Beyond Numbers: A swift EDA of Saudi Arabia's Population dataset

About Author

- Project: Population of Saudi Arabia
- Author: Syed Salman Bacha
- Code Submission Date: 25-11-2023
- Author's Contact Info:\

Email
Github
kaggle
Linkedin
twitter

About Data

- Data: Apple AppStore
- Data Age: The data was collected in the month of October 2021.
- **Dataset:** *link*

Description: Explore the diverse demographics of Saudi Arabia through this comprehensive dataset showcasing population statistics across various parameters. The dataset contains records detailing the population dynamics in terms of gender, nationality, region, and year.

Columns:

- **Gender:** Specifies the gender of individuals (e.g., male, female).
- **Nationality:** Represents the nationality of individuals residing in Saudi Arabia.
- **Region:** Indicates the geographical region within Saudi Arabia.
- **Year:** Represents the year of the recorded population data.
- **Population:** Denotes the population count based on the respective parameters.

Key Insights:

1. **Temporal Population Trends:** Unveil the changing population dynamics over the years within Saudi Arabia.
2. **Gender Distribution:** Explore the distribution and changes in population based on gender across different regions and years.
3. **Nationality Diversity:** Investigate the demographic diversity concerning various nationalities residing in Saudi Arabia.
4. **Regional Variations:** Analyze how population sizes differ across the diverse regions within the country.

Why Explore this Dataset?

- **Cultural Diversity:** Gain insights into the diverse cultural landscape of Saudi Arabia through population demographics.
- **Socio-Economic Analysis:** Understand how population dynamics impact socio-economic aspects within different regions.
- **Policy Implications:** Potentially inform policies based on trends observed in population demographics over time.

Loading the important libraries

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

Loading the data

In [ ]:
```
df = pd.read_csv('KSA_population.csv')
```

Lets take a look at the data

In [ ]:
```
df.head()
```

Out[ ]:

	Gender	Nationality	Region	Year	Population
0	Female	Non Saudi	Al Bahah	2010	16209
1	Female	Non Saudi	Al Bahah	2011	16521
2	Female	Non Saudi	Al Bahah	2012	16752
3	Female	Non Saudi	Al Bahah	2013	17508
4	Female	Non Saudi	Al Bahah	2014	17682

Shape of the data

In [ ]:
```
# Step 1: Data shape
print(df.shape)
rows, columns = df.shape
print(f"Num of Rows: {rows} ") # instances
print(f"Num of Columns: {columns} ") # series
print(f"The size (rows x columns) is: {df.size}") # size
print(f"The Dimensions are: {df.ndim}") # dimensions
```

```
(676, 5)
Num of Rows: 676
Num of Columns: 5
The size (rows x columns) is: 3380
The Dimensions are: 2
```

Columns In the Data

In [ ]:
```
df.columns
```

Out[ ]:
```
Index(['Gender', 'Nationality', 'Region', 'Year', 'Population'], dtype='object')
```

Number of Unique values in each Column

In [ ]:
```
df.nunique()
```

Out[ ]:
```
Gender           2
Nationality      2
Region          13
Year            13
Population     675
dtype: int64
```

View summary of the dataset

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 676 entries, 0 to 675
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Gender       676 non-null    object
 1   Nationality  676 non-null    object
 2   Region       676 non-null    object
 3   Year         676 non-null    int64 
 4   Population   676 non-null    int64 
dtypes: int64(2), object(3)
memory usage: 26.5+ KB
```

Important points:

- We can see that the dataset contains mixture of categorical and numerical variables.
- Categorical variables have data type `object`.
- Numerical variables have data type `int64`.

Statistical Summary of the Data

In [ ]:
```
df.describe()
```

Out[ ]:

	Year	Population
count	676.000000	6.760000e+02
mean	2016.000000	5.587169e+05
std	3.744428	7.062714e+05
min	2010.000000	1.430400e+04
25%	2013.000000	1.098975e+05
50%	2016.000000	2.495590e+05
75%	2019.000000	6.180745e+05
max	2022.000000	3.406281e+06

Important points to note

- The above command `df.describe()` helps us to view the statistical properties of numerical variables. It excludes character variables.
- If we want to view the statistical properties of character variables, we should run the following command -

