CSCI- 5832 - Martin - Natural Language Processing
NLP Assignment- 2

Name: Piyush Sudip Patel
StudentId: 104306264

Set-up:

- The files "*hotelPosT-train.txt*" and "*hoteNegT-train.txt*" are loaded as a part of the training data-set for the positive and negative movie reviews.
- The files are loaded in memory in the form of list.
- Before the testing data was available, out of the 95 positive reviews and 94 negative reviews, 5 positive and 4 negative reviews were extracted and that set was used as a test set (cross validation).
- The list loaded in the memory was shuffled by calling a random function, such that for every run, a different set of test data will be generated for cross validation (i.e. out of 95 positives, 5 docs will be splitted away for cross validation and out of 94 negatives, 4 docs will be splitted away for cross validation hence 9 set in total for cross validation)
- The entire dataset was loaded in a list separated by spaces and each word was lowercased. This was done so that, the word ***"Happy"*** and ***"happy"*** be treated as the same word.
- Multinomial boolean version of naive bayes was implemented - So, in every document, the repetition of words were removed. In sentiment analysis, we are more concerned about whether a word exists in our corpus or not rather than finding the frequency of that word.
- Additional set-up of naive bayes included calculating the log (base 10) of the terms since the values were very small, so chances were the floating point accuracy of the system would bump them to a value of **0** if the values became too less. Taking log of the values would prevent this mishap.
- An entire corpus of the vocabulary was made by adding all the unique words from all the docs.
- A global list of the likelihood of a word given that it's in a positive document was computed for all the words in the vocabulary and was maintained in a dictionary. The same was done for all the words given that it's in a negative document and the same was maintained in a dictionary as well.
- After the 50 test-data set was received, i read it and hand labelled it to measure how the performance of my model.

Building / Improving the accuracy of the system:

- To perform a faster calculation of the naive bayes, the denominator of the naive bayes and the prior was dropped.
    - The Denominator was dropped because it was constant across and hence would be a redundant in calculation.
    - The prior was dropped because it was **0.5** for both the positive and the negative reviews (number of positive docs = number of negative docs).
    - After the **50** test data were given the prior values for positive and negative reviews were changed to **95/189** and **94/189** respectively.
- To improve the accuracy, several techniques were used, some were successful and some were just plain horrible. I have discussed some of the techniques i've used in my program:
    - A unigram model of all the words in the document.
        - I also used *bigrams* and compared the accuracy with *unigrams*. Using *bigrams* gave a lesser accuracy than using unigram model. I concluded that since we have less data (189 docs), the probability of finding a particular bigram string would be really less and finally went with a unigram model.
    - A word-corpus of called **Vader-sentiment** was used. ***VADER (Valence Aware Dictionary and sEntiment Reasoner)*** *is a lexicon and rule-based sentiment analysis tool that is specifically*

attuned to sentiments expressed in social media, and works well on texts from other domains. https://github.com/cjhutto/vaderSentiment (Link to the wordlist).

- **System Development Approach:**

  - This word list contains a list of **7517** words, which provides a list of positive and negative word-list i.e. every word has been given a score of +'ve value and -'ve value. So, depending on the word's polarity, it was classified and stored in separate list in memory.
  - The list was added to the vocabulary (removing the duplicated word of-course) and the list of the positive words from **VADER** were added to the Positive training corpus and the list of negative words were added to the list of negative training corpus in the program.
  - After including all the words from the corpus, it gave me a really bad accuracy score (around **46%**) . However, after analyzing the **VADAR** data, there were many symbols including "emoticons" (No emoticons were found in the test data, hence emoticons were removed), "random non dictionary words", etc.. . I took a subset of the word corpus, with severals rules e.g.:
    - The word length between **5** and **12**, etc.. and **16** and **20**.
    - Ignoring the words not in dictionary.
  - Finally i included the word of lengths greater than **16**.
  - The positive corpus list was added to each of the **95** positive documents and the negative corpus list was added to each of the **94** negative documents.
    - It gave a bad accuracy score. I believe it's because the value of a word was overestimated by keeping it in all the documents.
  - I re-modified it by adding the positive wordlist of the VADAR corpus + the manual words identified by looking at mis-classified documents in *a single document* of the positive review and done the same for the negative VARDAR corpus + manual words with the negative reviews.
    - By doing so, the individuals words are not given an unnecessary high value, at the same time maintaining a positive word status.
    - After manually searching through the documents which were misclassified (around 5), i included some of the positive words along with the selected vader corpus of words (mentioned earlier).
    - After fiddling through multiple words, modifying words range from the VADAR dataset, I ended up with an *accuracy score of **96%***.

Analysis/ How well my system performs :

- **The baseline accuracy was found between 77.77 - 88.88 % averaged over several runs when the cross validation data set was used.**
- **Over the 50** test-data set, the baseline accuracy was found to be **88.0%** exactly.
  - Which signified **6** misclassifications out of **50** test-set data.
- In the improved model (and after manually reviewing all the **50** test data reviews and labelling them as positive and negative reviews) I matched it with the output of my program.
  - I found out there are **2** wrongly classified documents, **48** correctly classified documents.
- ***Hence, the final analysis score for the 50 test-data after improving the system was 96%.***