DATA 301 - Introduction to Data Analytics
Lab 5: Python Data Analysis

This lab can be done in pairs; however, pairs are not required. If working in a pair, only submit one file on CANVAS with a note of your partner in the comments. If you are enrolled in DATA 501, **bonus questions are mandatory**.

Total marks: 18 marks + 2 bonus marks

In this lab, we will use Python for automating data analysis tasks.

Objectives

1. Use Python to read from a file containing data.
2. Use Biopython library to manipulate bioinformatics data and request data from public NCBI databases.
3. Perform data analysis combining data from local and web sources.
4. Use Python libraries for charting, histograms, k-means clustering, and regression.

Marking and Evaluation (18 marks + 2 Bonus marks)

Biopython

1. By accident, a bioinformatics lab mixed up DNA sequences in a single data file for two organisms: fruit fly (Drosophila melanogaster) and a grape (Vitis vinifera). Your goal is to figure out which sequences belong to each organism while also learning about the NCBI databases, specifically BLAST. Create a Python program that reads the sequences from a text file (one per line), performs a BLAST search if not previously done, stores the results of each search in a separate file, and then performs analytics on the search results to help solve our sequence mix-up problem. Details:

   a. Put a comment at the top of the Python file called `lab5q1` with your name and student number(s).

   b. Download the data file called `input.txt` and place it in your local directory for reading. Each line of the data file contains a DNA sequence. There are 10 sequences in the file.

   c. (1 mark) Read each line of the data file and print the first 20 characters in the sequence and the length of the query sequence to the console.

   ```
   # Sample output for a single sequence:
   Sequence: GGCTGCGGAGACGTTGAAGG Length: 560
   ```

   d. (½ mark) Define a counter called `count` to keep track of how many sequences have been processed (the total count should be `10` once your program is complete).

   e. (2 marks) Using a `try-except` clause, look for a previously created BLAST result file named `dna_lab5_<count>.xml` (for example, `dna_lab5_1.xml`). Start the the search from `count=1`.

   f. (1 mark) If the file exists, open it and print `"Using saved file"`.

g. If the file does not exist or there is an error, perform a BLAST search using the sequence given and print `"Performing online BLAST search"`. Note that you will have to import some modules from Biopython which are not already installed with Anaconda. To install with Anaconda, run this command:

```
conda install -c https://conda.anaconda.org/anaconda biopython
```

Otherwise, see these download/install instructions.

h. Import the following libraries:

```
from Bio.Blast import NCBIWWW
from Bio.Blast import NCBIXML
```

i. (2 marks) Make a BLAST request using `NCBIWWW.qblast("blastn", "nt", seq)` method and store the results in a file with a filename as described above. `seq` is a variable storing your string sequence. Make sure to close the BLAST request and the saved file. Re-open the file for reading.

j. (1 mark) At this point, you will have an open file containing the BLAST output. Use `NCBIXML.parse(blast_infile)` to get the BLAST record.

k. (½ mark) Based on the title of the first BLAST record alignment (`blast_record.alignments[0].title`) determine if a given DNA sequence belongs to fruit fly or grape. **Hint:** You can check if a Python string contains a certain substring the `in` operator (read more here).

l. (½ mark) Save this information in a list called `classification` which stores the value of 1 if the DNA sequence is identified as a grape sequence and a 0 if it is identified as a fruit fly sequence.

m. (1 mark) Define another list called `sizes` to store the sizes of the alignment matched (`blast_record.alignments[0].length`).

n. (½ mark) Make sure to close the input file storing the BLAST result.

o. (1 mark) After all sequences in the DNA `input.txt` file are processed, print `count`, `classification` and `sizes`.

p. (½ mark) Make sure to close the `input.txt` file when done (you may use the `with` clause to do this automatically).

q. (2 marks) Create a bar chart that shows how many of the sequences were classified as a fruit fly and grape sequences. Ensure that they have the same titles, axis labels and legends as those displayed in Fig 1.

r. (2 marks (bonus)) Create a scatterplot which displays the sequence number (as it appears in the input.txt file) with the length of the sequence in 10000s (that is, if the sequence has a length of 23,011,485 it would be plotted as 2301.1485). Ensure that they have the same titles and axis labels as those displayed in Fig 2.

MapReduce

2. Create a Python program that uses Map-Reduce to analyze a data set. You must use map, filter, and reduce functions. Details:

   a. Put a comment at the top of the Python file called **lab5q2** with your name and student number(s).

   b. (½ mark) Create a list called `data` that consists of the numbers from 1 to 10 (inclusive), ie. $1, 2, \ldots, 9, 10$.

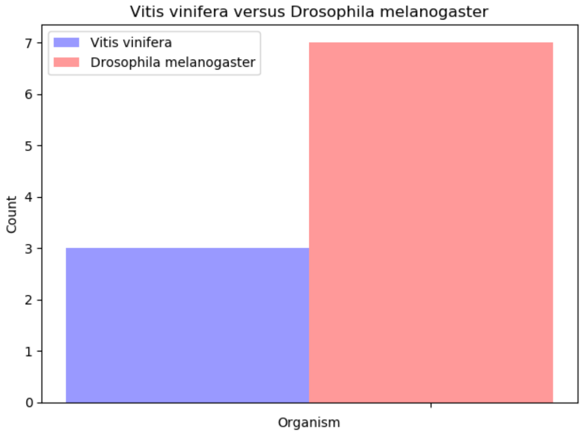

Figure 1: A bar chart containing the frequency of fruit fly (Drosophila melanogaster) and a grape (Vitis vinifera) sequences in our input.txt file

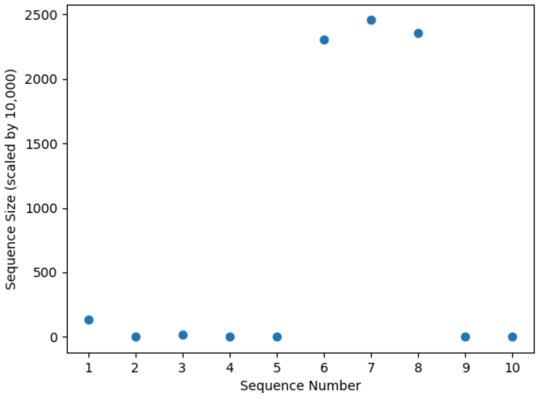

Figure 2: A scatterplot containing the length of the alignment sequences in 10000s.

c. (2 marks) Apply a map function to the data set that will divide the value by 2 if it is even or multiply by 3 if it is odd. Print the result after the map function is applied. (2 marks)

d. (1 mark) Apply a filter function that will keep the value if it is between 5 and 20 inclusive. Print the result after the filter function is applied.

e. (1 mark) Apply a reduce function that will add the two values if the first value is greater than the second otherwise it will multiply them. Print the result after reduce function is applied.

```
# Sample Output
Result after map: [3, 1.0, 9, 2.0, 15, 3.0, 21, 4.0, 27, 5.0]
Result after filter: [9, 15, 5.0]
Result after reduce: 140.0
```

When complete, submit your code to CANVAS:

- lab5q1 - May be a .txt, .py, or .ipynb file.
- lab5q2 - May be a .txt, .py, or .ipynb file.