**Exp 3**

User

**You**

**import numpy**

**speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]**

**x = numpy.mean(speed)**

**print(x)**

ChatGPT

**ChatGPT**

Sure, here's the meaning of each line in short:

1. **import numpy**: This line imports the numpy library, which is used for numerical computing in Python.
2. **speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]**: This line creates a list called **speed** containing a series of numerical values representing speeds.
3. **x = numpy.mean(speed)**: This line calculates the mean (average) of the values in the **speed** list using the **mean()** function from the numpy library, and assigns the result to the variable **x**.
4. **print(x)**: This line prints the value of **x**, which is the mean speed calculated in the previous line.

User

**You**

**import numpy**

**speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]**

**x = numpy.median(speed)**

**print(x)**

ChatGPT

**ChatGPT**

Here's the explanation for each line:

1. **import numpy**: Imports the numpy library for numerical computing.
2. **speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]**: Creates a list called **speed** containing a series of numerical values representing speeds.
3. **x = numpy.median(speed)**: Calculates the median (middle value) of the values in the **speed** list using the **median()** function from the numpy library, and assigns the result to the variable **x**.
4. **print(x)**: Prints the value of **x**, which is the median speed calculated in the previous line.

User

**You**

**import numpy**

**speed = [99,86,87,88,86,103,87,94,78,77,85,86]**

**x = numpy.median(speed)**

**print(x)**

ChatGPT

**ChatGPT**

Sure, here's the explanation for each line:

1. **import numpy**: Imports the numpy library for numerical computing.
2. **speed = [99,86,87,88,86,103,87,94,78,77,85,86]**: Creates a list called **speed** containing a series of numerical values representing speeds.
3. **x = numpy.median(speed)**: Calculates the median (middle value) of the values in the **speed** list using the **median()** function from the numpy library, and assigns the result to the variable **x**.
4. **print(x)**: Prints the value of **x**, which is the median speed calculated in the previous line.

User

**You**

**from scipy import stats**

**speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]**

**x = stats.mode(speed)**

**print(x)**

ChatGPT

**ChatGPT**

Here's the explanation for each line:

1. **from scipy import stats**: Imports the **stats** module from the **scipy** library, which provides statistical functions and tools.
2. **speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]**: Creates a list called **speed** containing a series of numerical values representing speeds.
3. **x = stats.mode(speed)**: Calculates the mode (the most frequent value) of the values in the **speed** list using the **mode()** function from the **stats** module, and assigns the result to the variable **x**.
4. **print(x)**: Prints the value of **x**, which contains both the mode value and its frequency as a mode-result object.

User

**You**

**n\_num = [1, 2, 3, 4, 5]**

**n = len(n\_num)**

**get\_sum = sum(n\_num)**

**mean = get\_sum/n print("Mean / Average is : " + str(mean))**

ChatGPT

**ChatGPT**

Here's the breakdown for each line:

1. **n\_num = [1, 2, 3, 4, 5]**: Defines a list called **n\_num** containing a series of numerical values.
2. **n = len(n\_num)**: Calculates the length of the list **n\_num** and assigns it to the variable **n**.
3. **get\_sum = sum(n\_num)**: Calculates the sum of all elements in the list **n\_num** using the **sum()** function and assigns it to the variable **get\_sum**.
4. **mean = get\_sum/n**: Calculates the mean (average) of the values in the list by dividing the sum (**get\_sum**) by the length of the list (**n**), and assigns it to the variable **mean**.
5. **print("Mean / Average is : " + str(mean))**: Prints out a message including the calculated mean, converted to a string using **str()** function, concatenated with the message indicating what it represents.

User

**You**

**n\_num = [1, 2, 3, 4, 5]**

**n = len(n\_num)**

**n\_num.sort()**

**if n % 2 == 0:**

**median1 = n\_num[n//2]**

**median2 = n\_num[n//2-1]**

**median = (median1 + median2)/2**

**else:**

**median = n\_num[n//2] print("Median is : " + str(median))**

ChatGPT

**ChatGPT**

Here's the explanation for each line:

1. **n\_num = [1, 2, 3, 4, 5]**: Defines a list called **n\_num** containing a series of numerical values.
2. **n = len(n\_num)**: Calculates the length of the list **n\_num** and assigns it to the variable **n**.
3. **n\_num.sort()**: Sorts the list **n\_num** in ascending order.
4. **if n % 2 == 0:**: Checks if the length of the list is even.
5. **median1 = n\_num[n//2]**: Calculates the first median value by accessing the element at the index **n//2** in the sorted list.
6. **median2 = n\_num[n//2-1]**: Calculates the second median value by accessing the element at the index **n//2-1** in the sorted list.
7. **median = (median1 + median2)/2**: Calculates the median (middle value) for even-length lists by taking the average of the two middle values.
8. **else:**: If the length of the list is odd, the following line will execute.
9. **median = n\_num[n//2]**: Calculates the median for odd-length lists by accessing the middle element directly.
10. **print("Median is : " + str(median))**: Prints out a message including the calculated median, converted to a string using **str()** function, concatenated with the message indicating what it represents.

