National Institute of Technology Calicut

Department of Computer Science & Engineering

Natural Language Processing

Course Project

Due Date 1: March 30, Due Date 2: April 11

Instructions

- Form groups of 3 people. Create one github project repository and upload your code and files there. Share the link with the instructors.
- There are two parts to the project. Part A has to be submitted by 30^{th} March, and part B by April 11^{th}
- Any kind of copying, if caught, will be dealt with appropriately, including awarding 0 marks in the project or an F grade in the course.

PART - A

1. Collect product reviews from e-commerce sites and label them as + or -
   You have to create a file *data.txt* which contains at least 500 reviews and their labels. Each review (called a document) should be in a single line. (So there will be at least 500 lines in the corpus)
2. Write code to remove stop words and convert the whole text to lower case. Also drop any characters other than alphabets. You may use this list of (stop words). Include the data file after the removal of stop words as well as the code used for this in a folder.
3. Create vocabulary. Vocabulary may be defined as the set of all words in the corpus which appears at least 2 times. Show the vocabulary as a text file named *vocabulary.txt*. The folder for this step should contain the code for creating the vocabulary as well.
4. Train a multinomial naive bayes model. Use laplace smoothing/add one smoothing. You should be able to save the model (eg:- pickle package for python or serialization in Java). Your code should be able to train given any input training data.
5. Evaluation: Do 10 fold cross validation. Divide your data into 10 equal parts (approximately containing same number of + and - samples). Train with 9 parts and test on the left out 1 part. Report the accuracy (not f-measure). repeat this 10 times each time leaving one part for testing and using the rest 9 part for training. Finally report the average accuracy. You can give the accuracy results in the **Readme** file of your github project page.(**Use log probabilities to prevent underflow as explained in the class**)

PART - B

6. In this part, use some bigram features as well. Find out all the bigrams (after stop word removal though) which appear ≥ 3 times. (You can change this if there are too many bigrams or too few bigrams, depending on your corpus). Now if $\{not\ good\}$ is a bigram, then consider this as a feature (means add this bigram to your vocabulary) in addition to the unigram features $\{not\}$ and $\{good\}$.

7. Repeat the training and testing as in Part A. Make sure that when you are counting a bigram in the training data, you shouldn't count the corresponding unigrams. For example if there is $\{not\ good\}$ appearing in the training corpus, you shouldn't add that to the $\{not\}$ and $\{good\}$ unigram counts, but only to the bigram count. Report the 10 fold cross validation performances as well as the average accuracy.