Forests of Stumps

Amirah Alharthi, Charles C. Taylor and Jochen Voss

Abstract Many numerical studies (Hansen and Salamon (1990), Schapire (1990)) indicate that bagged decision stumps perform more accurately than a single stump. In this work, we will investigate two approaches to create a forest of stumps for classification. The first method is bagging with stumps, that is growing a stump on different bootstrap sample size drawn from the training dataset. The second method is Gini-sampled stumps, where we sample split points with probability proportional to the Gini index. These two methods are combined with two aggregation methods: Majority vote and weighted vote. We use simulation studies to compare the performance and consumed time for these two methods. The computing time of generating split points by Gini-sampled stumps is less than half of the time needed to generate split points from bootstrap samples. Also, weighted vote aggregation results in more accurate performance than majority vote aggregation.

Amirah Alharthi
University of Leeds, Leeds, the UK & Taif University, Taif, Saudi Arabia
✉ mmasja@leeds.ac.uk
✉ amiraharthi@hotmail.com

Charles C.Taylor · Jochen Voss
University of Leeds, Leeds, the UK
✉ charles@maths.leeds.ac.uk
✉ j.voss@leeds.ac.uk

ARCHIVES OF DATA SCIENCE, SERIES A
(ONLINE FIRST)
KIT SCIENTIFIC PUBLISHING
Vol. 5, No. 1, 2018

DOI: 10.5445/KSP/1000087327/30
ISSN 2363-9881

1 Introduction

Over the past few decades, researchers around the world have increasingly focused on data science and its applications. Data science unifies statistics and computer science and their applications, methods, techniques and theories. There are many subfields of data science such as machine learning, computational statistics (these two subfields are closely connected to each other) and classification. While computational statistics is known as the science of designing algorithms to implement statistical methods on computers and deal with analytical problems (Lauro, 1996), machine learning is defined as a field of computer science in which computers have the ability to learn from the data without being explicitly programmed (Samuel, 1959). As the volume and nature of the gathered data becomes larger and more complex, analysing it is also more challenging. Therefore, researchers recently focus on developing efficient algorithms that tackle the data properly. Decision stump algorithms are one of the simplest kinds of algorithms and were mentioned first by Iba and Langley (1992). We will introduce decision stumps, bagging, the Gini function and some of the aggregating decision methods. We investigate here two main experiments: The first experiment is to consider bagging with stumps which is an ordinary stump fitted on bootstraps of different sample sizes and the decisions of these trees are combined by using two different aggregating methods (Majority vote or Weighted vote). The technique of the second approach (Gini-sampled stumps) is to generate split points for a variable according to Gini index values, splitting the training dataset into two subsets greater and smaller than this split point or "threshold" by using this set of split points, and then combining the results by using the same aggregation methods mentioned previously.

1.1 Decision Stumps

A decision stump is a one-level decision tree, with a root "the internal node" that is directly connected to two leaves "the terminal nodes".

We have n observations with p feature variables that are used in making predictions. Let x_{ik} represent the value of the k^{th} variable for the i^{th} observation, where $i = 1, 2, \dots, n$ and $k = 1, 2, \dots, p$. We have a dataset X that can be denoted as an $n \times p$ matrix whose $(i, k)^{th}$ element is x_{ik}.

Every row of X represents one observation and we write the observations as $x_1, x_2, \ldots, x_n$. Here x_i is a vector of length p, containing the p features for the i^{th} observation. That is, $x_i' = (x_{i1}, x_{i2}, \ldots, x_{ip})$. Every column of X represents one variable and we write the variables as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p$. Each is a vector of length n. By using this we can write our dataset as

$$X = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p). \tag{1}$$

We use y_i to denote the label of the i^{th} observation such that $y_i \in \{1, 2, \ldots, J\}$. Also, we have a test dataset and it contains m observations with the same structure. We have a classifier $f(x)$, such that $f(x)$ is a function defined on the sample space Ω_X and for every $x \in \Omega_X$, $f(x) \in \{1, 2, \ldots, J\}$. This classifier f partitions the data X into subsets where:

$$X_j = \{x \in X : f(x) = j\}. \tag{2}$$

A stump uses a variable $\mathbf{x}_k$ to split the training data and make the prediction on the testing data. For continuous feature variables, the most common approach is that a feature variable and a corresponding threshold feature value, i.e. split points are selected to create a stump with two leaves for values below and above that threshold.

The classic way is that each of the two leaves is labelled based on the most frequent class for the observations in that leaf, however, we will also keep track of the proportion of observations in each leaf because it happens frequently that the classes' proportions are almost equal, for example 0.51 in one class and 0.49 in the other. This idea is the basis of the weighted vote aggregation method which will be explained in Section 3. The stump selects the feature variable and the "split point" in a way such that the child nodes, which are here the terminal nodes, are always "purer" than their parent node. The most common function used to measure impurity for nodes is the Gini function, for which the split point is chosen. We will talk about the Gini function in more detail in Subsection 1.2.

A decision stump is called a weak learner due to its poor performance and decision stumps are commonly used as components in ensemble strategies such as bagging. A number of studies (Freund et al. (1996), Dimitriadou et al. (2003)) show that combining the decision of a group of classifiers can result in a better decision and improve the accuracy and performance of weak learners.

Nevertheless, in this research we focus on bagged decision trees and bagging will be discussed in Subsection 2.1.

