Broadview
www.broadview.com.cn

人生苦短，我用Python。
助力数据分析工程师、机器学习工程师快速成长！

跟老齐学Python数据分析

齐伟 编著

“跟老齐学Python”系列后续，基于新版本
详解与数据分析、机器学习相关的Python库的应用
提高读者Python综合应用能力

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

跟老齐学

Python

数据分析

齐伟 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

读者在本书中可以学习到与数据分析、机器学习相关的 Python 库的应用，并通过各种类型的应用示例将所学基本知识进行综合应用。

本书依然秉承“跟老齐学 Python”系列书的写作风格，力争以通俗易懂的内容与读者分享笔者的心得。虽然数据分析强调的是严谨的科学性和缜密的逻辑性，但本书并不会因为顾此特点而变得枯燥。

本书可作为数据分析工程师、机器学习工程师的入门教程。

图书在版编目（CIP）数据

跟老齐学 Python：数据分析 / 齐伟编著. —北京：电子工业出版社，2018.6
ISBN 978-7-121-34003-1

Ⅰ. ①跟… Ⅱ. ①齐… Ⅲ. ①软件工具—程序设计 Ⅳ. ①TP311.561

中国版本图书馆 CIP 数据核字（2018）第 070382 号

策划编辑：高洪霞
责任编辑：牛 勇
印　　刷：三河市良远印务有限公司
装　　订：三河市良远印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×1092　1/16　印张：20.75　字数：541 千字
版　　次：2018 年 6 月第 1 版
印　　次：2018 年 6 月第 1 次印刷
定　　价：79.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：（010）51260888-819，faq@phei.com.cn。

序

认真阅读序言，是读书的好习惯。

“跟老齐学 Python”系列已经出版了三本书，第一本是 Python 入门教程——《跟老齐学 Python：轻松入门》，第二本是 Web 开发教程——《跟老齐学 Python：Django 实战》，本书是第三本。在阅读本书之前，需要读者完成《跟老齐学 Python：轻松入门》的学习或者具有相当程度的知识。

我遇到的学生、软件工程师、大学教师等，他们以各种理由来说明在数据分析、机器学习中应该选择哪种语言或者什么工具。在实际工作中，也的确是百花齐放、百家争鸣。那么到底学什么呢？

这是一个令人苦恼的问题，也是一个浪费时间的问题。

幸亏，现实给出了一个统计性的答案：Python 已经胜出。

所以，本书就呈现在读者眼前了。

本书的目标如下。

- 数据分析和机器学习的入门读物。凡是有志于在此领域工作的读者，通过阅读本书，能够跨进该领域，为日后工作奠定基础。
- 在示例中学习。本书在适当时机，向读者提供各种类型的示例（因为面向对象中的“实例”有特别含义，所以本书中用“示例”表示“某知识技能的应用举例”）。
- 为读者展示一种学习方法。这也是我在前面两本书中所贯彻的核心思想，本书继承这个思想。

本书冠名“数据分析”，是因为绝大部分内容介绍了数据分析的知识和应用，读者学习完这些内容，即可从事相关的工作。在本书的最后一章，也以示例的方式简要介绍了机器学习中的一点内容，主要目的是“开个天窗”，让具有数据分析知识的读者能够看到更广阔的天空。当然，也夹带了私活，就是预告系列丛书的下一本“机器学习”。

跟本书有关的网址如下。

- 代码仓库：https://github.com/qiwsir/DataAnalysis
- 网站：http://itdiffer.com/

如果本书能够成为读者进入数据分析、机器学习领域的垫脚石，我当荣幸之至。

这本书的编写完全是在业余时间完成的，所幸有妻子相助，感谢我的妻子，她为我的写作提供了很多帮助，除日常生活外，还协助我查询和翻译一些资料，通读了全书内容，修正了很多语言表达方面的错误，比如语法错误、错别字等。

另外，还要感谢本书的编辑朋友们，正是有了她们细致、耐心的工作，才能够让本书呈现在读者面前。

齐　伟

2018 年 3 月

目录

第0章

数据分析概述

近几年，数据越来越受到重视了，各种跟数据有关的概念不断被“热炒”。但是，当嘴里不断说出带有“数据”的句子时，是不是也同时明晰了其真正含义？我们是人云亦云，还是基于对其深入研究后的理智思考？

从现在开始，我们共同学习数据分析，厘清各种概念，掌握应用方法，并且透过数据“把这纷扰看个清清楚楚明明白白真真切切”。

0.1 与数据相关的概念

在任何一个学科门类中，概念都非常重要。通过概念能够准确地说明所指对象，概念是我们准确表达所不可或缺的。数据分析及其相关领域正在蓬勃发展，各种新概念层出不穷。为了让读者不至于在面对纷繁多变的概念时惊慌失措，本节将数据分析及其相关领域的几个主要概念罗列出来，请读者阅读，特别是不愿意被名词忽悠的读者，一定要认真阅读这些概念的说明，虽然它们很枯燥，读起来不舒服。不过有“鸡汤”提供者说，越和自己的不舒服做斗争，越能高于一般人。

