

2022 年“泰迪杯”数据分析职业技能赛 A 题 竞赛作品的自动评判 分析报告

摘要

随着信息化的进一步发展, 对各类学科网络竞赛往往需要参赛者提交 Excel 或/和 PDF 格式的竞赛作品。评判组面对收集而来的众多参赛作品, 如果采用人工评阅的方式需要消耗大量的精力, 因此为了提高对参赛作品评分工作的效率和减轻评判人员的负担, 采用 Python 语言编程实现对竞赛作品的自动评判系统可以极大的缩减了评阅时间, 提高了评判效率。本次赛题以某届数据分析竞赛作品的评阅为背景, 对附件的所有作品的得分数据进行处理, 根据所给定的评分准则和标准答案, 使用 Python 编程完成竞赛作品的自动评判。

Abstract

With the further development of informatization, online competitions for various disciplines often require participants to submit competition works in Excel or/and PDF format. In the face of the large number of entries collected, if the manual review method requires a lot of energy, so in order to improve the efficiency of scoring the entries and reduce the burden of the judges, the use of Python language programming to realize the automatic evaluation system of the competition works can greatly reduce the review time and improve the judging efficiency. This competition question is based on the evaluation of works in a data analysis competition, the scoring data of all works attached to it is processed, and the automatic evaluation of competition works is completed using Python programming according to the given scoring criteria and standard answers.

目录

摘要.....	1
1 目标.....	4
1.1 解题思路 and 流程.....	4
1.1.1 XXX.....	4
1.2 解题思维.....	4
2 任务分析.....	4
2.1 任务 1 基本处理.....	4
2.1.1 任务 1.1.....	5
2.2 任务 2 数据分析.....	9
2.2.1 任务 2.1.....	9
2.2.2 任务 2.2.....	12
2.2.3 任务 2.3.....	12
2.3 任务 3 相似矩阵评分.....	12
2.3.1 任务 3.1.....	13
2.3.2 任务 3.2.....	13
2.3.3 任务 3.3.....	14
2.3.4 任务 3.4.....	15
2.3.5 任务 3.5.....	16
2.4 任务 4 评分结果汇总.....	16
2.4.1 任务 3.1.....	16
2.4.2 任务 3.2.....	16
2.4.3 任务 4.3.....	17
2.4.4 任务 4.4.....	17
2.5 任务 5 嵌套压缩文件处理.....	17
2.5.1 任务 5.1.....	17
2.5.2 任务 5.2.....	18
2.5.3 任务 5.3.....	20
参考文献.....	23

1 目标

1. 使用 Python 解压压缩文件，从中读取指定的文件。
2. 使用 Python 解析 PDF 文件，获取其中的图片信息。
3. 使用 Python 解析 Excel 和 PDF 文件，对数据进行处理与统计，根据评分准则对每份作品打分，并输出报表。

1.1 解题思路 and 流程

正文正文正文

1.1.1 XXX

正文正文正

1.2 解题思维

[图片]

图1-1 解题流程思维导图

正文正文正文正文正文正文正文正文正文正文正文正文正文正文正文正文
文正文正文正文

2 任务分析

2.1 任务 1 基本处理

基于任务 1 各点的要求，首先要对附件压缩文件中“DataA.rar”里所有待评分的作品使用 Python 进行基本处理，进行解压，因为每份作品是以作品号为文件

名、包含若干结果文件的压缩文件，这些压缩文件的文件格式有 rar、zip 或 7z 三种不同类型。因此，如何分别自动解压这三种压缩包对其进行文件读取和操作是解决问题的方向。

思维导图：

2.1.1 任务 1.1

常用的压缩格式有很多种，而不同的压缩包的解压需要用到不同的库。针对压缩文件“DataA.rar”中有 rar、7z、zip 三种压缩格式，通过 Python 首先导入解压各类压缩包的模块，python 可以解压缩五种文件：“.gz, .tar, .tgz, .zip, .rar”。处理思路：首先输入总压缩包“DataA.rar”的路径，解压该路径下的所有压缩文件。即把压缩包里的所有作品解压到“DataA”文件夹的同名子文件夹中：

```
def undataA():  
    # 解压DataA  
    if not os.path.isdir("D:\\桌面\\泰迪杯数据分析A题\\DataA"):  
        os.mkdir("D:\\桌面\\泰迪杯数据分析A题\\DataA")  
    # 放置DataA解压缩文件  
    if not os.path.isdir("D:\\桌面\\泰迪杯数据分析A题\\unDataA"):  
        os.mkdir("D:\\桌面\\泰迪杯数据分析A题\\unDataA")  
  
    path_rar = "D:\\桌面\\泰迪杯数据分析A题\\DataA.rar"  
  
    path_folder = "D:\\桌面\\泰迪杯数据分析A题\\DataA"  
  
    temp = rarfile.RarFile(path_rar) # 待解压文件  
  
    temp.extractall(path_folder)
```

