Brukernavn: syhd
INF3800 (Vår 2017) - Oblig 5

Coding Assignment

My implementation for the k-grams (i.e. fuzzy search) is pretty straight-forward, so I won't talk about it here. As for the NB classifier, the majority of my implementation is also taken directly out from the slides/textbook, including the Laplace smoothing formula.

I made one small change to the prekode though. In the prekode, the `likelihood` variable was a map of the type `HashMap<Integer,HashMap<Integer,Double>>`. But I thought that this was unnecessarily complicated and not efficient at all, so I changed it to a 2D-array (i.e. a table) such that it became `likelihood[i][j]` (where `i` represents the class number, and `j` represents the term number on the global lexicon). The main reason is because we need to store the likelihood value for each and every pair of `[i,j]` anyway, so there is really no superior performance from using a hash map over the simple 2D-array. Besides, the 2D-array gives a simpler and more compact data structure.

Using the Laplace smoothing technique, my implementation of the NB classifier gives about 84% accuracy. To improve this, I've tried to include a stop dictionary (as suggested). However, this gave only a 2%-jump in the level of accuracy. To determine if a term is a stop word or not, I've used the document frequency (which is basically the number of documents that the term is in). In the `MultinomialNaiveBayes` class, I defined the constant `THRESHOLD`, which can be used to set the level of document frequency at which we can declare a term to be a stop word. For example, let the threshold be 0.2, then a term is a stop word if and only if it appears in more than 20% of the total number of documents in the training set. Note that if the threshold is set to 1.0, then it's like we are not using the stop dictionary at all (because no term can possibly appear in more than 100% of all the documents).

This method of identifying the stop words is reasonable, because if a term appears in only a few documents in the training set, then it's quite likely that these documents actually belong to the same class, which then implies that this term is more likely to affect the chance of a document belonging to that one class rather than some other class. The testing result is as follow (just a bit disappointing):

$$Threshold = 1.00 \Rightarrow Accuracy = 84.0\%$$

$$Threshold = 0.20 \Rightarrow Accuracy = 86.5\%$$

$$Threshold = 0.05 \Rightarrow Accuracy = 87.6\%$$

$$Threshold = 0.01 \Rightarrow Accuracy = 86.3\%$$