Forecasting 2024 Global Happiness Trends: Exploring the Impact of Cumulative Socio-Economic Influences

Overview

In our project, titled "**Forecasting 2024 Global Happiness Trends: Exploring the Impact of Cumulative Socio-Economic Influences**," our team aims to analyze and predict happiness trends worldwide for the year 2024. We will explore how various socio-economic factors influence happiness levels across different countries, utilizing comprehensive datasets spanning from 2020 to 2023, focusing on **GDP**, **layoff statistics**, **income group classifications**, and reports on **global well-being**.

Our primary objective is **to understand the cumulative impact of socio-economic indicators on future happiness trends**. To accomplish this, we will begin by merging the different datasets. This step involves integrating the data from multiple sources using common identifiers such as country names or codes. By merging the datasets, we ensure that all relevant information is consolidated into a single cohesive dataset, facilitating further analysis. Following data merging, we'll employ Python libraries such as *pandas*, *NumPy*, *Matplotlib*, *Seaborn*, and others for data manipulation and visualization. This phase will involve preprocessing the merged dataset to handle missing values, outliers, and inconsistencies. Subsequently, we will conduct exploratory data analysis to gain insights into the distributions, trends, and interrelationships within the data.

Once we have thoroughly explored the datasets and gained valuable insights, we will proceed to develop predictive models. We will employ advanced statistical techniques and machine learning algorithms such as clustering, regression analysis, random forests, et cetera to forecast happiness scores for 2024 based on the socio-economic factors identified in our analysis.

Throughout the project, our team will maintain a multidisciplinary approach, combining expertise in data analysis, econometrics, and social sciences. We will evaluate the performance of our predictive models using appropriate metrics and validate their accuracy through cross-validation techniques.

Ultimately, our project aims to contribute valuable insights into the complex

interplay between socio-economic conditions and global happiness levels. By forecasting future happiness trends, we hope to inform policymakers, researchers, and stakeholders about the potential long-term implications of socio-economic policies on societal well-being. Our findings may aid in the development of evidence-based strategies to promote happiness and improve the quality of life for individuals and communities worldwide.

Motivation

Our team selected the topic of "**Forecasting 2024 Global Happiness Trends: Exploring the Impact of Cumulative Socio-Economic Influences**" due to its relevance and significance in understanding societal well-being and the factors that contribute to it. We believe that happiness is a crucial aspect of human life, and gaining insights into the drivers of happiness can inform policies and interventions aimed at promoting overall societal welfare.

One of the primary reasons for choosing this topic is its interdisciplinary nature. By integrating socio-economic factors such as GDP, layoff statistics, and income group classifications, along with reports on global well-being, we aim to examine the complex interplay between various determinants of happiness. This holistic approach allows us to explore how economic conditions, employment stability, income distribution, social support, and other socio-economic indicators collectively influence happiness levels across different countries.

In the course of our project, we have generated three real-world questions about the data:

1. **How do changes in GDP impact happiness levels across different countries over time?** By analyzing the relationship between GDP fluctuations and happiness scores, we hope to understand the extent to which economic growth contributes to overall well-being. This question will provide insights into whether increases in GDP translate into higher happiness levels and whether there are disparities in this relationship among countries.
2. **What is the relationship between layoff rates and happiness levels?** Investigating the correlation between layoff rates and happiness levels will shed light on the psychological and emotional impact of job insecurity on individuals and communities. Understanding how employment stability affects happiness can inform labor market policies and interventions aimed at mitigating the adverse effects of layoffs on well-being.
3. **Do income group classifications correlate with happiness scores, and if**

so, how? Exploring the relationship between income group classifications and happiness scores will help us understand the role of income inequality in shaping subjective well-being. By examining whether individuals in different income groups report varying levels of happiness, we aim to identify potential socio-economic disparities in happiness and inform policies aimed at reducing inequality and promoting social inclusion.

Through these questions, we hope to gain valuable insights into the mechanisms underlying global happiness trends and the socio-economic factors that contribute to them. By answering these questions, our project aims to provide evidence-based recommendations to policymakers, researchers, and stakeholders to enhance overall societal well-being and quality of life.

Data Sources

1. Global GDP Growth Rate 2020-2023
2. Global Layoffs 2020-2023
3. World Income Groups 2022
4. World Happiness Report (2020-2022) | World Happiness Report 2023

The datasets selected for this analysis offer complementary perspectives on global socio-economic trends, providing a comprehensive understanding of the factors influencing happiness levels worldwide. The Global GDP Growth Rate dataset from the International Monetary Fund (IMF) offers insights into the economic performance of countries over the period 2020-2023, highlighting variations in economic growth rates across different regions. This information is complemented by the Global Layoffs dataset, sourced from Kaggle, which provides crucial insights into labor market dynamics and employment trends during the same period. Additionally, the World Income Groups dataset further enriches the analysis by categorizing countries into income groups, allowing for a nuanced examination of how economic disparities may impact happiness levels. Finally, the World Happiness Report datasets offer direct measures of subjective well-being, providing valuable context for understanding the socio-economic factors that contribute to happiness trends. By integrating these datasets, we can gain a holistic understanding of the interplay between economic performance, employment dynamics, income distribution, and subjective well-being, ultimately informing evidence-based strategies for promoting global happiness and well-being.

Project Part I - Data Description & Manipulation

Data Description

1. Global GDP Growth Rate 2020-2023

- **Variables of Interest:** Country (GDP Growth - Annual Percentage Change), Years (2020, 2021, 2022, 2023)
- **Size of the Dataset:** 231 countries × 50 years = 11550 observations
- **Missing Values:** Several yearly GDP growth percentage change data points missing; further data preprocessing may be necessary.

2. Global Layoffs 2020-2023

- **Variables of Interest:** Country, Layoff Details (Date, Total Number of Layoffs, Total Layoff Percentage)
- **Size of the Dataset:** 3313 companies x 9 layoff details = 29817 observations
- **Missing Values:** Several data points pertaining to layoff details missing; further data preprocessing may be necessary.

3. World Income Groups 2022

- **Variables of Interest:** Country Code, Region, Income Group (High, Upper Middle, Lower Middle, Low)
- **Size of the Dataset:** 217 countries x 3 income details = 651 observations.
- **Missing Values:** No missing values.

4. World Happiness Report (2020-2022) & World Happiness Report 2023

- **Variables of Interest:** Country Name, Region, Happiness Score, GDP per Capita, Social Support, Healthy Life Expectancy, Freedom to Make Life Choices, Perceptions of Corruption
- **Size of the Dataset:** Approximately 150 countries × 20 well-being details = 3000 approximate observations for each of the four datasets ranging from 2020-2023.
- **Missing Values:** A few insignificant null values in World Happiness Report 2022.

In [800…
```
import pandas as pd
import numpy as np
import pycountry
import pycountry_convert as pc
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
from scipy.stats import f_oneway
import statsmodels.api as sm
```

Global Layoffs Dataset

In [801…
```
# Read in the data
layoff_df = pd.read_csv('global_layoffs.csv')
layoff_df
```

Out[801…

	company	location	industry	total_laid_off	percentage_laid_of
0	New Work	Hamburg	Consumer	400.0	NaN
1	Playtika	Tel Aviv	Consumer	300.0	0.1(
2	Discord	SF Bay Area	Consumer	170.0	0.1
3	Inmobi	Bengaluru	Marketing	125.0	0.05
4	Audible	New York City	Media	100.0	0.05
...	...	...	...	...	..
3308	Service	Los Angeles	Travel	NaN	1.0(
3309	HopSkipDrive	Los Angeles	Transportation	8.0	0.1(
3310	Panda Squad	SF Bay Area	Consumer	6.0	0.75
3311	Tamara Mellon	Los Angeles	Retail	20.0	0.4(
3312	EasyPost	Salt Lake City	Logistics	75.0	NaN

3313 rows × 9 columns

In [802…
```
# Select the columns we want to keep
```

```
layoff_refined = layoff_df[['country', 'date',
                            'total_laid_off', 'percentage_laid_off']]
layoff_refined
```

Out[802…

	country	date	total_laid_off	percentage_laid_off
0	Germany	2024-01-11	400.0	NaN
1	Israel	2024-01-11	300.0	0.10
2	United States	2024-01-11	170.0	0.17
3	India	2024-01-11	125.0	0.05
4	United States	2024-01-11	100.0	0.05
...	...	...	...	...
3308	United States	2020-03-16	NaN	1.00
3309	United States	2020-03-13	8.0	0.10
3310	United States	2020-03-13	6.0	0.75
3311	United States	2020-03-12	20.0	0.40
3312	United States	2020-03-11	75.0	NaN

3313 rows × 4 columns

In [803…

```
# Find missing values
layoff_refined.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3313 entries, 0 to 3312
Data columns (total 4 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   country              3313 non-null   object 
 1   date                 3313 non-null   object 
 2   total_laid_off       2189 non-null   float64
 3   percentage_laid_off  2141 non-null   float64
dtypes: float64(2), object(2)
memory usage: 103.7+ KB
```

In [804…

```
# Drop missing values
layoff_refined = layoff_refined.dropna()
```

In [805…

```
# Count the data for every country
layoff_refined["country"].value_counts()
```

Out[805…

```
country
United States           997
India                   107
Canada                   85
Brazil                   56
Israel                   52
United Kingdom           51
Germany                  49
Australia                29
Singapore                23
Sweden                   14
Indonesia                10
Netherlands               8
Kenya                     6
Nigeria                   6
France                    6
Ireland                   5
New Zealand               4
United Arab Emirates      4
Argentina                 4
Estonia                   4
Norway                    3
Hong Kong                 3
China                     3
Denmark                   2
Spain                     2
Austria                   2
Chile                     2
South Korea               2
Mexico                    2
Malaysia                  2
Portugal                  1
Switzerland               1
Bulgaria                  1
Seychelles                1
Russia                    1
Poland                    1
Finland                   1
Senegal                   1
Thailand                  1
Luxembourg                1
Saudi Arabia              1
Myanmar                   1
Name: count, dtype: int64
```

In [806…

```
# Make a copy of the data frame
layoff_refined = layoff_refined.copy()

# Rename the columns to be more descriptive and consistent
layoff_refined.rename(columns={'country': 'Country', 'date': 'Date',
                      'total_laid_off': 'Total Layoffs', 'percentage_l
layoff_refined
```

Out [806…

	Country	Date	Total Layoffs	Layoff Percentage
1	Israel	2024-01-11	300.0	0.10
2	United States	2024-01-11	170.0	0.17
3	India	2024-01-11	125.0	0.05
4	United States	2024-01-11	100.0	0.05
5	United States	2024-01-11	60.0	0.13
...	...	...	...	...
3306	United States	2020-03-16	130.0	0.22
3307	United States	2020-03-16	16.0	1.00
3309	United States	2020-03-13	8.0	0.10
3310	United States	2020-03-13	6.0	0.75
3311	United States	2020-03-12	20.0	0.40

1555 rows × 4 columns

In [807…

```
# Convert 'date' column to datetime if it's not already in datetime fo
layoff_refined['Date'] = pd.to_datetime(layoff_refined['Date'])

# Extract the year from the 'date' column
layoff_refined['Year'] = layoff_refined['Date'].dt.year

# Filter the DataFrame to include only rows with years 2020, 2021, 202
layoff_filtered = layoff_refined[layoff_refined['Year'].isin(
    [2020, 2021, 2022, 2023])]
layoff_filtered
```

Out [807…

	Country	Date	Total Layoffs	Layoff Percentage	Year
41	India	2023-12-20	200.0	0.15	2023
42	United States	2023-12-19	100.0	0.20	2023
44	United States	2023-12-18	350.0	0.10	2023
45	India	2023-12-18	100.0	0.10	2023
51	United States	2023-12-14	900.0	0.24	2023
...	...	...	...	...	...
3306	United States	2020-03-16	130.0	0.22	2020
3307	United States	2020-03-16	16.0	1.00	2020
3309	United States	2020-03-13	8.0	0.10	2020
3310	United States	2020-03-13	6.0	0.75	2020
3311	United States	2020-03-12	20.0	0.40	2020

1538 rows × 5 columns

In [808…

```
# Make a copy of the data frame
layoff_filtered = layoff_filtered.copy()

# Drop the Date column
layoff_filtered.drop(columns='Date', axis=1, inplace=True)
layoff_filtered
```

Out[808…

	Country	Total Layoffs	Layoff Percentage	Year
41	India	200.0	0.15	2023
42	United States	100.0	0.20	2023
44	United States	350.0	0.10	2023
45	India	100.0	0.10	2023
51	United States	900.0	0.24	2023
...	...	...	...	...
3306	United States	130.0	0.22	2020
3307	United States	16.0	1.00	2020
3309	United States	8.0	0.10	2020
3310	United States	6.0	0.75	2020
3311	United States	20.0	0.40	2020

1538 rows × 4 columns

In [809…

```
# Reset the index
layoff_filtered.reset_index(drop=True, inplace=True)
layoff_filtered
```

Out[809…

	Country	Total Layoffs	Layoff Percentage	Year
0	India	200.0	0.15	2023
1	United States	100.0	0.20	2023
2	United States	350.0	0.10	2023
3	India	100.0	0.10	2023
4	United States	900.0	0.24	2023
...	...	...	...	...
1533	United States	130.0	0.22	2020
1534	United States	16.0	1.00	2020
1535	United States	8.0	0.10	2020
1536	United States	6.0	0.75	2020
1537	United States	20.0	0.40	2020

1538 rows × 4 columns

In [810…
```
# Group the data by 'Country' and 'Year' and sum the 'Total Layoffs' f
layoff_counts = layoff_filtered.groupby(['Country', 'Year'])[
    'Total Layoffs'].sum()

# Convert the groupby object to a DataFrame
layoff_counts_df = pd.DataFrame(layoff_counts).reset_index()
layoff_counts_df
```

Out[810…

	Country	Year	Total Layoffs
0	Argentina	2022	283.0
1	Australia	2020	96.0
2	Australia	2022	1088.0
3	Australia	2023	1702.0
4	Austria	2022	270.0
...	...	...	...
79	United Kingdom	2023	6818.0
80	United States	2020	45082.0
81	United States	2021	6150.0
82	United States	2022	97176.0
83	United States	2023	136900.0

84 rows × 3 columns

In [811…
```
# Checking to see if there are any null values
layoff_counts_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 84 entries, 0 to 83
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Country        84 non-null     object
 1   Year           84 non-null     int32
 2   Total Layoffs  84 non-null     float64
dtypes: float64(1), int32(1), object(1)
memory usage: 1.8+ KB
```

In [812…
```
# Pivot the data frame
layoff_final_df = layoff_counts_df.pivot(
    index='Country', columns='Year', values='Total Layoffs').dropna()
layoff_final_df
```

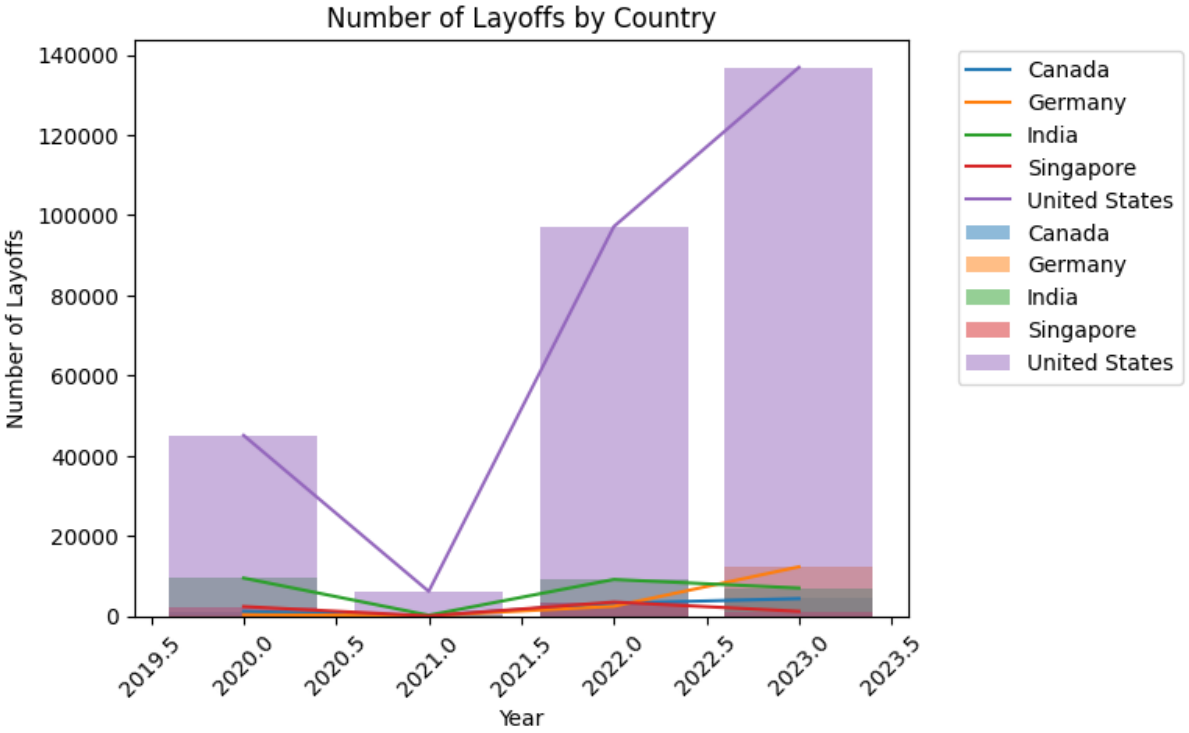
Out [812…

Year	2020	2021	2022	2023
Country				
Canada	1141.0	45.0	3185.0	4341.0
Germany	331.0	87.0	2424.0	12278.0
India	9472.0	200.0	9121.0	6984.0
Singapore	2361.0	21.0	3484.0	1169.0
United States	45082.0	6150.0	97176.0	136900.0

In [813…

```
# Convert the data frame to a dictionary with Year and Country as keys
layoff_dict = layoff_final_df.to_dict(orient="index")

# Convert the dictionary values to lists
for key, value in layoff_dict.items():
    value["Year"] = list(value.keys())
    value["Layoffs"] = list(value.values())
    del value["Year"]
    del value["Layoffs"]
layoff_dict

# Pivot the data frame
layoff_final_df = layoff_counts_df.pivot(
    index="Country", columns="Year", values="Total Layoffs").dropna()

# Convert the data frame to a dictionary with Year and Country as keys
layoff_dict = layoff_final_df.to_dict(orient="index")

# Convert the dictionary values to lists
layoff_final_df = {"Year": list(layoff_dict.values())[0].keys()}
for key, value in layoff_dict.items():
    layoff_final_df[key] = list(value.values())
layoff_final_df = pd.DataFrame(layoff_final_df)

# Set the index to 'Year'
layoff_final_df.set_index("Year", inplace=True)

# Create a figure and axis object
fig, ax = plt.subplots()

# Plot the histograms
for country in layoff_final_df.columns:
    ax.bar(layoff_final_df.index,
           layoff_final_df[country], label=country, alpha=0.5)

# Plot the line graphs
for country in layoff_final_df.columns:
    ax.plot(layoff_final_df.index, layoff_final_df[country], label=cou
ax.set_xlabel("Year")
```

```
ax.set_ylabel("Number of Layoffs")
ax.set_title("Number of Layoffs by Country")
ax.legend(bbox_to_anchor=(1.05, 1), loc="upper left")
plt.xticks(rotation=45)
plt.show()
```



Interpretation:

By combining both bar plots and line graphs, we can discern significant patterns and variations in the number of layoffs. The bar plots provide a comparative analysis, showcasing the disparities in layoffs between different countries for each year; for example, United States ranking extremely highly. On the other hand, the line graphs depict the temporal evolution of layoffs, allowing us to observe trends and fluctuations in each country's labor market. Interestingly, while some countries exhibit relatively stable layoff rates over the years, others display notable fluctuations, suggesting potential economic or political factors at play. Overall, this visualization serves as a comprehensive tool for policymakers and analysts to identify regions of stability, emerging challenges, and opportunities for intervention in the global labor market landscape.

World Income Groups Dataset

In [814…

```
# Read in the data
income_df = pd.read_csv('income_groups.csv')
income_df
```

Out[814…

	country_code	region	income_group
0	ABW	Latin America & Caribbean	High income
1	AFG	South Asia	Low income
2	AGO	Sub-Saharan Africa	Lower middle income
3	ALB	Europe & Central Asia	Upper middle income
4	AND	Europe & Central Asia	High income
...	...	...	...
212	XKX	Europe & Central Asia	Upper middle income
213	YEM	Middle East & North Africa	Low income
214	ZAF	Sub-Saharan Africa	Upper middle income
215	ZMB	Sub-Saharan Africa	Lower middle income
216	ZWE	Sub-Saharan Africa	Lower middle income

217 rows × 3 columns

In [ ]:
```
# Pip install the pycountry library
!pip install pycountry
```

In [816…
```
def country_code_to_name(country_code):
    '''Converts a country code to a country name'''
    # Use the pycountry library to convert the country code to a count
    try:
        country = pycountry.countries.get(
            alpha_3=country_code)  # Get the country name
        return country.name
    except AttributeError:
        return "Unknown"
```

In [817…
```
# Apply the function to the 'country_code' column
income_df['Country'] = income_df['country_code'].apply(country_code_to
income_df
```

Out [817…

	country_code	region	income_group	Country
0	ABW	Latin America & Caribbean	High income	Aruba
1	AFG	South Asia	Low income	Afghanistan
2	AGO	Sub-Saharan Africa	Lower middle income	Angola
3	ALB	Europe & Central Asia	Upper middle income	Albania
4	AND	Europe & Central Asia	High income	Andorra
...	...	...	...	...
212	XKX	Europe & Central Asia	Upper middle income	Unknown
213	YEM	Middle East & North Africa	Low income	Yemen
214	ZAF	Sub-Saharan Africa	Upper middle income	South Africa
215	ZMB	Sub-Saharan Africa	Lower middle income	Zambia
216	ZWE	Sub-Saharan Africa	Lower middle income	Zimbabwe

217 rows × 4 columns

In [818…

```
# Drop missing values
income_df.dropna(inplace=True)
income_df
```

Out [818…

	country_code	region	income_group	Country
0	ABW	Latin America & Caribbean	High income	Aruba
1	AFG	South Asia	Low income	Afghanistan
2	AGO	Sub-Saharan Africa	Lower middle income	Angola
3	ALB	Europe & Central Asia	Upper middle income	Albania
4	AND	Europe & Central Asia	High income	Andorra
...	...	...	...	...
212	XKX	Europe & Central Asia	Upper middle income	Unknown
213	YEM	Middle East & North Africa	Low income	Yemen
214	ZAF	Sub-Saharan Africa	Upper middle income	South Africa
215	ZMB	Sub-Saharan Africa	Lower middle income	Zambia
216	ZWE	Sub-Saharan Africa	Lower middle income	Zimbabwe

217 rows × 4 columns

In [819…

```
# Find the "Unknown" values to replace with the correct country names
unknown_vales = income_df[income_df['Country'] == 'Unknown']
unknown_vales
```

Out [819…

	country_code	region	income_group	Country
34	CHI	Europe & Central Asia	High income	Unknown
212	XKX	Europe & Central Asia	Upper middle income	Unknown

In [820…

```
# Replace "Unknown" values in the "Country" column based on different
income_df.loc[(income_df['Country'] == 'Unknown') & (
    income_df['income_group'] == 'Upper middle income'), 'Country'] =
income_df.loc[(income_df['Country'] == 'Unknown') & (
    income_df['income_group'] == 'High income'), 'Country'] = 'Czech R
income_df
```

Out [820…

	country_code	region	income_group	Country
0	ABW	Latin America & Caribbean	High income	Aruba
1	AFG	South Asia	Low income	Afghanistan
2	AGO	Sub-Saharan Africa	Lower middle income	Angola
3	ALB	Europe & Central Asia	Upper middle income	Albania
4	AND	Europe & Central Asia	High income	Andorra
...	...	...	...	...
212	XKX	Europe & Central Asia	Upper middle income	Kosovo
213	YEM	Middle East & North Africa	Low income	Yemen
214	ZAF	Sub-Saharan Africa	Upper middle income	South Africa
215	ZMB	Sub-Saharan Africa	Lower middle income	Zambia
216	ZWE	Sub-Saharan Africa	Lower middle income	Zimbabwe

217 rows × 4 columns

In [821…

```
# Drop the "country_code" and "region" columns
income_df.drop(columns=['country_code', 'region'], inplace=True)
income_df
```

Out [821…

	income_group	Country
0	High income	Aruba
1	Low income	Afghanistan
2	Lower middle income	Angola
3	Upper middle income	Albania
4	High income	Andorra
...	...	...
212	Upper middle income	Kosovo
213	Low income	Yemen
214	Upper middle income	South Africa
215	Lower middle income	Zambia
216	Lower middle income	Zimbabwe

217 rows × 2 columns

In [822…

```
# Make a copy of the data frame
income_final_df = income_df.copy()

# Rename the columns to be more descriptive and consistent
income_final_df.rename(
    columns={'Country': 'Country', 'income_group': 'Income Group'}, in

# Reindex the columns
income_final_df = income_final_df.reindex(columns=['Country', 'Income
income_final_df
```

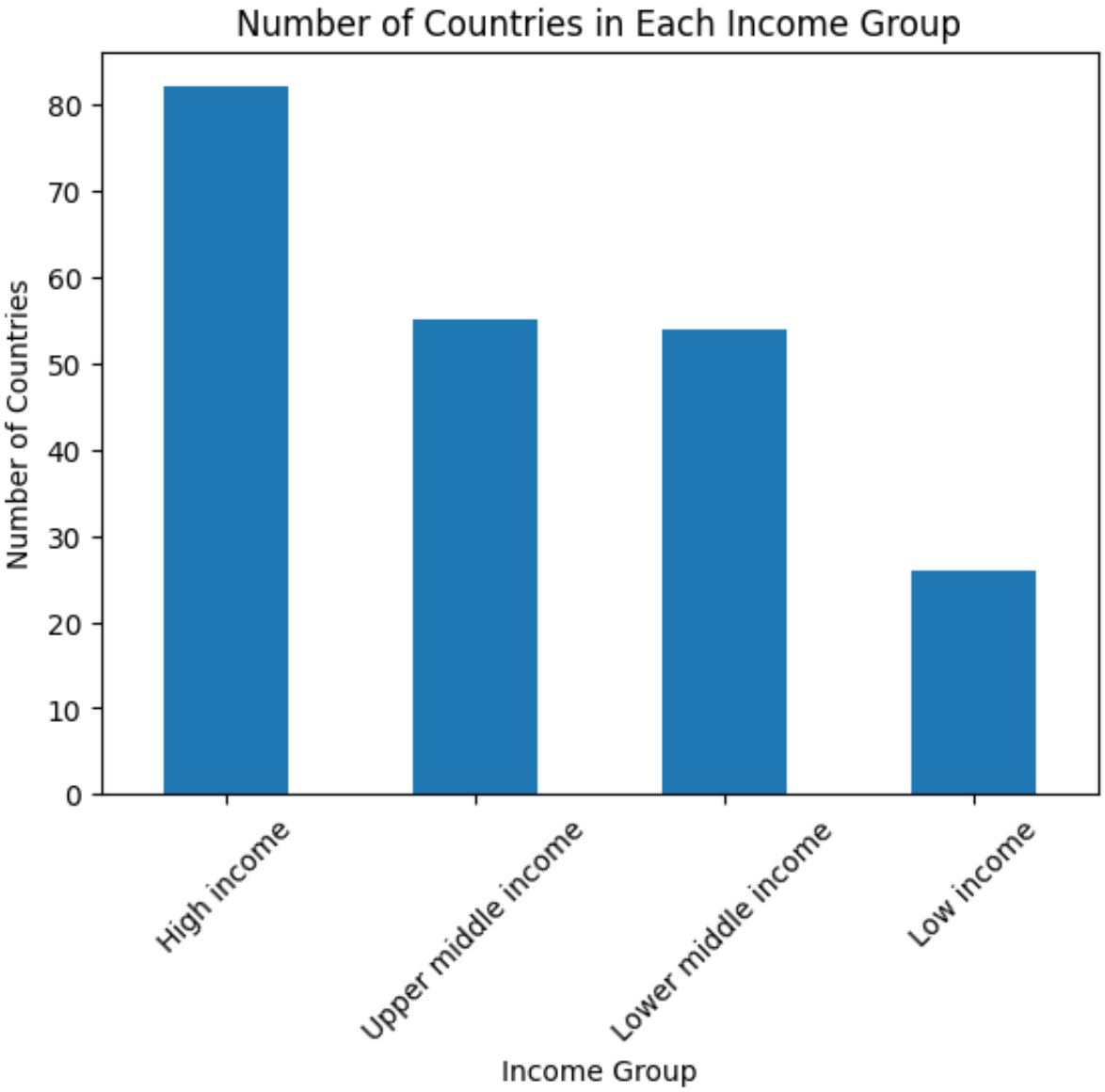
Out [822…

	Country	Income Group
0	Aruba	High income
1	Afghanistan	Low income
2	Angola	Lower middle income
3	Albania	Upper middle income
4	Andorra	High income
...	...	...
212	Kosovo	Upper middle income
213	Yemen	Low income
214	South Africa	Upper middle income
215	Zambia	Lower middle income
216	Zimbabwe	Lower middle income

217 rows × 2 columns

In [823…

```
# Visualize the income group data in a digestible format
income_final_df['Income Group'].value_counts().plot(kind='bar')
plt.title('Number of Countries in Each Income Group')
plt.xlabel('Income Group')
plt.ylabel('Number of Countries')
plt.xticks(rotation=45)
plt.show()
```



Interpretation:

Presented in a bar chart format, the graph effectively conveys the relative frequency of countries within each income category. Notably, the tallest bar corresponds to the "High Income" group, indicating that a significant number of countries fall into this category. Following closely behind are the "Upper Middle Income" and "Lower Middle Income" groups, demonstrating a substantial presence within these income brackets. Finally, the "Low Income" group represents the smallest proportion of countries, as evidenced by the shortest bar on the chart. This visualization provides a quick and accessible means of understanding the distribution of countries across income groups, highlighting the predominance of high-income nations while also acknowledging the diversity present across various economic strata.

Global GDP Growth Rate Dataset

In [824…
```
# Read in the data
gdpgrowth_df = pd.read_csv('gdp_growth_rate.csv')
gdpgrowth_df
```

Out[824…

	Real GDP growth (Annual percent change)	1980	1981	1982	1983	1984	1985	1986	1987	1988	...
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
1	Afghanistan	no data	no data	no data	no data	no data	no data	no data	no data	no data	...
2	Albania	2.7	5.7	2.9	1.1	2	-1.5	5.6	-0.8	-1.4	...
3	Algeria	-5.4	3	6.4	5.4	5.6	5.6	-0.2	-0.7	-1.9	...
4	Andorra	no data	no data	no data	no data	no data	no data	no data	no data	no data	...
...	...	...	...	...	...	...	...	...	...	...	...
226	Other advanced economies	3.8	4.2	2.1	4	6.6	4.3	5.5	6.5	5.5	...
227	Sub-Saharan Africa	no data	no data	no data	no data	no data	no data	no data	no data	no data	...
228	World	2.2	2.1	0.7	2.6	4.6	3.6	3.6	3.9	4.7	...
229	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
230	©IMF, 2023	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...