```
df.describe(include=['object'])
```

- If we want to view the statistical properties of all the variables, we should run the following command -

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

Univerate Analysis

1. Explore `Gender` Variable

In [ ]:
```
df['Gender'].unique()
```

Out[ ]:
```
array(['Female', 'Male'], dtype=object)
```

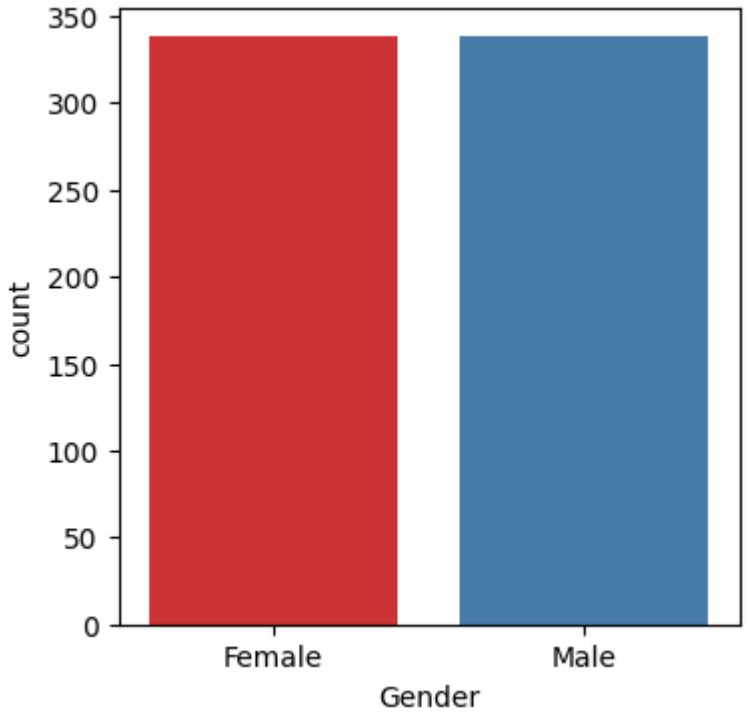
We can see that the number of unique values in `Gender` variable is 2.

The two unique values are `Male` and `Female`

In [ ]:
```
df['Gender'].value_counts()
```

Out[ ]:
```
Female    338
Male      338
Name: Gender, dtype: int64
```

In [ ]:
```
f, ax = plt.subplots(figsize=(4, 4))
ax = sns.countplot(x="Gender", data=df, palette="Set1")
plt.show()
```



2. Explore `Nationallity` variable

In [ ]:
```
df['Nationality'].nunique()
```

Out[ ]:
```
2
```

In [ ]:
```
df['Nationality'].unique()
```

Out[ ]:
```
array(['Non Saudi', 'Saudi'], dtype=object)
```

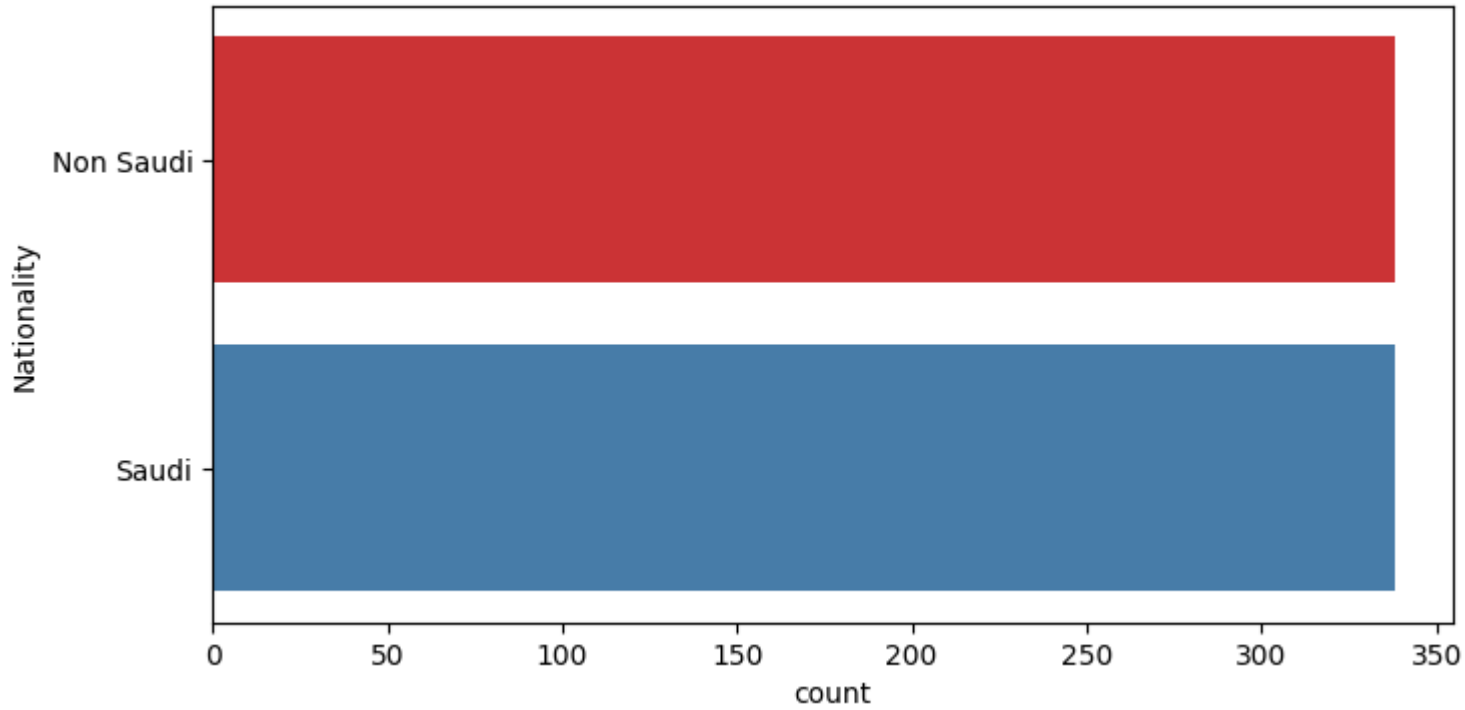
We can see that the number of unique values in `Nationality` variable is 2.

The two unique values are `Non Saudi` and `Saudi`

In [ ]:
```
df['Nationality'].value_counts()
```

Out[ ]:
```
Non Saudi    338
Saudi        338
Name: Nationality, dtype: int64
```

In [ ]:
```
f, ax = plt.subplots(figsize=(8, 4))
ax = sns.countplot(y="Nationality", data=df, palette="Set1")
plt.show()
```



3. Explore `Region` Variable

In [ ]:
```
df['Region'].nunique()
```

Out[ ]:
```
13
```

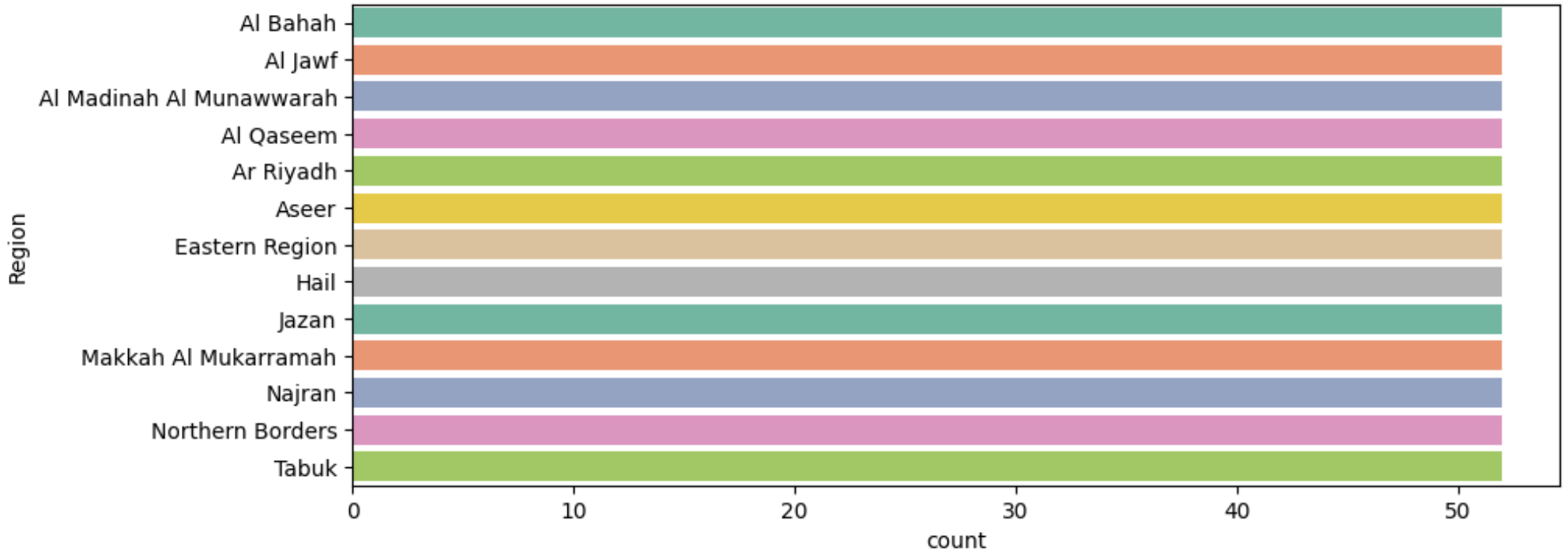
In [ ]:
```
df['Region'].unique()
```

Out[ ]:
```
array(['Al Bahah', 'Al Jawf', 'Al Madinah Al Munawwarah', 'Al Qaseem',
       'Ar Riyadh', 'Aseer', 'Eastern Region', 'Hail', 'Jazan',
       'Makkah Al Mukarramah', 'Najran', 'Northern Borders', 'Tabuk'],
      dtype=object)
```

We can see that the number of unique values in `Region` variable are 13.

The 13 unique values are `Al Bahah`, `Al Jawf`, `Al Madinah Al Munawwarah`, `Al Qaseem`, `Ar Riyadh`, `Aseer`, `Eastern Region`, `Hail`, `Jazan`, `Makkah Al Mukarramah`, `Najran`, `Northern Borders`, `Tabuk`

In [ ]:
```
f, ax = plt.subplots(figsize=(10, 4))
ax = sns.countplot(y="Region", data=df, palette="Set2")
plt.show()
```



4. Explore `Year` Variable

In [ ]:
```
df['Year'].nunique()
```

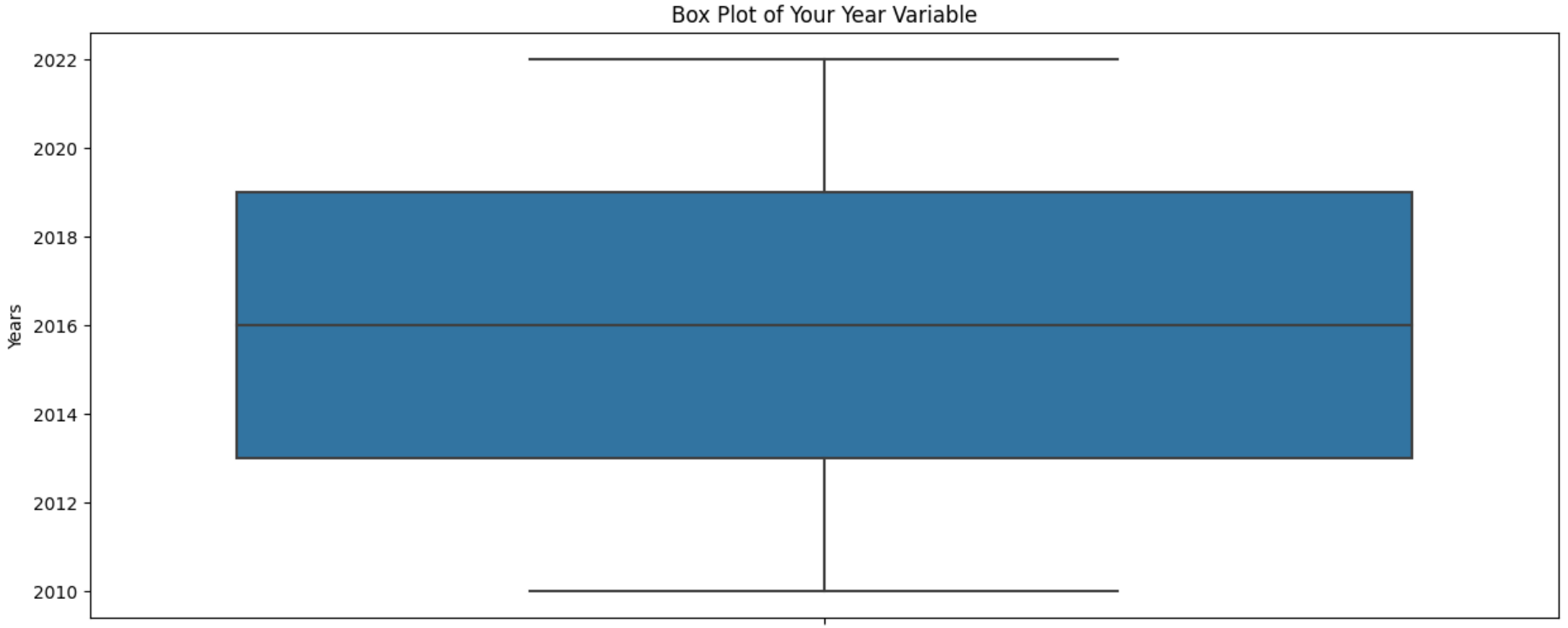
Out[ ]:
```
13
```

`Year` is a numeric variable.

In [ ]:
```
df['Year'].describe()
```

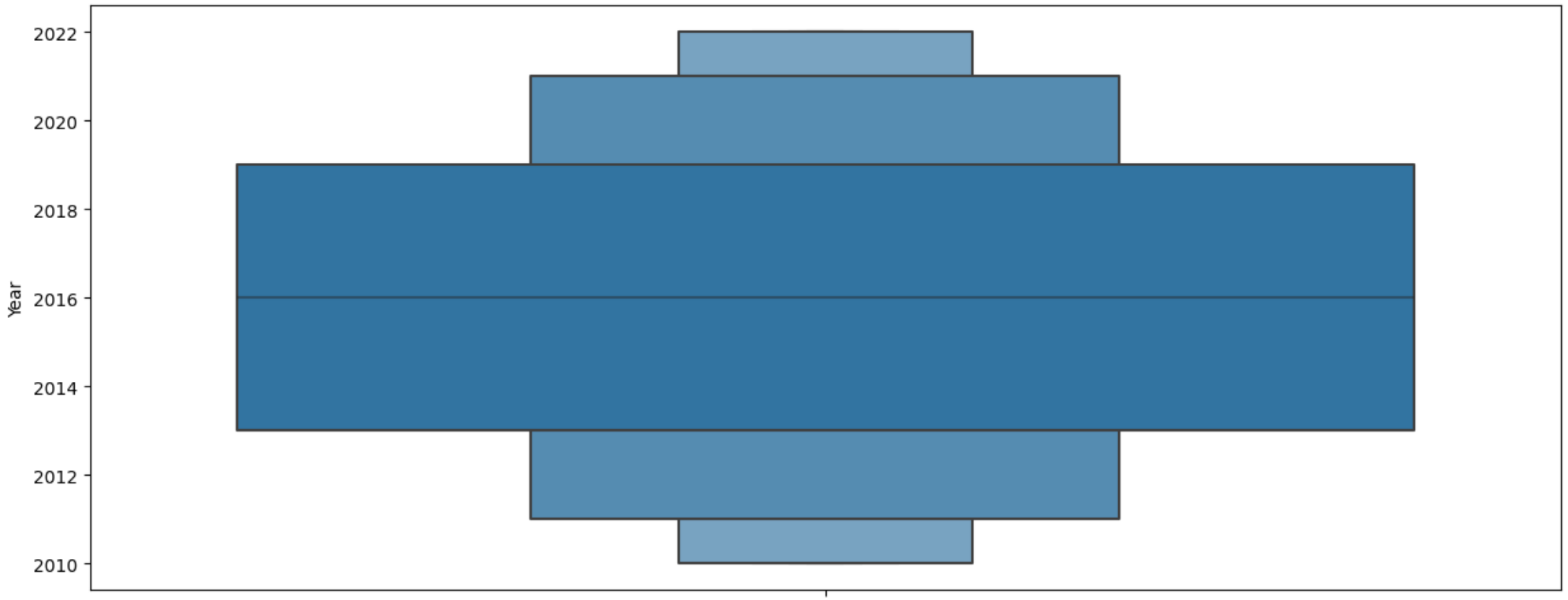
Out[ ]:
```
count     676.000000
mean     2016.000000
std         3.744428
min      2010.000000
25%      2013.000000
50%      2016.000000
75%      2019.000000
max      2022.000000
Name: Year, dtype: float64
```

In [ ]:
```
#boxplot
sns.boxplot(data=df, y='Year')
plt.ylabel("Years")
plt.title("Box Plot of Your Year Variable")
plt.show()
```



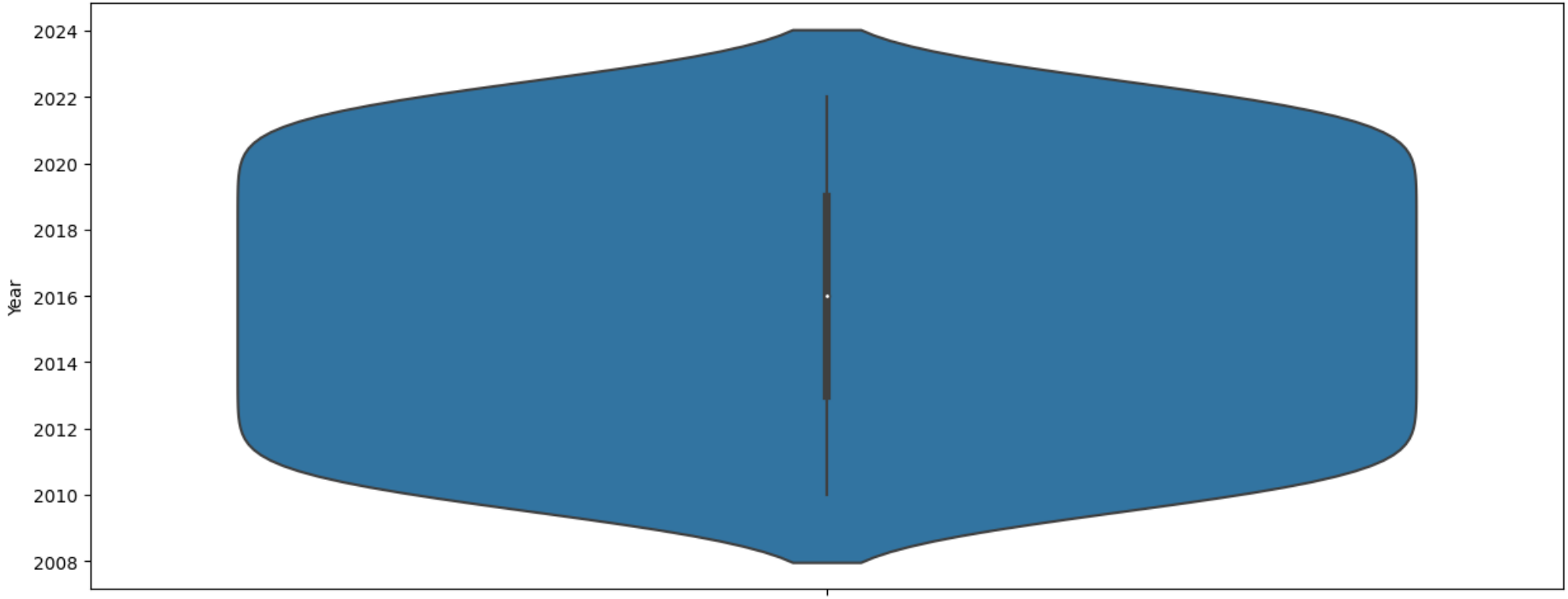
In [ ]:
```
#boxenplot
sns.boxenplot(data=df, y='Year')
```

Out[ ]: <AxesSubplot:ylabel='Year'>



In [ ]:
```
#violinplot
sns.violinplot(data=df, y='Year')
plt.figure(figsize=(6,8))
```

Out[ ]: <Figure size 600x800 with 0 Axes>




