Welcome to the course!

When was the first time you thought you wanted to learn how to code? or learning python? Welcome to this "course" of introduction to python oriented to Machine Learning.

After feedback from hundreds of students, we identified that many of them needed a tune-up before starting Saturdays AI. That's why we've created this course by guiding you through the best resources.

What do we do in Saturdays.AI?

As you know, the philosophy of Saturdays.AI is not to generate more resources. On the internet there are countless of them, and of a lot of quality. You just need to understand how to group and sort them to remove the first barriers in the learning curve.

👇 If you want to know how the methodology works 👇

https://youtu.be/O2QdXDQA9fY

Let's go from 0 to the minimum knowledge to use python with data science libraries. And before we start, we want to let you know that we **will leave a** survey at the end **of the course so that we can collect all** the feedback **and improve and iterate each module.**

You are not the only person, and that is why we have enabled a channel in our Slack where you can go asking questions, improvements, advances and everything you need. There will be different Saturdays.AI mentors who will help you prepare for AI Saturdays.

Link to join the Slack channel

This course follows a very practical methodology. We advise you to join the Slack channel to be in contact with us and to solve your doubts.

What are we relying on?

Most online courses are never finished. An MIT study [1] estimated that figure at 96%. In Saturdays.AI we have identified that it is a problem of motivation and sustained lack of attention.

It may take you 15 hours to get the knowledge of this course... or maybe 30. But the motivation is to be able to make the most of AI Saturdays, have the opportunity to be part of this community and not stay out.

Influenced by Jeremy Howard we continue in the top-down approach, which starts in a very practical way to consolidate the theoretical concepts later on. David Perkins, a professor at Harvard, wrote a book [2] about the 7 principles of this approach.

On the other hand, we also rely on several of the principles that Scott H.Young has been writing for many years. And specifically, his book Ultralearning [3] which tries to divide into 9 the phases of what he calls' Ultralearning'

Now yes, Welcome!

---
[1] WITH
[2] Making Learning a whole - David Perkins
[3] Ultralearning

Introduction

You will wonder: What is the best Python tutorial for ML? We intend to create that course, accessible to everyone.

There are a few points that cover, in a nutshell how you can learn Python in depth and master it.

1. Understand **the basics**: First, it's important to understand the complete Python workflow and build a solid foundation in this regard. Having a good understanding of basic syntax, functions, loops, class, and objects is very necessary to move forward. Like any programming language, the fundamentals.
2. **Learn packages and libraries**: There are a lot of libraries available in Python. You need to know them based on your mastery of experience. All have their use cases and functionalities in different domains. For example, Machine Learning-based libraries in Python are ***Pandas***, ***Numpy***, ***Scikit-learn***, *Scipy*, ***Matplotlib*** , etc., data analytics-based libraries are *Bokeh* , *Matplotlib*, *PyPlot*, etc., the UX design includes *PyQt*; web development libraries are *Bottle*, *Django*, ***Flask***, etc,Deep Learning Frameworks are *TensorFlow*, *NLTK* (NLP), *OpenCV* (Computer Vision), ***Keras*** and many more.
3. We will see that the **official Python documentation** is the best to know more about the libraries that are used. It is one of the best resources.
4. Finally,to deepen the **Python ecosystem** : Python has a very friendly interface through Notebooks in Anaconda and Jupyter. Also, the knowledge about PIP - the Python Package Installer and the Python and Anaconda prompt is a great way to move forward to the beginning.

And before we start, we want to give you some tips we've learned with years of practice:

1. On the internet we usually see the following: *You can watch videos instead* of *completing tasks* at the *beginning. And once* you're *comfortable* with *the theory behind ML, you can start* practicing. **We believe that the practice has to be** an **active part from** the **beginning of the learning process.**
2. "***Garbage in garbage out***" Don't underestimate the processing and cleansing of data. It takes up about 80% of the time. And there's no automatic way to do it. You need to merge many tables, use different data sources, remove noise, etc.

3. Don’t just clone projects, but also implement your own architectures and hacks from scratch. Machine Learning (ML) is based on research and practice.

