《python数据分析与挖掘实战》笔记第4章

**文章目录**

* + [第4章：数据预处理](https://cloud.tencent.com/developer?from=10680)
    - [4.1、数据清洗](https://cloud.tencent.com/developer?from=10680)
      * [4.1.1、缺失值处理](https://cloud.tencent.com/developer?from=10680)
      * [4.1.1、异常值处理](https://cloud.tencent.com/developer?from=10680)
    - [4.2、数据集成](https://cloud.tencent.com/developer?from=10680)
      * [4.2.1、实体识别](https://cloud.tencent.com/developer?from=10680)
      * [4.2.2、冗余属性识别](https://cloud.tencent.com/developer?from=10680)
    - [4.3、数据变换](https://cloud.tencent.com/developer?from=10680)
      * [4.3.1、简单函数变换](https://cloud.tencent.com/developer?from=10680)
      * [4.3.2、规范化](https://cloud.tencent.com/developer?from=10680)
      * [4.3.3、连续属性离散化](https://cloud.tencent.com/developer?from=10680)
      * [4.3.4、属性构造](https://cloud.tencent.com/developer?from=10680)
      * [4.3.5、小波变换](https://cloud.tencent.com/developer?from=10680)
    - [4.4、数据规约](https://cloud.tencent.com/developer?from=10680)
      * [4.4.1、属性规约](https://cloud.tencent.com/developer?from=10680)
      * [4.4.2、数值规约](https://cloud.tencent.com/developer?from=10680)
    - [4.5、Python主要数据预处理函数](https://cloud.tencent.com/developer?from=10680)
    - [4.6、小结](https://cloud.tencent.com/developer?from=10680)

**第4章：数据预处理**

数据预处理一方面是要提高数据的质量，另一方面是要让 数据更好地适应特定的挖掘技术或工具。统计发现，在数据挖掘的过程中，数据预处理工作量占到了整个过程的60%。

**4.1、数据清洗**

数据清洗主要是删除原始数据集中的无关数 据、重复数据，平滑噪声数据，筛选掉与挖掘主 题无关的数据，处理缺失值、异常值等。

**4.1.1、缺失值处理**

处理缺失值的方法可分为3类：删除记录、 数据插补和不处理。其中常用的数据插补方法

**表4-1常用的插补方法**

| 插补方法 | 方法描述 |
| --- | --- |
| 均值/中位数/众数插补 | 根据属性值的类型，用该属性取值的平均数/中位数/众数进行插补 |
| 使用固定值 | 将缺失的属性值用一个常量替换。如广州一个工厂普通外来务工人员的“基本工资”属性的空缺值可以用2015年广州市普通外来务工人员工资标准1895元/月，该 方法就是使用固定值 |
| 最近临插补 | 在记录中找到与缺失样本最接近的样本的该属性值插补 |
| 回归方法 | 对带有缺失值的变量，根据已有数据和与其有关的其他变量（因变量）的数据建立拟合模型来预测缺失的属性值 |
| 插值法 | 插值法是利用已知点建立合适的插值函数f(x),未知值由对应点X,求出的函数值f(xi),近似代替 |

如果通过简单的删除小部分记录达到既定的目标，那么删除含有缺失值的记录的方法是最有效的。然而，这种方法却有很大的局限性。它是以减少历史数据来换取数据的完备，会造成资源的大量浪费，将丢弃了大量隐藏在这些记录中的信息。尤其在数据集本来就包含很少记录的情况下，删除少量记录可能会严重影响到分析结果的客观性和正确性。一些模型可以将缺失值视作一种特殊的取值，允许直接在含有缺失值的数据上进行建模。

