Foulques Géraud

MOOC Report

Python for data science

With the

San Diego Supercomputer Center

0 – Introduction

0.1 – Why this course

0.1.1 – Why python in science

I often heard about python, its simplicity and its efficiency. As a programmer I really dreamed to use it and to learn this language. But overall I wanted to discover how to concretely apply computer science in an other science. We already know how to code by our school, but it was an occasion to me, to learn to use a new language for a concrete work.

0.1.2 – Why data science

At the beginning I asked for another python course. Then, as the other was too short, the Learning Unit Responsible, Mr Rasmaoely, decided to make me work on : python for data science. Thus I interested myself in this subject. I was lucky, as data science is a large domain with lots of interesting parts and a quite understandable theory. It was perfect with my goal to learn appliance of computer science in sciences and I was really excited about it.

0.2 – Proceedings

The course was a heavy one : 80 hours attended from people that did not use python before. Fortunately, I had the chance that it was very well explained. It was so a pleasure to work this course along some weeks. However, as the course was long, and the semester very short, I was not able to go until the end of the course. So is there the report of all the things I was able to do.

0.3 – Notations

As the test results were not available, I just explained honestly how I succeeded into passig the test at the end of each section.

1 : Getting started with data science

1.1 – Data science : generating value from data engagement

1.1.1 – Data science : getting value out of data

This course begins with a brief introduction to data science and the many utilization which are present in our lives of this domain, e.g. in NASA's mod. Those explanations come with introduction of numbers of current life which show how much data science is not only used, but needed.

1.1.2 – Why python for data science ?

Data scientists has really plentiful kinds of work to do, especially because their domain is in middle of mathematics, computing and analysis. Python is a way of learning properly data science using powerful programming tools without go deeply in this domain.

1.1.3 – Case study : Soccer Data Analysis

This is an introduction to the first case study, which is made in order to prepare myself using data by allocating data in groups which are analyzed separately.

1.1.4 – Learn about San Diego Supercomputer Center

An optional video which allow us to learn about the place our MOOC teachers are working.

1.2 – The data science process engagement

1.2.1 – How data science happens

In this part we learn about 'use case' of data science : when and how is this supposed to use, and how can we recognize when we are allowed to use it, or not, instead of making fake analysis.

1.2.2 – Asking the right question

In this video we learn to ask correctly the question we want to solve with data science. Problems have to be taken as an investment : how does it cost and how much can it make us earn ? Once the problem is understood, we can correctly ask the question we have to answer.

1.2.3 – Steps in data science

This part is about explaining to us how to process in data science : first we have to understand the nature, quality and format of data. Then arrives pre-processing : cleaning, filtering and sub-setting data (make them readable by a program). After are selected the data analytical techniques in order to build a model. Follows the step where we have to communicate the results. Finally comes the time to review our first problem and propose a solution.

1.2.4 – Step 1 : Acquiring data

We learn that data are often stocked with SQL, or in data files as texts, CSVs, which can be opened by lots of programming languages. There is a word about web-stocked data, in files like XML or Json. APIs can be found in internet.

1.2.5 – Step 2A : exploring data

Here is explained the importance of exploring data before to build a model : we have to master it as data scientists. First, build representations : correlations, general trends, outliers help us to see clearly through data. Mean, median, mode, range, standard deviation are so tools that will help us. Also are presented heat maps, histograms, line graphs, scatter plots and boxplots.

1.2.6 – Step 2B : pre-processing data

Real-world data is never in the format we want. We so have permanently to make them clean and usable, by removing data with missing values, merging duplicate records, generate best estimte for invalid issues and remove outliners. The ‘data munging’ that comes after includes dimensionality reduction (e.g. from unclear 3D to 2D), data manipulation, scaling, transformation (noise elimination) and feature selection).

1.2.7 – Step 3 : Analyze Data

The input data we prepared must now be involved in a model to get output data. To analyze data we must build a model. Classification, regression, clustering is an often-used way to process.

