

TURING

图灵程序设计丛书

[PACKT]
PUBLISHING

[美] Prateek Joshi 著 陶俊杰 陈小莉 译

Python 机器学习经典实例

Python Machine Learning Cookbook



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

作者简介

Prateek Joshi

人工智能专家，重点关注基于内容的分析和深度学习，曾在英伟达、微软研究院、高通公司以及硅谷的几家早期创业公司任职。

数字版权声明

图灵社区的电子书没有采用专有客户端，您可以在任意设备上，用自己喜欢的浏览器和PDF阅读器进行阅读。

但您购买的电子书仅供您个人使用，未经授权，不得进行传播。

我们愿意相信读者具有这样的良知和觉悟，与我们共同保护知识产权。

如果购买者有侵权行为，我们可能对该用户实施包括但不限于关闭该帐号等维权措施，并可能追究法律责任。



图灵程序设计丛书

Python Machine Learning Cookbook

Python机器学习经典实例

[美] Prateek Joshi 著

陶俊杰 陈小莉 译

人民邮电出版社
北京

图书在版编目 (C I P) 数据

Python机器学习经典实例 / (美) 普拉提克·乔西
(Prateek Joshi) 著 ; 陶俊杰, 陈小莉译. -- 北京 :
人民邮电出版社, 2017. 8
(图灵程序设计丛书)
ISBN 978-7-115-46527-6

I. ①P… II. ①普… ②陶… ③陈… 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(2017)第178314号

内 容 提 要

在如今这个处处以数据驱动的世界中, 机器学习正变得越来越大众化。它已经被广泛地应用于不同领域, 如搜索引擎、机器人、无人驾驶汽车等。本书首先通过实用的案例介绍机器学习的基础知识, 然后介绍一些稍微复杂的机器学习算法, 例如支持向量机、极端随机森林、隐马尔可夫模型、条件随机场、深度神经网络, 等等。

本书是为想用机器学习算法开发应用程序的 Python 程序员准备的。它适合 Python 初学者阅读, 不过熟悉 Python 编程方法对体验示例代码大有裨益。

◆ 著 [美] Prateek Joshi
译 陶俊杰 陈小莉
责任编辑 朱 巍
执行编辑 徐晓娟
责任印制 彭志环
◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京 印刷
◆ 开本: 800×1000 1/16
印张: 16.5
字数: 390千字 2017年8月第1版
印数: 1 - 35 00册 2017年8月北京第1次印刷
著作权合同登记号 图字: 01-2016-9523号

定价: 59.00元

读者服务热线: (010)51095186转600 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京东工商广登字 20170147 号

版 权 声 明

Copyright © 2016 Packt Publishing. First published in the English language under the title *Python Machine Learning Cookbook*.

Simplified Chinese-language edition copyright © 2017 by Packt Publishing. All rights reserved.

本书中文简体字版由Packt Publishing授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

译者序

有一天，忽然想到自己整天面对着52个英文字母、9个数字、32个符号^①和一个空格，经常加班没有双休日，好傻。时间不断被各种噪声碎片化，完全就是毛姆在《月亮和六便士》里写的，“If you look on the ground in search of a sixpence, you don't look up, and so miss the moon”，整天低头刷手机，却记得举头望明月。生活也愈发无序，感觉渐渐被掏空。薛定谔的《生命是什么》给我提了个醒，他在“以‘负熵’为生”(It Feeds On ‘negative Entropy’)一节指出：“要活着，唯一的办法就是从环境里不断地汲取负熵。”在介绍了熵的概念及其统计学意义之后，他紧接着在“从环境中引出‘有序’以维持组织”(Organization Maintained By Extracting ‘Order’ From The Environment)一节进一步总结：“一个有机体使本身稳定在较高的有序水平上(等于熵的相当低的水平上)的办法，就是从环境中不断地吸取秩序。”这个秩序(负熵、 $k\log(1/n)$)可以是食物，也可以是知识，按主流叫法就是“正能量”(有些所谓正能量却碰巧是增加系统无序水平的正熵)。于是，我开始渐渐放弃那些让人沮丧的老梗，远离那些引发混乱的噪声，重新读书，试着翻译，学会去爱。这几年最大的收获就是明白了“隔行如隔山”的道理，试着循序渐进，教学相长，做力所能及之事，让编程变简单。

一般人都不喜欢编程，更不喜欢动手编程(时间消耗：编写 & 测试 40%、重构 40%、风格 & 文档 20%)，却喜欢在心里、嘴上编程：“先这样，再那样，如果要XX，就YY，最后就可以ZZ了。”分分钟可以说完几万行代码的项目，水还剩大半杯。一旦大期将近，即使要亲自动手Copy 代码，也会觉得苦不堪言，键盘不是红与黑、屏幕不能左右推、小狗总是闹跑追，不断在数不清的理由中增加自己的熵。偶尔看编程书的目的也很明确，就是为了快速上手，找到答案。当然也是在Google、StackOverflow、GitHub网站上找不到答案之后，无可奈何之举。编程书把看着复杂的知识写得更复杂，虽然大多篇幅不输“飞雪连天射白鹿，笑书神侠倚碧鸳”等经典，且纲举目张、图文并茂，甚至有作者爱引经据典，却极少有令人拍案的惊奇之处。为什么同样是文以载道，编程书却不能像武侠小说一样简单具体，反而显得了无生趣，令人望而却步？虽然编程的目的就是用计算机系统解决问题，但是大多数问题的知识都在其他领域中，许多作者在介绍编程技巧时，又试图介绍一些并不熟悉的背景知识，显得生涩难懂，且增加了书的厚度。

^① 见文末Python示例代码。

有时我们真正需要的，就是能快刀斩乱麻的代码。（Talk is cheap, show me the code.）编程与研究数理化不同，没有任何假设、原命题、思维实验，并非科学；与舞剑、奏乐、炒菜相似，都是手艺，只要基础扎实，便结果立判。编程技巧也可以像剑谱、乐谱、食谱一般立竿见影，这本《Python机器学习经典实例》正是如此，直接上代码，照着做就行，不用纠结为什么。

机器学习是交叉学科，应用广泛，目前主流方法为统计机器学习。既然是以统计学为基础，那么就不只是计算机与数学专业的私房菜了，机器学习在自然科学、农业科学、医药科学、工程与技术科学、人文与社会科学等多种学科中均可应用。如果你遇到了回归、分类、预测、聚类、文本分析、语音识别、图像处理等经典问题，需要快速用Python解决，那么这本菜谱适合你。即使你对机器学习方法还一知半解，也不妨一试。毕竟是Python的机器学习，还能难到哪儿去呢？目前十分流行的Python机器学习库scikit-learn^①是全书主角之一，功能全面，接口友好，许多经典的数据集和机器学习案例都来自Kaggle^②。若有时间追根溯源，请研究周志华教授的《机器学习》西瓜书，周教授啃着西瓜把机器学习调侃得淋漓尽致，详细的参考文献尤为珍贵。但是想当作菜谱看，拿来就用，还是需要费一番功夫；若看书不过瘾，还有吴恩达（Andrew Ng）教授在Coursera上的机器学习公开课^③，机器学习入门最佳视频教程，吴教授用的工具是Matlab的免费开源版本Octave^④，你也可以用Python版^⑤演示教学示例。

学而时习之，不亦乐乎。学习编程技巧，解决实际问题，是一件快乐的事情。希望这本Python机器学习经典案例，可以成为你的负熵，帮你轻松化解那些陈年老梗。如果再努努力，也许陆汝钤院士在《机器学习》序言中提出的6个问题^⑥，你也有答案了。

示例代码：

```
"""打印ASCII字母表、数字、标点符号"""

import string

for item in [string.ascii_letters,
```

① scikit-learn网址：<http://scikit-learn.org/stable/>。

② Kaggle是一个2010年成立的数据建模和数据分析竞赛平台，全球数据科学家、统计学家、机器学习工程师的聚集地，上面有丰富的数据集，经典的机器学习基础教程，以及让人流口水的竞赛奖金，支持Python、R、Julia、SQLite，同时也支持jupyter notebook在线编程环境，2017年3月8日被谷歌收购。

③ 分免费版和付费版（购买结业证书），学习内容一样，<https://zh.coursera.org/learn/machine-learning>。

④ Octave下载地址：<https://www.gnu.org/software/octave/>。

⑤ GitHub项目：<https://github.com/mstampfer/Coursera-Stanford-ML-Python>。

⑥ 陆院士的6个问题是：1. 机器学习早期的符号机器学习，如何在统计机器学习主流中发展；2. 统计机器学习算法中并不现实的“独立同分布”假设如何解决；3. 深度学习得益于硬件革命，是否会取代统计机器学习；4. 机器学习用的都是经典的概率统计、代数逻辑，而目前仅有倒向微分方程用于预测，微分几何的流形用于降维（流形学习，Manifold learning，科普见博文<http://blog.pluskid.org/?p=533>），只是数学领域的一角，其他现代数学理论是否可以参与其中；5. 机器学习方法仍不够严谨，例如目前流形学习直接将高维数据集假设成微分流形，需要进一步完善；6. 大数据与统计机器学习是如何互动的。

```
    string.digits,  
    string.punctuation]:  
print('{}\t{}'.format(len(item), item))
```

