

Empirical Comparisons of Various Voting Methods in Bagging

Kelvin T. Leung, D. Stott Parker
UCLA Computer Science Department
Los Angeles, California 90095-1596
{kelvin,stott}@cs.ucla.edu

ABSTRACT

Finding effective methods for developing an ensemble of models has been an active research area of large-scale data mining in recent years. Models learned from data are often subject to some degree of uncertainty, for a variety of reasons. In classification, ensembles of models provide a useful means of averaging out error introduced by individual classifiers, hence reducing the generalization error of prediction.

The plurality voting method is often chosen for bagging, because of its simplicity of implementation. However, the plurality approach to model reconciliation is ad-hoc. *There are many other voting methods to choose from*, including the anti-plurality method, the plurality method with elimination, the Borda count method, and Condorcet's method of pairwise comparisons. Any of these could lead to a better method for reconciliation.

In this paper, we analyze the use of these voting methods in model reconciliation. We present empirical results comparing performance of these voting methods when applied in bagging. These results include some surprises, and among other things suggest that (1) plurality is not always the best voting method; (2) the number of classes can affect the performance of voting methods; and (3) the degree of dataset noise can affect the performance of voting methods. While it is premature to make final judgments about specific voting methods, the results of this work raise interesting questions, and they open the door to the application of voting theory in classification theory.

1. INTRODUCTION

Data mining is about knowledge or model extraction from raw data. It can potentially lead to a model of the underlying data that allows us to make non-trivial predictions on new data. Models can often be treated as high-level abstractions of raw data obtained by the data mining process (i.e. learning algorithm) with a mining objective. Those models are often subject to various degrees of uncertainty. Depending on the mining objective, an infinite number of models could potentially result from a data mining task. In ad-

dition, there are many techniques to define interesting patterns or model (knowledge) representations in a data mining process. They all depend on mining objectives, underlying assumptions and hypotheses. Unfortunately, no single methodology can reliably produce an accurate model about the underlying data. Recently ensembles of models have been investigated as a mechanism for averaging out the generalization error of individual classifiers. This paper reports on our experiments into the error performance of ensemble methods using model reconciliation processes based on various voting methods.

1.1 Definition of Model Reconciliation

Model Reconciliation is a general process of combining classification models. Abstractly speaking, a model is a form of subjective opinion and a classifier is an individual expert. Multiple opinions can then be presented to a black box that dictates a group opinion by using some selection or combination technique. Figure 1 depicts the process of model reconciliation.

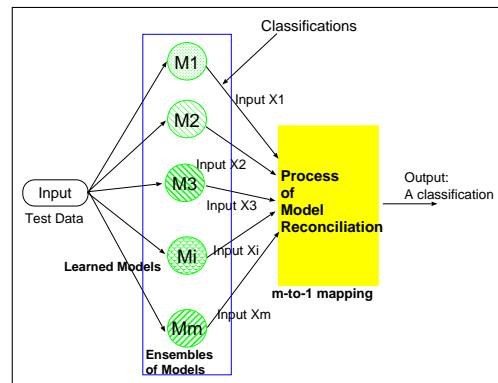


Figure 1: Process of Model Reconciliation

The process of model reconciliation can be regarded as a m -to-1 mapping from m experts' opinions to an output opinion. Given m experts, an opinion vector is a vector containing the output of m experts that depicts the distribution of the candidate classes (i.e. a poll). For example, a k -class- m -expert opinion vector is

$$V(k, m) = \langle C_1, C_2, \dots, C_i, \dots, C_m \rangle.$$

where each $C_i \in \{0, 1, \dots, k-1\}$ is one of the k known classes. In addition, the set of all possible opinion vectors C^m is called the **classification opinion space**, and $reconcile_c$:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGKDD '03, August 24-27, 2003, Washington, DC, USA
Copyright 2003 ACM 1-58113-737-0/03/0008 ...\$5.00.

$C^m \rightarrow C$ is a rule of model reconciliation. Given an opinion vector V , $\text{reconcile}_c(V)$ is the reconciled classification.

