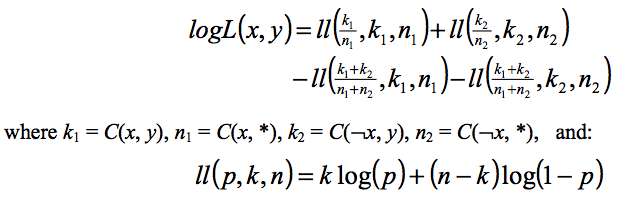
Tasks, somewhat prioritized

* Evaluation
  + Setup to run Xfold from cmd line rather than in pycharm
  + Try other classifiers: SVM, RF, NB
* aspect term model improvements
  + do some error analysis
    - features of FPs and FNs, stuck into a dictionary?
  + add a feature that indicates whether the word is one of the category words (food, etc.), and later extend to looking for its wordnet synset; could also, if parsing indicate whether the current word is in the same clause as a category word
  + SVM
    - Need to create binary classifiers
      * One vs. all
      * Pairwise (K X (K-1))/2 classifiers
      * and use weighted voting
    - Different chunk representations
    - See this paper for more: <http://acl.ldc.upenn.edu/N/N01/N01-1025.pdf?origin=publication_detail>
  + Can we use a heuristic such as has there been a punctuation “and” “or” or “but” between the current term and the prior term?
  + HMM
    - With specialized tags similar to <http://jmlr.org/papers/volume2/molina02a/molina02a.pdf>
* Aspect term: add more features
  + expand n-gram window
  + character n-grams (could work for sentiment too)
  + add the more complicated sentiment dictionary
* (C) sentiment task
  + measure acc of baseline sentiment classifier for entire sentence – does it align with majority label of aspect terms?
  + extend handling of negation
    - create “words” out of the negation concatenated with previous & next words (not sure what I meant here!)
    - get (from parser) the word it modifies
  + closest sentiment indicator word and/or its polarity (really want the one closest in a parse) – could chunks be good enough instead of full dep. P\parse?
  + feed sentiment classification back into aspect term extraction – if there is only neutral text then there is no aspect term
  + perhaps divide into two: subjective vs objective then pos/neg/mixed
* (C) figure out how to best generate figures and tables for inclusion in a paper
* (C) add ability to turn on & off different subsets of the features easily
  + create new dictionaries by copying then removing some key
  + wrapper around classifier (fit/train & predict/classify) that only uses the proper keys in the passed example
* If get to the category tasks: one classifier per topic/category and mixture of experts approach
* Bugs in xml reader:
  + quotes
  + Also doesn’t work if the word ends with a dash

Some other ideas

* could also use a NP chunker as additional evidence for aspect term extraction
* are verbs ever extracted? – could print out if we find one
* what is apply\_features – some kind of sparse data coding?
* add the actual sentiment indicator words (those from the lexicon) as features for aspect term id, but explore different ways to do it
  + closest
  + all
  + if in same (hi-level) NP
  + the actual word, its polarity, or just the presence of such a word
* use the classifier from the other domain as additional evidence (for all the tasks but starting with aspect term extraction)
* a feature that indicates we’ve seen the word but it’s never been in an aspect term (or the proportion of the time it’s been in an aspect term)
* How to incorporate PMI & log-likelihood into decision of whether to “combine” two words as part of an aspect phrase? (this paper suggest using a minimum for PMI as a threshold then log likelihood test: (where c(x,y) is count and \* is wildcard) <http://webdocs.cs.ualberta.ca/~lindek/papers/ai01.pdf>



