**Abstract**

The objective of this technical review was to compare three data visualisation libraries covered during the module: Matplotlib, Seaborn and Dash. Specifically, this was done through applying them to a chosen dataset and identifying their strengths and weaknesses regarding different types of analyses. It was found that each library was capable of visualising datasets to gain useful insights, however, for the stationary graphs, Seaborn was superior to Matplotlib. Dash is for interactive dashboards and thus has its own uses. All the strengths and weaknesses will be discussed further in the Conclusions and recommendations section of the report.

**Data collection**

The data chosen for this review was regarding nutrition, physical activity, and obesity rates in the US. My personal experience of being overweight prompted me to investigate further into obesity rates in the US specifically since it is described to have two thirds of its population either obese or overweight (Levine, J.A., 2011). The data was retrieved from the CDC website and contained an extensive amount of data on the relative health of adults in the US. There were over 80,000 rows of information and it was updated within the last two years therefore it is relatively current. Furthermore, as the data was collected by the CDC for their own research it is also original. All these features combined culminates in a very solid dataset that is suitable to test these different libraries. The data was then exported into a csv file for the purpose of this analysis. Then using PANDAS it was imported into Jupyter Notebook as a data frame to begin the analysis.

Before the raw data could be analysed, it first needed to be cleaned to remove any error or blank values. This was done identifying the columns that could be necessary for the analysis and discarding the rest. This step in the process was especially useful as many columns just contained blank cells thus removing them entirely streamlined the data cleaning process. The columns were removed by creating a subset of the original data frame only containing the ones that were deemed suitable for the analysis. Subsequently, after the unnecessary columns were removed the rest of the blank cells needed to be identified and removed. Using the .isna function from the PANDAS library the remaining blank cells were able to be found. A new clean dataset was then created by using the .dropna function which removed all rows containing blank cells. As the dataset was so comprehensive and there were very few blank cells remaining, relative to the whole dataset, this was a suitable method to clean the data.

**Data analysis**

The main method employed was Data frames as they allowed the data to be easily filtered into what was required at any given moment to then be visualised using the chosen libraries being reviewed in this study. To begin the analysis the clean data was split into the three categories that were investigated during data collection: Physical Activity, Obesity/ Weight scaling and Nutrition. This was done through the use of the st.contains function which filters the data based on a given condition. The data was then further filtered into the separate questions asked in each category. Finally, using the .groupby function more useful insights were able to be gained by retrieving average data values based on such things as the year or the states.

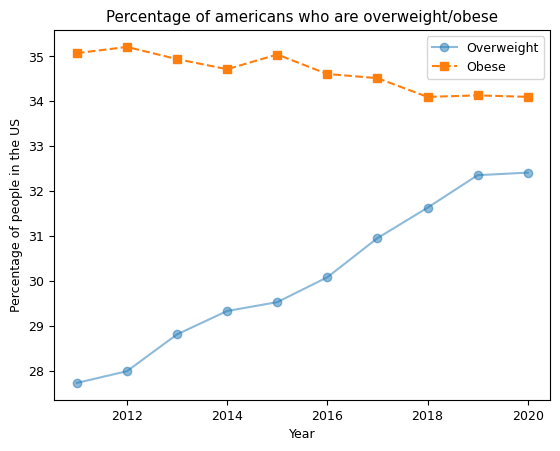
Next was testing the three data visualisation libraries and identifying which ones were most suitable for different types of data analyses. The first one used was matplotlib and this one provides a range of different visuals and features. Initially, a simple line graph was plotted using the .plot function to see how the obesity/overweight rates changed over the years.

Figure 1: Line graph

Matplotlib enables users to customise your lines to make them even more distinguishable, such as by adding markers, changing the type of line, changing the opacity of the line, changing the colour of the line and so on. Each of these customisations have their own catalogues of choices and can be called upon by adding them in to the .plot() function. Additionally, titles and legends can also be added to give your graphs more relevance and better understand what information is being shown.

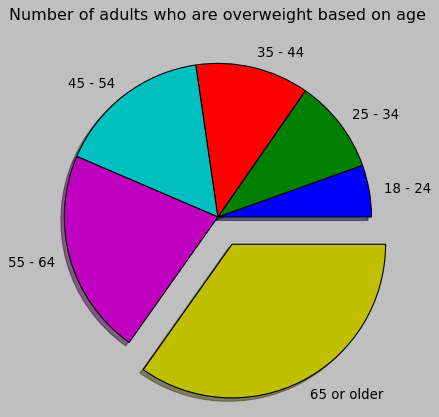


Figure 2: Pie chart

Next, we have pie charts, which are another feature of Matplotlib. This is very useful when a population must be broken down into different categories as they can be observed clearly through visuals. Additionally, the shadow feature makes the visual more 3 dimensional as well as the explode feature which allows different sections of the pie chart to be emphasised.

