Course title: INT6103 Python for Data Analysis and Visualization

Final Project

Student Name:

Christopher Pearson

Date: 07/18/2024

Documentation for the Popular Baby Names Analysis

Project Description

This project involves analyzing a dataset of popular baby names using Python. The objective is to write Python code to answer specific questions, and to produce a comprehensive report detailing the analysis and findings. The final deliverables include the Python code and a Jupyter Notebook.

Dataset Description

The dataset contains information on the frequency and rank of baby names in the United States from 2011 to 2019. It includes the following columns:

- **Year of Birth**: The year the baby was born.
- **Gender**: The gender of the baby (M for male, F for female).
- **Ethnicity**: The ethnicity of the baby (e.g., White, Black, Hispanic, Asian, etc.).
- **Child's First Name**: The first name of the baby.
- **Count**: The number of babies with that name.
- **Rank**: The rank of the name in terms of frequency (1 = most frequent).

Section 1: Dataset Exploration

1. **Load the dataset into a Pandas DataFrame and display the first 10 rows:**

```
[1]: # Import necessary libraries
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[2]: # Load the dataset into a Pandas DataFrame
     data = pd.read_csv("Popular_Baby_Names.csv")
```

- pandas is used for data manipulation and analysis.
- matplotlib.pyplot is used for plotting graphs.
- seaborn is used for creating statistical graphics.
- The dataset is loaded into a Pandas DataFrame from a CSV file.
- The first 10 rows of the dataset are displayed to get an initial look at the data.

  This step reads the CSV file into a DataFrame and prints the first 10 rows to get an initial look at the data.

2. **Display the shape of the dataset:**

   "print(data.shape)"

   This step prints the number of rows and columns in the dataset.

3. **Identify the data types of each column and check for any missing values:**

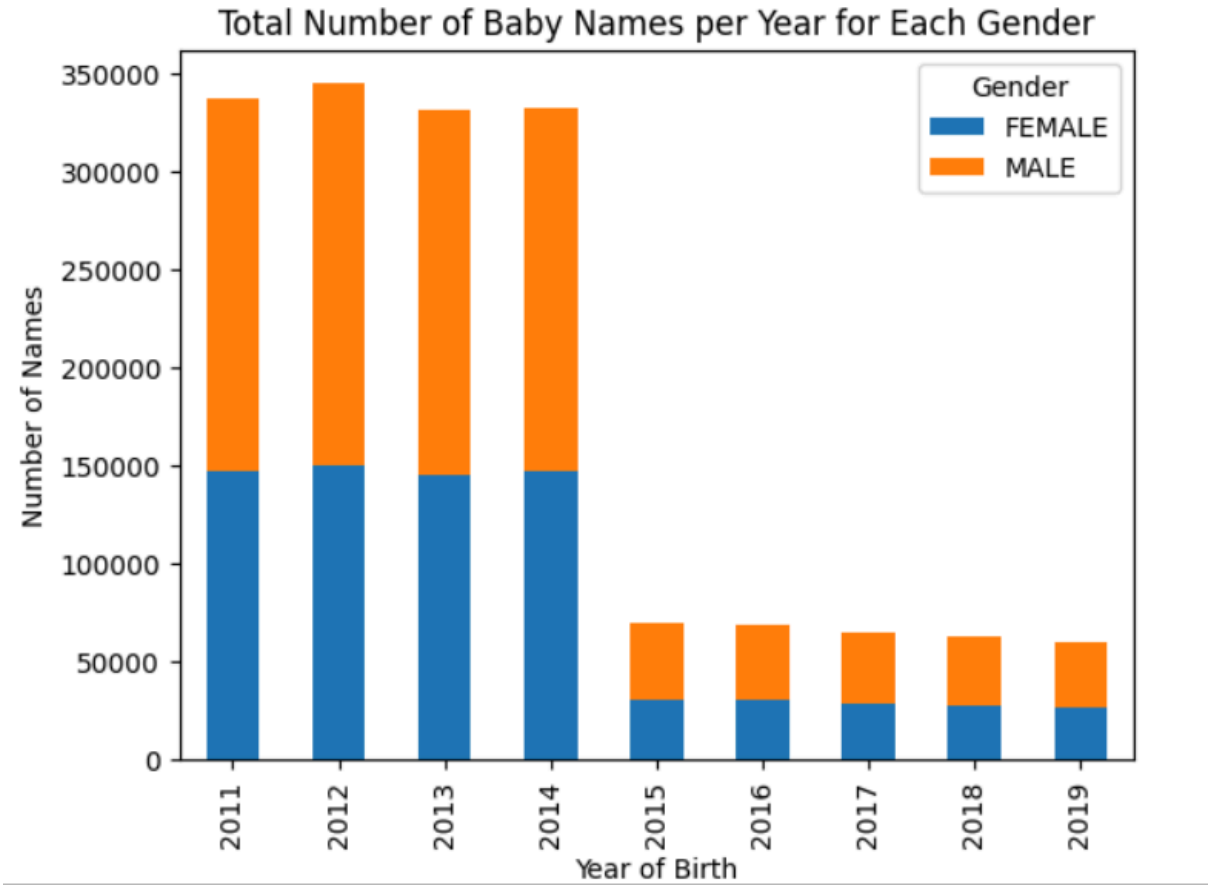
“print(data.info())

