Course Summary

Block 1

Day 1: Getting started and Basics of Python

Goals

1) Navigate in the shell application: terminal/cmd
2) Python installation (through anaconda installed)
3) Running Python (and Ipython) console in terminal/cmd
4) Creating (and running) Python scripts
5) Running Jupyter lab
6) Basics of Python: Python data types: int, float, string (a little bit of indexing)

Content

Shell: learn shell commands like `ls, cd, pwd, mkdir, cp, mv, rm, rm -r, touch, echo, cat` (and the associated commands on windows cmd)
Python: download and install anaconda, run Python console in terminal/cmd (i,e., open, close, etc.)
IPython: open console, getting help, advantage compared to python console.
Jupyter notebook: open it, creat a notebook, write code in cells, write markdown in cells, execute and create cells, delete cells, restart the kernel, close the notebook, stop the notebook server.

Resources

- https://www.codecademy.com/learn/learn-the-command-line
- https://www.python.org/about/
- https://ipython.readthedocs.io/en/stable/
- https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook
- https://jupyter-notebook.readthedocs.io/en/stable/
- https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet

Exercise

- https://www.practicepython.org/exercise/2014/01/29/01-character-input.html

Day 2: (Cont.) Basics of python

Goals

1) Recap of Day 1: Console, Script, Jupyter notebook, (some of the) Python data types
2) Contibuation with python variables and data types
3) Introduction Collections/sequences (Tuples, Lists, Dictionaries). Talk about differences between each data type (simmarize in a table)
4) Indexing, slicing
5) Loops
6) Functions
7) Booleans and flow control (if-else statements)

Content

- python data types: `boolean, int float, string, list, dict, tuple`
- operators: `+, -, /, *, **, %`
- methods associated with data types, e.g., essential string methods, essential list methods.
- if-elif-else statements, conditional variable assignment, `in` operator
- list and tuple indexing, dict indexing, mutability vs. immutability,
- `while` loops, `for` loops, concept of iterator and generator, using `enumerate`, and `zip`, combining them.
- iterating through dictionaries
- list comprehension, conditional list comprehension

Resources

- https://github.com/cne-tum/msne-datascience-2018/blob/master/notebooks/block1/block1_session2_basic_python.ipynb
- https://github.com/cne-tum/msne-datascience-2018/blob/master/notebooks/block1/block1_session2_programming_concepts.ipynb
- https://www.codecademy.com/learn/learn-python
- https://s3.amazonaws.com/assets.datacamp.com/blog_assets/PythonForDataScience.pdf

Exercises

- https://www.practicepython.org/exercise/2014/02/26/04-divisors.html

- https://www.practicepython.org/exercise/2014/03/19/07-list-comprehensions.html
- https://www.practicepython.org/exercise/2017/01/24/33-birthday-dictionaries.html
- https://www.datacamp.com/courses/intro-to-python-for-data-science

---

Day 2 & 3: Numpy and Matplotlib

Goals

1) Recap of Day 1 and Day 2
2) List comprehension (combines loop, slicing, and if statement)
3) What is a Numpy array, and why to use them? (motivation)
4) Importing and Generating Data
5) Getting insight about the Data (type, dimension, size, etc.)
6) Manipulating the array (arithmetic operations, transpose, etc.)
7) Slicing and Masking
8) Combining arrays
9) Saving data
10) Create different types of figures (e.g., line, scatter, etc.)
11) Customize the figure (e.g., color, style, size, etc.)
12) Put several figures together (i.e., subplots)
13) Save the figure

Content

- Import data:
  - `np.load()`
  - `np.loadtxt()`
  - `np.genfromtxt()`

- Creating Numpy arrays
  - `np.zeros()`
  - `np.ones()`
  - `np.random.random()`
  - `np.empty()`
  - `np.full()`
  - `np.full_like()`
  - `np.eye()`
  - `np.identity()`

- Data Inspection (assuming we have numpy a array object called `data`)

  - `data.dtype`
  - `data.ndim`
  - `data.shape`
  - `data.size`
  - `data.strides`
  - `data.min()`
  - `data.max()`
  - `data.mean()`
  - `data.std()`
  - `data.cumsum()`

- Data Transformation (assuming we have a numpy array object called `data`)
  - `data.T`
  - `data.reshape()`
  - `data.resize()`
  - `np.expand_dims()`
  - `np.ravel()`
  - `np.add()` , `np.subtract()` , `np.multiply()` , `np.divide()` , `np.remainder()`
  - `np..exp()` , `np.log()`
  - Masking using list (or array) of True and False values

- Combining multiple arrays ans splitting an array
  - `np.concatenate()`
  - `np.append()`
  - `np.hstack()`
  - `np.vstack()`
  - `np.hsplit()`
  - `np.vsplit()`

