

Statistical Computing with Python

Author: Laercio Serra

This is a quick tutorial and here I'll show you "how-to do" some statistical programming tasks using python. For that, is necessary to have some basic knowledge with python and be familiar with statistical programming in a language like R, Stata, SAS, SPSS or Matlab.

Why Python?

There are many good reasons to choose Python as your primary programming language. First of all Python is an easy to learn, powerful programming language. Furthermore it has efficient high-level data structures, which allow you to write complex operations in fewer statements than in C, C++ or Java. Object-oriented programming is a lot easier than in languages like Java.

Python has become one of the most popular programming languages among developers and programmers. They praise it for its clean syntax and code readability. Python is a general-purpose high-level programming language. Python is both object oriented and imperative and it can be even used in a functional style as well. Python programs are portable, they can be ported to other operating systems like Windows, Linux, Unix and Mac OS X, and they can be run on Java and .NET virtual machines.

David Beazley says in his foreword to the book "How to Think like a Computer Scientist Learning with Python" by Jeffrey Elkner, Allen B. Downey, and Chris Meyers: Despite Python's appeal to many different communities, you may still wonder "why Python?" or "why teach programming with Python?" Answering these questions is no simple task-especially when popular opinion is on the side of more masochistic alternatives such as C++ and Java. However, I think the most direct answer is that programming in Python is simply a lot of fun and more productive.

And I agree with this declaration. In particular I also think that Python is simply a lot of fun and more productive. And to close this discussion ... I bring these infographics. They are very interesting, and they are showing the leading differences between R and Python languages. These infographics were created by [Data Camp](#) Team.

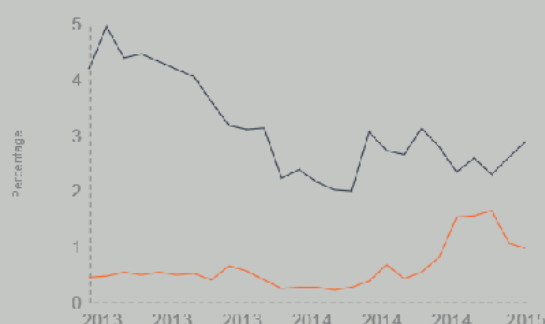
Use the rPython package to run Python code from R. Pass or get data from Python, call Python functions or methods.

Use the RPy2 library to run R code from within Python. It provides a low-level interface from Python to R.

R and Python: The Numbers

Popularity Rankings

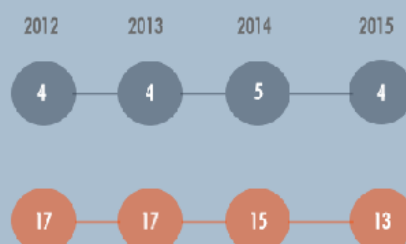
R and Python's popularity between 2013 and February 2015 (TioBE Index)



Redmonk ranking, comparing the relative performance of programming languages on GitHub and Stack Overflow (September 2012 and January 2013, 2014, 2015)

Python

R



R x Python – The Numbers

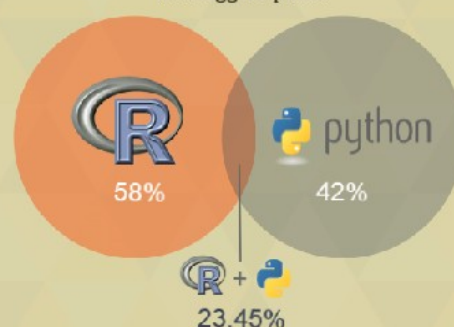
R And Python: The Quantified Battlefield

General

Languages for data analysis used in 2014 (KDnuggets polls)



Analysis of R and Python used together in 2014 (KDnuggets polls)