First steps

The course is divided into different modules **in** the form **of** a **guided itinerary.** This way, if you already have some experience in some of them you don't have to pass them. **This course will change to fit the best resources** and feedback from **the community.**

What will you learn?

Module 1: Programming from 0

- Types of data
- Variables
- Flow control
- Cycles
- Data Structures
- Functions

Module 2: First steps in Python

- Python "Hello World"
- Types of data and Variables in Python
- Data Structures in Python
- Inputs
- Statements
- Loops
- Functions
- Errors handling
- Classes & Objects
- File manipulations

Module 3: Basic libraries

- Numpy

- Pandas
- Seaborn
- Jupyter Notebooks

Module 4: Crash Course ML

- Model Workflow
- Basic Concepts
- Model Validation

Now you can jump to the module you need. Cheer up and report to Slack 👏

Are you ready?

Module – 1

Basic Concepts

You need to know that all programming languages share some basic elements that work and are used differently in each language, but that meet the same goal.

These elements are:

- Types of data
- Variables
- Flow control
- Cycles
- Data structures
- Functions

We know that understanding how these elements work not pure fun is, but it is the basis of programming. Remember that programming is nothing more than a tool, a means to achieve some purpose: write a game? make a web application? a mobile application?.
Read on

Basics in 10 minutes 😉

https://www.youtube.com/watch?v=9zOo4JkZgSI&feature=youtu.be

See, the basic elements of programming are really important to understand the language these machines speak.

We're not going to stand much more on this module because the really important thing is that you do your first projects with Python. However, we believe that Codecademy is a very simple platform where you can learn the basics.

- **Learn the keys to programming and write your first lines of code (6h) 👉 Here**

Module – 2

Python **Introduction**

All you need is Python. Python is all you need.

Some programming languages live at the heart of data science. Python is one of those languages. And actually, it would take a long time to explain why.

Let's start with the fact that Python provides great functionality for dealing with math, statistics and scientific functions.

It provides extensive libraries for processing data. Not to mention it is an open source and high level tool!

Most importantly, Python is widely used in the scientific and research communities due to its ease of use, its simple syntax makes it easy to adapt for people who they don't even have an engineering background.

What is Python?

Python is a high-level programming language. It can be easily used with other languages such as C, C++, extending the range of your application to almost all fields of programming. Programming in Python is simpler because selected lines of code are needed compared to other programming languages to get the desired output.

Python interpreters are available for installation on different operating systems, allowing code to run on a wide variety of systems.

For this 2nd module we will establish a fast path and a slower one. You may have a more technical profile and have knowledge in other languages, or maybe you may not...

If you are a beginner:

Read The Informal Intro to Python first in your own Python documentation.

Afterwards, you can continue with this tutorial, one of the most valued by several communities, which covers all the basics:

https://youtu.be/rfscVS0vtbw

We recommend that you try installing *jupyter notebook* through Anaconda, instead of installing the PyCharm IDE. As indicated in the following 4 classes.

- *What is an IDE?*
- *What is Jupyter Notebook?*

You can practice even more with the Codecademy course (if you wish).

If you already know any programming language:

https://youtu.be/N4mEzFDjqtA

- Start this video to get acquainted with operators, data structures and functions in general
- Follow with python in kaggle. It will take you in total about 6-7 hours to complete it.

To finish this module, we recommend that you perform the following test before moving on to the most practical part:

In this 10-minute test of Datacamp, specifically that of "Python **programming**" you will be able to discover in a short time your knowledge regarding the rest of people, strengths and weaknesses. *We ask you to take a screenshot of your result, we will ask you later to see your evolution.*

If you need to **review any concept we recommend** **W3School**. Later we will propose challenges for you to put your knowledge into practice.

RESOURCES

Beginners-Python-Cheat-Sheet.pdf

Test Python

The next step is critical for both profiles, in order to consolidate what you have learned so far:

If you are a beginner:

PracticePython.org - a platform that offers the full spectrum of programming tasks in Python and most importantly, its solutions. Here you can compare your solution with those of others and find the strengths and weaknesses of your approach.

If you already have experience:

Do you dare with these challenges? 👉 Beginner and Intermediate Level

---

In any case, if you need to *review any concept we recommend W3School.*

If you have any questions, we answer for Slack. You can also share your progress with the rest of the community.

