### Problem Statement - Part 1 (data.csv)

import pandas as pd

\_

# Generate data

data = {

'age': [25, 30, 22, 40, 55, 60, 33, 28, 45, 50],

'income': [50000, 60000, 45000, 70000, 80000, 90000, 65000, 62000, 75000, 85000],

'age\_group': ['20-30', '30-40', '20-30', '40-50', '50-60', '50-60', '30-40', '20-30', '40-50', '50-60']

}

# Define data in DataFrame

df = pd.DataFrame(data)

\_

# Group the data by age\_group and compute summary statistics for 'income'

summary\_stats = df.groupby('age\_group')['income'].describe()

# Print summary

print(summary\_stats)

\_

# Group the data by age\_group; Select income column for each of the groups created; Calculate median for income

median\_income = df.groupby('age\_group')['income'].median()

# Print dat median

print("Median Income by Age Group:")

print(median\_income)

\_

print("Column Names:", df.columns)

\_

# Modified dataset with repeated values

data = {

'age': [25, 30, 25, 40, 55, 60, 33, 28, 45, 50, 25, 30, 28, 30, 25],

'income': [50000, 60000, 50000, 70000, 80000, 90000, 65000, 62000, 75000, 85000, 50000, 60000, 62000, 70000, 75000],

'age\_group': ['20-30', '30-40', '20-30', '40-50', '50-60', '50-60', '30-40', '20-30', '40-50', '50-60', '20-30', '30-40', '20-30', '30-40', '20-30']

}

# Define data in DataFrame

df = pd.DataFrame(data)

\_

# Calculate the mode for each column

mode\_age = df['age'].mode()

mode\_income = df['income'].mode()

print(f"Mode of Age: {mode\_age.values}")

print(f"Mode of Income: {mode\_income.values}")

\_

**Problem Statement - Part 2 (iris.csv)**

# Load iris.csv in the DataFrame

df = pd.read\_csv('iris.csv')

print(df.head()) # Print first 5 columns

\_

# Group the data by species and display summary statistics

summary\_stats\_species = df.groupby('variety').describe()

# Compute specific percentiles and statistics

percentiles = df.groupby('variety').quantile([0.25, 0.5, 0.75])

# Display summary statistics and percentiles

summary\_stats\_species = df.groupby('variety').describe()

print("\nPercentiles by Species:")

print(percentiles)

\_

# Group the data by variety; Select sepal.width column for each of the groups created; Display summary statistics

summary\_stats\_species = df.groupby('variety')['sepal.width'].describe()

print("\nSummary Statistics by Species for Sepal Width:")

print(summary\_stats\_species)

\_

# Group by variety and compute the median for numeric columns

median\_values = df.groupby('variety').median()

print("Median Values by Species:")

print(median\_values)

\_

**# Group the data by variety; Select sepal.width column for each of the groups created; Display median**

**median\_sepal\_length = df.groupby('variety')['sepal.length'].median()**

**print("Median Sepal Length by Species:")**

**print(median\_sepal\_length)**

**\_**

**# Calculate & print mode for sepal.width**

**mode\_width = df['sepal.width'].mode()**

**print(f"Mode of Width: {mode\_width.values}")**

### ****Problem Statement - Part 1: Descriptive Statistics with Data Grouped by Age Group****

#### ****Objective****

The task here is to perform descriptive statistics on a dataset that includes information about individuals' **age**, **income**, and their corresponding **age group**. Specifically:

1. **Summary Statistics for Income by Age Group**: We need to calculate and display summary statistics (mean, median, minimum, maximum, standard deviation) for the **income** variable, grouped by **age group**.
2. **Median Income for Each Age Group**: For each **age group**, compute the median **income**.
3. **Mode Calculation**: Finally, calculate the **mode** for both **age** and **income** variables.

#### ****Step-by-Step Explanation of the Code for Part 1****

### ****Step 1: Create the Initial Dataset****

import pandas as pd

# Generate data

data = {

'age': [25, 30, 22, 40, 55, 60, 33, 28, 45, 50],

'income': [50000, 60000, 45000, 70000, 80000, 90000, 65000, 62000, 75000, 85000],

'age\_group': ['20-30', '30-40', '20-30', '40-50', '50-60', '50-60', '30-40', '20-30', '40-50', '50-60']

}

# Define data in DataFrame

df = pd.DataFrame(data)

* **data**: A dictionary with three keys: **age**, **income**, and **age\_group**. The values corresponding to each key are lists of numbers representing the age, income, and age group of 10 individuals.
* **df**: A **Pandas DataFrame** is created from the data dictionary, allowing easy manipulation and analysis of the data.

