

# assignment\_1

November 28, 2023

## 1 Assignment #1

In this assignment, you will use your knowledge of Python to assess the efficacy of an imaginary new drug for arthritis inflammation flare-ups.

You will use data from an imaginary clinical trial that had 60 patients take this new drug. Each `.csv` file has 60 rows, one for each patient, as well as 40 columns, one for each day they recorded the number flare-ups they experienced. There are 12 `.csv` files, one for each time patients met with the research team to report their experiences.

### 1.1 Setup

Download the `.zip` of data files from [here](#) and extract the `/data/` folder and all of its contents Google Colab.

### 1.2 Part 1: Reading in our files

Our data includes files that are named `small-xx.csv`, which are *not* relevant to our analysis, and `inflammation-xx.csv`, which *are*. Complete the lines of code to create a list of the full paths for the files that we are interested in analyzing further.

**Hint:** Refer to 03\_in\_out\_modules\_files\_oop slides.

```
[ ]: import os
import csv

path = '/content/data'
all_paths = []

#this for loop will iterate through all the contents of the folder
for i in os.listdir(path):
    #COMPLETE: check if the filename begins with 'inflammation'
    if :
        #COMPLETE: if True, append the full path to the array all_paths

print(all_paths)
```

Complete the lines of code to read in the first file in the list you created. Then, print each row in that file.

```
[ ]: with open(all_paths[0], 'r') as f:
      #COMPLETE: read the .csv file

      #COMPLETE: print each row
```

##2. Summarizing our data

We will now define a function, `patient_summary()`, that will summarize the data across each patient.

More specifically, `patient_summary(file_path, operation)` should: - Take `file_path`, a string of the path to the data, and - `operation`, a string (“mean”, “max”, or “min”) describing what operation to use summarize the number of flare-ups over the course of the 40 days across **each patient**.

**Hint 1:** Refer to 04a\_data\_numpy slides.

**Hint 2:** The shape of the output should be the same size as the number of patients (i.e. 60).

```
[ ]: import numpy as np

def patient_summary(file_path, operation):
    data = np.loadtxt(fname=file_path, delimiter=',')
    ax = 1

    # COMPLETE: fill in the rest of the if, elif, and else statements
    if operation == 'mean':

    elif operation == 'max':

    else:

    return summary_values
```

```
[ ]: # test it out on the data file we read in and make sure the size is what we
      ↪ expect
data_min = patient_summary(all_paths[0], 'min')
print(len(data_min))
```

### 1.3 3. Checking for errors

Sometimes, data entry results in some errors. As an example, if a patient has a mean inflammation of 0, this may indicate that a healthy individual was misgrouped into this dataset, or that there is some other issue requiring further attention.

We will now define a function, `detect_problems()`, that can check for any patients that have a mean inflammation of 0.

`detect_problems(file_path)` should: - Take a `file_path`, a string of the path to the data, and

- Return `True` or `False` depending whether a mean inflammation of 0 was found.

Note that we have created a helper function, `check_zeros(x)`, which returns `True` or `False` if there are values of zero in an array. Use your function, `patient_summary()`, from (2) and our helper function, `check_zeros()`, to create your new `detect_problems()` function.

Below is the helper function, `check_zeros(x)`. **Do not modify this code!** It is not necessary to understand all the code inside this helper function, but when using code others have written, you should have a good idea of: 1. what goes IN the function (`x`, an array of numbers), 2. what comes OUT of the function (`True` or `False`), and 3. what the output means (`True` if 0 found in array, `False` if 0 not found).

```
[ ]: # Run this cell so you can use this helper function

def check_zeros(x):
    """
    Given an array, x, check whether any values in x equal 0.
    Return True if any values found, else returns False.
    """
    # np.where() checks every value in x against the condition (x == 0) and
    ↪ returns a tuple of indices where it was True (i.e. x was 0)
    flag = np.where(x == 0)[0]

    # Checks if there are any objects in flag (i.e. not empty)
    # If not empty, it found at least one zero so flag is True, and vice-versa.
    flag = len(flag) > 0

    return flag
```

```
[ ]: # Define your function `detect_problems` here

def detect_problems(data):
    #COMPLETE: use patient_summary() to get the means and check_zeros() to check
    ↪ for zeros in the means

    return
```

Criteria	Pass Criteria	Fail Criteria
<b>General Criteria</b>		
Code Execution	All code cells execute without errors.	Any code cell produces an error upon execution.
Code Quality	Code is well-organized, concise, and includes necessary comments for clarity.	Code is unorganized, verbose, or lacks necessary comments.
Data Handling	Data files are correctly handled and processed.	Data files are not handled or processed correctly.

Criteria	Pass Criteria	Fail Criteria
Adherence to Instructions	Follows all instructions and requirements as per the assignment.	Misses or incorrectly implements one or more of the assignment requirements.
<b>Specific Criteria</b>		
Setup	Successfully downloads and extracts the data from the provided <code>.zip</code> file.	Fails to download or incorrectly extracts the data files.
Part 1: Reading in our files	Correctly filters and lists file paths for <code>inflammation-xx.csv</code> files and reads in the first file, displaying its content.	Fails to filter out <code>small-xx.csv</code> files, or errors in reading/displaying file contents.
Part 2: Summarizing our data	Correctly defines <code>patient_summary()</code> function. Function processes data as per <code>operation</code> and outputs correctly shaped data (60 entries).	Incomplete or incorrect definition of <code>patient_summary()</code> . Incorrect implementation of <code>operation</code> or wrong output shape.
Part 3: Checking for Errors	Correctly defines <code>detect_problems()</code> function. Function uses <code>patient_summary()</code> and <code>check_zeros()</code> to identify mean inflammation of 0 accurately.	Incorrect definition or implementation of <code>detect_problems()</code> function. Fails to accurately identify mean inflammation of 0.
<b>Overall Assessment</b>	Meets all the general and specific criteria, indicating a strong understanding of the assignment objectives.	Fails to meet one or more of the general or specific criteria, indicating a need for further learning or clarification.

## ##References

### 1.3.1 Data Sources

- Software Carpentry. *Python Novice Inflammation Data*.  
<http://swcarpentry.github.io/python-novice-inflammation/data/python-novice-inflammation-data.zip>