建立一个列表，然后将该路径下的所有子文件名放入列表中：

```
path = r'D:\桌面\泰迪杯数据分析A题\DataA'  
file_lst = glob.glob(path + '/*')  
filename_lst = [os.path.basename(i) for i in file_lst]  
un_files(filename_lst);
```

放入 for 语句循环里进行切割字符串，获取文件名后缀，获取到“zip”、“7z”、“rar”三种类型：

```
suffix = filename.split('.')[-1]
if suffix == 'zip':
    unzip(filename)
    # os.remove(filename)
if suffix == 'rar':
    unrar(filename)
    # os.remove(filename)
if suffix == '7z':
    un7z(filename)
    # os.remove(filename)
```

随后再将所得到的这三种压缩包分别放入相应的解压模块进行解压。

zip:

```
def unzip(filename):
    suffix = filename.split('.')[-1]
    print(suffix)
```

7z:

```
def un7z(filename):
    suffix = filename.split('.')[-1]
    print(suffix)
```

rar:

```
def unrar(filename):
    suffix = filename.split('.')[-1]
    print(suffix)
```

2.1.2 任务 1.2

用 for 循环语句实现遍历文件夹文件并在每个文件夹下创建 image 文件夹。
部分代码如下：

```
for file in fileList:
    if not os.path.isdir(r'D:\桌面\泰迪杯数据分析A题\unDataA\' + file + '\image'):
        os.mkdir(r'D:\桌面\泰迪杯数据分析A题\unDataA\' + file + '\image')
```

2.1.3 任务 1.3

解压所有作品之后，遍历各个作品文件夹中是否包含有相应的文件，即“task2_1.xlsx” “task2_2.xlsx” “task2_3.pdf”及“task3.xlsx”四个文件，命名四个不同的变量以此存储四个不同的得分点文件名。

```
#待搜索的名称
filename1 = "task2_1.xlsx"
filename2 = "task2_2.xlsx"
filename3 = "task2_3.pdf"
filename4 = "task3.xlsx"
```

