

**计算机与信息 学院实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验课程： | Python数据分析 | | | | |
| 实验编号： | 实验八 | | | | |
| 实验名称： | pandas数据处理与分析 | | | | |
| 实验人员： | 学号 | 18111207248 | | | |
| 姓名 | 吴钰 | | | |
| 班级 | 18计科创新班 | | | |
| 实验日期： | 2020.11.30 | | | | |
| 实验室： | 2060402 | | | | |
|  |  | | | | |
| 实验评价： |  | | | | |
| 实验成绩： | |  | 评价日期： |  |
|  | 指导教师： | |  | | |

# 一、实验目的

# 1. 掌握对缺失值数据处理的方法。

# 2. 掌握主键合并的几种方法。

# 3. 掌握数据标准化的方法。

# 4. 掌握时间信息提取的方法及其时间数据的算术运算。

# 5. 掌握分组聚合的处理方法与步骤。

# 二、实验工具

# Spyder(Python3.7)

# 三、实验要求

1. 用户用电量数据呈现一定的周期性关系，missing\_data.csv表中存放了用户A、用户B和用户C的用电量数据，其中存在缺失值，请使用Scipy库中interpolate模块中的拉格朗日插值法对缺失值进行插补，并检测插值操作是否成功。

2. 线路线损数据、线路用电量趋势下降数据和线路告警数据是识别用户窃漏电与否的3个重要特征。请读取ele\_loss.csv与alarm.csv表，查看两表的形状，以线路编号ID和时间date两个键值作为主键进行内连接，查看合并后的数据。

3. 算法的种类非常多，当涉及到空间距离计算、梯度下降等，就必须进行标准化处理。读取model.csv，定义标准差标准化函数，使用函数分别对线路线损特征、线路用电量趋势下降特征、线路告警特征进行标准化，查看标准化后的数据。

4. 用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo中均存在大量的时间数据，提取时间数据内存在的信息，一方面可以加深对数据的理解，另一方面能够探索这部分信息和目标的关联程度。同时用户登录时间、借款成交时间、用户信息更新时间这些时间的时间差信息也能够反映出P2P网络贷款不同用户的行为信息。（1）使用to\_datetime函数转换用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo的时间字符串。（2）使用year、month、week等方法提取用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo中的时间信息。（3）计算用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo中两时间的差，以分钟计算。

5. 分析用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo时，还可以结合用户编号进行分组聚合，然后进行组内分析。通过组内分析得出每组组内的最早和最晚信息更新时间、最早和最晚登录时间、信息更新的次数、登录的次数等信息。

# 四、实验内容

1. 用户用电量数据呈现一定的周期性关系，missing\_data.csv表中存放了用户A、用户B和用户C的用电量数据，其中存在缺失值，请使用Scipy库中interpolate模块中的拉格朗日插值法对缺失值进行插补，并检测插值操作是否成功。

代码：

#拉格朗日插值

#dropna().index用于记录非缺失值的下标

#dropna().values用于记录非缺失值的实际值

import pandas as pd

import numpy as np

from scipy.interpolate import lagrange

arr=np.array([0,1,2])

missing\_data=pd.read\_csv("D:/大三（上）专业课/Python/实验8/missing\_data.csv",names=arr)

#查询缺失值所在位置

print("lagrange插值前（False为缺失值所在位置）",'\n',missing\_data.notnull())

for i in range(0,3):

#"训练"lagrange模型

la=lagrange(missing\_data.loc[:,i].dropna().index,missing\_data.loc[:,i].dropna().values)

#list\_d用于记录当前列缺失值所在的行（记录缺失值下标）

list\_d=list(set(np.arange(0,21)).difference(set(missing\_data.loc[:,i].dropna().index)))

#将缺失值list\_d带入训练好的模型，并填入对应的位置

missing\_data.loc[list\_d,i]=la(list\_d)

print("第%d列缺失值的个数为:%d"%(i,missing\_data.loc[:,i].isnull().sum()))

print("lagrange插值后（False为缺失值所在位置）","\n",missing\_data.notnull())

