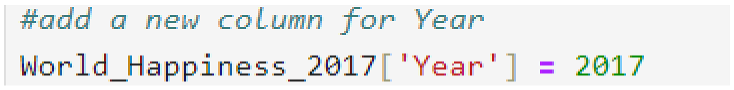
Add Year column for each datafram :



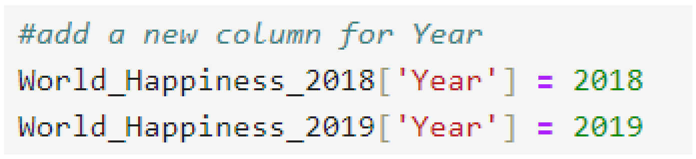
```
#add a new column for Year
World_Happiness_2015['Year'] = 2015
```



```
#add a new column for Year
World_Happiness_2016['Year'] = 2016
```



```
#add a new column for Year
World_Happiness_2017['Year'] = 2017
```



```
#add a new column for Year
World_Happiness_2018['Year'] = 2018
World_Happiness_2019['Year'] = 2019
```

Rename Columns :

```
[275]: #rename columns in dataframs
       #2015
       columns = {World_Happiness_2015.columns[i]: standard_columns_nameandorder[i] for i in range(len(standard_columns_nameandorder))}
       World_Happiness_2015.rename(columns=columns, inplace=True)
       #2016
       columns = {World_Happiness_2016.columns[i]: standard_columns_nameandorder[i] for i in range(len(standard_columns_nameandorder))}
       World_Happiness_2016.rename(columns=columns, inplace=True)
       #2017
       columns = {World_Happiness_2017.columns[i]: standard_columns_nameandorder[i] for i in range(len(standard_columns_nameandorder))}
       World_Happiness_2017.rename(columns=columns, inplace=True)
       #2018
       columns = {World_Happiness_2018.columns[i]: standard_columns_nameandorder[i] for i in range(len(standard_columns_nameandorder))}
       World_Happiness_2018.rename(columns=columns, inplace=True)
       #2019
       columns = {World_Happiness_2019.columns[i]: standard_columns_nameandorder[i] for i in range(len(standard_columns_nameandorder))}
       World_Happiness_2019.rename(columns=columns, inplace=True)
```

Merge dataframs again and view dataframe:

```
[263]: # new merge all data frame agin
       World_Happiness_v2 = pd.concat([World_Happiness_2015,World_Happiness_2016,World_Happiness_2017,World_Happiness_2018,World_Happiness_20
```

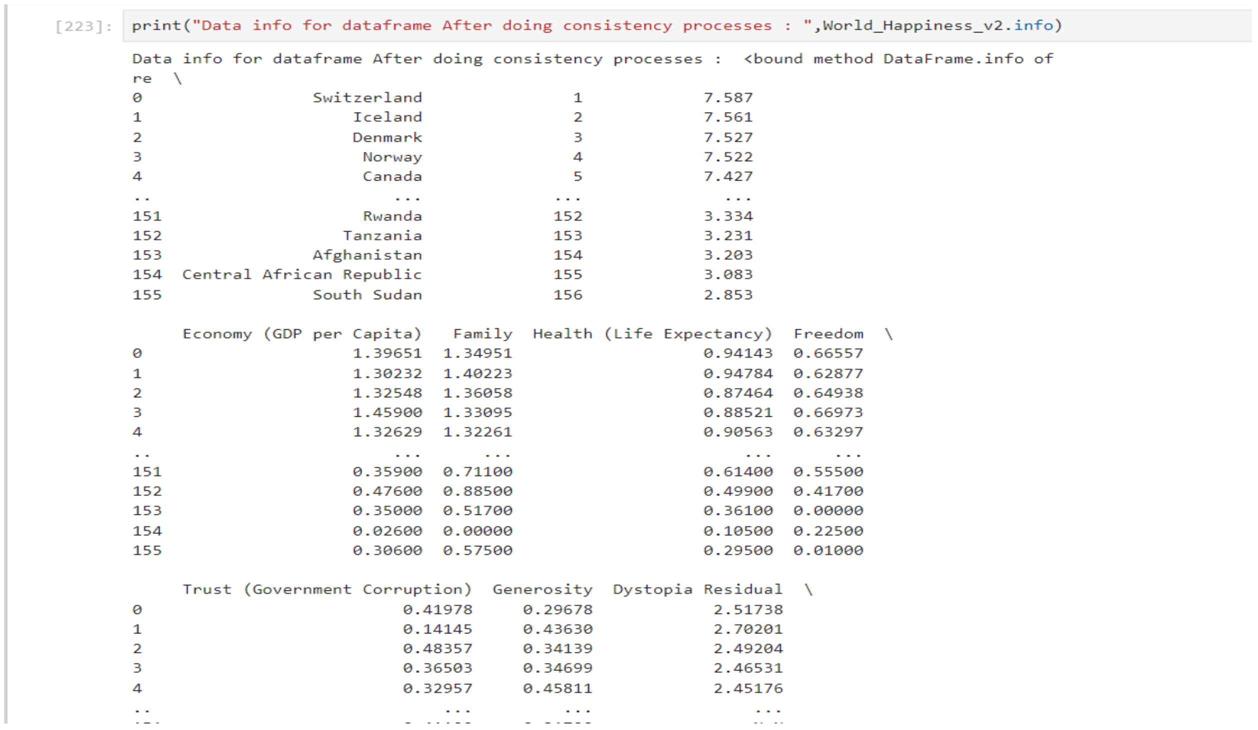
```
[265]: World_Happiness_v2
```

[265]:

