CENTRO UNIVERSITÁRIO
UNI METROCAMP

Inteligência Artificial

Prof Eduardo Nunes

Trabalho 1 – Listas de Recomendações

Adaptado de https://iaexpert.academy/

Os usuários de aplicativo de filmes em streaming pontuaram cada filme assistido, segundo a tabela a seguir (variável em Python no final do documento):

	Freddy x Jason	O Ultimato Bourne	Star Trek	Exterminador do Futuro	Norbit	Star Wars
Ana	2.5	3.5	3.0	3.5	2.5	3.0
Marcos	3.0	3.5	1.5	5.0	3.0	3.5
Pedro	2.5	3.0		3.5		4.0
Claudia		3.5	3.0	4.0	2.5	4.5
Adriano	3.0	4.0	2.0	3.0	2.0	3.0
Janaina	3.0	4.0		5.0	3.5	3.0
Leonardo		4.5		4.0	1.0	

Para cada usuário, a plataforma de filmes gostaria de sugerir novos títulos para assistir. Para isso, as notas dos demais usuários podem servir como base para esse sistema de recomendações, se conseguíssemos descobrir “afinidades” entre os usuários.

Na lista de filmes, tem-se ao todo, 6 títulos, e cada título pode ser compreendido com uma dimensão na qual cada usuário pode exprimir seu gosto. Se compararmos cada par de usuários em cada uma das dimensões, podemos inferir o grau de afinidade entre eles.

Matematicamente, iremos utilizar a distância euclidiana para quantificar essa afinidade, tendo em vista que neste conceito é possível incluir, teoricamente, infinitas dimensões(n).

A distância euclidiana entre os pontos $P = (p_1, p_2, \ldots, p_n)$ e $Q = (q_1, q_2, \ldots, q_n)$, num espaço euclidiano n-dimensional, é definida como:

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}.$$

Matematicamente, iremos utilizar a distância euclidiana para quantificar essa afinidade, tendo em vista que neste conceito é possível incluir, teoricamente, infinitas dimensões(n).

Em linguagem Python, podemos **calcular a distância euclidiana** da seguinte forma:

```
def euclidiana(base, usuario1, usuario2):
    si = {}
    for item in base[usuario1]:
        if item in base[usuario2]: si[item] = 1

    if len(si) == 0: return 0
    soma = sum([pow(base[usuario1][item] - base[usuario2][item], 2)
                for item in base[usuario1] if item in base[usuario2]])

    return 1/(1 + sqrt(soma))
```

Percorre-se a base da dados para verificar filmes que possuem avaliações dos 2 usuários!

Se não tem avaliações em comum, a distância é zero!!

Se o filme está na base de avaliações dos 2 usuários, calcula-se a soma dos quadrados da subtração.

Por fim, extrai-se a raiz e faz a ponderação entre 0 e 1. Zero para nenhuma correlação e 1 para correlação máxima

Uma vez definida a função “euclidiana”, podemos utiliza-la para **descobrir** quem tem **gostos similares** a cada um dos usuários.

```
def getSimilares(base, usuario):
    similaridade = [(euclidiana(base, usuario, outro), outro)
                    for outro in base if outro != usuario]
    similaridade.sort()
    similaridade.reverse()
    return similaridade[0:30]
```

Com as similaridades definidas entre os usuários, pode-se realizar **recomendações para cada usuário**:

```
def getRecomendacoesUsuario(base, usuario):
    totais={}
    somaSimilaridade={}
    for outro in base:
        if outro == usuario: continue
        similaridade = euclidiana(base, usuario, outro)
        if similaridade <= 0: continue
        for item in base[outro]:
            if item not in base[usuario]:
                totais.setdefault(item, 0)
                totais[item] += base[outro][item] * similaridade
                somaSimilaridade.setdefault(item, 0)
                somaSimilaridade[item] += similaridade
    rankings=[(total / somaSimilaridade[item], item) for item, total in totais.items()]
    rankings.sort()
    rankings.reverse()
    return rankings[0:30]
```

Se utilizarmos uma base de dados (avaliacoesFilme) com os filmes como chave primária e não os usuários, podemos encontrar itens similares.

```
def calculaItensSimilares(base):
    result = {}
    for item in base:
        notas = getSimilares(base, item)
        result[item] = notas
    return result
```

A partir desta similaridade de itens (Filmes), podemos elaborar **recomendações de itens**.

```
def getRecomendacoesItens(baseUsuario, similaridadeItens, usuario):
    notasUsuario = baseUsuario[usuario]
    notas={}
    totalSimilaridade={}
    for (item, nota) in notasUsuario.items():
        for (similaridade, item2) in similaridadeItens[item]:
            if item2 in notasUsuario: continue
            notas.setdefault(item2, 0)
            notas[item2] += similaridade * nota
            totalSimilaridade.setdefault(item2,0)
            totalSimilaridade[item2] += similaridade
    rankings=[(score/totalSimilaridade[item], item) for item, score in notas.items()]
    rankings.sort()
    rankings.reverse()
    return rankings
```