1. 数据

数据，英文是 Data。首先要注意不是“数、数字、数值”，虽然这些可以是数据。《维基百科》中的“数据”词条解释为“未经过处理的原始记录”。

由此可知，数据首先是“记录”，用某种方式把某对象保存下来，这就是数据。比如走进餐厅，用录音器材记录人们吃饭时的声音，这就是声音数据；用摄像器材拍摄公路上个别存在的开车违章现象，这就是影像数据。

再者，定义中所言“未经处理”，是说明这种数据缺乏有效组织，并没有被加工成某种序列信息。

或许，这只是对数据的诸多定义中的一种，仅供参考。

如果仅仅按照上面的方式理解，未免太不“计算机”了。在计算机行业，认为数据是能够

被计算机识别、存储和加工处理的。这虽然不是概念定义，但由此读者理解了软件工程师要处理的数据应当具备的属性。

2. 大数据

表述对某个物体的某种测量结果，常常用“数值+单位”的模式。在国际基本单位中，长度的单位是“米”，为了便于应用，在此基础上扩展了其他单位，比如“千米”，更大的单位如“光年”。

在计算机行业，数据的单位有比特（bit），比这个大一点的单位是字节（Byte，1 字节=8 比特）。此外，还有下面各种单位。

KB (Kilo Byte) : 1KB = 2^{10} Byte = 1024 Bytes
MB (Mega Byte) : 1MB = 2^{20} Byte = 1024 KB
GB (Giga Byte) : 1GB = 2^{30} Byte = 1024 MB
TB (Tera Byte) : 1TB = 2^{40} Byte = 1024 GB

打开计算机，通过查看文件属性，能直观地看到每个文件中数据量的多少。这几年，随着网络的发展，某些数据的数值太大了（也可以说数据量很大）。比如，据说 Facebook 每天增加的数据量超过 500TB（消息来源：http://www.infoq.com/cn/news/2012/08/FB-collect-500TB-everyday）。笔者有一个 3GB 的 U 盘，如果用这种 U 盘来存储 Facebook 每天增加的数据，需要多少个这样的 U 盘？

$500 \times 1024 / 3 = 170666.66666666666$

另外一个数据也挺吓人的。据传截至 2012 年，全世界每天产生 2.5 艾字节（2.5×10^{18} 字节）的数据（消息来源：https://www.ibm.com/big-data/us/en/），还用笔者那种 U 盘来存储，需要多少个？

$2.5\times10^{18}/(2^{30}\times3) = 776102145.5128988$

突然想起在 20 世纪 90 年代，笔者还在使用一种容量大约是 1.44MB 的 3.5 英寸的软盘。

可见，不同的对象，其数据大小不同，特别是随着数据量比较大的数据越来越多，很多资料上开始出现了“大数据”这个词语。所谓“大”，应该就是相对某种“小”而言的，这种在“数据”之前添加一个形容词的方法，某种程度上是为了吸引眼球。至少对于“大数据”而言，并不构成一个独立的严谨概念，它只是用来指代某种数据量比较大的对象。但是大于哪一个数值才算“大”呢？没有严格定义。

尽管如此，我们还会面对“大量数据”，此时对技术也有了新的要求，比如数据的存储、数据分析算法等，所以也常常将这些技术方法用“大数据”来指代。这就让本来含糊不清的“大数据”更笼统了，貌似是一个筐——什么都能装。所幸本书的目的不是进行严格的学术名词界定，也就不对“大数据”进行深究，只是从俗使用罢了。

3. 数据分析

数据分析，顾名思义，就是通过分析数据，得到某种结果。被分析的数据对象可“小”、可“大”。

假想一个可能荒唐的场景。我们的祖先，从树上到地面的时间还不是很长的时候，掌握了从 1 到 10 这几个数字（是不是通过观察双手习得的呢）。笔者基本可以保证他们那时的计数方法不是从 0 开始的。某天，祖先们经过长途跋涉，到了一个有食物的地方，带头大姐（注意，

