NLP Final Project
Fall 2015, Due Friday, December 18

For the final project, everyone is required to do some sentiment classification and then choose one of the other three types of projects: annotation, classification experiments or implementation. You may also propose your own project and you may work in groups of 2-3 people. For each person that is added to the group, plan on increasing the tasks by about 50%.

Required Part 0. Sentiment Classification

For this part, you are to do at least one experiment in the NLTK on the movie review data in which you compare the baseline performance of using just word features with some modified or additional features that you implement.

You will need to write a Python feature function to generate the features that you choose. One idea is to vary the representation of the subjectivity lexicon features or to use positive and negative emotion words from LIWC. Another idea would be to extend the representation of the negation features up to the next punctuation. Another idea would be to run the POS tagger on the text and include as features the counts of the POS tags of type verb (tag starts with V), noun (starts with N), adjective (starts with J) and adverb (starts with R).

Note that you will first want to develop your feature function until it works. Then to do the experiment, define a random training and test set; define the regular word features, run the classifier and get accuracy; and then define your new features, run the classifier and get accuracy. The important thing is the two classifier runs are made on the same training and test sets so that you can compare the accuracy. Note that two runs are the absolute minimum, and it is preferred that you make a series of related feature comparison runs.

Include a description of your experiments in the final project report, together with the feature function code that you wrote and the accuracies of the two classifications.

Now choose one of the following 3 options.

Option I. Annotation and Analysis of Data

For this task, you will annotate data in a corpus, compare your annotations with others in inter-annotator agreement, and design features for classification of the task.

The data will be twitter data obtained from the Governor's Campaign project headed by Jenny Stromer-Galley, Jeff Hemsley and Bryan Semaan. The tweets are those that are posted by the campaigns of candidates for governor in the various states in 2014. There is already a codebook developed that describes how to label each tweet as to one of 5 labels derived from communication theory as to how the candidates are communicating with the public.

If you choose this project, you will be expected to carry out three rounds of annotation. For the first two rounds, you will do trial annotation and then meet in a group with me to discuss your level of agreement with the other annotators. So this will require that we set up two group meeting times after the Thanksgiving break.

To complete this project, you will write a reflection report that describes what you did. First, write a reflection on the annotation process. Include in your report examples of tweets and labels that were straightforward and easy to decide, and also examples of those that were either hard for you to decide or hard for the group to come to agreement. Include the inter-annotator agreements for your group over the 2 week process.

For the last part of the reflection, look at some of the clue words or phrases and design a set of features that would be good to characterize the tweets for a ML classifier to label the tweets with the communication label. You may choose also to implement the features in a classifier, along with words and bigrams, in which case, this classifier can replace your required part 0.

Option 2. Processing and Classification of Sentiment or other Data

For this task, you should choose to work on classification of one of three datasets: the email spam data, the Kaggle movie review data, or the SemEval Twitter data. If you do enough experiments on this task, then you do not also have to do the required part 0 of the Final Project, as that will be replaced by the extra experiments that you do in this option.

Three Datasets for Text Classification Tasks

For this option, you should choose one of these three datasets to focus on (unless you propose your own).

1. Detection of SPAM in email

The first dataset is one produced for detecting Spam emails from the Enron public email corpus. In addition to some small numbers of Spam already in the corpus, additional spam emails were introduced into each user's email stream in order to have a sufficient number of spam examples to train a classifier. The non-Spam emails are labeled "ham". (See this paper for details: http://www.aueb.gr/users/ion/docs/ceas2006_paper.pdf) The dataset that we have was gleaned from their web site at http://www.aueb.gr/users/ion/data/enron-spam/.

Although there are 3 large directories of both Spam and Ham emails, only the first one is used here with 3,672 regular emails in the "ham" folder, and 1,500 emails in the "spam" folder.

2. Kaggle competition movie review phrase data, labeled for sentiment

This dataset was produced for the Kaggle competition, described here: https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews , and which uses data from the sentiment analysis by Socher et al, detailed at this web site: http://nlp.stanford.edu/sentiment/. The data was taken from the original Pang and Lee movie review corpus based on reviews from

the Rotten Tomatoes web site. Socher's group used crowd-sourcing to manually annotate all the subphrases of sentences with a sentiment label ranging over: "negative", "somewhat negative", "neutral", "somewhat positive", "positive".

