Project 4

Shun Zhang (sz4554)

December 3, 2012

Implementation

For KNN, I used the `retrieve` method, which returns the sorted results based on tf-idf weight. Then, use the first K ranked examples and find out the majority category of them. I also built a hash map which mapping from Document name to its category, this would make the testing more efficient.

For Rocchio, I just used a `HashMapVector` to represent the prototypes of each category. I added a negative scale to other categories when `neg` is enabled.

Comparative accuracy of the algorithms at different points on the learning curve for the training data.

KNN when K = 3, K = 5 are less overfitting the trainning data, so their performance when data is small is very bad. They start to learn well when enough data are fed.

KNN when K = 1, which is exactly nearest neighbor classifier, has the best consistency with training data. This doesn't make much sense, as it's overfitting the data - its performance on testing data is the worst.

Both versions of Rocchio and KNN when K = 3, K = 5 show some evidence of good generalization. They range from 75% to 85% in accuracy, which are consistent with most data, but not all.

Comparative accuracy of the algorithms at different points on the learning curve for the testing data.

They all start from 33%, where they are randomly choosing. Their accuracy increase and start to converge when size of trainng set increases.

Both versions of Rocchio and Naive Bayes have the best performance. According to `yahoo-science` corpus, Naive bayes is better than modified Rocchio, which is better than Rocchio.

KNN come in order of K = 5, K = 3 and K = 1. It has a good performance is K is larger. This doesn't mean the larger K is, the better the

accuracy is. When K equals size of corpus, it simply returns the same answer - the majority category of the corpus. In this case, the accuracy would drop down to 33% again.

Comparative running times of the algorithms in training and testing phases.

yahoo-science

Classifier	Training Time (sec.)	Test Time (ms.)
Naive Bayes	2.764	0.04
Rocchio	1.481	0.66
Rocchio (neg)	4.135	1.39
KNN (K = 1)	2.817	0.33
KNN (K = 3)	2.846	0.33
KNN (K = 5)	2.845	0.31

yahoo-top

Classifier	Training Time	Test Time
Naive Bayes	2.256	0.04
Rocchio	1.107	0.54
Rocchio (neg)	3.1	1.46
KNN (K = 1)	2.43	0.28
KNN (K = 3)	2.481	0.27
KNN (K = 5)	2.442	0.28

It shows that modified Rocchio costs more training time, as it needs to modifty the prototype of all the categories when one exmaple is seen (also approximiately 3 times of Rocchio).

Naive Bayes has the least test time, as it's just the calculation of probability, which is some multipulation operations.

All KNN have the similar training and test time. Most of the time is costed in retrieval process. I'm using retrieval here for coding convinience. For a perticular K, it can be optimized by using the algorithms of finding K-largest elements (for K = 1, just find the maximum value).