

2Data Mining and Decision Systems

Workshop 2

Getting to grips with Pandas

DataFrames and Series

Aims of the workshop

The aim of this session is to further introduce you to the Pandas library, by looking at some core functionality of DataFrame and Series objects. These are very powerful data wrappers, which make analysis of data easier to perform and, most importantly, understand.

Please see 'Useful Information' below on how to lookup certain Python functionality. The concept behind this workshop is about discovery, and experimentation surrounding topics covered so far.

Feel free to discuss the work with peers, or with any member of the teaching staff.

Useful Information

Throughout this workshop you may find the following useful.

Python Documentation

<https://docs.python.org/3.7/>

This allows you to lookup core language features of Python 3.7 as well as tangential information about the Python Language.

Jupyter Notebook Basics

<https://jupyter.org/try>

The above may be useful as an introduction to the Jupyter Notebook platform, and how to use it. Go to “Try Classic Notebook”. Once the Notebook opens, there should be a link on the page called “Notebook basics”.

Pandas Documentation

<https://pandas.pydata.org/pandas-docs/stable/index.html>

DataFrames:

<https://pandas.pydata.org/pandas-docs/stable/reference/frame.html>

Series:

<https://pandas.pydata.org/pandas-docs/stable/reference/series.html>

Similarly to the above, the pandas library itself is well-documented outlining function signatures, alongside example use-cases.

Functions of note: `read_csv`, `[]`, `loc`, `iloc`, `min`, `max`, `mean`, `median`, `copy`, `random.randint`, `insert`, `drop`.

Useful indexing introduction:

<https://www.geeksforgeeks.org/indexing-and-selecting-data-with-pandas/>

Reminder

We encourage you to discuss the contents of the workshop with the delivery team, and any findings you gather from the session.

Workshops are not isolated, if you have questions from previous weeks, or lecture content, please come and talk to us.

The contents of this workshop are not intended to be 100% complete within the 2 hours; as such it's expected that some of this work be completed outside of the session. Exercises herein represent an example of what to do; feel free to expand upon this.

Running A Jupyter/IPython Notebook

A jupyter notebook is a server which runs on the local machine. It will use whichever directory it is invoked within as the root folder. It is then able to see all sub-folders within that. This is currently installed on the lab machines; however, if you are wishing to install this at home, you may need to follow pip instructions for installing jupyter.

Jupyter Notebook can be launched from the command line by invoking:

```
jupyter notebook
```

This will start a server, typically on <http://localhost:8888/tree>, and should automatically open a new tab in the browser.

Jupyter notebook server uses a token, generated at launch, to authenticate access. By default, the notebook server is accessible to anybody with network access to your machine over the port, with the correct token.

If you are asked for a password or token, you can always invoke:

```
jupyter notebook list
```

This will provide you a list of currently active Notebook servers (you can run multiple, on different ports, for different folders), along with their token, all within a single URL.


NOTE: You may wish to `cd` into an appropriate starting folder, such as a USB Stick, or a folder where you can store all of your Data Mining Workshop documents.

Axes in Pandas

When dealing with multi-dimensional data, we often use the phrase 'axis' or 'axes', to represent which dimension we are working with.

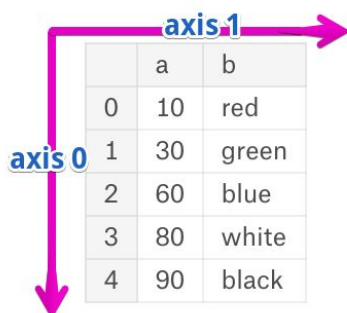
Many operations in Pandas, such as removing records, relies on an axis to work on. Typically, the axes are labelled sequentially. When dealing with 1D data (A Series), axis 0 is the only dimension. When dealing with 2D Data (DataFrame), Axis = 0 represents the rows, and Axis = 1 represents the columns, each of our attributes. As Pandas has a universal function, drop, for removing elements, it's important to consider whether you want to remove rows (records) or columns (attributes) and which axis this would be using.

```
[1]: import pandas as pd
      pd.Series(['red', 'green', 'blue', 'white', 'black'])
```



```
0    red
1   green
2    blue
3   white
4   black
dtype: object
```

```
[1]: import pandas as pd
      srs_a = pd.Series([10,30,60,80,90])
      srs_b = pd.Series(['red', 'green', 'blue', 'white', 'black'])
      df = pd.DataFrame({'a': srs_a, 'b': srs_b})
      df
```



	a	b
0	10	red
1	30	green
2	60	blue
3	80	white
4	90	black

For each of the exercises below, please use a separate code cell. This allows you to compute just the given exercise and execute it, checking the output. Subsequent cells all share the same variable space - variables you assign in cell 1 will be available afterwards in other cells.

Exercise 1: Read the Titanic Dataset into a DataFrame. Get the 'ID' attribute values only, as a Pandas Series, and store them in a variable called *id_values*.

Exercise 2: Using this Pandas Series, calculate the mean, and median. Print this in the following way: *"ID mean is 123.012; median value is: 5"* such that it appears beneath the Jupyter Notebook cell; replacing the values with those calculated programmatically.

Exercise 3: Find the minimum and maximum ID values. Represent this data attribute as a range description, as seen from the combinatorics lecture. Remember the difference between [10, 1000] and (10, 1000). What is the data type? And how many possible values exist? How many degrees of precision are there? Check your answer with your peers or a demonstrator.

Exercise 4: Create a copy of the original dataframe, storing this in a new variable. This new variable, *my_newly_copied_dataframe*, will be our working-copy. Any changes we make will only change this. The original dataframe should be left untouched, so that we don't accidentally modify our source data irreparably.

Exercise 5: Programmatically generate a random integer between the minimum and maximum ID found in Ex 3. Remove this data point from the copied dataframe. (hint: *iloc*)

Exercise 6: Programmatically generate a random integer, *r*, between 0 and *N*-1, where *N* is the number of records you have available in your copied dataframe. Locate this record (hint: *loc*), and display it.

Exercise 7: Consider the Titanic Attributes. Create a new person, making up values for attributes which are reasonable given the data taxonomy. Insert this into your copied dataframe. You will need to ensure that a unique ID is given, that doesn't conflict with any existing entries.

The Extended Exercises are optional, and are offered as an advanced supplement for those who have completed the existing work and wish to expand on their knowledge and challenge themselves further.

Extended Exercise 1: Replace the statistical functions in today's exercises with custom implementations. Create functions which will iterate through a Series object and return the statistic in question. E.g mean.