print(data.isnull().sum())”

The info() method provides information about the data types and non-null counts for each column, while isnull().sum() checks for missing values.

4. **Create a bar chart showing the total number of baby names per year for each gender:**

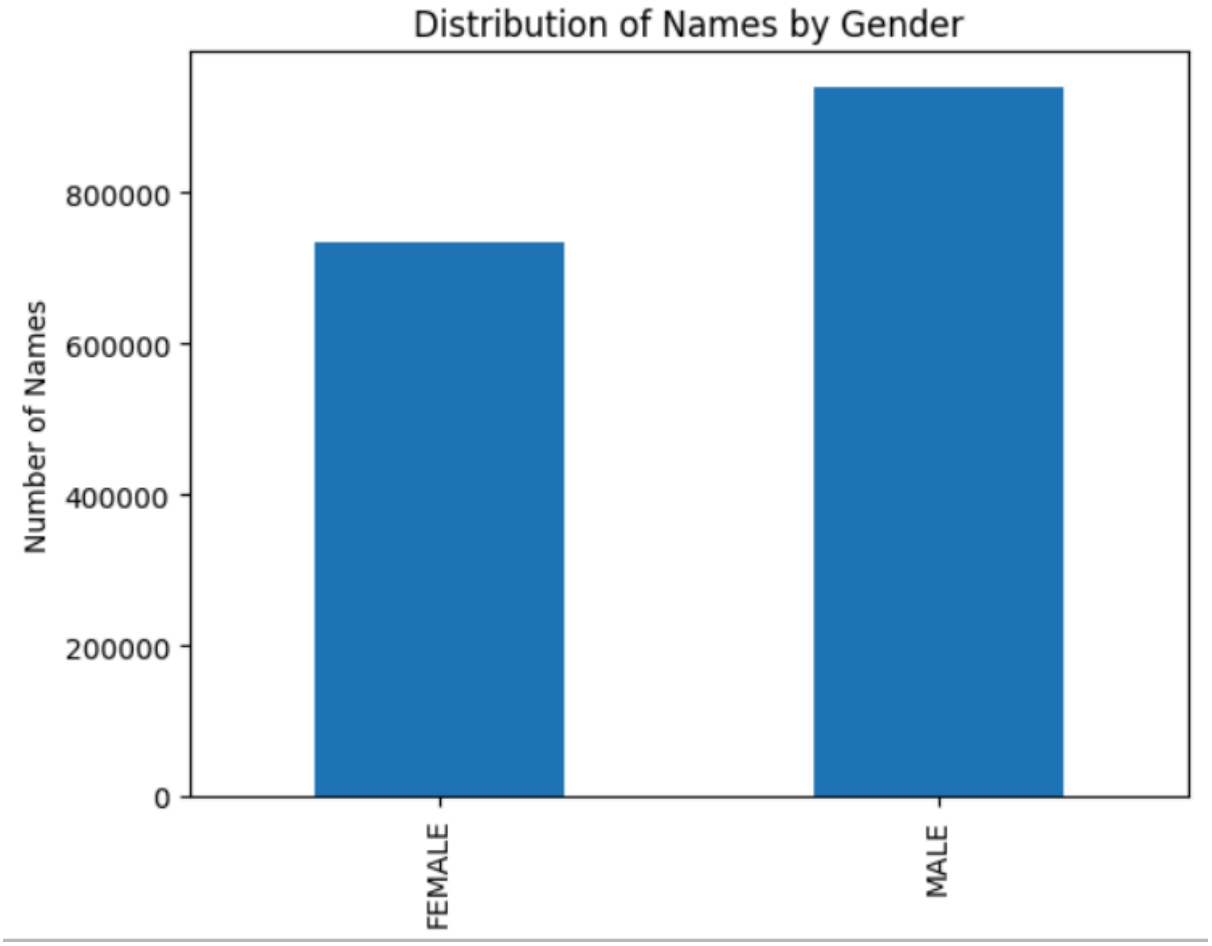
```
year_gender_counts = data.groupby(['Year of Birth', 'Gender'])['Count'].sum().unstack()
year_gender_counts.plot(kind='bar', stacked=True)
plt.title('Total Number of Baby Names per Year for Each Gender')
plt.xlabel('Year of Birth')
plt.ylabel('Number of Names')
plt.legend(title='Gender')
plt.show()
```



