

结果即可。这时找到的一定是全局最优解。

读者只需要关心所选择的损失函数是不是凸函数就可以了。

21.1 凸优化扫盲

数学上对凸函数是这样定义的。函数 f 的定义域为凸集，且满足：

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y), (0 \leq \theta \leq 1)$$

这个定义中出现了凸集，凸集的概念是这样的：连接集合 C 内任意两点间的线段均在集合 C 内，则称集合 C 为凸集。凸集的数学定义形式如下：

$$\forall \theta \in [0, 1], \forall x_1, x_2 \in C$$

$$x = \theta x_1 + (1 - \theta)x_2 \in C$$

所以凸集和凸函数是两个概念，前者强调的是集合，后者是一个函数。常见的凸函数有：

- 指数函数 e^{ax} ；
- 幂函数 x^a ， $a \geq 1$ 或 $a \leq 0$ ；
- 负对数函数 $-\log_a x$ ；
- 负熵函数 $x \log_a x$ ；
- 范数函数 $\|x\|_p, p \geq 1$ 。

另外，凸函数之间的某些运算具有保凸性质：比如凸函数的非负加权和还是凸

函数，即如果函数 f_1, f_2, \dots, f_m 都是凸函数， $w_1, w_2, \dots, w_m \geq 0$ ，则 $\sum w_i f_i$ 还是凸函数。

机器学习之所以喜欢凸函数，就是因为凸函数的局部最优解即为全局最优解。

21.2 正则化和凸优化

在回归问题中，我们定义的损失函数是均方误差损失函数：

$$J(\theta) = \|y - X\theta\|_2^2$$

从函数的凸凹性来说，均方误差损失函数是凸函数，感兴趣的读者可以自己证明。所以采用梯度下降算法找到的局部最优解一定是全局最优解。

为了避免过拟合，又引入了 L1、L2 两种正则项，L2 正则项为 $\|\theta\|_2^2$ ，加入后的损失函数如下：

$$J(\theta) = \|y - X\theta\|_2^2 + \lambda \|\theta\|_2^2$$

引入正则项后的损失函数还是凸函数吗？答案是肯定的，可以用前面提到的保凸性质证明。所以，尽管对损失函数加上了正则项，但是函数的凸凹性并没有被破坏，找到的局部最优解仍然还是全局最优解。

在逻辑回归中，并没有使用均方误差作为损失函数，而是使用 Log Loss 作为损失函数，其原因也是因为前者不是凸函数，而后者是凸函数，对这一点感兴趣的读者可以自己证明。

21.3 小结

凸优化本身是一个难度很大的主题，本章仅仅介绍了优化和凸优化的基本概念。其实我们的目标只有一个，就是希望千辛万苦找到的解是全局的最优解。但要保证这一点，就要求损失函数是凸函数，于是就要知道什么是凸函数，以及常见的算法中是不是用了凸函数。有了这些知识后，才能放心地建模。

附录 A 工作环境搭建说明

本书采用 Python 编程语言对书中案例进行编码实现。

近几年来，Python 编程语言炙手可热，国内已有很多地区将 Python 纳入中小学课程。现在，随着计算机软硬件的发展，人工智能在沉寂多年之后再次进入活跃期，Python 也凭借其特性成为人工智能领域的首选编程语言。

好吧，不给 Python 打广告了，接下来严肃地介绍一下在人工智能领域为何选择 Python 作为编程语言。

A.1 什么是 Python

1989 年圣诞节期间，阿姆斯特丹的 Guido van Rossum 为了打发圣诞节的无聊时间，开发了一个新的脚本解释程序，于是就有了 Python。之所以选 Python（大蟒蛇）作为该语言的名字，是因为他是一个名为 Monty Python 的喜剧团体的“死忠粉”。

所以，Python 其实是一门非常古老的语言，它的出生时间要比 Java 还早一年，算起来也算是步入中年了。可为什么 Python 在之前一直默默无闻，这几年却突然“老树开花”了呢？

其实，并不是 Python 语言本身有多大的改进，而是数据时代到来了。回想在大数据刚兴起时，很多人对此都一头雾水，更别提与之相关的云计算等技术了。没

想到短短几年时间，这些技术已经成为常规技术。预计在不远的将来，数据处理能力将成为每一位职场人员的基本技能，就像会操作电脑、懂英文、能驾驶汽车那样——谁让我们出生在数据时代呢！