* sentiment for the aspect terms
  + how well is the brute force sentiment classifier I wrote doing? Does it agree with the majority of labels? Are the neutral labels corresponding to no aspect terms? How well can we trust that lexicon?
    - Learn other indicators that flip the dominant sentiment (like one aspect term following another has a flipped label)
  + as part of it, should really figure out which sentiment-laden words (like “sluggish” modify which aspect terms (like driver, in the phrase sluggish driver)
  + types of patterns
    - the X was Y (steak was delicious)
    - Y X (poor service)
  + How to use sentences for which there was no aspect term & thus no sentiment? Words used in a neutral setting, perhaps?
  + PMI between sentiment words and heads of aspect phrases?
  + Dependency parse closest words
    - See spring 14 plan file for notes on dep parsing
  + Extending lexicon: words that occur in similar contexts to base words (but it’s a pretty large lexicon already – not sure in our small corpus if we’d find much, but maybe so since it’s domain specific)
* Features for twitter task
  + all caps
  + initial cap
  + hashtag (not for aspect task)
  + stemmed vs not
  + stopwords vs not
  + lowercase vs not
  + emoticons as a single feature
  + other normalization (numbers, hashtags)
  + whether a RT or not
  + removing URLs (but keeping them aside for later getting the text in them)
  + replacing @<uname> with just “@”
* What helps in WSD?
  + Character ngrams?

Dependency parse notes

* find the sentiment term that is closest to a given term

POS tagging notes:

* the first version of the code counts (create\_one\_withPOS) the character indexes incorrectly when punctuation is involved (for the POS tag case, may also be true for non-pos case, not sure)
  + The “to” position in the xml is the position after the EOW, with punctuation stripped, if any

Sentiment lexicon notes:

* to use is Bing Liu’s lexicon and Wilson etal’s lexicon
* B.Liu’s is just word list for pos & neg
* Wilson etals associates words with pos tags and stemmed versions of the words, so to use that well need stemming
* Not sure yet how to use s.lex. entries for evidence in aspect term extraction, but presence within a window is a first reasonable cut, and of course it should be used for the polarity tagging task

**To get started in interpreter:**

sys.path.extend(['/users/cindi/semeval'])

import semevalTask4

**results**

* Hmm R/P on 500 sentence subset of laptop:
  + with just pos tags as features: .03 / .5
  + with wd+pos: .08/.75
  + with stemmedWd+pos: .16 / .86
  + just stemmedWd, no POS: .32, .86
* (3/16) our version of P/R/F with maxent and these features: 'word': word, 'pos': pos, 'sentiment': sentiment, 'prevpos': prevpos, 'prevtag': prevtag, 'prev\_sentiment': prev\_sentiment,

'nextpos': nextpos, 'next\_sentiment': next\_sentiment}

restaurant\_trial: ave Prec: 0.82, Rec: 0.74, F1: 0.78

laptop trial: ave Prec: 0.72, Rec: 0.62, F1: 0.66 (more Unicode issues there?)

3/19, 'word': word, 'pos': pos, 'sentiment': sentiment, 'obj': objectivity,

'prevw': prevw, 'prevpos': prevpos, 'prevtag': prevtag, 'prev\_sentiment': prev\_sentiment, 'prev\_obj': prev\_obj,

'nextw': nextw, 'nextpos': nextpos, 'next\_sentiment': next\_sentiment, 'next\_obj': next\_obj}

After fixing Unicode issues, above features, laptop: ave Prec: 0.77, Rec: 0.53, F1: 0.63

Restaurant: ave Prec: 0.82, Rec: 0.69, F1: 0.75

Now running with turning on stemming:

ave Prec: 0.82, Rec: 0.69, F1: 0.75 (no difference but I believe it kicked in)

Before Unicode fix:

* testing on the training set for a 1000-example subset of laptop, gets .9/.85/.87 P/R/F1 results for the baseline of 3/19
* same subset, Xval: ave Prec: 0.78, Rec: 0.41, F1: 0.54
* adding unk handling to above: (running @ noon Wed): ave Prec: 0.70, Rec: 0.38, F1: 0.49; Okay, so much for that!
* The semeval\_base gets P = 0.401709 -- R = 0.381339 -- F1 = 0.391259 (#correct: 188, #retrieved: 468, #relevant: 493) on laptops & P = 0.539118 -- R = 0.514247 -- F1 = 0.526389 (#correct: 379, #retrieved: 703, #relevant: 737) on restaurants

**Results for subtask 2**

* 3/9, initial classifier = maxent iis, features as below, accuracy = 64%!
  + Overall sentence positive & negative sentiment score
  + Number of aspect terms
  + POS, word, iob, sentiment of word preceding & following aspect phrase
  + Word, pos & iob of head of aspect phrase