### ****Step 2: Group Data by Age Group and Compute Summary Statistics for Income****

# Group the data by age\_group and compute summary statistics for 'income'

summary\_stats = df.groupby('age\_group')['income'].describe()

# Print summary

print(summary\_stats)

* **df.groupby('age\_group')**: Groups the DataFrame by the **age\_group** column, meaning the data will be divided into subsets based on the unique values in the **age\_group** column (e.g., '20-30', '30-40', etc.).
* **['income']**: After grouping, we are selecting the **income** column for further analysis.
* **.describe()**: This function generates summary statistics for the **income** column within each **age\_group**. It calculates the following:
  + **count**: The number of non-null values in the **income** column.
  + **mean**: The average income for each age group.
  + **std**: The standard deviation of income within each age group.
  + **min**: The minimum income value within each age group.
  + **25%**: The 25th percentile (1st quartile) of income values.
  + **50%**: The median (2nd quartile) of income values.
  + **75%**: The 75th percentile (3rd quartile) of income values.
  + **max**: The maximum income value within each age group.
* The results are stored in **summary\_stats**, which is then printed out.

### ****Step 3: Calculate the Median Income by Age Group****

# Group the data by age\_group; Select income column for each of the groups created; Calculate median for income

median\_income = df.groupby('age\_group')['income'].median()

# Print median income

print("Median Income by Age Group:")

print(median\_income)

* **.groupby('age\_group')**: Again, the DataFrame is grouped by the **age\_group** column.
* **['income']**: We select the **income** column to perform operations on it.
* **.median()**: This function calculates the median income for each **age\_group**.
* The resulting **median\_income** is then printed out.

### ****Step 4: Print Column Names****

print("Column Names:", df.columns)

* **df.columns**: This retrieves the names of all columns in the DataFrame (age, income, and age\_group), and prints them.

### ****Step 5: Modify the Dataset with Repeated Values****

# Modified dataset with repeated values

data = {

'age': [25, 30, 25, 40, 55, 60, 33, 28, 45, 50, 25, 30, 28, 30, 25],

'income': [50000, 60000, 50000, 70000, 80000, 90000, 65000, 62000, 75000, 85000, 50000, 60000, 62000, 70000, 75000],

'age\_group': ['20-30', '30-40', '20-30', '40-50', '50-60', '50-60', '30-40', '20-30', '40-50', '50-60', '20-30', '30-40', '20-30', '30-40', '20-30']

}

# Define data in DataFrame

df = pd.DataFrame(data)

* The dataset is modified by adding more repeated values for **age**, **income**, and **age\_group**. This simulates a larger dataset, allowing for the calculation of modes and other statistics.

### ****Step 6: Calculate the Mode for Age and Income****

# Calculate the mode for each column

mode\_age = df['age'].mode()

mode\_income = df['income'].mode()

print(f"Mode of Age: {mode\_age.values}")

print(f"Mode of Income: {mode\_income.values}")

* **df['age'].mode()**: This computes the **mode** of the **age** column, which is the value that appears most frequently. If there are multiple values with the highest frequency, all of them are returned.
* **df['income'].mode()**: Similarly, the **mode** of the **income** column is computed.
* **mode\_age.values** and **mode\_income.values** are printed to show the mode values for both **age** and **income**.

### ****Explanation of the Output****

1. **Summary Statistics for Income by Age Group**: This section provides a detailed statistical summary (mean, standard deviation, min, max, etc.) of income within each age group. For example, the **mean** of income in the '20-30' age group could be $55,000, while the **standard deviation** might indicate how spread out the income values are in that group.
2. **Median Income by Age Group**: This shows the median income value for each age group, which is useful for understanding the central tendency in the presence of outliers.
3. **Mode of Age and Income**: The **mode** tells us the most frequently occurring values in the **age** and **income** columns. For example, if the most frequent **age** is 25, that would be the **mode** of the age.

### ****Problem Statement - Part 2: Descriptive Statistics on the Iris Dataset****

#### ****Objective****

The task here is to perform descriptive statistics on the famous **Iris dataset**, specifically focusing on three species: **Iris-setosa**, **Iris-versicolor**, and **Iris-virginica**. For each species, we need to display:

1. Basic statistical details like **percentiles**, **mean**, **standard deviation**, and other descriptive statistics.
2. Detailed summary statistics of specific features like **sepal width** for each species.
3. Calculate the **median** for each numeric feature, grouped by species.
4. Find the **mode** for the **sepal width**.

### ****Step-by-Step Explanation of the Code for Part 2****

### ****Step 1: Load the Iris Dataset****

# Load iris.csv in the DataFrame

df = pd.read\_csv('iris.csv')

print(df.head()) # Print first 5 rows

* **pd.read\_csv('iris.csv')**: This function reads the data from the **iris.csv** file and loads it into a **Pandas DataFrame**. The **iris.csv** dataset contains data on various Iris flowers, including features such as **sepal length**, **sepal width**, **petal length**, and **petal width**, along with the species of each flower.
* **df.head()**: This prints the first 5 rows of the dataset, allowing us to inspect the structure of the data (column names and some values).

