

**Métodos probabilísticos para a Engenharia informática**

## **Avaliação PL04**

Rafael Santos - 98466

Gustavo Silveira - 96141

## Introdução

Neste trabalho será apresentada a criação de um sistema online de disponibilização de filmes com o objetivo de utilizar métodos dados durante o semestre na cadeira de MPEI.

Vai ser criada uma ferramenta em que o utilizador, através do ficheiro " main.m " interage com o programa podendo realizar operações tais como:

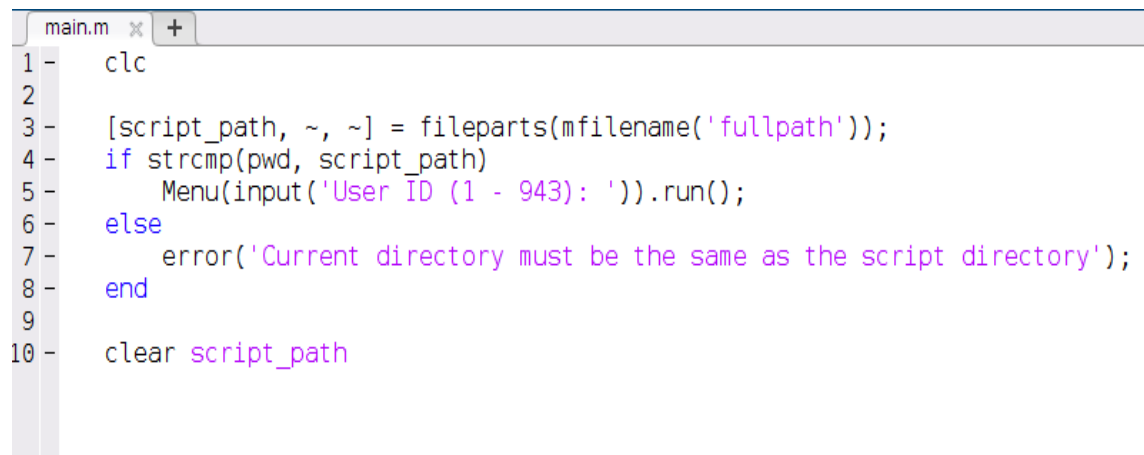
- Ver os filmes vistos pelo utilizador;
- Obter sugestões de filmes;
- Procurar o título de um filme;

Foram também utilizados mais scripts, "utils.m" ,"MovieGenre" e "menu.m" que servirão de apoio ao programa principal. Sem ser necessário qualquer contacto com eles.

## Main.m

Este vai ser o ficheiro principal.

Antes de pedir ao utilizador o ID e mostrar o respetivo menu, o programa vai encontrar o ficheiro e respetivo diretório (linha 3). Se o diretório atual (indicado pela função pwd) for o mesmo do obtido na linha anterior (comparação feita pela função strcmp ) então o programa irá correr normalmente. Caso não se verifique o programa irá parar e indicar uma mensagem de erro.



```
main.m x +
1 -   clc
2
3 -   [script_path, ~, ~] = fileparts(mfilename('fullpath'));
4 -   if strcmp(pwd, script_path)
5 -       Menu(input('User ID (1 - 943): ')).run();
6 -   else
7 -       error('Current directory must be the same as the script directory');
8 -   end
9
10 -  clear script_path
```

# Utils.m

Este ficheiro vai ser usado como base às operações de minHash e similaridade.

Se o ficheiro “database.mat” não existir vai ser pedido ao utilizador que introduza o ficheiro “u.data” e “u\_item.txt” de modo a obter a base de dados

A função “findSimilarUsers” vai ser utilizada para verificar a similaridade entre os utilizadores. Recebe como argumentos a distância de Jaccard dos utilizadores , o threshold e o id de cada utlizador. Começa por calcular a similiaridade recorrendo a uma outra função “Similarity”.

Na função “JaccardDistance” vai ser calculada a distância de Jaccard onde recebe como argumento os valores de minHash de cada utilizador e retorna a distância de Jaccard de cada par de filme ou usuário.

