SENTI_MEND

A recommender utility that provides recommendation of relevent kid's books based on parent's rating and feedback.

By: Raymond Ordona, Amitesh Shukla, Haibin Huang

Installation:

Senti_Mend requires python installation - if possible - on linux environment; although, it should work on other environments (mac, windows) given proper installation.

For python installation, visit this site: http://docs.python-guide.org/en/latest/starting/installation/

For pip installation, visit this site: https://pip.pypa.io/en/stable/installing/

Note: SENTI_MEND has been developed and tested against Python 2.7

Required python packages:

1. Make sure to install the following required python-based packages:

```
nltk (3.2.5)

  $ pip install nltk

twython (3.6.0)

  $ pip install twython

sklearn-pandas (1.6.0)

  $ pip install sklearn-pandas

subprocess32 (3.2.7)

  $ pip install subprocess32

pandas (0.20.3)

  $ pip install pandas

pandas-datareader (0.4.0)

  $ pip install pandas-datareader
```

```
    numpy (1.13.3)

      $ pip install numpy

    scipy (1.0.0)

      $ pip install scipy

    hashlib (20081119)

      $ pip install hashlib

    toml (0.9.3)

      $ pip install toml
```

2. For NLTK, there are extra downloads required.

First, download my_nltk.py and execute: python ./my_nltk.py

This will download sentiwordnet lexicon, wordnet lexicon, stopwords, perceptron tagger.

Example output:

```
$ python ./my_nltk.py
[nltk_data] Downloading collection u'all'
[nltk_data]    |
[nltk_data]    | Downloading package abc to
[nltk_data]    |     /Users/raymondordona/nltk_data...
[nltk_data]    |   Unzipping corpora/abc.zip.
[nltk_data]    | Downloading package alpino to
[nltk_data]    |     /Users/raymondordona/nltk_data...
[nltk_data]    |   Unzipping corpora/alpino.zip.
[nltk_data]    | Downloading package biocreative_ppi to
[nltk_data]    |     /Users/raymondordona/nltk_data...
[nltk_data]    |   Unzipping corpora/biocreative_ppi.zip.
...
nltk_data]    | Downloading package mwa_ppdb to
[nltk_data]    |     /Users/raymondordona/nltk_data...
[nltk_data]    |   Unzipping misc/mwa_ppdb.zip.
[nltk_data]    |
[nltk_data]  Done downloading collection all
```

Note: Running nltk.download('all') may require close to 4GB of disk space.

4. Download senti_mend.py and senti_mend.conf

5. Download Dataset.txt and Final_Feedback.txt

6. Edit senti_mend.conf and update the path to the two datasets.

```
    [dataset]
    book = "./dataset/Dataset.txt"
    rating = "./dataset/Final_Feedback.txt"
    mask_rating = "./dataset/Mask_Final_Feedback.txt"

    [tfidf]
    max_features=3000
    lemmatize_first="True"

    [sentiment]
    algo="vader"  # other choices:  vader,swn
```

Note: Ignore the "mask_rating" parameter for now.

Note: See Sentiment Analysis algorithm section for "lemmatize_First" parameter

Usage:

```
To list books (simulating listing book):

    senti_mend.py -l

    Note: You can derive the <book id> of a book by running  senti_mend.py -l

To display book information:

    senti_mend.py -i -t <book title|book id>

To add book:

    senti_mend.py -a -t <book title> -u <Author> -k <Math|Science|Bed Time> -n <Description>

    where [-k] is book category

To search a book:

    senti_mend.py -s -t <book title>

To check for recommended books based on given title:

    senti_mend.py -c -t <book title|book id> [-d]

    where [-d] is in debug mode

To rate a book (simulating click-throughs and feedback):
    senti_mend.py -r <rate between 1 and 5> -t "<book title|book id>" -f
"<feedback>" -u "<user>"
```

Quick Tutorial:

First, get a list of available books. To do this, run the following command:

```
$ ./senti_mend.py -l

Title          Category
0                          Trace Numbers, Ages 3 - 5 (Big Skills for
Little Hands)                Math
1
7 Ate 9               Math
2                               Numbers: Ages 3-5 (Collins Easy Learning
Preschool)               Math
3
Feast for 10               Math
4                                                              Chicka
Chicka 1, 2, 3              Math
5                            Maths Ages: Ages 4-5 (Collins Easy Learning
Preschool)               Math
6                                                      Sequencing &
Memory Workbook               Math
7                    Math Work Stations: Independent Learning You Can
Count On, K-2               Math
8                                             Common Core Connections
Math, Grade K               Math
9    Young Children's Mathematics: Cognitively Guided Instruction in
Early Childh...              Math
10                        Shapes, Grades PK - K: Gold Star Edition (Home
Workbooks)               Math
11                             What's the Place Value? (Little World Math
Concepts)               Math
12         Shapes, Colours and Patterns: Ages 3-5 (Collins Easy Learning
Preschool)               Math
13                     Numbers Workbook: Ages 3-5 (Collins Easy Learning
Preschool)               Math
...
```