运行截图：



2. 线路线损数据、线路用电量趋势下降数据和线路告警数据是识别用户窃漏电与否的3个重要特征。请读取ele\_loss.csv与alarm.csv表，查看两表的形状，以线路编号ID和时间date两个键值作为主键进行内连接，查看合并后的数据。

代码：

import pandas as pd

ele\_loss=pd.read\_csv("D:/大三（上）专业课/Python/实验8/ele\_loss.csv")

alarm=pd.read\_csv("D:/大三（上）专业课/Python/实验8/alarm.csv", encoding='utf8')

#合并前查看两个表的形状

print("ele\_loss表的形状为:",ele\_loss.shape)

print("alarm表的形状为: ",alarm.shape)

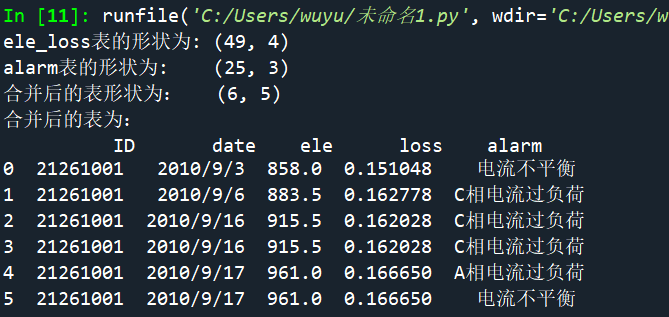
#合并后

merge=pd.merge(ele\_loss,alarm,left\_on=["ID","date"],right\_on=["ID","date"],how="inner")

print("合并后的表形状为： ",merge.shape)

print("合并后的表为：\n",merge)

运行截图：



3. 算法的种类非常多，当涉及到空间距离计算、梯度下降等，就必须进行标准化处理。读取model.csv，定义标准差标准化函数，使用函数分别对线路线损特征、线路用电量趋势下降特征、线路告警特征进行标准化，查看标准化后的数据。

代码：

import pandas as pd

model=pd.read\_csv("D:/大三（上）专业课/Python/实验8/model.csv",encoding="utf8")

#定义标准差标准化函数

def Standard(data):

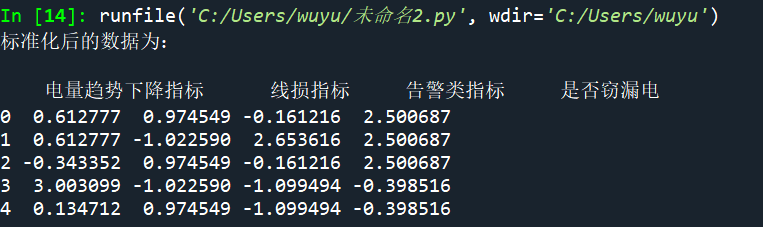
data=(data-data.mean())/data.std()

return data

S=Standard(model)

print("标准化后的数据为：\n",'\n',S.head())

运行截图：



4. 用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo中均存在大量的时间数据，提取时间数据内存在的信息，一方面可以加深对数据的理解，另一方面能够探索这部分信息和目标的关联程度。同时用户登录时间、借款成交时间、用户信息更新时间这些时间的时间差信息也能够反映出P2P网络贷款不同用户的行为信息。（1）使用to\_datetime函数转换用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo的时间字符串。（2）使用year、month、week等方法提取用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo中的时间信息。（3）计算用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo中两时间的差，以分钟计算。

实验代码：

#第四题

import pandas as pd

from datetime import datetime

from datetime import timedelta

LogInfo=pd.read\_csv('D:\大三（上）专业课\Python\实验8/Training\_LogInfo.csv',encoding='utf8')

Userupdate=pd.read\_csv('D:\大三（上）专业课\Python\实验8/Training\_Userupdate.csv',encoding='utf8')#转换时间字符串

Userupdate['ListingInfo1']=pd.to\_datetime(Userupdate['ListingInfo1'])

