PROGRAMMING FOR DATA SCIENCE: COURSEWORK

LUCA ZANETTI (LZ2040@BATH.AC.UK).

General instructions

Set: Thursday 28st of March 2024.

Due: The code and supplemental material should be submitted by **Friday 26th of April 2024 at 23.59 on Moodle**.

Estimated time required: The expected time to complete this coursework is about 25 hours in total.

Submission: Submit a single Jupyter notebook on Moodle that is named `*****.ipynb` where `*****` is your Bath username. For instance, my submission would be named `lz2040.ipynb`.

The Jupyter notebook should be clearly formatted and well commented to enable the marker to easily understand which question you are answering and the logic behind the Python commands you invoke. It should also clearly display the results. Open ended questions that ask you, for example, to discuss a certain result or to provide experimental evidence in the data to justify a certain statement should be answered in the same Jupyter notebook. This can be done using a Markdown cell in the notebook (similarly to what I have done in the flipped lectures). You should always make clear which question you are answering.

Value: This coursework accounts for 100% towards the final mark of the Programming for Data Science.

Length: There is no minimum or maximum length for this assignment

Feedback: You will receive feedback within a maximum of three semester weeks following the submission deadline. The feedback will consist of an overall feedback document commenting on the assessment across the cohort.

Late submission of coursework: If there are valid circumstances preventing you from meeting the deadline, your Director of Studies may grant you an extension to the specified submission date, if it is requested before the deadline. Forms to request an extension are available on SAMIS.

- If you submit a piece of work after the submission date, and no extension has been granted, the maximum mark possible will be the pass mark.
- If you submit work more than five working days after the submission date, you will normally receive a mark of 0 (zero), unless you have been granted an extension.

Gen AI use: The use of generative AI is not permitted (type A category).

Academic integrity statement: Academic misconduct is defined by the University as "the use of unfair means in any examination or assessment procedure". This includes (but is not limited to) cheating, collusion, plagiarism, fabrication, or falsification. The University's Quality Assurance Code of Practice, QA53 Examination and Assessment Offences, sets out the consequences of committing an offence and the penalties that might be applied.

Coursework Guidelines

This coursework will assess your ability at manipulating data, analysing it, and extracting information from it. It will also assess your knowledge of programming in Python.

The coursework has been split into three sections, each of which has been further subdivided into a set of guided tasks and open ended components.

While you are not allowed to copy code from the internet, you can use any standard Python library that you find useful (for example, `pandas`, `numpy`, `sklearn`, `networkx`, and `matplotlib`).

Provided Files

As part of this assessment you are provided with the following files:

- This document, which describes the problem and the assessment procedure.
- A zip file which contains the files you may use as part of the assessment.

1. Data Analysis (45 marks total)

This section is devoted to the processing and analysis of a data set containing economic and demographic information about geographic areas associated with local authorities of England and Wales ("Bath and North East Somerset" and "City of Bristol" are examples of such local authorities). The data set has been constructed from data made available by the Office for National Statistics (ONS).

This data set is stored in the file `la_stats.csv`. In particular, each row of `la_stats.csv` corresponds to a different local authority and contains information about the geographic area represented by the corresponding local authority. The following attributes are recorded:

`Code`: a string representing the code of the local authority responsible for the corresponding geographic area;

`Name`: a string representing the name of the corresponding local authority;

`Northing`: a number representing the *Northing* of a reference point in the geographic area;

`Easting`: a number representing the *Easting* of a reference point in the geographic area;

`Population`: the number of people residing in the corresponding geographic area;

`No. pubs`: the number of pubs in the corresponding geographic area;

`Life expectancy (female)`: Life expectancy at birth for people of female sex living in the corresponding geographic area;

`Life expectancy (male)`: Life expectancy at birth for people of male sex living in the corresponding geographic area;

`Median income`: Median total income per household living in the corresponding geographic area.

Northings and Eastings are a two-dimensional coordinate system to reference geographical places in the UK. For example, if a place has coordinates `Northing` 531474 and `Easting` 447160, it means it is 531.474 km north and 447.160 km east from the point that has coordinates Northing 0 Easting 0 (which is somewhere south west of the Isles of Scilly).

Note that `la_stats.csv` contains some missing values.

Task 1.1. Import the file `la_stats.csv` as a pandas dataframe. Find and display the row in the dataframe corresponding to Bath's local authority.

[3 marks]

Task 1.2. Notice that, in the columns corresponding to attributes `Population` and `Median income`, thousands are separated by a comma. Use the pandas method `str.replace` to delete all commas in the values corresponding to attributes `Population` and `Median income`.

[3 marks]

Task 1.3. Convert the values corresponding to attributes `Population`, `No. pubs`, and `Median income` to numeric type using the pandas function `to_numeric`. Use the option `errors='coerce'` to force the conversion to numeric in the presence of missing values.