那时是母系社会）清点了一下人数，数了好几遍后终于确定一共 9 个人。通过这个数据，她得出一个结论：这次迁徙比较幸运，一个成员也没少。

这是否算是最早的“数据分析”？如果算，那么数据分析与我们形影不离。

事实上，现在所说的数据分析不是这样简单的过程。

一般认为，数据分析是通过某种方法，从不同维度提炼出数据中所包含的信息。用于分析的方法可以概括为以下几个。

- 描述性分析法（Descriptive Analytics）。比如每个月所挣的工钱按照如下方式分配：房租 40%，食品 30%，交通 20%，通信 10%。也可用饼图等图示直观地表示这种分配方式。
- 预测性分析法（Predictive Analytics）。最典型的就是对股票数据的分析，通过分析结果预测买哪只股票会赚钱。“大数据”通常被认为可以很好地预测未来——准确地说应该是在统计意义上预测。
- 规范性分析（Prescriptive Analytics）。先重复一遍或许读者已经知道的“啤酒与尿布”的故事。据传在 20 世纪 90 年代，美国沃尔玛超市的管理人员分析销售数据时发现，在某些特定的情况下，“啤酒”与“尿布”两件看上去毫无关系的商品会经常出现在同一个购物篮中（对其原因的解释，读者可以上网搜索）。根据此分析结果，超市管理人员开始在卖场尝试将啤酒与尿布摆放在相同的区域。这就是非常典型的所谓“数据驱动”的决策。

或许现实中的“数据分析”不能简单地贴上某一个标签，但从上面三个类别中，也能初步理解数据分析的大体用途和方法。

虽然数据分析的数学基础早在 20 世纪 90 年代已经确立，但直到计算机发展起来之后，伴随着计算能力的提升和数据量的增加，“数据科学”才成为人们关注的焦点。可以说，它是数学与计算机科学相结合的产物。

另外，还有一个类似的名词“数据挖掘（Data Mining）”，有人专门撰文区分两者，也有不少资料将两者混用，还有资料只说其一，不提另外一个，言下之意是两者一样，用一个词语即可。总之，在当前盛产新词汇的时代，面对众多似是而非的名词做取舍，也的确不知所措。所以，笔者的做法就是“不争论”，重实务。

以上列出三个跟数据有关的概念，主要目的在于使读者对“数据”领域有一个概括的认识。而在实际的“数据”语境中，名词绝非上述三个，还有云计算、集群计算、暗数据、数据湖、脏数据、结构化数据、数据科学家、数据分析师、机器学习、人工智能、数据清洗等。或许在本书中读者也会看到一些似是而非、似懂非懂的词语，如果想深究，建议“自己动手，上网搜索”——身处一个造词的新时代，谁也逃不脱。

“透过历史，看××未来”，这是一句常用的话，其中也蕴含着数据分析方法，历史就是过去的数据，需要详查。

0.2　数据分析技术的发展

谁没有历史？！数据分析不是凭空出现的，有一个发展历程，这就是它的历史。了解历史的目的不仅是在茶余饭后聊天用，更是通过历史，在一定程度上判断其发展趋势，并用于今天

和明天的决策之中——这也是数据分析。通过下述的历史过程，或许读者能够判断这门学科的发展趋势，从而确定是否有必要认真阅读本书，是否立志在数据分析领域从业等——风闻，数据分析和机器学习的工程师都是高薪哦。

下面就是数据分析的极简编年史（参考 Gil Press 的 *A Very Short History Of Data Science*，网址是 https://www.forbes.com/sites/gilpress/2013/05/28/a-very-short-history-of-data-science/#5c5686a355cf）。

