Hands-on Activity 10.1 Data Analysis using Python

Names:

Carigo, Naira Jezreel B.

Corpuz, Micki Laurren B.

Section: CPE22S3

Date: 08 May 2025

Instructor: Engr. Roman Richard

Intended Learning Outcomes

1.Perform descriptive and correlation analysis to to analyze the dataset.

2.Interpret the results of descriptive and correlation analysis

Resources

- Personal Computer
- Jupyter Notebook
- Internet Connection

Instructions

1. Gather a dataset regarding your identified problem for the ASEAN Data Science Explorer. Make sure that the dataset includes multiple variables.

> Our duo, Dare2Pair, gathered verified dataset about SDG 12, "Responsible Consumption and Production," specifically target 12.3: Halve per capita global food waste at the retail and consumer levels and reduce food losses along production and supply chains.

> These datasets were obtained from the website of United Nations Department of Economic and Social Affairs.

2. Load the dataset into pandas dataframe.

I. Population --- Food waste

```
import pandas as pd


# Load the dataset about population
pop = pd.read_excel("Population.xlsx")
pop.head()
```

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unna
0	Table 1.1. Number of Population in ASEAN, 2013...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Next steps: Generate code with pop | View recommended plots | New interactive sheet

```
# Load the dataset about generated food waste (in tonnes)
FW = pd.read_excel("Food waste (in tonnes).xlsx")
FW.head()
```

	Goal	Target	Indicator	SeriesCode	SeriesDescription	GeoAreaCode	GeoAreaName
0	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
1	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
2	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
3	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
4	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	2	Africa

5 rows × 29 columns

II. Poverty rate --- Food Insecurity

```
# Load the dataset about proportion of population below international poverty line (%)
pop3 = pd.read_excel("SI_POV_DAY1.xlsx")
pop3.head()
```

	Goal	Target	Indicator	SeriesCode	SeriesDescription	GeoAreaCode	GeoAreaName	
0	1	1.1	1.1.1	SI_POV_DAY1	Proportion of population below international p...	2	Africa	A
1	1	1.1	1.1.1	SI_POV_DAY1	Proportion of population below international p...	8	Albania	
2	1	1.1	1.1.1	SI_POV_DAY1	Proportion of population below international p...	8	Albania	
3	1	1.1	1.1.1	SI_POV_DAY1	Proportion of population below international p...	8	Albania	
4	1	1.1	1.1.1	SI_POV_DAY1	Proportion of population below international p...	8	Albania	A

5 rows × 36 columns

```
# Load the dataset about population of severe food insecurity
pop2 = pd.read_excel("Food waste (in tonnes).xlsx")
pop2.head()
```

	Goal	Target	Indicator	SeriesCode	SeriesDescription	GeoAreaCode	GeoAreaName
0	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
1	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
2	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
3	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	4	Afghanistan
4	12	12.3	12.3.1	AG_FOOD_WST	Food waste (Tonnes)	2	Africa

5 rows × 29 columns

3. Prepare the data by applying appropriate data preprocessing techniques.

Population --- Food Waste

Population

a. Take a snapshot of the population over the last 10 years (i.e. 2013 to 2022)

```
pop_copy = pop.copy()
# Get an overview about the data
pop_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18 entries, 0 to 17
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   15 non-null     object
 1   Unnamed: 1   12 non-null     float64
 2   Unnamed: 2   12 non-null     float64
 3   Unnamed: 3   12 non-null     float64
 4   Unnamed: 4   12 non-null     float64
 5   Unnamed: 5   12 non-null     float64
 6   Unnamed: 6   12 non-null     float64
 7   Unnamed: 7   12 non-null     float64
 8   Unnamed: 8   12 non-null     float64
 9   Unnamed: 9   12 non-null     float64
 10  Unnamed: 10  13 non-null     object
 11  Unnamed: 11  1 non-null      object
dtypes: float64(9), object(3)
memory usage: 1.8+ KB
```

```
# # Step 1: Build the renaming dictionary
# first column is 'Country', rest are unnamed columns like 'Unnamed: 1', 'Unnamed: 2', etc.
num_years = pop_copy.shape[1] - 1  # exclude the first column
years = list(range(2013, 2013 + num_years))

# Step 2: Make a dictionary to rename columns properly
# Rename 'Unnamed: 0' to 'Country', and rest to the corresponding year
new_column_names = {
    f'Unnamed: {i}': ('Country' if i == 0 else str(years[i - 1]))
    for i in range(pop_copy.shape[1])
}

# Step 3: Rename the columns in the DataFrame using our dictionary
pop_copy = pop_copy.rename(columns=new_column_names)

# Step 4: Convert all year columns to numeric values
# This makes sure numbers are treated as numbers, not strings
for year in list(new_column_names.values())[1:]:  # skip 'Country'
    pop_copy[year] = pd.to_numeric(pop_copy[year], errors='coerce')

# Step 5: Drop unnecessary rows from top and bottom (first 4 and last 4)
rows_to_remove = list(range(0, 4)) + list(range(len(pop_copy) - 4, len(pop_copy)))
pop_copy = pop_copy.drop(index=rows_to_remove).reset_index(drop=True)

# Step 6: Drop the column for 2023 as it has "NO" values
if "2023" in pop_copy.columns:
    pop_copy = pop_copy.drop(columns="2023")

# Step 7: Show the first few rows to check our cleaned-up DataFrame
pop_copy.head()
```

