Bias-Variance Decomposition for model selection

Kees Luykx (9760768)

E-mail: luykx@science.uva.nl

Begeleider:

Maarten van Someren

FNWI, University of Amsterdam, Kruislaan 419, 1098 VA Amsterdam, The Netherlands

Abstract

Bias-variance decomposition is known to be a powerful tool when explaining the success of learning methods. By now it's common practice to analyze newly developed techniques in terms of their bias and variance. Apart from an academic context however, the bias-variance decomposition is rarely applied when data mining in a more practical context. In this article I argue that there are several reasons why this is the case. Those are most importantly the lack of a single definition, and the sensitivity of the technique to various factors besides the combination of learning method and dataset. The starting point of this thesis to investigate what the added value of the decomposition would be in the context of model selection. This is done by comparing results from the standard cross-validation procedure with results from the bias-variance decomposition. The conclusion is that under specific circumstances model-selection based on the outcome of the bias-variance composition will lead to bad choices. This is mainly due to instability issues which increase when the overall error is close to zero. This makes the case that results from the bias-variance decomposition are first and foremost estimations of concepts which are hard to directly measure.

1. Introduction

In 1992 Geman, Bienenstock and Doursat used the bias-variance decomposition to show the limitations of neural networks (*Geman,Bienenstock and Doursat, 1992*). It was the first time this technique, which has its origin in statistics, was used to evaluate a learning method from the field of machine learning. Interestingly, this was at a moment when feed forward neural networks using backpropagation were believed to be able to solve just about any imaginable learning task. Using the bias-variance decomposition, they successfully showed that, like nonparametric inference methods in statistics, neural networks suffer from what is called the bias/variance dilemma. The dilemma describes something which is essentially a trade-off between a biased and stable learner, and an unbiased but unstable learner. Here, bias is an indicator for (the strength and) the applicability of a learning method to the structure inherent in the data. Or, loosely speaking, bias can be seen as the tendency of a learner towards certain solutions. This tendency can have a strength, a direction and it can have a limiting effect on the solution-space. And these three factors, apart from the data, influence the stability of a learner. The downside of a strong bias can be that, although stable, the solution might be suboptimal.

Since Geman's 1992 article, this technique has been adapted so that it can be used to evaluate other machine learning techniques as well. In more recent years the evaluation method has been used to assess new data mining techniques like the recent ensemble learners (e.g. *Breiman, 1996; Dietterich & Kong, 1995;Webb, 2000; Brain & Webb, 2002*). But although it has become clear that bias-variance decomposition is a powerful tool for evaluating the success of learning methods, it is still a peculiarity in non-academic data mining. This is due to a couple of reasons. First, there have been a variety of proposals for bias-variance decomposition definitions and techniques which can lead to different results (e.g. *Webb & Conilione, 2003; James, 2000*). And second, the various techniques themselves can give unstable results. Most of the instability issues are connected to the sampling methods, the procedure and the stability of the learning method itself (*Bouckaert, 2008*).

In this paper I will investigate whether the bias-variance decomposition will provide information which will result in alternative choices during the model selection stage when compared to the accuracy measures based on standard cross-validation. Cross-validation is a widely used technique to estimate the general accuracy of a learning method (*Witten & Frank, 2005*). This estimate of accuracy is restricted however to overall accuracy which doesn't necessarily say something about the stability of the predictions on instance level. In theory a high overall accuracy could coincide with unstable predictions on the underlying instances. Using the bias-variance decomposition the overall accuracy (error) is decomposed into the bias and variance components. So, in principle this should give more

information on the accuracy of a learner, and in the context of model selection other models might be preferred.

In this paper results from standard cross-validation are compared to results from the bias-variance decomposition. In the comparisons the stability of the decomposition itself was also taken into account. Various articles have shown that the decomposition can be unstable (*Webb & Conilione, 2003; Bouckaert, 2008*). And this stability might be an important factor in deciding whether one should use the decomposition in addition to overall accuracy. I will start with a short introduction to the overall principle of the bias-variance decomposition and discuss the various techniques which have been proposed. After this introduction the experiments and their results will be discussed. And finally an answer is given to the question whether the bias-variance decomposition could provide new information which could potentially change the preferred model.

2. Estimating Bias and Variance

As already mentioned, various ways to estimate the bias and variance of classifiers have been proposed. The differences can be clustered into three groups: different procedures, different sampling methods, and different definitions of bias and variance. But despite these differences, there is a conceptual basis which is shared among all implementations.

In general, a learning method is being trained N times on different training-sets sampled from the same dataset. The resulting N models are then tested on a set of instances which were not used during training. The N predictions for each test-instance are recorded and from these recorded results, the total error, bias and variance on each test-instance are calculated. These results are then averaged over all test-instances and presented as the final result of the bias-variance decomposition.

This general concept can be implemented in several ways, each resulting in different bias-variance estimates. Two factors of difference will be discussed here, namely the procedure and the definition. The third one, the sampling method refers to the distinction between sampling methods with or without replacement. It has been discussed in Webb and Conilione (*2003*) and is not of particular interest in the scope of this paper.

2.1 Different procedures for estimating bias and variance

The first difference is due to the procedure being used to create test- and training-sets. For illustrative purposes the two most used procedures will be discussed here.

