In the name of God

استاد : دکتر هراتی

دانشجو : توحید حقیقی سیس

شماره دانشجویی : 830598021

موضوع : تمرین هفتم

تمرین اول :

دیتاست تمرین شامل دو فایل است. فایل matrix.txt شامل یک ماتریس کاربر − آیتم R است. تعداد کاربرها ۹۹۸۵ نفر و آیتم‌ها شامل ۵۶۳ برنامه تلوزیونی است. اگر کاربر آیتم را پسندیده باشد $R_{ij} = 1$ و در غیر اینصورت صفر است. فایل items.txt شامل لیست برنامه‌ها با همان ترتیب ستون‌های R است.

می‌خواهیم به کاربر ۵۰۰م (سطر ۴۹۹) تعدادی آیتم برای مشاهده پیشنهاد بدهیم. فرض کنید او هیچکدام از ۱۰۰ آیتم اول را مشاهده نکرده و می‌خواهیم ده آیتم از آنها را به او پیشنهاد بدهیم. این کار را بر اساس رفتار او در مشاهده دیگر آیتم‌ها انجام می‌دهیم.

الف) ابتدا ده آیتم پر طرفدار از این ۱۰۰ آیتم را پیدا کرده و به او پیشنهاد بدهید.

برای شروع ابتدا ماتریس ها را با numpy لود کرده تا بتوانیم عملیات مورد نیاز را روی داده ها انجام دهیم .

```
In [84]: import numpy as np
         import pandas as pd

In [85]: matrix = pd.read_csv('matrix.txt', header = None,sep=' ')

         items = pd.read_csv('items.txt', header = None,sep=' ')

In [86]: matrix
```

Out[86]:

	0	1	2	3	4	5	6	7	8	9	…	553	554	555	556	557	558	559	560	561	562
0	1	1	1	1	1	1	1	1	1	1	…	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	…	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
4	1	1	0	0	1	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
9980	0	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
9981	0	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
9982	0	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
9983	0	0	1	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
9984	0	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0

9985 rows × 563 columns

```
In [87]: len(items)
```

Out[87]: 563

2 تابعی که در این سوال استفاده کردم jacard similarity و bubble sort است که کد پایتون ان به صورت زیر است :

```
In [88]: def jaccard(x,y):
           x = np.asarray(x, np.bool) # Not necessary, if you keep your data
           y = np.asarray(y, np.bool) # in a boolean array already!
           return np.double(np.bitwise_and(x, y).sum()) / np.double(np.bitwise_or(x, y).sum())
```

```
In [89]: matrix=np.nan_to_num(matrix)
```

```
In [90]: def bubbleSort(arr,indexarr):
             n = len(arr)

             # Traverse through all array elements
             for i in range(n-1):
             # range(n) also work but outer loop will repeat one time more than needed.

                 # Last i elements are already in place
                 for j in range(0, n-i-1):

                     # traverse the array from 0 to n-i-1
                     # Swap if the element found is greater
                     # than the next element
                     if arr[j] > arr[j+1] :
                         arr[j], arr[j+1] = arr[j+1], arr[j]
                         indexarr[j],indexarr[j+1] = indexarr[j+1],indexarr[j]

             return arr,indexarr
```

بعضی خانه های ماتریس نال بود که در جمع به مشکل میخورد با یک خط کد وسطی ان را به صفر تبدیل میکند.

برای جل این قسمت یعنی انتخاب 10 ایتم پر طرفدار از 100 ایتم اول به صورت زیر عمل میکنیم .

```
122]: #انتخاب 100 ایتم اول
      column_sums = matrix[:,:100].sum(axis=0)
      #100 تولید یک ارایه با 100 ستون از 1 تا
      range_colums=np.arange(0, 100 , 1)
```

```
123]: # مرتب سازی ستون ها و یافتن ترتیب ان ها
      get_sorted_portarafdar,portarafdar_index=bubbleSort(column_sums,range_colums)
```

```
125]: # انتخاب 10 ایتم پر طرفدار
      for l in range(len(portarafdar_index)-10,len(portarafdar_index)):
          print(str(portarafdar_index[l]))

      62
      72
      82
      68
      5
      9
      60
      45
      96
      74
```

بعد از مرتب سازی بابل و انتخاب 10 تا بیشترین مقدار ان به مقادیر زیر میرسیم .

ستون های : 62 – 72 – 82 – 68 – 5 - 9 - 60 - 45 - 96 - 74

بخش دوم سوال :

یکی از موارد زیر (ب یا ج) را انجام دهید و نتایج را مقایسه و تحلیل کنید.

ب) با استفاده از روش user-user collaborative filtering ، ده آیتم به او پیشنهاد دهید.

ج) با استفاده از روش item-item collaborative filtering ، ده آیتم به او پیشنهاد دهید.

```
In [126]: #jacard similarity distance user and user
          #ارایه فاصله ژاکارد بین وکتور ها
          sorted_list_jacarddistance=[]

          #ارایه ایندکس وکتور ها
          sorted_list_jacarddistance_index=[]

          for i in range(len(matrix)):
              #یافتن فاصله ژاکارد
              sorted_list_jacarddistance.append(jaccard(matrix[i],matrix[499]))
              sorted_list_jacarddistance_index.append(i)
```

در کد بالا فاصله ژاکارد 500 را با تک تک نود ها پیدا میکند.

و در مرحله بعد فاصله ژاکارد رو با روش bubble sort مرتب میکند و بعد 20 تا از ان یوزر هایی که به یوزر 500 شبیه است رو برمیگردانیم .

بعد در این یوزر ها میبینیم چه فیلم هایی دیدن که یوزر 500 ندیده را به ان ها پیشنهاد میدهیم .

در این مثال فقط یک یوزر نزدیک را انتخاب کرده و فیلم هایی که او دیده و 500 ندیده را به او پیشنهاد میدهیم.

```
In [ ]: #مرتب کردن یوزر ها با شباهت ژاکارد
sorted_list,indexsorted_list=bubbleSort(sorted_list_jacarddistance,sorted_list_jacarddistance_index)

In [51]: # بر داشتن 20 یوزر که بیشترین شباهت را به یوزر 500 دارند
top_20_idx = np.argsort(sorted_list_jacarddistance_index)[-1:]
top_20_values = [sorted_list_jacarddistance_index[i] for i in top_20_idx]

In [52]: top_20_idx
Out[52]: array([1612], dtype=int32)

In [53]: array_Offer=[]
for i in top_20_idx:
    print(matrix[i])
    for j in range(len(matrix[i])):
        print(j)
        if(matrix[i][j]==1):
            print('--------------------------')
            if(matrix[499][j]==0):
                print(matrix[499][j])
                array_Offer.append(j)

6
--------------------------
0
7
8
9
--------------------------
0
10
11
--------------------------
0
12
13
--------------------------
0
14
15

In [54]: len(array_Offer)
Out[54]: 24

In [55]: print(np.unique(array_Offer))
[  6   9  11  13  15  25  43  45  46  54  60  64  96 116 119 124 152 248
 316 381 384 393 397 467]
```

ایتم های پیشنهادی به 500 به صورت بالا میشود. میبینیم که با قسمت اول در بخشی شبیه هستند اما در همه شبیه نیستند.