输出结果：

```
52 abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ  
10 0123456789  
32 !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

前　　言

在如今这个处处以数据驱动的世界中，机器学习正变得越来越大众化。它已经被广泛地应用于不同领域，如搜索引擎、机器人、无人驾驶汽车等。本书不仅可以帮你了解现实生活中机器学习的应用场景，而且通过有趣的菜谱式教程教你掌握处理具体问题的算法。

本书首先通过实用的案例介绍机器学习的基础知识，然后介绍一些稍微复杂的机器学习算法，例如支持向量机、极端随机森林、隐马尔可夫模型、条件随机场、深度神经网络，等等。本书是为想用机器学习算法开发应用程序的Python程序员准备的。它不仅适合Python初学者（当然，熟悉Python编程方法将有助于体验示例代码），而且也适合想要掌握机器学习技术的Python老手。

通过本书，你不仅可以学会如何做出合理的决策，为自己选择合适的算法类型，而且可以学会如何高效地实现算法以获得最佳学习效果。如果你在图像、文字、语音或其他形式的数据处理中遇到困难，书中处理这些数据的机器学习技术一定会对你有所帮助！

本书内容

第1章介绍各种回归分析的监督学习技术。我们将学习如何分析共享单车的使用模式，以及如何预测房价。

第2章介绍各种数据分类的监督学习技术。我们将学习如何评估收入层级，以及如何通过特征评估一辆二手车的质量。

第3章论述支持向量机的预测建模技术。我们将学习如何使用这些技术预测建筑物里事件发生的概率，以及体育场周边道路的交通情况。

第4章阐述无监督学习算法，包括K-means聚类和均值漂移聚类。我们将学习如何将这些算法应用于股票市场数据和客户细分。

第5章介绍推荐引擎的相关算法。我们将学习如何应用这些算法实现协同滤波和电影推荐。

第6章阐述与文本数据分析相关的技术，包括分词、词干提取、词库模型等。我们将学习如何使用这些技术进行文本情感分析和主题建模。

第7章介绍与语音数据分析相关的算法。我们将学习如何建立语音识别系统。

第8章介绍分析时间序列和有序数据的相关技术，包括隐马尔可夫模型和条件随机场。我们将学习如何将这些技术应用到文本序列分析和股市预测中。

第9章介绍图像内容分析与物体识别方面的算法。我们将学习如何提取图像特征，以及建立物体识别系统。

第10章介绍在图像和视频中检测与识别面部的相关技术。我们将学习使用降维算法建立面部识别器。

第11章介绍建立深度神经网络所需的算法。我们将学习如何使用神经网络建立光学文字识别系统。

第12章介绍机器学习使用的数据可视化技术。我们将学习如何创建不同类型的图形和图表。

阅读背景

Python 2.x 和 Python 3.x 的版本之争尚未平息^①。一方面，我们坚信世界会向更好的版本不断进化，另一方面，许多开发者仍然喜欢使用 Python 2.x 的版本。目前许多操作系统仍然内置 Python 2.x。本书的重点是介绍 Python 机器学习，而非 Python 语言本身。另外，考虑到程序的兼容性，书中用到了一些尚未被迁移到 Python 3.x 版本的程序库，因此，本书依然选择 Python 2.x 的版本。我们会尽最大努力保持代码兼容各种 Python 版本，因为这样可以让你轻松地理解代码，并且很方便地将代码应用到不同场景中。

读者对象

本书是为想用机器学习算法开发应用程序的 Python 程序员准备的。它适合 Python 初学者阅读，不过熟悉 Python 编程方法对体验示例代码大有裨益。

内容组织

在本书中，你会频繁地看到下面这些标题（准备工作、详细步骤、工作原理、更多内容、另请参阅）。

为了更好地呈现内容，本书采用以下组织形式。

^① 2020 年之前应该不会终结。——译者注

准备工作

这部分首先介绍本节目标，然后介绍软件配置方法以及所需的准备工作。

详细步骤

这部分介绍具体的实践步骤。

工作原理

这部分通常是对前一部分内容的详细解释。

更多内容

这部分会补充介绍一些信息，帮助你更好地理解前面的内容。

另请参阅

这部分提供一些参考资料。

排版约定

在本书中，你会发现一些不同的文本样式。这里举例说明它们的含义。

嵌入代码、命令、选项、参数、函数、字段、属性、语句等，用等宽的代码字体显示：“这里，我们将25%的数据用于测试，可以通过`test_size`参数进行设置。”

代码块用如下格式：

