Chapter 6
Making Early Predictions of the Accuracy of Machine Learning Classifiers

James Edward Smith, Muhammad Atif Tahir, Davy Sannen, and Hendrik Van Brussel

Abstract The accuracy of machine learning systems is a widely studied research topic. Established techniques such as cross validation predict the accuracy on unseen data of the classifier produced by applying a given learning method to a given training data set. However, they do not predict whether incurring the cost of obtaining more data and undergoing further training will lead to higher accuracy. In this chapter, we investigate techniques for making such early predictions. We note that when a machine learning algorithm is presented with a training set the classifier produced, and hence its error, will depend on the characteristics of the algorithm, on training set's size, and also on its specific composition. In particular we hypothesize that if a number of classifiers are produced, and their observed error is decomposed into bias and variance terms, then although these components may behave differently, their behavior may be predictable. Experimental results confirm this hypothesis, and show that our predictions are very highly correlated with the values observed after undertaking the extra training. This has particular relevance to learning in nonstationary environments, since we can use our characterization of bias and variance to detect whether perceived changes in the data stream arise from sampling variability or because the underlying data distributions have changed, which can be perceived as changes in bias.

J.E. Smith (✉)
Department of Computer Science and Creative Technologies, University of the West of England, Bristol, BS16 1QY, UK
e-mail: james.smith@uwe.ac.uk

M.A. Tahir
School of Computing, Engineering and Information Sciences, University of Northumbria, Newcastle, UK
e-mail: muhammad.tahir@northumbria.ac.uk

D. Sannen • H.V. Brussel
Department of Mechanical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium
e-mail: davysannen@gmail.com; hendrik.vanbrussel@mech.kuleuven.be

M. Sayed-Mouchaweh and E. Lughofer (eds.), *Learning in Non-Stationary Environments: Methods and Applications*, DOI 10.1007/978-1-4419-8020-5_6,

6.1 Introduction

Predicting the accuracy of a trained machine learning system when presented with previously unseen test data is a widely studied research topic. Techniques such as cross validation are well established and understood both theoretically and empirically, e.g., [18,34]. However, these techniques predict the accuracy on unseen data *given the existing training set*. For example, N-fold Cross Validation (NCV) averages the fitness estimated from N runs, each using a proportion $1 - 1/N$ of the available data to train a classifier and $1/N$ to evaluate it. Therefore, repeating with different values of N can give the user some indication of how the error rate changed as the training set increased to the current size, since lower values of N effectively equate to smaller training sets. However, NCV does not predict what accuracy might be achievable after further training. Thus if the current accuracy is not acceptable, and obtaining data comes at cost, NCV and similar techniques do not offer any insights into whether it is worth incurring the cost of further training.

This is of more than theoretical interest, because the successful application of machine learning techniques to "real-world" problems places various demands on the collaborators. Not only must the management of the industrial or commercial partner be sufficiently convinced of the potential benefits that they are prepared to invest money in equipment and time but, vitally, there must also be a significant investment in time and commitment from the end-users in order to provide training data from which the system can learn. This poses a problem if the system developed is not sufficiently accurate, as the users and management may view their input as wasted effort, and lose faith with the process.

In some cases this effort may be re-usable if, for example, the user has been labeling training examples that can be stored in their original form, and which come from a fairly stationary distribution. However, this is frequently not the case. For example, in many applications it may not be practical to store the physical training examples rather, it is necessary to characterize them by a number of variables. If the failure of the Machine Learning system in such cases stems from an inappropriate or inadequate choice of descriptors, then the whole process must be repeated. Not only has the user's input been a costly waste of time and effort but there also may be a loss of faith in the process which can manifest in reduced attention and consistency when classifying further samples. To give a concrete example from the field of diagnostic visual inspection (e.g., manufacturing process control or medical images), it frequently turns out that it is not sufficient to store each relevant image—other information is necessary such as process variables, or patients' history. If this data is not captured at the same time, and is not recoverable post-hoc, then the effort of collecting and labeling the database of examples has been wasted.

A significant factor that would help in gaining confidence and trust from end-users would be the ability to quickly and accurately predict whether the learning process was going to be successful. Perhaps more importantly from a commercial viewpoint, it would be extremely valuable to have an early warning that the users can save their effort while the system designer refines the choice of data, algorithms etc.

From the perspective of learning in nonstationary environments, such a tool could provide a number of advantages. Firstly, a deviation from the expected progress can be used as an indicator that there has been a fundamental shift in the nature of the training input provided to the algorithm. This could arise either because the user providing the labels has changed, or the underlying data set is dynamic and requires a classifier that is able to take account of this. In either case, early warning is needed to enable corrective action to be taken.

In this chapter, we investigate a technique for making such early predictions of future error rates. We will consider that we are given n samples, and that the system is still learning and refining its model at this stage. We are interested in predicting what final accuracy might be achievable if the users were to invest the time to create n' more samples. This leads us to focus on two questions. First, what are the most appropriate descriptors of the system's behavior after some limited number n of samples, and then later after an additional n' samples? Second, is it possible to find useful relationships for predicting the second of these quantities from the first?

Theoretical studies, backed up by empirical results, have suggested that the total error rate follows a power–law relationship, diminishing as extra training samples are provided. While these theoretic bounds on error are rather loose, they provide motivation for investigating practical approaches for quickly and reliably estimating the error rate that may be observed after future training. In general the error will be a complicated function, but the hypothesis of this chapter is that we can deal with it more easily if we decompose it into a number of more stable functions. Therefore this chapter concentrates on the use of the well-known bias–variance decomposition [8, 21] as a source of predictors when an algorithm is used to build a classification model from a data set. Specifically, our hypothesis is that if the observed error is decomposed into bias and variance terms, then although these components may behave differently, their behavior may be individually predictable.

To test our hypothesis we first apply a range of algorithms to a variety of data sets, for each combination periodically estimating the error components as more training samples are introduced, until the full data set has been used. All of the data arising from this (rather lengthy) process is merged and regression analysis techniques are applied to produce three sets of predictive models—one each for bias, variance and total error. Each of these models takes as input a measurement obtained from the classifier produced when only a few samples (n) from a data set have been presented to the learning algorithm, and predicts the value after all samples have been applied ($n + n'$). As the data have been merged, the intention is that these models are algorithm-data set independent. We examine the stability and valid range of these models using simple linear regressors. Moving on to consider trainable ensembles of different classifiers, we show how a similar approach can be applied to obtain estimates on the upper bound of the achievable accuracy, which can predict the progression of the ensemble's performance.

The rest of this chapter proceeds as follows. In Sect. 6.2, we review related work in the field, including the bias–variance decomposition of error that we will use. Following that, Sect. 14.2.1 describes the experimental methodology used to collect the initial statistics, and test the resulting models. Section 6.4 describes

and discusses the results obtained. In Sect. 6.5, we show how this approach may be extended to predict the future accuracy of trainable ensembles of classifiers. In Sect. 6.6, we discuss how these methods could be applied to detect changes in underlying data distributions that would trigger re-learning in no stationary environments. Finally in Sect. 6.7, we draw some conclusions and suggestions for further work.

6.2 Background

6.2.1 Notation

For the sake of clarity we will use a standard notation throughout this chapter, reinterpreting results from other authors as necessary.

We assume classification tasks, where we are given an instance space X and a predicted categorical variable Y. The "true" underlying function F is a mapping $F : X \to Y$.

Let D be the set of all possible training sets of size n sampled from the instance space X, and $d \in D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$.

When a machine learning algorithm C is presented with d it creates a classifier, which we may view as a hypothesis about the underlying mapping: $H_{Cd} : X \to Y$. The subscripts C and d make it explicit that the specific classifier H induced depends on the learning algorithm and the training set. For a specific learning algorithm C, the set of classifiers that it can induce is denoted $\mathscr{H}$.