Once you have tested your knowledge and skills Do you want to know your progress? 😉

We recommend you retest the 10-minute Test of Datacamp, specifically the "Python programming" test where you can discover in a short time your knowledge of other people, strengths and weaknesses . *We ask you to take a screenshot of your result, we will ask you later to see your evolution.*

Module – 3

Intro to Python for Data Science

Python libraries for Data Science

What are programming libraries? It is a collection of precompiled routines that a program can use. Routines, sometimes called modules, are stored in object format. Libraries are particularly useful for storing frequently used routines because you don't need to explicitly link them to each program that uses them. Libraries save time because you don't need to build functions from scratch.

What do you need?

It is essential that you learn how to manage libraries like NumPy, Pandas, Matplotlib and Seaborn. Installing Anaconda, Data Science environment and Python:

- Linux
- Macos

- Windows

Check out the Python 3 Jupyter Notebook web interpreter we'll use. Familiarize yourself with the libraries and tools that we will use; in recommended order

Libraries

1.First of all, start learning **NumPy,** as it is the fundamental package for scientific computing with Python. A good understanding of Numpy will help you use tools like Pandas more effectively.

- Things to learn: Numpy basics, most frequent operations in Numpy,such as working with N-dimensional arrays, indexing, indexing using integer arrays, transposition of an array, universal functions, data processing using arrays, more frequent statistical methods.

2. Pandas contain high-level data structures and manipulation tools to make data analysis fast. The work with this library is built on NumPy.

- Things to learn series, data frames, remove entries from an axis, work with lost values.

3. **Visuals library** (Seaborn and Matplotlib) - for the visualization of two-dimensional or three-dimensional data. They are complex but useful tools. With Matplotlib you can quickly generate linear charts, histograms and more.

- Things to learn: Create different types of visualizations depending on the message you want to transmit. Learn how to build complex and custom charts based on real data.

Our advice

One of the most common mistakes when learning Python is trying to learn too many things, especially libraries at once. When you try to learn like this, you spend too much time switching between different concepts, getting frustrated and moving on to something else.

So, when you start learning, focus on this process and be patient at every step:

1. Understand the basics of Python
2. Learn Numpy
3. Learn Pandas
4. Learn about data visualization
5. Practice your Data Science skills

If you are a beginner:

If you want something shorter, or perhaps to do a review, we recommend this 4-hour Datacamp "Introduction to Python for Data*Science"* course.

We recommend you try this 10-minute [test] of Datacamp, specifically that of "Data manipulation with Python" where you can discover in a short time your knowledge of other people, strengths and weaknesses . *We ask you to take a screenshot of your result, we will ask you later to see your evolution.*

NumPy

NumPy is the most basic and powerful package for working with data in python.
If you are going to work on data analysis or machine learning projects, then having a solid knowledge of NumPy is almost mandatory.

Because other packages for data analysis (such as pandas) are built on NumPy and the scikit-learn package used to build machine learning applications works strongly with NumPy too.

So, what does NumPy provide?

In the kernel, NumPy provides the excellent ndarray objects, short for n-dimensional arrays.

In an object 'ndarray', also known as 'array', you can store multiple elements of the same data type. It is the facilities around the array object that make NumPy so convenient for performing math and data manipulations.

You may wonder, 'I can store numbers and other objects in a python list itself and do all sorts of calculations and manipulations through understanding lists, search loops, etc. Why do I need a numeric array?

Well, there are very significant advantages of using numerical arrangements over lists.

https://youtu.be/QUT1VHiLmmI

Where is the code to follow the tutorial? 👉 Here

- Tutorial 👉 **http://cs231n.github.io/python-numpy-tutorial/#numpy**
- You want to test your knowledge 👉 **https://www.w3resource.com/python-exercises/numpy/basic/index.php**

Additional resources

- **Visual intro to NumPy** 👉 **Here**

RESOURCES

numpy_basics.pdf

100_Numpy_exercises_with_hint.ipynb

Pandas

https://youtu.be/vmEHCJofslg

Code for the tutorial

👉 https://www.kaggle.com/learn/pandas

Do you want to test your knowledge? 👉 Here

RESOURCES

pandas_basics.pdf

seaborn.pdf

pyplot.ipynb

Test Libraries

And now?