User

**You**

**from collections import Counter**

**n\_num = [1, 2, 3, 4, 5, 5]**

**n = len(n\_num)**

**data = Counter(n\_num)**

**get\_mode = dict(data)**

**mode = [k for k, v in get\_mode.items() if v == max(list(data.values()))]**

**if len(mode) == n:**

**get\_mode = "No mode found"**

**else:**

**get\_mode = "Mode is/are : " + ', '.join(map(str,mode)) print(get\_mode)**

ChatGPT

**ChatGPT**

Here's the explanation for each line:

1. **from collections import Counter**: Imports the **Counter** class from the **collections** module. **Counter** is used for counting the occurrences of elements in a list.
2. **n\_num = [1, 2, 3, 4, 5, 5]**: Defines a list called **n\_num** containing a series of numerical values.
3. **n = len(n\_num)**: Calculates the length of the list **n\_num** and assigns it to the variable **n**.
4. **data = Counter(n\_num)**: Creates a **Counter** object named **data** from the list **n\_num**, which counts the occurrences of each element.
5. **get\_mode = dict(data)**: Converts the **Counter** object **data** into a dictionary named **get\_mode**.
6. **mode = [k for k, v in get\_mode.items() if v == max(list(data.values()))]**: Finds the mode(s) by iterating over the items of **get\_mode** and selecting the key(s) (element(s)) that have the maximum value (frequency) in the dictionary.
7. **if len(mode) == n:**: Checks if the length of the **mode** list is equal to the length of the original list **n\_num**.
8. **get\_mode = "No mode found"**: If all elements in the list have the same frequency, sets the **get\_mode** variable to indicate that there is no mode.
9. **else:**: If there is at least one mode, the following line will execute.
10. **get\_mode = "Mode is/are : " + ', '.join(map(str,mode))**: Constructs a string representation of the mode(s) by joining the elements of the **mode** list into a single string separated by commas. If there are multiple modes, it separates them with commas and appends the string "Mode is/are :" to the beginning.
11. **print(get\_mode)**: Prints the string representation of the mode(s).

User

**You**

**import pandas as pd**

**df = pd.DataFrame({'A' : ['a', 'b', 'c', 'c', 'a', 'b'], 'B' : [0, 1, 1, 0, 1, 0]}, dtype = "category")**

**df.dtypes**

ChatGPT

**ChatGPT**

Here's the explanation for each line:

1. **import pandas as pd**: Imports the pandas library and aliases it as **pd**. Pandas is a powerful data manipulation library for Python.
2. **df = pd.DataFrame({'A' : ['a', 'b', 'c', 'c', 'a', 'b'], 'B' : [0, 1, 1, 0, 1, 0]}, dtype = "category")**: Creates a pandas DataFrame named **df** with two columns 'A' and 'B'. Column 'A' contains categorical data (strings) and column 'B' contains numerical data (integers). The **dtype="category"** argument specifies that the data type of column 'A' should be treated as categorical.
3. **df.dtypes**: Retrieves the data types of each column in the DataFrame **df**. It returns a Series containing the data types of each column.

User

**You**

**print(df)**

**print(df.groupby(['A']). count().reset\_index())**

ChatGPT

**ChatGPT**

Certainly! Here's the explanation for each line:

1. **print(df)**: This line prints the entire DataFrame **df**, which contains two columns labeled 'A' and 'B', with their respective values.
2. **print(df.groupby(['A']).count().reset\_index())**: This line performs the following operations:
   * **df.groupby(['A'])**: Groups the DataFrame **df** by the values in column 'A'.
   * **.count()**: Counts the number of occurrences of each unique value in column 'A' within each group.
   * **.reset\_index()**: Resets the index of the resulting DataFrame, converting the grouped values of column 'A' into a regular column again and giving the count of occurrences a new index.

