CS3244 Machine Learning

Assignment 2: Text Classification with WEKA

Nguyen Viet Dung – A0112068N

Overview

Text classification is a common problem in Machine Learning and Natural Language Processing. In this assignment, we will classify some posts on several newsgroups.

There are five newsgroups: **comp.graphics, comp.os.ms-windows. misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware and comp.windows.x.** About dataset, we are given 1425 instances to classify and 2935 instance for training. All the experiments in this report will be done using WEKA 3.7.13.

I. Performance of Individual Classifiers

First, we need to identify the number of features that we will use. As setting an overly high value of the feature selection may make the feature vector too specific to the training set. While a low value of the feature selection will reduce the precision of the learning and prediction.

So for the first step, we extract the feature vectors and start training using 4 classifiers: Naïve Bayes, J48, K-Nearest Neighbors (IBk) and Support Vector Machines (SMO). We will run these classifiers several times using different values of **fs_top_num** (that will affect the number of keywords / features). The weighted F-measures of the experiments are shown in Table 1.

fs_top_num	# Keywords/Features	Naïve Bayesian	J48	IBk	SMO
50	203	0.736	0.678	0.736	0.769
100	428	0.737	0.688	0.76	0.805
150	641	0.744	0.694	0.769	0.814
200	857	0.75	0.691	0.774	0.818
250	1073	0.754	0.7	0.777	0.835

Table 1. Weighted F-measure of classifiers with different numbers of features used.

We do the experiments in 5 founds, start with **fs_top_num=50** and increase it by **50** by each round. From the table, we observe that the weighted F-measure increases as the number of feature increases. However, large feature vectors may cause the classification

to be very slow and make repeated testing difficult. So we stop at **fs_top_num=250** as it is large enough and does not take too much time for training.

The following sections will describe the performance of each individual classifier. We will use **fs_top_num=250** and **10-fold cross validation** in our evaluations.

1. Naïve Bayes

The Naïve Bayes does not allow us to tweak parameters much. So we only use the default setting of Naïve Bayes and the result is follow (table 2).

	TP Rate	FP Rate	Precision	Recall	F-Measure
Class 0	0.723	0.047	0.793	0.723	0.756
Class 1	0.77	0.105	0.648	0.77	0.704
Class 2	0.675	0.059	0.741	0.675	0.706
Class 3	0.83	0.069	0.748	0.83	0.787
Class 4	0.767	0.029	0.87	0.767	0.815
Weighted Avg.	0.753	0.062	0.76	0.753	0.754

Table 2. Experiment result of Naïve Bayes classifier.

According to the table, we can see that Naïve Bayes works better for class 3 (F-Measure 0.787) and class 4 (F-Measure 0.815). Compare to other classifiers, the Weighted Average F-measure is still better than J48 with default setting (we can see from table 1).

2. J48

For J48, parameters that highly affect the result are the confident factor used for pruning and the minimum number of instances per leaf. We try some different values for these 2 parameters and note down the weighted average results as in Table 3.

Confident factor	Minimum instances	TP Rate	FP Rate	Precision	Recall	F-Measure
0.25	2	0.7	0.075	0.702	0.7	0.7
0.35	2	0.701	0.075	0.703	0.701	0.702
0.15	2	0.702	0.074	0.705	0.702	0.703
0.25	1	0.709	0.073	0.712	0.709	0.71
0.25	3	0.686	0.079	0.687	0.686	0.686

0.1	1	0.706	0.073	0.709	0.706	0.707

Table 3. Experiment with J48 using different confident factor and minimum number of instances.

As J48 takes quite some time to complete training, we only try 6 pairs of values of the confident factor and the minimum number of instances. All these values are taken near the default one so that we have a better comparison. From the table, we can see the F-Measure does not change much, in the range from 0.686 to 0.71. The precision is in the range from 0.687 to 0.712 and the recall is from 0.686 to 0.709. The highest F-Measure, Precision and Recall are 0.71, 0.712 and 0.709 with the confident value 0.25 and the minimum of number of instance 1.

So we will use this result of confident factor 0.25 and 1 minimum instance for further comparisons.

3. K-Nearest Neighbors (IBk)

By default, k is set to 1 for IBk on WEKA. It is followed that every new instance will be classified the same as its first nearest neighbor. So the first idea to test the performance of IBk is to try numerous values of k. However, better way to do this is to turn on the cross validate (-X code). This will find the best value of k between 1 and a specified number. However, this will slow down the program as well. So we will set this value to 50, which is large enough and able to run in acceptable time.

