sa_extract_m

May 29, 2016

1 Text data feature extraction

Here are parameters of the program user can easily change and estimate their impact on the performance.

```
In [1]: USE_MY_METHOD = True
        USE_STOP_WORDS = True
        USE_EMOTICONS = False
        USE_NEGATION = True

        # If set to false, number of occurances of words is calculated
        USE_BOOLEAN_REPRESENTATION = True

        NUMBER_OF_REVIEWS_TO_ANALYZE = 100000
        NUMBER_OF_POPULAR_WORDS_TO_USE = 1000
```

We will use a dataset consisting of baby product reviews on Amazon.com.

```
In [2]: import pandas as pd
```

```
In [3]: products_raw = pd.read_csv("../valt_sa_data/amazon_baby.csv")
        products = products_raw[['review', 'rating']][0:NUMBER_OF_REVIEWS_TO_ANALYZ
```

Let us see how the data looks like:

```
In [4]: products
```

Out[4]:

	review	rating
0	These flannel wipes are OK, but in my opinion ...	3
1	it came early and was not disappointed. i love...	5
2	Very soft and comfortable and warmer than it l...	5
3	This is a product well worth the purchase. I ...	5
4	All of my kids have cried non-stop when I trie...	5
5	When the Binky Fairy came to our house, we did...	5
6	Lovely book, it's bound tightly so you may not...	4
7	Perfect for new parents. We were able to keep ...	5
8	A friend of mine pinned this product on Pinter...	5
9	This has been an easy way for my nanny to reco...	4

10	I love this journal and our nanny uses it ever...	4
11	This book is perfect! I'm a first time new mo...	5
12	I originally just gave the nanny a pad of pape...	4
13	I thought keeping a simple handwritten journal...	3
14	Space for monthly photos, info and a lot of us...	5
15	I bought this calender for myself for my secon...	4
16	I love this little calender, you can keep trac...	5
17	This was the only calender I could find for th...	5
18	I completed a calendar for my son's first year...	4
19	We wanted to get something to keep track of ou...	5
20	I had a hard time finding a second year calend...	5
21	I only purchased a second-year calendar for my...	2
22	I LOVE this calendar for recording events of m...	5
23	Calendar is exactly as described, but I find t...	3
24	Wife loves this calender. Comes with a lot of ...	5
25	My daughter had her 1st baby over a year ago. ...	5
26	Extremely useful! As a new mom, tired and inex...	5
27	My son loves peek a boo at this age of 9 month...	3
28	One of baby's first and favorite books, and it...	4
29	I like how the book has a hook to attach it to...	5
...	...	...
99970	We had an earlier version of this cup for our ...	2
99971	It's hard to tell from the picture, but the sp...	4
99972	this is a good beaker except for that the spou...	4
99973	High quality step stool. My tot often stands r...	5
99974	I bought this to mainly help my toddler reach ...	5
99975	Fits well. Soft to touch.	5
99976	I love these paci's which are so helpful to my...	1
99977	I recommend this training cup. I have bought M...	5
99978	I just love MAM products. High quality and pri...	5
99979	We love this bottle. Mam bottles are great, a...	5
99980	I bought three of these, in addition to some o...	4
99981	Not as easy to remove as other brands, like Ro...	2
99982	After about a year and some time more, I'm tak...	5
99983	I love this wrap. My husband I both use it da...	5
99984	I bought this for my daughter and she loves it...	5
99985	Love this wrap. Cotton, soft and light. Inexpe...	5
99986	I love this wrap I purchased the large in colo...	5
99987	This is a wonderful wrap to carry baby or todd...	5
99988	Overall, I am pleased with this purchase. I w...	4
99989	I loved this bag! I didn't like the color but ...	4
99990	I spent a surprising amount of time searching ...	5
99991	I have 4 Bumble Bags. The quality is top-notch...	5
99992	ONLY WISH IT HAD A SPACE FOR THE PULL TIGHTENE...	4
99993	Boy car seat covers are expensive so I settled...	5
99994	The Snuzzler is perfect for my baby boy. It ma...	5
99995	This is excellent padding especially for your ...	5
99996	Despite what it says, you may not use this in ...	2

```
99997  Bought this to protect the leather seats in ou...       5
99998  After doing my research online, I found this t...       5
99999  We had a similar product for our first car sea...       5

[100000 rows x 2 columns]
```