The line then prints this modified DataFrame, which displays each unique value of column 'A' along with the count of its occurrences in the original DataFrame.

User

**You**

**import pandas as pd**

**df = pd.DataFrame({'A' : ['a', 'b', 'c', 'c', 'a', 'b'], 'B' : [0, 1, 1, 0, 1, 0], 'C' : [7, 8, 9, 5, 3, 6]})**

**df['A'] = df['A'].astype('category')**

**print(df)**

**print(df.groupby(['A', 'B']).mean().reset\_index())**

ChatGPT

**ChatGPT**

Here's the explanation for each line:

1. **import pandas as pd**: Imports the pandas library and aliases it as **pd**. Pandas is a powerful data manipulation library for Python.
2. **df = pd.DataFrame({'A' : ['a', 'b', 'c', 'c', 'a', 'b'], 'B' : [0, 1, 1, 0, 1, 0], 'C' : [7, 8, 9, 5, 3, 6]})**: Creates a pandas DataFrame named **df** with three columns 'A', 'B', and 'C'. Column 'A' contains categorical data (strings), column 'B' contains numerical data (integers), and column 'C' contains numerical data (integers) as well.
3. **df['A'] = df['A'].astype('category')**: Converts the data type of column 'A' to a categorical data type using the **astype()** method. This is done to explicitly define column 'A' as a categorical variable.
4. **print(df)**: Prints the entire DataFrame **df**, showing all columns and their respective values.
5. **print(df.groupby(['A', 'B']).mean().reset\_index())**: This line performs the following operations:
   * **df.groupby(['A', 'B'])**: Groups the DataFrame **df** by unique combinations of values in columns 'A' and 'B'.
   * **.mean()**: Computes the mean (average) of the numerical values in each group.
   * **.reset\_index()**: Resets the index of the resulting DataFrame, converting the grouped values of columns 'A' and 'B' into regular columns again and giving the mean values a new index.

The line then prints this modified DataFrame, which displays each unique combination of values in columns 'A' and 'B' along with the mean value of column 'C' for each combination.

User

**You**

**import pandas as pd**

**data = pd.read\_csv(r"D:\College\TE\SEM-2\Practical\DSBDA\3\Iris.csv")**

**print('Iris-setosa')**

**setosa= data['Species'] == 'Iris-setosa'**

**print(data[setosa].describe())**

**print('\nIris-versicolor')**

**versicolor= data['Species'] == 'Iris-versicolor'**

**print(data[versicolor].describe())**

**print('\nIris-virginica')**

**virginica = data['Species'] == 'Iris-virginica'**

**print (data[virginica].describe())**

ChatGPT

**ChatGPT**

This script is loading data from a CSV file named "Iris.csv", performing some analysis on it, and printing descriptive statistics for each species of iris (setosa, versicolor, and virginica). Here's the explanation for each part:

1. **import pandas as pd**: Imports the pandas library and aliases it as **pd**. Pandas is a powerful data manipulation library for Python.
2. **data = pd.read\_csv(r"D:\College\TE\SEM-2\Practical\DSBDA\3\Iris.csv")**: Reads the data from the CSV file located at the specified path and stores it in a DataFrame named **data**.
3. **print('Iris-setosa')**: Prints the label "Iris-setosa" to indicate the start of analysis for this species.
4. **setosa= data['Species'] == 'Iris-setosa'**: Creates a boolean mask **setosa** that filters rows where the 'Species' column is equal to 'Iris-setosa'.
5. **print(data[setosa].describe())**: Prints descriptive statistics (count, mean, standard deviation, minimum, quartiles, and maximum) for the subset of data where the species is 'Iris-setosa'.
6. **print('\nIris-versicolor')**: Prints the label "Iris-versicolor" to indicate the start of analysis for this species.
7. **versicolor= data['Species'] == 'Iris-versicolor'**: Creates a boolean mask **versicolor** that filters rows where the 'Species' column is equal to 'Iris-versicolor'.
8. **print(data[versicolor].describe())**: Prints descriptive statistics for the subset of data where the species is 'Iris-versicolor'.
9. **print('\nIris-virginica')**: Prints the label "Iris-virginica" to indicate the start of analysis for this species.
10. **virginica = data['Species'] == 'Iris-virginica'**: Creates a boolean mask **virginica** that filters rows where the 'Species' column is equal to 'Iris-virginica'.
11. **print (data[virginica].describe())**: Prints descriptive statistics for the subset of data where the species is 'Iris-virginica'.