# SCC.022 Making Sense of Data

## Studio Worksheet: Week 3

In this week's studio, we will build on the skills learnt in the previous studios. We will look at two data files using Python with Pandas and Matplotlib. The first task considers cleaning data to improve presentation and facilitate a more informative graphical representation. The second task involves building an automated diagnostic measure using multiple tests and selecting an appropriate threshold.

## Preparation

Please complete the following preparation items before the lab session.

It is assumed that you already have Python installed from the last session. If not, you can install Python from AppsAnywhere, or via the Python website if you have problems with AppsAnywhere.

**Step 1: Create a directory for this session**

We will need a new home directory (or folder) to house all of our files for this studio. In this worksheet, we will assume that you are using the address "H:\SCC022\Wk3". Please keep in mind that your home directory address might be different so, when you see this, you should substitute it for your home folder address.

**Step 2: Download the week3studio.zip file**

This can be downloaded from the Moodle module page. This contains the files needed for the rest of this worksheet. Save it to your home directory and extract the archive so the files are accessible (right click the file and select "Extract all..." or similar). Move the files to your home directory. This should include the files

- requirements.txt
- coronavirus-cases.csv
- dr-test-study.csv

**Step 3: Install Pandas and the supporting libraries**

We will use “pip” to install these:

1. Launch the command prompt:
    a. If you are using Windows: Open Windows Powershell from the start menu.
    b. If you are using Mac: Open Terminal.app
    c. If you are using Ubuntu: Open the Terminal
2. Go to the folder where the worksheet files are saved by typing “cd H:\SCC022\Wk3”
   **NB:** If there are spaces in your filename, you may need to use speech marks in your command. E.g. cd “H:\SCC 022\Wk3”
3. Install the required libraries by running “pip install -r requirements.txt” This will install the Pandas, Scipy and Matplotlib libraries.

## Task 1: Data Cleaning and Visualisation

In this task, we will revisit the Coronavirus data file and try to visualise the results in a meaningful way. This will require filtering rows and working with “for loops.” We will use the coronavirus-cases.csv file for this task.

**Step 1: Run IDLE**

We will be using the interactive window in IDLE to run our program step-by-step as we learn the steps. If you want to run it all automatically later, you can write it into a new file window in IDLE and save and run it like other Python programs. If you are doing so, remember to move all the import lines to the top of the script file.

**Step 2: Import the libraries**

First, we need to import the needed libraries into the program, so type these separate lines into IDLE's interactive window (the one with the >>> prompt):

```
>>> import pandas as pd
>>> import matplotlib as plt
>>> import datetime as dt
>>> import numpy as np
```

Tell the plotting library **matplotlib** to pop-up any graph windows and continue on with the program instead of waiting until the windows has closed to continue. This makes it easier to interactively experiment with.

```
>>> plt.interactive(True)
```

**NB:** you will not see any feedback from this command.

**Step 3: Load the CSV file.**

Store the home directory address as a string variable and load the CSV file into a Pandas DataFrame using the read_csv function. We can then use that variable to access all the values in the file.

```
>>> hdir = "H:\\scc022\\Wk3\\"
>>> cvd = pd.read_csv(hdir + "coronavirus-cases.csv")
```

**NB:** If there are any backslash characters in the path then we need to type them twice to escape them so they are stored in the Python string properly - if you forget to do this you will get an error. Note also that we have used the "+" symbol to join together two strings.

The CSV file should now be loaded. This will be stored in a DataFrame called cvd. Recall that we can check this by typing the variable name into the command window and it will show us some of the first and last rows and columns.

**Step 4: Data Visualisation**

Plot a graph of the daily lab confirmed cases for one area by Specimen date:

1. Convert the dates in the Specimen date column to the correct date format. HINT: we did this in week 1.
2. Get a list of the areas covered in the spreadsheet and chose one of these. HINT: the areas are stored in the "Area name" column. You can obtain a list of the areas using the unique() function. E.g. dataframe['colname'].unique().

