新工科建设之路 · 数据科学与大数据系列
新形态教材 · 数据工程师成长之路

数据准备和特征工程

——数据工程师必知必会技能

齐 伟 编著

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

批注+视频+交互
让学习更容易

数据准备和特征工程

——数据工程师必知必会技能

本书详细地介绍了大数据、人工智能等项目中不可或缺的环节和内容：数据准备和特征工程。书中的每节首先以简明方式介绍了基本知识；然后通过实际案例演示了基本知识的实际应用，并提供了针对性练习项目，将“知识、案例、练习”融为一体；最后以“扩展探究”方式引导读者进入更深广的领域。

本书既适合作为大学相关专业的教材，也适合作为大数据、人工智能等领域的开发人员的参考读物。

- 画龙点睛的批注，让学习更加简单
- 案例式教学，面向工程实践
- 渗透技术发展，基础与前沿结合
- 提供在线实验平台，学练融合
- 在专属公众号与作者对话，教学相长
- 配套视频，边看边学

责任编辑：章海涛
封面设计：田晨晨

ISBN 978-7-121-38263-
9 787121 382635 >
定价：45.00元

新工科建设之路·数据科学与大数据系列
新形态教材·数据工程师成长之路

数据准备和特征工程

——数据工程师必知必会技能

文 佳 编著

电子工业出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

本书详细地介绍了大数据、人工智能等项目中不可或缺的环节和内容：数据准备和特征工程。书中的每节首先以简明方式介绍了基本知识；然后通过实际案例演示了基本知识的实际应用，并提供了针对性练习项目，将“知识、案例、练习”融为一体；最后以“扩展探究”方式引导读者进入更深广的领域。

本书既适合作为大学相关专业的教材，也适合作为大数据、人工智能等领域的开发人员的参考读物。

图书在版编目（CIP）数据

数据准备和特征工程：数据工程师必知必会技能 / 齐伟编著 . —北京：电子工业出版社，2020.3
ISBN 978-7-121-38263-5

Ⅰ . ① 数… Ⅱ . ① 齐… Ⅲ . ① 数据处理—高等学校—教材 ② 人工智能—高等学校—教材
Ⅳ . ① TP274 ② TP18

中国版本图书馆 CIP 数据核字（2020）第 020000 号

责任编辑：章海涛　　特约编辑：张燕虹
印　　刷：北京虎彩文化传播有限公司
装　　订：北京虎彩文化传播有限公司
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 × 1 092　1/16　印张：13　字数：332 千字
版　　次：2020 年 3 月第 1 版
印　　次：2020 年 3 月第 1 次印刷
定　　价：45.00 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：192910558（QQ 群）。

前言 Preface

在计算机科学中，有一句名言：“Garbage in, garbage out”(GIGO)。这句话用到数据科学上也同样成立。另外，数据科学业界中还流传着另一句话：“数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已。”

除了“名言”，很多数据科学实践者的项目经验也一再证明高质量的数据永远是排在第一位的。

然而，现实世界的数据存在不完整、噪声、不一致、错误值、离群值、重复等问题。不仅如此，数据集的特征也是形形色色的，有的特征与项目无关，有的特征彼此强相关，还有的数据集因为特征太多而导致耗费极大的计算资源。诸如此类现象，可以概括为一句话：“理想很丰满，现实很骨感。”

因此，数据准备和特征工程的工作就成为数据科学项目中不可或缺的环节，每个从业者必须熟练掌握相关操作技能，并能耐心地从事这项工作。实践经验表明，数据准备和特征工程会占用项目开发的绝大部分时间。

本书相对于已有的类似书籍而言，在以下方面更具有特色。

- 强调工程实践，这也是本书作者所有书籍的共同特点。书中通过大量案例，向读者演示了各种方法的具体实现方式。
- 基础与前沿结合。虽然本书在“基础知识”中介绍了相关的基本实现方法，但因为现实项目的复杂性，在具体项目中还会用到各种工具及最新的研发成果，为此专设了“扩展探究”供读者了解更精彩的内容。
- 以案例为载体，传授思想方法。数据科学项目需要严谨、科学的思想方法，这些方法并非通过简单说教就能让读者掌握，本书以“项目案例”为载体，不仅讲述操作技法，而且还让读者体验其中的思想方法，并且在“动手练习”中提供了练习项目，供读者检验和巩固所学内容。