1.2 The Gini Function

Binary tree classifiers are constructed by recursively splitting subsets of the training dataset into two child nodes. To split any node into the child nodes we have to select the splits in a way such that the child nodes are always "purer" than their parent's. The "purity" of a node can be measured e.g. by a concentration measure. For this purpose we have chosen the Gini index (for a survey see e.g. Giorgi (1990)). The Gini index is defined as follows:

For a given node c with n_c observations let n_{cj} be the number of observations in class j, $j \in \{1, 2, \ldots, J\}$, such that

$$n_c = \sum_{j=1}^{J} n_{cj}. \tag{3}$$

Then, the Gini index can be defined as:

$$\Gamma(c) = 1 - \sum_{j=1}^{J} \left(\frac{n_{cj}}{n_c} \right)^2. \tag{4}$$

Now, let $c \in \{P, L, R\}$, the change in the Gini index resulting from splitting node P (parent) into L (left) and R (right) is

$$\triangle_\Gamma = \Gamma(P) - \left(\frac{n_L}{n_P} \Gamma(L) + \frac{n_R}{n_P} \Gamma(R) \right). \tag{5}$$

The classification tree algorithm searches through all possible candidate splits of P to select the one with the maximum change $\triangle_\Gamma$.

2 Splitting Schemes

We investigate two different splitting schemes to build an ensemble of decision stumps. The first scheme is bagging with stumps, which is basically combining stumps with the bagging technique and it will be described in more detail in Subsection 2.1. The second scheme is Gini-sampled stumps, which is a new subsampling method that builds a forest of stumps based on generated split points from Gini indices. This method will be explained in Subsection 2.2.

2.1 Bagging with Stumps

We need here to define bagging before explaining this method. Bagging stands for **B**ootstrap **agg**regat**ing** and was proposed by Breiman (1996). It is an ensemble method designed to improve the classification by combining the classifiers of randomly generated bootstrap samples of the training set. The size of the bootstrap sample size is n^* and it can be chosen to be equal to the training data size.

Now, suppose we are given a sequence of learning sets $\{X_b\}, b = 1, 2, \ldots, B$, each consisting of $n^* = n$ independent observations, drawn with replacement uniformly at random from the training data. This is the bootstrap technique and it has been used in many ensemble methods like "Random Forests" (Breiman, 2001) and "Bagging predictors" (Breiman, 1996). The goal is to use the $\{X_b\}$ to get a better classifier than the classifier using a single learning dataset. By sampling, each observation has a n^{-1} chance to be not repeated in this sampling and $(1 - n^{-1})$ being repeated. By repeating this n-times, we will get $(1 - n^{-1})^n$ of repeated observations. The expectation of having not repeated values is

$$n\left(1-\left(1-\frac{1}{n}\right)^{n}\right) \approx n\left(1-\frac{1}{e}\right), \tag{6}$$

since $\lim_{n\to\infty}(1 - n^{-1})^n = e^{-1}$. The percentage of unique observations is $\frac{n\,(1-e^{-1})}{n} = 1 - e^{-1} = 63.2\,\%$, so, we have, on average, 63.2 % of the unique observations of the training dataset in each bootstrap sample.

After fitting a classifier (a stump) on each bootstrap-sampled version of the training data, then we have to make a final decision by aggregating these classifiers decisions. There are many aggregation decision methods and we will talk about the two aggregation methods that are used in this work in Section 3.

The main idea of this method is fitting a decision stump on samples of size n^* drawn from the training data, making a prediction on the testing data by using these models and finally applying aggregating techniques on an ensemble of these stumps. The methodology of this ensemble can be summarised in the following steps:

1. Repeat B times:
    - Draw n^* observations with replacement from the training data
    - Fit a decision stump on the sampled data
2. Apply one of the aggregation methods that are mentioned in Section 3.

We consider different n^* here to test whether the bootstrap sample size has any affect on enhancing the performance of the method or not.

2.2 Gini-sampled Stumps

Here we take a random sample of split points derived from Gini indices to build an ensemble of stumps. This method is a new way of generating a set of split points and we refer to this as the Gini-sampled stumps method.

It is known that, any tree-based method will typically compute a Gini index at all possible split points and variables, select the highest value and then split the data by using that variable and corresponding split point. Using the generated split points to divide the training dataset into two subsets is a new approach which allows us to consider the possible occurrence of every possible split. Here, we are treating the Gini function as a probability density function and sampling from it. The following steps explain the method:

1. Let $x_1, \ldots, x_n$ be the data-points of variable $p = 1$ and $x_{(1)}, \ldots, x_{(q)}$ be the unique values in increasing order .
2. The Gini gain g is constant on the intervals $(x_{(i)}, x_{(i+1)})$ and we have $g(x) = 0$ for $x < x_{(1)}$ and $x > x_{(q)}$.

3. Generate random samples with density $\frac{1}{S}\,g$, where

$$S = \sum_{i=1}^{q-1} g\left(\frac{x_{(i)} + x_{(i+1)}}{2}\right)\left(x_{(i+1)} - x_{(i)}\right). \tag{7}$$

4. For the inverse transform method, find the cdf

$$G(x) = \frac{1}{S}\left(\sum_{i=1}^{k} g\left(\frac{x_{(i)} + x_{(i+1)}}{2}\right)\left(x_{(i+1)} - x_{(i)}\right) + g(x)\left(x - x_{(k+1)}\right)\right), \tag{8}$$

   where $k = \max\{i \,|\, x_{(i+1)} \leq x\}$, and let $G_k = G(x_{(k)})$.

5. Sample from G by using the inverse transform method as follows :

   For $(b \leftarrow 1$ to $B)$

   a. Generate a random value uniformly distributed $u \sim U(0, 1)$.
   b. Find $v = max\{i|G(x_i) \leq u\}$, so that $G_v \leq u \leq G_{v+1}$.
   c. Find the slope of the v^{th} line segment $a = \frac{u-G_v}{G_{v+1}-G_v}$.
   d. Finally, compute $x_{(v)} + a(x_{(v+1)} - x_{(v)})$.