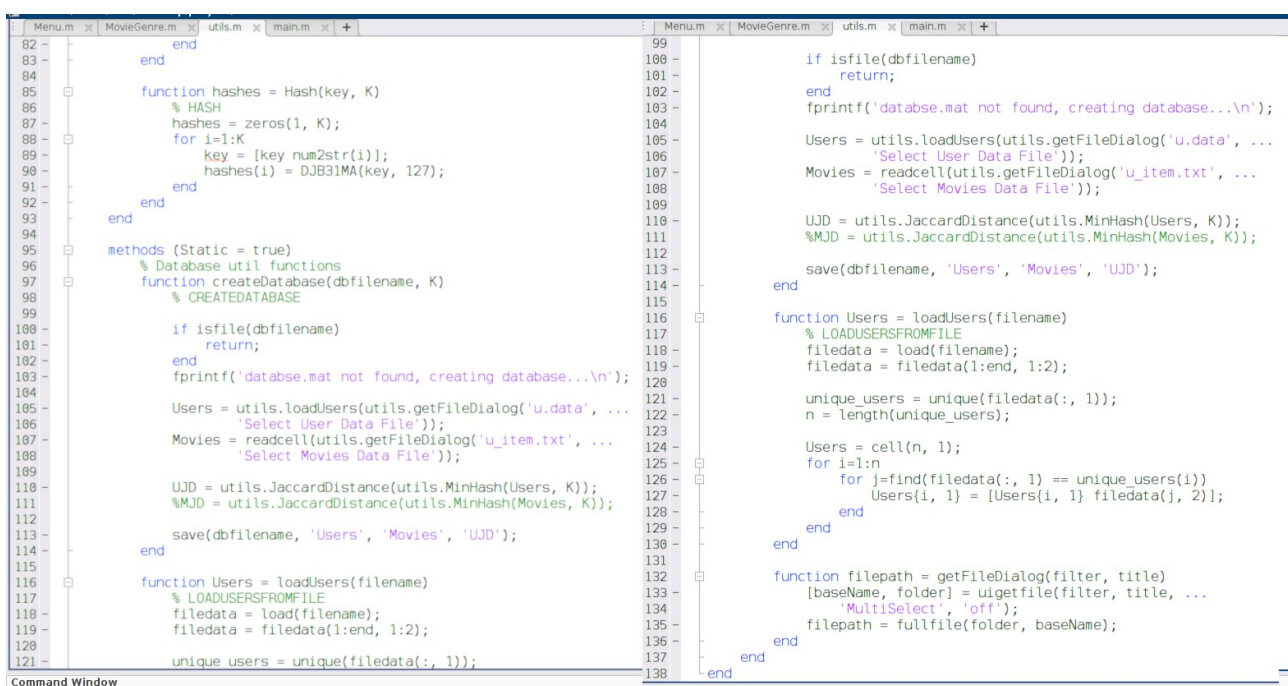
```
1 classdef utils
2     % UTILS utility class
3
4     methods (Static = true)
5         % Algo utils
6
7
8         function similar_users = findSimilarUsers(UJD, threshold, id)
9             % FINDSIMILARUSERS
10            similarity = utils.Similarity(UJD, threshold);
11            [rows, ~] = find(similarity == id);
12            similar_users = similarity(rows, :, :);
13
14            % fix order
15            for i=1:length(similar_users)
16                if similar_users(i, 2) == id
17                    similar_users(i, :) = [similar_users(i, 2) ...
18                                           similar_users(i, 1) similar_users(i, 3)];
19                end
20            end
21
22            function similarity = Similarity(JD, threshold)
23                % SIMILARITY
24                [rows, cols] = size(JD);
25                similarity = zeros(1, 3);
26                idx = 1;
27                for i=1:rows
28                    for j=i+1:cols
29                        if JD(i, j) <= threshold
30                            similarity(idx, :) = [i j JD(i, j)];
31                            idx = idx + 1;
32                        end
33                    end
34                end
35            end
36
37            function JD = JaccardDistance(mhValues)
38                % JACCARDDISTANCE
39                [n, k] = size(mhValues);
40                JD = zeros(n);
41                for i=1:n
```

Illustration 1: utils.m(parte 1 e 2 , respetivamente)

Na função “Shingles” recebe como argumento a string escrita pelo utilizador (text) e as k hash functions.De seguida organiza o titulo de cada filme em shingles e armazena num cell array.

A função Hash vai , através da hash function dada na aula DJB31MA , calcular os valores de hash para cada utilizador.

