Data Analytics

Assignment -Collaborative filtering-

윤장혁 교수님

산업공학과

201811527

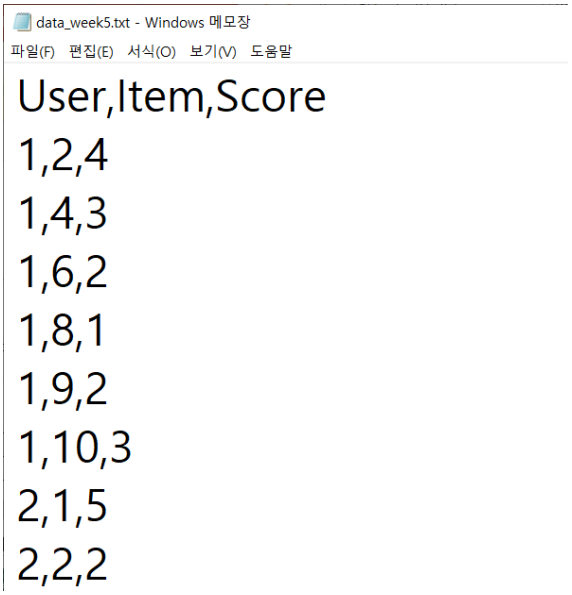
이영은

Week5

■ 주어진 평점 데이터(data_week5.txt)를 활용하여 User X에게 Item 추천

- 데이터는 각 Row가 User, Item, Score(1-5점)으로 구성되어 있습니다.

  data_week5.txt 파일은 다음과 같이 작성되어 있습니다.



  따라서 위의 data들을 분석에 용이하게 하기 위하여 pandas의 Dataframe구조로 바꾸었습니다.

■ data_week5.txt 파일 -> Dataframe으로 변환

결과를 순차적으로 확인하기 위하여 분석 과정에서 jupyter notebook을 사용하였습니다.

```
In [1]: from pandas import Series, DataFrame
        import pandas as pd
        import numpy as np

        f = open(r"C:₩Users₩zdudd₩Desktop₩DA₩20₩week5₩data_week5.txt", encoding='utf8')
        lines = f.readlines()

        item_frame = DataFrame(columns = ['User1','User2','User3','User4','User5','User6','User7','User8','User9','User10'], index = ['Item1','I

        for i in range(1, len(lines)):
            item_frame.ix["Item"+lines[i].split(',')[1]]["User"+lines[i].split(',')[0]] = lines[i].split(',')[2].strip()

        item_frame.fillna(0, inplace = True)

        item_frame
```

- from pandas import Series, DataFrame

  import pandas as pd

  import numpy as np

DataFrame과 Series를 import하기 위하여 numpy와 pandas 라이브러리를 불러왔습니다.

- f = open(r"C:₩Users₩zdudd₩Desktop₩DA₩20₩week5₩data_week5.txt", encoding='utf8')

  lines = f.readlines()

  # f는 week5 과제에서 주어진 평점 데이터 txt파일을 open하였고, 이를 readlines() 함수로 모든 줄을 읽어와, 각각의 줄을 요소로 갖는 리스트 lines를 만들었습니다.

- item_frame = DataFrame(columns = ['User1','User2','User3','User4','User5','User6','User7','User8','User9','User10'], index = ['Item1','Item2','Item3','Item4','Item5','Item6','Item7','Item8','Item9','Item10'])

  # item_frame이라는 DataFrame을 만들어 Column name과 row name을 정하였습니다.

  추후에 Item_based collaborative filtering을 수행하기 위하여 item을 행에 ,User를 열에 두었습니다.

- for i in range(1, len(lines)):

    item_frame.ix["Item"+lines[i].split(',')[1]]["User"+lines[i].split(',')[0]] = lines[i].split(',')[2].strip()

  # 리스트에 저장한 각각의 score를 item과 user를 각 행과 열로 갖는 dataframe에 저장하였습니다.

- item_frame.fillna(0, inplace = True)

  # cosine similarity를 구하기 위해 null값을 갖는 dataframe에 0으로 교체하였습니다.

- item_frame

  # item_frame의 결과는 다음과 같습니다.

Out[1]:

	User1	User2	User3	User4	User5	User6	User7	User8	User9	User10
Item1	0	5	1	0	2	2	1	0	3	1
Item2	4	2	1	0	1	0	3	1	0	2
Item3	0	1	0	4	0	4	0	1	2	1
Item4	3	3	2	5	0	2	2	2	0	3
Item5	0	2	3	0	4	0	5	3	1	4
Item6	2	0	5	0	3	1	0	3	0	0
Item7	0	1	0	3	0	3	2	0	4	0
Item8	1	0	2	0	1	5	0	3	0	2
Item9	2	0	4	3	1	1	1	1	5	1
Item10	3	0	4	0	2	2	4	0	0	1

