# Text Categorization Using Hybrid Multiple Model Schemes

In-Cheol Kim[1] and Soon-Hee Myoung[2]

[1] Dept. of Computer Science, Kyonggi University
San94-6, Yiui-dong, Paldal-gu, Suwon-si, Kyonggi-do, Korea,
kic@kyonggi.ac.kr

[2] Dept. of IMIS, Yong-In Songdam College
571-1 Mapyong-dong, Yongin-si, Kyonggi-do, Korea,
shmyoung@ysc.ac.kr

**Abstract.** Automatic text categorization techniques using inductive machine learning methods have been employed in various applications. In this paper, we review the characteristics of the existing multiple model schemes which include bagging, boosting, and stacking. Multiple model schemes try to optimize the predictive accuracy by combining predictions of diverse models derived from different versions of training examples or learning algorithms. In this study, we develop hybrid schemes which combine the techniques of existing multiple model schemes to improve the accuracy of text categorization, and conduct experiments to evaluate the performances of the proposed schemes on MEDLINE, Usenet news, and Web document collections. The experiments demonstrate the effectiveness of the hybrid multiple model schemes. Boosted stacking algorithms, that are a kind of the extended stacking algorithms proposed in this study, yield higher accuracies relative to the conventional multiple model schemes and single model schemes.

## 1 Introduction

The increasing amount of textual data available in digital form with the advancement of WWW, compounded by the lack of standardized structure, presents a formidable challenge for text management. Among various techniques to find useful information from these huge resources, automatic text categorization has been the focus of many research efforts in the areas of information retrieval and data mining. In the 1980s, expensive expert systems were widely used to perform the task. In recent years, inductive machine learning algorithms that are capable of generating automatic categorization rules have been adopted to text categorization with success. They offer less expensive, but more effective alternatives[1][2]. Naive Bayesian, k-nearest neighbors, and decision tree algorithms are the subset of machine learning algorithms employed in our study.

More recently, several multiple model schemes have been explored in an effort to achieve increased generalization accuracy by means of model combination.

M.R. Berthold et al. (Eds.): IDA 2003, LNCS 2810, pp. 88–99, 2003.

Multiple models are derived from different versions of training examples or different types of learning algorithms. In various practical applications, such as pattern recognition and data mining, methods of combining multiple classifiers have been widely used. Results demonstrate the effectiveness of this new approach. However, a great variation in error reduction has been reported from domain to domain or from dataset to dataset[5].

In this paper, we briefly review the characteristics of basic multiple model approaches and propose new hybrid approaches that include stacked bagging and boosting as well as bagged and boosted stacking. We also evaluate the performances of the hybrid multiple model schemes using real world document collection from the MEDLINE database, Usenet news articles, and web document collection.

## 2 Standard Multiple Model Schemes

The current standard multiple model schemes, namely bagging[6], boosting[7], and stacking[8] have evolved mainly from the need to address the problems inherent in models derived from single learning algorithms based on limited training examples. One of the problems is the instability observed in learned models. For example, while decision trees generated with the C4.5 algorithm is easy to comprehend, small changes in data can lead to large changes in the resulting tree[4]. The effect of combining multiple models can be evaluated with the concept of bias-variance decomposition. High variance in predictors is inherent in models derived from single algorithms based on a finite number of training examples. Inductive learning methods employ different representations for the learned model and different methods for searching the space of hypotheses. Representation and search methods make up the source of persistent learning biases for different algorithms.

Recent approaches deal with the challenge by using multi-strategy techniques. These multiple model approaches can intervene in all phases of the conventional machine learning process by modifying input, learning, and/or output phases. In the input phase, multiple different subsets of data can be selected as an input data set for the learning phase by applying different sampling methods into a given dataset. In the learning phase, multiple models can be induced by applying different learning algorithms on the same single set of training data or by applying the same learning algorithm on multiple different sets of training data. In the former case, homogeneous classifiers sharing the identical representation are derived. On the other hand, in the latter case, heterogeneous ones are generated. In the output phase, for the final decision, the predictions by multiple classifiers can be combined according to different combination methods, such as the weighted vote in boosting or the additional meta-learning in stacking. All in all, the approaches using multiple models can be characterized by many aspects: the data sampling methods, the model representing methods, the model learning methods, and the prediction combining methods. Whereas each model has its own bias and variance portions of errors, homogeneous models mainly