Como indicado no início , na secção Introdução , caso não haja uma base de dados criada o programa vai pedir para seleccionar 2 ficheiros e vai ser criada uma. Essa habilidade está ao encargo da função “createDataBase”.Caso já tenha sido criada , o programa funciona normalmente.



```
82 - end
83 -
84 -
85 - function hashes = Hash(key, K)
86 -     % HASH
87 -     hashes = zeros(1, K);
88 -     for i=1:K
89 -         key = [key num2str(i)];
90 -         hashes(i) = DJB31MA(key, 127);
91 -     end
92 - end
93 -
94 -
95 - methods (Static = true)
96 -     % Database util functions
97 -     function createDataBase(dbfilename, K)
98 -         % CREATEDATABASE
99 -
100 -         if isfile(dbfilename)
101 -             return;
102 -         end
103 -         fprintf('database.mat not found, creating database...\n');
104 -
105 -         Users = utils.loadUsers(utils.getFileDialog('u.data', ...
106 -             'Select User Data File'));
107 -         Movies = readcell(utils.getFileDialog('u_item.txt', ...
108 -             'Select Movies Data File'));
109 -
110 -         UJD = utils.JaccardDistance(utils.MinHash(Users, K));
111 -         %MJD = utils.JaccardDistance(utils.MinHash(Movies, K));
112 -
113 -         save(dbfilename, 'Users', 'Movies', 'UJD');
114 -     end
115 -
116 -     function Users = loadUsers(filename)
117 -         % LOADUSERSFROMFILE
118 -         filedata = load(filename);
119 -         filedata = filedata(1:end, 1:2);
120 -
121 -         unique_users = unique(filedata(:, 1));
122 -         n = length(unique_users);
123 -
124 -         Users = cell(n, 1);
125 -         for i=1:n
126 -             for j=find(filedata(:, 1) == unique_users(i))
127 -                 Users{i, 1} = [Users{i, 1} filedata(j, 2)];
128 -             end
129 -         end
130 -     end
131 -
132 -     function filepath = getFileDialog(filter, title)
133 -         [baseName, folder] = uigetfile(filter, title, ...
134 -             'MultiSelect', 'off');
135 -         filepath = fullfile(folder, baseName);
136 -     end
137 - end
138 -
139 -
140 -
141 -
142 -
143 -
144 -
145 -
146 -
147 -
148 -
149 -
150 -
151 -
152 -
153 -
154 -
155 -
156 -
157 -
158 -
159 -
160 -
161 -
162 -
163 -
164 -
165 -
166 -
167 -
168 -
169 -
170 -
171 -
172 -
173 -
174 -
175 -
176 -
177 -
178 -
179 -
180 -
181 -
182 -
183 -
184 -
185 -
186 -
187 -
188 -
189 -
190 -
191 -
192 -
193 -
194 -
195 -
196 -
197 -
198 -
199 -
200 -
201 -
202 -
203 -
204 -
205 -
206 -
207 -
208 -
209 -
210 -
211 -
212 -
213 -
214 -
215 -
216 -
217 -
218 -
219 -
220 -
221 -
222 -
223 -
224 -
225 -
226 -
227 -
228 -
229 -
230 -
231 -
232 -
233 -
234 -
235 -
236 -
237 -
238 -
239 -
240 -
241 -
242 -
243 -
244 -
245 -
246 -
247 -
248 -
249 -
250 -
251 -
252 -
253 -
254 -
255 -
256 -
257 -
258 -
259 -
260 -
261 -
262 -
263 -
264 -
265 -
266 -
267 -
268 -
269 -
270 -
271 -
272 -
273 -
274 -
275 -
276 -
277 -
278 -
279 -
280 -
281 -
282 -
283 -
284 -
285 -
286 -
287 -
288 -
289 -
290 -
291 -
292 -
293 -
294 -
295 -
296 -
297 -
298 -
299 -
300 -
301 -
302 -
303 -
304 -
305 -
306 -
307 -
308 -
309 -
310 -
311 -
312 -
313 -
314 -
315 -
316 -
317 -
318 -
319 -
320 -
321 -
322 -
323 -
324 -
325 -
326 -
327 -
328 -
329 -
330 -
331 -
332 -
333 -
334 -
335 -
336 -
337 -
338 -
339 -
340 -
341 -
342 -
343 -
344 -
345 -
346 -
347 -
348 -
349 -
350 -
351 -
352 -
353 -
354 -
355 -
356 -
357 -
358 -
359 -
360 -
361 -
362 -
363 -
364 -
365 -
366 -
367 -
368 -
369 -
370 -
371 -
372 -
373 -
374 -
375 -
376 -
377 -
378 -
379 -
380 -
381 -
382 -
383 -
384 -
385 -
386 -
387 -
388 -
389 -
390 -
391 -
392 -
393 -
394 -
395 -
396 -
397 -
398 -
399 -
400 -
401 -
402 -
403 -
404 -
405 -
406 -
407 -
408 -
409 -
410 -
411 -
412 -
413 -
414 -
415 -
416 -
417 -
418 -
419 -
420 -
421 -
422 -
423 -
424 -
425 -
426 -
427 -
428 -
429 -
430 -
431 -
432 -
433 -
434 -
435 -
436 -
437 -
438 -
439 -
440 -
441 -
442 -
443 -
444 -
445 -
446 -
447 -
448 -
449 -
450 -
451 -
452 -
453 -
454 -
455 -
456 -
457 -
458 -
459 -
460 -
461 -
462 -
463 -
464 -
465 -
466 -
467 -
468 -
469 -
470 -
471 -
472 -
473 -
474 -
475 -
476 -
477 -
478 -
479 -
480 -
481 -
482 -
483 -
484 -
485 -
486 -
487 -
488 -
489 -
490 -
491 -
492 -
493 -
494 -
495 -
496 -
497 -
498 -
499 -
500 -
501 -
502 -
503 -
504 -
505 -
506 -
507 -
508 -
509 -
510 -
511 -
512 -
513 -
514 -
515 -
516 -
517 -
518 -
519 -
520 -
521 -
522 -
523 -
524 -
525 -
526 -
527 -
528 -
529 -
530 -
531 -
532 -
533 -
534 -
535 -
536 -
537 -
538 -
539 -
540 -
541 -
542 -
543 -
544 -
545 -
546 -
547 -
548 -
549 -
550 -
551 -
552 -
553 -
554 -
555 -
556 -
557 -
558 -
559 -
560 -
561 -
562 -
563 -
564 -
565 -
566 -
567 -
568 -
569 -
570 -
571 -
572 -
573 -
574 -
575 -
576 -
577 -
578 -
579 -
580 -
581 -
582 -
583 -
584 -
585 -
586 -
587 -
588 -
589 -
590 -
591 -
592 -
593 -
594 -
595 -
596 -
597 -
598 -
599 -
600 -
601 -
602 -
603 -
604 -
605 -
606 -
607 -
608 -
609 -
610 -
611 -
612 -
613 -
614 -
615 -
616 -
617 -
618 -
619 -
620 -
621 -
622 -
623 -
624 -
625 -
626 -
627 -
628 -
629 -
630 -
631 -
632 -
633 -
634 -
635 -
636 -
637 -
638 -
639 -
640 -
641 -
642 -
643 -
644 -
645 -
646 -
647 -
648 -
649 -
650 -
651 -
652 -
653 -
654 -
655 -
656 -
657 -
658 -
659 -
660 -
661 -
662 -
663 -
664 -
665 -
666 -
667 -
668 -
669 -
670 -
671 -
672 -
673 -
674 -
675 -
676 -
677 -
678 -
679 -
680 -
681 -
682 -
683 -
684 -
685 -
686 -
687 -
688 -
689 -
690 -
691 -
692 -
693 -
694 -
695 -
696 -
697 -
698 -
699 -
700 -
701 -
702 -
703 -
704 -
705 -
706 -
707 -
708 -
709 -
710 -
711 -
712 -
713 -
714 -
715 -
716 -
717 -
718 -
719 -
720 -
721 -
722 -
723 -
724 -
725 -
726 -
727 -
728 -
729 -
730 -
731 -
732 -
733 -
734 -
735 -
736 -
737 -
738 -
739 -
740 -
741 -
742 -
743 -
744 -
745 -
746 -
747 -
748 -
749 -
750 -
751 -
752 -
753 -
754 -
755 -
756 -
757 -
758 -
759 -
760 -
761 -
762 -
763 -
764 -
765 -
766 -
767 -
768 -
769 -
770 -
771 -
772 -
773 -
774 -
775 -
776 -
777 -
778 -
779 -
780 -
781 -
782 -
783 -
784 -
785 -
786 -
787 -
788 -
789 -
790 -
791 -
792 -
793 -
794 -
795 -
796 -
797 -
798 -
799 -
800 -
801 -
802 -
803 -
804 -
805 -
806 -
807 -
808 -
809 -
810 -
811 -
812 -
813 -
814 -
815 -
816 -
817 -
818 -
819 -
820 -
821 -
822 -
823 -
824 -
825 -
826 -
827 -
828 -
829 -
830 -
831 -
832 -
833 -
834 -
835 -
836 -
837 -
838 -
839 -
840 -
841 -
842 -
843 -
844 -
845 -
846 -
847 -
848 -
849 -
850 -
851 -
852 -
853 -
854 -
855 -
856 -
857 -
858 -
859 -
860 -
861 -
862 -
863 -
864 -
865 -
866 -
867 -
868 -
869 -
870 -
871 -
872 -
873 -
874 -
875 -
876 -
877 -
878 -
879 -
880 -
881 -
882 -
883 -
884 -
885 -
886 -
887 -
888 -
889 -
890 -
891 -
892 -
893 -
894 -
895 -
896 -
897 -
898 -
899 -
900 -
901 -
902 -
903 -
904 -
905 -
906 -
907 -
908 -
909 -
910 -
911 -
912 -
913 -
914 -
915 -
916 -
917 -
918 -
919 -
920 -
921 -
922 -
923 -
924 -
925 -
926 -
927 -
928 -
929 -
930 -
931 -
932 -
933 -
934 -
935 -
936 -
937 -
938 -
939 -
940 -
941 -
942 -
943 -
944 -
945 -
946 -
947 -
948 -
949 -
950 -
951 -
952 -
953 -
954 -
955 -
956 -
957 -
958 -
959 -
960 -
961 -
962 -
963 -
964 -
965 -
966 -
967 -
968 -
969 -
970 -
971 -
972 -
973 -
974 -
975 -
976 -
977 -
978 -
979 -
980 -
981 -
982 -
983 -
984 -
985 -
986 -
987 -
988 -
989 -
990 -
991 -
992 -
993 -
994 -
995 -
996 -
997 -
998 -
999 -
1000 -
```

Illustration 2: utils.m( parte 3 e 4 , respetivamente)

Se não tiver uma base de dados criada , o programa através das funções “loadUsers”, “JaccardDistance”, “minHash” e do ficheiro de texto, vai extrair os dados e armazena-los.

A função “loadUsers” , como já referido , vai retirar os dados de todos os utilizadores do ficheiro e guarda-los.

# Menu.m

Na classe Menu vai ser projetado o menu interativo e a realização de cada uma das suas funcionalidades.

Começamos por definir definir properties em que o valor do ID introduzido pelo utilizador não poderá ser menor do que 0 nem maior que 943 , caso aconteça o programa terminará com uma mensagem de erro no terminal.

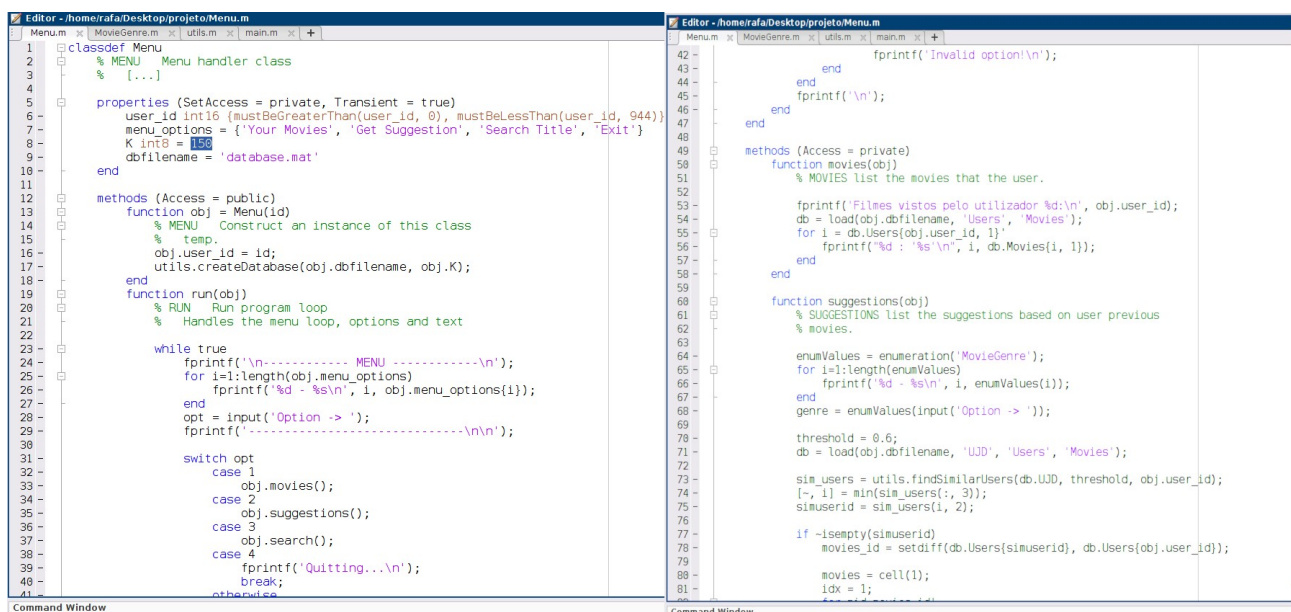
Começa-se por criar um construtor de modo a poder aceder a todos os atributos do utilizador( function obj).

Na função run é onde começa a utilização do menu.Quando é selecionada a opção 1 do menu , o sistema vai mostrar todos os filmes já visualizados por ele.Primeiro faz-se o load das informações do utilizador(ID e filmes vistos) para um cell array(linhas 53 e 54). Depois é só imprimir.

Na opção 2 é pedido que , com base nos outros utilizadores e no género de filme pretendido , o utilizador obtenha sugestões de filmes que ainda não tenha visto.

Começa-se por aceder ao ficheiro MovieGenre , dispondo-os no ecrã. De seguida volta-se a fazer o load das informações ,desta vez para além dos utilizadores e dos filmes também é pedido a distância de Jaccard entre os utilizadores.De seguida calcula-se a similaridade entre si(linha 73).

Caso haja similaridade entre utilizadores está na altura de verificar se o utilizador do menu atual e do utilizador com maior similaridade entre si se possui algum filme não visto dentro do género pedido anteriormente, se sim o filme é armazenado num cell array e os filmes são impressos no ecrã(linhas 77 a 93). Caso contrário o programa imprime a mensagem 'No suggestions found'.



```
1 classdef Menu
2     % MENU Menu handler class
3     % [...]
4
5     properties (SetAccess = private, Transient = true)
6         user_id int16 {mustBeGreaterThan(user_id, 0), mustBeLessThan(user_id, 944)}
7         menu_options = {'Your Movies', 'Get Suggestion', 'Search Title', 'Exit'}
8         K int8 = 150
9         dbfilename = 'database.mat'
10    end
11
12    methods (Access = public)
13        function obj = Menu(id)
14            % MENU Construct an instance of this class
15            % temp.
16            obj.user_id = id;
17            utils.createDatabase(obj.dbfilename, obj.K);
18        end
19        function run(obj)
20            % RUN Run program loop