- Classification : the goal is to predict the category of an input data
- Regression is classification is numeric values instead of categories
- Clustering is used to separate categories to study them apart, in order to have a better appreciation of data

Modeling is to choose a technique, and fill the model with data in order to validate it, and use, after, its properties.

1.2.8 – Step 4 : Reporting insights

It is important to present the results we get with data science, even if they are not pleasant, or uncomfortable : we have to show what we found, even if this is the opposite of what we hoped to find. In this kind of presentation, we ave to be aware the results are clear for the public and use good tools (plots, graphs…) to be understandable. We have to make sure we have a question and we answer it, without showing useless data.

1.2.9 – Step 5 : Turning insights into actions

Now that we have answers to our questions, we have to be able to propose actions based on what we found. Some questions can be asked about the action we want to do : can it be automated ? Is it easy to make ? Have we took the good action in mind ? Does already exists a similar case, and how is it treated ?

1.2.10 – Conclusion

Telling about next work.

1.2.11 – Setup

How to prepare our PC to compile python.

1.3 – Test

This test was mixed with both correct answers and fails for me.

2 – Background in python and Unix

2.1 – Python : basics

2.1.1 – Python overview

Python is free, simple, short and runs really fast for an interpretive language. There is some story about python and jupyter (created to support python, julia and R, and run now for like 40 languages).

2.1.2 – Value of python in data science

Python has a great ecosystem, lots of tools and libraries. It is so useful that python programmers are really valuable on the market. Python I also useful in data science as it helps to make data science from the first to the last step. Everything in data science is possible with python.

2.1.3 – Run python

A tutorial to launch python.

2.1.4 – Python : variables

As python is a dynamically-typed language, you don’t need to declare the variables. More, you can affect a int and then a float in the same variable without going in troubles. Finally we learn about the types of python, classified in numeric, sequence, binary, Boolean and text sections.

2.1.5 – Python : objects part 1

We learn that object already exists in python, and that each variable we can create in python is an object, and that a garbage collector destroys all objects (ex-variables) which are not referenced anymore. We learn about operators is and ==.

2.1.6 – Python : objects part 2

We learn how to use the built-in methods of python background object, as, for instance, call the capitalize or lower method on a text variable.

2.1.7 – Python : variables practice quiz

A one-question quiz on affectations.

2.1.8 – Python : variables quiz explanation

A short video explaining why I was right in the micro-test.

2.1.9 – Python : loops

Python loops work with double-comas, indentations, and overall sequence-typed objects as lists, enumerations or ranges to be crossed. We learn how to use the range function and short incremental += for the whiles.

2.1.10 – Python : loop practice quiz

A one-question quiz on loops.

2.1.11 – Python : loop practice explanation

A short video explaining why I was right in the micro-test.

2.1.12 – Python : conditions

In this part we learn to use the if, elif and else conditions in python, as the modulo (%) operator.

2.1.13 – Python : Functions

Then we learn about how to define, with the keyword “def”, functions in python. We explore the notions of return, of parameter, and how to use local and global variables in order to make correct functions. Moreover, we discover that a function can return None.

2.1.14 – Python : function practice quiz 1

A one-question quiz on functions returning.

2.1.15 – Python : function quiz 1 explanation

A short video explaining why I was right in the micro-test.

2.1.16 – Python : function practice quiz 2

A one-question quiz on functions global and local variables.

2.1.17 – Function quiz explanation : False vs Swap

A short video explaining why I was wrong in the micro-test.

2.1.18 – Python : scope

This part explains finally how to create and how to manage with the global variables in our python code.

2.2 – Python : key data structures

2.2.1 – Data structures and basic libraries in python

A 14 seconds introduction to this part

2.2.2 – String functions

In this video we learn to use the functions of concatenation, and then multiplication, on strings. We also learn about upper and lower functions, as well as the strip and split methods or word indexation of chars.

2.2.3 – Lists in python

We learn about lists creations, allocations and management, how to use append, pop, remove, copy, how to manage with the indices of each element of a list, how to iterate over a list, concatenation, loops and other things.

2.2.4 – Reference practice quiz

A one-question quiz on references of elements inside of a list

2.2.5 – Reference quiz explanation

A short video explaining why I was wrong in the micro-test.