- 1947 年，John W. Tukey 创造了“比特”这个词。
- 1948 年，Claude Shannon 在论文《通信的数学理论》中使用了“比特”一词。
- 1962 年，John W. Tukey 在《数据分析的未来》一文中提出“数据分析本质上是一门实证科学”。1977 年，他又发表了论文《探索性数据分析》。他认为需要更加重视利用数据来检验所提出的假设，并进一步提出“探索性数据分析”和“验证性数据分析”“可以且应该并行”。
- 1974 年，Peter Naur 出版了《计算机方法的简明综述》，这本书综述了被广泛应用的当代数据处理方法。
- 1989 年，Gregory Piatetsky-Shapiro 组织和主持了第一届 KDD（Knowledge Discover in Databases）研讨会。1995 年，KDD 研讨会成为 ACM SIGKDD 年度会议。
- 1994 年 9 月，《商业周刊》发表了关于“数据库营销”的封面故事，“公司正在收集关于你的大量信息，对这些数据信息进行运算，从而预测你购买某一种产品的可能性，并利用这些知识来制定出精确校准过的营销信息来促使你购买它”。
- 1996 年，Usama Fayyad、Gregory Piatetsky-Shapiro 和 Padhraic Smyth 发表了《从数据挖掘到数据库中的知识发现》，他们认为“KDD 是指从数据中发现有用的知识的全过程，而数据挖掘是指在这一过程中的具体步骤”。
- 1997 年，C. F. Jeff Wu 教授在密歇根大学的演说中呼吁将统计学改名为数据科学，并且将统计学家改名为数据科学家。
- 2002 年 4 月，《数据科学杂志》创刊。
- 2005 年 5 月，Thomas H. Davenport、Don Cohen 和 Al Jacobson 发表研究报告《分析学方面的竞争》，描述了一种新的竞争形式，即统计、定量分析及预测模型开始代替传统因素成为公司竞争的主要元素。这项研究后来被 Davenport 发表于《哈佛商业评论》（2006 年 1 月），之后他同 Jeanne G. Harris 一起将其研究成果编成图书《分析学方面的竞争：致胜的新科学》（2007 年 3 月）。
- 2007 年，数据科学研究中心在复旦大学建立。
- 2009 年 1 月，题为《利用数字数据的力量服务于科学和社会》的报告出版。报告指出，“许多学科中涌现出一种新型的数据科学和管理专家，他们擅长计算机、信息、数据科学领域及另一个科学领域。这些人是当前和未来科研事业成功的关键”。
- 2009 年 1 月，谷歌首席经济学家 Hal Varian 告诉麦肯锡季刊：“掌握数据的能力——能够理解数据、处理数据，从中提取有价值的信息，把数据可视化，传达数据——这将是未来几十年的一项非常重要的技能……因为现在我们确实拥有基本上免费和无处不在的数据。因此，理解这些数据并从中获取价值的能力成为稀缺因素……我的确认为这些技能——能够获取、理解和传达你从数据分析中所获得的见解——将是非常重要的。管

理人员需要能够访问和理解数据本身”。

上述简史终止于 2009 年，虽然那年之后，围绕数据发生的事情还有很多，并且发展迅猛，但是笔者认为 Hal Varian 已经做了很好的预见，后面的事情都在证明他的预见。

“滚滚长江东逝水，浪花淘尽英雄”，众多“牛人”们为我们今天的学习奠定了基础，是他们给了我们可以依靠并站立于上的肩膀，所以我们要“时刻准备着”，为“数据分析的伟大事业”贡献自己绵薄的力量（此处参考了小学作文的部分语句，请回忆）。

0.3　开发环境配置

本节讲解数据分析的工具和使用方法——以 Python 语言为基础的库。在现实中，能够用于数据分析的工具五花八门，概括起来，可以分为以下两大类。

- GUI 软件产品，比如 SPSS、电子表格等，这种工具操作简单，获得了不少用户的青睐。但是由于它们自身是一款软件产品，只能提供软件产品本身的功能，对于“大数据”中多样化的需求就显得无能为力了。还有一个问题也需要提醒读者，它们不都是免费的。很多从事数据分析方面研究、教学和学习的用户，以及使用数据分析工具的用户，或许应用的是盗版软件，对此笔者旗帜鲜明地反对。
- 以 Python 和 R 为代表的高级编程语言，在数据分析领域已经被广泛使用，特别是在处理“大数据”时，其优势不仅在于开源免费，更重要的是能够根据业务需要，灵活多样地进行各种计算，并且在所需要的方向上进行优化。本书当然选择 Python，因为它是“跟老齐学 Python”系列图书中的一员。更重要的是，现实已经表明，Python 在数据科学领域的霸主地位已经确立，它是每个试图进入此领域的人不得不学的——所以《跟老齐学 Python：轻松入门》之后要继续《跟老齐学 Python：数据分析》。

使用 Python 进行数据分析，不是用鼠标点来点去就能解决的，必须使用一些专门的第三方库。所以，安装相应的库是必需的。

笔者默认阅读本书的读者已经完成《跟老齐学 Python：轻松入门》的学习，或者具有相当的 Python 知识技能。

在本书中，通常都是使用 pip 命令安装有关数据分析的库，并且针对 Python 3。

尽管以下要安装的库已经成为数据分析的标配，但是至今它们也没有被纳入标准库，所以要自己安装。

本书所有代码都是基于 Ubuntu 16.04 操作系统调试的——笔者不厌其烦地在各种适合的场合推荐的操作系统。

1. 安装基本库

打开终端，依次输入如下安装指令，特别建议一个一个进行安装。根据经验，有的库可能由于连接超时而安装失败，遇到这种情况笔者也无能为力，除非改变安装方式。有的库安装时间比较长，要有耐心等待。

```
$ sudo pip3 install numpy
$ sudo pip3 install scipy
$ sudo pip3 install matplotlib
```

```
$ sudo pip3 install pandas
$ sudo pip3 install sympy
$ sudo pip3 install ipython
$ sudo pip3 install jupyter
```

