Sardar Patel Institute of Technology, Mumbai
Department of Electronics and Telecommunication Engineering
B.E. Sem-VII (2022-2023)
OEIT6 - Data Analytics

Experiment: Exploratory Data Analysis (EDA)

Name: Pushkar Sutar **Roll No. 2019110060**

Objective : Perform EDA such as number of data samples, number of features, number of classes, number of data samples per class, removing missing values, conversion to numbers, using seaborn library to plot different graphs.

Dataset Description:
World Happiness Report Dataset (2015-19) -
The happiness scores and rankings use data from the Gallup World Poll. The scores are based on answers to the main life evaluation question asked in the poll. The dataset contains 12 attributes.

- Country - name of the country
- Region - region the country belongs to.
- Happiness Rank - rank of the country based on the happiness score.
- Happiness Score - a metric measured by asking sample people with set questions.
- Standard Error - the standard error of happiness score.
- GDP per capita - The extent to which GDP contributes to the calculation of the Happiness Score.
- Family - The extent to which Family contributes to the calculation of the Happiness Score
- Life Expectancy - The extent to which Life expectancy contributed to the calculation of the Happiness Score
- Freedom - The extent to which Freedom contributed to the calculation of the Happiness Score.
- Government Corruption - The extent to which Perception of Corruption contributes to Happiness Score.
- Generosity - The extent to which Generosity contributed to the calculation of the Happiness Score.
- Dystopia Residual - The extent to which Dystopia Residual contributed to the calculation of the Happiness Score.

Code and Output:

To perform EDA Google Colab is used. Entering a command with shift executes the line and prints the result.

Importing necessary modules -
We first mount the drive so that we can fetch the dataset stored on Google drive.

```
from google.colab import drive
drive.mount('/content/drive')
```

We import all the required libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Using the “read_csv” command from Pandas we can import our csv data files. We have imported the happiness index from the year 2015 to 2019 for analysis.

```
df = pd.read_csv('/content/drive/MyDrive/DA/2016.csv')
```

```
df15 = pd.read_csv('/content/drive/MyDrive/DA/2015.csv')
df17 = pd.read_csv('/content/drive/MyDrive/DA/2017.csv')
df18 = pd.read_csv('/content/drive/MyDrive/DA/2018.csv')
df19 = pd.read_csv('/content/drive/MyDrive/DA/2019.csv')
```

EDA

The first thing we will check is the size of our dataset. We can use info() to get the number of entries of each column.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 15 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Country                    157 non-null    object 
 1   Year                       158 non-null    int64  
 2   Rank                       157 non-null    float64
 3   Score                      157 non-null    float64
 4   GDP_Capita                 157 non-null    float64
 5   Family                     157 non-null    float64
 6   Life_Expectancy            157 non-null    float64
 7   Freedom                    157 non-null    float64
 8   Gov_Corruption             157 non-null    float64
 9   Generosity                 157 non-null    float64
 10  Dystopia_Residual          157 non-null    float64
 11  St_Error                   0 non-null      float64
 12  Region                     157 non-null    object 
 13  Lower Confidence Interval  157 non-null    float64
 14  Upper Confidence Interval  157 non-null    float64
dtypes: float64(12), int64(1), object(2)
memory usage: 18.6+ KB
```

We can also get the size using shape() function.

```
df.shape
```

```
(158, 15)
```

To check all the attributes present in the dataset we use columns function.

```
df.columns
```

```
Index(['Country', 'Year', 'Rank', 'Score', 'GDP_Capita', 'Family',
       'Life_Expectancy', 'Freedom', 'Gov_Corruption', 'Generosity',
       'Dystopia_Residual', 'St_Error', 'Region', 'Lower Confidence Interval',
       'Upper Confidence Interval'],
      dtype='object')
```

To see how the data looks like we use head() which returns the first 5 rows by default. We can note here that St_Error is NaN for first 5 rows.

```
df.head()
```

	Country	Year	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Dystopia_Residual	St_Error	Region
0	Denmark	2016	1.0	7.526	1.44178	1.16374	0.79504	0.57941	0.44453	0.36171	2.73939	NaN	Western Europe
1	Switzerland	2016	2.0	7.509	1.52733	1.14524	0.86303	0.58557	0.41203	0.28083	2.69463	NaN	Western Europe
2	Iceland	2016	3.0	7.501	1.42666	1.18326	0.86733	0.56624	0.14975	0.47678	2.83137	NaN	Western Europe
3	Norway	2016	4.0	7.498	1.57744	1.12690	0.79579	0.59609	0.35776	0.37895	2.66465	NaN	Western Europe
4	Finland	2016	5.0	7.413	1.40598	1.13464	0.81091	0.57104	0.41004	0.25492	2.82596	NaN	Western Europe

We access a particular column by using bracket notation and naming the column.

```
df['Country']
```

```
0           Denmark
1       Switzerland
2           Iceland
3            Norway
4           Finland
           ...
153     Afghanistan
154            Togo
155           Syria
156         Burundi
157             NaN
Name: Country, Length: 158, dtype: object
```

For categorical features we get contributions by each value using value_counts(). We note that majority of the countries are from Sub-Saharan Africa and other values can also be noted.

```
df['Region'].value_counts()
```

```
Sub-Saharan Africa                  38
Central and Eastern Europe          29
Latin America and Caribbean         24
Western Europe                      21
Middle East and Northern Africa     19
Southeastern Asia                    9
Southern Asia                        7
Eastern Asia                         6
North America                        2
Australia and New Zealand            2
Name: Region, dtype: int64
```

To get the basic statistical measures of each numerical attribute we use describe(). We can see that the maximum happiness score is 7.5 out of 10 and the St_Error column is empty.