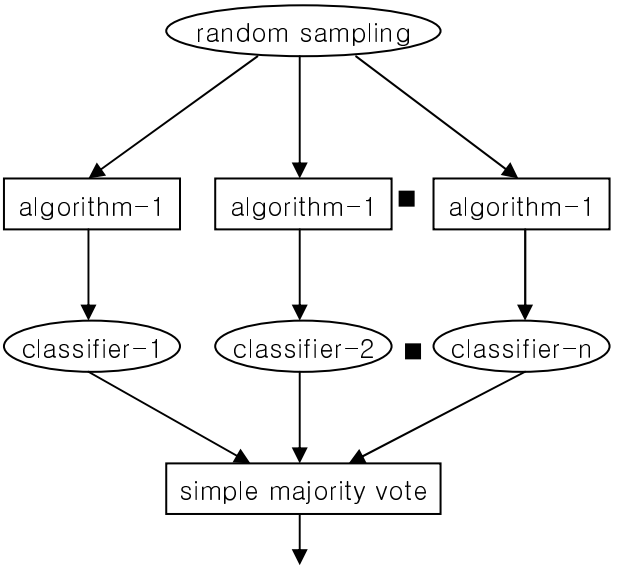


**Fig. 1.** Bagging

address the variance effects of the error, using perturbation of training data. Heterogeneous models mitigate the bias resulting from participating algorithms by combining the models induced from different learning algorithms by means of meta-learning. Bagging, the simplest of voting or committee schemes, uses random sampling with replacement (also called bootstrap sampling) in order to obtain different versions of a given dataset. The size of each sampled dataset equals the size of the original dataset. On each of these versions of the dataset the same learning algorithm is applied. Classifiers obtained in this manner are then combined with majority voting. Fig. 1 illustrates the concept of bagging. Boosting, such as the popular AdaBoost, is drastically different from bagging in practice in its method for sampling and drawing decisions. Fig. 2 represents the concept of boosting. Boosting first builds a classifier with some learning algorithm from the original dataset. The weights of the misclassified examples are then increased and another classifier is built using the same learning algorithm. The procedure is repeated several times. Classifiers derived in this manner are then combined using a weighted vote. In boosting, the weights of training data reflect how often the instances have been misclassified by the classifiers produced so far. Hence the weights are altered depending on the current classifier's overall error. More specifically, if $e$ denotes the classifiers's error on the weighted data, then weights are updated by the expression (1).

$$weight \leftarrow weight \cdot \frac{e}{1-e} \tag{1}$$

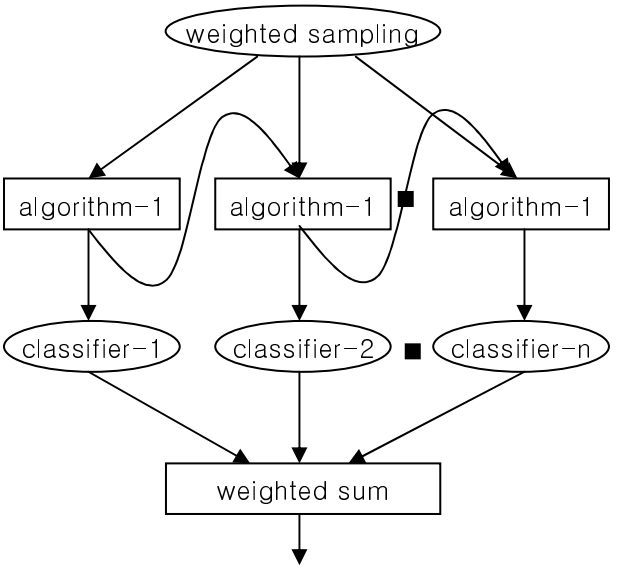


**Fig. 2.** Boosting

In order to make a prediction, the weights of all classifiers that vote for a particular class are summed, and the class with the greatest total is chosen. To determine the weights of classifiers, note that a classifier that performs well on the weighted training data from which it was built should receive a high weight, and a classifier that performs badly should receive a low one. More specifically, we use the expression (2) to determine the weight of each classifier.