Besides the cross validate, we need to consider the distance weighting also. As by default, this attribute is turned off. There are two ways of expressing the distance weighting is weight by 1/distance (-I code) and weight by 1-distance (-F code). By trying these experiments using the cross validate and different distance weighting, we get the result in table 4.

Label	Value for X	Distance Weighting	TP Rate	FP Rate	Precision	Recall	F-Measure
IBk -X	50	null	0.774	0.057	0.779	0.774	0.774
IBk -X	50	1/distance (-I)	0.792	0.052	0.795	0.792	0.792
IBk -X	50	1-distance (-F)	0.78	0.055	0.783	0.78	0.78

Table 4. Performance of IBk with cross validate and different distance weighting.

From the table, it is clearly that the 1/distance (-I) is the best among all. The F-Measure increases about 2% compared to no distance weighting and about 0.4% with 1-distance (-F). The precision and recall are highest also.

Hence, we will use these setting with the cross validation and 1/distance of distance weighting for further comparisons (-X -I).

4. SMO (WEKA's implementation of SVM)

In SMO, one of the things that we can change to test its performance is the exponent of the Poly Kernel. By default, SMO learns a classifier with a linear decision boundary (exponent 1). Then we will try to do experiments with different values of this exponent. The result is shown in table 5.

Exponent	TP Rate	FP Rate	Precision	Recall	F-Measure
0.5	0.757	0.061	0.76	0.757	0.758
1	0.834	0.041	0.836	0.834	0.835
1.5	0.85	0.037	0.851	0.85	0.851
2	0.845	0.039	0.846	0.845	0.846
2.5	0.842	0.04	0.842	0.842	0.842

Table 5. Performance of SMO with different exponent of Poly Kernel.

From the table, we observe that the results of exponent from 1 to 2.5 are slightly different (smaller than 1%). However, we can see that SMO will work the best with exponent 1.5. Then the F-Measure is about 0.851, precision 0.851 and recall 0.85.

So we will use SMO with exponent of Poly Kernel 1.5 for further comparisons.

II. Comparison

After do experiment for each of 3 classifiers, we have chosen the best setting for each of the classifier. That is:

- Naïve Bayes
- J48 –C 0.25 –M 1: J48 with confident factor 0.25 and minimum instance 1.
- IBk –K 50 –X –I: IBk with cross validation of value 50 and distance weighting is 1/distance.
- SMO –E 1.5: SMO with the exponent of Poly Kernel 1.5

We combine the data from table 1 to 5 and come up with table 6 to compare the performance of these classifiers with the best settings.

	TP Rate	FP Rate	Precision	Recall	F-Measure
Naïve Bayes	0.753	0.062	0.76	0.753	0.754
J48 –C 0.25 –M 1	0.709	0.073	0.712	0.709	0.71
IBk –K 50 –X –I	0.792	0.052	0.795	0.792	0.792

SMO –E 1.5	0.85	0.037	0.851	0.85	0.851

Table 6. Performance of classifiers with the best settings.

From table 5, it is clearly that SMO is dominating IBk, Naïve Bayes and J48. It is better than Naïve Bayes and J48 in most aspects, from correct prediction rate, precision, recall to the F-Measure.

Next we try to compare these classifiers using WEKA. As this process may take quite a lot of time, we set the number of folds to 5 and the number of repetitions to 5. This may speed up the process a little bit. The actual time for this process is approximately about 30 minutes. By the way, with the discussion above, SMO is likely the best classifier among all. So we set it as the test base and the result is as below:

Dataset	SMO	Naïve Bayes	J48	Ibk
5Newsgroups-train	83.86	74.75	70.39	78.24
	(v / / *)	(0/0/1)	(0/0/1)	(0/0/1)

Table 7. Comparison between classifiers using WEKA.

So WEKA gives the same result as our discussion. SMO performs better than all of the others with this dataset.

III. Conclusion

In general, all the selected classifiers perform well with over 0.75 for the F-Measure and above 70% for the true positive rate.

Through experiments, we also see the best settings for each of the classifiers. Base on the properties of the data set, these settings may change accordingly.

With the comparisons, we also see the performance of each classifier. For this data set, SMO is the best with about 85% of accurate predictions, while J48 is the worst with about 70% of accurate predictions. So we will use SMO with exponent of Poly Kernel about 1.5 for testing and prediction.