CSC401: Natural Language Processing

Tutorial: Assignment 1

Jan 20, 2017
TA: Aryan Arbabi
arbabi@cs.toronto.edu

(Slides adapted from Stefania Raimondo, Erin Grant, Siavash Kazemian, Varada Kolhatkar and Ka-Chun Won)
Goal

Perform sentiment analysis on individual tweets:
Binary classification of tweets as having positive or negative sentiment

Input

“I love my BrandNameProduct!”

“Never shop at X-Store. Total garbage.”

Output

Positive

Negative
Methodology
Tweet Corpus

- /u/cs401/A1/tweets
  - 1,600,000 training tweets - training.1600000.processed.noemoticon.csv
  - 359 testing tweets - testdata.manualSUBSET.2009.06.14.csv

- Format: 1 tweet/line in .csv

4, 2087, Sat May 16 23:58:44 UTC 2009, friends, bffever, Best @friendster ever! #omg

Tweet Id
Date of Tweet

Polarity: 0 (negative), 4 (positive)

User
Tweet text
Query (or NO_QUERY)
Tweet Corpus

• Use a subset of 20,000 tweets from the training file

\[
\begin{align*}
\text{ID} \times 10,000 + [0 \ldots 9999] \\
800,000 + \text{ID} \times 10,000 + [0 \ldots 9999]
\end{align*}
\]

• Where ID is your student ID module 80
Preprocessing: 3 steps

1. “Pre-processing” aka cleaning tweets
2. Tokenizing
3. Tagging

• There are 9 tasks
  • Implement one function for each (except 6)
  • Each function takes a string and outputs the modified string
  • Name the functions **twtt1** to **twtt9**
Table 2: Conversion from raw tweets to tagged tweets

Raw tweet:


Output from twtt.py:

...<A=4>
Meet/VB me/PRP today/NN at/IN the/DT FEC/NN in/IN DC/NN at/IN 4/NN ./.
Wear/VB a/DT carnation/NN so/RB I/PRP know/VB it/PRP ’s/POS you/PRP ./.
<A=0>
...
Preprocessing: Detailed Steps

• Remove HTML tags/attributes/characters
• Remove URLs
• Twitter # and @ symbol removal
• Sentence boundary identification
• Tokenize
• POS Tag
• Delimit Tweets
Removing HTML/URLs

• Regex is your friend!

• For fixed patterns, you can use string replace
  • Ex. mystring.replace("&amp;","&")
  • Note: strings are immutable

• For variable patterns, you’ll need regular expressions
  • Ex. For html start tags (e.g., <html>, <ahref=“google.com”>) use re.sub
  • Note: re is greedy! (so ‘<.+>’ isn’t good enough)
Sentence Boundaries: Hard

• Sentences end with ‘.’, ‘?’ , or ‘!’

• But not all periods are EOS (e.g. abbreviations)
  e.g., How much does the U.S. president get paid?

• But some abbreviations are EOS
  e.g., After the UK tour ends next week, he returns to the U.S.

• Possible solution: consider checking if the following letter is lowercase
  But what about: e.g., After U.S. Attorney General…

• List of common abbreviations:
  • /u/cs401/Wordlists/abbrev.english
Sentence Boundaries: Hard (con’t)

• Don’t break multiple times for multiple punctuation (e.g. !!!)

• But not all ellipsis are EOS
  e.g., I dunno Manny… do you want to go?

• Quotations: after the punctuation, but part of the sentence
  e.g., “You remind me,” she remarked, “of your mother.”

• There is no perfect sentence parser!

• See Manning and Schütze, Section 4.2.4 for some good ideas
Tokenization: Splitting sentences into tokens

• Simple words: Use line.strip().split()
  e.g., ‘an apple’ → [‘an’, ‘apple’]

• Punctuation should be it’s own token
  e.g., ‘she said,’ → [‘she’, ‘said’, ‘,’]

• But not always…
  e.g., ‘paid $10,000’ → [‘paid’, ‘$', ‘10,000’]

• Including clitics and contractions
  e.g., “can’t” → [“ca”, “n’t”]
Tokenization (con’t)

• Possessives
  e.g., “she’s” → [“she”, “’s”]