$$weight = -\log \frac{e}{1-e} \tag{2}$$

Better performance is observed in boosting compared with bagging, but the former varies more widely than the latter. In addition, bagging is amenable to parallel or distributed processing. One practical issue in machine learning is how to deal with multiplicity problem [9]. A critical problem is to select a learning algorithm that is expected to yield optimal result for a given problem domain. The strategy of stacked generalization or stacking is to combine the learning algorithms rather than to choose one amongst them. Stacking achieves a generalization accuracy using two phases of processing: one by reducing biases employing a mixture of algorithms, and the other by learning from meta-data the regularities inherent in base-level classifiers. Stacking introduces the concept of a meta-learner, which replaces the voting procedure. Stacking tries to learn which base-level classifiers are the reliable ones, using another learner to discover how best to combine the output of the base learners. Figure 3 depicts the way stacking is conceptualized. The input to the meta model (also called the level-1 model) are the predictions of the base models, or level-0 models. A level-1

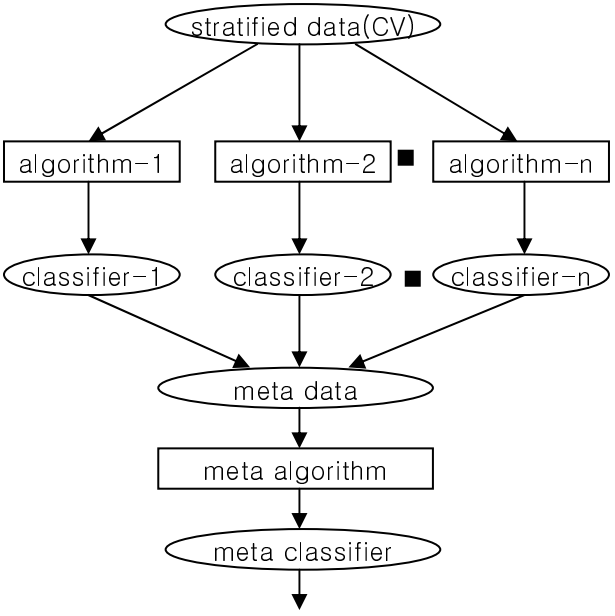


**Fig. 3.** Stacking

instance has as many attributes as there are level-0 learners, and the attribute values give the predictions of these learners on the corresponding level-0 instance. When the stacked learner is used for classification, an instance is first fed into the level-0 models, and each one guesses a class value. These guesses are fed into the level-1 model, which combines them into the final prediction. In stacking, cross-validation is used as a means for error estimation. It performs a cross-validation for every level-0 learner. Each instance in the training data occurs in exactly one of the test folds of the cross-validation, and the predictions of the level-0 inducers built from the corresponding training fold are used to build a level-1 training instance. Because a level-0 classifier has to be trained for each fold of the cross-validation, the level-1 classifier can make full use of the training data.

## 3 Hybrid Multiple Model Schemes

In this study, we develop two classes of hybrid schemes: one is the extended vote algorithms employing a meta-level learning phase instead of a simple majority vote to combine homogenous classifiers' predictions, and the other is the extended stacking algorithms, also augmented with different versions of dataset to train multiple heterogeneous base-level classifiers. Our implementation incorporates all the concepts and techniques of existing multiple model schemes.