We consider a $0/1$ misclassification error—in other words the error is zero if H correctly predicts the true class of an item $x \in X$, and 1 otherwise. More formally, the misclassification cost of a single data item x with a specific classifier H is:

$$\text{Cost}(H_{Cd}, x) = \begin{cases} 0 & H_{Cd}(x) = F(x) \\ 1 & H_{Cd}(x) \neq F(x) \end{cases}. \tag{6.1}$$

The expected error of the classifier created from n data points is then given by integrating over X and d, taking into account their conditional likelihood, i.e.:

$$\text{Error}(H_{Cn}, X) = \int_{x \in X, d \in D} P(x)P(d|n)\text{Cost}(H_{Cd}, x), \tag{6.2}$$

where $P(d|n)$ is the probability of generating a specific training set $d \in D$ given the training set size n, and $P(x)$ is the probability of selecting an item $x \in X$ to be classified. In practice of course it is not possible to exactly measure the true error, so approaches such as bootstrapping, hold-out, and cross validation are used to estimate the error, given a finite sized set of examples. In *bootstrapping*, new data sets are repeatedly generated from the original data set using random sampling

with replacement. The new data sets, which most likely contain duplicate examples, are then used to train a classifier and the examples that are not part of the data sets are used for testing. *Hold-out* approaches divide the available data into two sets (typically using a 70/30 split), train a classifier using the larger set and then estimate its accuracy using the "unseen" smaller set. As described above, *N-fold cross validation* (NCV) is an improvement on the hold-out approach which aims to avoid the possibility of accidentally selecting an "easy" test set. The available data is split into N (usually equally sized) blocks. Each of the blocks in turn is then used as a test set to estimate the accuracy of a classifier built from the remaining $N-1$ blocks. The average of these is then used as an estimate of the accuracy of the classifier that would be built from all of the available data. We will use the lower case "error" to denote an estimation is being used for the true error.

6.2.2 Relationship to Other Work

Cortes et al. [10] presented an empirical study where they characterized the behavior of classification algorithms using "learning curves". These suggest that the predicted error of the classifier after n samples have been presented will follow a power–law distribution in n:

$$\text{error}(n) = an^{-\alpha} + b, \tag{6.3}$$

where the constants a (the learning rate), α (the decay rate), and b (the asymptotic Bayes error rate) depend on the particular combination of classification algorithm and data set, but α is usually close to, or less than one. This suggests that given a particular classifier–data set combination, it should be possible to commence training, take periodic estimates of the error as n increased, and then use regression to find values for a, b, α that fit the data, and can be used for predicted future error rates. "Progressive sampling" uses training sets ("samples") with progressively larger sizes (i.e., increasing n) until some desired accuracy has been reached. This can be inefficient if a larger number of "samples" is used as each must be evaluated. Using a similar approach to Cortes et al. recent papers have attempted to fit a learning curve to a few samples in order to predict the size needed [23, 29]. Mukerhjee et al. [26] have pointed out a problem with this curve-fitting approach, namely that for low values of n the estimated error rates are subject to high variability, which leads to significant deviations when fitting the power–law curve. They have presented an extension of the method which uses a "significance permutation test" to establish the significance of the observed classifier error prior to curve fitting.

These results fit in with theoretical bounds from "Probably Approximately Correct" (PAC) theory such as those presented by Vapnik in [35]. These begin with the assumption that a training set $d = \{x_k, y_k\}, 1 \leq k \leq n, y_k \in \{-1, 1\}$ is drawn independently and identically distributed (iid) from a data set, and that future training and test data will be drawn from the data set in the same way.

Given the restriction $Y = \{-1, 1\}$, the test error $\text{Error}(H_{Cn})$, (the probability of misclassification) is defined to be:

$$\text{Error}(H_{Cn}) = E\left[\frac{1}{2} \mid F(x) - H_{Cn}(x) \mid\right], \tag{6.4}$$

where in comparison to (6.1), the division by two maps absolute values of differences in Y onto costs (rather than having a fixed cost of 1 for misclassification), and the H_{Cn} denotes that we are taking the expectation for the general case. The current empirically measured training error $\text{error}(H_{Cd})$ is:

$$\text{error}(H_{Cd}) = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{2}\left|y_k - H_{Cd}(x_k)\right|. \tag{6.5}$$

Note that since this estimates the error by classifying the n elements of the training set with a classifier trained on that data, it is calculated as a summation and will underestimate the true error. Vapnik showed that the amount of underestimation can be bounded [35]. If ψ represents the Vapnik–Chervonenkis (VC) dimension, and $0 \leq \eta \leq 1$, then with probability $1 - \eta$:

$$\text{Error}(H_{Cn}) \leq \text{error}(H_{Cd}) + \sqrt{\frac{\psi\log(2n) + \psi(1 - \log\psi) - \log(\eta/4)}{n}}. \tag{6.6}$$

Effectively this equation makes explicit an assumption that machine learning algorithms inherently produce classifiers which overfit the available training data. The VC-dimension ψ is a measure of the capacity of a hypothesis space of classification algorithm C, so may be thought of as the "power" of C. It is the maximum number of points that can be arranged so that C can always "shatter" them—for example, the VC-dimension of a linear classifier such as a perceptron is three, since no straight line can separate the four points of an XOR problem. Equation (6.6) makes it clear that more powerful algorithms (higher ψ) are more likely to over-fit the data, and so it may be used as grounds to select between two algorithms which produce the same training error but have different complexity (related to ψ). It also makes explicit the dependency on n: for a given training set error, the maximum amount by which this will underestimate the true error decreases by approximately $\sqrt{\psi\log n/n}$.

However, in practice these bounds tend to be rather "loose." There have been other more recent developments in Statistical Learning Theory which use a similar approach but exploit Rademacher complexity to provide tighter bounds, such as those in [1–3, 27]. Common to all of these approaches, as with the use of VC-dimension results, is the idea that on the basis of the available training data, an algorithm selects a classifier H_{Cd} from some class $\mathscr{H}$ available to it. To analyze the learning outcomes, the "error" observed when the training data is classified by H_{Cd} is broken down into the Bayes optimal error (which cannot be avoided) plus an amount by which best ($H* \in \mathscr{H}$) in the current class of classifiers would be more

than Bayes optimal, plus an amount by which the classifier H_{Cd} currently estimated by the algorithm to be "best" is different to the actual best $H*$. Thus for example, approaches such as *Structural Risk Minimisation* can be thought of as principled methods for increasing the size/complexity of the current class of classifiers $\mathscr{H}$ until it includes the Bayes optimal classifier.

The underlying assumption is that the error is estimated using the current training set, and that this almost certainly overfits the true underlying distribution (i.e. $H_{Cd} \neq H*$) so the current estimates of error for the chosen classifier H_{Cd} will be less than the "true" error that would be seen if it was applied to the whole data distribution. Therefore, bounds are derived which describe the extent to which the error on the training set underestimates the true error. Since this can be described in terms of the search problem of identifying $H* \in \mathscr{H}$, it is understandable that they take into account the amount of information available to the search algorithm—i.e., the size n of the training set.

While this is a valid and worthwhile line of theoretical research, we would argue that it is not currently as useful for the practitioner. Consider the example of a user who is highly skilled in his/her domain, but knows nothing about Machine Learning, and is providing the training examples from which a classifier is constructed. The theory above effectively says: *"Based on what you have told me, I've built a classifier which seems to have an error rate of x%. I can tell you with what probability the "true" error rate is worse than $x + y$%, for any positive y."* If they have provided enough labeled data items to create what appears to be an accurate classifier, then this is valuable. However, if they are still early on in the process, and the current error rates are high, it gives no clues as to whether they will drop. Instead we attempt to provide heuristics that answer a different question: *"Based on what you have told me, I've built some classifiers and although the current error rate is x% it will probably drop to y%, where $y \leq x$".*

To do this, we note that the analysis above relates the true test error to a specific estimated error from a given training set size, and that the variance in the predicted error depends strongly on n. This has prompted us to examine different formulations that explicitly decompose the error into terms arising from the inherent bias of the algorithm (related to its VC dimension, or to the difference between $H*$ and the Bayes optimal classifier) and the variability arising from the choice of $d \in D$.

6.2.3 Bias–Variance Decomposition

A number of recent studies have shown that the decomposition of a classifier's error into bias and variance terms can provide considerable insight into the prediction of the performance of the classifier [8, 21]. Originally, it was proposed for regression [17] but later, this decomposition has been successfully adapted for classification [8, 21, 31]. While a single definition of bias and variance is adopted for regression, there is considerable debate about how the definition can be extended to

classification [5,12,16,19,21,22]. In this chapter, we use Kohavi and Wolpert's [21] definition of bias and variance on the basis that it is the most widely used definition [37,38], and has strictly nonnegative variance terms.