```
df.describe()
```

	Year	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Dystopia_Residual	St_Error	Lower Confidence Interval	Upper Confidence Interval
count	158.0	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	0.0	157.000000	157.000000
mean	2016.0	78.980892	5.382185	0.953880	0.793621	0.557619	0.370994	0.137624	0.242635	2.325807	NaN	5.282395	5.481975
std	0.0	45.466030	1.141674	0.412595	0.266706	0.229349	0.145507	0.111038	0.133756	0.542220	NaN	1.148043	1.136493
min	2016.0	1.000000	2.905000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.817890	NaN	2.732000	3.078000
25%	2016.0	40.000000	4.404000	0.670240	0.641840	0.382910	0.257480	0.061260	0.154570	2.031710	NaN	4.327000	4.465000
50%	2016.0	79.000000	5.314000	1.027800	0.841420	0.596590	0.397470	0.105470	0.222450	2.290740	NaN	5.237000	5.419000
75%	2016.0	118.000000	6.269000	1.279640	1.021520	0.729930	0.484530	0.175540	0.311850	2.664650	NaN	6.154000	6.434000
max	2016.0	157.000000	7.526000	1.824270	1.183260	0.952770	0.608480	0.505210	0.819710	3.837720	NaN	7.460000	7.669000

Likewise, we get more such information about a particular attribute.

Hence, Rwanda has the maximum recorded corruption in the world. While Bosnia and Herzegovina has the least corruption

```
df[df.Gov_Corruption == df.Gov_Corruption.max()]
```

	Country	Year	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Dystopia_Residual	St_Error	Region
151	Rwanda	2016	152.0	3.515	0.32846	0.61586	0.31865	0.5432	0.50521	0.23552	0.96819	NaN	Sub-Saharan Africa

```
df[df.Gov_Corruption == df.Gov_Corruption.min()]
```

	Country	Year	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Dystopia_Residual	St_Error	Region
86	Bosnia and Herzegovina	2016	87.0	5.163	0.93383	0.64367	0.70766	0.09511	0.0	0.29889	2.48406	NaN	Central and Eastern Europe

We also group countries by region and observe the scores.

```
df[df.Region == 'Southern Asia']
```

	Country	Year	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Dystopia_Residual	St_Error	Region
83	Bhutan	2016	84.0	5.196	0.85270	0.90836	0.49759	0.46074	0.16160	0.48546	1.82916	NaN	Southern Asia
91	Pakistan	2016	92.0	5.132	0.68816	0.26135	0.40306	0.14622	0.13880	0.31185	3.18286	NaN	Southern Asia
106	Nepal	2016	107.0	4.793	0.44626	0.69699	0.50073	0.37012	0.07008	0.38160	2.32694	NaN	Southern Asia
109	Bangladesh	2016	110.0	4.643	0.54177	0.24749	0.52989	0.39778	0.12583	0.19132	2.60904	NaN	Southern Asia
116	Sri Lanka	2016	117.0	4.415	0.97318	0.84783	0.62007	0.50817	0.07964	0.46978	0.91681	NaN	Southern Asia
117	India	2016	118.0	4.404	0.74036	0.29247	0.45091	0.40285	0.08722	0.25028	2.18032	NaN	Southern Asia
153	Afghanistan	2016	154.0	3.360	0.38227	0.11037	0.17344	0.16430	0.07112	0.31268	2.14558	NaN	Southern Asia

Data Cleaning -

We check the sum of null and na values per column.

```
df.isnull().sum()
```

```
Country                          1
Year                             0
Rank                             1
Score                            1
GDP_Capita                       1
Family                           1
Life_Expectancy                  1
Freedom                          1
Gov_Corruption                   1
Generosity                       1
Dystopia_Residual                1
St_Error                       158
Region                           1
Lower Confidence Interval        1
Upper Confidence Interval        1
dtype: int64
```

```
df.isna().sum()
```

```
Country                          1
Year                             0
Rank                             1
Score                            1
GDP_Capita                       1
Family                           1
Life_Expectancy                  1
Freedom                          1
Gov_Corruption                   1
Generosity                       1
Dystopia_Residual                1
St_Error                       158
Region                           1
Lower Confidence Interval        1
Upper Confidence Interval        1
dtype: int64
```

We drop the null column St_Error.

```
df.drop('St_Error',inplace = True,axis=1)
```

We check for na rows and drop the irrelevant columns.