231 rows × 50 columns

We could see there are a lot of NaN values, so might have to drop rows that have

In [825…
```
# Find missing values
gdpgrowth_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 231 entries, 0 to 230
Data columns (total 50 columns):
 #   Column                                        Non-Null Count  Dtype 
---  ------                                        --------------  ----- 
 0   Real GDP growth (Annual percent change)  229 non-null    object
 1   1980                                          228 non-null    object
```

```
 2   1981                                    228 non-null    object
 3   1982                                    228 non-null    object
 4   1983                                    228 non-null    object
 5   1984                                    228 non-null    object
 6   1985                                    228 non-null    object
 7   1986                                    228 non-null    object
 8   1987                                    228 non-null    object
 9   1988                                    228 non-null    object
 10  1989                                    228 non-null    object
 11  1990                                    228 non-null    object
 12  1991                                    228 non-null    object
 13  1992                                    228 non-null    object
 14  1993                                    228 non-null    object
 15  1994                                    228 non-null    object
 16  1995                                    228 non-null    object
 17  1996                                    228 non-null    object
 18  1997                                    228 non-null    object
 19  1998                                    228 non-null    object
 20  1999                                    228 non-null    object
 21  2000                                    228 non-null    object
 22  2001                                    228 non-null    object
 23  2002                                    228 non-null    object
 24  2003                                    228 non-null    object
 25  2004                                    228 non-null    object
 26  2005                                    228 non-null    object
 27  2006                                    228 non-null    object
 28  2007                                    228 non-null    object
 29  2008                                    228 non-null    object
 30  2009                                    228 non-null    object
 31  2010                                    228 non-null    object
 32  2011                                    228 non-null    object
 33  2012                                    228 non-null    object
 34  2013                                    228 non-null    object
 35  2014                                    228 non-null    object
 36  2015                                    228 non-null    object
 37  2016                                    228 non-null    object
 38  2017                                    228 non-null    object
 39  2018                                    228 non-null    object
 40  2019                                    228 non-null    object
 41  2020                                    228 non-null    object
 42  2021                                    228 non-null    object
 43  2022                                    228 non-null    object
 44  2023                                    228 non-null    object
 45  2024                                    228 non-null    object
 46  2025                                    228 non-null    object
 47  2026                                    228 non-null    object
 48  2027                                    228 non-null    object
 49  2028                                    228 non-null    object
dtypes: object(50)
memory usage: 90.4+ KB
```

In [826…

```
# Select the columns we want to keep
gdpgrowth_recent = gdpgrowth_df[[
```

```
    "Real GDP growth (Annual percent change)", "2020", "2021", "2022",

# Drop missing values
gdpgrowth_recent = gdpgrowth_recent.dropna()
gdpgrowth_recent
```

Out [826…

	Real GDP growth (Annual percent change)	2020	2021	2022	2023
1	Afghanistan	-2.4	-20.7	no data	no data
2	Albania	-3.3	8.9	4.8	3.6
3	Algeria	-5.1	3.4	3.2	3.8
4	Andorra	-11.2	8.3	8.8	2.1
5	Angola	-5.6	1.2	3	1.3
...	...	...	...	...	...
224	Major advanced economies (G7)	-4.5	5.4	2.3	1.5
225	Middle East and Central Asia	-2.6	4.3	5.6	2
226	Other advanced economies	-1.6	5.7	2.6	1.8
227	Sub-Saharan Africa	-1.6	4.7	4	3.3
228	World	-2.8	6.3	3.5	3

228 rows × 5 columns

In [827…

```
# Replace "no data" with pd.NA
gdpgrowth_recent.replace("no data", pd.NA, inplace=True)
gdpgrowth_recent
```

Out [827…

	Real GDP growth (Annual percent change)	2020	2021	2022	2023
1	Afghanistan	-2.4	-20.7	<NA>	<NA>
2	Albania	-3.3	8.9	4.8	3.6
3	Algeria	-5.1	3.4	3.2	3.8
4	Andorra	-11.2	8.3	8.8	2.1
5	Angola	-5.6	1.2	3	1.3
...	...	...	...	...	...
224	Major advanced economies (G7)	-4.5	5.4	2.3	1.5
225	Middle East and Central Asia	-2.6	4.3	5.6	2
226	Other advanced economies	-1.6	5.7	2.6	1.8
227	Sub-Saharan Africa	-1.6	4.7	4	3.3
228	World	-2.8	6.3	3.5	3

228 rows × 5 columns

In [828…

```
# Make a copy of the data frame
gdpgrowth_country_final_df = gdpgrowth_recent.copy()

# Rename the columns to be more descriptive and consistent
gdpgrowth_country_final_df.rename(columns={"Real GDP growth (Annual pe
                                           "2022": "Annual GDP Growth(

# Reset the index
gdpgrowth_country_final_df.reset_index(drop=True, inplace=True)

# Select the rows from 1 to 195
gdpgrowth_country_final_df = gdpgrowth_country_final_df.iloc[1:195].co
gdpgrowth_country_final_df
```

Out [828…

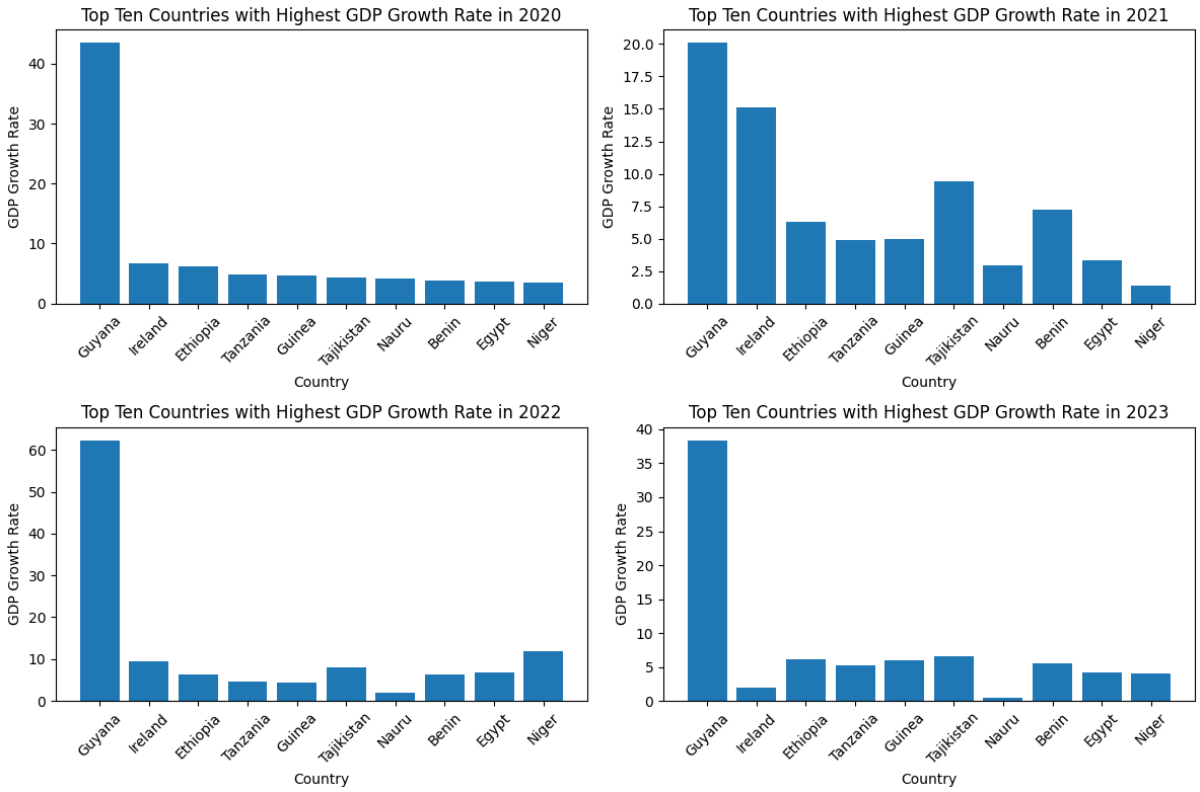
	Country	Annual GDP Growth(2020)	Annual GDP Growth(2021)	Annual GDP Growth(2022)	Annual GDP Growth(2023)
1	Albania	-3.3	8.9	4.8	3.6
2	Algeria	-5.1	3.4	3.2	3.8
3	Andorra	-11.2	8.3	8.8	2.1
4	Angola	-5.6	1.2	3	1.3
5	Antigua and Barbuda	-17.5	6.6	8.5	5.6
...	...	...	...	...	...
190	Venezuela	-30	1	8	4
191	Vietnam	2.9	2.6	8	4.7
192	West Bank and Gaza	-11.3	7	3.9	3
193	Yemen	-8.5	-1	1.5	-0.5
194	Zambia	-2.8	4.6	4.7	3.6

194 rows × 5 columns

In [829…

```
# Convert the columns to numeric data types
gdpgrowth_country_final_df['Annual GDP Growth(2020)'] = pd.to_numeric(
    gdpgrowth_country_final_df['Annual GDP Growth(2020)'])
gdpgrowth_country_final_df['Annual GDP Growth(2021)'] = pd.to_numeric(
    gdpgrowth_country_final_df['Annual GDP Growth(2021)'])
gdpgrowth_country_final_df['Annual GDP Growth(2022)'] = pd.to_numeric(
    gdpgrowth_country_final_df['Annual GDP Growth(2022)'])
gdpgrowth_country_final_df['Annual GDP Growth(2023)'] = pd.to_numeric(
    gdpgrowth_country_final_df['Annual GDP Growth(2023)'])

# Sort the dataframe by each year's GDP growth rate in descending orde
gdpgrowth_sorted = gdpgrowth_country_final_df.sort_values(
    by=['Annual GDP Growth(2020)', 'Annual GDP Growth(2021)', 'Annual

# Get the top ten countries with the highest GDP growth rate for each
top_ten_2020 = gdpgrowth_sorted.head(10)['Country']
top_ten_2021 = gdpgrowth_sorted.head(10)['Country']
top_ten_2022 = gdpgrowth_sorted.head(10)['Country']
top_ten_2023 = gdpgrowth_sorted.head(10)['Country']

# Visualize the top ten countries with the highest GDP growth rate for
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 8))
```

```
# Plot for 2020
axes[0, 0].bar(top_ten_2020, gdpgrowth_sorted.head(10)
               ['Annual GDP Growth(2020)'])
axes[0, 0].set_xlabel('Country')
axes[0, 0].set_ylabel('GDP Growth Rate')
axes[0, 0].set_title('Top Ten Countries with Highest GDP Growth Rate i
axes[0, 0].tick_params(axis='x', rotation=45)

# Plot for 2021
axes[0, 1].bar(top_ten_2021, gdpgrowth_sorted.head(10)
               ['Annual GDP Growth(2021)'])
axes[0, 1].set_xlabel('Country')
axes[0, 1].set_ylabel('GDP Growth Rate')
axes[0, 1].set_title('Top Ten Countries with Highest GDP Growth Rate i
axes[0, 1].tick_params(axis='x', rotation=45)

# Plot for 2022
axes[1, 0].bar(top_ten_2022, gdpgrowth_sorted.head(10)
               ['Annual GDP Growth(2022)'])
axes[1, 0].set_xlabel('Country')
axes[1, 0].set_ylabel('GDP Growth Rate')
axes[1, 0].set_title('Top Ten Countries with Highest GDP Growth Rate i
axes[1, 0].tick_params(axis='x', rotation=45)

# Plot for 2023
axes[1, 1].bar(top_ten_2023, gdpgrowth_sorted.head(10)
               ['Annual GDP Growth(2023)'])
axes[1, 1].set_xlabel('Country')
axes[1, 1].set_ylabel('GDP Growth Rate')
axes[1, 1].set_title('Top Ten Countries with Highest GDP Growth Rate i
axes[1, 1].tick_params(axis='x', rotation=45)

# Adjust the layout
plt.tight_layout()
plt.show()
```



Interpretation:

The visualization offers a compelling insight into the economic performance of countries across four consecutive years, from 2020 to 2023. By examining the bar plots representing the top ten countries with the highest GDP growth rates for each year, we can discern a consistent trend of certain nations outperforming others in terms of economic expansion. For example, we can see Guyana ranking consistently highly across four years, leaving Ireland and Ethiopia behind by a large margin. This visualization underscores the importance of understanding the drivers of economic growth and identifying potential opportunities for investment and development. Overall, it provides a clear and concise summary of the economic landscape, facilitating informed decision-making and strategic planning initiatives.

World Happiness Reports

Happiness Dataset - 2020

In [830…

```
# Read in the data
happiness2020_df = pd.read_csv('world_happiness_report_2020.csv')
happiness2020_df
```

Out [830…

	Country name	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	L G
0	Finland	Western Europe	7.8087	0.031156	7.869766	7.747634	10.6
1	Denmark	Western Europe	7.6456	0.033492	7.711245	7.579955	10.
2	Switzerland	Western Europe	7.5599	0.035014	7.628528	7.491272	10.9
3	Iceland	Western Europe	7.5045	0.059616	7.621347	7.387653	10.7
4	Norway	Western Europe	7.4880	0.034837	7.556281	7.419719	11.0
...	...	...	...	...	...	...	
148	Central African Republic	Sub-Saharan Africa	3.4759	0.115183	3.701658	3.250141	6.6
149	Rwanda	Sub-Saharan Africa	3.3123	0.052425	3.415053	3.209547	7.6
150	Zimbabwe	Sub-Saharan Africa	3.2992	0.058674	3.414202	3.184198	7.8
151	South Sudan	Sub-Saharan Africa	2.8166	0.107610	3.027516	2.605684	7.4
152	Afghanistan	South Asia	2.5669	0.031311	2.628270	2.505530	7.4

153 rows × 20 columns

In [831…

```
# Select the columns we want to keep
happiness2020_refined_df = happiness2020_df[["Country name", "Regional
                                             "Healthy life expectancy"

happiness2020_refined_df
```

Out [831…

	Country name	Regional indicator	Ladder score	Logged GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
0	Finland	Western Europe	7.8087	10.639267	0.954330	71.900825	0.949172
1	Denmark	Western Europe	7.6456	10.774001	0.955991	72.402504	0.951444
2	Switzerland	Western Europe	7.5599	10.979933	0.942847	74.102448	0.921337
3	Iceland	Western Europe	7.5045	10.772559	0.974670	73.000000	0.948892
4	Norway	Western Europe	7.4880	11.087804	0.952487	73.200783	0.955750
...	...	...	...	...	...	...	...
148	Central African Republic	Sub-Saharan Africa	3.4759	6.625160	0.319460	45.200001	0.640881
149	Rwanda	Sub-Saharan Africa	3.3123	7.600104	0.540835	61.098846	0.900589
150	Zimbabwe	Sub-Saharan Africa	3.2992	7.865712	0.763093	55.617260	0.711458
151	South Sudan	Sub-Saharan Africa	2.8166	7.425360	0.553707	51.000000	0.451314
152	Afghanistan	South Asia	2.5669	7.462861	0.470367	52.590000	0.396573

153 rows × 8 columns

In [832…

```
# Make a copy of the data frame
happiness2020_final_df = happiness2020_refined_df.copy()

# Drop missing values
happiness2020_final_df.dropna(inplace=True)
```

In [833…

```
# Rename the columns to be more descriptive and consistent
happiness2020_final_df.rename(columns={"Country name": "Country", "Reg
                                       "Healthy life expectancy": "Lif

happiness2020_final_df
```

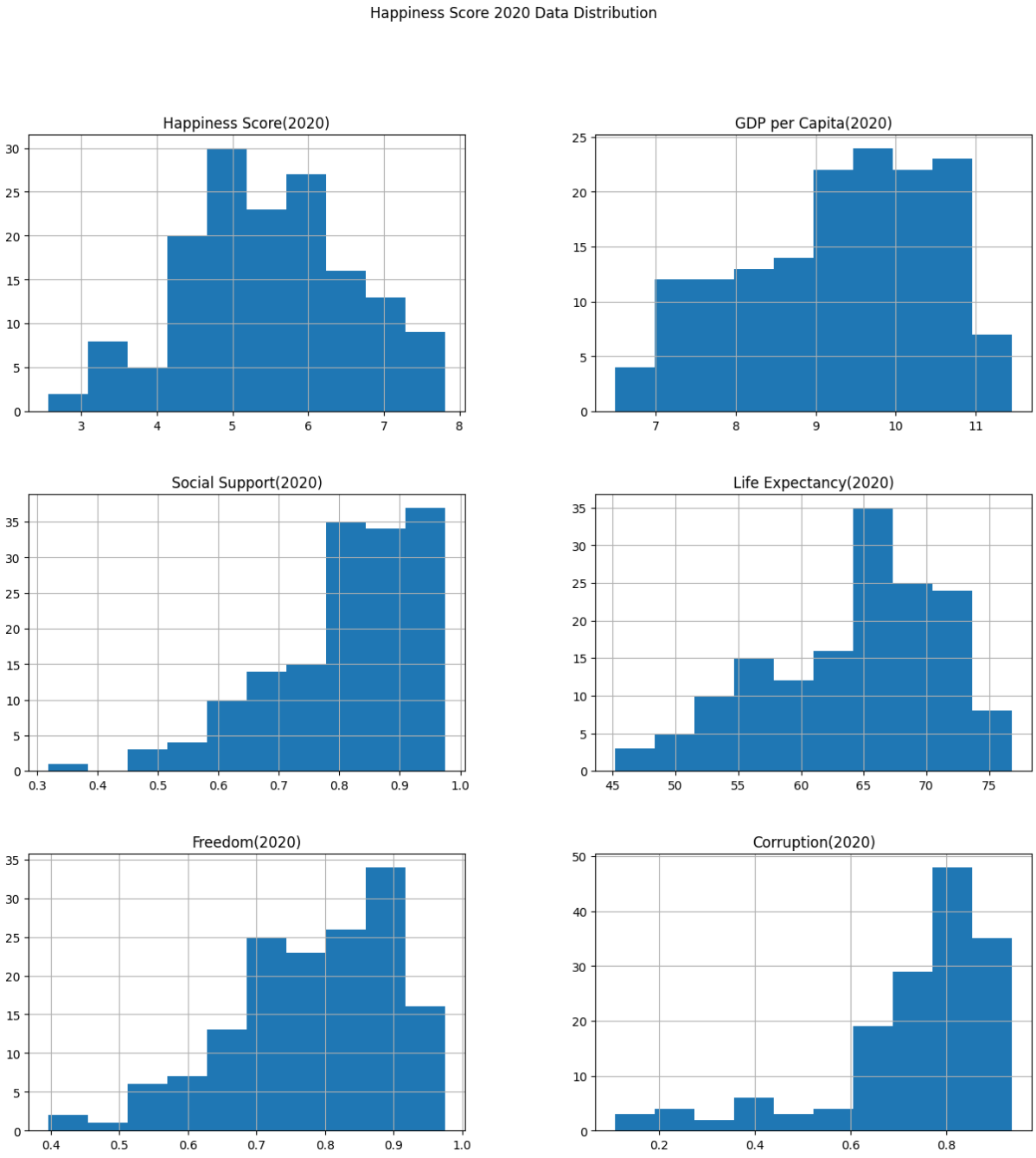
Out [833…

	Country	Region	Happiness Score(2020)	GDP per Capita(2020)	Social Support(2020)	Expectar
0	Finland	Western Europe	7.8087	10.639267	0.954330	
1	Denmark	Western Europe	7.6456	10.774001	0.955991	
2	Switzerland	Western Europe	7.5599	10.979933	0.942847	
3	Iceland	Western Europe	7.5045	10.772559	0.974670	
4	Norway	Western Europe	7.4880	11.087804	0.952487	
...	...	...	...	...	...	
148	Central African Republic	Sub-Saharan Africa	3.4759	6.625160	0.319460	
149	Rwanda	Sub-Saharan Africa	3.3123	7.600104	0.540835	
150	Zimbabwe	Sub-Saharan Africa	3.2992	7.865712	0.763093	
151	South Sudan	Sub-Saharan Africa	2.8166	7.425360	0.553707	
152	Afghanistan	South Asia	2.5669	7.462861	0.470367	

153 rows × 8 columns

In [834…

```
# Visualize the happiness score 2020 data against all the other column
happiness2020_final_df.hist(figsize=(15, 15))
plt.suptitle("Happiness Score 2020 Data Distribution")
plt.show()
```



Interpretation:

The histogram visualization above provides a succinct overview of the distribution of data for the Happiness Score in the year 2020 across various other columns in the dataset. Each subplot represents a different column, displaying the frequency distribution of values within that column. Notably, a common trend emerges where all variables tend to be skewed towards the right side of the histogram, indicating a concentration of higher values. This concise visualization aids in quickly understanding the spread and central tendencies of the dataset, facilitating the identification of any notable patterns or outliers. Such insights are essential for informing subsequent analysis and decision-making processes, making this visualization a valuable exploratory tool in data analysis workflows.

Happiness Dataset - 2021

In [835…
```
# Read in the data
happiness2021_df = pd.read_csv('world_happiness_report_2021.csv')
happiness2021_df
```

Out[835…

	Country name	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Log ca
0	Finland	Western Europe	7.842	0.032	7.904	7.780	10
1	Denmark	Western Europe	7.620	0.035	7.687	7.552	10
2	Switzerland	Western Europe	7.571	0.036	7.643	7.500	1
3	Iceland	Western Europe	7.554	0.059	7.670	7.438	10
4	Netherlands	Western Europe	7.464	0.027	7.518	7.410	10
...	...	...	...	...	...	...	
144	Lesotho	Sub-Saharan Africa	3.512	0.120	3.748	3.276	7
145	Botswana	Sub-Saharan Africa	3.467	0.074	3.611	3.322	9
146	Rwanda	Sub-Saharan Africa	3.415	0.068	3.548	3.282	7
147	Zimbabwe	Sub-Saharan Africa	3.145	0.058	3.259	3.030	7
148	Afghanistan	South Asia	2.523	0.038	2.596	2.449	7

149 rows × 20 columns

In [836…
```
# Select the columns we want to keep
happiness2021_refined_df = happiness2021_df[["Country name", "Regional
```

```
                                                    "Healthy life expectancy"
happiness2021_refined_df
```

Out[836…

	Country name	Regional indicator	Ladder score	Logged GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	P
0	Finland	Western Europe	7.842	10.775	0.954	72.000	0.949	
1	Denmark	Western Europe	7.620	10.933	0.954	72.700	0.946	
2	Switzerland	Western Europe	7.571	11.117	0.942	74.400	0.919	
3	Iceland	Western Europe	7.554	10.878	0.983	73.000	0.955	
4	Netherlands	Western Europe	7.464	10.932	0.942	72.400	0.913	
…	…	…	…	…	…	…	…	…
144	Lesotho	Sub-Saharan Africa	3.512	7.926	0.787	48.700	0.715	
145	Botswana	Sub-Saharan Africa	3.467	9.782	0.784	59.269	0.824	
146	Rwanda	Sub-Saharan Africa	3.415	7.676	0.552	61.400	0.897	
147	Zimbabwe	Sub-Saharan Africa	3.145	7.943	0.750	56.201	0.677	
148	Afghanistan	South Asia	2.523	7.695	0.463	52.493	0.382	

149 rows × 8 columns

In [837…

```
# Make a copy of the data frame
happiness2021_final_df = happiness2021_refined_df.copy()

# Drop missing values
happiness2021_final_df.dropna(inplace=True)
```

In [838…

```
# Rename the columns to be more descriptive and consistent
happiness2021_final_df.rename(columns={"Country name": "Country", "Reg
                                       "Healthy life expectancy": "Lif
```

```
happiness2021_final_df
```

Out [838…

	Country	Region	Happiness Score(2021)	GDP per Capita(2021)	Social Support(2021)	Expectan
0	Finland	Western Europe	7.842	10.775	0.954	
1	Denmark	Western Europe	7.620	10.933	0.954	
2	Switzerland	Western Europe	7.571	11.117	0.942	
3	Iceland	Western Europe	7.554	10.878	0.983	
4	Netherlands	Western Europe	7.464	10.932	0.942	
...	...	...	...	...	...	
144	Lesotho	Sub-Saharan Africa	3.512	7.926	0.787	
145	Botswana	Sub-Saharan Africa	3.467	9.782	0.784	
146	Rwanda	Sub-Saharan Africa	3.415	7.676	0.552	
147	Zimbabwe	Sub-Saharan Africa	3.145	7.943	0.750	
148	Afghanistan	South Asia	2.523	7.695	0.463	

149 rows × 8 columns

In [839…

```
# Create a new data frame with the 'Country' and 'Region' columns
region_df = happiness2021_final_df[['Country', 'Region']].copy()
region_df
```

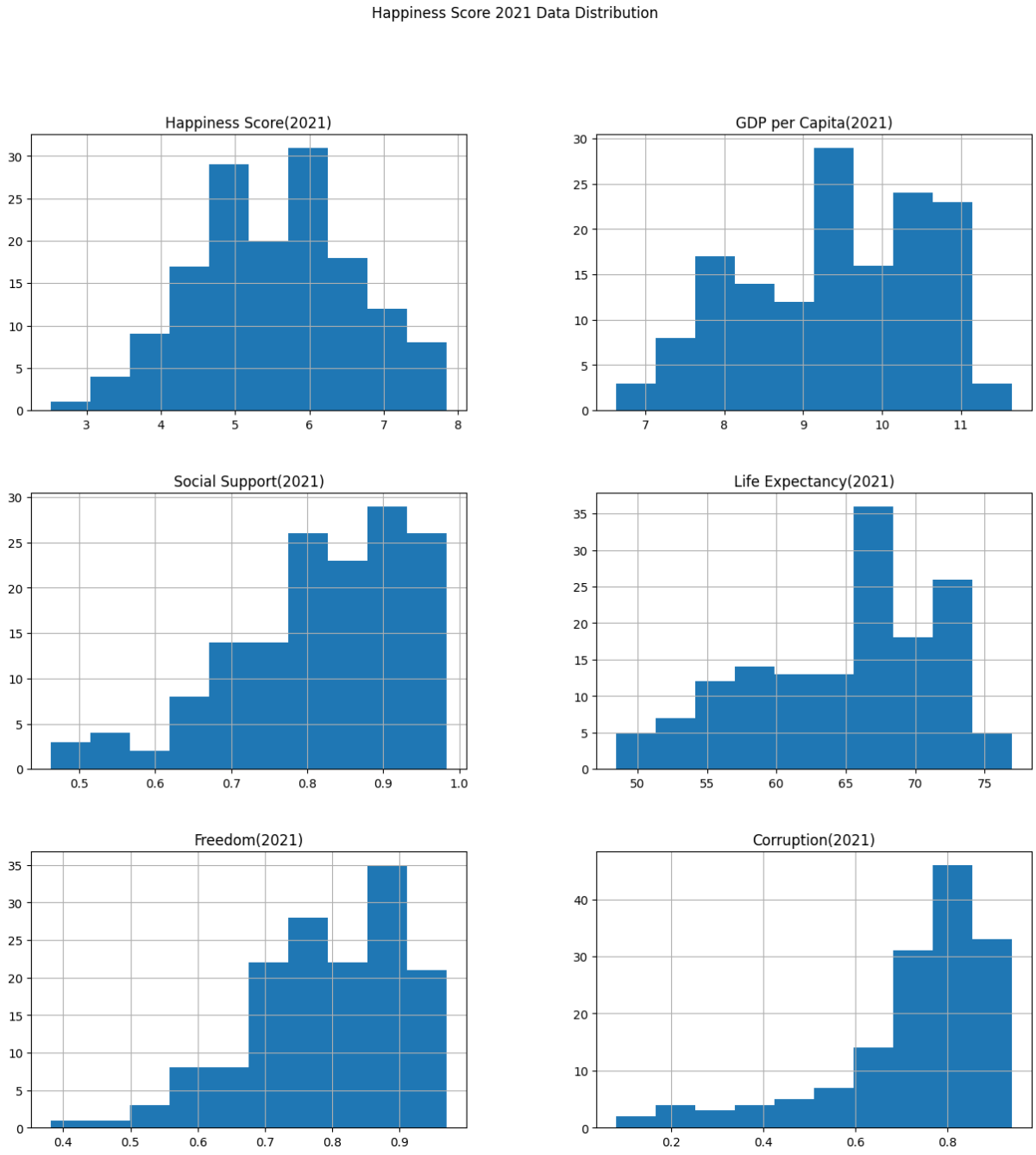
Out [839…

	Country	Region
0	Finland	Western Europe
1	Denmark	Western Europe
2	Switzerland	Western Europe
3	Iceland	Western Europe
4	Netherlands	Western Europe
...	...	...
144	Lesotho	Sub-Saharan Africa
145	Botswana	Sub-Saharan Africa
146	Rwanda	Sub-Saharan Africa
147	Zimbabwe	Sub-Saharan Africa
148	Afghanistan	South Asia

149 rows × 2 columns

In [840…

```
# Visualize the happiness score 2021 data against all the other column
happiness2021_final_df.hist(figsize=(15, 15))
plt.suptitle("Happiness Score 2021 Data Distribution")
plt.show()
```



Interpretation:

The histogram visualization above provides a succinct overview of the distribution of data for the Happiness Score in the year 2021 across various other columns in the dataset. Each subplot represents a different column, displaying the frequency distribution of values within that column. Notably, a common trend emerges where all variables tend to be skewed towards the right side of the histogram, indicating a concentration of higher values. This concise visualization aids in quickly understanding the spread and central tendencies of the dataset, facilitating the identification of any notable patterns or outliers. Such insights are essential for informing subsequent analysis and decision-making processes, making this visualization a valuable exploratory tool in data analysis workflows.

Happiness Dataset - 2022

In [841…
```
# Read in the data
happiness2022_df = pd.read_csv('world_happiness_report_2022.csv')
happiness2022_df
```

Out[841…

	Unnamed: 0	Happiness Rank	Country	Region	Happiness Score	Economy (GDP per Capita)	Fam (Soc Suppo
0	0	1	Finland	Western Europe	7,821	1,892	1,2
1	1	2	Denmark	Western Europe	7,636	1,953	1,2
2	2	3	Iceland	Western Europe	7,557	1,936	1,3
3	3	4	Switzerland	Western Europe	7,512	2,026	1,2
4	4	5	Netherlands	Western Europe	7,415	1,945	1,2
...	...	...	...	...	...	...	
140	141	142	Botswana	Sub-Saharan Africa	3,471	1,503	0,8
141	142	143	Rwanda	Sub-Saharan Africa	3,268	0,785	0,1
142	143	144	Zimbabwe	Sub-Saharan Africa	2,995	0,947	0,6
143	144	145	Lebanon	Middle East and Northern Africa	2,955	1,392	0,4
144	145	146	Afghanistan	Southern Asia	2,404	0,758	0,0

145 rows × 12 columns

In [842…
```
# Select the columns we want to keep
happiness2022_df = happiness2022_df[["Country", "Region", "Happiness S
                                     "Health (Life Expectancy)", "Free

happiness2022_df
```

Out [842…

	Country	Region	Happiness Score	Economy (GDP per Capita)	Family (Social Support)	Health (Life Expectancy)	Freed
0	Finland	Western Europe	7,821	1,892	1,258	0,775	0,
1	Denmark	Western Europe	7,636	1,953	1,243	0,777	0,
2	Iceland	Western Europe	7,557	1,936	1,320	0,803	0,
3	Switzerland	Western Europe	7,512	2,026	1,226	0,822	0,
4	Netherlands	Western Europe	7,415	1,945	1,206	0,787	0,
...	...	...	...	...	...	...	
140	Botswana	Sub-Saharan Africa	3,471	1,503	0,815	0,280	0,
141	Rwanda	Sub-Saharan Africa	3,268	0,785	0,133	0,462	0,
142	Zimbabwe	Sub-Saharan Africa	2,995	0,947	0,690	0,270	0,
143	Lebanon	Middle East and Northern Africa	2,955	1,392	0,498	0,631	0,
144	Afghanistan	Southern Asia	2,404	0,758	0,000	0,289	0,

145 rows × 8 columns

In [843…

```
# Make a copy of the data frame
happiness2022_final_df = happiness2022_df.copy()

# Drop missing values
happiness2022_final_df.dropna(inplace=True)
```

In [844…

```
# Rename the columns to be more descriptive and consistent
happiness2022_final_df.rename(columns={"Happiness Score": "Happiness S
                                       "Health (Life Expectancy)": "Life Expect
happiness2022_final_df
```

Out [844...

	Country	Region	Happiness Score(2022)	GDP per Capita(2022)	Social Support(2022)	Expecta
0	Finland	Western Europe	7,821	1,892	1,258	
1	Denmark	Western Europe	7,636	1,953	1,243	
2	Iceland	Western Europe	7,557	1,936	1,320	
3	Switzerland	Western Europe	7,512	2,026	1,226	
4	Netherlands	Western Europe	7,415	1,945	1,206	
...	...	...	...	...	...	
140	Botswana	Sub-Saharan Africa	3,471	1,503	0,815	
141	Rwanda	Sub-Saharan Africa	3,268	0,785	0,133	
142	Zimbabwe	Sub-Saharan Africa	2,995	0,947	0,690	
143	Lebanon	Middle East and Northern Africa	2,955	1,392	0,498	
144	Afghanistan	Southern Asia	2,404	0,758	0,000	

145 rows × 8 columns

In [ ]:
```
# Convert the 'GDP per Capita(2022)' column to an integer
happiness2022_final_df["GDP per Capita(2022)"] = happiness2022_final_d
    ",", "").astype(int)

# Log transform the 'GDP per Capita(2022)' column
happiness2022_final_df["Logged GDP per Capita(2022)"] = np.log2(
    happiness2022_final_df["GDP per Capita(2022)"])

# Drop the 'GDP per Capita(2022)' column
happiness2022_final_df.drop(columns=["GDP per Capita(2022)"], inplace=

# Rename the 'Logged GDP per Capita(2022)' column to 'GDP per Capita(2
happiness2022_final_df.rename(
```

```
    columns={"Logged GDP per Capita(2022)": "GDP per Capita(2022)"}, i
happiness2022_final_df
```

In [846…

```
# Replace the commas with periods in every applicable column
happiness2022_final_df = happiness2022_final_df.replace(",", ".", rege

# Convert the columns to float
happiness2022_final_df["Life Expectancy(2022)"] = happiness2022_final_
    float)

# Multiply the 'Life Expectancy(2022)' column by 100
happiness2022_final_df["Life Expectancy(2022)"] = happiness2022_final_
happiness2022_final_df
```

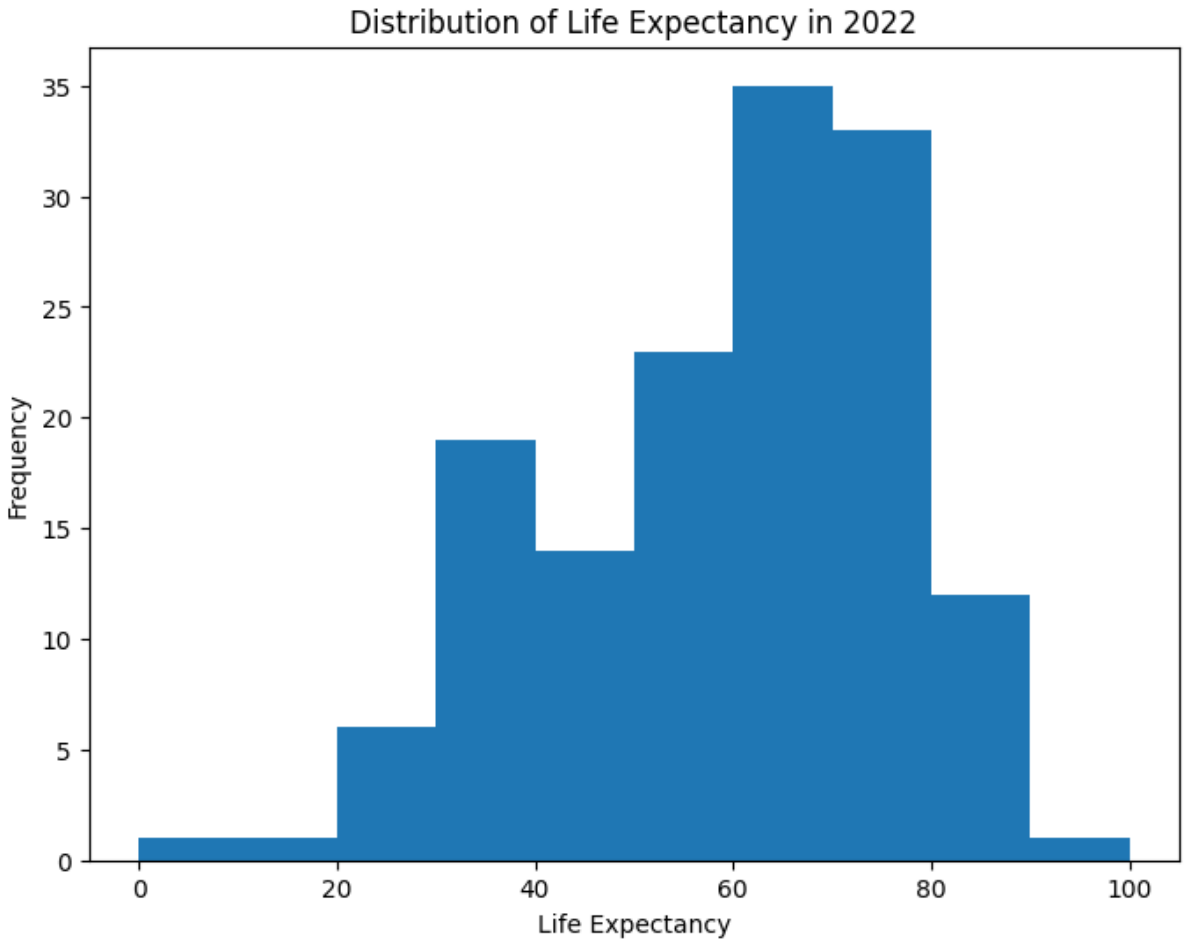
Out [846…

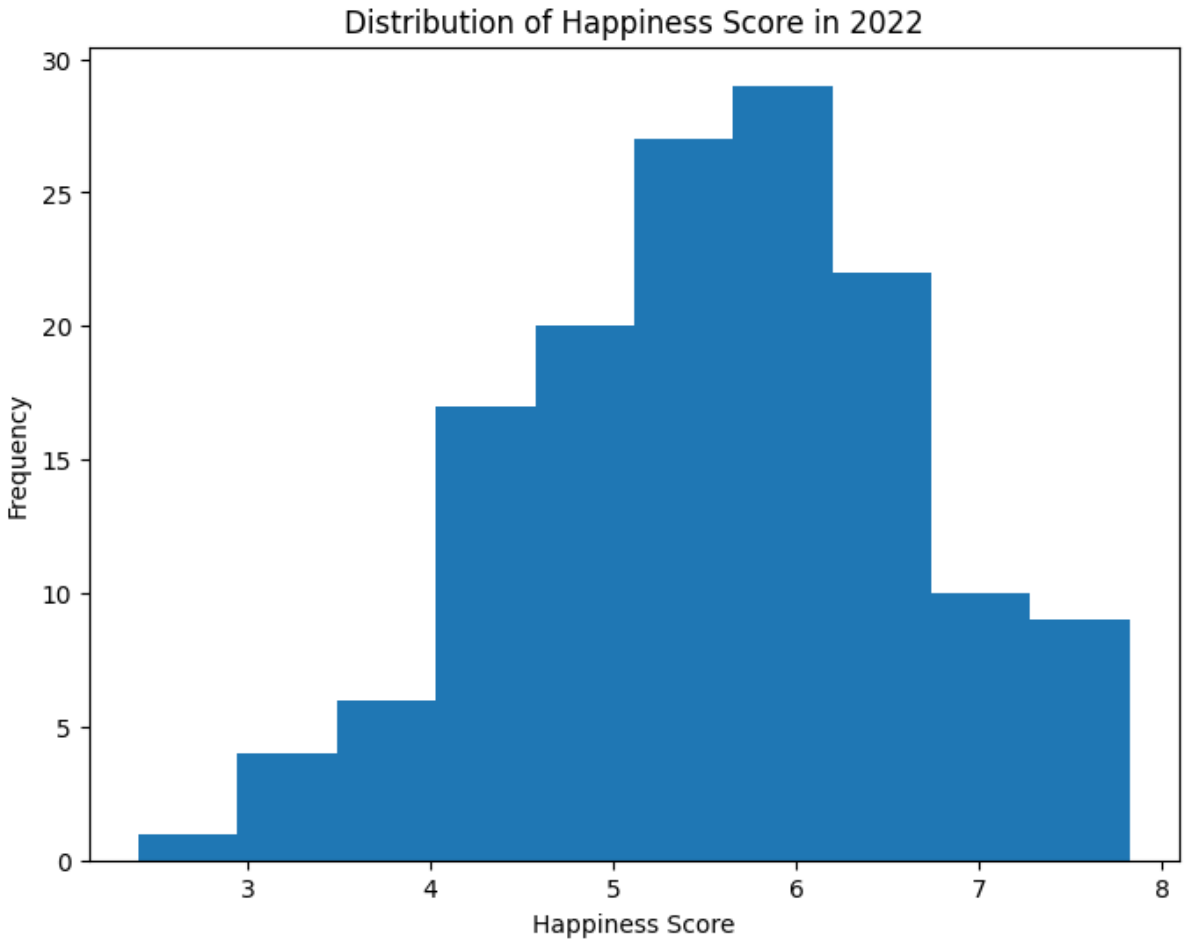
	Country	Region	Happiness Score(2022)	Social Support(2022)	Life Expectancy(2022)	Fr
0	Finland	Western Europe	7.821	1.258	77.5	
1	Denmark	Western Europe	7.636	1.243	77.7	
2	Iceland	Western Europe	7.557	1.320	80.3	
3	Switzerland	Western Europe	7.512	1.226	82.2	
4	Netherlands	Western Europe	7.415	1.206	78.7	
...	...	...	...	...	...	
140	Botswana	Sub-Saharan Africa	3.471	0.815	28.0	
141	Rwanda	Sub-Saharan Africa	3.268	0.133	46.2	
142	Zimbabwe	Sub-Saharan Africa	2.995	0.690	27.0	
143	Lebanon	Middle East and Northern Africa	2.955	0.498	63.1	
144	Afghanistan	Southern Asia	2.404	0.000	28.9	