安装完毕，会显示是否成功。若还不放心，可以用类似下述方法检验。

```
$ python3
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>>
```

import 后面的那个库，如果不报错，就说明安装成功。

以上安装的仅仅是基础设施，在学习过程中，还会根据需要安装别的库。

因为上述所安装的库都是开源的，所以能够将它们集结在一个大包中，这就是所谓集成式的安装包。通过这个安装包，就能获得上述所有库的环境。其中比较有名的是 Anaconda，用官方网站的说明，它是“The Most Popular Data Science Ecosystem”。下载之后，一次安装，获得上述要求的配置（其实比上述还多）。下载地址是 https://www.continuum.io/downloads，分别有 Windows、Mac OS、Linux 三种操作系统的安装包供下载使用。

诚然，跟其他开源软件一样，还可以下载源码编译安装，这种方式不是笔者在这里推荐的。喜欢用这种方式安装的读者，一定也是高手，笔者就不演示了。

如果读者穷尽所能，依然无法将所要求的环境配置好，那么只能求助搜索引擎了。为此特别奉上一句话：

“搜索决定成败”——老齐

推荐的搜索引擎当然是 google.com，不管你有什么理由，但凡想在数据分析和机器学习领域从业，必须用它——虽然它更多时候是图 0-3-1 这样的，但为了你的目标，也要努力去寻找。

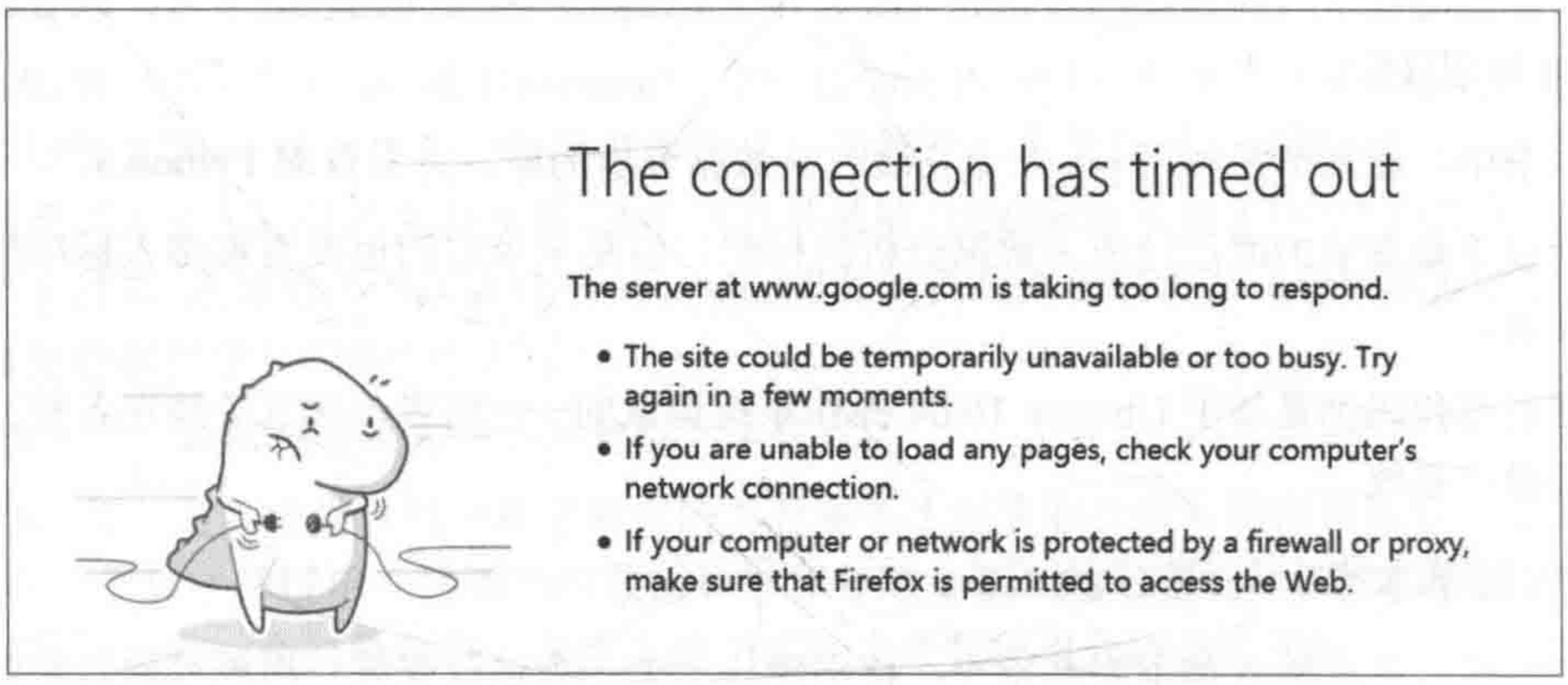

图 0-3-1　google.com 连接超时

准备工作完成，就要开始学习了。

2．使用 Jupyter

在学习 Python 的时候，我们经常在交互模式中进行一些操作。但是，Python 默认的交互模式其实很不友好，你不觉得吗？有这种感觉的人应该不少吧。所以，Jupyter 横空出世了。

