

# Rule Based Base Classifier Selection for Bagging Algorithm

A B M Shawkat Ali<sup>1</sup>, Ben Pang<sup>2</sup> and Kevin Tickle<sup>4</sup>

<sup>1</sup>School of Computing Sciences, Central Queensland University, Rockhampton, QLD 4702, Australia

<sup>2</sup>Australian Bureau of Statistics, Belconnen ACT 2617, Australia.

**Abstract** - Bagging is a popular method that improves the classification accuracy for any learning algorithm. A trial and error classifier feeding with the Bagging algorithm is a regular practice for classification tasks in the machine learning community. In this research we propose a rule based method using statistical information for unique classifier selection. The generated rules are verified using 113 classification problem with cross validation approach. That makes Bagging is a computationally faster algorithm and optimal solution for classification performance.

**Keywords:** Bagging, Classification, rule based method.

## 1. Introduction

Data Mining is getting popularity rapidly due to their expert knowledge extraction process from a huge database. To extract knowledge data mining user using machine learning technique. Now-a-days there have many techniques are available for public use. But better performed techniques are well popular among these lists. Bagging [1,2] a sobriquet for Bootstrap aggregating is one of the well established technique in the machine learning community for improving the performance of any learning algorithm. It does re-sampling training sets from the original data set to the learning algorithm which builds up a base classifier for each training set [3]. Bagging uses a voting technique which is unable to take into account the heterogeneity of the instance space. The philosophy is when majority of the base classifiers give a wrong prediction for a new instance then the majority vote will result in a wrong prediction [4]. Therefore the base classifier selection is a critical issue for Bagging. This research will propose a solution for Bagging on this issue.

First we classify 113 problems by boosting using different classifier and rank the classifier performance. After that we use statistical central tendency measure for these datasets to construct a data characteristics matrix. Now we add an additional attribute towards the end with the data characteristics, which explain the classifier identity. Finally we use decision tree algorithm to find out which classifier is the best suited for Bagging algorithm

for a specific problem. The solution came out as a set of rules.

The rest of the paper is organized as follows: first we provide a brief description of Bagging and some popular base classifiers. After that we summarize the experimental outcome of our current research. We conclude our research with a discussion of the limitations and future prospects of our research.

## 2. Algorithm Description and Experimental Setup

The following section will provide a brief description of each algorithm.

**Bagging:** Bagging is an ensemble method for improving unstable estimation or classification schemes. It has attracted much attention due to its easy formulation and the popularity of the bootstrap methodology.

Let us consider the data matrix  $(X_i, Y_i)$  ( $i = 1, \dots, n$ ), where  $X_i \in \mathbb{R}^d$  denotes the  $d$ -dimensional predictor variable and the response for classification  $Y_i \in \{0, 1, \dots, J-1\}$ , where  $J$  is the number of classes. The target multivariate function of interest is  $P[Y = j | X = x]$  ( $j = 0, \dots, J-1$ ) for classification task. We suppose  $X_e$  is the instance that we need to classify.

We define  $C_i(X)$  as the function that will be used to convert the classifier  $C_i^*(X)$  to the dummy variables in the  $i^{\text{th}}$  iteration of the bagging.

$$C_i(X) = \begin{cases} = 1 & \text{if } C_i^*(X) = y \\ = 0 & \text{otherwise} \end{cases}$$

Bagging algorithm works in the following three steps:

**Step 1:** For the  $i^{\text{th}}$  iteration, we first construct a bootstrap sample  $(X_1^*, Y_1^*), \dots, (X_n^*, Y_n^*)$  by randomly drawing  $n$  times with replacement from the data  $(X_1, Y_1), \dots, (X_n, Y_n)$ .

**Step 2:** Based on the data obtained in the step 1, we obtain the classifier  $C_i^*(X)$ .



**Step 3:** Now, repeat steps 1 and 2  $M$  times. The final classifier will be:

$$C^f(X_e) = \arg \max_{y \in Y} \sum_{i: C_i(X_e)=y} C_i(X_e)$$

This is the basic formulation of Bagging algorithm [5].