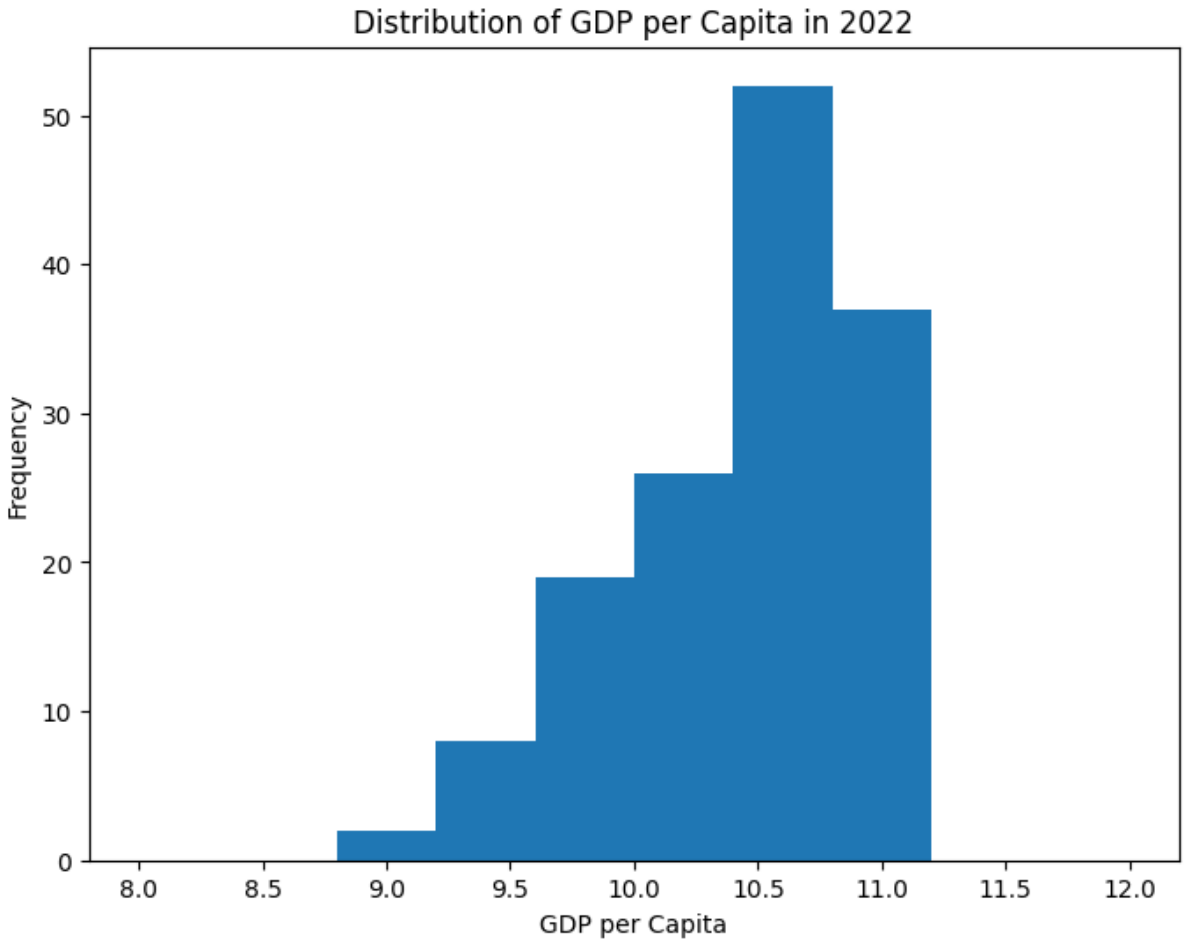
145 rows × 8 columns

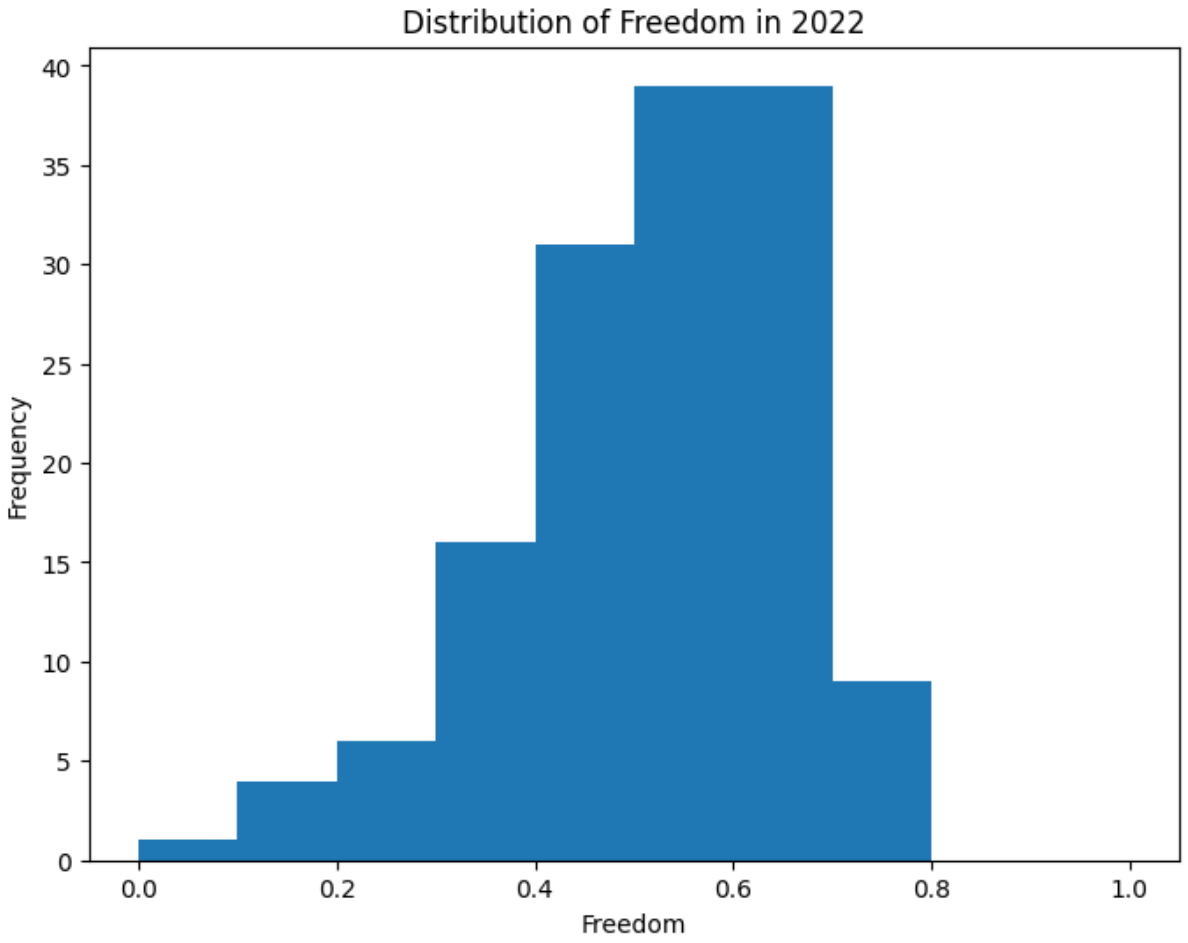
In [847…

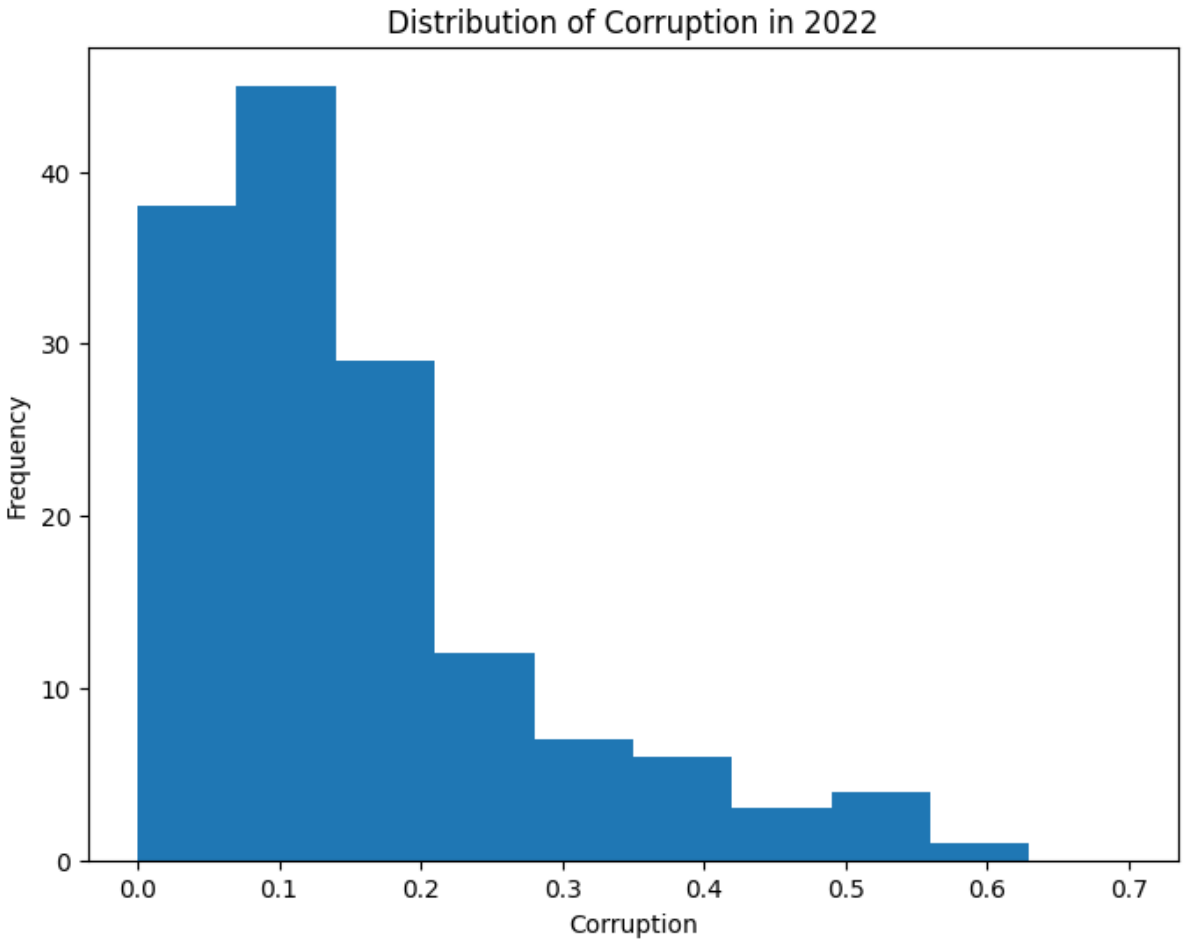
```
# Convert the 'Happiness Score(2022)', 'Freedom(2022)' and 'Corruption
happiness2022_final_df["Happiness Score(2022)"] = happiness2022_final_
    float)
happiness2022_final_df["Freedom(2022)"] = happiness2022_final_df["Free
    float)
happiness2022_final_df["Corruption(2022)"] = happiness2022_final_df["C
    float)

# Visualize the Life Expectancy data
happiness2022_final_df["Life Expectancy(2022)"].plot(
    kind="hist", figsize=(8, 6), range=(0, 100))
plt.xlabel("Life Expectancy")
plt.ylabel("Frequency")
plt.title("Distribution of Life Expectancy in 2022")
plt.show()

# Visualize the Happiness Score
happiness2022_final_df["Happiness Score(2022)"].plot(
    kind="hist", figsize=(8, 6))
plt.xlabel("Happiness Score")
plt.ylabel("Frequency")
plt.title("Distribution of Happiness Score in 2022")
plt.show()

# Visualize the GDP per Capita
happiness2022_final_df["GDP per Capita(2022)"].plot(
    kind="hist", figsize=(8, 6), range=(8, 12))
plt.xlabel("GDP per Capita")
plt.ylabel("Frequency")
plt.title("Distribution of GDP per Capita in 2022")
plt.show()

# Visualize the Freedom data
happiness2022_final_df["Freedom(2022)"].plot(
    kind="hist", figsize=(8, 6), range=(0, 1))
plt.xlabel("Freedom")
plt.ylabel("Frequency")
plt.title("Distribution of Freedom in 2022")
plt.show()

# Visualize the Corruption data
happiness2022_final_df["Corruption(2022)"].plot(
    kind="hist", figsize=(8, 6), range=(0, 0.7))
plt.xlabel("Corruption")
plt.ylabel("Frequency")
plt.title("Distribution of Corruption in 2022")
plt.show()
```


Distribution of Life Expectancy in 2022
Frequency
Life Expectancy
0
5
10
15
20
25
30
35
0
20
40
60
80
100


Distribution of Happiness Score in 2022
Frequency
Happiness Score
0
5
10
15
20
25
30
3
4
5
6
7
8


Distribution of GDP per Capita in 2022
Frequency
GDP per Capita
0
10
20
30
40
50
8.0
8.5
9.0
9.5
10.0
10.5
11.0
11.5
12.0


Distribution of Freedom in 2022
Frequency
Freedom
0
5
10
15
20
25
30
35
40
0.0
0.2
0.4
0.6
0.8
1.0



Interpretation:

The histogram visualization above provides a succinct overview of the distribution of data for the Happiness Score in the year 2022 across various other columns in the dataset. Each subplot represents a different column, displaying the frequency distribution of values within that column. Notably, a common trend emerges where all variables tend to be skewed towards the right side of the histogram, indicating a concentration of higher values, except for "Corruption". This concise visualization aids in quickly understanding the spread and central tendencies of the dataset, facilitating the identification of any notable patterns or outliers. Such insights are essential for informing subsequent analysis and decision-making processes, making this visualization a valuable exploratory tool in data analysis workflows.

Happiness Dataset - 2023

In [848…

```
# Read in the data
happiness2023_df = pd.read_csv('world_happiness_report_2023.csv')
happiness2023_df
```

Out[848…

	Country name	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Logged GDP per capita	So supp
0	Finland	7.804	0.036	7.875	7.733	10.792	0.9
1	Denmark	7.586	0.041	7.667	7.506	10.962	0.9
2	Iceland	7.530	0.049	7.625	7.434	10.896	0.9
3	Israel	7.473	0.032	7.535	7.411	10.639	0.9
4	Netherlands	7.403	0.029	7.460	7.346	10.942	0.9
...	...	...	...	...	...	...	
132	Congo (Kinshasa)	3.207	0.095	3.394	3.020	7.007	0.6
133	Zimbabwe	3.204	0.061	3.323	3.084	7.641	0.6
134	Sierra Leone	3.138	0.082	3.299	2.976	7.394	0.5
135	Lebanon	2.392	0.044	2.479	2.305	9.478	0.5
136	Afghanistan	1.859	0.033	1.923	1.795	7.324	0.

137 rows × 19 columns

In [ ]:
```
# Pip install the pycountry-convert library
!pip install pycountry-convert
```

In [850…
```
# Select the columns we want to keep
happiness2023_refined_df = happiness2023_df[["Country name", "Ladder s
                                             "Healthy life expectancy"

happiness2023_refined_df
```

Out [850…

	Country name	Ladder score	Logged GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Perceptions of corruption
0	Finland	7.804	10.792	0.969	71.150	0.961	0.182
1	Denmark	7.586	10.962	0.954	71.250	0.934	0.196
2	Iceland	7.530	10.896	0.983	72.050	0.936	0.668
3	Israel	7.473	10.639	0.943	72.697	0.809	0.708
4	Netherlands	7.403	10.942	0.930	71.550	0.887	0.379
...	...	...	...	...	...	...	...
132	Congo (Kinshasa)	3.207	7.007	0.652	55.375	0.664	0.834
133	Zimbabwe	3.204	7.641	0.690	54.050	0.654	0.766
134	Sierra Leone	3.138	7.394	0.555	54.900	0.660	0.858
135	Lebanon	2.392	9.478	0.530	66.149	0.474	0.891
136	Afghanistan	1.859	7.324	0.341	54.712	0.382	0.847

137 rows × 7 columns

In [851…

```
# Make a copy of the data frame
happiness2023_final_df = happiness2023_refined_df.copy()

# Drop missing values
happiness2023_final_df.dropna(inplace=True)
```

In [852…

```
# Rename the columns to be more descriptive and consistent
happiness2023_final_df.rename(columns={"Country name": "Country", "Lad
                                       "Social support": "Social Suppo
                                       "Perceptions of corruption": "C

# Merge the 'Region' column from the 'region_df' data frame with the '
happiness2023_final_df = happiness2023_final_df.merge(
    region_df, on='Country', how='left')
happiness2023_final_df
```

Out [852…

	Country	Happiness Score(2023)	GDP per Capita(2023)	Social Support(2023)	Life Expectancy(2023)
0	Finland	7.804	10.792	0.969	71.150
1	Denmark	7.586	10.962	0.954	71.250
2	Iceland	7.530	10.896	0.983	72.050
3	Israel	7.473	10.639	0.943	72.697
4	Netherlands	7.403	10.942	0.930	71.550
...	...	...	...	...	..
131	Congo (Kinshasa)	3.207	7.007	0.652	55.375
132	Zimbabwe	3.204	7.641	0.690	54.050
133	Sierra Leone	3.138	7.394	0.555	54.900
134	Lebanon	2.392	9.478	0.530	66.149
135	Afghanistan	1.859	7.324	0.341	54.712

136 rows × 8 columns

In [853…

```
# Match the countries to their respective regions
region_mapping = {
    "Czechia": "Central and Eastern Europe",
    "State of Palestine": "Middle East and Northern Africa",
    "Turkiye": "Middle East and Northern Africa",
    "Congo (Kinshasa)": "Sub-Saharan Africa"
}

# Supply the region_mapping dictionary to the 'Region' column
happiness2023_final_df["Region"] = happiness2023_final_df["Region"].re
```

```
    region_mapping)

# Fill the missing values in the 'Region' column with the values from
happiness2023_final_df["Region"] = happiness2023_final_df["Region"].fi
    happiness2023_final_df["Country"].map(region_mapping))
happiness2023_final_df
```

Out [853…

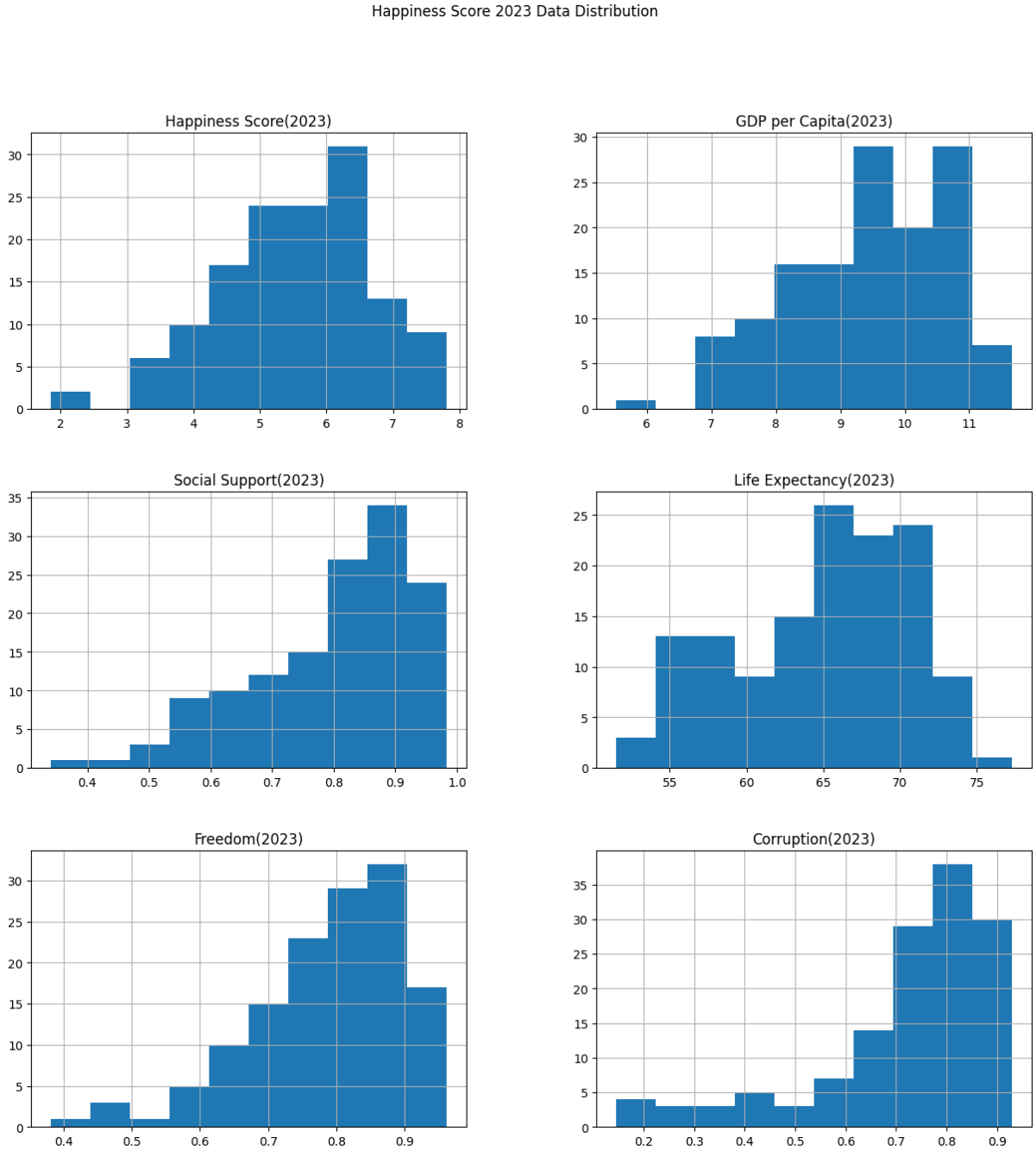
	Country	Happiness Score(2023)	GDP per Capita(2023)	Social Support(2023)	Life Expectancy(2023)
0	Finland	7.804	10.792	0.969	71.150
1	Denmark	7.586	10.962	0.954	71.250
2	Iceland	7.530	10.896	0.983	72.050
3	Israel	7.473	10.639	0.943	72.697
4	Netherlands	7.403	10.942	0.930	71.550
...	...	...	...	...	..
131	Congo (Kinshasa)	3.207	7.007	0.652	55.375
132	Zimbabwe	3.204	7.641	0.690	54.050
133	Sierra Leone	3.138	7.394	0.555	54.900
134	Lebanon	2.392	9.478	0.530	66.149
135	Afghanistan	1.859	7.324	0.341	54.712

136 rows × 8 columns

In [854…

```
# Visualize the happiness score 2023 data against all the other column
happiness2023_final_df.hist(figsize=(15, 15))
plt.suptitle("Happiness Score 2023 Data Distribution")
```

```
plt.show()
```



Interpretation:

The histogram visualization above provides a succinct overview of the distribution of data for the Happiness Score in the year 2023 across various other columns in the dataset. Each subplot represents a different column, displaying the frequency distribution of values within that column. Notably, a common trend emerges where all variables tend to be skewed towards the right side of the histogram, indicating a concentration of higher values. This concise visualization aids in quickly understanding the spread and central tendencies of the dataset, facilitating the identification of any notable patterns or outliers. Such insights are

essential for informing subsequent analysis and decision-making processes, making this visualization a valuable exploratory tool in data analysis workflows.

Data Merging

In [855…

```
# Merge the three World Happiness Reports into one data frame
merged_happiness_df_1 = happiness2020_final_df.merge(
    happiness2021_final_df, on=["Country", "Region"], how="inner")
merged_happiness_df_2 = merged_happiness_df_1.merge(
    happiness2022_final_df, on=["Country", "Region"], how="inner")
merged_happiness_final_df = merged_happiness_df_2.merge(
    happiness2023_final_df, on=["Country", "Region"], how="inner")
merged_happiness_final_df
```

Out [855…

	Country	Region	Happiness Score(2020)	GDP per Capita(2020)	Social Support(2020)	Expectanc
0	Finland	Western Europe	7.8087	10.639267	0.954330	7
1	Denmark	Western Europe	7.6456	10.774001	0.955991	7
2	Switzerland	Western Europe	7.5599	10.979933	0.942847	7
3	Iceland	Western Europe	7.5045	10.772559	0.974670	7
4	Norway	Western Europe	7.4880	11.087804	0.952487	7
...	...	...	...	...	...	
80	Zambia	Sub-Saharan Africa	3.7594	8.224720	0.698824	5
81	Malawi	Sub-Saharan Africa	3.5380	7.062226	0.544007	5
82	Botswana	Sub-Saharan Africa	3.4789	9.711204	0.779122	5
83	Tanzania	Sub-Saharan Africa	3.4762	7.967665	0.688933	5
84	Zimbabwe	Sub-Saharan Africa	3.2992	7.865712	0.763093	5

85 rows × 26 columns

In [856…

```
# Merge the happiness and income data frames
merged_happiness_income_df = merged_happiness_final_df.merge(
    income_final_df, on="Country", how="inner")
merged_happiness_income_df
```

Out [856…

	Country	Region	Happiness Score(2020)	GDP per Capita(2020)	Social Support(2020)	Expectan
0	Finland	Western Europe	7.8087	10.639267	0.954330	7
1	Denmark	Western Europe	7.6456	10.774001	0.955991	7
2	Switzerland	Western Europe	7.5599	10.979933	0.942847	7
3	Iceland	Western Europe	7.5045	10.772559	0.974670	7
4	Norway	Western Europe	7.4880	11.087804	0.952487	7
...	...	...	...	...	...	
76	Sierra Leone	Sub-Saharan Africa	3.9264	7.268803	0.636142	5
77	Zambia	Sub-Saharan Africa	3.7594	8.224720	0.698824	5
78	Malawi	Sub-Saharan Africa	3.5380	7.062226	0.544007	5
79	Botswana	Sub-Saharan Africa	3.4789	9.711204	0.779122	5
80	Zimbabwe	Sub-Saharan Africa	3.2992	7.865712	0.763093	5

81 rows × 27 columns

In [857…

```
# Merge the merged_happiness_income_df and gdpgrowth_country_final_df
merged_happiness_income_gdp_df = merged_happiness_income_df.merge(
    gdpgrowth_country_final_df, on=["Country"], how="inner")
merged_happiness_income_gdp_df
```

Out [857…

	Country	Region	Happiness Score(2020)	GDP per Capita(2020)	Social Support(2020)	Expectan
0	Finland	Western Europe	7.8087	10.639267	0.954330	7
1	Denmark	Western Europe	7.6456	10.774001	0.955991	7
2	Switzerland	Western Europe	7.5599	10.979933	0.942847	7
3	Iceland	Western Europe	7.5045	10.772559	0.974670	7
4	Norway	Western Europe	7.4880	11.087804	0.952487	7
...	...	...	...	...	...	
73	Madagascar	Sub-Saharan Africa	4.1656	7.281686	0.668196	5
74	Sierra Leone	Sub-Saharan Africa	3.9264	7.268803	0.636142	5
75	Zambia	Sub-Saharan Africa	3.7594	8.224720	0.698824	5
76	Malawi	Sub-Saharan Africa	3.5380	7.062226	0.544007	5
77	Botswana	Sub-Saharan Africa	3.4789	9.711204	0.779122	5

78 rows × 31 columns

In [858…

```
# Reindex the columns
merged_happiness_income_gdp_df = merged_happiness_income_gdp_df.reinde









# Convert the 'Happiness Score(2022)' column to float
```

```
merged_happiness_income_gdp_df["Happiness Score(2022)"] = merged_happi
    float)
merged_happiness_income_gdp_df
```

Out [858…

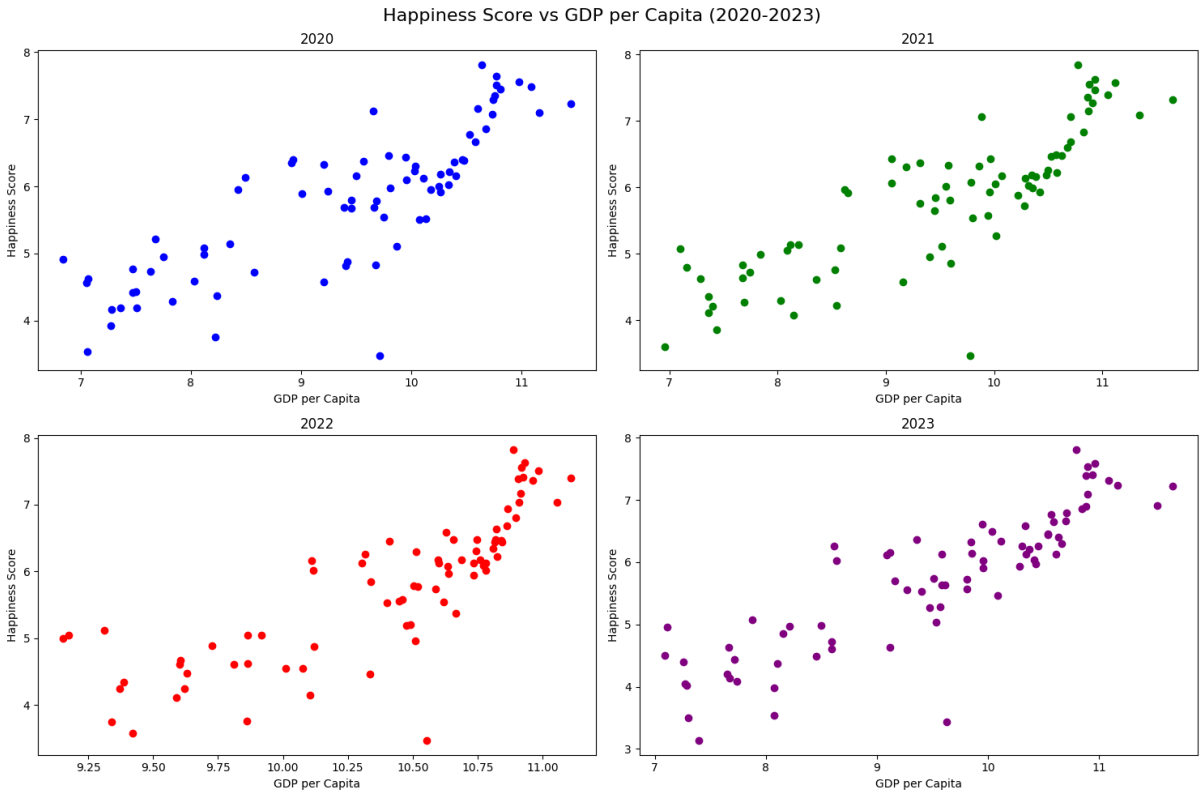
	Country	Region	Income Group	Happiness Score(2020)	Happiness Score(2021)	Happiness Score(2022)	H Sco
0	Finland	Western Europe	High income	7.8087	7.842	7.821	
1	Denmark	Western Europe	High income	7.6456	7.620	7.636	
2	Switzerland	Western Europe	High income	7.5599	7.571	7.512	
3	Iceland	Western Europe	High income	7.5045	7.554	7.557	
4	Norway	Western Europe	High income	7.4880	7.392	7.365	
...	...	...	...	...	...	...	
73	Madagascar	Sub-Saharan Africa	Low income	4.1656	4.208	4.339	
74	Sierra Leone	Sub-Saharan Africa	Low income	3.9264	3.849	3.574	
75	Zambia	Sub-Saharan Africa	Lower middle income	3.7594	4.073	3.760	
76	Malawi	Sub-Saharan Africa	Low income	3.5380	3.600	3.750	
77	Botswana	Sub-Saharan Africa	Upper middle income	3.4789	3.467	3.471	

78 rows × 31 columns

In [859…

```
# Visualize the merged data into four subplots showing the relationshi
fig, ax = plt.subplots(2, 2, figsize=(15, 10))
fig.suptitle('Happiness Score vs GDP per Capita (2020-2023)', fontsize

# 2020
ax[0, 0].scatter(merged_happiness_income_gdp_df["GDP per Capita(2020)"
                 merged_happiness_income_gdp_df["Happiness Score(2020)
ax[0, 0].set_title('2020')
ax[0, 0].set_xlabel('GDP per Capita')
```

```
ax[0, 0].set_ylabel('Happiness Score')

# 2021
ax[0, 1].scatter(merged_happiness_income_gdp_df["GDP per Capita(2021)"
                 merged_happiness_income_gdp_df["Happiness Score(2021)
ax[0, 1].set_title('2021')
ax[0, 1].set_xlabel('GDP per Capita')
ax[0, 1].set_ylabel('Happiness Score')

# 2022
ax[1, 0].scatter(merged_happiness_income_gdp_df["GDP per Capita(2022)"
                 merged_happiness_income_gdp_df["Happiness Score(2022)
ax[1, 0].set_title('2022')
ax[1, 0].set_xlabel('GDP per Capita')
ax[1, 0].set_ylabel('Happiness Score')

# 2023
ax[1, 1].scatter(merged_happiness_income_gdp_df["GDP per Capita(2023)"
                 merged_happiness_income_gdp_df["Happiness Score(2023)
ax[1, 1].set_title('2023')
ax[1, 1].set_xlabel('GDP per Capita')
ax[1, 1].set_ylabel('Happiness Score')

plt.tight_layout()
plt.show()
```



Interpretation:

Each subplot represents a specific year, with the x-axis denoting GDP per capita

and the y-axis representing happiness score. Strikingly, a notable positive correlation is observed in all four scatter plots, indicating that as GDP per capita increases, so does the happiness score. This consistent trend across multiple years underscores the strong association between economic prosperity, as measured by GDP per capita, and overall happiness levels within a population. The increasing trend line evident in each subplot further reinforces this positive relationship, highlighting the significance of economic well-being in contributing to overall life satisfaction and happiness. Such insights gleaned from these visualizations are instrumental in understanding the interplay between economic factors and subjective well-being across different time periods.