```
import numpy as np
import matplotlib.pyplot as plt

import utilities

# Load input data
input_file = 'data_multivar.txt'
X, y = utilities.load_data(input_file)
```

命令行输入或输出用如下格式：

```
$ python object_recognizer.py --input-image imagefile.jpg --model-file
erf.pkl --codebook-file codebook.pkl
```

新术语和重要文字将采用黑体字。你在屏幕上看到的内容，包括对话框或菜单里的文本，都将这样显示：“如果你将数组改为(0, 0.2, 0, 0, 0)，那么Strawberry部分就会高亮显示。”

读者反馈

我们非常欢迎读者的反馈。如果你对本书有些想法，有什么喜欢或是不喜欢的，请反馈给我们，这将有助于我们出版充分满足读者需求的图书。

一般性反馈请发送电子邮件至feedback@packtpub.com，并在邮件主题中注明书名。

如果你在某个领域有专长，并有意编写一本书或是贡献一份力量，请参考我们的作者指南，地址为<http://www.packtpub.com/authors>。

客户支持

你现在已经是引以为傲的Packt读者了。为了能让你的购买物超所值，我们还为你准备了以下内容。

下载示例代码

你可以用你的账户从<http://www.packtpub.com>下载所有已购买Packt图书的示例代码文件。如果你是从其他途径购买的本书，可以访问<http://www.packtpub.com/support>并注册，我们将通过电子邮件把文件发送给你。

可以通过以下步骤下载示例代码文件：

- (1) 用你的电子邮件和密码登录或注册我们的网站；
- (2) 将鼠标移到网站上方的客户支持 (SUPPORT) 标签；
- (3) 单击代码下载与勘误 (Code Downloads & Errata) 按钮；
- (4) 在搜索框 (Search) 中输入书名；
- (5) 选择你要下载代码文件的书；
- (6) 从下拉菜单中选择你的购书途径；
- (7) 单击代码下载 (Code Download) 按钮。

你也可以通过单击Packt网站上本书网页上的代码文件 (Code Files) 按钮来下载示例代码，该网页可以通过在搜索框 (Search) 中输入书名获得。以上操作的前提是你已经登录了Packt网站。

下载文件后，请确保用以下软件的最新版来解压文件：

-
- WinRAR / 7-Zip for Windows ;
 - Zipeg / iZip / UnRarX for Mac ;
 - 7-Zip / PeaZip for Linux 。

本书的代码包也可以在GitHub上获得，网址是<https://github.com/PacktPublishing/Python-Machine-Learning-Cookbook>。另外，我们在<https://github.com/PacktPublishing>上还有其他书的代码包和视频，请需要的读者自行下载。

下载本书的彩色图片

我们也为你提供了一份PDF文件，里面包含了书中的截屏和图表等彩色图片，彩色图片能帮助你更好地理解输出的变化。下载网址为https://www.packtpub.com/sites/default/files/downloads/PythonMachineLearningCookbook_ColorImages.pdf。

勘误

虽然我们已尽力确保本书内容正确，但出错仍旧在所难免。如果你在书中发现错误，不管是文本还是代码，希望能告知我们，我们将不胜感激。这样做，你可以使其他读者免受挫败，也可以帮助我们改进本书的后续版本。如果你发现任何错误，请访问<http://www.packtpub.com/submit-errata>，选择本书，单击勘误表提交表单（Errata Submission Form）的链接，并输入详细说明。^①勘误一经核实，你提交的内容将被接受，此勘误会上传到本公司网站或添加到现有勘误表。

访问<https://www.packtpub.com/books/content/support>，在搜索框中输入书名，可以在勘误（Errata）部分查看已经提交的勘误信息。

盗版

任何媒体都会面临版权内容在互联网上的盗版问题，Packt也不例外。Packt非常重视版权保护。如果你发现我们的作品在互联网上被非法复制，不管以什么形式，都请立即为我们提供相关网址或网站名称，以便我们寻求补救。

请把可疑盗版材料的链接发到copyright@packtpub.com。

保护我们的作者，就是保护我们继续为你带来价值的能力，我们将不胜感激。

^① 中文版勘误可以到<http://www.ituring.com.cn/book/1894>查看和提交。——编者注

问题

如果你对本书内容存有疑问，不管是哪个方面的，都可以通过questions@packtpub.com联系我们，我们会尽最大努力解决。

电子书

扫描如下二维码，即可获得本书电子版。



目 录

第 1 章 监督学习	1
1.1 简介	1
1.2 数据预处理技术	2
1.2.1 准备工作	2
1.2.2 详细步骤	2
1.3 标记编码方法	4
1.4 创建线性回归器	6
1.4.1 准备工作	6
1.4.2 详细步骤	7
1.5 计算回归准确性	9
1.5.1 准备工作	9
1.5.2 详细步骤	10
1.6 保存模型数据	10
1.7 创建岭回归器	11
1.7.1 准备工作	11
1.7.2 详细步骤	12
1.8 创建多项式回归器	13
1.8.1 准备工作	13
1.8.2 详细步骤	14
1.9 估算房屋价格	15
1.9.1 准备工作	15
1.9.2 详细步骤	16
1.10 计算特征的相对重要性	17
1.11 评估共享单车的需求分布	19
1.11.1 准备工作	19
1.11.2 详细步骤	19
1.11.3 更多内容	21
第 2 章 创建分类器	24
2.1 简介	24
2.2 建立简单分类器	25
2.2.1 详细步骤	25
2.2.2 更多内容	27
2.3 建立逻辑回归分类器	27
2.4 建立朴素贝叶斯分类器	31
2.5 将数据集分割成训练集和测试集	32
2.6 用交叉验证检验模型准确性	33
2.6.1 准备工作	34
2.6.2 详细步骤	34
2.7 混淆矩阵可视化	35
2.8 提取性能报告	37
2.9 根据汽车特征评估质量	38
2.9.1 准备工作	38
2.9.2 详细步骤	38
2.10 生成验证曲线	40
2.11 生成学习曲线	43
2.12 估算收入阶层	45
第 3 章 预测建模	48
3.1 简介	48
3.2 用 SVM 建立线性分类器	49
3.2.1 准备工作	49
3.2.2 详细步骤	50
3.3 用 SVM 建立非线性分类器	53
3.4 解决类型数量不平衡问题	55
3.5 提取置信度	58
3.6 寻找最优超参数	60
3.7 建立事件预测器	62
3.7.1 准备工作	62
3.7.2 详细步骤	62

2 目 录

3.8 估算交通流量	64
3.8.1 准备工作	64
3.8.2 详细步骤	64
第4章 无监督学习——聚类	67
4.1 简介	67
4.2 用 k-means 算法聚类数据	67
4.3 用矢量量化压缩图片	70
4.4 建立均值漂移聚类模型	74
4.5 用凝聚层次聚类进行数据分组	76
4.6 评价聚类算法的聚类效果	79
4.7 用 DBSCAN 算法自动估算集群数量	82
4.8 探索股票数据的模式	86
4.9 建立客户细分模型	88
第5章 构建推荐引擎	91
5.1 简介	91
5.2 为数据处理构建函数组合	92
5.3 构建机器学习流水线	93
5.3.1 详细步骤	93
5.3.2 工作原理	95
5.4 寻找最近邻	95
5.5 构建一个 KNN 分类器	98
5.5.1 详细步骤	98
5.5.2 工作原理	102
5.6 构建一个 KNN 回归器	102
5.6.1 详细步骤	102
5.6.2 工作原理	104
5.7 计算欧氏距离分数	105
5.8 计算皮尔逊相关系数	106
5.9 寻找数据集中的相似用户	108
5.10 生成电影推荐	109
第6章 分析文本数据	112
6.1 简介	112
6.2 用标记解析的方法预处理数据	113
6.3 提取文本数据的词干	114
6.3.1 详细步骤	114
6.3.2 工作原理	115
6.4 用词形还原的方法还原文本的基本形式	116
6.5 用分块的方法划分文本	117
6.6 创建词袋模型	118
6.6.1 详细步骤	118
6.6.2 工作原理	120
6.7 创建文本分类器	121
6.7.1 详细步骤	121
6.7.2 工作原理	123
6.8 识别性别	124
6.9 分析句子的情感	125
6.9.1 详细步骤	126
6.9.2 工作原理	128
6.10 用主题建模识别文本的模式	128
6.10.1 详细步骤	128
6.10.2 工作原理	131
第7章 语音识别	132
7.1 简介	132
7.2 读取和绘制音频数据	132
7.3 将音频信号转换为频域	134
7.4 自定义参数生成音频信号	136
7.5 合成音乐	138
7.6 提取频域特征	140
7.7 创建隐马尔科夫模型	142
7.8 创建一个语音识别器	143
第8章 解剖时间序列和时序数据	147
8.1 简介	147
8.2 将数据转换为时间序列格式	148
8.3 切分时间序列数据	150
8.4 操作时间序列数据	152
8.5 从时间序列数据中提取统计数字	154
8.6 针对序列数据创建隐马尔科夫模型	157
8.6.1 准备工作	158
8.6.2 详细步骤	158
8.7 针对序列文本数据创建条件随机场	161
8.7.1 准备工作	161
8.7.2 详细步骤	161

8.8 用隐马尔科夫模型分析股票市场 数据.....	164	第 11 章 深度神经网络	210
第 9 章 图像内容分析	166	11.1 简介	210
9.1 简介	166	11.2 创建一个感知器	211
9.2 用 OpenCV-Python 操作图像	167	11.3 创建一个单层神经网络	213
9.3 检测边	170	11.4 创建一个深度神经网络	216
9.4 直方图均衡化	174	11.5 创建一个向量量化器	219
9.5 检测棱角	176	11.6 为序列数据分析创建一个递归 神经网络	221
9.6 检测 SIFT 特征点	178	11.7 在光学字符识别数据库中将字 符可视化	225
9.7 创建 Star 特征检测器	180	11.8 用神经网络创建一个光学字符 识别器	226
9.8 利用视觉码本和向量量化创建特征	182	第 12 章 可视化数据	230
9.9 用极端随机森林训练图像分类器	185	12.1 简介	230
9.10 创建一个对象识别器	187	12.2 画 3D 散点图	230
第 10 章 人脸识别	189	12.3 画气泡图	232
10.1 简介	189	12.4 画动态气泡图	233
10.2 从网络摄像头采集和处理视频信息	189	12.5 画饼图	235
10.3 用 Haar 级联创建一个人脸识别器	191	12.6 画日期格式的时间序列数据	237
10.4 创建一个眼睛和鼻子检测器	193	12.7 画直方图	239
10.5 做主成分分析	196	12.8 可视化热力图	241
10.6 做核主成分分析	197	12.9 动态信号的可视化模拟	242
10.7 做盲源分离	201		
10.8 用局部二值模式直方图创建一个 人脸识别器	205		

第 1 章

监督学习



在这一章，我们将介绍以下主题：

- 数据预处理技术
- 标记编码方法
- 创建线性回归器（linear regressor）
- 计算回归准确性
- 保存模型数据
- 创建岭回归器（ridge regressor）
- 创建多项式回归器（polynomial regressor）
- 估算房屋价格
- 计算特征的相对重要性
- 评估共享单车的需求分布

1.1 简介

如果你熟悉机器学习的基础知识，那么肯定知道什么是监督学习。监督学习是指在有标记的样本（labeled samples）上建立机器学习的模型。例如，如果用尺寸、位置等不同参数建立一套模型来评估一栋房子的价格，那么首先需要创建一个数据库，然后为参数打上标记。我们需要告诉算法，什么样的参数（尺寸、位置）对应什么样的价格。有了这些带标记的数据，算法就可以学会如何根据输入的参数计算房价了。

无监督学习与刚才说的恰好相反，它面对的是没有标记的数据。假设需要把一些数据分成不同的组别，但是对分组的条件毫不知情，于是，无监督学习算法就会以最合理的方式将数据集分成确定数量的组别。我们将在后面章节介绍无监督学习。

建立书中的各种模型时，将使用许多Python程序包，像NumPy、SciPy、scikit-learn、matplotlib等。如果你使用Windows系统，推荐安装兼容SciPy关联程序包的Python发行版，网址为<http://www.scipy.org/install.html>，这些Python发行版里已经集成了常用的程序包。如果你使用

Mac OS X或者Ubuntu系统，安装这些程序包就相当简单了。下面列出来程序包安装和使用文档的链接：

- ❑ NumPy: <http://docs.scipy.org/doc/numpy-1.10.1/user/install.html>
- ❑ SciPy: <http://www.scipy.org/install.html>
- ❑ scikit-learn: <http://scikit-learn.org/stable/install.html>
- ❑ matplotlib: <http://matplotlib.org/1.4.2/users/installing.html>

现在，请确保你的计算机已经安装了所有程序包。

1.2 数据预处理技术

在真实世界中，经常需要处理大量的原始数据，这些原始数据是机器学习算法无法理解的。为了让机器学习算法理解原始数据，需要对数据进行预处理。

1.2.1 准备工作

来看看Python是如何对数据进行预处理的。首先，用你最喜欢的文本编辑器打开一个扩展名为.py的文件，例如preprocessor.py。然后在文件里加入下面两行代码：

```
import numpy as np
from sklearn import preprocessing
```

我们只是加入了两个必要的程序包。接下来创建一些样本数据。向文件中添加下面这行代码：

```
data = np.array([[3, -1.5, 2, -5.4], [0, 4, -0.3, 2.1], [1, 3.3, -1.9, -4.3]])
```

现在就可以对数据进行预处理了。

1.2.2 详细步骤

数据可以通过许多技术进行预处理，接下来将介绍一些最常用的预处理技术。

1. 均值移除（Mean removal）

通常我们会把每个特征的平均值移除，以保证特征均值为0（即标准化处理）。这样做可以消除特征彼此间的偏差（bias）。将下面几行代码加入之前打开的Python文件中：

```
data_standardized = preprocessing.scale(data)
print "\nMean =", data_standardized.mean(axis=0)
print "Std deviation =", data_standardized.std(axis=0)
```

现在来运行代码。打开命令行工具，然后输入以下命令：

```
$ python preprocessor.py
```

命令行工具中将显示以下结果：

```
Mean = [ 5.55111512e-17 -1.11022302e-16 -7.40148683e-17 -7.40148683e-17]
Std deviation = [ 1. 1. 1. 1.]
```

你会发现特征均值几乎是0，而且标准差为1。

2. 范围缩放（Scaling）

数据点中每个特征的数值范围可能变化很大，因此，有时将特征的数值范围缩放到合理的大
小是非常重要的。在Python文件中加入下面几行代码，然后运行程序：

```
data_scaler = preprocessing.MinMaxScaler(feature_range=(0, 1))
data_scaled = data_scaler.fit_transform(data)
print "\nMin max scaled data =", data_scaled
```

范围缩放之后，所有数据点的特征数值都位于指定的数值范围内。输出结果如下所示：

```
Min max scaled data:
[[ 1.          0.          1.          0.          ]
 [ 0.          1.          0.41025641  1.          ]
 [ 0.33333333  0.87272727  0.          0.14666667]]
```

3. 归一化（Normalization）

数据归一化用于需要对特征向量的值进行调整时，以保证每个特征向量的值都缩放到相同的
数值范围。机器学习中最常用的归一化形式就是将特征向量调整为L1范数，使特征向量的数值之
和为1。增加下面两行代码到前面的Python文件中：

```
data_normalized = preprocessing.normalize(data, norm='l1')
print "\nL1 normalized data =", data_normalized
```

执行Python文件，就可以看到下面的结果：

```
L1      normalized      data:
[[ 0.25210084 -0.12605042  0.16806723 -0.45378151]
 [ 0.          0.625       -0.046875   0.328125  ]
 [ 0.0952381   0.31428571 -0.18095238 -0.40952381]]
```

这个方法经常用于确保数据点没有因为特征的基本性质而产生较大差异，即确保数据处于同
一数量级，提高不同特征数据的可比性。

4. 二值化（Binarization）

二值化用于将数值特征向量转换为布尔类型向量。增加下面两行代码到前面的Python文件中：

```
data_binarized = preprocessing.Binarizer(threshold=1.4).transform(data)
print "\nBinarized data =", data_binarized
```

再次执行Python文件，就可以看到下面的结果：

