ERROR ESTIMATION AND MODEL SELECTION

vorgelegt von
Dipl.-Inform.
Tobias Scheffer

Vom Fachbereich 13 – Informatik –
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuß:
Vorsitzender: Prof. Dr. Stefan Jähnichen
Berichter: Prof. Dr. Fritz Wysotzki
Berichter: Prof. Dr. Claus Weihs

Tag der wissenschaftlichen Ausprache: 31. Mai 1999

Berlin, Mai 1999

D 83

Acknowledgment

I wish to thank everyone who helped me with my studies and contributed to my thesis. In particular, I wish to thank *Eckehardt Blanz* who supported my attempt to obtain the Ernst von Siemens fellowship and hosted me in Princeton. Thanks to *Paul Compton* for hosting me in Sydney and for helpful reviews on earlier papers. Thanks to *Chris Darken* who I enjoyed working with in Princeton. I also enjoyed working with *Russ Greiner* from whom I learned a lot about conducting scientific research. Thanks to *Ralf Herbrich* for contributions to Chapter 5. *Achim Hoffmann's* agnostic view of machine learning lead me to deeper insights into learning theory. Achim also contributed to Chapter 4. Thanks to *Hans-Peter Jacobs* for installing my laptop, uploading games to it, and contributing to the implementation of some experiments. *Thorsten Joachims* helped me out with some equations which did not behave themselves and made key contributions to Chapter 3. Thanks to *Katharina Morik* for hosting me in Dortmund. Thanks to *Frank Neumann* for help with my computer and contributions to the implementation of some experiments. I appreciate *Visvanathan Ramesh*'s lesson on what engineering is all about. Thanks to *Alberto Segre* who helped me to edit my IJCAI paper and taught me a lesson on english punctuation. *Arun Sharma*'s striving for scientific excellence was a source of inspiration to me. Thanks to *Ursula Sondhauss* for enjoyable discussions. I enjoyed working with *Frank Stephan* on the complexity of boosting. Thanks to *Claus Weihs* for an interesting discussion on the expected error analysis and for carefully proof-reading the proofs. Thanks to my advisor *Fritz Wysotzki* for giving me full autonomy during my PhD project and for detailed comments on my thesis.

The work reported here was supported by an *Ernst von Siemens* fellowship and partially supported by grants WY 20/1-1 and WY 20/1-2 of the German Research Council (DFG) and two travel grants of the German Research Association (Stifterverband).

Abstract

Machine learning algorithms search a space of possible hypotheses and estimate the error of each hypotheses using a sample. Most often, the goal of classification tasks is to find a hypothesis with a low true (or generalization) misclassification probability (or error rate); however, only the sample (or empirical) error rate can actually be measured and minimized. The true error rate of the returned hypothesis is unknown but can, for instance, be estimated using cross validation, and very general worst-case bounds can be given. This doctoral dissertation addresses a compound of questions on error assessment and the intimately related selection of a "good" hypothesis language, or learning algorithm, for a given problem.

In the first part of this thesis, I present a new analysis of the generalization error of the hypothesis which minimizes the empirical error within a finite hypothesis language. I present a solution which characterizes the generalization error of the apparently best hypothesis in terms of the distribution of error rates of hypotheses in the hypothesis language. The distribution of error rates can, for any given problem, be estimated efficiently from the sample. Effectively, this analysis predicts how good the outcome of a learning algorithm would be without the learning algorithm actually having to be invoked. This immediately leads to an efficient algorithm for the selection of a good hypothesis language (or "model"). The analysis predicts (and thus explains) the shape of learning curves with a very high accuracy and thus contributes to a better understanding of the nature of over-fitting. I study the behavior of the model selection algorithm empirically (in particular, in comparison to cross validation) using both artificial problems and a large scale text categorization problem.

In the next step, I study in which situations performing automatic model selection is actually beneficial; in particular, I study Occam algorithms and cross validation. Model selection techniques such as tree pruning, weight decay, or cross validation, are employed by virtually all "practical" learners and are generally believed to enhance the performance of learning algorithms. However, I show that this belief is equivalent to an assumption on the distribution of problems which the learning algorithm is exposed to. I specify these distributional assumptions and quantify the benefit of Occam algorithms and cross validations in these situations. When the distributional assumptions fail, cross-validation based model selection *increases* the generalization error of the returned hypothesis on average.

When several distinct learners are assessed with respect to a particular problem (or one learner is assessed repeatedly with distinct parameter settings), an effect arises which is very similar to over-fitting that occurs during error-minimization processes. The lowest observed error rate is an optimistic estimate of the corresponding generalization error. I quantify this bias. In particular, I study the bias which is imposed by repeated invocations of a learner with distinct parameter settings when n-fold cross validation is used to estimate the error rate. I pursue an information theoretic approach which does not require the assumption that empirical error rates measured in distinct cross validation folds are independent estimates. I discuss the implications of these results on the results of empirical studies which have been carried out in the past and propose an experimental setting which leads to almost unbiased results.

Finally, I address complexity issues of model selection. In model selection based learning, the learning algorithm is restricted to a (small) model, chosen by the model selection algorithm. By contrast, in the boosting setting, the hypothesis is allowed to grow dynamically, often until the hypothesis is fitted to the data. By giving new worst-case time bounds for the AdaBoost algorithm I show that in many cases the restriction to small sets of hypotheses causes the high complexity of learning

Contents

Chapter 1
Introduction

This Chapter provides an informal introduction to supervised machine learning and an overview over this doctoral dissertation. In particular, PAC Theory, Bayesian learning, the No-Free-Lunch Theorems, and the principle idea of model selection are discussed on an intuitive level.

1.1 Machine Learning

Learning is studied in many disciplines and many definitions try to capture the intuition of what we actually consider learning. Attempts to define learning were made by, among others, Simon (1983). Simon defines learning as a modification in the behavior of a system which leads to an improvement with respect to the repeated performance of some task. This definition may appear to be overly general as it includes, for instance, the physical modification of a system. More specifically, Michalski *et al.* (1986) define learning as the construction and modification of representations of experience.

Many categories of learning processes are distinguished in psychology, statistics, and computer science. The learning settings differ in the task which the student has to accomplish and the information which is provided by the teacher. In the *concept formation* setting, the student (or *learner*) is provided with a set of positive and often additionally a set of negative examples of a concept. It then has to identify the target concept from the data. *Classification* differs slightly from concept formation. Here, the learner has to discriminate finitely many classes from each other. *Regression* is studied in statistics; here, the learner has to identify a target function (from sample points) rather than just discriminate finitely many classes. *Language acquisition* has a somewhat different meaning in learning theory than it has in psychology: In Gold's mathematical model (Gold, 1967), language learning is considered the problem of finding a grammar which identifies a language syntactically (*i.e.*, discriminates its sentences from the complementary language); the acquisition of a semantic is disregarded. In the *skill acquisition* setting, a learner is able to perceive parameters of a system and has to select a control action, such that some performance criterion is maximized. In order to acquire such a control policy, the learner can either conduct control experiments (this setting is referred to as reinforcement learning; Sutton & Barto, 1998), or it can observe a perfect operator controlling the system (referred to as behavior cloning; Sammut *et al.*, 1992).

In this thesis, I will focus on *mathematical models* which also try to capture the intuition of learning. Many of these models and the whole of this thesis is committed to classification (a detailed overview on the various other fields of machine learning is given by Mitchell, 1997). I assume that there is an unknown target function which the learner has to "guess" from examples. The domain of this target function is generally referred to as the "instances" and the elements of the finite co-domain are called "class labels". Typically, the learner is given a finite sequence of input-output pairs (a

sample). The various models differ in their success criteria: While Gold's model of *Identification in the Limit* (Gold, 1967) requires that the learner identifies the target concept exactly and with certainty but in unbounded time, Valiant's framework of *Probably Approximately Correct* learning (Valiant, 1984) requires the learner to find, with high probability, a hypothesis which incurs an error of no more than ε from a batch of examples (preferably polynomial in size) where the error is measured in terms of the misclassification probability for new instances. The notation of the misclassification probability (or "error rate") of hypotheses implies the existence of an (unknown) underlying distribution over instances. The error rate is then the chance of the hypothesis making a false prediction on an instance drawn according to this distribution. What makes this setting nontrivial is that the learner is only able to perceive the error incurred on the sample; however, the success criterion refers to the actual target function which is unknown to the learner. The way that PAC theory argues to bound the *true (or generalization) error rate* of hypotheses is the following: The chance of a hypothesis with error of at least ε giving the correct answer for a new instance is at most $1 - \varepsilon$. Hence, the chance of our hypothesis classifying a sequence of m instances correctly (although the true error is ε or more) is at most $(1-\varepsilon)^m$, provided that these examples are independent and identically distributed. Suppose that the learner searches a finite hypothesis space H and returns an arbitrary hypothesis which is consistent with a sample of size m. In the worst case, *all* $|H|$ hypotheses incur an error of at least ε with each hypothesis having a chance of $(1-\varepsilon)^m$ of being consistent with the sample. Hence, the chance that at least *one* of $|H|$ hypotheses incurs a true error of at least ε *and* is consistent with the sample is at most $|H|(1-\varepsilon)^m$. When we employ the worst possible learner, it might return the hypothesis with the highest true error which is consistent with the sample. Therefore, if our learner returns a hypothesis that is consistent with a sample of size m, we can claim that, with probability at least $1-|H|(1-\varepsilon)^m$, the true error is no more than ε. This simple result gives an intuition on how PAC theory allows to make statistical claims on the true error of a hypothesis when only the empirical error is known. This result also demonstrates that our knowledge on the generalization error of a learned hypothesis decreases with the size of the hypothesis language which the hypothesis was learned from. If we consider a hypothesis space with one single hypothesis, the empirical error is an *unbiased* estimate of the true error (the chance of the empirical error being an optimistic estimate is just as large as the chance of it being a pessimistic estimate). Unbiased means that the expectation of the empirical error rate of a hypothesis is the true error rate. However, if we consider a hypothesis space with many hypotheses, then certainly some of them will have an empirical error which is greater than their true error while others will have an empirical error which is less than their true error. If we choose a hypothesis with low empirical error we are likely to select one with an optimistically estimated error. The more distinct hypotheses there are in the space, the less we can say about the true error of a hypothesis which is consistent with the sample. However, this result must not be misinterpreted as meaning that the generalization error of the returned hypothesis increases when the hypothesis language which it was learned from grows.

In *Bayesian Learning* (*e.g.,* Berger, 1985), the learner is assumed to have some extra information, compared to a PAC learner. While the latter is required to perform well for *any* target function (from a given set of possible functions), one averages the error rate of a Bayesian learner over all possible target functions, according to a known prior probability $P(f)$ on target functions. So, Bayesian learning is easier than PAC learning in some sense because it does not "hurt too much" when the learner performs poorly for unlikely target functions, and the learner knows the prior probability of target functions. The hypothesis which minimizes the expected error (or, more general, the loss) is called the *Bayes* hypothesis. The Bayes hypothesis makes a prediction for an instance x according to the weighted majority of all possible functions $f(x)$, where the weights are the posterior probabilities $P(f|S)$ of the function f having generated the observed sample S. Using Bayes' rule, this posterior

works out to $P(S|f)\frac{P(f)}{P(S)}$. $P(S|f)$ can often be determined and $P(f)$ is assumed to be known; $P(S)$ can in some cases be derived from $P(S|f)$ and $P(f)$. In this situation, one can make an *optimal* decision – *i.e.,* one that minimizes the expected misclassification probability in classification learning. When $P(S)$ is not known or summing over all possible f is not tractable, one can still determine the MAP (or "*Maximum a posteriori*") hypothesis which maximizes the posterior probability $P(f|S)$ of having generated the sample by maximizing $P(S|f)P(f)$ ($P(S)$ is constant for a given learning problem). Unfortunately, assuming that the prior $P(f)$ is known for a given environment is not entirely realistic. *Robust Bayesian learners* (Berger, 1993) can be guaranteed to perform well for a whole class of possible priors which differ from each other to some degree.

1.2 The Need for Bias in Learning

The term "learning bias" entails various mechanisms which have an influence on which hypothesis a learning algorithm is going to come up with. Mathematically, this learning bias can be written as $P_L(h|S)$ – *i.e.,* the chance that hypothesis h is returned by learner L given the data S. One factor which influences the learning result is the hypothesis language which is therefore also referred to as "language bias". Learning algorithms which minimize the empirical error rate have to decide which of the hypotheses with least empirical error within the language to return. One possible bias is to draw at random (under uniform distribution) from the hypotheses with least empirical error. I will assume this particular bias throughout Chapter 3. However, a learner might follow a completely different bias; *e.g.,* it might prefer the hypothesis which is least with respect to the alphabetical ordering. Instead of choosing among the error minimizing hypotheses, a learner might choose a particular hypothesis which maximizes some merit criterion (this is what complexity penalization algorithms do) which leads to some particular learning bias.

It is obvious that the learning bias has a major impact on the generalization ability of the resulting hypothesis. But precisely what does the relation between learning bias and generalization ability look like? Or, asked in another way, is there a learning bias which can be proven to be superior to all other biases? PAC and VC theory (discussed more carefully in Section 2.2.2) study the generalization ability as intrinsic properties of the learning bias. The PAC results are sometimes interpreted as suggesting that, in order to achieve good generalization, one should choose certain hypothesis languages while avoiding others. However, a careful analysis often reveals that such interpretations are undue. By contrast, the No-Free-Lunch Theorems (for a more detailed discussion, see Section 2.3) clarify that the generalization ability of a particular learning bias is a property of the focused problem, rather than a property of the bias itself.

PAC theory requires learners to produce a hypothesis with low true error when only minimal domain knowledge is available (only a class of possible target functions is known). One way of quantifying the generalization performance of learners is to look at the sample size which is required to guarantee a low true error. The "sample complexity" of learning has been studied intensely, and many classes of target functions and hypothesis languages have been identified which can be learned from sample sizes polynomial in the size parameter of the function or language class. Blumer *et al.* (1987) proved the well-known result that the generalization error of a hypothesis h which is consistent with a sample of size m and which has been learned from a hypothesis language H can be bounded by ε (with confidence $1-\delta$) when the sample size m is at least $\frac{1}{\varepsilon}\log\frac{1}{\delta}$. Assume that two hypotheses, h_1 and h_2, which are consistent with a sample S have been learned from hypothesis languages H_1 and H_2, respectively, with $|H_1| < |H_2|$. Then Blumer's result shows that we can prove a better error bound for h_1 than we can for h_2. It is very tempting to misinterpret this result as meaning that H_1 is

a better language bias than H_2. This, however, is not necessarily true; in particular, it is only likely to be true when the target function is known to lie in H_1. A different way of thinking about this problem reveals that, on average, h_1 and h_2 incur an equal generalization error.

Wolpert (1992) adopted a different perspective towards the question of how good learners can be and came to some very insightful results – the No-Free-Lunch Theorems. If we assume that all learners minimize the sample error anyway, the off-sample error (error on all instances that are not in the sample) becomes an interesting issue. Imagine that, for some classification problem, only four instances x_1 though x_4 are not in the sample. If the hypothesis language is powerful enough, there are 2^4 distinct hypotheses which are consistent with the sample (these four hypotheses form the *version space;* Mitchell, 1982) but behave differently on the remaining four instances. Every learner L which returns hypotheses that are consistent with the sample needs to have a built-in preference to choose between hypotheses which are equally consistent with the sample (this preference is sometimes also referred to as *learning bias*). Let learner L_{0000} return the hypothesis which classifies all four instances as 0, L_{0001} labels x_1 through x_3 as 0 and x_4 as 1, and so on. There are also 2^4 possible target functions which behave differently on x_1 though x_4, let us label them f_{0000} through f_{1111} as well. Let us now look at the off-sample error of L_{0000}, averaged over all possible target functions, under uniform distribution of the instances. For f_{0000}, learner L_{0000} incurs an off-sample error of zero, for f_{0001} of $\frac{1}{4}$, and so on. On average, L_{0000} incurs an off-sample error of $\frac{1}{2}$. Learner L_{1000} incurs an off-sample error of 0 for f_{1000}, of $\frac{1}{4}$ on f_{0000} and, averaged over all functions, of $\frac{1}{2}$ as well. In fact, all learners L_{0000} through L_{1111} impose an average off-sample error of $\frac{1}{2}$. This observation leads to the first No-Free-Lunch Theorem: Uniformly averaged over all possible target functions, the off-sample errors of two arbitrary learners are equal. This theorem holds for arbitrary learners and basically says that it is impossible to construct a learner which is better-than-average on *all* problems. But the important point is that we averaged the error uniformly over all target functions. If, however, some functions occur more frequently than others (*i.e.*, there is a known nonuniform prior $P(f)$), it *is* possible to construct a learner that performs well for this particular distribution. If the prior is nonuniform but unknown, the third No-Free-Lunch Theorem claims that, again, no better-than-average learner can be constructed. See Section 2.3 for a formal presentation of the No-Free-Lunch Theorems

This argues that, as far as the generalization error rate is concerned, there is no such thing as an "intrinsically good learning bias". One cannot construct a learner which is both general and accurate. Instead, a low generalization error is due to an alignment between the bias of the learner and the prior probability of target concepts which occur in some domain. This indeed justifies the need for a learning bias which is adequate for the given learning problem.

Another aspect of the learning bias is the complexity of learning algorithms which use that particular bias. Consider this example. When the target function is a k-term CNF(n) (a conjunction of up to k disjunctions over n variables), then no learner which uses k-term CNF(n) as hypothesis language can be guaranteed to find a hypothesis which is consistent with a sample of size m in time polynomial in n. If, however, k-DNF(n) (disjunctions of arbitrary many conjunctions which may consist of up to k Boolean literals each) is used as hypothesis language, then a polynomial algorithm can be found that finds a k-DNF which is consistent with the sample and approximates the target k-term CNF(n) well – although k-DNF(n) is a proper superset of k-term CNF(n). This is because, for k-DNF(n), a greedy algorithm exists while, when k-term CNF(n) is selected as hypothesis language, the whole hypothesis space has to be enumerated to check for a hypothesis which is consistent with the sample. In *Inductive Logic Programming* (ILP) (*e.g.,* Lavrac & Džeroski, 1994; Muggleton, 1992), the target function is a set of Horn clauses (usually the class of Horn clauses is subject to further restrictions). In this learning problem, even calculating the class label which a hypothesis assigns to an instance is undecidable (under logic implication) or NP-complete (under θ-subsumption). Consequently, the

learning problem is, in most cases, extremely expensive. It does, however, turn out that certain sets of clauses can be learned in polynomial time (usually those for which θ-subsumption can be proven efficiently; *e.g.,* Kietz & Dzeroski, 1994; Scheffer *et al.*, 1996) while other, equally large sets cannot be learned polynomially. So from a complexity oriented point of view, there are language biases which are intrinsically superior to others.

