recommender_user_based_cross_sell

December 13, 2020

[1]:

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

[50]:

```
#o primeiro elemento é o id do usuário, o segundo é a lista de discos comprados␣
 ↪na transação
trans_id=[
          ␣
 ↪(0,[0,1]),(1,[3,4]),(1,[7]),(2,[3,4,6]),(6,[4,5]),(3,[4,5,8]),(4,[9]),(5,[7,8]),(5,[0]),(6,
]
```

[51]:

```
#associação entre ids de bandas e seus nomes
dic={0:'metallica',1:'beatles',2:'stones',3:'black sabbath',4:'rush',5:'van␣
 ↪halen',6:'the who',7:'the police',8:'pink floyd',9:'led zeppelin'}
```

[52]:

```
#Criação da matriz A
A=[[0]*10 for i in range(10)]
A
```

[52]:

```
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
```

[53]:

```
#montagem da matriz A a partir das transações
for i in range(len(trans_id)):
  lin=trans_id[i][0]
  for j in range(len(trans_id[i][1])):
    A[lin][trans_id[i][1][j]]=1
A=np.array(A)
A
```

```
[53]: array([[1, 1, 0, 0, 0, 0, 0, 0, 0, 0],
              [0, 0, 0, 1, 1, 0, 0, 1, 0, 0],
              [0, 0, 0, 1, 1, 0, 1, 0, 0, 0],
              [0, 0, 0, 0, 1, 1, 0, 0, 1, 0],
              [0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
              [1, 0, 0, 0, 0, 0, 0, 1, 1, 0],
              [1, 1, 0, 0, 1, 1, 0, 0, 1, 0],
              [0, 0, 0, 0, 1, 0, 0, 1, 0, 0],
              [1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
              [0, 0, 0, 1, 0, 1, 1, 0, 0, 0]])
```

```
[54]: def sim(v1,v2):
          return np.inner(v1,v2)/(np.linalg.norm(v1)*np.linalg.norm(v2))
```

Gerar a Matriz de Similaridade WC

Obtenha a matriz de recomendações USER BASED

Gerar a função de recomendação

```
[55]: m,n=A.shape
      WC=np.identity(m)
      for i in range(m):
          for j in range(i+1,m):
              WC[i,j]=sim(A[i],A[j])
              WC[j,i]=WC[i,j]
      WC
```

```
[55]: array([[1.        , 0.        , 0.        , 0.        , 0.        ,
               0.40824829, 0.63245553, 0.        , 0.5       , 0.        ],
              [0.        , 1.        , 0.66666667, 0.33333333, 0.        ,
               0.33333333, 0.25819889, 0.81649658, 0.        , 0.33333333],
              [0.        , 0.66666667, 1.        , 0.33333333, 0.        ,
               0.        , 0.25819889, 0.40824829, 0.        , 0.66666667],
              [0.        , 0.33333333, 0.33333333, 1.        , 0.        ,
               0.33333333, 0.77459667, 0.40824829, 0.        , 0.33333333],
              [0.        , 0.        , 0.        , 0.        , 1.        ,
               0.        , 0.        , 0.        , 0.70710678, 0.        ],
              [0.40824829, 0.33333333, 0.        , 0.33333333, 0.        ,
               1.        , 0.51639778, 0.40824829, 0.40824829, 0.        ],
              [0.63245553, 0.25819889, 0.25819889, 0.77459667, 0.        ,
               0.51639778, 1.        , 0.31622777, 0.31622777, 0.25819889],
              [0.        , 0.81649658, 0.40824829, 0.40824829, 0.        ,
               0.40824829, 0.31622777, 1.        , 0.        , 0.        ],
              [0.5       , 0.        , 0.        , 0.        , 0.70710678,
               0.40824829, 0.31622777, 0.        , 1.        , 0.        ],
              [0.        , 0.33333333, 0.66666667, 0.33333333, 0.        ,
               0.        , 0.25819889, 0.        , 0.        , 1.        ]])
```

[56]:

```
#matriz de recomendação é só multiplicar WC por A (user based)
REC=np.dot(WC,A)
REC
```