- The data is grouped by Year of Birth and Gender, and the counts are summed.
- unstack() is used to pivot the data for easy plotting.
- A stacked bar chart is created to show the total number of baby names per year for each gender.

5. **Create a bar chart showing the distribution of names by gender:**

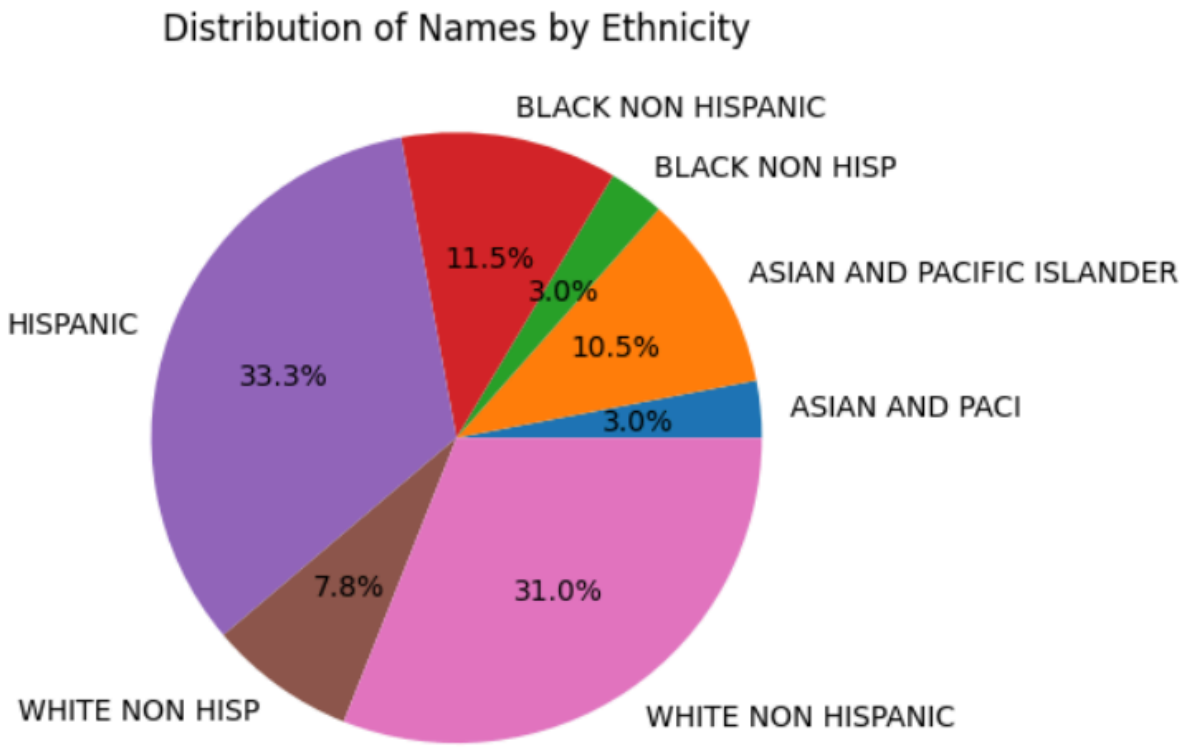
```
gender_counts = data.groupby('Gender')['Count'].sum()
gender_counts.plot(kind='bar')
plt.title('Distribution of Names by Gender')
plt.xlabel('Gender')
plt.ylabel('Number of Names')
plt.show()
```



