Fall term 2016/2017

SCIENTIFIC PYTHON

Course coordinator: Roberta Sinatra, robertasinatra@gmail.com
No. of Credits: 3
Prerequisites: Basic programming skills in any programming language (e.g. familiarity with logical statements, for loops, with different variables), Basic statistics
Course Level: Master and PhD
Cross-listed: Cognitive Science, Economics, Network Science, Environmental Science, Political Science, Environmental Sciences

Brief introduction to the course:
This course will provide a comprehensive, fast-paced introduction to Scientific Python. The course will run with theoretical classes, hands-on sessions and tutorials. We expect you to come to lectures and labs, ask questions when you get stuck, and develop a project taking advantage of tutorials. The course will have an intensive schedule, taking place mostly during the first month of the term.

The goals of the course:
The overarching goal is to equip students with enough programming experience to start working in any area of computation and data-intensive research. This course will lay a foundation from which new tools and techniques can be explored.

The learning outcomes of the course:
By the end of the course, students will have experience with techniques which are vital to effective scientific research, including:
- The basic syntax and use of Python as a scientific tool, including writing and executing scripts to automate common tasks, using the IPython interpreter for interactive exploration of data and code, and using the Jupyter notebook to share and collaborate.
- Loading data from a variety of common formats
- Manipulating data efficiently with Numpy
- Visualizing data with Matplotlib
- Performing basic data mining and machine learning analysis with Scipy
- Basic concepts of Natural Language Processing (NLP)

Course Organization:
Lectures: 8 classes of 120min (theory and hands-on sessions), 4 classes of 150min+15 min break (hands-on and tutorials). Use of a computer will be required during most of the lectures. Students can use their own laptops or the facilities provided by CEU.

Instructions for installation of Python and the required packages will be provided during the first class.
Project coaching and tutorials: 2 classes of 120 min.

Topics
Lectures:

- Introduction to Python Programming: why Python? Basics of Python and IPython
- Data structures: lists, tuples, dictionaries
- Functions and Packages
- Numerical computing (Numpy: how to handle data efficiently and statistics)
- Scientific computing (Scipy: special functions, integration, linear algebra. Scikit learn)
- Matplotlib
- Control flow and Pandas

Hands-on:

- Handling and visualizing datasets
- Analysis of one dataset (with elements of multivariate analysis)
- Complex networks models or use of an API
- Natural Language Processing (NLP)

Tutorials will be designed during the course, based on the students programming background and their feedback during the first lectures.

Tentative calendar
First week
1st class: introduction to Python and IPython notebook
2nd class: running a script, type of variables, data structures (lists, tuples, dictionaries)
3rd class: hands on session on lists, an example of web scraping.

Second week
4th class: Functions and packages. File parsing.
5th class: Hands-on session on dictionaries and file parsing
6th class: Visualization and Matplotlib

Third week
7th class: Characteristics of Numpy. Data analysis with Numpy
8th class: Hands-on session on networks (Networkx)
9th class: Scipy, Scikit-learn and machine learning applications

Fourth week
10th class: Pandas, data-frames
11th class: Pandas applications
12th class: Hands-on with NLP

Mid-November: 1 tutorial class on final projects
Early December:

Course Requirements/Assessment
Students are expected to attend lectures and hands-on sessions, to hand in one assignment during the course and to develop a project, alone or in pairs, during the entire term.

Grading:
- Attendance of the classes and hands-on sessions: 30% of the final grade
- Assignments: 30% of the final grade
- Final project: 40% of the final grade

Final Project
For the final project, students will have to apply and show proficiency with the tools studied during the course. Possible projects will include, but will not be limited to: the analysis of a dataset, implementation and application of an algorithm, development of an interactive tool. A number of options for projects will be suggested in class. The students will also be free to design their own project within the guidelines that will be provided in the lecture.

Suggested reading
- Bill Mark Lutz, Learning Python, O'Reilly (2013) – Also available for free online
- Bill Lubanovic, Introducing Python, O'Reilly (2014)
- Wes McKinney, Python for Data Analysis, O'Reilly (2013)
- Online resources and documentation provided during classes

Further information, such as the course website, assessment deadlines, office hours, contact details etc. will be given during the course.

The instructor reserves the right to modify this syllabus as deemed necessary any time during the term. Any modifications to the syllabus will be discussed with students during a class period. <u>Students are responsible for information given in class</u>.

Cheating

In short: don't do it! You may work with friends to help guide problem solving or consult stack overflow or similar to work out a solution, but copying—from friends, previous students, or the Internet—is strictly prohibited. If caught cheating, you will fail this course. Ask questions in recitation and at office hours. If you're really stuck and can't get help, write as much code as you can and write comments within your code explaining where you're stuck.