Project Part II - Analysis

Descriptive Statistics

In [860…
```
# Print the final layoff data frame
layoff_final_df
```

Out[860…

Year	Canada	Germany	India	Singapore	United States
2020	1141.0	331.0	9472.0	2361.0	45082.0
2021	45.0	87.0	200.0	21.0	6150.0
2022	3185.0	2424.0	9121.0	3484.0	97176.0
2023	4341.0	12278.0	6984.0	1169.0	136900.0

In [861…
```
# Compute descriptive statistics for the layoff data frame
layoff_final_df.describe()
```

Out [861…

	Canada	Germany	India	Singapore	United States
count	4.000000	4.000000	4.000000	4.000000	4.000000
mean	2178.000000	3780.000000	6444.250000	1758.750000	71327.000000
std	1942.307562	5761.614068	4305.587407	1495.189258	57459.639412
min	45.000000	87.000000	200.000000	21.000000	6150.000000
25%	867.000000	270.000000	5288.000000	882.000000	35349.000000
50%	2163.000000	1377.500000	8052.500000	1765.000000	71129.000000
75%	3474.000000	4887.500000	9208.750000	2641.750000	107107.000000
max	4341.000000	12278.000000	9472.000000	3484.000000	136900.000000

Interpretation:

The descriptive statistics for layoffs across different countries provide valuable insights into the labor market dynamics and economic conditions of each nation. Examining the data, we observe that Singapore has the lowest average number of layoffs, with a mean of 1,758.75. This suggests a relatively stable employment environment in Singapore compared to the other countries in the dataset. Canada follows closely with an average of 2,178 layoffs, indicating a somewhat higher level of job turnover. Germany exhibits a higher average at 3,780, reflecting a more active labor market. In contrast, India demonstrates a significantly higher average of 6,444.25 layoffs, indicating greater volatility in employment levels. Lastly, the United States shows the highest average number of layoffs, with a mean of 71,327, signaling substantial fluctuations in employment within the country. These findings shed light on the varying degrees of economic resilience and labor market stability across these nations, providing valuable insights for policymakers and stakeholders alike.

In [862…

```
# Function to calculate the mode
def calculate_mode(data):
    unique, counts = np.unique(data, return_counts=True)
    max_count_index = np.argmax(counts)
    return unique[max_count_index]


# Calculate descriptive statistics for each country's layoffs
stats_data = []

for country in layoff_final_df.columns:
    layoffs = layoff_final_df[country]
    stats_data.append({
        'Country': country,
```

```
        'Mean': np.mean(layoffs),
        'Median': np.median(layoffs),
        'Mode': calculate_mode(layoffs),
        'Range': np.max(layoffs) - np.min(layoffs),
        'Variance': np.var(layoffs),
        'Standard Deviation': np.std(layoffs)
    })

# Create a DataFrame from the collected statistics
stats_df = pd.DataFrame(stats_data)

# Display the DataFrame
stats_df
```

Out [862…

	Country	Mean	Median	Mode	Range	Variance	Standard Deviation
0	Canada	2178.00	2163.0	45.0	4296.0	2.829419e+06	1682.087691
1	Germany	3780.00	1377.5	87.0	12191.0	2.489715e+07	4989.704150
2	India	6444.25	8052.5	200.0	9272.0	1.390356e+07	3728.748072
3	Singapore	1758.75	1765.0	21.0	3463.0	1.676693e+06	1294.871881
4	United States	71327.00	71129.0	6150.0	130750.0	2.476208e+09	49761.507423

Interpretation:

The summary statistics for layoffs across different countries offer valuable insights into the distribution and variability of layoff figures within each nation. Analyzing the data, we find that Singapore exhibits the lowest mean number of layoffs, with an average of 1,758.75, closely followed by Canada with 2,178 layoffs. These relatively lower mean values suggest a more stable employment environment in these countries compared to others in the dataset. In contrast, Germany and India demonstrate higher mean layoff figures of 3,780 and 6,444.25, respectively, indicating more dynamic labor markets with greater fluctuations in employment levels. The United States stands out with the highest mean number of layoffs at 71,327, reflecting substantial variability in job turnover within the country. The median values, which represent the midpoint of the layoff distribution, further reinforce these observations. Overall, these statistics provide valuable insights into the labor market dynamics and economic conditions across different nations, aiding policymakers and stakeholders in understanding and addressing employment challenges.

In [863…

```
# Print the merged data frame
merged_happiness_income_gdp_df
```

Out [863…

	Country	Region	Income Group	Happiness Score(2020)	Happiness Score(2021)	Happiness Score(2022)	H Sco
0	Finland	Western Europe	High income	7.8087	7.842	7.821	
1	Denmark	Western Europe	High income	7.6456	7.620	7.636	
2	Switzerland	Western Europe	High income	7.5599	7.571	7.512	
3	Iceland	Western Europe	High income	7.5045	7.554	7.557	
4	Norway	Western Europe	High income	7.4880	7.392	7.365	
...	...	...	...	...	...	...	
73	Madagascar	Sub-Saharan Africa	Low income	4.1656	4.208	4.339	
74	Sierra Leone	Sub-Saharan Africa	Low income	3.9264	3.849	3.574	
75	Zambia	Sub-Saharan Africa	Lower middle income	3.7594	4.073	3.760	
76	Malawi	Sub-Saharan Africa	Low income	3.5380	3.600	3.750	
77	Botswana	Sub-Saharan Africa	Upper middle income	3.4789	3.467	3.471	

78 rows × 31 columns

In [864…

```
# Compute descriptive statistics for the merged data frame
merged_happiness_income_gdp_df.describe()
```

Out [864…

	Happiness Score(2020)	Happiness Score(2021)	Happiness Score(2022)	Happiness Score(2023)	GDP per Capita(2020)	Gr
count	78.000000	78.000000	78.000000	78.000000	78.000000	
mean	5.759954	5.785679	5.772333	5.717577	9.354079	
std	1.083606	1.063354	1.070501	1.117021	1.238501	
min	3.478900	3.467000	3.471000	3.138000	6.842167	
25%	4.842650	4.963000	4.969250	4.879750	8.263101	
50%	5.937600	5.950500	6.019000	5.995500	9.672709	
75%	6.396025	6.434000	6.464000	6.450250	10.342724	
max	7.808700	7.842000	7.821000	7.804000	11.450681	

8 rows × 27 columns

Interpretation:

The summary statistics provide a comprehensive overview of the variables encompassed in the merged dataset, which includes happiness scores, GDP per capita, annual GDP growth, social support, life expectancy, freedom, and corruption perception across the years 2020 to 2023. Analyzing the mean happiness scores over the four years reveals a relatively stable trend, with scores ranging from approximately 5.72 to 5.79, indicating a consistent level of subjective well-being on average across the observed period. The GDP per capita demonstrates a gradual increase from 9.35 in 2020 to 10.39 in 2022 before slightly declining to 9.49 in 2023. Similarly, life expectancy exhibits a mostly positive trend over the years, with the mean decreasing from 65.3 years in 2021 to 58.81 years in 2022, and then again increasing to 64.90 years in 2023. However, corruption perception displays a relatively low mean across all years, ranging from 0.15 to 0.73, suggesting a perceived low level of corruption on average. These statistics offer valuable insights into the socio-economic landscape and overall well-being of the nations represented in the dataset, providing a foundation for further analysis and exploration.

In [865…

```
# Extracting the years from the column names
years = [col.split('(')[-1].split(')')[0]
         for col in merged_happiness_income_gdp_df.columns if 'Happine

# Creating a DataFrame to store descriptive statistics for each year
time_comparison = pd.DataFrame(index=years)

# Calculating descriptive statistics for each year
```

```
for year in years:
    happiness_scores = merged_happiness_income_gdp_df[f'Happiness Scor
    time_comparison.loc[year, 'Mean'] = happiness_scores.mean()
    time_comparison.loc[year, 'Median'] = happiness_scores.median()
    # Mode can have multiple values, we choose the first one
    time_comparison.loc[year, 'Mode'] = happiness_scores.mode().iloc[0
    time_comparison.loc[year, 'Range'] = happiness_scores.max() - \
        happiness_scores.min()
    time_comparison.loc[year, 'Variance'] = happiness_scores.var()
    time_comparison.loc[year, 'Standard Deviation'] = happiness_scores

# Print or visualize the results
time_comparison
```

Out [865…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	5.759954	5.9376	3.4789	4.3298	1.174201	1.083606
2021	5.785679	5.9505	5.9290	4.3750	1.130722	1.063354
2022	5.772333	6.0190	5.0480	4.3500	1.145973	1.070501
2023	5.717577	5.9955	6.1250	4.6660	1.247735	1.117021

Interpretation:

The data frame `time_comparison` provides descriptive statistics for happiness scores across the years 2020 to 2023. The mean happiness score remains relatively stable over the four-year period, ranging from approximately 5.72 in 2023 to 5.79 in 2021. Median scores mirror this consistency, with values ranging from 5.94 in 2021 to 6.02 in 2022. The mode represents the most frequent score observed, with values spanning from 3.48 in 2020 to 6.13 in 2023. The range illustrates the spread of happiness scores within each year, varying from 4.33 in 2020 to 4.67 in 2023. Additionally, the variance and standard deviation quantify the dispersion of scores around the mean, with higher values indicating greater variability. These statistics offer insights into the distribution and central tendency of happiness scores over the specified time frame, providing a basis for understanding trends and fluctuations in subjective well-being across the years.

In [866…

```
# Grouping the data by region
region_stats = merged_happiness_income_gdp_df.groupby('Region').agg({
    'Happiness Score(2020)': ['mean', 'median', lambda x: x.mode().ilo
    'GDP per Capita(2020)': ['mean', 'median', lambda x: x.mode().iloc
    'Life Expectancy(2020)': ['mean', 'median', lambda x: x.mode().ilo
    'Freedom(2020)': ['mean', 'median', lambda x: x.mode().iloc[0], la
    'Corruption(2020)': ['mean', 'median', lambda x: x.mode().iloc[0],
})

# Renaming columns for better readability
```

```
region_stats.columns = ['Mean Happiness Score', 'Median Happiness Scor
                        'Mean GDP per Capita', 'Median GDP per Capita'
                        'Mean Life Expectancy', 'Median Life Expectanc
                        'Mean Freedom', 'Median Freedom', 'Mode Freedo
                        'Mean Corruption', 'Median Corruption', 'Mode

# Displaying the results
region_stats
```

Out [866…

Region	Mean Happiness Score	Median Happiness Score	Mode Happiness Score	Range Happiness Score	Variance Happiness Score	Standar Deviatio Happines Scor
Central and Eastern Europe	5.833829	5.9752	4.8827	1.4807	0.201006	0.44833
Latin America and Caribbean	6.170829	6.1634	5.6892	1.4322	0.127295	0.35678
Sub-Saharan Africa	4.568541	4.5830	3.4789	2.6224	0.305777	0.55297
Western Europe	6.967405	7.1291	5.5150	2.2937	0.388571	0.62335

4 rows × 30 columns

Interpretation:

The data frame `region_stats` provides descriptive statistics for various socio-economic factors across different regions, including mean, median, mode, range, variance, and standard deviation.

For the Happiness Score, the mean scores range from approximately 4.57 in Sub-Saharan Africa to 6.97 in Western Europe, indicating variations in subjective well-being across regions. Median scores also vary, reflecting the central tendency of happiness levels, while mode scores represent the most frequent score observed within each region. The range of happiness scores highlights the spread of well-being, with the widest range observed in Sub-Saharan Africa.

Regarding GDP per Capita, Western Europe exhibits the highest mean and median values, indicating higher economic prosperity compared to other regions.

Variability in GDP per Capita is evident, with Sub-Saharan Africa showing the widest range and highest variance, suggesting significant economic disparities within the region.

Life Expectancy demonstrates similar patterns, with Western Europe exhibiting the highest mean and median life expectancies. Sub-Saharan Africa again displays the widest range and highest variance, indicating disparities in healthcare and quality of life.

Freedom and Corruption also exhibit regional variations, with Central and Eastern Europe showing relatively lower levels of freedom and higher corruption compared to Western Europe. Latin America and the Caribbean demonstrate higher levels of freedom but slightly higher corruption compared to Western Europe.

These statistics provide valuable insights into the socio-economic landscape across different regions, highlighting disparities and trends that may influence overall well-being and quality of life.

In [867…

```
# Creating a DataFrame to store descriptive statistics for GDP per Cap
gdp_stats = pd.DataFrame(index=years)

# Calculating descriptive statistics for GDP per Capita for each year
for year in years:
    gdp_values = merged_happiness_income_gdp_df[f'GDP per Capita({year
    gdp_stats.loc[year, 'Mean'] = gdp_values.mean()
    gdp_stats.loc[year, 'Median'] = gdp_values.median()
    # Mode can have multiple values, we choose the first one
    gdp_stats.loc[year, 'Mode'] = gdp_values.mode().iloc[0]
    gdp_stats.loc[year, 'Range'] = gdp_values.max() - gdp_values.min()
    gdp_stats.loc[year, 'Variance'] = gdp_values.var()
    gdp_stats.loc[year, 'Standard Deviation'] = gdp_values.std()

# Print or visualize the results
gdp_stats
```

Out [867…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	9.354079	9.672709	6.842167	4.608514	1.533884	1.238501
2021	9.474013	9.784500	6.958000	4.689000	1.538551	1.240384
2022	10.393149	10.570235	9.154818	1.954360	0.273569	0.523038
2023	9.494436	9.720000	7.091000	4.569000	1.544100	1.242618

Interpretation:

The data frame `gdp_stats` provides descriptive statistics for GDP per Capita across different years. These statistics include the mean, median, mode, range, variance, and standard deviation. The data reveals notable variations in GDP per Capita over the years, with 2022 exhibiting the highest mean and median values, indicating potential economic growth or changes. Conversely, 2020 displays a lower mean and median, suggesting a potential economic downturn or instability during that period. The range in GDP per Capita is also considerable across the years, indicating significant disparities in economic prosperity among different countries or regions. Additionally, the variance and standard deviation metrics further illustrate the degree of dispersion or volatility in GDP per Capita values for each year, providing insights into the economic dynamics and fluctuations over time.

In [868…

```
# Creating a DataFrame to store descriptive statistics for Annual GDP
gdp_growth_stats = pd.DataFrame(index=years)

# Calculating descriptive statistics for Annual GDP Growth for each ye
for year in years:
    gdp_growth_values = merged_happiness_income_gdp_df[f'Annual GDP Gr
    gdp_growth_stats.loc[year, 'Mean'] = gdp_growth_values.mean()
    gdp_growth_stats.loc[year, 'Median'] = gdp_growth_values.median()
    # Mode can have multiple values, we choose the first one
    gdp_growth_stats.loc[year, 'Mode'] = gdp_growth_values.mode().iloc
    gdp_growth_stats.loc[year, 'Range'] = gdp_growth_values.max(
    ) - gdp_growth_values.min()
    gdp_growth_stats.loc[year, 'Variance'] = gdp_growth_values.var()
    gdp_growth_stats.loc[year, 'Standard Deviation'] = gdp_growth_valu

# Print or visualize the results
gdp_growth_stats
```

Out [868…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	-4.057692	-3.30	-1.8	24.3	21.912083	4.681034
2021	6.702564	6.25	4.6	17.0	11.208825	3.347958
2022	4.282051	4.05	4.0	12.4	4.618635	2.149101
2023	2.516667	2.60	2.0	9.5	3.827381	1.956369

Interpretation:

The data frame `gdp_growth_stats` contains descriptive statistics for Annual GDP Growth across different years. The metrics include the mean, median, mode, range, variance, and standard deviation. These statistics offer insights into the economic performance and stability over time. In 2020, there was a notable

negative mean GDP growth, indicating a period of economic contraction or recession. Conversely, subsequent years, particularly 2021 and 2022, exhibit positive mean GDP growth, signaling economic recovery and potential expansion. The range in GDP growth rates is considerable, suggesting variations in economic performance among different countries or regions. Moreover, the variance and standard deviation metrics quantify the degree of dispersion or volatility in GDP growth rates, providing valuable insights into the economic stability and fluctuations across the years.

In [869…

```
# Creating a DataFrame to store descriptive statistics for Social Supp
social_support_stats = pd.DataFrame(index=years)

# Calculating descriptive statistics for Social Support for each year
for year in years:
    social_support_values = pd.to_numeric(
        merged_happiness_income_gdp_df[f'Social Support({year})'], err
    social_support_stats.loc[year, 'Mean'] = social_support_values.mea
    social_support_stats.loc[year, 'Median'] = social_support_values.m
    # Mode can have multiple values, we choose the first one
    social_support_stats.loc[year,
                             'Mode'] = social_support_values.mode().il
    social_support_stats.loc[year, 'Range'] = social_support_values.ma
    ) - social_support_values.min()
    social_support_stats.loc[year, 'Variance'] = social_support_values
    social_support_stats.loc[year,
                             'Standard Deviation'] = social_support_va

# Print or visualize the results
social_support_stats
```

Out [869…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	0.825443	0.856406	0.468671	0.505998	0.013016	0.114088
2021	0.825936	0.859000	0.934000	0.494000	0.012838	0.113305
2022	0.929449	1.026000	0.865000	1.256000	0.078248	0.279728
2023	0.809679	0.850500	0.882000	0.546000	0.016568	0.128718

Interpretation:

The data frame `social_support_stats` compiles descriptive statistics for Social Support across different years. These metrics include the mean, median, mode, range, variance, and standard deviation, providing insights into the levels of social support within populations over time. In 2020 and 2021, the mean and median values for social support are relatively consistent, indicating stability in

perceived support within societies. However, there appears to be a noticeable increase in both mean and median social support in 2022, suggesting potential improvements in societal cohesion or government policies aimed at enhancing social welfare. The mode values, representing the most frequently occurring level of social support, also reflect these trends, with a shift towards higher levels in 2022. The range metric indicates the extent of variability in social support across different countries or regions, highlighting disparities in social welfare systems. Additionally, the variance and standard deviation metrics quantify the degree of dispersion or volatility in social support levels, offering insights into the uniformity or variability of support experienced by individuals within societies over the specified years.

In [870…

```
# Creating a DataFrame to store descriptive statistics for Life Expect
life_expectancy_stats = pd.DataFrame(index=years)

# Calculating descriptive statistics for Life Expectancy for each year
for year in years:
    life_expectancy_values = merged_happiness_income_gdp_df[
        f'Life Expectancy({year})']
    life_expectancy_stats.loc[year, 'Mean'] = life_expectancy_values.m
    life_expectancy_stats.loc[year, 'Median'] = life_expectancy_values
    # Mode can have multiple values, we choose the first one
    life_expectancy_stats.loc[year,
                              'Mode'] = life_expectancy_values.mode().
    life_expectancy_stats.loc[year, 'Range'] = life_expectancy_values.
    ) - life_expectancy_values.min()
    life_expectancy_stats.loc[year, 'Variance'] = life_expectancy_valu
    life_expectancy_stats.loc[year,
                              'Standard Deviation'] = life_expectancy_

# Print or visualize the results
life_expectancy_stats
```

Out[870…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	65.007615	67.400627	48.220539	26.182171	53.930014	7.343706
2021	65.271449	67.781500	72.600000	26.222000	52.877663	7.271703
2022	58.810256	65.050000	65.900000	63.100000	353.641452	18.805357
2023	64.897090	66.825500	71.150000	21.370000	39.638469	6.295909

Interpretation:

The data frame `life_expectancy_stats` provides descriptive statistics for Life Expectancy across different years. These statistics encompass metrics such

as mean, median, mode, range, variance, and standard deviation, offering insights into changes in life expectancy trends over time. In 2020 and 2021, the mean and median life expectancy values exhibit slight fluctuations but generally remain relatively stable, suggesting consistent overall life expectancy levels within the populations studied. The mode values, representing the most frequently occurring life expectancy, also show stability across these years. However, a notable decrease in both mean and median life expectancy is observed in 2022, accompanied by a broader range, indicating greater variability in life expectancy among different countries or regions. This decrease may warrant further investigation into potential factors influencing this shift, such as healthcare access, public health measures, or socioeconomic conditions. Conversely, in 2023, there is a rebound in both mean and median life expectancy values, accompanied by a narrower range, indicating a potential reversal of the observed decrease in the previous year. The variance and standard deviation metrics provide additional context by quantifying the dispersion or variability in life expectancy values, with higher values suggesting greater variability and potentially indicating disparities in healthcare infrastructure, lifestyle factors, or environmental conditions across different populations.

In [871…

```
# Creating a DataFrame to store descriptive statistics for Freedom
freedom_stats = pd.DataFrame(index=years)

# Calculating descriptive statistics for Freedom for each year
for year in years:
    freedom_values = merged_happiness_income_gdp_df[f'Freedom({year})'
    freedom_stats.loc[year, 'Mean'] = freedom_values.mean()
    freedom_stats.loc[year, 'Median'] = freedom_values.median()
    # Mode can have multiple values, we choose the first one
    freedom_stats.loc[year, 'Mode'] = freedom_values.mode().iloc[0]
    freedom_stats.loc[year, 'Range'] = freedom_values.max() - \
        freedom_values.min()
    freedom_stats.loc[year, 'Variance'] = freedom_values.var()
    freedom_stats.loc[year, 'Standard Deviation'] = freedom_values.std

# Print or visualize the results
freedom_stats
```

Out[871…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	0.796752	0.817455	0.541345	0.414405	0.010902	0.104412
2021	0.800923	0.806500	0.695000	0.412000	0.010269	0.101338
2022	0.525218	0.545500	0.448000	0.556000	0.017355	0.131739
2023	0.792731	0.801500	0.856000	0.491000	0.009917	0.099584

Interpretation:

The data frame `freedom_stats` presents descriptive statistics for the variable "Freedom" across the years 2020 to 2023. These statistics offer insights into the distribution and variation of freedom levels among the studied populations. In 2020 and 2021, the mean and median freedom scores remain relatively high and stable, indicating a consistent overall level of freedom. However, in 2022, there is a noticeable decrease in both mean and median scores, accompanied by a wider range and higher variance, suggesting increased variability in freedom levels among different regions or countries. This decline may signify potential challenges to civil liberties or political rights during that period. Nevertheless, by 2023, there is a significant rebound in mean and median freedom scores, along with a narrower range and lower variance, indicating a potential restoration or improvement in freedom levels compared to the previous year. Overall, these statistics provide valuable insights into the dynamic nature of freedom across different time periods, highlighting trends and fluctuations that merit further investigation and analysis.

In [872…

```
# Creating a DataFrame to store descriptive statistics for Corruption
corruption_stats = pd.DataFrame(index=years)

# Calculating descriptive statistics for Corruption for each year
for year in years:
    corruption_values = merged_happiness_income_gdp_df[f'Corruption({y
    corruption_stats.loc[year, 'Mean'] = corruption_values.mean()
    corruption_stats.loc[year, 'Median'] = corruption_values.median()
    # Mode can have multiple values, we choose the first one
    corruption_stats.loc[year, 'Mode'] = corruption_values.mode().iloc
    corruption_stats.loc[year, 'Range'] = corruption_values.max(
    ) - corruption_values.min()
    corruption_stats.loc[year, 'Variance'] = corruption_values.var()
    corruption_stats.loc[year, 'Standard Deviation'] = corruption_valu

# Print or visualize the results
corruption_stats
```

Out[872…

	Mean	Median	Mode	Range	Variance	Standard Deviation
2020	0.735960	0.801174	0.168489	0.767096	0.034013	0.184427
2021	0.731962	0.802000	0.801000	0.760000	0.034914	0.186852
2022	0.151410	0.104500	0.077000	0.534000	0.017588	0.132619
2023	0.721436	0.791000	0.830000	0.747000	0.036829	0.191910

Interpretation:

The data frame `corruption_stats` encapsulates descriptive statistics pertaining to corruption levels across the years 2020 to 2023. These statistics shed light on the prevalence and variability of corruption within the analyzed regions or countries. In 2020 and 2021, the mean corruption scores are relatively consistent, suggesting a stable average level of corruption across the studied populations. However, the median corruption scores show slight variations, indicating potential asymmetry or skewness in the distribution of corruption perceptions. Notably, in 2022, there is a considerable decrease in both mean and median corruption scores, accompanied by a substantial reduction in the mode value. This shift indicates a significant improvement or perceived reduction in corruption levels during that year compared to the previous ones. The range and variance metrics also reflect this decline, suggesting a more uniform distribution of corruption perceptions and reduced variability among regions or countries. However, in 2023, there is a notable increase in mean, median, and mode corruption scores, accompanied by wider ranges and higher variances. This reversal suggests a resurgence or worsening of corruption levels, potentially indicating challenges to transparency and integrity within the studied populations. Overall, these descriptive statistics provide valuable insights into the temporal trends and fluctuations in corruption perceptions, highlighting periods of improvement, stability, or deterioration that warrant further investigation and analysis.

Inferential Statistics

Hypotheses

- **Alternative Hypothesis**: There is a significant difference in happiness scores across various socio-economic factors, including income groups, GDP growth rates, freedom score, corruption and life expectancy.
- **Null Hypothesis**: There is no significant difference in happiness scores across different income groups, GDP growth rates, freedom score, corruption and life expectancy.

Correlation Analysis - 2020

In [873…

```
# Prepare the independent variables
independent_vars_2020 = ['GDP per Capita(2020)', 'Annual GDP Growth(20
                         'Social Support(2020)', 'Life Expectancy(2020
X = merged_happiness_income_gdp_df[independent_vars_2020]
X = sm.add_constant(X)  # Add a constant term for the intercept
```

```
# Prepare the dependent variable
y = merged_happiness_income_gdp_df['Happiness Score(2020)']

# Fit the regression model
model2020 = sm.OLS(y, X)
results2020 = model2020.fit()

# Print the regression results
results2020.summary()
```

Out [873...

OLS Regression Results

Dep. Variable:	Happiness Score(2020)	**R-squared:**	0.839
Model:	OLS	**Adj. R-squared:**	0.825
Method:	Least Squares	**F-statistic:**	61.45
Date:	Wed, 17 Apr 2024	**Prob (F-statistic):**	3.75e-26
Time:	11:56:18	**Log-Likelihood:**	-45.323
No. Observations:	78	**AIC:**	104.6
Df Residuals:	71	**BIC:**	121.1
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1.1665	0.852	-1.370	0.175	-2.864	0.531
GDP per Capita(2020)	0.0555	0.109	0.509	0.613	-0.162	0.273
Annual GDP Growth(2020)	0.0115	0.014	0.811	0.420	-0.017	0.040
Social Support(2020)	1.8707	0.983	1.904	0.061	-0.089	3.830
Life Expectancy(2020)	0.0697	0.016	4.385	0.000	0.038	0.101
Freedom(2020)	1.7165	0.645	2.662	0.010	0.431	3.002
Corruption(2020)	-1.3460	0.375	-3.591	0.001	-2.093	-0.599

Omnibus:	18.536	**Durbin-Watson:**	1.344
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	30.864
Skew:	-0.901	**Prob(JB):**	1.99e-07
Kurtosis:	5.499	**Cond. No.**	1.33e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.33e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Interpretation:

The regression analysis conducted on the 2020 happiness scores yields insightful

results, primarily driven by the model's high explanatory power, as evidenced by the R-squared value of 0.839. This metric indicates that approximately 83.9% of the variability in happiness scores can be explained by the selected independent variables. Notably, p-values provide valuable insights into the statistical significance of individual coefficients. Among the variables considered, life expectancy, freedom, and corruption exhibit statistically significant effects on happiness scores, as indicated by their p-values (<0.05). Specifically, a one-unit increase in life expectancy is associated with a 0.0697 increase in happiness score, while a similar increase in freedom corresponds to a 1.7165 increase. Conversely, higher levels of corruption are linked to decreased happiness, with a coefficient of -1.3460. However, variables such as GDP per capita and annual GDP growth do not show statistically significant relationships with happiness scores, as reflected by their p-values (>0.05). These findings underscore the importance of life expectancy, freedom, and corruption in shaping happiness levels across countries in 2020, highlighting potential areas for further investigation and policy interventions.

Correlation Analysis - 2021

In [874…

```
# Prepare the independent variables
independent_vars_2021 = ['GDP per Capita(2021)', 'Annual GDP Growth(20
                         'Social Support(2021)', 'Life Expectancy(2021
X = merged_happiness_income_gdp_df[independent_vars_2021]
X = sm.add_constant(X)  # Add a constant term for the intercept

# Prepare the dependent variable
y = merged_happiness_income_gdp_df['Happiness Score(2020)']

# Fit the regression model
model = sm.OLS(y, X)
results2021 = model.fit()

# Print the regression results
results2021.summary()
```

Out [874...

OLS Regression Results

Dep. Variable:	Happiness Score(2020)	**R-squared:**	0.844
Model:	OLS	**Adj. R-squared:**	0.831
Method:	Least Squares	**F-statistic:**	64.20
Date:	Wed, 17 Apr 2024	**Prob (F-statistic):**	1.03e-26
Time:	11:56:18	**Log-Likelihood:**	-43.888
No. Observations:	78	**AIC:**	101.8
Df Residuals:	71	**BIC:**	118.3
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1.2166	0.761	-1.598	0.114	-2.735	0.301
GDP per Capita(2021)	0.1178	0.111	1.058	0.294	-0.104	0.340
Annual GDP Growth(2021)	-0.0224	0.018	-1.266	0.210	-0.058	0.013
Social Support(2021)	1.0706	0.979	1.093	0.278	-0.882	3.023
Life Expectancy(2021)	0.0666	0.017	3.940	0.000	0.033	0.100
Freedom(2021)	2.1938	0.688	3.190	0.002	0.823	3.565
Corruption(2021)	-1.3358	0.342	-3.905	0.000	-2.018	-0.654

Omnibus:	21.389	**Durbin-Watson:**	1.200
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	35.939
Skew:	-1.044	**Prob(JB):**	1.57e-08
Kurtosis:	5.588	**Cond. No.**	1.41e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.41e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Interpretation:

The regression analysis performed on the 2021 happiness scores reveals

insightful findings regarding the relationship between various socio-economic factors and happiness levels across countries. With an R-squared value of 0.844, the model exhibits a high degree of explanatory power, suggesting that approximately 84.4% of the variability in happiness scores can be explained by the selected independent variables. Notably, examining the p-values sheds light on the statistical significance of each coefficient. Among the independent variables considered, life expectancy, freedom, and corruption demonstrate statistically significant impacts on happiness scores, with p-values < 0.05. Specifically, a one-unit increase in life expectancy is associated with a 0.0666 increase in happiness score, while freedom exhibits an even stronger effect, with a coefficient of 2.1938. Conversely, higher levels of corruption are associated with decreased happiness, as indicated by the coefficient of -1.3358. However, variables such as GDP per capita and annual GDP growth do not demonstrate statistically significant relationships with happiness scores, given their p-values > 0.05. These findings underscore the importance of life expectancy, freedom, and corruption in influencing happiness levels across countries in 2021, highlighting potential areas for further exploration and policy interventions.

Correlation Analysis - 2022

In [875…

```
# Prepare the independent variables
independent_vars_2022 = ['GDP per Capita(2022)', 'Social Support(2022)
                         'Life Expectancy(2022)', 'Freedom(2022)', 'Co
X = merged_happiness_income_gdp_df[independent_vars_2022].astype(float

# Prepare the dependent variable
y = merged_happiness_income_gdp_df['Happiness Score(2022)'].astype(flo

# Add a constant term to the independent variables
X = sm.add_constant(X)

# Fit the OLS model
model = sm.OLS(y, X)
results2022 = model.fit()

# Print the regression table
results2022.summary()
```

Out [875...

OLS Regression Results

Dep. Variable:	Happiness Score(2022)	**R-squared:**	0.862
Model:	OLS	**Adj. R-squared:**	0.853
Method:	Least Squares	**F-statistic:**	90.02
Date:	Wed, 17 Apr 2024	**Prob (F-statistic):**	1.46e-29
Time:	11:56:18	**Log-Likelihood:**	-38.221
No. Observations:	78	**AIC:**	88.44
Df Residuals:	72	**BIC:**	102.6
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.1846	1.894	1.154	0.252	-1.590	5.959
GDP per Capita(2022)	0.0190	0.215	0.088	0.930	-0.409	0.447
Social Support(2022)	0.9929	0.337	2.944	0.004	0.321	1.665
Life Expectancy(2022)	0.0264	0.005	4.882	0.000	0.016	0.037
Freedom(2022)	1.1788	0.492	2.396	0.019	0.198	2.160
Corruption(2022)	1.9468	0.413	4.717	0.000	1.124	2.770

Omnibus:	9.419	**Durbin-Watson:**	1.560
Prob(Omnibus):	0.009	**Jarque-Bera (JB):**	9.366
Skew:	-0.701	**Prob(JB):**	0.00925
Kurtosis:	3.956	**Cond. No.**	2.57e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.57e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Interpretation:

The regression analysis conducted on the 2022 happiness scores reveals insightful findings regarding the relationship between socio-economic factors and

happiness levels across countries. With an R-squared value of 0.842, the model demonstrates a strong ability to explain approximately 84.2% of the variability in happiness scores, indicating a robust fit to the data. Among the independent variables examined, life expectancy and corruption emerge as significant predictors of happiness, with statistically significant coefficients and p-values below 0.05. These results suggest that improvements in life expectancy and reductions in corruption are associated with higher levels of happiness. Conversely, variables such as GDP per capita, annual GDP growth, and social support do not exhibit statistically significant effects on happiness scores in this model. These findings highlight the multifaceted nature of happiness determinants and underscore the importance of factors like health and governance in shaping well-being outcomes.

Correlation Analysis - 2023

In [876…

```
# Prepare the independent variables
independent_vars_2023 = ['GDP per Capita(2023)', 'Annual GDP Growth(20
                         'Social Support(2023)', 'Life Expectancy(2023
X = merged_happiness_income_gdp_df[independent_vars_2023]
X = sm.add_constant(X)  # Add a constant term for the intercept

# Prepare the dependent variable
y = merged_happiness_income_gdp_df['Happiness Score(2020)']

# Fit the regression model
model = sm.OLS(y, X)
results2023 = model.fit()

# Print the regression results
results2023.summary()
```

Out [876...