1.3 Model Selection

Consider the following situation: In order to solve a learning problem we are free to choose a hypothesis language from a set of languages (or models) – decision trees of variable depth perhaps, or a neural network with a variable number of hidden neurons. One can think of a model as a collection of structurally identical (or similar) hypotheses. Statisticians like to think of models as parametric schemes and of hypotheses as models with instantiated parameters. If we choose too simple a model (a tree of depth one say) even the best hypothesis in that model is likely to incur both a high empirical and a high true error. On the other hand, if we choose too rich a model (*e.g.,* a neural network with a hundred hidden units) our hypothesis is likely to be poor due to over-fitting effects. This problem is often referred to as the bias-variance trade-off, based on Breiman *et al.* (1984) who distinguish a bias and a variance term in the generalization error. The bias part of the generalization error is the error rate of the best approximation of the target within a given model. By increasing the model size, the bias term decreases monotonically. The variance term quantifies the error that is imposed by improper labeling of the nodes caused by limited data which is available. Whether or not (and by how much) the variance term increases when we increase the model size depends on the problem. So what can we do in this situation? Three classes of approaches can be distinguished: Hold-out testing methods use parts of the data to assess the hypotheses learned from increasingly complex models; complexity penalization approaches minimize a demerit criterion (instead of the empirical error) which consists of the empirical error plus a complexity penalization term; Bayesian approaches exploit additional information in terms of a prior on target functions which is assumed to be known in advance.

Hold-out testing. One thing we might do is stratify the hypothesis language into increasingly complex models (*e.g.,* model H_i could consist of networks with i hidden units), and use cross validation (*e.g.,* Stone, 1974; Toussaint, 1974) or Bootstrapping (Efron, 1979) to obtain an estimate of the true error of the hypothesis which is returned by the learner when the learner operates on model H_i. Starting with the smallest model, the learning algorithm returns one hypothesis from each model. The hold-out set is then used to obtain an estimate of the expected generalization error; the model with the lowest estimate is selected and the learner is invoked for this model with the whole sample. In order to minimize the variance of the estimate the idea of n-fold cross validation, *e.g.,* (Stone, 1974), and bootstrapping (Efron, 1979) is to average many error measurements which are generated on re-sampled data sets (the instances in the re-sampled data set are drawn from the original data set, without replacement in case of cross-validation and with replacement in the case of bootstrapping). The model which incurred the least cross-validation error is then selected and the learner is run on that model using the complete training set. This method illustrates how intimately error assessment and model selection are related. It also shows a trivial bound on how good model selection techniques can be: Suppose that we stratify the hypothesis language H into models H_1 through $H_{|H|}$ – each containing exactly one distinct hypothesis. If we then use the sample S to decide which model to chose, we will necessarily end up with a hypothesis which is just as good as the one we would have obtained by using one model containing all hypotheses. For many applications, n-fold cross validation works quite well and reliably, although it should be noted that while this class of approaches yields a reasonably accurate

estimate of the expected error it does *not* yield a good estimate of the variance. More precisely, the empirical variance is generally much less than the true variance as the n error estimates are based on "very similar" data (rather than being *independent* measurements) (Dietterich, 1997). The main drawback of this approach is the high computational effort. Depending on the sample size, the learner has to be invoked about ten times *per model* (*e.g.,* Kohavi & John, 1997). In domains in which learning is very time consuming and the number of potential models is large, cross validation may incur an unacceptably large computational effort. This is the case, for instance, in Inductive Logic Programming (*e.g.,* Muggleton, 1992) and neural segmentation of satellite data (here, the number of attributes is often extremely large; Milne, 1997).

Complexity penalization. Instead of assessing the models by means of cross validation, we could have assigned a complexity based penalty term to each model. While hold-out testing based algorithms sequentially consider the hypotheses which have been learned from increasingly complex models, complexity penalization based methods minimize a demerit criterion which consists of the empirical error and a complexity based penalty term. One member of this class is Structural Risk Minimization (SRM) (Vapnik, 1982, 1996, 1998) which is based on the VC framework (*e.g.,* Vapnik & Chervonenkis, 1971). The Support Vector Machine (SVM) is an (approximate) implementation of SRM. The "vanilla" Support Vector Machine inflates the instance space by introducing polynomials of the original attributes as new attributes. Ideally, this should result in positive and negative examples being separable by a single hyper-plane. From all those planes which are consistent with the sample the SVM chooses the one which is least with respect to a stratification that is defined in terms of the width of the margin between positive and negative samples. Effectively, the SVM returns the plane (in the inflated space) which maximizes the margin between examples of distinct classes. This particular stratification which leads to a maximally large margin has proven to be beneficial for many practical learning problems. Intuitively, the SVM works best when the two classes are somehow clustered around distinct centers. However, the SVM does not actually trade off model complexity against empirical error as the empirical error is pinned down to zero. This may result in over-fitting in cases where there is no consistent hypothesis but the best approximation is fairly simple. This constraint is weakened in Soft-Margin-Machines (Cortes & Vapnik, 1995), but only by introducing a parameter that trades a higher VC-dimension against lower observed error and that has to be adjusted by cross validation or according to some heuristic. Similarly to other complexity penalization approaches like regularization (Moody, 1992), neural weight decay methods (*e.g.,* Cun *et al.*, 1989), or decision tree pruning algorithms (*e.g.,* Quinlan, 1993; Mingers, 1989), the merit of the selected model depends strongly on the value chosen for the penalization/regularization parameter. Effectively, this parameter forms a meta-level model selection problem and "trying out" different parameter settings incurs meta-level over-fitting (Ng, 1997).

Recently, a new penalization based model selection algorithm has been proposed by Schuurmans (1997) in the context of regression. Given the distribution of unlabeled instances, one can define a metric on hypotheses, and between hypotheses and the target distribution. Knowing the distance between hypotheses one can use the triangle inequality to decide when the distance to the target distribution must be increasing. This approach turns out to perform better than cross validation and complexity penalization methods for problems with a steep variance profile (Schuurmans *et al.*, 1997).

Bayesian learners (Berger, 1985) solve both the learning and the model selection problem at the same time. Under certain ideal conditions, one can, under high computational effort, derive the Bayes hypothesis from the posterior $P(f|S)$ which is guaranteed to have the least generalization error. Intuitively, the prior $P(f)$ relates to the hypothesis complexity (in an optimal coding scheme, frequent hypotheses have a small description length) and the likelihood $P(S|f)$ relates to the empirical behav-

ior of a hypothesis function f. The Bayes hypothesis yields the optimal trade-off between likelihood (empirical behavior) and prior (complexity). Often, the posterior is used for less expensive model selection heuristics such as MAP (the *maximum a posteriori* hypothesis maximizes the chance of having generated the data) or MDL (Rissanen, 1978, 1989) (the MDL hypothesis minimizes the description length required for the data by compressing it to a hypothesis and the exceptions to the hypothesis in the data). However, the prior distribution $P(f)$ is assumed to be known in advance – which is indeed a very strong assumption. The general belief is that Bayesian learners are fairly robust against some degree of misalignment between the actual and the assumed $P(f)$. The No-Free-Lunch Theorems (Wolpert, 1992) explain that Bayesian learners perform better than randomly guessing only if the actual and the assumed prior are "not completely unaligned".

In Section 2.3, I will discuss the No-Free-Lunch Theorems which claim that one cannot construct a learner that performs better than average for all possible problems. These theorems also imply that a learner which conducts model selection cannot be superior to one that does not, averaged over all possible target functions. This raises the question in which situations conducting model selection is actually beneficial.

1.4 Applications of Machine Learning

From an engineering point of view, the most interesting aspect of machine learning is perhaps that it provides methods for automatic adaptation of a system to a particular environment. Successful applications of machine learning techniques are numerous, only few can be mentioned here.

An area of applications which is gaining interest is *knowledge discovery in databases* (*e.g.,* Holsheimer & Siebes, 1991; Fayyad *et al.*, 1996). Currently existing commercial databases contain large quantities of potentially valuable knowledge regarding, for instance, typical patterns of customer behavior. The idea of "data mining" is to automatically extract potentially interesting patterns. One of the most frequently studied problems is the discovery of association rules. Association rules (Agrawal *et al.*, 1993) are simple implications between database items of the form "*if* the customer buys beer *then* the customer is likely to also buy potato chips". The large size of typical databases imposes a particular difficulty on data mining problems. Data mining algorithms are usually required to operate in time at most linear in the size of the database – preferably even sub-linear (*e.g.,* Toivonen, 1996). Discovering patterns of customer behavior and detecting fraudulent credit card transactions belong to the most popular data mining tasks.

A large number of applications fall into the general field of *pattern recognition*. This entails computer vision (*e.g.,* Jain *et al.*, 1997), optical character recognition (*e.g.,* Cun *et al.*, 1989), and recognition of spoken words (*e.g.,* Lee, 1989).

Text categorization (*e.g.,* Salton & Ruckley, 1988) is the problem of mapping texts to semantic categories. Interesting applications are automatic classification of news stories for later research (Lewis, 1991; Lang, 1995), and the classification of web pages (*e.g.,* Joachims *et al.*, 1997).

Automatic control is a very large field of application. Algorithms which automatically acquire a control skill fall into the classes reinforcement learning and behavior cloning (the relative benefits of these two approaches have been discussed by Scheffer *et al.*, 1997). Reinforcement learning algorithms (*e.g.,* Sutton & Barto, 1998) acquire a skill by conducting control experiments and receiving performance feedback. Perhaps the most successful application of reinforcement learning is TD-gammon (Tesauro, 1992, 1995), a program that plays backgammon on world championship level. Other applications include, for instance, automatic control of cars on highways (Pomerleau, 1989). Behavior cloning algorithms which learn from examples of good behavior have been applied

to problems such as satellite control (Müller & Wysotzki, 1995) and flight simulators (Sammut *et al.*, 1992).

Knowledge acquisition (*e.g.,* Quinlan, 1989) is the process of elaborating background knowledge on some domain from a domain expert and implementing this knowledge into an expert system. Knowledge acquisition is generally considered to be the bottleneck of the construction of expert systems. Machine learning algorithms have been used to support this process by extracting knowledge from examples of behavior rather than from descriptions of the expert (*e.g.,* Gaines & Compton, 1992; Kang *et al.*, 1995; Scheffer, 1996).

There are many other applications of machine learning; classification algorithms have been applied to problems of medical diagnosis (*e.g.,* Ulbrich & Wysotzki, 1972; Richter & et al., 1974; Kononenko, 1993), regression algorithms have often been applied to share price prediction, and many other problems.

1.5 Principle Contributions

In this doctoral dissertation, I discuss a compound of questions on assessment of hypotheses and the related selection of a good hypothesis language, and learner. The following is a sketch of the main results.

1. I conduct a new analysis of the expected true error of the hypotheses which minimize the observed error. The analysis characterizes the generalization error of the apparently best hypothesis in the model in terms of the prior distribution of error rates in the model which can be estimated efficiently. The results predict and thus explain learning curves much more precisely than PAC-style results and thus contribute to a better understanding to the nature of generalization.

2. As an immediate result, the analysis leads to an efficient algorithm for model selection; its primary benefit is that it is much more efficient than cross validation, while usually being at least as accurate. I conduct a series of empirical studies which support this claim. I demonstrate the scalability of the algorithm on a text categorization problem.

3. I study in which situations conducting model selection leads to better generalization than not conducting model selection. After a couple of generally negative results, I characterize a class of learning scenarios (located in the gap between Bayesian and PAC learning) in which Occam algorithms (which can be considered a “weak” form of model selection) perform better than PAC learners. I develop a framework which quantifies the expected error of cross validation based model selection.

4. The abovementioned framework has some practical implications: It provides an answer to the question how the sample should be split into training and hold-out sets, and it predicts whether cross validation based model selection (with respect to some given stratification) will do better or worse than simple error minimization.

5. When many instantiations of a parametric learning algorithm (many learners have parameters such as learning rates, regularization parameters, and so on) are being compared with respect to their performance on a collection of data sets, some results will necessarily be optimistic while other results will be pessimistic estimates of the true performance. The best observed accuracy

is likely to be an optimistic estimate. I quantify just how optimistic this estimate is and discuss the consequences of this result for the empirical assessment of learners.

6. In the model selection approach, the learner is constrained to a fixed model. By contrast, in the boosting approach the hypothesis space is allowed to grow dynamically. By giving new worst-case time bounds for the AdaBoost algorithm I show that in many cases the restriction to small sets of hypotheses causes the high complexity of learning.

1.6 Organization

This Section gives an overview on the structure of the following chapters. In Chapter 2, I discuss some principles of machine learning and introduce the necessary definitions and methodology. In Chapter 3, I present the new analysis of the error of hypotheses which minimize the empirical error rate and the resulting model selection algorithm. I also present empirical results on learning Boolean functions and on text categorization. Chapter 4 deals with the question when model selection is beneficial and, in particular, what the expected error of cross validation based model selection is. Chapter 5 analyzes the accuracy of the apparently best of many hypotheses, generated by differently parameterized learners. Chapter 6 addresses the complexity of model selection based learning and Chapter 7 contains some concluding remarks. The Appendix contains all proofs and derivations which exceed one page in length. Appendix H furthermore contains a list of all frequently used abbreviations.

Chapter 2
Preliminaries

In this thesis, I study the problem of classification learning from labeled examples. Most results refer to the expected generalization error of hypotheses which requires the existence of a "natural" distribution of instances.

2.1 Terminology Used throughout the Book

Instances. There is a set of instances X and a finite set of class labels Y (sometimes, for simplicity, Y is assumed to be the set $\{0, 1\}$). A classification problem is defined by an unknown distribution $D_{XY} = D_{Y|X}D_X$ over labeled instances ($X \times Y$), which has to be approximated as closely as possible. $D_{Y|X}(y|x)$ is the chance that y is the class label of an observed instance x and $D_X(x)$ is the probability distribution or density which governs the instances x. In classical PAC theory, learning problems are often defined as consisting of a function f from a class of target functions F together with a distribution D_X on instances. Sometimes it is more convenient to refer to the notation of the target function f but note that this definition is subsumed by the notation of D_{XY} (proposed by Kearns *et al.*, 1992). Given a D_X and f one can define D_{XY} as $D_{Y|X}D_X$ where $D_{Y|X}(y|x)$ is 1 iff $y = f(x)$, 0 otherwise.

Hypotheses and Error. A *hypothesis* $h : X \to Y$ is a mapping from instances to class labels. The *true (or generalization) error rate of a hypothesis*, with respect to the (unknown) distribution D_{XY} is the difference between the predicted value $h(x)$ and all class labels y, weighted with $D_{XY}(x, y)$ – more formally, $E_D(h) = \int_{(x,y)\in X\times Y} \ell(h(x), y)dD_{XY}(x, y)$, where ℓ is the zero-one loss function. Sometimes, when it is more convenient to talk about target *functions* f in conjunction with distributions D_X on instances, I write the error as $E_{f,D_X}(h) = \int \ell(h(x), f(x))dD_X(x)$. I will switch between the notations of target functions f and target distributions D_{XY}, using whichever is more appropriate in the given situation. Let the *sample* S be a sequence of labeled instances drawn *independently and identically distributed* according to D_{XY}. The *sample size* is abbreviated m throughout this book. Each example is drawn according to D_{XY} or, put in another way, the whole sample is drawn according to $(D_{XY})^m$. The *empirical or observed error* is the difference between the predicted value $h(x)$ and the class label observed in the sample, for all sample instances: $E_S(h) = \frac{1}{m}\sum_{(x,y)\in S} \ell(h(x), y)$. A hypothesis that incurs an empirical error of zero for a sample S is said to be *consistent* with that sample.

Hypothesis Language and Model. There is a given hypothesis language H which may be infinite and may even have an infinite VC-dimension. A *stratification* of the hypothesis language is a finite

sequence of models $\langle H_1, \ldots, H_k \rangle$, $H_i \subseteq H$. The models do not have to properly include each other (in fact, I do not even assume that the models are monotonically growing). But in Chapter 3, I will assume that each model is a *finite* subset of H.

Learner. A *learner* L takes as input a sample S and a model H_i and returns a hypothesis h_L. The learner may be deterministic (in which case $L_{H_i}(S)$ refers to the output of L for sample S), or stochastic. In the latter case, $P_L(h_L|S, H_i)$ is a distribution (for finite H_i) or a density (infinite H_i) over hypotheses. A learner may determine the set $H_i^*(S) = \{h \in H_i : E_S(h) = \min_{h' \in H_i}(E_S(h'))\}$ of hypotheses with least empirical error. There is at least one such hypothesis. I will call a learner which determines $H_i^*(S)$ and draws one hypothesis from this set at random an ERM learner (error minimizing learner) an the corresponding hypotheses ERM hypotheses.

Definition 2.1.1 (ERM Learner) *Given a sample S and a model H_i an ERM Learner $L(S, H_i)$ returns a hypothesis h_L with minimum empirical error $E_S(h_L) = min_{h \in H_i} E_S(h)$ on S. If there are multiple hypotheses with the same minimum error, the learner picks one of them at random under uniform distribution.*

Learning curve. Let $\langle H_1, \ldots, H_k \rangle$ be a stratification of models and let L be a learner. For a sample S, a learning curve is a set of points $(i, E_D(L_{H_i}(S)))$ – *i.e.,* the learning curve displays the generalization error incurred by learner L on model H_i for all models. When the sample is not fixed yet, the expected learning curve for a fixed sample size m can be plotted. When no target distribution but a prior distribution over targets, $P(D_{XY})$, is fixed, the expected learning curve over all targets can be plotted. Often, learning curves are "U" shaped. Model selection algorithms try to determine the minimum of the learning curve. Unless the target distribution D_{XY} is known, the learning curve can only be estimated; often, the cross validation error is plotted.