为了给读者使用本书提供更多的资源支持，在此推荐本书作者的微信公众号：老齐教室。通过此微信公众号，可以得到如下资源：

- 使用本书配套的在线实验平台。在实验平台中，读者可以运行本书的所有源码，应用书中所要求的数据集。
- 观看本书配套的视频课程。
- 及时获得本书的勘误内容。
- 阅读与本书相关的其他技术资料。
- 与本书的作者及其他读者进行专业交流。

非常感谢为本书的出版而辛苦工作的各位编辑。

书中内容难免错误，恳请读者不吝赐教。

齐 伟

目录 Contents

① 注：因没有需要介绍的基本概念和术语，故从 5.1 开始。

第1章 感知数据

数据（Data）是很常见的名词，如“大数据”“数据驱动决策”“数据科学”等。也正是因为常用，所以很难以某种公认的表述定义它。读者在本章可以通过若干示例，初步了解数据的可能来源和读取数据常规方法，亲身体验数据的存在，建立对数据的初步感知。

大数据（Big data）通常指传统软件不足以处理的数据集。随着技术的进步，“大数据”中“大”的标准也在发生变化。

第 1 章知识结构如图 1-0-0 所示。

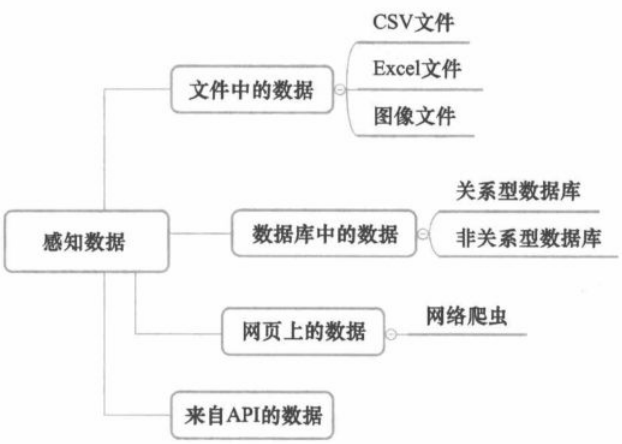


图 1-0-0 第 1 章知识结构

1.0 了解数据科学项目

什么是数据科学？业界对此定义多有不同，在此引用“维基百科”网站的“数据科学”词条部分内容：

“维基百科”中内容丰富，推荐读者参考。

“数据科学是一门利用数据学习知识的科学，其目标是通过从数据中提取有价值的部分来生产数据产品。它结合了诸多领域中的理论和技术，包括应用数学、统计、模式识别、机器学习、数据可视化、数据仓库等。”

根据这个定义，“数据科学项目”就应该是“生产数据产品”的工程实践，本书内容就是这个工程实践的一部分。

工程，常被认为“工人”按照“操作手册”执行既定操作。的确有的工程如此，那么数据科学的工程是否也有一份“操作手册”呢？本书作者总结的数据科学项目基本过程如图 1-0-1 所示。

扫描二维码，获得本章学习资源

在解释图 1-0-1 之前，先要建立如下认识——这些认识都来自实践中的经验教训：

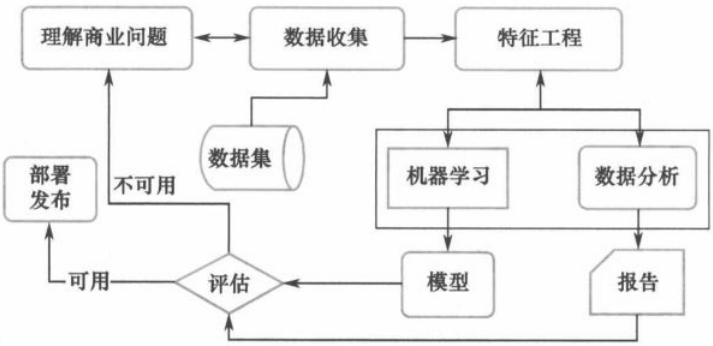


图 1-0-1　本书作者总结的数据科学项目过程

- 对于任何一个实践中的项目，不要寄希望于“操作手册”，更不要迷信著名专家的“名人名言”。
- 不存在“一招鲜吃遍天”，普遍真理是“具体问题具体分析”。

貌似上述认识否定了数据科学项目中的规律。非也！

数据科学项目在实践上没有固定的“操作手册”，但还是要遵循一定章法。这就类似于个人成长发展，每个人的秉性、才智不同，成才道路各有特色，但所有人都遵循着灵长目—人科—人属—智人种生物的普遍生长发育规律。