1.2 Existing Ensemble Methods use the Plurality Voting Method

The standard plurality voting method has been universally adopted in ensemble methods such as Boosting [8] and Bagging [4]. This is probably explained by its familiarity and simplicity of implementation. In ensemble methods, the process of model reconciliation is then essentially equivalent to an election, where each classifier casts one vote in support of its most preferred class. At the end of the election, the plurality voting method yields as output the input class having the greatest final tally. In case of a tie, the output class is chosen arbitrarily. However, the outcome of an election using the plurality voting method need not match the voters’ true preferences. It raises up a concern about the choice of voting method used in voting-based model reconciliation.

EXAMPLE 1. *Suppose a classification includes three classes, X , Y , and Z . An ensemble consisting of 15 classifiers is used to decide an output classification. The classifiers’ preferences on classification are as follows (where \succ means “is preferred over”):*

| Preference Order | # of Classifiers |
|---------------------|------------------|
| $X \succ Z \succ Y$ | 6 |
| $Y \succ Z \succ X$ | 4 |
| $Z \succ Y \succ X$ | 5 |

Table 1: Preferences of the 15 Classifiers

Using the plurality voting method, the winner is X , since $(X \succ Z \succ Y)$ wins with a $(6 : 5 : 4)$ tally. On the other hand, X receives only 6 votes, which is less than half of the total votes. Is X really the preferred choice of the ensemble?

With the Borda Count voting method, on the other hand, the outcome changes as shown in Table 2. The Borda Count method gives greater weight to more preferred classes. In a 3-class classification task using the Borda Count method, the most preferred class receives 2 points and the second most preferred class receives 1 point. The class with the greatest final point tally is declared the winner.

| Preference | Classifiers | X tally | Y tally | Z tally |
|---------------------|-------------|--------------|--------------|--------------|
| $X \succ Z \succ Y$ | 6 | 6×2 | | 6×1 |
| $Y \succ Z \succ X$ | 5 | | 4×2 | 4×1 |
| $Z \succ Y \succ X$ | 4 | | 5×1 | 5×2 |
| Total | 15 | 12 | 13 | 15 |

Table 2: Tally with the Borda Count voting method

Table 2 depicts the tally of the election using Borda count method and the Borda count outcome is that Z is the winner with a $(15 : 13 : 12)$ point tally. It is significant that the plurality outcome X is identified as the least preferred class by the Borda count method. This example illustrates a basic controversy about which candidate class should be selected as the output of the ensemble classification. The controversy has served as a basic motivation behind our study of the effect of using various voting methods in model reconciliation.

Ironically, mathematicians and voting theorists have been investigating voting methods over the last two centuries. It

is well-known that when there are at least three candidates, the winner need not be the most preferred candidate.

2. VOTING METHODS

Voting is a well-known aggregation procedure for combining opinions of voters in order to determine a consensus, an agreement on a given issue, within a given time frame. In short, voting methods can be divided into two types, namely 1) *Preferential voting methods* and 2) *Non-preferential voting methods* [6, 5, 7]. The distinction between these two is that the former uses information from a preference schedule while the latter does not. Here a **preference schedule** is a table that summarizes the results of an election by giving the tally of how many voters cast ballots for each preference ordering. The **Plurality method (P1)**, **Anti-plurality (Anti-P1) method**, **Plurality method with Elimination (P1-Elm)**, the **Borda Count (BC) method**, and the **Method of Pairwise Comparisons (PC)** are considered to be preferential (positional) voting methods. The **Approval Voting method** is considered to be a non-preferential voting method. This paper studies the performance of bagging with preferential voting methods.

2.1 Preferential voting methods

To formalize model reconciliation using different voting methods, we must first define an **expert’s preference (ranking) order** and the **expert preference profile**.

DEFINITION 1. *An expert’s preference order is defined as a ranking order of candidate class, from highest to lowest preference, given by an expert. Each expert, indexed by i from 1 to m , has a highest-to-lowest ranking order \succ_i on the $k \geq 2$ candidate classes in the nominee set C . We assume each \succ_i is a total order with no ties or indifference.*

DEFINITION 2. *The expert preference profile is defined as the list of experts’ preference orders $(\succ_1, \dots, \succ_m)$.*

EXAMPLE 2. *For a 3-class-15-expert model reconciliation problem where $C \in \{C_1, C_2, C_3\}$ is the set of possible class labels, there are $3! = 6$ (strict, transitive preferences) types of expert.*

Typically each expert can cast one vote for one of these types. In order to rank our candidate classes, we need to tabulate the preference ballots into a preference schedule using a voting vector. Different voting methods would assign different points (weights) to candidates based on their ranked positions. The intended goal of each method is to capture the voter’s “true beliefs.”

