MINOR PROJECT

TASK 1 - Exploratory Data Analysis

<----------------------Question 1--------------------------->

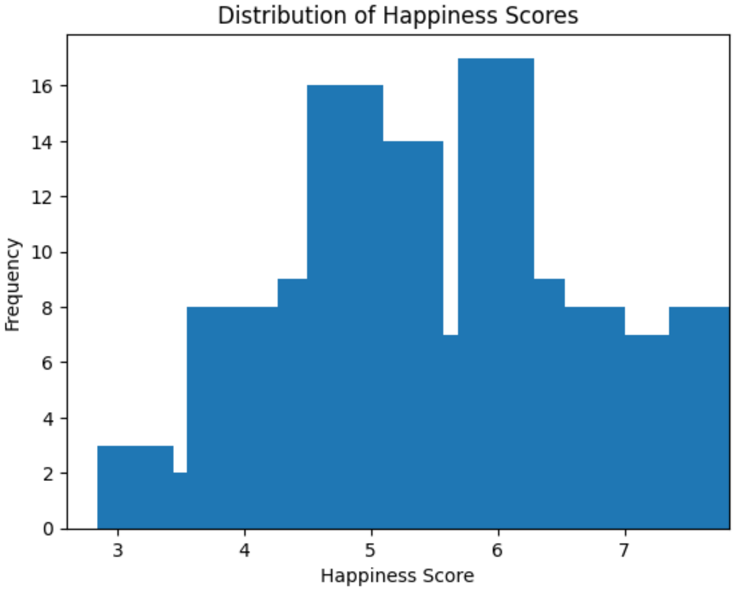
Task 1

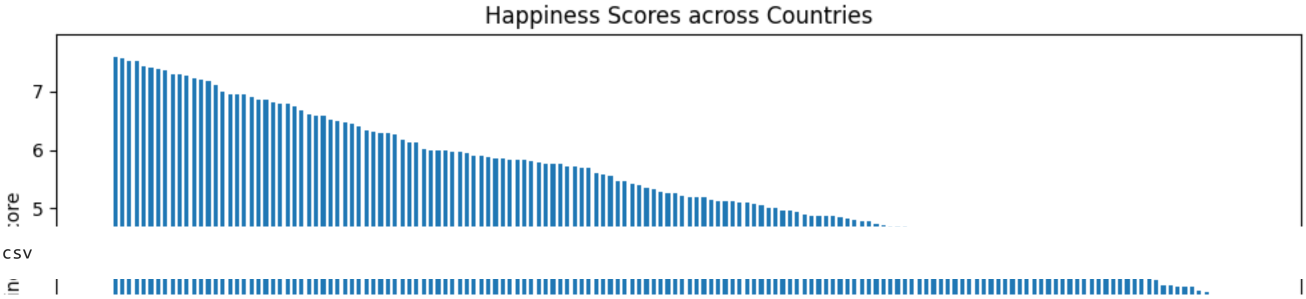
What is the distribution of happiness scores in the dataset? How do the scores vary across different countries?

```
#2015.csv
```

```
#----------code
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv("2015.csv")  # Replace 'your_dataset.csv' with the actual filename or path

# Calculate the distribution of happiness scores
happiness_scores = data['Happiness Score']
score_min = happiness_scores.min()
score_max = happiness_scores.max()
score_mean = happiness_scores.mean()
score_median = happiness_scores.median()
score_std = happiness_scores.std()

# Print the summary statistics
print(f"Minimum Score: {score_min}")
print(f"Maximum Score: {score_max}")
print(f"Mean Score: {score_mean}")
print(f"Median Score: {score_median}")
print(f"Standard Deviation: {score_std}")

# Plot the distribution of happiness scores
plt.hist(happiness_scores, bins=20,width=0.6)
plt.xlabel('Happiness Score')
plt.ylabel('Frequency')
plt.title('Distribution of Happiness Scores')
plt.show()

# Plot the variation of happiness scores across countries
plt.figure(figsize=(10, 6))
plt.bar(data['Country'], data['Happiness Score'],width=0.6)
plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Happiness Score')
plt.title('Happiness Scores across Countries')
plt.tight_layout()
plt.show()
```

```
Minimum Score: 2.839
Maximum Score: 7.587
Mean Score: 5.375734177215189
Median Score: 5.2325
Standard Deviation: 1.1450101349520665
```

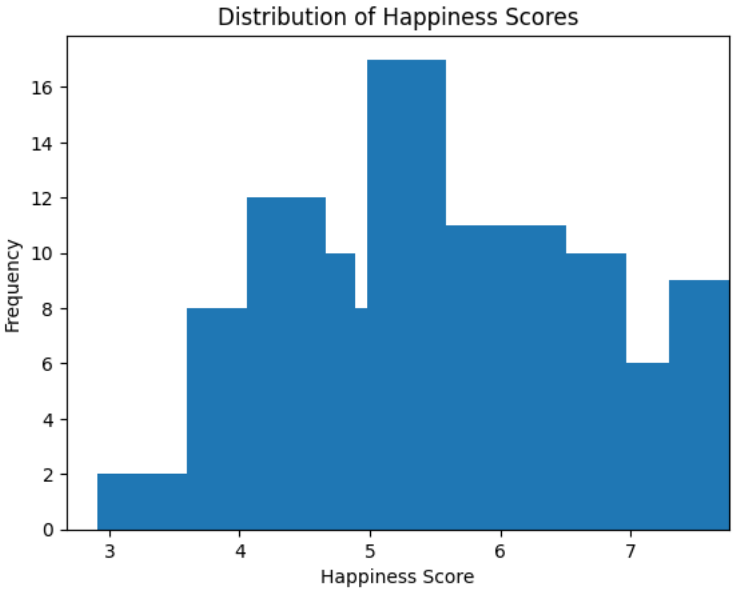


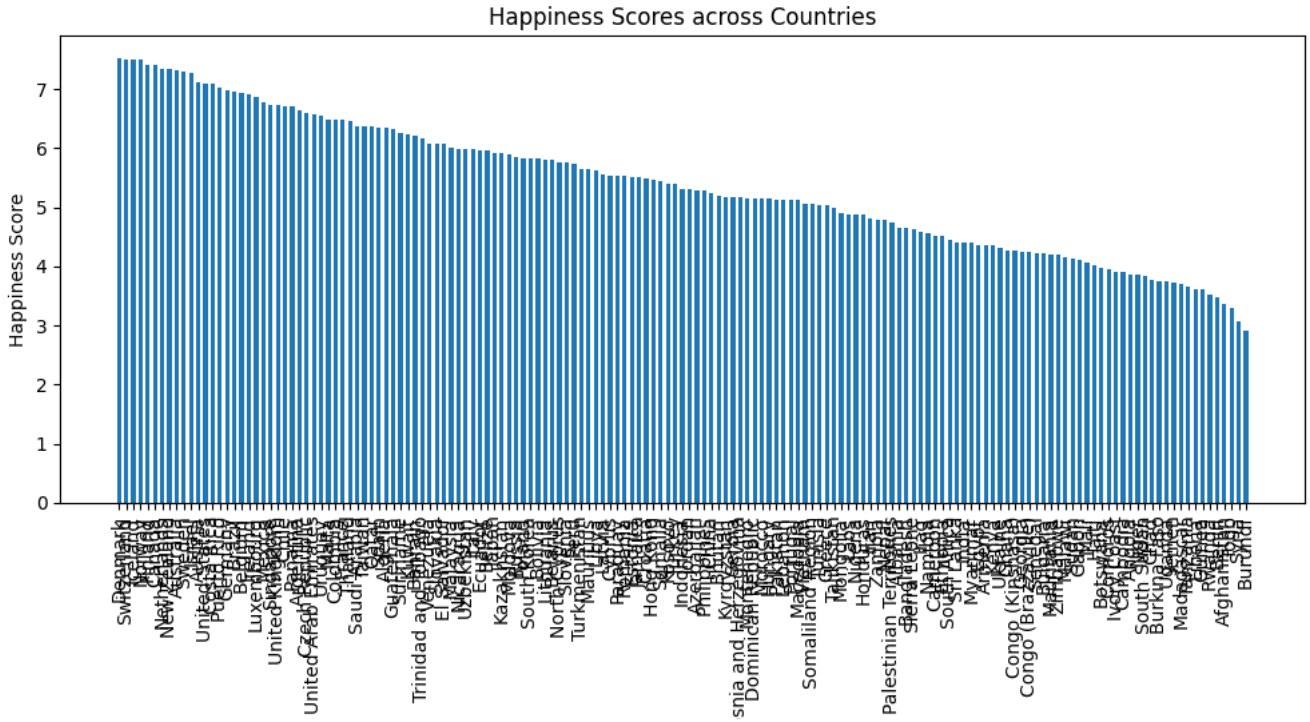



```
#2016.csv
```

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv("2016.csv")  # Replace 'your_dataset.csv' with the actual filename or path