```
<Figure size 600x800 with 0 Axes>
```

Important points

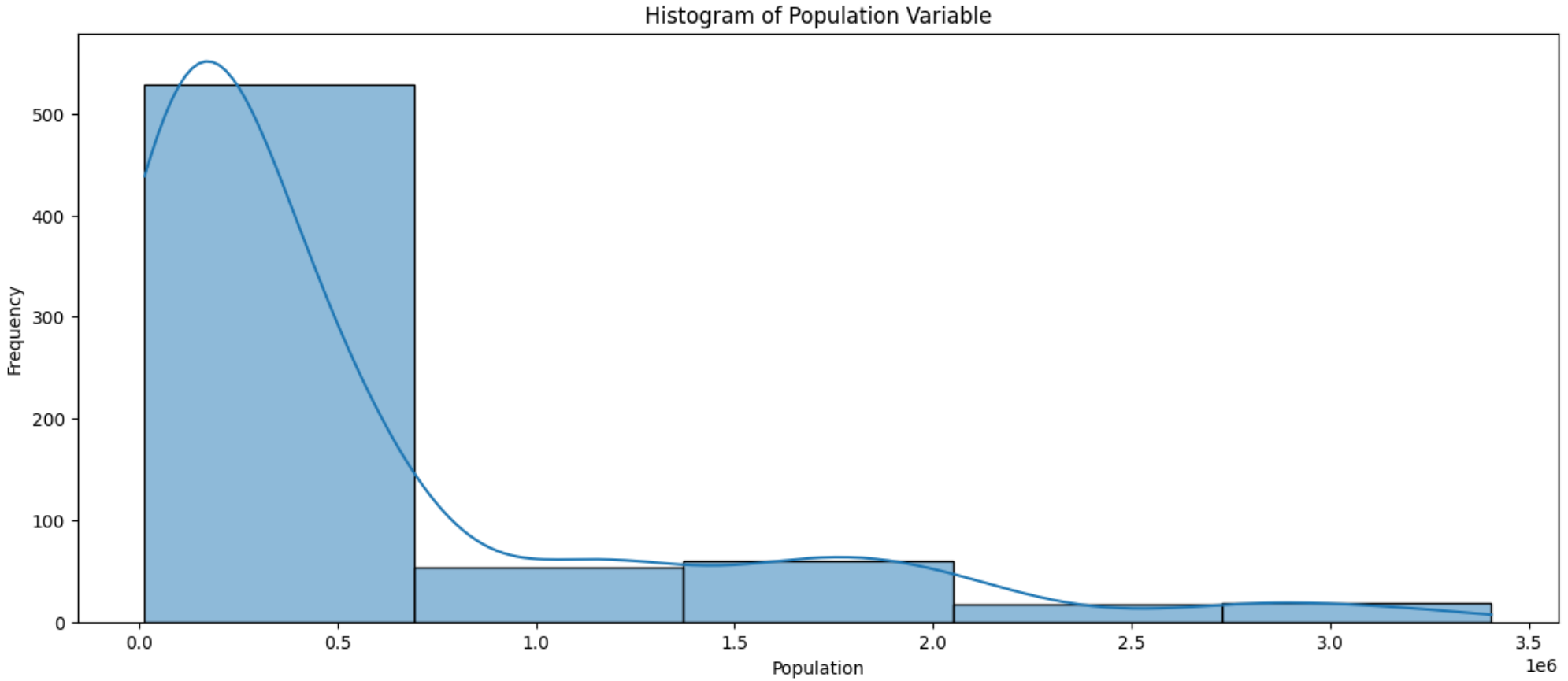
- The above three plots(`Boxplot, boxenplot and Violinplot`) shows us the distribution of a numeric column.
- It can help us to visualize outliers of a particular column

Explore `Population` Variable

In [ ]:
```
df['Population'].describe()
```

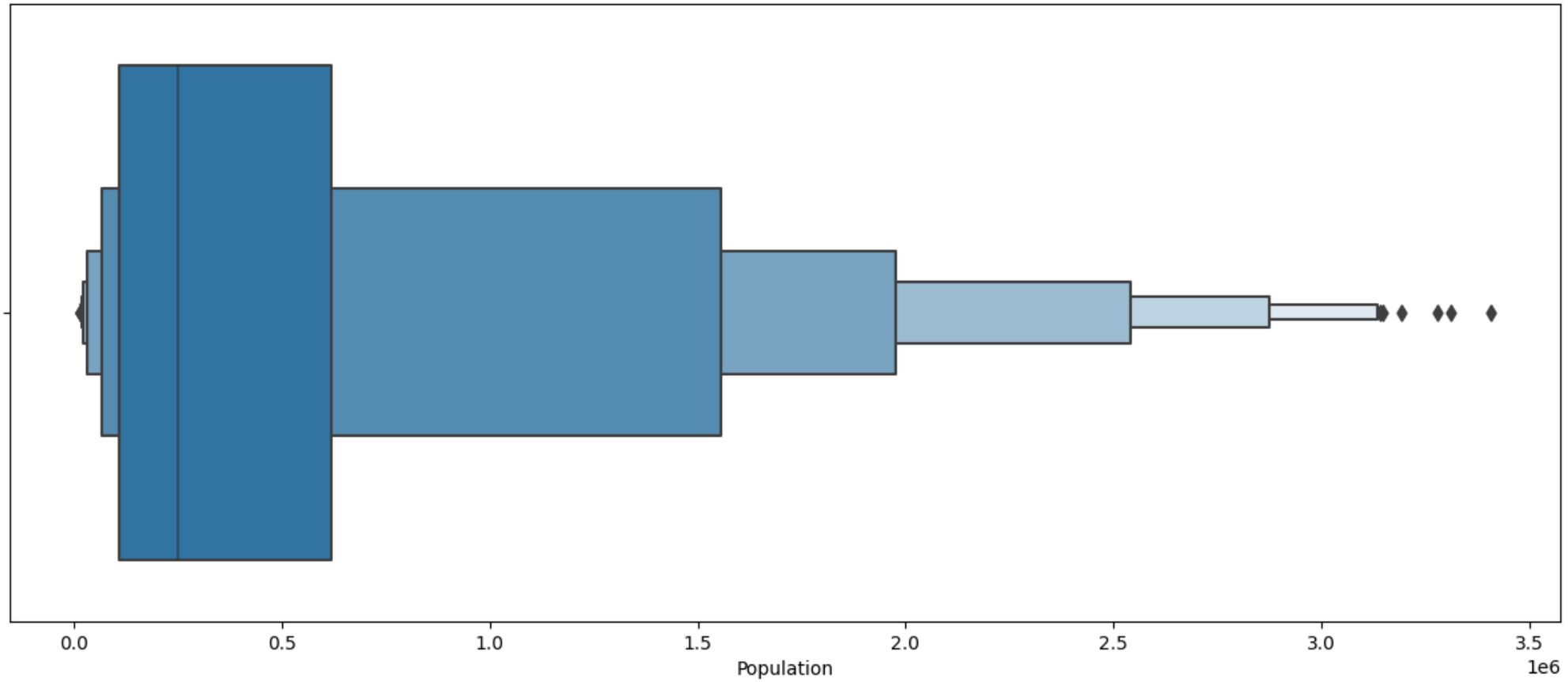
Out[ ]:
```
count    6.760000e+02
mean     5.587169e+05
std      7.062714e+05
min      1.430400e+04
25%      1.098975e+05
50%      2.495590e+05
75%      6.180745e+05
max      3.406281e+06
Name: Population, dtype: float64
```

In [ ]:
```
#Histogram
sns.histplot(data=df, x='Population', kde=True, bins= 5)
plt.xlabel("Population")
plt.ylabel("Frequency")
plt.title("Histogram of Population Variable")
plt.show()
```



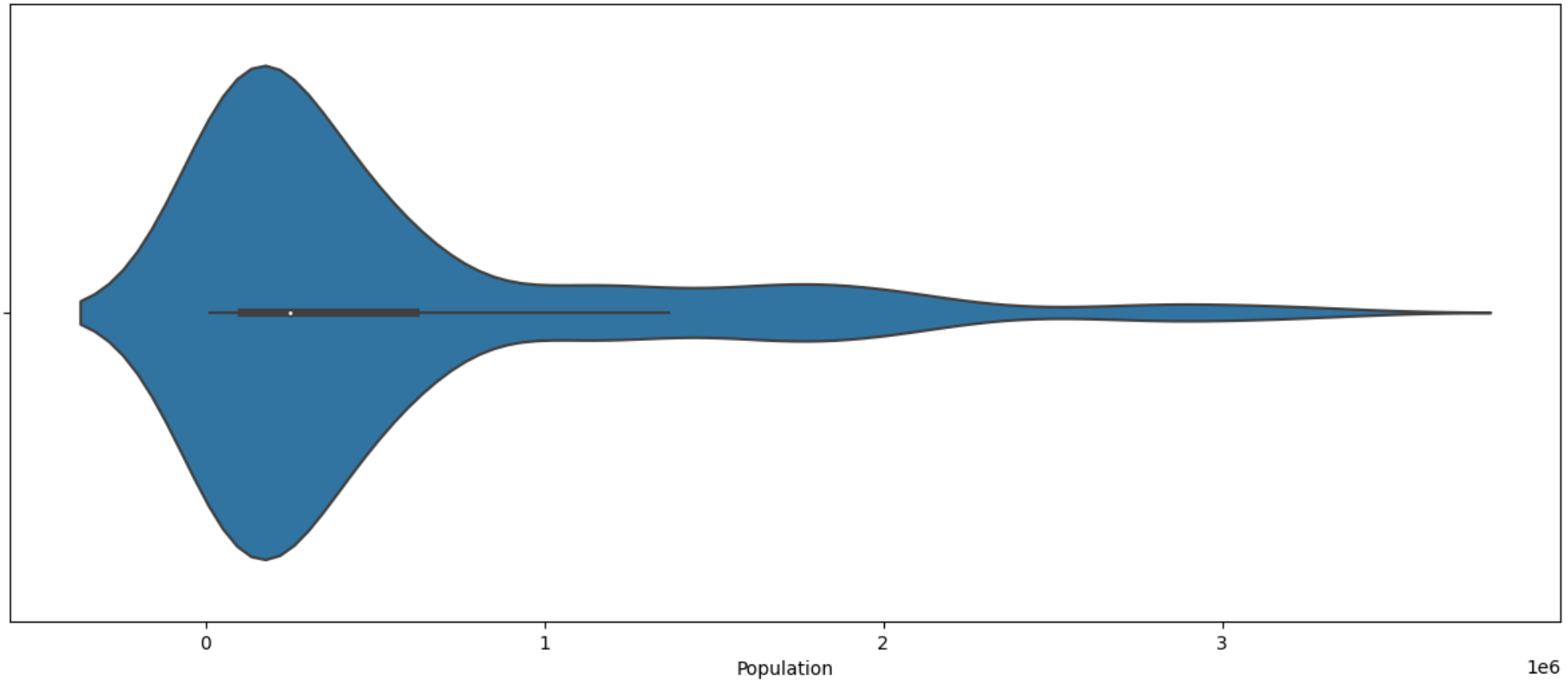
In [ ]:
```
#boxplot
sns.boxenplot(df, x= 'Population')
```

Out[ ]:
```
<AxesSubplot:xlabel='Population'>
```

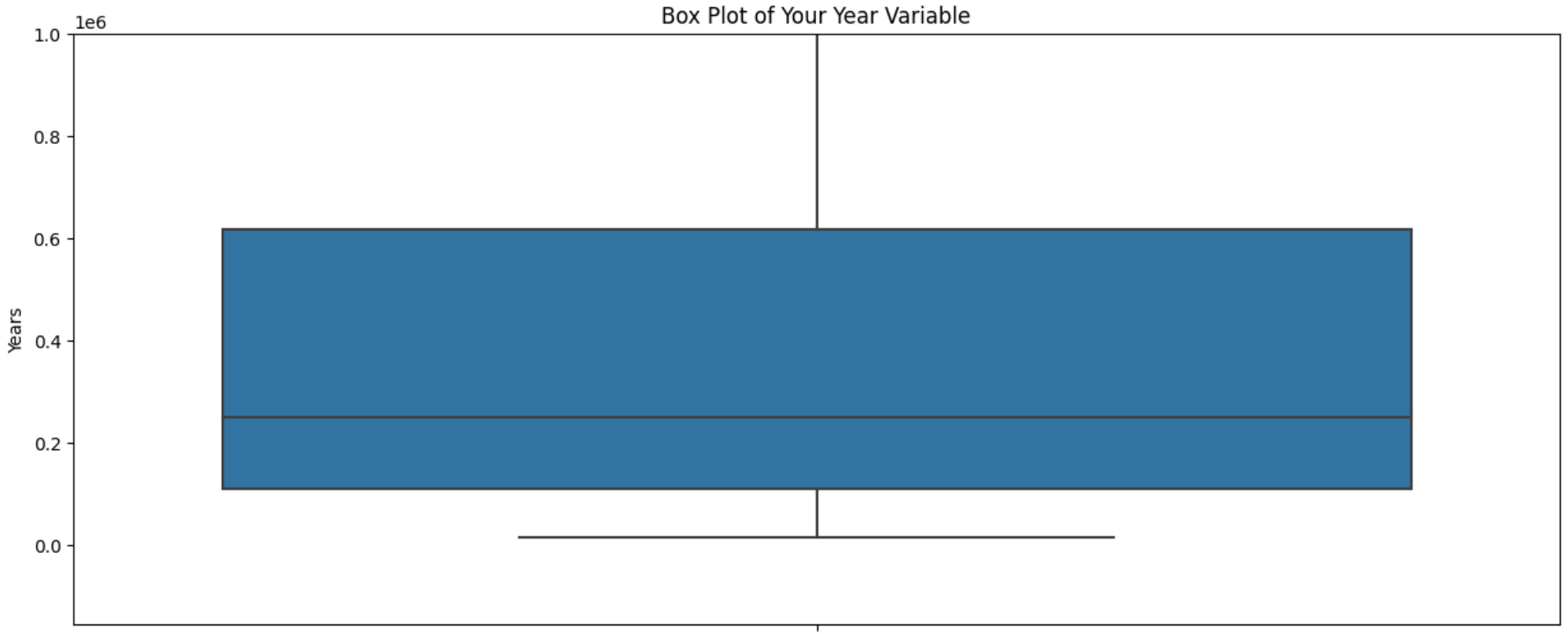


In [ ]:
```
#violinplot
sns.violinplot(df, x= 'Population')
```

Out[ ]:
```
<AxesSubplot:xlabel='Population'>
```



In [ ]:
```
#boxplot
sns.boxplot(data=df, y='Population')
plt.ylim(top= 1000000)
plt.ylabel("Years")
plt.title("Box Plot of Your Year Variable")
plt.show()
```



In [ ]:
```
# Checking skewness of the data
df['Population'].skew()
```

Out[ ]:
```
1.8921844651674402
```

- The skewness of the `Population` column is 1.89
- This means that the distribution of the population is slightly right-skewed.

Biavariate Analysis

Catagorical variables in the dataset

In [ ]:
```
categorical = [var for var in df.columns if df[var].dtype=='O']