2.2.6 – Tuples in python

All the syntax and the errors that can occur about tuples.

2.2.7 – Dictionaries in python

This video explains to us what is, concretely, a dictionary : a set of unsorted values to which are associated other values, independently from their types. We also learn all the syntax about creating, operating or removing about those dictionaries.

2.2.8 – List and dictionary comprehension

In a short video we learn here to use the python comprehension to quickly generate lists or dictionaries, e.g. with the syntax :

list = [f(i) for i in range(0,n)]

2.2.9 – Sets in python

finally we learn about sets, how to create and use it, union and removal operators, etc.

2.2.10 – Assignment : python word count

There is a little work to do over a document we have ton clean, to sort and to extract data from with python by mainly using all the functions we discovered about strings earlier.

2.3 – Unix

2.3.1 – Introduction to Unix

Unix is an operating system which disposes of lots of representations, as Linux, Debian and lots of others. Skills on Unix are really expected by job providers in computer science today, so it's important to master it. The three main parts of this operating system are the Kernel, the shell and the programs. Shell is interface between user and kernel, and launches programs with unique PID. Files architecture is built as a tree, with root as main container.

2.3.2 – Running Unix

This part teach us to launch the Unix-like terminal proposed on Git by the San Diego Supercomputer Center.

2.3.4 – Live code : intro to Unix

Explains the commands ls, cat, cd, pwd, clear and man.

2.3.5 – Basic Unix commands

This video explains us that we can and should use the standard output file stdout for the correct responses of our program, the stderr file for its fails, and of course stdin to provide arguments to our work.

2.3.6 – Live code : basic Unix commands

The commands mv, cp, mkdir, rmdir and ls -a are explained in this part, as well as the use of the * character to get files by replacing any of their name's section.

2.3.7 – Redirecting standard IO

Then we learn to use the >, >> and < operators in order to redirect (for instance) the stderr file in the stdout.

2.3.8 – Live code : redirecting standard IO

And now we can see the effective code which allow us to redirect our standard flows whenever we want, especially in a file, which is very useful because we can by it e.g. write what we want, collect results from a program, or collect errors occurred in stderr.

2.3.9 – Pipes and filters :

There is some theory about the grep command, its usage, and how we can use syntax to find or filter chains of characters in order to collect files by a grep or a ls.

2.3.9 – Live code : pipes and filters

And again concrete examples of code we can use in our folders to collect files with the grep and ls commands, using pipes too.

2.3.10 – Useful Unix commands for data science

Sort, uniq, word count, head, tail and other commands are really useful to explore data, as sed too. Also we learn to use the very very wonderful gnuplot, which allows us to create graphs and visualize data in a really simple way which is often used.

2.3.11 – Live code : useful Unix commands for data science

Here is a live video explaining concrete use of the commands which just were presented in last part.

2.4 – Test

This test was a total success.

3 – Jupyter notebooks and numpy

3.1 – Jupyter notebooks

3.1.1 – Why jupyter notebooks

By combining code, notes and graphics, jupyter notebooks is a powerful way to follow the evolution of our work in data science about a subject. This is a perfect tool for collaborative work and a good presentation of the incremental process of our studies in the steps we described in part 1.2.3 (steps in data science) .

3.1.2 – Live code : getting started

Here we go, and learn to create a new notebook on jupyter. This is very interesting to see that we're able to work from distance on other systems. We also learn to not be confused between the notebook, which will be the readable part of our work, and the notebook application, in which our code will be running and providing results. Finally we learn tips for the use of cells in jupyter notebooks in order to obtain clear results.

3.1.3 – Live code : documenting analysis with markdown text

Markdown on jupyter runs as the code cells : it lets the user enter text in cell, and is really flexible about presentation and beautification, providing its own tools, and supporting powerful editor languages as LateX and HTML and many others.

3.1.4 – Live code : additional tips

This session teaches us to import images in jupyter notebook and how to generate diagrams in it with the plot function. We also learn to close properly a jupyter notebook, to see when a program is running in background, an how to make correct saves.