Userupdate['UserupdateInfo2']=pd.to\_datetime(Userupdate['UserupdateInfo2'])

print('转换用户信息更新表的时间字符串后：\n',Userupdate)

LogInfo['Listinginfo1']=pd.to\_datetime(LogInfo['Listinginfo1'])

LogInfo['LogInfo3']=pd.to\_datetime(LogInfo['LogInfo3'])

print('转换登录信息表的时间字符串后:\n',LogInfo)

#提取用户信息更新表和登录信息表中的时间信息

# 定义一个提取用户信息的函数

def extract(file,time):

global year

year = [i.year for i in file[time]]

month = [i.month for i in file[time]]

week = [i.week for i in file[time]]

return year,month,week

#登录信息表

print('提取登录信息表中的时间信息：\n')

ETLog1 = extract(LogInfo,'Listinginfo1')

print('[年][月][周](前十个数据)：\n',ETLog1[0][0:10],ETLog1[1][0:10],ETLog1[2][0:10])

ETLog2 = extract(LogInfo,'LogInfo3')

print('[年][月][周](前十个数据)：\n',ETLog2[0][0:10],ETLog2[1][0:10],ETLog2[2][0:10])

#用户信息更新表

print('\n')

print('提取用户信息更新表中的时间信息：\n')

ETUser1 = extract(Userupdate,'ListingInfo1')

print('[年][月][周](前十个数据)：\n',ETUser1[0][0:10],ETUser1[1][0:10],ETUser1[2][0:10])

ETUser2 = extract(Userupdate,'UserupdateInfo2')

print('[年][月][周](前十个数据)：\n',ETUser2[0][0:10],ETUser2[1][0:10],ETUser2[2][0:10])

#计算用户信息更新表和登录信息表中两时间的差

TDLog=(LogInfo['Listinginfo1']-LogInfo['LogInfo3'])

TDUser=(Userupdate['ListingInfo1'] - Userupdate['UserupdateInfo2'])

def TransformDayIntoMinute(data):

for i in range(len(data)):