6. Make a stump of every split point from the previous step.

Please note that there are two differences between bagging with stumps and Gini-sampled stumps. The first difference is the way split points are generated. That is, selecting the split point to maximise the Gini index in each bootstrap sample, and generating random split points according to the distribution of the Gini indices. The second difference is the way to determine the class distribution in each leaf. For bagging with stumps, this is defined by considering only the bootstrap sample, while all the training dataset is used for the Gini-sampled stump method.

We also consider raising Gini index values g to power κ such that $\kappa \in [0, \infty)$ before Step 2 to have more flexibility and test whether decreasing the variation of the randomly generated split points will have any effect on the performance of this method. Note, $\kappa = 0$ gives uniformly distributed splits and $\kappa \to \infty$ will always return the optimal split.

3 Decision Aggregation

The aggregated decision is made by an aggregation function. The choice of the aggregation function plays an important role in many disciplines such as statistics, computer science and finance. It gives a single decision based on combining several decisions in which the single decision represents all the individual decisions well, for further information see Grabisch et al. (2011). There are many forms of aggregation functions like means (arithmetic, geometric, etc), mode, sum, minimum and maximum functions. In this work, we will talk about two aggregation methods, namely majority voting (see Subsection 3.1) and weighted voting (see Subsection 3.2).

3.1 Majority Vote

Majority vote (mode) is used in bagging, as introduced in Subsection 2.1 as an aggregation decision function for classification. The performance accuracy of a bagging classification ensemble tends to level off after combining a large number of classifiers (Martínez-Muñoz et al., 2007). A majority vote technique of aggregating decisions is counting the votes for each classifier, considers the highest voted predicted class as the final decision. If $f(x)$ predicts a class j, then the method of aggregating the $f_b(x)$ by voting is as follows:

$$f(x) = \underset{1 \leqslant j \leqslant J}{\arg\max} \sum_{b=1}^{B} I(f_b(x) = j), \tag{9}$$

where $I(\cdot)$ is the indicator function which is 1 if the statement inside the parentheses is true and 0 (false) otherwise.

3.2 Weighted Voting

A Weighted voting is another way of combining classifiers. This technique can be explained by the following illustrative example:

Consider a data set which contains n observations, two classes A and B, and a single explanatory feature variable $\mathbf{x}$. A forest of three stumps $\{(1), (2), (3)\}$

as in Figure 1 was grown on this data and the classification decision was made by using one of two methods: Majority voting (MV) or weighted voting (WV). Assume a new observation is equal to 30. Weighted voting works in the following way: From the information in Figure 1, this observation is in the left leaf of the decision stump (1) with the proportions $\hat{\pi}_{A1} = 0.52$ and $\hat{\pi}_{B1} = 0.48$. $\hat{\pi}_{A1}$ denotes the proportion of class A stump (1). The same rule is applied for the other two stumps, the position of the observation and proportions are as in Table 1. We add up

$$\sum_{b=1}^{B} \hat{\pi}_{jb}$$

for each class j separately. Then we choose the class which maximises this summation. If $f(x)$ predicts class j, then the weighted vote aggregation method is:

$$f(x) = \underset{1 \leqslant j \leqslant J}{\arg\max} \sum_{b=1}^{B} \hat{\pi}_{jb}. \tag{10}$$

Here, this observation will be allocated to class B due to its added-up proportion of 1.53 which is greater than class A with an added-up proportion of 1.47. Therefore, while this observation is allocated to class A by using the majority vote, it is allocated to class B by using the weighted vote.

Table 1: The positions of the new observation x=30 in the dataset and the aggregated decisions.

Stump	**Leaf**	**Majority vote**	**Weighted vote**	
			$\hat{\pi}_{\mathbf{A}}$	$\hat{\pi}_{\mathbf{B}}$
(1)	Left	A	0.52	0.48
(2)	Left	A	0.60	0.40
(3)	Right	B	0.35	0.65
			1.47	1.53
Decision		A	B	

By combining the two splitting schemes and aggregation methods, we have four methods, as in Table 2, that are compared in Section 4.

Table 2: Four methods investigated in Section 4.

	Aggregation Methods	
Method	**Majority vote**	**Weighted vote**
Bagging with stumps	Bagging with stumps MV	Bagging with stumps WV
Gini-sampled stumps	Gini-sampled stumps MV	Gini-sampled stumps WV