Notations. I generally write probability distributions and densities in the form $P_{\{x\}}(f(x) = y)$ where the subscript x indicates that x is a random variable. The distribution of x should become clear in the given context. $P_{\{x\}}(f(x) = y)$ refers to the density of $f(X)$ at x and can, for discrete distributions, be thought of as the chance of drawing an x such that $f(x) = y$. Similarly, I write $\mathbf{E}_{\{x\}}(f(x))$ for the expectation of $f(x)$ over all x (again, the distribution of x becomes clear in the context). I write the binomial distribution as $B[n, p](x)$, denoting the chance of observing x marked instances, when drawing n instances with replacement and the chance of observing a marked instance is p. The hypergeometric distribution is written $H[m, p, c](x)$ and quantifies the chance that x instances are marked when we draw c instances from a set of m instances of which $p \times m$ are marked.

2.2 Models of Generalization

In this Section, I discuss three distinct mathematical models of generalization and their relations. In the theory of computation, there is a canonical model of computability (the Turing machine) which entails all other models of computability (*e.g.,* the Lambda calculus) and completely entails the intuition of computability. Unfortunately, there is no such canonical model of *learnability* which is powerful enough to completely capture the intuition of learning. Instead, there is a hierarchy of learnability classes in the identification in the limit framework (*e.g.,* Angluin & Smith, 1983; Jain *et al.,* 1999) and orthogonal concepts of learnability in other frameworks, such as PAC theory (Valiant, 1984), the Bayesian framework (*e.g.,* Berger, 1985), and the Statistical Physics framework (*e.g.,* Tishby, 1995). In the following, I will briefly survey the "vanilla" versions of some of these models and their relations. In this survey, I will come to a new result on the relationship between identification in the limit and PAC theory.

2.2.1 Gold's Framework of Learning

The construction of the identification in the limit model (Gold, 1967) was guided by the intuition of language acquisition; the related study of language acquisition in linguistics (Wexler & Culicover, 1980) has revealed some restrictions on the grammatical structure of natural languages which are necessary for languages to be learnable. A learner is said to identify a function (or *language*) class in the limit iff it can be guaranteed to win the following game: (a) The learner is given the class of possible target functions. (b) The teacher starts producing a "text" (*i.e.*, a sequence of instances which may consist of only positive or of positive and negative examples) such that every example occurs eventually. (c) After each new instance is read, the learner may change its mind and make a new hypothesis (or *conjecture*) about the target function. The learner wins the game if, after only finitely many mind changes, the hypothesis is correct and does not change upon reading new instances any more. This definition of learning is referred to as *explanatory* learning; the corresponding class of learnable function classes is abbreviated EX. An easy example for a class of functions in EX is the class of functions $f : \mathbb{N} \to \mathbb{N}$ (where $\mathbb{N}$ denotes the natural numbers) that are zero almost everywhere (*i.e.*, everywhere except in finitely many places). A learner which reproduces all observed nonzero function values and guesses zero at all other places identifies this class in the limit. Since there are only finitely many nonzero values and every example occurs eventually, the learner will have observed all nonzero values after finitely many examples. Note, however, that the learner does not know when it has identifies the target and, in fact, the number of examples cannot be bounded (there may be arbitrarily many nonzero values). By contrast, the class of all computable functions cannot be learned in the limit by any computable machine (Gold, 1967).

There are two important variations of Gold's learning paradigm: A learner identifies the target function *behaviorently correct* (the corresponding learnability class is abbreviated BC) if it makes only finitely many erroneous predictions and, from some finite point on, keeps making true predictions although it may still change its conjecture about the target function. BC can be shown to be a proper superset of EX (Barzdin, 1974; Case & Smith, 1983). *Finite identification* (or "one-shot learning", learnability class FIN) imposes a further restriction on the learner: After finitely many examples, the learner has to come up with one correct hypothesis; the learner may not change its mind. Clearly, this setting is more restrictive than EX-learning since the learner has to be aware whether the data seen so far suffices for a correct conjecture. Not surprisingly, $FIN \subset EX$ (Lindner, 1972).

One of the many compounds of questions which are studied within the identification in the limit framework is the learnability relative to oracles (*e.g.,* Bshouty *et al.*, 1994; Kobler & Lindner, 1997; Stephan, 1998). A set A is computable relatively to an oracle B if there is an algorithm that computes A which is allowed to ask questions of the form "is x in B?". Similarly, a class of functions is learnable relatively to an oracle B if an algorithm which may ask questions of the form "is x in B?" can identify it. A massive corpus of results exists on identifiability of functions relative to queries to a teacher (*e.g.*, Gasarch & Smith, 1988; Angluin, 1993), and on learnability of function classes by teams of learners (*e.g.,* Jain & Sharma, 1990; Smith, 1994). For a detailed overview on the identification in the limit framework the reader is referred to (Jain *et al.*, 1999).

2.2.2 The PAC and VC Models of Generalization

The PAC framework of generalization (Valiant, 1984) (for an overview, see Kearns & Vazirani, 1994; Vidyasagar, 1997) is more strongly focused on efficient learning and learning from fixed-sized samples. In PAC theory, one distinguishes between a hypothesis language H and a class of target functions F. A learner L is a PAC generalizer if it can be guaranteed to win the following game: (a) The learner

gets to know the class of possible target functions F but no further information on the target. The learner is also given parameters ε and δ. (b) At this point, the learner can request a required sample size m. Note that the sample size must only depend on F, H, and the parameters as no further information is available at this point. (c) The "teacher" now fixes a target function $f \in F$ and an arbitrary distribution D_X on instances. The teacher is free to choose any target function and may, for instance, always select the function which matches the learning bias of L the worst. (d) When a sample S of labeled instances of size m is drawn according to D_X and f and given to the learner, the chance of the learner returning a hypothesis h_L such that $P(E_{D_X,f}(h_L) > \varepsilon)$ must not exceed δ.

Hence, the learner wins (and is a PAC generalizer) if the chance of drawing a sample S of size m such that the resulting hypothesis incurs an error of more than ε is below δ for every D_X and $f \in F$. PAC theory is a worst case theory in two respects: The chosen sample size has to suffice for *any* distribution D_X and for *any* target function $f \in F$. Furthermore, most PAC bounds on the required sample size hold for *any* learner – *i.e.,* the bounds are subject to the additional assumption that the learner manages to select the worst possible hypothesis. What makes this setting particularly hard is, as we will see later, that there has to be a fixed sample size (depending only on F, H, ε, and δ) which suffices for all D_X and f. A PAC learner is said to be a polynomial PAC learner for F if the required sample size and the total running time are polynomial in $\frac{1}{\varepsilon}$, $\frac{1}{\delta}$, and the size parameter of F (usually the number of attributes n).

PAC theory is most concerned about which function classes can be learned polynomially (*i.e.,* learned in polynomial time and from a polynomially sized sample). The most fundamental learnability result has been obtained by Blumer *et al.* (1987): The chance that the error of any hypothesis h_L which is consistent with a sample of size m exceeds an error of ε can be bounded by $P(E_{D_X,f}(h_L) > \varepsilon | E_S(h_L) = 0, m) \leq |H|(1-\varepsilon)^m$. As a corollary, one can conclude that for any hypothesis h_L (which is consistent with a sample of size m) $P(E_{D_X,f}(h_L) > \varepsilon | E_S(h_L) = 0, m) \leq \delta$ holds if the sample size is at least $m \leq \frac{1}{\varepsilon} \log \frac{|H|}{\delta}$, because

$$P(E_{D_X,f}(h_L) > \varepsilon | E_S(h_L) = 0, m) \leq |H|(1-\varepsilon)^{\left(\frac{1}{\varepsilon} \log \frac{|H|}{\delta}\right)} \leq \delta. \tag{2.1}$$

This elementary result provides an easy-to-follow scheme for conducting learnability proofs in the PAC framework: A function class F can be learned polynomially using hypothesis language $H \supseteq F$ if (a) $\log |H|$ (and hence the required sample size) is polynomial in n (where n is typically the number of attributes) and (b) there is an algorithm which constructs a hypothesis which is consistent with any sample in the class of target functions in polynomial time. Several function classes have been shown to be polynomially learnable: Conjunctive concepts (Valiant, 1984), linear threshold units (Blumer *et al.*, 1989), k-DNF (Boolean disjunction with up to k literals per conjunction Valiant, 1985), k-CNF (Valiant, 1985), and k-decision lists (Rivest, 1987) are polynomially learnable for fixed k. On the other hand, disjunctions of two conjunctions are not polynomially learnable (Valiant, 1984), neither are conjunctions of two linear threshold units (Blumer *et al.*, 1989), or conjunctive concepts in structural domains which consist of at least two nodes with n unary attributes (Haussler, 1989). Pattern languages with one variable (Mitchell *et al.*, 1998) are not PAC learnable at all (not only not *polynomially* learnable). It is still unknown whether Boolean formulae in disjunctive normal form (DNF) are polynomially learnable and whether there exists a decision tree learner which runs in time polynomial in the size of the smallest possible decision tree. In virtually any hypothesis space, a *locally* optimal hypothesis can be found in polynomial time (Greiner, 1996).

Some function classes are not polynomially learnable when H equals F because there is no polynomial algorithm which constructs a consistent hypothesis. In some cases, these classes *become* polynomially learnable when the hypothesis language H is *extended*. This is a puzzling finding as

learning generally becomes more difficult for larger hypothesis spaces. One example of this is k-term CNF(n) which contains conjunctive normal forms with up to k conjunctions of disjunctions (with arbitrarily many literals) over n variables (Pitt & Valiant, 1988). Since $\log|k\text{-term CNF}|$ is polynomial in n, the required sample size is polynomial, too. But in order to find a consistent hypothesis one has to exhaust the whole hypothesis space which requires exponential computational effort. The class k-DNF (disjunctions of conjunctions with up to k literals) is a superset of k-term CNF although $\log|k\text{-DNF}|$ is still polynomial (and so is the sample size required to learn k-DNF). However, while learning k-term CNF requires to exhaust the whole space, there is a greedy algorithm which learns k-DNF in polynomial time. The algorithm first finds a conjunction of up to k literals which covers at least one positive and no negative example. The procedure then commits to this conjunction, removes all covered instances, and recurs until all positive instances are covered. There are further examples for cases in which extending H eases the learning task: While it is NP-complete to find a consistent neural network with three units (Höffgen *et al.*, 1995), this task can be accomplished in $O(m^2 \log m)$ when arbitrarily many units can be introduced by the learner (Chapter 6 and Scheffer & Stephan, 1998).

Classical PAC theory is subject to two major restrictions. First, the target function is assumed to be a member of the hypothesis language (*i.e.,* a consistent hypothesis is always assumed to exist) and, second, the "standard" PAC bounds and proof techniques do only work for finite hypothesis languages. The finiteness of H is a particularly strong restriction because finite languages are always PAC learnable (albeit not necessarily polynomially learnable). Agnostic learning theory (Kearns *et al.*, 1992; Haussler, 1992) aims at extending PAC theory to account for cases in which no hypothesis which is consistent with the sample is available. Independently (but much earlier), Vapnik and Chervonenkis (1971) developed a theory which is connected to PAC theory by a fundamental result and allows for results which are very similar to the elementary results of agnostic learning theory. Vapnik tried to find a statistical explanation for the success of Rosenblatt's experiments on the perceptron (Rosenblatt, 1958). Since linear threshold units form an infinite hypothesis space, Vapnik and Chervonenkis focused on the number of behaviorently distinct hypothesis to find a bound on the true error of hypotheses. The VC-dimension is a combinatorial property of hypothesis languages which accounts for the number of behaviorently distinct hypotheses and is defined as follows: A set S is shattered by a set H if for any subset $S' \subset S$ there is an element $h \in H$ such that $S \cap h = S'$. Intuitively, S is shattered by H iff S can be labeled in all possible $2^{|S|}$ ways by H. The VC-dimension of H is the largest number d such that there is a set S with $|S| = d$ which is shattered by H. Intuitively, the VC-dimension of some language H is d iff H can realize all 2^d possible Boolean functions on d instances. The fundamental result that links PAC and VC theory together is that a class of functions F is PAC-learnable iff the VC-dimension of F is finite (Blumer *et al.*, 1989). The VC dimension can now be used to bound the difference between the empirical and the true error of any hypothesis, based on Chernoff bounds. This leads to lower (Ehrenfeucht *et al.*, 1989) and upper (*e.g.,* Vapnik, 1982, 1996) bounds on the sample size required for the empirical error of every hypothesis in the hypothesis space being ε-close to the corresponding true error. Vapnik (1982) proves that the largest difference between true and empirical error rate of any hypothesis in the model is, with a confidence of at least $1-\delta$, no more than

$$2\sqrt{\frac{d\left(\log\frac{2m}{d}+1\right)+\log\frac{9}{\delta}}{m}}$$

where d is the VC-dimension of H_i.

2.2.3 The Relationship between PAC and Gold's Framework

What is the main difference between Gold's and Valiant's models of generalization? If we analyze explanatory (EX) and PAC learning, two distinctions occur: (a) While EX learning requires an exact match between f and h (a BC learner is still required to emulate the behavior of f perfectly from some point on), a PAC generalizer is only required to find a hypothesis the behavior of which is ε-close to the behavior of f with respect to D_X (with high probability). This argues that Gold's framework might be stronger than PAC learning. (b) At no point in time does the EX learner know that the target function has been identified. A FIN learner, by contrast, does know when f has been identified but even with the FIN learner the required number of learning steps cannot be bounded *a priori* (*i.e.,* before any examples have been observed). However, the sample size required by the PAC learner has to be bounded, given only the class of target functions F and no further information. This argues that PAC is actually more restrictive than identification in the limit. I will now study the question which of these frameworks is stronger, referring to a function class which has been studied in both, the PAC and the identification in the limit framework – namely pattern languages.

The simple and intuitive notion of pattern languages was formally introduced by (Angluin, 1980a) and has been studied extensively, both in the context of formal language theory and computational learning theory. I refer the reader to Salomaa (1994a, 1994b) for a review of the work on pattern languages in formal language theory.

A quick definition of pattern languages is useful for discussion. Let Σ (the *alphabet*) be a countable set of constants of the language and V (disjoint from Σ) be a countable set of variables. Any element of $(\Sigma \cup V)^*$ is a *pattern*. Let p be a pattern and let $x_1, \ldots, x_n$ be the list of all distinct variables in p. Let strings $u_1, \ldots, u_n \in \Sigma^+$. Then $p\{x_1/u_1, \ldots, x_n/u_n\}$ denotes the string $w \in \Sigma^+$ obtained by substituting u_i for each occurrence of x_i in p. The *language generated by* p is defined as $L(p) = \{p\{x_1/u_1, \ldots, x_n/u_n\} \mid u_1, \ldots, u_n \in \Sigma^+\}$. Let Pat denote the set of all patterns and PAT denote the set of all pattern languages.

Angluin (1980b) showed that the class PAT is identifiable in the limit from only positive data in Gold's model. Since its introduction, pattern languages and their variants have been a subject of intense study in the identification in the limit framework (for a review, see Shinohara & Arikawa, 1995). The reader should note that the definition of Angluin's class, PAT, does not allow for empty substitutions. If empty strings are allowed to be substituted for variables, this leads to the larger class, $ePAT$, of extended pattern languages (see Shinohara, 1982). This class turns out to be very complex and it is still open whether for finite alphabets of size > 1, $ePAT$ can be identified in the limit from only positive data[1].

Since PAT is identifiable in the limit from only positive data, a natural question is if there is any gain to be had if negative data is also present. Lange and Zeugmann (1993) observed that in the presence of both positive and negative data, the class PAT is identifiable with 0 mind changes, that is, there is a learner that after looking at sufficient number of positive and negative examples comes up with the correct pattern for the language (this restricted "one-shot" version of identification in the limit is referred to as *finite identification*). Theorem 1, however, proves that even 1-variable pattern languages are not learnable in the PAC setting.

Theorem 1 (Mitchell, Scheffer, & Sharma, 1998) *Let $k > 0$. The VC-dimension of k-variable pattern languages is unbounded.*

[1] See Mitchell (1998) where a subclass of $ePAT$ is shown to be identifiable in the limit from only positive data. Mitchell *et al.* (1998) also show that $ePAT$ is learnable if the alphabet size is 1 or ∞.

The proof of Theorem 1 can be found in (Mitchell *et al.*, 1998). This Theorem is a significant strengthening of Schapire's negative result on general pattern languages (Schapire, 1990) and, at first blush, seems to contradict the learnability result for k variable pattern languages by Kearns and Pitt (1989). However, a closer look at the result of Kearns reveals that the learnability is only due to the assumption that the length of the substitution strings be bounded. Is there a more precise explanation why pattern languages are EX learnable but even 1-variable pattern languages are not PAC learnable?

Theorem 2 (Mitchell, Scheffer, & Sharma, 1998) *Let ε and δ be given. Let L be a k-variable pattern language and D be an arbitrary distribution on Σ^*. Let S be an initial set of positive sentences of size at least one and let $l_{min} = \min\{|x| \mid x \in S\}$. Let h be any pattern consistent with a sample of size at most $m \geq \frac{l_{min}}{\varepsilon} \log \frac{1}{\delta \times \log(l_{\min}+k)}$. Then $P(Err_{L,D}[h] > \varepsilon) \leq \delta$.*

Theorem 2 (the proof can be found in Mitchell *et al.*, 1998) claims that k-variables pattern languages *can* be learned, but that the required sample size can only be bounded after the first positive example has been observed. The reason is that, after the first positive sentence has been read, the length of the target pattern and the alphabet can be bounded which renders the space of possible target patterns finite and thus PAC learnable. So far, we have seen that FIN and PAC have a nonempty intersection. This raises the question whether PAC is a subset of FIN. This, however, is not the case. In fact, PAC is not even a subset of BC. Consider the class ZO of linear threshold functions over the one-dimensional real-valued interval $[0, 1]$. The VC dimension of this ZO is 2. But ZO cannot be BC-identified in the limit.

