### 1.9.4 *Bias and Variance Decomposition*

It is well known that the error can be decomposed into three additive components [Kohavi and Wolpert (1996)]: the intrinsic error, the bias error and the variance error.

The intrinsic error represents the error generated due to noise. This quantity is the lower bound of any inducer, i.e. it is the expected error of the Bayes optimal classifier (also known as irreducible error). The bias error of an inducer is the persistent or systematic error that the inducer is expected to make. Variance is a concept closely related to bias. The variance captures random variation in the algorithm from one training set to another, namely it measures the sensitivity of the algorithm to the actual training set, or error due to the training set's finite size. The following equations are a possible mathematical definition for the various components in case of a zero-one loss.

$$t(I,S,c_j,x) = \begin{cases} 1 & \hat{P}_{I(S)}(y=c_j\,|x) > \hat{P}_{I(S)}(y=c^*\,|x)\,\forall c^* \in dom(y), \neq c_j \\ 0 & Otherwise \end{cases}$$

$$bias^2(P(y\,|x), \hat{P}_I(y\,|x)) =$$

$$\frac{1}{2}\sum_{c_j \in dom(y)} \left[ P(y=c_j\,|x) - \sum_{\forall S,|S|=m} P(S\,|D)\cdot t(I,S,c_j,x) \right]^2$$

$$var(\hat{P}_I(y\,|x)) = \frac{1}{2}\left\{ 1 - \sum_{c_j \in dom(y)} \left[ \sum_{\forall S,|S|=m} P(S\,|D)\cdot t(I,S,c_j,x) \right]^2 \right\}$$

$$var(P(y\,|x)) = \frac{1}{2}\left\{ 1 - \sum_{c_j \in dom(y)} [P(y=c_j\,|x)]^2 \right\}$$

Note that the probability to misclassify the instance $x$ using inducer $I$ and a training set of size $m$ is:

$$\begin{aligned} \varepsilon(x) &= bias^2(P(y\,|x), \hat{P}_I(y\,|x)) + \mathrm{var}(\hat{P}_I(y\,|x)) + \mathrm{var}(P(y\,|x)) \\ &= 1 - \sum_{c_j \in dom(y)} P(y=c_j\,|x) \cdot \sum_{\forall S,|S|=m} P(S\,|D)\cdot t(I,S,c_j,x) \end{aligned}$$

It is important to note that in case of zero-one loss there are other definitions for the bias-variance components. These definitions are not necessarily consistent. In fact there is a considerable debate in the literature about what should be the most appropriate definition. For a complete list of these definitions please refer to [Hansen (2000)].

Nevertheless in context of regression a single definition of bias and variance has been adopted by the entire community. In this case it is useful to define the bias-variance components by referring to the quadratic loss, as follows:

$$E((f(x) - \hat{f}_R(x)^2) =$$
$$\text{var}(f(x)) + \text{var}(\hat{f}_R(x)) + bias^2(f(x), \hat{f}_R(x))$$

where $\hat{f}_R(x)$ represents the prediction of the regression model and $f(x)$ represents the actual value. The intrinsic variance and bias components are respectively defined as:

$$\text{var}(f(x)) = E((f(x) - E(f(x)))^2)$$
$$\text{var}(\hat{f}_R(x)) = E((\hat{f}_R(x) - E(\hat{f}_R(x)))^2)$$
$$bias^2(f(x), \hat{f}_R(x)) = E((E(\hat{f}_R(x)) - E(f(x)))^2$$

Simpler models tend to have a higher bias error and smaller variance error than complicated models. [Bauer and Kohavi (1999)] have provided an experimental result supporting the last argument for Naïve Bayes, while [Dietterich and Kong (1995)] have examined the bias-variance issue in decision trees. Figure 1.5 illustrates this argument. The figure shows that there is a trade-off between variance and bias. When the classifier is simple it has a large bias and small variance. As the classifier become more complicated, it has larger variance but smaller bias. The minimum generalization error is obtained somewhere in between, where both bias and variance are small.