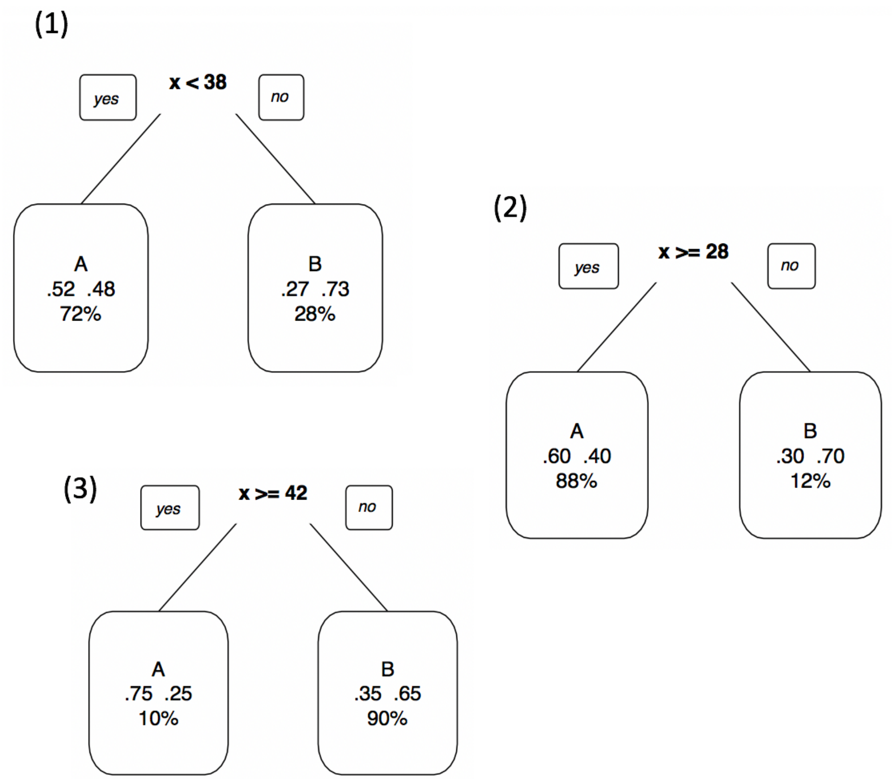


Figure 1: An example of aggregation methods applied on a forest of three stumps. The percentages indicate the percentages of the total number of observations in each leaf regardless of there classes.

4 Results

4.1 Simulated Data

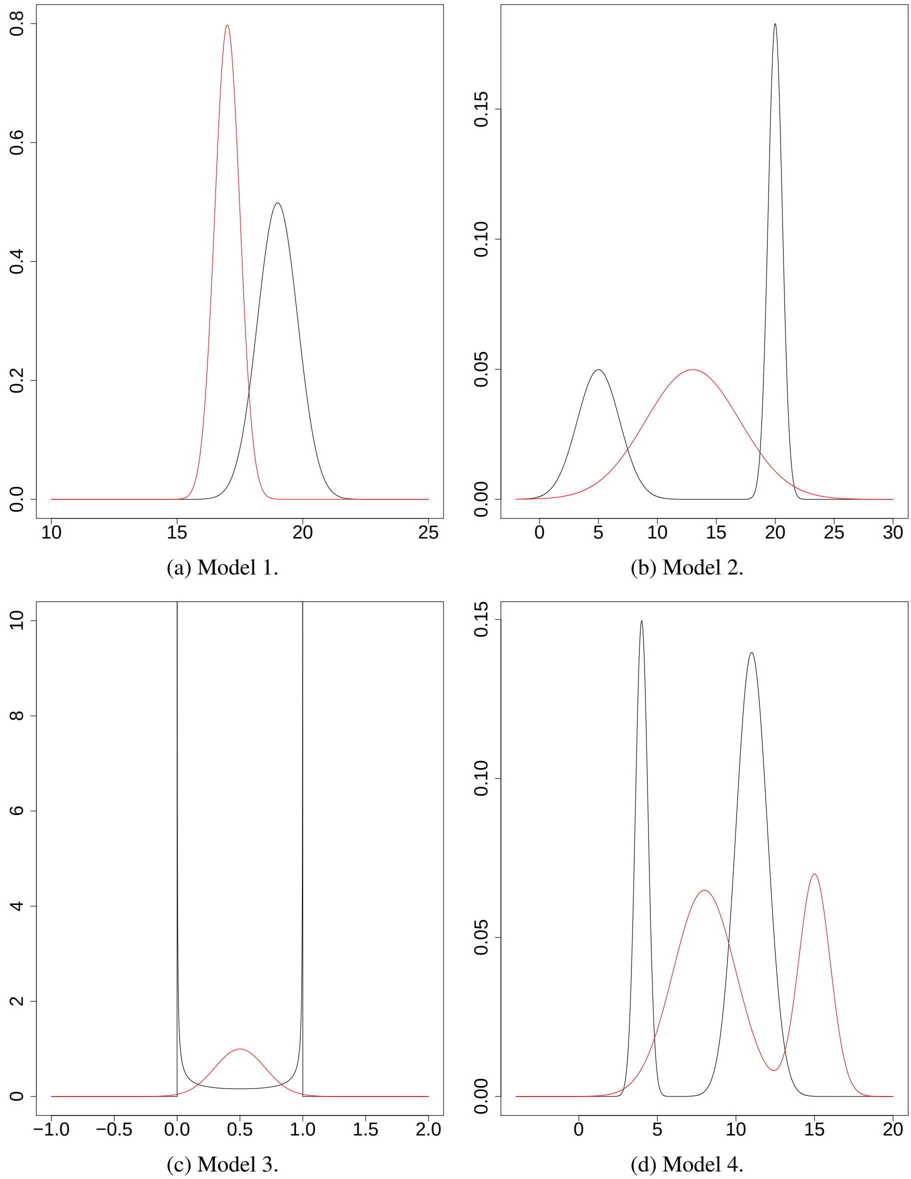
In this section, we consider five different models, each with two classes (A, B) and one explanatory variable. The densities of the explanatory variable for these models are described in Table 3. Table 4, shows maximal (Bayes rate) accuracies of these models. The densities of the explanatory variable for these models are shown in Figure 2.

Table 3: Densities of the explanatory variable for the five models.