Python 语言之所以是数据科学的标配工具，可以从两点进行解释。首先来看图 A-1。该图在一定程度上解释了 Python 是数据科学领域首选编程语言的原因。

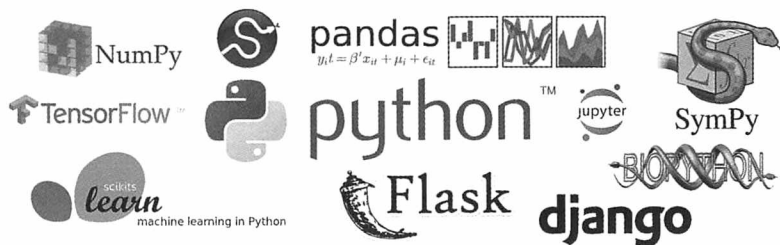
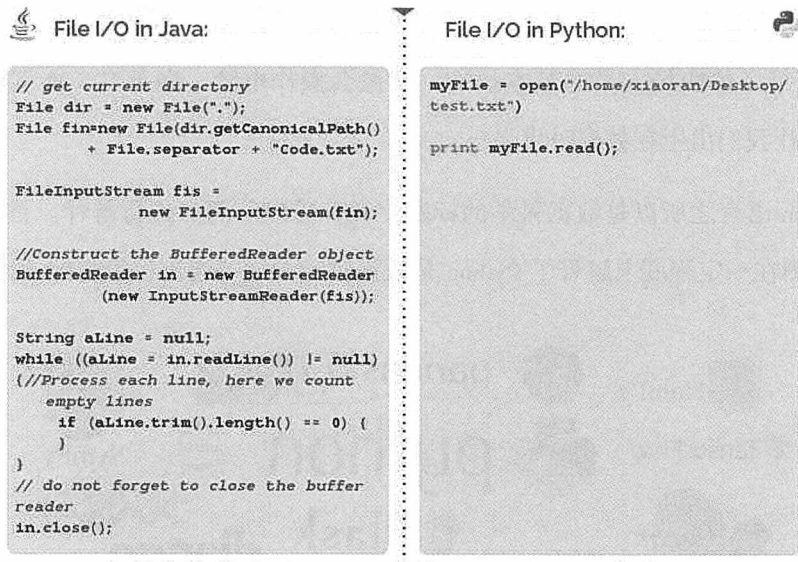


图 A-1 Python 生态圈

在 Python 生态圈中，针对数据处理有一套完整且行之有效的工具包，比如 NumPy、Pandas、scikit-learn、Matplotlib、TensorFlow、Keras 等。从数据采集到数据清洗、数据展现，再到机器学习，Python 生态圈都有非常完美的解决方案。

套用现在热门的说法，Python 的数据处理功能已经形成了一个完整的生态系统，这是其他编程语言（比如 Java、C++）望尘莫及的，所以 Python 已经成为数据科学领域事实上的标配工具。

再者，Python 语法极其简洁，相较于 Java、C 等编程语言，已经非常接近于人类语言。通过图 A-2 中两个代码片段的对比，大家可以对此有直观认识。



图A-2 Java代码与Python代码的对比

这两个代码片段做的事情相同：

- 打开磁盘上的一个文件；
- 读出其中的内容；
- 打印到屏幕上。

显而易见，Python 代码要更加清晰、明确。

需要说明的是，这个比较并不是说 Python 要比 Java 好，否则也无法解释 Java 多年来一直处于编程语言排行榜的首位，而 Python 只是在最近几年才开始“风头大盛”。编程语言之间没有比较的意义，只能说每种语言都有其特定的适用领域。

比如，Java 是企业级应用开发的首选编程语言，PHP 则是前几年网站开发的首选，最近因为推崇全栈式概念，导致越来越多的人转投 Node.js 阵营。虽然 Python 近乎无所不能，但是无论是企业级应用开发还是网站开发，均不是 Python 的强项。比如就网站开发来说，这么多年以来貌似只有“豆瓣”是采用 Python 开发的。

给初学者的建议

建议初学者先想清职业发展方向,然后再选择要学习的编程工具。如果打算以后从事网站开发,Node.js 是一个很好的选择;如果打算从事数据分析、机器学习相关的职业,Python 无疑是绝佳选择。

A.2 本书所需的工作环境

从一定程度上来说,编程是一个体力工作。要想学好 Python,必须通过高强度的编码实践来强化“肌肉记忆”。工欲善其事,必先利其器。为了提升学习效率,良好的学习环境是必不可少的。这里至少需要安装两个软件:Anaconda 和 PyCharm。