This step groups the data by gender, sums the counts, and creates a bar chart to show the distribution of names by gender.

6. **Create a pie chart showing the distribution of names by ethnicity:**

```
ethnicity_counts = data.groupby('Ethnicity')['Count'].sum()
ethnicity_counts.plot(kind='pie', autopct='%1.1f%%')
plt.title('Distribution of Names by Ethnicity')
plt.ylabel('')
plt.show()
```



This code groups the data by ethnicity, sums the counts, and creates a pie chart to show the distribution of names by ethnicity.

Section 2: Data Cleaning and Preparation

1. **Check for duplicated rows in the dataset for the year 2011:**

"duplicates_2011 = data[data['Year of Birth'] == 2011].duplicated()
print(f"Number of duplicated rows in 2011: {duplicates_2011.sum()}")"

This code checks for duplicated rows in the year 2011 and prints the number of duplicates.

Number of duplicated rows in 2011: 7826

2. **Rename columns:**

   “data.rename(columns={'Year of Birth': 'Year_of_Birth', "Child's First Name": 'Childs_First_Name'}, inplace=True)”

   This step renames columns to ensure consistent naming conventions.

3. **Create a new column 'Name_Length' for the length of each name excluding non-alphabetic characters:**

   “data['Name_Length'] = data['Childs_First_Name'].str.replace(r'[^A-Za-z]', '', regex=True).str.len()”

   A new column Name_Length is created that contains the length of each name, excluding non-alphabetic characters.

4. **Create a new column 'Percentage' for the percentage of babies with each name:**

   “total_counts_year_gender = data.groupby(['Year_of_Birth', 'Gender'])['Count'].transform('sum')
   data['Percentage'] = (data['Count'] / total_counts_year_gender) * 100”

   This step calculates the percentage of babies with each name by dividing the count by the total count for the same year and gender, then multiplying by 100.

5. **Replace 'Black or African American Non-Hispanic' with 'Black Non-Hispanic':**

   “data['Ethnicity'] = data['Ethnicity'].replace('Black or African American Non-Hispanic', 'Black Non-Hispanic')”

   This code replaces a specific ethnicity label to ensure consistency.

