deti universidade de aveiro
departamento de eletrónica,
telecomunicações e informática

Relatório

Guião PL04

Métodos Probabilísticos para Engenharia Informática

Departamento de Eletrónica, Telecomunicações e Informática

Prof. Amaro Sousa

Ano letivo 2022/2023

Turma P6

Isaac Moura, 105065

Miguel Gomes, 103826

06/01/2023

Índice

Introdução

A aplicação tem duas componentes, cada uma num script de MatLab. Uma componente é a estrutura de dados da aplicação, ou seja, vai conter as estruturas e funções usadas para manipular os dados dos utilizadores. O código presente neste script só necessita de ser executado uma vez para gerar todos os dados necessários sendo apenas preciso repetir o processo se quisermos inserir novos dados, como filmes ou utilizadores. O código armazena dados como nomes de filmes, nomes de utilizadores, função MinHash para cada filme e respetivo shingle. O outro script é responsável por executar a aplicação carregando os dados do outro script.

Estrutura de Dados

```
ufile = load('u.data');
ufilms = ufile(1:end,1:3);
users = unique(ufile(:,1));
dicFilms = readcell('films.txt', 'Delimiter', '\t');

Nu = length(users);
YourMoviesTable = cell(Nu, 1);

for i = 1:Nu
    x = find(ufilms(:,1) == users(i));
    YourMoviesTable{i} = [YourMoviesTable{i} ufilms(x,2)];
end



K = 100;
MinHashValue = inf(Nu,K);

for i = 1:Nu
    conjunto = YourMoviesTable{i};
    for j = 1:length(conjunto)
        chave = char(conjunto(j));
        hash = zeros(1,K);
        for kk = 1:K
            chave = [chave num2str(kk)];
            hash(kk) = DJB31MA(chave,127);
        end
        MinHashValue(i,:) = min([MinHashValue(i,:); hash]);  % Valor minimo
da hash para este título
    end
end

k= 100;
MinHashCat = inf(length(dicFilms),k);

Nc = length(dicFilms);

allcategories = strings(1,19);
Nallc = 19;
it = 1;
for i = 1:length(dicFilms)
    categorias = rmmissing(string(dicFilms(i,2:end)));
    for x = 1: length(categorias)
        categoria = categorias(x);
        if ~ismember(categoria,allcategories)
            allcategories(it) = categoria;
            it = it +1;
        end
    end
end

MinHashallCat = inf(length(allcategories),k);

for i = 1:Nallc
    conjunto = allcategories(i);
    for j = 1:length(conjunto)
        chave = char(conjunto(j));
        hash = zeros(1,k);
        for kk = 1:k
```

Na determinação do valor de k da MinHash dos filmes utilizamos as conclusões obtidas no exercício 4 da parte 4.3 do guião prático. Foi medido o tempo de calculo da MinHash para certos valores de k assim como as distâncias de Jaccard para os utilizadores que deram valores similares. Usando estes dados em conjunto com o real valor da distância de Jaccard chegamos a conclusão de que 100 seria um bom número para k pois junta a precisão dos valores obtidos com um tempo de execução razoável.

Quanto ao tamanho dos shingles o enunciado recomenda um valor entre 2 e 5 e , após alguns testes, a diferença do tempo de execução não foi muito relevante pelo que optamos por shingles com tamanho 3.

Aumentando os valores de k na MinHash conseguimos sempre aumentar a precisão do valor das distâncias de Jaccard. Aumentar estes valores tem como consequência o aumento do tempo de execução do código pelo que é necessário encontrar um ponto de equilíbrio conforme cada caso de MinHash.

Na decisão do valor de k para a MinHash dos shingles foi escolhido um valor intermédio que fornece uma precisão aceitável para a funcionalidade tendo em conta o tempo de execução. Não é benéfico ter um nível muito elevado de precisão se a aplicação demora um tempo descabido a correr.

Aplicação

```
load script1;


user = 0;
option = 0;

menu(user, option,MinHashSig, MinHashValue, Nu, dicFilms,
YourMoviesTable,MinHashallCat,MinHashCat);
```

Como temos os dados pré carregados basta fazer load. A função menu é chamada uma vez que serve de base a todo o programa.

