**INT 7623**

**Data Science for Business**

Due 2/26/2024 11:59 PM

Assignment 2

(40 points)

Note: Please refer to the syllabus for the submission format.

Python libraries are comprehensive, popular, and widely used tools that will assist you in working with data. In this assignment, you can use **Numpy** library to answer the question.

In this assignment, you have to include the following:

* **Python file.** Comment your code. Combine all the files including the word document in one folder and zip it (.zip)
* Screen shot of the code under each part.
* Screen shot of the output under each part.

**First Question (24 points):**

Consider the following numbers:

24, 5, 15, 60, 54, 82, 99, 80, 70, 98, 93, 60, 33, 22, 65, 61, 51, 58, 83, 86, 42, 67, 60

Write a python code to generate the following statistical Analysis (Create a list and name it *mylist*) Make sure to output the result (print function)

1. Mean
2. Median
3. Mode
4. Range
5. Variance
6. Standard deviation.

**Overall code:**

#####################

## Dalton Murray #

## 02/16/2024 #

## A2 - Exercise 1 #

#####################

import sys # Required for system functions

import numpy as np # Imports the numpy library and sets an alias to np

import statistics as stats # Imports the statistics library and sets an alias to sts

# Defines the \_\_main\_\_ function

def \_\_main\_\_():

# Creates a variable called myList and sets the list data to the provided data

myList = [24, 5, 15, 60, 54, 82, 99, 80, 70, 98, 93, 60, 33, 22, 65, 61, 51, 58, 83, 86, 42, 67, 60]

## Part 1

# Calculates the mean of the list using numpy mean function then prints it out

meanList = np.mean(myList)

print("The mean is:", meanList)

## Part 2

# Calculates the median of the list using numpy median function then prints it out

medianList = np.median(myList)

print("The median is:", medianList)

## Part 3

# Mode calculation method 1

# The assignment says we can use the numpy library but it doesn't say we cannot use other libraries, this would be the most simple way to calculate the mode

# by using a preexisting library with a function already built in, however, we can do this with regular numpy and python alone shown in method 2

modeList = stats.mode(myList)

print("Method 1\nThe mode is:", modeList)

# Mode calculation method 2

# This method uses only numpy and uses in-line lambda with bins

modeList = np.apply\_along\_axis(lambda x: np.bincount(x).argmax(), axis = 0, arr = myList)

print("Method 2\nThe mode is:", modeList)

# Other methods of mode calculation

# We did learn another method in class using a function with a counter and setting it to use the max function however,

# I opted not to use this in order to expand upon/learning different and more ways to do it and I didn't want to just

# take the method given to us and use it, similar with the other functions here, however, I didn't want to include

# 3 different ways to do every function so I just used numpy instead for them where easiest to

## Part 4

# Range calulcation method 1

# This uses numpy to calculate the range of the list and then prints it out, however, this isn't a way we learned to do it, so I will include the other easy way too

rangeList = np.ptp(myList)

print("Method 1\nThe range is:", rangeList)

# Range calculation method 2

# This performs the range calculation using other calculation other than the built-in range calculator, this method was taught in class

rangeList = np.max(myList, axis = 0) - np.min(myList, axis = 0)

print("Method 2\nThe range is:", rangeList)

## Part 5

# This calculates the variance using the numpy variance method and then prints it out (population variance not sample variance)

varianceList = np.var(myList)

print("The variance is:", varianceList)

## Part 6

# This calculates the standard deviation using numpy functions (population not sample)

standardDeviationList = np.std(myList)

print("The standard deviance is:", standardDeviationList)

