

第04课：为什么要学 Python 以及如何学 Python

### 学了模型，还需要学习编程吗？

对这个问题，****答案是肯定的！****

虽然我们学习的是机器学习原理，但是，并不等于说就可以停留在“原理”层面，彻底 Hands-Off。

恰恰是为了学好原理，我们需要具备最基本的编程能力。原因在于：

在讲述过程中，对于原理细节的展现，经常会以代码形式出现。在这个时候，就算是仅仅为了理解，也得能读得懂代码。

想要真的理解原理，就会涉及动手实践的问题。一般来说，经过自己去训练一个模型，测试一些数据，进一步去编写一个算法等动手的过程，那些公式、函数、算法才能真的在头脑中留下印象。

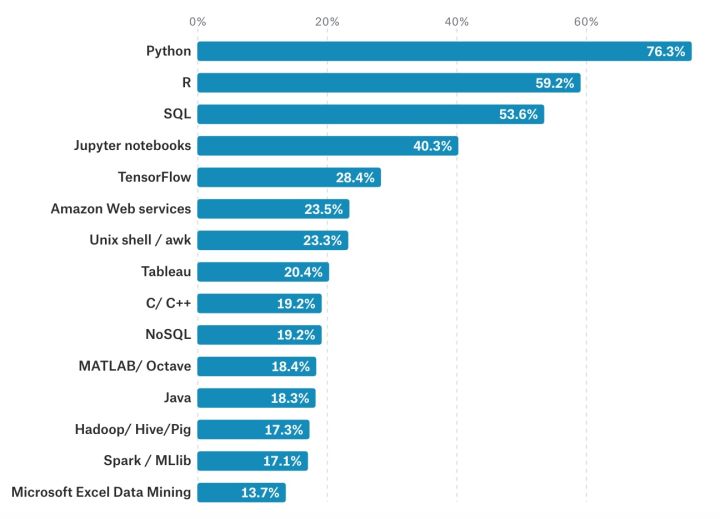
### 学哪种编程语言

既然，非学编程不可，那么，学哪种语言好呢？

如果你在开始学习的时候已经有了编程基础，那么继续使用自己擅长的语言就可以。

如果你还不会编程，或者觉得自己之前学的、用的语言不好，想换一种更适合机器学习的编程语言。那么我的建议是：****Python****。

根据数据平台 Kaggle 发布的****2017年机器学习及数据科学调查报告****，Python 是数据科学家和人工智能从业者使用最多的语言/工具：

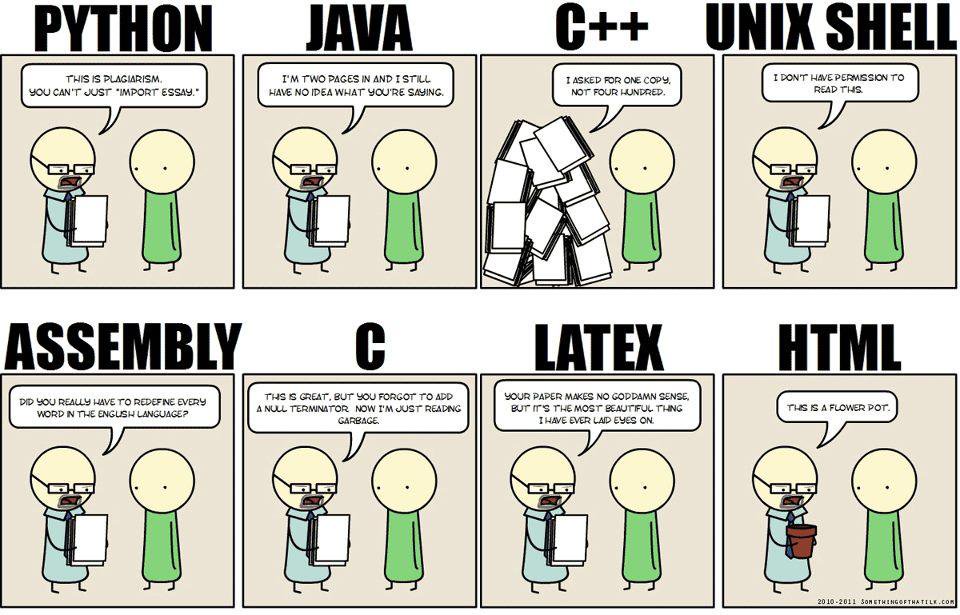


我并不是 Python 粉，做了十几年程序员，工作语言最开始是 C，后来用 Java，现在用 C#。却也不得不承认，在日常工作中，Python 所占的比例越来越大了。

即使不是开发最终提供给用户的产品，做 POC 或 Prototype 会用到 Python；处理数据会用到 Python；验证论文也会用 Python……

Python 在机器学习中拥有这样的地位，和它自身的特性分不开。

#### Python 是一种说人话的语言



所谓“说人话”，是指这种语言：

1. 开发者不需要关注底层； 编写 Python 代码时不用自己费心照顾存储、网络等资源。尤其是，写 AI 相关的程序时，处理大量非结构化数据是必然要求。针对这一场景，Python 的动态类型设计正好大展拳脚；
2. 语法简单直观，毫不啰嗦；
3. 表达形式一致。