We choose four classifiers, namely REPTree (Reduced Error Pruned Tree), IBK, NaïveBayes and PART as a base classifier for the Bagging algorithm. All these classifiers are available in the WEKA [6] implementation. WEKA is a Java based machine learning tools. The REPTree is a default classifier with Bagging in WEKA. The following section provides a brief explanation about the four base classifiers.

**REPTree:** A decision tree is a tool for carrying out classification of data instances input to it. Decision trees have production rules of the type IF – THEN (IF feathers = 'yes' THEN Animal = 'bird') [7]. RepTree is a fast decision tree learner and builds a decision/regression tree using information gain/variance reduction and prunes. It uses reduced-error pruning with backfitting. Only sorts values for numeric attributes once. Since this a fast algorithm so the pruned tree reduces the complexity in the classification process. Moreover pruning is used to find the best sub-tree of the initially growntree with the minimum error for the test set [8].

**IBK:** Instance Based Learning IBK is a very commonly used classification method with the exception that it is possible to define the number of nearest neighbours is considered in the K-nearest neighbour component of the algorithm. It works on the principle that first plot each training instance and then measure the distance of each test instance to the training instances. The class of the training instance with the least distance between it and the test instance is the class that we assign to the test instance. Basically  $k$  is chosen to be an odd number, and we take the smallest average distance of the  $k$  instances [9].

**NaiveBayes:** Naive Bayes is based on the well-known Bayes Theorem. It is termed 'naïve' because it assumes that attributes of the training set are conditionally independent and that the prediction procedure is not influenced by any hidden or latent attributes. It works by calculating the maximum posterior probability of each class [10, 11].

**PART:** Part is developed from the C4.5 and RIPPER algorithms and is a partial decision tree algorithm. However, unlike C4.5 and RIPPER, PART does not have to perform global optimization in order to generate rules [11].

We fed all these base algorithms with Bagging to classify a wide range of problems. First, we fed each classifier one-by-one with Bagging and kept a record of the classification performance for the 113 problems. We selected all data sets from two different data repositories [12, 13]. All classification problems descriptions are available in Appendix I. We chose ten-fold cross validation over the experiment. Then we collected the

descriptive statistical information about each of the 113 classification problems. The list of descriptive statistics is follows:

Statistical Name	Symbolic Name
<i>mean</i>	<i>m</i>
<i>standard deviation</i>	<i>std</i>
<i>skewness</i>	<i>s</i>
<i>kurtosis</i>	<i>k</i>

The explanations of these descriptive statistical terms are available in any statistical text book. Moreover one can find the implementation in the Statistical Toolbox in Matlab [14].

We constructed a data matrix with these statistical information's and the name of the best algorithm performance. Then we employed the C5.0 [15] algorithm to generate the rules. These rules have been considered to select a unique classifier for Bagging algorithm to classify any problem with better accuracy and faster computation.

### 3 Experimental Results

We observed from the experiment that the PART classifier is the best choice for the Bagging algorithm and it shows the highest percentage of average accuracy for the 113 problems. However, in terms of computational complexity REPTree is the best choice among the four classifiers.

The rules were generated using the C5.0 decision tree algorithm to select a unique classifier for the Bagging algorithm. C5.0 has two parameters, pruning confidence ( $c$ ) and minimum cases ( $m$ ). Pruning confidence affects error estimation and therefore how severely the tree may be pruned; a smaller value of  $c$  enables more pruning and a higher value less pruning. Minimum cases affect how the tree fits the data; a higher value of  $m$  allows more pre-pruning [15]. We tuned both parameters to produce the best rule and found the best suited values for  $c$  is 99 and  $m$  is 2. The generated rules were verified by ten-fold cross validation and the percentage of accuracy is summarized with the rules. These rules are as follows:

#### 3.0.1 Rules for REPTree Classifier

Rule 1: IF  $m > 2.1766$  &  $std \leq 14.396$  &  $s \leq 1.3015$  &  $3.7487 < k \leq 5.2539$  THEN select REPTree Classifier for Bagging Algorithm.

