## Contents

1. Python distribution and package managers
   1. Anaconda
   2. Pip
   3. Conda
   4. Broader overview: conda, pipenv, venv
2. Python
   1. Standard Python Library
   2. Numpy
   3. Pandas
   4. Code style
3. Jupyter Notebook
   1. Google Colab

## 

## Introduction

In this module, we cover the data science toolset and basics needed to complete the course. We start with Python distribution and installation of one. Then briefly go through the language and frequently used packages. After that learn what is and how to use Jupyter Notebook and one of its variations - the Google Colab. You can skip any part, if you feel confident about it, or return to some pages when you need to later in the course.

## 

## 

## Python distribution and package managers

### Anaconda

There are multiple ways to install Python. The suggested one is to install **Anaconda** distribution, which comes with a set of pre-installed libraries for data science. Although it's a good go-to option for newcomers, one can prefer to install miniconda distribution and all needed packages themselves.

To learn more and install Anaconda distribution go through [this page](https://www.anaconda.com/products/individual)

(recommended). To install more light-weighted Miniconda distribution you need [this page.](https://docs.conda.io/en/latest/miniconda.html)

### Pip

Data Science comes with many packages. The most popular way to install everything is via **pip**. Basic usage of it can be found [here.](https://pip.pypa.io/en/stable/quickstart/)

### Conda

Most data science packages can be installed via pip. However, some of them can have external Java, C, or other dependencies. While pip is a Python package manager, **conda** is a package manager for everything and so can be used to install all needed dependencies.

If there is a possibility to install a package via conda instead of pip, you should go for it. Chances are it will work right out of the box. To learn more about conda and how to use it look at [this](https://docs.conda.io/projects/conda/en/latest/user-guide/getting-started.html) official tutorial or [this](https://geohackweek.github.io/Introductory/01-conda-tutorial/) which is a bit more comprehensive.

(*optional*) There are more about pip and conda differences. If you want to learn more, look at this [page](https://www.anaconda.com/blog/understanding-conda-and-pip)  on the topic.

### Broader overview: conda, pipenv, venv

Conda is not the only tool to control and configure Python environments. Some alternatives can be represented by [pipenv](https://pypi.org/project/pipenv/) and [venv](https://docs.python.org/3/library/venv.html).

One reason why a Data Scientist could choose either pipenv or venv over conda is lightweightness. Another point is that it’s not quite clear at the point, whether Anaconda [is free for commercial use](https://www.reddit.com/r/Python/comments/iqsk3y/anaconda_is_not_free_for_commercial_use_anymore/) at the moment.

* [Why you need Python environments](https://www.freecodecamp.org/news/why-you-need-python-environments-and-how-to-manage-them-with-conda-85f155f4353c/).
* [Pipenv, venv, conda](https://medium.com/@krishnaregmi/pipenv-vs-virtualenv-vs-conda-environment-3dde3f6869ed).

## Python

Depending on the task people use other languages like R, Java, or C++ for Data Science. However, the best one in terms of available packages, simplicity, and power is **Python**. This course is written and intended to be taken in Python.

### Standard Python Library

There are many packages in the Python standard library and it's good to know what exists. [This](https://docs.python.org/3/tutorial/stdlib.html) is a brief tour through Python stdlib. [Here](https://docs.python.org/3/library/functions.html#built-in-funcs) you can find some handy and frequently used built-in functions.

(*optional*) When working with text it is very useful to use a re module for regular expressions. You can look at it now or come back during the module related to text classification.

### Numpy

**Numpy**

is a basis for many data science packages. It's important to understand the core basis of the library and how to use it.

Specific articles:

* [Numpy quickstart](https://numpy.org/doc/stable/user/quickstart.html)
* [Broadcasting](https://numpy.org/doc/stable/user/basics.broadcasting.html)

### Pandas

**Pandas** is a well-developed and the most popular tool for data wrangling in Python. It shares many principles with Numpy. A short introduction to Pandas can be found [here](https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html), but don't stop on that. Pandas will be used in every module of this course.

If you prefer video tutorials, there is a great series on Pandas [here](https://www.youtube.com/playlist?list=PL-osiE80TeTsWmV9i9c58mdDCSskIFdDS) which covers all base operations and common use cases.

Specific articles:

* [Merge, concatenate dataframes](https://pandas.pydata.org/pandas-docs/stable/user_guide/merging.html)
* [Groupby](https://pandas.pydata.org/pandas-docs/stable/user_guide/groupby.html)

### Code Style

Luckily, there are written guidelines for Python. You can find them [here](https://www.python.org/dev/peps/pep-0008/). Clear code is as important as other results of data scientist's work.

## Jupyter Notebook

Instead of developing in IDE data scientists prefer interactive environment for visualization and quick prototyping. One such environment is a **Jupyter Notebook**. The official documentation has a good starting [tutoйrial](https://jupyter-notebook.readthedocs.io/en/stable/notebook.html).

You can find a video tutorial on the installation and usage of the notebooks

[here.](https://www.youtube.com/watch?v=HW29067qVWk)

Most notable pages of Jupyter Notebook [documentation](https://jupyter-notebook.readthedocs.io/en/stable/index.html):

* The Jupyter Notebook
* User interface components
* Notebook Examples
* What to do when things go wrong

Jupyter Notebook is a powerful tool and has many extensions and capabilities except for running Python scripts. You can learn some advanced tricks [here](https://www.dataquest.io/blog/jupyter-notebook-tips-tricks-shortcuts/). They surely make your work more efficient and easy. Some notables are:

* shortcuts
* magic commands
* shell commands
* extensions

### Google Colab

**Google Colab** provides the same functionality as Jupyter notebook and extends it. Most popular data science tools are pre-installed on Colab and can be used right away. It's good for prototyping and training of simple neural networks since Google provides **free GPU instances** on Colab. See the Colab [introduction notebook](https://colab.research.google.com/notebooks/intro.ipynb) to learn more.