Second, get book information. You can get information of a book by providing the book id or the book title. For example, to get book information using book id 6 for book title 'Sequencing & Memory Workbook', you can issue the following command:

```
$ ./senti_mend.py -i -t 6
```

```
Book Information:

               **Id:** 6
            **Title:** Sequencing & Memory Workbook
           **Author:** by Brighter Child (Compiler),
         **Category:** Math
      **Description:** Carson-Dellosa Publishing (Compiler) Brighter
Child Sequencing & Memory helps young children master thinking skills and
concepts. Practice is included for numbers, patterns, classification,
critical thinking, and more. School success starts here! Workbooks in the
popular Brighter Child series are packed with plenty of fun activities
that teach a variety of essential school skills. Students will find help
for math, English and grammar, handwriting, and other important subject
areas. Each book contains full-color practice pages, easy-to-follow
instructions, and an answer key.
```

or you also can use:

```
$ ./senti_mend.py -i -t "Sequencing & Memory Workbook"
```

Third, try to search for a book title. Use the below command. Below, we are searching for book titles that matches for the **'Seq'** pattern.

```
$ ./senti_mend.py -s -t "Seq"
Book Information:

               **Id:** 6
            **Title:** Sequencing & Memory Workbook
           **Author:** by Brighter Child (Compiler),
         **Category:** Math
      **Description:** Carson-Dellosa Publishing (Compiler) Brighter
Child Sequencing & Memory helps young children master thinking skills and
concepts. Practice is included for numbers, patterns, classification,
critical thinking, and more. School success starts here! Workbooks in the
popular Brighter Child series are packed with plenty of fun activities
that teach a variety of essential school skills. Students will find help
for math, English and grammar, handwriting, and other important subject
areas. Each book contains full-color practice pages, easy-to-follow
instructions, and an answer key.
```

Fourth, check if a book title has already been rated. If a book is rated, a list of recommended books may also be available. To get recommendation for all other books, use the following command:

Below is a book that has not been rated yet.

```
$ ./senti_mend.py -c -t 6

**===============================================================**
                      RECOMMENDATION
**===============================================================**

The sparsity level of Book Reviews is 97.8%

Title:  Sequencing & Memory Workbook

Book has not been rated yet ... No relevant titles to recommend

To rate book:  senti_mend.py -r <rate between 1 and 5> -t "<book
title|book id>" -f "<feedback>" -u "<user>"
```

Here is an example of a book with recommendation:

```
$ ./senti_mend.py -c -t 244

**===============================================================**
                      RECOMMENDATION
**===============================================================**

The sparsity level of Book Reviews is 97.8%

Title:  Greek Myths for Young Children

**Note:** The following books received positive score and positive
feedback from parents
      who also read the book (Greek Myths for Young Children)

positives  score                                   title
     18.0  1.000          Greek Myths for Young Children
      7.0  1.000                        Beginning Sounds
      2.0  1.000  Sensational Seasons: Reproducible Fall
      7.0  0.875                       Same or Different
```

Fifth, To rate a book and give a good review, use the following command:

Here is a book that the user has already rated ...

```
$ ./senti_mend.py -r 5 -u "raymond5" -t "Greek Myths for Young Children"
-f "good book"
```

```
User (d196a91fb80e88) already rated the title (Greek Myths for Young
Children)...
```

And here is a book that another user has not rated yet ...

```
./senti_mend.py -r 5 -u "raymond ordona" -t "Greek Myths for Young
Children" -f "good book"

User review recorded:

      Parent User: f03e434e8b7c5f (Hashed)
       Book Title: Greek Myths for Young Children
             Rate: 5
         Feedback: good book
```

Sixth, To add a book, use the following command:

```
$ ./senti_mend.py -a -t "This is a new  book" -u "IAMAuthor" -n "Everything you
want to see"

New Book Added:

   Book Title: This is a new book
       Author: IAMAuthor
     Category: None
  Description: Everything you want to see
```

Sentiment Analysis Algorithm

The goal is to be able to interprete a comment and determine if it is suggestive of one being a positive feedback, a negative feedback, or neutral.