此处先定义一个空列表记录得分情况：

```
#定义空列表
theresult = []
```

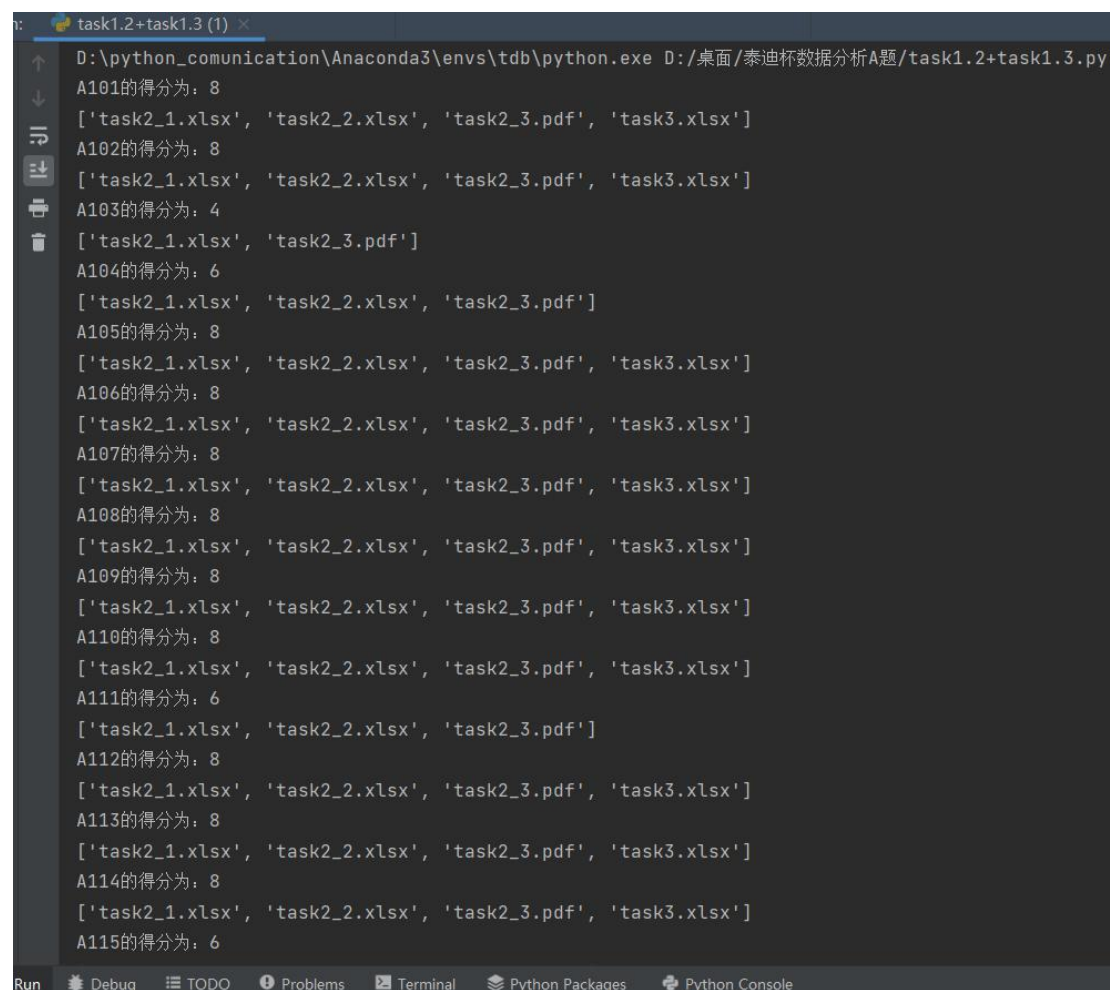
调用函数 *def findfiles(path)*:先遍历当前目录（DataA）所有文件及文件夹，再进入循环判断每个元素是文件还是文件夹，若是文件夹则递归。若是文件则判断该文件是否为特定文件名称，根据相应的特定文件名称个数获得相应的得分，存入空列表中。

```
# 判断是否是文件夹
if os.path.isdir(curpath):
    findfiles(curpath)
else:
    # 判断是否是特定文件名称
    if filename1 in file:
        theresult.append(file)
    elif filename2 in file:
        theresult.append(file)
    elif filename3 in file:
        theresult.append(file)
    elif filename4 in file:
        theresult.append(file)
```

最后，将空列表存储的特定文件名称个数，将该变量名传递给 `len()` 函数长度计算每一个作品的对应的得分。（每包含一个得 2 分，满分为 8 分。）

```
for file in fileList:
    theresult = []
    print(file)
    findfiles(path + '\\ ' + file)
    print('得分: ' + str(len(theresult) * 2))
    print(theresult)
```

对 DataA 里所有文件夹运行结果判断得分：



```
task1.2+task1.3 (1) x
D:\python_communication\Anaconda3\envs\tdb\python.exe D:/桌面/泰迪杯数据分析A题/task1.2+task1.3.py
A101的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A102的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A103的得分为: 4
['task2_1.xlsx', 'task2_3.pdf']
A104的得分为: 6
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf']
A105的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A106的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A107的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A108的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A109的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A110的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A111的得分为: 6
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf']
A112的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A113的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A114的得分为: 8
['task2_1.xlsx', 'task2_2.xlsx', 'task2_3.pdf', 'task3.xlsx']
A115的得分为: 6
```

2.1.4 任务 1.4

作品号 A118~A120 的运行结果截图

```
A119_1.png
DataA/A119/image/A119_1.png
(554, 434)
A278
A120_1.png
DataA/A120/image/A120_1.png
(741, 593)
A279
A120_2.png
DataA/A120/image/A120_2.png
(741, 593)
A360
A120_3.png
DataA/A120/image/A120_3.png
(600, 464)
A361
A120_4.png
DataA/A120/image/A120_4.png
(808, 619)
A362
A120_5.png
DataA/A120/image/A120_5.png
(1311, 663)
A363
A120_6.png
DataA/A120/image/A120_6.png
(1311, 663)
A364
```