■ Item 사이의 cosine similarity 구하기

In [2]:
```
from sklearn.metrics.pairwise import cosine_similarity
item_based = cosine_similarity(item_frame)

item_based_collabor = pd.DataFrame(data = item_based, index = item_frame.index, columns = item_frame.index)

item_based_collabor
```

- from sklearn.metrics.pairwise import cosine_similarity

  # cosine similarity를 구하기 위하여 scikit-learn 패키지를 설치하고, sklearn을 통하여 패키지를 import하였습니다.

- item_based = cosine_similarity(item_frame)

  # cosine_similarity를 이용하여 위의 item_frame의 cosine similarty를 구하였습니다.

- item_based_collabor = pd.DataFrame(data = item_based, index = item_frame.index, columns = item_frame.index)

  # item_based_collabor의 결과는 다음과 같습니다.

- item_based_collabor

Out[2]:

	Item1	Item2	Item3	Item4	Item5	Item6	Item7	Item8	Item9	Item10
Item1	1.000000	0.447214	0.477410	0.470016	0.550000	0.279715	0.596762	0.359573	0.485185	0.358391
Item2	0.447214	1.000000	0.133440	0.687184	0.689454	0.457069	0.213504	0.351763	0.412265	0.754247
Item3	0.477410	0.133440	1.000000	0.699062	0.196932	0.161788	0.846154	0.603506	0.583713	0.203810
Item4	0.470016	0.687184	0.699062	1.000000	0.542326	0.420084	0.543715	0.530173	0.599934	0.548795
Item5	0.550000	0.689454	0.196932	0.542326	1.000000	0.580948	0.286446	0.455085	0.480334	0.695701
Item6	0.279715	0.457069	0.161788	0.420084	0.580948	1.000000	0.069338	0.631032	0.582526	0.694022
Item7	0.596762	0.213504	0.846154	0.543715	0.286446	0.069338	1.000000	0.362103	0.708795	0.317038
Item8	0.359573	0.351763	0.603506	0.530173	0.455085	0.631032	0.362103	1.000000	0.412161	0.533002
Item9	0.485185	0.412265	0.583713	0.599934	0.480334	0.582526	0.708795	0.412161	1.000000	0.570756
Item10	0.358391	0.754247	0.203810	0.548795	0.695701	0.694022	0.317038	0.533002	0.570756	1.000000

- **User X에게 Item 추천**

User X에게 Item을 추천해 주는 방식은 다음과 같습니다.

1. User X가 지금까지 Item1 ~ Item10에 대해 점수를 준 score를 입력 받습니다.

   (아직 score가 없는 Item에는 0을 입력합니다.)

2. Item1 ~ Item10중 에서 가장 점수가 높은 Item을 선택합니다.
3. 선택된 Item과 cosine similarity가 높은 Item을 내림차순으로 나열합니다.
4. 이중 평가를 아직 메기지 않은 Item을 추천합니다.

- 코드의 재 사용성을 위하여 User X의 데이터를 input을 사용하여 입력 받았습니다.

```
print("user x의 item score를 입력하세요₩n score가 없을 경우 0을 입력하세요.")
UserX_lst = list()
UserX_lst.append(input("Item1 :"))
UserX_lst.append(input("Item2 :"))
UserX_lst.append(input("Item3 :"))
UserX_lst.append(input("Item4 :"))
UserX_lst.append(input("Item5 :"))
UserX_lst.append(input("Item6 :"))
UserX_lst.append(input("Item7 :"))
UserX_lst.append(input("Item8 :"))
UserX_lst.append(input("Item9 :"))
UserX_lst.append(input("Item10 :"))
```

- 실행 결과는 다음과 같습니다.

```
user x의 item score를 입력하세요
 score가 없을 경우 0을 입력하세요.
Item1 :0
Item2 :2
Item3 :1
Item4 :0
Item5 :3
Item6 :0
Item7 :0

Item8 :
```

- 주어진 User X의 데이터로 입력한 결과는 다음과 같습니다.

```
user x의 item score를 입력하세요
 score가 없을 경우 0을 입력하세요.
Item1 :0
Item2 :2
Item3 :1
Item4 :0
Item5 :3
Item6 :0
Item7 :0
Item8 :4
Item9 :3
Item10 :0

Out[2]: Item8     1.000000
        Item6     0.631032
        Item3     0.603506
        Item10    0.533002
        Item4     0.530173
        Item5     0.455085
        Item9     0.412161
        Item7     0.362103
        Item1     0.359573
        Item2     0.351763
        Name: Item8, dtype: float64
```

- User X가 가장 높은 score를 준 Item은 Item8이기 때문에 User1 ~ User10이 score를 준 Item 사이의 cosine similarity 결과를 토대로 Item8과 유사도가 높은 Item부터 내림차순으로 정렬하였습니다.