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual	Region	Standard Error	Year
0	Switzerland	1	7.587	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2.51738	Western Europe	0.03411	2015
1	Iceland	2	7.561	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2.70201	Western Europe	0.04884	2015
2	Denmark	3	7.527	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.49204	Western Europe	0.03328	2015
3	Norway	4	7.522	1.45900	1.33095	0.88521	0.66973	0.36503	0.34699	2.46531	Western Europe	0.03880	2015
4	Canada	5	7.427	1.32629	1.32261	0.90563	0.63297	0.32957	0.45811	2.45176	North America	0.03553	2015
...	...	...	...	...	...	...	...	...	...	...	...	...	...
151	Rwanda	152	3.334	0.35900	0.71100	0.61400	0.55500	0.41100	0.21700	NaN	NaN	NaN	2019
152	Tanzania	153	3.231	0.47600	0.88500	0.49900	0.41700	0.14700	0.27600	NaN	NaN	NaN	2019
153	Afghanistan	154	3.203	0.35000	0.51700	0.36100	0.00000	0.02500	0.15800	NaN	NaN	NaN	2019
154	Central African Republic	155	3.083	0.02600	0.00000	0.10500	0.22500	0.03500	0.23500	NaN	NaN	NaN	2019
	South												

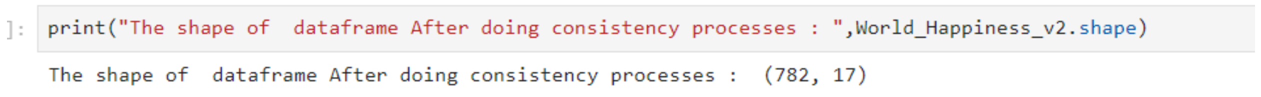
Profiling new dataframe :

Info :



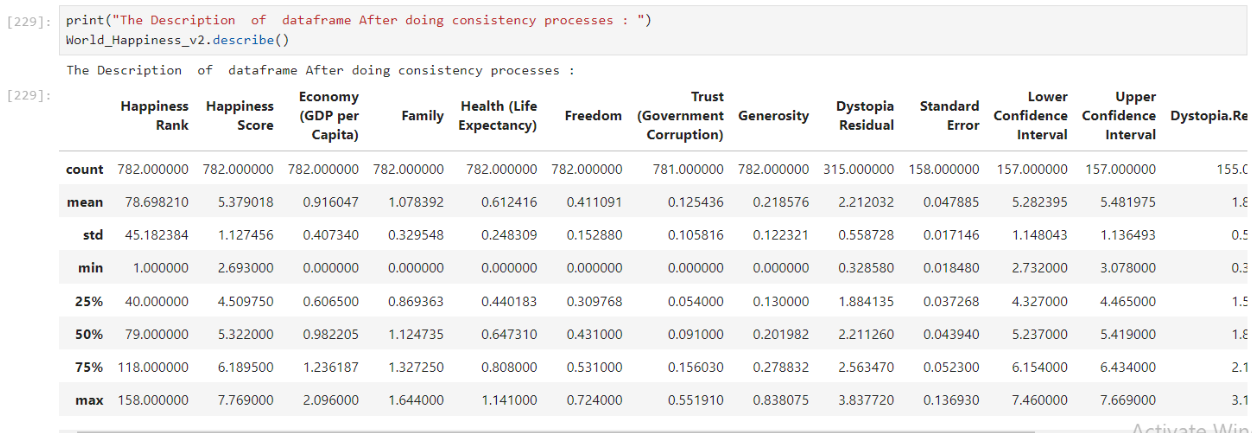
```
[223]: print("Data info for dataframe After doing consistency processes : ",World_Happiness_v2.info)

Data info for dataframe After doing consistency processes :  <bound method DataFrame.info of
re  \
0                 Switzerland            1          7.587
1                     Iceland            2          7.561
2                     Denmark            3          7.527
3                      Norway            4          7.522
4                      Canada            5          7.427
..                        ...          ...            ...
151                    Rwanda          152          3.334
152                  Tanzania          153          3.231
153               Afghanistan          154          3.203
154  Central African Republic          155          3.083
155               South Sudan          156          2.853

     Economy (GDP per Capita)  Family  Health (Life Expectancy)  Freedom  \
0                     1.39651 1.34951                   0.94143  0.66557
1                     1.30232 1.40223                   0.94784  0.62877
2                     1.32548 1.36058                   0.87464  0.64938
3                     1.45900 1.33095                   0.88521  0.66973
4                     1.32629 1.32261                   0.90563  0.63297
..                        ...     ...                       ...      ...
151                   0.35900 0.71100                   0.61400  0.55500
152                   0.47600 0.88500                   0.49900  0.41700
153                   0.35000 0.51700                   0.36100  0.00000
154                   0.02600 0.00000                   0.10500  0.22500
155                   0.30600 0.57500                   0.29500  0.01000

     Trust (Government Corruption)  Generosity  Dystopia Residual  \
0                          0.41978     0.29678            2.51738
1                          0.14145     0.43630            2.70201
2                          0.48357     0.34139            2.49204
3                          0.36503     0.34699            2.46531
4                          0.32957     0.45811            2.45176
..                             ...         ...                ...
```

Shape :



```
]: print("The shape of  dataframe After doing consistency processes : ",World_Happiness_v2.shape)

The shape of  dataframe After doing consistency processes :  (782, 17)
```

Description:



```
[229]: print("The Description  of  dataframe After doing consistency processes : ")
       World_Happiness_v2.describe()

The Description  of  dataframe After doing consistency processes :
```

[229]:

	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual	Standard Error	Lower Confidence Interval	Upper Confidence Interval	Dystopia.Re
count	782.000000	782.000000	782.000000	782.000000	782.000000	782.000000	781.000000	782.000000	315.000000	158.000000	157.000000	157.000000	155.C
mean	78.698210	5.379018	0.916047	1.078392	0.612416	0.411091	0.125436	0.218576	2.212032	0.047885	5.282395	5.481975	1.8
std	45.182384	1.127456	0.407340	0.329548	0.248309	0.152880	0.105816	0.122321	0.558728	0.017146	1.148043	1.136493	0.5
min	1.000000	2.693000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.328580	0.018480	2.732000	3.078000	0.3
25%	40.000000	4.509750	0.606500	0.869363	0.440183	0.309768	0.054000	0.130000	1.884135	0.037268	4.327000	4.465000	1.5
50%	79.000000	5.322000	0.982205	1.124735	0.647310	0.431000	0.091000	0.201982	2.211260	0.043940	5.237000	5.419000	1.8
75%	118.000000	6.189500	1.236187	1.327250	0.808000	0.531000	0.156030	0.278832	2.563470	0.052300	6.154000	6.434000	2.1
max	158.000000	7.769000	2.096000	1.644000	1.141000	0.724000	0.551910	0.838075	3.837720	0.136930	7.460000	7.669000	3.1

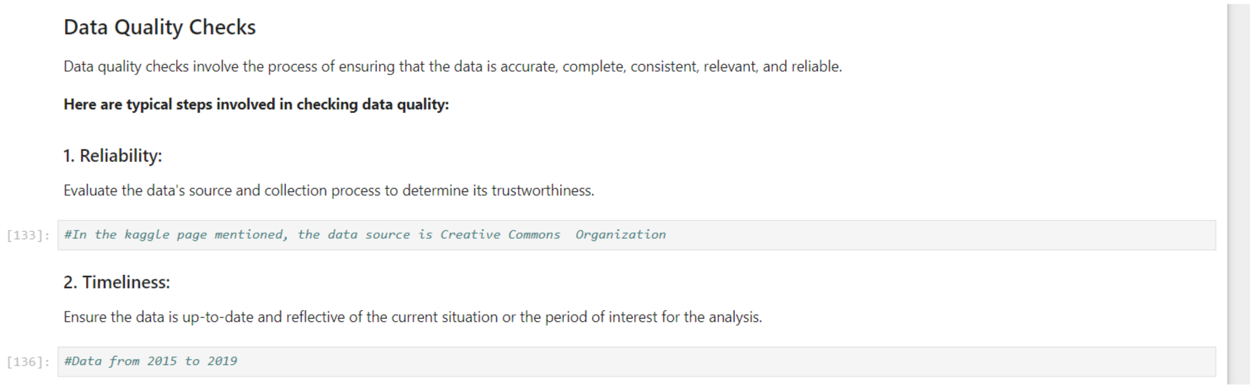
Columns :

```
[231]: print("The Columns  of  dataframe After doing consistency processes : ")
       World_Happiness_v2.columns
```

```
The Columns  of  dataframe After doing consistency processes :
```

```
[231]: Index(['Country', 'Happiness Rank', 'Happiness Score',
              'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)',
              'Freedom', 'Trust (Government Corruption)', 'Generosity',
              'Dystopia Residual', 'Region', 'Standard Error',
              'Lower Confidence Interval', 'Upper Confidence Interval',
              'Dystopia.Residual', 'Whisker.high', 'Whisker.low'],
             dtype='object')
```

Data Quality Checks :



Data Quality Checks

Data quality checks involve the process of ensuring that the data is accurate, complete, consistent, relevant, and reliable.

Here are typical steps involved in checking data quality:

1. Reliability:

Evaluate the data's source and collection process to determine its trustworthiness.

```
[133]: #In the kaggle page mentioned, the data source is Creative Commons  Organization
```

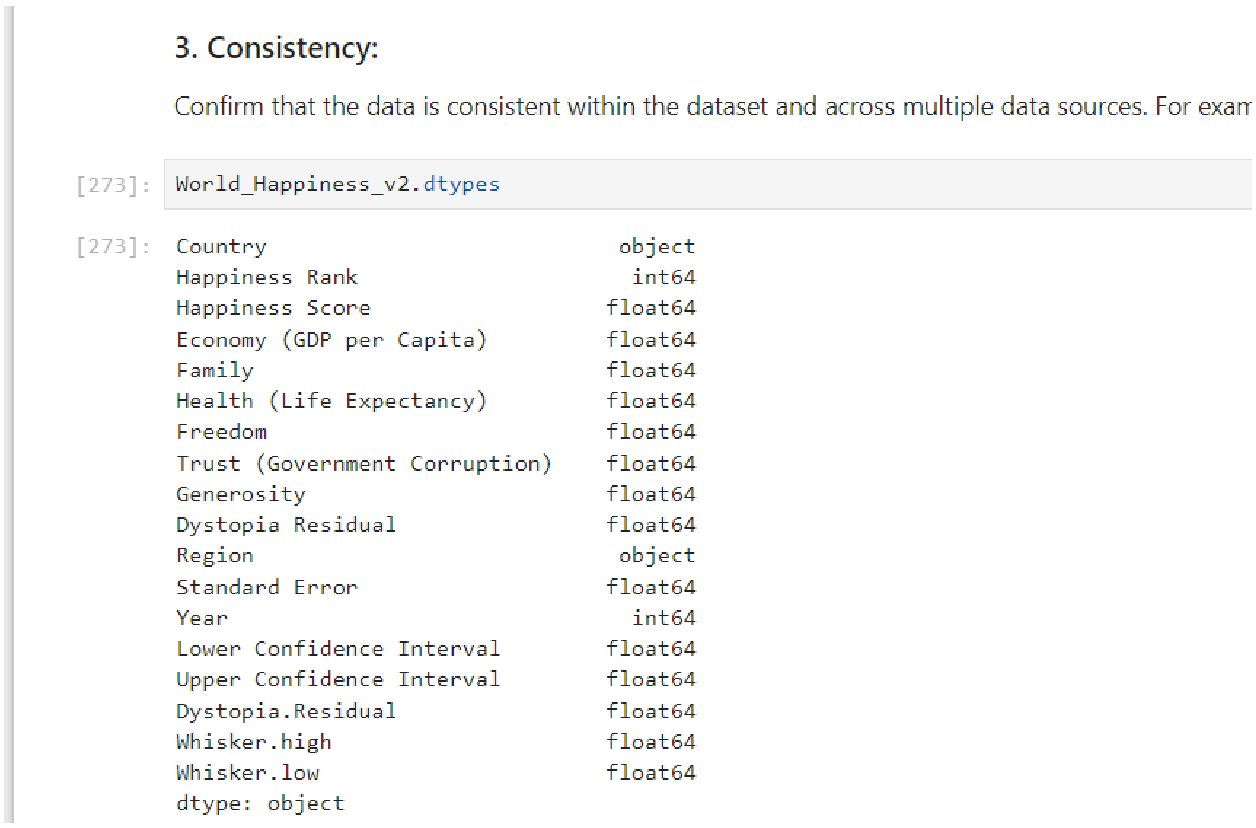
2. Timeliness:

Ensure the data is up-to-date and reflective of the current situation or the period of interest for the analysis.

```
[136]: #Data from 2015 to 2019
```

Consistency :

Check the data type :



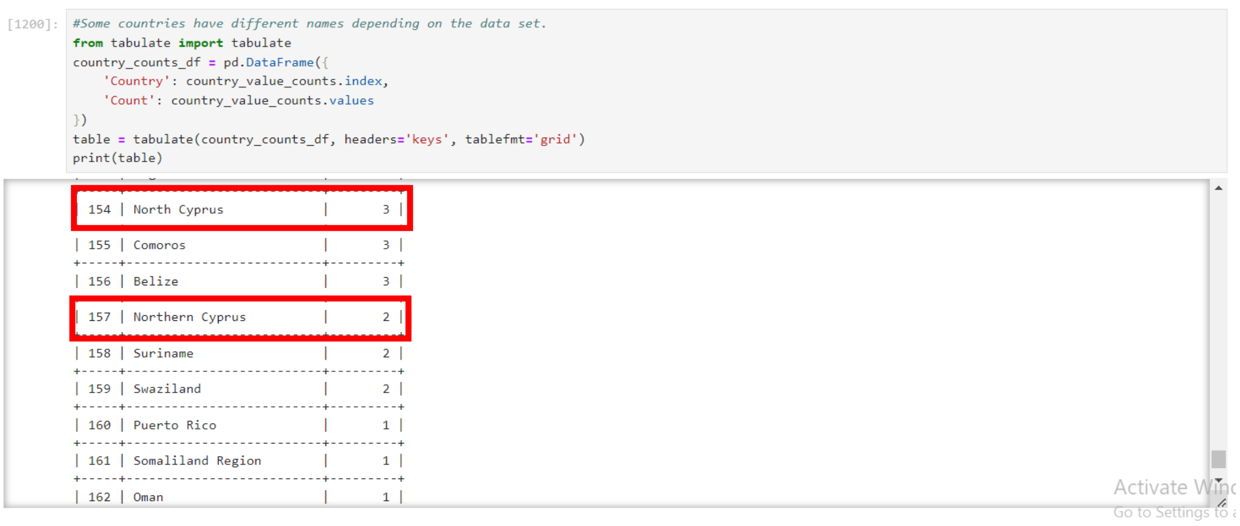
3. Consistency:

Confirm that the data is consistent within the dataset and across multiple data sources. For exam

```
[273]: World_Happiness_v2.dtypes
```

```
[273]: Country                              object
       Happiness Rank                        int64
       Happiness Score                     float64
       Economy (GDP per Capita)            float64
       Family                              float64
       Health (Life Expectancy)            float64
       Freedom                             float64
       Trust (Government Corruption)       float64
       Generosity                          float64
       Dystopia Residual                   float64
       Region                               object
       Standard Error                      float64
       Year                                  int64
       Lower Confidence Interval           float64
       Upper Confidence Interval           float64
       Dystopia.Residual                   float64
       Whisker.high                        float64
       Whisker.low                         float64
       dtype: object
```

Some countries have different names depending on the data set.



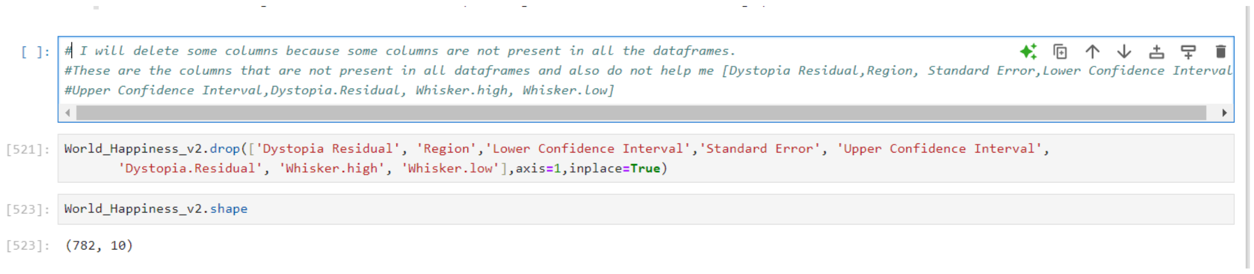
```
[1200]: #Some countries have different names depending on the data set.
from tabulate import tabulate
country_counts_df = pd.DataFrame({
    'Country': country_value_counts.index,
    'Count': country_value_counts.values
})
table = tabulate(country_counts_df, headers='keys', tablefmt='grid')
print(table)
```

```
| 154 | North Cyprus              |       3 |

| 155 | Comoros                   |       3 |
+-----+---------------------------+---------+
| 156 | Belize                    |       3 |

| 157 | Northern Cyprus           |       2 |

| 158 | Suriname                  |       2 |
+-----+---------------------------+---------+
| 159 | Swaziland                 |       2 |
+-----+---------------------------+---------+
| 160 | Puerto Rico               |       1 |
+-----+---------------------------+---------+
| 161 | Somaliland Region         |       1 |
+-----+---------------------------+---------+
| 162 | Oman                      |       1 |
```