2.2 任务 2 数据分析

2.2.1 任务 2.1

通用名称评分

调取 pandas、numpy 库，读取附件中 “criteria2_1.xlsx” 作为标准文件，记录 “criteria2_1.xlsx” 文件中的行数：

```
data = pd.read_excel("criteria2_1.xlsx")
index = data.shape[0] #行数
```

使用 for 循环对 DataA 文件夹里的所有子文件夹进行读取，这一步用于判断每个作品里面是否有“task2_1.xlsx”文件存在：

```
# 循环文件夹
for i in range(left, right):
    s=0
    if os.path.exists("dataA/A" + str(i) + "/task2_1.xlsx"):
```

若判断存在，则进入下一个 for 循环，将“criteria2_1.xlsx”文件里记录的每行的“正式登记证号（[j, 9]）”记录为 s1，将读取到每个“task2_1.xlsx”里的“正式登记证号（[k, 9]）”记录为 s2，随后将 s1 和 a2 进行匹配：

```
for j in range(index):
    s1 = str(data.iloc[j, 9]).replace("/t", "").replace(" " , "")

    for k in range(data_task2_1.shape[0]):
        s2 = str(data_task2_1.iloc[k, 9]).replace("/t", "").replace(" " , "") #?
```

匹配之后根据下列三个规则进行错误数 s 的记录（此前 s=0）

#问题重现 (1)：对匹配的记录，判断“产品通用名称”是否一致，如不一致，错误数 s 加 1。

比对两个表格的“产品通用记录”，此为确定错误数 s 的第一条规则

```
s3 = str(data.iloc[j, 2]).replace("/t", "").replace(" " , "")
s4 = str(data_task2_1.iloc[k, 2]).replace("/t", "").replace(" " , "")
if s3 == s4:
    break
else:
    s += 1
```

#问题重现 (2)：对“criteria2_1.xlsx”中的每条记录，查找“task2_1.xlsx”，如没有匹配的记录，错误数 s 加 1。

将“criteria2_1.xlsx”文件记录每一行与遍历得到每个“task2_1.xlsx”的所有行进行匹配。此为确定错误数 s 的第二条规则。

```

# 循环文件夹
for i in range(left, right):
    s=0
    if os.path.exists("dataA/A" + str(i) + "/task2_1.xlsx"):
        data_task2_1 = pd.read_excel("dataA/A" + str(i) + "/task2_1.xlsx")
        name = "A" + str(i)
        for j in range(index):
            flag = False
            s1 = str(data.iloc[j, 9]).replace("/t" "\t").replace(" " "")
            for k in range(data_task2_1.shape[0]):
                s2 = str(data_task2_1.iloc[k, 9]).replace("/t" "\t").replace(" " "")
                if s1 == s2:
                    flag=True
                    break
            if flag==False:
                s+=1
        if s==0:

```

#问题重现 (3): 对“task2_1.xlsx”中的每条记录, 查找“criteria2_1.xlsx”, 如没有匹配的记录, 错误数 s 加 1。

与问题 (2) 相反, 先遍历每个“task2_1.xlsx”中的每条记录, 抽取出来与“criteria2_1.xlsx”每一行进行匹配。

```

# 循环文件夹
for i in range(left, right):
    s=0
    if os.path.exists("dataA/A" + str(i) + "/task2_1.xlsx"):
        data_task2_1 = pd.read_excel("dataA/A" + str(i) + "/task2_1.xlsx")
        name = "A" + str(i)
        for j in range(index):
            flag = False
            s1 = str(data.iloc[j, 9]).replace("/t" "\t").replace(" " "")
            for k in range(data_task2_1.shape[0]):
                s2 = str(data_task2_1.iloc[k, 9]).replace("/t" "\t").replace(" " "")
                if s1 == s2:
                    flag=True
                    break
            if flag==False:
                s+=1
        if s==0:

```

最后依据每个作品的错误数 s 的数值, 输入相应的得分区间, 得到该作品通用名称的得分。错误数 s 的代码如下:

```

if s==0:
    res[name] = 15
elif s>=1 and s<=10:
    res[name] = 10
elif s>=11 and s<=20:
    res[name] = 5
else:
    res[name] = 0