### 1.9.5 *Computational Complexity*

Another useful criterion for comparing inducers and classifiers is their computational complexities. Strictly speaking computational complexity is the amount of CPU consumed by each inducer. It is convenient to differentiate between three metrics of computational complexity:

- Computational Complexity for generating a new classifier: This is the most important metric, especially when there is a need to scale the

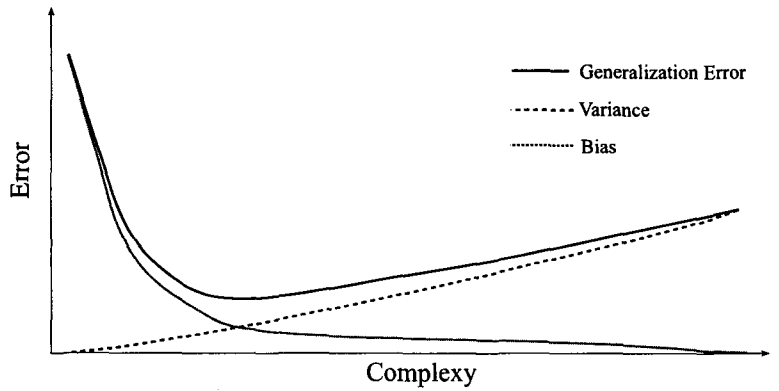


Fig. 1.5 Bias vs. Variance in the Deterministic Case: Hansen, 2000.

data mining algorithm to massive data sets. Because most of the algorithms have computational complexity, which is worse than linear in the numbers of tuples, mining massive data sets might be "prohibitively expensive".

- Computational Complexity for updating a classifier: Giving a new data — what is the computational complexity required for updating the current classifier such that the new classifier reflects the new data?
- Computational Complexity for classifying a new instance: Generally this type is neglected because it is relatively small. However, in certain methods (like k-Nearest Neighborhood) or in certain real time applications (like anti-missiles applications), this type can be critical.

### 1.9.6 *Comprehensibility*

Comprehensibility criterion (also known as Interpretability) refers to how well humans grasp the classifier induced. While the generalization error measures how the classifier fits the data, comprehensibility measures the "Mental fit" of that classifier.

Many techniques, like neural networks or SVM (Support Vector Machines), are designed solely to achieve accuracy. However, as their classifiers are represented using large assemblages of real valued parameters, they are also difficult to understand and are referred to as black-box models.

It is often important for the researcher to be able to inspect an induced classifier. For domains such as medical diagnosis, the users must understand how the system makes its decisions in order to be confident of the outcome.

Data mining can also play an important role in the process of scientific discovery. A system may discover salient features in the input data whose importance was not previously recognized. If the representations formed by the inducer are comprehensible, then these discoveries can be made accessible to human review [Hunter and Klein (1993)].

Comprehensibility can vary between different classifiers created by the same inducer. For instance, in the case of decision trees, the size (number of nodes) of the induced trees is also important. Smaller trees are preferred because they are easier to interpret. However, this is only a rule of thumb, in some pathologic cases a large and unbalanced tree can still be easily interpreted [Buja and Lee (2001)].

As the reader can see the accuracy and complexity factors can be quantitatively estimated, while the comprehensibility is more subjective.

Another distinction is that the complexity and comprehensibility depend mainly only on the induction method and much less on the specific domain considered. On the other hand, the dependence of error metric on specific domain can not be neglected.

## 1.10 "No Free Lunch" Theorem

Empirical comparison of the performance of different approaches and their variants in a wide range of application domains has shown that each performs best in some, but not all, domains. This has been termed the selective superiority problem [Brodley (1995)].

It is well known that no induction algorithm can be the best in all possible domains; each algorithm contains an explicit or implicit bias [Mitchell (1980)] that leads it to prefer certain generalizations over others, and it will be successful only insofar as this bias matches the characteristics of the application domain [Brazdil *et al.* (1994)]. Furthermore, other results have demonstrated the existence and correctness of the "conservation law" [Schaffer (1994)] or "no free lunch theorem" [Wolpert (1996)]: if one inducer is better than another in some domains, then there are necessarily other domains in which this relationship is reversed.

The "no free lunch theorem" implies that for a given problem a certain approach can yield more information from the same data than other approaches.

A distinction should be made between all the mathematically possible domains, which are simply a product of the representation languages used,

and the domains that occur in the real world, and are therefore the ones of primary interest [Rao *et al.* (1995)]. Without doubt there are many domains in the former set that are not in the latter, and average accuracy in the realworld domains can be increased at the expense of accuracy in the domains that never occur in practice. Indeed, achieving this is the goal of inductive learning research. It is still true that some algorithms will match certain classes of naturallyoccurring domains better than other algorithms, and so achieve higher accuracy than these algorithms, and that this may be reversed in other realworld domains; but this does not preclude an improved algorithm from being as accurate as the best in each of the domain classes.