CENTRO UNIVERSITÁRIO
UNI METROCAMP

Como base nos algoritmos em Python, elabore respostas para o seguintes itens a seguir. As respostas podem ser um print da IDE:

1. Calcular a distância euclidiana entre os seguintes usuários (1 ponto):
    a. Ana todos os demais.
    b. Leonardo e todos os demais.
    c. Entre Ana e ela mesma. Comente o valor obtido.

2. Elaborar a lista de usuários similares para cada um dos usuários (1 ponto).

3. Elaborar a lista de recomendações para cada usuário (2 pontos).

4. Elaborar a lista de filmes similares (2 pontos).

5. A lista de recomendações implementada até então, não leva e consideração a quantidade de vezes que aquele filme foi assistido para considerar suas notas atribuídas mais ou menos relevante.

    a. Implemente uma melhoria neste algoritmo para levar em conta também a quantidade de notas que cada filme recebeu. Gere uma nova lista de recomendações (2 pontos).
    b. Implemente uma melhoria para as notas de usuários mais assíduos terem mais relevâncias que os menos assíduos. Gere uma nova lista de recomendações (2 pontos).

```
# base de dados de avaliacoes de cada usuario
avaliacoesUsuario = {'Ana':
                        {'Freddy x Jason': 2.5,
                         'O Ultimato Bourne': 3.5,
                         'Star Trek': 3.0,
                         'Exterminador do Futuro': 3.5,
                         'Norbit': 2.5,
                         'Star Wars': 3.0},

                'Marcos':
                        {'Freddy x Jason': 3.0,
                         'O Ultimato Bourne': 3.5,
                         'Star Trek': 1.5,
                         'Exterminador do Futuro': 5.0,
                         'Star Wars': 3.0,
                         'Norbit': 3.5},

                'Pedro':

                 {'Freddy x Jason': 2.5,

                         'O Ultimato Bourne': 3.0,

                         'Exterminador do Futuro': 3.5,

                         'Star Wars': 4.0},

                'Claudia':
                        {'O Ultimato Bourne': 3.5,
                         'Star Trek': 3.0,
                         'Star Wars': 4.5,
                         'Exterminador do Futuro': 4.0,
                         'Norbit': 2.5},

                'Adriano':
                        {'Freddy x Jason': 3.0,
                         'O Ultimato Bourne': 4.0,
                         'Star Trek': 2.0,
                         'Exterminador do Futuro': 3.0,
                         'Star Wars': 3.0,
                         'Norbit': 2.0},

                'Janaina':
                  {'Freddy x Jason': 3.0,
                   'O Ultimato Bourne': 4.0,
                   'Star Wars': 3.0,
                   'Exterminador do Futuro': 5.0,
                   'Norbit': 3.5},

                'Leonardo':
                 {'O Ultimato Bourne':4.5,
                  'Norbit':1.0,
                   'Exterminador do Futuro':4.0}
}
```

```
# base de dados de avaliações para cada filme
avaliacoesFilme = {'Freddy x Jason':
                        {'Ana': 2.5,
                         'Marcos:': 3.0 ,
                         'Pedro': 2.5,
                         'Adriano': 3.0,
                         'Janaina': 3.0 },

                'O Ultimato Bourne':
                        {'Ana': 3.5,
                         'Marcos': 3.5,
                         'Pedro': 3.0,
                         'Claudia': 3.5,
                         'Adriano': 4.0,
                         'Janaina': 4.0,
                         'Leonardo': 4.5 },

                'Star Trek':
                        {'Ana': 3.0,
                         'Marcos:': 1.5,
                         'Claudia': 3.0,
                         'Adriano': 2.0 },

                'Exterminador do Futuro':
                        {'Ana': 3.5,
                         'Marcos:': 5.0 ,
                         'Pedro': 3.5,
                         'Claudia': 4.0,
                         'Adriano': 3.0,
                         'Janaina': 5.0,
                         'Leonardo': 4.0},

                'Norbit':
                        {'Ana': 2.5,
                         'Marcos:': 3.0 ,
                         'Claudia': 2.5,
                         'Adriano': 2.0,
                         'Janaina': 3.5,
                         'Leonardo': 1.0},

                'Star Wars':
                        {'Ana': 3.0,
                         'Marcos:': 3.5,
                         'Pedro': 4.0,
                         'Claudia': 4.5,
                         'Adriano': 3.0,
                         'Janaina': 3.0}
}
```