Kohavi and Wolpert define bias, variance and noise as follows [21]:

Squared Bias *"This quantity measures how closely the learning algorithm's average guess (over all possible training sets of the given training set size) matches the target."*

Variance *"This quantity measures how much the learning algorithm's guess bounces around for the different training sets of the given size."*

Intrinsic noise *"This quantity is a lower bound on the expected cost of any learning algorithm. It is the expected cost of the Bayes-optimal classifier."*

Given these definitions, we can restate (6.2) as:

$$\text{Error}(H_{C,n}) = \int_{x \in X} P(x) \left(\sigma_x^2 + \text{Bias}_x^2 + \text{Variance}_x\right). \tag{6.7}$$

Assuming a fixed cardinality for Y (finite set of classes), and noting D has finite cardinality, the summation terms in the integral are:

$$\text{Bias}_x^2 = \frac{1}{2} \sum_{d \in D} P(d|F,n) \sum_{y \in Y} \left[P(F(x) = y) - P(H_{Cd}(x) = y)\right]^2,$$

$$\text{Variance}_x = \frac{1}{2} - \frac{1}{2} \sum_{y \in Y} \sum_{d \in D} P(d|F,n) P(H_{Cd}(x) = y)^2,$$

$$\sigma_x^2 = \frac{1}{2} - \frac{1}{2} \sum_{y \in Y} P(F(x) = y)^2,$$

where the terms $P(F(x) = y), P(H_{Cd}(x) = y), P(d|F,n)$ make explicit that some terms are conditional probability distributions since the Bayes error may be nonzero, the classification output may not be crisp, and the specific choice of training set depends on the underlying function and the number of samples.

In practice, these values are estimated from repeated sampling of training sets to acquire the necessary statistics, which are then manipulated to give the different terms. Thus, the Bias term considers the squared difference between the actual and predicted probabilities that the label is y for a given input x for a given training set d. To calculate its value, the inner term sums over all possible values of y, and then the outer summation averages over all training sets of a given size. By comparison, the Variance term just considers the distribution of predicted values, $P(F(x) = y)$, but reverses the order of the summation to emphasize the effect of different training sets. The intrinsic noise term sums over all possible output values y the squared probabilities that the actual target $F(x) = y$ for a given input x. If the underlying class boundaries are crisp, then $P(F(x) = y)$ will be zero except for one value of y, the summation will be 1 and σ_x^2 will consequently be zero.

6.2.4 Bias as an Upper Limit on Accuracy

An alternative perspective on the analysis in Sect. 6.2.3 is that the bias term reflects an inherent limit on a classifier's accuracy resulting from the way in which it forms decision boundaries. For example, an elliptical class boundary can never be exactly replicated by a classifier which divides the space using axis-parallel decisions. A number of studies have been made confirming the intuitive idea that the size of variance term drops as the number of training samples increases, whereas the estimated bias remains more stable, e.g., [8]. Therefore, we can treat the sum of the inherent noise and bias terms as an upper limit on the achievable accuracy for a given classifier. Noting that in many prior works it is assumed that the inherent noise term is zero, and that for a single classifier it is not possible to distinguish between inherent noise and bias, we hereafter adopt the convention of referring to these collectively as bias.

6.3 Experimental Methodology

The hypothesis of the main part of this chapter is that values of the bias and variance components estimated after n training samples can be used to provide accurate predictions for their values after $n + n'$ samples, and hence for the final error rate observed. To do this prediction, we use statistical models built from a range of data set-algorithm combinations. The following sections describe our choice of experimental methodology, algorithms, and data sets.

6.3.1 Procedure for Building the Models

Our experimental procedure is as follows:

- For each data set x and classifier i, we estimated the values of error (e_{ixn}), bias (b_{ixn}) and variance (v_{ixn}) components using the first $n \in \{100, 200, \ldots, 1{,}000\}$ samples.
- For each data set, we then estimated the values of error, bias, and variance using all of the samples in the data set. Note that this results in different values of n' for different data sets. Note also that we do not use a separate "test set." We consider that since one is always making estimates of the error on unseen data it is more consistent to relate estimates of the error at different points in training *using the same estimation methodology*.
- After all of the collected data was pooled, we applied linear regression to create models of the form $Q_{(n+n')} = a_{Qn} \cdot Q_n + b_{Qn}$, where Q is one of bias, variance, or total error. In these models Q_n is the independent variable, $Q_{(n+n')}$ the dependent

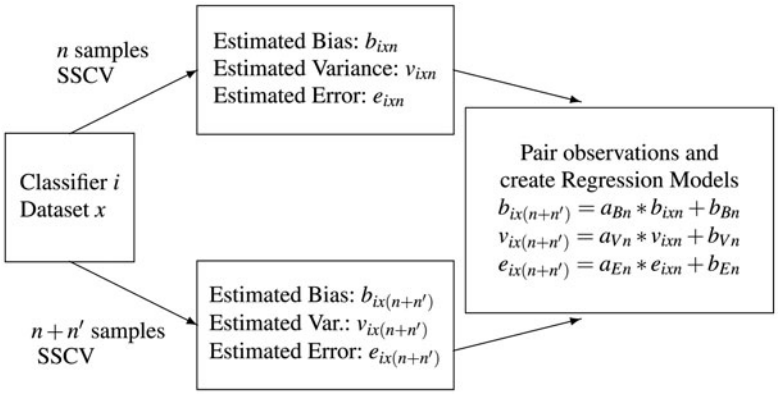


Fig. 6.1 Methodology for creating predictive models. This is repeated for $n \in \{100, 200, \ldots, 1{,}000\}$

variable and the constants a_{Qn} and b_{Qn} are estimated by the linear regression procedure for each variable $Q \in \{\text{total error, bias, variance}\}$ and for each value of n. We compute the coefficient of determination R^2 to measure how well the simple linear model explains the variability of the independent variable, and hence the quality of the predictions—the closer R^2 is to 1, the better is the prediction.

- Note that when used with a new classifier i or data set x, this gives us two ways of predicting the final error $e_{ix(n+n')}$ based on the first n samples: either directly from the observed error or by summing the predictions for bias and variance.
  - In the first case there is one independent variable, so $e_{ix(n+n')} = a_{En} \cdot e_{ixn} + b_{En}$, where a_{En} and b_{En} are taken from our models.
  - In the second case, the two decomposed components (bias b_{ixn} and variance v_{ixn}) are treated as independent variables, i.e. $e_{ix(n+n')} = a_{Bn} \cdot b_{ixn} + a_{Vn} \cdot v_{ixn} + b$, where $b\ (= b_{Bn} + b_{Vn})$, a_{Bn}, and a_{Vn} are given by our models.

Figure 6.1 shows this process for a single value of n.

We would like to re-iterate for the sake of clarity that we are not building models which relate error, bias, and variance as a function of the number of training samples n. In that case, it would certainly be true that by the two models (bias as a function of n) and (variance as a function of n) could be combined into a single linear model (error as a function of n). As the wealth of theoretical work described above shows, there is ample evidence to suggest that no simple predictive linear model exists. Instead we are building and combining linear models of the form *future bias/variance/error as a function of current bias/variance/error* and seeing how the predictive power of these models changes as a result of the value of n.

These linear models are of course an extremely simple way of modeling the relationship between our various predictors; more sophisticated techniques exist

in the fields of statistics and also Machine Learning, and will be examined in a later section. However, as the results will show, linear models are sufficient for our purposes.

6.3.2 Choice of Classifiers

In order to obtain the data for modeling ten different classification algorithms were selected, each with different bias and variance characteristics. These were: Naive Bayes [13], C4.5 [30], Nearest Neighbor [11], Bagging [4], AdaBoost [15], Random Forest [6], Decision Table [20], Bayes Network [13], Support Vector Machine [28], and Ripple-Down Rule learner [39]. Note that this set includes two methods for creating ensembles: AdaBoost (using Decision Stumps as the base classifier) and Bagging (using a decision tree with reduced error pruning). In these cases, since we are solely interested in the outputs, we treat the ensemble as a single entity, rather than attempt a bias–variance–noise–covariance decomposition [9]. For all these classifiers, the implementation in the *WEKA* library [39] is used, and the default parameters in WEKA are used for each classifier. We also used WEKA's Java implementation of Kohavi and Wolpert's definition of Bias and Variance (*weka.classifiers.BVDecompose*).

6.3.3 Data Sets

The data collection required to build the statistical models is carried out on data sets derived from four Artificial and five real-world visual surface inspection problems from the European DynaVis project[1] [14, 25]. Each artificial problem consists of 13,000 contrast images created by a tuneable randomized image generator. Class labels (good/bad) were assigned to the images by using different sets of rules of increasing complexity acting on the generator. The real-world data sets came from CD-imprint and egg inspection problems. There are 1,534 CD images, each labeled by four different operators, and 4,238 labeled images from the egg inspection problem. The same set of image processing routines are applied to segment and measure regions of interest (ROI) in each image. From each set of images are derived two data sets. The first has 17 features describing global characteristics of the image and the ROI it contains. In the second, these are augmented by the maximum value (over all the ROI) for each of 57 ROI descriptors. Adding the labels available provides a total of 18 different data sets with a range of dimensionality and cardinality.

[1] www.dynavis.org

To build the models, we used 14 of the data sets: the six derived from the first three artificial image sets, the six from the CD images labeled by the first three operators and the two from the egg data. The remaining four data sets, derived from the fourth artificial image set, and the CD labeled by Operator 4 are reserved for evaluation purposes. In each case we took $n' = \text{total\_set\_size} - 1{,}000$, so n' differs between data sets.

6.3.4 Prediction Methodology and Sampling Considerations

To create the model data, we repeatedly draw training and test sets from the n samples from which we can estimate the total error, together with its bias and variance components. This raises the issue of how we should do this repeated process.

If the variables in X are continuous, or unbounded integers, then the underlying distribution over which the classifier may have to generalize is of course infinite. For bounded integer or categorical variables, the number of potential training sets of size n drawn iid from an underlying distribution of X is of size $|D| = |X|!/n!(|X| - n)!$, so in practice even for nontrivial data sets it is not possible to evaluate all possible training sets d of size n. However the success (or otherwise) of the approach proposed in this chapter depends on the accuracy with which we can predict error components, particularly for when the training set sizes are low. This immediately raises the question of finding the most appropriate methodology for estimating the values of those quantities. To give a simple example of why this is important, a later result in this chapter partially relies on being able to distinguish between those data items that are *always* going to be misclassified by a given classifier, and those which will *sometimes* be misclassified, depending on the choice of training set. Since the well known N-fold cross validation approach only classifies each data item once, it does not permit this type of decomposition and cannot be used. In a preliminary paper [33], we have examined two possible approaches: the "hold-out" method proposed by Kohavi and Wolpert [21] and the "Sub-Samples Cross Validation" (SSCV) method proposed by Webb and Conilione [38]. The latter have argued that the hold-out approach proposed in [21] is fundamentally flawed, partly because it results in small training sets, leading to instability in the estimates it derives. This was confirmed by our results [33] which showed that the stability of the estimates, and hence the accuracy of the resulting prediction was far higher for the SubSampling method. Therefore, we restrict ourselves to this approach.

The SSCV procedure is designed to address weaknesses in to both the hold-out and bootstrap procedures by providing a greater degree of variability between training sets. In essence, this procedure repeats N-fold CV l times, thus ensuring that each sample x from the training set of size n is classified l times by the classifier i. The true b_{ix} and v_{ix} can be estimated as b_{ixn} and v_{ixn} from the resulting set of classifications. The final bias and variance is estimated from the average of all $x \in D$ [37, 38], thus using all n' samples.

6.4 Explanatory Power of the Models

Figures 6.2 and 6.3 show scatter plots of the values for error, bias, and variance as measured after $n \in \{100, 1{,}000\}$ samples and after all samples. Different markers indicate different total numbers of samples. Note that in each case the same range is used for x- and y-axes, so a 1:1 correspondence would from a diagonal from bottom-left to top-right of the plot. In each case we show the results of a linear regression, with 95% confidence intervals. Thus, the values for each classifier–data set pair as estimated after a few, then all, samples constitutes a single point marked on the plot. For each combination, the vertical distance between the actual point and the mean regression line shows the difference between the value as measured from all samples available, and the value predicted on the basis of just n samples .

From Fig. 6.2, we make the following observations:

- The models built from only 100 samples do not fit the data well: the plots are very scattered and the coefficient of determination is low—in other words the linear regression shown would only account for 31–32% of the observed variation in values for the final variables (bias, error, variance).
- The models predict that although the error, bias, and variance will all fall from the values observed after 100 samples: the total error by 65%, the variance by 70%, but the bias only by 50%.
- The models also predict a nonnegative residual component for each—4%, 1% and 3%, respectively, which clearly is incorrect since it suggests that no classifier-data set combination would have zero error.
- From the magnitude of the effects, we can see that the bias terms account for the majority of the observed error.
- Comparing the estimates of variance after $n = 100$ with the final values, the former are much higher. This makes it apparent that the small size of the data sets is leading to considerable noise, which introduces error into the modeling process.
- If we visualize a diagonal line through the plots for variance and total error, in each case the regression line lies below this—so the models show the observed values with $n = 100$ overestimate the final values.
- For the bias plot, the markers for all sized data sets would fall fairly evenly on either side of the diagonal. Thus, the "noise" in the bias plot does not seem to be particularly a function of the data set size.
- By contrast, for the error and variance the markers for $n = n' = 1{,}534$, which fall at the lower end of the scales, would fall around, or often above the 1:1 line, whereas those for the larger data sets would predominantly fall below the line.

This last observation is worthy of further consideration. It shows that the linear regression is a compromise. For the smaller data sets ($n + n' = 1{,}534$) whatever form the variance takes as a function of n, a Taylor expansion would give similar values to those observed after ($n = 100$), whereas for the larger data sets the variance clearly

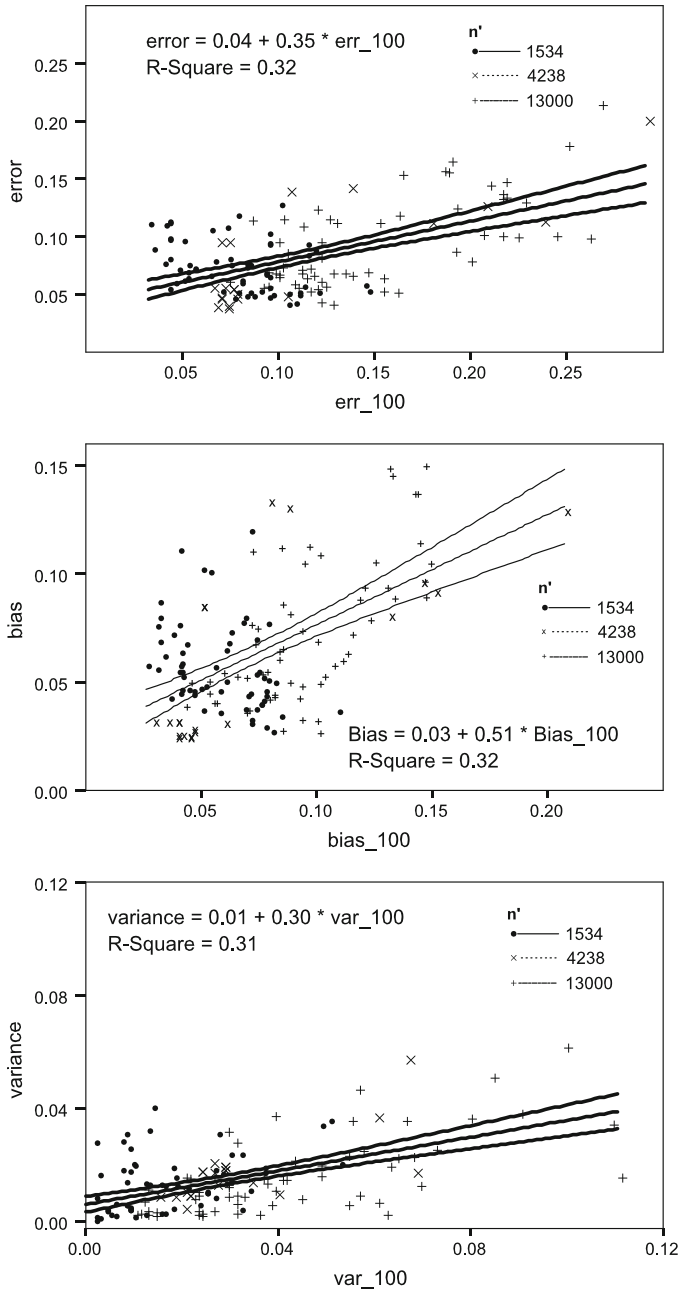


Fig. 6.2 Scatter plots of the Error (*top*), Bias (*middle*) and Variance (*bottom*) estimated after 100 samples (*x*-axis) and the same descriptors estimated using all samples (*y*-axis), together with results from linear regression (middle lines) and the 95% confidence intervals (upper and lower lines). Absolute values on individual plots vary, but in each case the *x*- and *y*-axes scale over the same range (so a 1:1 correspondence would form a diagonal of the plot)