21            % Handles the menu loop, options and text
22
23            while true
24                fprintf('\n----- MENU ----- \n');
25                for i=1:length(obj.menu_options)
26                    fprintf('%d - %s\n', i, obj.menu_options{i});
27                end
28                opt = input('Option -> ');
29                fprintf('\n----- \n\n');
30
31                switch opt
32                    case 1
33                        obj.movies();
34                    case 2
35                        obj.suggestions();
36                    case 3
37                        obj.search();
38                    case 4
39                        fprintf('Quitting...\n');
40                        break;
41                    otherwise
42                        fprintf('Invalid option!\n');
43                end
44            end
45        end
46    end
47
48    methods (Access = private)
49        function movies(obj)
50            % MOVIES list the movies that the user.
51
52            fprintf('Filmes vistos pelo utilizador %d:\n', obj.user_id);
53            db = load(obj.dbfilename, 'Users', 'Movies');
54            for i = db.Users(obj.user_id, 1)
55                fprintf('%d : %s\n', i, db.Movies(i, 1));
56            end
57        end
58
59        function suggestions(obj)
60            % SUGGESTIONS list the suggestions based on user previous
61            % movies.
62
63            enumValues = enumeration('MovieGenre');
64            for i=1:length(enumValues)
65                fprintf('%d - %s\n', i, enumValues{i});
66            end
67            genre = enumValues(input('Option -> '));
68
69            threshold = 0.6;
70            db = load(obj.dbfilename, 'UJD', 'Users', 'Movies');
71
72            sim_users = utils.findSimilarUsers(db.UJD, threshold, obj.user_id);
73            [~, i] = min(sim_users(:, 3));
74            simuserid = sim_users(i, 2);
75
76            if ~isempty(simuserid)
77                movies_id = setdiff(db.Users(simuserid), db.Users(obj.user_id));
78
79                movies = cell(1);
80                idx = 1;
81                for i = 1:length(movies_id)
82                    movies{idx} = db.Movies(movies_id(i), 1);
83                    idx = idx + 1;
84                end
85                fprintf('Sugestões de filmes que ainda não tenha visto:\n');
86                for i = 1:length(movies)
87                    fprintf('%d - %s\n', i, movies{i});
88                end
89            else
90                fprintf('No suggestions found.\n');
91            end
92        end
93    end
94 end
```