The method proposed by Kohavi and Wolpert (*1996*) is the most intuitive and is based on the hold-out procedure. In the hold-out procedure the dataset is initially

divided into two parts: a test-set and a learn-set from which the N training-sets are sampled (for all procedures N is a parameter). Typically, the sizes of the test-set, the training-sets and the difference between the learn-set and the training-sets should be as large as possible: the bigger the test-set, the better the estimation of the performance; the bigger the training-sets, the better the models; and last, the bigger the difference between the learn- and training-set, the more variation between the different training-sets can exist. More variation between the different training-sets gives better estimates of the bias and variance, or in other words, a better estimation of the sensitivity of the learning method to the training data. Because of the limited size of the dataset, the estimates of bias and variance are based on a compromise between these three conflicting demands. The decomposition is based on models trained on relatively small training-sets and tested on a relatively small test-set. Apart from the restrictions the procedure is also sensitive to the sampling method used to create the various sets of instances, and to aspects of the original dataset like size and relative frequencies of the different classes.

So, although intuitive, this procedure suffers from two problems. First, the estimates are based on a relatively small part of the dataset and may not be representative. And second, due to the conflicting demands and the dependence on sampling, the estimates can be unstable (*Webb & Conilione, 2003; Bouckaert, 2008*).

In an attempt to tackle some of these problems, Webb (*2000*) proposed a procedure using N times k-fold cross-validation. With this procedure all instances from the dataset are used both for testing and training. The “trick” that is being applied, is to do some extra “administrative work” where the test-results on each instance are recorded in a separate results set. In other words, each time an instance is being used as a test-instance during the cross-validation procedure, the prediction for that particular instance is recorded into a separate set of results. The result after the N times k-fold cross-validation procedure is a set where each instance in the original dataset has N predictions (and the actual class). This resulting set can be seen as the equivalent of the hold-out test-set of the previous procedure with the difference that each instance in the original dataset is guaranteed to have been tested (N times). The recorded results can then be handled the same way as in the hold-out procedure. Several experiments have shown that this procedure gives indeed more stable results than the hold-out procedure (*Webb & Conilione, 2003; Bouckaert, 2008*).

In addition to using cross-validation Webb and Conilione (*2003*) implemented a parameter into the procedure to influence what they call *inter-training-set variability*. This inter-training-set variability is a measure for the similarity of the different training-sets and describes the same phenomenon which is discussed in

the section on the hold-out procedure where the difference between the size of the learning-set and the training-sets influenced the variability between different training-sets. This is done by instead of variability can be influenced by slightly decreasing the size of the training-sets (which is (k-1)/k times the size of the total dataset in usual k-fold cross-validation). The range of influence is very restricted however and the need for such an extra parameter is questionable in the context of model selection (as opposed to assessing individual learning methods). In the context of model selection this parameter will be constant for all compared learning methods and the relative variance of all methods will present a sufficient indicator for relative train-set sensitivity.

2.2 Different definitions for calculating bias and variance

Numerous definitions have been proposed for calculating the bias and variance of classifiers (*Breiman, 1996; Kohavi & Wolpert, 1996; Kong & Dietterich, 1995; Friedman, 1997; James & Hastie, 1997; Domingos, 2000*), and a couple of comparative articles have been written (*James, 2003; Webb, 2000*). For a more thorough explanation the reader is advised to read these articles. Here, I will present a rough outline to illustrate the conceptual differences.

In essence the biggest difference between various definitions is due to the way the individual predictions are aggregated to the instance-level. To put it differently, each instance from the test-set will have N predictions, and the question is how these different predictions influence the estimate for bias and variance for that instance. Various methods have been proposed which lead to different estimates. In the context of this paper it suffices to give an example of the different flavors that one can expect when estimating bias and variance. The example originates from Webb's article (*2000*).

In this article the main differences between the various definitions are explained through the concept of *central tendency*. The central tendency $\mathbf{C_{L,T}(x)}$ for learner **L** over the distribution of training-sets **T** is the class with the greatest probability of selection for description **x** by classifiers learned by **L** from training-sets drawn from **T** (*Webb, 2000*).

$$\mathbf{C_{L,T}(x) = \operatorname{argmax} P_T(L_T(x) = y)}$$

The differences between the various definitions are then explained through the way they take into account this central tendency. Kong and Dietterich's (*1995*) definition for instance, only takes into account the error of the central tendencies without taking into consideration the strength and the frequency of predictions differing from this central tendency. Variance is defined by the subtraction of the bias component from the total error $\mathbf{P_{Y,X}(L_T(X) \neq Y \mid X=x)}$.

$$\mathbf{bias_{KD} = P_{(Y,X),T}(C_{L,T}(X) \neq Y)}$$

$$\mathbf{variance_{KD} = P_{Y,X}(L_T(X) \neq Y \mid X=x) - bias_{KD}}$$

So, when a learning method has a central tendency for a certain class on a given instance, it doesn't matter how many predictions differ from this tendency and how many different classes are predicted based on that same instance.