Theorem 3 *The class of threshold functions over the one-dimensional real-valued interval $[0, 1]$ cannot be BC-identified in the limit ($ZO \notin BC$).*

Proof. Let the target function be $x \leq t$ where t is drawn quasi-uniformly from the interval $[0, 1]$. I will show that, after m positive and m negative instances have been observed, the expected difference between h and f is still positive (*i.e.*, there is no sample size m from which on the hypothesis behaves correctly). Let $x_1, \ldots, x_m$ be the positive and $\bar{x}_1, \ldots, \bar{x}_m$ the negative examples. According to the definition of $P(f)$, the positive instances are drawn quasi-uniformly from the interval $[0, t]$ and the negatives from $(t, 1]$. Let $x_{max} = \max\{x_i\}$ be the largest positive and $\bar{x}_{min} = \min\{\bar{x}_i\}$ the least negative instances. Since the negatives are drawn from the open interval $(t, 1]$, $\bar{x}_{min} - x_{max} > 0$ holds. The quasi-uniform distribution of t together with the bounds of $\bar{x}_{min}$ and x_{max} for possible values of t induce a quasi-uniform distribution of t in the interval $[x_{max}, \bar{x}_{min})$. Let $\hat{t}$, $x_{max} \leq \hat{t} < \bar{x}_{min}$, be the guess for t made by some learner. Then, averaged over all possible t (distributed according to $P(f)$ restricted on $[x_{max}, \bar{x}_{min})$) the difference between t and $\hat{t}$ is $\int_{x_{max}}^{\bar{x}_{min}} |\hat{t} - t| dt = \frac{x_{max} - \bar{x}_{min}}{2} > 0$. Hence, the learner cannot be guaranteed to produce even a behaviorently correct hypothesis after $2m$ steps (for arbitrary m). ∎

These results prove that the learnability concepts of PAC and identification in the limit are orthogonal, as illustrated in Figure 2.1.

2.2.4 The Bayesian Framework

Bayesian learning (Bayes, 1763) is focused on the posterior distribution $P(f|S)$ of a function f having generated the observed data S. Given $P(f|S)$, one can construct the *Bayes* hypothesis $\hbar$ as

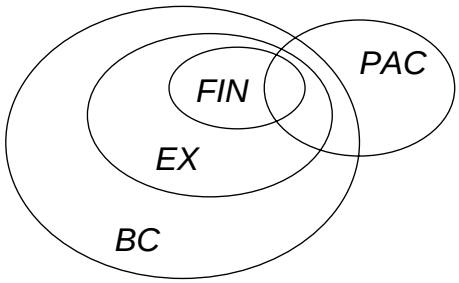

Figure 2.1: Relation between PAC and identification in the limit.

follows: $h^*(x) = \text{argmin}_{y \in Y}\{\sum_f \ell(f(x) - y)P(f|S)\}$ where ℓ is the zero-one loss function[2]. The fundamental Theorem of Bayes is that the Bayes hypothesis minimizes the expected generalization error. Unfortunately, $P(f|S)$ is hard to get hold of. Using Bayes rule, it can be reduced to $P(S|f)\frac{P(f)}{P(S)}$ and $P(S)$ can be reduced to $\sum_f P(S|f)P(f)$ (for the MAP hypothesis, $P(S)$ does not have to be determined as it is constant for all hypotheses, given a particular data set). $P(S|f)$ can in some cases be determined, but this still leaves us with two problems: $P(f)$ has to be known in advance (which is a strong assumption) and, in order to determine the Bayes hypothesis, one has to sum over the whole space of functions $f \in F$. There is an ongoing philosophical debate about what $P(f)$ "means". While frequentists like to think of probabilities as "objective" properties of intrinsically stochastic systems (*e.g.,* Neyman, 1967), Bayesians see probabilities as a means of modeling the behavior of a system by a subject who cannot predict the precise behavior of that system, due to a limited level of informedness (even if the system is not intrinsically stochastic). This motivates the notation of "subjective probabilities". Some Bayesians take this idea one step further and think of probabilities as subjective *beliefs* about the chances of events (again, the underlying systems are not required to be intrinsically stochastic). For a more detailed discussion on the meaning of probability, see Berger (1985), Jeffreys (1961), de Finetti (1972), Schafer (1981), Good (1983). As far as it concerns the results in this book, probabilities can be interpreted as both, intrinsically to the system and subjectively based on lack of information, but I will distinguish between "actual" probabilities of the physical reality and the belief of some agent.

Since computing the Bayes hypothesis (even if $P(f|S)$ is given) is extremely expensive, several heuristics have been proposed. A well-known heuristic is to choose the MAP (maximum *a posteriori*) hypothesis which maximizes $P(f|S)$. *Minimum description length* (MDL) (Rissanen, 1978, 1985, 1989) is another well-known heuristic. In order to communicate a sample of labeled instances one can transmit the data set by itself or, alternatively, find a hypothesis which covers some of the instances and append those instances which are not properly classified by the hypothesis (the exceptions). The more complex this hypothesis is, the more examples will be covered and the less exceptions have to be appended. The MDL hypothesis is the one which minimizes the description length of the hypothesis plus the description length of the exceptions. Of course, determining the description length of a hypothesis requires the definition of an optimal code which, in turn, requires knowledge of the prior distribution $P(f)$.

Certainly, one of the main drawbacks of the Bayesian framework (including MDL and MAP hypotheses) if its demand for $P(f)$ to be known. The third No-Free-Lunch Theorem (Wolpert, 1992)

[2]For function approximation (as opposed to classification), usually the quadratic loss is chosen as ℓ-function and the corresponding Bayes hypothesis can then be guaranteed to minimize the quadratic loss

claims that when $P(f)$ is not known and the learner assumes some $Q(f)$, the expected off-sample error (averaged over all possible priors $P(f)$) is just the error achieved by random guessing. It is, however, most interesting (and realistic) to assume that *some* knowledge on $P(f)$ is given (this scenario is referred to as *robust Bayesian learning*; *e.g.,* Berger, 1993). Technically, this is modeled by assuming that the true $P(f)$ is an element of a *class* of possible distributions. Several classes of distributions have been studied intensely. *ε-contamination classes* are defined as all distributions $Q(f)$ which can be written as $(1-\varepsilon)q + \varepsilon c$ where q is an assumed prior and $c \in C$ is the contamination; C can, for instance, be defined as the class of all distributions (*e.g.,* Berger & Berliner, 1986; Moreno & Cano, 1991; Bose, 1994). *Moment classes* contain all priors with a common set of moments (*e.g.,* Sivaganesan & Berger, 1993; Goutis, 1991). *Density bands* are classes of priors $Q(h)$ which lie between bounds $L(h) \leq Q(h) \leq U(h)$. Consequently, such priors do not necessarily have to integrate to 1. The justification of such *generalized priors* is that if it cannot be guaranteed that the target function is in H, then $\int_h P(h)$ can be less than 1 (*e.g.,* DeRobertis, 1978; Hartigan, 1983; Lavine, 1992; Sivaganesan, 1994). Many other classes of priors are studied, such as quantile classes (Cano *et al.*, 1985), mixture classes (Bose, 1993), and shape or smoothness classes (Bose, 1994). The hypothesis found by a robust Bayesian learner is off by $\sum_f (P(f|S) - Q(f|S))f(x))$ when $P(f|S)$ is the actual and $Q(f|S)$ the assumed posterior, which also yields implications on the gain on error incurred by a lack of knowledge on the prior.

2.2.5 Links between PAC/VC and Bayesian Learning

Relations between the VC framework and Bayesian learning have been characterized by Haussler *et al.* (1994). Haussler *et al.* show how the VC dimension can be related to the learning curves of Bayesian learners (under both correct and inaccurate priors). McAllester (1998) gives some PAC style error bounds for consistent hypotheses, in the presence of a (known) prior distribution. Another interesting link between PAC and Bayesian learning is given by the No-Free-Lunch Theorems (Wolpert, 1992): Due to the worst-case nature of PAC theory (worst-case with respect to the target function), the PAC error bounds can be achieved by any learner which reflects the data but behaves arbitrarily poorly on all other instances. The No-Free-Lunch Theorems prove that it is impossible to generalize a function for unobserved instances unless information on the prior is given. Baxter (1997a) has studied a link between *hierarchical* Bayesian learning and the PAC/VC framework. A hierarchical Bayesian learner (Good, 1983) is one that has a prior distribution $P(P(f))$ over priors over target functions available; *i.e.,* it does not know the actual prior $P(f)$ but it knows how likely certain priors are. By studying a *sequence* of learning problems, a hierarchical Bayesian learner can attempt to identify the prior $P(f)$ that actually generates the target functions which in turn generate the data. Baxter (1997b) gives a PAC-style analysis which bounds the number of learning problems which have to be observed for the prior to be estimated to some degree of accuracy. His main result are (PAC-style) error bounds for the n-th hypothesis (after $n-1$ problems have been observed) which are considerably lower than the known bounds on stand-alone learning problems.

In Chapter 4, I will establish further links by studying how partial knowledge on the prior can improve the sample complexity, compared to a PAC learner which has no such knowledge. I will also discuss the learnability of function classes which are not PAC-learnable without knowledge on the prior.

2.3 No Free Lunch

In Section 1.2, I discussed the question whether any learning bias can be superior to all other learning biases informally. PAC results by Blumer *et al.* (1987) and similar VC style results (Vapnik, 1998) which show that we can be more certain about the error rate of a hypothesis which has been learned from a small hypothesis language than we can be about the error rate of a hypothesis which origins from a large hypothesis language give rise to the idea that this might be the case. This idea experiences further intuitive support by Breiman's bias-variance decomposition (Breiman *et al.*, 1984). Breiman *et al.* split the error rate of the CART algorithm into a bias term (which is the error rate of the tree of a given depth which has the class labels assigned optimally to the leaf nodes) and the variance term (which is the error rate that is imposed by an improper labeling of the nodes caused by insufficient data). Breiman then shows that the bias term decreases when the hypothesis complexity (*i.e.*, the tree depth) is increased. He also shows that *we know less* about the variance term of the error when the complexity grows (*i.e.*, worse bounds can be proven) which is frequently misunderstood as meaning that the variance term increases. These results have lead to the general believe that there is such a thing as a "good" learning bias and that, in particular, the complexity of the hypothesis has to be regularized (*i.e.*, a model has to be selected) for the hypothesis to be accurate. However, examples of complementary behavior have been observed. Fisher and Schlimmer (1988) have observed problems for which unpruned decision trees outperform decision trees with regularized complexity. Schaffer (1993a, 1993b) shows several controlled experiments which support the idea that the suitability of a particular learning bias is a *property of the learning problem studied*, rather than being a property of the learning bias. Schaffer (1994) argues that conducting model selection is just a particular learning bias rather than being inherently useful. Wolpert (1993) has studied this question mathematically and came to a result which clarifies the meaning of the learning bias. The No-Free-Lunch Theorems claim that, uniformly averaged over all problems, two learning biases are equally good, provided that all learners minimize the empirical error.

Theorem 4 (Wolpert, 1992, 1993, 1995) *Let L_1 and L_2 be two arbitrary learners which implement distributions $P_1(h_L|S)$ and $P_2(h_L|S)$ over returned hypotheses. Let $E_D^{-S}(L_1(S)) = \int_{h_L} E_D^{-S} dP_1(h_L|S)$ and $E_D^{-S}(L_2(S))$ be the expected off-sample error of h_L incurred by learners L_1 and L_2, respectively (the true error on all instances except for those which occur in the sample). Let $D_{XY} = D_{Y|X} D_X$ where $D_{Y|X}$ implements a function from X to Y ($D_{Y|X}(y|x) = 1$ if $y = f(x)$ and 0 otherwise). Let, furthermore, X and Y be finite*[3]*. The following equations hold for all D_X and uniformly averaged over all f.*

1. *Uniformly averaged over all targets functions f, $\mathbf{E}_{\{D_{XY},S\}}(E_D^{-S}(L_1(S))|D_{XY}, m) - \mathbf{E}_{\{D_{XY},S\}}(E_D^{-S}(L_2(S))|D_{XY}, m) = 0$. In other words, for any D_X and uniformly averaged over all targets f, two learners incur an equal off-sample error.*

2. *Uniformly averaged over all targets f, for any sample S, $\mathbf{E}_{\{D_{XY}\}}(E_D^{-S}(L_1(S))|D_{XY}, S) - \mathbf{E}_{\{D_{XY}\}}(E_D^{-S}(L_2(S))|D_{XY}, S) = 0$. For any sample S, two learners incur an equal off-sample error, averaged over all targets.*

3. *Uniformly averaged over all distributions on target functions $P(f)$, $\mathbf{E}_{\{P(f),S\}}(E_D^{-S}(L_1(S))|m) - \mathbf{E}_{\{P(f),S\}}(E_D^{-S}(L_2(S))|m) = 0$. When the prior $P(f)$ is not known, two learners (which perhaps assume distinct priors) perform equally well, averaged over all possible priors.*

[3] The No-Free-Lunch Theorems can extended to cover countable (but infinite) domains X (Wolpert, 1993)

4. *Uniformly averaged over all $P(f)$ and for all samples S, $\mathbf{E}_{\{P(f)\}}(E_D^{-S}(L_1(S))|S) - \mathbf{E}_{\{P(f)\}}(E_D^{-S}(L_2(S))|S) = 0$. When the prior $P(f)$ is not known, two learners (which perhaps assume distinct priors) perform equally well, averaged over all possible priors, for any given sample.*

The first and third No-Free-Lunch Theorems discuss the expected (off-sample) error rate when no sample has been drawn yet. By contrast, Theorems two and four discuss the error rate for a particular sample. Note that the definition of generalization error used in this book (see Section 2.1) resembles the error rates used in Theorems (1) and (3) (but note that the No-Free-Lunch Theorems refer to the *off-sample* error rather than the generalization error). Claims (1) and (2) hold for a particular problem while Claims (3) and (4) discuss the expected error rate over a distribution of problems. In Section 1.2, I discussed an example which demonstrates that, when only four instances are not present in the sample, there are 2^4 hypotheses that are consistent with the sample and assign distinct combinations of class labels to the four remaining instances. There are also 2^4 target functions which assign distinct combinations of class labels to these instances. When we average the error rate of any of these hypotheses over the 2^4 possible target functions, it turns out that all error rates are, on average, equal.

The No-Free-Lunch Theorems explain that it is impossible to construct a learner which is both general and accurate. But this does not mean that, for a particular problem, all learners are equally good. But when learner L_1 is superior to L_2 for a given problem, this implies that there is at least one problem for which L_2 is superior, because both learners are equally good when we average over all possible problems. The only way in which we can construct a learner which is better than average for a particular (set of) problems is to implement additional assumptions on the target problems in the learner and thereby narrow the range of suited applications of the learner down. Note that this does not contradict the assumption that machine learning as such is useful. It only means that, in order to obtain good learning results, it is necessary to implement as much background knowledge into the learning algorithm as is available on the focused problems.

2.4 Model Selection

I discussed the intuition of model selection and some well-known model selection techniques informally in Section 1.3. In this Section, I will treat some techniques more formally and summarize some results on these approaches. Generally, the task of model selection is to select a model H_i such that a given learner minimizes the expected loss when using model H_i. Since all results presented in this thesis refer to the zero-one loss function (or generalization error), the discussion in this Section is restricted to the generalization error as loss function.

Model selection requires the definition of a stratification of models $\langle H_1, \ldots, H_k \rangle$, where each H_i is a set of hypotheses. Some algorithms require that the H_i be a sequence of nested models ($H_1 \subset H_2 \subset \ldots \subset H_k$) and, in some cases, the sequence is allowed to be infinite. Furthermore, a learner L_{H_i} is fixed which either maps samples to hypotheses in H_i or, when L_{H_i} is nondeterministic, is characterized by a distribution $P_L(h|S, H_i)$ over hypotheses, given a sample. A model selection problem is then given by $(\langle H_1, \ldots, H_k \rangle, L, m)$; the task of a model selection algorithm is to select a model H_j such that, for deterministic learners, the expected error of $L_{H_i}(S)$, $\mathbf{E}_{\{S\}}(E_D(L_{H_j}(S)))$ and, for nondeterministic learners, the expected generalization error over all possible resulting hypotheses $\mathbf{E}_{\{S,h_L\}}(E_D(h_L)) = \int_{h_L} E_D(h_L) dP_L(h_L|S, H_i, D_{XY})$ is minimized.

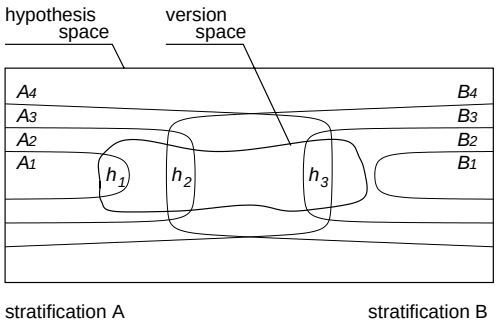

Figure 2.2: Occam's Razor: Two competing Occam learners give distinct guarantees for hypotheses h_1 through h_3: Learner A guarantees that (with high probability) h_1 incurs at most a very low error ε_1, h_2 incurs at most a slightly greater error ε_2, and h_3 incurs at most a fairly high error ε_3. By contrast, learner B guarantees a fairly low error ε_2 for hypothesis h_2 and gives only a weak guarantee of ε_3 for h_1. This is because the two learners operate with respect to distinct stratifications which give different definitions of "simple". The No-free-Lunch Theorems claim that h_1, h_2, and h_3 have equal errors, averaged over all target concepts, and are therefore equally good.