对于图 1-0-1，不同的研究者会有不同的见解，本书作者也仅是根据个人经验给予简述。

1. 理解商业问题

理解商业问题不同于理解项目需求，前者更强调对项目本质问题的把握。

这是数据科学项目的开始，参与者必须对相应的业务有所了解，并将业务中用描述性语言表述的问题转为“数据问题”——能够通过某些数据回答的问题。

2. 数据收集

“数据收集”与“理解商业问题”两者是互动关系。研究收集数据的方法，也是对商业问题的再度理解。例如，一个能够对学生学习过程进行评价和指导的系统所需要收集的数据包括但不限于学生写在作业本上的数据、学校服务器上的学籍数据、公安系统中学生的家庭和社区相关数据等。随着对此问题的深入研究，需要收集的数据还可能根据需求而变化。

运用技术从数据来源获得的数据通常称为“原始数据”。本章在后续各节中将介绍几种常见的数据收集方法。此外，因为数据源和技术的不同，所得到的原始数据格式、质量也会有很大差异，还可能存在重复、矛盾等。因此，必须处理这些数据。与此相关的技术构成了本书的主体内容。

3. 特征工程

特征工程是一个比较广泛的概念，它的起点是得到原始数据之后，终点是进行机器学习或数据分析之前。在这个过程中对数据集所做的操作统称为

“特征工程”——这是本书的界定，业界对“特征工程”有多种定义或说明。具体来讲，本书所界定的特征工程包括“数据清理”“特征变换”“特征选择”和“特征抽取”，是后续各章节内容。

4．数据生产

如果说第 2、3 步得到的是数据“原料”，本步就是用原料进行“生产”的过程，也就是前面关于“数据科学”的定义中提及的“从数据中提取有价值的部分来生产数据产品”的过程，可以比喻为“生产车间”。图 1-0-1 中的“数据分析”和“机器学习”只不过是目前常用的两条“生产线”。

机器学习是人工智能的基础和重要组成部分。本书所阐述的大多数技术是为机器学习算法而准备的。

5．评估

不论是机器学习，还是数据分析，其结果都要进行评估。根据评估结果，确定是否采用机器学习所获得的模型或数据分析的报告。

在机器学习中，有专门评估模型的算法。

6．部署和应用

作为商业项目，最终都要把产品部署到商业系统中，比如可能作为某个网站系统的一部分。这样才能让研究出来的算法（模型）处理新的数据，并满足商业需求。

在实际项目中，图 1-0-1 中的各环节不是截然分开、彼此互不相关的，也不是机械地按照单一方向进行的。

- 本书阐述的“数据准备和特征工程”，不只是在“数据收集”和“特征工程”环节实施，还可能贯穿整个项目过程。
- 各环节之间不仅前后衔接，而且还可能循环往复。例如，在进行“数据收集”时有可能要再次“理解商业问题”，才能确定所收集的数据内容，甚至决定项目是否具有可行性；经“评估”后发现模型不能解决实际问题，有可能要回到“数据收集”才能提升模型效果。

因此，在了解基本流程之后，再回到前面所述的认识，将两者结合，才是数据科学项目的实施原则。

1.1　文件中的数据

文件是计算机科学中的常用术语。文件能用于保存数据，而且是多种多样的，如文本文件、图像文件和办公软件生成的二进制文件等。不同文件中的数据也有差别，有的数据是结构化的，有的数据是非结构化的。本节将向读者介绍数据科学项目中常用到的三种文件，并重点演示如何从文件中读取数据。