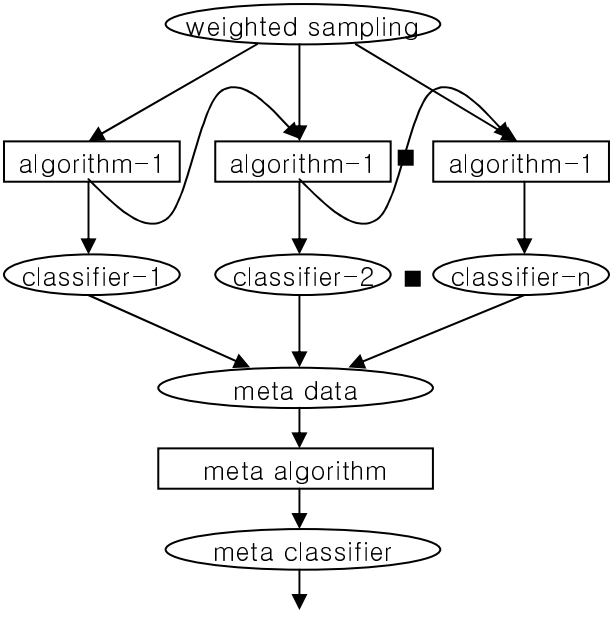


**Fig. 4.** Stacked boosting

Fig. 4 presents the concept of stacked boosting as an extended vote algorithm. Stacked bagging differs from stacked boosting only in that it samples training data for base model generation with uniform probability. The outputs generated by base-level classifiers are turned into a sequence of class probabilities attached to the original class to form meta-data. Instead of taking majority vote for the final decision as seen in simple bagging or boosting, the hybrid vote algorithm applies a meta-learning algorithm to meta-data to induce a meta-classifier which produces the final decision.

As for the extended stacking, we implement the stacking algorithms combined with model generation modules adapted from bagging and boosting. We call these extended stacking algorithms as bagged stacking and boosted stacking algorithms, respectively. The meta-data are produced in the same manner as in the extended vote algorithms, except that multiple models are generated from different algorithms independently and the output of multiple classifiers from each algorithm are linearly combined and turned into meta-data. Meta-level algorithm is applied to learn the regularities inherent in base models again, and the meta-classifier induced works as an arbiter of these heterogeneous base models. Fig. 5 shows the concept formulated for the implementation of the boosted stacking.

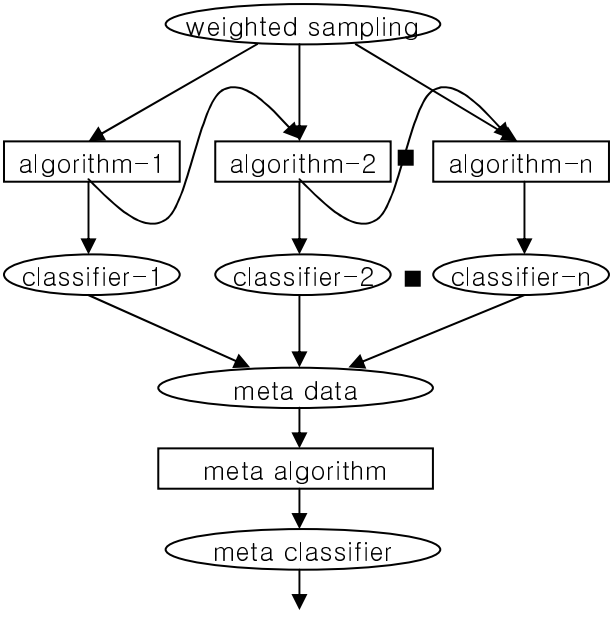


**Fig. 5.** Boosted stacking

## 4 Experiments

In this section, we compare the results of the empirical test of the existing basic multiple model algorithms and those of our proposed schemes, all based on Naive Bayesian, k-NN, and decision tree algorithms. We first describe the conditions under which the experiments were carried out, the data used, and how the experiments were executed. Next, we then evaluate the performances of various classifiers.

At the practical level, text categorization techniques can be regarded as a mixture of machine learning methods and advanced document retrieval techniques for document modelling. Documents need to be transformed into vectors of attributes prior to training to construct models or classifiers. A typical solution uses text preprocessing techniques such as stemming and indexing, the processes that turn documents into *Bag of Words* [10]. Feature selection should be conducted so as to reduce computation substantially without damaging the categorization performance. To achieve this, mainly filtering and wrapper approaches are taken into account. The filtering approach determines the features with some measures, and the wrapper approach searches for a subset of attributes more effective to a specific learning algorithm. Examples of the filtering approach are TFIDF, Information Gain(IG) and LSI techniques. For our experimentation, Information Gain(IG) is used as filtering measure in favor of its high accuracy based on entropy reported in various research outcomes [11].