3.1.5 – Live : using Unix in jupyter

Jupyter notebook proposes an easier way to use Unix commands in code than python's ones : as you can write in a cell using a ! character before your command line, you can enter Unix commands, and use shell instructions with python code in the notebook. We have to note that the default directory stays the one where the notebook is open.

3.2 – Numpy

3.2.1 – Why numpy ?

Numpy offers a number of key features for scientific computing, as multi-dimensional arrays (representing vectors and matrices), tools of linear algebra, and from global mathematics

3.2.2 – Numpy : ndarray basics

Here we learn how to use np.array in order to create 1D- or 2D -arrays (ndarrrays), and the tips about it, like the numpy's function zeros(m,n), which creates table of dimensions n and m filled with 0s. Moreover, we learn about mutability of numpy's arrays.

3.2.3 – Numpy : ndarray indexing

This study of indexations with ndarray give us the competences to use slice indexing to access subsets of an ndarray, and teaches us that such indexing creates a second reference to the same underlying data.

3.2.4 – Numpy : ndarray boolean indexing

Manipulating largely dimensionaly spread data make us needing to deal with clustering, and evaluating data to separate those which respects certain conditions from the others. To proceed, we use boolean indexing. This is possible by creating a predicate that we apply to each element of an array.

3.2.5 – Numpy : ndarray datatypes and operations

Here we learn to be able to examine and to set the datatype from a ndarray (integers, floats, etc.) as well as to use basic common ndarray functions, like add, substract and multiply for matrices.

3.2.6 – Numpy : statistical, sorting and set operations

And then we learn to use statistical operation, as median or average, with ndarrays. The interesting thing is that we can compute mean or median only on some rows and/or columns we select.

3.2.7 – Numpy : broadcasting

Broadcast is a numpy powerful tool which let us operate on arrays that don't have the same dimensions. For instance it lets us add a (1,7) array to a (9,7) array by adding 9 times the first array to the second, one for each column. The operator + between two arrays works for the arrays when they have 1) at east one similar dimension or 2) one of them is 1.

3.2.8 – Numpy : Speed test ndarray vs list

We learn to use the time library, to prepare a timer and then we use it to compare speeds of operations executed by numpy and directly on lists. We find that numpy is 13 times faster than the basic code, which shows that numpy is a great deal.

3.3 – Satellite image application in numpy

3.3.1 – Satellite image examples

In this part, we learn a bit about WIFIRE which is San Diego Supercomputer Center's project. We learn some basics things about images and their formats, as they are arrays made from pixels, that are made from RGB and intensity values, etc. Python type for images is array.

3.3.2 – WIFIRE overview

Here is a link to view a video that explain us that WIFIRE is a program that uses the supercomputer to determinate situation and future evolution of fire by satellite data collecting.

3.3.3 – Live code : satellite example part A

And now we can learn to import the libraries we need in order to analyze and impact images with python. In this part we learn to examine shape and data about image, how to get each pixel's values. Then we can modify them, to colorize an image, to put white or to make effects. We also can show what we get by those modifications.

3.3.4 – Live code : satellite example part B

Finally the video teaches us to cut out a circle from an image. This implies to be able to calculate the euclidean distance from each pixel to a defined center, to make a pixel selection, and set all value to 0 for each f them. Moreover, we can use the 'and' function of numpy to sum filters.

3.4 – Test

The test was not easy but I still succeeded in it.

4 – Pandas

4.1 – Working with pandas part 1

4.1.1 – Why pandas ?

As a numpy built-up library, pandas is a library which benefits from all numpys advantages and contains lots of features data scientists love to use. Moreover, pandas proposes tools to operate on big data sets, like merge or join. Time series, combinations, separations of data sets become easier with pandas.

4.1.2 – Live code : why pandas ?

Pandas offer two data structures that are useful to know. First, the pandas series are a set of data combined with a set of words that are their indexes. The advantages of this structure are that we can access to data both by word-indices and numeral-indices ; and that this data structure can contain data from different types as words, integers, and floats. The defined operators + and * are also able to work on the entire table, adapting to the type of each data.