```
Binarized data:  
[[ 1. 0. 1. 0.]  
 [ 0. 1. 0. 1.]  
 [ 0. 1. 0. 0.]]
```

如果事先已经对数据有了一定的了解，就会发现使用这个技术的好处了。

5. 独热编码

通常，需要处理的数值都是稀疏地、散乱地分布在空间中，然而，我们并不需要存储这些大数值，这时就需要使用独热编码（One-Hot Encoding）。可以把独热编码看作是一种收紧（tighten）特征向量的工具。它把特征向量的每个特征与特征的非重复总数相对应，通过*one-of-k*的形式对每个值进行编码。特征向量的每个特征值都按照这种方式编码，这样可以更加有效地表示空间。例如，我们需要处理4维向量空间，当给一个特性向量的第n个特征进行编码时，编码器会遍历每个特征向量的第n个特征，然后进行非重复计数。如果非重复计数的值是K，那么就把这个特征转换为只有一个值是1其他值都是0的K维向量。增加下面几行代码到前面的Python文件中：

```
encoder = preprocessing.OneHotEncoder()  
encoder.fit([[0, 2, 1, 12], [1, 3, 5, 3], [2, 3, 2, 12], [1, 2, 4, 3]])  
encoded_vector = encoder.transform([[2, 3, 5, 3]]).toarray()  
print "\nEncoded vector =", encoded_vector
```

结果如下所示：

```
Encoded vector:  
[[ 0. 0. 1. 0. 0. 0. 0. 1. 1. 0.]]
```

在上面的示例中，观察一下每个特征向量的第三个特征，分别是1、5、2、4这4个不重复的值，也就是说独热编码向量的长度是4。如果你需要对5进行编码，那么向量就是[0, 1, 0, 0]。向量中只有一个值是1。第二个元素是1，对应的值是5。

1.3 标记编码方法

在监督学习中，经常需要处理各种各样的标记。这些标记可能是数字，也可能是单词。如果标记是数字，那么算法可以直接使用它们，但是，许多情况下，标记都需要以人们可理解的形式存在，因此，人们通常会用单词标记训练数据集。标记编码就是要把单词标记转换成数值形式，让算法懂得如何操作标记。接下来看看如何标记编码。

详细步骤

(1) 新建一个Python文件，然后导入preprocessing程序包：

```
from sklearn import preprocessing
```

(2) 这个程序包包含许多数据预处理需要的函数。定义一个标记编码器 (label encoder)，代码如下所示：

```
label_encoder = preprocessing.LabelEncoder()
```

(3) label_encoder对象知道如何理解单词标记。接下来创建一些标记：

```
input_classes = ['audi', 'ford', 'audi', 'toyota', 'ford', 'bmw']
```

(4) 现在就可以为这些标记编码了：

```
label_encoder.fit(input_classes)
print "\nClass mapping:"
for i, item in enumerate(label_encoder.classes_):
    print item, '-->', i
```

(5) 运行代码，命令行工具中显示下面的结果：

```
Class mapping:
audi --> 0
bmw --> 1
ford --> 2
toyota --> 3
```

(6) 就像前面结果显示的那样，单词被转换成从0开始的索引值。现在，如果你遇到一组标记，就可以非常轻松地转换它们了，如下所示：

```
labels = ['toyota', 'ford', 'audi']
encoded_labels = label_encoder.transform(labels)
print "\nLabels =", labels
print "Encoded labels =", list(encoded_labels)
```

命令行工具中将显示下面的结果：

```
Labels = ['toyota', 'ford', 'audi']
Encoded labels = [3, 2, 0]
```

(7) 这种方式比纯手工进行单词与数字的编码要简单许多。还可以通过数字反转会单词的功能检查结果的正确性：

```
encoded_labels = [2, 1, 0, 3, 1]
decoded_labels = label_encoder.inverse_transform(encoded_labels)
print "\nEncoded labels =", encoded_labels
print "Decoded labels =", list(decoded_labels)
```

结果如下所示：

```
Encoded labels = [2, 1, 0, 3, 1]
Decoded labels = ['ford', 'bmw', 'audi', 'toyota', 'bmw']
```

可以看到，映射结果是完全正确的。

1.4 创建线性回归器

回归是估计输入数据与连续值输出数据之间关系的过程。数据通常是实数形式的，我们的目标是估计满足输入到输出映射关系的基本函数。让我们从一个简单的示例开始。考虑下面的输入与输出映射关系：

$$\begin{aligned}1 &\rightarrow 2 \\3 &\rightarrow 6 \\4.3 &\rightarrow 8.6 \\7.1 &\rightarrow 14.2\end{aligned}$$

如果你要估计输入与输出的关联关系，你可以通过模式匹配轻松地找到结果。我们发现输出结果一直是输入数据的两倍，因此输入与输出的转换公式就是这样：

$$f(x) = 2x$$

这是体现输入值与输出值关联关系的一个简单函数。但是，在真实世界中通常都不会这么简单，输入与输出的映射关系函数并不是一眼就可以看出来的。

1.4.1 准备工作

线性回归用输入变量的线性组合来估计基本函数。前面的示例就是一种单输入单输出变量的线性回归。

现在考虑如图1-1所示的情况。

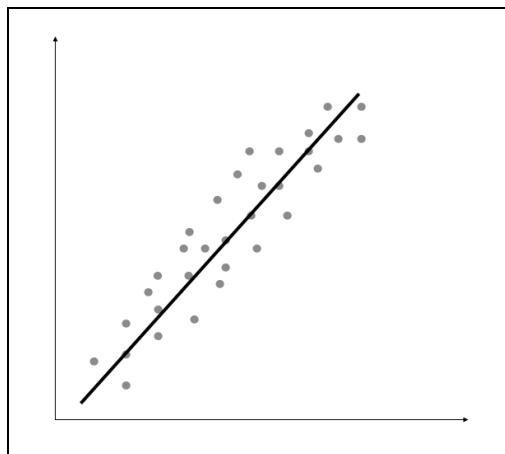


图 1-1

线性回归的目标是提取输入变量与输出变量的关联线性模型，这就要求实际输出与线性方程预测的输出的残差平方和（sum of squares of differences）最小化。这种方法被称为普通最小二乘法（Ordinary Least Squares，OLS）。

你可能觉得用一条曲线对这些点进行拟合效果会更好，但是线性回归不允许这样做。线性回归的主要优点就是方程简单。如果你想用非线性回归，可能会得到更准确的模型，但是拟合速度会慢很多。线性回归模型就像前面那张图里显示的，用一条直线近似数据点的趋势。接下来看看如何用Python建立线性回归模型。

1.4.2 详细步骤

假设你已经创建了数据文件data_singlevar.txt，文件里用逗号分隔符分割字段，第一个字段是输入值，第二个字段是与逗号前面的输入值相对应的输出值。你可以用这个文件作为输入参数。

(1) 创建一个Python文件regressor.py，然后在里面增加下面几行代码：

```
import sys
import numpy as np
filename = sys.argv[1]
X = []
y = []
with open(filename, 'r') as f:
    for line in f.readlines():
        xt, yt = [float(i) for i in line.split(',')]
        X.append(xt)
        y.append(yt)
```

把输入数据加载到变量X和y，其中X是数据，y是标记。在代码的for循环体中，我们解析每行数据，用逗号分割字段。然后，把字段转化为浮点数，并分别保存到变量X和y中。

(2) 建立机器学习模型时，需要用一种方法来验证模型，检查模型是否达到一定的满意度（satisfactory level）。为了实现这个方法，把数据分成两组：训练数据集（training dataset）与测试数据集（testing dataset）。训练数据集用来建立模型，测试数据集用来验证模型对未知数据的学习效果。因此，先把数据分成训练数据集与测试数据集：

```
num_training = int(0.8 * len(X))
num_test = len(X) - num_training

# 训练数据
X_train = np.array(X[:num_training]).reshape((num_training,1))
y_train = np.array(y[:num_training])

# 测试数据
X_test = np.array(X[num_training:]).reshape((num_test,1))
y_test = np.array(y[num_training:])
```

这里用80%的数据作为训练数据集，其余20%的数据作为测试数据集。

(3) 现在已经准备好训练模型。接下来创建一个回归器对象，代码如下所示：

```
from sklearn import linear_model  
  
# 创建线性回归对象  
linear_regressor = linear_model.LinearRegression()  
  
# 用训练数据集训练模型  
linear_regressor.fit(X_train, y_train)
```

(4) 我们利用训练数据集训练了线性回归器。向`fit`方法提供输入数据即可训练模型。用下面的代码看看它如何拟合：

```
import matplotlib.pyplot as plt  
  
y_train_pred = linear_regressor.predict(X_train)  
plt.figure()  
plt.scatter(X_train, y_train, color='green')  
plt.plot(X_train, y_train_pred, color='black', linewidth=4)  
plt.title('Training data')  
plt.show()
```

(5) 在命令行工具中执行如下命令：

```
$ python regressor.py data_singlevar.txt
```

就会看到如图1-2所示的线性回归。

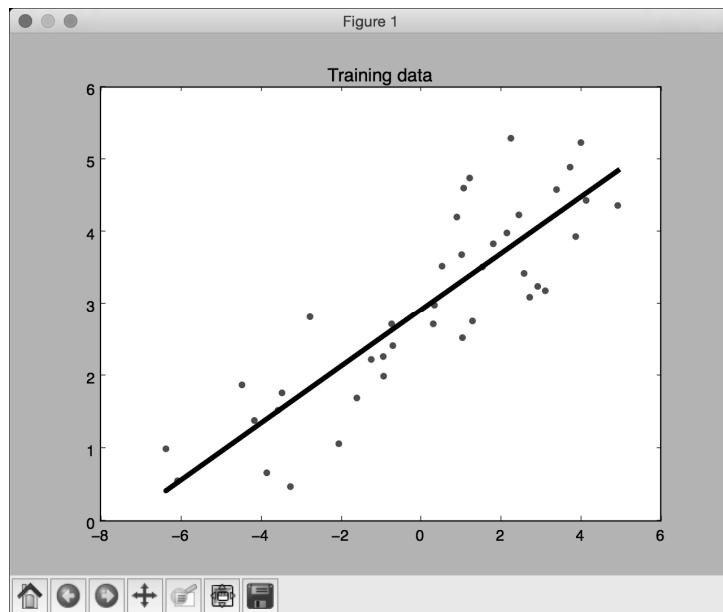


图 1-2

(6) 在前面的代码中，我们用训练的模型预测了训练数据的输出结果，但这并不能说明模型对未知的数据也适用，因为我们只是在训练数据上运行模型。这只能体现模型对训练数据的拟合效果。从图1-2中可以看到，模型训练的效果很好。

(7) 接下来用模型对测试数据集进行预测，然后画出来看看，代码如下所示：