Section 3: Data Analysis and Visualization

1. **Determine the number of children named Jacob in 2018 and the ethnicity distribution of those children:**

```python
# 1. How many children were named Jacob in 2018 and what is the ethnicity distribution of those children?
jacob_2018 = data[(data['Year_of_Birth'] == 2018) & (data['Childs_First_Name'] == 'Jacob')]
jacob_2018_count = jacob_2018['Count'].sum()
jacob_2018_ethnicity_distribution = jacob_2018.groupby('Ethnicity')['Count'].sum()
print(f"Number of children named Jacob in 2018: {jacob_2018_count}")
print("Ethnicity distribution of children named Jacob in 2018:")
print(jacob_2018_ethnicity_distribution)
```

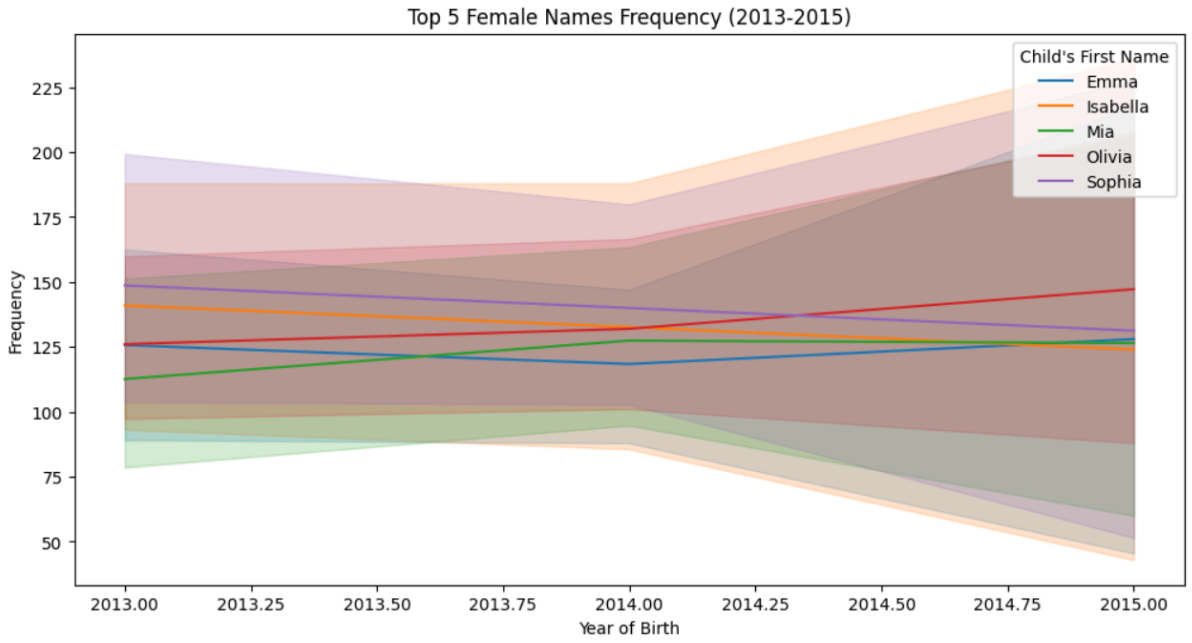
```
Number of children named Jacob in 2018: 540
Ethnicity distribution of children named Jacob in 2018:
Ethnicity
ASIAN AND PACIFIC ISLANDER     46
BLACK NON HISPANIC             36
HISPANIC                      241
WHITE NON HISPANIC            217
Name: Count, dtype: int64
```

This code filters the data for the year 2018 and the name 'Jacob', calculates the total count, and groups the data by ethnicity to show the distribution.

- The data is filtered for the year 2018 and the name 'Jacob'.
- The total count of children named Jacob is calculated.
- The ethnicity distribution of children named Jacob is calculated and printed.