```

2.2.2 任务 2.2

分组标签评分

#问题重现 (1): 对匹配的记录, 判断“分组标签”中的数值和顺序是否一致, 如不一致, 错误数 s 加 1。

#问题重现 (2): 对“criteria2_2.xlsx”中的每条记录, 查找“task2_2.xlsx”, 如没有匹配的记录, 错误数 s 加 1。

#问题重现 (3): 对“task2_2.xlsx”中的每条记录, 查找“criteria2_2.xlsx”, 如没有匹配的记录, 错误数 s 加 1。

该题的做法思路与上面任务 2.1 基本一致, 可参考任务 2.1 详细解题过程。

最后同样依据各个作品错误数 s 的数值, 输入相应条件的得分区间, 得到各个作品分组标签的得分。

2.2.3 任务 2.3

基于任务 1.3 的判断结果, 遍历所有含“task2_3.pdf”文件的作品,
def pdf2pic(path, picpath, filename):, 解析 PDF 文件, 从 PDF 中提取图片。提取图片表格。基于产品登记数量的排名来判断每个排名判断“分组标签”和“产品登记数量”的数值是否与标准答案一致。共有 6 个数值, 每个数值匹配则得 2 分。

2.3 任务 3 相似矩阵评分

这道题目利用 numpy 和 pandas 库就可以解决。

2.3.1 任务 3.1

```
data = pd.read_excel("criteria3.xlsx")
shape = str(data.shape)
ndim = str(data.ndim)

def juzhen(left, right):
    for i in range(left, right):
        name = "../dataA/A" + str(i) + "/task3.xlsx"
        s = 0
        if os.path.exists(name):
            data_task2_1 = pd.read_excel("../dataA/A" + str(i) + "/task3.xlsx")
            shape_2 = str(data_task2_1.shape)
            ndim_2 = str(data_task2_1.ndim)
            if shape == shape_2 and ndim == ndim_2:
                print(name+str("分数5"))
            else: print(name+str("分数0"))
```

2.3.2 任务 3.2

先将 criteria3.xlsx 的维度求出，利用 numpy 的 ndim 函数，再判断每个文件夹是否有这个 excel 表 task3.xlsx，再求出它的维度和标准版做比较可以得出结果。

```
import pandas as pd
import numpy as np
import os
data = pd.read_excel("criteria3.xlsx")
s = 0
index = data.shape[0]
column = data.columns.values
# 循环遍历
def id(left, right):
    for i in range(left, right):
        s = 0
        # 判断task3.xlsx是否存在。
        name = "../dataA/A" + str(i) + "/task3.xlsx"
        if (os.path.exists(name)):
            data_task = pd.read_excel(name)
            column_task3 = str(data_task.columns.values)
            # 遍历每个公司id 和这个task3.xlsx中id比较
            for j in column:
                if j in column_task3:
                    continue
                else:
                    s += 1
            else:
                continue
```

循环遍历每个文件夹，检查是否有这个 criteria3.xlsx 表，遍历标准表，标准表的每个记录都和 task3.xlsx 的列名比较，看是否 id 在这个 task3.xlsx 表内

2.3.3 任务 3.3

```
flag = False
name = "../dataA/A" + str(i) + "/task3.xlsx"
s = 0
data = pd.read_excel(name)
l = []
t = data.columns.values
data_col = data.columns.values
for k in range(data.shape[0]):
    l.append(data.loc[k][t[k + 1]])
for o in l:
    if abs(float(float(o)-1)) >= 0.000001:
        flag=True
    else:
        flag = False
if(flag):
    print(name + "分数: 5")
else:print(name+"分数: 0")
```

循环遍历每个文件夹，检查是否有这个 criteria3.xlsx 表，遍历 excel 表，然后拿到矩阵的对角线元素，然后循环进行误差判断，看是否符合条件，再算出分数。