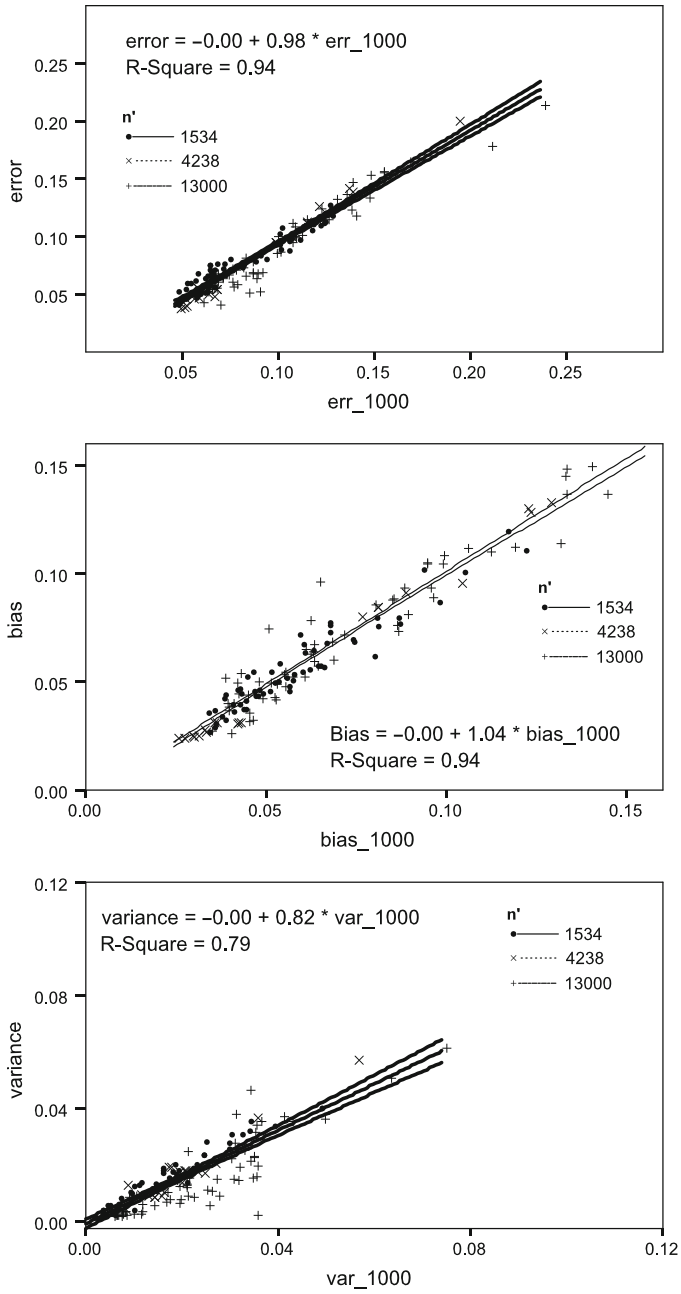


Fig. 6.3 Scatter plots of the Error (*top*), Bias (*middle*) and Variance (*bottom*) estimated after 1,000 (right) samples (*x*-axis) and the same descriptors estimated using all samples (*y*-axis), together with results from linear regression (middle lines) and the 95% confidence intervals (upper and lower lines). Absolute values on individual plots vary, but in each case the *x*- and *y*-axes scale over the same range (so a 1:1 correspondence would form a diagonal of the plot)

falls away. However, as the distribution of actual values for different data sets of the same size n' is wide, and overlaps those for different n', it is not possible for a single regression line to capture the differences.

Turning our attention to Fig. 6.3, we see a very different picture:

- The close fit of the models built from 1,000 samples to the observed data can be confirmed both visual inspection (all points fall very close to the regression line), and statistical analysis (coefficient of determination shows that for bias and total error, the model accounts for 94% of the observed differences).
- The regression models predicts that the final error will drop to 98% from its current value (direct error–error regression), and the variance component will fall to 82% of its value after 1,000 samples. However, the models predict that the bias component will rise to 104% of its current value. This is interesting since it suggests a comparison with the PAC results in (6.6). This will merit further study.
- The models now (correctly) predict zero residual components of error (i.e., $b_{Bn} = b_{Vn} = b_{En} = 0$).
- The variance accounts for a smaller proportion of the total error.
- There is no clear difference between the results for different values of n'.

This last observation is perhaps the least expected: if our arguments about the Taylor expansion of variance for $n = 100$ hold true, they should do even more so for $n = 1{,}000$ so we might see the difference in the distribution of variance markers for different sized data sets to be even more extreme. The fact that it is not can be explained by the hypothesis that the variance follows some inverse power–law in n—as suggested by (6.3). Intuitively, if elements of this variability are caused by the presence or absence in the training set of samples from particular regions of the data space, then both the probability of such elements not being present, and the averaged effect of their influence, fall nonlinearly as n increases.

However, the major point to be emphasized here is that even using a very simple model that is a linear regression from observed quantities, and does not take into account how far into the future (n') one is trying to predict, the models capture the characteristics of the observed data very closely. The results in Fig. 6.3 thus form strong evidence to confirm our original hypothesis—that the behaviors of the bias and variance, although different are predictable.

To show how the predictive quality of the models changes as they are built from increasing numbers of samples, Fig. 6.4 shows the coefficients of determination computed during the regression process as a function of n. To recapitulate, for each value of n, the bias, variance, and total error are estimated using SSCV, and regression models are built relating these to the final observed values. It is clear that the use of separate models for bias and variance provides better estimates of the predicted error. The plot also shows how rapidly the estimates (and correspondingly the predictive quality of the regression) stabilize in these two cases. What is apparent is that the method will hold well after only a few hundred data samples have been presented. Although the variance does not correlate highly (over 0.9) until closer

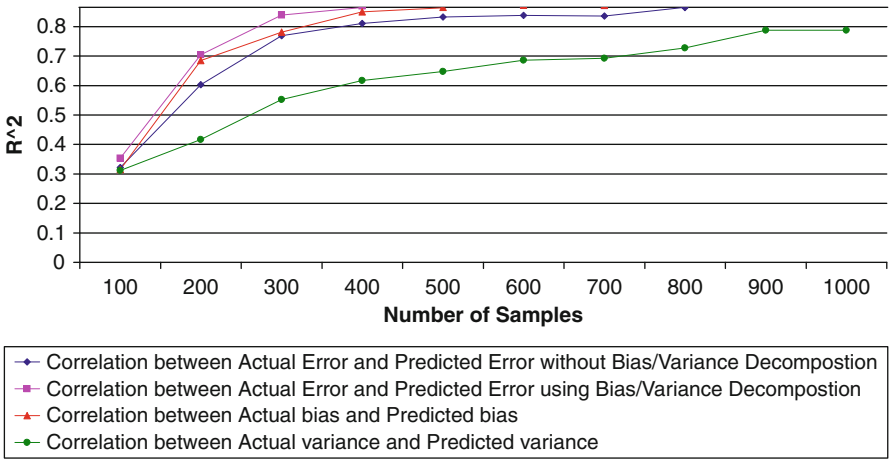


Fig. 6.4 Coefficient of Determination (R^2) between predicted and observed final error, and n, as a function of the number of samples used to build the statistical models

to 1,000 samples have been presented, the total error estimated via decomposition is well correlated because of the relatively greater size, and stability of the bias component.

6.5 Extension to Ensemble Classifiers

The concept of decomposing error into different terms has also been used to help explain the behavior of ensembles of algorithms. When the algorithms concerned are performing regression tasks, decomposing the error of an ensemble into terms representing the mean bias and variance of the individual algorithms, and the covariance between them is fairly straightforward. A good recent survey of both the bias–variance–covariance and ambiguity decompositions may be found in the first few pages of [9]. However, just as defining bias and variance for 0/1 loss functions was nontrivial, and there were several versions before Kohavi and Wolpert [21] created their formulation in which variance is always nonnegative, the extension to handle covariance in a natural way is also problematic. To the best of our knowledge there has not been a successful model decomposing 0/1 loss functions for ensembles of classifiers, so it is not immediately possible to simply extend the approach we took for single classifiers. However, in this section we present some initial findings from an approach in which we treat the entire ensemble as a single classifier. Revisiting the definitions of bias in Sect. 6.2.3, we next develop predictors for upper limits on its attainable accuracy based on simple observations of the behavior of the individual classifiers in the ensemble.