4. Relevance:

Next Day working :

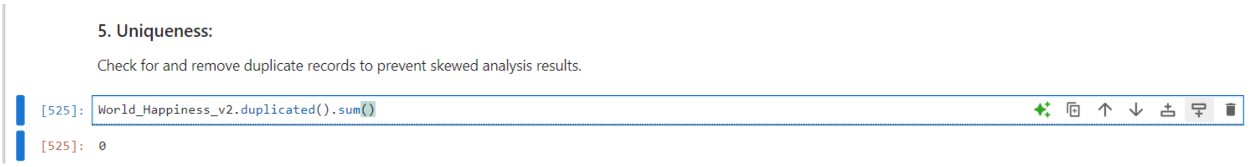
4. Relevance:



```
[ ]:   # I will delete some columns because some columns are not present in all the dataframes.
       #These are the columns that are not present in all dataframes and also do not help me [Dystopia Residual,Region, Standard Error,Lower Confidence Interval
       #Upper Confidence Interval,Dystopia.Residual, Whisker.high, Whisker.low]

[521]: World_Happiness_v2.drop(['Dystopia Residual', 'Region','Lower Confidence Interval','Standard Error', 'Upper Confidence Interval',
               'Dystopia.Residual', 'Whisker.high', 'Whisker.low'],axis=1,inplace=True)

[523]: World_Happiness_v2.shape

[523]: (782, 10)
```

5. Uniqueness:

The data has zero duplicate.



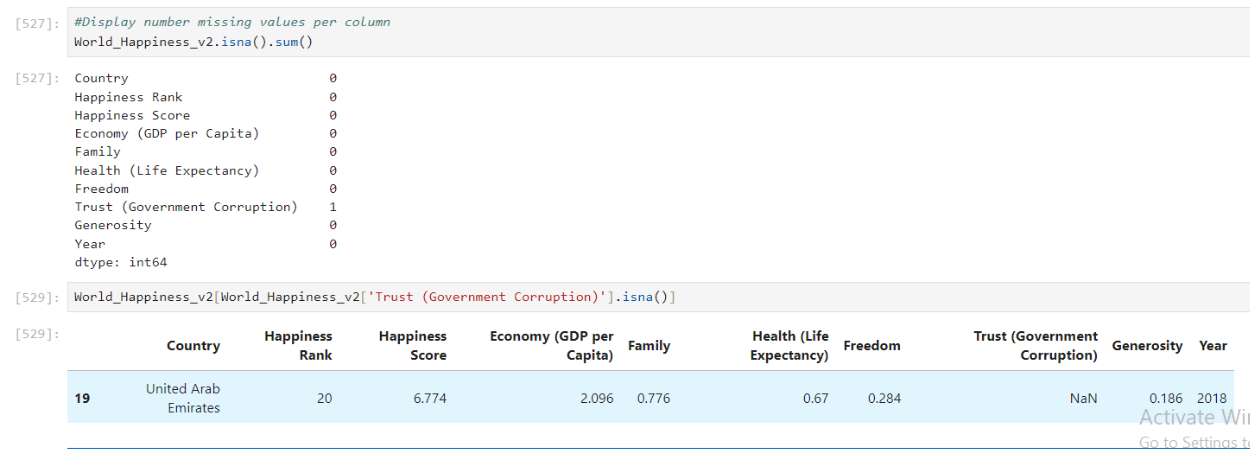
5. Uniqueness:

Check for and remove duplicate records to prevent skewed analysis results.

```
[525]: World_Happiness_v2.duplicated().sum()

[525]: 0
```

6.Completeness:

One null value in "Trust (Government Corruption)" column.



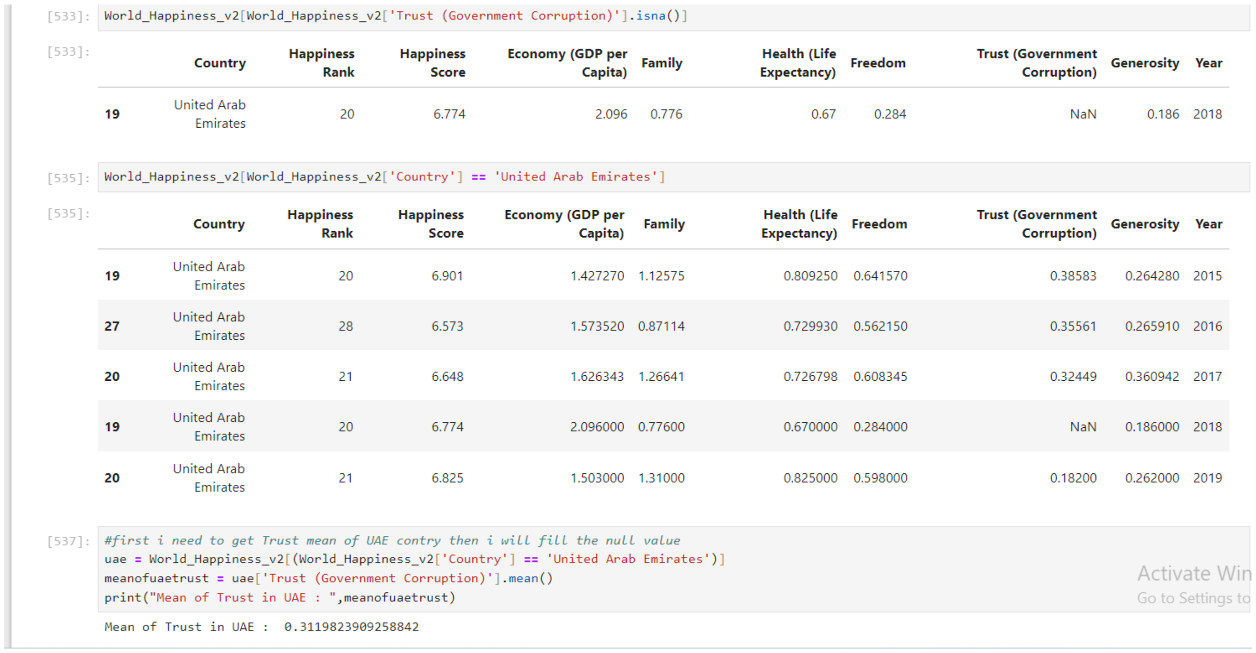
```
[527]: #Display number missing values per column
       World_Happiness_v2.isna().sum()

[527]: Country                          0
       Happiness Rank                   0
       Happiness Score                  0
       Economy (GDP per Capita)         0
       Family                           0
       Health (Life Expectancy)         0
       Freedom                          0
       Trust (Government Corruption)    1
       Generosity                       0
       Year                             0
       dtype: int64

[529]: World_Happiness_v2[World_Happiness_v2['Trust (Government Corruption)'].isna()]

[529]:
```