2. **Create a line plot showing the frequency of the top 5 names for the female gender between 2013 and 2015:**

```python
# 2. Create a line plot that shows the frequency of the top 5 names for the female gender between 2013 and 2015
female_data = data[(data['Gender'] == 'FEMALE') & (data['Year_of_Birth'].between(2013, 2015))]
top_5_female_names = female_data.groupby('Childs_First_Name')['Count'].sum().nlargest(5).index
top_5_female_data = female_data[female_data['Childs_First_Name'].isin(top_5_female_names)]
plt.figure(figsize=(12, 6))
sns.lineplot(data=top_5_female_data, x='Year_of_Birth', y='Count', hue='Childs_First_Name')
plt.title('Top 5 Female Names Frequency (2013-2015)')
plt.xlabel('Year of Birth')
plt.ylabel('Frequency')
plt.legend(title="Child's First Name")
plt.show()
```

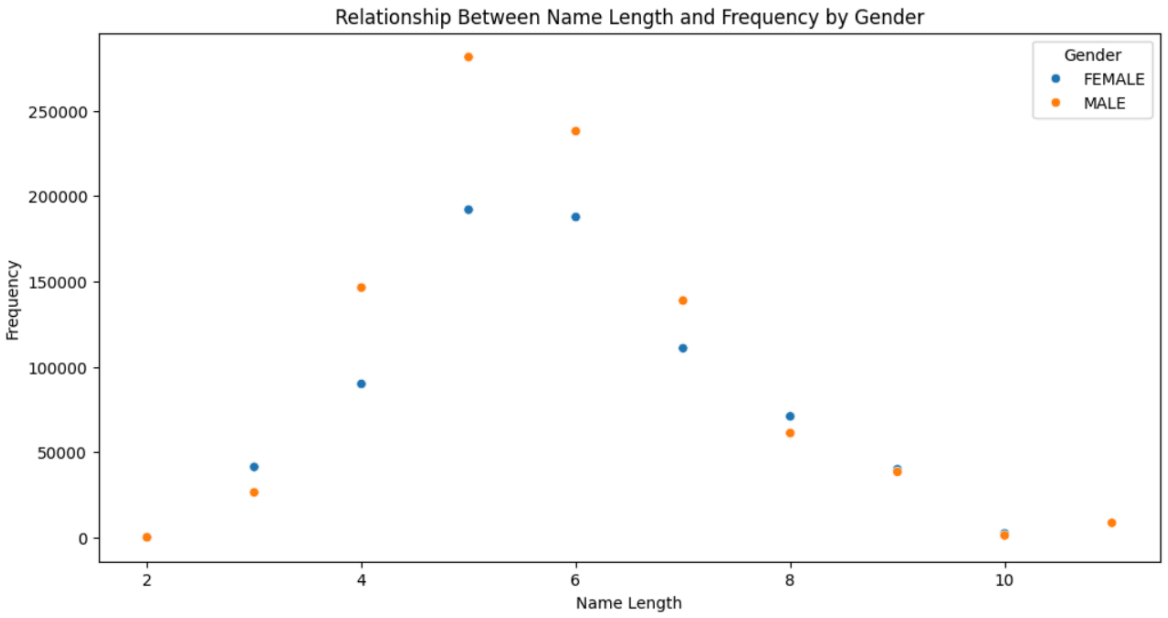


- The data is filtered for females born between 2013 and 2015.
- The top 5 names are identified by summing the counts.
- A line plot is created to show the frequency of these names over the years.

  This code filters the data for females born between 2013 and 2015, identifies the top 5 names, and creates a line plot to show the frequency of these names over the years.