本节重点介绍**拉格朗日插值法和牛顿插值法**。其他的插值方法还有**Hermite插值、分段插值、样条插值法**等。

牛顿插值法也是多项式插值，但采用了另一种构造插值多项式的方法，与拉格朗日插值相比，具有承袭性和易于变动节点的特点。从本质上来说，两者给出的结果是一样的(相同 次数、相同系数的多项式)，只不过表示的形式不同。因此，在Python的Scipy库中，只提 供了拉格朗日插值法的函数(因为实现上比较容易)，如果需要牛顿插值法，则需要自行编写

**代码清单4-1，用拉格朗日法进行插补**

# -\*- coding:utf-8 -\*-

#拉格朗日插值代码

import pandas as pd #导入数据分析库Pandas

from scipy.interpolate import lagrange #导入拉格朗日插值函数

inputfile = '../data/catering\_sale.xls' #销量数据路径

outputfile = '../tmp/sales.xls' #输出数据路径

data = pd.read\_excel(inputfile) #读入数据

data[u'销量'][(data[u'销量'] < 400) | (data[u'销量'] > 5000)] = None #过滤异常值，将其变为空值

#自定义列向量插值函数

#s为列向量，n为被插值的位置，k为取前后的数据个数，默认为5

def ployinterp\_column(s, n, k=5):

y = s[list(range(n-k, n)) + list(range(n+1, n+1+k))] #取数

y = y[y.notnull()] #剔除空值

return lagrange(y.index, list(y))(n) #插值并返回插值结果

#逐个元素判断是否需要插值

for i in data.columns:

for j in range(len(data)):

if (data[i].isnull())[j]: #如果为空即插值。

data[i][j] = ployinterp\_column(data[i], j)

data.to\_excel(outputfile) #输出结果，写入文件

复制

**4.1.1、异常值处理**

在数据预处理时，异常值是否剔除，需视具体情况而定，因为有些异常值可能蕴含着有 用的信息。异常值处理常用方法见表4-3。

表4-3异常值处理常用方法

| 异常值处理方法 | 方法描述 |
| --- | --- |
| 删除含有异常值的记录 | 直接将含有异常值的记录删除 |
| 视为缺失值 | 将异常值视为缺失值，利用缺失值处理的方法进行处理 |
| 平均值修正 | 可用前后两个观测值的平均值修正该异常值 |
| 不处理 | 直接在具有异常值的数据集上进行挖掘建模 |

**4.2、**[**数据集成**](https://cloud.tencent.com/product/datainlong?from=10680)

数据挖掘需要的数据往往分布在不同的数据源中，数据集成就是将多个数据源合并存放 在一个一致的数据存储（如数据仓库）中的过程。

在数据集成时，来自多个数据源的现实世界实体的表达形式是不一样的，有可能不匹配，要考虑实体识别问题和属性冗余问题，从而将源数据在最低层上加以转换、提炼和集成。

**4.2.1、实体识别**

实体识别是指从不同数据源识别出现实世界的实体，它的任务是统一不同源数据的矛盾之处，常见形式如下。

(1 )同名异义 数据源A中的属性ID和数据源B中的属性ID分别描述的是菜品编号和订单编号，即 描述的是不同的实体。

(2) 异名同义 数据源A中的sales\_dt和数据源B中的sales\_date都是描述销售日期的，即A. sales\_dt= B. sales\_date。

(3 )单位不统一

描述同一个实体分别用的是国际单位和中国传统的计量单位。 检测和解决这些冲突就是实体识别的任务。

**4.2.2、冗余属性识别**

数据集成往往导致数据冗余，例如，

1. 同一属性多次出现；
2. 同一属性命名不一致导致重复。

**4.3、数据变换**

数据变换主要是对数据进行规范化处理，将数据转换成“适当的”形式，以适用于挖掘任务及算法的需要。

**4.3.1、简单函数变换**

简单函数变换是对原始数据进行某些数学函数变换，常用的变换包括平方、开方、取对数、差分运算等。

简单的函数变换常用来将不具有正态分布的数据变换成具有正态分布的数据。

**4.3.2、规范化**

数据规范化(归一化)处理是数据挖掘的一项基础工作。不同评价指标往往具有不同的量纲，数值间的差别可能很大，不进行处理可能会影响到数据分析的结果。为了消除指标之间的量纲和取值范围差异的影响，需要进行标准化处理，将数据按照比例进行缩放，使之落 入一个特定的区域，便于进行综合分析。如将工资收入属性值映射到［-1,1］或者［0,1］内。