Practice makes the teacher and especially when it comes to data science.

Now the analytical work begins with Python. We believe that the most effective solution at this stage is to participate in the Kaggle contests, a platform that organizes data analysis contests.

Do you want to know your progress?

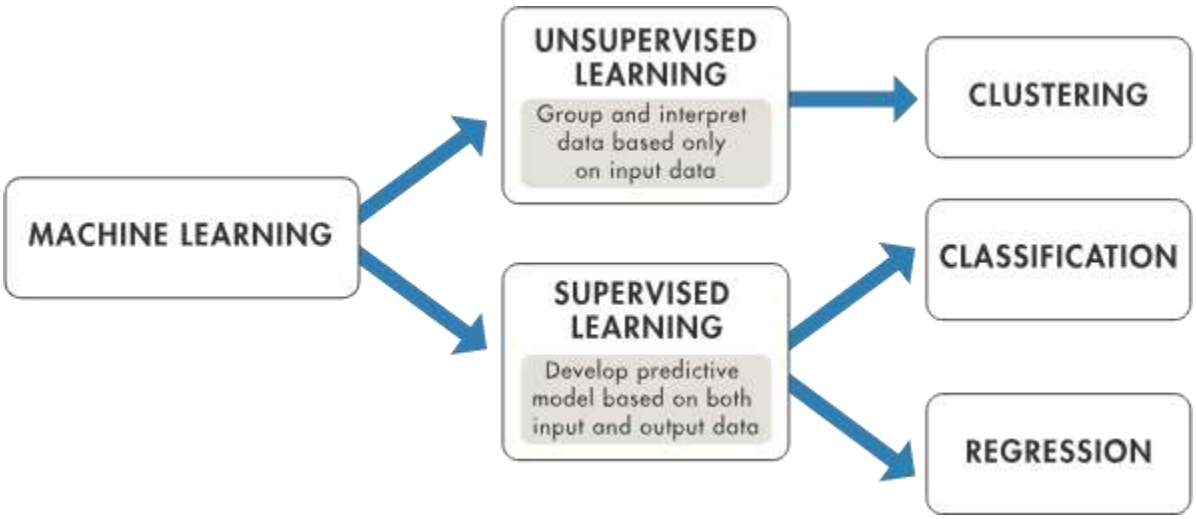
We recommend you retest the 10-minute Test of Datacamp, specifically that of "Data Manipulation with Python" where you can discover in a short time your knowledge of other people, strengths and weak. *We ask you to take a screenshot of your result, we will ask you later to see your evolution.*

Module – 4

ML Crash Course

Machine Learning has been in the spotlight for quite some time. The most important element of its success is purely the interest and motivation in it. And this widespread growth of ML can be seen in almost every other field, from healthcare to the information technology industry.

A brief description of the ML workflow



ML is an experience-based learning. For example, it's like a person who learns to play chess watching as another play. In this way, the machines can be programmed to be trained based on data, acquiring the ability to identify elements or their characteristics with high probability.

First you have to understand the following phases of ML:

- data collection
- data classification
- data analysis
- algorithm development

And to search for patterns, several algorithms are used, which are divided into two large groups:

- Supervised learning
- Unsupervised learning

With **unsupervised** learning, the machine receives only one input dataset. From that, you are in a position to determine the relationship between the data entered and any other hypothetical data. Unlike supervised learning, in which the algorithm receives some test data for learning, unsupervised learning implies that the algorithm itself will find patterns and relationships between different datasets.

Supervised learning involves the computer's ability to recognize elements based on the samples provided. The algorithm studies it and develops the ability to recognize new data based on this data. For example, you can train an algorithm to filter spam messages based on previously received information.

Introduction

https://youtu.be/iAFJUCQNIFI

As you can see, ML concepts can be intimidating for beginners without a well-made introduction:

👉 If you want something shorter https://www.kaggle.com/learn/intro-to-machine-learning

👉 If you want something longer https://developers.google.com/machine-learning/crash-course?hl=es-419

How to start implementing?

Start by participating in some Kaggle 👉 **Guide** challenges