```
y_test_pred = linear_regressor.predict(X_test)

plt.scatter(X_test, y_test, color='green')
plt.plot(X_test, y_test_pred, color='black', linewidth=4)
plt.title('Test data')
plt.show()
```

运行代码，可以看到如图1-3所示的线性回归。

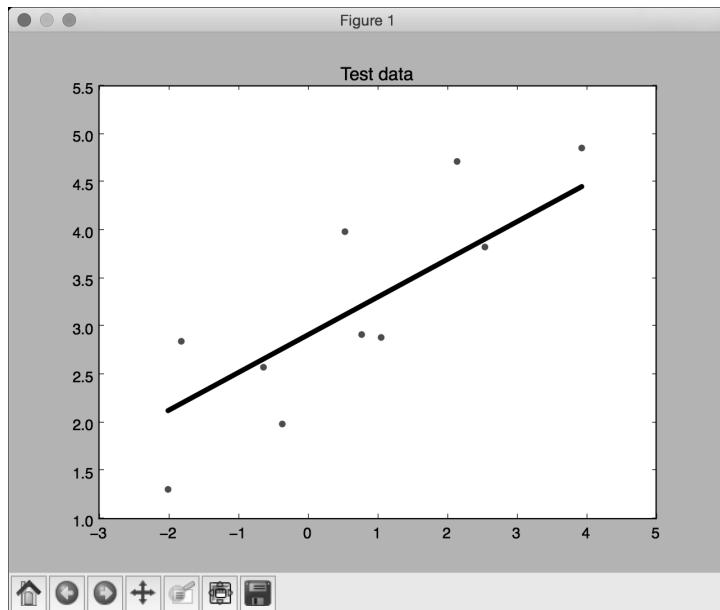


图 1-3

1.5 计算回归准确性

现在已经建立了回归器，接下来最重要的就是如何评价回归器的拟合效果。在模型评价的相关内容中，用误差（error）表示实际值与模型预测值之间的差值。

1.5.1 准备工作

下面快速了解几个衡量回归器拟合效果的重要指标（metric）。回归器可以用许多不同的指标

进行衡量，部分指标如下所示。

- **平均绝对误差 (mean absolute error)**：这是给定数据集的所有数据点的绝对误差平均值。
- **均方误差 (mean squared error)**：这是给定数据集的所有数据点的误差的平方的平均值。这是最流行的指标之一。
- **中位数绝对误差 (median absolute error)**：这是给定数据集的所有数据点的误差的中位数。这个指标的主要优点是可以消除异常值 (outlier) 的干扰。测试数据集中的单个坏点不会影响整个误差指标，均值误差指标会受到异常点的影响。
- **解释方差分 (explained variance score)**：这个分数用于衡量我们的模型对数据集波动的解释能力。如果得分1.0分，那么表明我们的模型是完美的。
- **R方得分 (R2 score)**：这个指标读作“R方”，是指确定性相关系数，用于衡量模型对未知样本预测的效果。最好的得分是1.0，值也可以是负数。

1.5.2 详细步骤

scikit-learn里面有一个模块，提供了计算所有指标的功能。重新打开一个Python文件，然后输入以下代码：

```
import sklearn.metrics as sm

print "Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2)
print "Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2)
print "Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2)
print "Explained variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2)
print "R2 score =", round(sm.r2_score(y_test, y_test_pred), 2)
```

每个指标都描述得面面俱到是非常乏味的，因此只选择一两个指标来评估我们的模型。通常的做法是尽量保证均方误差最低，而且解释方差分最高。

1.6 保存模型数据

模型训练结束之后，如果能够把模型保存成文件，那么下次再使用的时候，只要简单地加载就可以了。

详细步骤

用程序保存模型的具体操作步骤如下。

(1) 在Python文件regressor.py中加入以下代码：

```
import cPickle as pickle

output_model_file = 'saved_model.pkl'
with open(output_model_file, 'w') as f:
    pickle.dump(linear_regressor, f)
```

(2) 回归模型会保存在saved_model.pkl文件中。下面看看如何加载并使用它，代码如下所示：

```
with open(output_model_file, 'r') as f:
    model_linregr = pickle.load(f)

y_test_pred_new = model_linregr.predict(X_test)
print "\nNew mean absolute error =", round(sm.mean_absolute_error(y_test,
y_test_pred_new), 2)
```

(3) 这里只是把回归模型从Pickle文件加载到model_linregr变量中。你可以将打印结果与前面的结果进行对比，确认模型与之前的一样。

1.7 创建岭回归器

线性回归的主要问题是异常值敏感。在真实世界的数据收集过程中，经常会遇到错误的度量结果。而线性回归使用的普通最小二乘法，其目标是使平方误差最小化。这时，由于异常值误差的绝对值很大，因此会引起问题，从而破坏整个模型。

1.7.1 准备工作

先看图1-4。

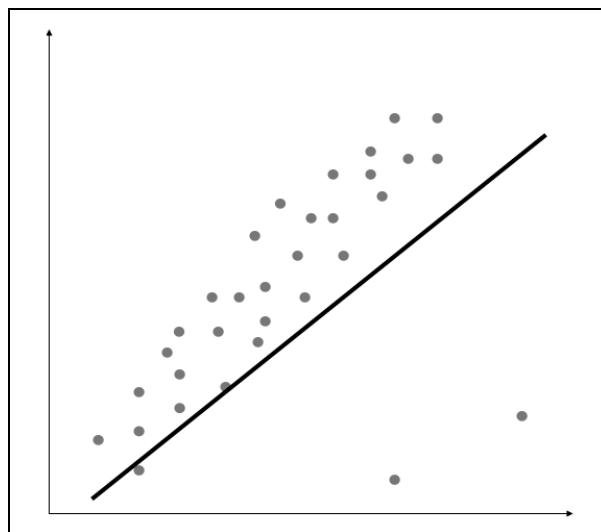


图 1-4

右下角的两个数据点明显是异常值，但是这个模型需要拟合所有的数据点，因此导致整个模型都错了。仅凭直觉观察，我们就会觉得如图1-5的拟合结果更好。

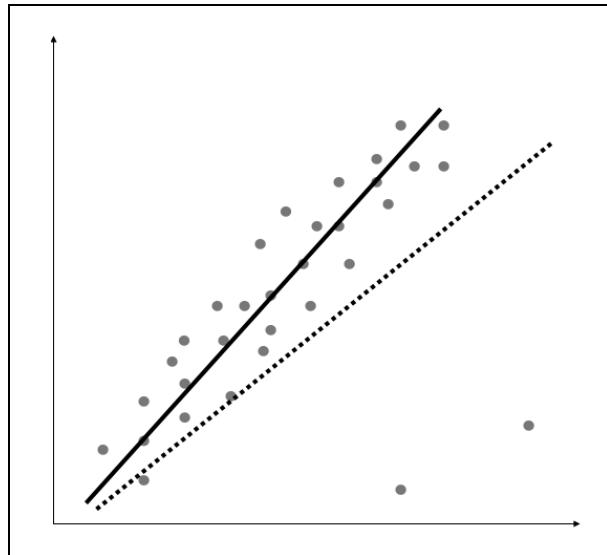


图 1-5

普通最小二乘法在建模时会考虑每个数据点的影响，因此，最终模型就会像图1-4显示的直线那样。显然，我们发现这个模型不是最优的。为了避免这个问题，我们引入正则化项的系数作为阈值来消除异常值的影响。这个方法被称为岭回归。

1.7.2 详细步骤

接下来看看如何用Python建立岭回归器。

(1) 你可以从data_multi_variable.txt文件中加载数据。这个文件的每一行都包含多个数值。除了最后一个数值外，前面的所有数值构成输入特征向量。

(2) 把下面的代码加入regressor.py文件中。我们用一些参数初始化岭回归器：

```
ridge_regressor = linear_model.Ridge(alpha=0.01, fit_intercept=True,  
max_iter=10000)
```

(3) alpha参数控制回归器的复杂程度。当alpha趋于0时，岭回归器就是用普通最小二乘法的线性回归器。因此，如果你希望模型对异常值不那么敏感，就需要设置一个较大的alpha值。这里把alpha值设置为0.01。

(4) 下面让我们来训练岭回归器。