数据规范化对于基于距离的挖掘算法尤为重要。

(1)最小-最大规范化 最小-最大规范化也称为离差标准化，是对原始数据的线性变换，将数值值映射到［0,1］之间。

(2 )零-均值规范化 零-均值规范化也称标准差标准化，经过处理的数据的均值为0,标准差为1。是当前用得最多的数据标准化方法。

(3) 小数定标规范化 通过移动属性值的小数位数，将属性值映射到［-1,1］之间，移动的小数位数取决于属性值绝对值的最大值。

代码清单4-2数据规范化代码

#-\*- coding: utf-8 -\*-

#数据规范化

import pandas as pd

import numpy as np

datafile = '../data/normalization\_data.xls' #参数初始化

data = pd.read\_excel(datafile, header = None) #读取数据

(data - data.min())/(data.max() - data.min()) #最小-最大规范化

(data - data.mean())/data.std() #零-均值规范化

data/10\*\*np.ceil(np.log10(data.abs().max())) #小数定标规范化

复制

**4.3.3、连续属性离散化**

一些数据挖掘算法，特别是某些分类算法(如ID3算法、Apriori算法等)，要求数据是 分类属性形式。这样，常常需要将连续属性变换成分类属性，即连续属性离散化。

1. 离散化的过程

连续属性的离散化就是在数据的取值范围内设定若干个离散的划分点，将取值范围划分为一些离散化的区间，最后用不同的符号或整数值代表落在每个子区间中的数据值。所以， 离散化涉及两个子任务：确定分类数以及如何将连续属性值映射到这些分类值。

1. 常用的离散化方法

常用的离散化方法有等宽法、等频法和(一维)聚类。

(1 )等宽法 将属性的值域分成具有相同宽度的区间，区间的个数由数据本身的特点决定，或者由用 户指定，类似于制作频率分布表。

(2 )等频法 将相同数量的记录放进每个区间。

这两种方法简单，易于操作，但都需要人为地规定划分区间的个数。同时，等宽法的缺点在于它对离群点比较敏感，倾向于不均匀地把属性值分布到各个区间。有些区间包含许多数据，而另外一些区间的数据极少，这样会严重损坏建立的决策模型。等频法虽然避免了上述问题的产生，却可能将相同的数据值分到不同的区间以满足每个区间中固定的数据个数。

(3)基于聚类分析的方法 一维聚类的方法包括两个步骤，首先将连续属性的值用聚类算法(如K-Means算法)进 行聚类，然后再将聚类得到的簇进行处理，合并到一个簇的连续属性值并做同一标记。聚类分析的离散化方法也需要用户指定簇的个数，从而决定产生的区间数。

代码清单4-3数据离散化

#-\*- coding: utf-8 -\*-

#数据规范化

import pandas as pd

datafile = '../data/discretization\_data.xls' #参数初始化

data = pd.read\_excel(datafile) #读取数据

data = data[u'肝气郁结证型系数'].copy()

k = 4

d1 = pd.cut(data, k, labels = range(k)) #等宽离散化，各个类比依次命名为0,1,2,3

#等频率离散化

w = [1.0\*i/k for i in range(k+1)]

w = data.describe(percentiles = w)[4:4+k+1] #使用describe函数自动计算分位数

w[0] = w[0]\*(1-1e-10)

d2 = pd.cut(data, w, labels = range(k))

def cluster\_plot(d, k): #自定义作图函数来显示聚类结果

import matplotlib.pyplot as plt

plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False #用来正常显示负号

plt.figure(figsize = (8, 3))

for j in range(0, k):

plt.plot(data[d==j], [j for i in d[d==j]], 'o')

plt.ylim(-0.5, k-0.5)

return plt

if \_\_name\_\_=='\_\_main\_\_':

from sklearn.cluster import KMeans # 引入KMeans

kmodel = KMeans(n\_clusters=k, n\_jobs=4) # 建立模型，n\_jobs是并行数，一般等于CPU数较好

kmodel.fit(data.values.reshape((len(data), 1))) # 训练模型

c = pd.DataFrame(kmodel.cluster\_centers\_).sort\_values(0) # 输出聚类中心，并且排序（默认是随机序的）

w = c.rolling(2).mean().iloc[1:] # 相邻两项求中点，作为边界点

w = [0] + list(w[0]) + [data.max()] # 把首末边界点加上

d3 = pd.cut(data, w, labels=range(k))

cluster\_plot(d1, k).show()

cluster\_plot(d2, k).show()

cluster\_plot(d3, k).show()

复制

* 等宽离散化结果
* 等频离散化结果
* （一维）聚类离散化结果

**4.3.4、属性构造**

在数据挖掘的过程中，为了提取更有用的信息，挖掘更深层次的模式，提高挖掘结果的 精度，我们需要利用已有的属性集构造出新的属性，并加入到现有的属性集合中。

代码清单4-4线损率属性构造

#-\*- coding: utf-8 -\*-

#线损率属性构造

import pandas as pd

#参数初始化

inputfile= '../data/electricity\_data.xls' #供入供出电量数据

outputfile = '../tmp/electricity\_data.xls' #属性构造后数据文件

data = pd.read\_excel(inputfile) #读入数据

data[u'线损率'] = (data[u'供入电量'] - data[u'供出电量'])/data[u'供入电量']

data.to\_excel(outputfile, index = False) #保存结果

复制

**4.3.5、小波变换**

小波变换的同是一种新型的数据分析工具，是近年来兴起的信号分析手段。小波分析的理论和方法在信号处理、[图像处理](https://cloud.tencent.com/product/tiia?from=10680)、语音处理、模式识别、量子物理等领域得到越来越广泛的应用，它被认为是近年来在工具及方法上的重大突破。小波变换具有多分辨率的特点，在 时域和频域都具有表征信号局部特征的能力，通过伸缩和平移等运算过程对信号进行多尺度 聚焦分析，提供了一种非平稳信号的时频分析手段，可以由粗及细地逐步观察信号，从中提取有用信息。

能够刻画某个问题的特征量往往是隐含在一个信号中的某个或者某些分量中，小波变换 可以把非平稳信号分解为表达不同层次、不同频带信息的数据序列，即小波系数。选取适当的小波系数，即完成了信号的特征提取。下面将介绍基于小波变换的信号特征提取方法。

(1)基于小波变换的特征提取方法 基于小波变换的特征提取方法主要有：基于小波变换的多尺度空间能量分布特征提取、 基于小波变换的多尺度空间的模极大值特征提取、基于小波包变换的特征提取、基于适应性小波神经网络的特征提取，详见表4-5。

表4-5基于小波变换的特征提取方法

| 基于小波变换的特征提取方法 | 方法描述 |
| --- | --- |
| 基于小波变换的多尺度空间 能量分布特征提取方法 | 各尺度空间内的平滑信号和细节信号能提供原始信号的时频局域信息，特别 是能提供不同频段上信号的构成信息。把不同分解尺度上信号的能量求解出来， 就可以将这些能量尺度顺序排列，形成特征向量供识别用 |
| 基于小波变换的多尺度空间 的模极大值特征提取方法 | 利用小波变换的信号局域化分析能力，求解小波变换的模极大值特性来检测 信号的局部奇异性，将小波变换模极大值的尺度参数S、平移参数，及其幅值作 为目标的特征量 |
| 基于小波包变换的特征提取 方法 | 利用小波分解，可将时域随机信号序列映射为尺度域各子空间内的随机系数 序列，按小波包分解得到的最佳子空间内随机系数序列的不确定性程度最低， 将最佳子空间的嫡值及最佳子空间在完整二叉树中的位置参数作为特征量，可 以用于目标识别 |
| 基于适应性小波神经网络的 特征提取方法 | 基于适应性小波神经网络的特征提取方法可以把信号通过分析小波拟合表示， 进行特征提取 |