# Calculate the distribution of happiness scores
happiness_scores = data['Happiness Score']
score_min = happiness_scores.min()
score_max = happiness_scores.max()
score_mean = happiness_scores.mean()
score_median = happiness_scores.median()
score_std = happiness_scores.std()

# Print the summary statistics
print(f"Minimum Score: {score_min}")
print(f"Maximum Score: {score_max}")
print(f"Mean Score: {score_mean}")
print(f"Median Score: {score_median}")
print(f"Standard Deviation: {score_std}")

# Plot the distribution of happiness scores
plt.hist(happiness_scores, bins=20,width=0.6)
plt.xlabel('Happiness Score')
plt.ylabel('Frequency')
plt.title('Distribution of Happiness Scores')
plt.show()

# Plot the variation of happiness scores across countries
plt.figure(figsize=(10, 6))
plt.bar(data['Country'], data['Happiness Score'],width=0.6)
plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Happiness Score')
plt.title('Happiness Scores across Countries')
plt.tight_layout()
plt.show()
```

```
Minimum Score: 2.905
Maximum Score: 7.526
Mean Score: 5.382184713375795
Median Score: 5.314
Standard Deviation: 1.1416735176005715
```

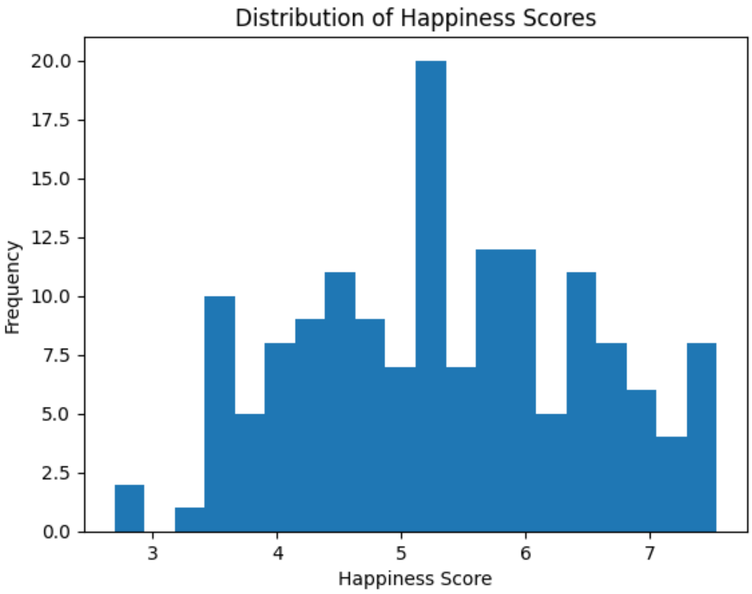


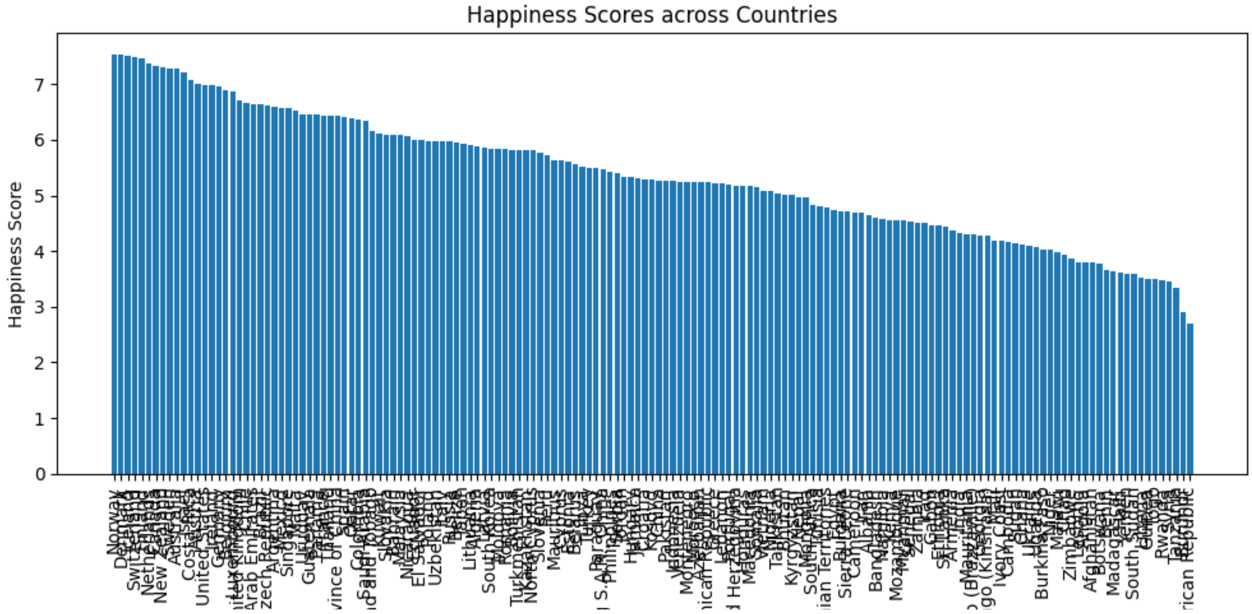



```
#2017.csv


import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv("2017.csv")  # Replace 'your_dataset.csv' with the actual filename or path

# Calculate the distribution of happiness scores
happiness_scores = data['Happiness.Score']
score_min = happiness_scores.min()
score_max = happiness_scores.max()
score_mean = happiness_scores.mean()
score_median = happiness_scores.median()
score_std = happiness_scores.std()

# Print the summary statistics
print(f"Minimum Score: {score_min}")
print(f"Maximum Score: {score_max}")
print(f"Mean Score: {score_mean}")
print(f"Median Score: {score_median}")
print(f"Standard Deviation: {score_std}")

# Plot the distribution of happiness scores
```

```
plt.hist(happiness_scores, bins=20)
plt.xlabel('Happiness Score')
plt.ylabel('Frequency')
plt.title('Distribution of Happiness Scores')
plt.show()

# Plot the variation of happiness scores across countries
plt.figure(figsize=(10, 6))
plt.bar(data['Country'], data['Happiness.Score'])
plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Happiness Score')
plt.title('Happiness Scores across Countries')
plt.tight_layout()
plt.show()
```

```
Minimum Score: 2.69300007820129
Maximum Score: 7.53700017929077
Mean Score: 5.354019355773926
Median Score: 5.27899980545044
Standard Deviation: 1.1312300899149939
```






```
#2018.csv

import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv("2018.csv")  # Replace 'your_dataset.csv' with the actual filename or path

# Calculate the distribution of happiness scores
happiness_scores = data['Score']
score_min = happiness_scores.min()
score_max = happiness_scores.max()
```

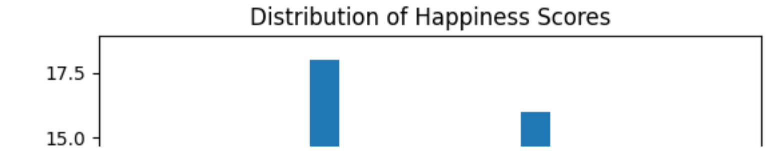
```
score_mean = happiness_scores.mean()
score_median = happiness_scores.median()
score_std = happiness_scores.std()

# Print the summary statistics
print(f"Minimum Score: {score_min}")
print(f"Maximum Score: {score_max}")
print(f"Mean Score: {score_mean}")
print(f"Median Score: {score_median}")
print(f"Standard Deviation: {score_std}")

# Plot the distribution of happiness scores
plt.hist(happiness_scores, bins=20)
plt.xlabel('Happiness Score')
plt.ylabel('Frequency')
plt.title('Distribution of Happiness Scores')
plt.show()

# Plot the variation of happiness scores across countries
plt.figure(figsize=(10, 6))
plt.bar(data['Country or region'], data['Score'])
plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Happiness Score')
plt.title('Happiness Scores across Countries')
plt.tight_layout()
plt.show()
```