Although the actual Kaggle competition is over, the data is still available at https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews/data. We are going to use the training data "train.tsv", and some test data is also available "test.tsv". There appear to be 156,060 phrases in the training data file.

3. SemEval shared task Twitter data, labeled for sentiment

The Semantic Evaluation conference, SemEval, has recently added sentiment detection in Twitter messages and other social media genres to its shared tasks. For example, in 2014, it ran as Task 9, sub-task B, Message Polarity Classification: http://alt.qcri.org/semeval2014/task9/. The data was manually labeled and each dataset is available as a Twitter id paired with the manual label. There are 5 labels used for each message: "positive", "negative", "objective", "neutral", "objective OR neutral".

To actually get the tweets, there is a script you can run to get them from Twitter itself. If a Twitter user has retracted their tweet, then Twitter will no longer send it out and it is marked as "Not Available".

The dataset given here was collected in the Spring 2014 from Twitter. You are given access to this data through Blackboard, and it should not be made publically available.

The Text Classification Tasks

Step 1: For your choice of dataset, you will first process the text, tokenize it and choose whether to do further pre-processing or filtering. If you do some pre-processing or filtering, then using the text with and without it can be one of your experiments.

For each dataset, there is a program template that reads the data. You should run the program on your data of choice and investigate some of the data in order to choose pre-processing or filtering.

Step 2: The second step is to produce the features in the notation of the NLTK. For this you should write feature functions as we have been doing in lab. You should start with the "bag-of-words" features where you collect all the words in the corpus and select some number of most frequent words to be the word features.

Now use the NLTK Naïve Bayes classifier to train and test a classifier on your feature sets. If possible, i.e. if time and space permit, you should use cross-validation to obtain precision, recall and F-measure scores. Or you can choose to produce the features as a csv file and use Weka or Sci-Kit Learn to train and test a classifier, using cross-validation scores.

Step 3: Experiments

A. For a base level completion of experiments, carry out at least several experiments where you use two different sets of features and compare the results. For example, you may take the unigram word features as a baseline and see if the features you designed improve the accuracy of the classification. Here are some of the types of experiments that we have done so far:

- filter by stopwords or other pre-processing methods
- representing negation (if using twitter data, note the difference in tokenization)
- using a sentiment lexicon with scores or counts: Subjectivity or LIWC
- different sizes of vocabularies
- possibly POS tag features on restricted data

B. Choose an additional, more advanced type of task from this list, or propose your own

- using Weka or Sci Kit Learn classifiers with features produced in NLTK
- using an additional type of lexicon besides Subjectivity or LIWC
- in addition to using cross-validation on the training set, train the classifier on the entire training set and test it on a separately available test set (both the Kaggle and SemEval data have these)
- implement additional features
    - in the email dataset, use word frequency or tfidf scores as the values of the word features, instead of Boolean values
    - use POS tagging from the ARK on Twitter
    - twitter emoticons or other features based on internet usage or informal text, such as repeated letters, or all caps words

Step 4: Write a report that describes the data processing, the features and the classification experiment(s). As one of your experiments, you may instead compare results from different classifier algorithms in Weka or Sci-Kit Learn.

Option 3. Programming Projects

Write a program to process text and discover lexical chains. We will work from the paper: Barzilay and Elhadad, "Using Lexical Chains for Text Summarization", 1999. We will identify a subset of their algorithm that is reasonable to implement in NLTK using WordNet. More details will be forthcoming.

Write a Python program with a window interface that allows a user to specify a file or directory of files to process. The program should use the Stanford Named Entity Recognizer to process the text. There is a Python interface to the Stanford NER by Dat Hoang at https://github.com/dat/pyner. After the NER processes the text, the python program should make and display most frequent words, pruned by a stop word list, and most frequent named entities for the categories Person, Organization and Location. More details are found in the Programming Projects document.

The programs should be well-documented in a report that is handed in with the code and with test results, using a small number of documents of your choice.

What to Hand In

Each person should hand in a report with the description of all that you did and the discussion of the results. Also submit any python programs that you write, particularly to show additional features that you implement, or the annotation files that you did.

If you worked in a group, the report should include which tasks each person worked on. Each person in the group should submit the (same) report . (Having every person submit the report makes it easier for me to record individual grades in Blackboard.)

As usual, submit these documents to the Blackboard system by the end of the day on the due date.