Python 可以说是现在最流行的机器学习语言，而且你也能在网上找到大量的资源。你现在也在考虑从 Python 入门机器学习吗？本教程或许能帮你成功上手，从 0 到 1 掌握 Python 机器学习，至于后面再从 1 到 100 变成机器学习专家，就要看你自己的努力了。本教程原文分为两个部分，机器之心在本文中将其进行了整合，原文可参阅：7 Steps to Mastering Machine Learning With Python 和 7 More Steps to Mastering Machine Learning With Python。本教程的作者为 KDnuggets 副主编兼数据科学家 Matthew Mayo。

「开始」往往是最难的，尤其是当选择太多的时候，一个人往往很难下定决定做出选择。本教程的目的是帮助几乎没有 Python 机器学习背景的新手成长为知识渊博的实践者，而且这个过程中仅需要使用免费的材料和资源即可。这个大纲的主要目标是带你了解那些数量繁多的可用资源。毫无疑问，资源确实有很多，但哪些才是最好的呢？哪些是互补的呢？以怎样的顺序学习这些资源才是最合适的呢？

首先，我假设你并不是以下方面的专家：

- 机器学习
- Python
- 任何 Python 的机器学习、科学计算或数据分析库

当然，如果你对前两个主题有一定程度的基本了解就更好了，但那并不是必要的，在早期阶段多花一点点时间了解一下就行了。

基础篇

第一步：基本 Python 技能

如果我们打算利用 Python 来执行机器学习，那么对 Python 有一些基本的了解就是至关重要的。幸运的是，因为 Python 是一种得到了广泛使用的通用编程语言，加上其在科学计算和机器学习领域的应用，所以找到一个初学者教程并不十分困难。你在 Python 和编程上的经验水平对于起步而言是至关重要的。

首先，你需要安装 Python。因为我们后面会用到科学计算和机器学习软件包，所以我建议你安装 Anaconda。这是一个可用于 Linux、OS X 和 Windows 上的工业级的 Python 实现，完整包含了机器学习所需的软件包，包括 numpy、scikit-learn 和 matplotlib。其也

包含了 iPython Notebook，这是一个用在我们许多教程中的交互式环境。我推荐安装 Python 2.7。

如果你不懂编程，我建议你从下面的免费在线书籍开始学习，然后再进入后续的材料：

- Learn Python the Hard Way，作者 Zed A. Shaw：Learn Python the Hard Way

如果你有编程经验，但不懂 Python 或还很初级，我建议你学习下面两个课程：

- 谷歌开发者 Python 课程（强烈推荐视觉学习者学习）：http://suo.im/toMzq
- Python 科学计算入门（来自 UCSB Engineering 的 M. Scott Shell）（一个不错的入门，大约有 60 页）：http://suo.im/2cXycM

如果你要 30 分钟上手 Python 的快速课程，看下面：

- 在 Y 分钟内学会 X（X=Python）：Learn python in Y Minutes

当然，如果你已经是一位经验丰富的 Python 程序员了，这一步就可以跳过了。即便如此，我也建议你常使用 Python 文档：Welcome to Python.org

第二步：机器学习基础技巧

KDnuggets 的 Zachary Lipton 已经指出：现在，人们评价一个「数据科学家」已经有很多不同标准了。这实际上是机器学习领域领域的一个写照，因为数据科学家大部分时间干的事情都牵涉到不同程度地使用机器学习算法。为了有效地创造和获得来自支持向量机的洞见，非常熟悉核方法（kernel methods）是否必要呢？当然不是。就像几乎生活中的所有事情一样，掌握理论的深度是与实践应用相关的。对机器学习算法的深度了解超过了本文探

讨的范围，它通常需要你将非常大量的时间投入到更加学术的课程中去，或者至少是你自己要进行高强度的自学训练。

好消息是，对实践来说，你并不需要获得机器学习博士般的理论理解——就想要成为一个高效的程序员并不必要进行计算机科学理论的学习。

人们对吴恩达在 Coursera 上的机器学习课程内容往往好评如潮；然而，我的建议是浏览前一个学生在线记录的课堂笔记。跳过特定于 Octave（一个类似于 Matlab 的与你 Python 学习无关的语言）的笔记。一定要明白这些都不是官方笔记，但是可以从它们中把握到吴恩达课程材料中相关的内容。当然如果你有时间和兴趣，你现在就可以去 Coursera 上学习吴恩达的机器学习课程：Machine Learning - Stanford University | Coursera

- 吴恩达课程的非官方笔记：Machine Learning - complete course notes

除了上面提到的吴恩达课程，如果你还需要需要其它的，网上还有很多各类课程供你选择。比如我就很喜欢 Tom Mitchell，这里是他最近演讲的视频（一起的还有 Maria-Florina Balcan），非常平易近人。

- Tom Mitchell 的机器学习课程：Machine Learning

目前你不需要所有的笔记和视频。一个有效地方法是当你觉得合适时，直接去看下面特定的练习题，参考上述备注和视频恰当的部分，

第三步：科学计算 Python 软件包概述

好了，我们已经掌握了 Python 编程并对机器学习有了一定的了解。而在 Python 之外，还有一些常用于执行实际机器学习的开源软件库。广义上讲，有很多所谓的科学 Python 库（scientific Python libraries）可用于执行基本的机器学习任务（这方面的判断肯定有些主观性）：

- numpy——主要对其 N 维数组对象有用 NumPy - NumPy
- pandas——Python 数据分析库，包括数据框架（dataframes）等结构 Python Data Analysis Library
- matplotlib——一个 2D 绘图库，可产生出版物质量的图表 Python plotting - Matplotlib 2.0.0 documentation
- scikit-learn——用于数据分析和数据挖掘人物的机器学习算法 scikit-learn: machine learning in Python