**Table 1.** Classification accuracy(%): MEDLINE collection

| Classification schemes | Generalizers | Document models | | | |
|---|---|---|---|---|---|
| | | 50 bin | 50 wt | 100 bin | 100 wt |
| Single classifiers | k-NN | 63.38 | 63.43 | 63.14 | 61.62 |
| | C4.5 | 68.87 | 70.58 | 68.24 | 67.89 |
| | NB | 76.91 | 74.11 | 76.76 | 74.12 |
| Basic multiple models(vote) | BagC4.5 | 72.51 | 69.80 | 70.49 | 70.59 |
| | BoostC4.5 | 68.48 | 67.76 | 67.99 | 68.09 |
| Basic multiple models(stacking) | StackDS | 33.18 | 30.78 | 31.27 | 30.25 |
| | StackC4.5 | 78.48 | 77.94 | 79.07 | 79.07 |
| | StackNB | 76.78 | 76.42 | 73.92 | 73.48 |
| Hybrid multiple models(vote) | StackedBag | 75.58 | 75.65 | 75.95 | 75.67 |
| | StackedBoost | 75.50 | 75.03 | 75.48 | 75.38 |
| Hybrid multiple models(stacking) | BaggedStack | 75.98 | 74.20 | 76.48 | 75.32 |
| | BoostedStack | 76.80 | 77.17 | 77.68 | 78.35 |

**Table 2.** Classification accuracy(%): USENET news collection

| Classification schemes | Generalizers | Document models | | | |
|---|---|---|---|---|---|
| | | 50 bin | 50 wt | 100 bin | 100 wt |
| Single classifiers | k-NN | 96.94 | 91.94 | 97.47 | 89.47 |
| | C4.5 | 97.29 | 96.59 | 96.88 | 86.70 |
| | NB | 97.65 | 66.65 | 97.06 | 70.00 |
| Basic multiple models(vote) | BagC4.5 | 97.35 | 97.18 | 97.48 | 97.00 |
| | BoostC4.5 | 97.41 | 97.18 | 97.76 | 97.41 |
| Basic multiple models(stacking) | StackDS | 38.94 | 38.88 | 39.00 | 38.88 |
| | StackC4.5 | 97.65 | 96.65 | 97.52 | 96.71 |
| | StackNB | 97.82 | 96.71 | 98.00 | 96.79 |
| Hybrid multiple models(vote) | StackedBag | 97.38 | 96.94 | 97.38 | 97.00 |
| | StackedBoost | 97.42 | 97.32 | 97.42 | 96.92 |
| Hybrid multiple models(stacking) | BaggedStack | 97.40 | 97.08 | 97.50 | 97.50 |
| | BoostedStack | 97.48 | 97.26 | 97.40 | 97.08 |

**Table 3.** Classification accuracy(%): Web document collection

| Classification schemes | Generalizers | Document models | | | |
|---|---|---|---|---|---|
| | | 50 bin | 50 wt | 100 bin | 100 wt |
| Single classifiers | k-NN | 74.30 | 75.80 | 74.62 | 75.21 |
| | C4.5 | 77.03 | 77.55 | 77.88 | 79.05 |
| | NB | 76.97 | 69.88 | 81.33 | 69.49 |
| Basic multiple models(vote) | BagC4.5 | 79.90 | 79.77 | 82.56 | 83.41 |
| | BoostC4.5 | 77.29 | 80.09 | 81.78 | 83.80 |
| Basic multiple models(stacking) | StackDS | 52.37 | 51.01 | 53.03 | 51.59 |
| | StackC4.5 | 78.59 | 77.90 | 82.50 | 79.12 |
| | StackNB | 79.38 | 78.66 | 82.30 | 75.63 |
| Hybrid multiple models(vote) | StackedBag | 78.80 | 78.71 | 78.51 | 78.71 |
| | StackedBoost | 78.49 | 77.78 | 77.82 | 78.38 |
| Hybrid multiple models(stacking) | BaggedStack | 80.15 | 77.29 | 83.98 | 80.06 |
| | BoostedStack | 80.61 | 78.84 | 82.63 | 80.13 |