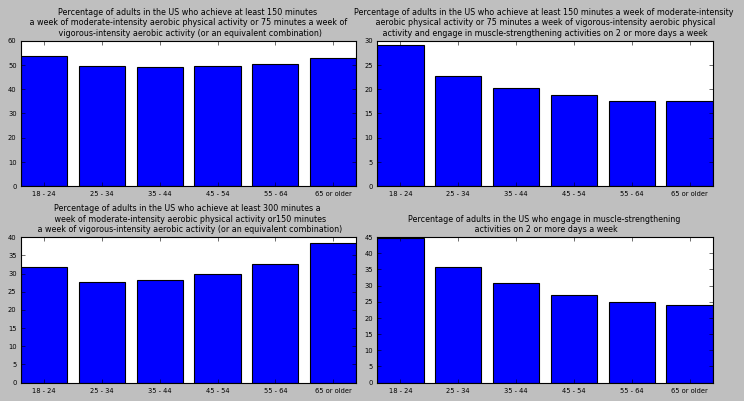
****

Figure 3: Simple dashboard of bar charts

Lastly, we have a simple dashboard showing 4 different bar charts. This is the .subplots function of Matplotlib and it allows multiple charts to be displayed at once, therefore allowing easier comparison of data.

The next library that was explored was Seaborn. One of the features of seaborn is to act as an add on to Matplotlib. It does this by changing the formatting of charts such as the .set\_theme function where the style of the table can be changed or the font or colour scheme. The one used in this study was the (style) and this was switched to “darkgrid” which made the graphs a bit more attractive and better displayed the lines passing through each point.

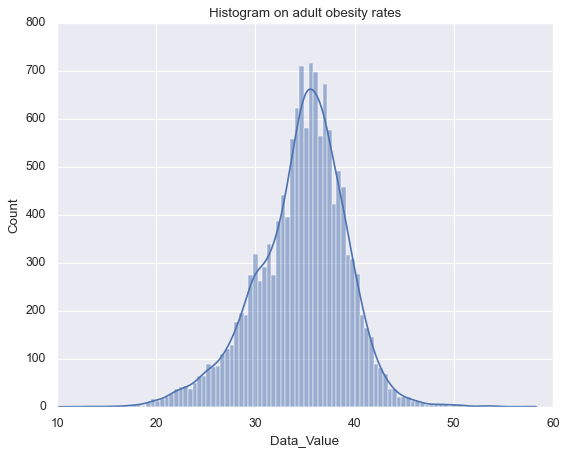


Figure 4: Histogram

Here is a histogram in the Seaborn package with the darkgrid feature present, additionally using the kde add-on a trend line can be seen to give a better understanding of the visual.

Finally, the last library reviewed was the dash library which allows interactive dashboards to be created. This is particularly useful as large amounts of data can be inputted into these dashboards and users can look through and identify the information that is important to their needs.

**Conclusions and recommendations**

The results of this study indicate that each library is very useful in visualising data, however each has strengths and weaknesses that may make them more useful for certain types of data analysis. Specifically, Matplotlib cannot analyse data from data frames and thus can only retrieve information from arrays. However, this can be combatted by using Matplotlib in conjunction with the Numpy library as this can convert columns in data frames into arrays using the np.array() function. Additionally, Matplotlib plot is relatively low-level in comparison to the libraries being explored to this study as it does require more lines of code to achieve similar results.

Using Seaborn, data can be extracted and visualised straight from data frames. Additionally, the customising features in Seaborn exceed that of Matplotlib allowing users to create attractive and informative visualisations with less lines of code. It is recommended that Seaborn is a better library to use as it possesses all the same strengths as Matplotlib with none of the same weaknesses. It also allows the user to produce more varied types of graphs such as joint plots and is easier to customise.

The dash library is an excellent tool for the reasons stated in the analysis. Therefore, it should be used when compiling large amounts of data for users to view. Due to its interactivity, an entire dataset could be visualised in one dashboard allowing onlookers to understand the whole thing without viewing any figures, which is particularly useful for stakeholders who may not have the time to go through the data thoroughly. However, it does require more code to implement and can be more difficult to understand. On the hand, if used in conjunction with some of the geographical libraries present in python, very attractive heat maps can be created, and users can zoom in on the information they require.

**Bibliography**

|  |
| --- |
|  |

Levine, J.A., 2011. Poverty and obesity in the US. *Diabetes*, *60*(11), pp.2667-2668.

**Appendix**

Original dataset

<https://chronicdata.cdc.gov/Nutrition-Physical-Activity-and-Obesity/Nutrition-Physical-Activity-and-Obesity-Behavioral/hn4x-zwk7/data>