Python 的设计理念就是让不同的人在撰写同样功能实现的代码时，所用的表达形式一致。

这样的代码既容易读懂，又容易整合。

而且，Python 程序不用编译，直接运行。甚至都可以不用写文件，一条条语句可以直接作为命令行运行。不要太方便咯。

#### 大量机器学习支持库

由于 Python 语言自身所具备的特性，使得它成了做数据处理、机器学习和深度学习的人们的首选。毕竟这些领域早期的主体是研究人员和数据科学家。他们并不是职业程序员，编程语言本身于他们而言仅仅是一种工具。

他们没有兴趣搞什么“XXX 是最好的编程语言”，也不想陷入语言类型、特征的争论，仅仅是需要一种能够在付出最小学习代价的前提下，尽快将他们的想法实现为代码的载体。

如此一来，Python 就成了他们的掌中宝。作为回报，他们向 Python 反哺了大量用于数据处理和机器学习的支持库。

这就使得 ****Python 在机器学习、深度学习领域拥有了各种语言中堪称最强的公共 AI 支持库****。

其中最著名的 NumPy 和 sklearn（scikit-learn），它们是现在每一个有志于入行 AI 的人都不可能忽略的。

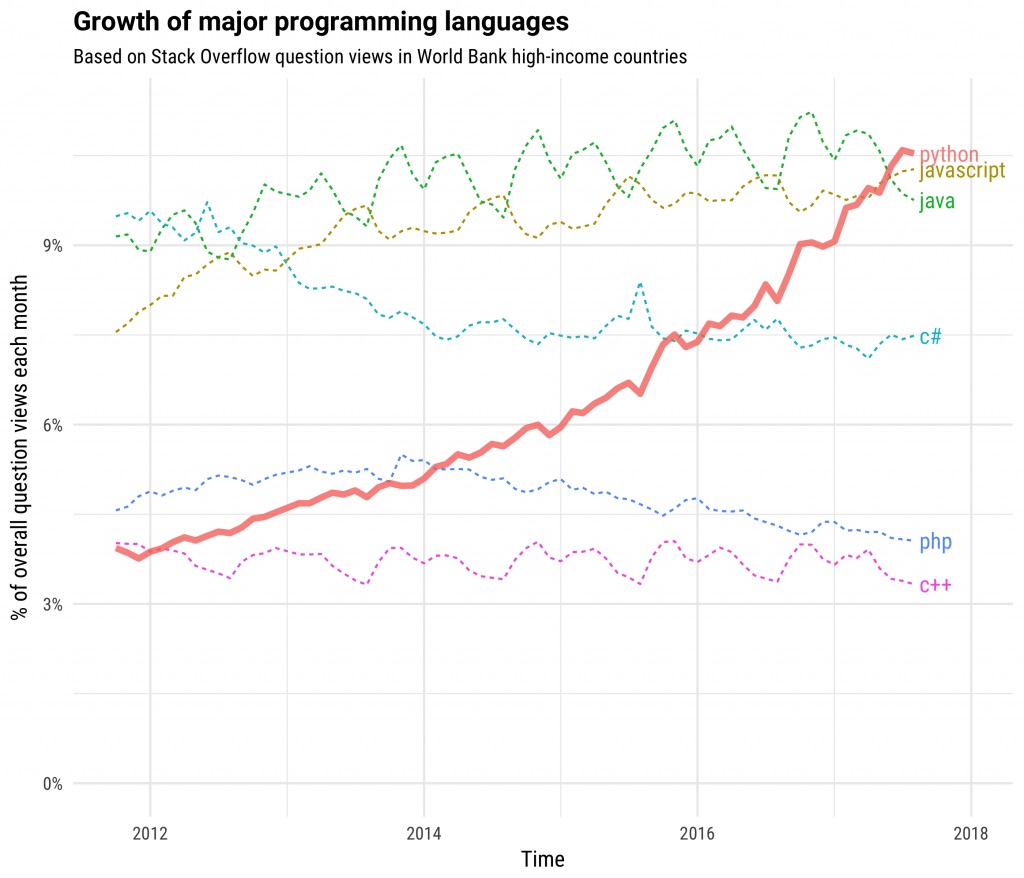
本课所讲述的全部内容，都用这两个库来进行实践。

#### Python 的江湖地位

****语言简单易学****，****支持库丰富强大****，这两大支柱从早期就奠定了 Python 的江湖地位。

不过当然咯，罗马不是一天建成的，凡事都需要积累。虽然 NumPy 的前身在1995年就出现了，不过正式的 NumPy 却是到了2006年才发布。sklearn 则发布于2007年。

根据以****高收入国家 Stack Overflow 问题阅读量****为基础的主要编程语言趋势统计，可以看出，近年来，Python 已然力压 Java 和 Javascript，成为发达国家增长最快的编程语言：