Kohavi and Wolpert's definition (*1996*) on the other hand only takes into account what the frequency of the various predicted classes is, regardless of the central tendency. The decomposition originally had a third component for noise, but because this error is usually an unknown it was finally aggregated into the bias component and assumed to be either 0 or 1 (*Kohavi & Wolpert, 1996; Webb, 2000*). The original definition was also proven to give biased estimations (*Kohavi & Wolpert, 1996*). It was shown that estimations based on a small number of training sets lead to higher estimates for bias when compared to estimates based on a higher number of training-sets. Although higher number of training-sets will give better estimates for bias, aspects like computational cost and size of the dataset often demand for a lower number of training-sets. To correct this, they added a correction where **N** is the number of training-sets. The second part of the equation for bias is the correction on the original definition.

$$\mathbf{bias_{KW\ x}^2 = 1/2 \sum_{y \in Y} [P_{Y,X}(Y = y|X = x) - P_T(L_T(x) = y)]^2 - P_T(L_T(x) = y)\left(1 - P_T(L_T(x) = y)\right) / (N - 1)}$$

$$\mathbf{variance_{KWx} = P_{Y,X}(L_T(X) \neq Y \mid X=x) - bias_{KWx}^2}$$

A consequence of this definition is that situations are possible where the strength of the central tendency is equal, but the estimates of bias and variance differ because of a different distribution of deviating classes. Take for instance the situation where a set of 10 models, created by the same learning method predict 6 times class "a", 2 times class "b" and 2 times class "c" on a given test-instance. When compared to another set of 10 models with again 6 predictions for class "a", but this time 4 predictions for class "b" and none for class "c" the estimates for bias and variance will differ. In both cases the central tendency (class "a") is equally strong, but because the distribution of the deviating classes differs, the estimates for bias and variance do as well.

Although this last definition resembles the definition used in numeric regression more closely, it doesn't necessarily provide more or better information. It is not the point which definition is the right one. The point is that each definition differs in the way individual predictions are aggregated to the instance-level. Put differently, each aggregation method communicates different information to the higher level and each type of information has its own purpose (and essentially its own meaning). Kong and Dietterich's (*1995*) definition provides a better insight into the

quality of the central tendency, while Kohavi and Wolpert's (*1996*) definition gives a better insight into overall performance independent of the central tendency. One can imagine that other definitions differ with respect to what extent they take into account the strength of the central tendency and the distribution and frequency of the deviating classes when aggregating to instance-level.

Webb (*2000*) proposes a definition which can be seen as an intermediate between the previous two definitions in the sense that it's based on both the central tendency and the frequency and distribution of the predictions deviating from the central tendency.

$$\mathbf{bias_W = P_{(Y,X),T}\left(L_T(X) \neq Y \wedge L_T(X) = C_{L,T}(X)\right)}$$

$$\mathbf{variance_W = P_{(Y,X),T}\left(L_T(X) \neq Y \wedge L_T(X) \neq C_{L,T}(X)\right)}$$

$$\mathbf{= P_{Y,X}(L_T(X) \neq Y \mid X=x) - bias_W}$$

In the experiments both Webb's and Kohavi & Wolpert's definitions will be used to see if the decomposition of the total error would give new insights which lead to alternative choices of models when compared to selection based on overall performance and stability.

2.3 Comparing overall performance with the bias-variance decomposition

The used indicators for overall performance are defined by the average total error (**ATE**) over **R** runs and the relative standard deviation (**Stdev**) of the total error (**RDE**) is used as measure for stability. The relative measure is being used to normalize the stability with respect to the overall performance.

$$\mathbf{ATE_{LTR} = 1/R \sum_{r \in R} P_{Y,X}(L_{Tr}(X) \neq Y \mid X=x)}$$

$$\mathbf{RDE_{LTR} = Stdev(error_{LTR}) / ATE_{LTR}}$$

Here, a high RDE says something about the stability of the total amount of misclassifications. It doesn't say anything about the stability on an instance level however.

To compare these results with the outcomes of the decompositions the main focus was on the estimates for bias. In the used definitions the variance component was the result of the difference between the total error and the estimate for bias. As such, the variance component wouldn't offer new information. The same two aspects of the bias estimates were used: the average relative bias (**ARB**) and the relative standard deviation of the bias component (**RDB**) over **R** runs, for each definition **d** (note that the results for the different definitions were not aggregated).

$$\mathbf{ARB_{LTdR} = 1/R \sum_{r \in R} bias_{LTdr} / error_{LTr}}$$

$$\mathbf{RDB_{LTdR} = Stdev_{r \in R} (bias_{LTdr} / error_{LTr}) / ATE_{LTR}}$$

The RDB has a mostly superficial meaning in the sense that it isn't directly based on the predictions on the instance level. A high RDB is the consequence of highly varying results between different estimates, where each estimate is an average error of the central tendency, and each central tendency is an 'average' performance on an instance. What 'average' performance exactly means depends on the used definition of bias. So, although the RDB does say something about the stability of a learner and the sensitivity of the definition, I would argue that the impact of the instability of a learner should be minimal due to various averaging effects. And this is what the decomposition procedure tries to achieve.

Based on these measures a comparison will be made between overall performance and estimates for bias. Specific points of interest were the stability of the decomposition with respect to overall stability and average relative bias with respect to overall stability. In order for the decomposition to be useful I argue that: 1. the RDB should be independent from the RDE; and 2. that high RDE should coincide with low ARB.