结构化数据是指关系型数据库中用二维表格表达和存储的数据，每个数据有严格的数据格式和规范。非结构化数据（如图像、视频等）则不能用二维表格形式表达和存储。

1.1.1　CSV 文件

CSV（Comma-Separated Values，逗号分隔值）是以纯文本方式保存数据的常用文件格式，其中的数据属于结构化数据。

基础知识

用电子表格工具软件可以打开 CSV 文件，如图 1-1-1 所示。

处理电子表格文件的常用工具，包括微软的 Excel 和 WPS 中的电子表格软件，也可以使用在线电子表格，如腾讯文档。

	A	B	C	D	E
1	name	area	population	longd	latd
2	Nanjing	6582.31	8004680	118.78	32.04
3	Wuxi	4787.61	6372624	120.29	31.59
4	Xuzhou	11764.88	8580500	117.2	34.26
5	Changzhou	4384.57	4591972	119.95	31.79
6	Soochow	8488.42	10465994	120.62	31.32
7	Nantong	8001	7282835	120.86	32.01
8	Lianyungang	7615.29	4393914	119.16	34.59
9	Huaian	9949.97	4799889	119.15	33.5
10	Yancheng	16972.42	7260240	120.13	33.38
11	Yangzhou	6591.21	4459760	119.42	32.39
12	Zhenjiang	3840.32	3113384	119.44	32.2
13	Taizhou	5787.26	4618558	119.9	32.49
14	Suqian	8555	4715553	118.3	33.96
15					

图 1-1-1　用电子表格工具软件打开的 CSV 文件

读者对电子表格软件一定不陌生，本书不再赘述操作方法。在数据科学项目实践中，也会有很多场景应用这类软件，请读者不要排斥。一切工具的目的都是得到符合预期的数据。只不过因为电子表格软件不是本书的重点内容，所以后面不再提及该工具，但不意味着不可以应用它。

下面重点说明使用 Python 编程语言读取 CSV 文件的数据。

path 是本书开发环境中的数据集目录，读者要根据自己的开发环境进行修改。

```
In [1]: import csv
        path = "/Users/qiwsir/Documents/Codes/DataSet"
        csv_file = path + "/jiangsu/cities.csv"
        f = open(csv_file)
        data = csv.reader(f)     # ①
        for line in data:
            print(line)
        # 以下是输出信息
        ['name', 'area', 'population', 'longd', 'latd']
        ['Nanjing', '6582.31', '8004680', '118.78',
         '32.04']
        ['Wuxi', '4787.61', '6372624', '120.29', '31.59']
        ['Xuzhou', '11764.88', '8580500', '117.2', '34.26']
        ['Changzhou', '4384.57', '4591972', '119.95',
         '31.79']
        ['Soochow', '8488.42', '10465994', '120.62',
         '31.32']
        ['Nantong', '8001', '7282835', '120.86', '32.01']
        ['Lianyungang', '7615.29', '4393914', '119.16',
         '34.59']
        ['Huaian', '9949.97', '4799889', '119.15', '33.5']
        ['Yancheng', '16972.42', '7260240', '120.13',
         '33.38']
        ['Yangzhou', '6591.21', '4459760', '119.42',
```

In[1] 第 5 行里的“# ①”表示注释，程序执行时忽略此内容。

```
 '32.39']
['Zhenjiang', '3840.32', '3113384', '119.44',
 '32.2']
['Taizhou', '5787.26', '4618558', '119.9', '32.49']
['Suqian', '8555', '4715553', '118.3', '33.96']
```

老齐教室
微信扫描二维码，关注我的公众号

为了便于学习，可以按照“前言”中的说明关注与本书相关的微信公众号，从而获得学习用数据集和加入在线实验平台。在代码示例中，用“/jiangsu/cities.csv”方式表示此文件在数据源目录中的地址。

附录 A：简要介绍 Jupyter。

以上代码在 Jupyter notebook 中调试。Jupyter 是数据科学中常用的工具——详细使用方法请阅读“扩展探究”中推荐的资料。

用 Python 标准库的 csv 模块能够读取 CSV 文件内容，但在 In[1] 的①中得到的是一个迭代器对象，必须使用循环语句才能将文件中每一行读入内存，这使得后续操作很不方便。开发者不允许存在任何“不方便”，因为会降低工作效率，并且，CSV 文件是常见的保存数据的文件。因此，必然有更简单且能够更适用于后续操作的方法——如果没有，则是创新的机会。

关于“迭代器”对象，请参阅《Python 大学实用教程》（电子工业出版社出版）。

Python 语言生态中的 Pandas 提供了实现上述诉求的函数——关于 Pandas 的使用方法请阅读“扩展探究”中推荐的资料。当然，如果读者对 In[2] 的②所示的函数不满意，也可以自己创造。