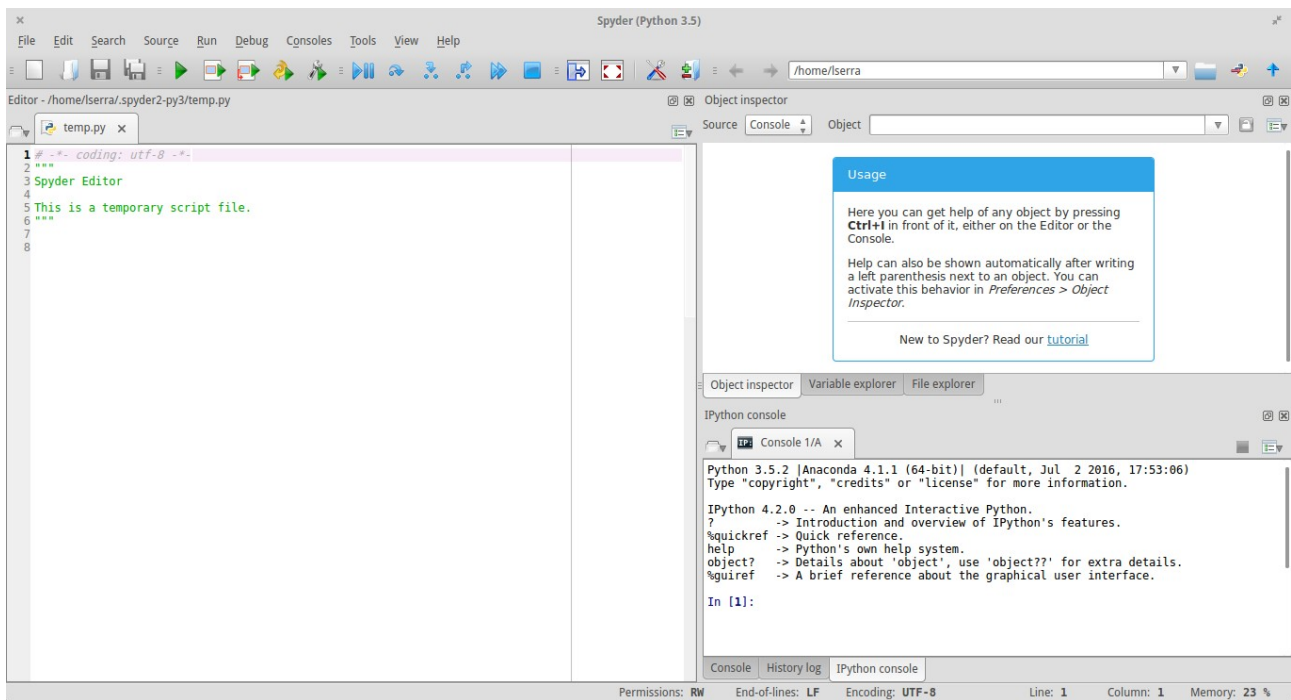
R x Python – The Quantified Battlefield

For more details or information about it, please visit the following address:

<https://www.datacamp.com/community/tutorials/r-or-python-for-data-analysis#gs.47mrfLw>.

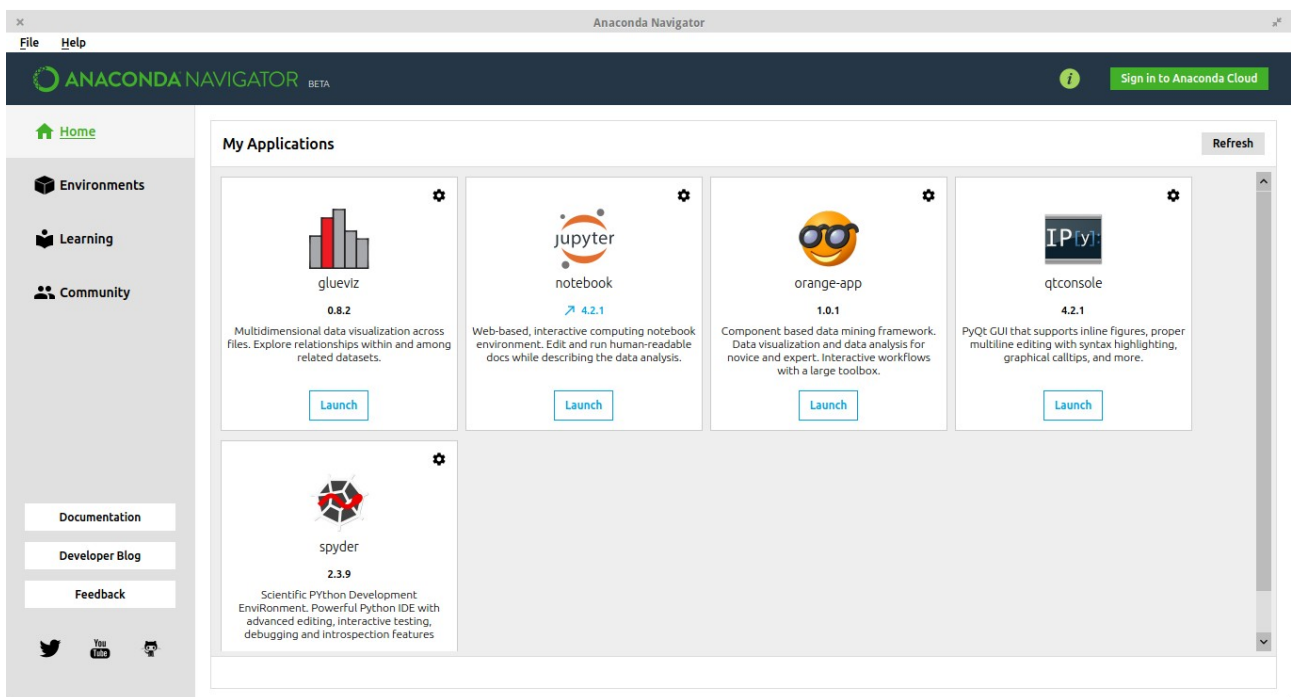
The environment (setup)

