PA3-README

#132b

Xinyi Jiang

Title:

CS 132 (Spring 2017) Programming assignment 3

Author:

Xinyi Jiang

Description

Build a simple Boolean Information Retrieval System Based on Wikipedia movie corpus

Dependencies

- nltk
- flask
- json
- shelve
- redis
- nose

Build Instructions

1. `pip install nltk`
2. `pip install nose`
3. An instruction of installing redis: Redis

Run Instruction

1. type `redis-server` in terminal
2. open another terminal window, type `python app.rb`

Modules

db folder:

4 shelve files, storing the inverted index of query, director, starring and location

static folder:

a css file

test folder:

- test_corpus.json: a test corpus with simple data
- unittests.py: a unit test file

templates folder:

4 html files:

- search box view
- search results view
- no search results view
- display a document

index.py:

define an 'Index' class which has 2 field: length and posting list

boolean_query.py:

- open json file, read movie dictionaries
- process string
- create Index instance for each word
- store Index object in shelve

boolean_query

- intersect: do intersection of 2 lists
- get_index: given a list of terms and the path of shelve file, find the lists of posting lists of those terms
- query_search: given the query and keyword, do search in corresponding db file
- final_search: give the 4 values of query(query, dire, star, loc), returns the search results

app.py

- run as the server
- index: search box
- search_results: read query from search box, call query method , get search results, update session, render html page
- display_doc: for request of displaying document: find the data in json file and display

process_text.py

Take a string as input, returns a list of terms and stop words.

Stop words list:

```
set([u'all', u'just', u'being', u'over', u'both', u'through',
u'yourselves', u'its', u'before', u'o', u'hadn', u'herself',
u'll', u'had', u'should', u'to', u'only', u'won', u'under',
u'ours', u'has', u'do', u'them', u'his', u'very', u'they',
u'not', u'during', u'now', u'him', u'nor', u'd', u'did',
u'didn', u'this', u'she', u'each', u'further', u'where',
u'few', u'because', u'doing', u'some', u'hasn', u'are',
u'our', u'ourselves', u'out', u'what', u'for', u'while',
u're', u'does', u'above', u'between', u'mustn', u't', u'be',
u'we', u'who', u'were', u'here', u'shouldn', u'hers', u'by',
u'on', u'about', u'couldn', u'of', u'against', u's', u'isn',
```

```
u'or', u'own', u'into', u'yourself', u'down', u'mightn',
u'wasn', u'your', u'from', u'her', u'their', u'aren',
u'there', u'been', u'whom', u'too', u'wouldn', u'themselves',
u'weren', u'was', u'until', u'more', u'himself', u'that',
u'but', u'don', u'with', u'than', u'those', u'he', u'me',
u'myself', u'ma', u'these', u'up', u'will', u'below', u'ain',
u'can', u'theirs', u'my', u'and', u've', u'then', u'is',
u'am', u'it', u'doesn', u'an', u'as', u'itself', u'at',
u'have', u'in', u'any', u'if', u'again', u'no', u'when',
u'same', u'how', u'other', u'which', u'you', u'shan',
u'needn', u'haven', u'after', u'most', u'such', u'why', u'a',
u'off', u'i', u'm', u'yours', u'so', u'y', u'the', u'having',
u'once'])
```

1. case folding
2. tokenize (`nltk.word_tokenize()`)
3. delete stop words(`stop = set(stopwords.words('english'))`)
4. do stemming (`nltk.stem.snowball.EnglishStemmer`)
5. delete any words that are entirely consisted by punctuation
6. delete "'s", delete stop words again

Testing

Run `nosetests unittests.py` in terminal, and it will show the result

- test_stopword(): test if the stop words returned is correct, examples: "is" is a stop word in query "is apple"
- test_stem(): test if stermmer works, examples: sentences ➜ sentenc, happiness ➜ happi