```
ridge_regressor.fit(X_train, y_train)
y_test_pred_ridge = ridge_regressor.predict(X_test)
print "Mean absolute error =", round(sm.mean_absolute_error
    (y_test, y_test_pred_ridge), 2)
print "Mean squared error =", round(sm.mean_squared_error
    (y_test, y_test_pred_ridge), 2)
print "Median absolute error =", round(sm.median_absolute_error
    (y_test, y_test_pred_ridge), 2)
print "Explain variance score =", round(sm.explained_variance_score
    (y_test, y_test_pred_ridge), 2)
print "R2 score =", round(sm.r2_score(y_test, y_test_pred_ridge), 2)
```

运行代码检查误差指标。可以用同样的数据建立一个线性回归器，并与岭回归器的结果进行比较，看看把正则化引入回归模型之后的效果如何。

1.8 创建多项式回归器

线性回归模型有一个主要的局限性，那就是它只能把输入数据拟合成直线，而多项式回归模型通过拟合多项式方程来克服这类问题，从而提高模型的准确性。

1.8.1 准备工作

先看图1-6。

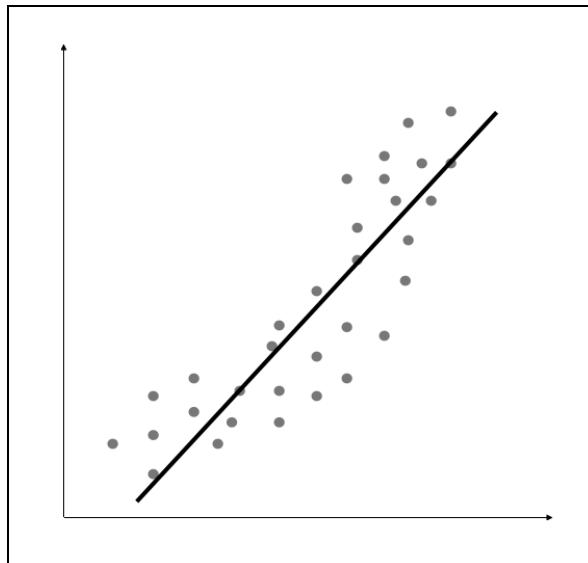


图 1-6

从图1-6中可以看到，数据点本身的模式中带有自然的曲线，而线性模型是不能捕捉到这一点的。再来看看多项式模型的效果，如图1-7所示。

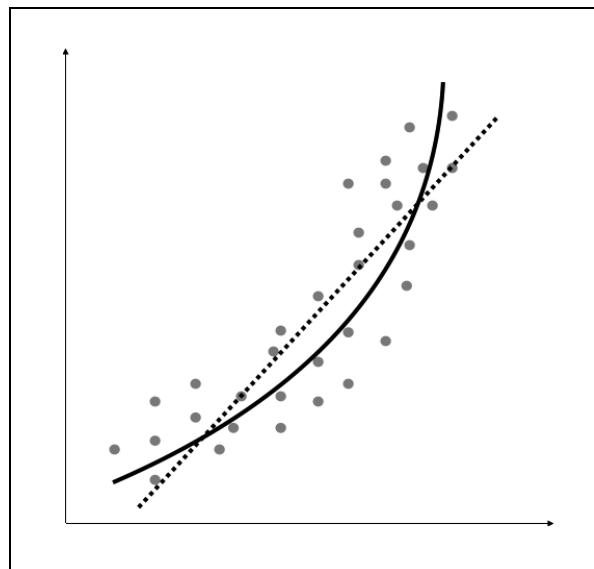


图 1-7

图1-7中的虚线表示线性回归模型，实线表示多项式回归模型。这个模型的曲率是由多项式的次数决定的。随着模型曲率的增加，模型变得更准确。但是，增加曲率的同时也增加了模型的复杂性，因此拟合速度会变慢。当我们对模型的准确性的理想追求与计算能力限制的残酷现实发生冲突时，就需要综合考虑了。

1.8.2 详细步骤

(1) 将下面的代码加入Python文件regressor.py中：

```
from sklearn.preprocessing import PolynomialFeatures  
  
polynomial = PolynomialFeatures(degree=3)
```

(2) 上一行将曲线的多项式的次数的初始值设置为3。下面用数据点来计算多项式的参数：

```
x_train_transformed = polynomial.fit_transform(x_train)
```

其中，`x_train_transformed`表示多项式形式的输入，与线性回归模型是一样大的。

(3) 接下来用文件中的第一个数据点来检查多项式模型是否能够准确预测：

```
datapoint = [0.39, 2.78, 7.11]
poly_datapoint = polynomial.fit_transform(datapoint)

poly_linear_model = linear_model.LinearRegression()
poly_linear_model.fit(X_train_transformed, y_train)
print "\nLinear regression:", linear_regressor.predict(datapoint) [0]
print "\nPolynomial regression:", poly_linear_model.predict(poly_datapoint) [0]
```

多项式回归模型计算变量数据点的值恰好就是输入数据文件中的第一行数据值。再用线性回归模型测试一下，唯一的差别就是展示数据的形式。运行代码，可以看到下面的结果：

```
Linear regression: -11.0587294983
Polynomial regression: -10.9480782122
```

可以发现，多项式回归模型的预测值更接近实际的输出值。如果想要数据更接近实际输出值，就需要增加多项式的次数。

(4) 将多项式的次数加到10看看结果：

```
polynomial = PolynomialFeatures(degree=10)
```

可以看到下面的结果：

```
Polynomial regression: -8.20472183853
```

现在，你可以发现预测值与实际的输出值非常地接近。

1.9 估算房屋价格

是时候用所学的知识来解决真实世界的问题了。让我们用这些原理来估算房屋价格。房屋估价是理解回归分析最经典的案例之一，通常是一个不错的切入点。它符合人们的直觉，而且与人们的生活息息相关，因此在用机器学习处理复杂事情之前，通过房屋估价可以更轻松地理解相关概念。我们将使用带AdaBoost算法的决策树回归器（decision tree regressor）来解决这个问题。

1.9.1 准备工作

决策树是一个树状模型，每个节点都做出一个决策，从而影响最终结果。叶子节点表示输出数值，分支表示根据输入特征做出的中间决策。AdaBoost算法是指自适应增强（adaptive boosting）算法，这是一种利用其他系统增强模型准确性的技术。这种技术是将不同版本的算法结果进行组合，用加权汇总的方式获得最终结果，被称为弱学习器（weak learners）。AdaBoost算法在每个阶段获取的信息都会反馈到模型中，这样学习器就可以在后一阶段重点训练难以分类的样本。这种学习方式可以增强系统的准确性。

首先使用AdaBoost算法对数据集进行回归拟合，再计算误差，然后根据误差评估结果，用同样的数据集重新拟合。可以把这些看作是回归器的调优过程，直到达到预期的准确性。假设你拥

有一个包含影响房价的各种参数的数据集，我们的目标就是估计这些参数与房价的关系，这样就可以根据未知参数估计房价了。

1.9.2 详细步骤

(1) 创建一个新的Python文件housing.py，然后加入下面的代码：

```
import numpy as np
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn import datasets
from sklearn.metrics import mean_squared_error, explained_variance_score
from sklearn.utils import shuffle
import matplotlib.pyplot as plt
```

(2) 网上有一个标准房屋价格数据库，人们经常用它来研究机器学习。你可以在<https://archive.ics.uci.edu/ml/datasets/Housing>下载数据。不过scikit-learn提供了数据接口，可以直接通过下面的代码加载数据：

```
housing_data = datasets.load_boston()
```

每个数据点由影响房价的13个输入参数构成。你可以用housing_data.data获取输入的数据，用housing_data.target获取对应的房屋价格。

(3) 接下来把输入数据与输出结果分成不同的变量。我们可以通过shuffle函数把数据的顺序打乱：

```
x, y = shuffle(housing_data.data, housing_data.target, random_state=7)
```

(4) 参数random_state用来控制如何打乱数据，让我们可以重新生成结果。接下来把数据分成训练数据集和测试数据集，其中80%的数据用于训练，剩余20%的数据用于测试：

```
num_training = int(0.8 * len(X))
X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]
```

(5) 现在已经可以拟合一个决策树回归模型了。选一个最大深度为4的决策树，这样可以限制决策树不变成任意深度：

```
dt_regressor = DecisionTreeRegressor(max_depth=4)
dt_regressor.fit(X_train, y_train)
```

(6) 再用带AdaBoost算法的决策树回归模型进行拟合：

```
ab_regressor = AdaBoostRegressor(DecisionTreeRegressor(max_depth=4),
n_estimators=400, random_state=7)
ab_regressor.fit(X_train, y_train)
```

这样可以帮助我们对比训练效果，看看AdaBoost算法对决策树回归器的训练效果有多大改善。

(7) 接下来评价决策树回归器的训练效果：

```
y_pred_dt = dt_regressor.predict(X_test)
mse = mean_squared_error(y_test, y_pred_dt)
evs = explained_variance_score(y_test, y_pred_dt)
print "\n#### Decision Tree performance ####"
print "Mean squared error =", round(mse, 2)
print "Explained variance score =", round(evs, 2)
```

(8) 现在评价一下AdaBoost算法改善的效果：

```
y_pred_ab = ab_regressor.predict(X_test)
mse = mean_squared_error(y_test, y_pred_ab)
evs = explained_variance_score(y_test, y_pred_ab)
print "\n#### AdaBoost performance ####"
print "Mean squared error =", round(mse, 2)
print "Explained variance score =", round(evs, 2)
```

(9) 命令行工具显示的输出结果如下所示：

```
#### 决策树学习效果 ####
Mean squared error = 14.79
Explained variance score = 0.82

#### AdaBoost算法改善效果 ####
Mean squared error = 7.54
Explained variance score = 0.91
```

前面的结果表明，AdaBoost算法可以让误差更小，且解释方差分更接近1。

1.10 计算特征的相对重要性

所有特征都同等重要吗？在这个案例中，我们用了13个特征，它们对模型都有贡献。但是，有一个重要的问题出现了：如何判断哪个特征更加重要？显然，所有的特征对结果的贡献是不一样的。如果需要忽略一些特征，就需要知道哪些特征不太重要。scikit-learn里面有这样的功能。

详细步骤

(1) 画出特征的相对重要性，在housing.py文件中加入下面几行代码：

```
plot_feature_importances(dt_regressor.feature_importances_,
    'Decision Tree regressor', housing_data.feature_names)
plot_feature_importances(ab_regressor.feature_importances_,
    'AdaBoost regressor', housing_data.feature_names)
```

回归器对象有一个feature_importances_方法会告诉我们每个特征的相对重要性。

(2) 接下来需要定义plot_feature_importances来画出条形图：

```
def plot_feature_importances(feature_importances, title, feature_names):
    # 将重要性值标准化
    feature_importances = 100.0 * (feature_importances / max(feature_importances))