```
df[df.isna().any(axis=1)]
```

	Country	Year	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Dystopia_Residual	Region
157	NaN	2016	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
df.dropna(inplace=True)
```

```
df.drop(['Dystopia_Residual','Lower Confidence Interval','Upper Confidence Interval','Year'],axis=1,inplace = True)
```

We perform data normalization using box cox transformation. This ensures that the data better fits the Gaussian distribution and is normal.

```
df['GDP_Capita'] = df['GDP_Capita'].apply(lambda x: boxcox1p(x,1e-4))
df['Family'] = df['Family'].apply(lambda x: boxcox1p(x,1e-4))
df['Life_Expectancy'] = df['Life_Expectancy'].apply(lambda x: boxcox1p(x,1e-4))
df['Freedom'] = df['Freedom'].apply(lambda x: boxcox1p(x,1e-4))
df['Gov_Corruption'] = df['Gov_Corruption'].apply(lambda x: boxcox1p(x,1e-4))
df['Generosity'] = df['Generosity'].apply(lambda x: boxcox1p(x,1e-4))
```

```
num = df.select_dtypes(include='number')
num.drop('Rank',inplace=True,axis=1)
num.head()
```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:4913: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.
  errors=errors,
```

	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity
0	7.526	0.892767	0.771868	0.585044	0.457062	0.367791	0.308746
1	7.509	0.927206	0.763281	0.622224	0.460955	0.345034	0.247511
2	7.501	0.886555	0.780850	0.624529	0.448688	0.139546	0.389872
3	7.498	0.946841	0.754694	0.585462	0.467568	0.305841	0.321328
4	7.413	0.877996	0.758327	0.593847	0.451748	0.343624	0.227074

We further remove outliers which lie 3 standard deviations away from the mean.

```
def remove_outliers(df,columns,n_std):
    for col in columns:
        print('Working on column: {}'.format(col))

        mean = df[col].mean()
        sd = df[col].std()

        df = df[(df[col] <= mean+(n_std*sd))]

    return df
```

```
df2 = remove_outliers(num,num.columns,3)
```

```
Working on column: Score
Working on column: GDP_Capita
Working on column: Family
Working on column: Life_Expectancy
Working on column: Freedom
Working on column: Gov_Corruption
Working on column: Generosity
```

Finally, we check the shape of the data after cleaning.

```
df.shape
```

```
(157, 10)
```

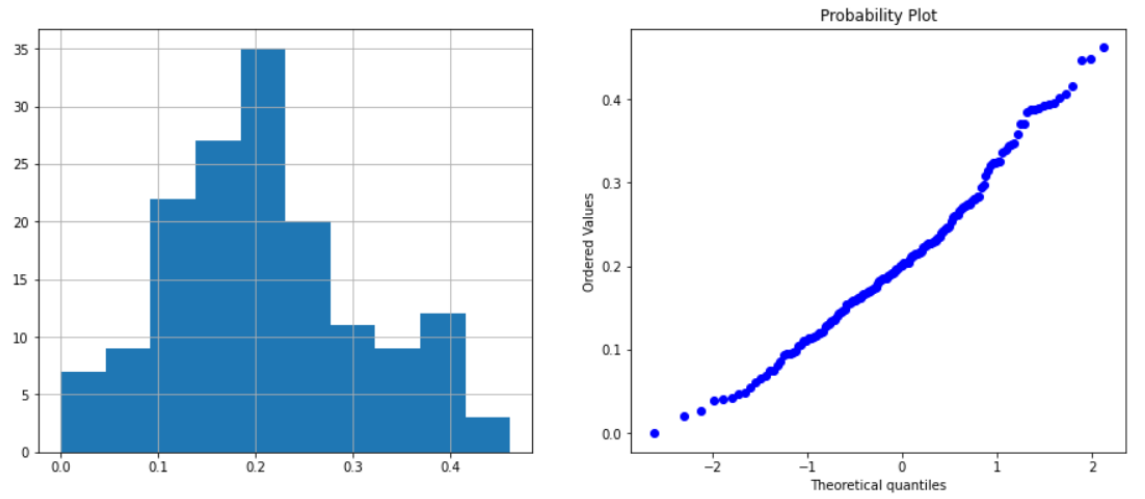
Visualization-
Univariate -

We create a function for QQ and histogram

```
def diagnostic_plots(df, variable):
    # function to plot a histogram and a Q-Q plot
    # side by side, for a certain variable

    plt.figure(figsize=(15,6))
    plt.subplot(1, 2, 1)
    df[variable].hist()

    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)

    plt.show()
```

We can observe that Generosity is fairly normal after cleaning and normalization.

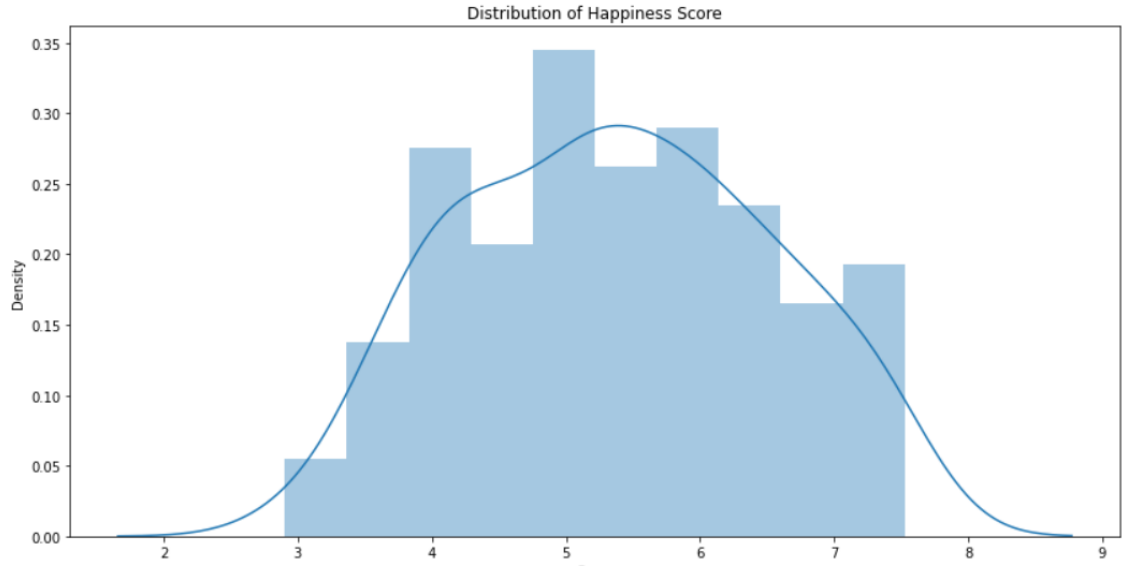
```
diagnostic_plots(df,'Generosity')
```



We can also plot the histogram for happiness score. We observe that very few countries have score greater than 7 and less than 3. Majority of the countries have score around 5.