	Country	2013	2014	2015	2016	2017	2018
0	Brunei Darussalam	406.200	411.900000	412.400	417.256	421.300	442.400
1	Cambodia	14676.590	14932.301000	15191.739	15453.921	15717.674	15981.798
2	Indonesia	248818.090	252164.786000	255587.900	258496.500	261355.500	264161.700
3	Lao PDR	6644.018	6809.053838	6671.680	6787.007	6900.846	7012.995

Next steps: Generate code with pop_copy | View recommended plots | New interactive sheet

```
pop_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 11 columns):
```

```
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Country  10 non-null     object
 1   2013     10 non-null     float64
 2   2014     10 non-null     float64
 3   2015     10 non-null     float64
 4   2016     10 non-null     float64
 5   2017     10 non-null     float64
 6   2018     10 non-null     float64
 7   2019     10 non-null     float64
 8   2020     10 non-null     float64
 9   2021     10 non-null     float64
 10  2022     10 non-null     float64
dtypes: float64(10), object(1)
memory usage: 1012.0+ bytes
```

Food waste

b. Filter the ASEAN nations

```
fw_copy = FW.copy()

fw_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1134 entries, 0 to 1133
Data columns (total 29 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Goal                1134 non-null   int64
 1   Target              1134 non-null   float64
 2   Indicator           1134 non-null   object
 3   SeriesCode          1134 non-null   object
 4   SeriesDescription   1134 non-null   object
 5   GeoAreaCode         1134 non-null   int64
 6   GeoAreaName         1134 non-null   object
 7   Food Waste Sector   1134 non-null   object
 8   Observation Status  1134 non-null   object
 9   Reporting Type      1134 non-null   object
 10  Units               1134 non-null   object
 11  2005                1 non-null      float64
 12  2006                1 non-null      float64
 13  2007                1 non-null      float64
 14  2008                5 non-null      float64
 15  2009                5 non-null      float64
 16  2010                5 non-null      float64
 17  2011                5 non-null      float64
 18  2012                5 non-null      float64
 19  2013                5 non-null      float64
 20  2014                5 non-null      float64
 21  2015                6 non-null      float64
 22  2016                5 non-null      float64
```

```
 23  2017               6 non-null      float64
 24  2018               9 non-null      float64
 25  2019               1014 non-null   float64
 26  2020               118 non-null    float64
 27  2021               116 non-null    float64
 28  2022               1083 non-null   float64
dtypes: float64(19), int64(2), object(8)
memory usage: 257.1+ KB
```

Correlation

```
# Create the list of ASEAN member countries
asean_countries = [
    'Brunei Darussalam', 'Cambodia', 'Indonesia', 'Lao People\'s Democratic Republic',
    'Malaysia', 'Myanmar', 'Philippines', 'Singapore', 'Thailand', 'Viet Nam'
]

# Step 2: Filter the dataset for:
# - Countries in ASEAN
# - Food Waste Sector is "ALL"
filtered_df = fw_copy[
    (fw_copy["GeoAreaName"].isin(asean_countries)) &
    (fw_copy["Food Waste Sector"] == "ALL")
]

# Create a new DataFrame with only 'GeoAreaName' and '2022' columns
asean_food_waste_2022 = filtered_df[["GeoAreaName", "2022"]]

# Reset the index
asean_food_waste_2022 = asean_food_waste_2022.reset_index(drop=True)

asean_food_waste_2022
```

	GeoAreaName	2022
0	Brunei Darussalam	7.057748e+04
1	Cambodia	2.838261e+06
2	Indonesia	3.953579e+07
3	Lao People's Democratic Republic	1.310729e+06
4	Malaysia	7.110918e+06
5	Myanmar	8.804563e+06
6	Philippines	1.272880e+07
7	Singapore	8.938128e+05
8	Thailand	1.263670e+07
9	Viet Nam	1.538485e+07

Next steps: Generate code with asean_food_waste_2022 View recommended plots New interactive sh

Poverty Rate --- Food Insecurity

Poverty rate

```
# Make a copy of the population DataFrame and filter ASEAN countries
asean_pop_2022 = pop_copy.copy()
asean_pop_2022 = asean_pop_2022[asean_pop_2022["Country"].isin(asean_countries)][["Country",
asean_pop_2022 = asean_pop_2022.rename(columns={"Country": "GeoAreaName", "2022": "Populatic

# Make a copy of the food waste DataFrame
asean_fw_2022 = asean_food_waste_2022.copy()
asean_fw_2022 = asean_fw_2022.rename(columns={"2022": "FoodWaste_2022"})

# Merge  copies on 'GeoAreaName'
merged_df = pd.merge(asean_fw_2022, asean_pop_2022, on="GeoAreaName")

merged_df
```