- Saving numpy arrays
  - save(): saves data in .npy format
  - savez(): Save several arrays into an uncompressed .npz archive
  - savez_compressed():
  - savetxt():
- Create a simple line plot using `plt.plot()`
- Create a simple scatter plot using `plt.scatter()`
- Modify the data representation (line color, width, point size, markers, and style)
- Modify the axes (xlim, ylim, ticks and ticklabels, etc.)
- Save a (high quality) figure using `plt.savefig()`

Resources

- https://www.datacamp.com/community/tutorials/python-numpy-tutorial
- Cheat sheet
- https://www.datacamp.com/courses/introduction-to-data-visualization-with-python (first two blocks)
- https://matplotlib.org/gallery.html
- https://github.com/matplotlib/AnatomyOfMatplotlib
- https://github.com/jbmouret/matplotlib_for_papers
- https://jakevdp.github.io/blog/2013/07/10/XKCD-plots-in-matplotlib/
- https://jakevdp.github.io/blog/2012/10/07/xkcd-style-plots-in-matplotlib/
- Cheat sheet

Day 3: Pandas and Seaborn

Goals

1. Understand the positioning of Pandas in the data science pipeline and the convenience supplied by /labeled data structures/, /automatic missing data handling/, /column-oriented layouts/, /embodiment of relational algebra/ and /rich C-level implementation of a functional map/reduce like API/.
   1) Understand row and column-oriented access patterns, know the customary layout of observations x attributes for data science.
   2) Understand `pd.DataFrames` data structures and its properties as compared to known basic Python data structures.
   3) Be able to read textual tabular data from the file system and remote URLs.
   4) Practice access to data and metadata.
   5) Use grouping operations and allowable reductions on them ('split-apply-combine')
   6) Express composable map operations in Pandas with anonymous functions
   7) Appreciate the advantages of splitting (/normalizing/) observational statements to prevent duplication and how table joins allow to operate practically in this setting.

Content

- Series and data frames created from dictionaries and (nested) lists.
- `read_csv` with different urls, separators.
- metadata accessors (`index` , `columns` , `info` , `dtypes` , `shape` , `len`) and data (`.values` , . `iloc` , `loc` , `head` , `tail`). Multiple uses of `[]` : element access in series, column access in data frames, boolean indexing with conforming arrays.
- grouping operations `groupby` , `value_counts` , `(n)unique` .
- chainable mapping with `apply` and `lambda` functions. Compare with for-loop iteration.
- caveats in assignment, `df.attribute = something` will not create a new column

- reduction operations `min` , `max` , `mean` , `std` , `median` , `count` and descriptive statistics with `describe`
- sorting with `sort_values`
- `merge` and the join key
- understand the relationship of seaborn with matplotlib and the added value of the former.

Resources

- Pandas cheatsheet
- Pandas exercises
- Pandas documentation
- Seaborn Gallery
- Seaborn introduction
- Jake van der Plas' intro to Seaborn

Exercises

Pandas

Looking at the Pandas cheatsheet and recurring to the interactive documentation or the API docs linked above, try to solve the following three notebooks. Then, check your answers against the provided solutions, try to understand and come up with precise questions for any remaining doubt.

- https://github.com/guipsamora/pandas_exercises/tree/master/02_Filtering_%26_Sorting/Fictional%20Army
- https://github.com/guipsamora/pandas_exercises/tree/master/03_Grouping/Alcohol_Consumption
- https://github.com/guipsamora/pandas_exercises/tree/master/05_Merge/Fictitous%20Names

If you want more challenges, go for the following:

- https://github.com/guipsamora/pandas_exercises/tree/master/02_Filtering_%26_Sorting/Euro12
- https://github.com/guipsamora/pandas_exercises/tree/master/03_Grouping/Regiment
- https://github.com/guipsamora/pandas_exercises/tree/master/05_Merge/Housing%20Market

Seaborn

Choose one graphical display from the Seaborn gallery (see Resources above), choose one of the datasets now known to you from the exercises (or any other of your interest, check out kaggle.com or data.world) and make a Seaborn plot with at least four dimensions of variation reflecting a mixture of continuous and discrete attributes.

For your upcoming final project, make at least one of the final presentation plots using Seaborn.

Day 4: Debugging, OOP, git (and GitHub)

Goals

1) Recap of Day 1, 2, and 3
2) Learning how to debug (or explore) the code and the tools available for such purpose
3) An introduction to Object-oriented programming. What are classes and objects
4) An introduction to git and GitHub

- Start by a recap from previous days, and a warm up exercise which integrates and refreshes as many taught concepts as possible
- A brief explanation (and discussion) about utility of a debugger
- What are different tools available for debugging Python code, and how to use them
- And introduction to OOP (classes, objects, and inheritance)
- Introduction to GitHub. Everyone is to have a GitHub account during this session. Create Repos, Clone, Make changes, Push, Pull, create branch(es), Push, Merge, Pull, etc.
- We will finish by a fun project (something that involves a group discussion and planning, have most of the concepts within it and can be visualized) and we will practice collaboration, and peer programming for this project and at the end we have something nice which reminds us of this Workshop and will publish it on out GitHub accounts