Anaconda 是一个比较流行的 Python 解释器,并且还是免费的。初学人员在学习 Python 时,建议不要选择 Python 官方提供的解释器,因为这需要自行手动安装许多第三方扩展包,这对于初学人员来说是一个不小的挑战,甚至会耗尽你的学习热情从而放弃。

读者可以去 Anaconda 官网下载最新的 Anaconda 安装包。

A.2.1 Anaconda 版本选择

众所周知,当前存在 Python 2 和 Python 3 两个版本,这两个版本并不完全兼容,两者在语法上存在明显的差异。下面列出了 Python 2 和 Python 3 的差别:

- 支持 Python 2 的工具包多于 Python 3;
- 目前很多 Python 入门教程采用的都是 Python 2;
- TensorFlow 在 Windows 平台上只支持 Python 3.5 以上的版本。

因此，Anaconda 在 Python 2 和 Python 3 的基础之上也推出了两个发行版本，即 Anaconda 2 和 Anaconda 3。建议大家同时安装 Anaconda 2 和 Anaconda 3，以便从容应对各种情况。

A.2.2 多版本共存的 Anaconda 安装方式

如果要在计算机上同时安装 Anaconda 2 和 Anaconda 3，并希望能在两者之间自由切换，通行的做法是以其中一个版本为主，另外一个版本为辅，后期即可根据需要在两个版本之间自由切换。作者习惯将 Anaconda 2 作为主版本，将 Anaconda 3 作为辅版本，所以下面的演示也以这种顺序为基础。如果大家想把 Anaconda 3 作为主版本，只需将下面两个安装过程换个顺序即可。

A.2.3 安装 Anaconda 主版本（Anaconda 2）

Anaconda 主版本的安装很简单，就像安装普通的 Windows 软件那样，一路单击“Next”按钮即可。这里只介绍几个重要的安装节点。

在安装 Anaconda 2 时，首先要设置好安装路径。在图 A-3 中，Anaconda 2 安装在 D 盘的 Anaconda 目录下。

提示

Anaconda 3 辅版本也会安装在这个目录下。

选中图 A-4 中的两个选项，它们各自的作用如下：

- 第一个选项是将 Anaconda 的安装目录添加到系统的 PATH 环境变量中，以便后续在命令行窗口中可以直接用 Python 命令进入 Python 的交互式环境；
- 第二个选项是让 IDE 工具（比如我使用的 PyCharm）能够检测到 Anaconda