In the next section more information will be given regarding the experiments.

3. Experiments

To compare the overall accuracy with the estimates for bias 6 learning methods were applied to 81 datasets. This gives a total of 486 learner-dataset combinations each of which were used to create 10 different decompositions and overall accuracy. In each of the 10 runs the only factor changed was the randomization of the data. This resulted in 4.860 decompositions which were then used to calculate the 4 previously defined indicators. From the results the impact of various variables on the stability of the total error and the bias estimate will be discussed:

- The impact of the overall accuracy on the stability of the decomposition and relative size of the bias component.
- The impact of the size of the dataset
- The impact of noise
- The impact of the learning method

3.1 The decomposition-procedure

The bias-variance procedure that's being used is based on the N-times k-fold cross-validation procedure proposed by Webb and Conilione (*2003*). This procedure is

implemented in the *BVDecomposeSegCVSub* function that's been made available for WEKA (*Witten & Frank, 2005*). This implementation gives two different estimations for bias and variance: one based on the definition proposed by Kohavi and Wolpert (*1996*) and one based on the definition initially proposed by Breiman and later modifications by Webb (*Breiman, 1996; Webb, 2000*).

This function uses 10 times 2-fold cross-validation by default. For the experiments only default-settings were used. As a result, each instance in the dataset has been used 10 times for testing. To measure the stability of the procedure, each 10-times 2-fold cross-validation run was repeated 10 times. Each time a different seed was used for randomization of the data.

3.2 Data

The datasets used, consist of both self-generated and UCI datasets. The self-generated datasets were used to be able to control the distributions of the different classes, the amount of observations and the amount of noise. Note that the effects of distributions are not investigated here, but are merely made explicit to keep them constant and not interfere with the results.

Type	Dataset	#instances	#attributes	#classes	Distribution of classes (%)
UCI	hypothyroid	3772	30	4	92%, 5%, 3%, < 1%
	kr-vs-kp	3196	37	2	52%, 48%
	monks-2	601	7	2	66%, 34%
	page-blocks	5473	11	5	90%, 6%, 2% (2), 1%
	segment	2310	20	7	14% (all)
	sick	3772	30	2	94%, 6%
	soybean	683	36	19	13% (4), 6% (2), 3% (9), 2% (3), 1%
	spambase	4601	58	2	61%, 39%
	splice	3190	62	3	52%, 24% (2)
Self	circle	2000/5000/10000	3	2	59%, 41%
	corner	2000/5000/10000	3	2	59%, 41%
	lines	2000/5000/10000	3	2	59%, 41%
	m_een	2000/5000/10000	3	2	59%, 41%
	m_twee	2000/5000/10000	3	2	59%, 41%
	m_drie	2000/5000/10000	3	2	59%, 41%

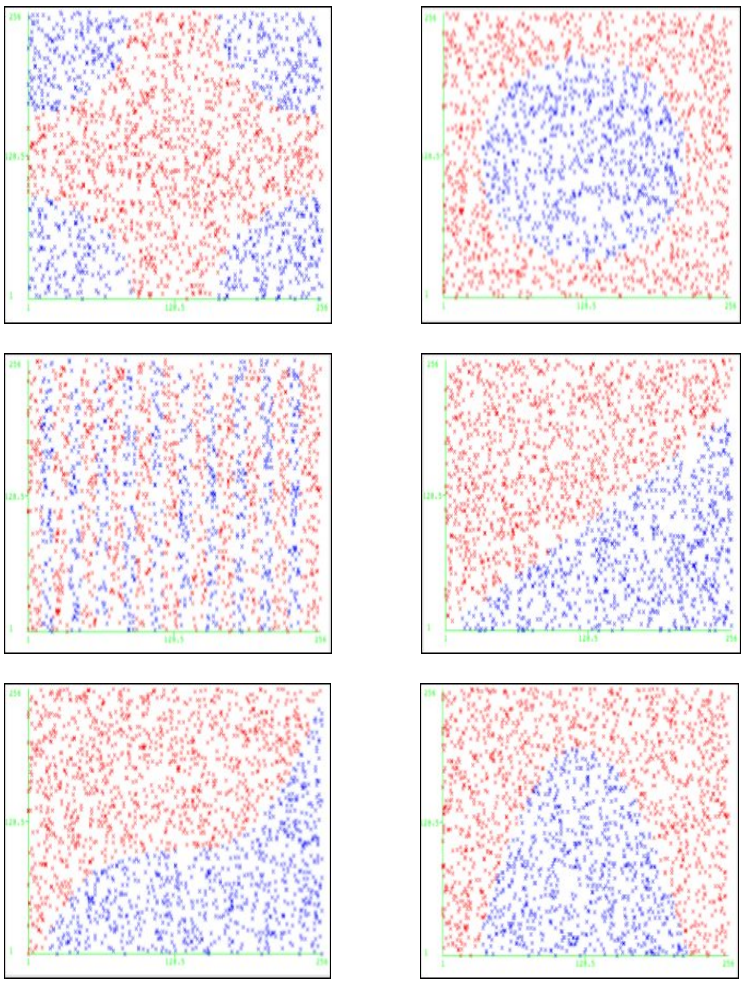
The data from which these self-generated datasets were drawn were based on six 256 by 256 coordinate systems, where in each system coordinates were assigned a class, "A" or "B". There were six different functions used to assign classes.

