



## **Relatório**

### **Guião PL04-Teste2**

Métodos Probabilísticos para Engenharia Informática

Departamento de Eletrónica, Telecomunicações e Informática

Ano letivo 2022/2023

Turma P7

Adalberto Júnior da Trindade Vaz do Rosário, 105589

08/01/2023



## Conteúdo

Introdução .....	3
ProcessData .....	4
<b>Main</b> .....	8
Menu .....	8
A função listName.....	9
Função listSuggestion .....	9
Jacard.....	10
Similaridade .....	10
ListSuggestInterests .....	11
feedback .....	12
Nota Final.....	14



## Introdução

Como um dos requisitos obrigatório, Comecei por fazer 2 scripts para o funcionamento desta aplicação, a um deles chamei de “processData.m” onde tenho código que apenas é preciso ser executado periodicamente caso haja uma mudança nos ficheiros dado. Este código armazena todos os dados que achei relevantes, para além dos que são pedidos nos requisitos, para a execução da aplicação, e tem alguns que não cheguei de usar. Ao outro script chamei de “main.m” que vai correr a app em si e dá *load* às variáveis que foram criadas no “processData.m”

Considere que:

- O ficheiro “u.data”, “users.txt” e o “film\_info.txt” estão no mesmo diretório dos scripts para o guião;
- Algum código é completamente reutilizado, logo código usado mais do que uma vez não irá ser comentado novamente;



## ProcessData

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Ler os ficheiros
%% u.data:
udata=load('u.data'); % Carrega o ficheiro dos dados dos filmes
%% users.txt
dic = readcell('users.txt', 'Delimiter', ';'); %informações tiradas do
users.txt
%% filme_info.txt
dic2 = readcell('filme_info.txt', 'Delimiter', '\t'); %informações tiradas do
filme_info.txt
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% processar os dados
%%udata

u= udata(1:end,1:3); clear udata;
users = unique(u(:,1));
Nu= length(users);

set = cell(Nu,1);
idFil_Aval = cell(Nu,1); % conjunto para guardar id de filmes e avaliações
de cada user

for n = 1:Nu % Para cada utilizador

    ind = u(:,1) == users(n);
    set{n} = [set{n} u(ind,2)];
    idFil_Aval{n} = [idFil_Aval{n} u(ind,2) u(ind,3)];
end

%% dic

interesses = {}; % conjunto de interesses de cada users
Id_Name = {}; %Conjunto com id e nome dos users
for j = 1:Nu
    for k=4:length(dic(j,:))
        interesses{j,k-3} = dic{j,k};
    end
    for i = 1:2
        Id_Name{j,i} = dic{j,i};
    end
end
```



```
end
end

titulos = {}; % conjunto de titulo dos filmes
for j = 1:length(unique(u(:,2)))
    titulos{j,1} = lower(dic2{j,1});
end

id_Fil = unique(u(:,2));
id_name_fil = cell(size(titulos));
for n = 1:id_Fil % Para cada utilizador

    id_name_fil{n} = [id_name_fil{n} {id_Fil(n) titulos{n}}];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% MinHash:
kf = 100;
MinHasUseAv = inf(Nu,kf);
for n1 = 1:Nu
    conjunto = set{n1};
    for i = 1:length(conjunto)
        hash = zeros(1,kf) ;
        chv = char(conjunto(i));
        for hf = 1:kf
            chv = [chv num2str(hf)];
            hash(hf) = DJB31MA(chv,hf);
        end
        MinHasUseAv(n1,:) = min([MinHasUseAv(n1,:); hash]);
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Minhash conjunto de interesses

shingSiz=3;
kf = 150; % Número de funções de dispersão
MinHasUseIntrs = inf(length(dic),kf);
siz = size(interesses);
for n1 = 1:length(dic)
    for x = 1: siz(2)
        conjunto = interesses{n1,x};
        if ~(ismissing(conjunto))
            shingles = {};
            for j= 1 : length(conjunto) - shingSiz+1 % Criando shingles para
cada filme
                shingle = conjunto(j:j+shingSiz-1);
                shingles{j} = shingle;
            end

            for j = 1:length(shingles)
                chav = char(shingles(j));
                hash = zeros(1,kf);
                for hf = 1:kf
                    chav = [chav num2str(hf)];
                    hash(hf) = DJB31MA(chav,hf);
                end
                MinHasUseIntrs(n1,:) = min([MinHasUseIntrs(n1,:);hash]);
            end
        end
    end
end
```