    # 将得分从高到低排序
    index_sorted = np.flipud(np.argsort(feature_importances))

    # 让X坐标轴上的标签居中显示
    pos = np.arange(index_sorted.shape[0]) + 0.5

    # 画条形图
    plt.figure()
    plt.bar(pos, feature_importances[index_sorted], align='center')
    plt.xticks(pos, feature_names[index_sorted])
    plt.ylabel('Relative Importance')
    plt.title(title)
    plt.show()
```

(3) 我们从feature_importances方法里取值，然后把数值放大到0~100的范围内。运行前面的代码，可以看到两张图（不带AdaBoost算法与带AdaBoost算法两种模型）。仔细观察图1-8和图1-9，看看能从决策树回归器中获得什么。

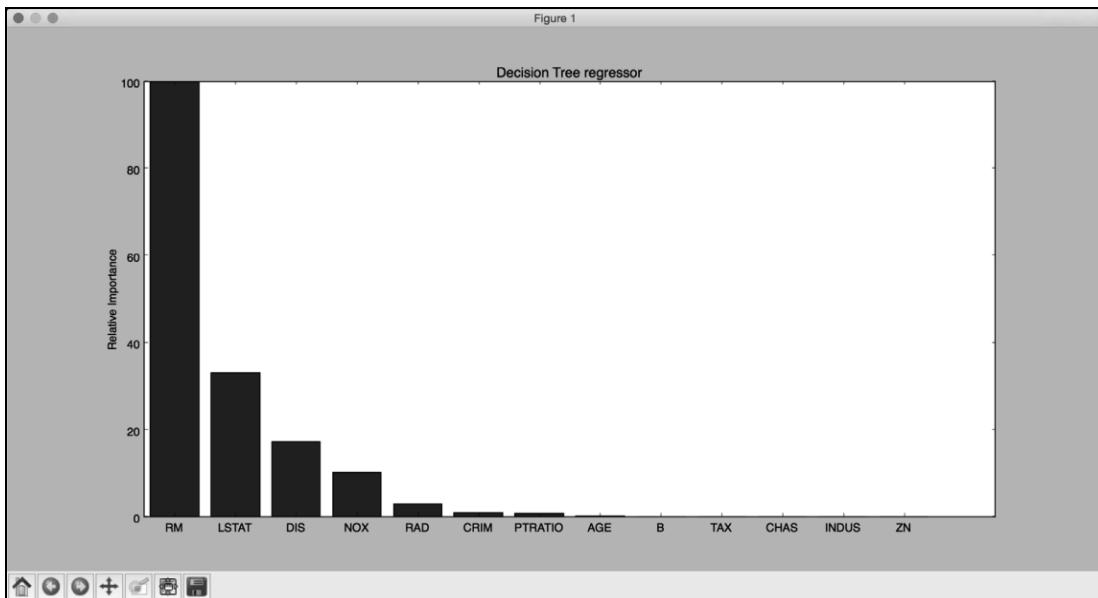


图 1-8

(4) 从图1-8可以发现，不带AdaBoost算法的决策树回归器显示的最重要特征是RM。再看看带AdaBoost算法的决策树回归器的特征重要性排序条形图，如图1-9所示。

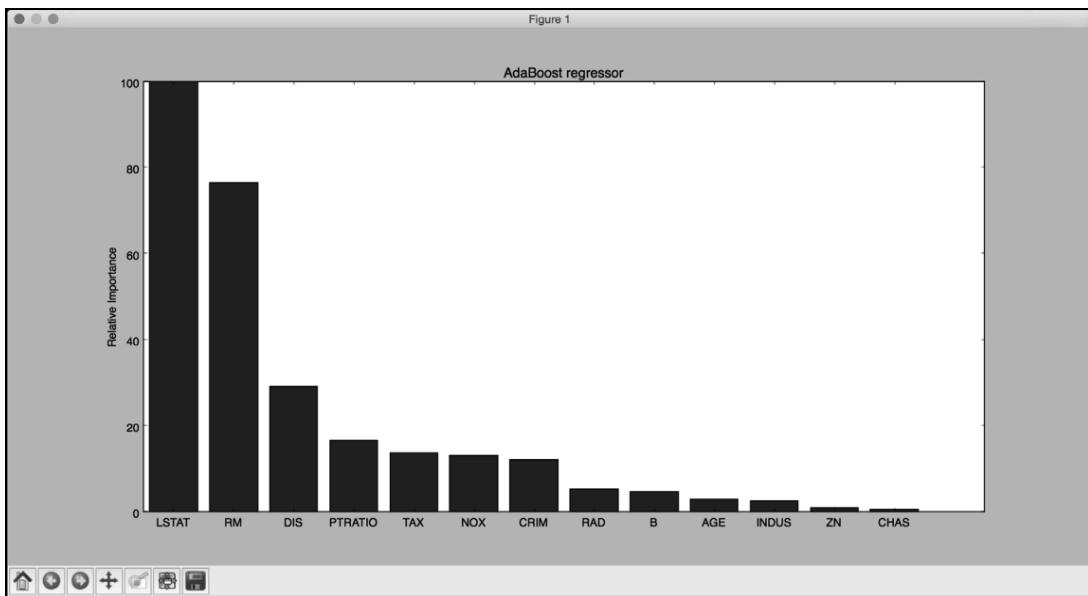


图 1-9

加入AdaBoost算法之后，房屋估价模型的最重要特征是LSTAT。在现实生活中，如果对这个数据集建立不同的回归器，就会发现最重要的特征是LSTAT，这足以体现AdaBoost算法对决策树回归器训练效果的改善。

1.11 评估共享单车的需求分布

本节将用一种新的回归方法解决共享单车的需求分布问题。我们采用随机森林回归器（random forest regressor）估计输出结果。随机森林是一个决策树集合，它基本上就是用一组由数据集的若干子集构建的决策树构成，再用决策树平均值改善整体学习效果。

1.11.1 准备工作

我们将使用bike_day.csv文件中的数据集，它可以在 <https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset> 获取。这份数据集一共16列，前两列是序列号与日期，分析的时候可以不用；最后三列数据是不同类型的输出结果；最后一列是第十四列与第十五列的和，因此建立模型时可以不考虑第十四列与第十五列。

1.11.2 详细步骤

接下来看看Python如何解决这个问题。如果你下载了本书源代码，就可以看到bike_sharing.py

文件里已经包含了完整代码。这里将介绍若干重要的部分。

(1) 首先导入一些新的程序包，如下：

```
import csv
from sklearn.ensemble import RandomForestRegressor
from housing import plot_feature_importances
```

(2) 我们需要处理CSV文件，因此加入了csv程序包来读取CSV文件。由于这是一个全新的数据集，因此需要自己定义一个数据集加载函数：

```
def load_dataset(filename):
    file_reader = csv.reader(open(filename, 'rb'), delimiter=',')
    X, y = [], []
    for row in file_reader:
        X.append(row[2:13])
        y.append(row[-1])

    # 提取特征名称
    feature_names = np.array(X[0])

    # 将第一行特征名称移除，仅保留数值
    return np.array(X[1:]).astype(np.float32), np.array(y[1:]).astype(np.float32),
    feature_names
```

在这个函数中，我们从CSV文件读取了所有数据。把数据显示在图形中时，特征名称非常有用。把特征名称数据从输入数值中分离出来，并作为函数返回值。

(3) 读取数据，并打乱数据顺序，让新数据与原来文件中数据排列的顺序没有关联性：

```
X, y, feature_names = load_dataset(sys.argv[1])
X, y = shuffle(X, y, random_state=7)
```

(4) 和之前的做法一样，需要将数据分成训练数据和测试数据。这一次，我们将90%的数据用于训练，剩余10%的数据用于测试：

```
num_training = int(0.9 * len(X))
X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]
```

(5) 下面开始训练回归器：

```
rf_regressor = RandomForestRegressor(n_estimators=1000, max_depth=10,
min_samples_split=1)
rf_regressor.fit(X_train, y_train)
```

其中，参数n_estimators是指评估器（estimator）的数量，表示随机森林需要使用的决策树数量；参数max_depth是指每个决策树的最大深度；参数min_samples_split是指决策树分裂一个节点需要用到的最小数据样本量。

(6) 评价随机森林回归器的训练效果：

```
y_pred = rf_regressor.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
evs = explained_variance_score(y_test, y_pred)
print "\n#### Random Forest regressor performance ####"
print "Mean squared error =", round(mse, 2)
print "Explained variance score =", round(evs, 2)
```

(7) 由于已经有画出特征重要性条形图的函数plot_feature_importances了，接下来直接调用它：

```
plot_feature_importances(rf_regressor.feature_importances_, 'Random Forest
regressor', feature_names)
```

执行代码，可以看到如图1-10所示的图形。

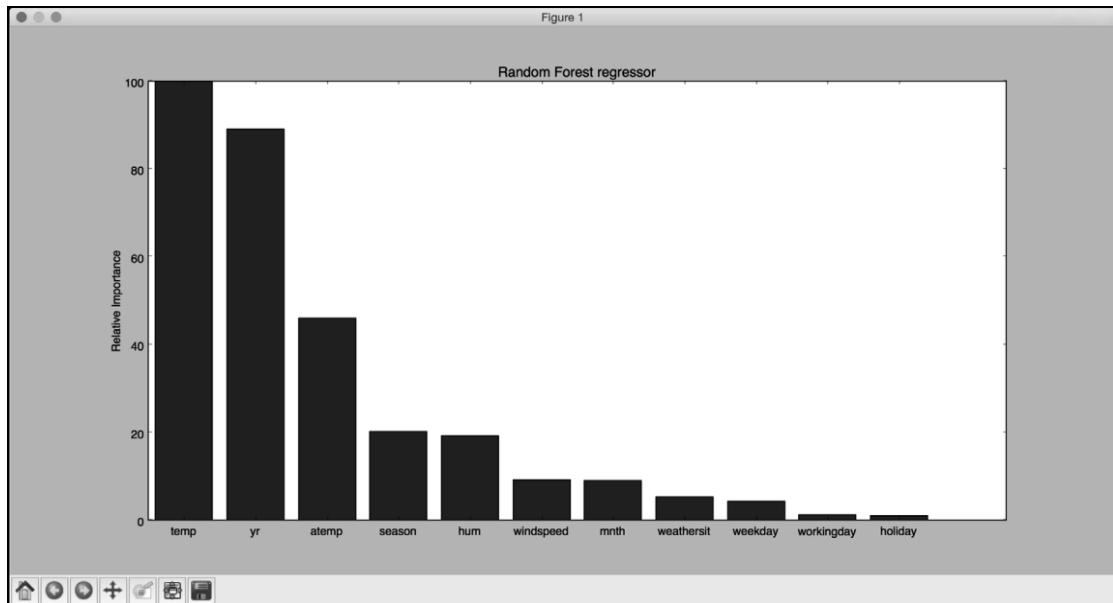


图 1-10

看来温度（temp）是影响自行车租赁的最重要因素。

1.11.3 更多内容

把第十四列与第十五列数据加入数据集，看看结果有什么区别。在新的特征重要性条形图中，除了这两个特征外，其他特征都变成了0。这是由于输出结果可以通过简单地对第十四列与第十五列数据求和得出，因此算法不需要其他特征计算结果。在数据集加载函数load_dataset中，

我们需要对for循环内的取值范围稍作调整：