```
plt.figure(figsize=(14,7))
plt.title("Distribution of Happiness Score")
sns.distplot(a=df['Score'],bins=10);
```

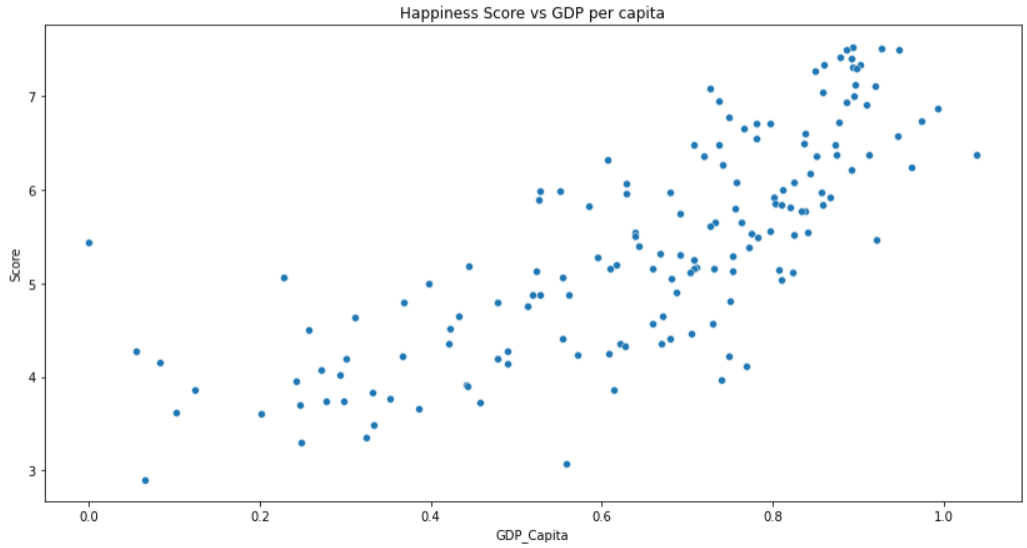
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function
  warnings.warn(msg, FutureWarning)
```



Bivariate Analysis-

We can compare the GDP with the happiness score with the help of a scatter plot. It can easily be observed that the score highly depends on the GDP per capita.

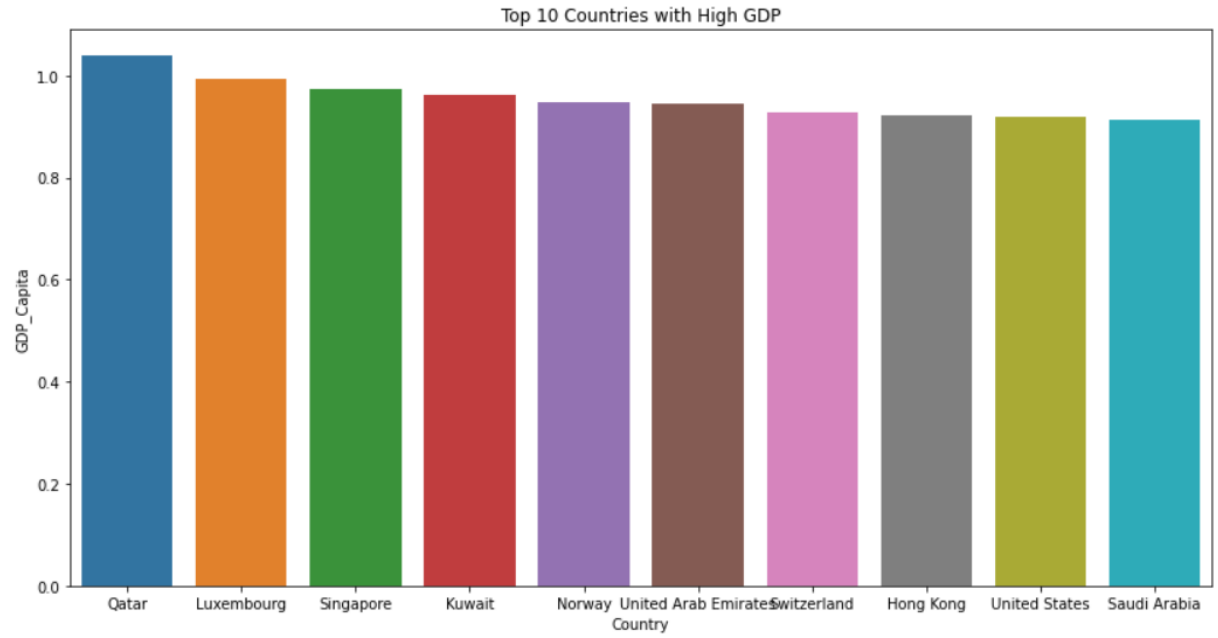
```
plt.figure(figsize=(14,7))
plt.title("Happiness Score vs GDP per capita")
sns.scatterplot(data=df, x='GDP_Capita', y='Score');
```