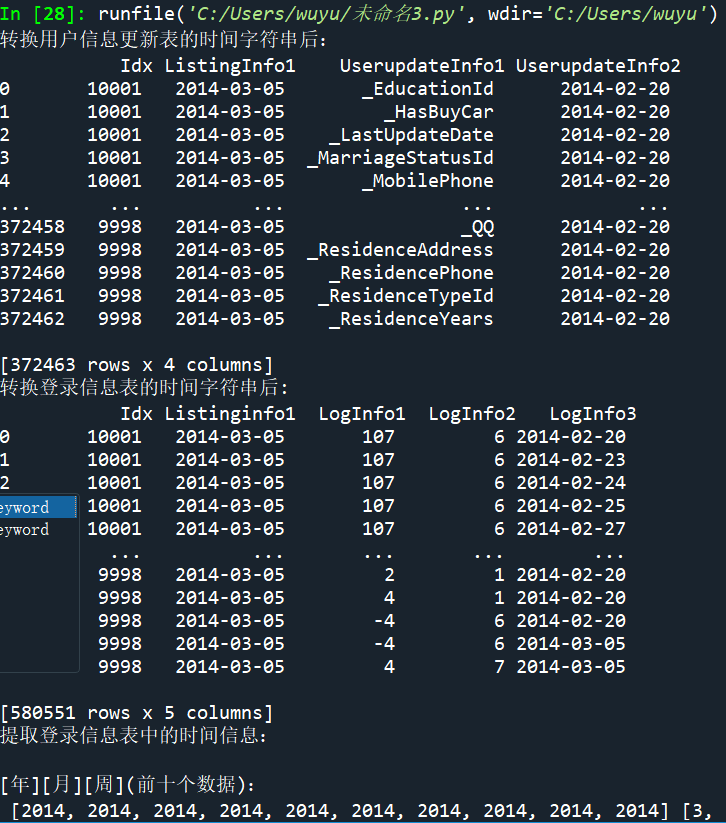
data[i]=data[i].total\_seconds()/60

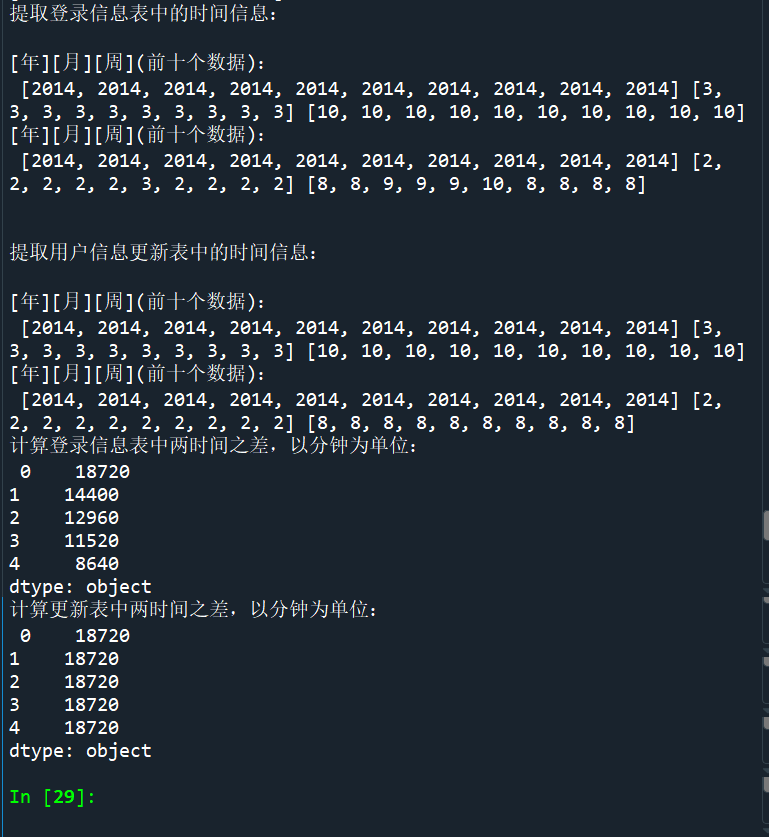
return data

print("计算登录信息表中两时间之差，以分钟为单位：\n",TransformDayIntoMinute(TDLog).head())

print("计算更新表中两时间之差，以分钟为单位：\n",TransformDayIntoMinute(TDUser).head())

运行截图：





5. 分析用户信息更新表Training\_Userupdate和登录信息表Training\_LogInfo时，还可以结合用户编号进行分组聚合，然后进行组内分析。通过组内分析得出每组组内的最早和最晚信息更新时间、最早和最晚登录时间、信息更新的次数、登录的次数等信息。

代码：

import pandas as pd

import numpy as np

Userupdate=pd.read\_csv("D:\大三（上）专业课\Python\实验8/Training\_Userupdate.csv",encoding="utf8")

LogInfo=pd.read\_csv("D:\大三（上）专业课\Python\实验8/Training\_LogInfo.csv",encoding="utf8")

#groupby方法对用户信息更新表和登录信息表进行分组

UserupdateGroup=Userupdate[["Idx","UserupdateInfo2"]].groupby(by="Idx")

UselogGroup=LogInfo[["Idx","LogInfo3"]].groupby(by="Idx")

LogInfoGroup=LogInfo[["Idx","LogInfo3"]].groupby(by="Idx")

print('分组后的用户信息更新表为：\n',UserupdateGroup.head())

print('分组后的登录信息表为：\n',LogInfoGroup.head())

#agg方法求取分组后的最早和最晚更新及登录时间

print("分组后最早更新时间：\n",UserupdateGroup.agg(np.min).head())

print("分组后最晚更新时间：\n",UserupdateGroup.agg(np.max).head())

print("分组后最早登录时间：\n",UselogGroup.agg(np.min).head())