```
X.append(row[2:15])
```

现在再画出特征重要性条形图，可以看到如图1-11所示的柱形图。

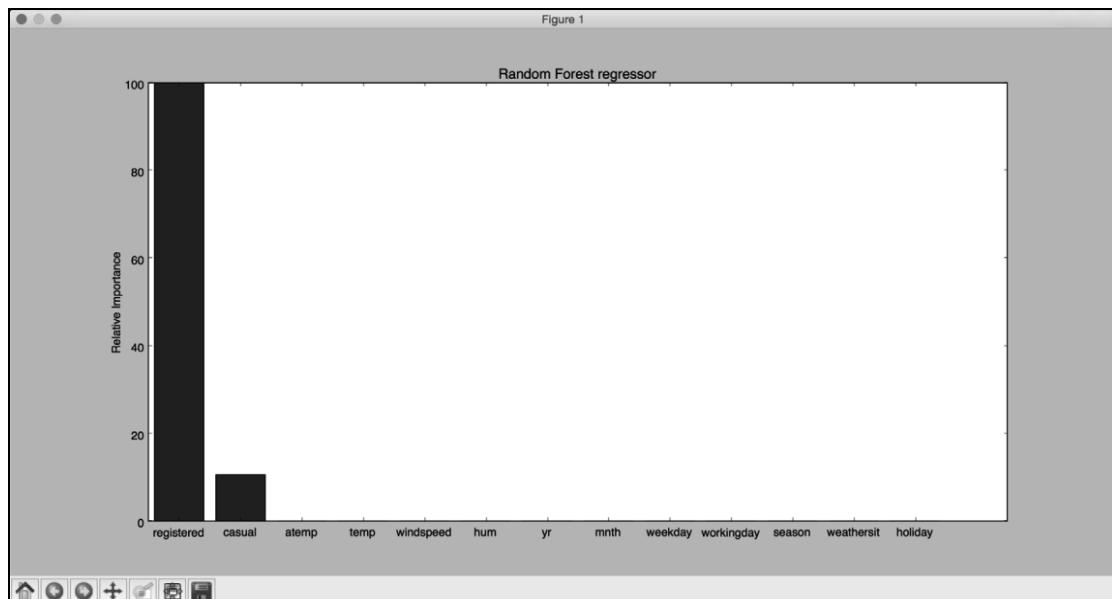


图 1-11

与预想的一样，从图中可以看出，只有这两个特征是重要的，这确实也符合常理，因为最终结果仅仅是这两个特征相加得到的。因此，这两个变量与输出结果有直接的关系，回归器也就认为它不需要其他特征来预测结果了。在消除数据集冗余变量方面，这是非常有用的工具。

还有一份按小时统计的自行车共享数据bike_hour.csv。我们需要用到第3~14列，因此先对数据集加载函数load_dataset做一点调整：

```
X.append(row[2:14])
```

运行代码，可以看到回归器的训练结果如下：

```
#### 随机森林学习效果 ####
Mean squared error = 2619.87
Explained variance score = 0.92
```

特征重要性条形图如图1-12所示。

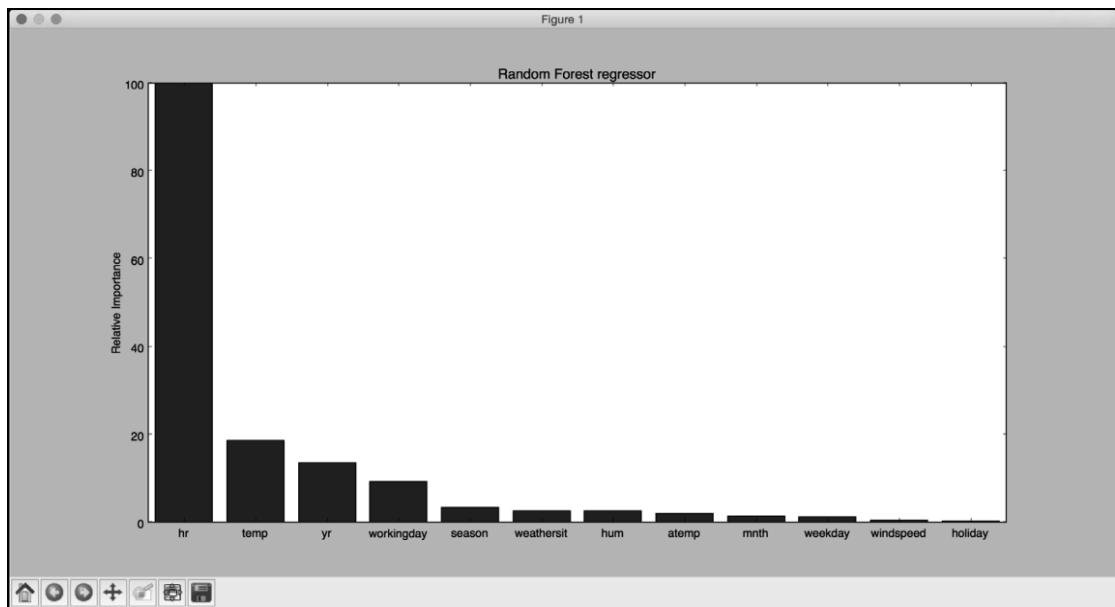


图 1-12

图1-12中显示，最重要的特征是一天中的不同时点（hr），这也完全符合人们的直觉；其次重要的是温度，与我们之前分析的结果一致。



微信连接



回复“Python”查看相关书单



微博连接

关注@图灵教育 每日分享IT好书



QQ连接

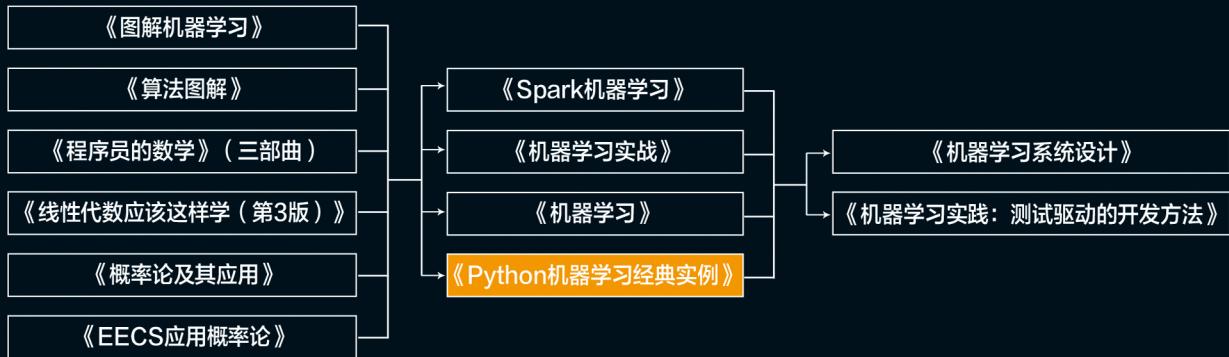
图灵读者官方群I: 218139230

图灵读者官方群II: 164939616

图灵社区 iTuring.cn

在线出版，电子书，《码农》杂志，图灵访谈

- 监督学习技术、预测建模、无监督学习算法等前沿话题的实例代码展示
- 来自Kaggle的经典数据集和机器学习案例
- 用流行的Python库scikit-learn解决机器学习问题



[PACKT]
PUBLISHING

图灵社区: iTuring.cn
热线: (010)51095186转600

分类建议 计算机/程序设计/Python

人民邮电出版社网址: www.ptpress.com.cn



欢迎加入

图灵社区

最前沿的IT类电子书发售平台

电子出版的时代已经来临。在许多出版界同行还在犹豫彷徨的时候，图灵社区已经采取实际行动拥抱这个出版业巨变。作为国内第一家发售电子图书的IT类出版商，图灵社区目前为读者提供两种DRM-free的阅读体验：在线阅读和PDF。

相比纸质书，电子书具有许多明显的优势。它不仅发布快，更新容易，而且尽可能采用了彩色图片（即使有的书纸质版是黑白印刷的）。读者还可以方便地进行搜索、剪贴、复制和打印。

最方便的开放出版平台

图灵社区向读者开放在线写作功能，协助你实现自出版和开源出版的梦想。利用“合集”功能，你就能联合二三好友共同创作一部技术参考书，以免费或收费的形式提供给读者。（收费形式须经过图灵社区立项评审。）这极大地降低了出版的门槛。只要你有写作的意愿，图灵社区就能帮助你实现这个梦想。成熟的书稿，有机会入选出版计划，同时出版纸质书。

图灵社区引进出版的外文图书，都将在立项后马上在社区公布。如果你有意翻译哪本图书，欢迎你来社区申请。只要你通过试译的考验，即可签约成为图灵的译者。当然，要想成功地完成一本书的翻译工作，是需要有坚强的毅力的。

图灵社区进一步把传统出版流程与电子书出版业务紧密结合，目前已实现作译者网上交稿、编辑网上审稿、按章发布的电子出版模式。这种新的出版模式，我们称之为“敏捷出版”，它可以让读者以较快的速度了解到国外最新技术图书的内容，弥补以往翻译版技术书“出版即过时”的缺憾。同时，敏捷出版使得作、译、编、读的交流更为方便，可以提前消灭书稿中的错误，最大程度地保证图书出版的质量。

最直接的读者交流平台

在图灵社区，你可以十分方便地写作文章、提交勘误、发表评论，以各种方式与作译者、编辑人员和其他读者进行交流互动。提交勘误还能够获赠社区银子。

你可以积极参与社区经常开展的访谈、审读、评选等多种活动，赢取积分和银子，积累个人声望。

ituring.com.cn