

## AI-Assignment-5

Abhinav Saurabh MT20127

- First I have made an interface.py to extract the facts and store them into the file which can be used further by the prolog program.
- Then prolog program takes the input from the facts saved in the text file from the previous program.
- The further program gives the recommended suggestions.

### Interface Program

- I have used python programming and Natural language processing libraries to get the facts from the input sentences.
- Preprocessing steps.
  - Lowering the words
  - Removing Punctuations
  - Lemmatizing words
  - Removed stop words
- Keywords are extracted and stored in the facts.txt.
- Following format is followed
  - yes('keyword')
  - no('keyword')

```
10 import nltk
11 from nltk.tokenize import word_tokenize
12 from nltk.stem import PorterStemmer
13 from nltk.stem import WordNetLemmatizer
14 from nltk.corpus import stopwords
15 import string
16 import warnings
17 warnings.filterwarnings('ignore')
18
19 nltk.download('punkt')
20 nltk.download('wordnet')
21 nltk.download('stopwords')
22
23 inplist = []
24 stopWords = set(stopwords.words('english'))
25
26 wordnet_lemmatizer = WordNetLemmatizer()
27
28 print("What are your interests and hobbies ?\n") #Taking user input
29
30 inp1 = input()
31
32 text = inp1.lower() #lowering the words
33 for sgn in string.punctuation: #removing punctuations
34     text= text.replace(sgn, ' ')
35
36
37 text = wordnet_lemmatizer.lemmatize(text) #lemmatizing_words
38
39
40 tok1 = word_tokenize(text) #extract tokens
41
42 for wod in tok1:
```

```

42 for wod in tok1:
43     if wod not in stopWords: #remove stop words
44         inplist.append(wod)
45
46
47 f = open("facts.txt", 'w')
48
49 oracle = ['python','programming','ml','database','advanceprogramming','probability','discreetmaths','cn',
50
51 for key in oracle:
52     if key in inplist:
53         f.write("yes('")
54         f.write(key)
55         f.write("').\n")
56     else:
57         f.write("no('")
58         f.write(key)
59         f.write("').\n")
60
61 inp1 = input("Are you interested in Artificial Intelligence?")
62 f.write("ai('")
63 f.write(inp1)
64 f.write("').\n")
65
66 inp2 = input("Are you interested in Data Engineering?")
67 f.write("de('")
68 f.write(inp2)
69 f.write("').\n")
70
71 inp3 = input("Are you interested in Information Security?")
72 f.write("is('")
73 f.write(inp3)
74 f.write("').\n")
75
76 inp4 = input("Are you interested in Mobile Computing?")
77 f.write("mc('")
78 f.write(inp4)
79 f.write("').\n")
80
81
82
83 f.close()

```

Input given:

```

I am interested in ai and have some programming experience. I also know python.
Are you interested in Artificial Intelligence?yes
Are you interested in Data Engineering?no
Are you interested in Information Security?no
Are you interested in Mobile Computing?no

```

The output obtained in facts.txt.

```

yes('python').
yes('programming').
no('ml').
no('database').
no('advanceprogramming').
no('probability').
no('discreetmaths').
no('cn').
no('electronics').
no('cryptography').
ai(yes).
de(no).
is(no).
mc(no).

```

## Prolog Program:

The program will give the output of the electives using the facts generated by the previous interface program.

```
1  get_advice:-
2
3
4      advice(_),
5      preferences(List),nl,
6
7      (isempty(List)
8          ->write('Sorry cannot recommend you anything' ),nl
9          ;write('Electives for you are :'),show(List)
10     ),
11     retractall(recommend(_)),
12     retractall(ai(_)),
13     retractall(de(_)),
14     retractall(is(_)),
15     retractall(mc(_)),
16     clear.
17
18  get_facts:-
19  open('/Users/abhinavsaurabh/Desktop/IIITD/Abhinav-MT20127/facts.txt', read, Str),
20  read_file(Str,Lines),
21  close(Str),
22  facts(Lines).
23
24  facts([Head|Tail]):-
25      ((Head == end_of_file)
26          ->facts(Tail)
27          ;assert(Head), facts(Tail)
28      ).
29  facts([]).
30
31  advice('Machine Learning') :- ml,fail.
32  advice('Information Retrieval') :- ir,fail.
33  advice('Data Mining') :- dm,fail.
34  advice('Mobile Computing') :- mc,fail.
35  advice('Collaborative Filtering') :- cf, fail.
36  advice('Big Data Mining in Healthcare') :- bdmh,fail.
37  advice('Artificial Intelligence') :- ai,fail.
38  advice('Deep learning') :- dl,fail.
39  advice('Applied Cryptography') :- ac,fail.
40  advice('Advanced Machine Learning') :- aml,fail.
41  advice('Big Data Analytics') :- bda,fail.
42  advice('Distributed Systems Security') :- dss,fail.
43  advice('Embedded Systems') :- es,fail.
```

```

44 advice('Cellular Data Networks') :- cdn,fail.
45 advice('Network Security') :- ns,fail.
46 advice('Ad Hoc Wireless Networks') :- ahwn,fail.
47 advice('Secure Coding') :- sc,fail.
48 advice('Sorry, No Recommendation !').
49
50
51 preferences([Head|Tail]):- retract(recommend(Head)), preferences(Tail).
52 preferences([]).
53
54 show([Head|Tail]):-
55     format('~n ~w',[Head]),show(Tail).
56
57 show([]).
58
59 isempty([]).
60
61
62 ml :-
63     retract(ai(A)),
64     assert(ai(A)),
65     retract(de(D)),
66     assert(de(D)),
67     ((A == yes ; D == yes)
68      ->true
69      ;fail
70     ),
71     questioninterest('python'),
72     questioninterest('programming'),
73     assert(recommend('Machine Learning')).
74
75 ir :-
76     retract(ai(A)),
77     assert(ai(A)),
78     retract(de(D)),
79     assert(de(D)),
80     ((A == yes ; D == yes)
81      ->true
82      ;fail
83     ),
84     questioninterest('python'),
85     questioninterest('programming'),
86     questioninterest('database'),