The datasets were drawn with each instance having equal probability and without replacement. Each instance in the dataset has three attributes: the x-coordinate, the y-coordinate, and a nominal class "A" or "B". Although not representative for real-world data the low dimensionality allows for a simple visualization of the trained

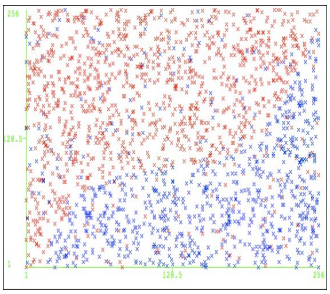
model in contrast to the actual. The proportion of both classes is (virtually) constant over all datasets, 41% class “A” and 59% “B”. The amount of instances varied from 2.000, 5.000 and 10.000 instances. Noise was added by swapping the class of a given percentage of instances. These were 0%, 3%, 5% and 10%. Note that the proportions of classes of the noisy datasets remained 41% and 59%.

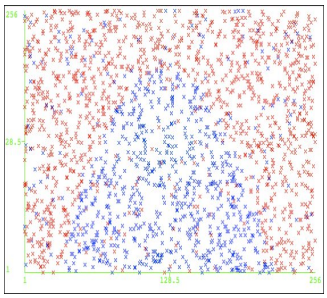
This resulted in total of 6 (relations)* 3 (sizes) * 4 (noise-ratios) = 72 self-generated datasets. With 9 UCI sets, this gave a total of 81 sets.

In the following images the relations of the self-generated sets are visible. The images represent respectively the corner, circle, lines, m_een, m_drie and m_twee datasets (10.000 instances, 0% swap).



The next images are examples of datasets where 10% of the instances are swapped.





3.3 Classifiers

The classifiers used in the first experiments were the WEKA implementations using the default parameters of the following six methods:

(1) NaiveBayes "

(2) Logistic '-R 1.0E-8 -M -1'

(3) ConjunctiveRule '-N 3 -M 2.0 -P -1 -S 1'

(4) J48 '-C 0.25 -M 2'

(5) AdaBoostM1 '-P 100 -S 1 -I 10 -W DecisionStump'

(6) Bagging '-P 100 -S 1 -I 10 -W REPTree -- -M 2 -V 0.0010 -N 3 -S 1 -L -1'

The reasoning behind this set of learners was to have various types of learners some of which are known for their specific bias-variance footprint. Naïve bayes is known to be a stable learner with a relative high bias component of total error. J48 is known for its instability and therefore high relative variance component. And both ensemble learners AdaBoost and Bagging are known for their ability to reduce the bias and variance components with respect to non-ensemble learners. More specifically, AdaBoost is known for its ability to reduce both bias and variance, and Bagging is known for its ability to reduce the variance component (*Webb, 2000*). Logistic and ConjunctiveRule were added to provide some extra references for a better general picture. The ConjunctiveRule learner was a typical mismatch for the self-generated data with a strong bias which should results in low accuracy with a stable and large bias component.

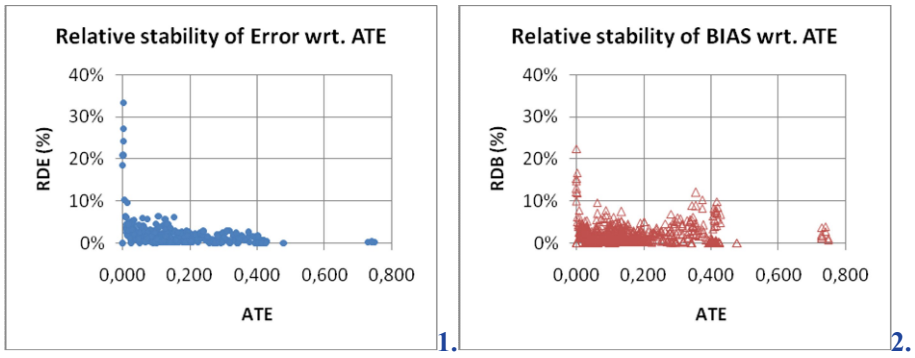
4. Results

One of the reasons for using the bias-variance decomposition is that it provides more information with respect to the stability of a model on the instance-level. In figures 1-3 the impact of overall accuracy (ATE) is shown. From figure 1 and 2 follows that stability of the learning method (RDE) is closely tied to the stability of

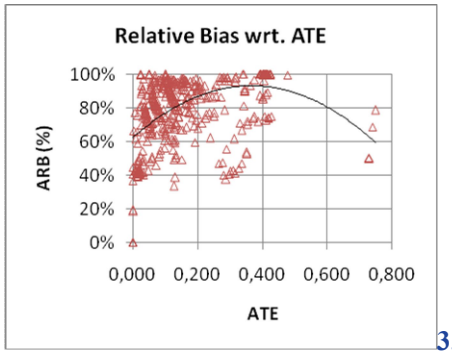
the decomposition (RBD). The only exception is the extra bump of the RBD around the 0,400 mark (figure 2). This bump is entirely due to the ConjunctiveRule learner on the self-generated datasets. Seeing that the self-generated datasets consist of two classes (“A” or “B”) the overall accuracy is close to that of a model which assigns all instances to the same class. An explanation might therefore be that the ConjunctiveRule learner creates models where all the instances are either classified as “A” or “B”. And that different trainsets could result in opposite models. Because the distribution of the classes is close to 40%-60% this may very well be the case. For all other cases, the general result that the RDE is closely tied to RDB remains however.