3. Create a copy of this dataframe called cvd_subset, containing only the records for the area that you chose in the previous step.
4. Sort the cvd_subset dataframe by date.
5. Plot the daily lab confirmed cases from cvd_subset with the Specimen date on the x-axis. HINT: you can use the plot() function, e.g. dataframe.plot(x = 'xcolname', y = 'ycolname')

**Step 5: Data Smoothing and Visualisation**

What do you notice about the graph you produced in the previous step? Is it clear and easy to interpret? In order to reduce noise in the graph and make it more interpretable, the rates are usually presented as a rolling average of the past 7 days.

Plot a graph of the 7 day average:

1. Add a new column to the cvd_subset dataframe called "7 Day Average"
2. Use a for loop to populate this column with the 7 day averages
3. Plot this column against the specimen date.

Compare this graph with that of the previous step.

## Task 2: Automated Diagnosis

In this task, we will build an automated diagnosis of Diabetic Retinopathy using test data from a clinical study. This is a study of 10,000 patients who are either healthy or have diabetic retinopathy. All of the records are anonymous and each has a record number. Three diagnostic tests have been carried out on each patient and the scores are recorded in the spreadsheet: Fundus Score, OCT Score, and Vision Score. All scores are between 0 and 1 such that lower scores are associated with DR and higher scores are associated with Healthy We will use the dr-test-study.csv file for this task.

**Steps 1-2: Run IDLE and import libraries**

If necessary, carry out steps 1-2 from Task 1.

**Step 3: Load the CSV file.**

Store the home directory address as a string variable and load the CSV file into a Pandas DataFrame using the read_csv function. We can then use that variable to access all the values in the file.

```
>>> hdir = "H:\\scc022\\Wk3\\"
>>> drtests = pd.read_csv(hdir + "dr-test-study.csv")
```

**NB:** If there are any backslash characters in the path then we need to type them twice to escape them so they are stored in the Python string properly - if you forget to do this you will get an error. Note also that we have used the "+" symbol to join together two strings.

The CSV file should now be loaded. This will be stored in a DataFrame called drtests. Recall that we can check this by typing the variable name into the command window and it will show us some of the first and last rows and columns.

**Step 4: Automated Diagnosis Measure**

Calculate the new diagnostic score based on combining the tests and determine the diagnosis using the threshold approach:

1. Create a new column in drtests called "New Score"
2. Create another new column called "New Diag" to store the diagnosis based on the new score
3. Set a variable thr = 0.5. This will act as a threshold for the new diagnosis
4. Populate the New Score column with the mean value of the scores from the study
5. Populate the New Diag column with the new diagnosis. Given the properties of the scores:
    a. the diagnosis will be "DR" if the score is less than or equal to the threshold
    b. the diagnosis will be "Healthy" if the score is greater than the threshold

HINT: For steps 4 and 5, you will need to use a for loop and if clause. You can carry out both these steps within the same for loop.

**Step 5: Evaluation**

Evaluate the new diagnosis, comparing with the clinical diagnosis:

1. Calculate the numbers of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) of this result. HINT: “Positive” refers to “DR” and “negative” refers to “Healthy.” The result is considered “true” if it agrees with the clinician’s diagnosis and “false” if it doesn’t.
2. Calculate the
    a. Accuracy = (TP + TN) / (Number of records)
    b. Sensitivity = TP / (TP + FN)
    c. Specificity = TN / (TN + FP)
3. Consider what this implies for new patients? I.e. If 100 new patients come with test results, how many can we expect to diagnose correctly? If 100 DR/ Healthy patients come with test scores, how many can we expect to diagnose correctly?

**Step 6: Refinement and Comparison**

So far, we have used a threshold of 0.5 but it is possible that a different threshold could be more appropriate for this data. Consider whether an alternative threshold can achieve better results:

1. Repeat step 4 but with a threshold of 0.45 and columns called “New Score 2” and “New Diag 2.”
2. Calculate the accuracy, sensitivity and specificity of this new diagnosis.
3. Compare these results with the previous results achieved with a threshold of 0.5. Are these results better?