利用小波变换可以对声波信号进行特征提取，提取出可以代表声波信号的向量数据，即完成从声波信号到特征向量数据的变换。

在Python中，Scipy本身提供了一些信号处理函数，但不够全面， 而更好的信号处理库是PyWavelets (pywt)。

代码清单4-5，小波变换特征提取代码

#-\*- coding: utf-8 -\*-

#利用小波分析进行特征分析

#参数初始化

inputfile= '../data/leleccum.mat' #提取自Matlab的信号文件

from scipy.io import loadmat #mat是MATLAB专用格式，需要用loadmat读取它

mat = loadmat(inputfile)

signal = mat['leleccum'][0]

import pywt #导入PyWavelets

coeffs = pywt.wavedec(signal, 'bior3.7', level = 5)

#返回结果为level+1个数字，第一个数组为逼近系数数组，后面的依次是细节系数数组

复制

**4.4、数据规约**

在[大数据](https://cloud.tencent.com/solution/bigdata?from=10680)集上进行复杂的数据分析和挖掘需要很长的时间，数据规约产生更小但保持原数据完整性的新数据集。在规约后的数据集上进行分析和挖掘将更有效率。

数据规约的意义在于：

* 降低无效、错误数据对建模的影响，提高建模的准确性；
* 少量且具代表性的数据将大幅缩减数据挖掘所需的时间；
* 降低储存数据的成本。

**4.4.1、属性规约**

**属性规约**通过属性合并来创建新属性维数，或者直接通过删除不相关的属性（维）来减少数据维数，从而提高数据挖掘的效率、降低计算成本。**属性规约的目标**是寻找出最小的属性子集并确保新数据子集的概率分布尽可能地接近原来数据集的概率分布。

**逐步向前选择、逐步向后删除和决策树归纳是属于直接删除不相关属性（维）方法**。**主成分分析**是一种用于连续属性的数据降维方法，它构造了原始数据的一个正交变换，新空间的基底去除了原始空间基底下数据的相关性，只需使用少数新变量就能够解释原始数据中的 大部分变异。在应用中，通常是选出比原始变量个数少，能解释大部分数据中的变量的几个新变量，即所谓主成分，来代替原始变量进行建模。

**计算主成分：** 在Python中,主成分分析的函数位于Scikit-Leam下：

sklearn.decomposition.PCA(n\_components = None,copy = True,whiten = False)

复制

参数说明：

（1） n\_components

意义：PCA算法中所要保留的主成分个数n，也即保留下来的特征个数n。

类型：int或者string，缺省时默认为None，所有成分被保留。赋值为int，比如n\_components =1将把原始数据降到一个维度。赋值为string，比如n\_components =‘mle’，将自动选取特征个数n，使得满足所要求的方差百分比。

(2 ) copy

类型：bool, True或者False，缺省时默认为True。

意义：表示是否在运行算法时，将原始训练数据复制一份。若为True，则运行PCA 算法后，原始训练数据的值不会有任何改变，因为是在原始数据的副本上进行运算；若为 False，则运行PCA算法后，原始训练数据的值会改，因为是在原始数据上进行降维计算。