	GeoAreaName	FoodWaste_2022	Population_2022
0	Brunei Darussalam	7.057748e+04	445.400
1	Cambodia	2.838261e+06	16843.333
2	Indonesia	3.953579e+07	275719.910
3	Myanmar	8.804563e+06	55770.200
4	Philippines	1.272880e+07	111572.254
5	Singapore	8.938128e+05	5637.022
6	Thailand	1.263670e+07	66090.000
7	Viet Nam	1.538485e+07	99461.715

Next steps: Generate code with merged_df | View recommended plots | New interactive sheet

```
pop3_copy = pop3.copy()
# Get an overview about the data
pop3_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1219 entries, 0 to 1218
Data columns (total 36 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Goal               1219 non-null   int64
 1   Target             1219 non-null   float64
 2   Indicator          1219 non-null   object
 3   SeriesCode         1219 non-null   object
 4   SeriesDescription  1219 non-null   object
 5   GeoAreaCode        1219 non-null   int64
 6   GeoAreaName        1219 non-null   object
 7   Age                1219 non-null   object
 8   Location           1219 non-null   object
 9   Reporting Type     1219 non-null   object
 10  Sex                1219 non-null   object
 11  Units              1219 non-null   object
 12  2000               114 non-null    float64
 13  2001               105 non-null    float64
 14  2002               140 non-null    float64
 15  2003               245 non-null    float64
 16  2004               336 non-null    float64
 17  2005               396 non-null    float64
 18  2006               395 non-null    float64
 19  2007               395 non-null    float64
 20  2008               394 non-null    float64
 21  2009               442 non-null    float64
 22  2010               487 non-null    float64
 23  2011               458 non-null    float64
 24  2012               514 non-null    float64
```

```
 25  2013              471 non-null    float64
 26  2014              479 non-null    float64
 27  2015              546 non-null    float64
 28  2016              522 non-null    float64
 29  2017              505 non-null    float64
 30  2018              617 non-null    float64
 31  2019              514 non-null    float64
 32  2020              434 non-null    float64
 33  2021              491 non-null    float64
 34  2022              195 non-null    float64
 35  2023              16 non-null     float64
dtypes: float64(25), int64(2), object(9)
memory usage: 343.0+ KB
```

Food Insecurity

```
pop2_copy = pop2.copy()
# Get an overview about the data
pop2_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1134 entries, 0 to 1133
Data columns (total 29 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Goal                1134 non-null   int64
 1   Target              1134 non-null   float64
 2   Indicator           1134 non-null   object
 3   SeriesCode          1134 non-null   object
 4   SeriesDescription   1134 non-null   object
 5   GeoAreaCode         1134 non-null   int64
 6   GeoAreaName         1134 non-null   object
 7   Food Waste Sector   1134 non-null   object
 8   Observation Status  1134 non-null   object
 9   Reporting Type      1134 non-null   object
 10  Units               1134 non-null   object
 11  2005                1 non-null      float64
 12  2006                1 non-null      float64
 13  2007                1 non-null      float64
 14  2008                5 non-null      float64
 15  2009                5 non-null      float64
 16  2010                5 non-null      float64
 17  2011                5 non-null      float64
 18  2012                5 non-null      float64
 19  2013                5 non-null      float64
 20  2014                5 non-null      float64
 21  2015                6 non-null      float64
 22  2016                5 non-null      float64
 23  2017                6 non-null      float64
 24  2018                9 non-null      float64
 25  2019                1014 non-null   float64
 26  2020                118 non-null    float64
 27  2021                116 non-null    float64
```

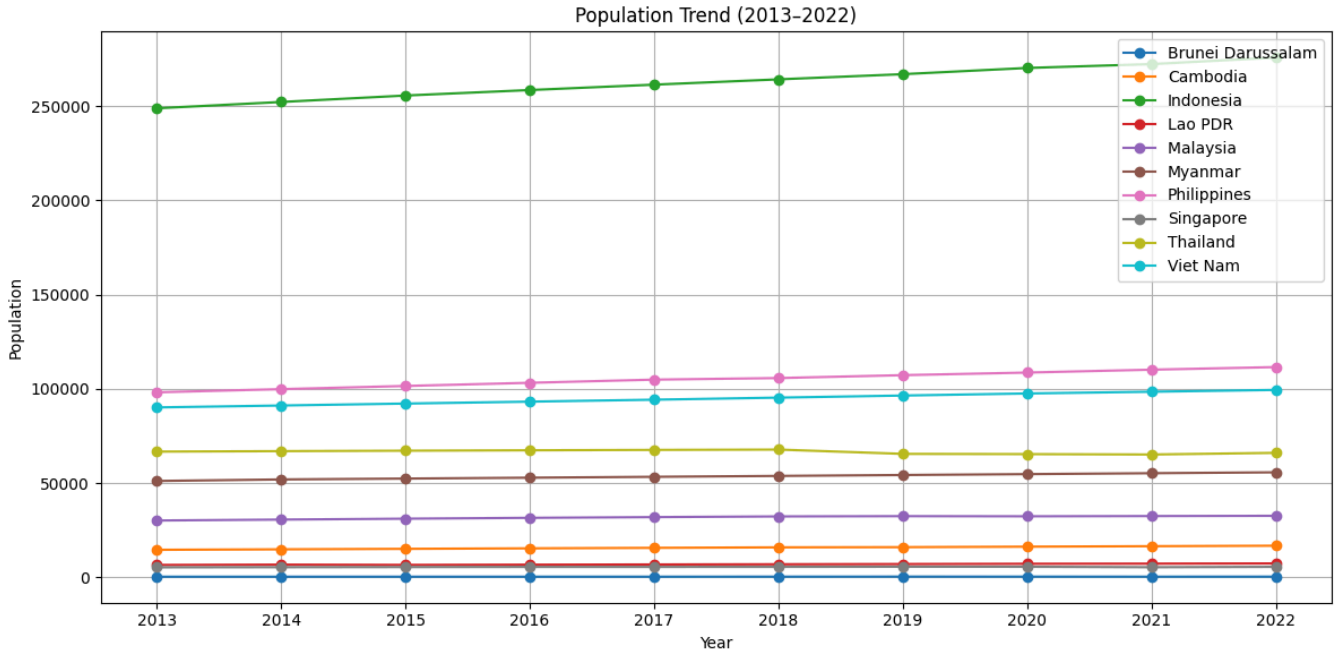
```
 28  2022                1083 non-null   float64
dtypes: float64(19), int64(2), object(8)
memory usage: 257.1+ KB
```