附录 C：简要介绍 Pandas。

```
In [2]: import pandas as pd
        df = pd.read_csv(csv_file)     # ②
        df
Out[2]:
```

	name	area	population	longd	latd
0	Nanjing	6582.31	8004680	118.78	32.04
1	Wuxi	4787.61	6372624	120.29	31.59
2	Xuzhou	11764.88	8580500	117.20	34.26
3	Changzhou	4384.57	4591972	119.95	31.79
4	Soochow	8488.42	10465994	120.62	31.32
5	Nantong	8001.00	7282835	120.86	32.01
6	Lianyungang	7615.29	4393914	119.16	34.59
7	Huaian	9949.97	4799889	119.15	33.50
8	Yancheng	16972.42	7260240	120.13	33.38
9	Yangzhou	6591.21	4459760	119.42	32.39
10	Zhenjiang	3840.32	3113384	119.44	32.20
11	Taizhou	5787.26	4618558	119.90	32.49
12	Suqian	8555.00	4715553	118.30	33.96

比较 Out[2] 和 In[1] 的输出，此处的结果在显示方式上友好了很多。不仅如此，这里所得到的对象（变量 df 引用）是数据科学项目中用途最广泛的 DataFrame 类型的对象。

DataFrame 是 Pandas 中的一种对象类型，类似于二维表格。

In[2] 的②使用 Pandas 的 read_csv 函数读取了指定的 CSV 文件，此函数的完整参数列表是：

```
pd.read_csv(filepath_or_buffer, sep=',', delimiter=None,
```

```
header='infer', names=None, index_col=None, usecols=None,
squeeze=False, prefix=None, mangle_dupe_cols=True, dtype=None,
engine=None, converters=None, true_values=None, false_values=None,
skipinitialspace=False, skiprows=None, nrows=None, na_values=None,
keep_default_na=True, na_filter=True, verbose=False, skip_blank_
lines=True, parse_dates=False, infer_datetime_format=False, keep_
date_col=False, date_parser=None, dayfirst=False, iterator=False,
chunksize=None, compression='infer', thousands=None, decimal=b'.',
lineterminator=None, quotechar='"', quoting=0, escapechar=None,
comment=None, encoding=None, dialect=None, tupleize_cols=None,
error_bad_lines=True, warn_bad_lines=True, skipfooter=0,
doublequote=True, delim_whitespace=False, low_memory=True,
memory_map=False, float_precision=None)
```

不需要对这些参数的含义死记硬背，可以使用帮助文档了解。建议读者浏览一遍 pd.read_csv 函数的帮助文档，当以后需要处理某个特殊问题的时候，可以再次借助帮助文档，查询相应参数。

```
In [3]: pd.read_csv?
```

在 Jupyter 中输入 In[3] 的代码并执行，能够显示函数 read_csv 的完整文档，其中包含对所有参数的解释。

例如，在 Out[2] 输出的二维数据表格中，以数字序号表示索引。在读取此 CSV 文件的时候，也可以通过参数指定文件中的某一列作为索引。

```
In [4]: pd.read_csv(csv_file, index_col=0)
Out[4]:
```

name	area	population	longd	Latd
Nanjing	6582.31	8004680	118.78	32.04
Wuxi	4787.61	6372624	120.29	31.59
Xuzhou	11764.88	8580500	117.20	34.26
Changzhou	4384.57	4591972	119.95	31.79
Soochow	8488.42	10465994	120.62	31.32
Nantong	8001.00	7282835	120.86	32.01
Lianyungang	7615.29	4393914	119.16	34.59
Huaian	9949.97	4799889	119.15	33.50
Yancheng	16972.42	7260240	120.13	33.38
Yangzhou	6591.21	4459760	119.42	32.39
Zhenjiang	3840.32	3113384	119.44	32.20
Taizhou	5787.26	4618558	119.90	32.49
Suqian	8555.00	4715553	118.30	33.96

In[4] 中函数 read_csv 增设了参数 index_col=0，意思是用 CSV 文件的第 0 列作为索引，最终得到了 Out[4] 输出效果。

推荐阅读《跟老齐学 Python：数据分析》（电子工业出版社出版），系统化了解 Pandas 的各项知识。

在 In[2] 的②中读取到 CSV 文件之后，返回的是 DataFrame 对象（②中用变量 df 引用此对象），有的资料将 DataFrame 翻译为“数据框”，本书使用英文名称。

项目案例

1. 项目描述

读取“/kaggle/diabetes.csv”数据，并了解此数据集的概况。

2. 实现过程

```
In [5]: f = "/kaggle/diabetes.csv"
        diabetes = pd.read_csv(path + f)     # ③
        diabetes.shape     # ④
Out[5]: (768, 9)
```