Jupyter 官方网站的网址是 http://jupyter.org/，如图 0-3-2 所示。首页这样描述："Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages"。

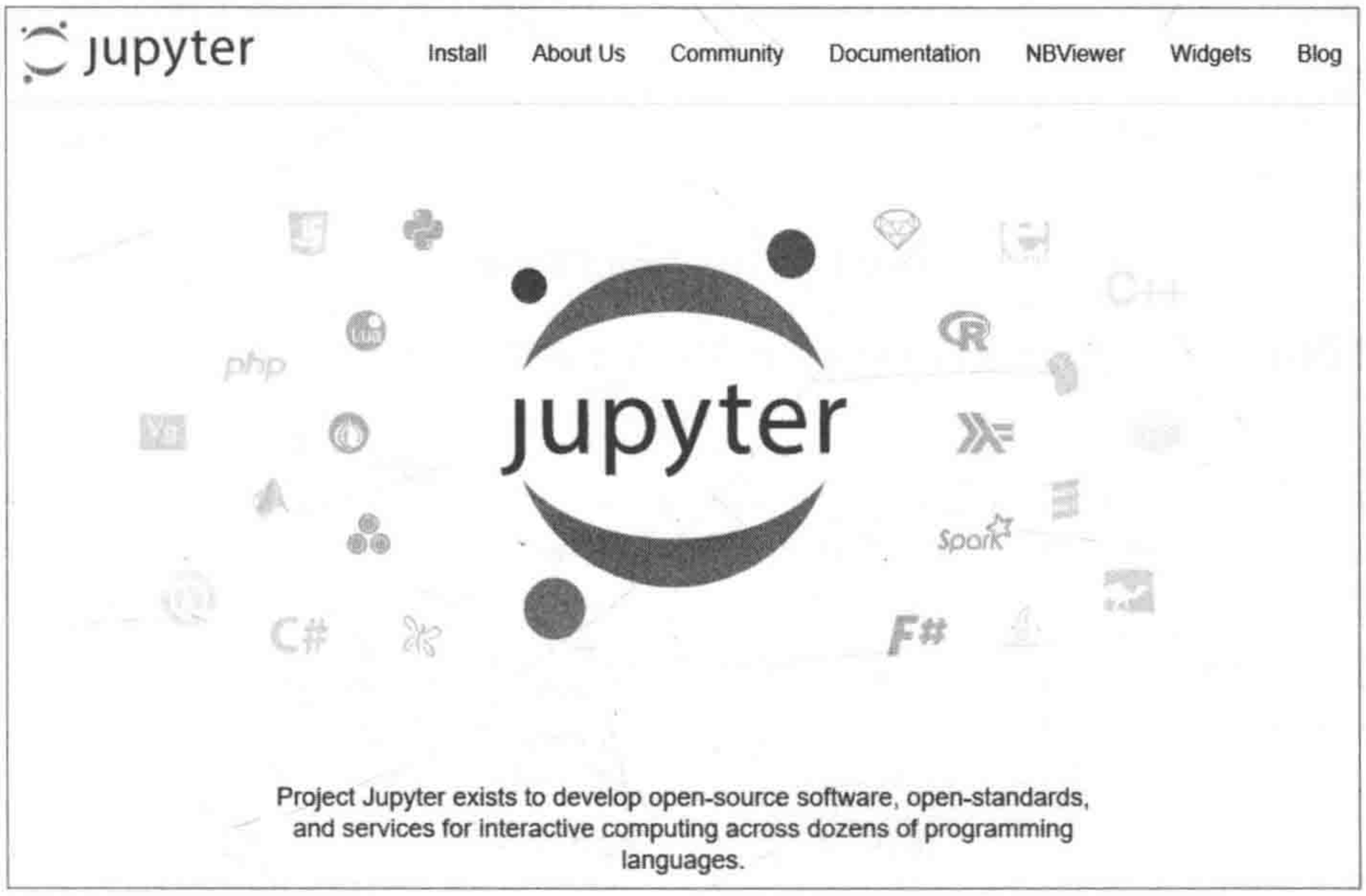

图 0-3-2　Jupyter 官网

Jupyter Notebook 是一种基于浏览器的交互环境，支持的不仅有 Python，也有别的语言。读者在有的资料里还会看到 IPython Notebook，这是它的曾用名。