学习这些库的一个好方法是学习下面的材料：

- Scipy Lecture Notes，来自 Gaël Varoquaux、Emmanuelle Gouillart 和 Olav Vahtras：Scipy Lecture Notes

- 这个 pandas 教程也很不错：10 Minutes to Pandas：10 Minutes to pandas

在本教程的后面你还会看到一些其它的软件包，比如基于 matplotlib 的数据可视化库 Seaborn。前面提到的软件包只是 Python 机器学习中常用的一些核心库的一部分，但是理解它们应该能让你在后面遇到其它软件包时不至于感到困惑。

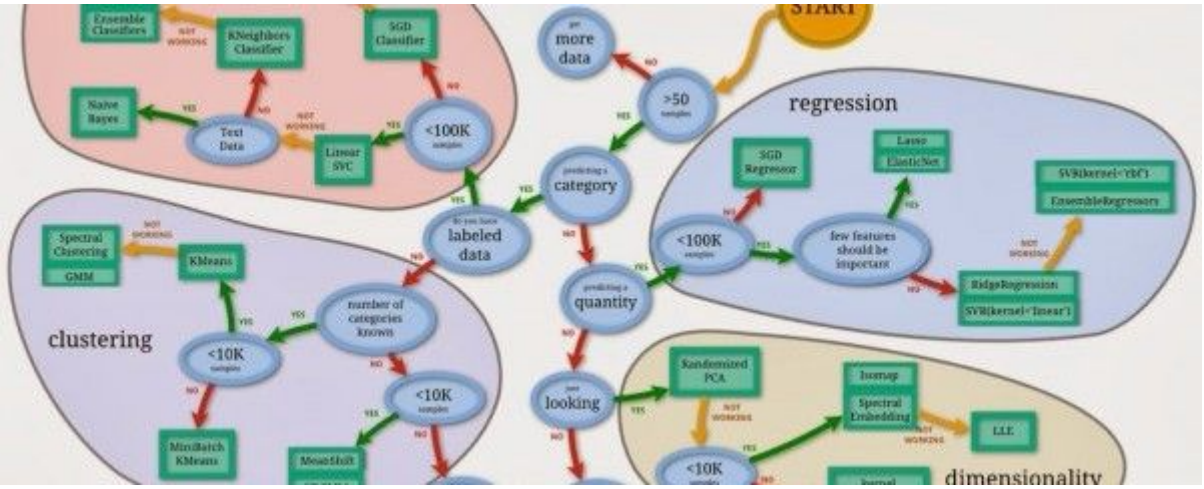
下面就开始动手吧！

第四步：使用 Python 学习机器学习

首先检查一下准备情况

- Python：就绪
- 机器学习基本材料：就绪
- Numpy：就绪
- Pandas：就绪
- Matplotlib：就绪

现在是时候使用 Python 机器学习标准库 scikit-learn 来实现机器学习算法了。



scikit-learn 流程图

下面许多的教程和训练都是使用 iPython (Jupyter) Notebook 完成的，iPython Notebook 是执行 Python 语句的交互式环境。iPython Notebook 可以很方便地在网上找到或下载到你的本地计算机。

- 来自斯坦福的 iPython Notebook 概览：IPython Tutorial

同样也请注意，以下的教程是由一系列在线资源所组成。如果你感觉课程有什么不合适的，可以和作者交流。我们第一个教程就是从 scikit-learn 开始的，我建议你们在继续完成教程前可以按顺序看一看以下的文章。

下面是一篇是对 scikit-learn 简介的文章，scikit-learn 是 Python 最常用的通用机器学习库，其覆盖了 K 近邻算法：

- Jake VanderPlas 写的 scikit-learn 简介：Jupyter Notebook Viewer

下面的会更加深入、扩展的一篇简介，包括了从著名的数据库开始完成一个项目：

- Randal Olson 的机器学习案例笔记：Jupyter Notebook Viewer

下一篇关注于在 scikit-learn 上评估不同模型的策略，包括训练集/测试集的分割方法：

- Kevin Markham 的模型评估：justmarkham/scikit-learn-videos

第五步：Python 上实现机器学习的基本算法

在有了 scikit-learn 的基本知识后，我们可以进一步探索那些更加通用和实用的算法。我们从非常出名的 K 均值聚类（k-means clustering）算法开始，它是一种非常简单和高效的方法，能很好地解决非监督学习问题：