In[5] 的③读取指定的 CSV 文件，得到了变量 diabetes 引用的 DataFrame 对象。④通过 DataFrame 实例的属性 shape 得到了 diabetes 的形状，Out[5] 的输出结果表示 diabetes 共有 765 行、9 列。如果直接调用 diabetes，就会将所有内容显示出来（读者可以在 Jupyter 中尝试），在页面上占用较多篇幅，为避免这种情况，可以显示部分样本。

```
In [6]: diabetes.head()
Out[6]:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

In[6] 中出现的 head 方法是 DataFrame 对象常用的显示部分样本的方法，默认显示前 5 个，传入整数类型的参数，就可以根据指定数量显示样本。

与 head 方法类似的，还有 tail 和 sample 方法。

```
In [7]: diabetes.info()
        # 以下为输出信息
        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 768 entries, 0 to 767
        Data columns (total 9 columns):
        Pregnancies                 768 non-null int64
        Glucose                     768 non-null int64
        BloodPressure               768 non-null int64
        SkinThickness               768 non-null int64
        Insulin                     768 non-null int64
        BMI                         768 non-null float64
        DiabetesPedigreeFunction    768 non-null float64
        Age                         768 non-null int64
        Outcome                     768 non-null int64
        dtypes: float64(2), int64(7)
        memory usage: 54.1 KB

In [8]: diabetes.dtypes
Out[8]: Pregnancies                 int64
```

比较 In[7] 和 In[8] 的执行结果，了解二者的异同。

```
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                           float64
DiabetesPedigreeFunction      float64
Age                           int64
Outcome                       int64
dtype: object
```

In[7] 和 In[8] 的功能类似，能够显示出 DataFrame 对象中每列的数据类型。

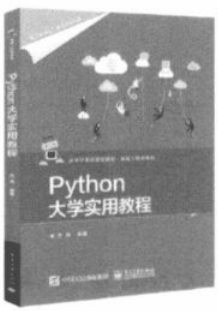
动手练习

git 是源码管理工具，目前已经被普遍采用。

1. 先在 github.com 网站完成用户注册和登录操作，然后完成如下操作。

- 在本地计算机安装 git，熟悉常用的 git 命令。
- 在 github.com 网站创建个人公开代码仓库。
- 应用 git 的 push 命令将本地指定目录中的文件上传到个人的代码仓库中。

本题目与《Python 大学实用教程》的“练习和编程 1”第 5 题相呼应，建议读者查阅有关资料，完成本题各项操作。



2. 参考本书附录或者推荐的书籍，完成如下操作。

（1）在本地计算机安装并运行 Jupyter。

（2）在本地计算机安装 Pandas、Numpy。

3. 用 Pandas 读取“/bicycle/Bicycle_Counts.csv”文件的数据，并完成如下操作。

（1）以第 1 列为索引，并显示前 10 个样本。

（2）返回此数据集的样本总数。

（3）将（1）所显示的数据保存到一个新的 CSV 文件中。

扩展探究

1. Jupyter 是基于浏览器的代码编辑工具，在数据科学中被广泛采用，其官方网站是 https://jupyter.org/。建议读者根据网站文档安装此工具，并学会使用。

2. Numpy 和 Pandas 是 Python 语言在数据科学中的重要工具，使用 Python 语言的数据科学项目都必须使用它们。本书在后续各种操作中会对涉及的一些函数（方法）给予必要的介绍，但是不能替代读者系统化学习。建议阅读《跟老齐学 Python：数据分析》，系统学习 Numpy 和 Pandas 的有关知识。

3. 在数据科学中，安装第三方模块（包）的方式，依然可以使用 Python 语言中常用的 pip 命令（参阅《Python 大学实用教程》）。此外，还有另一个专门的数据科学集成开发工具 Anaconda（官方网站：https://www.anaconda.com/），安装此工具之后，数据科学中常用的模块（包）就已经集成在其中，

未集成进来的其他模块一般也提供了 conda 命令的安装方法。更详细的内容请查阅官方文档（https://docs.anaconda.com/）。

1.1.2　Excel 文件

Excel 文件也是常用于保存数据的文件，《Python 大学实用教程》的 9.2.2 节专门介绍了如何使用 Python 第三方包读 / 写此类文件，请读者参阅。本节将重点介绍如何用 Pandas 从 Excel 文件中读取数据。

基础知识

在 Jupyter 中输入 pd.read_，然后按下 Tab 键，就可以出现如图 1-1-2 所示的效果，从这里可以看到多个以“read”开始的函数名称——Python 中规范的命名方式遵循着“望文生义”的原则。

利用 Tab 键可以查找函数，减轻记忆负担。

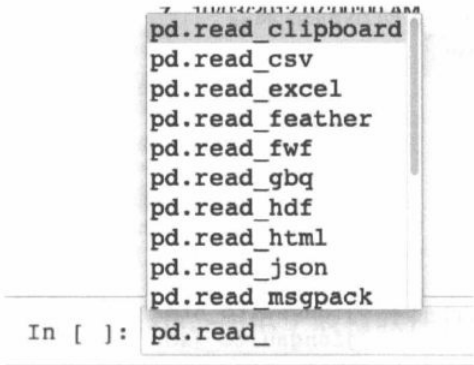


图 1-1-2　Tab 键辅助记忆

在学习和工作过程中，都应该充分利用帮助文档。In[3] 演示了获得 pd.read_csv 函数帮助信息的方法，用同样的方法，也可以查看 pd.read_excel 函数的文档内容。