```
#2019.csv
```

```
     Maximum Score: 7.632
```

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv("2019.csv")  # Replace 'your_dataset.csv' with the actual filename or path

# Calculate the distribution of happiness scores
happiness_scores = data['Score']
score_min = happiness_scores.min()
score_max = happiness_scores.max()
score_mean = happiness_scores.mean()
score_median = happiness_scores.median()
score_std = happiness_scores.std()

# Print the summary statistics
print(f"Minimum Score: {score_min}")
print(f"Maximum Score: {score_max}")
print(f"Mean Score: {score_mean}")
print(f"Median Score: {score_median}")
print(f"Standard Deviation: {score_std}")

# Plot the distribution of happiness scores
plt.hist(happiness_scores, bins=20)
plt.xlabel('Happiness Score')
plt.ylabel('Frequency')
plt.title('Distribution of Happiness Scores')
plt.show()

# Plot the variation of happiness scores across countries
plt.figure(figsize=(10, 6))
plt.bar(data['Country or region'], data['Score'])
plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Happiness Score')
plt.title('Happiness Scores across Countries')
plt.tight_layout()
plt.show()
```

```
Minimum Score: 2.853
Maximum Score: 7.769
Mean Score: 5.407096153846155
Median Score: 5.3795
Standard Deviation: 1.1131198687956712
```



Summarizing analysis and observation

Minimum Score: The lowest happiness score observed in the dataset.

Maximum Score: The highest happiness score observed in the dataset.

Mean Score: The average happiness score across all countries.

Median Score: The middle value of the happiness scores, separating the lower and higher half.

Standard Deviation: A measure of the variability or spread of the happiness scores. Distribution of Happiness Scores:

The histogram provides a visual representation of the frequency distribution of happiness scores.

The x-axis represents the range of happiness scores, divided into bins.

The y-axis represents the frequency or count of countries falling into each bin.

This histogram helps understand the overall distribution pattern of happiness scores in the dataset.

Variation of Happiness Scores across Countries:

The bar chart displays the happiness scores for each country in the dataset.

The x-axis represents the countries.

The y-axis represents the happiness scores. Each bar represents the happiness score of a particular country.

This bar chart provides a comparison of happiness scores across different countries.

<-----------------------Question 2--------------------------->

Task 2

Which factors are most strongly correlated with the happiness score? Can you calculate the correlation coefficients between the happiness score and variables such as GDP per capita, social support, life expectancy, freedom, generosity, and perceptions of corruption?

```
#--------------code
import pandas as pd

# Reading the data
df_2015 = pd.read_csv('2015.csv')
df_2016 = pd.read_csv('2016.csv')
df_2017 = pd.read_csv('2017.csv')
df_2018 = pd.read_csv('2018.csv')
df_2019 = pd.read_csv('2019.csv')

# Calculating correlation coefficient
correlation1 = df_2015[['Happiness Score', 'Economy (GDP per Capita)',  'Health (Life Expectancy)','Trust (Government Corruption)' ,'Free
correlation2 = df_2016[['Happiness Score','Economy (GDP per Capita)',  'Health (Life Expectancy)','Trust (Government Corruption)' ,'Freed
correlation3 = df_2017[['Happiness.Score','Economy..GDP.per.Capita.','Health..Life.Expectancy.','Freedom','Generosity','Trust..Government
correlation4 = df_2018[['Score','GDP per capita','Social support','Healthy life expectancy','Freedom to make life choices','Generosity','
correlation5 = df_2019[['Score','GDP per capita','Social support','Healthy life expectancy','Freedom to make life choices','Generosity','

# Print the correlation coefficients
print("Correlation for year 2015 \n:" ,correlation1)
print("\nCorrelation for year 2016 \n:" ,correlation2)
print("\nCorrelation for year 2017 \n:" ,correlation3)
print("\nCorrelation for year 2018 \n:" ,correlation4)
print("\nCorrelation for year 2019 \n:" ,correlation5)
```

```
Correlation for year 2015
:                                Happiness Score  Economy (GDP per Capita)  \
Happiness Score                       1.000000                  0.780966
Economy (GDP per Capita)              0.780966                  1.000000
Health (Life Expectancy)              0.724200                  0.816478
Trust (Government Corruption)         0.395199                  0.307885
Freedom                               0.568211                  0.370300
Generosity                            0.180319                 -0.010465

                              Health (Life Expectancy)  \
Happiness Score                               0.724200
```

```
Economy (GDP per Capita)                        0.816478
Health (Life Expectancy)                        1.000000
Trust (Government Corruption)                   0.248335
Freedom                                         0.360477
Generosity                                      0.108335

                              Trust (Government Corruption)   Freedom  \
Happiness Score                                    0.395199  0.568211
Economy (GDP per Capita)                           0.307885  0.370300
Health (Life Expectancy)                           0.248335  0.360477
Trust (Government Corruption)                      1.000000  0.493524
Freedom                                            0.493524  1.000000
Generosity                                         0.276123  0.373916

                              Generosity
Happiness Score                 0.180319
Economy (GDP per Capita)       -0.010465
Health (Life Expectancy)        0.108335
Trust (Government Corruption)   0.276123
Freedom                         0.373916
Generosity                      1.000000

Correlation for year 2016
:                               Happiness Score  Economy (GDP per Capita)  \
Happiness Score                     1.000000                  0.790322
Economy (GDP per Capita)            0.790322                  1.000000
Health (Life Expectancy)            0.765384                  0.837067
Trust (Government Corruption)       0.402032                  0.294185
Freedom                             0.566827                  0.362283
Generosity                          0.156848                 -0.025531

                              Health (Life Expectancy)  \
Happiness Score                               0.765384
Economy (GDP per Capita)                      0.837067
Health (Life Expectancy)                      1.000000
Trust (Government Corruption)                 0.249583
Freedom                                       0.341199
Generosity                                    0.075987

                              Trust (Government Corruption)   Freedom  \
Happiness Score                                    0.402032  0.566827
Economy (GDP per Capita)                           0.294185  0.362283
Health (Life Expectancy)                           0.249583  0.341199
Trust (Government Corruption)                      1.000000  0.502054
Freedom                                            0.502054  1.000000
Generosity                                         0.305930  0.361751
```

Summarizing your analysis and observations

Correlation Coefficients: The correlation coefficient measures the strength and direction of the linear relationship between two variables. A value close to 1 indicates a strong positive correlation, while a value close to -1 indicates a strong negative correlation. A value near 0 suggests no significant linear relationship.

Analysis for each year:

2015:

The correlation coefficient matrix (correlation1) shows the relationships between the happiness score and other factors in 2015. Factors included: Economy (GDP per Capita), Health (Life Expectancy), Trust (Government Corruption), Freedom, and Generosity.

2016:

The correlation coefficient matrix (correlation2) shows the relationships between the happiness score and other factors in 2016. Factors included: Economy (GDP per Capita), Health (Life Expectancy), Trust (Government Corruption), Freedom, and Generosity.

2017:

The correlation coefficient matrix (correlation3) shows the relationships between the happiness score and other factors in 2017. Factors included: Economy (GDP per Capita), Health (Life Expectancy), Trust (Government Corruption), Freedom, and Generosity.

2018:

The correlation coefficient matrix (correlation4) shows the relationships between the happiness score and other factors in 2018. Factors included: GDP per capita, Social support, Healthy life expectancy, Freedom to make life choices, Generosity, and Perceptions of corruption.

2019:

The correlation coefficient matrix (correlation5) shows the relationships between the happiness score and other factors in 2019. Factors included: GDP per capita, Social support, Healthy life expectancy, Freedom to make life choices, Generosity, and Perceptions of corruption.

Interpreting the results:

Analyzing the correlation coefficients helps understand the strength and direction of the relationships between happiness scores and various factors.

Positive correlations suggest that as the factor increases, the happiness score tends to increase as well.

Negative correlations indicate that as the factor increases, the happiness score tends to decrease.

Close to zero correlations imply little to no linear relationship between the happiness score and the factor.