执行如下命令。

```
qiwsir@ubuntu:~$ jupyter notebook
[I 14:28:46.797 NotebookApp] Serving notebooks from local directory: /home/qiwsir
[I 14:28:46.797 NotebookApp] 0 active kernels
[I 14:28:46.797 NotebookApp] The Jupyter Notebook is running at:
http://localhost:8888/?token=b87a995704e95d9b7ca653e4065aa3c380c73f0aa3a8dd16
[I 14:28:46.797 NotebookApp] Use Control-C to stop this server and shut down all kernels
(twice to skip confirmation).
[C 14:28:46.801 NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=b87a995704e95d9b7ca653e4065aa3c380c73f0aa3a8dd16
[I 14:28:51.780 NotebookApp] Accepting one-time-token-authenticated connection from
127.0.0.1
```

执行上述命令后会自动打开默认浏览器，显示类似如图 0-3-3 所示的界面。

图 0-3-3　启动 Jupyter Notebook

单击图 0-3-3 所示界面中的 New 下拉按钮，在下拉菜单中选择 Python 3，如图 0-3-4 所示。

图 0-3-4　选择 Python 3 交互界面

这时会创建一个新的 Tab，这就是我们的工作界面，如图 0-3-5 所示。

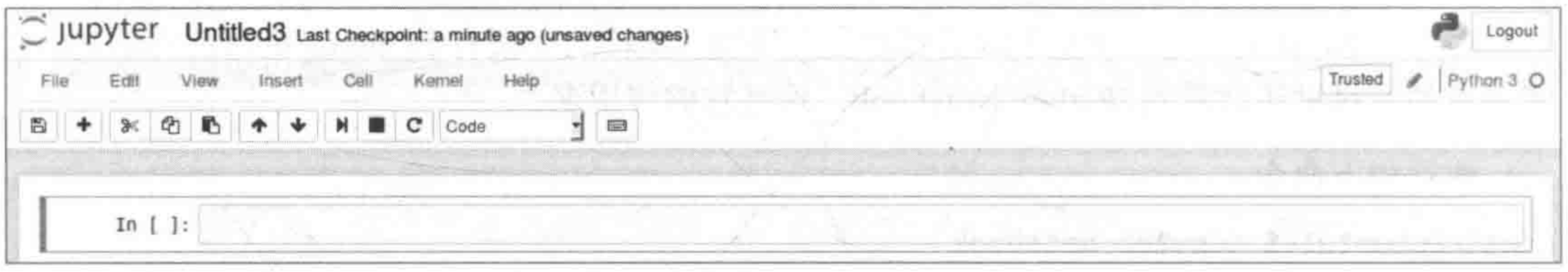

图 0-3-5　Jupyter Notebook 工作界面

关于 Jupyter 的一些使用技巧不是本书的重点，建议读者上网查找资料了解。它的操作非常简单，读者根据已有的软件操作经验，操作它肯定不在话下。

在 Jupyter Notebook 中操作，最终可以将当前页面上的内容保存为扩展名是.ipynb 的文件，这个文件可以传播，也可以将其他的这种文件导入到当前的 Jupyter Notebook 界面中。比如，读者可以在本书代码仓库中下载相应文件并导入——其实笔者不提倡，笔者更提倡自己敲代码。

万事俱备只欠东风，还有你的决心——开始练“神功”。

第1章

NumPy 基础和应用

NumPy 是 Python 语言的一个第三方库，被广泛应用于数据分析领域。它实现了多维数组与矩阵的高效运算，还提供了大量的数学函数。用更高、更快、更强来描述 NumPy 并不为过，“更高”即开发效率高，“更快”即运行速度快，“更强”即数据处理方面的功能强大。毫不夸张地说，NumPy 是任何打算进入数据分析乃至机器学习、人工智能等领域的读者必须要学习并掌握的。

NumPy 的前身是一款名为 Numeric 的库，由 Jim Hugunin 与其他协作者共同开发。2005 年，Travis Oliphant 在 Numeric 中结合另一个同性质的库 Numarray 的特点，并加入了其他扩展而开发了 NumPy。

NumPy 是开源的，其最大的好处就是免费，因此其代码质量能够得到最大限度地保证——开源，意味着最大限度的安全。

NumPy 是数据分析“降龙十八掌”的第一招。

1.1 数组对象基础

“ndarray”是 NumPy 的核心功能，其含义为 n-dimensional array，即多维数组。在后面的叙述中，会经常用到“数组”这个词，就是指的 ndarray。数组是 NumPy 的一个重要数据结构，正如 Python 中“万物皆对象”原则，数组也是一个对象，这个对象具有自身的独特之处，具体表现在其属性和方法上。