6.5.1 Estimating Lower Bounds on the Bias for Finite Data Sets

The analysis in Sect. 6.2.3 used a very general model predicated on the fact that the data items x could be drawn from a large, potentially infinite universe of samples, corresponding to unlimited future use of the classifiers. Here we are concerned with the more limited case where our future estimates are still drawn from a finite set of size $n+n'$. In particular, we consider whether we can predict the values of those estimates, before completing the training process. In order to achieve this, we can reformulate the models above slightly as follows.

To start with, let us assume that we have a finite set X of sample data points. For consistency with above note that $|X| = n+n'$. Because we are treating the ensemble as a single high-level entity, we need not worry about the effects of Boosting or Bagging approaches to creating ensembles by repeatedly sampling from training sets. Therefore, we assume that at our higher level training sets of size n are created by sampling from X uniformly without replacement. Let D denote the set of training sets created in this way, and d be any member of D, then we note that under these conditions $P(d|X,n) = \frac{1}{|D|} = \frac{(n)!(n')!}{(n+n')!}$.

Now let A^+, A^-, B partition X such that $A^+ \cup A^- \cup B = X$, and $A^+ \cap A^- = A^+ \cap B = A^- \cap B = \emptyset$, where:

- A^+ is the (possibly empty) subset of data items where for all training sets a classifier trained on that set correctly predicts the class of item x.

$$\forall x \in A^+, d, d' \in D, y \in Y \quad Y_H(y|x,d) = Y_H(y|x,d') = Y_F(y|x).$$

- A^- is the (possibly empty) subset of data items where for all training sets a classifier trained from that set incorrectly predicts the class of item x.

$$\forall x \in A^-, d, d' \in D, y \in Y \quad Y_H(y|x,d) = Y_H(y|x,d') \neq Y_F(y|x).$$

- B is the (possibly empty) set of data items where $Y_H(y|x,d)$, the hypothesis describing the predicted class of item x depends on the choice of training sets d.

$$\forall x \in B \quad \exists d, d' \in D \bullet Y_H(y|x,d) \neq Y_H(y|x,d').$$

So now lets look at what this means in terms of our estimates of the bias of the classifier. This will of course depend on the methods used for the estimates. Following well-established previous research, we will assume that each item in the data set is predicted exactly k times. This is true with $k = 1$ for N-fold cross validation, and for $k > 1$ for the Webb and Conilione approach, in general although interestingly not for the Kohavi approach [21]. This means that when we sum over the data items x in the counterpart of (6.7) each term occurs with equal probability.

Note that bias_x as stated above is composed of terms which themselves depend on the choice of training sets, and that we are assuming a fixed set of data points

and a fixed size training sets. We therefore refine the definition of bias to take these into account, and average over all possible training sets.

$$\text{bias}^2 = \frac{1}{2}\sum_{x\in X} P(x)\sum_{d\in D} P(d|X,n)\sum_{y\in Y}[P(F(x)=y) - P(H_{Cd}(x)=y)]^2. \tag{6.8}$$

If we assume we are sampling iid then $P(x) = 1/|X|$ and $P(d|X,n) = 1/|D|$. We now turn our attention to the case where each data item $x \in X$ is unambiguously associated with one of two possible class labels $y \in Y$, and we will further constrain our ensemble to output crisp decisions so that $P(H_{Cd}(x) = y) \in \{0,1\}$. Partitioning the data set X as above, we note that we make use of the following conditions when performing the summation. First, the set A^+ does not contribute to the bias since the predicted class for this subset of items is always correct. Second, $\forall x \in X, C, d \in D, \exists y_1, y_2 \in Y, y_1 \neq y_2 : F(x) = y_1 \wedge H_{CD}(x) = y_2$. This means that within the partition A^- for each combination of x and d, there are exactly two values of y which both contribute $+1$ to the summation. This yields:

$$\begin{aligned}\text{bias}^2 &= \frac{1}{2}\sum_{x\in A^-} P(x)\sum_{d\in D} p(d|X,n)\cdot 2 \\ &\quad + \frac{1}{2}\sum_{x\in B} P(x)\sum_{d\in D} P(d|X,n)\sum_{y\in Y}[P(F(x)=y) - P(H_{Cd}(x)=y)]^2, \qquad (6.9)\\ &= \frac{|A^-|}{|X|} + \frac{1}{2}\frac{|B|}{|X|}\cdot\frac{n!(|X|-n)!}{|X|!}\sum_{x,d}\sum_{y\in Y}[P(F(x)=y) - P(H_{Cd}(x)=y)]^2. \qquad (6.10)\end{aligned}$$

The last term will take a value between 0 and $|B|/|X|$ since for each value of y the difference will be 0 for some training sets and 1 for others which the gives bounds:

$$|A^-|/|X| < \text{bias}^2 < (|A^-| + |B|)/|X|. \tag{6.11}$$

This reformulation makes it explicit that considering the proportion of samples which the ensemble always misclassifies will yield a strict underestimate of the bias provided that there exist any items for which the prediction made is dependent on the training set. Furthermore, since according to (6.7) the variance term is always nonnegative, we can say that the quantity $|A^-|/|X|$ constitutes a strict lower bound on the error rate of a classifier—or an ensemble treated as a single entity.

6.5.2 *Experimental Approach*

Previous sections illustrated the successful use of regression models built from a variety of data set–classifier combinations to predict the error rates that could

be attained after future training. However, decomposing the error into different components is not straightforward for ensembles of classifiers [9]. Moreover, this would require running N-fold cross validation a number of times to get accurate estimates of bias and variance components for each combination of data set, algorithm, and n. This becomes computationally expensive when extended to a heterogeneous ensemble, particularly if the ensemble is itself trainable.

For this section, we use a slightly different approach. Previously we pooled the results from many experiments to build regression models relating observations of bias and variance after different values of n training data to the same variables of $n+n'$ items. Here we treat each data set independently, and build regression models to characterize the ensemble's learning curve as a function of n. As noted above, there is theoretical [36] as well as empirical [10, 13, 26] evidence that these learning curves have a power–law dependency on the number of training samples, i.e., they are of the form

$$\text{error}_{\text{ensemble}} = a \cdot n^b + c, \tag{6.12}$$

where a is the learning rate, b the decay rate, and c the Bayes error (the minimum achievable error or, in the error-decomposition framework, the "noise").

In our experiments, the bound on the ensemble's error derived in Sect. 6.5.1, $|A^-|/|X|$, was used as an estimate of the minimum achievable error. When faced with a new data set–ensemble combination, we make observations of $|A^-|$ and the ensemble error at regular intervals, and then feed these into the power–law regression model in order to fine-tune the parameters of the model so that it fits the new data and predicts the future development of the ensemble error, as will be detailed in Sect. 6.5.4. Before elaborating on these results, in Sect. 6.5.3 we analyze the stability of the estimation of the lower bound on the error by using $|A^-|/|X|$.

6.5.3 *Analysis of the Stability of Estimators of Lower Bounds on Error*

For the experiments performed here, 22 Machine Vision data sets from the DynaVis project were used (2 different feature spaces—17 and 74 features—for each of 5 CD-Print, 5 Artificial, and the Egg image sets). The CART [7] and C4.5 [30] decision trees, the Naive Bayes [13], Nearest Neighbor [11], and eVQ [24] classifiers were used as base classifiers, the decisions of which were combined using the Discounted Dempster–Shafer ensemble training method [32]. For each data set, each classifier, and each value of $n \in \{100, 120, \ldots, 1{,}000\}$ samples, N-fold cross-validation was repeated l times to make l predictions of the class of each item in the training set. From this data, we calculated the values of $|A^-|/|X|$ as a function of n for each data set (i.e., 22 values for each value of n). For clarity, we denote the values $|A^-|/|X|$ hereafter as Or_n.