2.4.1 Occam Algorithms

Occam's Razor is one of the oldest elements of folk-lore in machine learning. It dates back to the work of William of Ockham, a 14th century philosopher. Back then, William claimed that "items should not be multiplied unnecessarily". In machine learning, this statement is generously translated to "the simplest of many explanations is the best one" and generally referred to as the Occam's Razor principle. The justification that PAC theory gives for this statement (Blumer *et al.*, 1987) is that the error bounds for hypotheses that are consistent with the sample become weaker as the size (or complexity) of the hypothesis grows. As we have seen in Chapter 1, it can easily be proven that $P(E_D(h^*) > \varepsilon | E_S(h^*) = 0, m) \leq |H|(1-\varepsilon)^m$. This motivates Occam algorithms (which require the stratification to be nested, $H_1 \subset \ldots \subset H_k$) to select the smallest index i such that $L_{H_i}(S)$ is consistent with the sample. An example of such an Occam algorithm is the vanilla Support Vector Machine. The Support Vector Machine defines the stratification of the models in terms of the distance between the hyper-plane and positive and negative examples. Because all models are infinite (hyper-planes are defined in terms of real-valued vectors), the VC dimension is considered as a measure of the number of behaviorently distinct hypotheses. Note, however, that the stratification of models depends on the sample (obviously, the distance between the plane and positive and negative examples depends on the examples) which makes all guarantees that the Support Vector Machine comes with approximate. At first blush, returning a hypothesis from a restricted subset of the hypothesis space seems reasonable as it will enable us to give better guarantees on the accuracy of the returned hypothesis. However, we have to wonder whether it becomes only possible to prove a better bound on the largest difference between empirical and true error of any hypothesis, or whether the accuracy of the returned hypothesis actually increases when we choose the smallest sufficient model. Suppose that there are two Occam learners, A and B with different stratifications, $A_1 \subset \ldots \subset A_4$ and B_1 through B_4, respectively, as illustrated in Figure 2.2. The version space entails all hypotheses which are consistent with some sample. Perhaps learners A and B use distinct encoding schemes for hypotheses and so their opinions about which hypotheses are "simple" and thus belong into a model with small index differ. According

to Vapnik (1982), learner A can prove a much better bound for $h_1 \in A_1$ than it can for h_3 which lies only in A_3. This is because model A_3 (which is larger than A_1) is much more likely to contain a hypothesis that is consistent with the sample that incurs an unusually great true error and, in the worst case, A could return this hypothesis. By contrast, learner B guarantees a fairly low error ε_2 for hypothesis h_2 and gives only a weak guarantee of ε_3 for h_1 (because B_3 is larger than B_2 and more likely to contain an unusually poor hypothesis which learner B could return, in the worst case). Is h_1 (which learner A prefers to h_2 and h_3) intrinsically better than h_3? The clear answer which is provided by the first claim of the No-Free-Lunch Theorems is no. Uniformly averaged over all target functions, all three hypotheses incur an equally large generalization error (because all three of them are consistent with the sample). Therefore, neither of the learners A and B is intrinsically superior to the other one. Learner A does better for target functions which lie "on the left hand side" of the hypothesis space of Figure 2.2 (because in this case there is a hypothesis in a model with small index of stratification A which is consistent with the sample which will allow learner A to guarantee a low error) while learner B will be performs better for target functions "on the right hand side" of Figure 2.2. This argues that in a Bayesian scenario (where certain target functions are more likely than others) Occam's razor can indeed be justified. Suppose that the models are stratified such that $P(H_1) \geq \ldots \geq P(H_k)$ (where $P(f)$ is the prior probability of f being the target). Thus, when one picks the hypothesis which is consistent with the sample from the model with the least index, one maximizes the prior and thereby the posterior probability that the selected hypothesis is the target function. When the prior $P(f)$ is known, one can construct an optimal representation scheme H. Such an optimal representation minimizes the message length which is required to transmit functions, when these functions are drawn with respect to $P(f)$. In an optimal coding scheme, functions which occur frequently are represented by shorter texts than those occurring only very infrequently. In this context, Occam algorithms certainly make sense: By choosing the shortest hypothesis that is consistent with the sample one maximizes the prior probability (and thereby implicitly the posterior probability) of picking the correct target function.

Occam algorithms are a special case of model selection algorithms because they do not trade off increases in empirical error against increases in complexity, but only select between several hypotheses that are consistent with the sample.

2.4.2 Complexity Penalization

Occam algorithms which only return hypotheses that are consistent with the sample are a special case of complexity penalization algorithms. A complexity penalization algorithm uses a stratification $\langle H_1, \ldots, \rangle$ with $H_i \subset H_{i+1}$ and returns the hypothesis h which minimizes a demerit criterion $G(E_S(h), H_i)$ where H_i is the model with the smallest index i in which hypothesis h occurs. The penalty term $G(\cdot, \cdot)$ penalizes the size or, (more frequently) when the models are infinite, the number of free parameters (Moody & Utans, 1992; Moody, 1992) or VC dimension (Vapnik, 1998) of the model H_i in which a focused hypothesis h occurs for the first time in the stratification. Frequently, $G(E_S(h), H_i) = E_S(h) + penalty(H_i)$. By means of a penalization term, complexity penalization algorithms try to reconstruct the learning curve $(i, E_D(L_{H_i}(S)))$ from only $E_S(h)$ and some complexity measure of H_i. Ideally, G should be such that $G(E_S(h), H_i)$ is a close guess of $E_D(h)$. Sometimes, complexity penalization algorithms are thought of as penalizing the complexity (or VC dimension) of a hypothesis. But note that the underlying suggestion that hypotheses possess an intrinsic complexity (or even a VC dimension) is misleading. The complexity of a hypothesis can – if at all – only be measured relatively to an encoding scheme. By varying the encoding scheme, we can assign a hypothesis any description length which we wish to. It is even more confusing to speak of the

VC dimension of a hypothesis, because the VC dimension is only defined for sets, not for elements (see Section 2.2.2). The VC dimension of the model H_i which h is included in depends on which hypotheses we wish to group into H_i – a completely arbitrary decision. An example of complexity penalization is the Support Vector Machine (SVM). While the original SVM (Vapnik, 1982) required the empirical error to be zero (see Section 2.4.1), the SVM as extended by Cortes and Vapnik (1995) minimizes the empirical error plus an approximated worst-case bound on the difference between true and empirical error of any hypothesis in the model.

Often, Minimum Description Length (MDL; Rissanen, 1978, 1989) is considered to be a complexity penalization based model selection algorithm. This is essentially correct because MDL tries to reconstruct the learning curve $(i, E_D(L_{H_i}(S)))$ from only the models H_i and the empirical error $E_S(L_{H_i}(S))$. It does so by considering the description length required to encode a hypothesis h from H_i plus the description length required to encode the difference between the examples $(x, y) \in S$ and $(x, h(x))$, the empirical behavior of h on the instances x which occur in the sample. MDL then chooses the model H_i which minimizes the sum of these description lengths. However, one also might instead think of MDL as a heuristic strategy of Bayesian learning, because MDL assumes that the prior $P(f)$ be known (the prior is required to determine the optimal code) but selects a hypothesis which is distinct from (but much easier to determine than) the Bayes hypothesis (see Section 2.2.4). For a discussion of the MDL principle and its applications to machine learning see, *e.g.*, , (Oliver & Baxter, 1994; Mehta *et al.*, 1995; Oliver *et al.*, 1996; Atteson, 1991; Grünwald, 1996; Quinlan & Rivest, 1987).

Complexity penalization based model selection algorithm reflect the idea that the generalization error is an intrinsic property of the hypothesis language – because they try to reconstruct the learning curve from only the empirical error and some complexity measure of the hypothesis space. The considerations discussed in Sections 1.2 and 2.3 show that this approach has certain limitations. For some problems and stratifications, the (variance term of the) generalization error increases steeply with increasing model index while for other problems and stratifications this is not the case. It is always possible to construct a pair of learning problems, one with a steeply increasing and one with a flat generalization error curve. Any complexity penalization algorithm can only perform well for one of these two problems and will fail (*i.e.,* produce an additional constant error λ which does not decrease with increasing sample size) for the other. This has been proven by Kearns *et al.* (1997) and has been observed empirically by Schuurmans *et al.* (1997). This is an inherent weakness of complexity penalization; by contrast, a cross validation based model selection algorithm can perform "reasonable" for all model selection problems.

Virtually all "practical" learners employ some sort of complexity penalization technique; for instance, decision tree learners (*e.g.,* Mingers, 1989; Quinlan, 1993; Müller & Wysotzki, 1992), and many neural network based algorithms (*e.g.,* Cun *et al.*, 1989).

2.4.3 Cross Validation

The term *cross validation* or, slightly more general, *hold-out testing* entails a wide range of techniques for estimation of misclassification probabilities and, intimately related, choosing a model, or a learning algorithm, which minimizes this estimate. Many cross validation based techniques do not only provide an estimate of the error but also an (often biased) estimate of the variance, cross validation results are also used to support (approximate) guarantees that, for a given problem, one technique performs better than another one (with high confidence).

Suppose that we have a hypothesis h with true error $E_D(h)$. When we draw a sample S with m independent and identically distributed instances, we can measure the empirical error $E_S(h)$. When h

has been chosen such that it minimizes $E_S(h)$, then $E_S(h)$ is a strongly optimistically biased estimate of $E_D(h)$ – *i.e.,* $\mathbf{E}_{\{S\}}(E_D(h) - E_S(L_{H_i}(S)))$ is in this case positive. In the earliest days of pattern recognition, this training set error has been proposed as an estimate of the true error rate (Smith, 1947) but such an estimate cannot be used to select a model because it would always tend towards greater models which impose a stronger bias. However, when the choice of h is independent of the sample S (*i.e.,* S is a hold-out sample which has not been used for training), then $E_S(h)$ is distributed according to $B[m, E_D(h)]$, where B is the binomial distribution. The technique of splitting the available sample into a training set and a hold-out set (originally known as H method, or hold-out testing) has been proposed by Highleyman (1962). When m is greater than 30, the empirical error is approximately normally distributed. This is because the empirical error is essentially a sum of many random variables – and the distribution of a sum of random variables always converges towards a normal distribution (central limit theorem). Keeping this in mind, one can easily determine the chance that the difference between true and empirical error exceeds a certain threshold. Such a threshold which is not exceeded (with high confidence) is called a confidence bound. $\frac{|E_D(h)-E_S(h)|}{\sigma}$ is governed by the normal distribution when $\sigma = \sqrt{(1/m)E_D(h)(1-E_D(h))}$ is the true standard deviation of $E_S(h)$ (this follows immediately from the central limit theorem). Of course, the standard deviation is not known (the standard deviation requires the true mean value $E_D(h)$ which is unknown) but it can be estimated. Let $\hat{\sigma} = \sqrt{(1/m)E_S(h)(1-E_S(h))}$ be this estimate. Note that σ can only be estimated unbiasedly when the m examples are independent and identically distributed. Then $\frac{|E_D(h)-E_S(h)|}{\hat{\sigma}}$ is governed by Student's t distribution with $m-1$ degrees of freedom. Thus, using a table of the t distribution one can easily determine the chance that $E_S(h)$ is off by a threshold ε. As a rule of thumb, with a confidence of 95%, this difference is no more than $1.96\sqrt{\frac{E_S(h)(1-E_S(h))}{m}}$. This estimate, however, is subject to a pessimistic bias because not the whole sample has been used for training and we can expect the learner to do better when we do not hold back a part of the sample.

In order to reduce the variance of the estimate further (*i.e.,* to tighten the confidence interval) and to minimize the pessimistic bias which is caused by not using the hold-out set for training, other methods have been proposed. n-fold cross validation (also known as double cross-validation, or Π-method Mosier, 1951; Mosteller & Tukey, 1968) is conducted as follows. n runs of the learner L are conducted. The sample S is re-sampled into training sets S'_1 through S'_n and hold-out sets S''_1 through S''_n such that the S''_j are disjoint, S'_j and S''_j are disjoint, $S'_j \cup S''_j = S$, and $\bigcup_j S''_j = S$. This process simulates drawing a sample S according to D_{XY}. In each fold j, the hypothesis $L_{H_i}(S'_j)$ is assessed on the hold-out data which yields n hold-out error rates $E_{S''_j}(h_j)$. When n equals m, the method is called leave-one-out, or U-method (Lachenbruch, 1967). Let $E_{CV}(L_{H_i})$ be the averaged hold-out error over all n folds. Of course, we can use Chernoff bounds to bound the difference between $\mathbf{E}_{\{S\}}(E_D(h_L)|L, H_i, m, D_{XY})$ and $E_{CV}(L_{H_i})$, but these bounds would be too loose for practical purposes. Usually, the number of folds n is not large enough to assume that the sum of n random numbers is governed by the normal distribution. Therefore, one usually assumes instead that the empirical error rate measured in each fold is governed by normal distribution (this assumption is reasonable when the sample size m is at least $30 \times n$; when the sample size is large, the binomial distribution converges towards a normal distribution). Then,

$$\frac{\sqrt{n(n-1)}|\mathbf{E}_{\{S\}}(E_D(h_L)|L, H_i, D_{XY}, m) - E_{CV}(L_{H_i})|}{\sqrt{\sum_{j=1}^{n}(E_{S''_j}(h_j) - E_{CV}(L_{H_i}))^2}}$$

is governed by Student's t distribution of degree $n-1$, and we can, again, use a Student's distribution table to determine the confidence of the cross validation error (*i.e.,* the chance that it is off by more

than a certain amount). This, however, requires an estimate of the variance, $\hat{\sigma}$, and it is assumed that the n error estimates are independent. Unfortunately, cross validation does not yield an unbiased estimate of the variance. The n hypotheses are learned from samples which are drawn from one given sample. 90% of the examples in two such samples are identical. Therefore, the estimated variance is an optimistically biased estimate. This bias has been quantified empirically by Dietterich (1997). The bias is even worse when the n hold-out examples are drawn with replacement; this happens with bootstrapping (Efron, 1979, 1983) or when cross-validation is restarted several times (*e.g.,* Kohavi & John, 1995). In this case, several error measurements on the same example are treated as independent estimates.

By conducting arbitrarily many repetitions of cross validation or bootstrapping, it is always possible to obtain an empirical variance which is almost arbitrarily low. This "trick" can be used to "prove" a performance difference between two learners which are really equally good. The biased variance is not a major problem for cross validation based model selection, because the estimated error itself is almost unbiased but it imposes a danger when one wants to study whether an apparent difference is "really there".

Bootstrap experiments are conducted by re-sampling a number of training sets of size m from an original data set of size m by randomly drawing examples *with replacement*. On average, $(1 - 1/e)m \approx .632m$ distinct examples will appear in the training set, and the averaged accuracies on the remaining test sets provide an optimistically biased estimate. The variance is claimed to be lower in many cases than the variance of cross validation (Efron, 1983), but this observation may be due to the a stronger under-estimation of the variances (compared to cross validation).

Cross validation based model selection algorithms use a stratification $\langle H_1, \ldots, H_k \rangle$ and select the model i which minimizes $E_{CV}(L_{H_i})$. An advantage of cross validation is that an almost unbiased estimate of the target is minimized, rather than a (loose) bound. The generalization error of a learning algorithm which uses hold-out testing to decide which model to use can be bounded as follows (Kearns, 1996): With high probability $(1 - \delta)$, the resulting hypothesis h_L incurs at most a generalization error of $E_S(h_L)$, plus a bound on the difference between true and empirical error within the chosen model, plus an additional penalty term which accounts for the possibility of the model selection algorithm choosing a sub-optimal model. The bound on the difference between true and empirical error can be chosen as $O\left(\sqrt{\frac{d_i}{m'} \log \frac{m}{d_i}}\right)$ (according to Vapnik, 1982), and we can use the minimum (over all models H_i) of $E_S(h_L)$ and this bound (d_i is the VC-dimension of model H_i, m' is the training set size and m'' is the hold-out sample size). The additional penalty term for possibly choosing the wrong model index is $O\left(\sqrt{\frac{\log d_k m'}{m''}}\right)$, where d_k is the VC-dimension of the greatest model H_k.

Cross validation based model selection comes with a general but relatively weak guarantee: With probability $1 - \delta$, no other model selection algorithm can be more than $O\left(\sqrt{\frac{\log(m/\delta)}{\gamma m}}\right)$ better than one-fold cross validation (Kearns *et al.*, 1997). In contrast to complexity penalization algorithms which always fail on some problems, cross validation always performs at least reasonably.

Many "practical" learners use cross validation based techniques to select a set of attributes (*e.g.,* among many others, Kohavi, 1995; Kohavi & John, 1995, 1997), or to adapt their regularization parameters (one of many examples is Müller & Wysotzki, 1992). Hold-out testing has been used excessively to assess and adapt the architecture of back-propagation networks (*e.g.,* Koza & Rice, 1991; Musial & Scheffer, 1994). For an overview on cross-validatory error estimation techniques, see (Toussaint, 1974).

2.5 Empirical Methodology of Machine Learning

From the beginning of research on machine learning, there have been attempts to evaluate the performance of generalizers empirically. It is not clear when this beginning was, but one might want to consider Rosenblatt's experiments on the perceptron (Rosenblatt, 1958) as the first experiments on machine learning in the tradition of artificial intelligence (Rosenblatt was studying learning in humans). However, statistical learning emerged much earlier; Fisher (1936) has observed over-fitting effects in regression back in the thirties. Unfortunately, artificial intelligence emerged from computer science which is not a research discipline in which empirical studies (and statistics in general) play a strong role.

Before assessing the performance of some learner, one first has to specify the term "performance" more precisely. When the learner is to be assessed with respect to a particular problem D_{XY} (of which a sample S is given), a natural and commonly used performance criterion is $\mathbf{E}_{\{S\}}(E_D(L_{H_i}(S))|D_{XY}, H_i, m)$ – the expected generalization error (expected over all samples) of learner L (*e.g.,* Wapnik & Tscherwonenkis, 1979; Stone, 1974). However, it should be noted that the expected generalization error rate requires the existence of a fixed distribution over the instances which the learner is exposed to (and which the hypothesis is exposed to once it has been generated). Assuming the existence of such a distribution is not always reasonable.

When the performance is to be assessed not only with respect to a particular learning task, a meaningful criterion is more difficult to define. In order to talk about the expected error (expected over all problems) one has to refer to a prior distribution or density $P(D_{XY})$ over problems. Such a distribution of problems which the learner will be exposed to in the future can usually not be assumed. Also, remember that the No-Free-Lunch Theorems imply that all error minimizing learners are precisely equally good, uniformly averaged over all target functions. However, meaningful questions to study are *"for which targets D_{XY} does some particular technique perform well?"*, or *for which targets D_{XY} does some particular technique L_1 perform better than technique L_2?* Most often, only problems which are archived in certain collections (*e.g.,* Murphy & Aha, 1998; Michie *et al.,* 1994) are studied which imposes a particular bias on the results.