DEFINITION 3. *A voting vector $\mathbf{w} \in \mathbb{R}^k$ is a $k \times 1$ vector*

$$\mathbf{w} = \langle w_1, w_2, \dots, w_{k-1}, 0 \rangle^T$$

and $w_1 > 0, w_1 \geq w_2 \geq w_3 \geq \dots \geq w_{k-1} \geq w_k = 0$.

A voting vector is used to assign w_j points to an expert’s j th ranked candidate and rank the candidates according to the sum of received points in a preference schedule.

DEFINITION 4. *The preference schedule can be expressed as a preference profile matrix A . The rows of this matrix are indexed by the candidate classes of the nominee set $C \in \{C_1, C_2, \dots, C_k\}$ and the columns of the matrix are indexed*

by the different preference orderings depicted in the expert preference profile, ranging from Type 1 to Type $k!$, which is the maximum number of distinct (strict, transitive) types of preference for a nominee set with k classes. As defined by the voting vector \mathbf{w} , let α_{ij} denote the points (weights) assigned to the i -th ranked candidate by the j -th type of preference. Thus, the preference profile matrix, $A = (\alpha_{ij})$, is an $k \times k!$ matrix as shown below.

| | \succ_1 | \succ_2 | \succ_3 | ... | \succ_j | ... | $\succ_{k!}$ |
|----------|-----------|-----------|-----------|----------|---------------|----------|----------------|
| C_1 : | 1 | 1 | 0 | ... | . | ... | . |
| C_2 : | 0 | 0 | 0 | ... | . | ... | . |
| C_3 : | 0 | 0 | 1 | ... | . | ... | . |
| \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots |
| C_i : | . | . | . | ... | α_{ij} | ... | . |
| \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots |
| C_k : | . | . | . | ... | . | ... | $\alpha_{kk!}$ |

DEFINITION 5. The **scoring vector** \mathbf{s}

$$\mathbf{s} = \langle s_1, s_2, \dots, s_i, \dots, s_{k!} \rangle^T$$

is a $k! \times 1$ vector giving the total number of votes received for each type of preference order by the voters. For example, $\mathbf{s} = \langle 0, 6, 0, 5, 0, 4 \rangle^T$ is the scoring vector for Example 1.

Notice that, the scoring vector is similar to the opinion vector (or poll, described in section §1.1) that depicts the distribution among the candidates.

DEFINITION 6. The **tally vector** \mathbf{t}

$$\mathbf{t}_{k \times 1} = A_{k \times n} s_{n \times 1}$$

gives the total number of votes (points) received by each candidate class.

2.1.1 Plurality voting method

The Plurality method is a commonly used method in voting algorithms such as Boosting and Bagging. By definition, a candidate class with a plurality of the votes has more votes than any other candidate classes. The winner does not have to receive a majority of the first place votes. For a k -class classification task, the voting vector

$$\mathbf{w}^k = \langle w_1, w_2, \dots, w_k \rangle^T$$

satisfies $w_1 = 1, w_2 = w_3 = \dots = w_k = 0$.

The rule of model reconciliation is to select an output class $x \in C$ from nominee set $C = \langle C_1, C_2, \dots, C_k \rangle$ such that

$$x = \operatorname{argmax}_i \mathbf{t}(i) \quad \text{where } \mathbf{t} = A \mathbf{s}.$$

2.1.2 Anti-Plurality voting method

The Anti-Plurality method requires each expert to *vote against his/her bottom ranked candidate*. For a k -class classification task, the voting vector

$$\mathbf{w}^k = \langle w_1, w_2, \dots, w_{k-1}, 0 \rangle^T$$

satisfies $w_1 = w_2 = w_3 = \dots = w_{k-1} > w_k = 0$.