In order to examine the stability of the predicted bounds as n increased, we plotted Or_n against Or_{final} and used linear regression as before to fit a model of

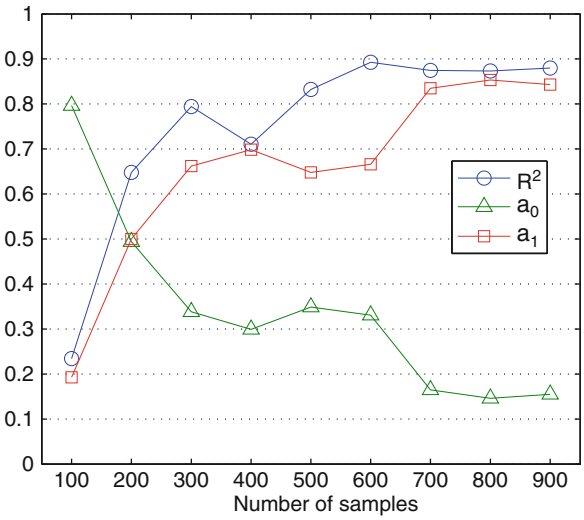


Fig. 6.5 The linear regression components for the correlation of Or_n vs. $\mathrm{Or}_{\mathrm{final}}$ together with the coefficient of determination R^2 as a function of n

the form $\mathrm{Or}_{\mathrm{final}} = a_1 \cdot \mathrm{Or}_n + a_0$, and to estimate the quality of the model via R^2. Figure 6.5 shows the progression of the coefficients a_0 and a_1 and the corresponding values of R^2 as a function of n.

As can be seen in Fig. 6.5, the models generated as n increases do produce predictions which correlate well to the observed values after further training. However, as can be seen by the progression of the coefficients, the nature of the regression models changes. For low values of n the models predict a high constant value for $\mathrm{Or}_{\mathrm{final}}$ with a low component related to the observed value of Or_n—essentially the system has not seen enough "difficult" samples. Since the major component of the predicted value of $\mathrm{Or}_{\mathrm{final}}$ is fixed for $n = 100$, the correlation is fairly low. As n increases and a more representative sample of the data is seen, the situation changes. Thus for training set sizes $n \geq 700$, the predicted value is dominated by the observed value ($a_1 \approx 0.85$) with only a low constant component ($a_0 \approx 0.15$). For these training set sizes, R^2 increases to approximately 0.9.

6.5.4 *Empirical Results for Predicting Lower Bounds and Total Errors*

The values Or_n for different n can be used for predicting not just a lower bound on, but also an estimate of the error of a trained ensemble. The following procedure can be used:

1. Or_n is measured for different n and a constant regression is performed for these values, i.e., we obtain the constant OR which minimizes the Mean Square Error with the values of Or_n across different values of n. This value forms our estimate of the lower bound on the achievable error.

2. The errors the ensemble makes are also recorded for different n.
3. A power–law regression is performed for the ensemble errors, asymptotically approaching the estimated constant OR as calculated in step 1:

$$\text{error}_{\text{ensemble}} = a \cdot n^b + \text{OR}, \tag{6.13}$$

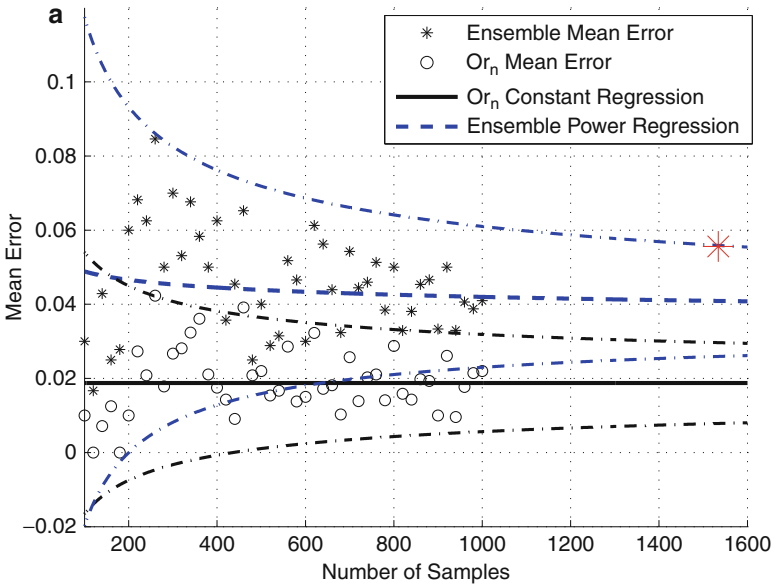
   where a and b are the regression parameters which are optimized in the regression procedure.
4. An analogous procedure is used to model the standard deviation of the observed values.
5. From the power–law regression model, we can estimate the error of the ensemble after $n + n'$ samples are presented and also some estimates of how the variation changes.

The results of this procedure are illustrated in Fig. 6.6. The five base classifiers listed above are combined using the Discounted Dempster–Shafer combination ensemble [32]. Or_n was measured for $n = \{100, 120, \ldots, 1{,}000\}$ samples. A constant regression was performed to model Or_n with a constant value and the obtained value is then used as an asymptote when modeling the ensemble errors. The errors the ensemble makes are again recorded for $n = \{100, 120, \ldots, 1{,}000\}$ samples and a (robust) regression model is built according to (6.12). The results of this procedure are illustrated in Fig. 6.6a for CD-Operator 4 and in Fig. 6.6b for Artificial 04. The values Or_n and the errors of the ensemble are shown for different n, as well as the regression models that are built for them, together with the estimated standard errors. Also the final error after evaluating the performance of the ensemble when it is trained on the entire data set is indicated, to show how accurately the errors are predicted for the ensemble when it would be trained using a larger number of training samples ($n + n'$).

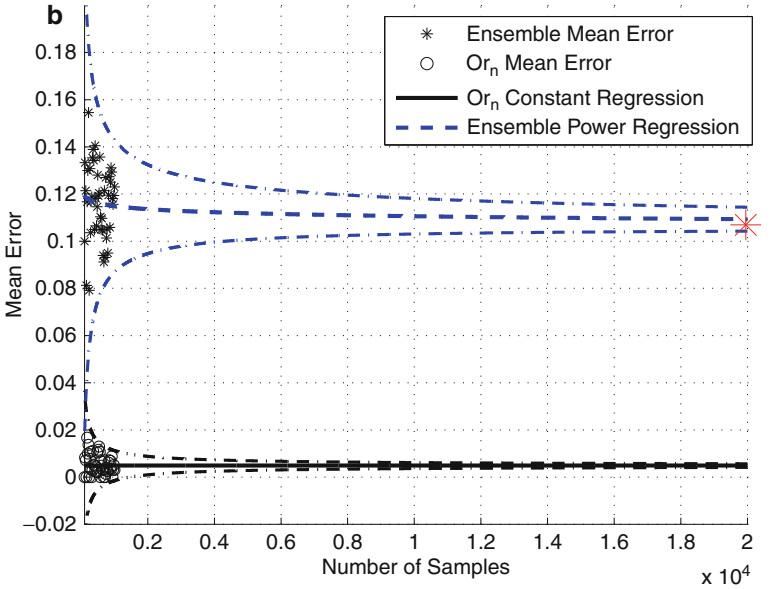
First, in both cases the results show that the model of OR does as expected form a lower bound on the error. As can be seen from Fig. 6.6b, the use of the secondary robust regression method to predict the mean and standard deviation of the observed ensemble error (top set of curves) for the artificial data set, extrapolates well and the final observed error (large asterisk at $n + n' \approx 20{,}000$) falls inside these values. For the much smaller CD print data set the figure is less clear, and the estimated standard errors on the predicted asymptote Or_n (bottom set of curves) overlap those of the robust regression prediction. Nevertheless, again the observed final ensemble error lies within one standard deviation of the value predicted by the robust regression procedure.

6.6 Application to Nonstationary Environments

In this chapter, we have outlined two ways in which a decomposition of the observed error into bias and variance terms can lead to useful predictions of future behavior. In the last section, we also introduced a means of rapidly estimating a



Results for CD-Operator 4



Results for Artificial 04

Fig. 6.6 Prediction of the errors of a (trainable) ensemble for $n + n'$ samples using a regression model which is built using n training samples. The mean values of Or_n and the mean errors of the ensemble are shown, together with their regression models (including standard errors). The error when training on the entire data set is depicted by the large $*$

lower bound on the bias when an ensemble of classifiers is used. Whereas the discussion has been partly couched in terms of making early predictions of accuracy in static environments, we now turn our attention to dynamic environments. We will characterize such environments by observing that the target function is now a function of time, i.e., $F : X, T \rightarrow Y$ and $\exists x \in X, t_1, t_2, t_1 < t_2 \bullet F(x, t_1) \neq F(x, t_2)$.