• Compounds (your choice)
  e.g., time-consuming

• Don’t break up ellipsis...
POS Tagging

• Use the module we’ve provided: import NLPlib

• Only load the tagger once!
  tagger = NLPlib.NLPlib()

• Pass a list of tokens to the tag method:
  tags = tagger.tag([‘the’, ‘boy’])
  Returns [‘DT’, ‘NN’]

• Do not tag empty strings
Tag list (see handout)

<table>
<thead>
<tr>
<th>Tag</th>
<th>Name</th>
<th>Example</th>
<th>POS</th>
<th>Personal pronoun</th>
<th>PRP $</th>
<th>Possessive pronoun</th>
<th>PRP</th>
<th>Possessive ending</th>
<th>'s, '</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>Coordinating conjunction</td>
<td>and three the there [is] d‘oeuvre in, of, like</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD</td>
<td>Cardinal number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td>Determiner</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EX</td>
<td>Existential there</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FW</td>
<td>Foreign word</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IN</td>
<td>Preposition or subordinating conjunction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JJ</td>
<td>Adjective</td>
<td>green, good greener, better greenest, best (1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JJR</td>
<td>Adjective, comparative</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JJS</td>
<td>Adjective, superlative</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS</td>
<td>List item marker</td>
<td>table tables</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MD</td>
<td>Modal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN</td>
<td>Noun, singular or mass</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNS</td>
<td>Noun, plural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNP</td>
<td>Proper noun, singular</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNPS</td>
<td>Proper noun, plural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDT</td>
<td>Predeterminer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TO</td>
<td>to</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UH</td>
<td>Interjection</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VB</td>
<td>Verb, base form</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBD</td>
<td>Verb, past tense</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBG</td>
<td>Verb, gerund or present participle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBN</td>
<td>Verb, past participle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBP</td>
<td>Verb, non-3rd-person singular present</td>
<td>take</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBZ</td>
<td>Verb, 3rd-person singular present</td>
<td>takes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WDT</td>
<td>wh-determiner</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WP</td>
<td>wh-pronoun</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WP$</td>
<td>Possessive wh-pronoun</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRB</td>
<td>wh-adverb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tag list (see handout)

<table>
<thead>
<tr>
<th>Tag</th>
<th>Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>Pound sign</td>
<td>£</td>
</tr>
<tr>
<td>$</td>
<td>Dollar sign</td>
<td>$</td>
</tr>
<tr>
<td>.</td>
<td>Sentence-final punctuation</td>
<td>!, ?, .</td>
</tr>
<tr>
<td>,</td>
<td>Comma</td>
<td></td>
</tr>
<tr>
<td>:</td>
<td>Colon, semi-colon, ellipsis</td>
<td></td>
</tr>
<tr>
<td>(</td>
<td>Left bracket character</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td>Right bracket character</td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>Straight double quote</td>
<td></td>
</tr>
<tr>
<td>'</td>
<td>Left open single quote</td>
<td></td>
</tr>
<tr>
<td>“</td>
<td>Left open double quote</td>
<td></td>
</tr>
<tr>
<td>'</td>
<td>Right close single quote</td>
<td></td>
</tr>
<tr>
<td>”</td>
<td>Right close double quote</td>
<td></td>
</tr>
</tbody>
</table>
Delimit tweets

• Output file: (*.twt)...
• Space between tokens (" ".join(tokens))
• Each line is a sentence, not a tweet ("\n".join(sents))
• Each tweet is separated "<A=#>" on a separate line

• If a tweet is empty (e.g. only url), include the empty tweet!
  • Your feature extractor must handle this condition
Example .twt file

Hindsight/NN ./.
Yeah/UH ,/, that/IN was/VBD probably/RB a/DT poorly/RB worded/VBN tweet/NN ./.
Pick/VB up/IN the/DT jacket/NN .../:
Tips

• Sanity check often
• Peek at the tweets
• Use your best judgement
  • Check out how these tools handle specific cases:
    • https://code.google.com/p/splitta/
    • http://nlp.stanford.edu/software/tokenizer.shtml

• Finish Part 1 ASAP!
  • Get it working. Don’t worry about perfecting it. There’s no such thing as a perfect parser.