Illustration 3: Menu.m (parte 1 e 2 , respetivamente)

Na opção 3 é pedido ao utilizador que introduza uma String de modo a pesquisar por um filme. Para realizar esse pedido começa-se por fazer load da base de dados da parte dos filmes( db ).

De seguida armazenamos a String num cell array e calculamos os shingles( linha 107 ). Dentro do ciclo vamos calcular os shingles de todos os filmes e armazena-los no array Smovies.

Em seguida calculamos a distância de Jaccard entre todos os filmes utilizando as funções “MinHash” e “JaccardDistance”, organizamos por ordem crescente de distâncias de Jaccard (Função sort) e por fim são imprimidos no máximo 5 valores cuja distância de Jaccard não ultrapasse o threshold(0.99 , neste caso).

```
Editor - /home/rafa/Desktop/projeto/Menu.m
Menu.m x MovieGenre.m x utils.m x main.m x +
81 -
82 -     for mid=movies_id'
83 -         if db.Movies[mid, genre] == 1
84 -             movies(idx) = db.Movies[mid, 1];
85 -             idx = idx + 1;
86 -         end
87 -     end
88 -
89 -     if ~isempty(movies)
90 -         fprintf("Title: '%s'\n", movies{:});
91 -         return;
92 -     end
93 - end
94 -
95 - fprintf('No suggestions found!\n');
96 - end
97 -
98 - function search(obj)
99 -     % SEARCH search movies based on user choice.
100 -
101 -     db = load('database.mat', 'Movies');
102 -     k = 3;
103 -
104 -     title = input('Search: ', 's');
105 -
106 -     SMovies = cell(1);
107 -     SMovies{1, 1} = utils.Shingles(lower(title), k);
108 -     for i=2:length(db.Movies)+1
109 -         SMovies{i, 1} = utils.Shingles(lower(db.Movies{i-1, 1}), k);
110 -     end
111 -
112 -     MJD = utils.JaccardDistance(utils.MinHash(SMovies, obj.K));
113 -     movies = sort(utils.Similarity(MJD[1, :], 0.99), 'ascend');
114 -     [rows, ~] = size(movies);
115 -
116 -     for i=1:min([rows 5])
117 -         fprintf("%.2f: %s\n", movies(i, 3), db.Movies[movies(i, 2)-1, 1]);
118 -     end
119 - end
120 - end
```

Illustration 4: Menu.m (parte 3)

```
Menu.m x MovieGenre.m x utils.m x main.m x +
1  classdef MovieGenre < uint8
2      enumeration
3          Action (3)
4          Adventure (4)
5          Animation (5)
6          Children (6)
7          Comedy (7)
8          Crime (8)
9          Documentary (9)
10         Drama (10)
11         Fantasy (11)
12         FilmNoir (12)
13         Horror (13)
14         Musical (14)
15         Mystery (15)
16         Romance (16)
17         SciFi (17)
18         Thriller (18)
19         War (19)
20         Western (20)
21     end
22 end
23
```

Illustration 5: Movie Genre\