To accomplish the tasks of this tutorial is necessary to have the Spyder (IDE) and the following libraries: numpy, scipy, pandas, matplotlib, requests. You can use the “pip install”, to download and install these libraries above.



Spyder (IDE)

For the new users I recommend to install the “*Anaconda - Open Data Science Platform*”.
(For more details about it, please visit the site <https://www.continuum.io/>)



Anaconda Navigator

Let's start

This tutorial was divided in two parts.

Part 1

This first part is easier than the second part. In this part, I talk about the capture and manipulation of data tasks, using the Pandas library. Pandas, is an open-source library providing high-performance,

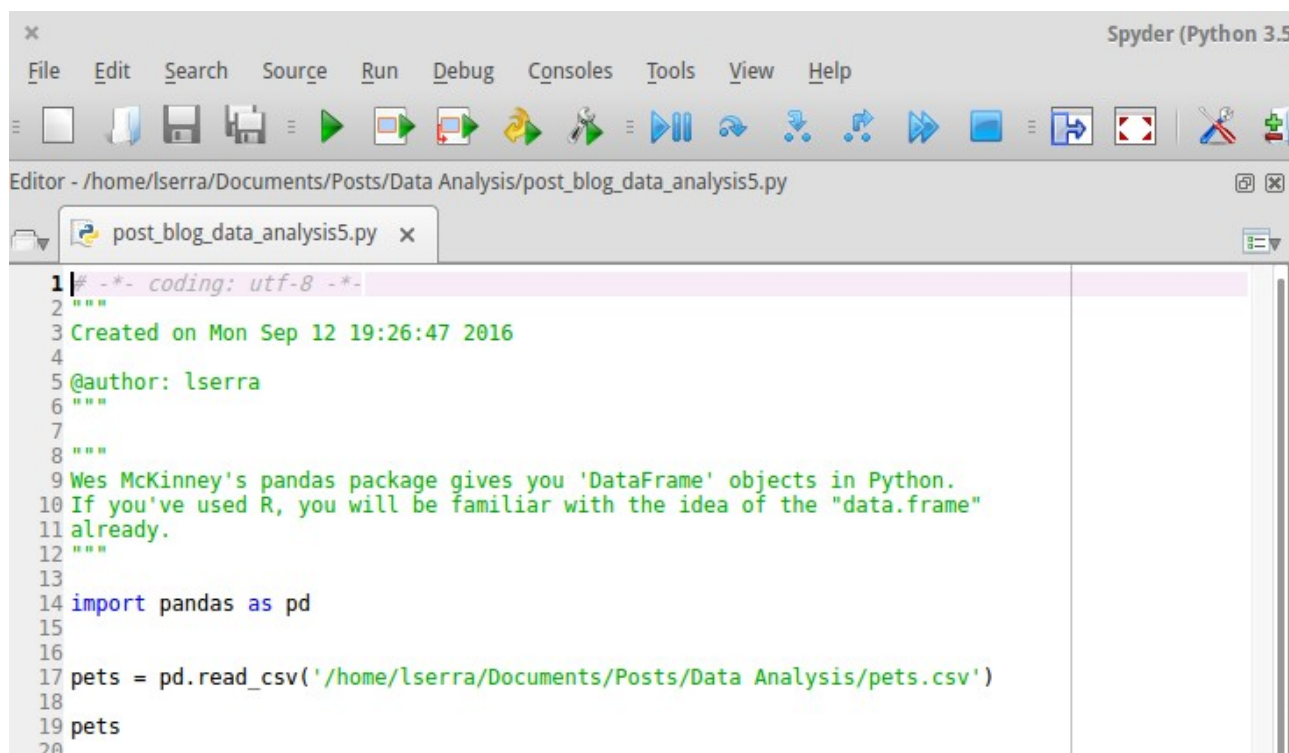
easy-to-use data structures and data analysis tools for the [Python](#) programming language.