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Year
19	United Arab Emirates	20	6.774	2.096	0.776	0.67	0.284	NaN	0.186	2018

Data Cleaning

Handling missing values:

Here the missing value was the trust in the government for the UAE in 2018. I took the mean of the trust in the government for all years for the UAE and then compensated for it in the missing value.



```
[533]: World_Happiness_v2[World_Happiness_v2['Trust (Government Corruption)'].isna()]
```

[533]:

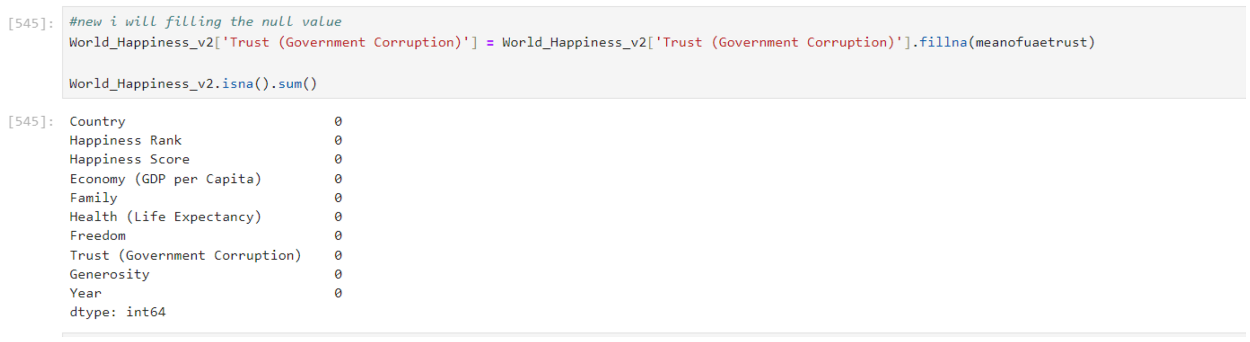
	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Year
19	United Arab Emirates	20	6.774	2.096	0.776	0.67	0.284	NaN	0.186	2018

```
[535]: World_Happiness_v2[World_Happiness_v2['Country'] == 'United Arab Emirates']
```

[535]:

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Year
19	United Arab Emirates	20	6.901	1.427270	1.12575	0.809250	0.641570	0.38583	0.264280	2015
27	United Arab Emirates	28	6.573	1.573520	0.87114	0.729930	0.562150	0.35561	0.265910	2016
20	United Arab Emirates	21	6.648	1.626343	1.26641	0.726798	0.608345	0.32449	0.360942	2017
19	United Arab Emirates	20	6.774	2.096000	0.77600	0.670000	0.284000	NaN	0.186000	2018
20	United Arab Emirates	21	6.825	1.503000	1.31000	0.825000	0.598000	0.18200	0.262000	2019

```
[537]: #first i need to get Trust mean of UAE contry then i will fill the null value
       uae = World_Happiness_v2[(World_Happiness_v2['Country'] == 'United Arab Emirates')]
       meanofuaetrust = uae['Trust (Government Corruption)'].mean()
       print("Mean of Trust in UAE : ",meanofuaetrust)

       Mean of Trust in UAE :  0.3119823909258842
```



```
[545]: #new i will filling the null value
       World_Happiness_v2['Trust (Government Corruption)'] = World_Happiness_v2['Trust (Government Corruption)'].fillna(meanofuaetrust)

       World_Happiness_v2.isna().sum()

[545]: Country                          0
       Happiness Rank                   0
       Happiness Score                  0
       Economy (GDP per Capita)         0
       Family                           0
       Health (Life Expectancy)         0
       Freedom                          0
       Trust (Government Corruption)    0
       Generosity                       0
       Year                             0
       dtype: int64
```

missing data in Region column

```
[198]: #missing data in Region column
       World_Happiness_v2[World_Happiness_v2['Region'].isna()]
```

[198]:

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Region	Year
0	Norway	1	7.537	1.616463	1.533524	0.796667	0.635423	0.315964	0.362012	NaN	2017
1	Denmark	2	7.522	1.482383	1.551122	0.792566	0.626007	0.400770	0.355280	NaN	2017
2	Iceland	3	7.504	1.480633	1.610574	0.833552	0.627163	0.153527	0.475540	NaN	2017
3	Switzerland	4	7.494	1.564980	1.516912	0.858131	0.620071	0.367007	0.290549	NaN	2017
4	Finland	5	7.469	1.443572	1.540247	0.809158	0.617951	0.382612	0.245483	NaN	2017
...	...	...	...	...	...	...	...	...	...	...	...
151	Rwanda	152	3.334	0.359000	0.711000	0.614000	0.555000	0.411000	0.217000	NaN	2019
152	Tanzania	153	3.231	0.476000	0.885000	0.499000	0.417000	0.147000	0.276000	NaN	2019
153	Afghanistan	154	3.203	0.350000	0.517000	0.361000	0.000000	0.025000	0.158000	NaN	2019
154	Central African Republic	155	3.083	0.026000	0.000000	0.105000	0.225000	0.035000	0.235000	NaN	2019
155	South Sudan	156	2.853	0.306000	0.575000	0.295000	0.010000	0.091000	0.202000	NaN	2019

Now I will fill in the blank values by taking the region values from countries with similar names.

```
#Now I will fill in the blank values by taking the region values from countries with similar names.
World_Happiness_v2['Region'] = World_Happiness_v2.groupby('Country')['Region'].transform(lambda x: x.fillna(method='bfill').fillna(method='ffill'))
```

Region column after fill

```
[266]: World_Happiness_v2[World_Happiness_v2['Region'].isna()]
```

[266]:

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Region	Year
119	Gambia	120	4.516	0.308	0.939	0.428	0.382	0.167	0.269	NaN	2019

Now I will fill Gambia by make a search and fill it

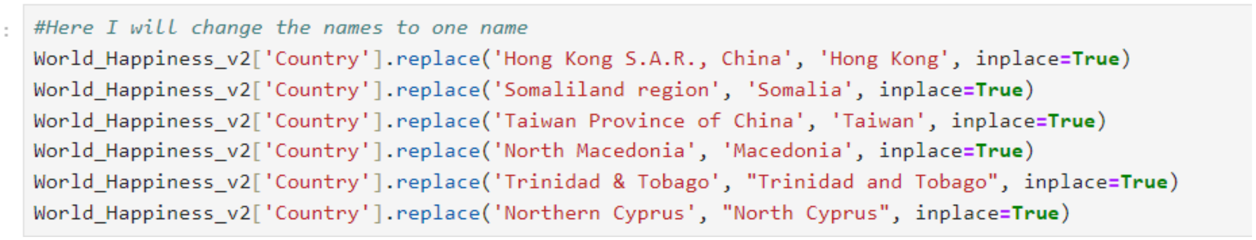
```
[276]: World_Happiness_v2.loc[(World_Happiness_v2['Country'] == 'Gambia') & (World_Happiness_v2['Region'].isna()), 'Region'] = 'West Africa'

[278]: World_Happiness_v2[World_Happiness_v2['Country'] == 'Gambia']
```

[278]:

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Region	Year
119	Gambia	120	4.516	0.308	0.939	0.428	0.382	0.167	0.269	West Africa	2019

Correcting errors

Change the value name:

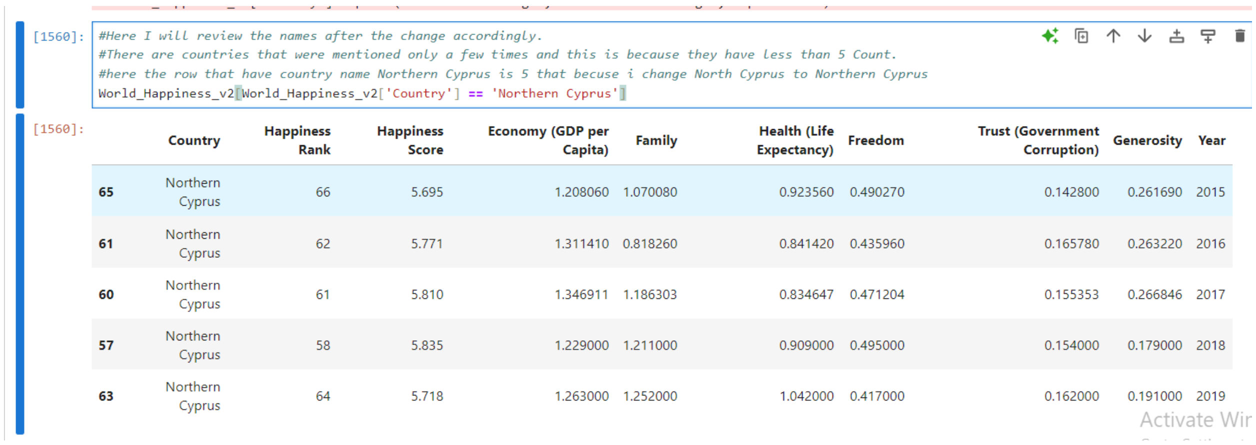


```
#Here I will change the names to one name
World_Happiness_v2['Country'].replace('Hong Kong S.A.R., China', 'Hong Kong', inplace=True)
World_Happiness_v2['Country'].replace('Somaliland region', 'Somalia', inplace=True)
World_Happiness_v2['Country'].replace('Taiwan Province of China', 'Taiwan', inplace=True)
World_Happiness_v2['Country'].replace('North Macedonia', 'Macedonia', inplace=True)
World_Happiness_v2['Country'].replace('Trinidad & Tobago', "Trinidad and Tobago", inplace=True)
World_Happiness_v2['Country'].replace('Northern Cyprus', "North Cyprus", inplace=True)
```

After make change :

There are countries that were mentioned only a few times and this is because they have less than 5 Count.