```
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Minhash conjunto de titulos:
%shingSiz=3;
kf = 150; % Número de funções de dispersão
MinHasFilmTitul = inf(length(dic2),kf);
for i = 1:length(dic2)
    conjunto = titulos{n1};
    shingles = {};
    for j= 1 : length(conjunto) - shingSiz+1 % Criando shingles para cada
filme
        shingle = conjunto(j:j+shingSiz-1);
        shingles{j} = shingle;
    end

    for j = 1:length(shingles)
        chav = char(shingles(j));
        hash = zeros(1,kf);
        for hf = 1:kf
            chav = [chav num2str(hf)];
            hash(hf) = DJB31MA(chav,hf);
        end
        MinHasFilmTitul(i,:) = min([MinHasFilmTitul(i,:);hash]);
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Countinh Bloom Filter

%Inicializar o Counter Bloom Filter:
n = 10000;
m = 1682; %numero de filmes
khf = uint8((n * 0.693)/m); % k hash function

B = zeros(1,n,"uint8"); % Inicializar o Bloom construir o countinh
for n1 = 1:Nu
    conjunto = idFil_Aval{n1};
    for i = 1:length(conjunto)
        if conjunto(i,2) >= 3
            Filtro = add(B,titulos{i},khf);
            B = Filtro;
        end
    end
end

save ('dados', 'dic', 'dic2', 'Id_Name', 'interesses', 'idFil_Aval',
'titulos', 'users', 'set', 'MinHasUseIntrs', 'MinHasUseAv',
'MinHasFilmTitul', 'Filtro', 'khf')

%Adicionar o elemento
function F = add(B,elemento,k)
key = elemento;

for i = 1:k
    key = [key num2str(i)];
```



```
hash = string2hash(key);  
hash = rem(hash,length(B))+1;  
B(hash) = B(hash) + 1;  
end  
F = B;  
return  
end
```

Para decidi o valor de K para a MinHash da avaliação filmes por users, rever o exercício 4 da secção 4.3 do último guião prático, reparei que para K = 50, 100 e 200 os tempos de calculo de MinHash são 71, 167 e 408 segundos respetivamente, e que o cálculo das distâncias de Jaccard para todos os pares possíveis de utilizadores é demoram 9, 2 e 3 segundos respetivamente e com tempos de cálculo do par mais similar bastante semelhantes entre todos. Com estes resultados obtidos e ao ver qual o verdadeiro valor da distância de Jaccard entre 2 pares de utilizadores, achai por bem utilizar o valor de K = 100, vistos que este script apenas irá ser executado periodicamente. E com isso o cálculo no “main.m” fica mais rápido.

Ao variar o tamanho do shingle entre 2 e 5 pude ver que o cálculo de todos os shingles possíveis para todos os filmes varia entre os 0,7 segundos para um size de 2 até 0,3 segundos para um size de 5, chegando à conclusão de que pelo menos para esta fase na criação dos shingles o tamanho não afeta a aplicação, e com escolhi o 3.

Para decidir o valor K para a MinHash dos shingles voltei a fazer o teste de K para 50, 100, 150 e 200 para ver quanto tempo o código demoraria, obtendo 25, 53, 90 e 134 segundos respetivamente. Tal como na função MinHash anterior o balanço é muito simples, um valor de K maior vai consequentemente dar um valor mais exato para o cálculo das distâncias de Jaccard mas demorando mais tempo a fazer as suas MinHash, decidir então ficar pelo valor K = 150.



## Main

### Menu

```
clear;
load("dados.mat") % todos os dados necessário
%main--> programa principal
prompt = "Insert Film ID (1 to 1682): ";
id = input(prompt);
op = 0;
while(op ~= 5)
    fprintf("\n")
    fprintf("1- Users that evaluated current movie\n")
    fprintf("2- Suggestion of users to evaluate movie\n")
    fprintf("3- Suggestion of users to based on common interests\n")
    fprintf("4- Movies feedback based on popularity\n")
    fprintf("5- Exit\n")
    prompt = "Select choice: ";
    op = input(prompt);
    switch(op)

        case 1
            listName(id,Id_Name,set);
        case 2
            listSuggestion(id,users,Id_Name,set,MinHasUseAv);
        case 3
            ListSuggetInterests(id,users,Id_Name,set,MinHasUseIntrs);
        case 4
            feedback(titulos,Filtro,MinHasFilmTitul,4);
        case 5
            fprintf("Exit...\n")
        otherwise
            fprintf("Invalid choise! Try aguain\n")
    end
end
```





Comecei por fazer a leitura dos dados armazenados no “dados.m”, e depois dando o início ao programa pedindo o id do filme e depois apresentará o menu. E de acordo com o as opções do menu o user irá receber os dados pretendido. Infelizmente não consegui fazer todas as opções funcionarem.

### A função listName

```
function listName(id,setName,setAll)
    fprintf("Users that evaluated this movie: ID = %d\n",id);
    fprintf("ID - NAME:\n");
    for n1 = 1:length(setName) % por cada users
        conjunto = setAll{n1}; %os filmes avaliados pelo mesmo
        for i = 1:length(conjunto)
            if conjunto(i) == id %se o mesmo avaliou o filme atual printa
                fprintf("%3d - %s\n",setName{n1,1},setName{n1,2});
            end
        end
    end
end
```

Esta função tem como objetivo listar todos os nomes dos users que avaliaram o filme inserido pelo utilizador. Aceita como parâmetros o ID do filme escolhido (id), o Set que contem todos os nomes dos users e seus Ids (setName ) e o setAll que contem todos os filmes avaliados por cada um do user.

### Função listSuggestion

```
function listSuggestion(id,users,setName,setAll,minaHash)
    J = jcard(minaHash,length(setName)); % calcular a distancia de jcard
    SimilarUsers = similaridade(J,0.4,length(setName),users); %pares
    similares

    [maxSiliar ind] = max(SimilarUsers(:,3)); % os pares com maior
    similaridade e o seu indice
    maisSimilar = SimilarUsers(ind,:);

    filmEvaUser1 = setAll{maisSimilar(1)}; %os filmes avaliados pelo user 1
    dos pares
    filmEvaUser2 = setAll{maisSimilar(2)}; %os filmes avaliados pelo user 2
    dos pares
    Suggestion = [];

    if ~ismember(filmEvaUser1,id) % se o user ainda não avaliou o atual
    filme
        Suggestion = [ Suggestion maisSimilar(1)];
    end
end
```



```
end
if ~ismember(filmEvaUser2,id)
    Suggestion = [Suggestion maisSimilar(2)];
end

if isempty(Suggestion)
    fprintf("\nThere is no film suggestions.\n");
else
    fprintf('\nSuggestions of users:\n');
    for i = 1:length(Suggestion)
        fprintf("%3d - %s\n",setName{Suggestion(i),1},setName{Suggestion(i),2})
    end
end
end
```

A função listSuggestion tem como objetivo listar dois users similares nos filmes avaliados e apresentar ao utilizador o nome e o Id do(s) user(s) que não avaliaram o filme atual. Se os users mais similares avaliaram o filme atual não é apresentado ao utilizador. Esta função usa outras duas funções: jacard e similaridade.

## Jacard

```
%distancia de Jacard
function J = jacard(MinHash,Nu)
    khf = 100; % o mesmo usado no processData
    J = zeros(Nu);
    h= waitbar(0,'Calculating');
    for n1= 1:Nu
        waitbar(n1/Nu,h);
        for n2= n1+1:Nu
            num = sum(MinHash(n1,:) == MinHash(n2,:));
            J(n1,n2) = 1 - (num/khf);
        end
    end
    delete (h)
end
```

## Similaridade

```
%Similaridade usando distancia dado por minhash
function SimilarUsers = similaridade(J,threshold,Nu,users)
%threshold = 0.4;
SimilarUsers = zeros(1,3);
j = 1;
for n1 = 1:Nu
    for n2 = n1+1: Nu
        if J(n1,n2) <= threshold
            SimilarUsers(j,:) = [users(n1) users(n2) J(n1,n2)];
            j = j+1;
        end
    end
end
```



```
end  
end
```

## ListSuggestInterests

```
function ListSuggetInterests(id,users,setName,setAll,minaHash)  
    J = jacard(minaHash,length(setName)); % calcular a distancia de  
jacard  
    threshold = 0.9;  
    setId = {};  
  
    for n1 = 1:length(setName) % por cada users  
        conjunto = setAll{n1}; %os filmes avaliados pelo mesmo  
        setId{n1} = [];  
        for n2 = n1+1: length(setName)  
            conjunto2 = setAll{n2};  
            if ~ismember(conjunto2,id)  
                if ismember(conjunto,id)  
                    if J(n1,n2) < threshold  
                        setId{n1} = [setId{n1} n2];  
                    end  
                end  
            end  
        end  
        NumOcors = [];  
        for n1 = 1:length(setName)  
            conjunto = setId{n1};  
            ocorencias = [];  
            for n2 = 1:length(conjunto)  
                ocorencias(n2) = 0;  
                for i = 1:length(conjunto)  
                    if n2 == conjunto(i,1)  
                        ocorencias(n2) = ocorencias(n2) + 1;  
                        NumOcors(n2) = [NumOcors(n2) ocorencias(n2) ];  
                    end  
                end  
            end  
        end  
        for i = 1:2  
            [maxi, ind] = max(NumOcors); % Calcular o valor maximo  
            fprintf("%3d - %s\n",setName{ind,1},setName{ind,2})  
            NumOcors(ind) = 0; % Retirar  
        end  
    end  
end
```

Esta função tem como objetivo listar os Ids e os nomes de dois utilizadores que apareceram mais vezes no conjunto de interesses cuja a distancia de jacard seja menor ou igual a 0.9. Infelizmente não consegui colocar esta função a funcionar. E não consegui achar o erro que condicionou o mal funcionamento.



## feedback

```
function feedback(titulos,filtro,MinHashSig,khf)
    nome = lower(input('\nWrite the name : ', 's'));
    shin_size = 3; % o mesmo numero de shingles usado no processamento de
dados
    kf = size(MinHashSig, 2); % kf igual ao kf utilizado no processamento
para os shingles dos titulos

    % Cell array com os shingles do nome introduzido
    shinglesNom = {};
    for i = 1:length(nome) - shin_size+1
        shingle = nome(i:i+shin_size-1);
        shinglesNom{i} = shingle;
    end

    % MinHash dos shingles do nome introduzido
    MinHashNome = inf(1,kf);
    for j = 1:length(shinglesNom)
        chave = char(shinglesNom{j});
        hash = zeros(1, kf);
        for hf = 1: kf
            chave = [chave num2str(hf)];
            hash(hf) = DJB31MA(chave, hf);
        end
        MinHashNome(1,:) = min([MinHashNome(1,:); hash]);
    end

    % Distancia de Jaccard entre a string e cada filme
    J = ones(1, length(titulos)); % array para guardar distancias
    h = waitbar(0, 'Calculating');
    for i=1:length(titulos) % cada hashcode da string
        waitbar(i/ kf, h);
        num = sum(MinHashSig(i,:) == MinHashNome);
        J(i) = 1 - (num/ kf);
    end
    delete(h);

    filme = false; %para saber se houve algum filme encontrado com uma
distancia menor ou igual a threshold

    for i = 1:3
        [maximo, ind] = max(J); % Calcular o valor minimo (mais
similaridade)
        filme = true;
        contador = contagem(filtro,titulos{ind,1},khf);
        fprintf('Filme: %s foi avaliado %d com nota superior ou igual a
3\n', titulos{ind,1}, contador);
        J(ind) = 0; % Retirar esse filme dando uma distancia igual a 1
contagem(B,elemento,k)
    end

    if (~filme)
        fprintf('No movies found.\n');
    end
end
```



```
%% contagem()
function multipli = contagem(B,elemento,k)
    member = zeros(1,k);
    key = elemento;

    for i = 1:k
        key = [key num2str(i)];
        hash = string2hash(key);
        hash = rem(hash,length(B))+1;
        B(hash) = B(hash) + 1;
        member(i) = B(hash);
    end
    multipli = min(member);

end
```

A função feedback, tem como objetivo achar e listar 3 nomes do filme similar ao nome introduzido pelo utilizador e o número de vezes que os mesmo tiveram a avaliação maior ou igual que 3. E comecei por calcular o shingles do nome dado pelo utilizador, para calcular o minhash do mesmo, em seguida é comparado com a minhash dos títulos dos filmes para calcular a distancia de jaccard. Por algum motivo não consigo calcular a distancia, o mesmo dá sempre 1. E para o número de vezes que os filmes encontrados tiveram a avaliação maior ou igual que 3 é usado a função contagem que irá calcular o hash de cada filme, armazenar na posição hash no filtro o número de ocorrências do mesmo e finalmente é enviado o número mínimo das ocorrências.



## Nota Final

Fiz o trabalho sozinho, visto que o elemento do meu grupo desistiu da cadeira, o código foi implementado por mim, mas contei com ajuda de dois dos meus colegas de outra turma para corrigir alguns erros e esclarecer o enunciado. Usei alguns dos exercícios implementados nas aulas praticas.