But also, how to do data visualization using the Matplotlib library. Matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in python scripts, the python and [ipython](#) shell (ala MATLAB® or Mathematica®), web application servers, and six graphical user interface toolkits.

Well to demonstrate all that, we'll use a little dataset. Please [click here](#) to download the file (".csv").

1. Reading a .CSV file using the Pandas package

Launch the Spyder IDE and click on the menu File > New file, or press the buttons (Ctrl+N). Thus write the code bellow, after that press the button F5 (Run File).



The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Tools, View, and Help. Below the menu is a toolbar with various icons for file operations and execution. The main window is titled 'Editor - /home/lserra/Documents/Posts/Data Analysis/post_blog_data_analysis5.py'. The code editor contains the following Python code:

```
1 |# -*- coding: utf-8 -*-
2 |"""
3 |Created on Mon Sep 12 19:26:47 2016
4 |
5 |@author: lserra
6 |"""
7 |
8 |"""
9 |Wes McKinney's pandas package gives you 'DataFrame' objects in Python.
10 |If you've used R, you will be familiar with the idea of the "data.frame"
11 |already.
12 |"""
13 |
14 |import pandas as pd
15 |
16 |
17 |pets = pd.read_csv('/home/lserra/Documents/Posts/Data Analysis/pets.csv')
18 |
19 |pets
20 |
```

The right side of the IDE, which would typically contain the IPython Console, is currently empty.

The result appears in the right side (IPython Console). In this console you can see the content of the file (.CSV) in columns x rows (tabular format).

IPython console

IP: Console 1/A x

Out[4]:

	name	age	weight	species
0	fluffy	1	13	cat
1	vesuvius	2	3	fish
2	rex	3	33	dog
3	garu	4	43	dog
4	annie	5	13	cat
5	thor	6	14	cat
6	pokemon	5	4	fish
7	kalunga	4	34	dog
8	storm	3	44	dog
9	tom	2	14	cat
10	travis	1	15	cat
11	red	2	5	fish
12	sparrow	3	35	dog
13	spark	4	45	dog
14	hadoop	5	15	cat
15	linux	6	12	cat
16	windows	7	2	fish
17	google	8	32	dog
18	furious	9	42	dog

Console History log IPython console

For R users note that Python, like most normal programming languages, starts indexing from 0. R is the unusual one for starting from 1.

Now, write the second part of the code (described bellow) and after that you must have to execute each piece of the code (line by line). So, you can see and also to understand the results that has been generate. For example, select the lines 23 and 24 then press the buttons CTRL+ENTER (Run current cell). Repeat this procedure to all the others lines.

```

22 # two different ways to print out a column
23 pets.age
24 pets["age"]
25
26 pets.head(2) # prints first 2 rows
27 pets.tail(1) # prints last row
28
29 pets.name[1] # 'vesuvius'
30 pets.species[0] # 'cat'
31 pets["weight"][2] # 34
32
33 # in R, you would expect to get 3 rows doing this, but here you get 2:
34 pets.age[0:2]
35
36 sum(pets.age) * 2 # 28
37 max(pets.weight) - min(pets.weight) # 20
38

```

If you are doing some serious linear algebra and number-crunching, you may just want arrays, not Data Frames. Data Frames are ideal for combining columns of different types.

For more details or information about the use of this library, you can see the documentation on <http://pandas.pydata.org/>