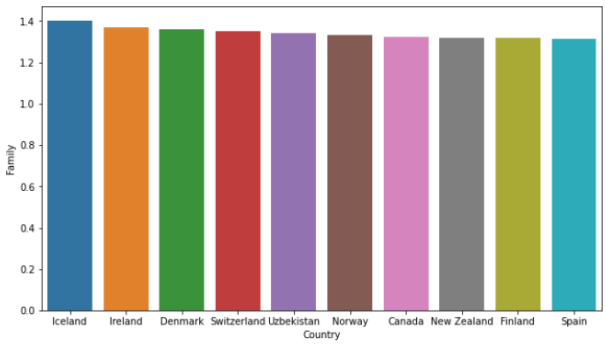


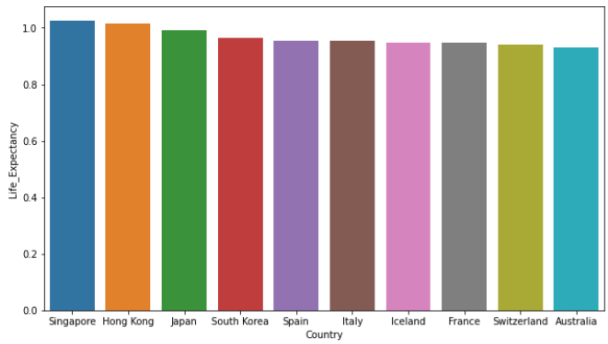
Using bar plots we can visualize the top countries by parameter. Qatar has the highest GDP per capita.
Similarly for other attributes we can note the top countries.

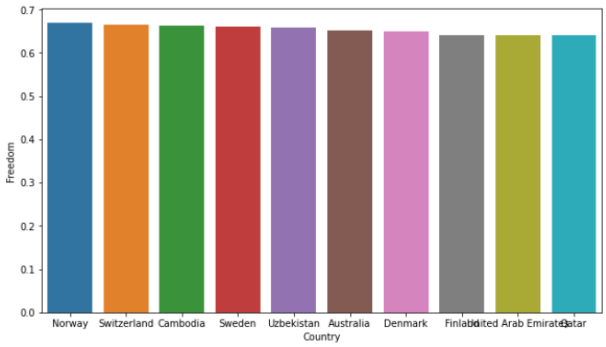
```
plt.figure(figsize=(14,7))
plt.title("Top 10 Countries with High GDP")
sns.barplot(data = df.sort_values('GDP_Capita', ascending= False).head(10), y='GDP_Capita', x='Country')
```