Menu

```
function menu(user,option, MinHashSig, MinHashValue, Nu, dicFilms,
YourMoviesTable,MinHashallCat,MinHashCat)
    while(option ~=5)
        clc
        if(user == 0)
            user = str2num(input(['Insert User ID (1 to ' num2str(Nu) '): '],
's'));
        elseif (user < 1 || user > Nu)
            fprintf('User Id not valid. ');
            clc;
            user = 0;
        else

            while(option <5)
                fprintf('\nUser ID: %d Menu:', user)
                fprintf('\n1 - Your movies\n2 - Suggestion of movies based on
other users\n3 - Suggestion of movies based on already evaluated movies\n4 -
Movies feedback based on popularity\n5 - Exit\n')

                option = str2num(input('','s'));

                if isempty(option)
                    continue;
                end

                switch option
                    case 1
                        yourMovies(user, YourMoviesTable, dicFilms)
                    case 2

SuggestionsOtherUsers(Nu,MinHashValue,user,YourMoviesTable,dicFilms)
                    case 3

SuggestionsCategories(Nu,MinHashValue,user,YourMoviesTable,dicFilms,MinHashal
lCat,MinHashCat)
                    case 4
                        searchTitle(uFilms, MinHashSig)
                    case 5

                        break;


                    otherwise
                        fprintf('Choose a number more than 1 and less than
5');
                end
            end
        end
    end
    end
```

Temos uma função menu que vai fornecer ao utilizador todas as operações que pode efetuar na aplicação. Depois de efetuar o "login", ou seja, de inserir o seu numero(id), o programa vai listar as operações disponíveis. As opções vão do 1 ao 5 e utilizam como dados os conjuntos da estrutura de dados como a MinHash, lista de filmes, lista de utilizadores e lista de filmes vistos e avaliados.

Tivemos em atenção possíveis inputs errados por parte do utilizador e por isso a aplicação ignora-os e volta a pedir o número de utilizador quando este é inválido.

Função yourMovies

```
function yourMovies(user,YourMoviesTable, dicFilms)
    fprintf('\nYour Movies:');
    keyMovie = YourMoviesTable{user};
    for i = 1:length(keyMovie)
        linha = keyMovie(i);
        fprintf(char(dicFilms(linha,1)))
        fprintf('\n')
    end
    pause; clc;
end
```

Esta função lista todos os filmes que o utilizador que está a usar o sistema já viu. Tem como parâmetros o Id do utilizador, a lista de todos os filmes vistos por cada utilizador (YourMoviesTable) e ainda a lista que tem todos os filmes e respetivas categorias(uFilms).

Função SuggestionsOtherUsers

```
function SuggestionsOtherUsers(Nu,MinHashValue,user,YourMoviesTable,dicFilms)
    k = 100;
    Jdistances = ones(1,Nu);

    for n = 1:Nu
        if n~= user
            Jdistances(n) = sum(MinHashValue(n,:) ~= MinHashValue(user,:))/k;
% Distancia je Jaccard para todos os pares possiveis do
            %respetivo user
        end
    end

    [val,SimilarId] = min(Jdistances);
    Jdistances(:,SimilarId) = [];
    [val,SimilarId2] = min(Jdistances);

    suggestions = [];
    for n = 1: length(YourMoviesTable{SimilarId})
        if(~ismember(YourMoviesTable{SimilarId}(n), YourMoviesTable{user}))
            suggestions = [suggestions string(dicFilms(n,1))];
        end
    end

    if isempty(suggestions)
        fprintf('Don t exists any suggestion')
    else
        fprintf('Film Suggestions of the user most similar:\n')
        for i = 1:length(suggestions)
            fprintf(suggestions(i) + '\n');
        end
    end

    suggestions = [];
    for n = 1: length(YourMoviesTable{SimilarId2})
        if(~ismember(YourMoviesTable{SimilarId2}(n), YourMoviesTable{user}))
            suggestions = [suggestions string(dicFilms(n,1))];
        end
    end

    if isempty(suggestions)
        fprintf('Don t exists any suggestion')
    else
        fprintf('Film Suggestions of the seconds user most similar:\n:\n')
        for i = 1:length(suggestions)
            fprintf(suggestions(i) + '\n');
        end
    end
    fprintf('\nPress enter to continue');
    pause; clc;
end
```

Esta função sugere títulos de filmes que ainda não foram avaliados pelo utilizador atual, provenientes dos dois utilizadores que são mais similares a ele.

Para encontrar os utilizadores semelhantes é usado a MinHashValue com os valores previamente calculados. Usando estes valores é possível calcular a distância de Jaccard. Foram mantidos os valores de K usados na estrutura de dados. A função ismember é responsável por filtrar aqueles filmes recomendados que o nosso utilizador já avaliou.