1. 初识数组对象

（1）使用 Jupyter Notebook

如果读者按部就班地跟随本书操作，那么已经打开了 Jupyter Notebook 界面。

执行如下操作。

```
In [1]: import numpy as np
        np.__version__
```

```
Out[1]: '1.13.0'
```

In[1]表示输入的内容。输入完毕，按住 Shift 键，再按 Enter 键，就会执行输入语句。如果有结果出来，就会在 Out[1]中显示。In[1]中的数字是程序单元的序号。看一下笔者执行 In[1]的截图，如图 1-1-1 所示。

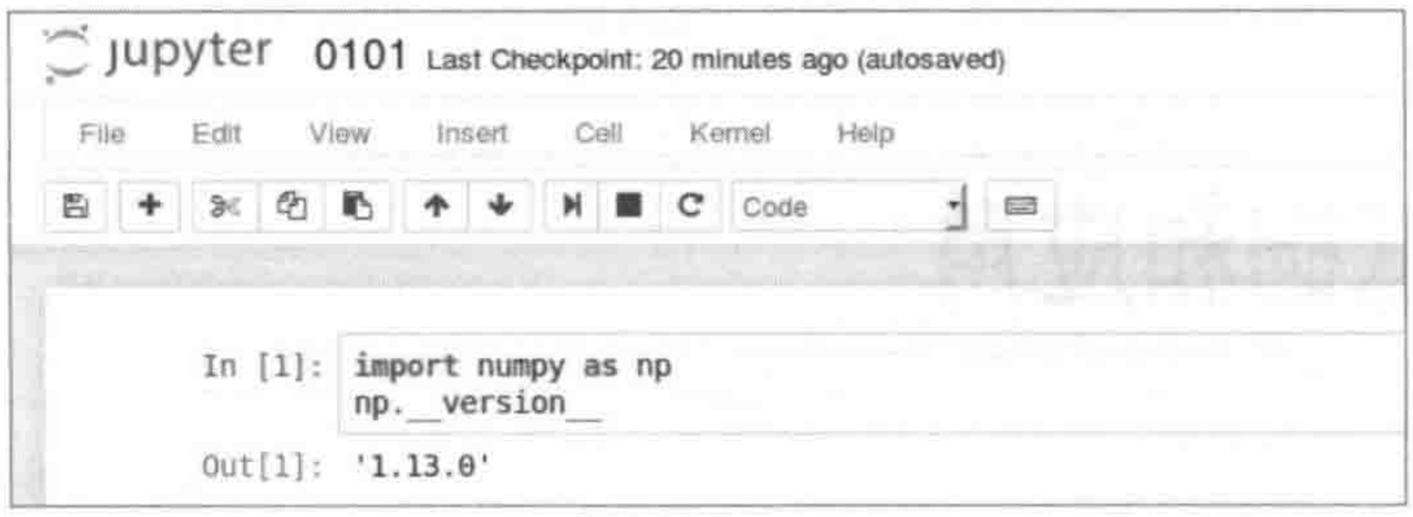

图 1-1-1　执行结果图示

Out[1]的输出结果是当前 NumPy 的版本。或许读者所使用的版本跟笔者所演示的不同，这是很正常的。因为 NumPy 一直在发展，这也是我们使用它的重要原因。如果它的版本号不变化了，你还敢用吗？这又不是古董。当然，新旧版本会有一些差异，但读者对此不必太担心。一方面主体内容不会有太大变化（除非比较大的版本变化，比如升级为 2.xx.x），另一方面笔者在本书中各章节重点强调的不是掌握哪些知识，而是要掌握学习知识的方法，本书中的知识只不过是方法的载体罢了。再者，NumPy 有非常好的帮助文档，甚至在操作的时候，都会有非常友好的提示。

在 In[1]中，就是引入 NumPy。通常都用这种方式引入，请读者也用这种通常的方式——要获得自由，必须遵守规范——这种引入方式能够保证你跟别人快乐地一起玩耍，不至于弄翻友谊的小船。

为了认识数组对象，先要创建一个数组。下面就创建本书的第一个数组。

```
In [2]: data = np.array([1, 2, 3, 4, 5])
        data
Out[2]: array([1, 2, 3, 4, 5])

In [3]: type(data)
Out[3]: numpy.ndarray
```

关于如何创建数组，后面会专门讲解，这里先来认识一下数组。

刚才所创建的数组，其类型为 numpy.ndarray，这个数组即为一个对象。既然如此，它就有一些属性和方法，所以 dir()在这里依然有效（如果读者不知道 dir()的作用，请查阅《跟老齐学 Python：轻松入门》）。

```
In [4]: dir(data)
Out[4]: ['T',
         '__abs__',
         '__add__',
         ......                    #省略很多内容
         'transpose',
         'var',
         'view']
```