For example, $\mathbf{w}^k = \langle w_1, w_2, \dots, w_{k-1}, 0 \rangle^T$ where $w_1 = w_2 = w_3 = \dots = w_{k-1} = 1$. Like the plurality voting method, the rule of model reconciliation is to select an output class $x \in C$ from the nominee set $C = \langle C_1, C_2, \dots, C_k \rangle$ such that

$$x = \operatorname{argmax}_i \mathbf{t}(i) \quad \text{where } \mathbf{t} = A \mathbf{s}.$$

2.1.3 Borda Count voting method

Borda Count Method was proposed by Jean-Charles de Borda [6]. It is a positional-scoring procedure. For the Borda Count Method, each candidate class gets 0 for each last place vote received, 1 point for each next-to-last point vote, etc., up to $k-1$ points for each first place vote (where k is the number of candidate classes). Thus, the voting vector

$$\mathbf{w}^k = \langle w_1, w_2, \dots, w_{k-1}, 0 \rangle^T$$

has $w_1 = (k-1), w_2 = (k-2), \dots, w_{k-2} = 2, w_{k-1} = 1$. The candidate class with the largest point total wins the election. The Borda Count Method, or some variation of it, is often used for things like polls which rank sporting teams or academic institutions.

The rule of model reconciliation using the Borda Count voting method is to select an output candidate class $x \in C$ from the nominee set $C = \{C_1, C_2, \dots, C_k\}$ so as to maximize the votes (points) in the resultant tally vector:

$$x = \operatorname{argmax}_i \mathbf{t}(i) \quad \text{where } \mathbf{t} = A \mathbf{s}.$$

2.1.4 Plurality with Elimination voting method

Plurality with elimination is carried out in rounds. After each round of voting, the candidate with the fewest first place votes is eliminated and a new round of voting begins with the remaining candidates. When only two candidates remain in a round, the candidate with the most votes (plurality method) wins the election. For an N candidate election, the method of plurality with elimination requires $N-1$ rounds. The following depicts the algorithm for model reconciliation with plurality with elimination method.

Algorithm for Plurality with Elimination:

Inputs: Nominee set, $C \in \{C_1, C_2, \dots, C_k\}$, Expert preference profile, $(\succ_1, \succ_2, \dots, \succ_j, \dots, \succ_{k!})$, Voting vector, \mathbf{w}^k .

Output: a candidate class x that belongs to the nominee set $C \in \{C_1, C_2, \dots, C_k\}$

1. Initialize $\mathbf{w}^k = \langle w_1, w_2, \dots, w_{k-1}, w_k \rangle^T$ where $w_1 = 1$ and $w_2 = \dots = w_{k-1} = w_k = 0$
2. Generate the *scoring vector* \mathbf{s} .
3. Compute the *preference profile matrix* A_i using the voting vector \mathbf{w}^k and \succ_j .
4. For $i = 0$ to $k-2$ {
5. $j = \#C$
6. For $x = 1$ to j {
7. For $y = 1$ to $k!$ {
8. $A_i(x, y) = w^{k-i}(\succ_y(x))$
9. }
10. }
11. Compute the tally vector $\mathbf{t}_i = A_i \mathbf{s}$
12. Obtain $x = \operatorname{argmin}_l \mathbf{t}_i(l)$
13. Update $C = C \setminus x$
14. Initialize \mathbf{w}^{k-i}
15. }
16. Select $x \in C$ such that $x = \operatorname{argmax}_l \mathbf{t}_{k-2}(l)$.

Output: a candidate class $x \in C$ such that $x = \operatorname{argmax}_l \mathbf{t}_{k-2}(l)$.

2.1.5 Condorcet’s Pairwise Comparison voting method

In Condorcet’s method of Pairwise Comparisons [5], each candidate is matched head-to-head (one-on-one) with each of the other candidates. Each candidate gets 1 point for a one-on-one win and a half a point for a tie. The candidate with the most total points is the winner. In a k -candidate election, there are $\frac{k}{2}$ head-to-head matchups.