主版本，并将其作为默认的 Python 2.7 解释器。

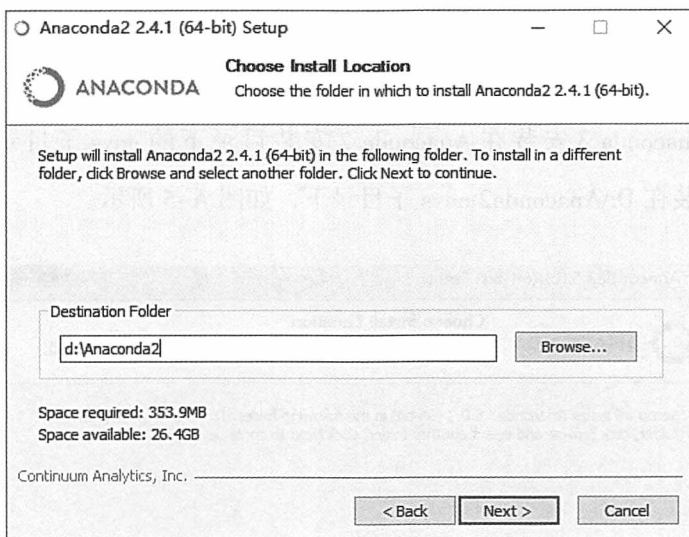


图 A-3 设置 Anaconda 主版本的安装路径

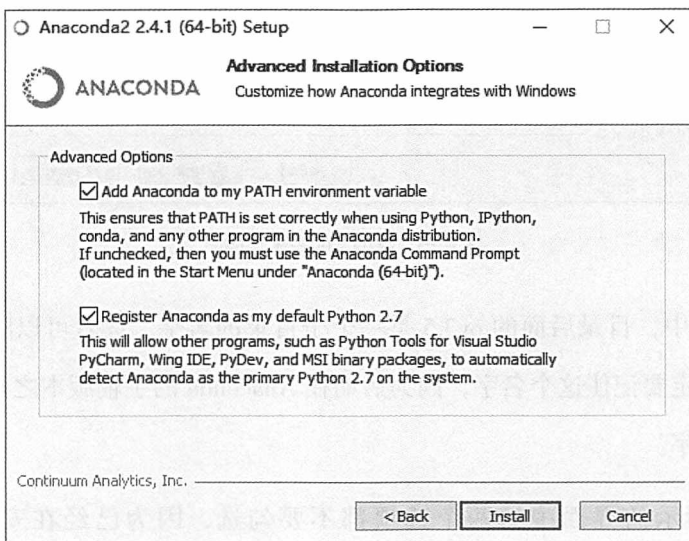


图 A-4 两个选项都要勾选

在安装完 Anaconda 主版本之后，接下来要安装辅版本。本书将 Anaconda 3 作

为辅版本，同样只关注几个重要的安装节点。

A.2.4 安装 Anaconda 辅版本 (Anaconda 3)

必须将 Anaconda 3 安装在 Anaconda 2 安装目录下的 envs 子目录下。下面将 Anaconda 3 安装在 D:\Anaconda2\envs 子目录下，如图 A-5 所示。

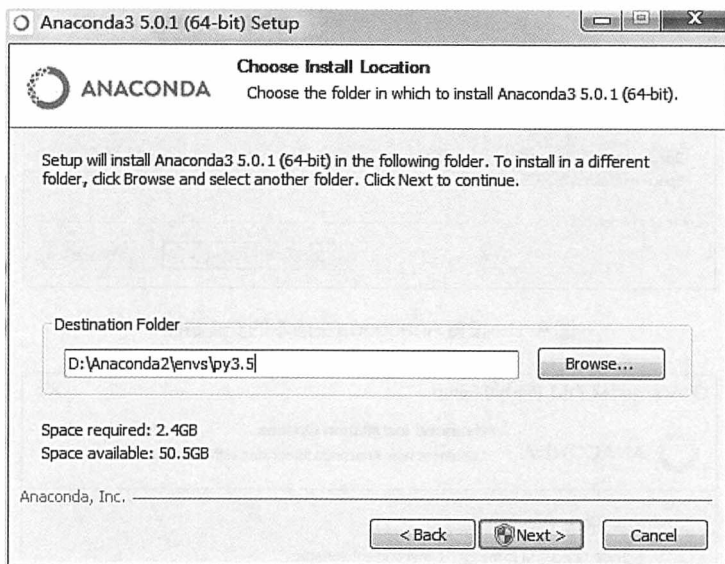
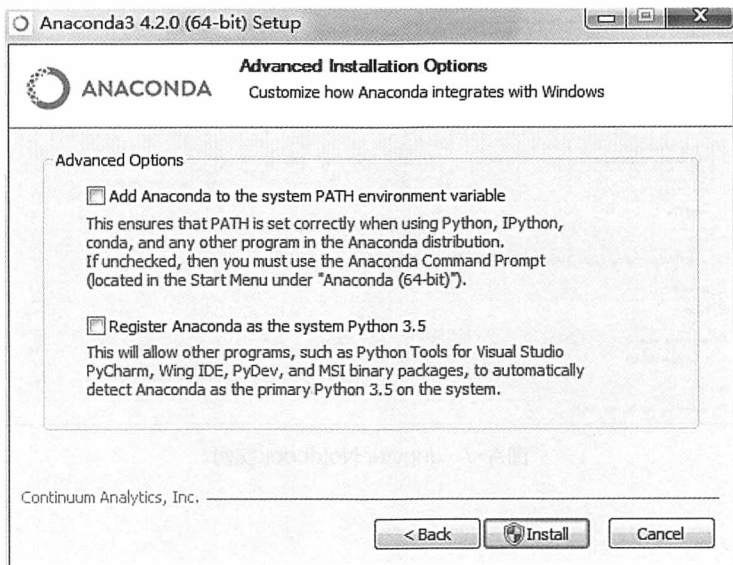


图 A-5 辅版本的安装路径

在图 A-5 中，目录后面的 py3.5 是一个子目录的名字。读者可以随意命名该子目录，但是一定要记住这个名字，因为后期在 Anaconda 的主辅版本之间进行切换时会用到这个名字。