The relatively high frequency of cases with low average relative bias (ARB) in combination with the low ATE (figure 3) is as expected. Models with low error and a relatively small bias component are the “overfitters” one would be looking for during the model selection stage. So in this sense the decomposition tells us what we want to know. Another observation is that there does seem to be a relation between a lower ARB with a higher RDB (figures 2 and 3). To an extent, the stability of the decomposition is related to the variance component. This leaves the possibility that a single run of the bias-variance decomposition could falsely estimate a relatively high bias when in fact it should be otherwise. Cases with an ATE close to 0,0 are especially suspect. In general “overfitters” will be recognized, but a single run might not be sufficient.

Figure 1 also shows that model selection based on a single run of cross-validation does not appear to be a good way to recognize “overfitters”. Especially models with low ATE are troublesome due to the potentially high RDE. So, model selection entirely based on overall accuracy after cross-validation could be problematic.



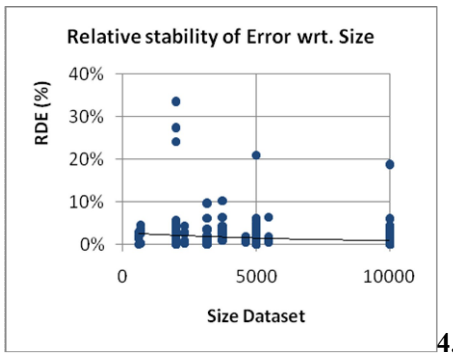
1. 2.



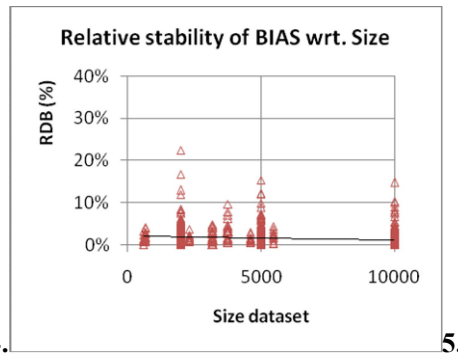
3.

Comparing the RDB to the size of the dataset (figure 5) gave some unexpected results. The expectation was that larger datasets would lead to generally more stable decompositions. Larger datasets allow for larger samples upon which the estimates are based. And the larger the samples, the better the estimates.

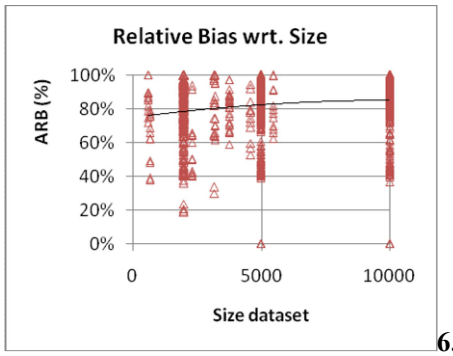
The results show no indication that this is the case. The decomposition seems to be able to work just as well on larger datasets as on smaller datasets. There does seem to be a weak tendency for higher ARB in the case of larger datasets (figure 6). This tendency seems stronger on the UCI data (datasets with size unequal to 2.000, 5.000 and 10.000). Whether the 9 UCI datasets are a good representation for general results however remains to be seen.



4.

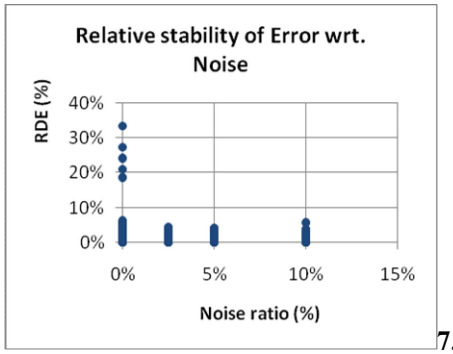


5.

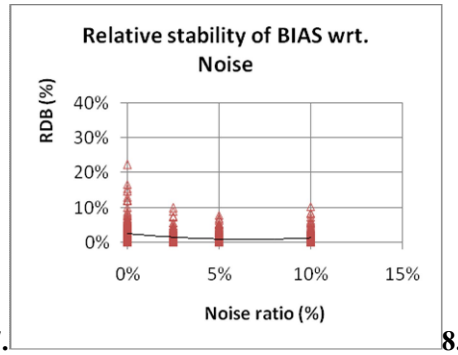


6.