print('There are {} categorical variables\n'.format(len(categorical)))
```

```
print('The categorical variables are :', categorical)
```

```
There are 3 categorical variables

The categorical variables are : ['Gender', 'Nationality', 'Region']
```

In [ ]:
```
# Lets take a look
df[categorical].head()
```

Out[ ]:

	Gender	Nationality	Region
0	Female	Non Saudi	Al Bahah
1	Female	Non Saudi	Al Bahah
2	Female	Non Saudi	Al Bahah
3	Female	Non Saudi	Al Bahah
4	Female	Non Saudi	Al Bahah

Number of labels: Cardinality

The number of labels within a categorical variable is known as **cardinality**. A high number of labels within a variable is known as **high cardinality**. High cardinality may pose some serious problems in the machine learning model. So, I will check for high cardinality.

In [ ]:
```
# check for cardinality in categorical variables

for var in categorical:

    print(var, ' contains ', len(df[var].unique()), ' labels')
```

```
Gender  contains  2  labels
Nationality  contains  2  labels
Region  contains  13  labels
```

Explore problems within categorical variables

1) Missing values in categorical variables

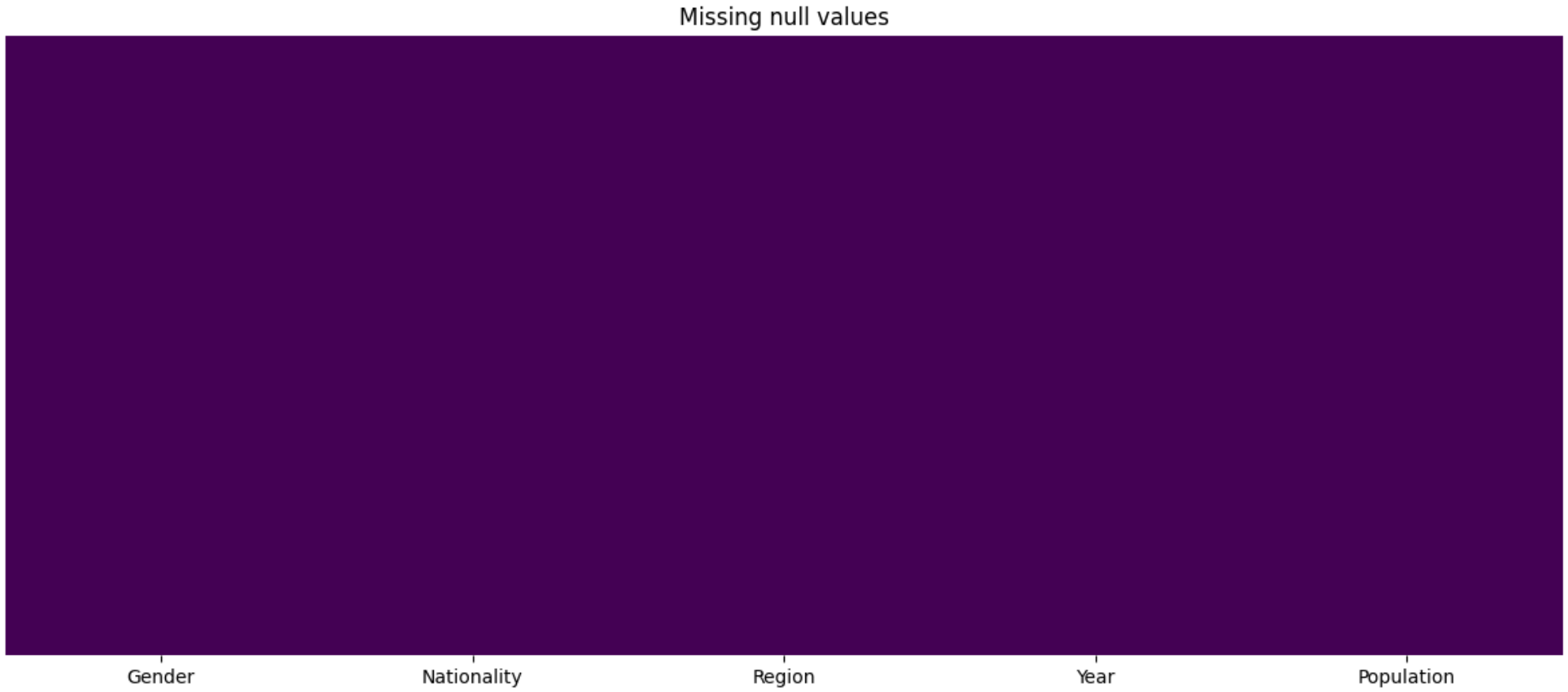
In [ ]:
```
df[categorical].isnull().sum()
```

Out[ ]:
```
Gender         0
Nationality    0
Region         0
dtype: int64
```

2) Plotting the missing values

In [ ]:
```
plt.rcParams['figure.figsize'] = (15,6)
sns.heatmap(df.isnull(),yticklabels = False, cbar = False , cmap = 'viridis')
plt.title("Missing null values")
```

Out[ ]:
```
Text(0.5, 1.0, 'Missing null values')
```



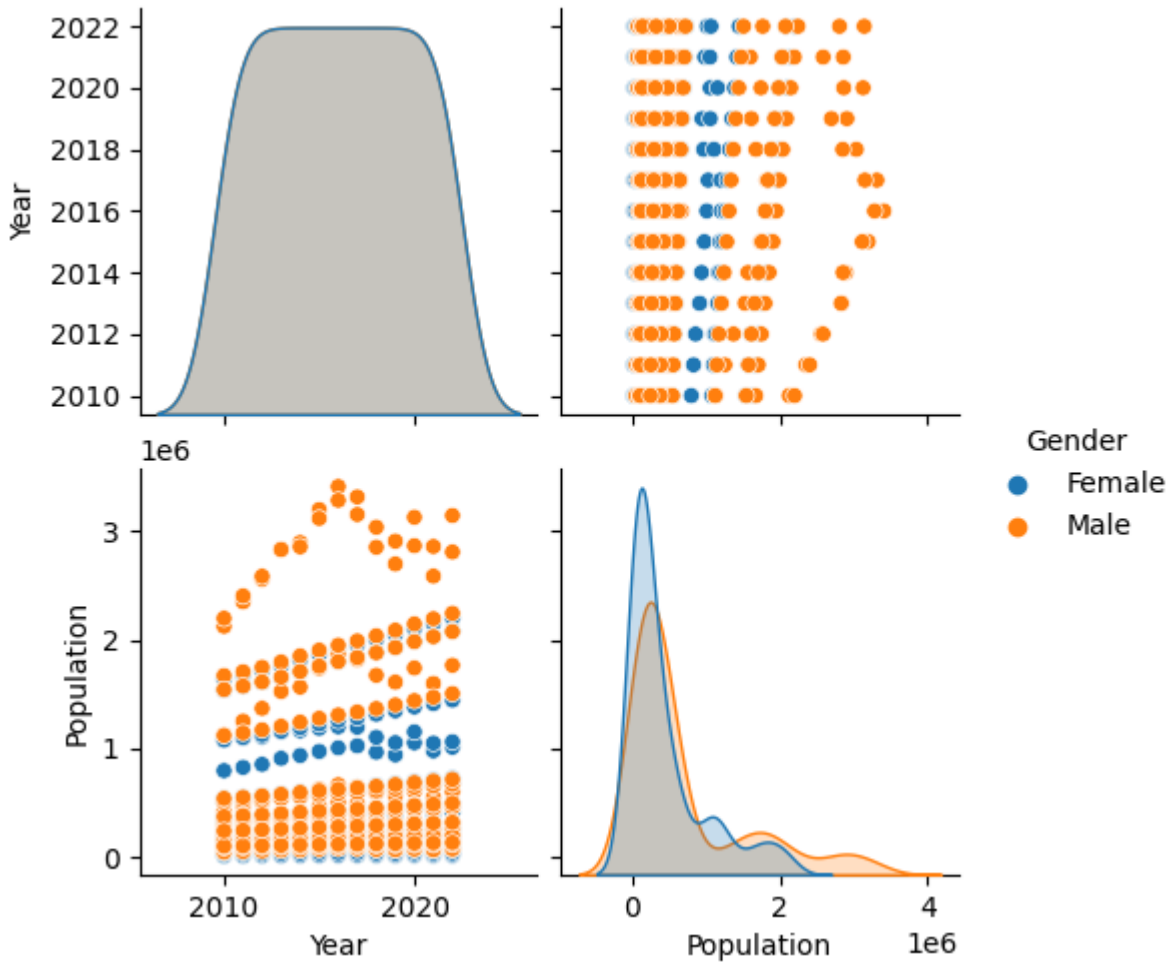
3) Data Distribution of the Variables

In [ ]:
```
import seaborn as sns

