Course Summary

Block 1

Session 1: Getting started

Goals

1) navigate in the shell application: cmd or terminal
2) have python and anaconda installed, all required packages installed
3) now how to open and close, Python console, IPython console, jupyter notebook

Content

Shell: learn shell commands like `ls, cd, pwd, mkdir, cp, mv, rm, rm -r, touch, echo, cat` (and the associated commans on windows cmd) Python: download and installing anaconda, open python console, close it, install packages using anaconda. 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

---

Session 2: Basics of python

Goals

1) Know different ways to write and execute python code: console, script, jupyter notebook
2) Be familiar with python variables and data types, know how to access the documentation, "everything is an object!"
3) Be familiar with booleans, dictionaries, if-else statements, loops, indexing, slicing, mutability, generators, iterators

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_bas
- https://github.com/cne-tum/msne-datascience-2018/blob/master/notebooks/block1/block1_session2_pro
- 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

---

Session 3: Git and GitHub

Goals

1) Understanding the concept of version control and its importance
2) Differentiation between Git and GitHub
3) Familiarize with the Git (version) control terminology
4) Creating repositories and practicing common workflow

Content

- Git is a version control system that:
  - Keeps track of changes to files (**who** made **what** changes, **when** and **why**).
  - Notice conflicts between changes made by different people.
  - Synchronize files between different computers.
- GitHub is a web-based hosting service for version control using Git, with some extra features (e.g, access control, etc.)
- Common practices (workflow) in git
  - clone (i.e, donwload) a (remote) repository from GitHub
  - (Make branches and) make changes on your local repository (copy of the remote repo)
  - commit (i.e., save) the changes
  - push the changes to the remote repo (update the remote repo wrt the local repo)
  - pull changes from the remote repo (update the local repo wrt the remote repo)
- Create a branch for a change, choose descriptive names for the branch, and keep the number of branches low. Once new changes are stable, merge them with the master branch and remove the other branch.

Resources

- A nice interactive course offered by DataCamp.com (link)
- A collection of resources to learn git (link)

Exercises

1. you can complete the DataCamp course mentioned above.
2. (if you haven;t done it already,) please crete a repository on github and clone it.
3. withing this repo, on your local machine, open a juputer notebook and solve the exercises for Numpy and Matplotlib (can be found in the following sections).

Session 4: numpy

Goals

1) What is a Numpy array, and why to use them? (motivation)
2) Importing and Generating Data
3) Getting insight about the Data (type, dimension, size, etc.)
4) Manipulating the array (arithmetic operations, transpose, etc.)
5) Slicing and Masking
6) Combining arrays
7) Saving data

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():

Resources

- https://www.datacamp.com/community/tutorials/python-numpy-tutorial
- Cheat sheet

Exercises

Please refer to the notebook (called "Numpy_exercises.ipynb") provided in the course repository.

Session 5: matplotlib

Goals

- Create different types of figures
- Customize the figure
- Put several figures together
- Save the figure

Content

- 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/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

Exercises

Re-create the figure below.

```
<img src ="notebooks/block1/img/ExercisePlot.png" height="600"/>
```

Block 2

Session 6: 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/.
2. Understand row and column-oriented access patterns, know the customary layout of observations x attributes for data science.
3. Understand `pd.Series` and `pd.DataFrames` data structures and their properties as compared to known basic Python data structures.
4. Be able to read textual tabular data from the filesystem and remote urls.
5. Practice access to data and metadata.

6. Use grouping operations and allowable reductions on them ('split-apply-combine')
7. Express composable map operations in Pandas with anonymous functions
8. Appreciate the advantages of splitting (/normalizing/) observational statements to prevent duplication and how table joins allow to operate practically in this setting.
9. Enumerate available dimensions of graphical variation to accomodate categorical and continuous data

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%20
- 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/maste
- https://github.com/guipsamora/pandas_exercises/tree/master/03_Grouping/Regiment
- https://github.com/guipsamora/pandas_exercises/tree/master/05_Merge/Housing%20Market

For even more you can choose directly from the list of available topics.

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.

Session 7: scikit-learn

Goals

1. Understand (short) programs
2. Exposure to central concepts of machine learning
3. Examples of data analysis workflows
4. Learn scikit syntax

Content

- Short introduction to machine learning
- sklearn tutorials:
  - Load datasets and explore (identify input data, target data within numpy arrays)
  - Visualize data (with matplotlib)
  - Apply simple ML methods with .fit(): PCA, KMeans

Resources

- https://scikit-learn.org/
- https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html
- https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_iris.html

Block 3

Session 1

Goals

- have an overview of editors and programming environments for python.
- know why and how to use anaconda environments.
- find a project and decide.
- create an environment for your project, choose an editor for your project.
- start working on your project.

Lecture 1 Editors

Overview of editors: - pycharm, - atom, - emacs, - vi

Slides in a notebook: link

Lecture 2 Anaconda environments

Short intro to anaconda envs, demo. Have a look at - https://medium.freecodecamp.org/why-you-need-python-environments-and-how-to-manage-them-with-conda-85f155f4353c - https://conda.io/docs/user-guide/tasks/manage-environments.html

Choosing projects

overview of projects: here

Block 4

Session 2 – Introduction to deep learning frameworks

Goals

- Understand the scaffolding of deep learning: function learning, empirical risk minimization + regularization, matrix products, taking derivatives, following gradients, expressing complex computational graphs.
- Understand the forward pass in terms of function composition of matrix products and nonlinearities
- Appreciate the interest of moving matrix products to the GPU and how to do it in PyTorch.
- Understand the principle of backpropagation, and how to indicate in PyTorch that gradients will be required.

- Obtain a qualitative understanding of the pitfalls and limitations of gradient learning. Know where to find optimizers and tune learning rates in PyTorch.
- Find in PyTorch the building blocks to conveniently create multilayer neural networks.
- Identify helpful framework services for the industrialisation of deep networks: distributed training, model serving, checkpointing, model porting (onnx), monitoring, export to embedded/mobile.

Content

See the slides.

References

- Deep learning with PyTorch: a 60-minute blitz
- Géron: hands-on machine learning with scikit-learn and tensoflow
- Deep Learning Book

Exercises

Find an example of a network defined in PyTorch and identify in the code the places where a) network definition b) loss function specification c) optimizer choice d) learning rate scheduling (if applicable) e) checkpointing or monitoring (if applicable) are happening. Discuss what aspects of the code represent regularization strategies and if there are any explicit regularization terms in the cost function.