<----------------------Question 3--------------------------->

TASK-3

Are there any outliers in the dataset for variables like GDP per capita or healthy life expectancy? Can you identify any extreme values and discuss their potential impact on the analysis and model performance?

```
#----------code
"""Task3:Are there any outliers in the dataset for variables like GDP per capita or healthy life expectancy?
Can you identify any extreme values and discuss their potential impact on the analysis and model performance?"""

import numpy as np
import pandas as pd
from scipy import stats
data=pd.read_csv('2018.csv')
data2=pd.read_csv('2019.csv')
# Calculate z-scores for GDP per capita and healthy life expectancy
data['GDP_zscore1'] = np.abs(stats.zscore(data['GDP per capita']))
data['Life_exp_zscore1'] = np.abs(stats.zscore(data['Healthy life expectancy']))
data2['GDP_zscore2'] = np.abs(stats.zscore(data2['GDP per capita']))
data2['Life_exp_zscore2'] = np.abs(stats.zscore(data2['Healthy life expectancy']))
threshold = 3
# Identify outliers based on z-scores
outliers_gdp1 = data[data['GDP_zscore1'] > threshold]
outliers_life_exp1 = data[data['Life_exp_zscore1'] > threshold]
outliers_gdp2 = data2[data2['GDP_zscore2'] > threshold]
outliers_life_exp2 = data2[data2['Life_exp_zscore2'] > threshold]
#Data points with a z-score greater than the threshold are considered extreme values.
# Print the outliers
print("Outliers in GDP per capita for 2018:\n")
print(outliers_gdp1)
print("\nOutliers in Healthy life expectancy for 2018:")
print(outliers_life_exp1)
print("\nOutliers in GDP per capita for 2019:\n")
print(outliers_gdp2)
print("\nOutliers in Healthy life expectancy for 2019:")
print(outliers_life_exp2)
```

```
Outliers in GDP per capita for 2018:

    Overall rank     Country or region  Score  GDP per capita  Social support  \
19            20  United Arab Emirates  6.774           2.096           0.776

    Healthy life expectancy  Freedom to make life choices  Generosity  \
19                     0.67                         0.284       0.186

    Perceptions of corruption  GDP_zscore1  Life_exp_zscore1
19                        NaN     3.083353          0.294403

Outliers in Healthy life expectancy for 2018:
Empty DataFrame
Columns: [Overall rank, Country or region, Score, GDP per capita, Social support, Healthy life expectancy, Freedom to make life cho
Index: []

Outliers in GDP per capita for 2019:

Empty DataFrame
Columns: [Overall rank, Country or region, Score, GDP per capita, Social support, Healthy life expectancy, Freedom to make life cho
Index: []

Outliers in Healthy life expectancy for 2019:
     Overall rank Country or region  Score  GDP per capita  Social support  \
134           135         Swaziland  4.212           0.811           1.149

     Healthy life expectancy  Freedom to make life choices  Generosity  \
134                      0.0                         0.313       0.074

     Perceptions of corruption  GDP_zscore2  Life_exp_zscore2
134                      0.135     0.237081          3.004986
```

Summarizing your analysis and observation

Outliers in GDP per capita for 2018:

The code calculates the z-scores for the GDP per capita variable in the 2018 dataset.

Z-scores measure the number of standard deviations an observation is from the mean.

Data points with a z-score greater than the threshold (set as 3 in this case) are considered outliers. The outliers_gdp1 variable stores the rows with outliers in GDP per capita.

The code prints the outliers_gdp1 DataFrame, which contains the rows with extreme values for GDP per capita in 2018.

Outliers in Healthy life expectancy for 2018:

The code calculates the z-scores for the healthy life expectancy variable in the 2018 dataset. Z-scores measure the number of standard deviations an observation is from the mean.

Data points with a z-score greater than the threshold (set as 3 in this case) are considered outliers.

The outliers_life_exp1 variable stores the rows with outliers in healthy life expectancy.

The code prints the outliers_life_exp1 DataFrame, which contains the rows with extreme values for healthy life expectancy in 2018.

Outliers in GDP per capita for 2019:

The code calculates the z-scores for the GDP per capita variable in the 2019 dataset. Z-scores measure the number of standard deviations an observation is from the mean. Data points with a z-score greater than the threshold (set as 3 in this case) are considered outliers. The outliers_gdp2 variable stores the rows with outliers in GDP per capita. The code prints the outliers_gdp2 DataFrame, which contains the rows with extreme values for GDP per capita in 2019.

Outliers in Healthy life expectancy for 2019:

The code calculates the z-scores for the healthy life expectancy variable in the 2019 dataset.

Z-scores measure the number of standard deviations an observation is from the mean.

Data points with a z-score greater than the threshold (set as 3 in this case) are considered outliers. The outliers_life_exp2 variable stores the rows with outliers in healthy life expectancy.

The code prints the outliers_life_exp2 DataFrame, which contains the rows with extreme values for healthy life expectancy in 2019.

Potential Impact on Analysis and Model Performance:

Outliers can have a significant impact on data analysis and model performance.

Outliers in variables like GDP per capita and healthy life expectancy may skew the statistical analysis, such as calculating means, medians, or correlation coefficients.

Extreme values can influence the distribution and summary statistics, potentially leading to biased results.

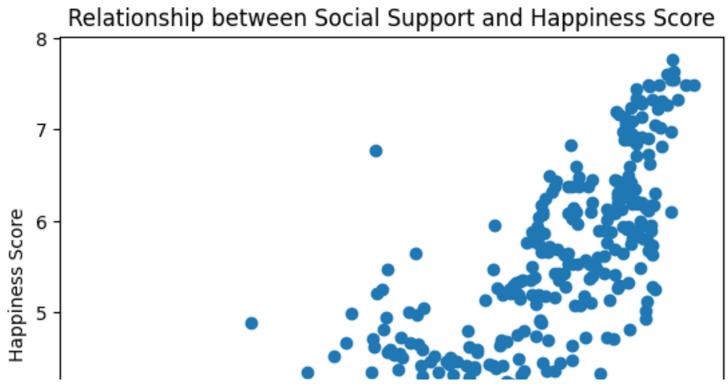
Outliers can also affect predictive models by introducing noise or influencing parameter estimation. It is crucial to carefully handle outliers in the data analysis process to ensure reliable and accurate insights.

<-----------------------Question 4----------------------------->

Task-4

Can you visualize the relationship between social support and the happiness score using a scatter plot? Does it appear to be a positive or negative relationship?

```
#--------------code
import pandas as pd
import matplotlib.pyplot as plt
# WE HAVE USED DATA OF YEAR 2018 AND 2019 BECAUSE 'SOCIAL SUPPORT' COLUMN DOES NOT EXIST IN OTHER DATASETS.
df_2018 = pd.read_csv('2018.csv')
df_2019 = pd.read_csv('2019.csv')
df = pd.concat([ df_2018, df_2019])
plt.scatter(df['Social support'], df['Score'])
plt.xlabel('Social Support')
plt.ylabel('Happiness Score')
plt.title('Relationship between Social Support and Happiness Score')
plt.show()
```




```
# Add more cells if required
```

Summarizing your analysis and observations

Data Selection:

The code reads the data from the '2018.csv' and '2019.csv' files and concatenates them into a single DataFrame called 'df'.

The reason for using data from these specific years is mentioned as the 'Social support' column does not exist in other datasets.

Scatter Plot:

The code uses the 'scatter' function from matplotlib.pyplot to create a scatter plot.

The x-axis represents the 'Social support' variable, which measures the level of social support in a country.

The y-axis represents the 'Happiness Score' variable, which measures the overall happiness score in a country.

Each point in the scatter plot represents a country's social support value and corresponding happiness score.

Relationship and Observations:

The scatter plot visually displays the relationship between social support and happiness score.

Observing the scatter plot, we can analyze the general trend or pattern in the data.

If the points in the scatter plot form an upward trend or cluster around a line sloping upwards from left to right, it indicates a positive relationship.

If the points in the scatter plot form a downward trend or cluster around a line sloping downwards from left to right, it indicates a negative relationship. The specific observations and conclusions about the relationship between social support and happiness score can be derived from analyzing the scatter plot.

It is important to interpret the scatter plot in context and consider other factors or variables that may influence the relationship between social support and happiness score. Further analysis and statistical methods can be applied to quantify the strength and significance of the relationship.