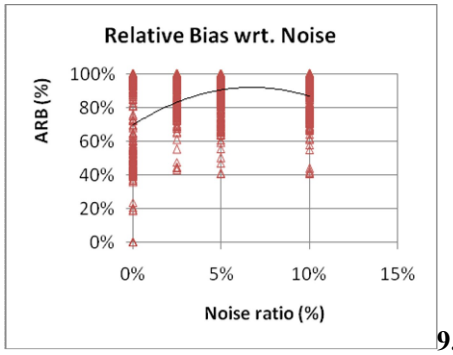
The expected effect of noise was that it would lead to relatively unstable overall accuracy and therefore also relatively unstable decompositions. The biggest instability was shown on datasets with no noise (figure 7 and 8). The explanation is that the cases with an ATE close to 0,0 were typically the cases where the noise was 0% (figure 1 and 7) . Without these extremely unstable cases, the impact of noise is insignificant. The reason why this is the case, is probably due to the kind of noise which was introduced into the data. The noise is randomly distributed over the dataset, and samples drawn from the data will generally have a similar noise ratio as the total dataset. The relations used to create the various classes in the self-generated sets are robust to noise. The predicted classes of the swapped instances will therefore be generally stable, which leaves the overall estimates for bias unaffected (figure 9). If the noise would be unevenly distributed however, another outcome would be expected.



7.



8.



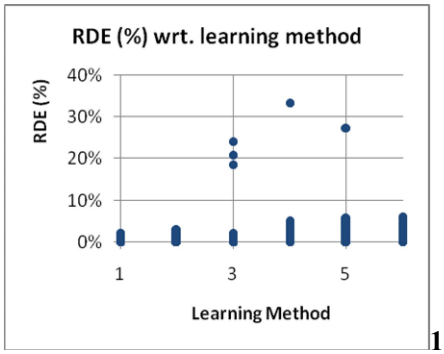
9.

Finally, the results will be compared to the individual learning methods. As mentioned in the section covering the experiments, the expected results are that NaïveBayes has a low RDE, a low RDB, a high ARB and potentially high ATE. ConjunctiveRule is expected to show similar results and J48 should show a sharp contrast with low ATE, high ARB and hopefully low RDB and low RDE. In other words, J48 is the reference “overfitter”. The both ensemble learners are expected to show relatively low ATE, low RDE, low RDB. The ARB of AdaBoost should on

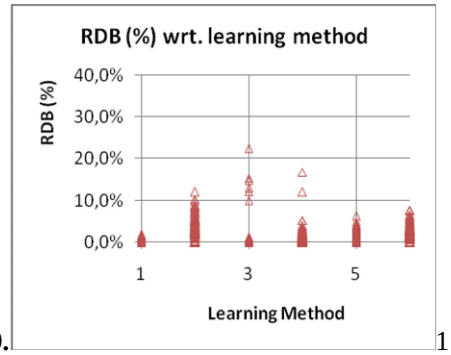
average be lower than the ARB of Bagging due to the better variance reduction of Bagging (*Webb, 2000*).

In figures 10 – 13 the learning methods are numbered according to the following table.

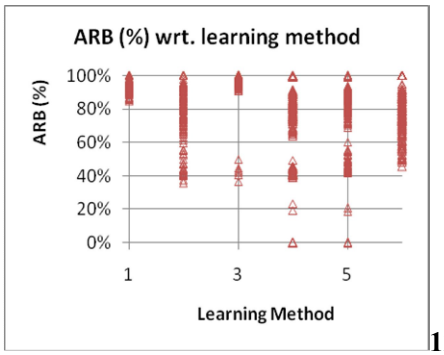
Learning Method	#
NaiveBayes	1
ConjunctiveRule	2
Logistic	3
J48	4
Bagging	5
AdaBoost	6



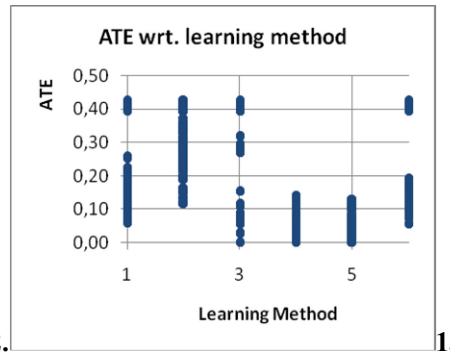
10.



11.



12.



13.

According to these results, NaiveBayes (#1) behaves exactly as expected while ConjuntivRule (#2) does show some high RDB and low ARB. Which is odd, because the low ARB results are mainly due to the self-generated data and the expectations were that ARB would be relatively high on these cases. These unexpected results are similar to the ones from figure 2. The given explanation for results in figure 2 hold here as well: the models either predict all instances to be of class “A” or “B” which is due to sampling of the decomposition.

The Logistc learner (#3) shows varied results. In general the RDE and RDB are even lower than NaiveBayes, but the exceptions are more extreme. These extreme results coincide with low ARB and they come from 3 datasets: splice, soybean and m_een. On m_een, which is a linear relation, the ATE is close to 0,0 while on splice and soybean are close to 0,15. The extreme reactions of the RDE might be the consequence of the sensitivity of the RDE to low ATE. When the ATE is close to zero, even a small value for RDE could result in extreme behavior. The same reasoning could explain the high RDB. An explanation for the low ARB might be that because of the low error-rate the misclassifications that are being made are unstable. The few misclassifications that are being made, are not made on the exact same instances. This example shows that low bias in combination with very low ATE does not necessarily imply that the learner gives unstable results. In other words, a low ARB could be the result of a small amount of "random" errors in combination with a low error ($error_{LTr}$, not the ATE!)**.** The results on splice and soybean show low RDE and RDB in combination with a low ARB. An explanation for the low bias was not found, but I suspect this is due to aspects of the data itself (many attributes and uneven distribution of classes in combination with a weak relationship).