Rule Accuracy = 64%

#### 3.0.2 Rules for NB Classifier

Rule 2: IF  $s > 1.25$  &  $5.2539 < k \leq 6.2028$  OR  $1.3626 < k \leq 1.9518$  OR  $s \leq 0.94824$  THEN select NB Classifier for Bagging Algorithm.

Rule Accuracy = 87.57%



### 3.0.2 Rules For IBK Classifier

Rule 3: IF std > 14.396 & 2.983 < k <= 5.2539 OR std > 8.4814 & s > 0.20412 & k <= 2.4958 OR m <= 2.1766 & s > 0.94824 & 2.983 < k <= 5.2539 OR 6.2028 < k <= 6.5193 THEN select IBK Classifier for Bagging Algorithm.

Rule Accuracy = 80%

### 3.0.4 Rules for PART Classifier

Rule 4: IF s <= 0.28279 & k <= 1.3626 OR m > 50.557 OR k > 6.5193 THEN select PART Classifier for Bagging Algorithm.

Rule Accuracy = 82.14%

The default classifier of this approach is NB, since it shown the highest accuracy among the generated rules. That means if any data set does not satisfy the above rules then state way we suggest to select the NB classifier for the existing problem.

## 4 Conclusions

This research contributes a new approach to selecting a unique classifier for the Bagging algorithm. A rule based approach has been introduced for the unique classifier selection. These rules are generated based on descriptive statistical information of 113 classification problems. All generated rules showed higher accuracy during the ten-fold cross validation except for the REPTree classifier. REPTree showed the best classification performance for only a few data sets. This performance could be increased by considering more classification problems. We have planned to extend our research using more problems from different domains with a variety of classifiers.

## 5 References

- [1] Breiman, L. "Bagging Predictors". Machine Learning, vol. 24 (1996) 123-140.
- [2] Bauer, E., Kohavi, R. "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants". Machine Learning, vol.36 (1999) 105-139.
- [3] Webb, G.I., MultiBoosting. "A Technique for Combining Boosting and Wagging". Machine Learning, 40, 159-196, 2000.
- [4] Tsymbal, A., Puuronen, S. "Bagging and boosting with dynamic integration of classifiers". Principles of Data Mining and Knowledge Discovery, Proc. PKDD 2000.
- [5] Jerome H. Friedman, "Stochastic gradient boosting", Computational Statistics & Data Analysis, v.38 n.4, p.367-378, 2002.
- [6] Witten I.H. and Frank, E. "Data Mining: Practical Machine Learning Tools and Techniques". Morgan Kaufmann, 2005.
- [7] J. Quinlan, "C4.5: Programs for Machine Learning". Morgan Kaufmann, 1993.
- [8] Park, J. Hsiao-Rong, T. and Kuo, C.-C.J. "GA-Based Internet Traffic Classification Technique for QoS Provisioning". IEEE Proceedings of the International Conference on Intelligent Information Hiding and Multimedia Signal, pp. 251-254. 2006.
- [9] Aha, D., and D. Kibler. "Instance-based learning algorithms". Machine Learning, vol.6, pp.37-66, 1991.
- [10] G.H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, CA, pp. 338-345, 1995.
- [11] S. Ali and K.A. Smith, "On Learning Algorithm Selection for Classification," Applied Soft Computing, Elsevier Science, vol.6, no. 2, pp.119-138, 2006.
- [12] C. Blake and C.J. Merz, "UCI Repository of machine learning databases". University of California, Irvine, CA, 2002. <http://www.ics.uci.edu/mllearn/mlrepository.html>.
- [13] T. S. Lim, "Knowledge Discovery Central datasets". 2002, <http://www.kdcentral.com/>.
- [14] Statistics Toolbox User's Guide, version 3, The MathWorks Inc., USA, 2001. URL: <http://www.mathworks.com>.
- [15] See 5: <http://www.rulequest.com/see5-info.html>, Accessed: 02 February 2007.

