**pandas用法及数据预处理实例**

[](https://www.zhihu.com/people/xianyang94)

**1、pandas常用命令**

数据的导入及结构查看：

**import** numpy **as** np

**import** pandas **as** pd

path**=**'train.txt' *#将下载的原始数据放到项目文件夹，即可不用写路径*

data**=**pd**.**read\_csv(path,sep**=**" +",header**=**None, engine**=**'python') *#具体参数可查看官方文档*

**print**("原始数据data", data**.**shape,type(data)) *#原始数据的结构*

查看前十行第一列的值：

**print**(data**.**values[0:10, 0])

数据描述：每列的元素数、均值、标准差、最小值、25%、50%、75%分位数和最大值

**print**(data**.**describe())

前3行的值：

**print**(data**.**head(3))

第一行的值：

**print**(data**.**loc[1])

第一行第一列的值：

**print**(data**.**loc[1, 1])

第0列为1的所有行：

**print**（data[data[0]**==** 1]）

第0列为1或2的所有行：

**print**（data[data[0]**.**isin([1, 2])]）

画出第0列的数据折线图、直方图：

data[0]**.**plot() *#折线图*

data[0]**.**hist() *#直方图*

将第0行第0列的元素替换为10：

data**.**loc[0,0] **=** 10

获取每列的缺失值个数：

n\_missings\_each\_col **=** data**.**apply(**lambda** x: x**.**isnull()**.**sum())

获取第一列的种类、每个种类的个数：

**print**(n\_missings\_each\_col**.**head())

**print**（data[0]**.**value\_counts()）

**2、数据转存**

将数据按列拆分并存储到不同的csv文件中：

转存数据

**def** **save**(data, name):

data**.**to\_csv(name**+** ".csv")

*# 将第一列按不同类保存到不同的表中*

new\_data**=**[]

**for** i **in** range(219): *#测试数据中有219个设备数据，将每个设备数据单独存储*

new\_data **=** data[data[0]**==**i]

save(new\_data, str(i))

文件批量删除小技巧：

（1）删除文件夹中所有.csv文件

path **=** '文件路径'

**import** os

**import** glob

**for** infile **in** glob**.**glob(os**.**path**.**join(path, '\*.csv')):

os**.**remove(infile)

**print**(infile)

（2）删除连续的文件，比如删除文件夹0-218.csv共218个文件

**import** os

**for** i **in** range(0,218):

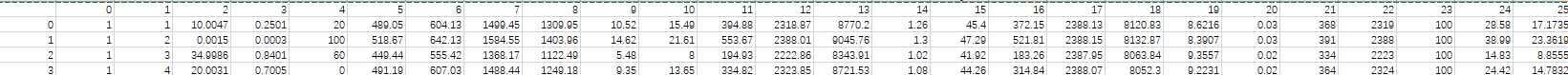
excel\_path **=** r"F:\pycharm program\VAE\{}.csv"**.**format(i)

os**.**remove(excel\_path)

**print**(excel\_path)

**3、聚类分析**

测试数据为单个设备的操作状态数据和监测数据，具体格式如下，第2-3列代表操作状态数据，第5-25列代表监测数据。先将数据按照操作状态进行聚类，接着对每种聚类结果的数据进行归一化处理。

测试数据明细

**from** sklearn.cluster **import** KMeans

*#对第i设备进行聚类*

**def** **cluster**(path, n): *#path为文件路径， n为聚类数量*

data1 **=** pd**.**read\_csv(path, sep**=**",", index\_col**=**0, engine**=**'python')

input **=** data1**.**values[:, 2:5] *#将输入数据的3-5列作为聚类输入数据*

clf **=** KMeans(n\_clusters**=**n, max\_iter**=**300, n\_init**=**10)

y **=** clf**.**fit(input) *#初始化分类器*

o **=** clf**.**cluster\_centers\_ *#样本中心*

lab **=** clf**.**labels\_ *#样本标签*

score **=** clf**.**inertia\_ *#距离，越小聚类结果越好*

**print**("初始化分类器", y)

**print**("样本中心", o**.**shape)

**print**("距离", score)

**print**("样本标签大小", lab**.**shape)

**print**("样本标签", lab)

data2 **=** data1**.**values[:, 5:] *#从输入数据的第5列取值，过滤掉聚类数据*

result **=** np**.**insert(data2, 0, values**=**lab, axis**=**1) *# 将分类结果插入第一列*

order **=** np**.**arange(0, 223, dtype**=**np**.**int) *#为聚类数据加上顺序*

cluster\_result **=** np**.**insert(result, 0, values**=**order, axis**=**1) *#将第一列设为数据顺序*