Let us explore a specific example of a baby product.

```
In [5]: products.iloc[9]

Out[5]: review    This has been an easy way for my nanny to reco...
        rating                                                    4
        Name: 9, dtype: object
```

Let us define an emoticons extraction function.

```
In [6]: emoticons = [
            ':)', ':))', ':)))', ':(', ':((',
            ':(((', '=)', '=(', '=))', '=(('
        ]

        def extract_emoticons(text):
            emoticons_in_text = []
            for emoticon in emoticons:
                i = text.find(emoticon)
                if i > -1:
                    emoticons_in_text.append(emoticon)
            return emoticons_in_text
```

The helper functions below are also useful.

```
In [7]: punctuation_to_remove = '!"#$%&()*+,-./:;<=>?@[\]^_`{|}~'

        def remove_punctuation(text):
            return text.translate(None, punctuation_to_remove)

        pos_dict = {
            'NN': 'n', 'VB': 'v', 'VBD': 'v', 'VBG': 'v',
            'VBN': 'v', 'VBP': 'v', 'VBZ': 'v',
            'JJ': 'a', 'JJR': 'a', 'JJS': 'a', 'JJT': 'a'
        }

        def get_pos_for_lemmatirzer(brown_post):
            if not brown_post in pos_dict:
                return 'n'
            else:
                return pos_dict[brown_post]
```

Now let us define a more sophisticated function for review analysis.
First the punctuation is removed.
Then every word is pos tagged to prepare for lemmatization.
After that lemmatization is performed to find the root form of each word.
All the stop words are removed if the corresponding program parameter is set. Also if set, emoticons are extracted and processed.

```
In [8]: import nltk
        from nltk.stem import WordNetLemmatizer

        stop_words = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves',
         'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his',
         'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself',
         'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which',
         'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are',
         'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having',
         'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or',
         'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about
         'against', 'between', 'into', 'through', 'during', 'before', 'after', 'abo
         'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'un
         'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why
         'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'suc
         'nor', 'not', "n't", 'only', 'own', 'same', 'so', 'than', 'too', 'very', '
         'will', 'just', 'don', 'should', 'now']

        def analyze_review(text):
            if USE_EMOTICONS:
                emoticons_features = extract_emoticons(text)
            else:
                emoticons_features = []

            text_without_punctuation = remove_punctuation(text)
            tokens = nltk.word_tokenize(text_without_punctuation)
            tagged_tokens = nltk.pos_tag(tokens)
            tokens_prepared_for_lemmatization = [(t[0], get_pos_for_lemmatirzer(t[1

            lemmatizer = WordNetLemmatizer()
            lemmas = []

            not_count = 0
            words_after_not_count = 0
            for tpl in tokens_prepared_for_lemmatization:
                current_word = lemmatizer.lemmatize(tpl[0], tpl[1]).lower()
                if words_after_not_count > 2:
                    not_count = 0
                    words_after_not_count = 0
                if current_word == 'not' or current_word == "n't":
                        not_count += 1
```

```
            elif (not USE_STOP_WORDS) or (not current_word in stop_words):
                if USE_NEGATION and not_count % 2 == 1:
                    current_word = 'NOT_' + current_word
                    words_after_not_count += 1
                lemmas.append('F_' + current_word) # F - meaning feature

        review_words = lemmas + emoticons_features
        return review_words
```

