Tutorial 01 Introduction

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 separate notebook. Name them Tut01-A.ipynb and Tut01-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 of the Notebook 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.

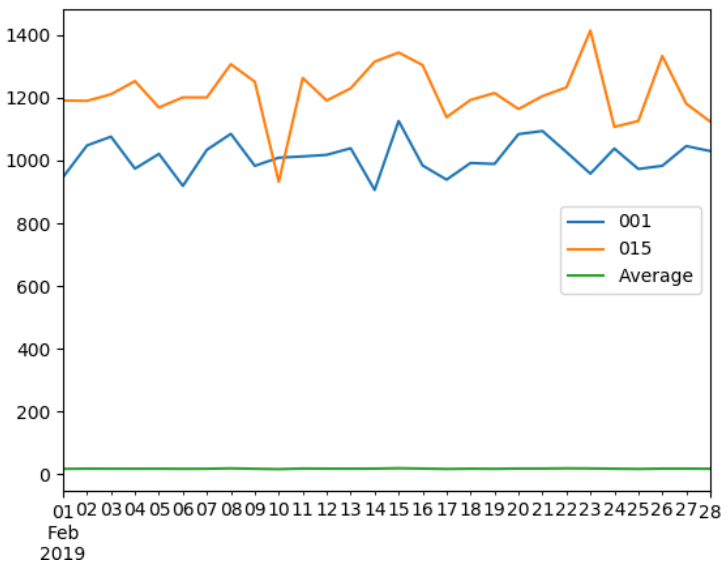
1. Read in the data from the file **Pages/DailyHits.csv** (https://tinyurl.com/ChrisCoDV/Pages/DailyHits.csv) and do some basic statistical analysis using the .head(), .tail() and .describe() functions. Then write a quick summary in markdown (e.g. what is the time period under investigation, how many pages are there). Also describe the range of values for page 001 – what is the min, max and average number of hits each day.

2. Now sort the columns according to the **average** number of hits per day, largest first. Which two pages have the highest average. [**Hint**: use .mean() to get the average.]

3. **Filter** the data for these two pages by name and create a line plot showing the page hits per day. Almost every day one of the pages has more hits than the other … except for one day. Which month did that happen?

4. Next add a column which shows the average page hits for the entire site and then **select** the columns containing two most popular pages and the site average. Use the resulting dataframe to draw a line plot. [**Hint**: this is slightly different to the lecture example – you need to calculate the average *before* you select the columns.]

5. Finally restrict the rows to just show February and create a line plot of the result. If you have done it all correctly the final plot should look something like the one on the right. Notice that the site average is tiny (less than 20 hits per page per day as there are so many pages that get almost no traffic).



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.

   Next drop the three the initial rows which contain any descriptive data – the rows you need to drop are 'Country Code', 'Indicator Name' and 'Indicator Code'. [**Hint**: see example 12 for an example of how to drop data but note that because these are rows you will need to use axis=0 rather than axis=1.]

   How many countries are listed (should be the same as the number of columns).

2. Next sort the countries by their average population density, largest first. Which are the top three countries in terms of average population density? [**Hint**: use .mean() to do the sorting and then, once you have sorted them, use print(data.mean()) to get a list of countries in order of density.]

3. Finally select the top three countries, using column selection by value, and draw a line plot of their population density. If you have done it all correctly the final plot should look something like the one below.