The results on J48 (#4) are as expected: low ARB and low ATE next to relatively low RDB and RDE. The only signs of instability are due to the cases where the ATE is close to 0,0. Generally speaking, this means that the bias-variance decomposition is useful for recognizing stable (in terms of RDE!) "overfitters".

Both ensemble learners show slightly disappointing results. Bagging (#5) shows very similar results to J48 (#4). Purely on the basis of the ATE, Bagging would be the preferred learner on most datasets (figure 13). But the low ARB (figure 12) in combination with the low RDB (figure 11), are the signs of a typical "overfitter". To an extent, this might be due to the same effect the Logistic learner suffered from: a few random misclassifications in combination with a small error could result in a relatively low bias. What's different here is that the RDB is low as well, so the estimates for bias are fairly stable. The relatively frequent low ARB for AdaBoost (#6) is disappointing as well. Especially when there are no strong signs of instability of the RDB (figure 11). To an extent these results may confirm the suspects made by Webb and Conilione (*2003*) and Bouckaert (*2008*) who argued that previous results using the bias-variance decomposition may have suffered from the instability of the estimates. Signs of instability are not present in the present results with respect to both ensemble learners however, but this could be explained by the improved stability of the used decomposition procedure (*Webb & Conilione, 2003*). It does put into question the effectiveness of the used ensemble learners.

5. Conclusion

The following conclusions are drawn from the results:

- Although the bias-variance decomposition is able to recognize unstable learners, where normal accuracy indicators based on cross-validation can not, the results on cases with extreme low error can be very unstable. Relative measures like the ARB can also be problematic in the context of ATE close to 0,0. The relative bias would therefore not always give representative results.

- The size of the dataset (and therefore the size of the samples) didn't have a huge impact on the stability of the decomposition. This seems to contradict the results from Bouckaert (*2008*). And although it is not made clear in that article, I suspect his results were mostly due to the different sampling methods (bootstrap, sampling with replacement). If this is the case, this would show that the sampling method has a bigger impact in the stability than the size of the dataset.

- Randomly distributed noise, as used in these experiments, does not have a huge impact on the stability of the bias-variance decomposition. I suspect that unevenly distributed noise will lead to more instability. This has not been tested however.

When comparing the overall performance based on cross-validation with results from the bias-variance decomposition there is a strong case for using the decomposition. When comparing different learners using the decomposition one would be tempted to use relative measures for bias and variance similar to the ones used in this paper. It is shown that under certain conditions wrong conclusions could be drawn from these measures. In the cases where the overall error is close to 0,0 the relative measures could misrepresent the relative components of the bias-variance decomposition. In the other cases however no problems were found. In those cases it useful to have the decompositions of a NaiveBayes and a J48 learner as references for respectively biased stable learners, and unbiased unstable learners.

6. References

Brain, D., Webb, G.I. (2002). The need for low bias algorithms in classifications learning from large data sets. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2002)*, pp. 62-73, Berlin. Springer-Verlag.

Breiman, L. (1996). Bias, Variance, and Arcing classifiers. Technical Report 460, Statistics Department, University of California, Berkeley, CA.

Bouckaert, R. R. (2008). Practical Bias Variance Decomposition. In *Proc 21st Australasian Joint Conference on Artificial Intelligence Auckland, New Zealand.*

Dietterich, T.G., Kong, E.B. (1995). Machine Learning Bias, Statistical Bias, and Statistical Variance of Decision Tree Algorithms. (?)

Domingos, P. (2000). A Unified Bias-Variance Decomposition for Zero-One and Squared Loss. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pp. 564-569, Austin, TX.

Geman, S., Bienenstock, E., Doursat, R. (1992). Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4, pp. 1-58. Massachusetts Institute of Technology.

Gordon, D. F. , Des Jardins, M., Dietterich, G. (1995). Evaluation and Selection of Biases in Machine Learning. *ACM Computing Surveys*, 4, pp. 255-306.

James, G., Hastie, T. (1997) Generalizations of the Bias/Variance Decomposition for Prediction Error. (?)

Kohavi, R. (1995). A study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *(?)*

Kohavi, R. Wolpert, D. (1996). Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 275-283, San Fransisco. Morgan Kaufman.

Kong, E.B., Dietterich, T.G., (1995). Error-Correcting Output Coding Corrects Bias and Variance. (?)

Mitchell, T.M. (1980). The need for biases in learning generalizations. Tech. rep. CBM-TR-117, Rutgers University, New Brunswick, NJ.

Tibshirani, R. (1996) Bias, variance and prediction error for classification rules. University of Toronto. Canada.

Van der Putten, P., van Someren, M., (2004). A Bias-Variance Analysis of a Real World Learning Problem: the CoIL Challenge 2000. *Machine Learning*, 57, pp. 177-195.

Webb, G.I. (2000). Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, 40(2), pp. 159-196.

Webb, I.G., Conilione, P. (2003). Estimating Bias and Variance from Data. (unpublished)

Witten, I.H., Frank, E. (2005). Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufman, San Fransisco, CA.