- K-均值聚类：jakevdp/sklearn_pycon2015

接下来我们可以回到分类问题，并学习曾经最流行的分类算法：

- 决策树：Tutorial: Decision Trees

在了解分类问题后，我们可以继续看看连续型数值预测：

- 线性回归：Jupyter Notebook Viewer

我们也可以利用回归的思想应用到分类问题中，即 logistic 回归：

- logistic 回归：justmarkham/gadsdc

第六步：Python 上实现进阶机器学习算法

我们已经熟悉了 scikit-learn，现在我们可以了解一下更高级的算法了。首先就是支持向量机，它是一种依赖于将数据转换映射到高维空间的非线性分类器。

- 支持向量机：jakevdp/sklearn_pycon2015

随后，我们可以通过 Kaggle Titanic 竞赛检查学习作为集成分类器的随机森林：

- Kaggle Titanic 竞赛（使用随机森林）：Jupyter Notebook Viewer

降维算法经常用于减少在问题中所使用的变量。主成份分析法就是非监督降维算法的一个特殊形式：

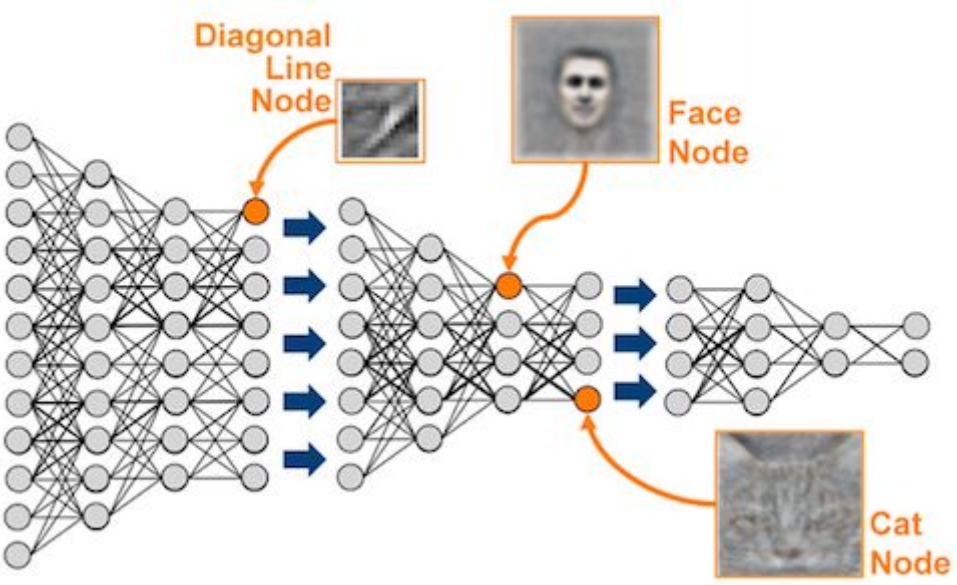
- 降维算法：jakevdp/sklearn_pycon2015

在进入第七步之前，我们可以花一点时间考虑在相对较短的时间内取得的一些进展。

首先使用 Python 及其机器学习库，我们不仅已经了解了一些最常见和知名的机器学习算法（k 近邻、k 均值聚类、支持向量机等），还研究了强大的集成技术（随机森林）和一些额外的机器学习任务（降维算法和模型验证技术）。除了一些基本的机器学习技巧，我们已经开始寻找一些有用的工具包。

我们会进一步学习新的必要工具。

第七步：Python 深度学习



神经网络包含很多层

深度学习无处不在。深度学习建立在几十年前的神经网络的基础上，但是最近的进步始于几年前，并极大地提高了深度神经网络的认知能力，引起了人们的广泛兴趣。如果你对神经网络还不熟悉，KDnuggets 有很多文章详细介绍了最近深度学习大量的创新、成就和赞许。

最后一步并不打算把所有类型的深度学习评论一遍，而是在 2 个先进的当代 Python 深度学习库中探究几个简单的网络实现。对于有兴趣深挖深度学习的读者，我建议从下面这些免费的在线书籍开始：

- 神经网络与深度学习，作者 Michael Nielsen：Neural networks and deep learning

1.Theano

链接：Welcome - Theano 0.8.2 documentation

Theano 是我们讲到的第一个 Python 深度学习库。看看 Theano 作者怎么说：

> *Theano 是一个 Python 库，它可以使你有效地定义、优化和评估包含多维数组的数学表达式。*

下面关于运用 Theano 学习深度学习的入门教程有点长，但是足够好，描述生动，评价很高：

- Theano 深度学习教程，作者 Colin Raffel：Jupyter Notebook Viewer

2.Caffe

链接：Caffe | Deep Learning Framework

另一个我们将测试驱动的库是 Caffe。再一次，让我们从作者开始：

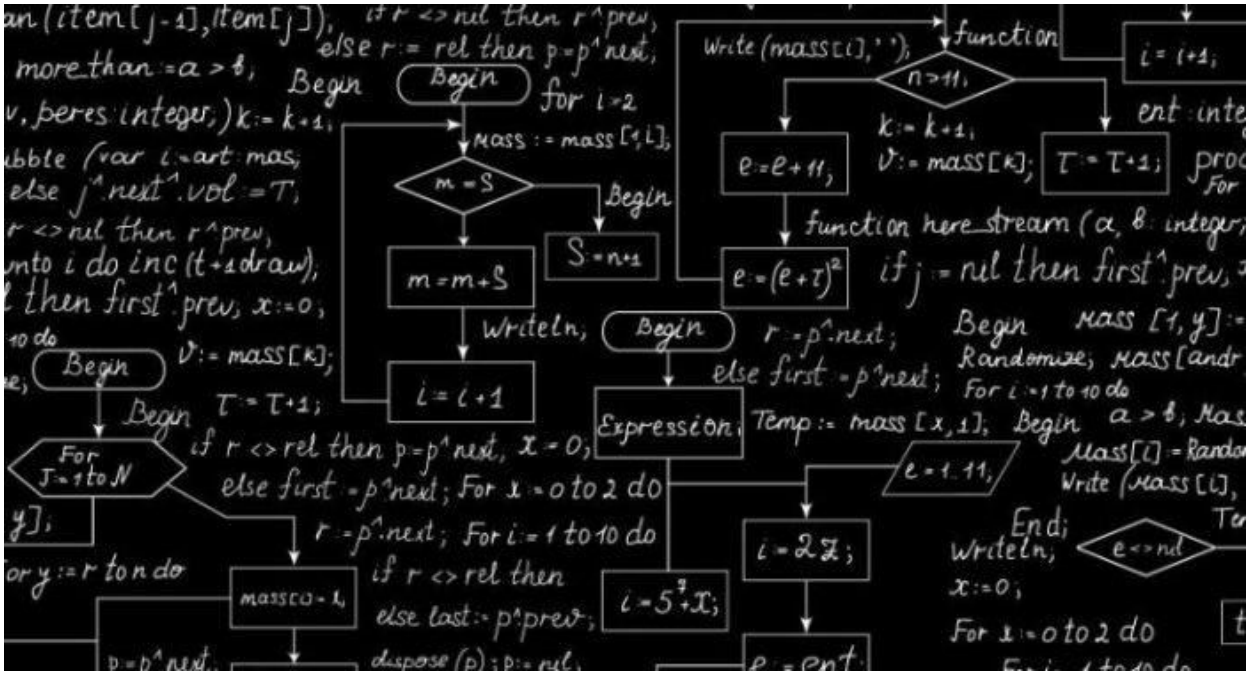
> *Caffe 是一个深度学习框架，由表达、速度和模块性建构，Bwekeley 视觉与学习中心和社区工作者共同开发了 Caf fe。*