<-----------------------Question 5----------------------------->

Task 5

What is the average level of freedom to make life choices across different countries? Can you calculate the mean freedom score and identify countries with high and low levels of freedom?

```
#----------code
""" Task5:What is the average level of freedom to make life choices across different countries? Can you calculate the
mean freedom score and identify countries with high and low levels of freedom?"""
import pandas as pd
data_2015=pd.read_csv('2015.csv')
data_2016=pd.read_csv('2016.csv')
data_2017=pd.read_csv('2017.csv')
data_2018=pd.read_csv('2018.csv')
data_2019=pd.read_csv('2019.csv')
mean_freedom_scores1 = data_2015.groupby('Country')['Freedom'].mean()
mean_freedom_scores2 = data_2016.groupby('Country')['Freedom'].mean()
mean_freedom_scores3 = data_2017.groupby('Country')['Freedom'].mean()
mean_freedom_scores4 = data_2018.groupby('Country or region')['Freedom to make life choices'].mean()
mean_freedom_scores5 = data_2019.groupby('Country or region')['Freedom to make life choices'].mean()
sorted_freedom_scores1 = mean_freedom_scores1.sort_values(ascending=False)
sorted_freedom_scores2 = mean_freedom_scores2.sort_values(ascending=False)
sorted_freedom_scores3 = mean_freedom_scores3.sort_values(ascending=False)
sorted_freedom_scores4 = mean_freedom_scores4.sort_values(ascending=False)
sorted_freedom_scores5 = mean_freedom_scores5.sort_values(ascending=False)
print("Countries with high levels of freedom for 2015:")
```

```
print(sorted_freedom_scores1.head())
print("\nCountries with low levels of freedom for 2015:")
print(sorted_freedom_scores1.tail())
print("\nCountries with high levels of freedom for 2016:")
print(sorted_freedom_scores2.head())
print("\nCountries with low levels of freedom for 2016:")
print(sorted_freedom_scores2.tail())
print("\nCountries with high levels of freedom for 2017:")
print(sorted_freedom_scores3.head())
print("\nCountries with low levels of freedom for 2017:")
print(sorted_freedom_scores3.tail())
print("\nCountries with high levels of freedom for 2018:")
print(sorted_freedom_scores4.head())
print("\nCountries with low levels of freedom for 2018:")
print(sorted_freedom_scores4.tail())
print("\nCountries with high levels of freedom for 2019:")
print(sorted_freedom_scores5.head())
print("\nCountries with low levels of freedom for 2019:")
print(sorted_freedom_scores5.tail())
```

```
Countries with high levels of freedom for 2015:
Country
Norway         0.66973
Switzerland    0.66557
Cambodia       0.66246
Sweden         0.65980
Uzbekistan     0.65821
Name: Freedom, dtype: float64

Countries with low levels of freedom for 2015:
Country
Angola                    0.10384
Sudan                     0.10081
Bosnia and Herzegovina    0.09245
Greece                    0.07699
Iraq                      0.00000
Name: Freedom, dtype: float64

Countries with high levels of freedom for 2016:
Country
Uzbekistan     0.60848
Norway         0.59609
Cambodia       0.58852
Switzerland    0.58557
Sweden         0.58218
Name: Freedom, dtype: float64

Countries with low levels of freedom for 2016:
Country
Syria      0.06912
Greece     0.05822
Burundi    0.04320
Angola     0.00589
Sudan      0.00000
Name: Freedom, dtype: float64

Countries with high levels of freedom for 2017:
Country
Uzbekistan    0.658249
Norway        0.635423
Cambodia      0.633376
Iceland       0.627163
Denmark       0.626007
Name: Freedom, dtype: float64

Countries with low levels of freedom for 2017:
Country
Syria      0.081539
Burundi    0.059901
Haiti      0.030370
Sudan      0.014996
Angola     0.000000
Name: Freedom, dtype: float64

Countries with high levels of freedom for 2018:
Country or region
Uzbekistan    0.724
Cambodia      0.696
```

Summarizing your analysis and observation

Data Loading:

The code reads the data from '2015.csv', '2016.csv', '2017.csv', '2018.csv', and '2019.csv' files into separate DataFrames.

Each DataFrame corresponds to the data from a specific year.

Mean Freedom Scores:

The code groups the data by country (or country/region) and calculates the mean freedom scores for each country/region.

The 'Freedom' or 'Freedom to make life choices' column is used as the indicator of freedom in the respective datasets.

The mean freedom scores are calculated separately for each year.

Sorting and Display:

The mean freedom scores are sorted in ascending order using the sort_values() function. The highest and lowest mean freedom scores are displayed for each year, representing countries with high and low levels of freedom, respectively.

Observations:

The output displays the countries with high and low levels of freedom for each year separately. Countries with high levels of freedom are listed at the top, while countries with low levels of freedom are listed at the bottom.

The specific countries and their rankings provide insights into the distribution of freedom scores across different years.

By comparing the rankings across years, one can observe any changes or trends in the level of freedom in different countries.

It is important to note that the interpretation of freedom scores should be done in the context of the specific dataset and variables used. Other factors and variables may influence the level of freedom, and further analysis can be conducted to explore the relationships and patterns in more depth.

TASK 2 - Classification/Regression

Perform following steps on the same dataset which you used for EDA.

> - Data Preprocessing (as per requirement)
> - Feature Engineering
> - Split dataset in train-test (80:20 ratio)
> - Model selection
> - Model training
> - Model evaluation
> - Fine-tune the Model
> - Make predictions

Summarize your model's performance by evaluation metrices

Data Understanding and Preprocessing

```
#Data Preprocessing



import pandas as pd                 # for data manipulation

import numpy as np                  # for mathematical calculations

import seaborn as sns               # for data visualization

import matplotlib.pyplot as plt     # for plotting graphs
plt.style.use('seaborn')            # the seaborn stylesheet will make our plots look neat and pretty.

%matplotlib inline
# "%matplotlib inline" ensures commands in cells below the cell that outputs a plot does not affect the plot

import warnings                     # to ignore any warnings
warnings.filterwarnings("ignore")
```

```
<ipython-input-1-bdcdfe7e01bc>:8: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6,
  plt.style.use('seaborn')            # the seaborn stylesheet will make our plots look neat and pretty.
```

```
# Loading the data

whr_2015 = pd.read_csv('2015.csv')
whr_2016 = pd.read_csv('2016.csv')
```

```
whr_2017 = pd.read_csv('2017.csv')
whr_2018 = pd.read_csv('2018.csv')
whr_2019 = pd.read_csv('2019.csv')
#previewing 2015 report
# .head() returns the first 5 rows in the dataframe

whr_2015.head()

# Alternatively, you can use 'train.sample(5)'' to get the same output
```

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystc Resic
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2.5
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2.7(
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.4
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.66973	0.36503	0.34699	2.4
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.63297	0.32957	0.45811	2.4

```
#previewing 2016 report

whr_2016.head()
```

	Country	Region	Happiness Rank	Happiness Score	Lower Confidence Interval	Upper Confidence Interval	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Ger
0	Denmark	Western Europe	1	7.526	7.460	7.592	1.44178	1.16374	0.79504	0.57941	0.44453	
1	Switzerland	Western Europe	2	7.509	7.428	7.590	1.52733	1.14524	0.86303	0.58557	0.41203	
2	Iceland	Western Europe	3	7.501	7.333	7.669	1.42666	1.18326	0.86733	0.56624	0.14975	
3	Norway	Western Europe	4	7.498	7.421	7.575	1.57744	1.12690	0.79579	0.59609	0.35776	
4	Finland	Western Europe	5	7.413	7.351	7.475	1.40598	1.13464	0.81091	0.57104	0.41004	

```
#previewing 2017 report

whr_2017.head()
```

	Country	Happiness.Rank	Happiness.Score	Whisker.high	Whisker.low	Economy..GDP.per.Capita.	Family	Health..Life.E
0	Norway	1	7.537	7.594445	7.479556	1.616463	1.533524	
1	Denmark	2	7.522	7.581728	7.462272	1.482383	1.551122	
2	Iceland	3	7.504	7.622030	7.385970	1.480633	1.610574	
3	Switzerland	4	7.494	7.561772	7.426227	1.564980	1.516912	
4	Finland	5	7.469	7.527542	7.410458	1.443572	1.540247	

```
#previewing 2018 report

whr_2018.head()
```

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.632	1.305	1.592	0.874	0.681	0.202	0.393
1	2	Norway	7.594	1.456	1.582	0.861	0.686	0.286	0.340
2	3	Denmark	7.555	1.351	1.590	0.868	0.683	0.284	0.408
3	4	Iceland	7.495	1.343	1.644	0.914	0.677	0.353	0.138
4	5	Switzerland	7.487	1.420	1.549	0.927	0.660	0.256	0.357

```
#previewing 2019 report

whr_2019.head()
```