Usually, the performance of learning techniques is assessed by hold-out testing, or n-fold cross validation (see Section 2.4.3). When a learner L_1 incurs a hold-out error which is lower than the hold-out error of learner L_2, this might just be due to chance. In Section 2.4.3, I discussed how we can determine the confidence that the hold-out error is ε-close to the true error. Similar considerations can show whether an apparent performance difference between two learners for some problem is "really there" or just pure chance. Let h_1 and h_2 be the outcome of learners L_1 and L_2 for a given problem. Let us then define Δ as $\Delta = \frac{1}{m}\sum_{(x_i,y_i)\in S}(\ell(h_1(x_i), y_i) - \ell(h_2(x_i), y_i))$. Then, $\frac{\sqrt{m}(E_D(h_1)-E_D(h_2)-\Delta)}{\sigma_\Delta}$ is governed by Student's t distribution. We can use the following procedure to support the claim that L_1 really does better than L_2 (paired t-test).

1. Calculate $T = \frac{\sqrt{m(m-1)}\Delta}{\sqrt{\sum_{i=1}^{m}((E_S(h_1)-E_S(h_2))-\Delta)^2}}$.
2. Use a Student's distribution table to determine the smallest δ such that $t[m-1, \delta] \geq T$.
3. If $p = 1 - \delta$ is below .05 then claim that, with p-value $1 - \delta$, we are better off using L_1 rather than L_2 for this particular problem.

In Chapter 3, I will conduct a set of empirical studies in which I will draw target functions at random under uniform distribution over the set of all Boolean functions. This procedure has two

advantages over the use of benchmark data sets: First, the true error of all hypotheses can be determined exactly (because the target functions are known) and, second, the results support claims on all Boolean functions whereas experiments on benchmark data sets support only claims on the performance of learners for problems which are drawn with respect to the same distribution on problems according to which the benchmark problems have been drawn.

Another methodological problem occurs with learners which have parameters. If several parameter settings of some learner are "tried out", then the lowest measured error is not an unbiased estimate of the true error for that learner and parameter setting. I will discuss this issue in Chapter 5. A more detailed overview on methodological aspects is given by Cohen (1995).

2.6 Summary

- This book is focused on *classification learning*. The task is to minimize $E_D(h_L)$, the zero-one loss of the returned hypothesis of the learner L, while only the empirical error $E_S(h)$ of all hypotheses $h \in H$ can be observed.

- PAC and VC theory support claims on the true error of h_L by bounding the chance (in terms of the sample size) that there is a hypothesis in H_i the empirical error of which is off by more than ε from its true error.

- Loss functions such as the generalization error require the existence of a stationary distribution of instances which the learner is exposed to. By contrast, in the identification in the limit framework, the learner is successful when it can be guaranteed to identify the target function *exactly* but *eventually*. "Exactly" means that no loss function is required but "eventually" means that no bound on the sample size sufficient for learning can be given. This is the principle difference between Valiant's and Gold's frameworks.

- The Bayesian framework exploits knowledge of the prior distribution of target functions, $P(f)$. This requires that the learner is exposed to a stationary distribution of target function, which is also assumed to be known. Under ideal conditions, one can determine the $Bayes$ hypothesis which minimizes the expected loss.

- Occam algorithms have a particular "learning bias" according to which they select between hypotheses that are consistent with the sample. Often, the consistent hypothesis with the least description length in a particular (arbitrarily chosen) coding scheme is preferred. The intuition is that, when the hypothesis comes from a restricted subset of the hypothesis space, one is able to prove better bounds on the largest difference between true and empirical error of any hypothesis in the language. However, this does not improve the expected error of the returned hypothesis and, on average, all learners which return a hypothesis that is consistent with the sample are equally good.

- Complexity penalization algorithms try to reconstruct the learning curve from the empirical error and the complexity of the models. Sometimes, worst-case bounds on the largest difference between empirical and generalization error of any hypothesis in the model are used to construct an assumed learning curve. Since learning curves can differ considerably from each other, no complexity penalization algorithm can perform reasonably well for any model selection problem.

- Cross validation based model selection algorithms stratify the hypothesis language into models $H_1, \ldots, H_k$ and select a model H_i such that the (estimated) expected error of the hypothesis returned by learner L on model H_i is minimized. This error is estimated by recording the average error on hold-out data.

Chapter 3

Expected Error Analysis

Suppose that we have to solve some learning problem for which two possible models are available. Model H_1 contains just one single hypothesis h_1 while model H_2 contains two hypotheses, h_{21} and h_{22}. All three hypotheses incur (unknown) generalization errors. When we draw a sample, each hypothesis exhibits an empirical error which is an *unbiased* estimate of its true error. Unbiased means that the expected empirical error of each hypothesis is just its generalization error. The relation between true and empirical error is known: the empirical error is governed by the binomial distribution with the true error as its mean value. Now suppose that we minimize the empirical error in H_2; let h_2^* be the hypothesis in h_2 with the least empirical error (H_1 contains only one hypothesis, so minimizing the empirical error in H_1 is trivial). When both hypotheses in H_2 incur equal empirical errors, we draw one at random. Unfortunately, the empirical error of h_2^* is not an unbiased estimate of its true error. Why is that? With a chance of (almost) $\frac{1}{2}$, the empirical error of each hypothesis is an optimistic estimate of its true error, and with a chance of (almost)$\frac{1}{2}$ it is a pessimistic estimate. An optimistically assessed hypothesis has a greater chance of being selected as h_2^* than a pessimistically assessed one. In return, the empirical error of h_2^* is, on average, optimistically biased. So, what should we do when the empirical error of h_2^* is less than the empirical error of h_1? The empirical error of h_1 is unbiased while the empirical error of h_2^* (which is lower) is known to be optimistic. Here, the question is how strong this bias is. In this Chapter, I present an answer to this question. We will see that the expected error of the hypothesis which minimizes the empirical error (the ERM hypothesis) in model H_2 depends on the distribution of error values in that model. The distribution of error values of all hypotheses in H_2 contains (at most) two occurring error values; however, the true error rates are unknown. But we can estimate the distribution of true error rates by recording the distribution of empirical error rates in H_2. The distribution of empirical error values in H_2 is, in this case, very simple because only (at most) two distinct values occur which can be observed. When these two error values are known, this suffices to determine an estimate of the expected generalization error of h_2^*. The most interesting aspect of the analysis is that no learning has to be conducted in order to determine the error of the hypothesis that would be the result of learning.

After giving a general overview, I will first present a solution which is based on two independence assumptions but which is rather simple and can be implemented easily (Scheffer & Joachims, 1998a, 1998b). I will then eliminate the stronger of these independence assumptions and come to a very general solution which is slightly more complicated. But I will discuss how the resulting formula can be evaluated efficiently.

In Chapter 2, I defined the model selection problem as consisting of a target D_{XY}, a stratification $\langle H_1, \ldots, H_k \rangle$, a learner L, and a sample size m. In the following, I will assume that the learner is an ERM learner: It minimizes the empirical error, determining $H_i^*(S)$, the set of hypotheses which

minimize the empirical error, and then returns one of these hypotheses, breaking ties by drawing at random under uniform distribution. Most results of this Chapter are based on (Scheffer & Joachims, 1998a, 1998b, 1999a, 1999b).

3.1 Overview on the Framework

First, I will give a brief description of the most important distributions which I will refer to in the following theorems.

D_{XY}, H_i, S, and $H_i^*(S)$: D_{XY} is the unknown distribution on labeled instances, H_i is the focused model, and S the sample of size m (the examples are drawn according to D_{XY}). $H_i^*(S) = \{h \in H_i | E_S(h) = \min_{h' \in H_i}\{E_S(h')\}$ is the set of ERM hypotheses. These apparently best hypotheses only minimize the empirical error rates and must not be confused with hypotheses that minimize the true error rate. Of course, there may be only one or, if several hypotheses have an equally low error, more than one hypothesis in this set. Usually, the random variable h refers to a hypothesis which is drawn uniformly from H_i whereas h_L refers to a hypothesis which is drawn uniformly from $H_i^*(S)$ and is returned by the learner.

$P_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$: Distribution of error values of a hypothesis h_L which is drawn uniformly from the set of ERM hypotheses $H_i^*(S)$ (for a given sample size m). This probability depends on two random variables: The sample (drawn according to the unknown D_{XY}) and h_L (drawn uniformly from $H_i^*(S)$). The model H_i and the sample size m are fixed values. $P_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, m, D_{XY}, h_L \in D_{XY})$ can be read as the chance of drawing a sample S and, subsequently, a hypothesis h_L uniformly from $H_i^*(S)$ with a generalization error of e_D. This distribution is the *error posterior*.

$P_{\{S,h_L\}}(E_D(h_L) > \varepsilon|H_i, m, D_{XY}, h_L \in H_i^*(S))$: Integrating $P_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$ from ε to 1 yields the chance of drawing a sample S and, subsequently, a hypothesis h_L from $H_i^*(S)$ such that the true error of h_L exceeds ε. Note the difference between this chance and the probability of interest in PAC theory $P_{\{S\}}(\exists h : E_D(h) > \varepsilon, E_S(h) = 0|H_i, m)$: The latter denotes the chance of drawing a sample such that *there is* a hypothesis that is consistent with the sample but exceeds a generalization error of ε while $P_{\{S,h_L\}}(E_D(h_L) > \varepsilon|H_i, m, D_{XY}, h_L \in H_i^*(S))$ refers to the chance that the returned hypothesis (which is drawn uniformly from the $H_i^*(S)$) exceeds an error of ε.

$\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$: Expected error of a hypothesis h_L which is drawn uniformly from the set of hypotheses with least empirical error $H_i^*(S)$. From the distribution of error values of h_L the expectation can easily be derived. Cross validation is a straightforward (but expensive) way of estimating this expected value.

$P_{\{h\}}(E_D(h)|H_i, D_{XY})$: Distribution of true error values in the model. I assume that each model contains finitely many hypotheses which induces a distribution of their errors. $P(E_D(h) = e_D|H_i, D_{XY})$ is the chance of picking a hypothesis at random from H_i with a true error of e_D. This distribution is the *error prior*.

$P_{\{S,h\}}(E_S(h)|H_i, m, D_{XY})$ is the distribution of empirical error rates of the hypotheses in the current model. $P_{\{S,h\}}(E_S(h) = e_S|H_i, m, D_{XY})$ is the chance of drawing a sample S and a hypothesis h (uniformly from H_i) such that the empirical error of h is e_S. The relation between $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and $P_{\{S,h\}}(E_S(h)|H_i, m, D_{XY})$ is relatively straightforward:

The chance of a randomly drawn hypothesis incurring a true error of e_D is $P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$ and a hypothesis with true error e_D incurs an empirical error which is governed by the binomial distribution with mean e_D (each new example can be classified properly or be misclassified; the probability of the latter happening is e_D). Hence, $P_{\{S,h\}}(E_S(h) = e_S|H_i, m, D_{XY}) = \int_{e_D} B[e_D, m](e_S)\, dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$.

3.2 Solution for Independent Error Values

In this Section, I discuss a derivation which is based on the additional assumption that the true error rates of the hypotheses considered by the learner are independent random variables. This solution will be generalized by the solution which I will present in Section 3.3. But the solution discussed in this Section is much easier than the more general one and I will refer to both solutions for further results in Chapter 4.

The crucial relation in this analysis is the one between the posterior $P_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$ and the prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. The main claim (given in Theorem 5) is that $P_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$ is a function of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. The high-level intuition of the derivation is the following: by repeatedly applying Bayes' formula one can reduce $P_{\{S,h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S))$ to some computable terms plus three priors. These are $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ (the chance of a randomly drawn hypothesis having a certain error), $P_{\{S,h\}}(E_S(h)|H_i, m, D_{XY})$ (the chance of a randomly drawn hypothesis having a certain empirical error, and $P_{\{S,h\}}(h \in H_i^*(S)|E_S(h) = e, H_i, D_{XY}, m)$ (the chance of a randomly drawn hypothesis with empirical error e being an ERM hypothesis). $P_{\{S,h\}}(E_S(h)|H_i, m, D_{XY})$ can be derived from $P_{\{h\}}(E_D(h)|H_i, D_{XY})$: Each hypothesis h incurs an empirical error which is binomially distributed with mean value $E_D(h)$. Later, I will discuss how $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ can be estimated efficiently from H_i and S which will ground the framework empirically. In order to determine $P_{\{S,h\}}(h \in H_i^*(S)|E_S(h) = e, H_i, D_{XY}, m)$, I make the following consideration: h is in $H_i^*(S)$ if no other h' achieves an error which is strictly less than e. When browsing the model H_i, the learner observes $|H_i|$ error values, these values are distributed according to $P_{\{S,h\}}(E_S(h)|H_i, m, D_{XY})$. Assuming that these values and the corresponding true error values are independent random variables, one can easily calculate the chance of *none* of them having an error of less than e. In doing this, I see the process of exhausting a hypothesis space as a stochastic process in which error values (which are instantiations of a random variable) are observed.

Assumption 1 *The empirical error rates of hypotheses $h \in H_i$ are independent estimates of the corresponding true errors.* $P(E_S(h_1), \ldots, E_S(h_{|H_i|})|H_i, D_{XY}, m, E_D(h_1), \ldots, E_D(h_{|H_i|})) = \prod_{j=1}^{|H_i|} P(E_S(h_j)|H_i, D_{XY}, m, E_D(h_j))$

Assumption 2 *The true error rates of distinct hypotheses are independent random variables.* $P(E_D(h_1), \ldots, E_D(h_{|H_i|})|H_i, D_{XY}) = \prod_{j=1}^{|H_i|} P(E_D(h_j)|H_i, D_{XY})$

Assumption 1 is relatively mild and is often made implicitly; for instance, the calculation of p-values which is required to compare n-fold cross validation results (the p-value gives the chance that one learner does better than another learner for some problem, given the cross validation results) is based on the assumptions that the hold-out errors are independent estimates of the corresponding true error rates.

The intuition of assumption 2 is best captured by considering a counter-example: Let $H_i = \{h_1, h_2\}$ where h_1 maps all instances to 0 and h_2 maps all instances to 1 – *i.e.*, h_1 and h_2 are complementary. Once that $E_D(h_1)$ is known, $E_D(h_2)$ is pinned down to $1 - E_D(h_1)$ (the same holds for E_S). Hence, $P(E_D(h_2)|E_D(h_1)) \neq P(E_D(h_2))$ – *i.e.*, $E_D(h_1)$ and $E_D(h_2)$ are not independent. Most "practical" hypothesis languages such as decision trees (generally, all hypothesis languages which are closed under complement) violates assumption 2 to some degree. However, in Section 3.3 Assumption 2 will be dropped.

Theorem 5 (Scheffer & Joachims, 1998) *Let D_{XY} be a distribution of labeled instances, and let H_i be a finite model. Under Assumptions 1 and 2, the error distribution $P_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$ is a function of $|H_i|$, m, and $P_{\{h\}}(E_D(h)|H_i, D_{XY})$:*

$$P_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, m, D_{XY}, h_L \in H_i^*(S)) \tag{3.1}$$
$$= \frac{\sum_{e_S} B[e_D, m](e_S)(P_{\{S,h\}}(E_S(h) \geq e_S|H_i, m, D_{XY}))^{|H_i|-1} dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{\sum_e (P_{\{S,h\}}(E_S(h) \geq e|H_i, m, D_{XY}))^{|H_i|-1} P_{\{S,h\}}(E_S(h) = e_S|H_i, D_{XY})}$$

where $P_{\{S,h\}}(E_S(h)|H_i, m, D_{XY}) = \int_{e_D} B[e_D, m](e_S) dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$.

The proof of Theorem 5 can be found in Appendix A. As an immediate corollary, the *expected* true error of h_L can be determined.

Corollary 1 *The expected true error of h_L is a function of m,$|H_i|$, and $P_{\{h\}}(E_D(h)|H_i, D_{XY})$:*

$$\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$$
$$= \int_{e_D} e_D dP_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, m, D_{XY}, h_L \in H_i^*(S)) \tag{3.2}$$

Similarly, the chance that h_L exceeds a generalization error of ε can be determined (not just bounded).

Corollary 2 *The chance that h_L exceeds an error of ε is a function of m, $|H_i|$, and $P_{\{h\}}(E_D(h)|H_i, D_{XY})$:*

$$P_{\{S,h_L\}}(E_D(h_L) > \varepsilon|H_i, D_{XY}, m, h_L \in H_i^*(S))$$
$$= 1 - \int_{e_D=0}^{\varepsilon} dP_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, m, D_{XY}, h_L \in H^*(S)) \tag{3.3}$$

A careful implementation of Theorem 5 and Corollary 1 runs in $O(m^2)$ (see Appendix B).

One remaining question is how to get hold of the distribution $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. When this distribution is known we can, according to Theorem 5 and Corollary 2, determine the expected generalization error of h_L and thus conduct model selection. Before discussing this issue, I will present the more general analysis which does not require Assumption 2.

3.3 General Solution

In this Section, I will present an analysis of the expected generalization error of h_L which does not rely upon Assumption 2.

Let us look at a model H_i and a target distribution D_{XY}. The target D_{XY} defines an error $E_D(h)$ for each hypothesis $h \in H_i$. These error values define a distribution of error values in H_i, which I write as $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and which is the "prior" in the analysis. $P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$ is the chance of drawing hypothesis h from H_i (when drawing under uniform distribution) which incurs an error of e_D. (In this Section I study finite models and therefore $P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$ is actually a discrete distribution).