Model	A	B
1	$A \sim N(17, 0.5^2)$	$B \sim N(19, 0.8^2)$
2	$A \sim \begin{cases} N(5, 1.8^2), \text{probability } 0.45 \\ N(20, 0.6^2) \end{cases}$	$B \sim N(13, 4)$
3	$A \sim B(0.2, 0.2^2)$	$B \sim N(0.5, 0.2^2)$
4	$A \sim \begin{cases} N(4, 0.4^2), \text{probability } 0.30 \\ N(11, 1^2) \end{cases}$	$B \sim \begin{cases} N(8, 2^2), \text{probability } 0.65 \\ N(15, 1^2) \end{cases}$
5	$A \sim \begin{cases} N(5, 1.3^2), \text{probability } 0.35 \\ N(11, 1.4^2) \end{cases}$	$B \sim N(8, 1.6^2)$

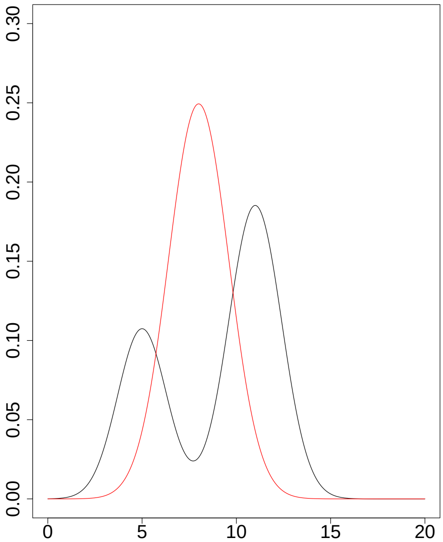
Table 4: Best possible classification accuracies (%) for the five models.

Model	1	2	3	4	5
Accuracy	94.21	92.87	83.50	87.36	78.50



(a) Model 1. (b) Model 2. (c) Model 3. (d) Model 4.

Figure 2: Densities of the explanatory variable of the five models for each class.



(e) Model 5.

Figure 2: Densities of the explanatory variable of the five models for each class.

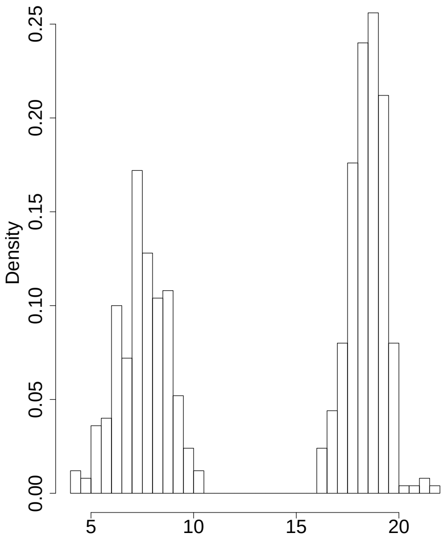
4.2 Bagging with Stumps

Table 5 shows the prediction performance of the different tree-based methods for the five models of Subsection 4.1. These methods are: Bagging with Stumps MV, Bagging with Stumps WV, a forest of trees MV, a forest of trees WV, a single tree and a decision stump. The term bagging with stumps is a relatively new name for a group of B one-level trees that are bagged by using one of the aggregation methods mentioned previously whereas forest of trees is generally understood to mean a group of B standard trees with no feature selection. These models are fitted on a bootstrap of size n^* that is drawn from a training dataset has n observations. Finally, their predictions are evaluated on a test data has m observations. Here, $n = 1000, m = 1000, B = 500$ and $n^* = (20, 40, 60, 80, 100, 200)$. It can clearly be seen from Table 5, that models 1 and 5 have similar results for the two bagging with stumps methods. However, models 2 with WV, 3 and 4 perform more accurately in smaller bootstrap sizes.

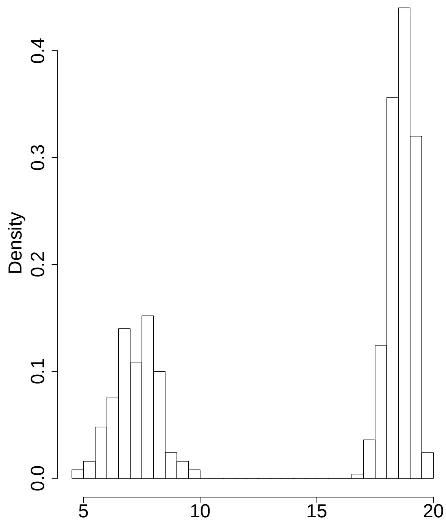
Model 2 MV shows the opposite behaviour, it shows a very small increase in the percentages as the bootstrap sizes increased.

In terms of the aggregation methods, the WV aggregation method wins regardless whether it combines with bagging with stumps or forest of trees.For models 2, 3, and 4 the forest of tree method outperforms the other methods. The best result (in bold) in Table 5 occur with WV and they are close to the maximal accuracies shown in Table 4.

Figure 3 shows that the variation of split points decreases as n^* increases for model 2. The split point distribution is more diverse when n^* is smaller and the diversity decreases by increasing the bootstrap sample size.

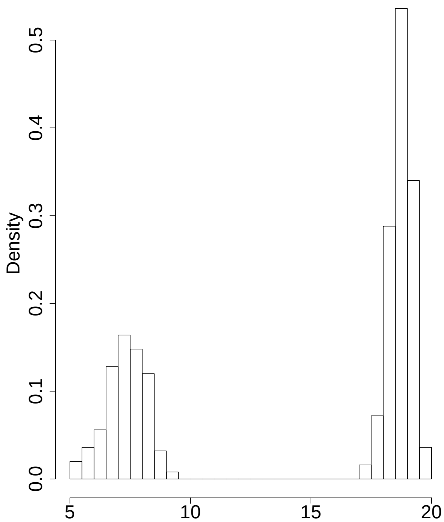


(a) $n^* = 20$.

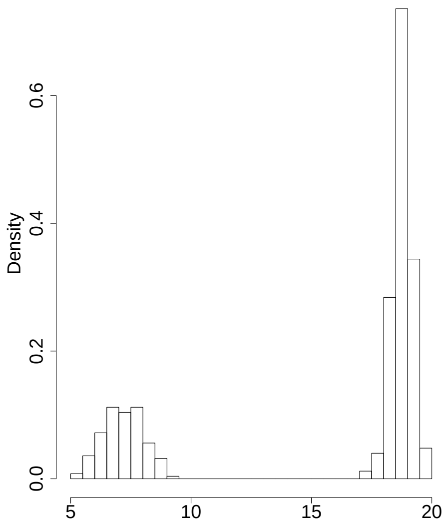


(b) $n^* = 40$.