# make sure that the variable 'categorical' is defined in a previous cell
```

```
sns.pairplot(df, hue= 'Gender')
```

Out[ ]: <seaborn.axisgrid.PairGrid at 0x22c6b331490>

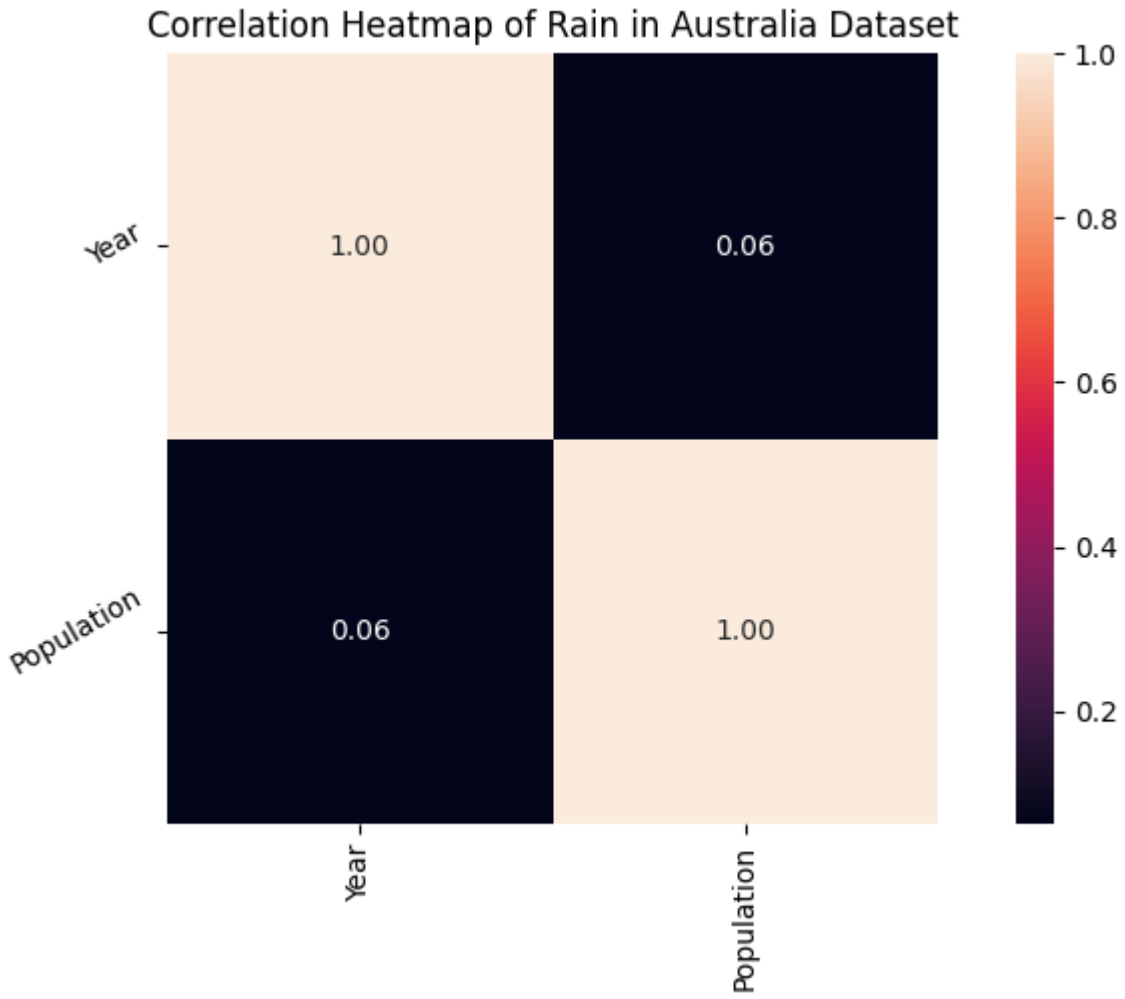


- ## Correlation in the dataset

In [ ]:
```
correlation = df.corr()
```

In [ ]:
```
plt.figure(figsize=(10,5))
plt.title('Correlation Heatmap of Rain in Australia Dataset')
ax = sns.heatmap(correlation, square=True, annot=True, fmt='.2f', linecolor='white')
ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
```