Suppose that H_i contains two hypotheses (just as an easy example). The prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ tells us which error values occur in H_i. There are either two values with a chance of $\frac{1}{2}$ or one value with a chance of 1 (if both hypotheses have equal errors). Let us invent names h_1 and h_2 for the hypotheses and let $E_D(h_1)$ and $E_D(h_2)$ be the two occurring true error values. When a sample S is drawn, the hypotheses will show empirical error values of $E_S(h_1)$ and $E_S(h_2)$, respectively. Now the following happens. On a sample S of size m, each hypothesis incurs an empirical error rate (in the case of two hypotheses $E_S(h_1)$ and $E_S(h_2)$, respectively), governed by the binomial distribution $B[E_D(h_1), m]$ and $B[E_D(h_2), m]$, respectively. (Each example is classified correctly or wrongly, the chance of a wrong answer being $E_D(h_1)$ and $E_D(h_2)$, respectively. This results in a binomial distribution.) Let us now select the hypothesis with the smaller empirical error, call it h_L. In the general case, there might be a set $H_i^*(S)$ of ERM hypotheses and the learner is then assumed to draw a hypothesis h_L at random under uniform distribution from this set. The chance that h_L has a particular error value e_D is now no longer $P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$, because h_L is not a randomly drawn hypothesis. It is, instead, the hypothesis which minimizes the empirical error. The expected true error of h_L is likely to be greater than its empirical error (which is optimistically biased) but less than the error of a randomly drawn hypothesis. Assume that the sample size m is fixed and given *a priori*. By contrast, the sample S itself is a random variable, governed by the distribution $(D_{XY})^m$. This implies that $H_i^*(S)$ is a random variable (as it depends on S) and so is h_L; h_L is drawn randomly from $H_i^*(S)$. This leads us to the posterior distribution $P_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, D_{XY}, m, h_L \in H_i^*(S))$ which is the chance of drawing a sample S (of fixed size m) and, consequently, a hypothesis h_L from $H_i^*(S)$, such that the true error of h_L is e_D. The principle difference between the prior and posterior distribution is that the prior gives the distribution of error rates of hypotheses which are drawn uniformly from H_i, whereas the posterior gives the distribution of error values for hypotheses which have been generated by an error minimization process. The posterior $P_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, D_{XY}, m, h_L \in H_i^*(S))$ immediately leads to the expectation $\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, D_{XY}, m, h_L \in H_i^*(S))$. It is the expected true error of the hypothesis h_L returned by the ERM learner using model H_i and a sample of size m.

The expected true error of h_L, $\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, D_{XY}, m, h_L \in H_i^*(S))$ is quantified in Theorem 6. The crucial part of the proof is how to determine the minimum error e_S and the number of hypotheses $|H_i^*(S)|$ which achieve this error. The idea is that the chance that $H_i^*(S)$ is a particular subset H^* can be determined by factorizing the least error e_S and calculating the chances that each hypothesis in H^* has an empirical error of e_S and each hypothesis outside incurs a strictly greater error. This, however, imposes another difficulty. The empirical error of a hypothesis is distributed binomially, given the true error. But here the chance of *several hypotheses* incurring a certain empirical error e_S has to be determined. Unfortunately, this probability cannot be determined unless the empirical error rates of distinct hypotheses are assumed to be independent estimates of the corresponding true error rates (in reality, the empirical error rates are slightly dependent because they are measured on the same sample). This assumption (Assumption 1) is relatively mild and common in statistics. The more questionable Assumption 2 is no longer necessary.

Theorem 6 (Scheffer & Joachims, 1999) *For a distribution D_{XY} of labeled instances and a finite*

model H_i let $E_D(h)$ be the true error of each hypothesis $h \in H$. Let m be the fixed sample size. Let h_L be a hypothesis which is drawn uniformly from the set $H_i^*(S)$ of hypotheses in H with least empirical error with respect to a sample S, drawn according to $(D_{XY})^m$. Under Assumption 1, the expected error of h_L is

$$\begin{aligned}
&\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|m, H_i, D_{XY}, h_L \in H_i^*(S)) \\
&= \sum_{i=1}^{|H_i|} E_D(h) \sum_{e_S} B[E_D(h), m](e_S) \sum_{n=1}^{|H_i|} \frac{1}{n} \sum_{\substack{H^* \subseteq H\setminus\{h\} \\ |H^*|=n-1}} \qquad (3.4) \\
&\quad \prod_{h^* \in H^*} P_{\{S\}}(E_S(h^*) = e_S | E_D(h^*), m, D_{XY}) \prod_{h \in H\setminus\{h\}\setminus H^*} P_{\{S\}}(E_S(h) > e_S | E_D(h), m, D_{XY})
\end{aligned}$$

The proof can be found in Appendix C. Equation 3.4 can, in principle, be evaluated, given the distribution of true error values $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. Unfortunately, a straightforward evaluation of Equation 3.4 would require a run time which would be exponential in $|H_i|$. Therefore, I will now make an additional technical assumption.

Assumption 3 *I assume that*
$P(|H^*| = n | h_i \in H^*, H_i, m, D_{XY}) = P(|H^*| = n | h_j \in H^*, H_i, m, D_{XY})$.

Assumption 3 means that the chance of the set of hypotheses with least empirical error being of size m when it is known that a hypothesis h_i belongs to this set is not dependent on *which* hypothesis is known to be in this set. This assumption is reasonable in all practical cases as $|H_i|$ grows doubly exponential for Boolean functions and, at least singly exponential for languages such as conjunctions (*i.e.*, it is very large).

Theorem 7 (Scheffer & Joachims, 1999) *Let D_{XY} be a distribution over labeled instances H_i be a finite model. Under Assumptions 1 and 3, the expected error of the hypothesis h_L returned by an ERM learner given an* i.i.d. *sample S drawn according to $(D_{XY})^m$ is*

$$\begin{aligned}
&\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S)) \qquad (3.5) \\
&= \frac{\int_{e_D} e_D P_{\{h\}}(E_D(h) = e_D | H_i, D_{XY}) dP_{\{S\}}(h_{e_D} \in H_i^*(S) | H_i, m, E_D(h_{e_D}))}{\int_{e_D} P_{\{h\}}(E_D(h) = e_D | H_i, D_{XY}) dP_{\{S\}}(h_{e_D} \in H_i^*(S) | H_i, m, E_D(h_{e_D}))}
\end{aligned}$$

where

$$\begin{aligned}
P_{\{S\}}(h_{e_D} \in H_i^*(S) | H_i, m, E_D(h_{e_D})) &= \sum_{e_S} B[e_D, m](e_S) \prod_{e'_D} \left(\sum_{e \geq e_S} B[e'_D, m](e) \right)^{f(e_D, e'_D)} \qquad (3.6) \\
f(e_D, e'_D) &= \begin{cases} |H_i| P_{\{h\}}(E_D(h) = e'_D | H_i, D_{XY}) & \text{iff } e_D \neq e'_D \\ |H_i| P_{\{h\}}(E_D(h) = e'_D | H_i, D_{XY}) - 1 & \text{iff } e_D = e'_D \end{cases}
\end{aligned}$$

and h_{e_D} is an arbitrary hypothesis with true error $E_D(h_{e_D}) = e_D$.

The proof if given in Appendix D. Theorem 7 solves the primary complexity problem by removing the product over all subsets of H_i from Equation 3.4. While a straightforward evaluation of Equation 3.5 would still run in $O(m^4)$, a careful implementation (see Appendix E) runs in $O(m^2)$.

3.4 Estimating $P_{\{h\}}(E_D(h)|H_i, D_{XY})$.

As $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ depends on D_{XY}, it cannot be determined exactly. All information on D_{XY} which we can access is contained in S. We have to find an efficient way of obtaining an estimate of the distribution of error rates based on the sample.

One possible way of estimating this distribution is to measure the empirical counterpart $P_{\{h\}}(E_S(h)|H_i, S)$ and use it as an estimate of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. As m grows, $P_{\{h\}}(E_S(h)|H_i, S)$ converges towards $P_{\{h\}}(E_D(h)|H_i, D_{XY})$; so, for a reasonably large S a good estimate of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ can be obtained. We will see in the experimental section, that small samples impose an optimistic bias that vanishes as the sample size grows. In fact, the experiments reported on in the following sections show that even samples of size 50 allow for reasonably accurate estimates. Note that this is a one-dimensional distribution only; the dimensionality does not increase when H_i grows. How many hypotheses do we have to draw under uniform distribution from H_i in order to obtain a good estimate of this distribution? $P_{\{h\}}(E_S(h)|H_i, D_{XY}, S)$ is a discrete distribution with m individual probabilities. Therefore, if we draw $\frac{1}{\varepsilon}\log\frac{m}{\delta}$ hypotheses, the chance of mis-estimating at least one of them by more than ε is at most δ. This statement, however, is not strong enough because it does not say anything about how strongly a misestimation of the error distribution will influence the quality of the estimate of h_L's generalization error. Let us consider the worst possible case that can occur. Suppose that a hypothesis space contains one single hypothesis with error rate zero, and an exponentially fast growing number of hypotheses with an error rate of one. If we fail to "hit" the extremely good hypothesis, then our estimate of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ will be a single point with mass one. Although this estimate is not far from the true density (which converges towards a single point exponentially fast), it will impose a strong inaccuracy on the estimated error rate of h_L: while the true error rate of h_L is zero with certainty (provided that the learner is exhaustive and finds the isolated good hypothesis in an exponentially fast growing set of bad hypotheses), Theorem 7 will estimate this error as one. Hence, in order to avoid such failures with high confidence, we need to draw an exponentially fast growing number of hypotheses to estimate the distribution of error rates. Many hypothesis languages, however, have a certain property of symmetry which we can exploit to achieve this in linear time. Suppose that we have a decision tree with n leaf nodes and assume that these leafs are unlabeled yet. By assigning combinations of the class labels zero and one to the leafs we can generate 2^n distinct trees which are have equal "stems" and differ in the labelings of their leafs. We can exploit this property of symmetry and construct an algorithm that prints the distribution of the corresponding 2^n empirical error rates in only $O(n)$. For details on the algorithm, see "Algorithm Estimate-$P_{\{h\}}(E_D(h)|H_i, D_{XY})$" in Section 3.8.

Unfortunately, the estimator for $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ discussed above is not unbiased. If, for some model, $P_{\{h\}}(E_D(h) = 0|H_i, D_{XY})$ is zero, the chance of some hypothesis incurring an empirical error of zero is still greater than zero which imposes a bias. Fortunately, though, this bias vanishes when the sample size grows. Two unbiased estimators exist, but both have considerable disadvantages (one estimator has a prohibitive variance and the second relies on a distributional assumption which may easily fail). See (Scheffer & Joachims, 1998b) for a detailed discussion.

When $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is assumed to be a normal distribution, the parameters μ and σ can be determined very easily from an observed $P_{\{h\}}(E_S(h)|H_i, S)$. This assumption holds when the target is a Boolean function over 5 or more attributes and the instances are governed by the uniform distribution. However, experiments on the artificial and the text categorization problem have shown that this assumption fails frequently.

Efficient model selection algorithm. Now Theorem 7 can be implemented into the following algo-

rithm: (a) For all models H_i, record $P_{\{h\}}(E_S(h)|H_i, S)$ by drawing a small number of hypotheses at random under uniform distribution from H_i and measuring the empirical error. (b) Use $P_{\{h\}}(E_S(h)|H_i, S)$ as an estimate of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and use Theorem 7 to determine the expected error of the ERM hypothesis of H_i. (c) After the expected error has been estimated for all models, select the model with the lowest estimated error and (d) invoke the learner with the selected model and the sample S.

3.5 What is Over-Fitting?

A learning curve is a function which maps a model index i to the error of the hypothesis which is generated by a given learner on model H_i for a given learning problem. Often, the learning curve grows for large models which is generally referred to as over-fitting. This is frequently considered to be due to the high hypothesis complexity in models with high indeces. However, over-fitting does not necessarily occur at all. Boosting algorithms have often exhibited a complementary behavior (Schapire *et al.*, 1997), unpruned decision trees have been observed to outperform pruned decision trees (Fisher & Schlimmer, 1988) and Schaffer (1993a) has presented experiments which support his claim that the generalization ability of a learner is a property of the learning problem rather than a property which is intrinsic to the learner.

Using Chernoff bounds, one can guarantee that (with high probability) the difference between true and empirical error of *no* hypothesis in some model H_i exceeds a certain threshold – which immediately leads to worst-case error bounds. This is the way that PAC and VC theory argue. The empirical error is binomially distributed, so even a poor hypothesis has a small chance (depending on the sample size) of exhibiting a low empirical error. When H_i grows, the chance of *some* hypothesis in H_i exhibiting a large difference between true and empirical error grows steeply. Therefore, given two hypotheses with equal empirical error which come from distinct models, PAC theory gives better guarantees for the one which comes from the smaller model. But these guarantees are extremely pessimistic as they rely on the assumption that the learner selects the ERM hypothesis with the greatest true error rate. The expected error analysis implies that when the prior distribution of error rates in the model remains constant, the expected error of the returned hypothesis *converges from above* as H_i grows.

Let us look at the expected error of the returned hypothesis h_L when the model size, $|H_i|$, approaches infinity or, more formally, $\lim_{|H_i|\to\infty} \mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$. For the moment, I assume that all other influential factors, in particular $P_{\{h\}}(E_D(h)|H_i, D_{XY})$, stay constant.

Theorem 8 *When $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is constant with $P_{\{h\}}(E_D(h) = 1|H_i, D_{XY}) < 1$, the expected error of an ERM hypothesis h_L converges as $|H_i|$ grows.*

$$\begin{aligned}&\lim_{|H_i|\to\infty} \mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S)) \\ &= \frac{\int_{e_D} e_D \times (1-e_D)^m dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{\int_{e_D} (1-e_D)^m dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}\end{aligned} \tag{3.7}$$

The proof can be found in Appendix F. Theorem 8 implies that the limit exists which means that the learning curve converges towards a fixed number rather than diverging. Note that the above consideration was subject to the assumption that $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ remains fixed while H_i grows. Let us now study how $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ actually behaves when the target is a Boolean function.

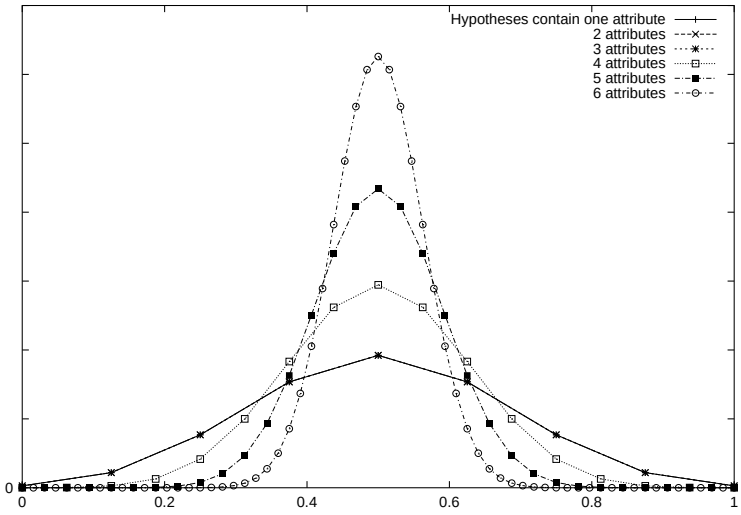

Figure 3.1: Various shapes of $P_{\{h,D_{XY}\}}(E_D(h)|H_i)$ curves for models which contain Boolean attributes $x_1, \ldots x_i$ when the target function requires attributes x_1, x_2, x_3. The distributions are equal in the first three models; models which contain irrelevant attributes incur a smaller ratio of hypotheses with extremal error values which causes a greater expected error for h_L.

Let us now study how $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ actually behaves when we want to learn Boolean functions under uniform distribution of the Boolean instances. In this case, the prior can be determined analytically. When the target function uses attributes x_1 through x_n and the model H_i contains hypotheses over attributes x_1 through x_i, then the prior is a certain binomial distribution depending on i and n (function and hypothesis must agree on all possible $2^{\max\{i,n\}}$ instances which can be distinguished by target function or hypothesis). By plugging the exact prior into Theorem 6 we can determine the learning curve analytically. Appendix G gives the derivation of the expected learning curve (expected for the uniform distribution over Boolean functions) when H_i contains all Boolean functions over i attributes.

Figure 3.1 shows some examples of the distribution $P_{\{D_{XY},h\}}(E_D(h)|H_i)$ (here, the target D_{XY} is a random variable because we draw targets at random) where $P(D_{XY})$ is the uniform distribution over all Boolean functions over attributes x_1, x_2, and x_3 and H_i is the set of functions over attributes x_1 through x_i. H_1, H_2, and H_3 impose equal distributions $P_{\{D_{XY},h\}}(E_D(h)|H_i)$, but from H_4 towards H_6, the tails of the distribution become skinnier – intuitively, the concentration of really good (and really bad) hypotheses decreases and more hypotheses incur an error of close to $\frac{1}{2}$. This causes the learning curve to rise; Figure 3.2 shows the learning curve for Boolean functions over x_1 through x_3. The "simulation" curve shows the error measured in an experiment, averaged over 200 randomly drawn target functions. The slight deviation originates from three sources: The simulation curve is only measured in an experiment and hence subject to some inaccuracy, the independence assumption 1 on the empirical error causes a modest bias, and the simplification (Assumption 3) on which the implementation is based incurs a very small error. However, the learning curve is still predicted very accurately. To my knowledge, this is the first time that a mathematical model of generalization quantitatively predicts the shape of a learning curve. This prediction indicates that the expected error

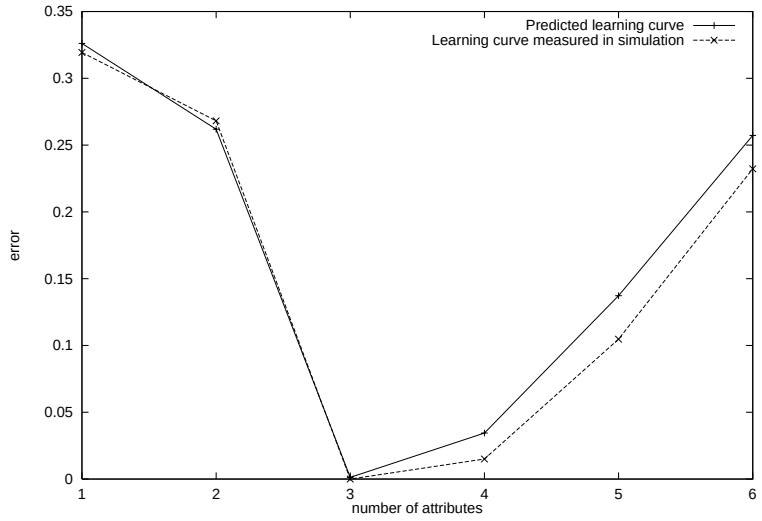

Figure 3.2: Learning curve: Expected generalization error (theoretical values and values measured from simulation) when the target function requires attributes x_1 through x_3 and model H_i (i is on the horizontal axis) uses attributes x_1 through x_i.

analysis provides a good understanding of the nature of over-fitting.

