Tutorial 02 Proportion

This tutorial consists of two data explorations. The first uses data that is relatively similar to the data in the lecture notes and examples. The second is more unstructured and open ended.

We will be using **python notebooks** (.ipynb files) to do the explorations. You can run these in a number of ways including Jupyter, if you are familiar with that software. However, in the labs I recommend that you use Colab from Google. **For more details see Tutorials.pdf or watch the "How to get started with …" videos from Moodle.**

Naming: For each exploration you should create a notebook and save it when you have finished. You should name the two notebooks Tut02-A.ipynb and Tut02-B.ipynb.

Structure: Every numbered item in the Exploration should have its own code section and a markdown section underneath where you discuss your findings. There should also be a code section at the top with the imports.

Exploration A

The company who supplied the Products data in the lecture notes also want an investigation into their website. The data is in a similar format to the Products, showing the number of page hits per day, rather than the number of product sales, but there are many more web pages than products.

As with the lecture examples the company wants the pages classified into High Volume (used by most visitors to the site), Medium Volume (used by some visitors to the site) and Low Volume (only used occasionally). You will need to decide on what the classifications are.

1. Read in the data from the file **Pages/DailyHits.csv** (https://tinyurl.com/ChrisCoDV/Pages/DailyHits.csv) and do some basis statistical analysis using the .head() and .describe() functions. Then write a quick summary in markdown (e.g. what is the time period under investigation, how many pages are there, which page has the most hits, what is the smallest number of hits for any page, …).

2. Create a bar chart showing the total number of hits for all pages. Most pages only have relatively few hits – how many pages have a significant number? Note chart will almost certainly be unreadable, so you will need to do this by sorting the data first and then printing out the first 15 entries or so using something like
```
data = data.reindex(data.sum().sort_values(ascending=False).index, axis=1)
print(data.sum().head(15))
```

3. Create a bar chart showing the total number of hits for high volume pages.

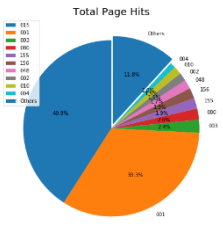
4. Create a bar chart showing the total number of hits for medium volume pages.

5. Create a bar chart showing the total number of hits for low volume pages. [**Hint**: this is a bit harder than the lecture example as there are too many low volume pages for you to list them all. Instead, select the columns by value as described last week.]

6. Now by modifying the code in 06BarChartAutomatic.py, create a code block in your Notebook that automatically classifies pages into high, medium and low volume pages. You will need to decide the boundaries for each type. Your code block should produce 3 bar charts. [**Hint**: if you modified the data in item 5 above by dropping columns, you will need to read it in again in this block.]

7. Finally create a pie chart summarising your findings. The pie sections should be sorted in order of decreasing page hits (like the ones in the lecture are sorted in order of decreasing sales) and all the low volume pages should be grouped together into one pie segment.

   Your final plot should look something like the one on the right.



Exploration B

The file world_population.csv (available at https://tinyurl.com/ChrisCoDV/world_population.csv) contains data about population densities from 1960 to 2016. Note this is **population density** (i.e. the number of people per square kilometer) and not **absolute population**. So some of the smallest countries have the highest densities.

Some of the countries in the data have missing for some years (possibly because the country didn't exist that year or just because the data is missing). For that reason, and because it doesn't really make sense to add up densities, in the following exploration we will calculate mean (average) population density for each country. Therefore you should use .mean() rather than .sum() **throughout** this exploration.

1. The data in this exploration also involves more work to get it into shape so read in the data and wrangle it as follows:

   First, each column contains the data from a particular year, whilst each row contains the data for a country. We would like it the other way around to match the previous examples, so transpose it. [**Hint:** see last week to find out how to do this.]

   Next drop some of the initial rows which contain any descriptive data – the rows you need to drop are 'Country Code', 'Indicator Name' and 'Indicator Code'. [**Hint**: see last week for the .drop() function.]

   Finally write a quick summary of the data in markdown.

2. Create a bar chart showing the mean population density for all countries (although this will produce too much information to make sense of).

3. Using the same techniques as item 6 in exploration A, create a code block in your Notebook that automatically classifies countries into extreme, high, moderate and low density. You will need to decide the boundary between moderate and low densities. There is no right answer for this but extreme density might be more than 10,000 people per square kilometer and high density more than 1,000. Your code block should produce 4 bar charts.

4. The data is still rather difficult to deal with, so pick five countries of your choice and create a bar chart showing the mean population density for these countries.

5. Now create a bar chart showing the mean population density for your selected countries for the years 1961-1970, inclusive. [**Hint**: see last week to find out how to select certain rows of the dataset.] You may find the following code useful (copy & paste).

   years = ['1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968', '1969', '1970']

6. Finally create a bar chart showing the mean population density for your selected countries for the years 2001-2010, inclusive. Again you may find the following code useful.

   years = ['2001', '2002', '2003', '2004', '2005', '2006', '2007', '2008', '2009', '2010']

   You should make sure that the three charts used for items 4, 5 & 6 have the same scale on the y-axis. [**Hint**: use 'plt.ylim(ymax=*value*)' to do this, where you choose an appropriate *value*.]

   Your final chart should look something like the one on the right, depending on which countries you chose.

   Comment on your findings.