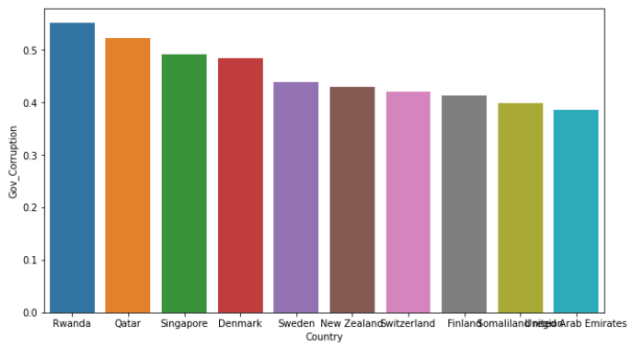
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ce3f1b1d0>
```





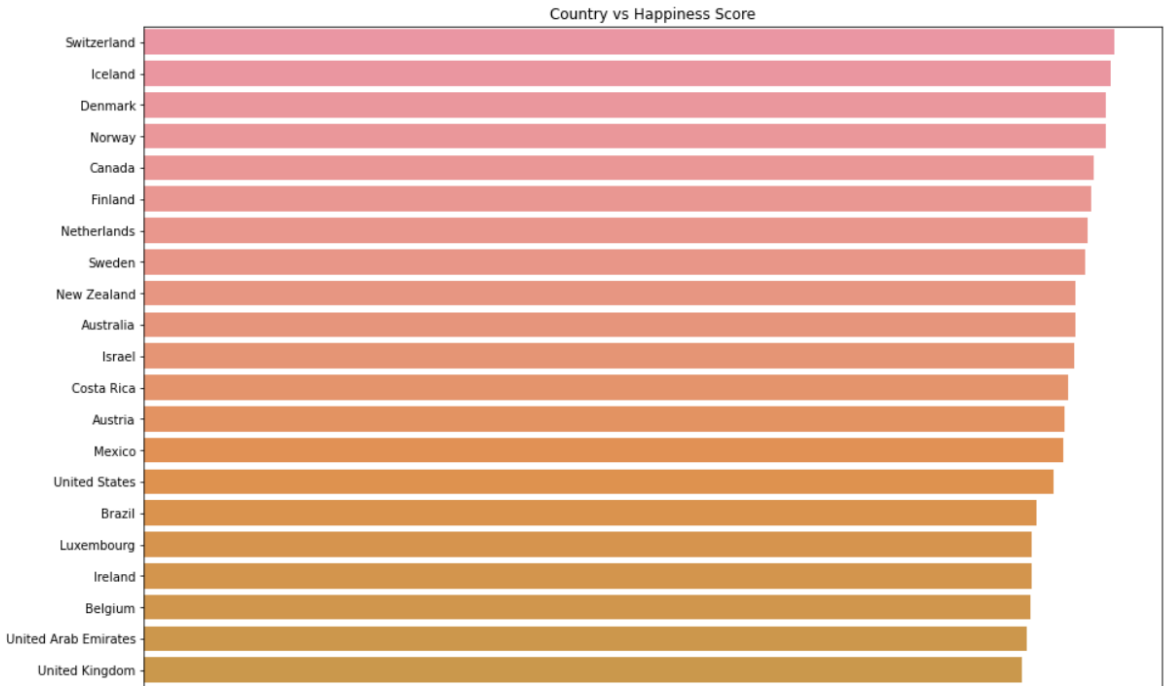






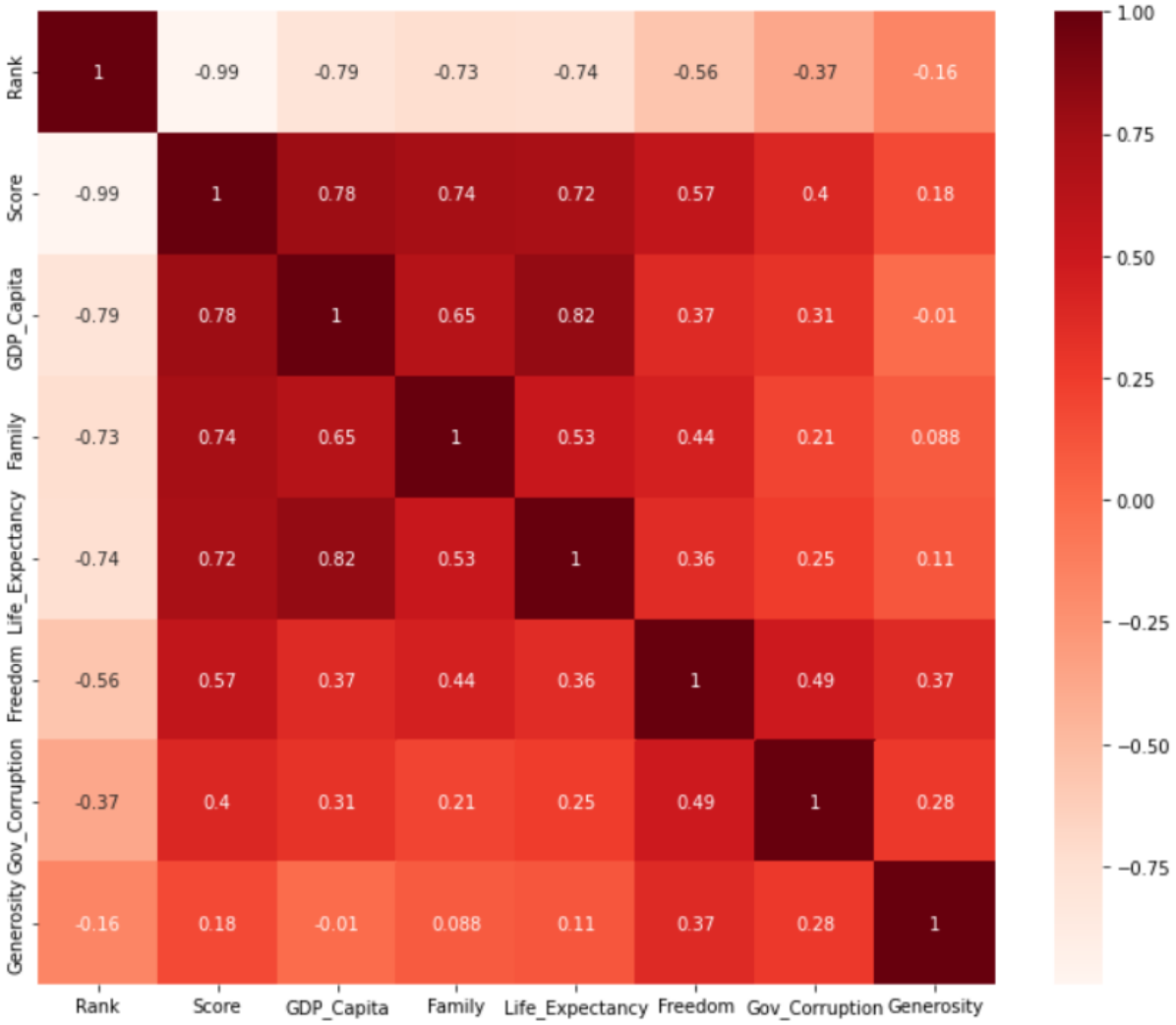
We can also plot the countries by their rank. We note that Switzerland is the happiest country and Burundi has the least happiness score.