这个教程是本篇文章中最好的一个。我们已经学习了上面几个有趣的样例，但没有一个可与下面这个样例相竞争，其可通过 Caffe 实现谷歌的 DeepDream。这个相当精彩！掌握教程之后，可以尝试使你的处理器自如运行，就当作是娱乐。

- 通过 Caffe 实现谷歌 DeepDream：google/deepdream

我并没有保证说这会很快或容易，但是如果你投入了时间并完成了上面的 7 个步骤，你将在理解大量机器学习算法以及通过流行的库（包括一些在目前深度学习研究领域最前沿的库）在 Python 中实现算法方面变得很擅长。

进阶篇



机器学习算法

本篇是使用 Python 掌握机器学习的 7 个步骤系列文章的下篇，如果你已经学习了该系列的上篇，那么应该达到了令人满意的学习速度和熟练技能；如果没有的话，你也许应该回顾一下上篇，具体花费多少时间，取决于你当前的理解水平。我保证这样做是值得的。快速回顾之后，本篇文章会更明确地集中于几个机器学习相关的任务集上。由于安全地跳过了一些基础模块——Python 基础、机器学习基础等等——我们可以直接进入到不同的机器学习算法之中。这次我们可以根据功能更好地分类教程。

第1步：机器学习基础回顾&一个新视角

上篇中包括以下几步：

1. Python 基础技能

2. 机器学习基础技能

3. Python 包概述

4. 运用 Python 开始机器学习：介绍&模型评估

5. 关于 Python 的机器学习主题：k-均值聚类、决策树、线性回归&逻辑回归

6. 关于 Python 的高阶机器学习主题：支持向量机、随机森林、PCA 降维

7. Python 中的深度学习

如上所述，如果你正准备从头开始，我建议你按顺序读完上篇。我也会列出所有适合新手的入门材料，安装说明包含在上篇文章中。

然而，如果你已经读过，我会从下面最基础的开始：

- 机器学习关键术语解释，作者 Matthew Mayo。地址：Machine Learning Key Terms, Explained
- 维基百科条目：统计学分类。地址：Statistical classification
- 机器学习：一个完整而详细的概述，作者 Alex Castrounis。地址：Machine Learning: A Complete and Detailed Overview

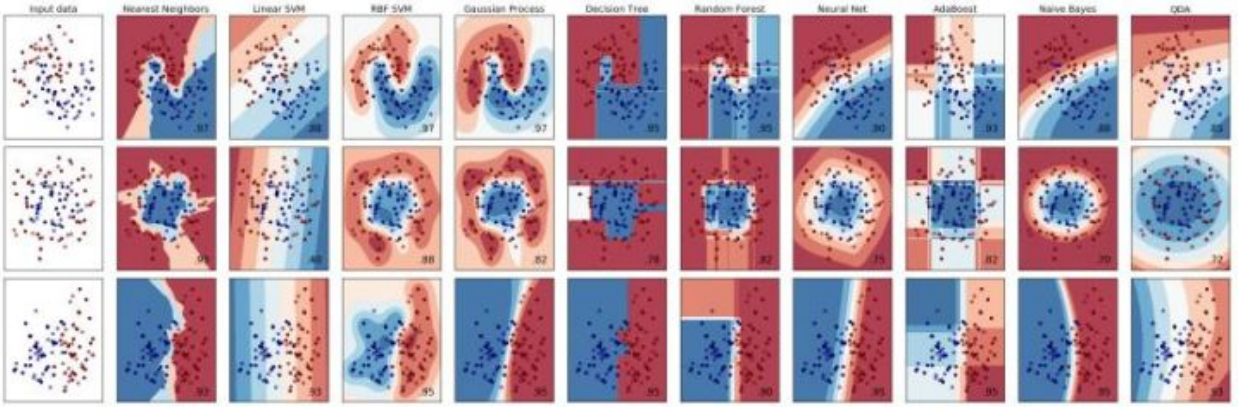
如果你正在寻找学习机器学习基础的替代或补充性方法，恰好我可以把正在看的 Shai Ben-David 的视频讲座和 Shai Shalev-Shwartz 的教科书推荐给你：

- Shai Ben-David 的机器学习介绍视频讲座，滑铁卢大学。地址：http://suo.im/1TFlK6
- 理解机器学习：从理论到算法，作者 Shai Ben-David & Shai Shalev-Shwartz。地址：http://suo.im/1NL0ix

记住，这些介绍性资料并不需要全部看完才能开始我写的系列文章。视频讲座、教科书及其他资源可在以下情况查阅：当使用机器学习算法实现模型时或者当合适的概念被实际应用在后续步骤之中时。具体情况自己判断。

第2步：更多的分类

我们从新材料开始，首先巩固一下我们的分类技术并引入一些额外的算法。虽然本篇文章的第一部分涵盖决策树、支持向量机、逻辑回归以及合成分类随机森林，我们还是会添加 k-最近邻、朴素贝叶斯分类器和多层感知器。



