Code.Hub

Week 1 exercise

You will be given a table in CSV format. This format allows us to store a table in a human-readable text file. The first line represents the first row of the table, the second line represents the second row etc. The values for each of the columns are separated by commas (dots are reserved for decimals).

For example the first 3 rows and 10 columns of the table would look like this:

17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471	0.2419
20.57	17.77	132.9	1326	0.08474	0.07864	0.0869	0.07017	0.1812
19.69	21.25	130	1203	0.1096	0.1599	0.1974	0.1279	0.2069

The goal of this exercise is to get you acquainted with Python's built-in functionality. As such, for this exercise **the use of external libraries is prohibited**.

The tasks are the following:

1. **Read the file** in python with the use of python's built-in function: `open()`.
    - The easiest way to read it is as a list of lists, meaning that we will have an outer list, whose elements are also lists. Each inner list will represent a row. The outer list will simply be the collection of inner lists, i.e. the collection of rows.
    - The list-of-lists format, however, isn't convenient for what we want to do with the next tasks.
    - Note that at no point during this exercise will we represent the table as the 2D structure that it is.
    - Hint: the string method `str.split()` might be of use

2. **Answer** the following two questions about the shape of the table:
    - How many rows does it have?
    - How many columns does it have?
    - Hint: the built-in function `len()` might be of use.

3. Compute the **mean** of **each column**.

- Here is where the format we chose isn't ideal. For example to compute the mean of the first column, we'll have to sum the elements of the first column and divide them by the total number of rows in the table. However these elements are all stored in separate lists. You will need to find a way to overcome this issue.
- Hint: the built-in function `len()` might be of use.

4. Compute the **variance** of **each column**.
    - This calculation requires the mean and it faces the same difficulties as we mentioned previously.

5. **Print** the results.

6. [Bonus 1] Compute the **median** of each column.
    - This is a lot trickier than the mean.
    - You might have to change the way you represent your table.

7. [Bonus 2] Incorporate the above functionality into a **python class**.
    - Create a class called `Dataset` that loads and stores the data.
    - This class should have methods for computing the column means, medians, etc.
    - During instantiation it should:
        i. read the dataset
        ii. compute the above statistics
        iii. print these to the user