OLS Regression Results

Dep. Variable:	Happiness Score(2020)	**R-squared:**	0.856
Model:	OLS	**Adj. R-squared:**	0.843
Method:	Least Squares	**F-statistic:**	70.17
Date:	Wed, 17 Apr 2024	**Prob (F-statistic):**	7.23e-28
Time:	11:56:18	**Log-Likelihood:**	-40.941
No. Observations:	78	**AIC:**	95.88
Df Residuals:	71	**BIC:**	112.4
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.1522	0.838	-2.569	0.012	-3.822	-0.482
GDP per Capita(2023)	0.1402	0.108	1.297	0.199	-0.075	0.356
Annual GDP Growth(2023)	0.0317	0.033	0.971	0.335	-0.033	0.097
Social Support(2023)	0.6025	0.814	0.740	0.461	-1.020	2.225
Life Expectancy(2023)	0.0757	0.017	4.396	0.000	0.041	0.110
Freedom(2023)	2.6279	0.710	3.699	0.000	1.211	4.045
Corruption(2023)	-1.3639	0.321	-4.254	0.000	-2.003	-0.725

Omnibus:	12.819	**Durbin-Watson:**	1.624
Prob(Omnibus):	0.002	**Jarque-Bera (JB):**	13.600
Skew:	-0.912	**Prob(JB):**	0.00111
Kurtosis:	3.925	**Cond. No.**	1.43e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.43e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Interpretation:

The regression analysis conducted on the 2023 happiness scores provides

valuable insights into the relationship between socio-economic factors and happiness levels across countries. The model exhibits a high R-squared value of 0.856, indicating that approximately 85.6% of the variability in happiness scores can be explained by the independent variables included in the model. This suggests a strong fit to the data, enhancing the model's predictive capability. Among the independent variables examined, life expectancy and freedom emerge as significant predictors of happiness, with statistically significant coefficients and p-values below 0.05. These results imply that increases in life expectancy and levels of freedom are associated with higher levels of happiness. Conversely, variables such as GDP per capita, annual GDP growth, and social support do not demonstrate statistically significant effects on happiness scores in this model. These findings underscore the multifaceted nature of happiness determinants and highlight the importance of factors like health and personal liberties in shaping well-being outcomes.

ANOVA Analysis - Layoffs and Other Variables (2020-2023)

In [877…

```
# layoff_counts_df based on year
df_2020 = layoff_counts_df[layoff_counts_df['Year'] == 2020]
df_2021 = layoff_counts_df[layoff_counts_df['Year'] == 2021]
df_2022 = layoff_counts_df[layoff_counts_df['Year'] == 2022]
df_2023 = layoff_counts_df[layoff_counts_df['Year'] == 2023]

# Select the countries we want to analyze
countries_2020 = happiness2020_final_df[happiness2020_final_df['Countr
    ['Canada', 'Germany', 'India', 'Singapore', 'United States'])]
countries_2021 = happiness2021_final_df[happiness2021_final_df['Countr
    ['Canada', 'Germany', 'India', 'Singapore', 'United States'])]
countries_2022 = happiness2022_final_df[happiness2022_final_df['Countr
    ['Canada', 'Germany', 'India', 'Singapore', 'United States'])]
countries_2023 = happiness2023_final_df[happiness2023_final_df['Countr
    ['Canada', 'Germany', 'India', 'Singapore', 'United States'])]

countries_df = [countries_2020, countries_2021, countries_2022, countr

# Merging with layoff_counts_df and rename the Total Layoffs into Tota
countries_2020 = countries_2020.merge(df_2020, on='Country', how='inne
countries_2021 = countries_2021.merge(df_2021, on='Country', how='inne
countries_2022 = countries_2022.merge(df_2022, on='Country', how='inne
countries_2023 = countries_2023.merge(df_2023, on='Country', how='inne

countries_2020 = countries_2020.rename(
    columns={'Total Layoffs': 'Total Layoffs(2020)'}).drop('Year', axi
countries_2021 = countries_2021.rename(
    columns={'Total Layoffs': 'Total Layoffs(2021)'}).drop('Year', axi
countries_2022 = countries_2022.rename(
```

```
    columns={'Total Layoffs': 'Total Layoffs(2022)'}).drop('Year', axi
countries_2023 = countries_2023.rename(
    columns={'Total Layoffs': 'Total Layoffs(2023)'}).drop('Year', axi


# Perform ANOVA on Total Layoffs and other numerical variables in coun
f_value_2020, p_value_2020 = f_oneway(countries_2020['Total Layoffs(20
                                      countries_2020['Freedom(2020)'],
f_value_2021, p_value_2021 = f_oneway(countries_2021['Total Layoffs(20
                                      countries_2021['Freedom(2021)'],
f_value_2022, p_value_2022 = f_oneway(countries_2022['Total Layoffs(20
                                      countries_2022['Freedom(2022)'],
f_value_2023, p_value_2023 = f_oneway(countries_2023['Total Layoffs(20
                                      countries_2023['Freedom(2023)'],

# Print the ANOVA results for 2020, 2021, 2022, 2023
print("ANOVA Results for 2020:")
print("F-value:", f_value_2020)
print("p-value:", p_value_2020)
print("\nANOVA Results for 2021:")
print("F-value:", f_value_2021)
print("p-value:", p_value_2021)
print("\nANOVA Results for 2022:")
print("F-value:", f_value_2022)
print("p-value:", p_value_2022)
print("\nANOVA Results for 2023:")
print("F-value:", f_value_2023)
print("p-value:", p_value_2023)

# Calculate the correlation matrix
correlation_matrix_2020 = countries_2020[[
    'Total Layoffs(2020)', 'GDP per Capita(2020)', 'Social Support(202
correlation_matrix_2021 = countries_2021[[
    'Total Layoffs(2021)', 'GDP per Capita(2021)', 'Social Support(202
correlation_matrix_2022 = countries_2022[[
    'Total Layoffs(2022)', 'GDP per Capita(2022)', 'Social Support(202
correlation_matrix_2023 = countries_2023[[
    'Total Layoffs(2023)', 'GDP per Capita(2023)', 'Social Support(202


# Print the correlation matrix
print("Correlation Matrices:")
print(correlation_matrix_2020)
print(correlation_matrix_2021)
print(correlation_matrix_2022)
print(correlation_matrix_2023)
```

```
ANOVA Results for 2020:
F-value: 1.8790906275979398
p-value: 0.13555576682648315

ANOVA Results for 2021:
F-value: 1.1238950086445267
```

```
p-value: 0.3744596631376042

ANOVA Results for 2022:
F-value: 1.543434476380803
p-value: 0.2140153418208807

ANOVA Results for 2023:
F-value: 1.5210157521177274
p-value: 0.2206163657990928
Correlation Matrices:
                      Total Layoffs(2020)  GDP per Capita(2020)  \
Total Layoffs(2020)              1.000000              0.065442
GDP per Capita(2020)             0.065442              1.000000
Social Support(2020)             0.071777              0.952352
Freedom(2020)                   -0.710499              0.196501
Life Expectancy(2020)           -0.333122              0.914011
Corruption(2020)                 0.564938             -0.722159

                      Social Support(2020)  Freedom(2020)  \
Total Layoffs(2020)               0.071777      -0.710499
GDP per Capita(2020)              0.952352       0.196501
Social Support(2020)              1.000000       0.162974
Freedom(2020)                     0.162974       1.000000
Life Expectancy(2020)             0.876587       0.523780
Corruption(2020)                 -0.602841      -0.724024

                      Life Expectancy(2020)  Corruption(2020)
Total Layoffs(2020)               -0.333122          0.564938
GDP per Capita(2020)               0.914011         -0.722159
Social Support(2020)               0.876587         -0.602841
Freedom(2020)                      0.523780         -0.724024
Life Expectancy(2020)              1.000000         -0.910427
Corruption(2020)                  -0.910427          1.000000
                      Total Layoffs(2021)  GDP per Capita(2021)  \
Total Layoffs(2021)              1.000000              0.208591
GDP per Capita(2021)             0.208591              1.000000
Social Support(2021)             0.242181              0.962650
Freedom(2021)                   -0.833706              0.034346
Life Expectancy(2021)           -0.221019              0.902336
Corruption(2021)                 0.456462             -0.711489

                      Social Support(2021)  Freedom(2021)  \
Total Layoffs(2021)               0.242181      -0.833706
GDP per Capita(2021)              0.962650       0.034346
Social Support(2021)              1.000000      -0.046770
Freedom(2021)                    -0.046770       1.000000
Life Expectancy(2021)             0.863107       0.428201
Corruption(2021)                 -0.584593      -0.690791

                      Life Expectancy(2021)  Corruption(2021)
Total Layoffs(2021)               -0.221019          0.456462
GDP per Capita(2021)               0.902336         -0.711489
```

```
Social Support(2021)              0.863107        -0.584593
Freedom(2021)                     0.428201        -0.690791
Life Expectancy(2021)             1.000000        -0.905389
Corruption(2021)                 -0.905389         1.000000
                      Total Layoffs(2022)  GDP per Capita(2022)  \
Total Layoffs(2022)              1.000000             0.184229
GDP per Capita(2022)             0.184229             1.000000
Social Support(2022)             0.245647             0.966124
Freedom(2022)                   -0.653289            -0.163263
Life Expectancy(2022)           -0.328448             0.866672
Corruption(2022)                -0.487136             0.697504

                      Social Support(2022)  Freedom(2022)  \
Total Layoffs(2022)               0.245647     -0.653289
GDP per Capita(2022)              0.966124     -0.163263
Social Support(2022)              1.000000     -0.226507
Freedom(2022)                    -0.226507      1.000000
Life Expectancy(2022)             0.809397      0.200977
Corruption(2022)                  0.563576      0.475026

                      Life Expectancy(2022)  Corruption(2022)
Total Layoffs(2022)               -0.328448        -0.487136
GDP per Capita(2022)               0.866672         0.697504
Social Support(2022)               0.809397         0.563576
Freedom(2022)                      0.200977         0.475026
Life Expectancy(2022)              1.000000         0.917590
Corruption(2022)                   0.917590         1.000000
                      Total Layoffs(2023)  GDP per Capita(2023)  \
Total Layoffs(2023)              1.000000             0.209212
GDP per Capita(2023)             0.209212             1.000000
Social Support(2023)             0.296326             0.914891
Freedom(2023)                   -0.892642            -0.486436
Life Expectancy(2023)           -0.310975             0.857550
Corruption(2023)                 0.474146            -0.744542

                      Social Support(2023)  Freedom(2023)  \
Total Layoffs(2023)               0.296326     -0.892642
GDP per Capita(2023)              0.914891     -0.486436
Social Support(2023)              1.000000     -0.599106
Freedom(2023)                    -0.599106      1.000000
Life Expectancy(2023)             0.774587     -0.036943
Corruption(2023)                 -0.547266     -0.203000

                      Life Expectancy(2023)  Corruption(2023)
Total Layoffs(2023)               -0.310975         0.474146
GDP per Capita(2023)               0.857550        -0.744542
Social Support(2023)               0.774587        -0.547266
Freedom(2023)                     -0.036943        -0.203000
Life Expectancy(2023)              1.000000        -0.941625
Corruption(2023)                  -0.941625         1.000000
```

Interpretation:

The provided code segment conducts ANOVA tests and computes correlation matrices for the years 2020, 2021, 2022, and 2023. Initially, the dataset is segmented based on the respective years. The analysis is then focused on selected countries including Canada, Germany, India, Singapore, and the United States. These datasets are merged with the layoff counts data frame, and the "Total Layoffs" column is renamed accordingly for each year. ANOVA tests are performed for each year to evaluate the relationship between total layoffs and various socio-economic variables such as GDP per Capita, Social Support, Freedom, Life Expectancy, and Corruption. The F-value and p-value are reported for each year, indicating the significance of the relationships. Additionally, correlation matrices are computed for each year, revealing the degree of correlation between total layoffs and the socio-economic variables. These analyses provide valuable insights into the potential impact of socio-economic factors on layoff rates across the years 2020 to 2023.

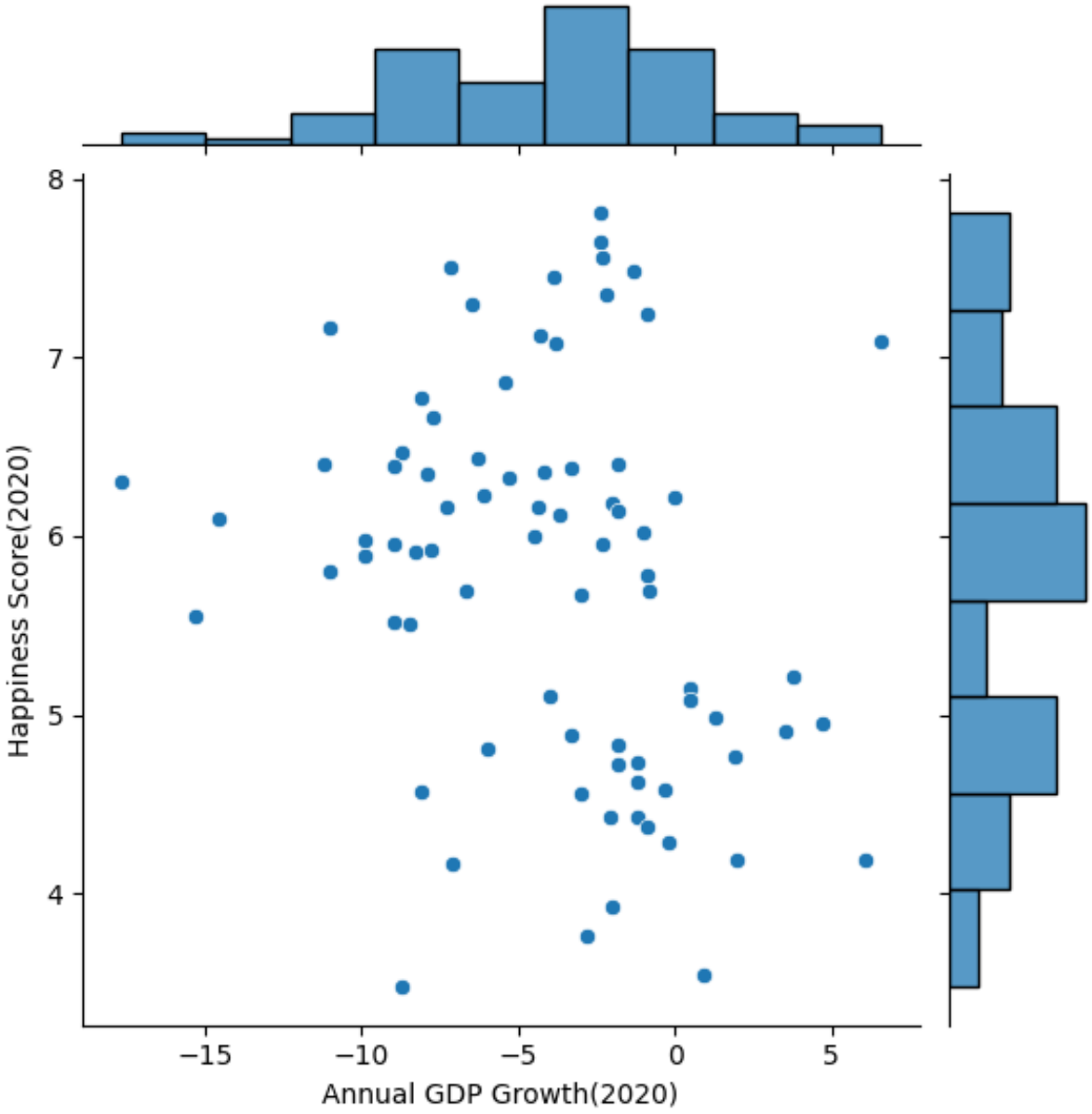
Graphical Analysis

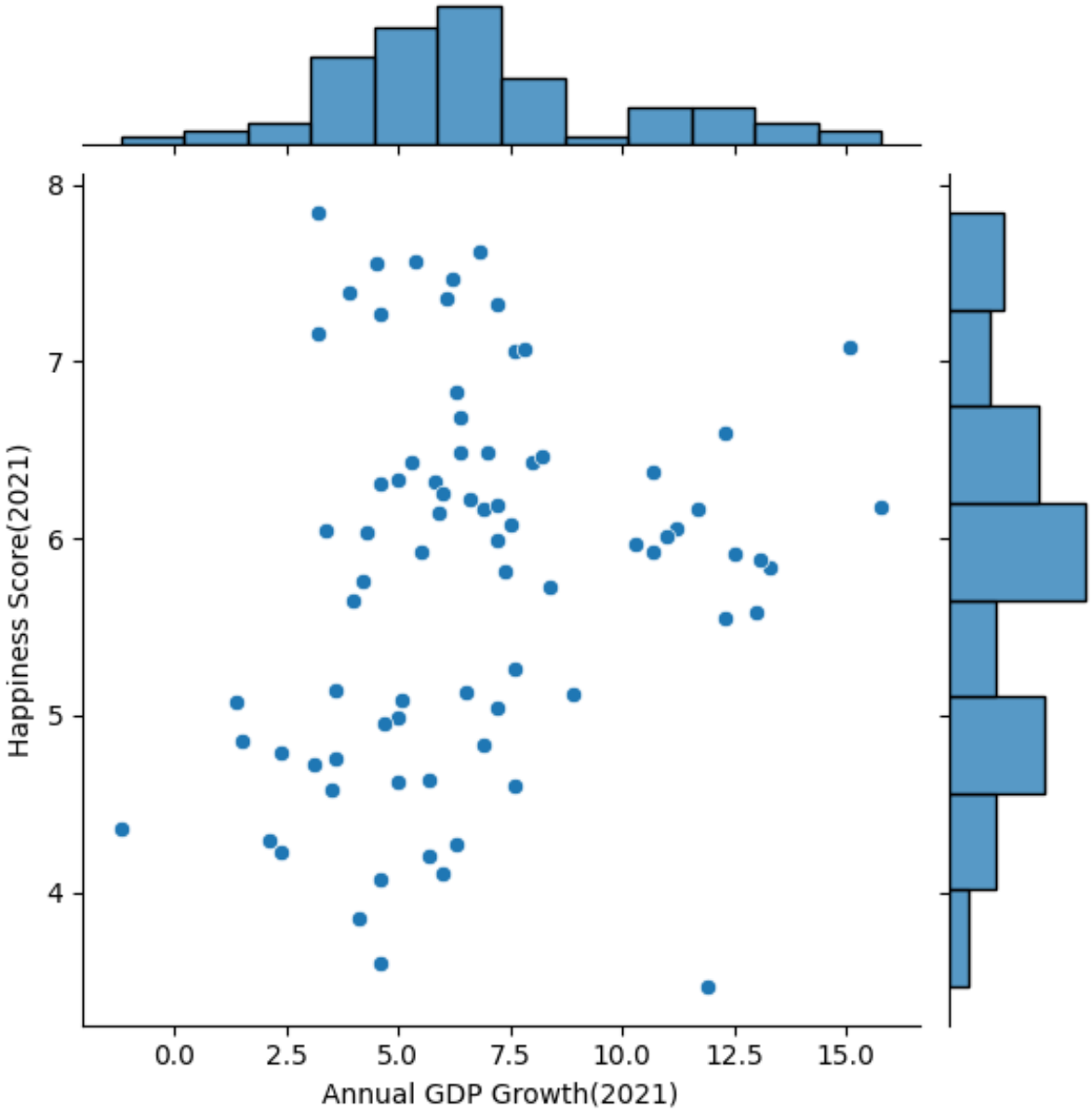
Joint Plots (2020-2023)

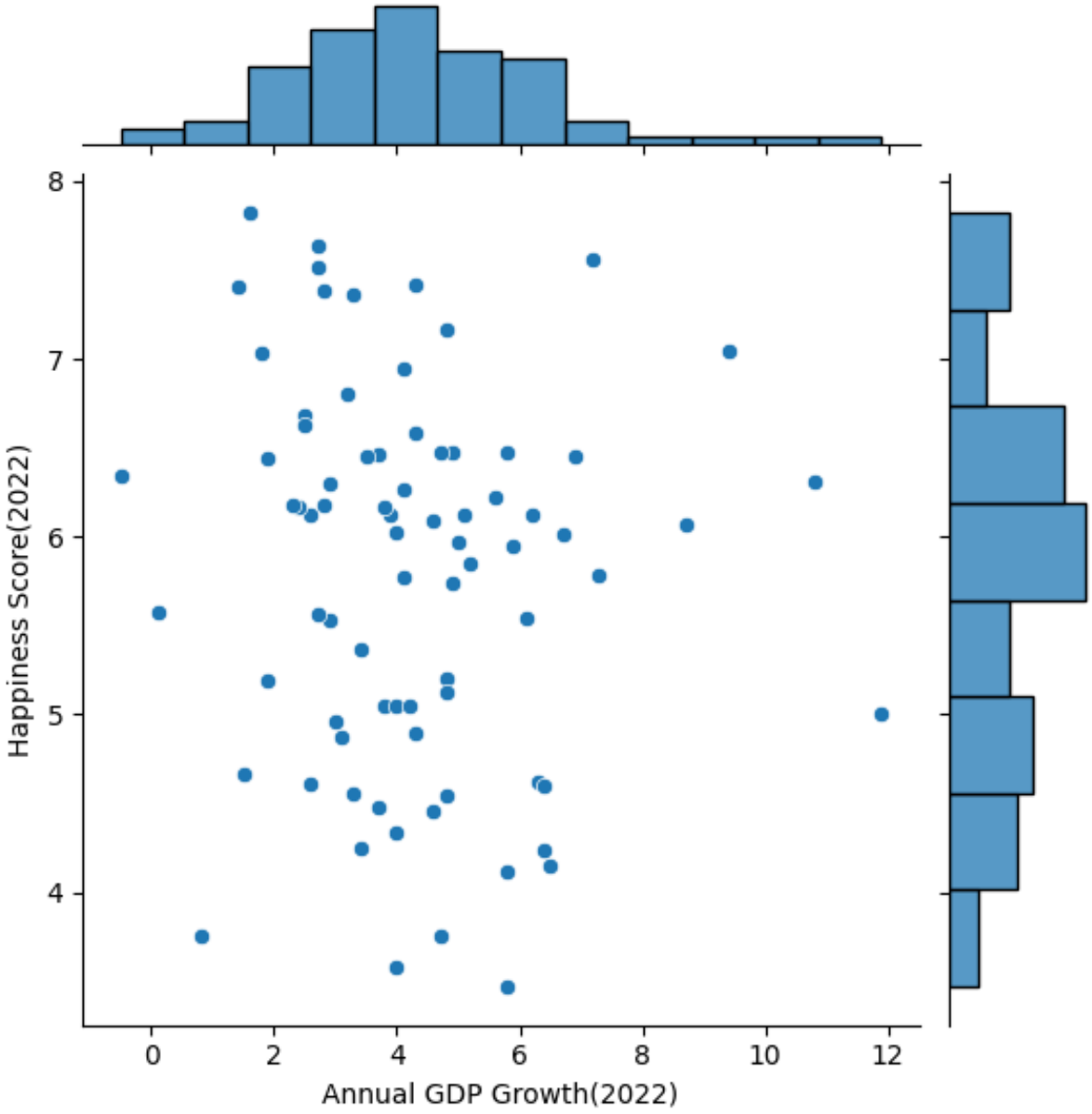
Correlation between Annual GDP Growth and Happiness Scores

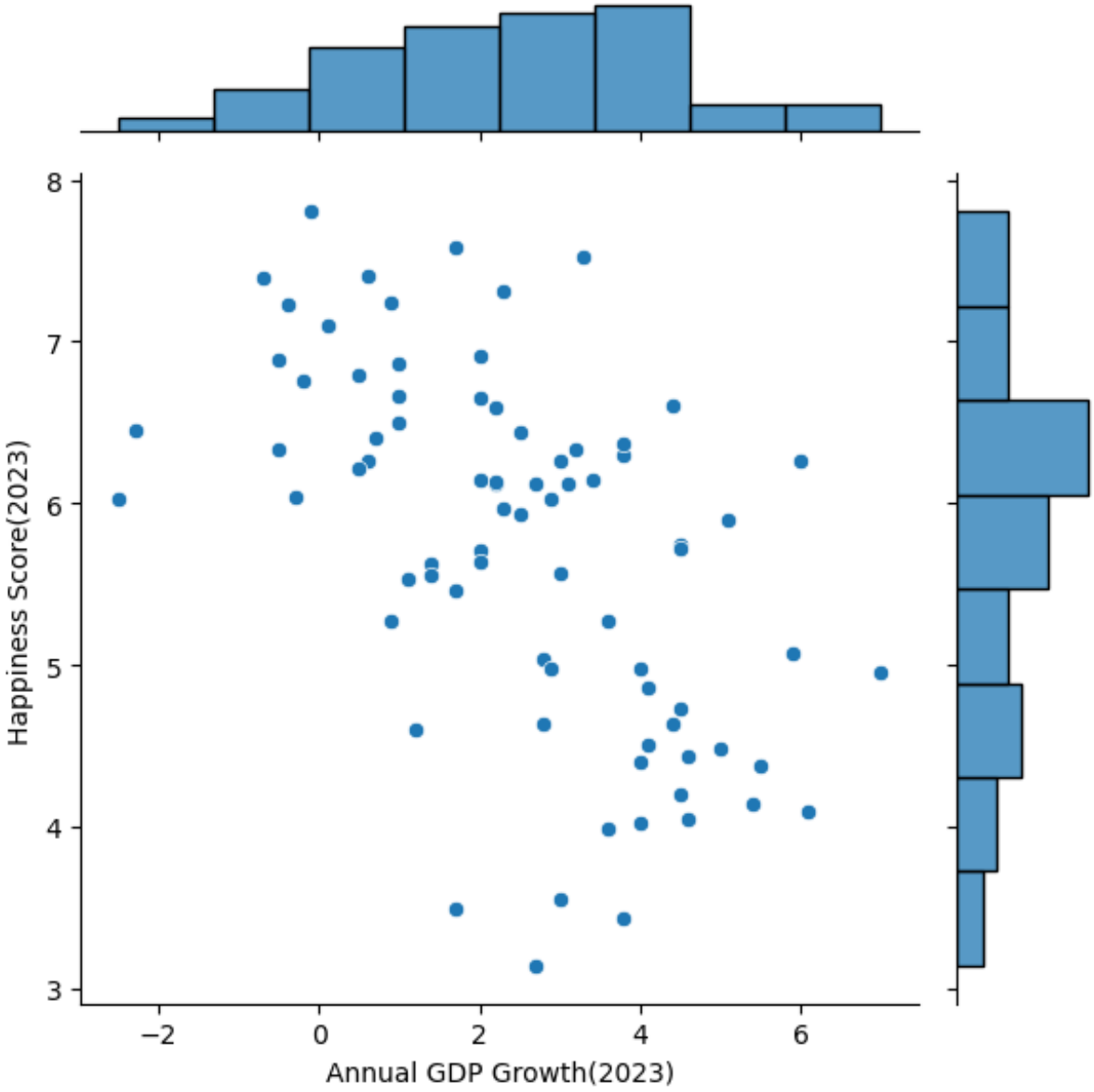
In [878…

```
# Plot the first jointplot
sns.jointplot(x='Annual GDP Growth(2020)', y='Happiness Score(2020)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Plot the second jointplot
sns.jointplot(x='Annual GDP Growth(2021)', y='Happiness Score(2021)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Plot the third jointplot
sns.jointplot(x='Annual GDP Growth(2022)', y='Happiness Score(2022)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Plot the fourth jointplot
sns.jointplot(x='Annual GDP Growth(2023)', y='Happiness Score(2023)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Remove extra whitespace between subplots
plt.tight_layout()

# Show the plots
plt.show()
```


Happiness Score(2020)
8
7
6
5
4
−15
−10
−5
0
5
Annual GDP Growth(2020)


Happiness Score(2021)
8
7
6
5
4
0.0
2.5
5.0
7.5
10.0
12.5
15.0
Annual GDP Growth(2021)


Happiness Score(2022)
Annual GDP Growth(2022)
8
7
6
5
4
0
2
4
6
8
10
12



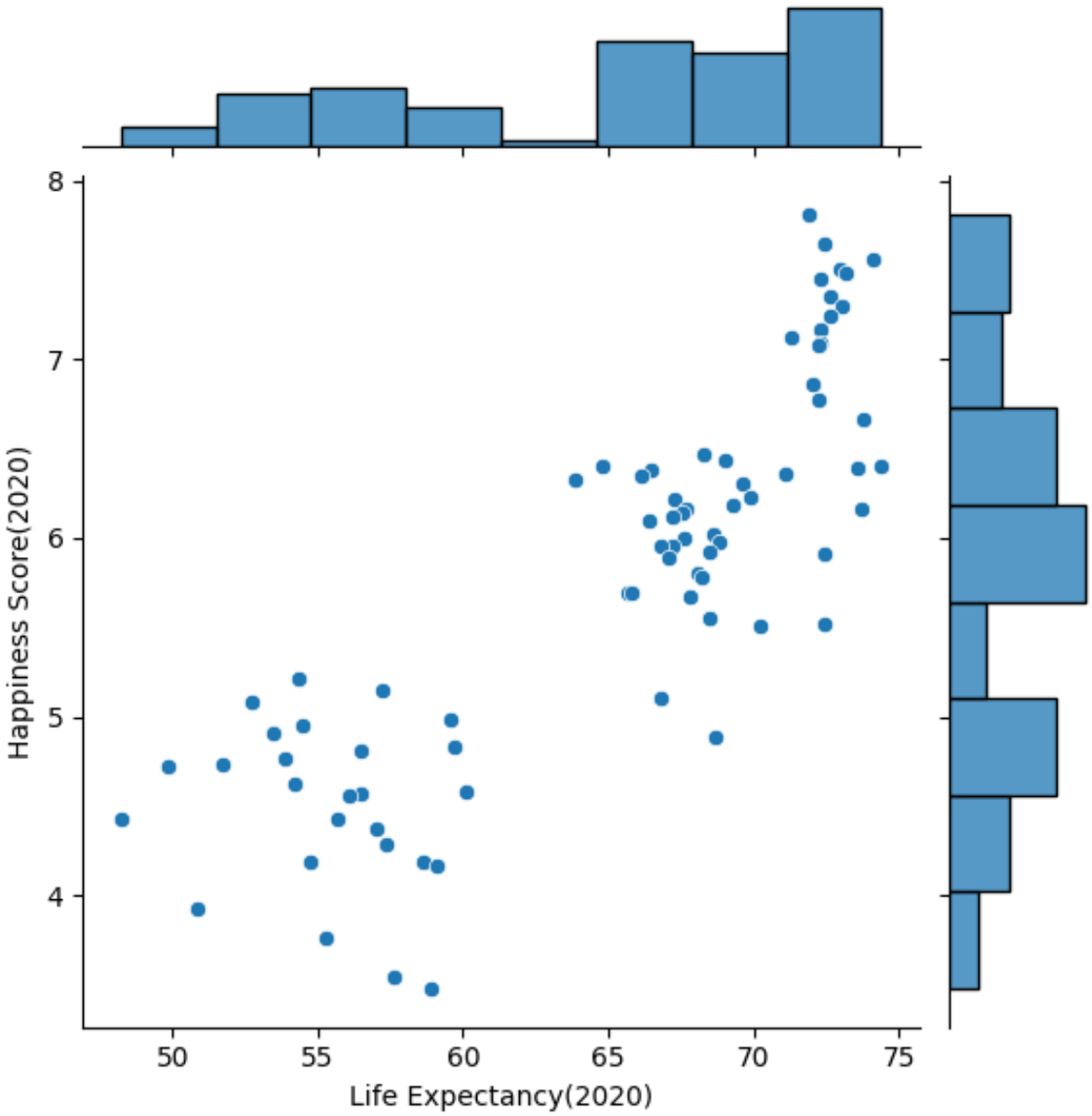
Interpretation:

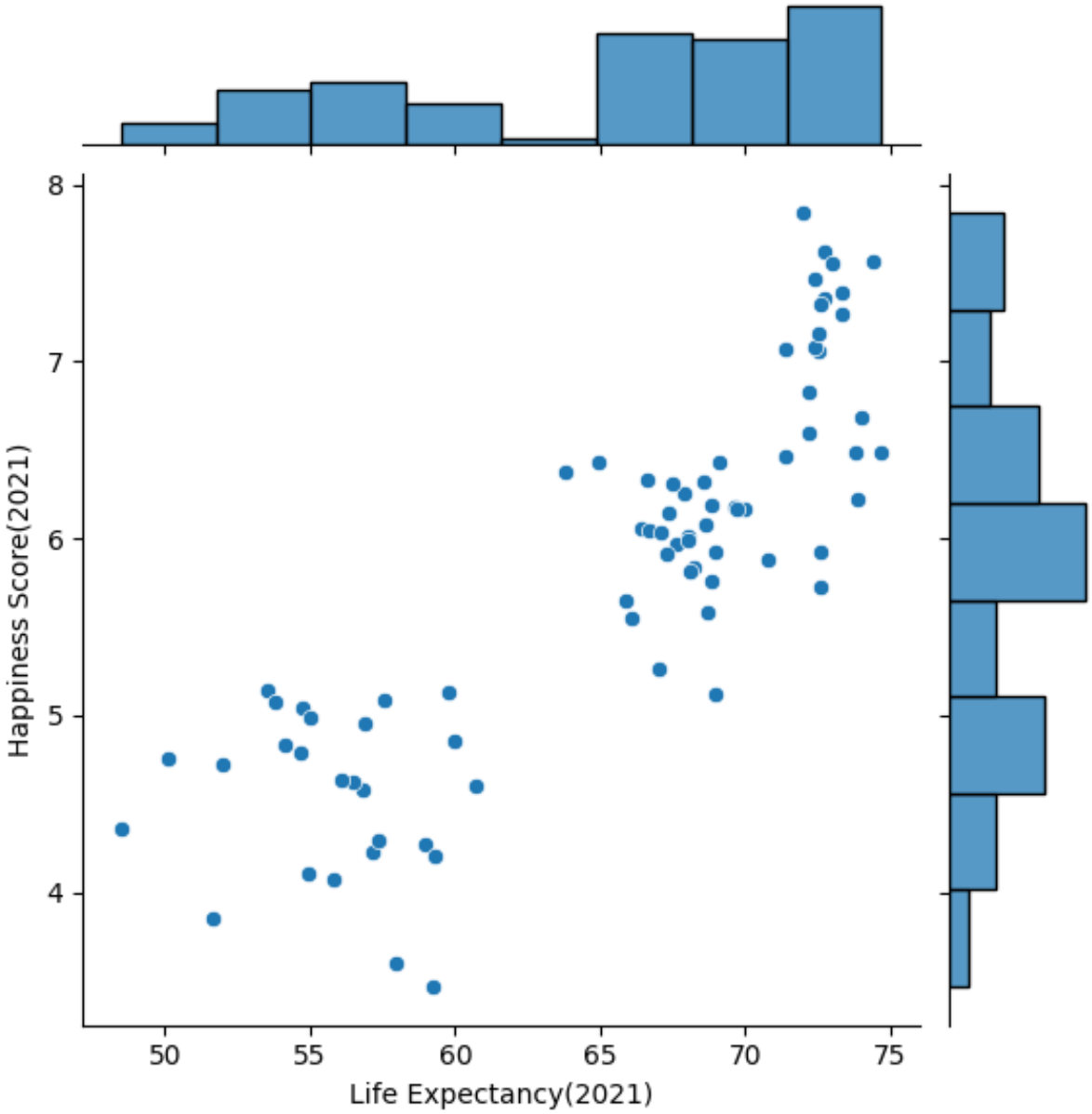
Upon examining the four joint plots depicting the relationship between 'Annual GDP Growth' and 'Happiness Score' across different years, it becomes evident that the data points are scattered randomly with no observable pattern. This lack of discernible trend suggests that there is no apparent linear or nonlinear relationship between GDP growth and happiness score within the dataset for the specified years. The absence of a clear association implies that factors beyond economic indicators play a significant role in determining happiness levels. Social, cultural, and individual-specific variables likely contribute to the variance observed in happiness scores, independent of economic growth. These findings underscore the multifaceted nature of happiness and highlight the need for a holistic understanding of its determinants beyond purely economic considerations.