```
plt.figure(figsize=(15,75))
plt.title('Country vs Happiness Score')
sns.barplot(data=df.sort_values('Score', ascending=False), x='Score', y='Country');
```



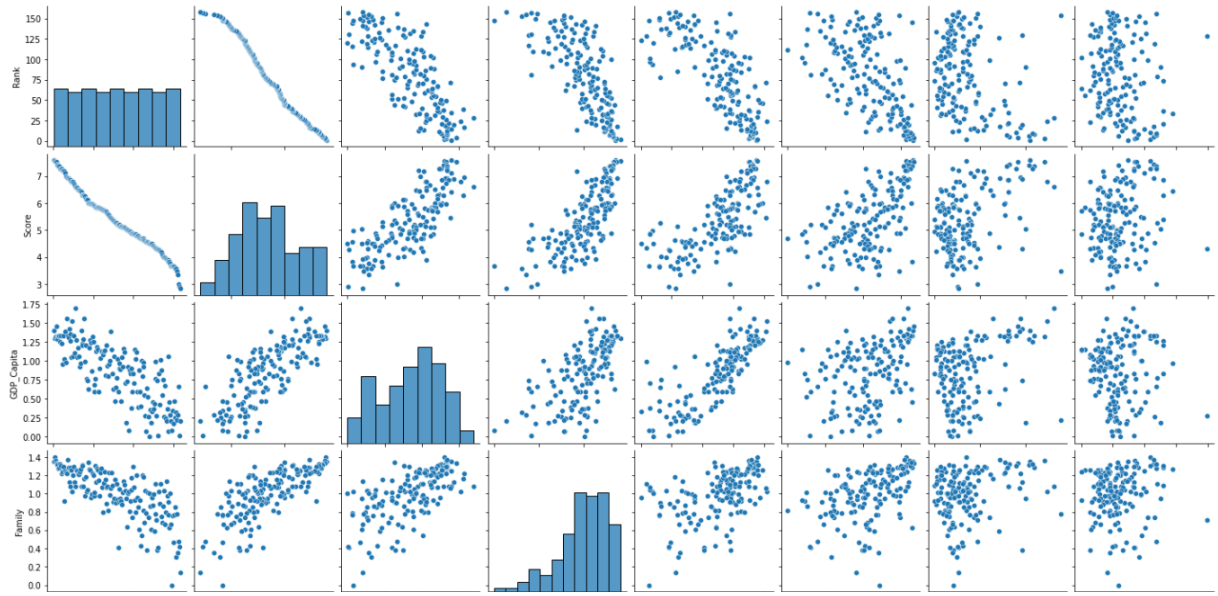
We can see that the score is highly correlated with GDP/capita and Family. Generosity has very little effect on the score. We can also observe that GDP and Life expectency have a high correlation.

```
plt.figure(figsize=(12,10))
corrmat = df.corr()
sns.heatmap(corrmat,annot = True, cmap=plt.cm.Reds )
plt.show()
```



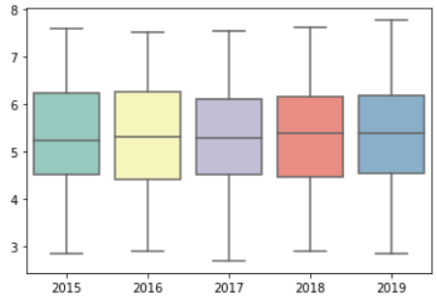
Pairplot also provides information about and relationship between any two attributes.

```
sns.pairplot(df)
```



The box plot shows the change of the happiness score of countries over the years. We can see that there is not a significant difference between their scores by year.

```
scores=pd.DataFrame(data={'2015':df15['Score'],'2016':df['Score'],'2017':df17['Score'],'2018':df18['Score'],'2019':df19['Score']})
sns.boxplot(data=scores,palette='Set3');
```



We can also get geographical visualization of the score.

```
import plotly.graph_objs as go
from plotly.offline import iplot

data = dict(type = 'choropleth',
           locations = df['Country'],
           locationmode = 'country names',
           colorscale='RdYlGn',
           z = df['Score'],
           text = df['Country'],
           colorbar = {'title':'Happiness Score'})

layout = dict(title = 'Geographical Visualization of Happiness Score',
              geo = dict(showframe = True, projection = {'type': 'azimuthal equal area'}))

choromap3 = go.Figure(data = [data], layout=layout)
iplot(choromap3)
```

European countries are generally the happiest countries and Sub-Saharan Africa has most of the countries with scores below the average value.

EDA on India-

India ranks 118 on the list of happiest countries.

```
df.loc[df['Country']=='India']
```

	Country	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Region
117	India	118.0	4.404	0.554107	0.256558	0.372198	0.338512	0.083624	0.22337	Southern Asia

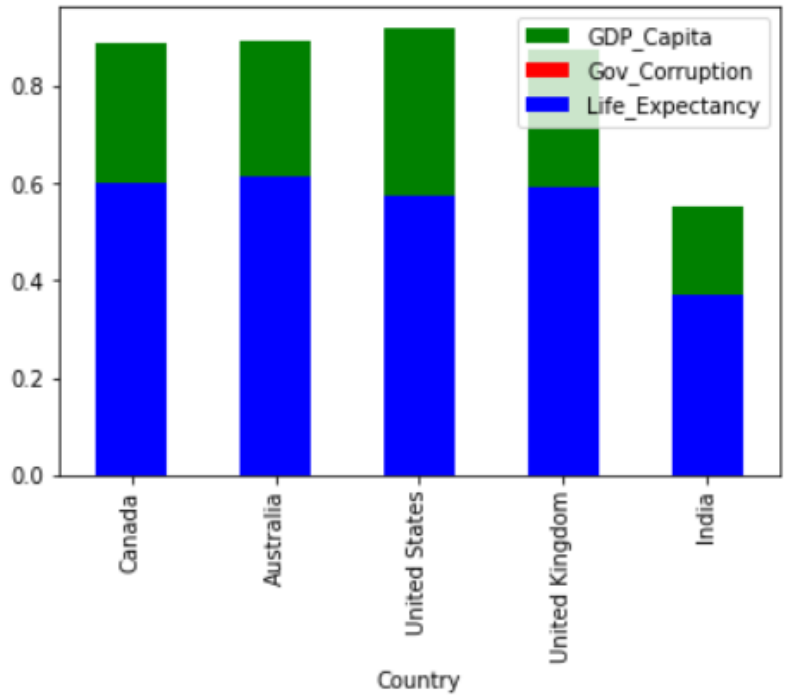
Let us compare India with other major countries like the USA, Canada.