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298

```
years = [whr_2015, whr_2016, whr_2017, whr_2018, whr_2019]

for year in years:
    print(year.shape)                                           # returns the no. of rows and columns
    print(year.columns)
```

```
(158, 12)
Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
       'Standard Error', 'Economy (GDP per Capita)', 'Family',
       'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
       'Generosity', 'Dystopia Residual'],
      dtype='object')
(157, 13)
Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
       'Lower Confidence Interval', 'Upper Confidence Interval',
       'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)',
       'Freedom', 'Trust (Government Corruption)', 'Generosity',
       'Dystopia Residual'],
      dtype='object')
(155, 12)
Index(['Country', 'Happiness.Rank', 'Happiness.Score', 'Whisker.high',
       'Whisker.low', 'Economy..GDP.per.Capita.', 'Family',
       'Health..Life.Expectancy.', 'Freedom', 'Generosity',
       'Trust..Government.Corruption.', 'Dystopia.Residual'],
      dtype='object')
(156, 9)
Index(['Overall rank', 'Country or region', 'Score', 'GDP per capita',
       'Social support', 'Healthy life expectancy',
       'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption'],
      dtype='object')
(156, 9)
Index(['Overall rank', 'Country or region', 'Score', 'GDP per capita',
       'Social support', 'Healthy life expectancy',
       'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption'],
      dtype='object')
```

```
years = [whr_2015, whr_2016, whr_2017, whr_2018, whr_2019]

for year in years:
    print(year.shape)                                           # returns the no. of rows and columns
    print(year.columns)
```

```
(158, 12)
Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
       'Standard Error', 'Economy (GDP per Capita)', 'Family',
       'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
       'Generosity', 'Dystopia Residual'],
      dtype='object')
```

```
(157, 13)
Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
       'Lower Confidence Interval', 'Upper Confidence Interval',
       'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)',
       'Freedom', 'Trust (Government Corruption)', 'Generosity',
       'Dystopia Residual'],
      dtype='object')
(155, 12)
Index(['Country', 'Happiness.Rank', 'Happiness.Score', 'Whisker.high',
       'Whisker.low', 'Economy..GDP.per.Capita.', 'Family',
       'Health..Life.Expectancy.', 'Freedom', 'Generosity',
       'Trust..Government.Corruption.', 'Dystopia.Residual'],
      dtype='object')
(156, 9)
Index(['Overall rank', 'Country or region', 'Score', 'GDP per capita',
       'Social support', 'Healthy life expectancy',
       'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption'],
      dtype='object')
(156, 9)
Index(['Overall rank', 'Country or region', 'Score', 'GDP per capita',
       'Social support', 'Healthy life expectancy',
       'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption'],
      dtype='object')
```

```
# dropping irrelevant columns
# (inplace=True) argument means that changes made to the dataframe remains permanent.

whr_2015.drop(columns=['Standard Error', 'Region', 'Dystopia Residual'], inplace=True)

whr_2016.drop(columns=['Lower Confidence Interval', 'Upper Confidence Interval', 'Region', 'Dystopia Residual'], inplace=True)

whr_2017.drop(columns=['Whisker.high', 'Whisker.low', 'Dystopia.Residual'], inplace=True)

#nothing to drop for 2018 and 2019 dataframe
```

```
# Adding a new column 'Year' to indicate the year of report collation

whr_2015['Year'] = 2015
whr_2016['Year'] = 2016
whr_2017['Year'] = 2017
whr_2018['Year'] = 2018
whr_2019['Year'] = 2019
```

Double-click (or enter) to edit

Double-click (or enter) to edit

```
# Reordering the columns to acheive uniformity

whr_2015 = whr_2015[['Happiness Rank', 'Country', 'Happiness Score', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'F
whr_2016 = whr_2016[['Happiness Rank', 'Country', 'Happiness Score', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'F
whr_2017 = whr_2017[['Happiness.Rank', 'Country', 'Happiness.Score', 'Economy..GDP.per.Capita.', 'Family', 'Health..Life.Expectancy.', 'F

# whr_2018 and whr_2019 already have the correct order so no need to reorder
```


```
# Reordering the columns to acheive uniformity

whr_2015 = whr_2015[['Happiness Rank', 'Country', 'Happiness Score', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'F
whr_2016 = whr_2016[['Happiness Rank', 'Country', 'Happiness Score', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'F
whr_2017 = whr_2017[['Happiness.Rank', 'Country', 'Happiness.Score', 'Economy..GDP.per.Capita.', 'Family', 'Health..Life.Expectancy.', 'F

# whr_2018 and whr_2019 already have the correct order so no need to reorder
```


```
#renaming the columns to ensure all columns have same name across the years

#New column names = new_cols
new_cols = ['Happiness Rank', 'Country', 'Happiness Score', 'GDP per capita', 'Social Support', 'Healthy Life Expectancy', 'Freedom to Ma

years = [whr_2015, whr_2016, whr_2017, whr_2018, whr_2019]
for year in years:
    year.columns = new_cols
    #print(year.columns)
```

Features Description

Happiness Rank: A country's rank on a world scale - determined by how high their happiness score is.

Happiness Score: A score given to a country based on adding up the rankings that a population has given to each category (normalized)

Country: The country in question

GDP per capita: individuals rank they quality of life based on the amount they earn

Social Support: quality of family life, nuclear and joint family

Healthy Life Expectancy: ranking healthcare availability and average life expectancy in the country

Freedom to make life choices: how much an individual is able to conduct them self based on their free will

Perceptions of Corruption: Trust in the government to not be corrupt

Generosity: how much their country is involved in peacekeeping and global aid

```
#merging all 5 dataframes into one

whr_all = [whr_2015, whr_2016, whr_2017, whr_2018, whr_2019]

happiness = pd.concat(whr_all)

happiness.head()
```

	Happiness Rank	Country	Happiness Score	GDP per capita	Social Support	Healthy Life Expectancy	Freedom to Make Life Choices	Generosity	Perceptions of Corruption	Year
0	1	Switzerland	7.587	1.39651	1.34951	0.94143	0.66557	0.29678	0.41978	2015
1	2	Iceland	7.561	1.30232	1.40223	0.94784	0.62877	0.43630	0.14145	2015
2	3	Denmark	7.527	1.32548	1.36058	0.87464	0.64938	0.34139	0.48357	2015
3	4	Norway	7.522	1.45900	1.33095	0.88521	0.66973	0.34699	0.36503	2015
4	5	Canada	7.427	1.32629	1.32261	0.90563	0.63297	0.45811	0.32957	2015

```
# This is used to view basic statistical details like percentile, mean, std etc.

happiness.describe(include='all')
```

	Happiness Rank	Country	Happiness Score	GDP per capita	Social Support	Healthy Life Expectancy	Freedom to Make Life Choices	Generosity	Perceptions of Corruption	Yea
count	782.000000	782	782.000000	782.000000	782.000000	782.000000	782.000000	782.000000	781.000000	782.00000
unique	NaN	170	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
top	NaN	Switzerland	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
freq	NaN	5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
mean	78.698210	NaN	5.379018	0.916047	1.078392	0.612416	0.411091	0.218576	0.125436	2016.99360
std	45.182384	NaN	1.127456	0.407340	0.329548	0.248309	0.152880	0.122321	0.105816	1.41736
min	1.000000	NaN	2.693000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2015.00000
25%	40.000000	NaN	4.509750	0.606500	0.869363	0.440183	0.309768	0.130000	0.054000	2016.00000
50%	79.000000	NaN	5.322000	0.982205	1.124735	0.647310	0.431000	0.201982	0.091000	2017.00000
75%	118.000000	NaN	6.189500	1.236187	1.327250	0.808000	0.531000	0.278832	0.156030	2018.00000

```
# Checking to see if any feature has empty/missing values

happiness.isnull().sum()
```

```
Happiness Rank                0
Country                       0
```

```
Happiness Score                0
GDP per capita                 0
Social Support                 0
Healthy Life Expectancy        0
Freedom to Make Life Choices   0
Generosity                     0
Perceptions of Corruption      1
Year                           0
dtype: int64
```

```
# filling the empty row with the median of the colunm

median = happiness['Perceptions of Corruption'].median()
#print(median)

happiness['Perceptions of Corruption'].fillna(median, inplace = True)
# checking for duplicate values

happiness.duplicated().sum()
```

```
0
```

```
# The info() function is used to print a concise summary of a DataFrame

happiness.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 782 entries, 0 to 155
Data columns (total 10 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Happiness Rank                782 non-null    int64  
 1   Country                       782 non-null    object 
 2   Happiness Score               782 non-null    float64
 3   GDP per capita                782 non-null    float64
 4   Social Support                782 non-null    float64
 5   Healthy Life Expectancy       782 non-null    float64
 6   Freedom to Make Life Choices  782 non-null    float64
 7   Generosity                    782 non-null    float64
 8   Perceptions of Corruption     782 non-null    float64
 9   Year                          782 non-null    int64  
dtypes: float64(7), int64(2), object(1)
memory usage: 67.2+ KB
```