The Method of Pairwise Comparisons was explicitly designed to satisfy the fairness criterion called the Condorcet Criterion (see section § 3.5). As stated in [7] and some other literature, profiles with *cyclic majorities* or *Condorcet’s phenomenon* would have no majority candidate. For example, if the voters’ orders on $X = \{a, b, c\}$ are $a \succ b \succ c$, $c \succ a \succ b$, $b \succ c \succ a$ with $n = m = 3$, then every candidate loses to one of the other two under majority comparison. Also, there are profiles with a majority candidate x who will not be elected by *any Borda-type* positional-scoring method [7]. For example, suppose $n = 7$, $X = \{a, b, x\}$, and the final tally is

| | |
|---------|-----------------------|
| 3 votes | $x \succ a \succ b$ |
| 2 votes | $a \succ b \succ x$ |
| 1 vote | $a \succ x \succ b$ |
| 1 vote | $b \succ x \succ a$. |

Since x has a 4-to-3 (pairwise comparison) majority over a and b , if the voting vector $\mathbf{w}^3 = \langle w_1, w_2, w_3 \rangle^T$ where $w_1 > w_2 > w_3$, a always gets more points ($3w_1 + 3w_2 + w_3$) than the ($3w_1 + 2w_2 + w_3$) obtained by x using the Borda method [7]. The Condorcet Criterion addresses the fairness of declaring a candidate the winner even when some other candidate wins all possible head-to-head matchups. With the Method of Pairwise Comparisons, any candidate who wins all possible head-to-head matchups will have a higher point total than any other candidate and thus be the winner.

3. EXPERIMENTS AND RESULTS

Table 3 and 4 below summarize the generalization error performance of bagging using each of the voting methods introduced earlier. In this section we summarize several outcomes that have emerged from these experiments, some of which are surprising and merit further investigation.

3.1 The Data

We used 39 UCI datasets [2] for our experiments. We give their sizes, number of training instances used, number of test instances used and numbers of attributes and classes in Table 3.

To evaluate the generalization error of the various preferential voting methods over all datasets in bagging, we divided each dataset roughly into three parts where two-thirds of the datasets is used as training samples to produce 10 random bootstraps and the remaining one-third of the datasets is used for testing. The average size of the bootstraps is given in Table 3. This procedure was carried out ten times for each dataset and the average results were recorded in Table 3. Our choice of data partitioning strategy may seem arbitrary, but was primarily due to the small size of some of the datasets.

We used the `tree` [3] package in R [9] to create the base classifier for each bootstrap. By randomly selecting different values of *deviance* and *gini* for the splitting criterion, classification trees with various confusions were introduced

in the ensemble. In testing, for every unseen test instance, we obtained a tally vector of each voting method considered and compared the outcome of each voting method with the correct classification to determine generalization error. The generalization error is defined as

$$error = \left(1 - \frac{\text{no. of correctly classified test instances}}{\text{total no. of test instances used}}\right)$$

3.2 Empirically, plurality is not always the best voting method

Table 4 is a digest of Table 3. It gives, for each voting method, the total number of wins and losses (i.e., the number of times the voting method had the least, and the greatest, generalization error) over all test datasets.

Table 4 also gives the win/loss ratio for each voting method. This ratio represents an overall measure of performance of each method. It is certainly not the only viable measure here, but by penalizing loss it does give us an indication of the *risk* involved in adopting a given voting method, or (looking at it inversely) a measure of *reliability* or *trustworthiness* of the method.

Plurality voting is the method typically used in the model reconciliation literature. A surprise of this effort is that, over all datasets considered here, plurality has exhibited inferior performance — both with respect to win/loss ratio and to other measures. This evidence suggests strongly that *voting methods other than plurality have a role to play in model reconciliation, and deserve greater attention from the data mining community*. An intriguing question here is whether there is some stacking scheme for voting methods.

3.3 The number of classes matters

The sixth column of Table 3 shows the number of classes in each classification problem dataset considered here. From these it is possible to see that only 8 are two-class problems, while the others involve three or more classes.

Two-class problems are discrimination problems, and therefore qualitatively different from multi-class problems. Empirically, some voting methods perform better when there are only two candidate classes. For example, in Table 3 Anti-plurality and Plurality with Elimination perform better for two-class problems when there are missing attributes in the dataset.

Often preference can be important additional information in problems that involve more than two classes. Voting methods that consider preference among classes can therefore gain an advantage over methods that do not, such as plurality.

3.4 Noise matters

The *led* datasets in Table 3, for example, shows great sensitivity of the voting methods to noise. The amount of noise in *led* datasets increases from 0% to 50%.

One somewhat surprising outcome of the experiments, plainly visible in the table, is that the Borda Count method performed better in the presence of noise than the other methods. We also obtained the same outcome with other artificial datasets, giving the general impression that the Borda Count method is more resilient to noise than other methods.