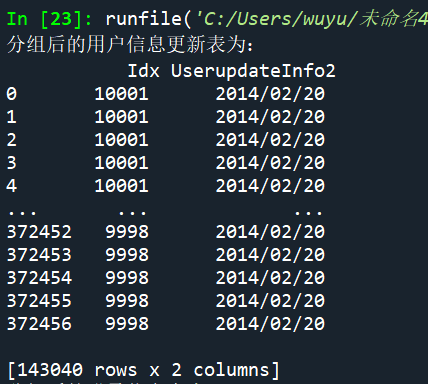
print("分组后最晚登录时间：\n",UselogGroup.agg(np.max).head())

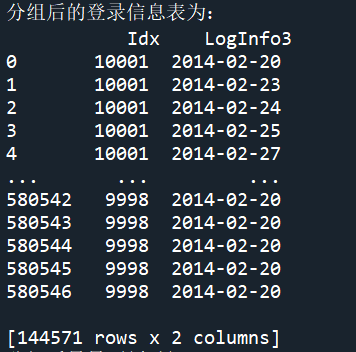
#size方法求取分组后的数据的信息更新次数与登录次数

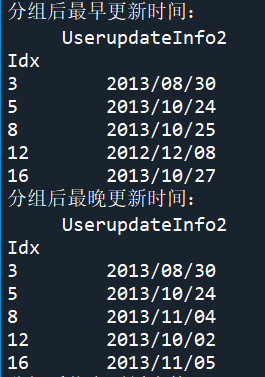
print("分组后信息更新次数：\n",UserupdateGroup.size().head())

print("分组后登录次数：\n",LogInfoGroup.size().head())

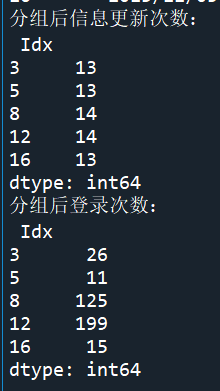
运行截图：

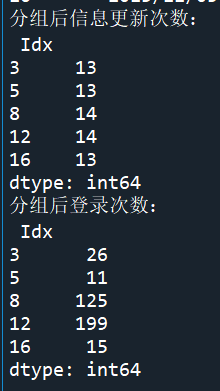












# 五、实验思考

1、标准化建模除了标准差标准化还有离差标准化、小数定标差标准化，三种标准化各有其优势，离差标准化方法简单，便于理解，标准化后的数据限定在[0,1]区间内。标准差标准化受数据分布的影响较小。小数定标标准化方法的适用范围广，并且受数据分布的影响较小，相比较于前两种方法，该方法适用程度适中。

2、分组分析

根据分组字段，将分析对象划分成不同的部分，以进行对比分析各组之间差异性的一种分析方法（定性分组、定量分组）

分组统计函数：

groupby(by=[分组列1,分组列2,...])

[统计列1,统计列2,...]

.agg({统计列别名1:统计函数1,统计列别名2:统计函数2,...})

其中，

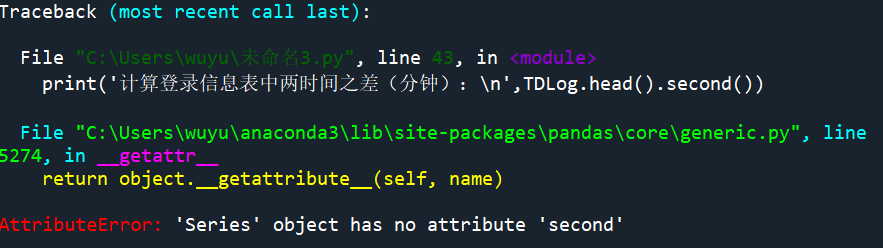
by 用于分组的列

中括号 用于统计的列

agg 统计别名显示统计值的名称，统计函数用于统计数据

3、遇到的问题：

两个datatime类型的日期之差得到的时间差是days为单位，在转换成minute为单位的时候，直接调用方法的时候会出错



解决方法：

写一个方法，将原本的数据转化成一个数组类型，对每一个数据进行操作就不会出现上述问题