(3 ) whiten

类型：bool,缺省时默认为False。

意义：白化，使得每个特征具有相同的方差。

使用主成分分析降维的程序如代码清单4-6所示。

代码清单4-6，主成分分析降维代码

#-\*- coding: utf-8 -\*-

#主成分分析 降维

import pandas as pd

#参数初始化

inputfile = '../data/principal\_component.xls'

outputfile = '../tmp/dimention\_reducted.xls' #降维后的数据

data = pd.read\_excel(inputfile, header = None) #读入数据

from sklearn.decomposition import PCA

pca = PCA()

pca.fit(data)

print(pca.components\_ )#返回模型的各个特征向量

print(pca.explained\_variance\_ratio\_) #返回各个成分各自的方差百分比(贡献率)

'''

方差百分比越大，说明向量的权 重越大

当选取前4个主成分时，累计贡献率已达到97.37%,说明选取前3个主成分进行计算已经相当不错了，

因此可以重新建立PCA模型，设置n\_components=3，计算出成分结果。

'''

pca = PCA(n\_components=3)

pca.fit(data)

low\_d = pca.transform(data) #用它来降低维度

print(low\_d)

pd.DataFrame(low\_d).to\_excel(outputfile) #保存结果

pca.inverse\_transform(low\_d) #必要时可inverse\_transform()函数来复原数据

复制

**4.4.2、数值规约**

数值规约指通过选择替代的、较小的数据来减少数据量，包括有参数方法和无参数方法两类。**有参数方法**是使用一个模型来评估数据，只需存放参数，而不需要存放实际数据，例如回归（线性回归和多元回归）和对数线性模型（近似离散属性集中的多维概率分布）。**无参数方法**就需要存放实际数据，例如直方图、聚类、抽样（采样）。

**4.5、Python主要数据预处理函数**

表4-7 Python主要数据预处理函数

| 函数名 | 函数功能 | 所属扩展库 |
| --- | --- | --- |
| interpolate | 一维、高维数据插值 | Scipy |
| unique | 去除数据中的重复元素，得到单值元素列表，它是对象的方法名 | Pandas/Numpy |
| isnull | 判断是否空值 | Pandas |
| notnull | 判断是否非空值 | Pandas |
| PCA | 对指标变量矩阵进行主成分分析 | Scikit-Leam |
| random | 生成随机矩阵 | Numpy |

(1 ) interpolate

1 ) 功能：interpolate是Scipy的一个子库，包含了大量的插值函数，如拉格朗日插值、 样条插值、高维插值等。使用前需要用from scipy.interpolate import \*引入相应的插值函数， 读者应该根据需要到官网查找对应的函数名。

2 ) 使用格式：f = scipy.interpolate.lagrange(x, y)。这里仅仅展示了一维数据的拉格朗日插值的命令，其中x,y为对应的自变量和因变量数据。插值完成后，可以通过f(a)计算新的 插值结果。类似的还有样条插值、多维数据插值等，此处不一一展示。

(2) unique

1 ) 功能：去除数据中的重复元素，得到单值元素列表。它既是Numpy库的一个函数 (np.unique()),也是Series对象的一个方法。

2 ) 使用格式：

np.unique(D), D 是一维数据，可以是 list、array、Series； D.unique(), D 是 Pandas 的 Series 对象。

3 ) 实例：求向量A中的单值元素，并返回相关索引。

> D = pd.Series ([1, 1, 2, 3, 5])

> D.unique ()

array([1, 2, 3, 5], dtype=int64)

> np.unique (D)

array([1, 2, 3, 5], dtype=int64)

复制

(3) isnull/ notnull

1 ) 功能：判断每个元素是否空值/非空值。

2 ) 使用格式：D.isnull()/ D.notnull()。这里的D要求是Series对象，返回一个布尔 Series。可以通过D[D.isnull()]或D[D.notnull()]找出D中的空值/非空值。

(4) random

1 ) 功能：random是Numpy的一个子库(Python本身也自带了 random,但Numpy的更加强大)，可以用该库下的各种函数生成服从特定分布的随机矩阵，抽样时可使用。