While this hypothesis has intuitive appeal, it is premature to form conclusions about the Borda Count method and noise. A general question resulting from these experiments

is whether inclusion of preference in voting methods gives them some kind of resilience in the face of noise.

3.5 Classification theory can benefit by incorporating concepts from voting theory

Voting theory [1] is a mathematical discipline that has concentrated on development of ‘fair’ voting procedures. A number of fairness criteria have been identified:

1. **Majority Criterion:** Any candidate receiving a majority of first place votes should be the winner.
2. **Condorcet Criterion:** A candidate who wins head-to-head matchups with all other candidates should be the winner.
3. **Monotonicity Criterion:** If an election is held and a winner is declared, the winning candidate should remain the winner in any revote in which all preference changes are in favor of the winner of the original election.
4. **Independence of Irrelevant Alternatives Criterion:** If an election is held and a winner is declared, this winning candidate should remain the winner in any recalculation of votes as a result of one or more of the losing candidates dropping out.

Arrow’s work in 1952 abruptly ended these efforts when he showed that such a search was in vain [1]. In terms of the four fairness criteria defined above, Arrow’s result means that there is *no* voting method that can satisfy all four. Thus, for example, a method designed to satisfy the Majority Criterion will always violate at least one of the other criteria. This, in turn, means that there is no ‘perfect’ voting method — and any decision about the choice of voting method to be used is therefore, by necessity, subjective. The best one can hope for is to be able to objectively analyze the strengths and weaknesses of various methods and to apply that knowledge to the task of selecting a ‘good’ method for a particular situation.

The examples in Section 1.2 contradict the optimality of the plurality voting method in situations involving preference among candidates. Thus, for any classification involving at least 3 candidate classes, if preference among classes is essential to represent an expert’s opinion, plurality voting may be a suboptimal voting scheme. Because the plurality voting method ignores preference, it omits important information that can be used in learning to derive better decisions.

4. CONCLUSIONS

In light of our work here, it is premature for us to make any final judgment about specific voting methods. While the results of this work are encouraging, they are not conclusive — they are only a piece in the large puzzle of model reconciliation. In other words, this research is more a first step than a conclusion.

At the same time, the results here appear positive, in that they suggest new directions for model reconciliation methods, and possibly for classification methods as well. These directions deserve attention. It seems plausible that, for some problems, varying the voting method used could provide a percentage reduction in classification error. At minimum, the discussion here offers a new source of tools for data mining practitioners.

The contributions of this paper rest on decoupling the process of model reconciliation from ensemble methods, such as boosting and bagging. Voting methods define an important class of methods for model reconciliation. On a number of experiments, voting methods yielded performance statistics recorded in tables below. For these experiments, the results suggest the following conclusions:

- plurality is not always the best voting method.
- the number of classes can influence the performance of voting methods
- noise can influence the performance of voting methods.

If true, these conclusions would be in line with results from voting theory. Even if not, we believe that classification theory can benefit significantly from greater use of concepts from voting theory.

The win/loss ratio considered in this paper is an overall measure of risk or trustworthiness of individual voting methods. In some ways this seems a natural measure for model reconciliation, but other measures will also give valuable information.

This work still must be extended to consider criteria for broader data mining success, such as implementation performance and scalability of individual voting methods. It also must consider aspects of the classification problem, such as the kinds of domains under consideration, and the size and degree of noise in the existing dataset. Although the Borda Count method performed well for noisy datasets in the experiments here, it is open whether this behavior reflects some deeper data mining principle.

5. REFERENCES

- [1] K. Arrow. *Social Choice and individual values (2nd Edition)*. Wiley, 1963.
- [2] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.
- [3] L. Breiman, J. H. Friedan, R. A. Olshen, and C. J. Stone. Classification and regression trees. *Wadsworth*, 1984.
- [4] L. Brieman. Bagging predictors. Technical Report 421, Department of Statistics, University of California at Berkeley, CA 94720, Department of Statistics, University of California at Berkeley, CA 94720, 1994.
- [5] M.-J. Condorcet. *Éssai sur l’application de l’analyse à la probabilité des décisions rendues à la pluralité des voix*. Paris, 1785.
- [6] J. de Borda. *Mémoire sur les élections au scrutin, histoire de l’académie royale des sciences*. Paris, 1781.
- [7] P. C. Fishburn. Discrete mathematics in voting and group choice. *SIAM J. ALG. DISC. METH.*, 5(2):pp263–275, June 1984.
- [8] Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):pp. 771–780, September 1999.
- [9] R. Gentleman. R project for statistical computing, 1997.