[3 marks]

Task 1.4. Create numpy arrays named `pop` and `pubs` that contain, respectively, the population of each geographic areas and the number of pubs in each area. Make sure `pop` and `pubs` have the same length by taking care of the missing values in the two corresponding columns of the dataframe.

[3 marks]

Task 1.5. What can you say about the relationship between the population and number of pubs of each geographic area? Justify your answer using one of the scikit-learn models seen in class. You can also use plots to support your answer.

[5 marks]

Task 1.6. Does Bath's corresponding geographic area has more or less pubs than its population suggest? Justify your answer.

[5 marks]

Task 1.7. Do geographic areas with a higher income have more or less pubs? Justify your answer. You can also use plots to support your answer.

[4 marks]

Task 1.8. After you publish your finding regarding the relation between the no. of pubs and income, the X (formerly known as Twitter) user *VeryImportantStatisticsProf89* argues that your findings are flawed because you didn't control for the fact that the number of pubs in a geographic area strongly depends on the population of that area, and the median income is also correlated with the population in a geographic area. In order to reply to *VeryImportantStatisticsProf89*, investigate the relationship between median income and population. Does knowledge about the relationship between median income and population help you better understand the relationship between income and number of pubs in a geographic area? Justify your answer.

[5 marks]

Task 1.9. Construct a numpy matrix X in which each row corresponds to a local authority; the matrix should have three columns that, respectively, contain information about the population, median income, and number of pubs of the geographic area corresponding to each local authority. The matrix should contain only numeric values. Beware, again, that the original data set contains missing values and there is not necessarily a unique way to handle these missing values. Explain and justify how you handle such values.

[4 marks]

Task 1.10. Apply k-means clustering with $k = 3$ on the three-dimensional points corresponding to rows of the matrix X you have constructed in the previous task. Consider the smallest cluster found by k-means. Display the name of the local authorities corresponding to points in this cluster. What characteristics do points in this cluster share?

Create a 3D plot to visualise the results of k-means clustering. In order to do that, use the following two lines of code.

```
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
```

Check the matplotlib documentation to understand what these lines do.

Make your plot intelligible by adding, for example, a label to each axis.

[10 marks]

2. Network Analysis (40 marks total)

This question is focussed on the manipulation and analysis of networks.

The files `graph1.txt` and `graph2.txt` contain the edge lists of two different graphs. One of them represents a network that has been synthetically generated, while the other represents a "real world" network. In this section, you will first try to deduce which graph represents the "real world" network and which one represents the one that is synthetically generated. Finally, you will use tools from network analysis, together with information provided in the file `la_stats.csv`, to answer questions about demographic properties related to geographic regions of England and Wales.

Task 2.1. Load the graphs stored in `graph1.txt` and `graph2.txt` into two networkx Graph objects named `G1` and `G2`, respectively.

[2 marks]

The "real world" network has been generated as follows. Nodes represent geographic areas associated with a local authority in England or Wales. In particular, node i corresponds to the local authority represented by row i of the file `la_stats.csv`. Furthermore, two nodes u and v are connected by an edge if and only if *many* people have moved between the two corresponding geographic areas in a certain period of time. Here, *many* means a number of people larger than a certain number t, which is not known to us.

The synthetic network, instead, represents a graph sampled from the $G(n, p)$ model, i.e., a graph of n nodes where each pair of nodes is connected by an edge with probability p, for some $p \in (0, 1)$.

Task 2.2. Explain which of the two graphs `G1` or `G2` represents the real world network. Provide numerical evidence to support your answer.

[10 marks]

Task 2.3. Find the graph distance between the northernmost and southernmost local authorities belonging to the largest connected component of the real world network.

[8 marks]

Task 2.4. Provide and discuss empirical evidence that proves or disproves the following hypothesis: "people tend to move between areas that are geographically close".

[20 marks]

3. Computational complexity (15 marks total)

Consider the following snippet of Python code.

```
def f(p,q,r):
    if len(r) == 0:
        r.append(("1","10"))
        r.append(("1","11"))
        r = f(0,q-1,r)
        r = f(1,q-1,r)
    elif q > 0:
        r.append((r[p][1],r[p][1] + "0"))
        r.append((r[p][1],r[p][1] + "1"))
        p = len(r) - 2
        r = f(p,q-1,r)
        r = f(p+1,q-1,r)
    return r
def surprise(n):
    return f(0,n,[])
```

Task 3.1. What does the function `surprise(n)` compute? Justify your answer. You can assume n is a positive integer.

[10 marks]

Task 3.2. What is the computational complexity of `surprise(n)` as a function of n? Justify your answer.

[5 marks]