CSE 5525 Speech and Language Processing (Spring 2017)
Homework #1: Text Classification

Prof. Wei Xu, The Ohio State University

Due: 11:59PM, Tuesday, February 28th, 2017

Instructions The goal of this homework is for you to execute what you have learned in the class and implement the naïve bayes and perceptron algorithm. This homework will count 12% towards your final grade. Depending on the efficiency of your implementation the experiments required to complete the assignment may take some time to run, so it is a good idea to start early. **You can find starter code and data for this homework in Piazza's resources.** If you have any questions, the best way is to ask on Piazza.

In this assignment you will implement naïve Bayes and perceptron algorithms for text classification. You will train your models on a (provided) dataset of 25,000 positive and negative movie reviews and report prediction accuracy on a test set. We provide you with starter Python code to help read in the data and evaluate the results of your model's predictions. Please finish by yourself and turn in the following in Carmen:

- Your code
- A brief writeup that includes the numbers / evaluation requested below.

1 Naïve Bayes

Implement a naïve Bayes classifier. The provided code in `imdb.py` reads the data into a document-term matrix using scipy's `csr_matrix` format (For more details, see http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.sparse.csr_matrix.html#scipy.sparse.csr_matrix). You would need to work with log probabilities instead of multiplying them directly to avoid the possibility of floating point underflow (see: https://en.wikipedia.org/wiki/List_of_logarithmic_identities).

You can run the sample code like so:

```
python NaiveBayes.py ../../data/aclImdb 1.0
```

Classification and Evaluation (4 Points)

The first two methods you will need to implement are `NaiveBayes.Train` and `NaiveBayes.PredictLabel`. Before you do this, the classifier in the starter code always predicts +1 (positive). Once you have implemented these methods, the code will print out accuracy. Try running with different values of the smoothing hyperparameter (`ALPHA`) (suggested values to try: 0.1, 0.5, 1.0, 5.0. 10.0), and record the evaluation results for your report.

Probability Prediction (2 Points)

Then you will need to implement two methods `NaiveBayes.LogSum` and `NaiveBayes.PredictProb`. You would need to work with log probabilities and use the `log-sum-exp trick` to prevent potential numerical underflows (see: http://stats.stackexchange.com/questions/105602/example-of-how-the-log-sum-exp-trick-works-in-naive-bayes). Record the probability estimated for the first 10 reviews in the test data for your report.

2 Perceptron

Perceptron (3 points)

Implement the perceptron classification algorithm (analagous starter code is provided in `Perceptron.py`). The only hyperparameter is the number of iterations. Run the classifier and report results on the test set with various numbers of iterations (for example: 1, 10, 50, 100, 1000).

Parameter Averaging (2 points)

Modify the perceptron code to implement parameter averaging. Instead of using parameters from the final iteration, w_n, to classify test examples, use the average of the parameters from every iteration, $\sum_{i=1}^{N} w_i$. A nice trick for doing this efficiently is described in section 4.6 of Hal Daume III's CIML notes (http://www.ciml.info/dl/v0_99/ciml-v0_99-ch04.pdf).

Features (1 points)

Print out the 20 most positive and 20 most negative words in the volcabulary sorted by their weight according to your model This will require a bit of thought how to do because the words in each document have been converted to IDs (see `Vocab.py`). The output should look something like so:

```
word1_pos weight1
word2_pos weight2
word3_pos weight3
...

word1_neg weightk
word2_neg weightk+1
word3_neg weightk+2
...
```

Where `wordn_pos` and `wordn_neg` are the top 20 positive and negative words. (Hint: you might find the `numpy.argsort` method useful - http://docs.scipy.org/doc/numpy/reference/generated/numpy.argsort.html). Please include this output in your report.