| Data Set | #Trn | #Tst | #Att | M | C | Pl | Anti-Pl | BC | Pl-Elm | PC |
|-------------------------------------|-------|-------|------|---|----|---------------|---------------|---------------|---------------|---------------|
| autos | 137 | 68 | 25 | N | 6 | 0.4412 | <i>0.4706</i> | 0.4118 | 0.4412 | 0.4412 |
| balance scale | 417 | 208 | 4 | N | 3 | 0.2596 | <i>0.3990</i> | 0.2163 | 0.2500 | 0.2163 |
| credit screening | 460 | 230 | 4 | Y | 2 | <i>0.1870</i> | <i>0.1870</i> | <i>0.1870</i> | 0.1783 | <i>0.1870</i> |
| echocardiogram | 88 | 44 | 10 | Y | 2 | 0.3864 | 0.3636 | 0.3864 | <i>0.4091</i> | 0.3864 |
| glass | 143 | 71 | 9 | N | 6 | 0.3099 | 0.2817 | <i>0.3380</i> | 0.3239 | 0.3099 |
| heart disease (cleveland) | 202 | 101 | 13 | Y | 5 | 0.4257 | 0.3663 | <i>0.4356</i> | <i>0.4356</i> | 0.4257 |
| heart disease (hungarian) | 196 | 98 | 13 | Y | 2 | 0.3469 | 0.3469 | 0.3469 | <i>0.6531</i> | 0.3469 |
| heart disease (switzerland) | 82 | 41 | 13 | Y | 5 | 0.9512 | 0.9512 | 0.9512 | <i>1.0000</i> | 0.9512 |
| heart disease (va) | 133 | 67 | 13 | Y | 5 | 0.7015 | 0.7015 | 0.7015 | <i>0.9701</i> | 0.7015 |
| image | 210 | 2100 | 19 | N | 7 | <i>0.1248</i> | 0.1229 | <i>0.1248</i> | <i>0.1248</i> | <i>0.1248</i> |
| ionosphere | 234 | 117 | 34 | N | 2 | 0.1282 | 0.1282 | 0.1282 | <i>0.1453</i> | 0.1282 |
| led creator 7 (0% noise) | 667 | 333 | 7 | N | 10 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| led creator 7 (10% noise) | 667 | 333 | 7 | N | 10 | 0.3453 | <i>0.6426</i> | 0.3363 | 0.3363 | 0.3393 |
| led creator 7 (25% noise) | 667 | 333 | 7 | N | 10 | 0.6787 | <i>0.8679</i> | 0.6817 | 0.6997 | 0.6637 |
| led creator 7 (50% noise) | 667 | 333 | 7 | N | 10 | <i>0.9219</i> | 0.8979 | <i>0.9219</i> | <i>0.9219</i> | <i>0.9219</i> |
| led creator 17 (0% noise) | 667 | 333 | 24 | N | 10 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| led creator 17 (10% noise) | 667 | 333 | 24 | N | 10 | 0.3123 | <i>0.5435</i> | 0.3063 | 0.3123 | 0.3123 |
| led creator 17 (25% noise) | 667 | 333 | 24 | N | 10 | 0.6937 | <i>0.7988</i> | 0.7087 | 0.6907 | 0.6877 |
| led creator 17 (50% noise) | 667 | 333 | 24 | N | 10 | 0.8739 | 0.8799 | 0.8529 | <i>0.8919</i> | 0.8649 |
| lenses | 16 | 8 | 4 | N | 3 | 0.5000 | 0.5000 | <i>0.6250</i> | 0.2500 | 0.5000 |
| page blocks | 3649 | 1824 | 10 | N | 5 | 0.0334 | <i>0.0636</i> | 0.0323 | 0.0444 | 0.0334 |
| pima indians diabetes | 512 | 256 | 8 | N | 2 | <i>0.2451</i> | <i>0.2451</i> | <i>0.2451</i> | 0.2372 | <i>0.2451</i> |