4.a. Analyze the data using descriptive analysis.

I. Population --- Food Waste

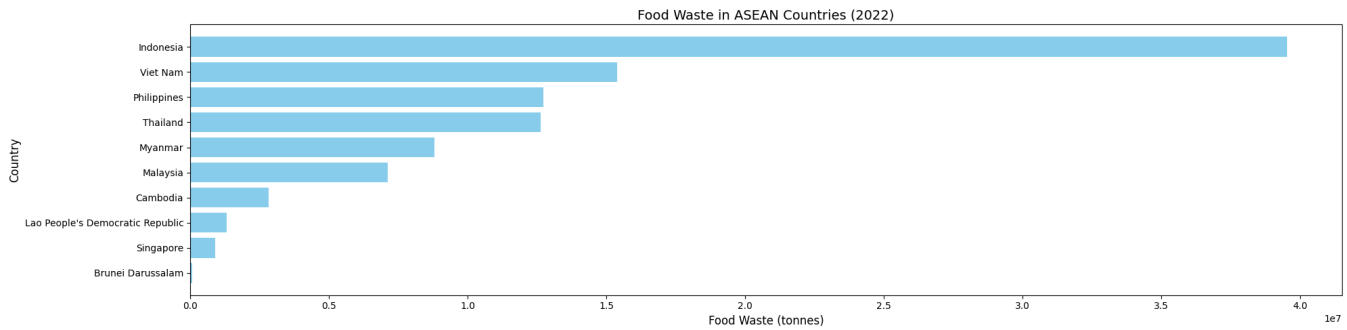
a. Population

```
import matplotlib.pyplot as plt

# Step 1: Set the index to 'Country'
pop_copy = pop_copy.set_index('Country')

# Step 2: Transpose so years are rows
df_t = pop_copy.T  # Now rows = years, columns = countries

# Step 3: Plot each country as a separate line
plt.figure(figsize=(12, 6))

for country in df_t.columns:
    plt.plot(df_t.index, df_t[country], label=country, marker='o')

# Step 4: Final touches
plt.title("Population Trend (2013–2022)")
plt.xlabel("Year")
plt.ylabel("Population")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



b. Food waste (in tonnes)

```
import matplotlib.pyplot as plt

# Sort data by 2022 values for better visualization
plot_df = asean_food_waste_2022.sort_values(by="2022", ascending=True)

plt.figure(figsize=(20, 5))
plt.barh(plot_df["GeoAreaName"], plot_df["2022"], color='skyblue')

plt.title("Food Waste in ASEAN Countries (2022)", fontsize=14)
plt.xlabel("Food Waste (tonnes)", fontsize=12)
plt.ylabel("Country", fontsize=12)