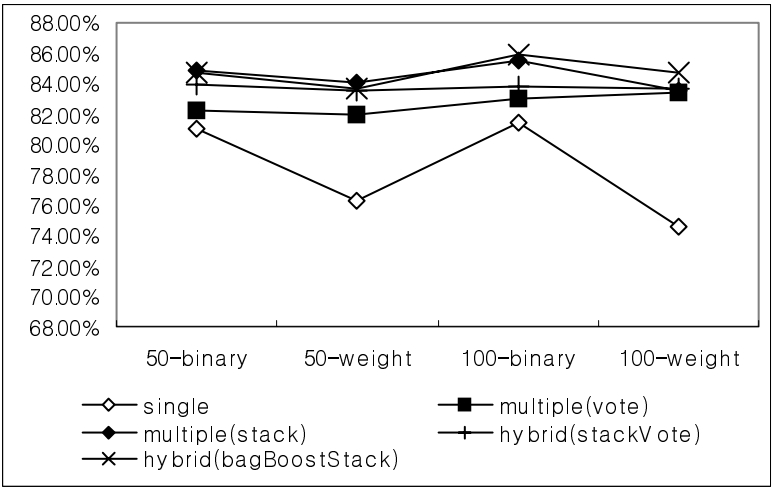


**Fig. 6.** Comparison among different schemes: classification accuracy (%)

The experiments were performed on a computer system equipped with Linux operating system, Intel Pentium IV processor, and 256 MB of memory. Rainbow[4] was used as the tool for text preprocessing and, WEKA[9] for categorization. These programs were modified with Java and C programming languages for the purpose of our research. Since we intend to compare the performance of each of these methods, two standard datasets and a real world data from MEDLINE database were used to evaluate each method. The MEDLINE data collection was obtained as the result of queries posed to MEDLINE database. The size of MEDLINE dataset is 6000 documents with 1000 documents assigned to each of six classes. Usenet news articles collection consists of five classes, each of which also holds 1000 documents. Web collection has 4,518 documents with six classes, and the class distribution varies. Two thirds of each collection was used as training examples and the rest of the data was set aside as test examples.

In order to examine the effect of suitable meta-level learner, the decision stump algorithm known to be an inefficient method is tested as meta algorithm. The category denoted as StackDS contains the results. Table 1, 2, and 3 summarize the results of the experiments to measure and compare the predictive accuracy of multiple classifiers with baseline performance by single classifiers. All of the error estimates were obtained using ten-fold cross validation.

Table 1, 2, and 3 present the performance data of various classifiers and Fig. 6 illustrates the performance of classifiers represented with average cross-collection accuracy. The accuracy rates of StackDS were excluded since this algorithm gives unrealistically pessimistic statistics. It is concluded from the results of decision stump algorithm that the meta algorithm as combining function itself substantially affects the overall performance.

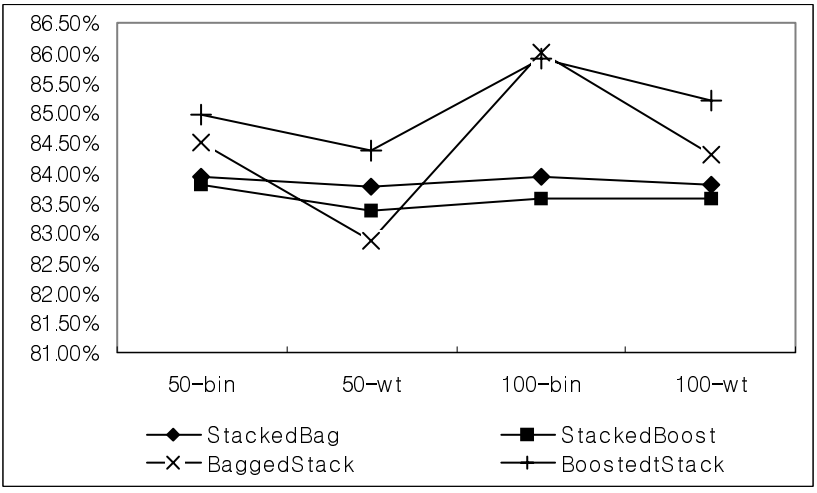


**Fig. 7.** Comparison among hybrid multiple model schemes: classification accuracy (%)

According to the results of the experiments, the methods of document representation developed in IR community has a trifling effect in performance with binary document models showing slightly better performance than weighted models. The size of the feature subset, however, appears to have no correlation with the distribution of accuracy rates. Among the document collection, the Usenet news articles records the highest classification accuracy, apparently owing to discriminating properties of keywords contained in computer newsgroup articles.