Scikit-learn 分类器

k-最近邻（kNN）是一个简单分类器和懒惰学习者的示例，其中所有计算都发生在分类时间上（而不是提前在训练步骤期间发生）。kNN 是非参数的，通过比较数据实例和 k 最近实例来决定如何分类。

- 使用 Python 进行 k-最近邻分类。地址：K-Nearest Neighbor classification using python

朴素贝叶斯是基于贝叶斯定理的分类器。它假定特征之间存在独立性，并且一个类中任何特定特征的存在与任何其它特征在同一类中的存在无关。

- 使用 Scikit-learn 进行文档分类，作者 Zac Stewart。地址：Document Classification with scikit-learn

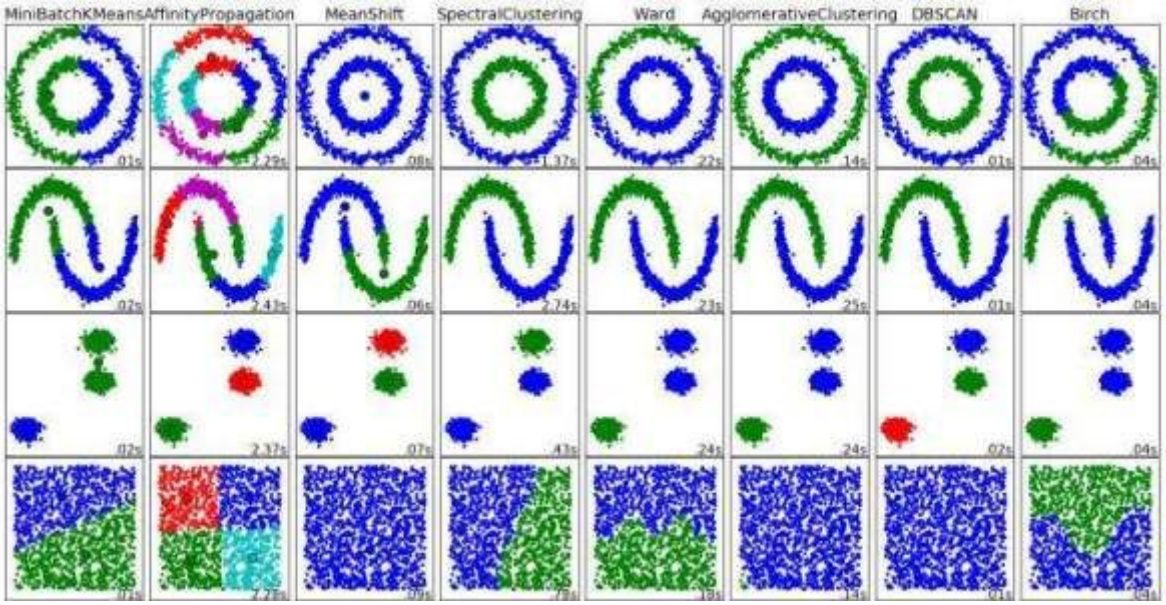
多层感知器（MLP）是一个简单的前馈神经网络，由多层节点组成，其中每个层与随后的层完全连接。多层感知器在 Scikit-learn 版本 0.18 中作了介绍。

首先从 Scikit-learn 文档中阅读 MLP 分类器的概述，然后使用教程练习实现。

- 神经网络模型（监督式），Scikit-learn 文档。地址：1.17. Neural network models (supervised)
- Python 和 Scikit-learn 的神经网络初学者指南 0.18！作者 Jose Portilla。地址：A Beginner’ s Guide to Neural Networks with Python and SciKit Learn 0.18!

第3步：更多聚类

我们现在接着讲聚类，一种无监督学习形式。上篇中，我们讨论了 k-means 算法; 我们在此介绍 DBSCAN 和期望最大化（EM）。



Scikit-learn聚类算法

首先，阅读这些介绍性文章; 第一个是 k 均值和 EM 聚类技术的快速比较，是对新聚类形式的一个很好的继续，第二个是对 Scikit-learn 中可用的聚类技术的概述：

- 聚类技术比较：简明技术概述，作者 Matthew Mayo。地址：Comparing Clustering Techniques: A Concise Technical Overview

- 在玩具数据集中比较不同的聚类算法，Scikit-learn 文档。地址：Comparing different clustering algorithms on toy datasets

期望最大化（EM）是概率聚类算法，并因此涉及确定实例属于特定聚类的概率。EM 接近统计模型中参数的最大似然性或最大后验估计（Han、Kamber 和 Pei）。EM 过程从一组参数开始迭代直到相对于 k 聚类的聚类最大化。

首先阅读关于 EM 算法的教程。接下来，看看相关的 Scikit-learn 文档。最后，按照教程使用 Python 自己实现 EM 聚类。

- 期望最大化（EM）算法教程，作者 Elena Sharova。地址：A Tutorial on the Expectation Maximization (EM) Algorithm
- 高斯混合模型，Scikit-learn 文档。地址：2.1. Gaussian mixture models。
- 使用 Python 构建高斯混合模型的快速介绍，作者 Tiago Ramalho。地址：Quick introduction to gaussian mixture models with python