```
ax.set_yticklabels(ax.get_yticklabels(), rotation=30)
plt.show()
```



Import points:

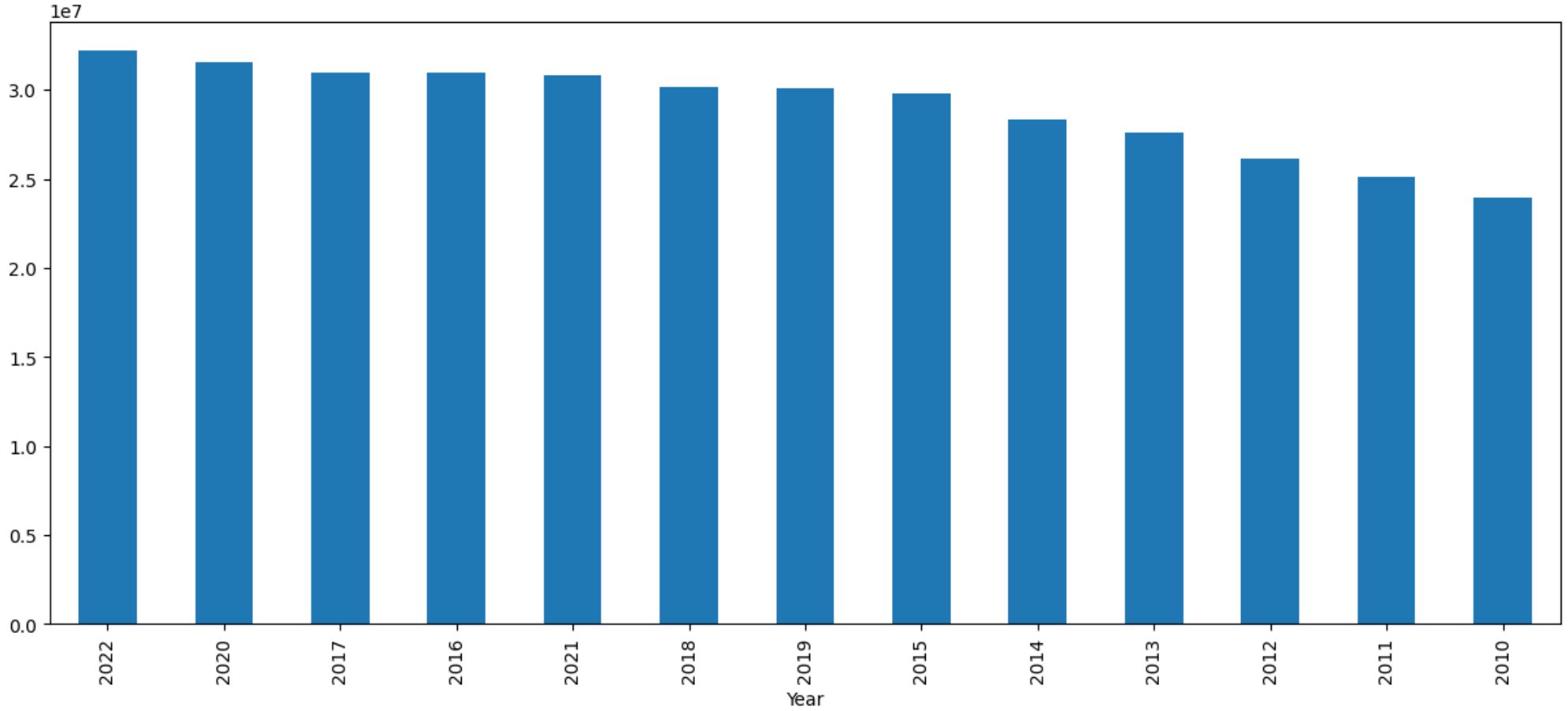
- `Population` and `Year` are positively correlated

Questions

Q1. In which Year the population was the Highiest

In [ ]:
```
df.groupby('Year')['Population'].sum().sort_values(ascending= False).plot(kind= 'bar')
plt.get_backend()
```

Out[ ]:
```
'module://matplotlib_inline.backend_inline'
```



- **Creating new Column : `Total_population_by_year`**

In [ ]:
```
# Creating column('Total_population_by_year')
df['Total_population_by_year'] = df.groupby('Year')['Population'].transform('sum')
df.head()
```

Out[ ]:

	Gender	Nationality	Region	Year	Population	Total_population_by_year
0	Female	Non Saudi	Al Bahah	2010	16209	23978487
1	Female	Non Saudi	Al Bahah	2011	16521	25091867
2	Female	Non Saudi	Al Bahah	2012	16752	26168861
3	Female	Non Saudi	Al Bahah	2013	17508	27624004
4	Female	Non Saudi	Al Bahah	2014	17682	28309273

In [ ]:
```
# Converting 'total_population_by_year' into millions
df['Total_population_by_year']= round(df['Total_population_by_year'] / 1e6, 1)
df.head()
```

Out[ ]:

	Gender	Nationality	Region	Year	Population	Total_population_by_year
0	Female	Non Saudi	Al Bahah	2010	16209	24.0
1	Female	Non Saudi	Al Bahah	2011	16521	25.1
2	Female	Non Saudi	Al Bahah	2012	16752	26.2
3	Female	Non Saudi	Al Bahah	2013	17508	27.6
4	Female	Non Saudi	Al Bahah	2014	17682	28.3

In [ ]:
```
# Renaming the column
df.rename(columns= {"Total_population_by_year": "Year_population(million)"}, inplace= True)
df.head()
```

Out[ ]:

	Gender	Nationality	Region	Year	Population	Year_population(million)
0	Female	Non Saudi	Al Bahah	2010	16209	24.0
1	Female	Non Saudi	Al Bahah	2011	16521	25.1
2	Female	Non Saudi	Al Bahah	2012	16752	26.2
3	Female	Non Saudi	Al Bahah	2013	17508	27.6
4	Female	Non Saudi	Al Bahah	2014	17682	28.3

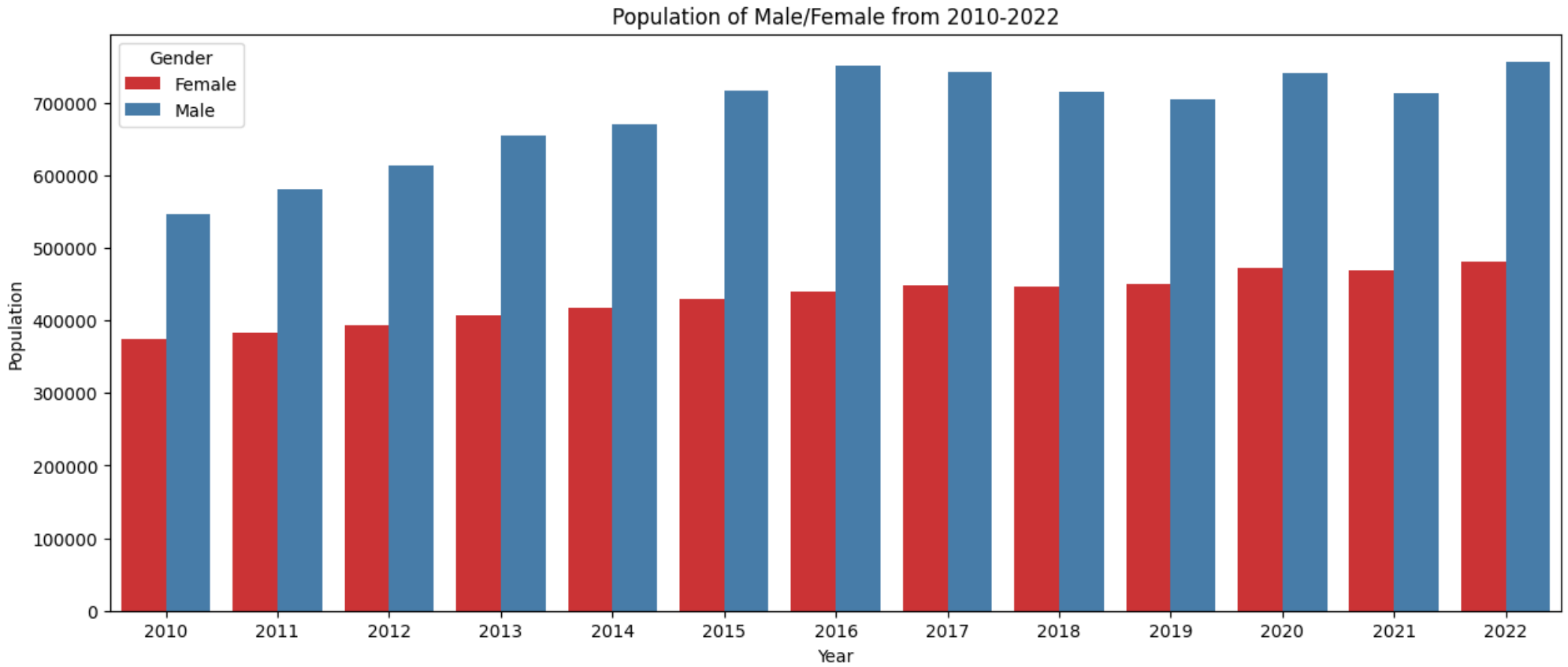
Important points

- **Division by 1e6:** The expression `df['Total_population_by_year'] / 1e6` divides the values in the 'Total_population_by_year' column by 1,000,000, effectively converting the population counts to millions.
- **Rounding to One Decimal Place:** The `round()` function is used to round the division result to one decimal place. This ensures the population values are presented with a reduced precision, providing a clearer view of the data in millions.
- **Assignment to a New Column:** The transformed values are then assigned back to the 'Total_population_by_year' column, replacing the original population counts with their corresponding values in millions, rounded to one decimal place.

Q2. What is the Popoulatipn of Male vs female in each Year

In [ ]:
```
sns.barplot(data=df, x='Year', y='Population', hue='Gender',palette='Set1', errorbar= ('ci', 0))
plt.title("Population of Male/Female from 2010-2022")
```

Out[ ]:
```
Text(0.5, 1.0, 'Population of Male/Female from 2010-2022')
```



In [ ]:
```
df.groupby(df['Gender']=="Male")["Population"].count()
```

Out[ ]:
```
Gender
False    338
True     338
Name: Population, dtype: int64
```

Question 3: What is the average population growth rate per year?

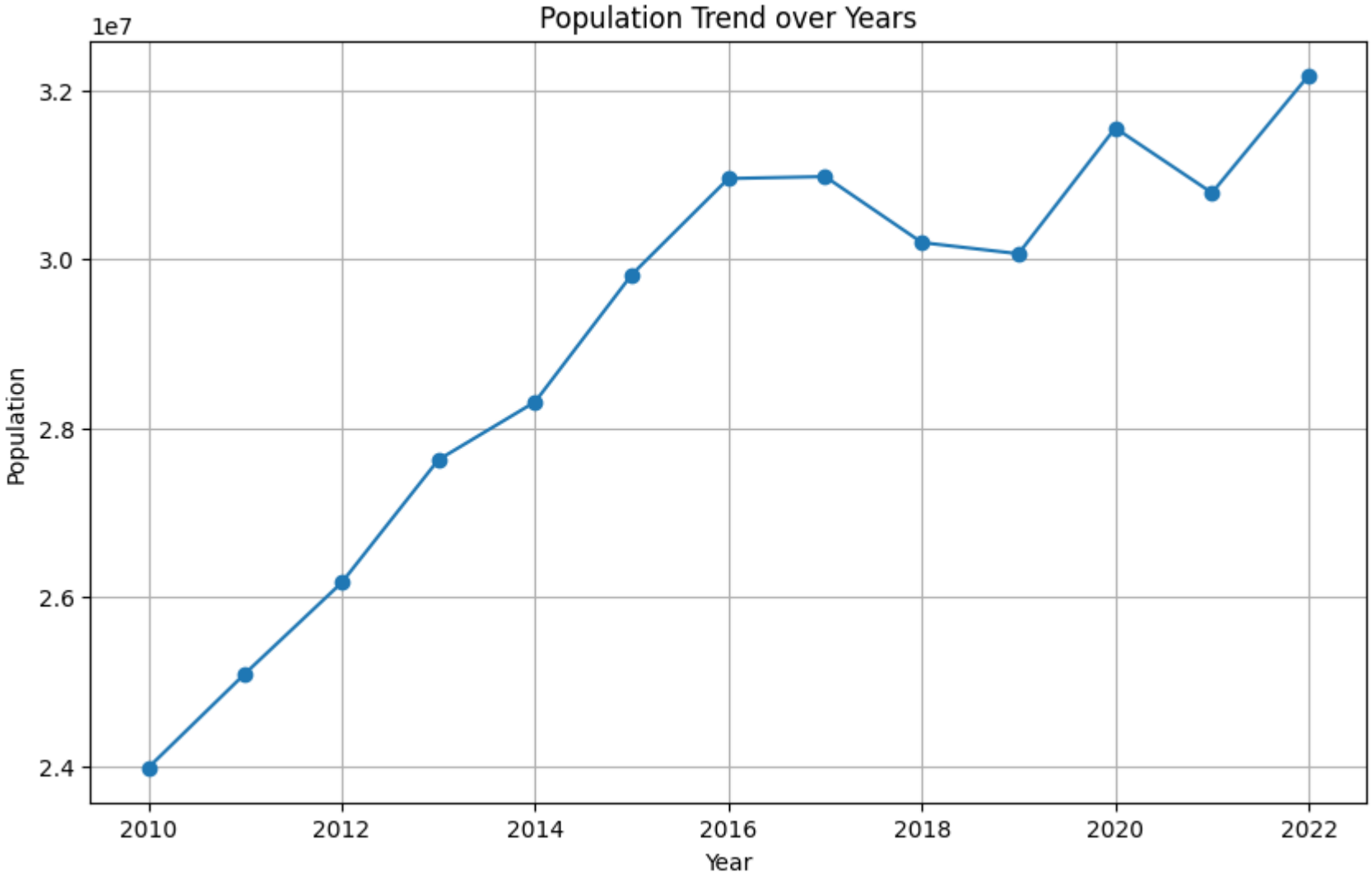
In [ ]:
```
population_by_year = df.groupby('Year')['Population'].sum()
average_growth_rate = (population_by_year.iloc[-1] - population_by_year.iloc[0]) / (population_by_year.index[-1] - population_by_year.index
print("Average population growth rate per year:", average_growth_rate)
```