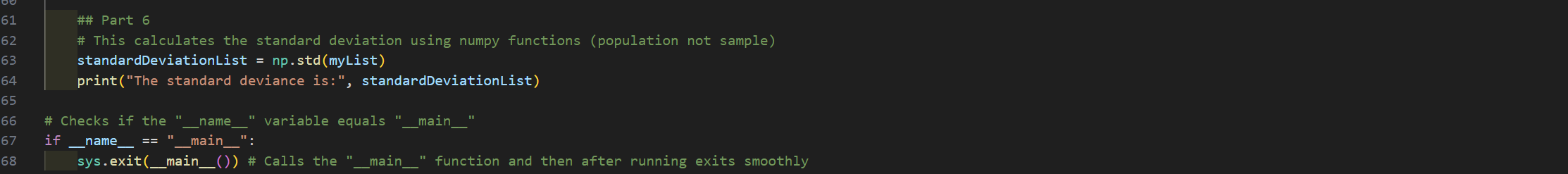
# Checks if the "\_\_name\_\_" variable equals "\_\_main\_\_"

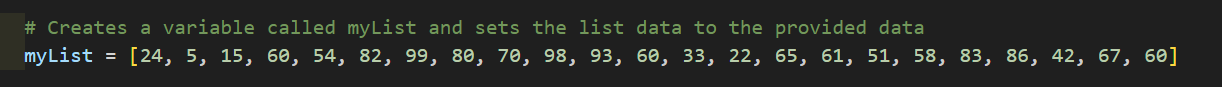
if \_\_name\_\_ == "\_\_main\_\_":

sys.exit(\_\_main\_\_()) # Calls the "\_\_main\_\_" function and then after running exits smoothly

A screenshot of a computer program

Description automatically generated





1. Mean

A black background with green text

Description automatically generated



For this first part I already have numpy and the list setup with the already given data and run it through numpy’s mean function then I print it out.

2. Median

A black screen with green text

Description automatically generated



This part is doing the same as the first part except is doing median instead of mean.

3. Mode

A screenshot of a computer

Description automatically generated

A black background with white text

Description automatically generated

The assignment said I can use the numpy library however it didn’t say I cannot use any other library so I wanted to show that the statistics library has the mode function, however, I also wanted to show a way to do it strictly with numpy. There are a few methods of calculating mode such as my first one using stats, the second one with numpy only, and the third using the way the slides showed it, however, there are also many other ways to calculate mode. My second method is using numpy’s apply along axis and I am using the lambda function. I use bincount in order to count the occurrences of each number in the list to find the index value with the highest count by using argmax. The axis function tells it to apply along the first axis and arr tells it in which array or list for it to work in.

4. Range

A black background with green and yellow text

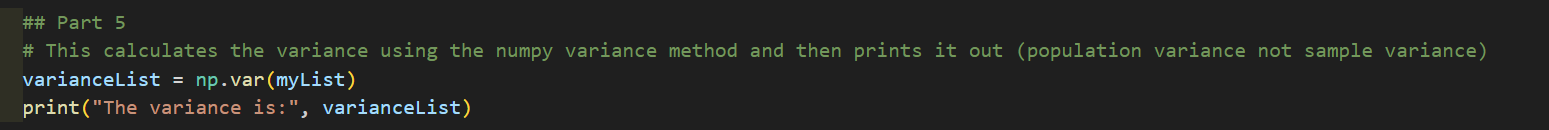
Description automatically generated

A black background with white text

Description automatically generated

Here I also have two different ways to calculate the range, the first one using numpy’s built in range function which as far as I could tell wasn’t taught in class and the second uses numpy to manually calculate the range which was taught in class.

5. Variance





This part also just uses numpy’s variance function calculation and then prints it out.

6. Standard deviation.

A screen shot of a computer

Description automatically generated



This part too just uses numpy’s built in standard deviation calculation and prints it out.

**Second Question (16 points):**

1. Consider the attached CSV file (Sample.text)
   1. Read the file using open() and output the content to the screen. ( 5points)
   2. Append the following sentence at the end of the file “This is my second assignment” (5points)
2. Create new file name it (myfile.txt) and write some information like your name, class number/name, and assignment number, etc. Output the content to the screen. (6 points)

**Overall code**

#####################

## Dalton Murray #

## 02/16/2024 #

## A2 - Exercise 2 #

#####################

import sys # Required for system functions

# Defines the \_\_main\_\_ function

def \_\_main\_\_():

## Part 1 - A

# Using with allows for the automatic handling of the file better/closing the file rather than typing additional code to close it