- User X 가 추천받을 Item은 Item6입니다. (Item8은 이미 score를 주었기 때문입니다.) Item8과 cosine similarity가 두번째로 높은 Item3은 User X가 이미 score를 주었기 때문에 제외하고, 두번째로는 Item 10을 추천해 주게 됩니다.

- 한계점 : 가장 높은 점수를 준 Item만 고려하여, 다른 Item에 준 점수는 고려하지 못하였습니다. (User X가 Item3에는 score를 1을 주었음에도 불구하고 Item 8과 유사도가 높다는 이유로 2번째로 추천되는 점 등)

- **User X의 score가 달라질 경우**

  - User X의 score data가 주어진 score가 아니라 변동이 될 경우를 분석해 보았습니다.

```
user x의 item score를 입력하세요
 score가 없을 경우 0을 입력하세요.
Item1 :5
Item2 :4
Item3 :0
Item4 :3
Item5 :0
Item6 :2
Item7 :0
Item8 :0
Item9 :3
Item10 :1
```

```
Out[4]: Item1     1.000000
        Item7     0.596762
        Item5     0.550000
        Item9     0.485185
        Item3     0.477410
        Item4     0.470016
        Item2     0.447214
        Item8     0.359573
        Item10    0.358391
        Item6     0.279715
        Name: Item1, dtype: float64
```

  - Item1에 가장 높은 점수를 주었으므로 Item1의 유사도와 가장 높은 Item부터 차례대로 추천을 해 주었습니다.

■ item_based 의 cosine similarity를 이용한 item 추천함수

```
def recommend_item(user):

    largest = int(UserX_lst[0])
    for i in range(0,len(UserX_lst)):
        if int(UserX_lst[i]) > largest:
            largest = int(UserX_lst[i])
    best_rate = UserX_lst.index(str(largest))+1
    rec = item_based_collabor["Item"+str(best_rate)]
    fin_rec = rec.sort_values(ascending=False)
    return fin_rec


recommend_item("UserX")
```

  - def recommend_item(user):

largest = int(UserX_lst[0])

UserX_lst에서 가장 첫번째 값을 largest로 설정합니다.

for i in range(0,len(UserX_lst)):

if int(UserX_lst[i]) > largest:

largest = int(UserX_lst[i])

가장 점수를 높게 준 item을 구합니다.

- best_rate = UserX_lst.index(str(largest))+1

rec = item_based_collabor["Item"+str(best_rate)]

fin_rec = rec.sort_values(ascending=False)

return fin_rec

■ 코드 전체

In [4]:
```
from pandas import Series, DataFrame
import pandas as pd
import numpy as np

f = open(r"C:₩Users₩zdudd₩Desktop₩DA₩20₩week5₩data_week5.txt", encoding='utf8')
lines = f.readlines()

item_frame = DataFrame(columns = ['User1','User2','User3','User4','User5','User6','User7','User8','User9','User10'], index = ['Item1','

for i in range(1, len(lines)):
    item_frame.ix["Item"+lines[i].split(',')[1]]["User"+lines[i].split(',')[0]] = lines[i].split(',')[2].strip()
item_frame.fillna(0, inplace = True)

from sklearn.metrics.pairwise import cosine_similarity
item_based = cosine_similarity(item_frame)
item_based_collabor = pd.DataFrame(data = item_based, index = item_frame.index, columns = item_frame.index)

print("user x의 item score를 입력하세요₩n score가 없을 경우 0을 입력하세요.")
UserX_lst = list()
UserX_lst.append(input("Item1 :"))
UserX_lst.append(input("Item2 :"))
UserX_lst.append(input("Item3 :"))
UserX_lst.append(input("Item4 :"))
UserX_lst.append(input("Item5 :"))
UserX_lst.append(input("Item6 :"))
UserX_lst.append(input("Item7 :"))
UserX_lst.append(input("Item8 :"))
UserX_lst.append(input("Item9 :"))
UserX_lst.append(input("Item10 :"))
```

```
def recommend_item(user):

    largest = int(UserX_lst[0])
    for i in range(0,len(UserX_lst)):
        if int(UserX_lst[i]) > largest:
            largest = int(UserX_lst[i])
    best_rate = UserX_lst.index(str(largest))+1
    rec = item_based_collabor["Item"+str(best_rate)]
    fin_rec = rec.sort_values(ascending=False)
    return fin_rec


recommend_item("UserX")
```

■ 한계점

- User X가 계속하여 새롭게 추가될 경우의 User X의 data는 DataFrame에 저장하지 못하였습니다.
- 가장 높은 점수를 받은 Item과의 cosine 유사도를 통한 추천으로 구현하여 다른 Item들이 받은 점수는 고려하지 못하였습니다.