NLP HW 2 Answer and Report

Fengzhang Du - 23628652

Q1

Predict “I always like foreign films.”
positive: 0.4* 0.09* 0.07* 0.29* 0.04* 0.08 = 0.00000233856
negative: 0.6* 0.16* 0.06* 0.06* 0.15* 0.11 = 0.0000057024
Predict : negative

Q2

a)
Done. please see the code in the python file.

b) movie_review_BOW_small.json in log probability


```
{
   "comedy": {
       "shoot": -4.0,
       "love": -2.415037499278844,
       "furious": -4.0,
       "fast": -3.0,
       "fly": -3.0,
       "fun": -2.0,
       "couple": -2.415037499278844
   },
   "action": {
       "shoot": -1.84799690655495,
       "love": -3.1699250014423126,
       "furious": -2.584962500721156,
       "fast": -2.584962500721156,
       "fly": -3.1699250014423126,
       "fun": -3.1699250014423126,
       "couple": -4.169925001442312
   },
   "prior": {
       "comedy": -1.3219280948873622,
       "action": -0.7369655941662062
   }
}
```


c)
“Fast couple shoot fly”
predict: action
probabilities: {'comedy': -13.736965594166206, 'action': -12.509775004326936}.

d)
I examined some documents which have the wrong predictions. These documents tend to have long descriptions on the movies itself, and show less preference or feelings of the reviewer. The review overall shows less sentiment and can be ambiguous even for a human reader. Also we can observe the log probabilities for each class. These log probabilities are very close for some wrong predicted document.

Program Report

This is a very good practice that helped me to understand the Naive Bayes classifier.
I did the preprocessing(pre_processing.py) for the first step and turned each document into vector format. The challenge is how to properly structure the data for it to be used in the Naive Bayes classifier. I utilized the dictionary and preserved the file name when constructing the data. This helps us to locate the document when we had a wrong prediction later in the testing phase, so we can go back to examine the document again.

Then, the constructed vectors were fed into the Naive Bayes classifier (NB.py) to be trained and classified. In the training phase, I calculated all the probabilities(prior and conditional) in log space using the bag_of_word feature and add_one_smoothing. Then wrote these parameters into the movie_review_BOW.json file for testing use.

Final step is for testing. The program made predictions for each file in the directory. Then output the given label, the predict label and the probabilities for each class in output.txt. When there is a wrong prediction, The program also outputs the file name so we can go back to examine the original document. In addition, in Question 2 d), I discussed the possible cause of the wrong prediction.

For the future improvement, I think it's worth trying to extract the word more precisely that only relates to the reviewer's opinion. For the summary of the movie itself. Maybe we should filter out those noises. Because if a movie's theme is negative, but the reviewer is positive about the movie. The summary of the movie will lead us in the wrong direction, and vice versa.