### ****Step 2: Group the Data by Species and Display Summary Statistics****

# Group the data by species and display summary statistics

summary\_stats\_species = df.groupby('variety').describe()

# Print summary statistics by species

print(summary\_stats\_species)

* **df.groupby('variety')**: Groups the data by the **'variety'** column, which contains the species of the flowers (Iris-setosa, Iris-versicolor, Iris-virginica).
* **.describe()**: After grouping, this function computes summary statistics for each species, including:
  + **count**: The number of flowers in each species.
  + **mean**: The mean of each numeric column (e.g., **sepal length**, **sepal width**, etc.) for each species.
  + **std**: The standard deviation of each numeric column within each species.
  + **min**: The minimum value of each column for each species.
  + **25%**: The 25th percentile (1st quartile) of the numeric columns for each species.
  + **50%**: The median (2nd quartile) of each column for each species.
  + **75%**: The 75th percentile (3rd quartile) of the numeric columns for each species.
  + **max**: The maximum value of each column for each species.
* The result is stored in **summary\_stats\_species**, and it is printed for inspection.

### ****Step 3: Compute Specific Percentiles for Each Species****

# Compute specific percentiles and statistics

percentiles = df.groupby('variety').quantile([0.25, 0.5, 0.75])

# Display percentiles

print("\nPercentiles by Species:")

print(percentiles)

* **df.groupby('variety').quantile([0.25, 0.5, 0.75])**: This computes specific percentiles (25th, 50th, and 75th) for each species. The **quantile** function is used to get specific percentiles for numeric columns.
* The resulting percentiles for each species are printed out to show the distribution of values across different percentiles.

### ****Step 4: Group by Species and Display Summary Statistics for Sepal Width****

# Group the data by variety; Select sepal.width column for each of the groups created; Display summary statistics

summary\_stats\_species = df.groupby('variety')['sepal.width'].describe()

print("\nSummary Statistics by Species for Sepal Width:")

print(summary\_stats\_species)

* **df.groupby('variety')['sepal.width']**: The data is grouped by **variety** (species), and we focus on the **sepal.width** column to get its statistics.
* **.describe()**: This provides summary statistics for **sepal width** specifically. It will include count, mean, standard deviation, min, max, and percentiles for each species' sepal width.

### ****Step 5: Compute Median for Each Numeric Column by Species****

# Group the data by variety and compute the median for numeric columns

median\_values = df.groupby('variety').median()

print("Median Values by Species:")

print(median\_values)

* **df.groupby('variety')**: Groups the data by species.
* **.median()**: This function computes the median value for each numeric column (e.g., **sepal length**, **sepal width**, etc.) for each species. The result is printed to show the median values for each species.

### ****Step 6: Compute the Median Sepal Length by Species****

# Group the data by variety; Select sepal.length column for each of the groups created; Display median

median\_sepal\_length = df.groupby('variety')['sepal.length'].median()

print("Median Sepal Length by Species:")

print(median\_sepal\_length)

* **df.groupby('variety')['sepal.length']**: Groups the data by **variety** and selects the **sepal length** column.
* **.median()**: Calculates the median of **sepal length** for each species. The result is printed for each species.

### ****Step 7: Calculate the Mode for Sepal Width****

# Calculate & print mode for sepal.width

mode\_width = df['sepal.width'].mode()

print(f"Mode of Sepal Width: {mode\_width.values}")

* **df['sepal.width'].mode()**: This function computes the **mode** for **sepal width**. The mode represents the most frequent value in the column.
* **mode\_width.values**: The mode value(s) are printed out. If there are multiple modes, they will all be printed.

### ****Explanation of the Output****

1. **Summary Statistics by Species**: This section provides detailed summary statistics for each numeric column (sepal length, sepal width, etc.) within each species. It includes metrics such as **mean**, **standard deviation**, **min**, **max**, and various percentiles.
2. **Percentiles by Species**: This shows the 25th, 50th (median), and 75th percentiles for each numeric feature in the dataset, broken down by species. It helps in understanding the distribution of values for each species.
3. **Summary Statistics for Sepal Width**: This section shows the descriptive statistics specifically for the **sepal width** variable. It's useful to observe how the sepal width varies across different species.
4. **Median Values by Species**: The median value for each numeric column (e.g., **sepal length**, **sepal width**) for each species is printed. The median is a central tendency measure, which helps to understand the central value of each feature for each species.
5. **Median Sepal Length by Species**: This specifically focuses on the **sepal length** and displays the median value for each species.
6. **Mode of Sepal Width**: The **mode** represents the most frequently occurring value in the **sepal width** column. This is useful to understand which value appears most often in the dataset.

### ****Conclusion****

* **Part 1** helped us analyze a custom dataset by grouping it by **age group** and calculating descriptive statistics such as **mean**, **median**, **mode**, and others.
* **Part 2** focuses on the **Iris dataset**, where we analyzed various numeric features for three species by computing **summary statistics**, **percentiles**, **median values**, and **mode**.

Let me know if you'd like further clarification or assistance with any part of the code or output!