|  |  |  |  |
| --- | --- | --- | --- |
|  | | logo_black.gif | |
|  |  |  |  |
|  | GEOG5995M: Programming for the Social Sciences – Core Skills |  |  |
|  | Assessment 2 – The Black Death |
|  |  |

|  |
| --- |
|  |
| 1. |
| 2. |
| 3. |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

**Intention of the software**

The objectives for this assignment were to:

* Read in some data
* Process it in some way
* Display the results
* Write the results to a file

I chose to use one of the given projects – The Black Death (Evans, 2017). The aim of this project is to calculate the total deaths per week from the Black Death in 1665 for 16 parishes using a given equation and enable the user to change the parameters of the equation to examine the impact on the deaths.

Using the guidance given (Evans, 2017), this is achieved through the development of the Jupyter Notebook that performs the following steps:

1. *Reading in the two datasets*

The first dataset is a raster map of the average number of rats caught per week by rat catchers per 100m x 100m square (parameter ‘r’). The second is a raster map of the average population density per 100m x 100m square for 16 square parishes (parameter ‘p’).

1. *Processes these and displays the maps*

This step involves creating images of both datasets that are appropriately labelled with colour schemes that make sense for the type of data they are displaying.

1. *Calculates the average number of deaths per week and displays this as a map*

The average number of deaths per week (‘d’) is calculated using the equation given by Evans (2017):

This output is displayed as another map.

1. *Displays the three maps together*
2. *Saves the average deaths map as a text file with each line in the file equally a line on the map*
3. *Calculates and displays the total deaths per week*
4. *Allows the user to change the parameter weights for the equation (achieved using a Graphical User Interface (GUI).*

**UML diagram**

**Development process and issues**

Overall this was a relatively challenging piece of work for me to deliver given that I had never coded anything before this module.

The first challenge was ensuring I was confident that I had imported and displayed the raster data appropriately. I utilised the information learnt from the module about raster data (each row is an increase in the y-direction and subsequent values in a row being an increase in the x-direction) and reading such data into a 2D array to inform my code. I cross-checked this by visualising the data in a spreadsheet and by checking the example images shown in Evans (2017).

As the data was in list format I converted it into a numpy array to allow me to use floats for the equation calculation and to enable utilisation of the numpy functions for the calculations.

Once I had imported the data, I then displayed it as two images. INtiially these images had axis on that caused confusion and no indication of what the colours meant. Using Python documentation I discovered how to remove the default axis labels and add appropriate labelling and a colorbar. I tried different colormap schemes to identify the ones I felt conveyed the data well to the viewer. A same colour sequential colormap was used for population density whereas a different sequential colormap that enabled the viewer to clearly demark the different rat catcher patches was used for this data.

I then developed code to run the equation by creating new variables using for loops. One looped through each item within each row (and then subsequent rows) multiplying each r value by 0.8 and the second multiplying each p value by 1.3. The numpy structure ensured that this worked correctly. I then calculated the average death for each 100 x 100 square by using a numpy multiply function. These average deaths per week were then displayed as an image with appropriate labelling and a colorbar. I tested that these calculations were running correctly in numpy by performing test calculations (code not shown).

The next step, which proved to be very tricky, was to create an image with all 3 maps together. On starting to code this I had to go back to the earlier code where the images were created and alter some of this (e.g. save the images, clear the figure) to enable the subsequent code to work correctly. Whilst it was relatively easy to get three images in the same horizontal plane and close enough together it proved impossible to get them to be the same size to enable direct comparison. I tried altering the saving of the 3 individual images (by specifying image size and dpi); used various matplot options such as figure, subplot, gridspec and close(); and searched StackOverflow and Python documentation but I was unable to get them all to be equal size.

The total number of deaths were calculated by summing the values in the numpy array and then displaying them.

The final step was to create something to allow the user to change the parameter weights for the equation. Having looked at various options (such as scrollbars etc), I decided to use PyQT5 to create a GUI to enable the equation parameters to be altered. I watched a series YouTube videos created by Forogh (2018) on PyQT5 GUI programming to learn how to create a GUI. I also utilised another You Tube video to understand how to create the calculations and display the answers in the GUI (Programming Liftoff, 2017). The GUI programming was very much code and then run to see what happened and whether that corresponded to what I wanted. This part (the GUI) was extremely satisfying as it felt like the culmination of all my learning.

**Sources of assistance**

The key sources of assistance were Stackover flow and the official Python documentation such as matplot.

**Initial data vis – to help make sense of the data**

**equation**

Evans, A. 2017. *Geography Programming Courses – The Black Death.* [Online]. [Accessed 5 December 2018]. Available from: <http://www.geog.leeds.ac.uk/courses/computing/study/core-python-phd/assessment2/death.html>

Forogh, P. 2018. *PyQT5 GUI Programming with Python 3.6*. [Online]. [Accessed 29 November 2018]. Available from:

<https://www.youtube.com/channel/UCD6ArU-AYbfIj5sx2L4SZAQ>

Programming Liftoff. 2017. *PyQT5 Calculator Gui Tutorial Part 3.* [Online]. [Accessed 29 November 2018]. Available from:

<https://www.youtube.com/watch?v=urEz0Q7nXaU>