Homework 2 – Statistics

Arya Rahmanian

COMP 614

September 22, 2024

CodeSkulptor Link: https://py3.codeskulptor.org/#user309_LdTDlu52RpJHXdV.py

1. **Question 2.C.i: In the case where n<m, how many pieces of data from the input list should be used to compute an individual element in the output list? Express your answer in terms of m and/or n.**

   n pieces of data should be used to calculate the "moving" average if $m > n$

2. **Question 2.C.ii: Without using iteration, how could you extract a contiguous sequence of elements from the input list? Provide a Python expression in terms of data, i, n, and/or m that will give you the sequence of elements that will be used to compute the ith element in the output list in the case that n<m. Your expression should work for any arbitrary n<m and any arbitrary value of i that is greater than or equal to zero and less than the size of the output list.**

   I could get the contiguous sequence of element by using index slicing. The slice would look something like data[i:i+m]. If i+m > n, then python will return a slice up until the end of the list and not throw any errors.

3. **Question 2.C.iii: In the case where n<m, how many pieces of data will be in your output list? Why? Express your answer in terms of m and/or n.**

Just one item will be in the list. Since $m > n,$ I will only compute one average and that will be for all n elements.

1. **Question 2.D.i: For clean_with_deletion, describe in English a strategy for generating a deep copy of the original list while excluding rows that originally contained None**

   To accomplish clean_with_deletion, I will create a new 2D list with the first element as the first element of the original list (list of column headers). After that, I will loop through the original list row's starting from index 1, and if there are any **None** values in the rows, I will not append it to the new list, and if there are no **None** values in the row, I append it to the new list.

2. **Question 2.D.ii: For column_avgs, provide a Python expression for the number of columns in the data set. Explain your answer.**

   columns = len(data[0])

   This is because data[0] is a list of the column headers. So, the length of how many column headers we have is the number of columns in the list.

3. **Question 2.D.iii: For column_avgs, describe a strategy for building a list of all of the numeric elements in the ith column. You may include Python expression(s) to help explain your answer, but your description should be in English. If iteration is involved, you should clearly specify what you are iterating over (i.e. what structure and/or what indices, including whether they are inclusive or exclusive).**

   To complete column_avgs, I will iterate over each column index with:

   for idx in range(len(data[0]))

With that for-loop, I can access the i-th element in each row that corresponds to the column I am interested in. With each index corresponding to the column row, I am able to collect each column in a list by iterating over each row, accessing the i-th element that I want to collect with:

column = [row[idx] for row in data[1:]]

And to remove all the None values, I would then iterate over every item in the column list I created above, removing all the None values. When it is cleaned, I will have a list of all the items in i-th row that I can average.

Reflection:

1. The two most important concepts and skills this assignment focused on are **list indexing/slicing** and **data filtering**. This assignment reinforced the fact that real-world data is often not clean and needs to be filtered to ensure it fits our expectations. Lists are incredibly important because they form the foundation of data science in Python, as many other libraries, such as Pandas and NumPy, build on lists.

2. Data filtering and aggregation is very important because data will never be perfect, and programmers must include precautions to make sure everything is formatted as expected before moving forward. This is a recurring theme with data analysis, and we should not expect perfect data. Additionally, python lists are a fundamental part of the language, so understanding lists slicing and manipulation is vital.

3. I believe I performed well on the entire assignment. My work with list indexing and slicing felt concise and clear, creating less clutter for anyone else who needs to understand my thought process. If I could do this assignment over, I would ensure I fully understand the concept of deep copy, as I encountered issues with my code not running properly because I failed to make deep copies of the original data list.

4. I am comfortable enough with these concepts to teach to a peer. I have prior experience with python and if a peer needs assistance, I would be glad to help as I am confident with lists.