```
d = df[(df['Country'].isin(['India','Canada','United Kingdom', 'United States','Australia']))]
d
```

	Country	Rank	Score	GDP_Capita	Family	Life_Expectancy	Freedom	Gov_Corruption	Generosity	Region
5	Canada	6.0	7.404	0.892099	0.740106	0.603022	0.453440	0.272539	0.370425	North America
8	Australia	9.0	7.313	0.893852	0.744229	0.615853	0.450047	0.280140	0.388035	Australia and New Zealand
12	United States	13.0	7.104	0.919512	0.716802	0.576068	0.393151	0.138614	0.344142	North America
22	United Kingdom	23.0	6.725	0.876686	0.735621	0.593295	0.405713	0.242157	0.406513	Western Europe
117	India	118.0	4.404	0.554107	0.256558	0.372198	0.338512	0.083624	0.223370	Southern Asia

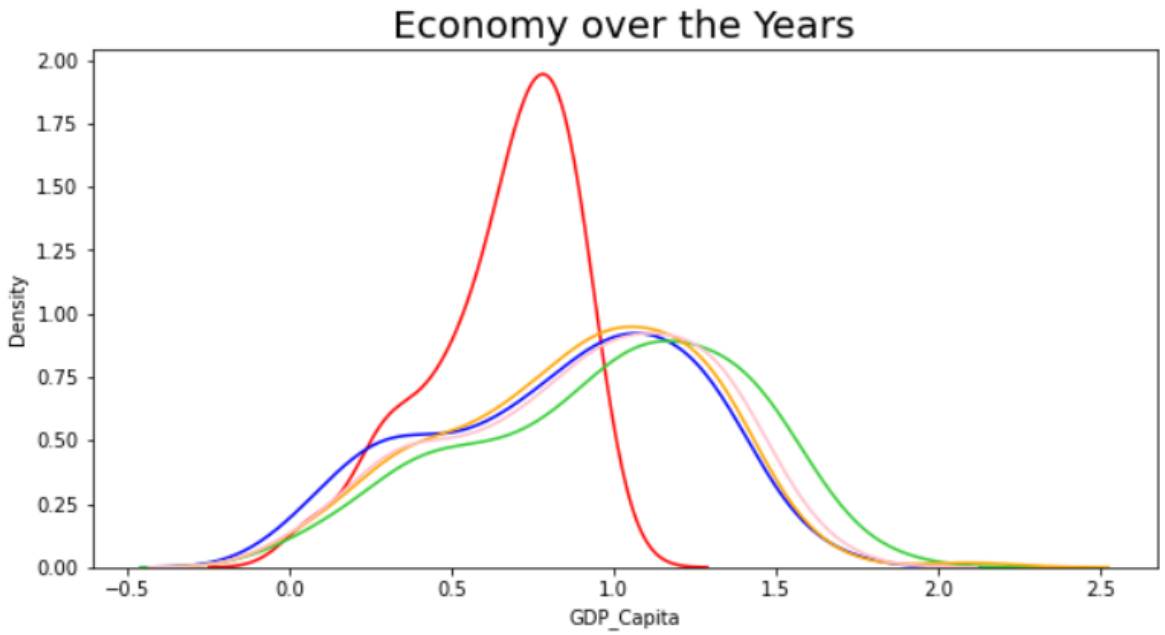
Let us see how these countries differ in attribute scores with India. All these countries have much higher life expectancy, GDP per capita than India. While Government Corruption score is similar.

```
ax = d.plot(y="GDP_Capita", x="Country", kind="bar",color='green')
d.plot(y="Gov_Corruption", x="Country", kind="bar", ax=ax, color="red")
d.plot(y="Life_Expectancy", x="Country", kind="bar", ax=ax, color="blue")

plt.show()
```



We can observe that how GDP of India has kept of declining over the years and so did its happiness score.

```
plt.figure(figsize=(10,5))
sns.kdeplot(df['GDP_Capita'],color='red')
sns.kdeplot(df15['GDP_Capita'],color='blue')
sns.kdeplot(df17['GDP_Capita'],color='limegreen')
sns.kdeplot(df18['GDP_Capita'],color='orange')
sns.kdeplot(df19['GDP_Capita'],color='pink')
plt.title('Economy over the Years',size=20)
plt.show()
```



Conclusions :

- After performing EDA on the World Happiness Report we were able to discover the impact of each different factor in determining “happiness.”
- We had also found that among the different factors, Economic GDP tends to have the greatest happiness with Health following close by.
- The group determined that the “happiest” countries were located in Europe, particularly Scandinavia and Switzerland. Meanwhile the “least happy” countries were located in Africa and the Middle East. This suggests that countries in close proximity or those in the same region often have similar living conditions and are thus affected by factors similarly.
- One bigger concern is how Government Corruption has the lowest scores of all conditions looked at. There is a higher positive correlation between Government Corruption and Freedom which tells that the citizens feel deprived and are unable to make free life choices.
- India’s declining rank can be owed primarily to lower GDP and Health score.