如果高斯混合模型初看起来令人困惑，那么来自 Scikit-learn 文档的这一相关部分应该可以减轻任何多余的担心：

> *高斯混合对象实现期望最大化（EM）算法以拟合高斯模型混合。*

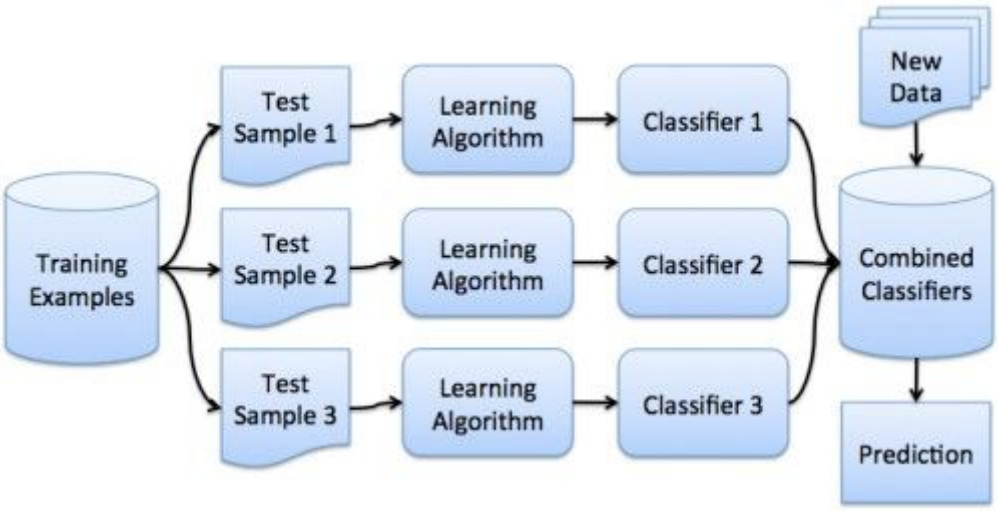
基于密度且具有噪声的空间聚类应用（DBSCAN）通过将密集数据点分组在一起，并将低密度数据点指定为异常值来进行操作。

首先从 Scikit-learn 的文档中阅读并遵循 DBSCAN 的示例实现，然后按照简明的教程学习：

- DBSCAN 聚类算法演示，Scikit-learn 文档。地址：Demo of DBSCAN clustering algorithm
- 基于密度的聚类算法（DBSCAN）和实现。地址：http://suo.im/1LEoXC

第4步：更多的集成方法

上篇只涉及一个单一的集成方法：随机森林（RF）。RF 作为一个顶级的分类器，在过去几年中取得了巨大的成功，但它肯定不是唯一的集成分类器。我们将看看包装、提升和投票。



给我一个提升

首先，阅读这些集成学习器的概述，第一个是通用性的；第二个是它们与 Scikit-learn 有关：

- 集成学习器介绍，作者 Matthew Mayo。地址：Data Science Basics: An Introduction to Ensemble Learners
- Scikit-learn 中的集成方法，Scikit-learn 文档。地址：1.11. Ensemble methods

然后，在继续使用新的集成方法之前，请通过一个新的教程快速学习随机森林：

- Python 中的随机森林，来自 Yhat。地址：Random Forests in Python

包装、提升和投票都是不同形式的集成分类器，全部涉及建构多个模型; 然而，这些模型由什么算法构建，模型使用的数据，以及结果如何最终组合起来，这些都会随着方案而变化。

- 包装：从同一分类算法构建多个模型，同时使用来自训练集的不同（独立）数据样本——Scikit-learn 实现包装分类器
- 提升：从同一分类算法构建多个模型，一个接一个地链接模型，以提高每个后续模型的学习——Scikit-learn 实现 AdaBoost
- 投票：构建来自不同分类算法的多个模型，并且使用标准来确定模型如何最好地组合——Scikit-learn 实现投票分类器

那么，为什么要组合模型？为了从一个特定角度处理这个问题，这里是偏差-方差权衡的概述，具体涉及到提升，以下是 Scikit-learn 文档：

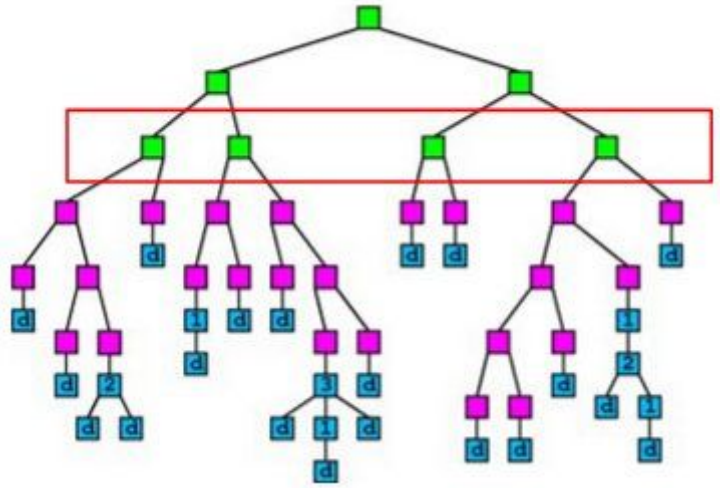
- 单一评估器 vs 包装：偏差-方差分解，Scikit-learn 文档。地址：http://suo.im/3izlRB

现在你已经阅读了关于集成学习器的一些介绍性材料，并且对几个特定的集成分类器有了基本了解，下面介绍如何从 Machine Learning Mastery 中使用 Scikit-learn 在 Python 中实现集成分类器：

- 使用 Scikit-learn 在 Python 中实现集成机器学习算法，作者 Jason Brownlee。地址：Ensemble Machine Learning Algorithms in Python with scikit-learn - Machine Learning Mastery

