# Assignment 1 – Data science tools and libraries

*The purpose of this assignment is to recognize Python libraries and commands/syntax that are relevant for data science and statistical analysis*

This assignment provides you with an opportunity to demonstrate the achievement of the following course learning outcomes:

* Know most important libraries in Python that will be essential for implementation of data science projects in Python
* Be able to manipulate datasets in Python

## Key Information

* **Type:** *Individual*
* **Weight:** 6.5%
* **Delivery:** Course website upload
* **Due Date:** End of lab session

## Expectations

You are expected to complete this assignment individually.

Respect for academic integrity is crucial to your success. Make sure you understand what constitutes acts of academic dishonesty in the page: [What is Academic Dishonesty?](http://mcmaster.ca/academicintegrity/students/whatis.html)

## Instructions

*Using Python, you are to complete the following questions.* ***Please submit your answers (CODE USED AND OUTPUT) as PDF* *files to the course website submission folder.***

This lab will introduce Python libraries that are most important for data science. Below is a list and description of the most popular and important Python libraries:

**NumPy**

[NumPy](http://www.numpy.org/) is an open source extension module for Python. It provides fast precompiled functions for numerical routines. It’s very easy to work with large multidimensional arrays and matrices using NumPy.

Another advantage of NumPy is that you can apply standard mathematical operations on an entire data set without having to write loops. It is also very easy to export data to external libraries that are written in low-level languages (such as C or C++), and for data to then be imported from these external libraries as NumPy arrays.

Even though NumPy does not provide powerful data analysis functionalities, understanding NumPy arrays and array-oriented computing will help you use other Python data analysis tools more effectively.

**Scipy**

[SciPy](http://www.scipy.org/) is a Python module that provides convenient and fast N-dimensional array manipulation. It provides many user-friendly and efficient numerical routines, such as routines for numerical integration and optimization. SciPy has modules for [optimization](http://en.wikipedia.org/wiki/Optimization_%28mathematics%29),  [linear algebra](http://en.wikipedia.org/wiki/Linear_algebra),  [integration](http://en.wikipedia.org/wiki/Integral) and other common tasks in data science.

**Matplotlib**

[Matplotlib](http://matplotlib.org/) is a Python module for visualization. Matplotlib allows you to quickly make line graphs, pie charts, histograms and other professional grade figures. Using Matplotlib, you can customise every aspect of a figure. When used within [IPython notebook](http://ipython.org/notebook.html), Matplotlib has interactive features like zooming and panning. It supports different GUI backends on all operating systems, and can also export graphics to common vector and graphic formats like PDF, SVG, JPG, PNG, BMP, GIF, etc.

**Scikit-Learn**

[Scikit-Learn](http://scikit-learn.org/) is a Python module for machine learning built on top of SciPy. It provides a set of common machine learning algorithms to users through a consistent interface. Scikit-Learn helps to quickly implement popular algorithms on datasets. Have a look at the list of [algorithms available in Scikit-Learn](http://scikit-learn.org/stable/user_guide.html),  and you will realise that it includes tools for many standard [machine-learning tasks](http://strata.oreilly.com/2013/09/gaining-access-to-the-best-machine-learning-methods.html) (such as clustering, classification, regression, etc.).

**Pandas**

[Pandas](http://pandas.pydata.org/) is a Python module that contains high-level data structures and tools designed for fast and easy data analysis operations. Pandas is built on NumPy and makes it easy to use in NumPy-centric applications, such as data structures with labelled axes. Explicit data alignment prevents common errors that result from misaligned data coming in from different sources.

It is also easy to handle missing data using Python. Pandas is the best tool for doing [data munging](http://en.wikipedia.org/wiki/Data_wrangling).

**Theano**

[Theano](http://deeplearning.net/software/theano/) is a Python library for numerical computation, and is similar to Numpy. Some libraries such as Pylearn2 use Theano as their core component for mathematical computation. Theano allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently.

**NLTK**

[NLTK](http://www.nltk.org/) is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to [over 50 corpora and lexical resources](http://nltk.org/nltk_data/) such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, and wrappers for industrial-strength NLP libraries. NLTK has been used successfully as a platform for prototyping and building research systems.

**Statsmodels**

[Statsmodels](http://statsmodels.sourceforge.net/) is a Python module that allows users to explore data, estimate statistical models, and perform statistical tests. An extensive list of descriptive statistics, statistical tests, plotting functions, and result statistics are available for different types of data and each estimator.

**PyBrain**

[PyBrain](http://statsmodels.sourceforge.net/) is an acronym for “Python-Based Reinforcement Learning, Artificial Intelligence, and Neural Network”. It is an open source library mainly used for neural networks, reinforcement learning and unsupervised learning.

Work in Python:

1. Import Numpy, Pandas, Scipy, matplotlib, sklearn, cvs libraries
2. Open supplied CVS file “Office”
   1. Use: pandas.read\_cvs(“Name,cvs”)
3. Characterize the dataset:
   1. Use: name.describe()
4. Create two new variables X and Y with number of values matching length of columns in the imported file:
   1. Use: numpy:random.random(length)
5. Scale created variables to be within 0-100 value range
6. Create third variable Z as a function between X and Y that utilizes functions “+”, “\*”, “/”
7. Create forth dataset ZZ as ascending range of values from 0 to the number of elements each vector should have:
   1. Use: ZZ =np.arange(numberofelements)
8. Combine X, Y, Z and ZZ informing a table T:
   1. Use: T = numpy.array ( list (zip (x,y,z,zz)))
9. Combine T with the original dataset imported from CVS:
   1. Use: dataset3 = numpy.c\_(dataset1, dataset2)
10. Create labels for new dataset so X, Y and Z columns are labeled as “Demand”, Supply”, “Projected growth”, ‘Index’
    1. Use: column\_labels = [‘Name’, ‘Name2, “Name3’]
11. Combine labels with the dataset:
    1. Use: pandas.dataframe (dataset, columns = column\_labels)
12. Select rows that have values satisfying following conditions:
    1. Units>20; Price<100, Demand<500
    2. USE: df[(df.Col1 < XYZ) & (df.PP < 60)]
13. Plot demand as a function of price:
    1. Use: import matplotlib.pyplot as plt

plt.plot(DF.K[:],DF.PP[:])

plt.show()

1. Export the dataset as a CVS file:
   1. Use: datasetname.to\_CVS(‘outputfilename.cvs’)