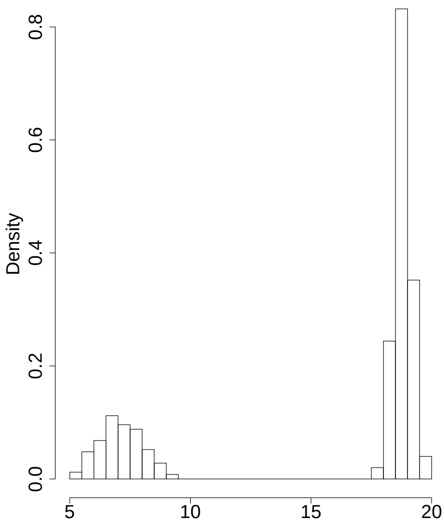
Figure 3: Split points of 500 stumps for different bootstrap sample sizes n^* from model 2.



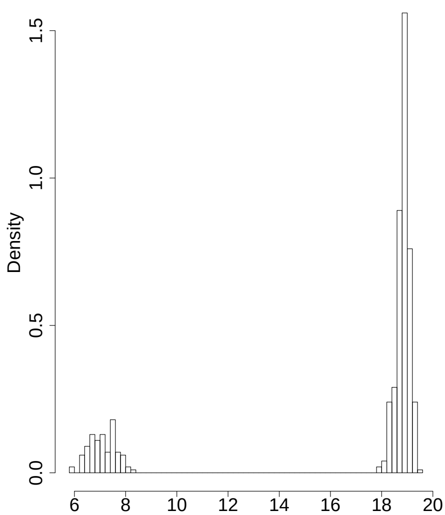
(c) $n^* = 60$.



(d) $n^* = 80$.

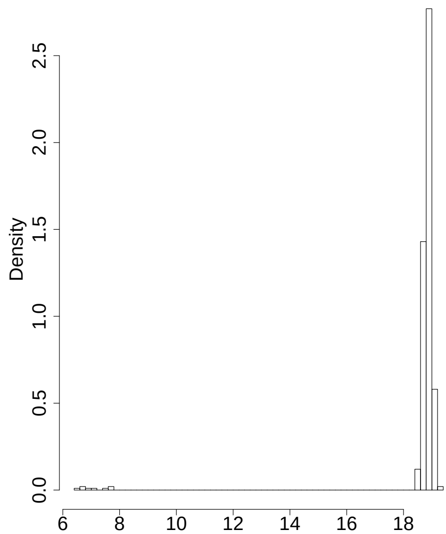


(e) $n^* = 100$.



(f) $n^* = 200$.

Figure 3: Split points of 500 stumps for different bootstrap sample sizes n^* from model 2 (continuation).



(g) $n^* = 1000$.

Figure 3: Split points of 500 stumps for different bootstrap sample sizes n^* from model 2 (continuation).

Table 5: The average of percentages of correctly predicted classes of 50 simulations from five models by using different tree-based methods (1/2).

Model	Sample Size n^*	Bagging with Stumps		Forest of Trees		Standard Tree	Decision Stump
		MV	WV	MV	WV		
1	20	94.09	94.10	94.13	94.13	91.00	92.51
	40	94.11	94.13	94.07	94.06	90.42	92.89
	60	94.12	94.13	94.11	94.11	91.38	93.20
	80	94.13	**94.15**	94.10	94.10	90.96	93.33
	100	94.13	94.14	94.05	94.05	92.04	93.61
	200	94.14	94.09	94.08	94.09	92.99	93.73
2	20	72.48	87.24	88.58	88.59	85.51	67.76
	40	72.53	83.71	88.78	88.90	84.38	70.86
	60	72.77	77.95	88.53	88.89	84.43	69.59
	80	72.83	76.12	89.24	89.25	84.77	70.80
	100	72.93	73.73	89.72	89.84	84.51	70.85
	200	73.02	73.17	90.20	**90.30**	85.43	72.49

Table 5: The average of percentages of correctly predicted classes of 50 simulations from five models by using different tree-based methods (2/2).

Model	Sample Size	Bagging with Stumps		Forest of Trees		Standard Tree	Decision Stump
	n*	**MV**	**WV**	**MV**	**WV**		
3	20	72.84	83.16	83.16	83.17	76.51	63.84
	40	70.50	83.18	83.18	83.19	76.61	65.44
	60	68.25	83.12	83.12	83.10	75.54	64.77
	80	66.27	83.07	83.07	83.06	76.33	65.43
	100	65.71	83.11	83.11	83.12	76.60	66.03
	200	65.14	$\boxed{\mathbf{83.21}}$	83.11	83.20	80.56	65.44
4	20	77.01	83.78	87.11	87.14	80.34	61.24
	40	66.54	66.72	87.30	87.30	80.88	63.22
	60	66.54	66.14	87.34	87.33	80.58	64.58
	80	66.56	66.31	87.25	87.33	80.82	64.70
	100	65.57	66.33	87.28	$\boxed{\mathbf{87.35}}$	81.80	65.14
	200	66.55	66.48	87.28	87.28	81.30	65.99
5	20	69.43	69.82	78.41	**78.42**	69.04	64.13
	40	69.34	69.90	78.41	78.41	68.83	66.06
	60	69.32	69.89	78.30	78.31	68.84	66.79
	80	69.35	69.89	78.31	78.30	69.83	66.88
	100	69.35	69.88	78.31	78.28	70.08	67.89
	200	69.31	69.84	$\boxed{\mathbf{78.42}}$	78.40	74.78	68.54

4.3 Gini-sampled Stumps

Table 6 shows the percentages of correctly predicted classes by using the Gini-sampled stumps method. The performance of model 1 and models 2, 4 and 5 with MV get more accurate as κ increases whilst the performance of models 2, 4 and 5 with WV and 3 with MV decrease for larger κ.