#here the row that have country name Northern Cyprus is 5 that because I changed North Cyprus to Northern Cyprus



```
[1560]: #Here I will review the names after the change accordingly.
        #There are countries that were mentioned only a few times and this is because they have less than 5 Count.
        #here the row that have country name Northern Cyprus is 5 that becuse i change North Cyprus to Northern Cyprus
        World_Happiness_v2[World_Happiness_v2['Country'] == 'Northern Cyprus']
```

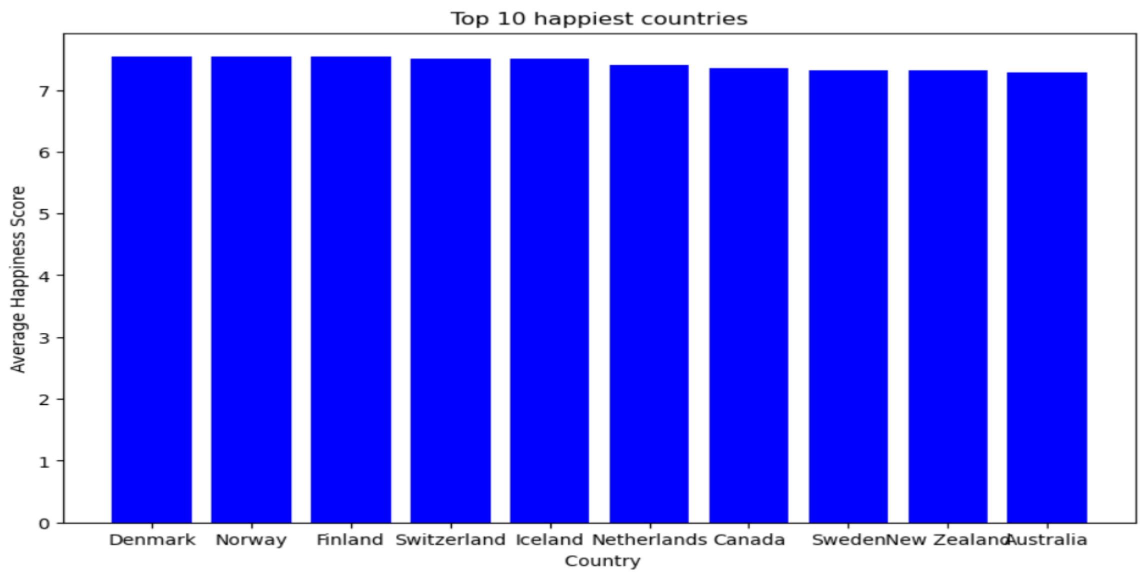
[1560]:

	Country	Happiness Rank	Happiness Score	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Year
65	Northern Cyprus	66	5.695	1.208060	1.070080	0.923560	0.490270	0.142800	0.261690	2015
61	Northern Cyprus	62	5.771	1.311410	0.818260	0.841420	0.435960	0.165780	0.263220	2016
60	Northern Cyprus	61	5.810	1.346911	1.186303	0.834647	0.471204	0.155353	0.266846	2017
57	Northern Cyprus	58	5.835	1.229000	1.211000	0.909000	0.495000	0.154000	0.179000	2018
63	Northern Cyprus	64	5.718	1.263000	1.252000	1.042000	0.417000	0.162000	0.191000	2019

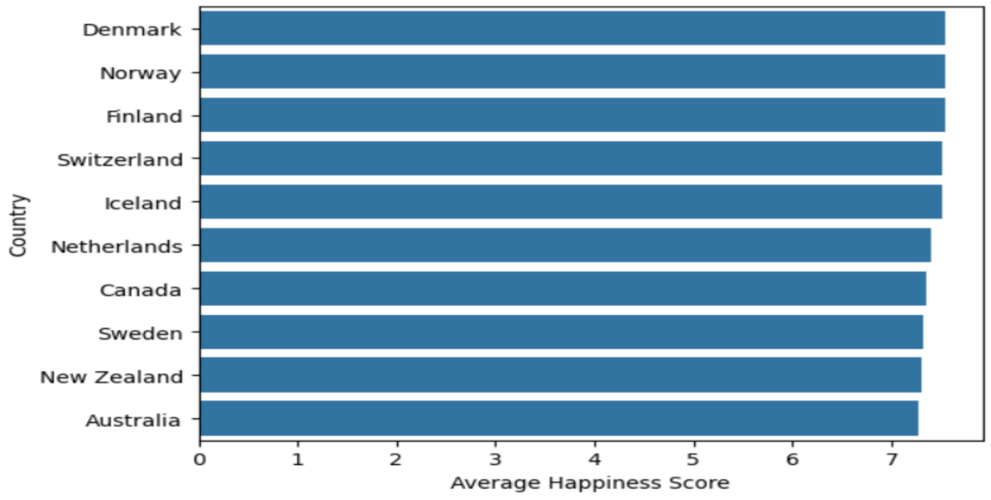
Univariate Graphical Analysis

Q.1 What countries or regions rank the highest in overall happiness and each of the six factors contributing to happiness?

```
]: average_values = World_Happiness_v2.groupby('Country')['Happiness Score'].mean().reset_index()
   average_values.columns = ['Country', 'Average Happiness Score']
   sorted_values = average_values.sort_values(by='Average Happiness Score', ascending=False)
   top10 = sorted_values.head(10)
   Country_counts = average_values['Average Happiness Score'].value_counts()
   plt.figure(figsize=(10, 6))
   plt.barh(top10['Country'], top10['Average Happiness Score'], color='blue')
   plt.title('Top 10 happiest countries')
   plt.xlabel(' Country')
   plt.ylabel('Average Happiness Score')
   plt.show()
```



```
[388]: sns.barplot(y="Country", x="Average Happiness Score", data=top10)

[388]: <Axes: xlabel='Average Happiness Score', ylabel='Country'>
```


Denmark
Norway
Finland
Switzerland
Iceland
Netherlands
Canada
Sweden
New Zealand
Australia
Country
Average Happiness Score