由图可见，2012年之后，对于 Python 相关问题的浏览量迅速增长，从时间上看，这一趋势正好和近几年人工智能的发展重合。

技术的普及推广就像滚雪球，早期的积累相对缓慢，一旦过了临界点，就是大爆发。别的不说，就说现在 TensorFlow、Caffe 之类的深度学习框架，主体都是用 Python 来实现，提供的原生接口也是 Python。

### 怎么学 Python

如果读者完全没有编程实践经验，该如何着手学习呢？是不是应该先买本书？

买书当然可以，不过不是必须，特别是，当你学习编程的目的是要训练机器学习模型的时候。

个人认为，阅读[官网文档](https://www.python.org/doc/)或[中文版教程](http://www.pythondoc.com/pythontutorial3/index.html)，并在实践中通过 Stack Overflow 直接查找所需实现问题的 Solution，可能是比读书更好的学习编程语言的方法。

如果你非要买书，那么推荐这本：由 Eric Matthes 撰写的《Python 编程从入门到实践》。

不管买不买书，下面的过程都差不多。个人建议按照下面的顺序入手 Python。

（1）安装 Python 运行环境。

Python 2 和 Python 3 又是一个悲伤的故事。

关于两者的争论罄竹难书，但毕竟，更高版本总是看着顺眼点。更何况，Python 官方到2020年就不支持 Python 2 了。如今 TensorFlow for Windows 也只有 Python 3 版本。

废话少说，就选 Python 3吧。

第一步，在你的系统上安装好 Python 3。

（2）编写第一个 Python 程序：“Hello world”。

虽然有效代码只有：print（”hello world”）。

但注意，你需要尝试至少三种不同的运行方法：

1. 在命令行直接运行；
2. 编写一个 Python 文件，将 print hello world 封装为一个函数，通过 main 函数调用它来运行；
3. 编写一个 class，将 print hello world 封装为一个 method，通过 main 函数创建 class 实例来运行 method。

（3）编写一个或者多个复杂点的程序，用（2）中第2种或第3种方式执行。

具体程序的功能可以自己定义，目的是借此了解下面这些****概念****：****数据类型、变量、函数、参数、返回值、调用、递归……****

学习****流程控制****：****顺序、条件、循环****。

搞清几种不同类型：****array、list、dict、set**** 的用法。

在这个过程中，学习什么是****静态类型****、什么是****动态类型****、什么是****强类型****、什么是****弱类型****，这些不同设计的目的和应用场景分别是什么。

如果你实在不知道写什么程序好，那就写写 binary search 和 quick sorting 吧。并顺便尝试一下递归和非递归的不同实现。

（4）编写程序练习文件读写，文件和目录操作。

这一点非常重要，对于最初级的机器学习实践者，如果使用支持库封装好的模型算法，那么实际要做的工作其实就是把数据在文件和各种类型的变量之间导来导去。

所以务必学会将 tsv、csv 之类的文件读入 array、list、dict 等结构，以及将这些变量打印到文本文件中的方法。

（5）开始写第一个机器学习程序。

首先 import numpy 和 sklearn。之前当然要安装这两个支持库，一般安装支持库使用 apt-get 或者 pip，可以根据需要选用。

其次，找一个模型，比如 Logistic Regression，网络搜索该模型训练和测试的 Example Code，运行后看结果，并阅读 Example Code（下面是个例子）。

**from** numpy **import** \*

**from** sklearn.datasets **import** load\_iris *# import datasets*

*# load the dataset: iris*

iris = load\_iris()

samples = iris.data

*#print samples*

target = iris.target

*# import the LogisticRegression*

**from** sklearn.linear\_model **import** LogisticRegression

classifier = LogisticRegression() *# 使用类，参数全是默认的*

classifier.fit(samples, target) *# 训练数据来学习，不需要返回值*

x = classifier.predict(array([5, 3, 5, 2.5]).reshape(1,-1)) *# 测试数据，分类返回标记*

**print** x

一则学习 sklearn 和 Numpy 库函数的调用，一方面对 LR 有个感性认识。

（6）自己设置一个实际问题，并准备数据。修改（5）的程序，用自己的数据训练并测试模型。

（7）逐行解读（6）中调用的 sklearn 函数的实现代码，结合该模型的原理，对比印证，彻底搞懂一个算法的实现细则。

比如例子中的：classifier.fit() 和 classifier.predict()。

（8）不依赖 sklearn，自己动手实现一个训练 LR 模型的算法。

完成上面这些步骤后，你算是可以用 Python 辅助学习机器学习基础知识了。

以上就是绪论部分需要学习的内容，不知道同学们通过这两天是否对为什么学习和怎么学习有了更深的理解？请把你的心得和问题通过学习笔记提交出来吧，我们一起研究，共同进步。

[邀请好友一起学，获得 25% 返现奖励](https://gitbook.cn/m/mazi/columns/5bc6ac7442d7d32f50f19a98/topics/5be5477e2c33167c317c028b)

[IMG_260](https://gitbook.cn/m/mazi/comp/column?columnId=5bc6ac7442d7d32f50f19a98%26tag=2#catalog)