We see that after cleaning we have 782 rows of clean data with no null values. There are 10 columns and three dtypes(int, float and object)

The data is now clean and void of unnecessary features, we can now proceed to visualizing the data to see the relationship between features

```
#happiness.to_csv('cleaned_happiness.csv', index =False)
```

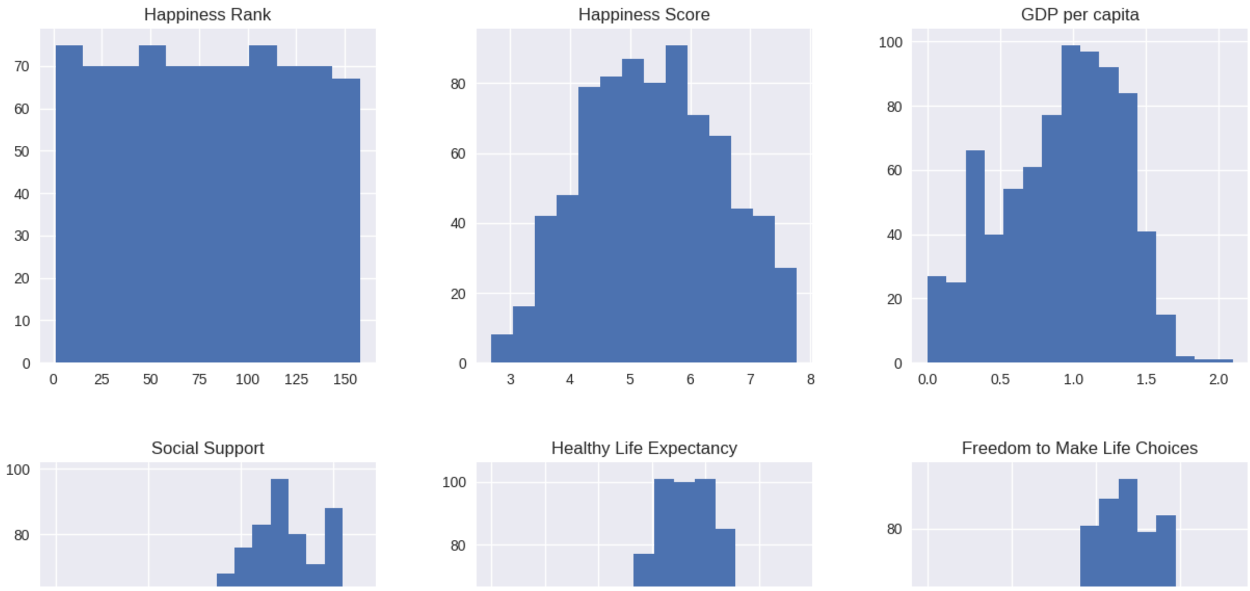
Data Visualization and Analysis

Univariate Plots

This is the plots of each individual variable. They help us understand each attribute better.

```
# checking the frequency distribution of the variables

happiness.hist(bins='auto', figsize=(15,15))
```

```
array([[<Axes: title={'center': 'Happiness Rank'}>,
        <Axes: title={'center': 'Happiness Score'}>,
        <Axes: title={'center': 'GDP per capita'}>],
       [<Axes: title={'center': 'Social Support'}>,
        <Axes: title={'center': 'Healthy Life Expectancy'}>,
        <Axes: title={'center': 'Freedom to Make Life Choices'}>],
       [<Axes: title={'center': 'Generosity'}>,
        <Axes: title={'center': 'Perceptions of Corruption'}>,
        <Axes: title={'center': 'Year'}>]], dtype=object)
```




```
# We can check for outliers using boxplots

happiness[['Happiness Score', 'GDP per capita', 'Social Support', 'Healthy Life Expectancy', 'Freedom to Make Life Choices']]
```

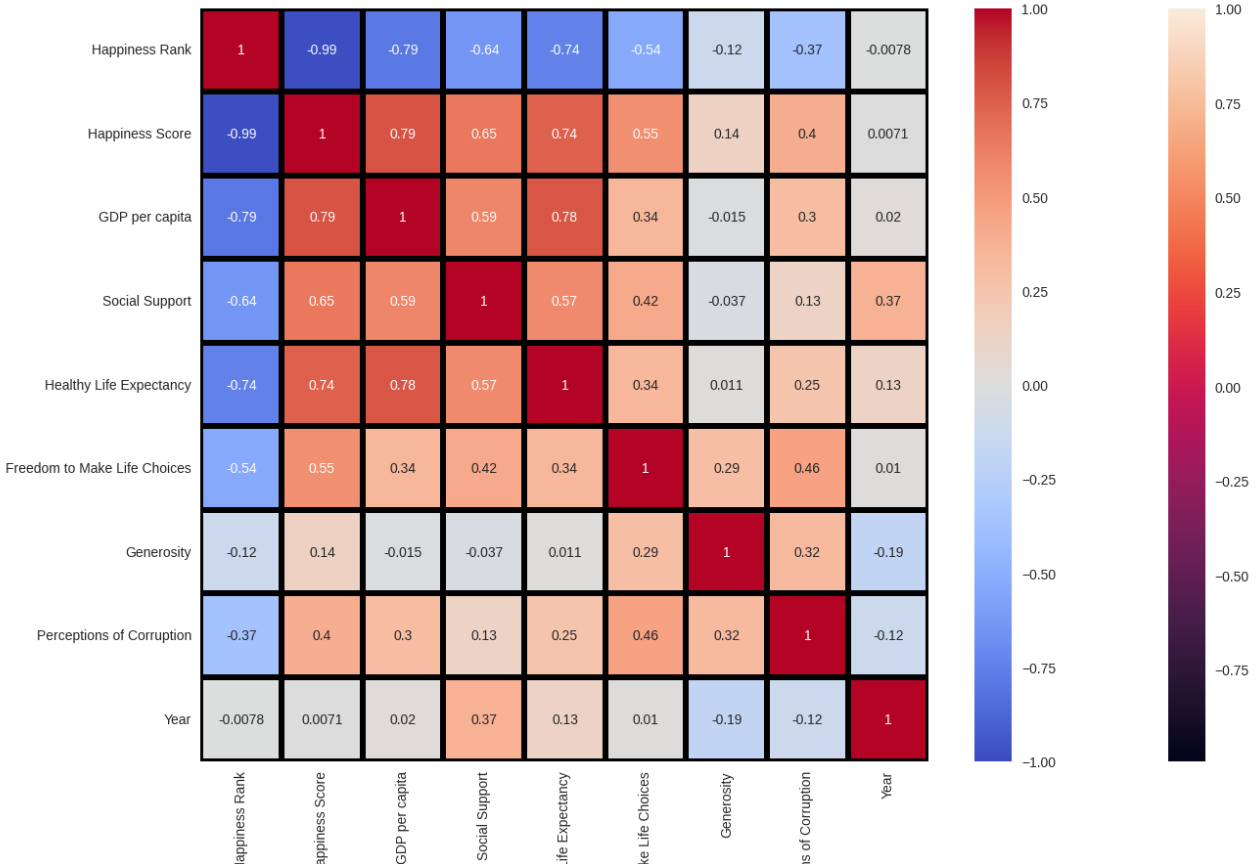
	Happiness Score	GDP per capita	Social Support	Healthy Life Expectancy	Freedom to Make Life Choices
0	7.587	1.39651	1.34951	0.94143	0.66557
1	7.561	1.30232	1.40223	0.94784	0.62877
2	7.527	1.32548	1.36058	0.87464	0.64938
3	7.522	1.45900	1.33095	0.88521	0.66973
4	7.427	1.32629	1.32261	0.90563	0.63297
...	...	...	...	...	...
151	3.334	0.35900	0.71100	0.61400	0.55500
152	3.231	0.47600	0.88500	0.49900	0.41700
153	3.203	0.35000	0.51700	0.36100	0.00000
154	3.083	0.02600	0.00000	0.10500	0.22500
155	2.853	0.30600	0.57500	0.29500	0.01000