Indeed, in many application domains the generalization error of even the best methods is far above 0%, and the question of whether it can be improved, and if so how, is an open and important one. One part of answering this question is determining the minimum error achievable by any classifier in the application domain (known as the optimal Bayes error). If existing classifiers do not reach this level, new approaches are needed. Although this problem has received considerable attention (see for instance [Tumer and Ghosh (1996)]), no generally reliable method has so far been demonstrated.

The "no free lunch" concept presents a dilemma to the analyst approaching a new task: which inducer should be used?

If the analyst is looking for accuracy only, one solution is to try each one in turn, and by estimating the generalization error, to choose the one that appears to perform best [Schaffer (1994)]. Another approach, known as *multistrategy learning* [Michalski and Tecuci (1994)], attempts to combine two or more different paradigms in a single algorithm. Most research in this area has been concerned with combining empirical approaches with analytical methods (see for instance [Towell and Shavlik (1994)]. Ideally, a multistrategy learning algorithm would always perform as well as the best of its "parents" obviating the need to try each one and simplifying the knowledge acquisition task. Even more ambitiously, there is hope that this combination of paradigms might produce synergistic effects (for instance by allowing different types of frontiers between classes in different regions of the example space), leading to levels of accuracy that neither atomic approach by itself would be able to achieve.

Unfortunately, this approach has often been only moderately successful. Although it is true that in some industrial applications (like in the case of demand planning) this strategy proved to boost the error performance, in many other cases the resulting algorithms are prone to be cumbersome,

and often achieve an error that lie between those of their parents, instead of matching the lowest.

The dilemma of what method to choose becomes even greater, if other factors such as comprehensibility are taken into consideration. For instance for a specific domain, neural network may outperform decision trees in accuracy. However, from the comprehensibility aspect, decision trees are considered better. In other words, in this case even if the researcher knows that neural network is more accurate, he still has a dilemma what method to use.

## 1.11 Scalability to Large Datasets

Obviously induction is one of the central problems in many disciplines like: machine learning, pattern recognition, and statistics.

However the feature that distinguishes data mining from traditional methods is its scalability to very large sets of varied types of input data. In this book the notion, "scalability" refers to datasets that fulfill at least one of the following properties: high number of records, high dimensionality, high number of classes or heterogeneousness.

"Classical" induction algorithms have been applied with practical success in many relatively simple and small-scale problems. However, trying to discover knowledge in real life and large databases, introduce time and memory problems.

As large databases have become the norm in many fields (including astronomy, molecular biology, finance, marketing, health care, and many others), the use of data mining to discover patterns in them has become a potentially very productive enterprise. Many companies are staking a large part of their future on these "data mining" applications, and looking to the research community for solutions to the fundamental problems they encounter.

While a very large amount of available data used to be a dream of any data analyst, nowadays the synonym for "very large" has become "terabyte", a hardly imaginable volume of information. Information-intensive organizations (like telecom companies and banks) are supposed to accumulate several terabytes of raw data every one to two years.

However, the availability of an electronic data repository (in its enhanced form known as a "data warehouse") has caused a number of previously unknown problems, which, if ignored, may turn the task of

efficient data mining into mission impossible. Managing and analyzing huge data warehouses requires special and very expensive hardware and software, which often causes a company to exploit only a small part of the stored data.

According to [Fayyad *et al.* (1996)] the explicit challenges for the data mining research community is to develop methods that facilitate the use of data mining algorithms for real-world databases. One of the characteristics of a real world databases is high volume data.

Huge databases pose several challenges:

- Computing complexity: Since most induction algorithms have a computational complexity that is greater than linear in the number of attributes or tuples, the execution time needed to process such databases might become an important issue.
- Poor classification accuracy due to difficulties in finding the correct classifier. Large databases increase the size of the search space, and thus it increases the chance that the inducer will select an over fitted classifier that is not valid in general.
- Storage problems: In most machine learning algorithms, the entire training set should be read from the secondary storage (such as magnetic storage) into the computer's primary storage (main memory) before the induction process begins. This causes problems since the main memory's capability is much smaller than the capability of magnetic disks.

The difficulties in implementing classification algorithms as-is on high volume databases derives from the increase in the number of records/instances in the database and from the increase in the number of attributes/features in each instance (high dimensionality).

Approaches for dealing with a high number of records include:

- Sampling methods — statisticians are selecting records from a population by different sampling techniques.
- Aggregation — reduces the number of records either by treating a group of records as one, or by ignoring subsets of "unimportant" records.
- Massively parallel processing — exploiting parallel technology — to simultaneously solve various aspects of the problem.
- Efficient storage methods — enabling the algorithm to handle many records. For instance [Shafer *et al.* (1996)] presented the SPRINT which constructs an attribute list data structure.