[56]:

```
array([[2.54070382, 1.63245553, 0.        , 0.        , 0.63245553,
        0.63245553, 0.        , 0.40824829, 1.04070382, 0.5       ],
       [0.59153222, 0.25819889, 0.        , 2.        , 3.07469547,
        0.92486556, 1.        , 2.14982991, 0.92486556, 0.        ],
       [0.25819889, 0.25819889, 0.        , 2.33333333, 2.66644718,
        1.25819889, 1.66666667, 1.07491496, 0.59153222, 0.        ],
       [1.10793   , 0.77459667, 0.        , 1.        , 2.84951163,
        2.10793   , 0.66666667, 1.07491496, 2.10793   , 0.        ],
       [0.70710678, 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 1.70710678],
       [2.33289436, 0.92464607, 0.        , 0.33333333, 1.59131274,
        0.84973111, 0.        , 1.74158162, 1.84973111, 0.40824829],
       [2.46508108, 1.63245553, 0.        , 0.77459667, 2.60722221,
        2.03279556, 0.51639778, 1.09082444, 2.29099445, 0.31622777],
       [0.72447606, 0.31622777, 0.        , 1.22474487, 2.94922093,
        0.72447606, 0.40824829, 2.22474487, 1.13272435, 0.        ],
       [2.22447606, 0.81622777, 0.        , 0.        , 0.31622777,
        0.31622777, 0.        , 0.40824829, 0.72447606, 1.70710678],
       [0.25819889, 0.25819889, 0.        , 2.        , 1.59153222,
        1.59153222, 1.66666667, 0.33333333, 0.59153222, 0.        ]])
```

[57]:

```
def rec_user(id,REC,A,d):
    va=A[id]
    ra=REC[id]
    for i in range(len(va)):
        if va[i]==1:ra[i]=-1
    return d[np.argmax(ra)]
```

[58]:

```
rec_user(0,REC,A,dic)
```

[58]:

```
'pink floyd'
```

[59]:

```
for i in range(m):
    print('user ',i,' recomendação ',rec_user(i,REC,A,dic))
```

```
user  0  recomendação  pink floyd
user  1  recomendação  the who
user  2  recomendação  van halen
user  3  recomendação  metallica
user  4  recomendação  metallica
user  5  recomendação  rush
user  6  recomendação  the police
user  7  recomendação  black sabbath
user  8  recomendação  beatles
user  9  recomendação  rush
```

```
[60]: #instalando apyori
      # !pip install apyori
      from apyori import apriori
```

```
[62]: #parte 1: transforme a lista trans_id em␣
       ↪trans=[['metallica','beatles'],['stones','black sabbath],.................]
      trans = []
      for i in trans_id:
          trans.append(i[1])
      trans
```

```
[62]: [[0, 1],
       [3, 4],
       [7],
       [3, 4, 6],
       [4, 5],
       [4, 5, 8],
       [9],
       [7, 8],
       [0],
       [8, 0, 1],
       [0, 9],
       [3],
       [5, 6],
       [4, 7]]
```

```
[78]: for i in range(len(trans)):
          for j in range(len(trans[i])):
              trans[i][j] = dic[trans[i][j]]
```

```
[80]: trans
```

```
[80]: [['metallica', 'beatles'],
       ['black sabbath', 'rush'],
       ['the police'],
       ['black sabbath', 'rush', 'the who'],
       ['rush', 'van halen'],
       ['rush', 'van halen', 'pink floyd'],
       ['led zeppelin'],
       ['the police', 'pink floyd'],
       ['metallica'],
       ['pink floyd', 'metallica', 'beatles'],
       ['metallica', 'led zeppelin'],
       ['black sabbath'],
 ['van halen', 'the who'],
 ['rush', 'the police']]
```

[82]:

```
#Parte 2)Rode apriori e identifique os 2 conjuntos com maior suporte
resultado = list(apriori(trans))
```

[94]:

```
sup_list = []
set_list = []
resultado[0][1], len(resultado)
for i in range(len(resultado)):
    sup_list.append(resultado[i][1])
    set_list.append(resultado[i][0])
```

[109]:

```
sup_list
```

[109]:

```
[0.14285714285714285,
 0.21428571428571427,
 0.14285714285714285,
 0.2857142857142857,
 0.21428571428571427,
 0.35714285714285715,
 0.21428571428571427,
 0.14285714285714285,
 0.21428571428571427,
 0.14285714285714285,
 0.14285714285714285,
 0.14285714285714285]
```

[107]:

```
set_list[3], sup_list[3]
```

[107]:

```
(frozenset({'metallica'}), 0.2857142857142857)
```

[108]:

```
set_list[5], sup_list[5]
```

[108]:

```
(frozenset({'rush'}), 0.35714285714285715)
```

[138]:

```
#Parte 3: faça sugestões de duplas com os dois itens de maior suporte (quem␣
 ↪comprou a, também levou b)
quant=2
sug = []
for r in resultado:
  if len(r[0])==quant:
    print(r[0])
```

```
frozenset({'metallica', 'beatles'})
frozenset({'black sabbath', 'rush'})
frozenset({'van halen', 'rush'})
```

[ ]:

```
```