# APA-Intro-Python-for-ML

September 6, 2018

## 1 Python for Machine Learning

This notebook does not have code, it just has links to resources and other notebooks that show the basics of the python libraries used for Machine Learning.

As you will notice the APA course is oriented to R, so these notebooks are for people interested in going a little beyond. My intention is to open to you the possibility to know the other most used language in Machine Learning and Data Science.

### 1.1 Why Python?

Python is one of the preferred languages in the Machine Learning community and it has gained a lot of popularity because it has been chosen as one of their preferred languages by the main internet companies in the current AI revolution (Google, Facebook, Amazon, Microsoft, ...)

It has several advantages

- It easy to learn
- It is a general purpose programing language and it is used for almost evertyhing
- It has libraries/tools for anything that can be needed in a ML project (from gathering data, preprocessing, prototyping, modeling and deployment)
- There are only a few basic libraries for ML well designed with common interfaces and usually compatible among them
- A large community of ML users that improve and add new tools/libraries constantly
- First language in job demand according to the IEEE Spectrum 2018 programming languages ranking

### 1.2 What about R?

The discussion about if python is better than R for Machine Learning/Data Science or viceversa is at the same level of windows vs linux, PC vs Mac or vi vs emacs. Mostly is a matter of taste, you like R or you don't, the same goes for python.

Full disclosure, I am a python fan, so I may be biased ;-)

More computer science oriented people hate R with all their soul because it has not been really designed as a programming language, its syntax is a little odd and it takes time to get used to it and despite there are a lot of packages they have been designed by different people without really a plan using different interfaces/parameters and sometimes is difficult to do something without having to use many libraries together.

More data analysis/statistics oriented people love R because they are used to it, it has been there forever and it is similar to other statistical tools or languages like minitab, SAS or SPSS. The python fans would say that R fans do not now any better.

R is ok for most of the tasks for ML, it is older than python and it is very oriented to statistical analysis so it has more specialized libraries for that and if you have to develop a ML proyect in a specialized area more statistics oriented probably you will have R libraries that will solve your problem. Although, once you have your model you will have to resort to other language for deployment.

The advantage of this hate/love relationship is that both languages are taking the good things of the other, putting some order in the libraries/tools and allowing to use one language from the other so both communities have access to all tools. See for example the collection of R packages in the Tidyverse (https://www.tidyverse.org/).

In the end, the funny thing is that the core of R and python is the well optimized matrix operations/linear algebra/optimization libraries that were implemented in Fortran/C decades ago.

### 1.3 Why jupyter notebooks?

Jupyter notebooks are the best way to experiment with data and document the results, they have several advantages:

- You can use the different free on-line platforms for running notebooks (Microsoft Azure, IBM cognitive class lab, Google colaboratory)
- If you use a python distribution like anaconda all that you need is pre installed or easy to install, if you are using the standard python distribution everything it is also easy to install (just use pip)
- If you do not want to bother installing anything, there is a docker image data scientist workbench that has everything installed (but you have to install docker)
- They are interactive, not oly on the sense that they embbed an interpreter of the language you use, but also you can integrate interactive graphics (via libraries using javascript)
- They allow to use different languages, for example python and R, but also https://github.com/jupyter/jupyter/wiki/Jupyter-kernels
- They can be integrated with big data processing frameworks like Hadoop and Spark
- They can be shared on line
- They can be exported to multiple formats for creating documentation (html, markdown, latex, pdf)
- They are sexy and all the cool kids are using it ;-)

### 1.4 Basic Python

If you have taken the Programming Languages course you already have the basics.

If not, you have:

- these slides explaining the basics
- the python official tutorial
- some notebooks of basic python
- Another notebook

The material that you have in the notebooks works for python 3 and probably it will work also for python 2, but be aware the end of life of python 2 is january 2020

## 1.5 ML libraries

The notebooks use 4 mainly libraries for ML tasks

- Pandas for data manipulation
- numpy/scipy for matrix operations and linear algebra
- scikit-learn for most of the ML algorithms
- stats-model for some regression algorithms

Visualization is an important thing in data analysis and ML and like in R there are basic plotting libraries and more advanced ones.

For plotting, the main library is matplotlib, but there are other libraries based on it that make more easy to create some plots like seaborn. Pandas uses matplotlib under the hood for plotting and most of the basic plots can be obtained directly as method from Pandas DataFrames.

If you are used to R's ggplot2, there are python implemetations like ggplot or plotnine.

There are also other python plotting libraries that allow interactivity and are based on javascript and the bowser like plotly and bokeh

### 1.5.1 Pandas

Pandas (http://pandas.pydata.org/) is the port to python of R's dataframes.

