基于用户的协同过滤评分预测报告

作者：黄宇辰　学号：10185102253

1-方法介绍

1.基于用户的协同过滤算法模型(UCF)

2.使用User-Business数据集

3.构建User-Business评分矩阵

4.构建User-User相似度评分矩阵

5.利用测试集预测评分

2-核心代码介绍

环境：python3.7

```
import pandas as pd
import numpy as np
import math
```

1.读取数据

读取数据的类，分别用于读取训练数据与测试数据

```
class clearinfo:
    def __init__(self,train_filepath,test_filepath):
        self.train_filepath = train_filepath
        self.test_filepath = test_filepath
    def getinfo_train(self):
        train_data = pd.read_csv(self.train_filepath,header=0)
        return train_data
    def getinfo_test(self):
        test_data = pd.read_csv(self.test_filepath,header=0)
        return test_data
```

2.得到User-Business评分矩阵

将DataFrame类型数据，通过遍历每条评分记录构成二维字典。

如

User	Business	stars
1	a	3.0
2	a	4.0
1	b	2.0

得到如下评分矩阵

```
{
    1:{a:3.0 , b:2.0},
    2:{a:4.0}
}
```

```python
def getUserBus(train_info):
    UB_dict = {}
    for row in train_info.itertuples():
        if row[1] not in UB_dict:
            UB_dict.setdefault(row[1],{})
        UB_dict[row[1]][row[2]]=row[4]
    return UB_dict
```

3.构建共同评分向量

为计算相似度，通过评分矩阵获得两用户共同评分过的business的评分向量

```python
def getsameVec(user1,user2,uudic):
    user1_info = uudic[user1]
    user2_info = uudic[user2]
    #获取两位用户的评分向量
    sameBusiness = []
    for key in user1_info.keys():
        if key in user2_info.keys():
            sameBusiness.append(key)
    x = {}
    y = {}
    for ele in sameBusiness:
        x[ele]=user1_info[ele]
        y[ele]=user2_info[ele]
    return x,y
```

4.计算相似度的三种方法

分别实现了欧几里得距离、余弦相似度、皮尔逊相似度三种相似度计算方式

```python
#欧几里得距离
def euclidean(user1,user2,uudic):
    x,y=getsameVec(user1,user2,uudic)
    distance = 0
    for key in x.keys():
        distance+=(x[key]-y[key])**2
    try:
        distance = 1+math.sqrt(distance)
    except ZeroDivisionError:
        distance = 0
    return distance

#余弦相似度
def cosine(user1,user2,uudic):
    x, y= getsameVec(user1,user2,uudic)
    up=0
    down1=0
    down2=0
    for key in x.keys():
        up += x[key]*y[key]
        down1 += x[key]**2
        down2 +=y[key]**2
    down = (down1*down2)**0.5
    try:
        dis = up/down
    except ZeroDivisionError:
        dis=0
    return dis

#皮尔逊相关系数
def pearson(user1,user2,uudic):
    x,y =getsameVec(user1,user2,uudic)
    length = len(x)
    if length == 0:
        return 0
    sumx=0
    sumy=0
    for key in x.keys():
        sumx+=x[key]
        sumy+=y[key]
    mean1 = sumx/length
    mean2 = sumy/length
    up=0
    down1=0
    down2=0
    for key in x.keys():
        up += (x[key]-mean1)*(y[key]-mean2)
        down1 +=(x[key]-mean1)**2
        down2 +=(y[key]-mean2)**2
    down = (down1*down2)**0.5
    try:
        value = up / down
    except ZeroDivisionError:
        value = 0
    return value
```

5.得到User-User相似度矩阵

以下以欧氏距离相似度矩阵代码为例

```
def getUU_euc(UB):
    UU_dict = {}
    for key in UB.keys():
        UU_dict.setdefault(key,{})
        for key_other in UB.keys():
            if not key == key_other:
                if euclidean(key,key_other,UB) > 0:
#使用欧式距离
                    UU_dict[key][key_other] = euclidean(key,key_other,UB)
        #UU_dict[key] = sorted(UU_dict[key].items(),key = lambda 
x:x[1],reverse=True)
    return UU_dict
```

6.评分预测函数

通过相似度高于0的相似用户对于某一Business的评分进行相似度加权评估进行评分预测，结果保留一位小数

```
def preScore(user,business,UB,UU):
    up = 0
    down = 0
    for simiUser in UU[user].keys():
        if business in UB[simiUser].keys():
            up += UB[simiUser][business]*UU[user][simiUser]
            down += UU[user][simiUser]
    try:
        value = up/down
    except ZeroDivisionError:
        value = 0
    return  round(value,1)
```

3-预测结果

欧氏距离相似度预测结果文件: Result1.csv

余弦相似度预测结果文件: Result2.csv

皮尔逊相似度预测结果文件： Result3.csv

代码文件: UCF.ipynb (基于python3.7)