In this case the approaches outlined above can also act as a valuable "early warning system", because since for example $F(x, t_2)$ is not available to be sample at t_1 and vice versa, dynamic environments explicitly violate one of the assumptions of the analysis, namely that the training data sets are drawn iid from the underlying sample space. Thus the use of on-the-fly estimates of the *current* values of error, bias and variance can be used to detect changes in dynamic environments—for example:

- Both the "learning curve approaches" and the linear regressions models predict that the variance component will decrease with the number of samples. Any observed increase can be taken as a sign of a dynamic environment.
- Although the relative size of the bias and variance components will vary between data set–classifier combinations, our models predict that as the size of the training set increases then the ratio of bias to variance term should increase. Any departure from this should be treated as a warning sign.
- If the bias and variance components of error are periodically re-estimated from the last few samples, then the ratio of bias to variance term should remain constant. Any departure from this should be treated as a warning sign.

These indicators can be used as "early warning" signs in a number of ways. They could be used to trigger the classification algorithm—for example, to rebuild a decision tree. Alternatively they could be used to recognize that the underlying algorithm itself need changing to one which can explicitly account for dynamic situations, such as those outlined in other chapters of this book.

6.7 Conclusion

In this chapter, we have investigated techniques for making early predictions of the error rate achievable after further interactions. We have provided several example scenarios where the ability to do this would be of great value in practical data mining applications. Our approach is based on our observations that although the different components of the error progress in different ways as the number of training samples is increased, the behavior displayed by each component appeared to be qualitatively similar across different combinations of data set and classification algorithm. To investigate this finding, we have created a large set of results for many different combinations of data set, algorithm, and training set size (n) and applied statistical techniques to examine the relationship between the values observed after partial training (with n samples) and those after full training.

Perhaps surprisingly, the experimental results showed that in fact a simple linear model provided a highly accurately predictor for the subsequent behavior of different components. The results confirmed our hypothesis that these could be combined to produce highly accurate predictions of the total observed error. As there is no bias–variance(–covariance) decomposition available for 0/1 loss functions for ensembles of classifiers, it is not straightforward to apply the methodology used to accurately predict the performance of classifiers after further training to ensembles of classifiers. We have shown how a reformulation of the bias component can provide an estimate of the lower bound on the achievable error which may be more easily computed. This is especially important when the cost of training is high—for example, with trainable ensembles of classifiers. This bound is used as an asymptote in a power–law regression model to accurately predict the progression of the ensemble's error, independently for each data set.

For future work, we will focus in two directions. First, we will combine previous theoretical findings and the successful results from the two different approaches here. Taken together they suggest that for even more accurate predictions, it is worth combining the linear model for bias with an inverse power law model for variance using both the current estimates and the period over which to predict (n') as factors. This can be expected to prove particularly useful for classifiers where variance forms a major part of the observed error. Second, the work presented in this chapter used Kohavi and Wolpert's definition of bias and variance, and we will investigate whether using other definitions of bias and variance further improve the predicted accuracy.

Acknowledgements This work was supported by the European Commission (project Contract No. STRP016429, acronym DynaVis). This publication reflects only the authors' views.

References

1. P. L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Machine Learning*, **48**(1), 85–113 (2002)
2. S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: P&S*, **9**, 323–375 (2005)
3. O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. *Advanced Lectures on Machine Learning*, pp. 169–207 (2004)
4. L. Breiman. Bagging predictors. *Machine Learning*, **24**(2), 123–140 (1996)
5. L. Breiman. Bias, variance, and arcing classifiers. Technical Report 460, Statistics Department, University of California, Berkeley, CA, 1996
6. L. Breiman. Random forests. *Machine Learning*, **45**(1), 5–32 (2001)
7. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, California. (1994)
8. D. Brian and G.I. Webb. On the effect of data set size on bias and variance in classification learning. In *Proceedings of the 4th Australian Knowledge Acquisition Workshop*, pp. 117–128 (1999)
9. G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: A survey and categorisation. *Journal of Information Fusion*, **6**(1), 5–20 (2005)

10. C. Cortes, L.D. Jackel, S.A. Solla, V. Vapnik, and J.S. Denker. Learning curves: Asymptotic values and rate of convergence. In *Advances in Neural Information Processing Systems: 6*, pp. 327–334 (1994)
11. T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, **13**(1), 21–27 (1967)
12. Pedro Domingos. A unified bias–variance decomposition and its applications. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 231–238. Morgan Kaufmann, San Francisco (2000)
13. R O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2nd edition, New York (2000)
14. C. Eitzinger, W. Heidl, E. Lughofer, S. Raiser, J.E. Smith, M.A. Tahir, D. Sannen and H. Van Brussel. Assessment of the Influence of Adaptive Components in Trainable Surface Inspection Systems, *Machine Vision and Applications*, **21**(5), 613–626 (2010)
15. Yoav Freund and Robert E. Shapire. Experiments with a new boosting algorithm. In *Proceedings of 13th International Conference on Machine Learning*, pp. 148–156. (1996)
16. J. H. Friedman. On bias, variance, 0/1-loss, and the curse of dimensionality. *Data Mining and Knowledge Discovery*, **1**(1), 55–77 (2000)
17. S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, textbf4, 1–48 (1995)
18. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Springer Verlag, New York, Heidelberg, London (2001)
19. G. James. Variance and bias for general loss functions. *Machine Learning*, **51**(2), 115–135 (2003)
20. R. Kohavi. The power of decision tables. In *Proceedings of the 8th European Conference on Machine Learning*, pp. 174–189. Springer Verlag, London, UK (1995)
21. R. Kohavi and D. H. Wolpert. Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the 13th International Conference on Machine Learning.*, pp. 275–283 (1996)
22. B. E. Kong and T. G. Dietterich. Error-correcting output coding corrects bias and variance. In *Proceedings of the 12th International Conference on Machine Learning*, pp. 313–321, San Francisco, Morgan Kaufmann (1995)
23. R. Leite and P. Brazdil. Improving Progressive Sampling via Meta-learning on Learning Curves. In *Proceedings of the European Conference on machine Learning (ECML)* pp. 250–261 (2004)
24. E. Lughofer. Extensions of vector quantization for incremental clustering. *Pattern Recognition*, **41**(3), 995–1011 (2008)
25. E. Lughofer, J.E. Smith, M.A. Tahir, P. Caleb-Solly, C. Eitzinger, D. Sannen and M. Nuttin. Human–Machine Interaction Issues in Quality Control Based on On-Line Image Classification, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, **39**(5), 960–971 (2009)
26. S. Mukherjee, P. Tamayo, S. Rogers, R.M. Rifkin, A. Engle, C. Campbell, T.R. Golub, and J.P. Mesirov. Estimating dataset size requirements for classifying DNA microarray data. *Journal of Computational Biology*, **10**(2), 119–142 (2003)
27. Bartlett P.L. and S Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, **3**, 463–482 (2002)
28. J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*. MIT Press, Mass (1998)
29. F.J. Provost, D. Jensen and T. Oates. Efficient Progressive Sampling. In *Proceedings of Knowledge Discovery in Databases (KDD)*, pp. 23–32 (1999)
30. J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco (1993)
31. J. J. Rodriguez, C. J. Alonso, and O. J. Prieto. Bias and variance of rotation-based ensembles. In *Computational Intelligence and Bioinspired Systems*, number 3512 in Lecture Notes in Computer Science, pp. 779–786. Springer, Berlin, Heidelberg (2005)

32. D. Sannen, H. Van Brussel, and M. Nuttin. Classifier fusion using discounted Dempster–Shafer combination. In *Proceedings of the 5th International Conference on Machine Learning and Data Mining, Poster Proceedings*, pp. 216–230 (2007)
33. J. E. Smith and M. A. Tahir. Stop wasting time: On predicting the success or failure of learning for industrial applications. In *Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL'08)*, number 4881 in Lecture Notes in Computer Science, pp. 673–683. Springer Verlag, Berlin, Heidelberg (2007)
34. M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, **36**, 111–147 (1974)
35. V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York (2000)
36. V. Vapnik and O. Chapelle. Bounds on error expectation for support vector machines. *Neural Computation*, **12**(9), 2013–2036 (2000)
37. G. I. Webb. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, **40**(2), 159–196 (2000)
38. Geoffrey I. Webb and Paul Conilione. Estimating bias and variance from data. Technical report, Monash University, http://www.csse.monash.edu.au/webb/Files/WebbConilione03.pdf (2004)
39. I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition (2005)