```
Average population growth rate per year: 683061.4166666666
```

Question 4: Population Trends over Years

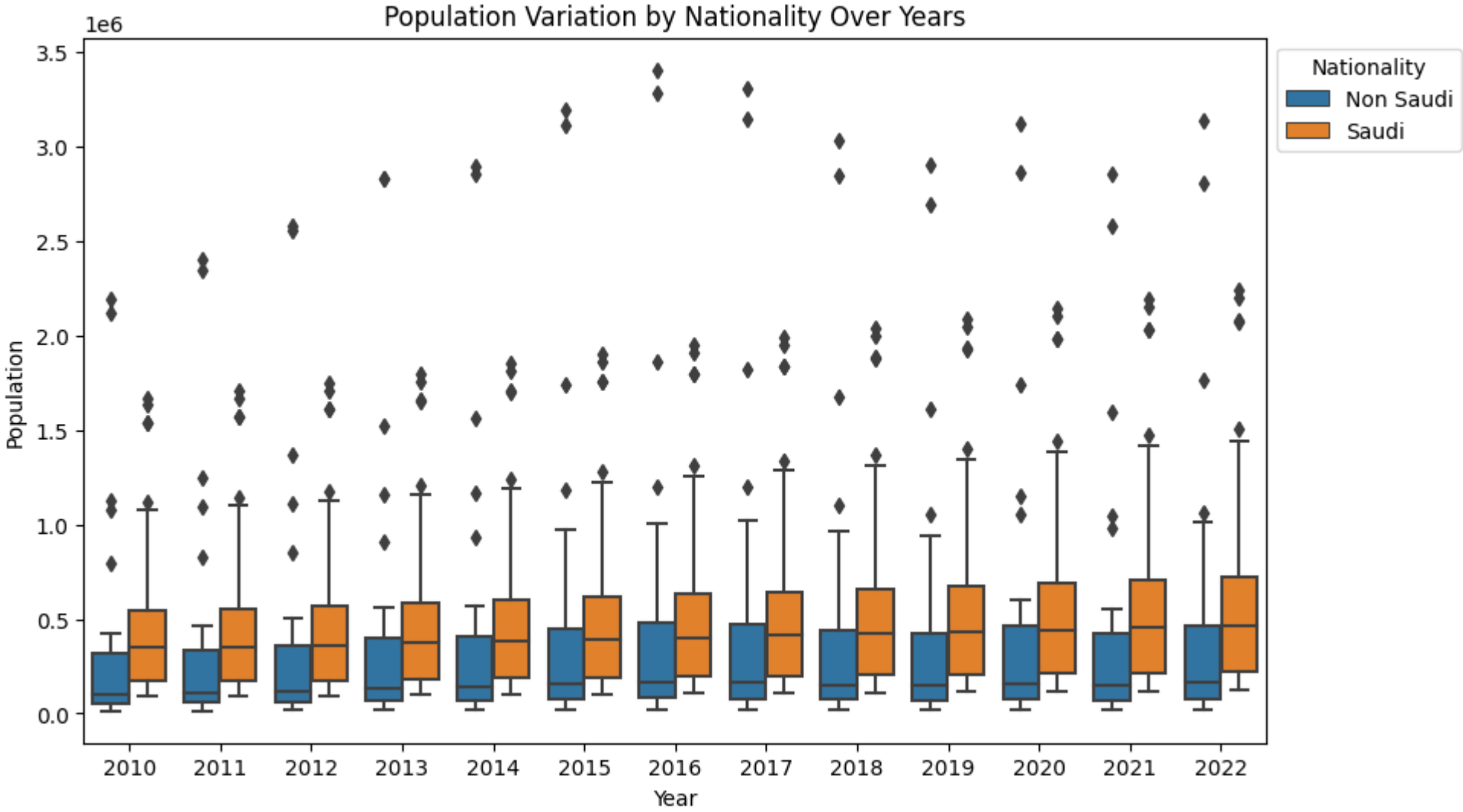
In [ ]:
```
population_over_years = df.groupby('Year')['Population'].sum()

plt.figure(figsize=(10, 6))
population_over_years.plot(kind='line', marker='o')
plt.title('Population Trend over Years')
plt.xlabel('Year')
plt.ylabel('Population')
plt.grid(True)
plt.show()
```



Question 5: What is the Population Variation by Nationality over the years?

In [ ]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Year', y='Population', hue='Nationality')
plt.title('Population Variation by Nationality Over Years')
plt.xlabel('Year')
plt.ylabel('Population')
plt.legend(title='Nationality', bbox_to_anchor=(1, 1))
plt.show()
```



Question 6: Which Region has the highiest population

In [ ]:
```
px.bar( df.groupby('Region')['Population'].max().sort_values(ascending= False))
```

Question 7: What is the Male vs female ratio in the population

In [ ]:
```
px.bar(df.groupby('Gender')['Population'].max().sort_values(ascending= False))
```

In [ ]:
```
#plotting the data
px.bar(df, x= "Year", y= 'Population', color= 'Nationality')

# The code generates a bar plot using the specified DataFrame and columns, providing a visual representation of how population counts
# in thousands vary over different years, segmented by nationality.
# Each bar in the plot represents the population count for a specific nationality category in a given year.
```

- ## Population division in the region of Jazan

In [ ]:
```
px.bar(df.groupby(df['Region']== 'Jazan')['Population'].mean())
```

In [ ]:
```
fig = px.bar(df, x='Region', y='Population', title='Population by Region',
             labels={'Population': 'Total Population', 'Region': 'Region'},
             color='Region')

fig.update_layout(barmode='stack')
fig.show()
```

Two libraries for Quick EDA

- I will use two Libraries for quick analysis of the dataset
- The two libraries are `Skimpy` and `Dataprep`

(1) Skimpy

In [ ]:
```
import skimpy
from skimpy import skim
skim(df)
```

skimpy summary

Data Summary

dataframe	Values
Number of rows	676
Number of columns	6

Data Types

Column Type	Count
string	3
int32	2
float64	1

number

column_name	NA	NA %	mean	sd	p0	p25	p75	p100	hist
Year	0	0	2000	3.7	2000	2000	2000	2000	
Population	0	0	560000	710000	14000	110000	620000	3400000	
Year_population(mill	0	0	29	2.5	24	28	31	32	

string

column_name	NA	NA %	words per row	total words
Gender	0	0	1	680
Nationality	0	0	1	680
Region	0	0	1	680

End

(2) Dataprep

In [ ]:
```
!pip install dataprep
```

In [ ]:
```
from dataprep.eda import create_report
create_report(df)
```

```
  0%|          | 0/884 [00:00<?, ?it/s]
c:\Users\Tariq Laptops\AppData\Local\Programs\Python\Python39\lib\site-packages\dask\core.py:119: RuntimeWarning:

invalid value encountered in divide

c:\Users\Tariq Laptops\AppData\Local\Programs\Python\Python39\lib\site-packages\dataprep\eda\distribution\render.py:274: FutureWarning:

The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat instead.
```

Out[ ]:

Overview

Dataset Statistics

Number of Variables	6
Number of Rows	676
Missing Cells	0
Missing Cells (%)	0.0%
Duplicate Rows	0
Duplicate Rows (%)	0.0%
Total Size in Memory	143.5 KB
Average Row Size in Memory	217.3 B
Variable Types	Categorical: 3 Numerical: 3

Dataset Insights

`Population`	is s
`Year_population(million)`	is s

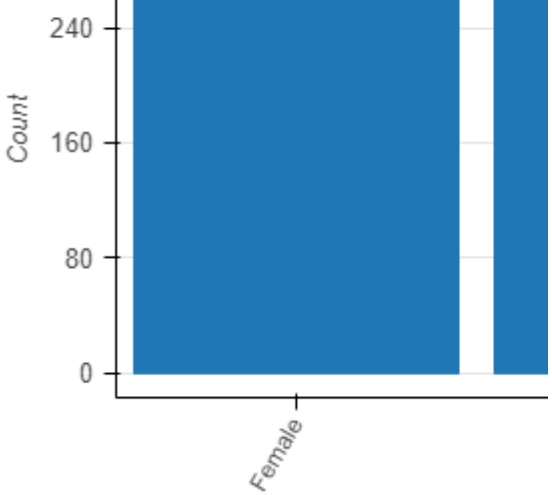
Variables

Sort by Feature order ☐ Reverse order

Gender
categorical

Show Details

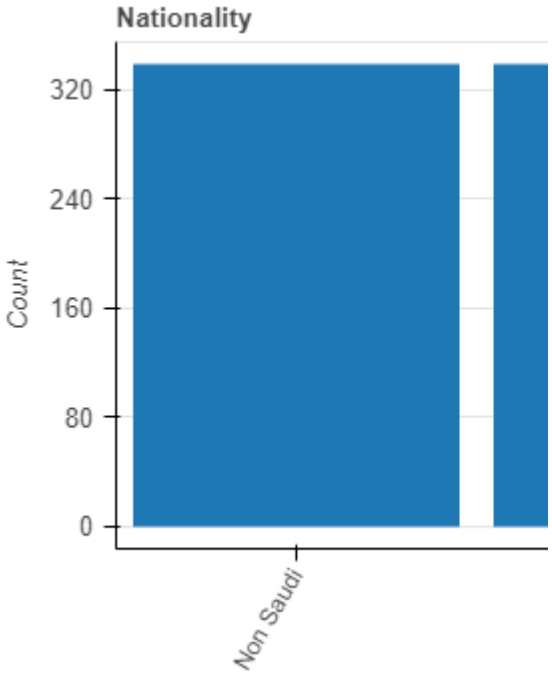
Approximate Distinct Count	2
Approximate Unique (%)	0.3%
Missing	0
Missing (%)	0.0%
Memory Size	47320



Nationality
categorical

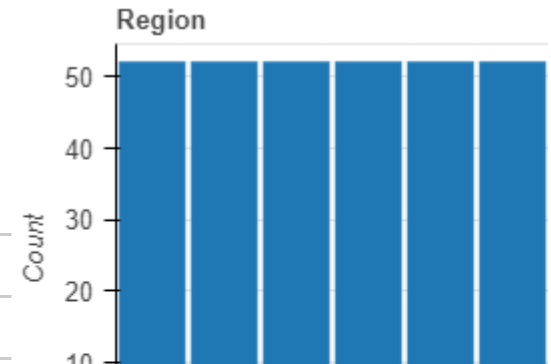
Show Details

Approximate Distinct Count	2
Approximate Unique (%)	0.3%
Missing	0
Missing (%)	0.0%
Memory Size	48672



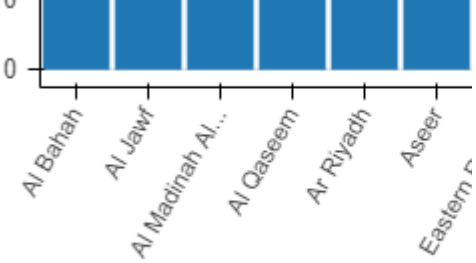
Region
categorical

Approximate Distinct Count	13
Approximate Unique (%)	1.9%
Missing	0



Show Details

Missing (%)	0.0%
Memory Size	50804



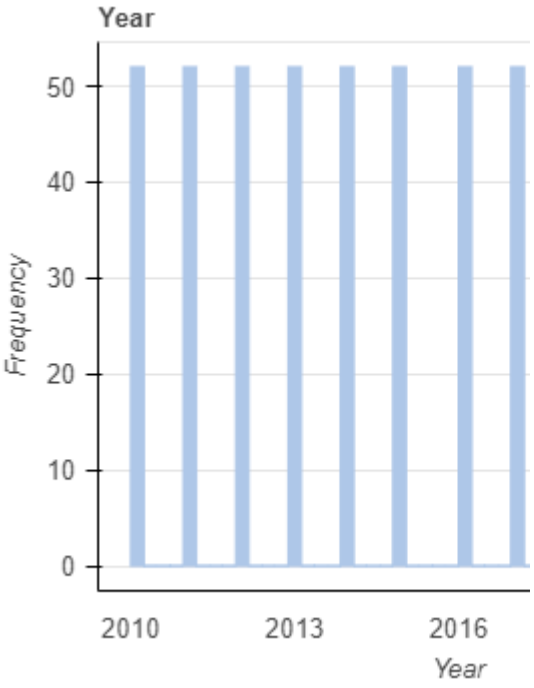
Top 10 of 13 Re

Year
numerical

Show Details

Approximate Distinct Count	13
Approximate Unique (%)	1.9%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%
Memory Size	10816