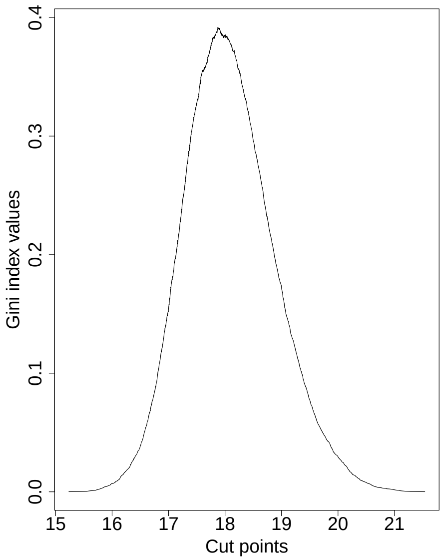
Comparing general results of the two aggregation methods, WV also wins here especially with smaller values of $\kappa = (0, 1/3, 1/2, 1)$. The best results (in bold) appear in the WV column with smaller values of κ and they are similar to the maximal accuracies in Table 4, especially for models 1, 3 and 5.

The plots in Figure 4 indicate Gini gains as a function of split points from the five models in Figure 2. By comparing Figure (3a) with Figure (4b), we find that the Gini curve almost resembles the histogram of the bootstrap sample size 20.

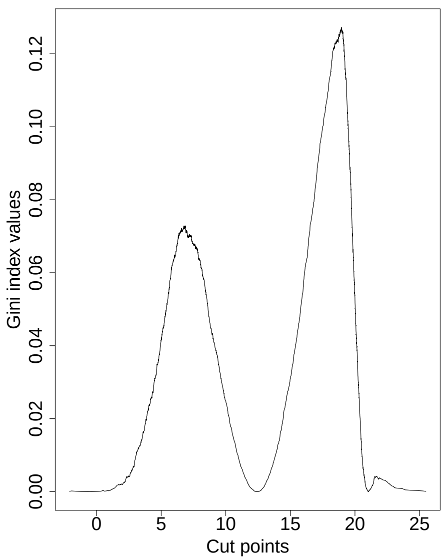
4.4 Comparing The Two Methods for Generating Splits

A comparison is carried out between these two methods in terms of their consumed time to generate 500 split points. Gini-sampled stumps are fitted on training data and bagging with stumps are fitted on different bootstrap sample sizes n^* are drawn from training data with $n = 1000$ observations. Here, the bootstrap sample size n^* takes the values $(40, 60, 80, 100, 200, 500, 1000)$.

Figure 5 indicates the time needed to generate 500 split points by the Gini-sample stumps method and bagging with stumps. It is clear that the Gini-sample stumps method is faster than bagging with stumps.

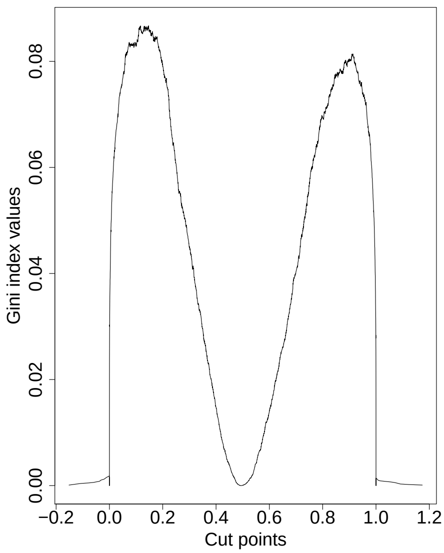


(a) Model 1.

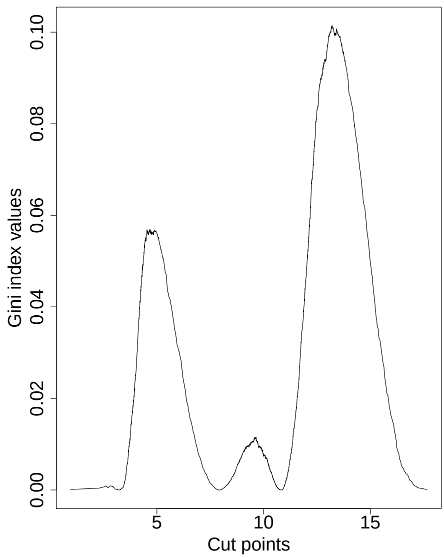


(b) Model 2.

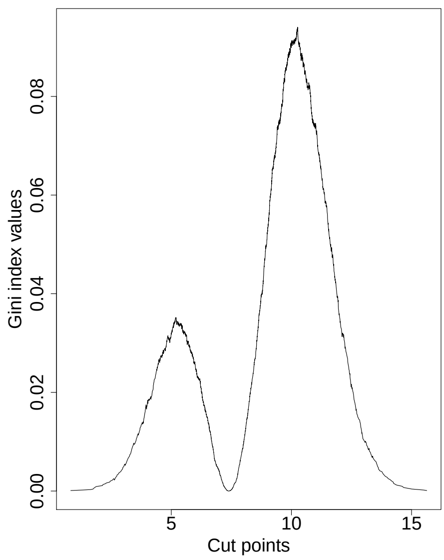
Figure 4: Gini gain as a function of split points from the five models. The highest values in these curves occur at the overlapping areas between the densities in the corresponding model as in Figure 2.



(c) Model 3.



(d) Model 4.



(e) Model 5.