# What this does is opens the file in read mode and sets it to the variance file, it then reads everything in the file and sets it

# to the variable contents and then prints it out, since I am using with I do not have to worry about using file.close()

with open("./Sample.txt", "r") as file:

contents = file.read()

print("Method 1\n", contents)

# Another method of reading the contents without using with

file = open("./Sample.txt", "r")

contents = file.read()

print("Method 2\n", contents)

file.close()

## Part 1 - B

# Rather than including the appending to the first part, I will include the first part but in appending mode just to show that I know

# how to do both read only and appending modes, I will also only do this using with

with open("./Sample.txt", "a") as file:

file.write("This is my second assignment!\n")

with open("./Sample.txt", "r") as file:

contents = file.read()

print("Appended file\n", contents)

# Although the above method is efficient, it has additional unnecessary lines because it has to open it in appending mode then close then put it in reading then close again

# This is where different modes come into play to be more efficient

# The mode a+ stands for appending mode plus read operations

with open("./Sample.txt", "a+") as file:

file.write("This is my second assignment!")

# Appending mode opens the file at the bottom/end of the position and then writes to it, because of this I then have to change the position I read from in order

# to actually get the contents of the whole file

file.seek(0)

contents = file.read()

print("Appended file\n", contents)

# This does make for a weird file with the two written things, the first one when opening in only append mode, on the same line, this is because I'm not telling it to go to

# the next line yet, which would be done by adding \n at the first write statement which I have included now even though in the instructions it doesn't say to do so, in order to make

# my program run and look better

## Part 2

# This creates a new file called myfile.txt if it does not exist, it then puts into write+ mode so I can perform both writing and reading of the file at the same time to make sure everytime I run,

# the file doesn't keep infinitely expanding like a+ mode, where I then create a first row having the columns of the file

# I then perform another write with the contents for the first row, also using tabs for spacing, I then go back to the top of the file and read and print it out

with open("myfile.txt", "w+") as file:

file.write("firstName\tlastName\tclassNumber\tclassName\tassigntmentNumber\n")

file.write("Dalton\tMurray\tINT7623\tData Science for Business\t02")

file.seek(0)

contents = file.read()

print("New file\n", contents)

# Checks if the "\_\_name\_\_" variable equals "\_\_main\_\_"

if \_\_name\_\_ == "\_\_main\_\_":

sys.exit(\_\_main\_\_()) # Calls the "\_\_main\_\_" function and then after running exits smoothly

A screen shot of a computer program

Description automatically generated

1 – A.

**A screen shot of a computer

Description automatically generated**

**A screenshot of a computer screen

Description automatically generated**

Part 1 – a asks to read and print out the contents of the sample.txt file, which I am doing with two separate methods. The first method is using the with function which automatically handles some file needs by opening and closing it automatically. I then tell it to open in read only mode, read all of the contents, set it to the variable contents, and then print it out. The second method does the same as the first method, however, I have to explicitly tell it to close the file.

1 – B.

A screen shot of a computer

Description automatically generated

A screenshot of a computer program

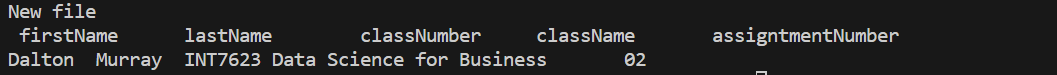
Description automatically generated

Part 1 - b I also made with two different methods. The first method it has to open the files twice one in appending mode and the second in read only mode. The appending mode just writes at the end of the file the given text and in the read mode I tell it to read and print it all out. The second method is where I use a+ mode, or appending plus, which allows me to not need to open and close the file twice. It tells it to append the given statement and then seek the start of the file and read it all and print it out.

2.

A black screen with green text

Description automatically generated



For part 2 all I am doing is opening the file in w+, write plus mode which tells it to make a new file if it doesn’t exist and then I am writing to the line with columns and then adding a new line with given information for the columns such as first name, last name, class number, class name, and assignment number. I am also doing all of this with tab formatting. I then seek the start of the file, read it all, and print it all out.