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



II. Poverty rate --- Food Insecurity

Poverty rate

```
# Descriptive statistics for poverty
print("\nDescriptive Statistics: Poverty")
print(pop3_copy.describe())
```

```
Descriptive Statistics: Poverty
          Goal        Target  GeoAreaCode         2000        2001        2002  \
count  1219.0  1.219000e+03  1219.000000  114.000000  105.000000  140.000000
mean      1.0  1.100000e+00   434.600492   16.951754   20.371429   16.600000
std       0.0  9.329701e-15   248.705064   19.008229   18.007614   18.695516
min       1.0  1.100000e+00     1.000000    0.000000    0.000000    0.000000
25%       1.0  1.100000e+00   218.000000    3.025000    5.000000    2.000000
50%       1.0  1.100000e+00   430.000000   10.750000   14.000000   11.000000
75%       1.0  1.100000e+00   643.000000   24.875000   34.000000   23.500000
max       1.0  1.100000e+00   894.000000   84.000000   71.000000   81.000000

             2003        2004       2005        2006  ...        2014  \
count  245.000000  336.000000  396.00000  395.000000  ...  479.000000
mean     8.677551    7.095238    8.07803    5.222785  ...    5.592902
std     15.320821   13.434524   13.68931   10.556399  ...   11.204976
min      0.000000    0.000000    0.00000    0.000000  ...    0.000000
25%      0.000000    0.000000    0.30000    0.000000  ...    0.000000
50%      1.000000    1.000000    1.05000    1.000000  ...    1.000000
75%     10.000000    7.250000   10.70000    6.000000  ...    5.000000
max     82.000000   92.000000   80.50000   72.000000  ...   75.000000
```

```
              2015        2016        2017        2018        2019        2020  \
count  546.000000  522.000000  505.000000  617.000000  514.000000  434.000000
mean     5.153297    6.875479    3.794059    6.199352    4.134241    2.964516
std     10.658781   14.710965    8.142162   11.317616   11.551540    9.482944
min      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
25%      0.200000    0.000000    0.000000    0.000000    0.000000    0.100000
50%      1.000000    1.000000    1.000000    1.000000    1.000000    0.500000
75%      4.775000    4.000000    3.000000    5.000000    2.000000    1.575000
max     83.000000   76.000000   61.000000   59.000000   77.000000   78.900000

              2021        2022       2023
count  491.000000  195.000000  16.000000
mean     5.101833    5.094872   1.750000
std     11.807720   13.350180   1.064581
min      0.000000    0.000000   0.000000
25%      0.000000    0.500000   1.000000
50%      1.000000    1.200000   2.000000
75%      3.000000    3.200000   2.000000
max     76.000000   82.700000   4.000000

[8 rows x 27 columns]
```