3. **Create a scatter plot showing the relationship between the length of names and their frequency, differentiated by gender:**

```
# 3. Relationship between the length of names and their frequency, differentiated by gender
name_length_frequency = data.groupby(['Gender', 'Name_Length'])['Count'].sum().reset_index()
plt.figure(figsize=(12, 6))
sns.scatterplot(data=name_length_frequency, x='Name_Length', y='Count', hue='Gender')
plt.title('Relationship Between Name Length and Frequency by Gender')
plt.xlabel('Name Length')
plt.ylabel('Frequency')
plt.show()
```

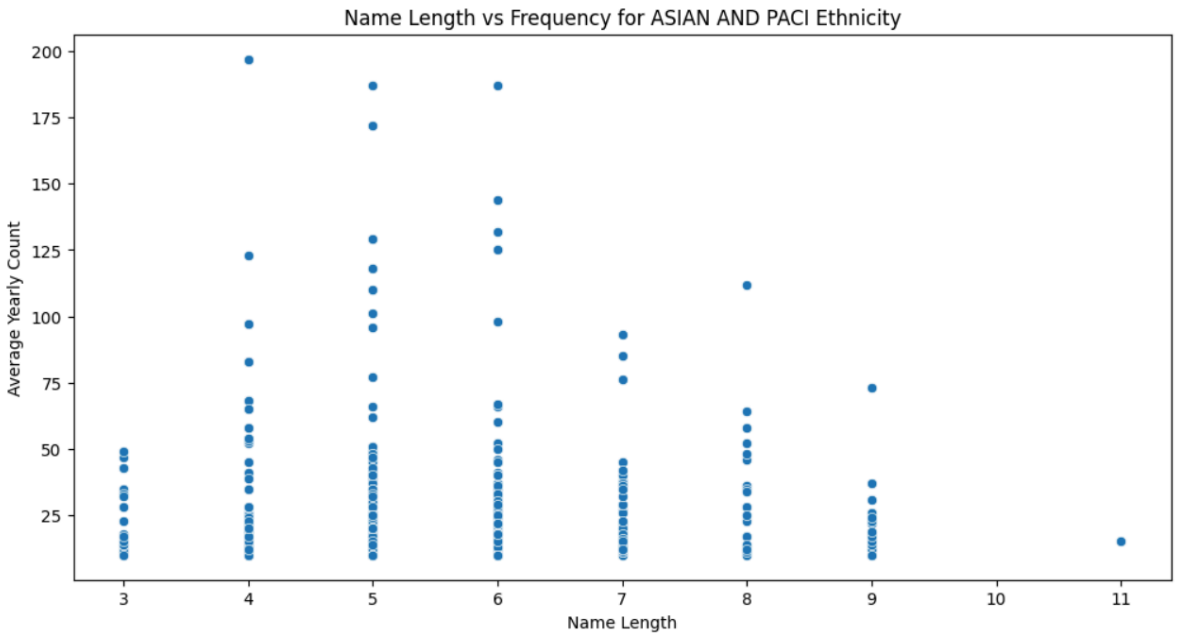


- The data is grouped by gender and name length, and the counts are summed.
- A scatter plot is created to show the relationship between name length and frequency, differentiated by gender.

This code groups the data by gender and name length, sums the counts, and creates a scatter plot to show the relationship between name length and frequency.

4. **Create a scatter plot showing the correlation between name length and frequency for the 'ASIAN AND PACI' ethnicity:**

```
# 4. Scatter plot showing the correlation between name length and frequency for the 'ASIAN AND PACI' ethnicity
asian_paci_data = data[data['Ethnicity'] == 'ASIAN AND PACI']
asian_paci_data_avg = asian_paci_data.groupby(['Childs_First_Name', 'Name_Length'])['Count'].mean().reset_index()
plt.figure(figsize=(12, 6))
sns.scatterplot(data=asian_paci_data_avg, x='Name_Length', y='Count')
plt.title('Name Length vs Frequency for ASIAN AND PACI Ethnicity')
plt.xlabel('Name Length')
plt.ylabel('Average Yearly Count')
plt.show()
```



This code filters the data for the 'ASIAN AND PACI' ethnicity, calculates the average count of each name, and creates a scatter plot to show the relationship between name length and frequency.

- The data is filtered for the 'ASIAN AND PACI' ethnicity.
- The average count of each name is calculated.
- A scatter plot is created to show the relationship between name length and frequency for this ethnicity.