2.3.4 任务 3.4

```
#判断task3.xlsx是否存在。
s = 0
if(os.path.exists("../dataA/A"+str(i)+"/task3.xlsx")):
    data_task = pd.read_excel("../dataA/A"+str(i)+"/task3.xlsx")
    #将列不规范的列删掉，留下有ID的列
    for z in data_task.columns.values:
        if "ID" not in z:
            #删除
            data_task.drop(z, axis=1, inplace=True)
    #转置矩阵
    data_task_T = data_task.T
    data_task_column = data_task.columns.values
    data_task_T_index = data_task_T.index.values
    # # print(data_task_column)
    for j in range(data_task.shape[0]):
        t1=str(data_task.loc[j][data_task_column[j]])
        t2 = str(data_task_T.loc[data_task_T_index[j]][j])
        if t1==t2:
            continue
    ,
```

循环判断 task3.xlsx 文件是否存在，将不规范的 excel 表的列删除，得到需要的数据，然后将矩阵转置，遍历矩阵，判断矩阵和矩阵转置是否相等，来得出是否是主对角线堆成。

2.3.5 任务 3.5

```
#判断task3.xlsx是否存在...
s = 0
if(os.path.exists("../dataA/A"+str(i)+"/task3.xlsx")):
    data_task = pd.read_excel("../dataA/A"+str(i)+"/task3.xlsx")
    #将列不规范的列删掉，留下有ID的列
    for z in data_task.columns.values:
        if "ID" not in z:
            data_task.drop(z, axis=1, inplace=True)
    data_task_columns = data_task.columns.values
    for j in range(data_task.shape[0]-1):
        k = j+1
        #技术绝对误差
        while k<=data_task.shape[0]-1:
            t1=float(data_task.loc[j][data_col[k+1]])
            t=float(data_task.loc[j][data_task_columns[k]])
            sum_res =abs(t1-t)
            # print(sum_res)
            if sum_res>=0.005 and sum_res<0.01:
                s+=0.5
            elif sum_res >=0.01:
                s+=1
            else:sum_res+=1
            k+=1
```

循环判断 task3.xlsx 这个表是否存在，然后将不规范的表进行修改，删除不含运 ID 的列，拿到想要的表，然后通过两次循环，拿出上三角元素，每一次遍历，列的索引+1，就可以拿到上三角元素然后再算出绝对误差，进行比较，得出响应的结果。

2.4 任务 4 评分结果汇总

2.4.1 任务 4.1

代码思路

2.4.2 任务 4.2

代码思路

2.4.3 任务 4.3

代码思路

（将结果放在报告中。）

2.4.4 任务 4.4

代码思路

（将柱形图放在报告中。）

2.5 任务 5 嵌套压缩文件处理

2.5.1 任务 5.1

处理思路与任务 1.1 基本相同。

首先也是输入总压缩包“DataB.rar”的路径，解压该路径下的所有压缩文件。即把压缩包里的所有作品解压到“DataB”文件夹的同名子文件夹中：

```
def undataB():  
    # 解压DataB  
    if not os.path.isdir("D:\\桌面\\泰迪杯数据分析A题\\DataB"):  
        os.mkdir("D:\\桌面\\泰迪杯数据分析A题\\DataB")  
    # 放置DataB解压缩文件  
    if not os.path.isdir("D:\\桌面\\泰迪杯数据分析A题\\unDataB"):  
        os.mkdir("D:\\桌面\\泰迪杯数据分析A题\\unDataB")  
  
    path_rar = "D:\\桌面\\泰迪杯数据分析A题\\DataB.rar"  
  
    path_folder = "D:\\桌面\\泰迪杯数据分析A题\\DataB"  
  
    temp = rarfile.RarFile(path_rar) # 待解压文件  
  
    temp.extractall(path_folder)  
    # 解压DataB  
    if not os.path.isdir("D:\\桌面\\泰迪杯数据分析A题\\unDataB"):  
        undataB()
```

建立一个列表，然后将该路径下的所有子文件名放入列表中：

```
path = r'D:\桌面\泰迪杯数据分析A题\DataB'
file_lst = glob.glob(path + '/*')
filename_lst = [os.path.basename(i) for i in file_lst]
un_files(filename_lst):
```

放入 for 语句循环里进行切割字符串，获取文件名后缀，获取到“zip”、“7z”、“rar”三种类型：

```
suffix = filename.split('.')[-1]
if suffix == 'zip':
    unzip(filename)
    # os.remove(filename)
if suffix == 'rar':
    unrar(filename)
    # os.remove(filename)
if suffix == '7z':
    un7z(filename)
    # os.remove(filename)
```

