

Palavras-chave: Similaridade de conjuntos, distância de Jaccard, índice de Jaccard, minhash.

O presente trabalho prático tem por objectivo criar um “módulo” que suporte a descoberta de conjuntos similares (ex: [1]) e testar esse módulo.

Neste trabalho iremos utilizar o ficheiro `u.data` do conjunto de dados (release 4/1998) MovieLens 100k, disponível em <http://grouplens.org/datasets/movielens/>. Este ficheiro contém informação sobre 943 utilizadores e 1682 filmes. Tem cerca de 100 000 linhas, como as seguintes:

```
196 242 3 881250949
186 302 3 891717742
22 377 1 878887116
```

As colunas são separadas por *tabs*; a primeira coluna contém o ID do utilizador; a segunda contém o ID de um filme (avaliado pelo utilizador mencionado na primeira coluna); a terceira é a avaliação; a quarta um *timestamp*.

O nosso objectivos é descobrir utilizadores que avaliaram conjuntos similares de filmes. Para este objectivo as colunas 3 e 4 não são necessárias.

- 1) Analise o código Matlab disponibilizado conjuntamente com este guião e complete-o por forma a conseguir calcular a **distância de Jaccard** entre os conjuntos de filmes avaliados pelos vários utilizadores.

Inclua no código a possibilidade de calcular o tempo que demora cada uma das partes (cálculo da distância e determinação das distâncias abaixo de um determinado limiar). Veja a informação relativa `tic`, `toc`, `cputime`, `etime`.

No final, o programa deve mostrar informação com: (1) número de pares de utilizadores com distâncias inferiores ao limiar definido; (2) informação sobre cada par (utilizadores e distância).

Adicione, também, a capacidade de gravar em ficheiro a matriz de distâncias calculada. Sugere-se que consulte a informação de `save`.

- 2) Com base no código que adaptou, crie funções para:

- Criar a estrutura de dados com os conjuntos de filmes;
- Calcular as distâncias ;
- Processar as distâncias e devolver os itens similares. Esta função deve ter como um dos parâmetros o limiar de decisão.

Teste o código com 100 utilizadores seleccionados de forma aleatória.

Depois do teste anterior com um número reduzido de utilizadores e eventual resolução de problemas detectados, execute o seu programa com todo o conjunto de dados por forma a determinar todos os pares de utilizadores com uma distância de Jaccard inferior a 0.4

Tome nota dos tempos e dos resultados obtidos.

- 3) Crie uma nova versão da função de cálculo de distância recorrendo a uma aproximação probabilística usando minhash. Comece por testar esta nova implementação com um número pequeno de utilizadores e depois teste-a com o conjunto total de utilizadores.

Compare os pares considerados como similares com os obtidos com a implementação não probabilística. Comente.

- 4) Adapte o seu código para utilizar outro(s) conjunto(s) de dados (datasets) do Movilens e utilize as funções desenvolvidas anteriormente para detetar os pares de utilizadores similares.

Referências

[1] Jure Leskovec, Anand Rajaraman, and Jeff Ullman. *Mining of Massive Datasets*, chapter Finding Similar Items. Cambridge University Press, 2014.

% Código base para guião PL07 MPEI 2018-2019

```
udata=load('u.data'); % Carrega o ficheiro dos dados dos filmes
% Fica apenas com as duas primeiras colunas
u= udata(1:end,1:2); clear udata;
```

```
% Lista de utilizadores
users = unique(u(:,1)); % Extrai os IDs dos utilizadores
Nu= length(users); % Número de utilizadores
```

```
% Constrói a lista de filmes para cada utilizador
Set= cell(Nu,1); % Usa células
```

```
for n = 1:Nu, % Para cada utilizador
    % Obtém os filmes de cada um
    ind = find(u(:,1) == users(n));
    % E guarda num array. Usa células porque utilizador tem um número
    % diferente de filmes. Se fossem iguais podia ser um array
    Set{n} = [Set{n} u(ind,2)];
end
```

```
%% Calcula a distância de Jaccard entre todos os pares pela definição.
```

```
J=zeros(...); % array para guardar distâncias
h= waitbar(0,'Calculating');
for n1= 1:Nu,
    waitbar(n1/Nu,h);
    for n2= n1+1:Nu,
        %% Adicionar código aqui
    end
end
delete (h)
```

```
%% Com base na distância, determina pares com
%% distância inferior a um limiar pré-definido
```

```
threshold =0.4 % limiar de decisão
% Array para guardar pares similares (user1, user2, distância)
SimilarUsers= zeros(1,3);
k= 1;
for n1= 1:Nu,
    for n2= n1+1:Nu,
        if % .....
            SimilarUsers(k,:)= [users(n1) users(n2) J(n1,n2)]
            k= k+1;
        end
    end
end
```

©AT+CB+AS, 2017, 2018