第5步：梯度提升

下一步我们继续学习集成分类器，探讨一个当代最流行的机器学习算法。梯度提升最近在机器学习中产生了显著的影响，成为了 Kaggle 竞赛中最受欢迎和成功的算法之一。



给我一个梯度提升

首先，阅读梯度提升的概述：

- 维基百科条目：梯度提升。地址：http://suo.im/TslWi

接下来，了解为什么梯度提升是 Kaggle 竞赛中「最制胜」的方法：

- 为什么梯度提升完美解决了诸多 Kaggle 难题？Quora，地址：Why does Gradient boosting work so well for so many Kaggle problems?
- Kaggle 大师解释什么是梯度提升，作者 Ben Gorman。地址：A Kaggle Master Explains Gradient Boosting

虽然 Scikit-learn 有自己的梯度提升实现，我们将稍作改变，使用 XGBoost 库，我们提到过这是一个更快的实现。

以下链接提供了 XGBoost 库的一些额外信息，以及梯度提升（出于必要）：

- 维基百科条目：XGBoost。地址：Xgboost - Wikipedia
- Ghub 上的 XGBoost 库。地址：dmlc/xgboost
- XGBoost 文档。地址：Introduction to Boosted Trees

现在，按照这个教程把所有汇聚起来：

- Python 中 XGBoost 梯度提升树的实现指南，作者 Jesse Steinweg-Woods。地址：A Guide to Gradient Boosted Trees with XGBoost in Python

你还可以按照这些更简洁的示例进行强化：

- XGBoost 在 Kaggle 上的示例（Python）。地址：Titanic: Machine Learning from Disaster
- Iris 数据集和 XGBoost 简单教程，作者 Ieva Zarina。地址：Iris Dataset and Xgboost Simple Tutorial

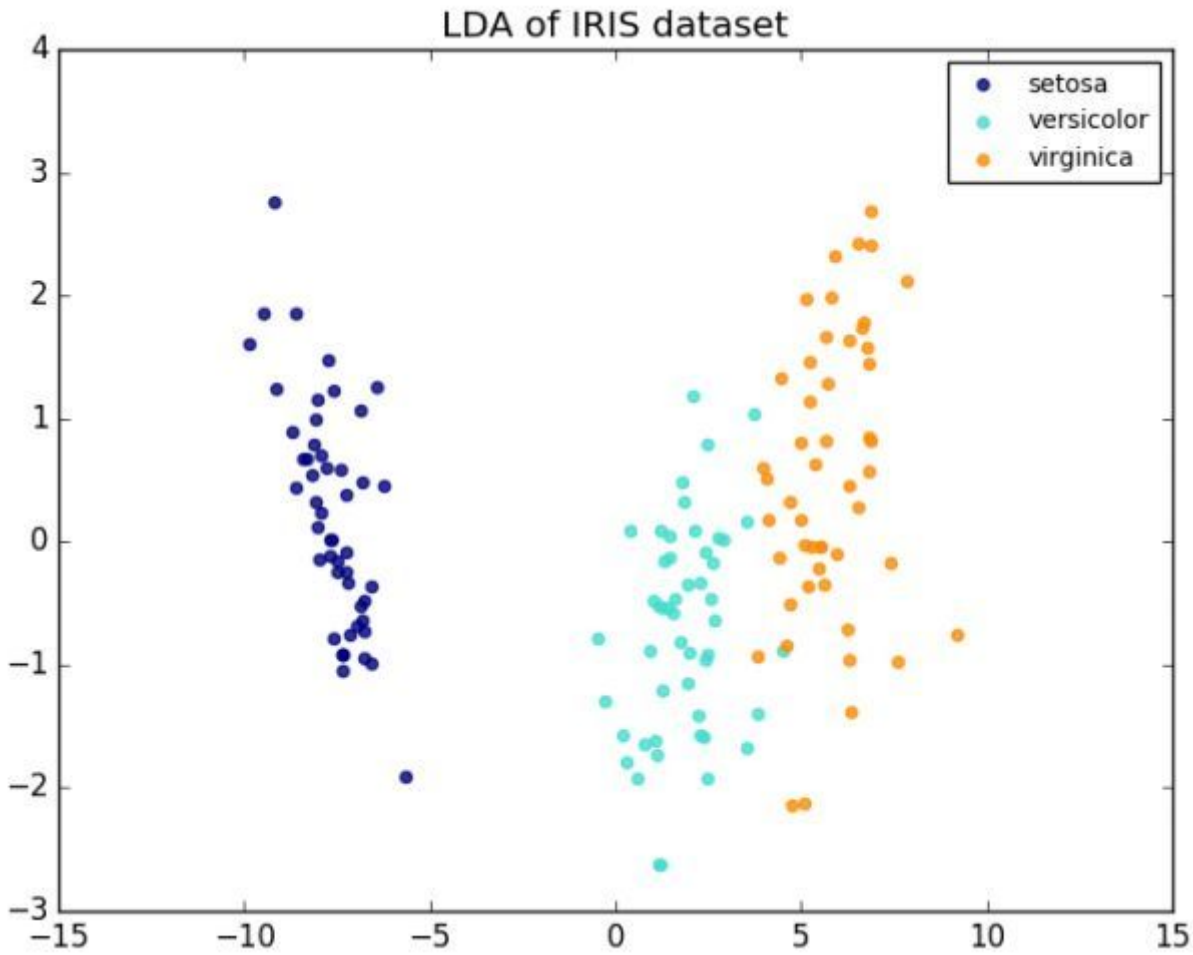
第6步：更多的降维

降维是通过使用过程来获得一组主变量，将用于模型构建的变量从其初始数减少到一个减少数。

有两种主要形式的降维：

- 1. 特征选择——选择相关特征的子集。地址：Feature selection
- 2. 特征提取——构建一个信息性和非冗余的衍生值特征集。地址：Feature extraction

