MPEI 2024/25 - PL 7

Algoritmos Probabilísticos 2 - MinHash

Palavras chave: Similaridade, distância de Jaccard, MinHash.

Nos exercícios seguintes explora-se MinHash na descoberta de conjuntos similares. Como base para os exercícios, iremos utilizar o ficheiro `u.data` do conjunto de dados MovieLens 100k (release 4/1998), disponível em `http://grouplens.org/datasets/movielens/`. O MovieLens é um conjunto de dados contendo informações sobre utilizadores, filmes e classificações, permitindo estudos e experiências em aprendizagem automática. O ficheiro `u.data` 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 dada pelo utilizador ao filme e a quarta coluna é um *timestamp* do momento da avaliação.

7.1 Utilizadores que avaliaram conjuntos similares de filmes

O objectivo é descobrir utilizadores que avaliaram conjuntos similares de filmes, sendo desnecessárias, para este fim, as colunas 3 e 4.

1. Analise o código Matlab disponibilizado no final desta secção e complete-o por forma a conseguir calcular a **distância de Jaccard** entre os conjuntos de filmes avaliados pelos vários utilizadores.

   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 (identificação dos utilizadores e distância de Jaccard).

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

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

   (a) criar a estrutura de dados com os conjuntos de filmes;

   (b) calcular as distâncias entre conjuntos;

   (c) processar as distâncias e devolver os pares de conjuntos similares. Um dos parâmetros de entrada desta função deve ser o limiar de decisão.

   Faça um primeiro teste ao código considerando 100 utilizadores seleccionados de forma aleatória.

3. Depois do teste anterior com um número reduzido de utilizadores e da resolução de eventuais 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.

4. Crie uma nova versão da função de cálculo de distância recorrendo a uma aproximação probabilística usando *MinHash*. Um dos parâmetros da função deve ser o número de funções de dispersão (k).

   Teste com o conjunto total de utilizadores considerando 100 funções de dispersão ($k = 100$) na implementação do *MinHash*.

5. Teste com $k = 50$ e $k = 200$. Compare os pares considerados como similares (e o seu valor de similaridade) para cada valor de k com os obtidos com a implementação não probabilística e retire conclusões.

7.2 Filmes com títulos similares

Considere agora o conjunto de dados no ficheiro `film_info.txt` [1] [2] com o seguinte conteúdo:

Toy Story (1995)	Adventure	Animation	Thriller
GoldenEye (1995)	Action	Adventure	Thriller
Four Rooms (1995)	Thriller		
Get Shorty (1995)	Action	Comedy	Drama

em que os dados de cada coluna estão separados por `tabs`. A linha número n contém a informação do filme com o ID n usado na segunda coluna do ficheiro `u.data`. A primeira coluna contém o nome do filme e respetivo ano de estreia. As restantes colunas contêm um número variável de géneros cinematográficos associados ao filme.

1. Desenvolva um método *MinHash* adequado a estimar a similaridade entre vetores de caracteres, escolhendo um número de funções de dispersão k e tamanho dos *shingles* adequados [3].

   Como resultado deve obter a matriz com as assinaturas dos nomes de todos os filmes, que deve ser guardada num ficheiro `.mat`.

2. Desenvolva uma função que devolva o nome dos 3 filmes com os títulos mais similares a uma string introduzida pelo utilizador (com o nome ou parte do nome de um filme).

7.3 Utilizadores com interesses similares

Para além da informação das colunas 1 e 2 utilizadas no exercício 7.1, consideremos agora informação sobre cada um dos utilizadores, disponibilizada no ficheiro, `users.txt` [4] [5] , com o seguinte conteúdo:

```
4 ; Carol ; Jesus ; Música ; Fotografia ; Filmes ; Jogos ; Leitura ; ...
49 ; Naísa ; Rodrigues ; Fotografia ; Viagens ; Futebol ;  ...
```

em que os dados de cada coluna estão separados por ";". A linha número n contém a informação do utilizador com o ID n usado no ficheiro `u.data`. A primeira coluna contém o número, a segunda o nome (próprio) e a terceira o apelido. As restantes colunas contêm um número variável de interesses do utilizador, como, por exemplo, "Jogos".

[1] O ficheiro `film_info.txt` encontra-se disponível para download em `https://bit.ly/4fKBBt3`

[2] Executando no Matlab a instrução: `dic2= readcell('film_info.txt', 'Delimiter','\t');` é criado o cell array `dic2` em que a célula `dic2{i,j}` contém a informação da linha i e da coluna j do ficheiro `film_info.txt`

[3] Sugere-se que experimente tamanhos de *shingle* entre 2 e 5 caracteres.

[4] O ficheiro `users.txt` encontra-se disponível para download em `https://bit.ly/3Z1KeZ8`

[5] Executando no Matlab a instrução: `dic= readcell('users.txt', 'Delimiter',';');` é criado o cell array `dic` em que a célula `dic{i,j}` contém a informação da linha i e da coluna j do ficheiro.

Desenvolva uma função em Matlab que para um dos filmes, definido pelo seu ID, de 1 a 1682, **determine utilizadores com interesses similares**. A função deve:

1. Para cada utilizador que já avaliou o filme escolhido, selecionar os utilizadores cuja distância de Jaccard estimada (em termos de interesses) seja menor que 0.9;

2. Filtrar os resultados por forma a reter apenas os utilizaores que ainda não tenham avaliado o filme passado como parâmetro. Isto resulta num conjunto de potenciais utilizadores por cada avaliador do filme;

3. No final apresentar os IDs e os nomes dos dois utilizadores que aparecem no maior número de conjuntos.

Deve desenvolver um método *MinHash* adequado à similaridade entre conjuntos de vectores de caracteres (interesses dos utilizadores).

Sugestão: Deve pré-calcular a matriz de asssinaturas com os vectores *MinHash* correspondentes ao conjunto de interesses de cada utilizador para tornar mais rápido o processo de determinação dos utilizadores com interesses similares.

Anexo

```
% Código base para deteção de pares similares

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
```