随后再将所得到的这三种压缩包分别放入相应的解压模块进行解压。

zip:

```
def unzip(filename):
    suffix = filename.split('.')[-1]
    print(suffix)
```

7z:

```
def un7z(filename):
    suffix = filename.split('.')[-1]
    print(suffix)
```

rar:

```
def unrar(filename):
    suffix = filename.split('.')[-1]
    print(suffix)
```

2.5.2 任务 5.2

代码思路与任务 1.2,

解压所有作品之后，遍历各个作品文件夹中是否包含有相应的文件，即“task2_1.xlsx” “task2_2.xlsx” “task2_3.pdf” 及 “task3.xlsx” 四个文件，命名四个不同的变量以此存储四个不同的得分点文件名。

```
#待搜索的名称
filename1 = "task2_1.xlsx"
filename2 = "task2_2.xlsx"
filename3 = "task2_3.pdf"
filename4 = "task3.xlsx"
```

此处先定义一个空列表记录得分情况：

```
#定义空列表
theresult = []
```

调用函数 *def findfiles(path)*:先遍历当前目录（DataA）所有文件及文件夹，再进入循环判断每个元素是文件还是文件夹，若是文件夹则递归。若是文件则判断该文件是否为特定文件名称，根据相应的特定文件名称个数获得相应的得分，存入空列表中。

```
# 判断是否是文件夹
if os.path.isdir(curpath):
    findfiles(curpath)
else:
    # 判断是否是特定文件名称
    if filename1 in file:
        theresult.append(file)
    elif filename2 in file:
        theresult.append(file)
    elif filename3 in file:
        theresult.append(file)
    elif filename4 in file:
        theresult.append(file)
```

bigArr 数组存储了各作品号的得分信息，然后将这个 bigArr 转化为数据框，进而通过 pandas 保存为 excel 文件。

```
bigArr = [] #分作品号存储信息
```

```

obj = {
    'name':file,
    'source':str(len(result) * 2)
}
bigArr.append(obj)
print(result)
print(bigArr)
DataFrame =pd.DataFrame(bigArr)
#unData中创建summary文件夹
if not os.path.isdir(r'D:\\桌面\\泰迪杯数据分析A题\\unDataB\\summary'):
    os.mkdir(r'D:\\桌面\\泰迪杯数据分析A题\\unDataB\\summary')
DataFrame.to_excel("D:\\桌面\\泰迪杯数据分析A题\\unDataB\\summary\\result5_2.xlsx")
print(DataFrame)

```

2.5.3 任务 5.3

任务 5.3 的基本思路在遍历寻找目标文件之后，把该文件信息存储到字典上，把字典放入一个大数组，最后将该大数组变成数据框，再通过 pandas 转化为 excel 文件。


```

# 作品名单元格合并
def to_merge(df):
    # 按照第一列id进行每行单元格合并
    # id列去重，确定一列需要合并成几个值
    print(df)
    df_key = list(set(df['作品号'].values))
    wb = Workbook()
    ws = wb.active
    print("df_key", df_key)
    # 将每行数据写入ws中
    for row in dataframe_to_rows(df, index=False, header=True):
        # print(row)
        ws.append(row)
    # 遍历第一列去重后id
    for i in df_key:
        # 获取id等于指定值的几行数据
        print(i)
        print(df.loc[df["作品号"]==i, :].index.tolist())
        df_id = df.loc[df["作品号"]==i, :].index.tolist() # 索引值从0开始
        for j in range(1, 2): # 遍历 需要合并两列，openpyxl中，读excel等的序号都是从1开始，所以合
            ws.merge_cells(start_row=df_id[0] + 2, end_row=df_id[-1] + 2, start_column=
                           end_column=j) # 序号从1开始，所以行序号需要加2

excel_name = r"D:\桌面\泰迪杯数据分析A题\unDataB\summary\result5_3.xlsx"

```

参考文献

- [1] CSDN博客, pandas用法大全【速查手册】,
链接: <https://blog.csdn.net/Viewinfinitely/article/details/124726749>.
- [2] CSDN博客, Python模块——os模块详解,
链接: https://blog.csdn.net/weixin_41261833/article/details/108047966.
- [3] CSDN博客, 用 Python 压缩文件方法汇总,
链接: https://blog.csdn.net/m0_63394128/article/details/125131301.