2 ) 使用格式：

np.random.rand(k, m, n,…)生成一个k x m x n x…随机矩阵，其元素均匀分布在区 间(0,1)上；

np.random.randn(k, m, n,…)生成一个k x m x n x …随机矩阵,其元素服从标准正态分布。

(5) PCA

1 ) 功能：对指标变量矩阵进行主成分分析。使用前需要用from skleam.decomposition import PCA引入该函数。

2 ) 使用格式：model = PCA()。注意，Scikit-Leam下的PCA是一个建模式的对象，也 就是说，一般的流程是建模，然后是训练model.fit(D), D为要进行主成分分析的数据矩阵， 训练结束后获取模型的参数，如.components\_获取特征向量，以及.explained\_variance\_ratio\_获取各个属性的贡献率等。

3 ) 实例：使用PCA()对一个10x4维的随机矩阵进行主成分分析。

from sklearn.decomposition import PCA

D = np.random.rand(10,4)

pca = PCA()

pca.fit (D)

PCA(copy=True, n\_components=None, whiten=False)

pca. components\_ #返回模型的各个特征向量

pca.explained\_variance\_ratio\_ #返回各个成分各自的方差百分比

复制

**4.6、小结**

本章介绍了数据预处理的4个主要任务：数据清洗、数据集成、数据变换和数据规约。

数据清洗主要介绍了对缺失值和异常值的处理，延续了第3章的缺失值和异常值分析的内容，本章所介绍的处理缺失值的方法分为3类：删除记录、数据插补和不处理，处理异常值 的方法有删除含有异常值的记录、不处理、平均值修正和视为缺失值；

数据集成是合并多个数据源中的数据，并存放到一个数据存储的过程，对该部分的介绍从实体识别问题和冗余属性两个方面进行；

数据变换介绍了如何从不同的应用角度对已有属性进行函数变换；

数据规约从属性（纵向）规约和数值（横向）规约两个方面介绍了如何对数据进行规约，使挖掘的性 能和效率得到很大的提高。通过对原始数据进行相应的处理，将为后续挖掘建模提供良好的数据基础。

本文参与[腾讯云自媒体分享计划](https://cloud.tencent.com/developer/support-plan) ，欢迎热爱写作的你一起参与！

本文分享自作者个人站点/博客：https://blog.csdn.net/xc\_zhou/[复制](javascript:void(0);)

如有侵权，请联系 cloudcommunity@tencent.com 删除。

[展开阅读全文](javascript:;)

[数据集成](https://cloud.tencent.com/developer/tag/11212?entry=article)[编程算法](https://cloud.tencent.com/developer/tag/10663?entry=article)[数据分析](https://cloud.tencent.com/developer/tag/10802?entry=article)[Python](https://cloud.tencent.com/developer/tag/10169?entry=article)

[举报](javascript:;)

点赞 2分享

[**登录**](javascript:;) 后参与评论

0 条评论

相关文章

* 《python数据分析与挖掘实战》笔记第5章

分类和预测是预测问题的两种主要类型，分类主要是预测分类标号（离散属性），而预测 主要是建立连续值函数模型，预测给定自变量对应的因变量的值。

[周小董](https://cloud.tencent.com/developer/user/4908043)

* 《python数据分析与挖掘实战》笔记第2章

python有4个内建的数据结构–list（列表）、tuple（元组）、dictionary（字典）以及set（集合），它们可以统称为容器。

[周小董](https://cloud.tencent.com/developer/user/4908043)

* 《python数据分析与挖掘实战》笔记第3章

数据质量分析是数据挖掘中数据准备过程的重要一环，是数据预处理的前提，也是数据挖掘分析结论有效性和准确性的基础，没有可信的数据，数据挖掘构建的模型将是空中楼阁。

[周小董](https://cloud.tencent.com/developer/user/4908043)

* 《python数据分析与挖掘实战》笔记第1章

数据挖掘是从大量数据（包括文本）中挖掘出隐含的、先前未知的、对决策有潜在价值的关系、模式和趋势，并用这些知识和规则建立用于决策支持的模型，提供预测性决策支持的方...