图 A-6 所示的界面中的两个选项都不要勾选，因为已经在安装主版本的 Anaconda 2 时进行了相应设置。



图A-6 辅版本的两个选项不要勾选

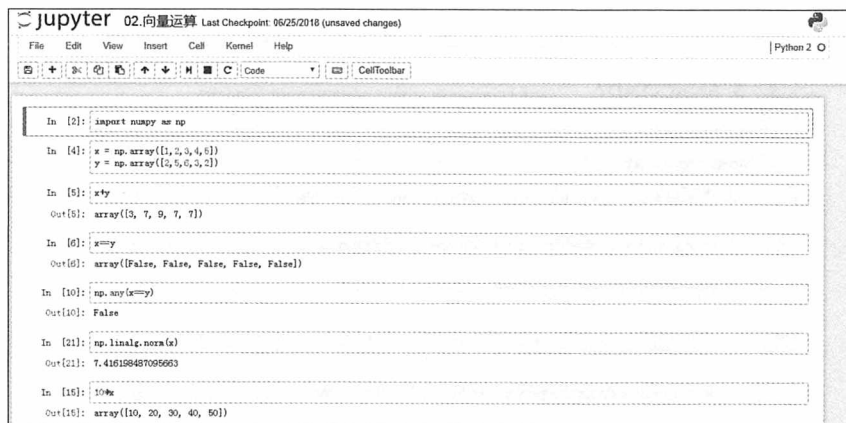
A.2.5 开发工具的选择

在安装好 Anaconda 之后，就可以编写代码了。当前有两种常见的代码编辑工具：Jupyter Notebook 和 IDE。

Jupyter Notebook 是一种常见的代码编辑工具，类似于在 Web 页面上编写代码，如图 A-7 所示。这种代码编辑工具的优势是，可以像记笔记那样编写代码，非常便于编程人员之间的交流。但是，这种代码编辑工具并不是工业界的首选。

在企业开发中，IDE 更为常见、通用，因为此时我们要做的并不是代码演示，而是需要做一些真实的工作：代码开发、模块编写、单元测试、集成测试以及版本控制等。这样一来，Jupyter Notebook 这样的工具就无法胜任了。

推荐大家选择 PyCharm 或者微软的 VS Code 作为自己的 IDE 工具。有关 Python IDE 工具的更多信息，感兴趣的读者可自行查询相关资料。



The screenshot displays a Jupyter Notebook window titled "jupyter 02.向量运算 Last Checkpoint: 06/25/2018 (unsaved changes)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for saving, undo, redo, and other actions. The notebook contains several code cells, each with an input prompt (In [n]:) and an output (Out[n:]).

```
In [2]: import numpy as np
In [4]: x = np.array([1, 2, 3, 4, 5])
        y = np.array([1, 5, 0, 3, 2])
In [5]: x*y
Out[5]: array([3, 7, 0, 7, 7])
In [6]: x==y
Out[6]: array([False, False, False, False, False])
In [10]: np.any(x==y)
Out[10]: False
In [21]: np.linalg.norm(x)
Out[21]: 7.416198487095663
In [15]: 10*x
Out[15]: array([10, 20, 30, 40, 50])
```

图A-7 Jupyter Notebook实例

结语

恭喜大家坚持学完了本书。

所有知识点都掌握了吗？（太过基础的知识需要读者自己默默啃课本哦！）

很多时候理论学习总是枯燥，然而我们却无法忽略它。这时，想象自己独自行走在荒无人烟的塔克拉玛干大沙漠，有千年胡杨做伴，也不觉孤独。

如果读者觉得这本书太浅显，不够烧脑，可以到网易云课堂上搜索我（“大圣”老师）的“人工智能的数学三部曲”。挑战不是一点点呢！



 <h3>程序员讲线性代数</h3> <p>讲师：张晓明（大圣）</p> <p>人工智能中的数学</p> <p>系列课程之一</p>	<h4>程序员讲线性代数</h4> <p>大圣数据学堂</p> <p>对大多数人来说，数学就是牛头马面，大圣教你只需要中学数学知识就能开挂！</p> <p>现在参加，25人与你并肩作战！</p> 
 <h3>程序员讲凸优化</h3> <p>人工智能中的数学</p> <p>系列课程之二</p>	<h4>凸优化</h4> <p>大圣数据学堂</p> <p>凸优化可以简单如神经网络，也可以复杂如SVM，思想非常朴实！</p> <p>现在参加，13人与你并肩作战！</p> 
 <h3>程序员讲概率统计</h3> <p>讲师：张晓明（大圣）</p> <p>人工智能中的数学</p> <p>系列课程之三</p>	<h4>概率和统计</h4> <p>大圣数据学堂</p> <p>想知道BAT面试时都会问哪些概率题吗？我来告诉你！</p> <p>现在参加，11人与你并肩作战！</p> 