## Appendix I: Datasets description.

# Data set	Data set name	# Instances	# Attributes	# Class
1	abalone	1253	9	3
2	adp	1351	12	3
3	adult+stret	20	5	2
4	adult-stret	20	5	2
5	allbp	840	7	3
6	ann1	1131	7	3
7	ann2	1028	7	3
8	aph	909	19	2
9	art	1051	13	2
10	australian	690	15	2
11	balance-sca	625	5	3
12	bcw	699	10	2
13	bcw_noise	683	19	2
14	bld	345	7	2
15	bld_noise	345	16	2
16	bos	910	14	3
17	bos_noise	506	26	2
18	breast-canc	286	7	2
19	breast-canc	699	10	2
20	bupa	345	7	2
21	c	1500	16	2
22	cleveland-heart	303	14	5
23	cmc	1473	10	3
24	crx	490	16	2
25	dar	1378	10	5

26	dhp	1500	8	2
27	DNA-n	1275	61	3
28	dna	2000	61	3
29	dna noise	2000	81	3
30	dph	590	11	2
31	echocardiogram	131	8	2
32	flare	1389	11	2
33	german	1000	25	2
34	glass	214	10	6
35	h-d	303	14	2
36	hayes-roth	132	6	3
37	hea	270	14	2
38	hea noise	270	21	2
39	heart	270	14	2
40	hepatitis	155	20	2
41	horse-23	368	23	2
42	horse-colic	368	28	2
43	house-votes-84	435	17	2
44	hyp	2847	16	2
45	hypothyroid	1265	26	2
46	iris	150	5	3
47	khan	1063	6	2
48	kr-vs-kp	1279	37	2
49	labor-neg	40	17	2
50	led-noise	1047	10	10
51	lenses	24	6	3
52	letter-a	1334	17	2
53	lung-cancer	32	57	2
54	lymphography	148	19	8
55	mha	1269	9	4
56	monk1	556	7	2
57	monk2	601	7	2
58	monk3	554	7	2
59	mushroom	1137	12	2
60	nettalk str	1141	8	5
61	page-blocks	1149	11	5
62	pendigits-8	1399	17	2
63	pha	1070	10	5
64	phm	1351	12	3
65	phn	1500	10	2
66	pid	532	8	2
67	Pima	768	9	2
68	poh	527	12	2
69	post-operative	90	9	3
70	primary-tum	339	18	2
71	pro	1257	13	2
72	promoter	106	58	2
73	pvro	590	19	2
74	rph	1093	9	2
75	satimage	1351	11	6
76	shuttle-landing control	15	7	2
77	sick-euthyroid	1582	16	2
78	sma	409	8	4
79	smo	1855	9	2
80	smo noise	1855	16	2
81	sonar	208	61	2
82	splice	1589	61	3

83	t series	62	3	2
84	tae	151	6	3
85	tae noise	151	11	2
86	thy	1887	22	3
87	thynoise	1132	11	3
88	tic-tac-toe	958	10	2
89	titanic	2201	4	2
90	tmris	100	4	2
91	tqr	1107	12	2
92	trains-transformed	10	17	2
93	va-heart	200	9	4
94	veh	846	19	4
95	veh noise	761	31	4
96	vehicle	658	20	0
97	votes noise	391	31	2
98	waveform	5000	22	2
99	waveform noise	5000	41	2
100	wdbc	569	31	2
101	wine	178	14	3
102	wdbc	199	34	2
103	xaa	94	19	4
104	xab	94	19	4
105	xac	94	19	4
106	xad	94	19	4
107	xae	94	19	4
108	xaf	94	19	4
109	xag	94	19	4
110	xah	94	19	4
111	xai	94	19	4
112	yha	1601	10	2
113	zoo	101	17	7



Copyright © 2008 CSREA Press  
ISBN: 1-60132-060-4, 1-60132-061-2 (1-60132-062-0)  
Printed in the United States of America