[周小董](https://cloud.tencent.com/developer/user/4908043)

* Python数据分析与实战挖掘

基础篇 书推荐：《用python做科学计算》 ? 扩展库 简介 Numpy数组支持，以及相应的高效处理函数 Scipy矩阵支持，以及相应的矩阵数值计算模块 Ma...

[企鹅号小编](https://cloud.tencent.com/developer/user/1008345)

* Python数据分析与挖掘实战.pdf

[学一学大数据](https://cloud.tencent.com/developer/user/1685822)

* 关于《Python数据挖掘入门与实战》读书笔记二（亲和性分析）

首先确保最新的版本吧，python最让人头痛和无解的就是版本兼容性问题，不过既然选择了就不要怕麻烦，不断的纠错不断的尝试，在频繁的错误中提高自己。

[python与大数据分析](https://cloud.tencent.com/developer/user/7190337)

* 关于《Python数据挖掘入门与实战》读书笔记七（主成分分析二）

主成分分析算法（Principal Component Analysis，PCA）的目的是找到能用较少信息描述数据集的特征组合。它意在发现彼此之间没有相关性、能...

[python与大数据分析](https://cloud.tencent.com/developer/user/7190337)

* Python业务分析实战｜共享单车数据挖掘

自行车共享系统是传统自行车租赁的新一代，从注册会员、租赁到归还的整个过程都是自动化的。通过这些系统，用户可以很容易地从一个特定的位置租用自行车，并在另一个位置归...

[数据STUDIO](https://cloud.tencent.com/developer/user/8756457)

* 关于《Python数据挖掘入门与实战》读书笔记六（主成分分析一）

特征抽取是数据挖掘任务最为重要的一个环节，一般而言，它对最终结果的影响要高过数据挖掘算法本身。但怎样选取好的特征，还没有严格、快捷的规则可循，这也是数据挖掘科学...

[python与大数据分析](https://cloud.tencent.com/developer/user/7190337)

* Python数据分析与挖掘实战 (大数据技术丛书) - 张良均 等著

10余位数据挖掘领域资深专家和科研人员，10余年大数据挖掘咨询与实施经验结晶。从数据挖掘的应用出发，以电力、航空、医疗、互联网、生产制造以及公共服务等行业真实案...

[用户3157710](https://cloud.tencent.com/developer/user/3157710)

* Python 爬取淘宝商品数据挖掘分析实战

? 作者 孙方辉 本文为CDA志愿者投稿作品，转载需授权 项目内容 本案例选择>> 商品类目：沙发； 数量：共100页 4400个商品； 筛选条件：天猫...

[CDA数据分析师](https://cloud.tencent.com/developer/user/1283928)

* 关于《Python数据挖掘入门与实战》读书笔记一（数据挖掘、机器学习、深度学习、人工智能）

Python数据挖掘、Python机器学习、Python深度学习的书籍买了不少本了，但真正读下来的却很少，为何？涉及高等数学，图论、离散数学等等，对我来说和天文...

[python与大数据分析](https://cloud.tencent.com/developer/user/7190337)

* 慕课网数据分析与挖掘实战（一）-数据获取

其中，urllib，urllib2，requests，scrapy主要用于数据抓取 xpath主要用于分析提取有价值的内容 phantomjs，beauti...

[Meet相识](https://cloud.tencent.com/developer/user/3090887)

* 《数据挖掘与数据化运营实战 思路、方法、技巧与应用》—— 读书笔记

[用户1154259](https://cloud.tencent.com/developer/user/1154259)

* 关于《Python数据挖掘入门与实战》读书笔记三（估计器和近邻算法）

Scikit-learn库，实现了一系列数据挖掘算法，提供通用编程接口、标准化的测试和调参工具，便于用户尝试不同算法对其进行充分测试和查找优参数值。

[python与大数据分析](https://cloud.tencent.com/developer/user/7190337)