NLP HW 4

Name: Vincent Lin (Zhenye)

Written

1. sense 1:
P(POS_{i-2}=$Verb|s$)=1/1
P(POS_{i-2}=$Noun|s$)=0
P(POS_{i-2}=$Num|s$)=0
sense 2:
P(POS_{i-2}=$Verb|s$)=1/3
P(POS_{i-2}=$Noun|s$)=2/3
P(POS_{i-2}=$Num|s$)=0
sense 3:
P(POS_{i-2}=$Verb|s$)=0
P(POS_{i-2}=$Noun|s$)=0
P(POS_{i-2}=$Num|s$)=1

2.
a. P($POS_{i+1} = noun$)= log(3/6)= -0.30102999566
b. P($w_{i+1} = art$)= log(4/5) = -0.096910013
note for b, I considered "arts" as a stem of art.

Programming

Preprocessing:
For this lab, I first parsed the xml file into list of a tuple of (word, tag) or a list of word. For every list, I created a tuple of the list with its tuple, which I then parsed into feature vectors. For the actual text itself, I preprocessed the test that it filters out all the stopwords in the text, meaning words such as "to, the, a". I did this using the nltk stopword vocabulary. I believe that the stopword should not affect the actual meaning of the word. So the window, I take the words from would be more meaningful. Furthermore, I used the Lancaster stemming algorithm in nltk to reduce every single words to its stem. I chose the Lancaster stemming algorithm because its the most aggressive algorithms out of the common Porter and Snowball stemming algorithm, which means it reduces words like banker, banks to bank. These preprocessing allowed me to extract the most meaningful words out of the windows in the feature vectors.

Expectation:
As learned in class, the predicted value is that the more features that one has the better the accuracy and the F1 Score will be.

Design Choices:
For Collocational feature vectors, I chose the window of 2 (the two words before bank and after bank) as it gave me the best result. For the co-occurrence feature vectors, I chose the bag of words from the most common words given a certain definition of 'bank'. For every definition, there are 7 most common words, which I combined into one list. I used the list to create a feature vector by measuring the count of the

word that appears in the text. For Collocational and for co-occurence, I simply added the list of vectors together. I used the same method for anything with Part of Speech Tagging. For the classifier, I used the NLTK's naive Bayesian classifier. To calculate the F1 score, I used sklearn.

collocational:
Results:
Accuracy:0.436781609195
F1Score:0.281450379195
With POS:
Accuracy:0.350574712644
F1Score:0.254503840619

co-occurrence:
Results:
Accuracy:0.379310344828
F1Score:0.261885880443
With POS:
Accuracy:0.32183908046
F1Score:0.180049575655

co-occurrence + collocational:
Results:
Accuracy:0.408045977011
F1Score:0.274365201342
With POS
Accuracy:0.295019157088
F1Score:0.267729139065

POS:
Accuracy:0.275862068966
F1Score:0.131168831169

Conclusion:
The general relationship between accuracy and F1 scores appears to be true, that the higher the accuracy, the higher the F1Score. The result were surprising in that whenever we combine the part of speech with any other features, it decreases both the accuracy and the F1Score. To explain this, I ran the part of speech as a separate classifier and I was able to obtained the accuracy of 0.275 and F1 score of 0.131. This is substantially lower than the accuracy and F1 scores of co-occurrence, collocational, and co-occurrence +collocational. This means that whenever that part of speech feature vector is added to the other feature vectors it will decrease its disambiguation rate. This appears to be a general trend: whenever you add a feature vector with a higher F1score to a feature vector with a lower F1score, the resulting F1score will be higher than the lower-score feature vector and lower than the higher-score feature vector. The reason for this is that extra features that have low disambiguation rate will cloud the pool of feature vectors with higher rate. In conclusion, because the low disambiguation rate of the POS tagging, it lowers the accuracy and the F1Score whenever it is used. Overall, I think that the Bayesian classifier, which does not give weight to certain feature vectors, decreased the disambiguation rate. Furthermore, I do think that using other classifiers might be able to get a higher accuracy rate than the Naive Bayesian Classifier.