| post operative patient data | 60 | 30 | 8 | Y | 3 | 0.3000 | <i>0.3333</i> | 0.3000 | 0.3000 | 0.3000 |
| restricted primary tumor | 226 | 113 | 17 | Y | 21 | 0.6903 | <i>0.7257</i> | 0.6991 | 0.6903 | 0.6903 |
| soybean (large) | 205 | 102 | 35 | Y | 19 | <i>0.3803</i> | 0.3191 | 0.3112 | 0.3617 | 0.3644 |
| soybean (small) | 31 | 16 | 35 | N | 4 | <i>0.7500</i> | 0.3750 | 0.5625 | <i>0.7500</i> | <i>0.7500</i> |
| space shuttle (o-ring-erosion-only) | 15 | 8 | 4 | N | 3 | <i>0.3750</i> | 0.2500 | <i>0.3750</i> | <i>0.3750</i> | <i>0.3750</i> |
| statlog German | 667 | 333 | 20 | N | 2 | 0.2342 | 0.2342 | 0.2342 | <i>0.2372</i> | 0.2342 |
| statlog Heart | 180 | 90 | 13 | N | 2 | <i>0.2333</i> | <i>0.2333</i> | <i>0.2333</i> | 0.2000 | <i>0.2333</i> |
| statlog shuttle | 43500 | 14500 | 9 | N | 7 | 0.0037 | <i>0.9972</i> | 0.0028 | 0.0037 | 0.0037 |
| statlog vehicle (xaa.dat) | 63 | 31 | 18 | N | 4 | 0.3871 | <i>0.4839</i> | 0.3871 | 0.3871 | 0.3871 |
| statlog vehicle (xab.dat) | 63 | 31 | 18 | N | 4 | <i>0.5484</i> | 0.5161 | 0.5161 | 0.5161 | 0.5161 |
| statlog vehicle (xac.dat) | 63 | 31 | 18 | N | 4 | 0.3548 | <i>0.3871</i> | <i>0.3871</i> | <i>0.3871</i> | 0.3548 |
| statlog vehicle (xad.dat) | 63 | 31 | 18 | N | 4 | <i>0.3871</i> | 0.3226 | 0.3548 | <i>0.3871</i> | <i>0.3871</i> |
| statlog vehicle (xae.dat) | 63 | 31 | 18 | N | 4 | <i>0.5161</i> | 0.4194 | 0.4839 | 0.4516 | 0.4839 |
| statlog vehicle (xag.dat) | 63 | 31 | 18 | N | 4 | <i>0.6452</i> | 0.6129 | 0.5161 | 0.5806 | <i>0.6452</i> |
| statlog vehicle (xah.dat) | 63 | 31 | 18 | N | 4 | 0.3226 | <i>0.3871</i> | 0.2903 | 0.3226 | 0.3226 |
| tic tac toe | 639 | 319 | 9 | N | 2 | 0.1285 | 0.1285 | 0.1285 | <i>0.1975</i> | 0.1285 |
| wine | 119 | 59 | 13 | N | 3 | <i>0.0847</i> | 0.0169 | 0.0508 | <i>0.0847</i> | <i>0.0847</i> |

Legends:

Bold/Red=Win (lowest error) *Italic/Blue=Loss (highest error)*
 #Trn: No. of training sets, #Tst: No. of testing sets
 #Att:No. of Attributes M: Missing data, C: No. of Classes

Table 3: Generalization Error of Bagging (#bootstraps=10) using various voting methods

| Voting method | No. of wins | No. of losses | No. of ties | Win/Loss Ratio | Ranking |
|---------------------------------|-------------|---------------|-------------|----------------|----------|
| Borda Count | 19 | 10 | 2 | 1.900 | 1 |
| Condorcet's Pairwise Comparison | 14 | 10 | 2 | 1.400 | 2 |
| Anti-Plurality | 17 | 16 | 2 | 1.063 | 3 |
| Plurality | 10 | 13 | 2 | 0.769 | 4 |
| Plurality with Elimination | 9 | 16 | 2 | 0.563 | 5 |

Table 4: Comparisons of various voting methods in Bagging (#bootstraps=10)