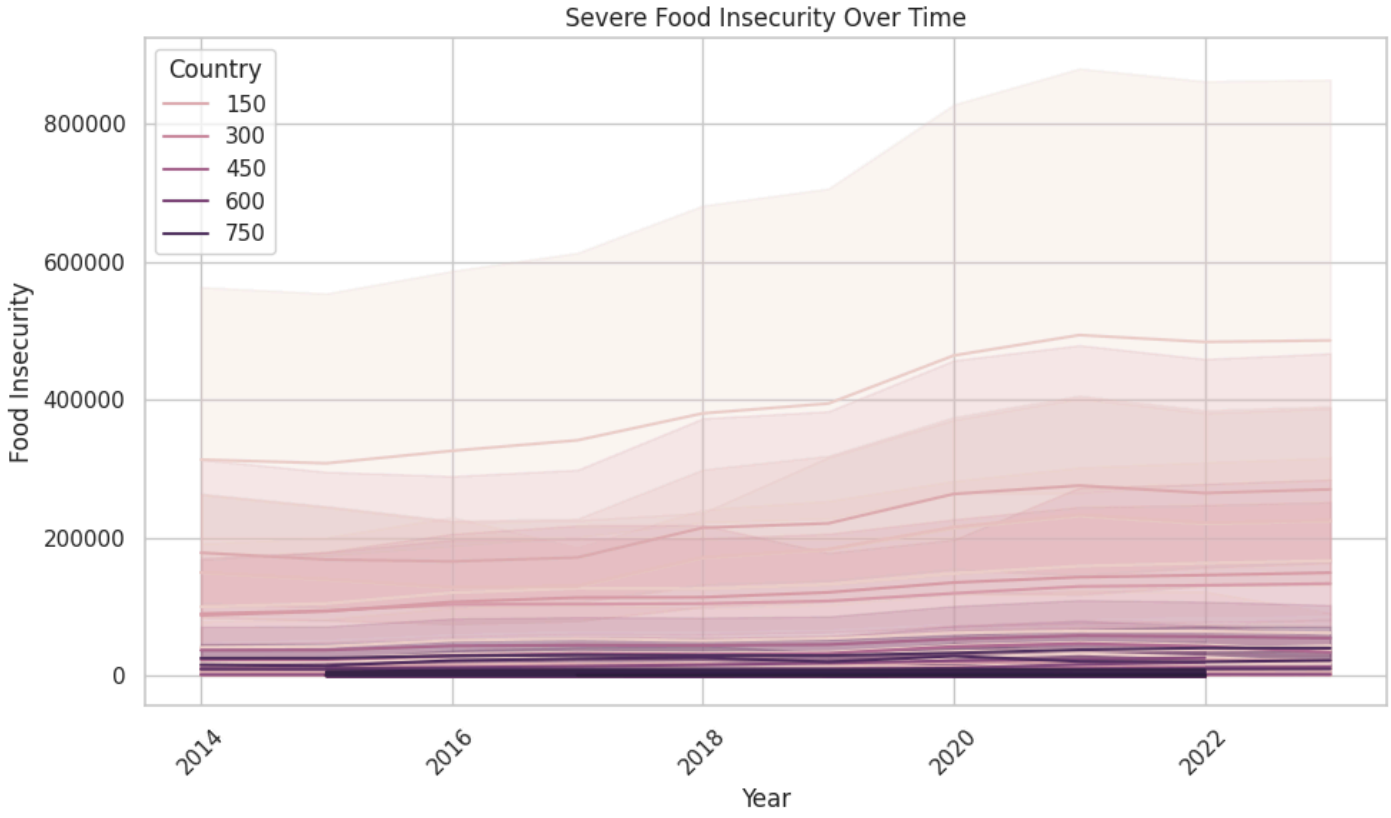
Food Insecurity

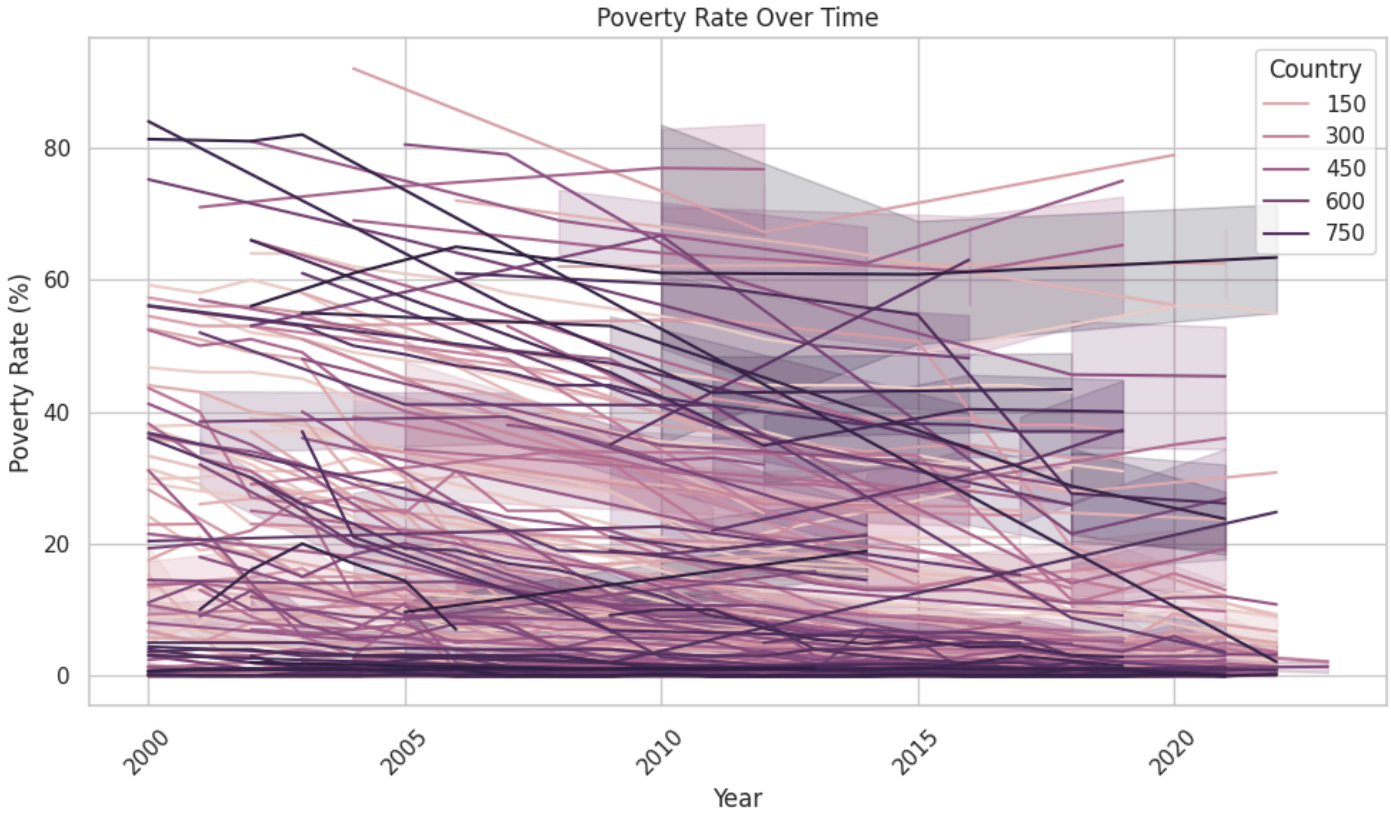
```
# Descriptive statistics for food insecurity
print("Descriptive Statistics: Food Insecurity")
print(pop2_copy.describe())
```

```
Descriptive Statistics: Food Insecurity
         Goal        Target  GeoAreaCode      2005      2006      2007  \
count  1134.0  1.134000e+03  1134.000000       1.0       1.0       1.0
mean     12.0  1.230000e+01   401.331570  728060.0  749890.0  772310.0
std       0.0  2.239197e-13   261.950366       NaN       NaN       NaN
min      12.0  1.230000e+01     1.000000  728060.0  749890.0  772310.0
25%      12.0  1.230000e+01   171.000000  728060.0  749890.0  772310.0
50%      12.0  1.230000e+01   406.000000  728060.0  749890.0  772310.0
75%      12.0  1.230000e+01   629.000000  728060.0  749890.0  772310.0
max      12.0  1.230000e+01   894.000000  728060.0  749890.0  772310.0

               2008          2009          2010          2011  ...  \
count  5.000000e+00  5.000000e+00  5.000000e+00  5.000000e+00  ...
mean   6.159064e+06  5.899430e+06  5.849868e+06  5.489624e+06  ...
std    6.352576e+06  6.077542e+06  6.183632e+06  5.802706e+06  ...
min    7.953200e+05  8.171500e+05  8.413400e+05  8.661200e+05  ...
25%    1.309000e+06  1.348000e+06  1.192000e+06  1.275000e+06  ...
50%    2.971000e+06  2.672000e+06  2.292000e+06  1.876000e+06  ...
75%    1.072000e+07  1.032000e+07  1.072000e+07  1.014000e+07  ...
max    1.500000e+07  1.434000e+07  1.420400e+07  1.329100e+07  ...

               2013          2014          2015          2016          2017  \
count  5.000000e+00  5.000000e+00  6.000000e+00  5.000000e+00  6.000000e+00
mean   4.913020e+06  4.761660e+06  4.274338e+06  4.662035e+06  3.867583e+06
std    5.017813e+06  4.774845e+06  4.529367e+06  4.598125e+06  4.518101e+06
```

```
min    9.191020e+05  9.463010e+05  9.740310e+05  1.002174e+06  9.500000e+03
