R04922170 林于智

Naive Bayes Classifier

Utilizing the concept of unigram model to do text categorization, we can estimate the probability (likelihood) of a category given a document through ***the product of probabilities*** of all the words in that document. And the probability of a word with respect to a category is estimated with the occurrences of this word in the category.

Therefore, every document in the test dataset will come with `|C|` estimations, where `|C|` is the number of categories in the corpus. Documents will be classified as the category with the greatest estimation of all.

EM Algorithm

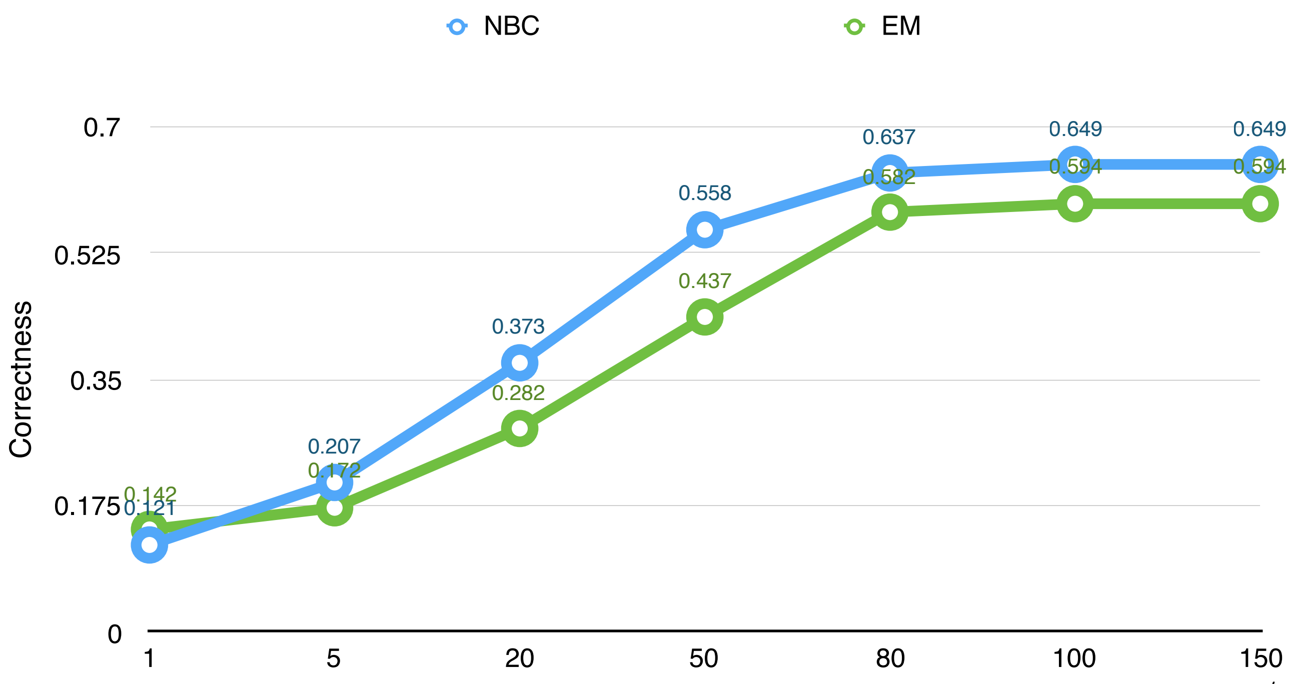
The purpose of EM algorithm in this homework is to consider the unlabeled dataset, and we can see its effects in the Experiments section below.

Basically EM algorithm consists of 2 steps, i.e., ***E-step and M-step***. E-step is where we estimate the unlabeled data with our model, while M-step uses the result from E-step in order to maximize the likelihood, i.e., to get better parameters.

Since we have the concept of unigram model in NBC above, the new likelihood-maximized model generated in M-step will be the original model with the prediction of unlabeled data in E-step added. As for E-step, it is actually identical to the NBC described above.

Experiments

For my experiment, I ran both NBC and EM for labeled data size = 1, 5, 20, 50, 80, 100, 150. And the results are shown in the graph below.



We can infer from the graph above that EM algorithm helps when labeled dataset is insufficient but no significant improvement found when the labeled dataset is relatively large.

Some Techniques & Their Impacts

For those words that don't appear in a certain category, smoothing technique is somewhat required. In a small test with NBC and with labeled-data size 150, the number of documents in test data that are correctly predicted with smoothing technique is several hundred more than that with no smoothing technique. The smoothing technique that I applied is additive smoothing which gives 0.5 occurrences to those words that are actually not seen in that category.

Other Observations

I found in a small test that additive smoothing with 0.5 is better than additive smoothing with 1.0 because the corpus is not so large. However, additive smoothing with 0.2 doesn't appear to be persuasively better than 0.5.