782 rows × 5 columns

Bivariate Plots

This is used to understand the relationship between variables

```
# let's see the correlation between the features
# Checking the correlation of features helps us decide which features affect the target variable the most, and in turn, get used in predi



plt.figure(figsize=(15, 10))              # This specifies the size, the bigger the map, the easier we can understand the map

sns.heatmap(happiness.corr())             # This is sufficient but adding the 'annot' argument makes interpretaton easier

sns.heatmap(happiness.corr(), annot = True, vmin=-1, vmax=1, center= 0, cmap= 'coolwarm', linewidths=3, linecolor='black')  # 'annot' hel


plt.show()
```

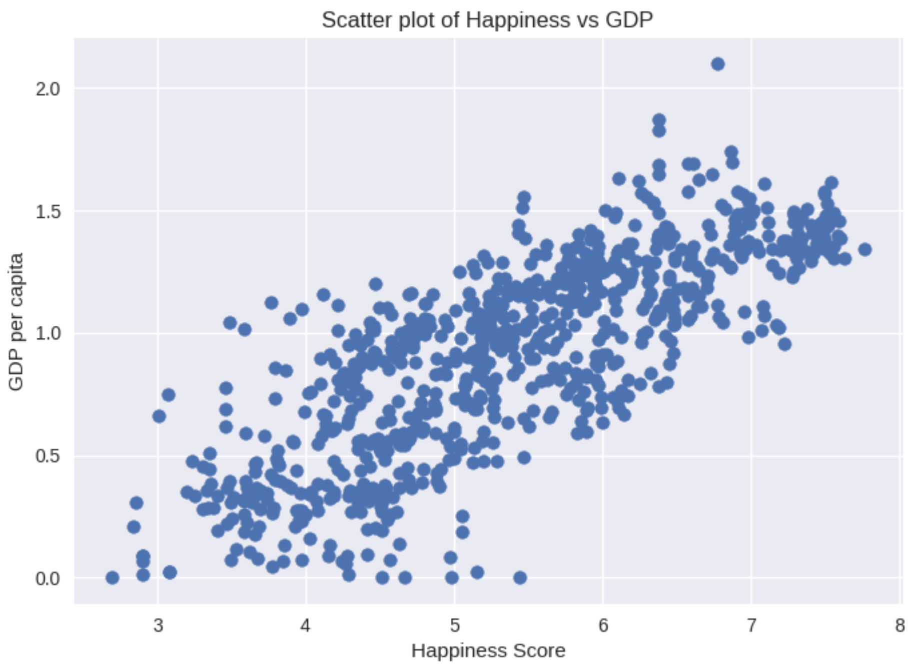


The darker the box, the stronger/higher the correlation.

Happiness Score correlates strongly with GDP per capita and Healthy Life Expectancy. It has low correlates with Generosity and Perceptions of Corruption.

Also, there is an inverse relationship between Happiness rank and Happiness score; the higher the score, the lower the rank.

```
# lets's further investigate the relationship between happiness score and GDP

happiness_score = happiness['Happiness Score']
gdp= happiness['GDP per capita']

plt.scatter(happiness_score, gdp)
plt.title('Scatter plot of Happiness vs GDP')
plt.xlabel('Happiness Score')
plt.ylabel('GDP per capita')
plt.show()
```



The diagram above shows that, the higher the gdp, the higher the happiness score

```
# happiest countries

happy_countries = happiness[['Country', 'Happiness Rank']].groupby('Country').mean().sort_values(by = 'Happiness Rank', ascending = True)
happy_countries.head()
```

Country	Happiness Rank
Denmark	2.2
Norway	2.8
Iceland	3.2
Switzerland	3.6
Finland	3.6

```
# saddest countries

sad_countries = happiness[['Country', 'Happiness Rank']].groupby('Country').mean().sort_values(by = 'Happiness Rank', ascending = True)
sad_countries.tail()
```

Country	Happiness Rank
Tanzania	150.80
Rwanda	152.00
Syria	152.60
Central African Republic	153.25
Burundi	153.80

```
# What countries have the highest GDP per capita over the years?

rich_countries = happiness[['Country', 'GDP per capita']].groupby('Country').mean().sort_values(by='GDP per capita', ascending=False)
rich_countries.head()
```

Country	GDP per capita
Qatar	1.743691
United Arab Emirates	1.645227
Luxembourg	1.637675
Singapore	1.592138
Kuwait	1.555662

Modeling and Prediction

Now that the data is clean and we have an understandng of the variables, we can now construct a model.

First, we drop any categorical variables, and the happiness rank as that is not something we are exploring in this report.

```
# We'll drop the Country variable because it's categorical, we'll also drop the happiness rank and year variable beacuse it's irrelevant
# this leaves only numerical features in the data frame


new_happiness = happiness.drop(['Country', 'Happiness Rank', 'Year'], axis=1)
new_happiness.head()
#new_happiness.info()
```

	Happiness Score	GDP per capita	Social Support	Healthy Life Expectancy	Freedom to Make Life Choices	Generosity	Perceptions of Corruption
0	7.587	1.39651	1.34951	0.94143	0.66557	0.29678	0.41978

```
# let's split our data into training(80%) and testing(20%) sets

from sklearn.model_selection import train_test_split

# features with low corelation has been removed

X = new_happiness[['GDP per capita', 'Social Support', 'Healthy Life Expectancy', 'Freedom to Make Life Choices']]
y = new_happiness['Happiness Score']

# X = features, y = target variable

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)


# Next we need to scale the data before feeding it to the model
# To standardize our data, we need to import the StandardScaler from the sklearn library

from sklearn.preprocessing import StandardScaler

scale = StandardScaler()

new_happiness = scale.fit_transform(new_happiness)
# Training the algorithm

from sklearn.linear_model import LinearRegression

lm = LinearRegression()                         # instantiating the model

lm.fit(X_train, y_train)                        # fitting the model with the training dataset


#print(lm.coef_)
```

```
▾ LinearRegression
LinearRegression()
```

```
# making predictions on the test data

y_pred = lm.predict(X_test)

# comparing actual values with predicted values
actual_vs_pred = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
actual_vs_pred
```

	Actual	Predicted
67	5.525	5.750436
17	6.886	6.626951
36	6.344	6.278800
14	7.119	6.560101
145	3.781	4.192592
...	...	...
64	5.709	5.547671
152	3.303	4.678981
52	5.878	5.596648
21	6.627	6.750217
127	4.340	5.045486

157 rows × 2 columns

```
coefficient = lm.coef_

#making a dataframe of the coeffiecients to help us easily determine which variable carries more weight


coefficient_df = pd.DataFrame(list(zip(X.columns, lm.coef_)), columns=['features', 'coefficients'])
coefficient_df
```

	features	coefficients
0	GDP per capita	1.210445
1	Social Support	0.470579
2	Healthy Life Expectancy	1.041019
3	Freedom to Make Life Choices	1.889035

From the output above, we see that, Freedom to make life choices not GDP per capita is the most important factor contributing to happiness of citizens, as opposed to the result we got from the heatmap.

This shows that correlation doesn't necessarily mean causation. Health and social support are also important but carry less weight in this model.

Model Evaluation

For this model, I will use the most common evaluation metric for regressions: Root Mean Squared Error (RMSE) is the square root of the mean of the squared errors

```
from sklearn import metrics
from sklearn.metrics import mean_squared_error as MSE
```

```
print('Root Mean Squared Error:', np.sqrt(MSE(y_test, y_pred)))
```

```
Root Mean Squared Error: 0.5601640575464227
```

The lower the RMSE, the better the model is at making predictions.

The RMSE is low, further feature engineering can lead to a lower RMSE score.

Conclusion

Answer to the question posed at the beginning; What factors influence the happiness of citizens the most?

GDP per capita, Freedom to make life choices and Life expectancy are great determinants of Happiness score and can be used to predict the future scores. However, this is not conclusive because unforeseen occurences like pandemic, natural disasters and economic meltdown happen, even to the most stable countries so these scores can actually change.

```
%who
```

```
LinearRegression         MSE      StandardScaler  X        X_test  X_train         actual_vs_pred  coefficient     coefficient_df
gdp       happiness       happiness_score         happy_countries         lm      median  metrics         new_cols        new_happin
np        pd      plt     rich_countries  sad_countries   scale   sns     train_test_split        warnings
whr_2015          whr_2016        whr_2017        whr_2018        whr_2019        whr_all         y       y_pred  y_test
y_train   year    years
```

```
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:100% !important; }</style>"))
```