25%    1.239000e+06  1.269000e+06  1.331250e+06  1.271000e+06  1.015500e+06
50%    1.884000e+06  1.938000e+06  1.747500e+06  1.994000e+06  1.646000e+06
75%    8.700000e+06  8.224000e+06  6.735750e+06  7.889000e+06  6.391000e+06
max    1.182300e+07  1.143100e+07  1.158600e+07  1.115400e+07  1.112600e+07

                2018          2019          2020          2021          2022
count  9.000000e+00  1.014000e+03  1.180000e+02  1.160000e+02  1.083000e+03
mean   7.281602e+06  1.055311e+07  1.026134e+06  9.505668e+05  1.015716e+07
std    6.794794e+06  5.263254e+07  2.046426e+06  1.927371e+06  5.086142e+07
min    1.141480e+00  6.402003e+01  2.377066e+02  2.067315e+02  0.000000e+00
25%    1.223000e+06  5.330650e+04  6.360200e+04  7.136600e+04  2.839771e+04
50%    7.662000e+06  4.405835e+05  2.305400e+05  2.228270e+05  3.264564e+05
75%    1.100000e+07  2.554019e+06  9.240092e+05  8.194900e+05  2.476867e+06
max    2.041821e+07  9.308639e+08  1.091889e+07  1.093680e+07  1.051963e+09

[8 rows x 21 columns]
```

```
import seaborn as sns

# Optional: Convert 'Year' to numeric
pop2_long['Year'] = pd.to_numeric(pop2_long['Year'], errors='coerce')
pop3_long['Year'] = pd.to_numeric(pop3_long['Year'], errors='coerce')

# Drop rows with missing values
pop2_long.dropna(subset=['Food_Insecurity'], inplace=True)
pop3_long.dropna(subset=['Poverty'], inplace=True)

# Plot
plt.figure(figsize=(10, 6))
sns.lineplot(data=pop2_long, x="Year", y="Food_Insecurity", hue="Country")
plt.title("Severe Food Insecurity Over Time")
plt.xlabel("Year")
plt.ylabel("Food Insecurity")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

plt.figure(figsize=(10, 6))
sns.lineplot(data=pop3_long, x="Year", y="Poverty", hue="Country")
plt.title("Poverty Rate Over Time")
plt.xlabel("Year")
plt.ylabel("Poverty Rate (%)")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

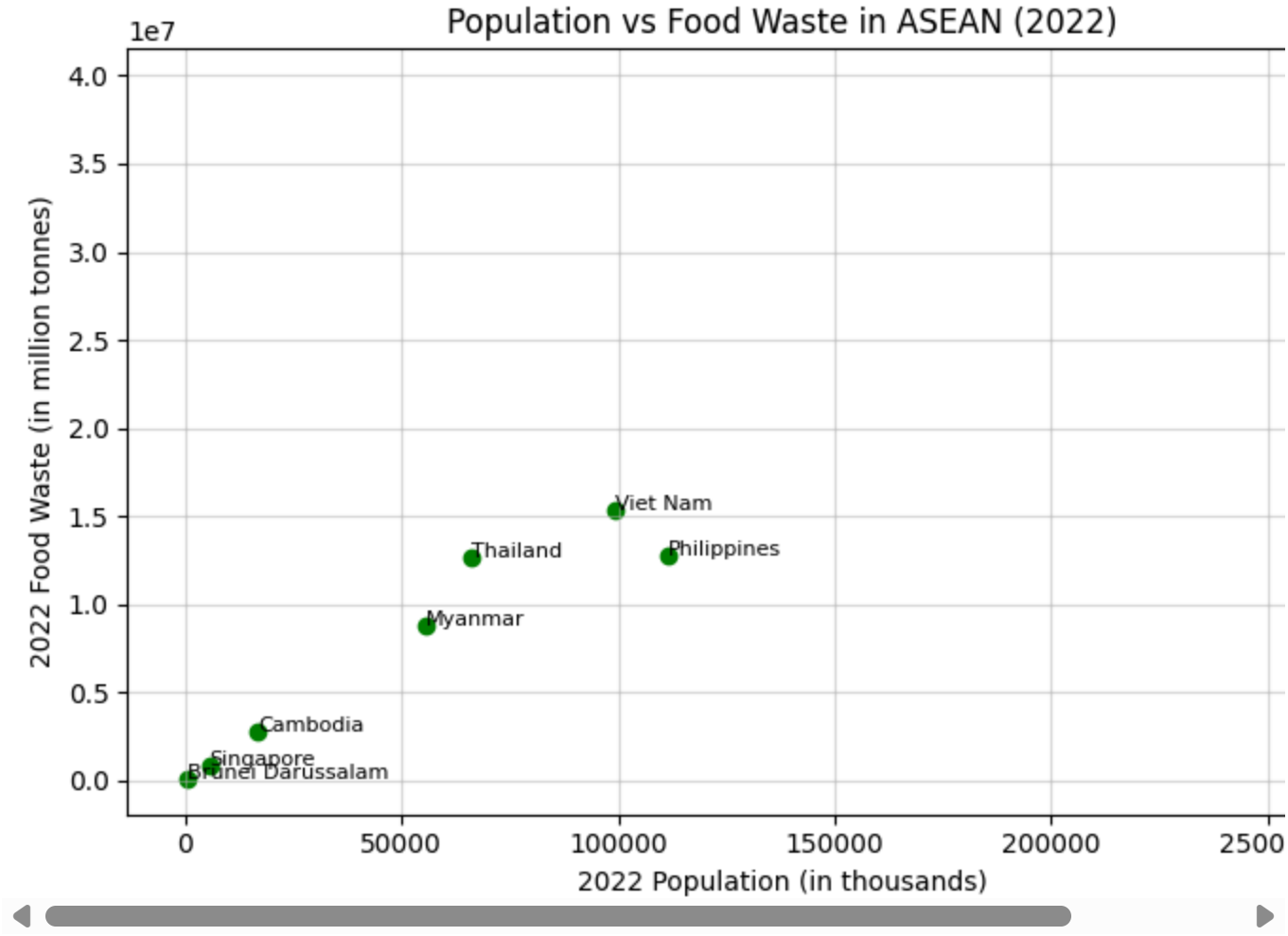




4.b. Perform correlation analysis.