```
In [9]: pd.read_excel?
```

依然建议读者认真阅读文档内容，了解此函数的基本使用方法。在帮助文档的后面，通常还会有学习示例。

下面就使用这个函数读取 Excel 文件的数据。

path 为 In[1] 中创建的变量。

```
In [10]: jiangsu = pd.read_excel(path +
                                 "/jiangsu/jiangsu.xls")
         jiangsu
Out[10]:
              name        area        population longd    latd
         0    Nanjing     6582.31     8004680    118.78   32.04
         1    Wuxi        4787.61     6372624    120.29   31.59
         2    Xuzhou      11764.88    8580500    117.20   34.26
```

3	Changzhou	4384.57	4591972	119.95	31.79
4	Soochow	8488.42	10465994	120.62	31.32
5	Nantong	8001.00	7282835	120.86	32.01
6	Lianyungang	7615.29	4393914	119.16	34.59
7	Huaian	9949.97	4799889	119.15	33.50
8	Yancheng	16972.42	7260240	120.13	33.38
9	Yangzhou	6591.21	4459760	119.42	32.39
10	Zhenjiang	3840.32	3113384	119.44	32.20
11	Taizhou	5787.26	4618558	119.90	32.49
12	Suqian	8555.00	4715553	118.30	33.96

除了如 In[10] 中的代码那样读取 Excel 文件，还可以利用电子表格软件将 Excel 文件转化为 CSV 文件，然后利用 Pandas 的 read_csv 函数读取文件。

如果将已有的数据，如 DataFrame 类的数据，保存为 Excel 或者 CSV 文件，应当如何操作？

继续使用如图 1-1-2 所示的方法，在 Jupyter 中输入“jiangsu.to_”，然后按 Tab 键，显示如图 1-1-3 所示的结果。

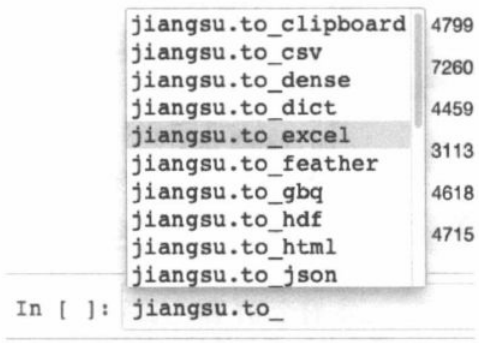


图 1-1-3　保存为某种文件的函数

从这里可以看到，DataFrame 对象实例保存为某种格式文件的方法（用这种方法找到解决 1.1.1 节“动手练习”中第 3 题所需要的方法）。读者应该认真观察图 1-1-2 和图 1-1-3 显示的函数（方法）名称，从而了解到 Pandas 可以读、写什么格式的文件。

```
In [11]: jiangsu.to_excel('./chapter01/jiangsu.xlsx')
```

如果没有报错和其他显示，则说明已经保存成功。若不放心，可以到目录中查看是否已有保存的文件。

在 Jupyter 中执行 shell 命令查看当前目录内容的方式：

```
!ls
```

即在命令前面写上“！”符号（英文状态）。

项目案例

1. 项目描述

从“国家数据”网站（http://data.stats.gov.cn/）下载“全国居民消费价格分类指数”，并用柱形图表示指数的变化。