**Data mining project 2 - report**

计23 田家申 2012011296

**I. Data preprocessing**

1. Overview

The bag-of-words model is used to extract feature from the raw texts. For every text, a word frequency vector is made indicating words that appear in the text. Words containing only Latin letters are counted.

2. Tools

The compressed files in each directory in decompressed using FOR command under Windows.

Using the Java code provided with the corpus, the documents are parsed. Another Java class (Files, in files.java) is written to get the documents and produce the vectors as well as extract the taxonomy information.

**II. Classification**

**1. Methods**

Support Vector Machine and k-nearest neighbors algorithm are used to make classifiers of the documents.

For SVM, the tool LIBSVM is used. The aforementioned Files class produces text files in LIBSVM format containing the vectors and their taxonomy information to be used by LIBSVM as train set data or test set data.

For k-NN algorithm, I implement it in Java myself. To run the whole train and test process, one can run the knn\_test function in Classify class (Classify.java). K-NN itself is in the knn function. In my implementation, every test item’s presumed class must receive a vote above indicated lower limit. When picking out the k nearest neighbours, cosine distance is used to measure the distance between two vectors. A prediction is considered true with one of its classes coincides with any class the document really belongs to.

**2. Results**

Because the large amount of documents in the whole corpus, only a selection of them is used.

**i. K-NN**

Train set: 1000

Test set size: 200

k = 50

Minimum vote = 5

Accuracy: 87.5%

**ii. SVM**

Because one document in the corpus can belong to several different classes, even multi-class SVM does not perform well in this scenario. To relieve this problem, each class is treated separately. Attempt is made to classify the documents on account of whether it belongs to a specific class.

Train set: 1000

Test set size: 2000

Parameters: default

Data is scaled using svm-scale.exe

Accuracy: 69.1%

It seems that SVM doesn’t work well in this case. Future work can be done to find a good set of parameters.

**III. Hierarchical Classification**

**1. Method**

In hierarchical classification, the relations of the classes matters. Here k-nn algorithm is used, and the taxonomical information of processed beforehand to remove duplication, namely a parent class with its children (e.g. “Top/News” and “Top/News/World”), in which situation the parent class is removed. The other things is the same as mentioned in the previous section about k-nn.

**2. Results**

Train set: 1000

Test set size: 200

k = 50

Minimum vote = 5

Accuracy: 85.5%

As we can see, there is a drop of accuracy comparing with pure classification. The reason might be that the elimination of duplicated classed reduces the possibility of a vague but true prediction or some random factors.

**IV. Conclusion**

Through the project, we can see that the task of text classification requires suitable ways to be accomplished with good results. K-nn performs not so well on the same data when using L2 distance; with cosine distance the results are better. On the other hand, SVM is generally considered to be more powerful than k-nn, but in this case its performance is poor. This is because the default parameters are used, and they are by no means optimized for text classification. Should a better parameter be found, the result will turn out better.

**V. Notes**

The source code is contained in the tools\src folder.

Data and executable files of SVM is located in svm\_data folder.

Thank you for reading!