Required Dataset:

Senti_Mend utility requires two datasets (see senti_mend.conf).

Book Dataset: Dataset.txt

The dataset comes in the form of: <Title>~<Author>~<Category>~<Short Description>

Review Dataset: Final_Feedback.txt

The dataset comes in the form of: <Title>~<Hashed User>~<Rating>~<Review>~<Published Date>~<Annotated Sentiment>

The utility reads both datasets (a.l.a CSV file delimited by a tilde (~)) into a pandas Dataframe for text processing.

The review dataset serves as the training set for the recommender. The recommender algorithm is evaluated based on the annotated sentiment (POS, NEU, NEG) against the computed sentiment score (a score above 0.5 renders to a positive feedback, a score of 0.5 receives a neutral feedback, and a score less than 0.5 receives a negative feedback). This is used to calculate precision, recall, and F1 for evaluation and comparison with other sentiment analysis algos (e.g. swn vs vader). The 300+ reviews are carefully validated and annotated with POS, NEUTRAL, and NEGATIVE labels; thus making this a gold standard feedback.

Senti_Mend, in this version, uses two sentiment tools: swn and vader. Edit sent_mend.conf and choose the proper tool by updating 'algo=' parameter, e.g.

```
[sentiment]
algo="vader"
```

By default, the algorithm is set to "vader".

However, should you try to use "swn", please see below how we implemented pre-processing (e.g. tokenize, stopwords, bigram, lemmatization, stemming, etc.) then sentiment weighing.

First, we rely on the following 'sklearn.feature_extraction.text.TfidfVectorizer' module to help with the following functionalities:

1. Tokenize - split the comments into terms (words)
2. Convert the terms into lowercase
3. Use sublinear-tf scaling in place of just term frequency
4. Use of L2-norm (may not have effect)
5. Use of stopwords
6. Use of IDF and smoothing IDF (may not be required for weights)
7. Use both unigram and bigram
8. Exclude numeric
9. Finally, limit features to maximum of 3000

Second, we rely on 'nltk.pos_tag' module to associate each term with pattern-of-speech tags.

Third, we rely on 'nltk.stem.wordnet.WordNetLemmatizer' module to help with lemmatizing words. Because wordnetlemmatizer requires that each word needs the POS tag, then pos-tagging has to come first.

The term 'loving' cannot be lemmatized without a pos-tag:

```
nltk.stem.WordNetLemmatizer().lemmatize('loving')
'loving'

nltk.stem.WordNetLemmatizer().lemmatize('loving', 'v')
```

```
'love'
```

Note: Edit sent_mend.conf to and set "Lemmatize_first" to "True" if you want to lemmatize terms before taking pos-tags.

Fourth, we rely on 'nltk.corpus.sentiwordnet' module to get the sentiment weight against the given term and pos-tag:

```
senti = swn.senti_synset('happy.a.1')
print(senti.pos_score())
print(senti.neg_score())
print(senti.obj_score())

<happy.a.01: PosScore=0.875 NegScore=0.0>
0.875
0.0
0.125
```

Fifth, for how we computed for the score, please visit this PPT to understand the scoring, computation of weight, selection of related books and finally ranking:

https://github.com/rmordona/myrepo/blob/master/cs410/senti_mend/senti_mend.pptx

Possible PPT version compatibility: TESTED ON PPT version 2013 and 2016

Computing for precision, recall, F1

```
$ ./senti_mend.py -e

Precision-Recall analysis  ...

       ------- SENTIMENT ----------------
S            POS         NEG
C POS        266    |      22
O      ----------------------------------
R NEG         37    |      11
E      ----------------------------------

**Precision:**  0.88
**Recall:**  0.92
**F1-MEASURE:**  0.9
```

Test a comment

```
$ ./senti_mend.py -p "i am not happy"

Evaluating polarity ...

Negative

$ ./senti_mend.py -p "i am happy"

Evaluating polarity ...

Positive
```

Limitation and Challenges

- The sentiment analysis algorithm at the moment does not classify objectivity vs subjectivity and only assumes subjectivity and polarity (positive feedback vs negative feedback).
- POS-tags for words from nltk.stem.wordnet.WordNetLemmatizer may not always match those POS-tags from nltk.corpus.sentiwordnet, e.g. love (n) does not equate to love(v)
- This utility does not utilize DB, cache, or indexing given the small sample dataset used. However, utility can be enhanced to utilize REDIS or other IN-MEMORY DBs for faster access.
- There are other ways to improve the analysis: collocation, intensity, etc. which at this current stage are not included in this utility.

licensing: