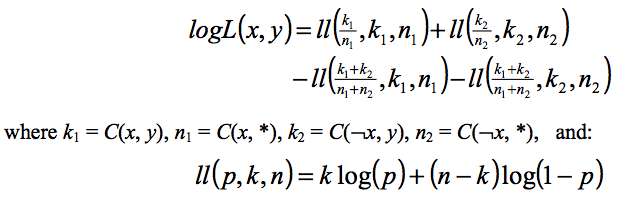
Tasks, somewhat prioritized

* Evaluation
  + Incorporate XValidation machinery and look at the training sets instead of trial sets for both domains
  + Try other classifiers: SVM, RF, NB
* Aspect term: add more features
  + prev & next word weren’t actually added to dictionary – I did this? Or Hao to do (I did in subtask 2 code)
  + expand n-gram window
  + character n-grams (could work for sentiment too)
  + add the more complicated sentiment dictionary
* aspect term model improvements
  + explicitly add unk handling &/or only use most common words as features; the clfs don’t do anything exciting here; see ch 5 of nltk book and search for unk
  + 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>
  + HMM
    - With specialized tags similar to <http://jmlr.org/papers/volume2/molina02a/molina02a.pdf>
* (C) start on the sentiment task over spring break
  + 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
* Bug in xml reader:
* I recommend the garlic shrimp, okra (bindi), and anything with lamb. – okra bindi should be a term
* [('I', 'PRP', 'O'), ('recommend', 'VBP', 'O'), ('the', 'DT', 'O'), ('garlic', 'JJ', 'B-Aspect'), ('shrimp', 'NN', 'I-Aspect'), (',', ',', 'O'), ('okra', 'NN', 'O'), ('(', ':', 'O'), ('bindi', 'NN', 'O'), (')', ':', 'O'), (',', ',', 'O'), ('and', 'CC', 'O'), ('anything', 'NN', 'O'), ('with', 'IN', 'O'), ('lamb', 'NN', 'B-Aspect'), ('.', '.', 'O')]
* A few tips: skip the turnip cake, roast pork buns and egg custards. – roast pork buns term
* [('A', 'DT', 'O'), ('few', 'JJ', 'O'), ('tips', 'NNS', 'O'), (':', ':', 'O'), ('skip', 'NN', 'O'), ('the', 'DT', 'O'), ('turnip', 'NN', 'B-Aspect'), ('cake', 'NN', 'I-Aspect'), (',', ',', 'O'), ('roast', 'JJ', 'O'), ('pork', 'NN', 'O'), ('buns', 'NNS', 'O'), ('and', 'CC', 'O'), ('egg', 'NN', 'B-Aspect'), ('custards', 'NNS', 'I-Aspect'), ('.', '.', 'O')]
* The dishes offered were unique, very tasty and fresh from the lamb sausages, sardines with biscuits, large whole shrimp to the amazing pistachio ice cream (the best and freshest I've ever had). – sardines w/ biscuits
* [('The', 'DT', 'O'), ('dishes', 'NNS', 'B-Aspect'), ('offered', 'VBD', 'O'), ('were', 'VBD', 'O'), ('unique', 'JJ', 'O'), (',', ',', 'O'), ('very', 'RB', 'O'), ('tasty', 'JJ', 'O'), ('and', 'CC', 'O'), ('fresh', 'JJ', 'O'), ('from', 'IN', 'O'), ('the', 'DT', 'O'), ('lamb', 'NN', 'B-Aspect'), ('sausages', 'NNS', 'I-Aspect'), (',', ',', 'O'), ('sardines', 'NNS', 'O'), ('with', 'IN', 'O'), ('biscuits', 'NNS', 'O'), (',', ',', 'O'), ('large', 'JJ', 'O'), ('whole', 'NN', 'O'), ('shrimp', 'NN', 'B-Aspect'), ('to', 'TO', 'O'), ('the', 'DT', 'O'), ('amazing', 'NN', 'O'), ('pistachio', 'NN', 'B-Aspect'), ('ice', 'NN', 'I-Aspect'), ('cream', 'NN', 'I-Aspect'), ('(', ':', 'O'), ('the', 'DT', 'O'), ('best', 'JJS', 'O'), ('and', 'CC', 'O'), ('freshest', 'JJS', 'O'), ('I', 'PRP', 'O'), ("'ve", 'VBP', 'O'), ('ever', 'RB', 'O'), ('had', 'VBD', 'O'), (')', 'CD', 'O'), ('.', '.', 'O')]
* The tech guy then said the service center does not do 1-to-1 exchange and I have to direct my concern to the "sales" team, which is the retail shop which I bought my netbook from. – sales
* [('The', 'DT', 'O'), ('tech', 'NN', 'B-Aspect'), ('guy', 'NN', 'I-Aspect'), ('then', 'RB', 'O'),
* ('said', 'VBD', 'O'), ('the', 'DT', 'O'), ('service', 'NN', 'B-Aspect'), ('center', 'NN', 'I-Aspect'), ('does', 'VBZ', 'O'), ('not', 'RB', 'O'), ('do', 'VB', 'O'), ('1-to-1', 'CD', 'O'), ('exchange', 'NN', 'O'), ('and', 'CC', 'O'), ('I', 'PRP', 'O'), ('have', 'VBP', 'O'), ('to', 'TO', 'O'), ('direct', 'VB', 'O'), ('my', 'PRP$', 'O'), ('concern', 'NN', 'O'), ('to', 'TO', 'O'), ('the', 'DT', 'O'), ('``', '``', 'O'), ('sales', 'NNS', 'O'), ("''", "''", 'O'), ('team', 'NN', 'O'), (',', ',', 'O'), ('which', 'WDT', 'O'), ('is', 'VBZ', 'O'), ('the', 'DT', 'O'), ('retail', 'JJ', 'O'), ('shop', 'NN', 'O'), ('which', 'WDT', 'O'), ('I', 'PRP', 'O'), ('bought', 'VBD', 'O'), ('my', 'PRP$', 'O'), ('netbook', 'NN', 'O'), ('from', 'IN', 'O'), ('.', '.', 'O')]
* Also doesn’t work when the sentence starts with spaces or (I think) if the word ends with a dash

Some other ideas

* could also use a NP chunker as additional evidence for aspect term extraction
* really do sequence modeling – seems that the head is stronger evidence than earlier words in the phrase
* 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?

POS tagging notes:

* The create\_exs method now assumes each aspect token appears only once in a given sentence, rather than worrying about which occurrence of the token actually is the aspect term if there are multiple ones. See next bullet for problems faced with dealing with both POS tagging and punctuation in finding the aspect terms (doable but messy!)
* 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
  + Could use a dictionary with start index as keys, (word, BIO-tag) as values, and include punctuation in that? Then iterate through range(len(sentence) and see if those are in the dict keys. But how to get that dictionary
    - Assume punctuation always attached to a word? So don’t add 1 in that case, just add len of punct? Or see if the punctuation is in the split sentence at the appropriate position.
    - Or iterate through the original characters. Can tokenize give us access to the original positions?

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

**Try train and test on the trial data first to see how it works. (don’t trust these scores anymore as of 3/11)**

ChunkParse score on restaurants-trial 80/20 split with no POS tagging (~2/23)

IOB Accuracy: 89.0%

Precision: 75.0%

Recall: 24.0%

F-Measure: 36.4%

Added POS tags, 3/1: Slightly hurts precision but awesome (in comparison to prev) on recall – makes sense, can incorporate syntax info

IOB Accuracy: 91.7%

Precision: 73.3%

Recall: 44.0%

F-Measure: 55.0%

Adding sentiment dictionary: helps precision quite a bit while hurting recall and F-measure:

IOB Accuracy: 91.4%

Precision: 83.3%

Recall: 40.0%

F-Measure: 54.1%

HMM tagger, basic using both w/pos (crazy, but…), gets:

IOB Accuracy: 90.7%

Precision: 100.0%

Recall: 16.7%

F-Measure: 28.6%

Took out words:

IOB Accuracy: 86.6%

Precision: 31.6%

Recall: 25.0%

F-Measure: 27.9%

**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