Now, we will perform text analysis. We will also find and print most common words and total number of words in the dictionary.

```
In [9]: analyzed_reviews = products['review'].apply(str).apply(analyze_review)

        review_words_list = [] # conaints duplicates, so that count of each word ca
        review_dictionary = set()

        for w_l in analyzed_reviews:
            for word in w_l:
                review_words_list.append(word)
                review_dictionary.add(word)

        from collections import Counter

        review_counter = Counter(review_words_list)
        most_common_words = map(lambda x: x[0], review_counter.most_common(NUMBER_C
        print most_common_words
        print len(review_dictionary)
```

```
["F_'s", 'F_use', 'F_baby', 'F_one', 'F_get', 'F_love', 'F_would', 'F_great', 'F_li
108322
```

We perform feature extraction on the analyzed text. The matrix for machine learning is formed. Only features stored in the variable `significant_words` are included.

```
In [10]: if USE_MY_METHOD:
             if USE_EMOTICONS:
                 significant_words = most_common_words + emoticons
             else:
                 significant_words = most_common_words
         else:
             significant_words = ['love', 'great', 'easy', 'old', 'little', 'perfec
                                  'well', 'able', 'car', 'broke', 'less', 'even', '
                                  'work', 'product', 'money', 'would', 'return']

         def count_number_of_significant_words(text):
             words = text['review']
             word_dict = {}
```

```
        for word in significant_words:
            word_dict[word] = 0
        for word in words:
            if word in significant_words:
                if not word in word_dict:
                    word_dict[word] = 1
                else:
                    if USE_BOOLEAN_REPRESENTATION:
                        word_dict[word] = 1
                    else:
                        word_dict[word] = word_dict[word] + 1
        significant_words_counts = []
        for word in significant_words:
            significant_words_counts.append(word_dict[word])
        return pd.Series(significant_words_counts, index=significant_words)

    word_counts_df = pd.DataFrame(analyzed_reviews).apply(count_number_of_sign
    word_counts_df.columns = significant_words

    products_with_words = products.join(word_counts_df)
```

Now, let us explore what the sample looks like after all the transformations.
The resulting matrix is very sparse, as was expected.

```
In [11]: products_with_words.iloc[9]
```

```
Out[11]: review          This has been an easy way for my nanny to reco...
         rating                                                          4
         F_'s                                                            0
         F_use                                                           0
         F_baby                                                          1
         F_one                                                           1
         F_get                                                           0
         F_love                                                          0
         F_would                                                         0
         F_great                                                         0
         F_like                                                          0
         F_buy                                                           0
         F_old                                                           0
         F_month                                                         0
         F_easy                                                          1
         F_seat                                                          0
         F_time                                                          0
         F_make                                                          0
         F_little                                                        0
         F_product                                                       0
         F_son                                                           0
         F_good                                                          0
```

```
F_go                                                    0
F_work                                                  0
F_well                                                  0
F_also                                                  0
F_really                                                0
F_put                                                   0
F_daughter                                              0
F_look                                                  0
                                                     ...
F_separate                                              0
F_upright                                               0
F_texture                                               0
F_pregnancy                                             0
F_length                                                0
F_rid                                                   0
F_careful                                               0
F_11                                                    0
F_terrible                                              0
F_laundry                                               0
F_create                                                0
F_NOT_'m                                                0
F_eventually                                            0
F_bounce                                                0
F_burp                                                  0
F_name                                                  0
F_condition                                             0
F_disappoint                                            0
F_NOT_first                                             0
F_contain                                               0
F_dark                                                  0
F_accident                                              0
F_attachment                                            0
F_temperature                                           0
F_alot                                                  0
F_tire                                                  0
F_NOT_small                                             0
F_gas                                                   0
F_theme                                                 0
F_alone                                                 0
Name: 9, dtype: object
```

1.1 Save prepared data into a file

```
In [12]: X = products_with_words[significant_words]
         y = products_with_words['rating']
         X.to_csv('../valt_sa_data/x_m.csv', index=False)
         y.to_csv('../valt_sa_data/y_m.csv', index=False)
```