Função SuggestionsCategories

```
SuggestionsCategories(Nu,MinHashValue,user,YourMoviesTable,dicFilms,MinHashal
lCat,MinHashCat)

    allmovies = YourMoviesTable{user};
    %usermov = cell2mat(allmovies);
    Nusermovies = length(allmovies);

    Nc = length(dicFilms);
    J=zeros(Nc);
    k=100;

    h=waitbar(0, 'Calculating');
    for i = 1:Nusermovies
        waitbar(i/Nusermovies, h);
        n1 = allmovies(i);
        for n2 = 1:Nc
            if n2 ~= n1 && ~ismember(n2,allmovies)
                J(n1,n2) = sum(MinHashCat(n1,:) ~= MinHashCat(n2,:))/k;
            end
        end
    end

    films = dicFilms(:,1);
    threshold = 0.8;
    SimilarMovies = cell(1,3);
    k = 1;

    h=waitbar(0, 'Calculating');
    for i = 1:Nusermovies
        waitbar(i/Nusermovies, h);
        n1 = allmovies(i);
        for n2 = 1:Nc
            if J(n1,n2) < threshold && ~ismember(n2,allmovies)
                SimilarMovies(k,:) = {n1 n2 J(n1,n2)};
                k = k+1;
            end
        end
    end


    allsimilarMovies = cell2mat(SimilarMovies(:,2));
    similar2Movies = mode(allsimilarMovies,'all');

    ismember(similar2Movies, allmovies)

    fprintf('\nTwo movies most common to you: ');
    fprintf('%s, ',char(films(similar2Movies)));
    x = find(allsimilarMovies == similar2Movies);

    allsimilarMovies(x) = [];

    similar2Movies = mode(allsimilarMovies,'all');
    fprintf(char(films(similar2Movies)));

    fprintf('\nPress enter to continue');
    pause; clc;
    end
```

A função SuggestionsCategories sugere filmes baseados nos filmes que o utilizador já avaliou. O critério de escolha está na distância de Jaccard, neste caso é 0.8. São apresentados os dois filmes que mais se aproximam ao conjunto de filmes avaliados pelo utilizador.

Função searchTitle

```
function searchTitle(dicFilms, MinHashSig)
    str = lower(input('\nWrite a String: ', 's'));
    shingle_size = 3;  % igual numero de shingles
    K = size(MinHashSig, 2);  % manter K
    threshold = 0.99;  % Definir um threshold que nos é dado

    % Estrutura de cell array para os shingles
    shinglesAns = {};
    for i = 1:length(str) - shingle_size+1
        shingle = str(i:i+shingle_size-1);
        shinglesAns{i} = shingle;
    end

    % MinHash para os shingles
    MinHashString = inf(1,K);
    for j = 1:length(shinglesAns)
        chave = char(shinglesAns{j});
        hash = zeros(1,K);
        for kk = 1:K
            chave = [chave num2str(kk)];
            hash(kk) = DJB31MA(chave, 127);
        end
        MinHashString(1,:) = min([MinHashString(1,:); hash]);
    end

    % Distância de Jaccard
    distJ = ones(1, size(dicFilms,1));  % guarda distâncias
    h = waitbar(0,'Calculating');
    for i=1:size(dicFilms, 1)  % cada hashcode da string
        waitbar(i/K, h);
        distJ(i) = sum(MinHashSig(i,:) ~= MinHashString)/K;
    end
    delete(h);

    flag = false;  % flag para ver se foi encontrado filme
    for i = 1:5
        [val, pos] = min(distJ);  % Calcular o valor mais proximo(minimo)
        if (val <= threshold)  % Se o valor nao pertence, exclui
            flag = true;
            fprintf('%s\n', dicFilms{pos, 1});
        end
        distJ(pos) = 1;  % Retirar esse filme dando uma distancia igual a 1
    end

    if (~flag)
        fprintf('No movies found.\n');
    end
    fprintf(2, 'Press any key to continue. ');
    pause;clc;  % Manter info
end
```

A função searchTitle tem como objetivo fornecer uma lista de filmes baseada numa string inserida pelo utilizador. Esta lista contém os 5 filmes cujo título é mais parecido com a string inserida. Usamos uma estrutura semelhante da função SuggestionsCategories para criar os shingles e formação da MinHash para cada um. As distâncias de Jaccard entre os shingles da string introduzida e dos filmes foram comparadas para encontrar um valor mínimo.
Caso a nossa string não esteja dentro do limite de pesquisa o programa responde que não foi possível encontrar uma correspondência.
A lista de filmes aparece ordenada pelas strings que mais são parecidas com a string inserida pelo utilizador.

Anotações

A função de hash usada na elaboração deste trabalho foi a "DJB31MA.m" que foi fornecida no guião prático 4. O valor de k da função foi sendo alterado conforme as necessidades o que permitiu obter várias funções hash dentro de uma só, variando um parâmetro.

```
function h= DJB31MA( chave, seed)
% implementação da hash function DJB31MA com base no algoritmo obtido
% no resumo 2014(PJF) que está em C
%
%  chave    array de caracteres com a chave
%  seed     semente que permite obter vários hash codes para a mesma chave
%
%  h        hashcode devolvido
len= length(chave);
chave= double(chave);
h= seed;
for i=1:len
    h = mod(31 * h + chave(i), 2^32 -1) ;
end
```