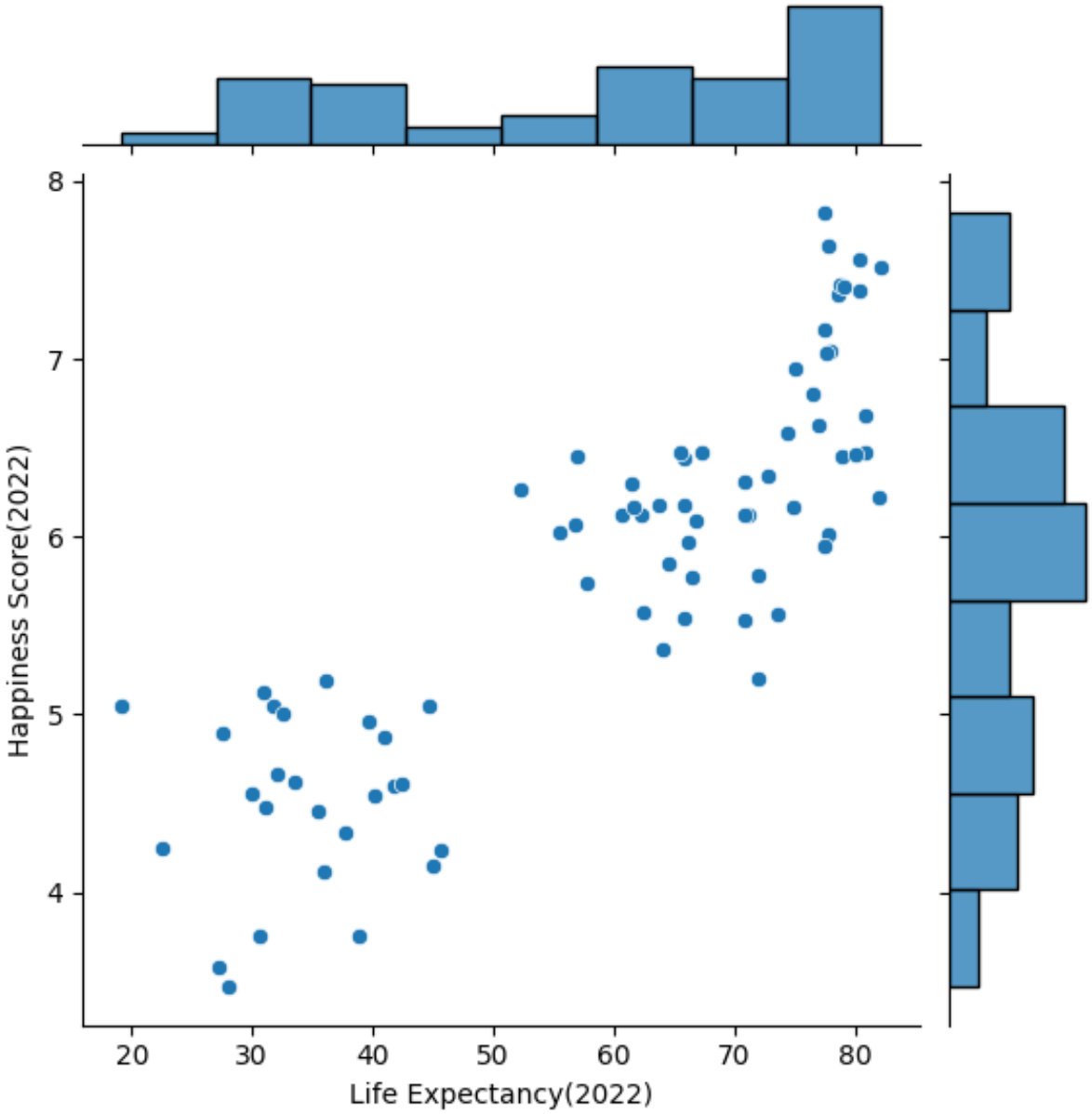
Correlation between Life Expectancy and Happiness Scores

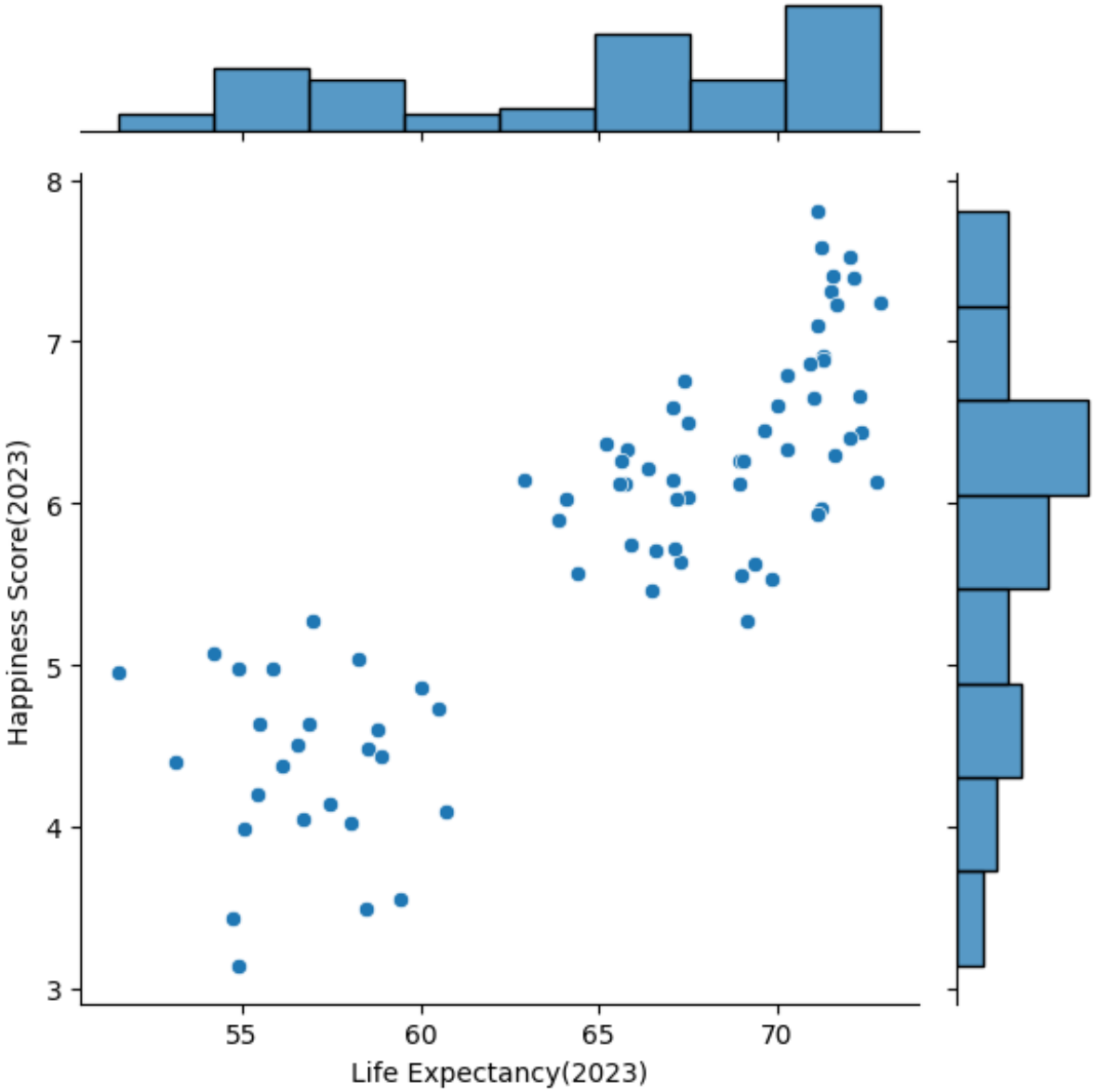
In [879…

```
# Plot the first jointplot
sns.jointplot(x='Life Expectancy(2020)', y='Happiness Score(2020)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Plot the second jointplot
sns.jointplot(x='Life Expectancy(2021)', y='Happiness Score(2021)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Plot the third jointplot
sns.jointplot(x='Life Expectancy(2022)', y='Happiness Score(2022)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Plot the fourth jointplot
sns.jointplot(x='Life Expectancy(2023)', y='Happiness Score(2023)',
              data=merged_happiness_income_gdp_df, kind='scatter')

# Remove extra whitespace between subplots
plt.tight_layout()

# Show the plot
plt.show()
```


Happiness Score(2020)
Life Expectancy(2020)
8
7
6
5
4
50
55
60
65
70
75


Happiness Score(2021)
Life Expectancy(2021)
8
7
6
5
4
50
55
60
65
70
75


Happiness Score(2022)
Life Expectancy(2022)
8
7
6
5
4
20
30
40
50
60
70
80



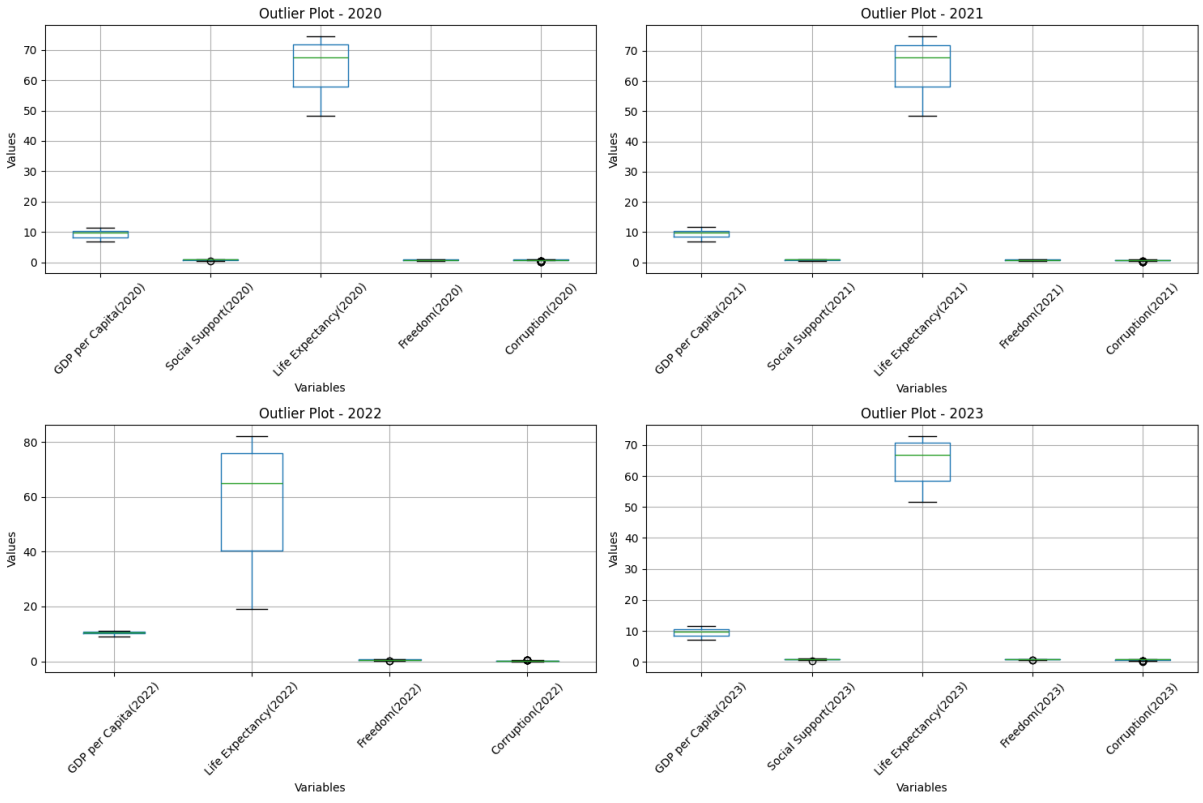
Interpretation:

Upon examining the four joint plots displaying the relationship between 'Life Expectancy' and 'Happiness Score' across different years, a clear positive correlation emerges in all instances. The data points are clustered in a manner that indicates an upward trend, suggesting that as life expectancy increases, so does the happiness score. This consistent pattern underscores the significant influence of life expectancy on happiness levels over time. The observed positive correlation implies that improvements in life expectancy within the dataset are associated with higher reported levels of happiness. While other factors may also contribute to overall well-being, these findings emphasize the crucial role of health-related factors in shaping subjective happiness assessments. Such insights highlight the importance of investing in public health initiatives and healthcare systems to promote overall happiness and well-being within populations.

Outlier Boxplots (2020-2023)

In [880…

```
# Select the variables for the outlier plot
variables_2020 = ['GDP per Capita(2020)', 'Social Support(2020)',
                  'Life Expectancy(2020)', 'Freedom(2020)', 'Corruptio
variables_2021 = ['GDP per Capita(2021)', 'Social Support(2021)',
                  'Life Expectancy(2021)', 'Freedom(2021)', 'Corruptio
variables_2022 = ['GDP per Capita(2022)', 'Social Support(2022)',
                  'Life Expectancy(2022)', 'Freedom(2022)', 'Corruptio
variables_2023 = ['GDP per Capita(2023)', 'Social Support(2023)',
                  'Life Expectancy(2023)', 'Freedom(2023)', 'Corruptio

# Create subplots for all four box plots
fig, axs = plt.subplots(2, 2, figsize=(15, 10))

# Plot boxplots for each year
for i, year in enumerate([variables_2020, variables_2021, variables_20
    row = i // 2
    col = i % 2
    merged_happiness_income_gdp_df[year].boxplot(ax=axs[row, col])
    axs[row, col].set_title(f'Outlier Plot - {years[i]}')
    axs[row, col].set_xlabel('Variables')
    axs[row, col].set_ylabel('Values')
    axs[row, col].tick_params(axis='x', rotation=45)

plt.tight_layout()
plt.show()
```



Interpretation:

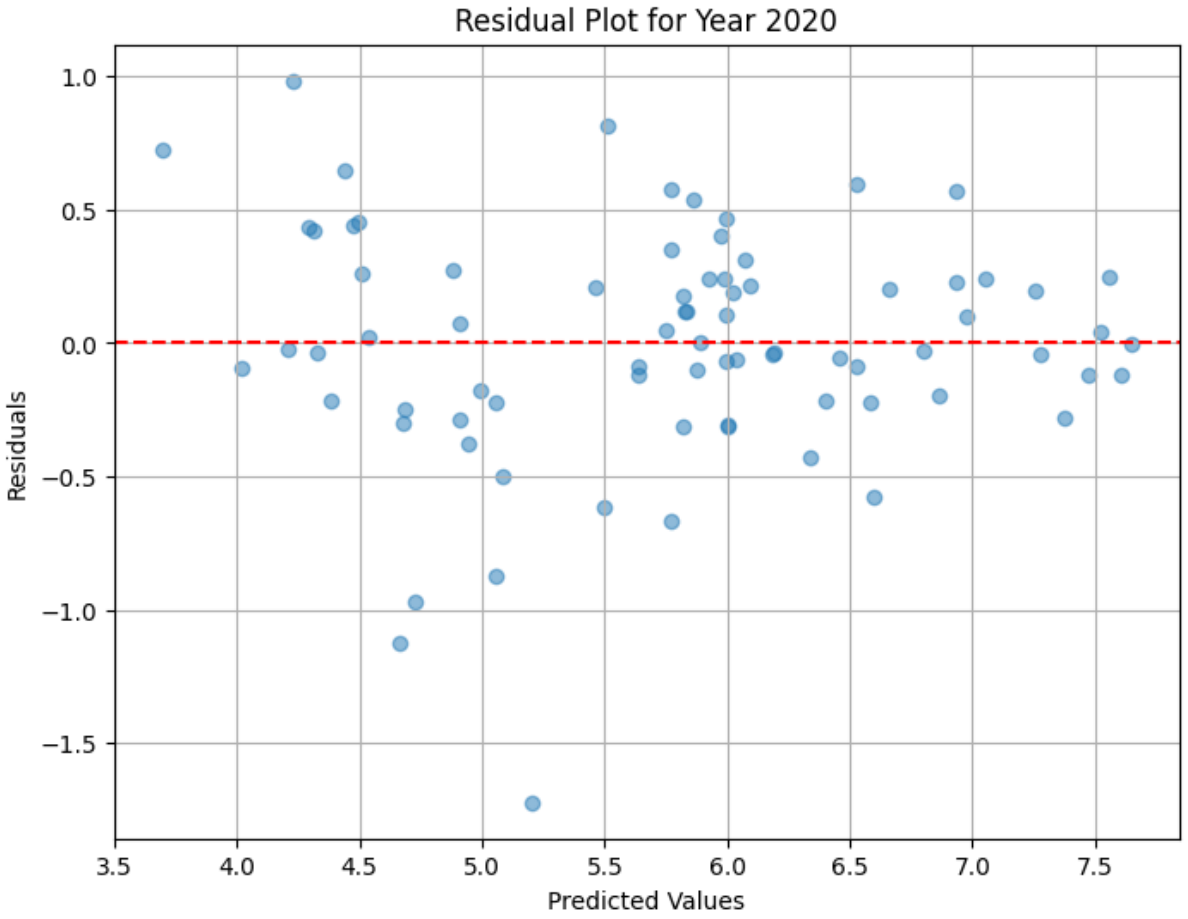
The boxplots generated from the provided code offer valuable insights into the distribution of various variables across four consecutive years. One notable observation is the consistent pattern observed across all four plots, where the variable representing life expectancy exhibits more significant variability compared to other variables. This finding suggests that life expectancy experiences wider fluctuations and possibly more extreme values over the years, as evidenced by the longer whiskers and larger interquartile range (IQR) in its boxplots. Such variability in life expectancy may indicate fluctuating health conditions or healthcare access across different regions or countries over the studied period. Understanding these fluctuations can be crucial for policymakers and public health authorities in identifying areas for targeted interventions aimed at improving overall population health and well-being. Further analysis and exploration into the underlying factors driving these fluctuations in life expectancy could provide valuable insights for designing effective health policies and interventions.

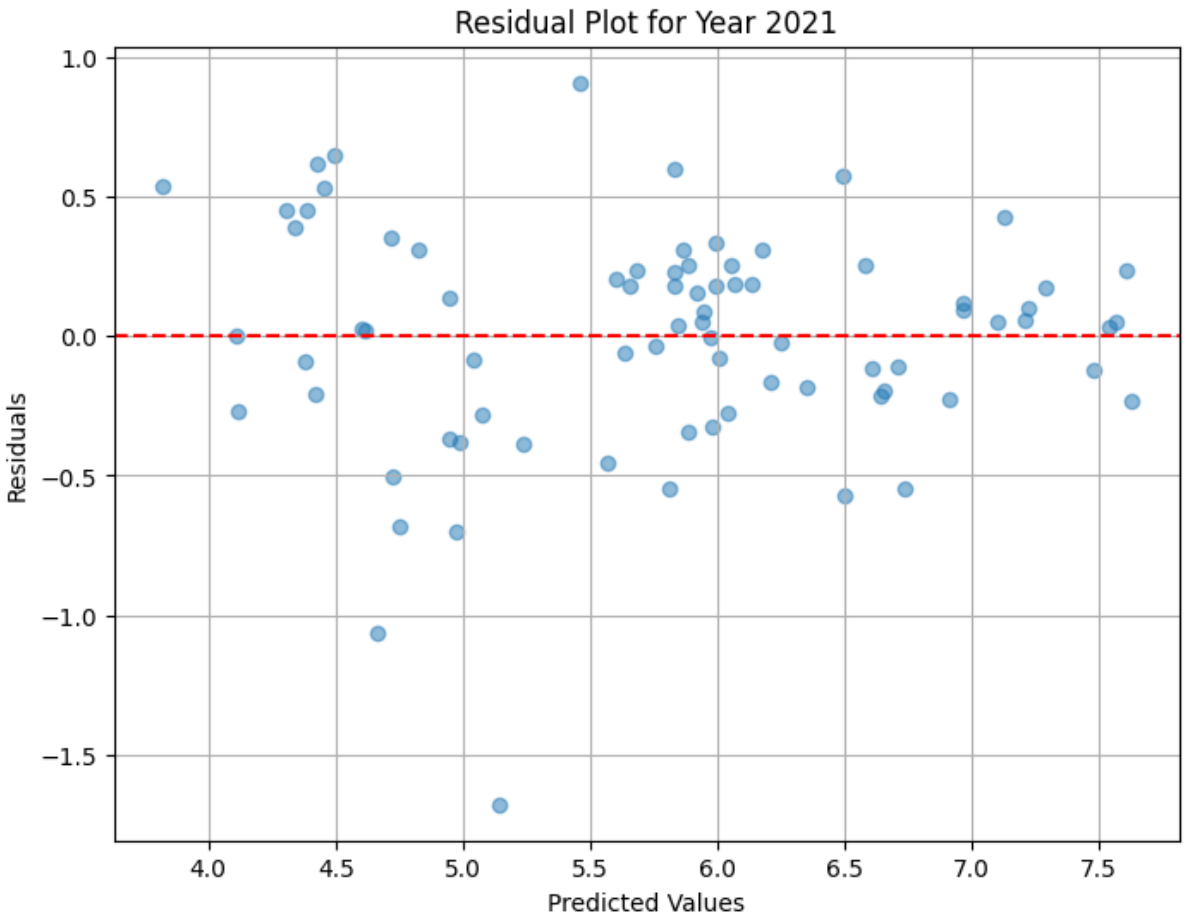
Residual Plots (2020-2023)

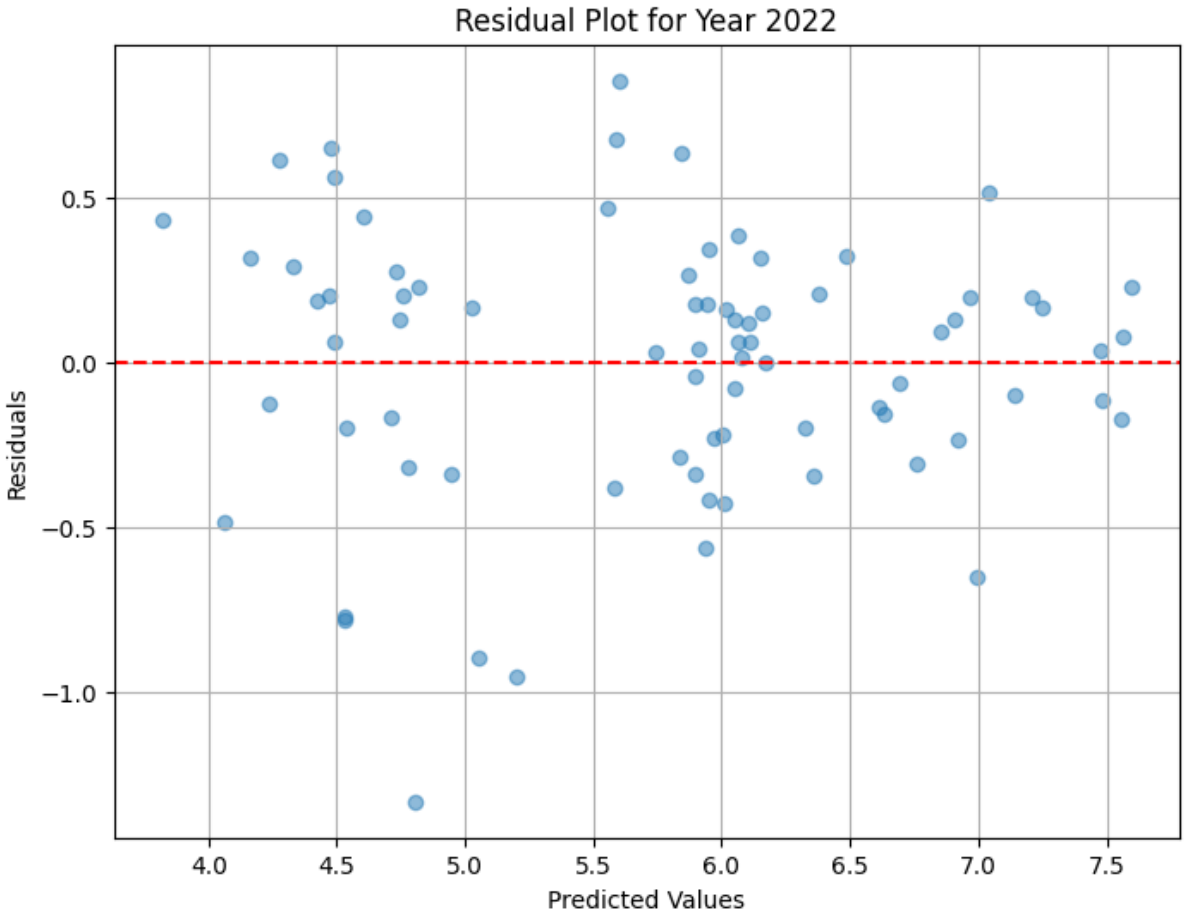
In [881…

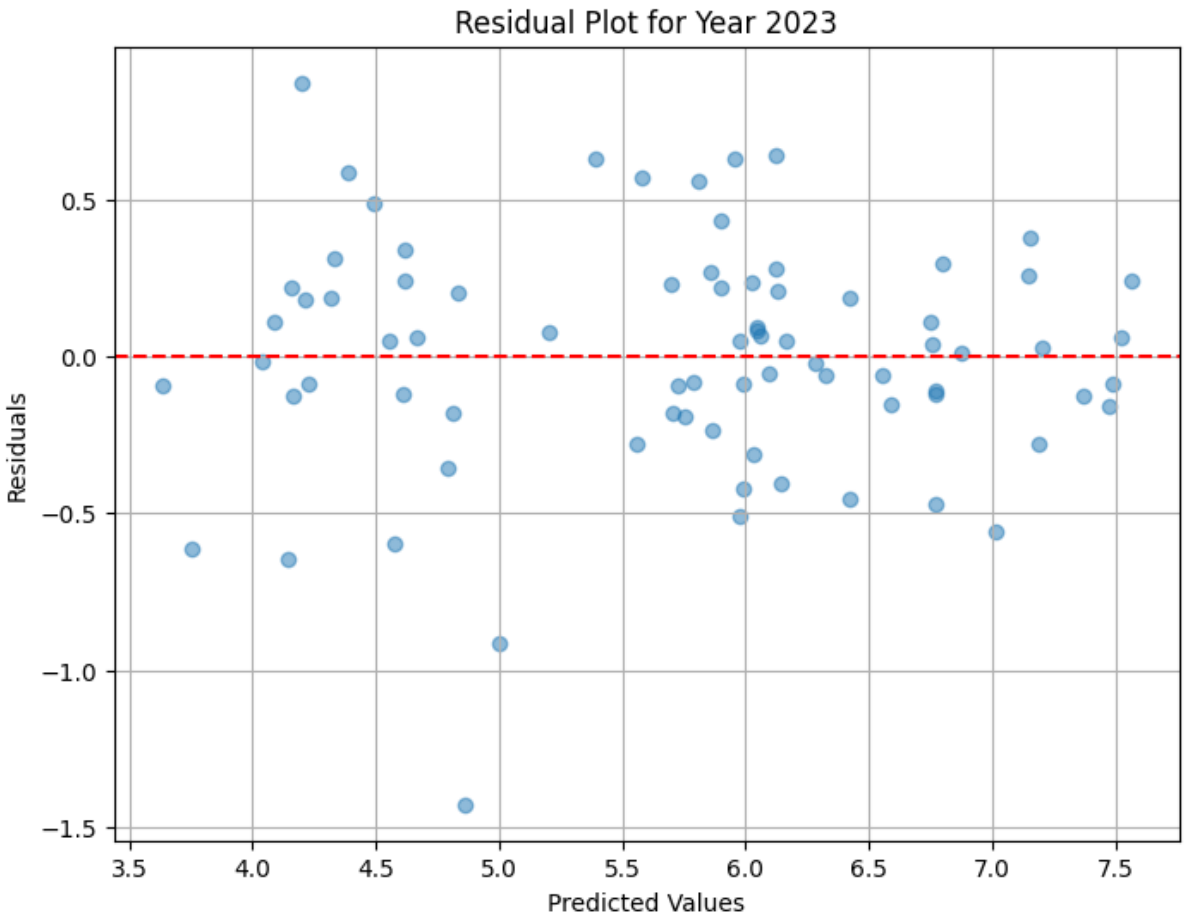
```
# Loop through each year and generate residual plots
for year, independent_vars in zip(range(2020, 2024), [independent_vars
    X = merged_happiness_income_gdp_df[independent_vars].astype(
        float)  # Convert independent variables to float

    # Prepare the dependent variable
    y = merged_happiness_income_gdp_df[f'Happiness Score({year})'].ast
        float)  # Convert dependent variable to float

    # Add a constant term to the independent variables
    X = sm.add_constant(X)

    # Fit the regression model
    model = sm.OLS(y, X)
    results = model.fit()

    # Generate predicted values
    predicted_values = results.predict(X)

    # Calculate residuals
    residuals = y - predicted_values

    # Create residual plot
    plt.figure(figsize=(8, 6))
    plt.scatter(predicted_values, residuals, alpha=0.5)
    # Add a horizontal line at y=0
    plt.axhline(y=0, color='red', linestyle='--')
```

```
    plt.title(f'Residual Plot for Year {year}')
    plt.xlabel('Predicted Values')
    plt.ylabel('Residuals')
    plt.grid(True)
    plt.show()
```




Residual Plot for Year 2021
Residuals
Predicted Values
1.0
0.5
0.0
−0.5
−1.0
−1.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5


Residual Plot for Year 2022
Residuals
Predicted Values
0.5
0.0
−0.5
−1.0
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5



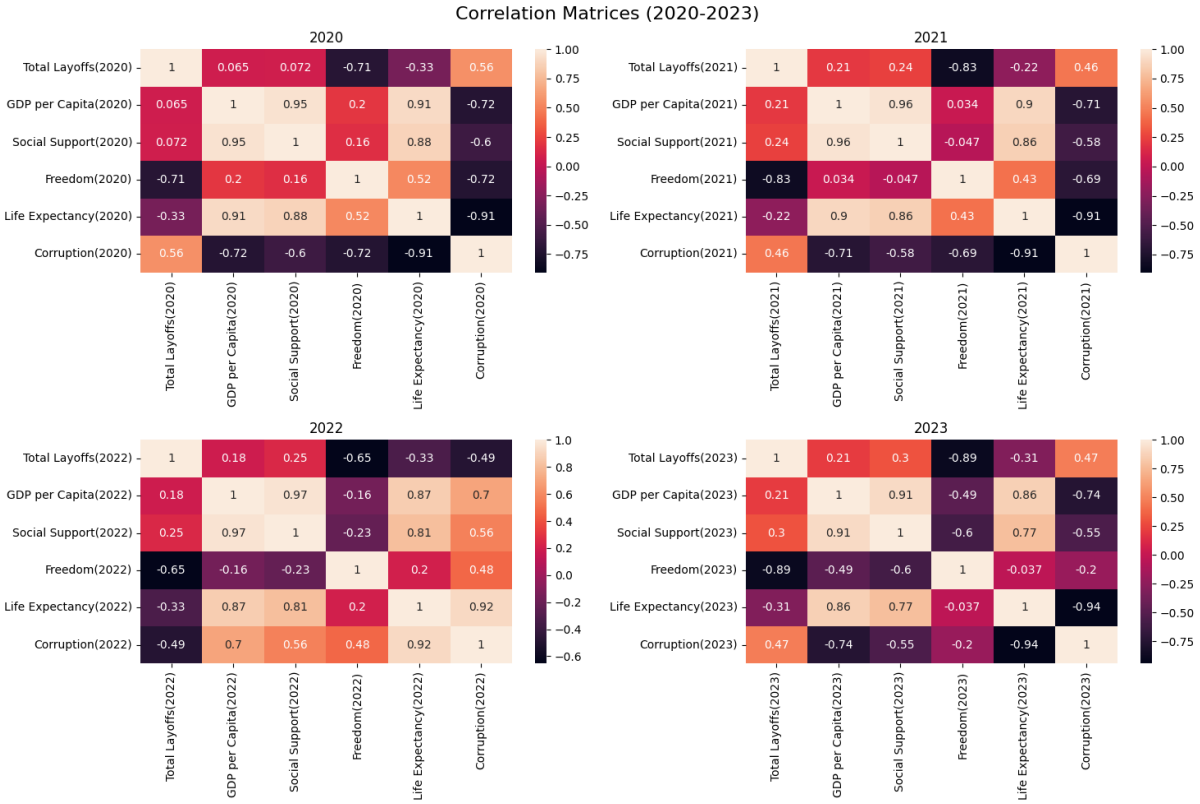
Interpretation: Upon examining the residual plots for the years 2020 to 2023, it is evident that the residuals are randomly scattered around the horizontal line at zero. This random distribution suggests that the linear regression models fitted to the data adequately capture the relationships between the independent variables (such as GDP per Capita, Social Support, Life Expectancy, Freedom, and Corruption) and the dependent variable (Happiness Score). The absence of any discernible pattern in the residuals indicates that the models are appropriately capturing the variation in the data, without any systematic bias or unexplained trends. Overall, the random scattering of residuals around the zero line signifies that the regression models provide a satisfactory fit to the data across the four years analyzed.

Heatmaps (2020-2023)

In [882…

```
# Visualize the four correlation matrices with a different color schem
fig, axes = plt.subplots(2, 2, figsize=(15, 10))
fig.suptitle('Correlation Matrices (2020-2023)', fontsize=16)

# 2020
sns.heatmap(correlation_matrix_2020, annot=True, cmap='rocket', ax=axe
axes[0, 0].set_title('2020')
```

```
# 2021
sns.heatmap(correlation_matrix_2021, annot=True, cmap='rocket', ax=axe
axes[0, 1].set_title('2021')

# 2022
sns.heatmap(correlation_matrix_2022, annot=True, cmap='rocket', ax=axe
axes[1, 0].set_title('2022')

# 2023
sns.heatmap(correlation_matrix_2023, annot=True, cmap='rocket', ax=axe
axes[1, 1].set_title('2023')

# Adjust the layout
plt.tight_layout()
plt.show()
```



Interpretation:

The visualization and analysis of correlation matrices from 2020 to 2023 provide valuable insights into the relationships between socio-economic factors and layoffs across these years. In 2020, we observe a moderate positive correlation between Total Layoffs and Corruption, indicating that countries with higher levels of corruption experienced more layoffs. However, there is a strong negative correlation between Total Layoffs and Freedom, suggesting that nations with greater freedom tend to have fewer layoffs. The following years show varying patterns: in 2021, there's a strong negative correlation between Total Layoffs and

Freedom, while in 2022, there's a moderate negative correlation. Interestingly, by 2023, this negative correlation weakens significantly, indicating a potential shift in the dynamics between layoffs and freedom. Additionally, consistent positive correlations are observed between GDP per Capita and Social Support across all years, suggesting a strong socio-economic foundation may mitigate the impact of layoffs. Overall, these findings highlight the complex interplay between socio-economic variables and layoffs, underscoring the importance of understanding these relationships for effective policy-making and economic planning.

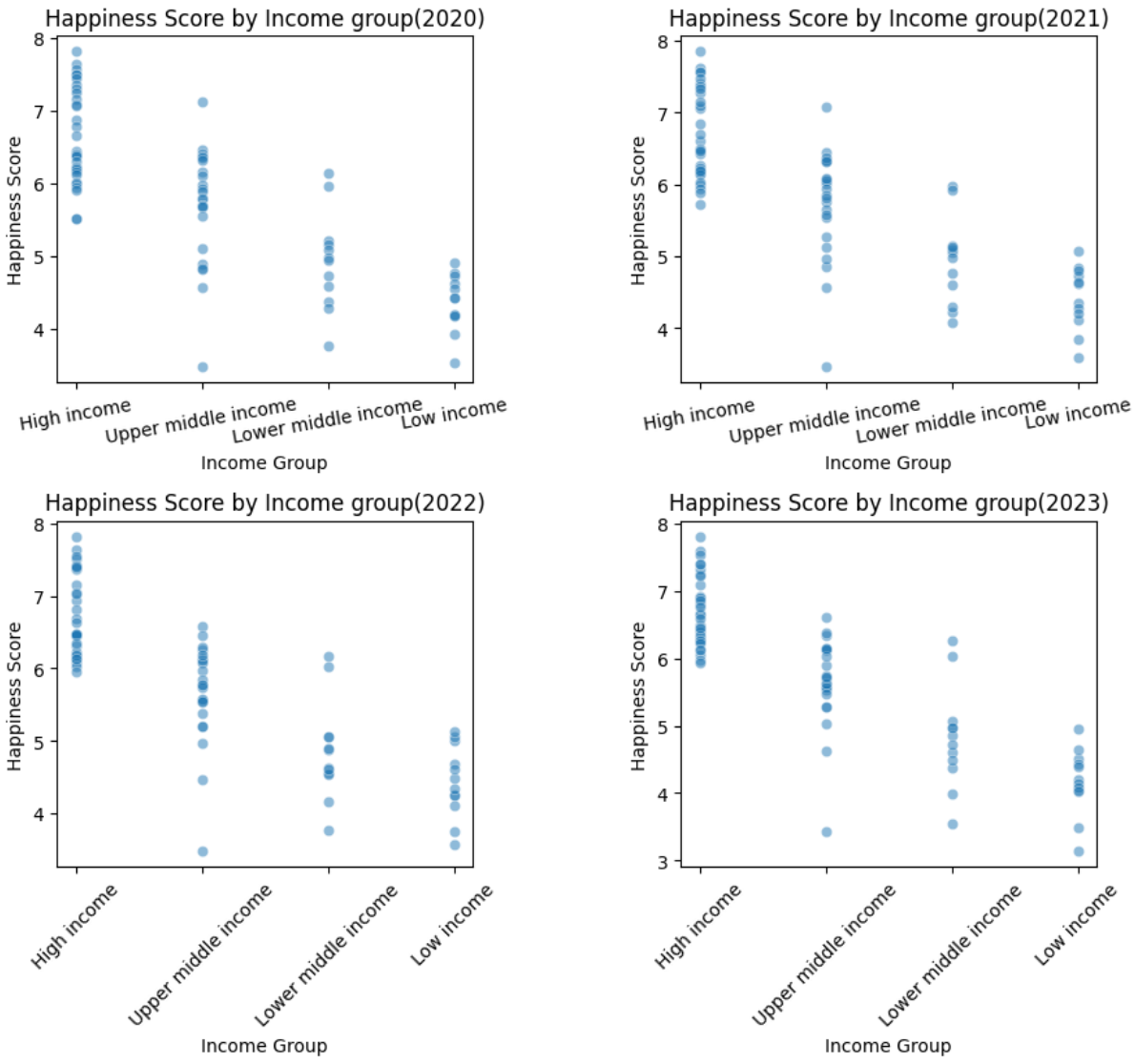
Comparative Analysis

Scatter Plots (2020-2023)

In [883…

```
# Create a figure and four subplots
fig, axes = plt.subplots(2, 2, figsize=(10, 8))

sns.scatterplot(x="Income Group", y="Happiness Score(2020)",
                sizes=(10, 40), alpha=.5,
                data=merged_happiness_income_gdp_df, ax=axes[0, 0])
axes[0, 0].set_title('Happiness Score by Income group(2020)')
axes[0, 0].set_xlabel('Income Group')
axes[0, 0].set_ylabel('Happiness Score')

# Plot the second subplot
sns.scatterplot(x="Income Group", y="Happiness Score(2021)",
                sizes=(10, 40), alpha=.5,
                data=merged_happiness_income_gdp_df, ax=axes[0, 1])
axes[0, 1].set_title('Happiness Score by Income group(2021)')
axes[0, 1].set_xlabel('Income Group')
axes[0, 1].set_ylabel('Happiness Score')

# Plot the third subplot
sns.scatterplot(x="Income Group", y="Happiness Score(2022)",
                sizes=(10, 40), alpha=.5,
                data=merged_happiness_income_gdp_df, ax=axes[1, 0])
axes[1, 0].set_title('Happiness Score by Income group(2022)')
axes[1, 0].set_xlabel('Income Group')
axes[1, 0].set_ylabel('Happiness Score')

# Plot the fourth subplot
sns.scatterplot(x="Income Group", y="Happiness Score(2023)",
                sizes=(10, 40), alpha=.5,
                data=merged_happiness_income_gdp_df, ax=axes[1, 1])
axes[1, 1].set_title('Happiness Score by Income group(2023)')
axes[1, 1].set_xlabel('Income Group')
axes[1, 1].set_ylabel('Happiness Score')
```

```
# Rotate the x-axis labels for better readability
axes[0, 0].tick_params(axis='x', rotation=10)
axes[0, 1].tick_params(axis='x', rotation=10)
axes[1, 0].tick_params(axis='x', rotation=45)
axes[1, 1].tick_params(axis='x', rotation=45)

# Adjust spacing between subplots
plt.subplots_adjust(left=0.1, right=0.9, top=0.9,
                    bottom=0.1, wspace=0.5, hspace=0.4)

# Show the plots
plt.show()
```



Interpretation:

The provided code generates a series of scatter plots illustrating the relationship between income groups and happiness scores across four consecutive years. Across all four scatter plots, a clear pattern emerges: countries categorized as high-income consistently exhibit the highest happiness scores, followed by

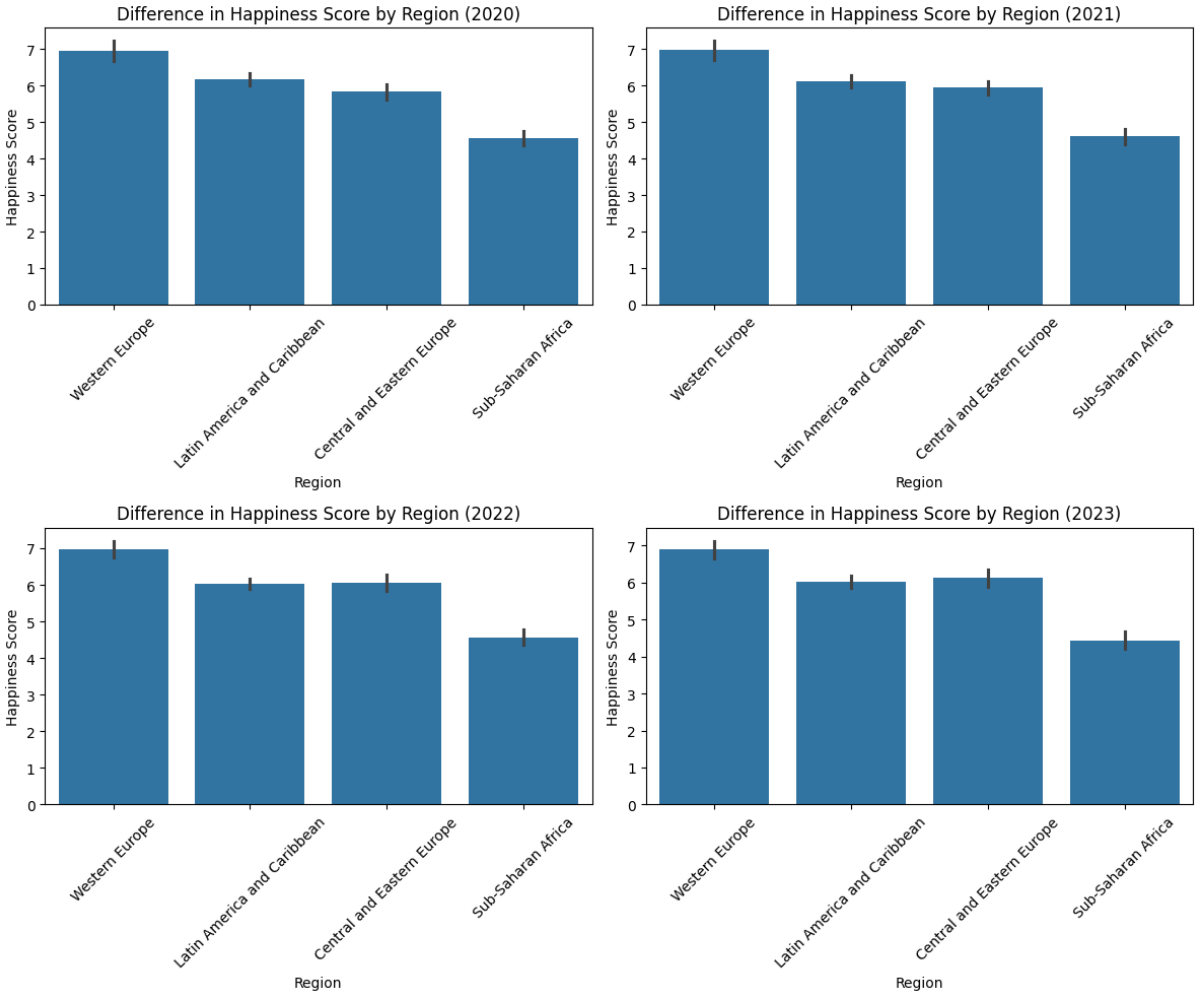
upper-middle-income, lower-middle-income, and low-income countries. This consistent hierarchy suggests a robust correlation between income levels and happiness scores over the studied period. Interestingly, the pattern remains consistent despite potential fluctuations in global economic conditions or other external factors. Such findings underscore the significance of economic well-being in shaping overall happiness levels within different socio-economic contexts. Understanding these dynamics can inform policymakers and international organizations in designing targeted interventions aimed at improving overall happiness and well-being, particularly in low-income regions where happiness scores tend to be lower.

Bar Plots (2020-2023)

In [884…

```
# Create a figure and four subplots
fig, axes = plt.subplots(2, 2, figsize=(12, 10))

# Plot the first subplot
sns.barplot(x='Region', y='Happiness Score(2020)',
            data=merged_happiness_income_gdp_df, ax=axes[0, 0])
axes[0, 0].set_title('Difference in Happiness Score by Region (2020)')
axes[0, 0].set_xlabel('Region')
axes[0, 0].set_ylabel('Happiness Score')

# Plot the second subplot
sns.barplot(x='Region', y='Happiness Score(2021)',
            data=merged_happiness_income_gdp_df, ax=axes[0, 1])
axes[0, 1].set_title('Difference in Happiness Score by Region (2021)')
axes[0, 1].set_xlabel('Region')
axes[0, 1].set_ylabel('Happiness Score')

# Plot the third subplot
sns.barplot(x='Region', y='Happiness Score(2022)',
            data=merged_happiness_income_gdp_df, ax=axes[1, 0])
axes[1, 0].set_title('Difference in Happiness Score by Region (2022)')
axes[1, 0].set_xlabel('Region')
axes[1, 0].set_ylabel('Happiness Score')

# Plot the fourth subplot
sns.barplot(x='Region', y='Happiness Score(2023)',
            data=merged_happiness_income_gdp_df, ax=axes[1, 1])
axes[1, 1].set_title('Difference in Happiness Score by Region (2023)')
axes[1, 1].set_xlabel('Region')
axes[1, 1].set_ylabel('Happiness Score')

# Rotate the x-axis labels for better readability
for ax in axes.flat:
    ax.tick_params(axis='x', rotation=45)
```

```
# Adjust spacing between subplots
plt.tight_layout()

# Show the plots
plt.show()
```



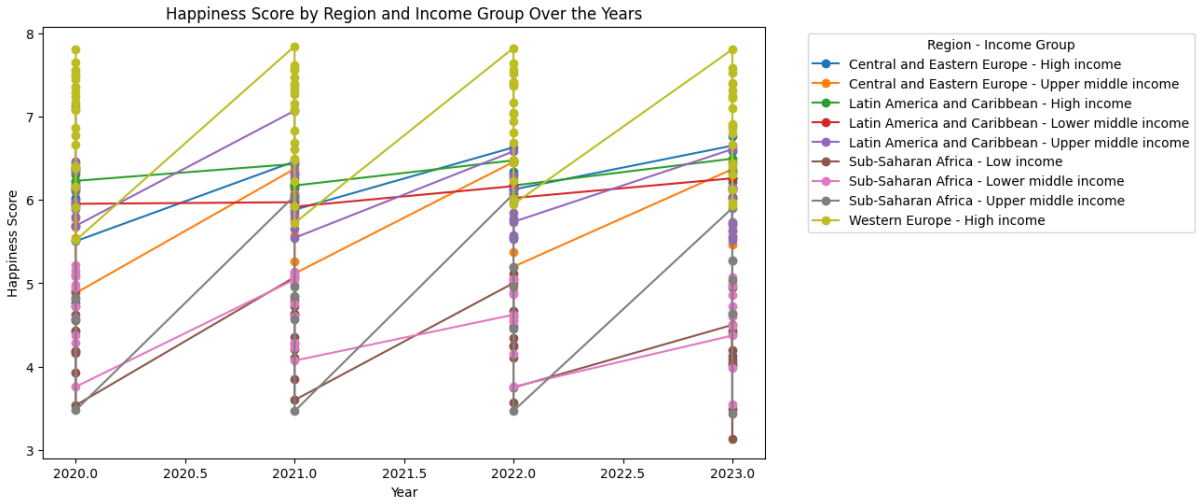
Interpretation:

The analysis of happiness scores by region across the years 2020 to 2023 reveals consistent patterns in happiness levels. Western Europe consistently exhibits the highest happiness scores, followed by Latin America and the Caribbean, Central and Eastern Europe, and Sub-Saharan Africa, in descending order. Notably, Latin America and the Caribbean, and Central and Eastern Europe, appear almost equal in happiness scores for the years 2022 and 2023. Despite this convergence, the overarching hierarchy in happiness levels remains unchanged across all four plots, emphasizing enduring disparities in well-being across different regions. These findings highlight the complexity of factors influencing happiness levels and suggest the presence of nuanced dynamics within and between regions over time.

Line Plots (2020-2023)

In [885…

```
# Reshape the DataFrame
merged_happiness_income_gdp_df_melted = pd.melt(merged_happiness_incom
                                                value_vars=[
                                                    'Happiness Score(2
                                                var_name='Year', value

# Extract the year from the 'Year' column and convert it to an integer
merged_happiness_income_gdp_df_melted['Year'] = merged_happiness_incom
    '(\d+)').astype(int)

# Create a line plot
fig, ax = plt.subplots(figsize=(10, 6))

# Loop through each region and income group and plot the data
for (region, income_group), group in merged_happiness_income_gdp_df_me
    group.plot(x='Year', y='Happiness Score', ax=ax,
               label=f"{region} - {income_group}", marker='o')

# Set the title, x-axis label, and y-axis label
plt.title('Happiness Score by Region and Income Group Over the Years')
plt.xlabel('Year')
plt.ylabel('Happiness Score')
plt.legend(title='Region - Income Group',
           bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



Interpretation:

The line plot effectively captures and illustrates the relationship between regions and income groups in terms of their respective happiness scores over the years. Each line on the plot represents a specific combination of region and income group, allowing for a nuanced examination of how these factors interact to influence happiness levels. Upon observation, certain trends emerge. Generally,

regions with higher income groups tend to exhibit higher happiness scores across the years, while regions with lower income groups show comparatively lower scores. However, there are exceptions and variations within regions, indicating that factors beyond income, such as social and cultural dynamics, also play significant roles in determining happiness levels. Additionally, over the years, some regions and income groups may experience fluctuations in happiness scores, reflecting changes in socio-economic conditions, policy interventions, or other external factors. Overall, the visualization highlights the complex interplay between region, income group, and happiness, underscoring the multifaceted nature of well-being and the need for comprehensive approaches to address global happiness disparities.

Multivariate Analysis

Cross-Tabulation (2020-2023)

In [886…

```
# Create a cross-tabulation to analyze the relationship between 'Regio
cross_tab = pd.crosstab(
    # Specify 'Region' as the rows (index)
    index=merged_happiness_income_gdp_df['Region'],
    # Specify 'Income Group' as the columns
    columns=merged_happiness_income_gdp_df['Income Group']
)
# Display the cross-tabulation results
cross_tab
```

Out[886…

Income Group	High income	Low income	Lower middle income	Upper middle income
Region				
Central and Eastern Europe	8	0	0	6
Latin America and Caribbean	3	0	2	12
Sub-Saharan Africa	0	12	10	5
Western Europe	20	0	0	0

Interpretation:

Upon conducting a cross-tabulation analysis, it becomes evident that the distribution of income groups varies significantly across different regions. In Central and Eastern Europe, the majority of countries fall into the 'High income'

and 'Upper middle income' categories, reflecting a relatively prosperous economic landscape. Conversely, in Sub-Saharan Africa, the predominant income groups are 'Low income' and 'Lower middle income,' indicating pervasive economic challenges within the region. Interestingly, Western Europe stands out as predominantly comprising 'High income' countries, suggesting a high level of economic prosperity across the region. Latin America and the Caribbean exhibit a more diverse distribution of income groups, with representation across 'High income,' 'Lower middle income,' and 'Upper middle income' categories. These findings underscore the intricate interplay between regional dynamics and income distribution, highlighting disparities that may inform targeted policy interventions aimed at promoting economic development and reducing inequality.

Pivot Table (2020-2023)

In [887…

```
# Create pivot tables for each year
pivot_table_2020 = pd.pivot_table(merged_happiness_income_gdp_df, inde
                                  columns='Income Group', values='Happ
pivot_table_2021 = pd.pivot_table(merged_happiness_income_gdp_df, inde
                                  columns='Income Group', values='Happ
pivot_table_2022 = pd.pivot_table(merged_happiness_income_gdp_df, inde
                                  columns='Income Group', values='Happ
pivot_table_2023 = pd.pivot_table(merged_happiness_income_gdp_df, inde
                                  columns='Income Group', values='Happ

# Combine the pivot tables into one
combined_pivot_table = pd.concat([pivot_table_2020, pivot_table_2021,
                                  pivot_table_2023], axis=1, keys=['202

# Display the combined pivot table
combined_pivot_table
```

Out [887...

	2020						
Income Group	**High income**	**Low income**	**Lower middle income**	**Upper middle income**	**High income**	**Low income**	**Lower middle income**
Region							
Central and Eastern Europe	6.045725	0.0000	0.00000	5.551300	6.139625	0.000000	0.0000
Latin America and Caribbean	6.324467	0.0000	6.04515	6.153367	6.261000	0.000000	5.9455
Sub-Saharan Africa	0.000000	4.3706	4.71087	4.758940	0.000000	4.423333	4.7346
Western Europe	6.967405	0.0000	0.00000	0.000000	6.983850	0.000000	0.0000

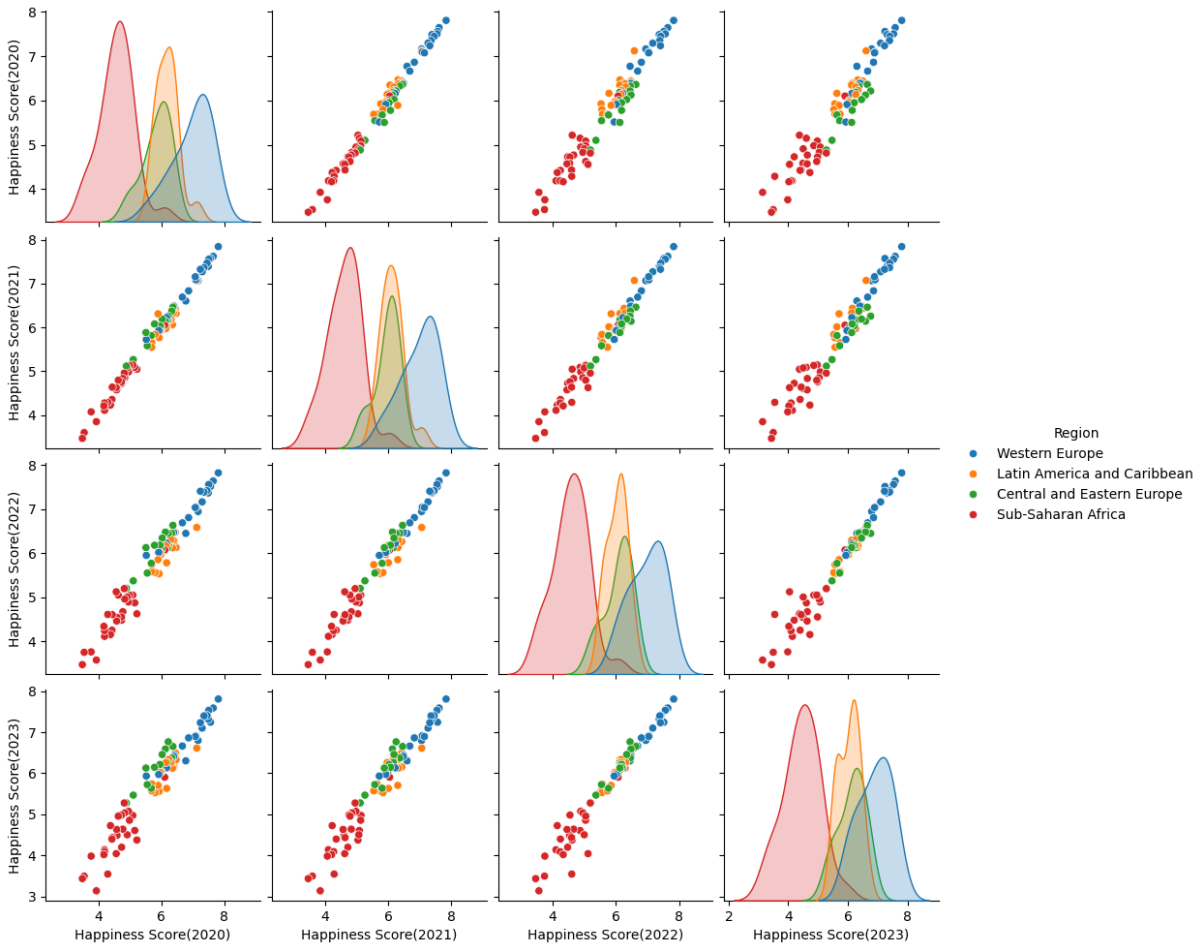
Interpretation:

Upon examining the combined pivot table encompassing mean happiness scores across income groups and regions for the years 2020 through 2023, intriguing insights emerge regarding the dynamics of happiness within diverse socio-economic contexts. Notably, within Central and Eastern Europe, countries classified as 'High income' and 'Upper middle income' consistently demonstrate elevated happiness scores, suggesting a positive correlation between economic prosperity and subjective well-being. In Latin America and the Caribbean, a remarkable stability in happiness scores is observed across income groups, indicative of a resilient societal fabric that transcends economic fluctuations. Conversely, Sub-Saharan Africa presents a sobering narrative, with uniformly lower happiness scores across income strata, underscoring persistent challenges to well-being irrespective of economic standing. Noteworthy is Western Europe's consistently high happiness scores, particularly evident among 'High income' nations, reflecting the region's robust socio-economic infrastructure conducive to overall life satisfaction. These findings collectively underscore the complex interplay between income, regional dynamics, and happiness, emphasizing the need for nuanced policy interventions to foster well-being across diverse global landscapes.

Pair Plot (2020-2023)

In [888…
```
# Create a pairplot to visualize the relationship between happiness sc
sns.pairplot(data=merged_happiness_income_gdp_df,
             hue="Region",
             vars=['Happiness Score(2020)', 'Happiness Score(2021)', '
```

Out[888… <seaborn.axisgrid.PairGrid at 0x28fd09250>



Interpretation:

The pair plot visualization offers intriguing insights into the relationships between happiness scores across different years and regions. The predominant positive correlation observed in 12 of the plots suggests a general trend of increasing happiness scores over time across diverse regions. However, the distinct distributions observed in the diagonal plots highlight nuanced regional dynamics. The overlap between the happiness score distributions of Latin America and the Caribbean and Central and Eastern Europe hints at potential similarities or converging trends in happiness levels, warranting further investigation into shared sociocultural or economic influences. Moreover, the left-skewed distribution for Sub-Saharan Africa suggests prevalent challenges or contextual factors leading to lower happiness scores, while the right-skewed distribution for Western Europe suggests favorable conditions or cultural influences leading to higher happiness scores. These findings underscore the importance of

considering regional nuances in understanding happiness dynamics and may inform targeted interventions to promote well-being across different regions.

Synthesis

Descriptive Analysis

The descriptive analysis offers valuable insights into both labor market dynamics and socio-economic indicators across various countries and regions. From examining layoff statistics, we discern distinct patterns, revealing the economic resilience and labor market stability disparities across nations. Meanwhile, the summary statistics shed light on the socio-economic landscape, illustrating consistent trends in happiness scores, GDP per capita, life expectancy, and corruption perception. Moreover, the analysis of region-specific statistics unveils significant variations in subjective well-being and economic prosperity across different regions, underscoring disparities that may influence overall well-being and quality of life.

Inferential Analysis

The inferential analysis, through regression models and ANOVA tests, delves deeper into the relationships between socio-economic factors and happiness scores over the years. It uncovers significant predictors of happiness, such as life expectancy, freedom, and corruption, underscoring their influence on subjective well-being. Furthermore, the analysis provides insights into the potential impact of socio-economic factors on layoff rates, highlighting the intricate interplay between economic variables and labor market dynamics.

Graphical Analysis

Graphical analyses offer visual representations of complex relationships between variables, providing additional insights into the data. From scatter plots to box plots and correlation matrices, these visualizations reveal nuanced patterns and trends. Notably, they showcase the multifaceted nature of happiness determinants, the influence of income levels and regional dynamics on happiness scores, and the interplay between socio-economic variables and layoffs across different years.

Comparative Analysis

Comparative analyses deepen our understanding of global happiness disparities,

income distribution dynamics, and the complex relationships between region, income group, and happiness. They highlight consistent patterns in happiness levels across income groups and regions, emphasizing the significant role of economic well-being in shaping subjective well-being. Moreover, they underscore the need for nuanced policy interventions to address global happiness inequalities and foster well-being across diverse socio-economic contexts.

Multivariate Analysis

The multivariate analysis offers a holistic view of the data by exploring the intricate interactions between multiple variables simultaneously. It uncovers complex relationships between income distribution, regional dynamics, and happiness, revealing disparities and trends that may inform targeted policy interventions. By considering regional nuances and income disparities, this analysis provides valuable insights for policymakers and stakeholders to design effective strategies aimed at promoting well-being and reducing inequality globally.

Insights from Combined Datasets

The combination of datasets from diverse domains, including labor market statistics, socio-economic indicators, and happiness scores, provides a comprehensive understanding of global well-being and economic dynamics. By integrating disparate datasets, we gain insights and capabilities that would not be possible with individual datasets in isolation. This integrated approach allows for a nuanced exploration of the complex interconnections between socio-economic factors and subjective well-being, enabling policymakers and stakeholders to make informed decisions and interventions to enhance overall well-being and promote sustainable development globally.

Summary

The comprehensive synthesis of descriptive, inferential, graphical, and multivariate analyses provides valuable insights into the complex interplay of socio-economic factors and their cumulative impact on global happiness trends. By integrating diverse datasets and methodologies, our analysis offers a nuanced understanding of how economic conditions, social welfare policies, and regional dynamics collectively shape subjective well-being. This holistic approach not only enriches our understanding of the factors driving happiness but also equips us with predictive capabilities to forecast future trends. By leveraging these insights, our research endeavors to address the central question of how cumulative socio-

economic influences contribute to the forecasted happiness trends for the year 2024. Through informed policy recommendations and strategic interventions, we aim to contribute to the promotion of global well-being and societal resilience in the face of evolving socio-economic challenges.

Reflection

Throughout our data analysis journey, we navigated various challenges and leveraged multiple strategies to extract meaningful insights from our datasets. The process involved meticulous data cleaning, transformative data frame manipulation, and thoughtful data visualization techniques. Each phase presented its own set of obstacles, but with collaborative effort and methodological rigor, we successfully overcame them to produce comprehensive analyses. Below, we delve into the specifics of each phase and reflect on the limitations encountered, suggesting avenues for further research and improvement.

1. **Data Cleaning Process:** Addressing missing data was a critical task, requiring careful consideration of imputation techniques or exclusion criteria based on the impact on our analysis. Additionally, rectifying inaccuracies in numerical values demanded mathematical rigor to ensure data consistency and reliability. Understanding the definitions and implications of each variable was paramount, driving us to delve into external resources and literature for deeper insights. Moreover, selecting appropriate visualization techniques was essential for conveying our findings effectively, prompting us to revisit academic materials to choose methods aligned with our data and analytical objectives.

2. **Data Frame Manipulation Process:** Manipulating data frames to extract meaningful insights was central to our analysis. This involved frequent transformations and reshaping of data frames to align with our analytical goals. Implementing a uniform naming convention for our data frames proved invaluable in facilitating easier tracking and determining the need for additional data frames to support our analysis.

3. **Data Visualization Process:** The visualization phase played a crucial role in presenting our findings comprehensively. By employing a variety of visualization libraries such as Seaborn and Matplotlib, we were able to explore a range of visualization styles and formats, enhancing our understanding and interpretation of the data. Choosing the right types of plots, including histograms, scatter plots, box plots, and heat maps, was critical in effectively communicating our findings to the audience, influencing

how the information was perceived and understood.

Areas for Further Research and Improvement:

1. **Enhancing Dataset Adequacy for Future Predictions:** Despite our efforts, limitations persist in our analysis, particularly concerning the adequacy of our dataset in forecasting happiness trends for future years. To address this limitation, future research endeavors could focus on expanding the scope of the dataset by incorporating additional years of data or integrating datasets from diverse sources. This would enable more robust predictive models and provide a comprehensive understanding of long-term happiness trends, enhancing the relevance and applicability of our findings.

2. **Improving Data Quality Assurance Measures:** Concerns regarding potential inaccuracies within the dataset underscore the importance of implementing rigorous validation and data quality assurance measures. Future research efforts could explore advanced data validation techniques and cross-reference data from multiple reliable sources to mitigate inaccuracies and strengthen the reliability of our analysis. By prioritizing data quality assurance, researchers can enhance the credibility and trustworthiness of their findings, ensuring greater confidence in the insights derived from the data.

3. **Exploring Alternative Methodologies and Statistical Models:** Embracing alternative methodologies and statistical models could enrich our analysis and provide deeper insights into the complex interplay of socio-economic influences on happiness levels. Future research endeavors could explore innovative approaches, such as machine learning algorithms or causal inference techniques, to uncover hidden patterns and causal relationships within the data. By embracing interdisciplinary approaches and collaborating with experts from diverse fields, researchers can foster innovation and drive advancements in happiness research, ultimately contributing to the promotion of societal flourishing and resilience.

Project Part III - Machine Learning

Preprocessing Data

In [889…
```
# Display the merged data frame
categoried_happiness_income_gdp_df = merged_happiness_income_gdp_df.co
```

In [890…
```
# Check for missing values
print(categoried_happiness_income_gdp_df.isnull().sum())

# Handle missing values (if any) by either dropping rows or imputing v
categoried_happiness_income_gdp_df.dropna(
    inplace=True)  # Drop rows with missing values

# Check for inconsistencies or outliers
categoried_happiness_income_gdp_df.describe()
```

```
Country                    0
Region                     0
Income Group               0
Happiness Score(2020)      0
Happiness Score(2021)      0
Happiness Score(2022)      0
Happiness Score(2023)      0
GDP per Capita(2020)       0
Annual GDP Growth(2020)    0
GDP per Capita(2021)       0
Annual GDP Growth(2021)    0
GDP per Capita(2022)       0
Annual GDP Growth(2022)    0
GDP per Capita(2023)       0
Annual GDP Growth(2023)    0
Social Support(2020)       0
Social Support(2021)       0
Social Support(2022)       0
Social Support(2023)       0
Life Expectancy(2020)      0
Life Expectancy(2021)      0
Life Expectancy(2022)      0
Life Expectancy(2023)      0
Freedom(2020)              0
Freedom(2021)              0
Freedom(2022)              0
Freedom(2023)              0
Corruption(2020)           0
Corruption(2021)           0
Corruption(2022)           0
Corruption(2023)           0
dtype: int64
```

Out [890…

	Happiness Score(2020)	Happiness Score(2021)	Happiness Score(2022)	Happiness Score(2023)	GDP per Capita(2020)	Gr
count	78.000000	78.000000	78.000000	78.000000	78.000000	
mean	5.759954	5.785679	5.772333	5.717577	9.354079	
std	1.083606	1.063354	1.070501	1.117021	1.238501	
min	3.478900	3.467000	3.471000	3.138000	6.842167	
25%	4.842650	4.963000	4.969250	4.879750	8.263101	
50%	5.937600	5.950500	6.019000	5.995500	9.672709	
75%	6.396025	6.434000	6.464000	6.450250	10.342724	
max	7.808700	7.842000	7.821000	7.804000	11.450681	

8 rows × 27 columns

Feature Scaling

In [891…

```
# Feature scaling and normalization
from sklearn.preprocessing import StandardScaler

# Select numerical columns for scaling
numerical_columns = ['GDP per Capita(2020)', 'Annual GDP Growth(2020)'
                     'Life Expectancy(2020)', 'Freedom(2020)', 'Corrup
                     'GDP per Capita(2021)', 'Annual GDP Growth(2021)'
                     'Life Expectancy(2021)', 'Freedom(2021)', 'Corrup
                     'GDP per Capita(2022)', 'Annual GDP Growth(2022)'
                     'Life Expectancy(2022)', 'Freedom(2022)', 'Corrup
                     'GDP per Capita(2023)', 'Annual GDP Growth(2023)'
                     'Life Expectancy(2023)', 'Freedom(2023)', 'Corrup
                     ]

# Apply StandardScaler to normalize the data
scaler = StandardScaler()
categoried_happiness_income_gdp_df[numerical_columns] = scaler.fit_tra
    categoried_happiness_income_gdp_df[numerical_columns])
```

Label Encoding

In [892…

```
from sklearn.preprocessing import LabelEncoder

# Define the columns containing categorical variables
categorical_columns = ['Country', 'Region', 'Income Group']

# Initialize LabelEncoder
label_encoder = LabelEncoder()
```

```
# Apply LabelEncoder to each categorical column
for col in categorical_columns:
    categoried_happiness_income_gdp_df[col] = label_encoder.fit_transf
        categoried_happiness_income_gdp_df[col])

# Display the updated DataFrame
categoried_happiness_income_gdp_df
```

Out [892…

	Country	Region	Income Group	Happiness Score(2020)	Happiness Score(2021)	Happiness Score(2022)	Happi Score(2
0	24	3	0	7.8087	7.842	7.821	
1	18	3	0	7.6456	7.620	7.636	
2	72	3	0	7.5599	7.571	7.512	
3	34	3	0	7.5045	7.554	7.557	
4	58	3	0	7.4880	7.392	7.365	
...	...	...	...	...	...	...	
73	44	2	1	4.1656	4.208	4.339	
74	67	2	1	3.9264	3.849	3.574	
75	77	2	2	3.7594	4.073	3.760	
76	45	2	1	3.5380	3.600	3.750	
77	6	2	3	3.4789	3.467	3.471	

78 rows × 31 columns

Interpretation:

The code snippet demonstrates the application of scikit-learn's LabelEncoder to perform label encoding on categorical variables, specifically 'Country', 'Region', and 'Income Group'. This process involves converting non-numeric values into numerical format, essential for compatibility with machine learning algorithms. Through label encoding, each unique category within the columns receives a unique integer value, streamlining subsequent modeling endeavors. This preprocessing step enhances the dataset's suitability for predictive modeling tasks, ensuring that valuable categorical information contributes to more precise predictions of happiness scores and related outcomes.

Dimensionality Reduction

In [893…