The utility of the predictive models derived is in its performance assessed in terms of generalization accuracy. The hybrid schemes show the highest accuracy rate, and the lowest standard deviation in their performance which indicates the stability and reliability of the predictions produced by the models. Single classifiers records 78.36 % of accuracy rate in cross-collection average while standard multiple models show 83.55 %, and the hybrid multiple models 84.30 percent of accuracy rate throughout the experiments. The standard deviation of single classifiers is 0.066 while those for basic multiple classifiers and hybrid classifiers are 0.023 and 0.01 respectively. Our research results in text categorization using multiple classifiers clearly demonstrate that the model combination approaches are more effective than single classifiers.

The experiments prove the utility of meta-learner as a combination function. The only difference between the standard vote algorithms and stacking algorithms is in model combining method. The extended vote algorithms implemented with meta-learning technique demonstrate higher accuracy than the standard methods by 1.09 %. Among the experiments dealing with hybrid algorithms, it is noted that the extended stacking methods like boosted and bagged stacking yield higher accuracy than the extended vote approach as shown in

Fig. 7. This apparently indicates that combining heterogeneous multiple models is effective in reducing the bias component of prediction error. The difference in performance is observed whereas all the algorithms use the same dataset with the same combination function which is meta-learning, and the same base and meta-algorithms. This boils down to a conclusion that the bias effect contributes more to the classification error than the variance resulting from a specific set of training examples does.

## 5 Conclusions

A primary goal in text categorization is to develop a classifier that correctly assigns documents into pre-defined categories. This paper reviews several methods applied to generate efficient classifiers for text documents, compares their relative merits, and shows the utility of the combination of multiple classifiers. Among the multiple classifiers experimented, those derived from the boosted stacking algorithms, that are a kind of the extended stacking algorithms proposed in this study, produce the best results. Our study demonstrates that the variants of the stacking algorithm contribute most to error reduction, mitigating the bias effect from learning algorithms. And meta-learning is proved to be a powerful tool for combining models. We also have found that the perturbation of training data reduces the error component caused by the variance due to the training dataset used for induction.

Our research findings confirm that hybrid model combination approaches improve the existing standard multiple model schemes. All of these findings can be a basis for formulating an inexpensive and efficient text classification system. Even though computation cost increases substantially with the implementation of hybrid multiple schemes, the cost may be justified by the enhanced reliability, stability and improved classification accuracy of the new approaches. We conclude that the techniques to address the problems of variance of training data and bias of learning algorithms can be implemented within a single classification system.

## References

1. Kjersti A. and Eikvil L.: Text categorization: a survey. (1999)
2. Apte et al.: Automated learning of decision rules for text categorization. ACM Transactions on Information Systems, Vol.12, No.3, (1994) 233–251
3. Mladeni'c, D. and Grobelnik, M.: Efficient text categorization. In Text Mining workshop on the 10th European Conference on Machine Learning ECML-98 (1998)
4. Mitchell, Tom: Machine learning. New York: McGraw-Hill (1997)
5. Bauer, Eric and Ron Kohavi.: An empirical comparison of voting classification algorithms: bagging boosting and variants. Machine Learning Vol.36, (1999) 105–142
6. Breiman, Leo: Bagging predictors. Machine Learning Vol.24 (1996) 49–64
7. Schaphire, Robert E.: Theoretical views of boosting. Proceedings of the European Conference on Computational Learning Theory EuroCOLT-99 (1999)

8. Wolpert, D. and Macready, W.: Combining stacking with bagging to improve a learning algorithm. Technical report. Santa Fe Institute (1996)
9. Witten, Ian H. and Eibe, Frank: Data Mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufman (2000)
10. Hong, Se June and Sholom M. Weiss: Advances in predictive model generation for data mining. IBM Research Report RC-21570 (1999)
11. Salton, Gerard: Introduction to information retrieval. McGraw-Hill (1983)
12. Yang, Yiming and Jan O. Pedersen: A comparative study on feature selection in text categorization. Proceedings of the 4th International Conference on Machine Learning (1997)