cluster\_result **=** pd**.**DataFrame(cluster\_result) *#聚类后的数据转换为Dataframe型*

**print**("聚类数据result\_oldr", cluster\_result**.**shape)

**return** cluster\_result *#聚类结果*

**4、数据标准化**

将上述聚类后的数据作为标准化数据的输入，根据不同的聚类结果对数据进行标准化

**from** sklearn **import** preprocessing

*#数据标准化*

**def** **scaler**(data, n): *#data为输入数据，n为数据的类别*

z\_result **=** pd**.**DataFrame() *#定义一个空的dataframe*

z\_scaler **=** preprocessing**.**StandardScaler() *# 建立 StandardScaler 对象*

**for** i **in** range(n):

new\_data **=** data[data[1] **==** i] *#取不同类别的数据*

z\_data **=** z\_scaler**.**fit\_transform(new\_data**.**values[:, 2:]) *#数据标准化（从第三列开始）*

z\_data **=** pd**.**DataFrame(z\_data) *#将数据转为Dataframe*

older\_data **=** pd**.**DataFrame(new\_data**.**values[:, 0:2]) *#取出数据的顺序和聚类结果*

z\_data **=** pd**.**concat([z\_data, older\_data], axis**=**1, join\_axes**=**[z\_data**.**index]) *#将数据的顺序和类别插入到后两行*

z\_result **=** z\_result**.**append(z\_data) *#按种类汇总标准化后的数据*

z\_result**.**columns **=** list(range(23)) *#更改列名*

z\_result **=** z\_result**.**rename({21: 'older', 22: 'cat'}, axis**=**'columns') *#更改列名*

**return** z\_result

**5、数据处理的结果**

将原始数据和处理后的数据分别进行部分展示，可以观察到原始数据是杂乱无章的，处理后的数据可以呈现一定的规律性，可以用于后续的研究中。该数据处理流程还应包括缺失值、异常值处理、滤波等操作，后续会继续进行改进研究。

**from** matplotlib.font\_manager **import** FontProperties *#字体设置*

font\_set **=** FontProperties(fname**=**r"c:\windows\fonts\simsun.ttc", size**=**15) *#字体设置*

path **=** '单个设备的数据路径'

cluster\_result **=** cluster(path, 6) *#聚类*

z\_result **=** scaler(cluster\_result, 6) *#按聚类结果进行标准化*

result **=** z\_result**.**sort\_values(by**=**"older", ascending**=**True) *#按照时间进行排序*

data1 **=** pd**.**read\_csv(path, sep**=**",", index\_col**=**0, engine**=**'python')

plt**.**subplot(2, 1, 1)

plt**.**title("处理前设备1某个监测数据", fontproperties**=**font\_set)

plt**.**plot(range(0, 223, 1), data1**.**values[:, 6])

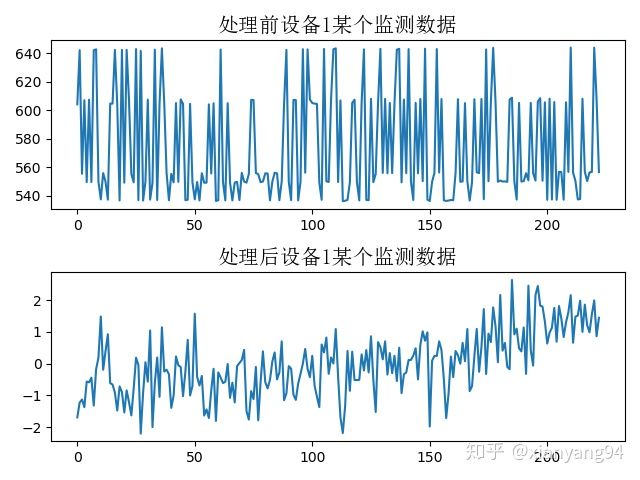
plt**.**subplot(2, 1, 2)

plt**.**title("处理后设备1某个监测数据", fontproperties**=**font\_set)

plt**.**plot(range(0, 223, 1), result**.**values[:, 1])

plt**.**show()

结果展示如下：



发布于 2020-06-11 08:40

[数据预处理](https://www.zhihu.com/topic/19965355" \t "_blank)

[Pandas(Python)](https://www.zhihu.com/topic/20179633" \t "_blank)

[Python 库](https://www.zhihu.com/topic/19644560" \t "_blank)