```
# Pearson correlation coefficient
correlation = merged_df["FoodWaste_2022"].corr(merged_df["Population_2022"])
print(f"Correlation between population and food waste in ASEAN (2022): {correlation:.2f}")

# Scatter plot
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 5))
plt.scatter(merged_df["Population_2022"],
            merged_df["FoodWaste_2022"],
            color='green')


plt.title("Population vs Food Waste in ASEAN (2022)")
plt.xlabel("2022 Population (in thousands)")
plt.ylabel("2022 Food Waste (in million tonnes)")

for i, row in merged_df.iterrows():
    plt.text(row["Population_2022"], row["FoodWaste_2022"], row["GeoAreaName"], fontsize=8)


plt.grid(True,
         alpha = 0.5)

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

```
Correlation between population and food waste in ASEAN (2022): 0.99
```




```
# Identify columns that are years
year_cols_pop2 = [col for col in pop2_copy.columns if str(col).isdigit()]
year_cols_pop3 = [col for col in pop3_copy.columns if str(col).isdigit()]

# Melt the datasets safely
pop2_long = pop2_copy.melt(id_vars=['GeoAreaCode'], value_vars=year_cols_pop2,
                           var_name='Year', value_name='Food_Insecurity')

pop3_long = pop3_copy.melt(id_vars=['GeoAreaCode'], value_vars=year_cols_pop3,
                           var_name='Year', value_name='Poverty')

# Convert values to numeric in case there are strings
pop2_long['Food_Insecurity'] = pd.to_numeric(pop2_long['Food_Insecurity'], errors='coerce')
pop3_long['Poverty'] = pd.to_numeric(pop3_long['Poverty'], errors='coerce')

# Merge on GeoAreaCode and Year
merged_df = pd.merge(pop2_long, pop3_long, on=['GeoAreaCode', 'Year'])

# Drop missing data
merged_df = merged_df.dropna()

# Correlation
```

```
correlation = merged_df['Food_Insecurity'].corr(merged_df['Poverty'])
print(f"Correlation between Food Insecurity and Poverty: {correlation:.3f}")
```

```
Correlation between Food Insecurity and Poverty: 0.054
```

```
import seaborn as sns
import matplotlib.pyplot as plt

# Set plot style
sns.set(style="whitegrid")

# Create scatter plot with regression line
plt.figure(figsize=(10, 6))
sns.regplot(data=merged_df, x='Poverty', y='Food_Insecurity', scatter_kws={'alpha':0.6})

# Customize labels and title
plt.title("Correlation Between Poverty and Food Insecurity", fontsize=14)
plt.xlabel("Poverty Rate (%)", fontsize=12)
plt.ylabel("Food Insecurity (count)", fontsize=12)

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