The data frame structure takes dictionaries (or pandas series) in parameters tat it naturally merges to provide a clear data frame. It has tools to put a beautiful on-board representation of tables, and lots of internal operations.

4.1.3 – Pandas : data ingestion

Now, we can begin to use pandas. In this part we learn to import CSV files with pandas, and that we can operate with SQL query on the databases we imported with pandas, as we did in the first homework.

4.1.4 – Live code : data ingestion

We work now on a concrete case of ingestion. Instead of struggling with python read function, and split with separators, etc. , we discover, working on a bench of movies data provided by the internet; that the pandas.read() function does all the work by itself, in a clear, powerful and simple way.

4.1.5 – Pandas : descriptive statistics

This part is a description and an explanation of the main data science functions that will be used, as mean, max, min, correlation, variance and others.

4.1.6 – Live code : descriptive statistics

In this part the curse teaches us to us pandas methods as min, max, mean, std to get the minimum, maximum, mean and variance of our data sets. We also learn about the .describe method that give us all those indicators, more the quartiles.

4.2 – Working with pandas part 2

4.2.1 – Pandas : data cleaning

As the real data world is messy, python can have to deal with missing data, as NaN values. In order to make the code cleaner, we can remove the missing value rows with the dropna method, but it can be a better alternative to give them a well-thought value, for instance with an estimation.

4.2.2 – Live code : data cleaning

Here is a short video of removing missing data from a table using the dropna method.

4.2.3 – Pandas : data visualization

We learn that bar charts, box plots and histograms are available with pandas, and what do they mean. We learn their representation and sense.

4.2.4 – Live code : data visualization

Then we can see how the functions corresponding to those graphs. First of all, we have to ‘import’ matplotlib. For this we use the persent symbol, which allows us to go through jupyter to matplotlib in order to catch the ‘magic functions’ that are useful for us. We learn to use .hist() and .boxplot() methods and how to manage their arguments.

4.2.5 – Pandas : frequent data operations

And now we are formed to use the groupby, del, loc and drop functions, their syntaxes, use and options.

4.2.6 – Live code : frequent data operations

And finally there are a lot of exemples and usecases about those functions.

4.3 – Working with pandas part 3

4.3.1 – Pandas : merging dataframes

There is often a need in data science to merge tables : as the .concat() or .append() pandas methods can be used on pandas dataframes, they are not really useful because they just make data successive, regardless of their key-values. By using the .concat method with join parameter, we can obtain a good representation of the dataframe, but wit the key-columns present twice. Finally, we have to use the .merge() method to obtain a correct fusion of dataframes.

4.3.2 – Pandas : frequent string operations

The string method are called by .str.method() with pandas. The method split separates a string in two tables of string, based on a separator that is passed in argument. The extract method is used to compose a text with pieces of data. The contains method returns a Boolean indicating if a character is contained within a chain, and the replace method can be used to modify such strings.

4.3.3 – Pandas : parsing timestamps

Here we learn how works the POSIX time, equal to the number of past seconds from the 1st of January 1970. We also learn to use the to_datetime method to work with it, and all the ways to manage different dates and times formats with python.

4.3.4 – Pandas : summary of movie rating notebooks

Just a video that summarize all we did about pandas.

4.4 – Test

A well-done test for me with a high majority of good answers.

5 – Conclusion

5.1 – What I did

To summarize, I succeeded into going quite far into this MOOC. I was not able to reach the introduction to machine learning section, but I could touch and begin to master the Data visualization part, that I did not describe in this report, as uncompleted. I really played with code ore than doing big exercises, and this permitted to me to really master python, its subtilities, and overall to master data science.

5.2 – What I learned

This was an occasion to learn a lot about data science. Not in the mathematics point f view, which we had to master last year, but in the situation of : how concretely do data science, what is the spirit of the thing, the real method to follow from the beginning of a concrete problem to the solutions we have to propose at the end.

It also was an occasion to discover new tools, and news for of coding, especially when we learned about images treatment. Also data management, building, merging, cleaning were domains I never heard about and I was excited to discover.