下面是一对常用的特征提取方法。



主成分分析（PCA）是一种统计步骤，它使用正交变换将可能相关变量的一组观测值转换为一组称为主成分的线性不相关变量值。主成分的数量小于或等于原始变量的数量。这种变换以这样的方式定义，即第一主成分具有最大可能的方差（即考虑数据中尽可能多的变率）

以上定义来自 PCA 维基百科条目，如果感兴趣可进一步阅读。但是，下面的概述/教程非常彻底：

- 主成分分析：3 个简单的步骤，作者 Sebastian Raschka。地址：Principal Component Analysis

线性判别分析（LDA）是 Fisher 线性判别的泛化，是统计学、模式识别和机器学习中使用的一种方法，用于发现线性组合特征或分离两个或多个类别的对象或事件的特征。所得到的组合可以用作线性分类器，或者更常见地，用作后续分类之前的降维。

LDA 与方差分析（ANOVA）和回归分析密切相关，它同样尝试将一个因变量表示为其他特征或测量的线性组合。然而，ANOVA 使用分类独立变量和连续因变量，而判别分析具有连续的独立变量和分类依赖变量（即类标签）。

上面的定义也来自维基百科。下面是完整的阅读：

- 线性判别分析——直至比特，作者 Sebastian Raschka。地址：Linear Discriminant Analysis

你对 PCA 和 LDA 对于降维的实际差异是否感到困惑？Sebastian Raschka 做了如下澄清：

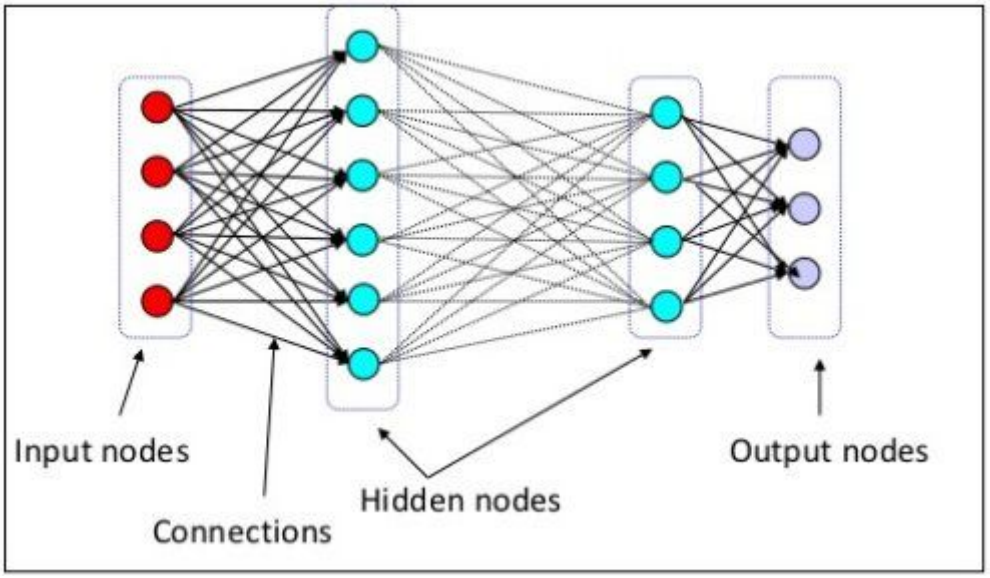
> *线性判别分析（LDA）和主成分分析（PCA）都是通常用于降维的线性转换技术。PCA 可以被描述为「无监督」算法，因为它「忽略」类标签，并且其目标是找到使数据集中的方差最大化的方向（所谓的主成分）。与 PCA 相反，LDA 是「监督的」并且计算表示使多个类之间的间隔最大化的轴的方向（「线性判别式」）。*

有关这方面的简要说明，请阅读以下内容：

- LDA 和 PCA 之间的降维有什么区别？作者 Sebastian Raschka。地址：Machine Learning FAQ

第 7 步：更多的深度学习

上篇中提供了一个学习神经网络和深度学习的入口。如果你的学习到目前比较顺利并希望巩固对神经网络的理解，并练习实现几个常见的神经网络模型，那么请继续往下看。



首先，看一些深度学习基础材料：

- 深度学习关键术语及解释，作者 Matthew Mayo

- 理解深度学习的 7 个步骤，作者 Matthew Mayo。地址：7 Steps to Understanding Deep Learning

接下来，在 Google 的机器智能开源软件库 TensorFlow（一个有效的深度学习框架和现今几乎是最好的神经网络工具）尝试一些简明的概述 / 教程：

- 机器学习敲门砖：任何人都能看懂的 TensorFlow 介绍（第 1、2 部分）
- 入门级解读：小白也能看懂的 TensorFlow 介绍（第 3、4 部分）

最后，直接从 TensorFlow 网站试用这些教程，它实现了一些最流行和常见的神经网络模型：

- 循环神经网络，谷歌 TensorFlow 教程。地址：http://suo.im/2gtkze
- 卷积神经网络，谷歌 TensorFlow 教程。地址：http://suo.im/g8Lbg

此外，目前一篇关于 7 个步骤掌握深度学习的文章正在写作之中，重点介绍使用位于 TensorFlow 顶部的高级 API，以增模型实现的容易性和灵活性。我也将在完成后在这儿添加一个链接。

相关的：

- 进入机器学习行业之前应该阅读的 5 本电子书。地址：5 EBooks to Read Before Getting into A Machine Learning Career
- 理解深度学习的 7 个步骤。地址：7 Steps to Understanding Deep Learning
- 机器学习关键术语及解释。地址：Machine Learning Key Terms, Explained

选自kdnuggets（1）（2） **机器之心编译**

只需十四步：从零开始掌握Python机器学习