Mean	2016
Minimum	2010
Maximum	2022
Zeros	0
Zeros (%)	0.0%
Negatives	0
Negatives (%)	0.0%

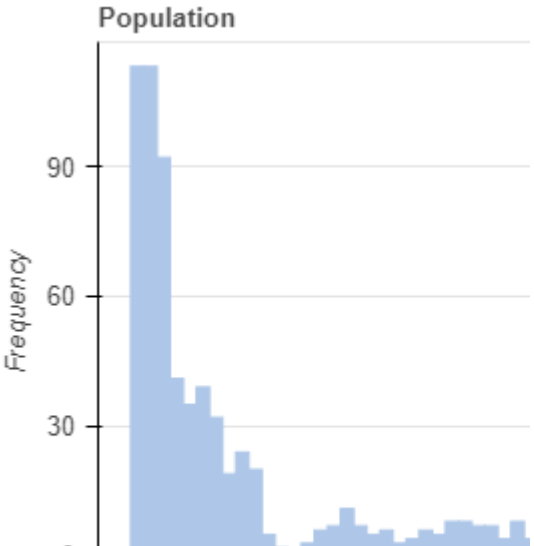


Population
numerical

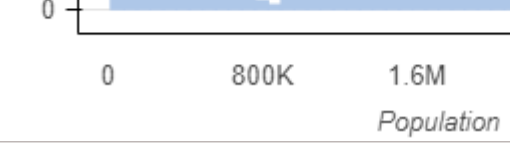
Show Details

Approximate Distinct Count	675
Approximate Unique (%)	99.9%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Memory Size	10816
Mean	558716.8994
Minimum	14304
Maximum	3406281
Zeros	0
Zeros (%)	0.0%
Negatives	0



Negatives (%)	0.0%

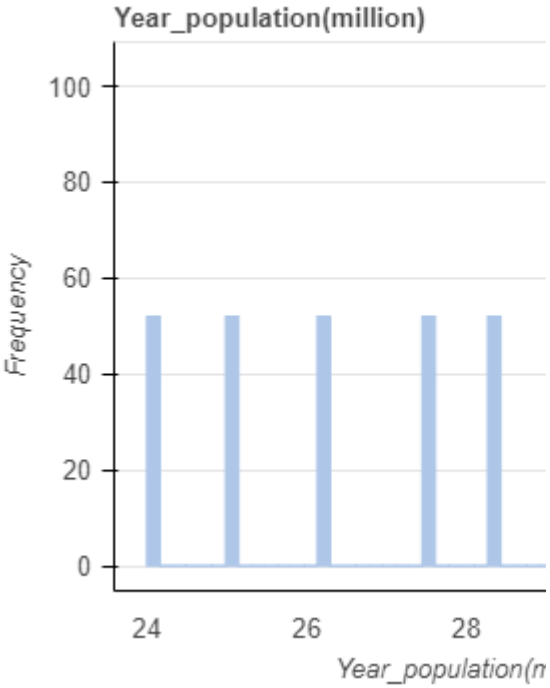


Year_population(million)
numerical

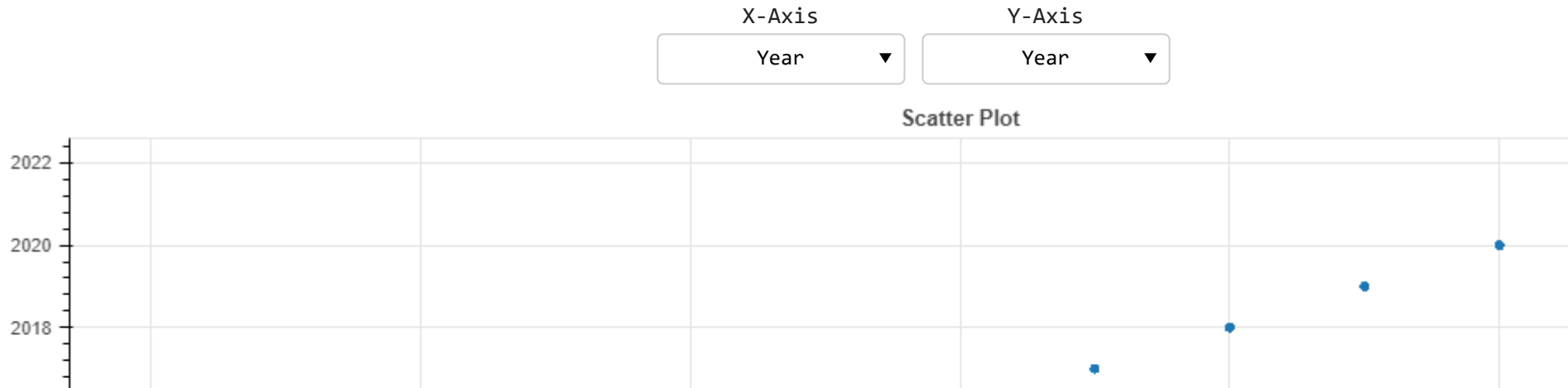
Show Details

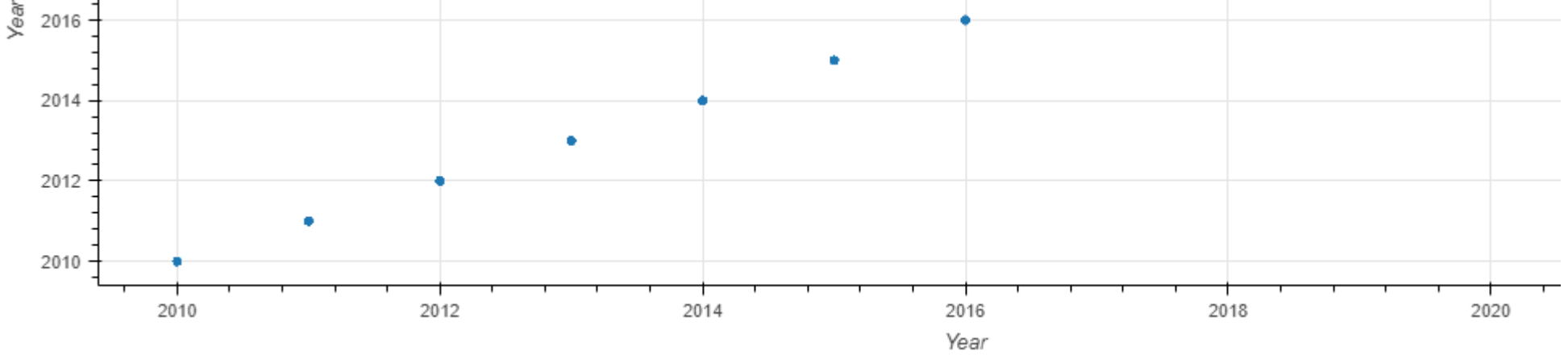
Approximate Distinct Count	12
Approximate Unique (%)	1.8%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%
Memory Size	10816

Mean	29.0692
Minimum	24
Maximum	32.2
Zeros	0
Zeros (%)	0.0%
Negatives	0
Negatives (%)	0.0%

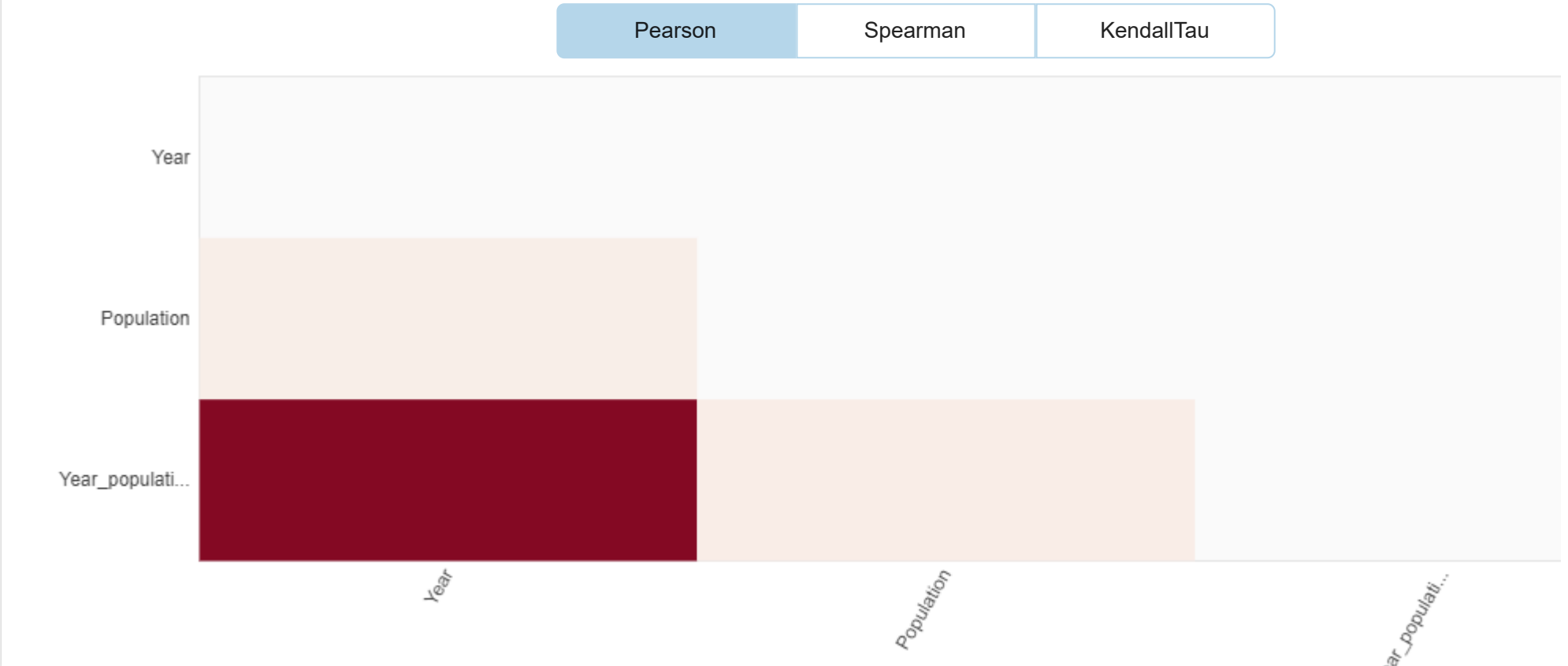


Interactions





Correlations



Missing Values

Bar Chart | Spectrum | Heat Map | Dendrogram