Figure 4: Gini gain as a function of split points from the five models. The highest values in these curves occur at the overlapping areas between the densities in the corresponding model as in Figure 2.

Table 6: The average of percentages of correctly predicted classes of 50 simulations from five models by using Gini-sampled stumps method with raising Gini indices to different powers κ.

Model	κ	**MV**	**WV**
1	0	86.74	93.89
	1/3	92.36	93.98
	1/2	92.91	94.02
	1	93.67	94.07
	2	94.00	94.11
	3	94.05	$\boxed{\textbf{94.16}}$
	4	94.09	94.11
2	0	66.54	$\boxed{\textbf{88.50}}$
	1/3	58.57	88.02
	1/2	58.48	87.83
	1	66.36	86.69
	2	72.71	79.03
	3	73.12	73.68
	4	73.16	72.77
3	0	82.46	83.05
	1/3	75.50	83.13
	1/2	68.26	$\boxed{\textbf{83.19}}$
	1	62.47	83.14
	2	62.93	83.12
	3	63.72	83.06
	4	64.27	82.94
4	0	61.96	$\boxed{\textbf{71.18}}$
	1/3	66.19	64.42
	1/2	66.21	64.31
	1	66.27	65.13
	2	66.31	65.79
	3	66.28	65.99
	4	66.23	66.14
5	0	50.24	$\boxed{\textbf{76.01}}$
	1/3	55.87	74.32
	1/2	60.12	73.25
	1	66.54	70.03
	2	68.76	69.87
	3	69.18	69.91
	4	69.26	69.92

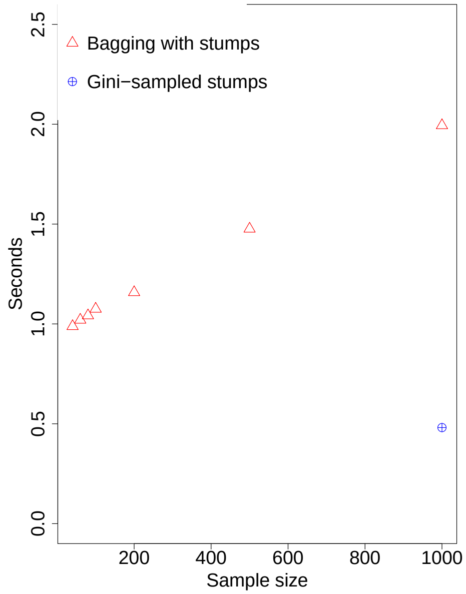


Figure 5: Consumed time to generate 500 split-points by the two stump approaches. Gini-sampled stumps are fitted on a training data of size n and bagging with stumps are trained on different bootstrap sample sizes n^*. These bootstrap sizes are drawn from a training dataset with the same number of observations n and $n = 1000$ samples.

4.5 Conclusion

The Gini-sampled stumps method is a promising method because of its accurate performance and fast speed. The Gini-sampled stumps method is more accurate in model 2 and 5 which both consist of densities of similar shape and it has almost the same accuracy in model 1 and 3 as the bagging with stumps method. However, for some reason bagging with stumps performs better in model 4. The classification performance has a large variation between the two aggregation methods and it is higher for the weighted vote method regardless whether it is with bagging with stumps or Gini-sampled stumps.

References

Breiman L (1996) Bagging Predictors. Machine Learning 24(2):123–140. DOI: 10.1007/BF00058655.

Breiman L (2001) Random Forests. Machine Learning 45(1):5–32. DOI: 10.1023/A:1010933404324.

Dimitriadou E, Weingessel A, Hornik K (2003) A Cluster Ensembles Framework. In: Design and Application of Hybrid Intelligent Systems, pp. 528–534. IOS Press, NLD, Abraham A, Köppen M, Franke KY (eds.). ISBN: 15-8603-394-8.

Freund Y, Schapire RE, et al. (1996) Experiments with a new boosting algorithm. In: Saitta L (ed.), Proceedings of the Thirteenth International Conference on International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., Vol. 96, pp. 148–156. ISBN: 15-5860-419-7.

Giorgi GM (1990) Bibliographic portrait of the Gini concentration ratio. Metron 48(1-4):183–221.

Grabisch M, Marichal J, Mesiar R, Pap E (2011) Aggregation functions: Means. Information Sciences 181(1):1–22, Elsevier. DOI: 10.1016/j.ins.2010.08.043.

Hansen LK, Salamon P (1990) Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(10):993–1001, Institute of Electrical and Electronics Engineers (IEEE). DOI: 10.1109/34.58871.

Iba W, Langley P (1992) Induction of one-level decision trees. In: Sleeman D, Edwards P (eds.), Machine Learning Proceedings. Morgan Kaufmann, San Francisco, pp. 233–240. DOI: 10.1016/B978-1-55860-247-2.50035-8.

Lauro C (1996) Computational statistics or statistical computing, is that the question? Computational Statistics & Data Analysis 23(1):191–193, North-Holland. DOI: 10.1016/0167-9473(96)88920-1.

Martínez-Muñoz G, Hernández-Lobato D, Suárez A (2007) Selection of Decision Stumps in Bagging Ensembles. In: de Sá J, Alexandre L, Duch W, Mandic D (eds.), International Conference on Artificial Neural Networks. Lecture Notes in Computer Science, Springer, Berlin, Vol. 4668, pp. 319–328. ISBN: 978-3-540746-89-8, DOI: 10.1007/978-3-540-74690-4_33.

Samuel AL (1959) Some studies in machine learning using the game of checkers. IBM Journal of Research and Development 3(3):210–229. DOI: 10.1147/rd.33.0210.

Schapire RE (1990) The strength of weak learnability. Machine Learning 5(2):197–227. DOI: 10.1007/BF00116037.