The library is very efficient, it is more easy to use for more CS oriented people. It has all the functionality of R dataframes in a more understandable manner (more pythonic and OO).

The full documentation is long and very good, but for a basic preview you can read the 10 minutes to Pandas summary.

You can also have a look to this notebook

### 1.5.2 Numpy/Scipy

Numpy/Scipy (http://www.numpy.org/, https://www.scipy.org/) are the scientific libraries for python. They are very efficient implementation of data array/matrices for python.

These libraries have all the matrix operations, linear algebra algorithms and statistical tools that you will need for ML.

The numpy official tutorial is here

You can learn the basics of numpy with this notebook

### 1.5.3 Scikit-learn

Scikit-learn (http://scikit-learn.org/stable/) is the main library for ML in python and it has set the standard for the implementation of other libraries. Other complementary libraries and extensions follow verbatim their classes signatures and implementation style. This makes for a very large ecosystem of ML methods implementations completely compatible with each other.

Basically all the algorithms are objects with a `fit` (train) and `predict` method.

This is their tutorial, but their documentation is very extensive and comprehensible with lots of examples.

Most of the algorithms that we are going to use are from this library, but it has also all the tools for all the ML process, from loading and transforming data, preprocessing, dimensionality reduction, training and model selection and evaluation.

### 1.5.4 Matplotlib

Matplotlib (https://matplotlib.org/) is the basic library for plottin in python. It is low level as the R plotting functions are. It has all the basic plots, and the best way to learn how to use it is to go to the examples section of the documentation. This notebook shows all the basics.

Most of the basic plots are also functions of Pandas (that uses matplotlib underneath) so if you have your data in a DataFrame is easier to plot this way. Also if you have a numpy array with results sometimes is easier to transform it into a Pandas DataFrame for plotting.

Seaborn (https://seaborn.pydata.org/) is another plotting library based on matplotlib. It makes easy to create specific plots and is also Pandas compatible. The best way to lear how to use it is to look at its gallery

For the people used to R ggplot2 (very brave people) there are ports of this library line ggplot(http://ggplot.yhathq.com/) or plotnine (https://plotnine.readthedocs.io/en/stable/) but they are still on development.

### 1.5.5 Statsmodels

Statsmodels (http://www.statsmodels.org/) is a statistical analysis python library. From this library we are only interested on the linear regression and logistic regression algorithms. Although, the scikit-learn library has already those algorithms and basically we are using it because it is very similar to the R package for regresion and its functions returns a report for the model that is also similar to the report from R functions.

You can substitute these algorithms by the scikit-learn counterparts without problem if you are not interested in the fancy report for the model.

### 1.5.6 Other libraries

In the era of Deep Learning there are more advanced packages that can be used for machine learning, they are considered mostly neural network libraries by the common public, but they are actually linear algebra packages that are the basis for most of the ML algorithms, so many of the algorithms from the previous packages can be implemented from scratch using these libraries. Obviously these packages go beyond the methods that we will see during the course, they are at the bleeding edge of neural networks research but they are also focused on industrial applications. Their developement cicle is also very fast with a new version almost every month.

The basic principle of all these libraries is that all ML algorithms are just functions that can be described by a graph and that can be optimized using clever optimization algorithms (basically advanced gradient descent) using automatic differentiation.

This makes the implementation of algorithms uniform, you describe the graph and the optimization algorithm takes care or the rest. They are also oriented for real big data (millions of example) with distributed and GPU training

These libraries were developed initially in academia by the leaders on neural networks/ML research, but now have been taken over literally by the big internet companies like Google, Facebook, Amazon and Microsoft (literally means that most of the big guns in ML research and their teams now work for these companies).

The main libraries are:

- Google's Tensorflow (https://www.tensorflow.org) (with Keras if you do not want to get your hands dirty with low level nuisance)
- Facebook's Pytorch (https://pytorch.org/)

- Amazon/MS/intel/Baidu backed open source MXNet (https://mxnet.apache.org/) (with Gluon for avoiding going low level)
- Microsoft's CNTK (https://www.microsoft.com/en-us/cognitive-toolkit/)

These libraries are hard stuff and complex to use for beginners so they are the next level if you are really interested on machine learning and AI applications in industry.

## 2 Additional material

There are plenty material about python for ML and data analysis using pandas/scikit-learn/numpy/matplotlib in the internet, some useful resurces are:

- Notebooks from the Python Data Science Handbook
- Notebooks from MS Azure - Python for DS 101
- The Scipy Lectures