```

```

87     questioninterest('advancedprogramming'),
88     assert(recommend('Information Retrieval')).
89
90 dm :-
91     retract(ai(A)),
92     assert(ai(A)),
93     retract(de(D)),
94     assert(de(D)),
95     ((A == yes ; D == yes)
96      ->true
97      ;fail
98     ),
99     questioninterest('python'),
100    questioninterest('programming'),
101    questioninterest('probability'),
102    assert(recommend('Data Mining')).
103
104
105 mc :-
106     retract(mc(M)),
107     assert(mc(M)),
108
109     (M == yes
110      ->true
111      ;fail
112     ),
113     questioninterest('programming'),
114     assert(recommend('Mobile Computing')).
115
116 cf :-
117     retract(de(D)),
118     assert(de(D)),
119     ( D == yes
120      ->true
121      ;fail
122     ),
123     questioninterest('python'),
124     questioninterest('programming'),
125     questioninterest('ml'),
126     assert(recommend('Collaborative Filtering')).
127
128 bdmh :-
129     retract(ai(A)),

```

```
129     retract(ai(A)),
130     assert(ai(A)),
131     retract(de(D)),
132     assert(de(D)),
133     ((A == yes ; D == yes)
134      ->true
135      ;fail
136      ),
137
138     questioninterest('ml'),
139     assert(recommend('Big Data in Healthcare')).
140
141 ai :-
142     retract(ai(A)),
143     assert(ai(A)),
144     (A == yes
145      ->true
146      ;fail
147      ),
148
149     assert(recommend('Artificial Intelligence')).
150
151
152
153
154 dl :-
155     retract(ai(A)),
156     assert(ai(A)),
157     (A == yes
158      ->true
159      ;fail
160      ),
161
162     questioninterest('ml'),
163     assert(recommend('Deep Learning')).
164
165
166 ac :-
167     retract(is(A)),
168     assert(is(A)),
169     (A == yes
170      ->true
171      ;fail
```

```

172         ),
173         questioninterest('discreetmaths'),
174         assert(recommend('Applied Cryptography'))).
175
176 aml :-
177     retract(ai(A)),
178     assert(ai(A)),
179
180     (A == yes
181      ->true
182      ;fail
183     ),
184
185     questioninterest('ml'),
186     assert(recommend('Advanced Machine Learning')).
187
188 bda :-
189     retract(de(D)),
190     assert(de(D)),
191     ( D == yes
192      ->true
193      ;fail
194     ),
195
196     questioninterest('programming'),
197     questioninterest('database'),
198     assert(recommend('Big Data Analytics')).
199
200 dss :-
201     retract(is(A)),
202     assert(is(A)),
203     retract(mc(D)),
204     assert(mc(D)),
205     ((A == yes ; D == yes)
206      ->true
207      ;fail
208     ),
209     questioninterest('cn'),
210     assert(recommend('Distributed System Security')).
211
212 es :-
213     retract(mc(D)),
214     assert(mc(D)),

```

```

214         assert(mc(D)),
215         (D == yes
216         ->true
217         ;fail
218         ),
219         questioninterest('electronics'),
220         assert(recommend('Embedded Systems')).
221
222     cdn :-
223         retract(mc(D)),
224         assert(mc(D)),
225         ( D == yes
226         ->true
227         ;fail
228         ),
229         questioninterest('cn'),
230         assert(recommend('Cellular Data Networks')).
231
232     ns :-
233         retract(is(A)),
234         assert(is(A)),
235         retract(mc(D)),
236         assert(mc(D)),
237         ((A == yes ; D == yes)
238         ->true
239         ;fail
240         ),
241         questioninterest('cn'),
242         assert(recommend('Network Security')).
243
244     ahwn :-
245         retract(mc(D)),
246         assert(mc(D)),
247         (D == yes
248         ->true
249         ;fail
250         ),
251         questioninterest('cn'),
252         assert(recommend('Ad Hoc Wireless Networks')).
253
254
255     sc :-
256         retract(is(A)),

```



```

257         assert(is(A)),
258
259         (A == yes
260          ->true
261          ;fail
262          ),
263         questioninterest('cryptography'),
264         assert(recommend('Secure Coding')).
265
266
267 questioninterest(In) :-
268     (yes(In)
269      ->true
270      ;no(In)
271      ->fail
272      ;ask(In))
273     ).
274
275 ask(Que) :-
276     format('~w ?',[Que]),
277     read(Ans),
278     ( (Ans == yes ; Ans == y)
279      ->assert(yes(Que))
280      ;assert(no(Que)), fail
281      ).
282
283 :- dynamic yes/1,no/1.
284
285
286 clear :- retract(yes(_)),fail.
287 clear :- retract(no(_)),fail.
288 clear.
289
290
291 read_file(Stream,[]) :-
292     at_end_of_stream(Stream).
293 read_file(Stream,[X|L]) :-
294     \+ at_end_of_stream(Stream),
295     read(Stream,X),
296     read_file(Stream,L).

```

## Output:

```
(base) abhinavsaurabh@Abhinavs-MacBook-Pro ~ % swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- ['/Users/abhinavsaurabh/Desktop/IIITD/Abhinav-MT20127/a5.pl'].
true.

?- get_facts.
true .

?- get_advice.

Electives for you are :
  Machine Learning
  Artificial Intelligence
true █
```