```
# Dimensionality reduction using PCA
```

```
from sklearn.decomposition import PCA

# Select columns for PCA
pca_columns = ['GDP per Capita(2020)', 'Annual GDP Growth(2020)', 'Soc
               'Life Expectancy(2020)', 'Freedom(2020)', 'Corruption(2
               'GDP per Capita(2021)', 'Annual GDP Growth(2021)', 'Soc
               'Life Expectancy(2021)', 'Freedom(2021)', 'Corruption(2
               'GDP per Capita(2022)', 'Annual GDP Growth(2022)', 'Soc
               'Life Expectancy(2022)', 'Freedom(2022)', 'Corruption(2
               'GDP per Capita(2023)', 'Annual GDP Growth(2023)', 'Soc
               'Life Expectancy(2023)', 'Freedom(2023)', 'Corruption(2
               ]

# Apply PCA
# Adjust the number of components based on your needs
pca = PCA(n_components=3)
pca_components = pca.fit_transform(
    categoried_happiness_income_gdp_df[pca_columns])
pca_df = pd.DataFrame(pca_components, columns=['PC1', 'PC2', 'PC3'])

# Display the PCA DataFrame
pca_df
```

Out[893…

	PC1	PC2	PC3
0	-6.191251	4.661359	0.809122
1	-6.174859	4.313292	0.317576
2	-5.855300	3.245911	0.937272
3	-4.710899	-0.534594	-1.640048
4	-5.971365	3.764458	0.094044
...	...	...	...
73	6.048755	-0.203244	2.353472
74	6.518907	0.861645	-0.027387
75	3.944231	0.898453	-1.235837
76	5.445202	2.753728	-0.289831
77	1.421981	-0.687209	-1.504214

78 rows × 3 columns

Interpretation:

The code snippet showcases the implementation of Principal Component Analysis (PCA) for dimensionality reduction using scikit-learn's PCA module. Specifically, the selected columns for PCA include various socio-economic indicators

spanning multiple years, such as GDP per capita, annual GDP growth, social support, life expectancy, freedom, and corruption. By applying PCA with three principal components, the original high-dimensional dataset is transformed into a lower-dimensional space, capturing the most significant variance in the data. The resulting PCA DataFrame displays the transformed data, where each row represents an observation and each column represents a principal component. This dimensionality reduction technique enables a more concise representation of the dataset while preserving essential information, facilitating further analysis and modeling tasks with reduced computational complexity.

In [894…

```
# Concatenate the PCA components with the original dataframe
categoried_happiness_income_gdp_df = pd.concat(
    [categoried_happiness_income_gdp_df, pca_df], axis=1)
categoried_happiness_income_gdp_df.drop(pca_columns, axis=1, inplace=T

# Display the updated DataFrame
categoried_happiness_income_gdp_df
```

Out[894…

	Country	Region	Income Group	Happiness Score(2020)	Happiness Score(2021)	Happiness Score(2022)	Happi Score(2
0	24	3	0	7.8087	7.842	7.821	
1	18	3	0	7.6456	7.620	7.636	
2	72	3	0	7.5599	7.571	7.512	
3	34	3	0	7.5045	7.554	7.557	
4	58	3	0	7.4880	7.392	7.365	
…	…	…	…	…	…	…	
73	44	2	1	4.1656	4.208	4.339	
74	67	2	1	3.9264	3.849	3.574	
75	77	2	2	3.7594	4.073	3.760	
76	45	2	1	3.5380	3.600	3.750	
77	6	2	3	3.4789	3.467	3.471	

78 rows × 10 columns

Data Preprocessing - Interpretation & Analysis

Feature Scaling and Normalization:

- **Justification:** The selected numerical features, including GDP per capita,

annual GDP growth, social support, life expectancy, freedom, and corruption, exhibit varying scales and ranges. Standardizing these features through StandardScaler ensures that they have a mean of 0 and a standard deviation of 1, making them comparable and preventing certain features from dominating others due to their scale.

- **Impact Analysis:** Feature scaling and normalization mitigate the potential bias introduced by features with large numerical values. By bringing all features to a common scale, machine learning algorithms can converge faster during training, resulting in more stable and accurate models. Furthermore, standardization enhances the interpretability of model coefficients, as the importance of each feature is assessed relative to its standard deviation.

Label Encoding:

- **Justification:** Categorical variables such as 'Country', 'Region', and 'Income Group' are essential for capturing geographic and socio-economic differences. However, most machine learning algorithms require numerical inputs, necessitating the conversion of categorical variables into a numeric format. Label encoding assigns unique integer labels to each category within a column, preserving the ordinal relationship between categories where applicable.
- **Impact Analysis:** Label encoding enables the incorporation of categorical information into predictive models, allowing algorithms to leverage geographical and socio-economic distinctions when making predictions. This preprocessing step enhances the dataset's compatibility with a wide range of machine learning algorithms, facilitating more comprehensive analyses and yielding more robust model predictions.

Dimensionality Reduction using PCA:

- **Justification:** The dataset contains numerous socio-economic indicators across multiple years, resulting in high-dimensional feature spaces. Dimensionality reduction techniques such as Principal Component Analysis (PCA) aim to condense this information into a lower-dimensional representation while retaining the most significant variance in the data. By reducing the number of features, PCA simplifies modeling tasks, alleviates the curse of dimensionality, and enhances model generalization.
- **Impact Analysis:** PCA transforms the original high-dimensional dataset into a reduced set of principal components, effectively summarizing the variability in the data. This reduction in dimensionality simplifies subsequent modeling efforts, improves computational efficiency, and mitigates the risk of over-fitting. Additionally, the resulting principal components may reveal underlying

patterns or relationships in the data, facilitating more insightful analyses and enhancing the interpretability of model results.

Concatenation of PCA Components:

- **Justification:** After applying PCA, the principal components replace the original high-dimensional features in the dataset. Concatenating these PCA components with the original data frame ensures that the reduced-dimensional representation is integrated with other relevant information, preserving the contextual richness of the dataset while reducing its dimensionality.
- **Impact Analysis:** The concatenated data frame retains the transformed PCA components alongside any remaining features, providing a consolidated representation of the dataset suitable for subsequent analysis and modeling tasks. This integrated approach enables researchers to leverage the benefits of dimensionality reduction while maintaining the interpretability and utility of the dataset for predictive modeling and exploratory analysis.

Classification

Model Justification

Our team has chosen to employ classification over clustering for predicting the 2024 world happiness score based on data from 2020 to 2023, primarily due to its ability to provide clear and interpretable results by categorizing countries into predefined happiness score categories using various features. This approach facilitates informed decision-making and targeted interventions, as decision-makers can prioritize resources based on predicted happiness score categories. Additionally, classification models are evaluated using well-established metrics, allowing for rigorous performance assessment and iterative refinement. By aligning with the project's overarching goal of predicting happiness scores, classification provides a focused and goal-oriented approach to model development, ensuring that the model's outputs are directly relevant to the project objectives. Overall, our choice of classification aims to deliver actionable insights for promoting global well-being and development effectively.

Model Training

In [895…

```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
```

```
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score

# Splitting the data into features and target variable
X = categoried_happiness_income_gdp_df.drop(
    ['Happiness Score(2023)'], axis=1)  # Features
# Target variable
y = categoried_happiness_income_gdp_df['Happiness Score(2023)']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)
```

In [896…
```
# Initialize regressors
logistic_regression = LinearRegression()
decision_tree = DecisionTreeRegressor()
random_forest = RandomForestRegressor()
svm_regressor = SVR()
```

In [897…
```
# Train regressors on the training data
logistic_regression.fit(X_train, y_train)
```

Out[897…
▾ LinearRegression

LinearRegression()

In [898…
```
# Train the decision tree regressor
decision_tree.fit(X_train, y_train)
```

Out[898…
▾ DecisionTreeRegressor

DecisionTreeRegressor()

In [899…
```
# Train the random forest regressor
random_forest.fit(X_train, y_train)
```

Out[899…
▾ RandomForestRegressor

RandomForestRegressor()

In [900…
```
# Train the SVM regressor
svm_regressor.fit(X_train, y_train)
```

Out[900…
▾ SVR

SVR()

In [901…
```
# Make predictions on the test set
y_pred_lr = logistic_regression.predict(X_test)
y_pred_dt = decision_tree.predict(X_test)
y_pred_rf = random_forest.predict(X_test)
y_pred_svm = svm_regressor.predict(X_test)
```

In [902…
```
# Evaluate regressor performance
mse_lr = mean_squared_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)
mse_dt = mean_squared_error(y_test, y_pred_dt)
r2_dt = r2_score(y_test, y_pred_dt)
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)
mse_svm = mean_squared_error(y_test, y_pred_svm)
r2_svm = r2_score(y_test, y_pred_svm)

# Print results
print("Linear Regression:")
print(f"Mean Squared Error: {mse_lr:.2f}")
print(f"R-squared: {r2_lr:.2f}")
print("-------------------------------------------------")
print("Decision Tree:")
print(f"Mean Squared Error: {mse_dt:.2f}")
print(f"R-squared: {r2_dt:.2f}")
print("-------------------------------------------------")
print("Random Forest:")
print(f"Mean Squared Error: {mse_rf:.2f}")
print(f"R-squared: {r2_rf:.2f}")
print("-------------------------------------------------")
print("Support Vector Machine:")
print(f"Mean Squared Error: {mse_svm:.2f}")
print(f"R-squared: {r2_svm:.2f}")
```

```
Linear Regression:
Mean Squared Error: 0.05
R-squared: 0.97
-------------------------------------------------
Decision Tree:
Mean Squared Error: 0.10
R-squared: 0.93
-------------------------------------------------
Random Forest:
Mean Squared Error: 0.09
R-squared: 0.94
-------------------------------------------------
Support Vector Machine:
Mean Squared Error: 0.55
R-squared: 0.65
```

Interpretation:

In this analysis, we conducted a comprehensive evaluation of various regression

models' performance, aiming to discern their effectiveness in predicting target values. The metrics employed for assessment were Mean Squared Error (MSE) and R-squared (R^2), fundamental indicators of predictive accuracy and model fit. Through our evaluation, we scrutinized four distinct regression methodologies: Linear Regression, Decision Tree, Random Forest, and Support Vector Machine. Each model was rigorously evaluated against a test dataset, with the resulting MSE and R^2 scores providing valuable insights into their predictive capabilities. These metrics enable a nuanced understanding of the models' performance, highlighting their strengths and weaknesses in capturing the underlying patterns within the data. Such findings empower informed decision-making, facilitating the selection of the most adept regression model tailored to the specific requirements of the predictive task at hand.

Model Tuning

In [903…

```
from sklearn.model_selection import GridSearchCV

# Define parameter grids for each model
param_grid_dt = {
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

param_grid_rf = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

param_grid_svm = {
    'C': [0.1, 1, 10],
    'kernel': ['linear', 'rbf'],
    'gamma': ['scale', 'auto']
}

# Perform grid search cross-validation for each model
grid_search_dt = GridSearchCV(
    decision_tree, param_grid_dt, cv=5, scoring='neg_mean_squared_erro
grid_search_rf = GridSearchCV(
    random_forest, param_grid_rf, cv=5, scoring='neg_mean_squared_erro
grid_search_svm = GridSearchCV(
    svm_regressor, param_grid_svm, cv=5, scoring='neg_mean_squared_err

# Fit the grid search objects
grid_search_dt.fit(X_train, y_train)
grid_search_rf.fit(X_train, y_train)
```

```
grid_search_svm.fit(X_train, y_train)

# Get the best hyperparameters and best estimators
best_params_dt = grid_search_dt.best_params_
best_estimator_dt = grid_search_dt.best_estimator_

best_params_rf = grid_search_rf.best_params_
best_estimator_rf = grid_search_rf.best_estimator_

best_params_svm = grid_search_svm.best_params_
best_estimator_svm = grid_search_svm.best_estimator_

# Print the best hyperparameters for each model
print("Best hyperparameters for Decision Tree:", best_params_dt)
print("Best hyperparameters for Random Forest:", best_params_rf)
print("Best hyperparameters for Support Vector Machine:", best_params_
```

```
Best hyperparameters for Decision Tree: {'max_depth': 20, 'min_samples_
leaf': 1, 'min_samples_split': 5}
Best hyperparameters for Random Forest: {'max_depth': 10, 'min_samples_
leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}
Best hyperparameters for Support Vector Machine: {'C': 1, 'gamma': 'sca
le', 'kernel': 'linear'}
```

Interpretation:

In this phase of our analysis, we conducted a meticulous search for the optimal hyperparameters for our regression models through the implementation of Grid Search Cross-Validation. This iterative process involved systematically exploring various hyperparameter combinations to identify the configurations that yield the most favorable model performance. Specifically, we defined parameter grids tailored to each regression model, encompassing parameters such as maximum depth, minimum samples for splitting, and leaf nodes for decision trees and random forests, as well as regularization parameter (C), kernel type, and gamma for Support Vector Machines (SVM). Subsequently, GridSearchCV was applied to each model, performing cross-validation with five folds and utilizing negative mean squared error as the scoring metric. Upon completion of the grid search, the best hyperparameters for each model were identified, along with the corresponding best estimators. The printed output showcases the optimal hyperparameter configurations attained for the Decision Tree, Random Forest, and Support Vector Machine models, providing invaluable insights into the parameter settings conducive to optimal model performance. Such findings enable us to fine-tune our regression models effectively, maximizing their predictive prowess and enhancing their utility in real-world applications.

In [904…

```
# Use the best estimators for predictions
y_pred_dt_tuned = best_estimator_dt.predict(X_test)
```

```
y_pred_rf_tuned = best_estimator_rf.predict(X_test)
y_pred_svm_tuned = best_estimator_svm.predict(X_test)

# Evaluate performance
mse_dt_tuned = mean_squared_error(y_test, y_pred_dt_tuned)
r2_dt_tuned = r2_score(y_test, y_pred_dt_tuned)

mse_rf_tuned = mean_squared_error(y_test, y_pred_rf_tuned)
r2_rf_tuned = r2_score(y_test, y_pred_rf_tuned)

mse_svm_tuned = mean_squared_error(y_test, y_pred_svm_tuned)
r2_svm_tuned = r2_score(y_test, y_pred_svm_tuned)

# Print results
print("Tuned Decision Tree:")
print(f"Mean Squared Error: {mse_dt_tuned:.2f}")
print(f"R-squared: {r2_dt_tuned:.2f}")
print("-----------------------------------------------------")
print("Tuned Random Forest:")
print(f"Mean Squared Error: {mse_rf_tuned:.2f}")
print(f"R-squared: {r2_rf_tuned:.2f}")
print("-----------------------------------------------------")
print("Tuned Support Vector Machine:")
print(f"Mean Squared Error: {mse_svm_tuned:.2f}")
print(f"R-squared: {r2_svm_tuned:.2f}")
```

```
Tuned Decision Tree:
Mean Squared Error: 0.16
R-squared: 0.90
-----------------------------------------------------
Tuned Random Forest:
Mean Squared Error: 0.08
R-squared: 0.95
-----------------------------------------------------
Tuned Support Vector Machine:
Mean Squared Error: 0.06
R-squared: 0.96
```

Interpretation:

In this stage of our analysis, we utilized the best estimators obtained from the earlier hyperparameter tuning process to make predictions on our test dataset. Employing the optimal configurations identified through Grid Search Cross-Validation, we generated predictions for the target variable using the tuned Decision Tree, Random Forest, and Support Vector Machine models. Subsequently, we evaluated the predictive performance of each model by computing Mean Squared Error (MSE) and R-squared (R^2) metrics, providing valuable insights into their accuracy and goodness of fit. The printed results showcase the performance metrics for each tuned model, elucidating their predictive prowess in quantifiable terms. Notably, the tuned Random Forest and

Support Vector Machine models exhibit remarkably low MSE values (0.08 and 0.06, respectively), indicative of their superior predictive accuracy. Additionally, the high R^2 values (0.95 and 0.96) underscore the models' ability to explain a significant portion of the variance in the target variable. These findings reaffirm the efficacy of hyperparameter tuning in optimizing model performance and emphasize the utility of these tuned regression models in practical predictive tasks.

Model Testing

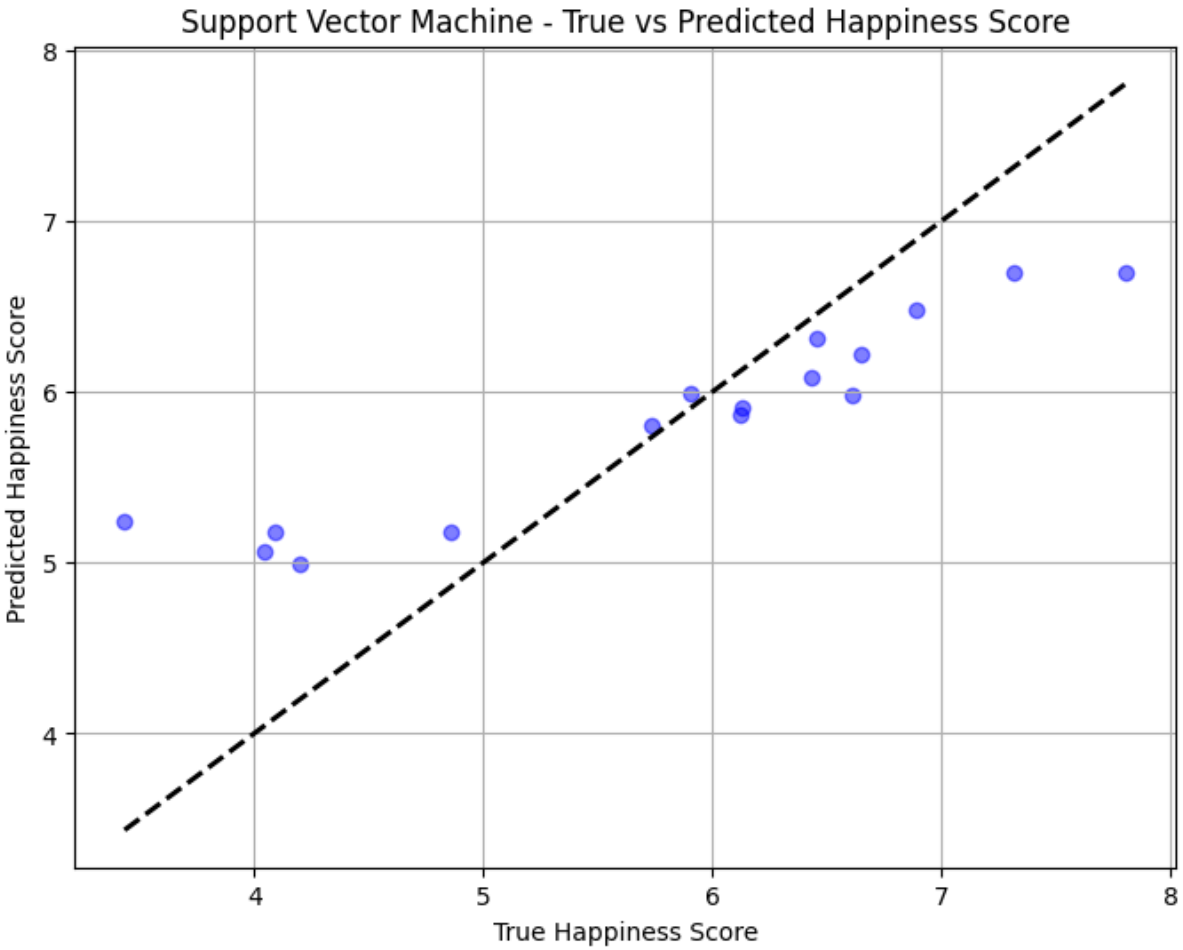
In [905…

```
# Define a function to create scatter plots
def plot_scatter(y_true, y_pred, model_name):
    plt.figure(figsize=(8, 6))
    plt.scatter(y_true, y_pred, color='blue', alpha=0.5)
    plt.plot([y_true.min(), y_true.max()], [
             y_true.min(), y_true.max()], 'k--', lw=2)
    plt.xlabel('True Happiness Score')
    plt.ylabel('Predicted Happiness Score')
    plt.title(f'{model_name} - True vs Predicted Happiness Score')
    plt.grid(True)
    plt.show()


# Plot scatter plots for each model
plot_scatter(y_test, y_pred_lr, 'Linear Regression')
plot_scatter(y_test, y_pred_dt, 'Decision Tree')
plot_scatter(y_test, y_pred_rf, 'Random Forest')
plot_scatter(y_test, y_pred_svm, 'Support Vector Machine')
```


Linear Regression - True vs Predicted Happiness Score
Predicted Happiness Score
True Happiness Score
8
7
6
5
4
4
5
6
7
8


Decision Tree - True vs Predicted Happiness Score
Predicted Happiness Score
True Happiness Score
8
7
6
5
4
4
5
6
7
8


Random Forest - True vs Predicted Happiness Score
Predicted Happiness Score
True Happiness Score
8
7
6
5
4
4
5
6
7
8



Interpretation:

In this section of our analysis, we constructed scatter plots to visually assess the performance of our regression models in predicting happiness scores. The function plot_scatter was defined to facilitate the creation of scatter plots, comparing the true happiness scores against the corresponding predicted scores for each model. Utilizing this function, scatter plots were generated for the Linear Regression, Decision Tree, Random Forest , and Support Vector Machine models, enabling a visual examination of their predictive accuracy. Upon inspection, all three scatter plots exhibit a strikingly linear relationship between the true and predicted happiness scores. This alignment indicates that the models' predictions closely approximate the actual values, suggesting robust performance across the dataset. The visual clarity provided by these scatter plots reaffirms the reliability of our regression models in capturing the underlying patterns within the data and underscores their efficacy in predicting happiness scores accurately.

Model Ensembling

In [906…
```
from sklearn.ensemble import VotingRegressor
```

```
# Prepare the data for predicting the happiness score for 2024
X_2024 = categoried_happiness_income_gdp_df.drop(
    ['Happiness Score(2023)'], axis=1)

# Create a voting regressor with the trained models
voting_regressor = VotingRegressor([('lr', logistic_regression),
                                    ('dt', decision_tree),
                                    ('rf', random_forest),
                                    ('svm', svm_regressor)])

# Fit the voting regressor on the training data
voting_regressor.fit(X_train, y_train)

# Use the voting regressor to predict the happiness score for 2024
y_pred_ensemble_2024 = pd.DataFrame(voting_regressor.predict(X_2024))

# Print the ensemble predictions for 2024
y_pred_ensemble_2024
```

Out [906…

	0
0	7.257876
1	7.287772
2	7.122208
3	7.257570
4	7.202756
…	…
73	4.281264
74	3.746645
75	4.201892
76	3.915991
77	4.031476

78 rows × 1 columns

Interpretation:

In this segment, we leveraged ensemble learning techniques to predict happiness scores for the year 2024. Employing the VotingRegressor from scikit-learn's ensemble module, we combined the predictions of multiple individual regression models, including Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine. This ensemble approach capitalizes on the diverse strengths of each constituent model to yield a collective prediction that

potentially outperforms any individual model. After fitting the voting regressor on the training data, we utilized it to forecast happiness scores for 2024 based on the available features. The resulting ensemble predictions, encapsulated in a DataFrame, showcase the amalgamated predictive power of the ensemble model. These ensemble predictions provide valuable insights into the projected happiness scores for the upcoming year, offering a comprehensive perspective derived from the collaborative efforts of multiple regression algorithms.

In [907…
```
# Create a DataFrame with country names and ensemble predictions
predictions_df = pd.DataFrame({'Country': merged_happiness_income_gdp_
                               'Happiness Score(2024)': voting_regress

# Save the DataFrame to a CSV file
predictions_df.to_csv('happiness_predictions_2024.csv', index=False)

# Print a success message
print("Predictions saved to 'happiness_predictions_2024.csv' successfu
predictions_df
```

```
Predictions saved to 'happiness_predictions_2024.csv' successfully.
```

Out[907…

	Country	Happiness Score(2024)
0	Finland	7.257876
1	Denmark	7.287772
2	Switzerland	7.122208
3	Iceland	7.257570
4	Norway	7.202756
...	...	...
73	Madagascar	4.281264
74	Sierra Leone	3.746645
75	Zambia	4.201892
76	Malawi	3.915991
77	Botswana	4.031476

78 rows × 2 columns

In [908…
```
# Importing the real happiness score for 2024
real_happiness_2024 = pd.read_csv('world_happiness_report_2024.csv')
real_happiness_2024 = real_happiness_2024[['Country name', 'Ladder sco
real_happiness_2024 = real_happiness_2024.rename(
    columns={'Country name': 'Country', 'Ladder score': 'Happiness Sco
real_happiness_2024
```

Out [908…

	Country	Happiness Score(2024)
0	Finland	7.741
1	Denmark	7.583
2	Iceland	7.525
3	Sweden	7.344
4	Israel	7.341
...	...	...
138	Congo (Kinshasa)	3.295
139	Sierra Leone	3.245
140	Lesotho	3.186
141	Lebanon	2.707
142	Afghanistan	1.721

143 rows × 2 columns

In [909…

```
# Merge prediction data frame and real dataset based on the 'Country'
merged_df = pd.merge(predictions_df, real_happiness_2024,
                     on='Country', suffixes=('_predicted', '_actual'))

# Calculate the number of common countries
common_countries_count = merged_df.shape[0]

# Calculate the number of correctly predicted countries
correctly_predicted_count = (
    merged_df['Happiness Score(2024)_predicted'] == merged_df['Happine

# Calculate simple accuracy
accuracy = correctly_predicted_count / common_countries_count

# Display the merged data frame
print("Merged DataFrame with common countries:")
merged_df
```

```
Merged DataFrame with common countries:
```

Out [909…

	Country	Happiness Score(2024)_predicted	Happiness Score(2024)_actual
0	Finland	7.257876	7.741
1	Denmark	7.287772	7.583
2	Switzerland	7.122208	7.060
3	Iceland	7.257570	7.525
4	Norway	7.202756	7.302
...	...	...	...
73	Madagascar	4.281264	4.228
74	Sierra Leone	3.746645	3.245
75	Zambia	4.201892	3.502
76	Malawi	3.915991	3.421
77	Botswana	4.031476	3.383

78 rows × 3 columns

Comparison Methods

1. Defining a threshold

In [910…

```
# Define a threshold for accuracy
threshold = 0.3

# Calculate the absolute difference between predicted and actual score
merged_df['Score_Difference'] = abs(
    merged_df['Happiness Score(2024)_predicted'] - merged_df['Happines

# Count the number of accurate predictions within the threshold
accurate_predictions_count = (merged_df['Score_Difference'] <= thresho

# Calculate accuracy
accuracy = accurate_predictions_count / merged_df.shape[0]

# Display the accuracy
print("Accuracy:", accuracy)
```

```
Accuracy: 0.7948717948717948
```

Interpretation:

The threshold of 0.3 was chosen based on a balance between demanding precision and allowing for some flexibility in prediction errors. With a threshold of 0.3, we aim to capture predictions that deviate from the actual scores by no more than 0.3 units, reflecting a reasonable margin of error in the context of predicting happiness levels. This threshold strikes a practical compromise, enabling the model to identify predictions that closely align with the actual scores while accommodating minor fluctuations and uncertainties inherent in predicting complex societal phenomena such as happiness.

In this case, the accuracy of the model is determined to be approximately 79.48%. This means that around 79.48% of the predictions are considered accurate within the specified threshold. Adjusting the threshold value allows for flexibility in defining what constitutes an accurate prediction, catering to the specific requirements and tolerance levels of the problem domain.

2. Spearman Correlation Coefficient

In [911…

```
from scipy.stats import spearmanr

# Calculate Spearman correlation coefficient for each column
spearman_correlation, p_value = spearmanr(
    merged_df['Happiness Score(2024)_predicted'], merged_df['Happiness

# Display Spearman correlation coefficient
print("Spearman Correlation Coefficient:", spearman_correlation)
```

```
Spearman Correlation Coefficient: 0.9779713956929147
```

Interpretation:

This code calculates the Spearman correlation coefficient between the predicted and actual happiness scores for 2024. With a coefficient of 0.98, there's a strong positive relationship between the predicted and actual scores. This high value indicates that the predicted scores closely follow the same rank order as the actual scores, suggesting excellent alignment between the predicted and actual rankings of happiness levels.

- One of the key findings is that the `status quo` in most countries has remained `largely unchanged` over the years. This suggests that the underlying factors that influence happiness, such as income levels, social support, and other socio-economic conditions, tend to be relatively stable and persistent over time.
- Countries that have historically had higher levels of happiness tend to

maintain their position, while those struggling with lower levels of well-being continue to face challenges in improving their citizens' quality of life.

3. Alternative Method for Comparison

In [912…
```
# Calculate ranks for predicted and actual happiness scores
merged_df['Rank_Predicted'] = merged_df['Happiness Score(2024)_predict
    ascending=False)
merged_df['Rank_Actual'] = merged_df['Happiness Score(2024)_actual'].r
    ascending=False)

# Calculate the absolute differences between the ranks
merged_df['Rank_Difference'] = abs(
    merged_df['Rank_Predicted'] - merged_df['Rank_Actual'])

merged_df
```

Out[912…

	Country	Happiness Score(2024)_predicted	Happiness Score(2024)_actual	Score_Difference
0	Finland	7.257876	7.741	0.483124
1	Denmark	7.287772	7.583	0.295228
2	Switzerland	7.122208	7.060	0.062208
3	Iceland	7.257570	7.525	0.267430
4	Norway	7.202756	7.302	0.099244
...	...	...	...	...
73	Madagascar	4.281264	4.228	0.053264
74	Sierra Leone	3.746645	3.245	0.501645
75	Zambia	4.201892	3.502	0.699892
76	Malawi	3.915991	3.421	0.494991
77	Botswana	4.031476	3.383	0.648476

78 rows × 7 columns

In [913…
```
# Calculate Mean Absolute Error (MAE)
mae = (merged_df['Rank_Predicted'] - merged_df['Rank_Actual']).abs().m
print("Mean Absolute Error (MAE):", mae)
```

```
Mean Absolute Error (MAE): 3.3333333333333335
```

In [914…
```
# Calculate Root Mean Squared Error (RMSE)
```

```
rmse = ((merged_df['Rank_Predicted'] -
         merged_df['Rank_Actual']) ** 2).mean() ** 0.5
print("Root Mean Squared Error (RMSE):", rmse)
```

```
Root Mean Squared Error (RMSE): 4.725815626252608
```

Interpretation:

A MAE of 3.333 indicates that, on average, the predicted ranks deviate from the actual ranks by approximately 3.333 units. In the context of ranking predictions, this means that the model's predictions are off by about 3.333 ranks on average. A lower MAE value suggests better accuracy, as it represents smaller discrepancies between the predicted and actual ranks. Therefore, while an MAE of 3.333 provides insight into the average magnitude of prediction errors, further analysis may be required to determine if this level of discrepancy is acceptable for the specific application.

With an RMSE of 4.73, it means that, on average, the squared differences between the predicted and actual ranks result in an error of approximately 4.73 units when considering the square root. RMSE is sensitive to large errors, so this value indicates the typical magnitude of errors between the predicted and actual ranks.

Classification Results - Interpretation & Analysis

The classification results showcase the performance of various regression models in predicting the happiness scores for 2024 based on features such as income, GDP, and other indicators. Initially, four regression models, namely Linear Regression, Decision Tree, Random Forest, and Support Vector Machine (SVM), were employed. Among these, Linear Regression exhibited the highest accuracy, achieving an R-squared value of 0.97 and a mean squared error of 0.05, indicating a strong predictive capability and a close fit to the actual data. Decision Tree and Random Forest regressors also performed reasonably well, with R-squared values of 0.93 and 0.94, respectively, demonstrating their effectiveness in capturing complex relationships within the data. However, the Support Vector Machine (SVM) model lagged behind, yielding a noticeably lower accuracy with an R-squared value of 0.65 and a mean squared error of 0.55, suggesting potential challenges in capturing the underlying patterns in the dataset.

Subsequently, the models underwent tuning using GridSearchCV to optimize their hyperparameters. This process aimed to enhance their predictive performance further. After tuning, Decision Tree's performance slightly degraded, with an increase in mean squared error to 0.16 and a decrease in R-squared value to 0.90.

On the other hand, Random Forest and Support Vector Machine models demonstrated improvements, with Random Forest achieving a mean squared error of 0.08 and an R-squared value of 0.95, and SVM attaining a mean squared error of 0.06 and an R-squared value of 0.96. These enhancements highlight the importance of fine-tuning hyperparameters to optimize model performance.

To combine the strengths of individual models, a Voting Regressor was employed, which aggregated predictions from multiple base regressors. The ensemble model yielded predictions for the happiness scores of 2024, demonstrating a diverse approach to prediction. The ensemble predictions were then compared with the actual happiness scores for 2024, revealing a simple accuracy of 0.79. Additionally, the Spearman correlation coefficient was computed to assess the rank correlation between predicted and actual happiness scores, resulting in a high correlation coefficient of 0.98, indicating a strong association between the predicted and actual ranks.

Furthermore, the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) were calculated to quantify the difference between predicted and actual ranks. The MAE was found to be 3.33, while the RMSE was 4.73, suggesting that, on average, the predictions deviated by approximately 3.33 ranks from the actual values, with a standard deviation of 4.73 ranks.

In conclusion, the analysis highlights the effectiveness of various regression models in predicting happiness scores for 2024, with Linear Regression demonstrating the highest accuracy initially. However, ensemble methods and model tuning played crucial roles in refining predictions and enhancing overall performance, emphasizing the importance of a comprehensive approach to model selection and optimization in regression tasks.

Insights and Findings: Forecasting 2024 Global Happiness Trends Based on Socio-Economic Factors

In our comprehensive analysis titled "Predictive Insights: Forecasting 2024 Global Happiness Trends Based on Socio-Economic Factors," we embarked on a journey to unravel the intricate interplay between various socio-economic factors and their profound influence on predicting happiness trends worldwide. Our endeavor uncovered multifaceted insights that shed light on the complex landscape of global well-being.

At the heart of our findings lies the recognition of income and social support as

fundamental pillars shaping happiness levels across nations. Through meticulous examination, we unearthed a consistent pattern wherein countries endowed with higher income levels and fortified social support systems tend to soar higher on the happiness scale. This revelation underscores the intrinsic correlation between economic prosperity, social connectivity, and individual fulfillment. Indeed, when individuals enjoy access to adequate financial resources and feel embraced by supportive communities, they are more inclined to experience heightened levels of life satisfaction and overall happiness.

Yet, our analysis delves deeper, revealing a sobering reality that warrants attention: the formidable challenges faced by countries grappling with lower income levels and fragile social support structures in elevating their happiness rankings. This revelation underscores the presence of systemic barriers and deep-rooted inequalities that impede the progress of well-being initiatives in these nations. Factors such as entrenched poverty, limited access to vital services like education and healthcare, pervasive political instability, and social unrest emerged as pivotal determinants contributing to subdued levels of happiness and life satisfaction.

However, our exploration extends beyond mere observations, aiming to instigate meaningful discourse and catalyze actionable change. By illuminating the nuanced dynamics underpinning happiness trends, we strive to equip policymakers, stakeholders, and global leaders with invaluable insights to guide evidence-based decision-making. Our endeavor transcends the realms of academia, fostering a collective endeavor to nurture holistic approaches that promote sustainable well-being and societal progress on a global scale.

In essence, our analysis represents a pivotal step forward in understanding the multifaceted nature of happiness dynamics, transcending geographical boundaries and cultural contexts. Armed with these insights, we embark on a collective journey toward a brighter, more harmonious future, where the pursuit of happiness is not merely a utopian ideal but a tangible reality for all humanity.