- Reducing the algorithm's Search space — For instance the PUBLIC algorithm [Rastogi and Shim (2000)] integrates the growing and pruning of decision trees by using MDL cost in order to reduce the computational complexity.

## 1.12 The "Curse of Dimensionality"

High dimensionality of the input (that is, the number of attributes) increases the size of the search space in an exponential manner, and thus increases the chance that the inducer will find spurious classifiers that are not valid in general. It is well known that the required number of labeled samples for supervised classification increases as a function of dimensionality [Jimenez and Landgrebe (1998)]. [Fukunaga (1990)] showed that the required number of training samples is linearly related to the dimensionality for a linear classifier and to the square of the dimensionality for a quadratic classifier. In terms of nonparametric classifiers like decision trees, the situation is even more severe. It has been estimated that as the number of dimensions increases, the sample size needs to increase exponentially in order to have an effective estimate of multivariate densities ([Hwang *et al.* (1994)].

This phenomenon is usually called "curse of dimensionality". Bellman (1961) was the first to coin this term, while working on complicated signal processing. Techniques like decision trees inducers that are efficient in low dimensions fail to provide meaningful results when the number of dimensions increases beyond a "modest" size. Furthermore, smaller classifiers, involving fewer features (probably less than 10), are much more understandable by humans. Smaller classifiers are also more appropriate for user-driven data mining techniques such as visualization.

Most of the methods for dealing with high dimensionality focus on Feature Selection techniques, i.e. selecting a single subset of features upon which the inducer (induction algorithm) will run, while ignoring the rest. The selection of the subset can be done manually by using prior knowledge to identify irrelevant variables or by using proper algorithms.

In the last decade, Feature Selection has enjoyed increased interest by many researchers. Consequently many Feature Selection algorithms have been proposed, some of which have reported remarkable accuracy improvement. As it is too wide to survey here all methods, the reader is referred to the following sources: [Langley (1994)], [Liu and Motoda (1998)] for further

reading.

Despite its popularity, the usage of feature selection methodologies for overcoming the obstacles of high dimensionality has several drawbacks:

- The assumption that a large set of input features can be reduced to a small subset of relevant features is not always true; in some cases the target feature is actually affected by most of the input features, and removing features will cause a significant loss of important information.
- The outcome (i.e. the subset) of many algorithms for Feature Selection (for example almost any of the algorithms that are based upon the wrapper methodology) is strongly dependent on the training set size. That is, if the training set is small, then the size of the reduced subset will be small also. Consequently, relevant features might be lost. Accordingly, the induced classifiers might achieve lower accuracy compared to classifiers that have access to all relevant features.
- In some cases, even after eliminating a set of irrelevant features, the researcher is left with relatively large numbers of relevant features.
- The backward elimination strategy, used by some methods, is extremely inefficient for working with large-scale databases, where the number of original features is more than 100.

A number of linear dimension reducers have been developed over the years. The linear methods of dimensionality reduction include projection pursuit [Friedman and Tukey (1973)], factor analysis [Kim and Mueller (1978)], and principal components analysis [Dunteman (1989)]. These methods are not aimed directly at eliminating irrelevant and redundant features, but are rather concerned with transforming the observed variables into a small number of "projections" or "dimensions". The underlying assumptions are that the variables are numeric and the dimensions can be expressed as linear combinations of the observed variables (and vice versa). Each discovered dimension is assumed to represent an unobserved factor and thus provide a new way of understanding the data (similar to the curve equation in the regression models).

The linear dimension reducers have been enhanced by constructive induction systems that use a set of existing features and a set of predefined constructive operators to derive new features [Pfahringer (1994); Ragavan and Rendell (1993)]. These methods are effective for high dimensionality applications only if the original domain size of the input feature can be in fact decreased dramatically.

One way to deal with the above mentioned disadvantages is to use a very large training set (which should increase in an exponential manner as the number of input features increases). However, the researcher rarely enjoys this privilege, and even if it does happen, the researcher will probably encounter the aforementioned difficulties derived from a high number of instances.

Practically most of the training sets are still considered "small" not due to their absolute size but rather due to the fact that they contain too few instances given the nature of the investigated problem, namely the instance space size, the space distribution and the intrinsic noise. Furthermore, even if a sufficient dataset is available, the researcher will probably encounter the aforementioned difficulties derived from high number of records.