2. Plotting a chart using the Matplotlib package

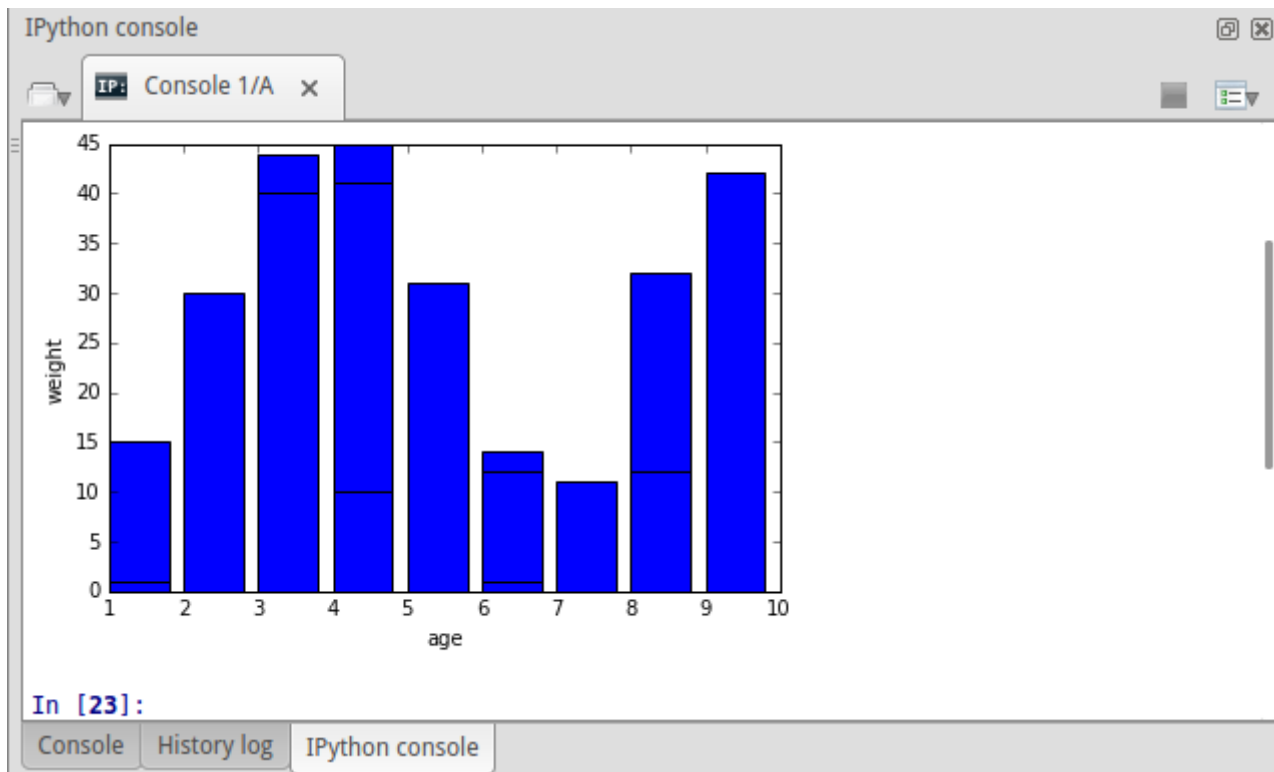
To plot a chart we use the “matplotlib” library. Write the lines bellow:

```

39 # To do data vizualization in Python, use matplotlib
40 import matplotlib.pyplot as plt
41
42
43 # plotting a vertical bar chart
44 plt.bar(pets.age, pets.weight)
45 plt.xlabel("age")
46 plt.ylabel("weight")
47

```

This is the result:



Now, let's reproduce the code bellow:

```

48
49 # plotting a pie chart
50 # The slices will be ordered and plotted counter-clockwise.
51 cols = pets["species"].unique()
52
53 labels = []
54 for c in range(len(cols)):
55     labels.append(cols[c])
56
57 rows = pets["species"].value_counts()
58
59 sizes = []
60 for r in range(len(rows)):
61     sizes.append(rows[r])
62
63 colors = ['yellowgreen', 'lightskyblue', 'lightcoral']
64 explode = (0, 0.1, 0) # only "explode" the 2nd slice (i.e. 'fish')
65
66 plt.pie(sizes, explode=explode, labels=labels, colors=colors,
67         autopct='%1.1f%%', shadow=True, startangle=90)
68
69 # Set aspect ratio to be equal so that pie is drawn as a circle.
70 plt.axis('equal')
71 plt.show()

```

This is the result:

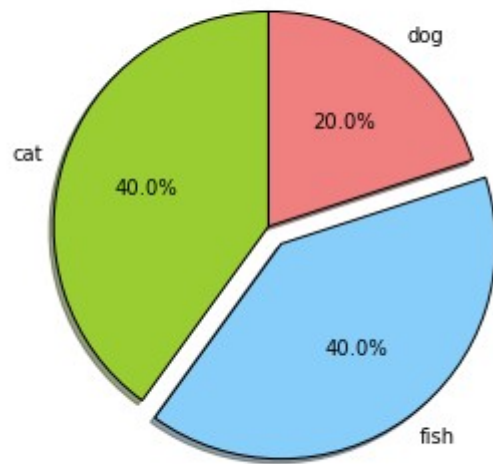
IPython console



IP: Console 1/A



....:



In [154]:

Console

History log

IPython console