3.6 Robustness against Inaccurate Estimates of $|H_i|$

For virtually all practically relevant hypothesis languages, only bounds on the language size are known (rather than the exact size). Can the expected error analysis be applied in cases in which the exact size of the models is not known? Let us assume that the number of independent hypotheses is over-estimated by a factor of c. What effect will this have on the estimated error of h_L? Figure 3.3 shows how inaccurate the error estimate will be, depending on the model size. While the topmost curve displays the error estimate based on the actual size of H_i, the two lower curves show the estimated error based on a model size which was over-estimated by a factor of 2 and 4, respectively. In order to obtain these curves, I fixed some $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ (using $|H_2|$ from the experiment in Section 3.7.1), and plotted the true error (according to Theorem 7) for various assumed model sizes with a fixed sample size of 50. The generalization error converges exponentially fast (exponential in $\log|H_i|$ *i.e.*, doubly exponential in $|H_i|$) towards a threshold which depends on the sample size and the error prior. As all the curves converge towards this threshold, the difference between them converges to zero exponentially fast. This fast convergence explains why knowing the precise size of the models is not necessary.

3.7 Empirical Studies

In this Section, I will study the bias and variance of the error estimates obtained by the expected error analysis. In particular, I will compare the estimates to estimates obtained by 10-fold cross-validation. In the first set of experiments (Section 3.7.1) I will use a fixed target concept and learn decision trees

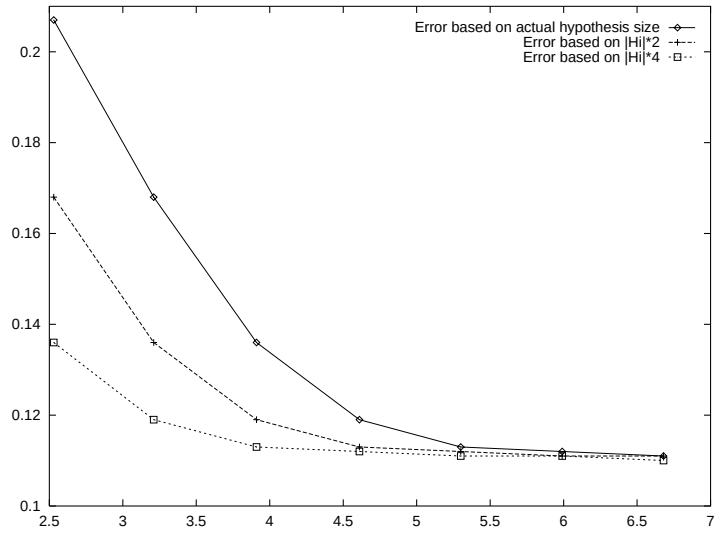

Figure 3.3: How inaccurate will the error estimate be if we mis-estimate the number of independent hypotheses? The topmost curve shows the true error as a function of $|H_i|$ (the horizontal axis is in units of $\log |H_i|$, the vertical axis shows the error); the second curve shows the error when the actual number of independent hypotheses is half the estimated number, the third curve gives the error estimate when the estimated number of hypotheses is four times the true number.

with one continuous and one discrete attribute. In the next set of experiments (Section 3.7.2), I will study Boolean functions. I will run a large number of model selection experiments in which the target functions are randomly drawn Boolean functions. Thus, I will be able to make claims on the performance for all possible Boolean target functions. The controlled experiments of Sections 3.7.1 and 3.7.2 are on a relatively small scale. Therefore, in order to study how well the algorithm scales, I will study a text categorization problem in Section 3.8. In this learning task there are 1000 relevant attributes and 12,000 examples. Applying cross validation to this problem would not be feasible.

3.7.1 Artificial Problem

I designed a model selection problem for which the true error of each hypothesis can be calculated. I chose an instance space with one continuous (x_1) and one discrete ($x_2 \in \{1, 2, 3, 4\}$) parameter. The distribution D_{XY} consists of four overlapping Gaussians (two for each class); the amount of overlap gives rise to a nonzero intrinsic target noise. I chose the hypothesis space such that H_1 consists of all decision trees with only one fixed split and four resulting hypotheses. H_2 contains all binary decision trees with one split of x_1 (discretized into 10 different values, 40 different trees). H_3 entails all 160 binary decision trees of depth 2, and the 2560 hypotheses in H_4 consist of a four-ary split of x_2 and a binary split of x_1.

Results. Table 3.1 shows the true error values, Table 3.2 the predictions of 10-fold cross validation. The predictions for small samples are poor and, generally, the variance of the predictions is quite high. Table 3.3 shows the predictions of the expected error analysis. Again, the predictions for a sample size of 10 are poor, unless another 40 instances are used to estimate the prior more accurately (bracketed values). The accuracy of the error rate estimates is comparable to the accuracy of the 10-fold cross

m	H_1	H_2	H_3	H_4
10	$.42 \pm .11$	$.41 \pm .09$	$.20 \pm .10$	$.26 \pm .10$
50	$.35 \pm .05$	$.35 \pm .02$	$.12 \pm .02$	$.12 \pm .02$
100	$.38 \pm .04$	$.35 \pm .02$	$.10 \pm .00$	$.10 \pm .01$
200	$.34 \pm .00$	$.34 \pm .01$	$.10 \pm .01$	$.10 \pm .01$

Table 3.1: True error rates.

m	H_1	H_2	H_3	H_4
10	$.45 \pm .02$	$.44 \pm .27$	$.18 \pm .11$	$.25 \pm .13$
50	$.43 \pm .02$	$.36 \pm .09$	$.11 \pm .05$	$.13 \pm .06$
100	$.41 \pm .05$	$.35 \pm .04$	$.11 \pm .04$	$.11 \pm .04$
200	$.40 \pm .03$	$.36 \pm .04$	$.10 \pm .05$	$.11 \pm .02$

Table 3.2: 10-fold cross validation error rates.

validation based estimates provided that the prior can be estimated with at least 50 instances). When the sample size is too small for $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ to be estimated accurately, the estimate is expected error analysis based estimate is optimistically biased.

3.7.2 Learning Boolean Decision Trees

The results reported in the previous Section might be due to intrinsic properties of the (fixed) target distribution. Therefore, in this Section, I will report on a set of experiments in which I consider many different target functions, drawn randomly from the space of all Boolean functions. The stratification is such that H_i contains all Boolean decision trees with i Boolean attributes (from a total of 6 possible attributes). I assumed a uniform distribution D_X of instances. Figures 3.4 through 3.6 show some learning curves for Boolean decision trees for sample sizes of 10, 50, and 100, respectively.

For all three figures, the target is a function over three attributes. Each of these figures compares three curves. The first is the average learning curve of a simulation (labeled "simulation") for 200 randomly drawn functions. This curve gives an unbiased estimate of the expected learning curve but is subject to some variance as only 200 runs were conducted. The second (labeled "predicted error, exact prior") is the learning curve predicted by the expected error analysis where the prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ was determined analytically (such a curve has already been presented in Section 3.5 where the error prior for Boolean functions has been determined). The curves show a small pessimistic bias caused by the independence assumption. The third (labeled "predicted

m	H_1	H_2	H_3	H_4
10	$.41 \pm .07$	$.31 \pm .10$	$.18 \pm .11$	$.25 \pm .13$
	$(.43 \pm .02)$	$(.39 \pm .05)$	$(.21 \pm .03)$	$(.26 \pm .02)$
50	$.37 \pm .07$	$.34 \pm .06$	$.11 \pm .04$	$.12 \pm .05$
100	$.39 \pm .04$	$.34 \pm .06$	$.11 \pm .03$	$.10 \pm .03$
200	$.38 \pm .03$	$.31 \pm .03$	$.10 \pm .02$	$.11 \pm .01$

Table 3.3: Expected error analysis based estimates.

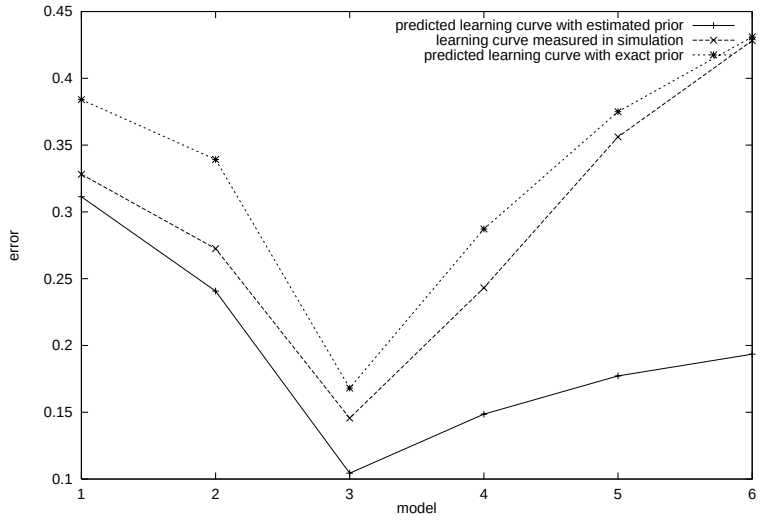

Figure 3.4: Expected learning curves for Boolean functions, 3 attributes are relevant, sample size 10.

error, estimated prior") is the predicted learning curve where the prior distribution on error values $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is estimated by recording $P_{\{h\}}(E_S(h)|H_i, S)$. The difference between the predicted curves (with exact prior) and the curves measured in the simulation is due to the assumption of independent empirical errors and the variance of the error measurement in the simulation. The difference between the predicted curve with exact prior and the predicted curve with estimated prior is due to the identification of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and the observed $P_{\{h\}}(E_S(h)|H_i, S)$. For small sample sizes, the estimate is poor and the predicted learning curve shows a considerable optimistic bias. However, when the sample size grows, the estimated learning curve converges towards the learning curve which is based on the exact error prior. In the following experiments, I will compare the expected error analysis to 10-fold cross validation.

In this experiment, first the depth of the target function is drawn at random under uniform distribution between 2 and 6 within the target model, the target function is drawn uniformly. Algorithm "10-CV" uses 10-fold cross validation to select a model and then minimizes the error within the chosen model using the complete sample. The true error of this returned hypothesis is then averaged over 200 runs (with different, randomly drawn target functions). The expected error analysis based algorithm predicts the error of each model, selects the apparently best model and minimizes the empirical error within that model using the whole sample. Two slightly different versions of expected error analysis based model selection are compared: One version ("EEA"), estimates the error prior by recording the empirical error of all hypotheses in the model. This is computationally as expensive as a run of a learning algorithm (and is therefore approximately ten times faster than 10-fold cross validation). The other version ("EEA-1000"), estimates the error prior by measuring only the empirical error rates of 1000 randomly drawn hypotheses. This can be accomplished in $O(1)$. Figure 3.7 shows the results (Table 3.4 gives the errors and standard deviation numerically); each point is averaged over 200 distinct target functions. The resulting error decreases, of course, with growing sample size. For a sample size of 30, the expected error analysis does significantly better than 10-fold cross validation (p-value is .002); for $m = 40$, cross validation does better than the expected error analysis when the prior is only estimated by drawing 1000 hypotheses. The error rate achieved by expected error anal-

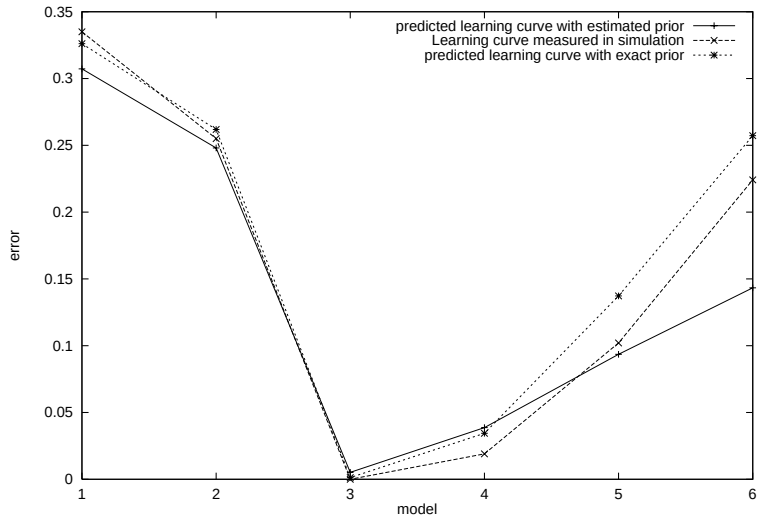

Figure 3.5: Expected learning curves for Boolean functions, 3 attributes are relevant, sample size 50.

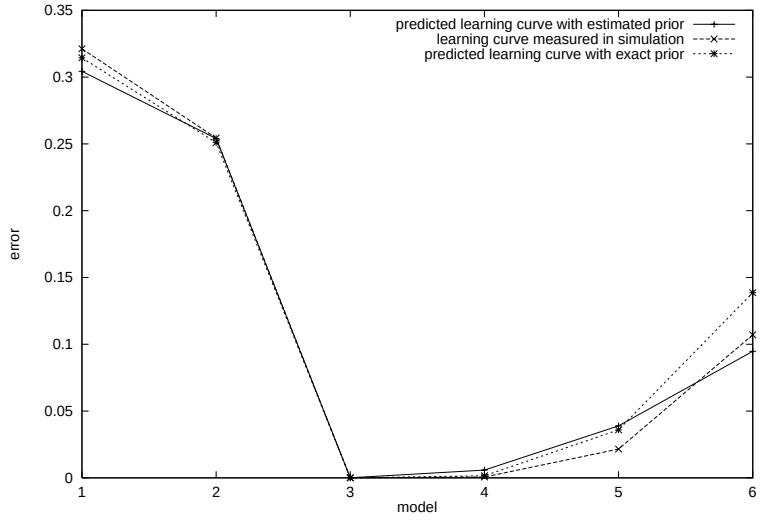

Figure 3.6: Expected learning curves for Boolean functions, 3 attributes are relevant, sample size 100.

m	10	20	30	40	50	100
EEA	$.29 \pm .001$	$.15 \pm .001$	$.07 \pm .001$	$.05 \pm .001$	$.04 \pm .000$	$.006 \pm .000$
EEA-1000	$.28 \pm .001$	$.16 \pm .001$	$.10 \pm .001$	$.09 \pm .001$	$.06 \pm .001$	$.004 \pm .001$
10-CV	$.27 \pm .001$	$.17 \pm .011$	$.10 \pm .001$	$.06 \pm .001$	$.04 \pm .000$	$.004 \pm .000$

Table 3.4: True error of the returned hypotheses; target models drawn uniformly.

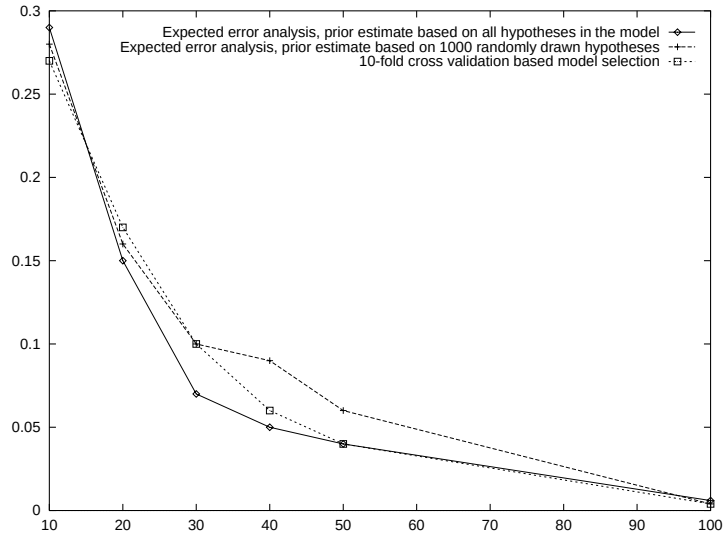

Figure 3.7: True error of returned hypothesis under uniform distribution of target models (horizontal axis: sample size, vertical axis: error); see also Table 3.4.

ysis based model selection is at least as accurate as the error rate obtained by cross validation based model selection. Even though the expected error analysis yields optimistically biased estimates for small samples, the expected error analysis based model selection algorithm performs well. At first blush, this may appear to be surprising. But note that, when the goal is to determine which of several error values is the lowest, it does not harm if *all* error rates are over-estimated by an (almost) constant bias. The principle advantage of expected error analysis, however, is that no learning has to be done, the estimate is obtained in an extremely efficient manner.

3.8 Scaling Up: Text Categorization

In this Section, I will apply the expected error analysis framework to the problem of text categorization. Text categorization (*e.g.,* Salton & Ruckley, 1988) is the problem of mapping texts to semantic categories. Important applications of this are classification of newspaper articles and classification of web pages. In both cases, the large number of available documents makes a manual classification very expensive. But since the problem is difficult and a high quality of the result is important, the task is frequently performed manually. For instance, the Yahoo web index was created and is being maintained manually. Often, documents are represented as sparse attribute vectors, where every word stem that occurs in the document is one feature. When linear classifiers are used, the features are

then weighted with the inverse word frequency (Salton & Ruckley, 1988). Clearly, this representation incurs some loss of information as the resulting classification can only be based on key words. But in many cases, the word ordering is not relevant to decide the category of the text. The induced problem of mapping feature vectors to text categories is still difficult, because the number of attributes is typically greater than thousand and the required sample size can be as much as ten thousand. In this experiment, I used the Reuters corpus (*e.g.,* Joachims, 1998) of 12,902 pre-classified newspaper articles. The data set is split into 9430 training and 3472 hold-out instances. In order to apply the expected error analysis to this problem, several steps have to be carried out.

Selecting a hypothesis language and a stratification. Experiments with decision trees (Joachims, 1998) indicated that C4.5 produced trees which were very imbalanced. In order to exploit this apparent property of the problem, I chose k-decision lists, with k fixed to 2 after some initial experiments (3-DL is infeasible due to the high computational complexity and 1-DL performed outstandingly poor). A k decision list (Rivest, 1987) is a list $\langle b_1 \rightarrow c_1, \ldots, b_l \rightarrow c_l \rangle$ of arbitrarily many rules $b_i \rightarrow c_i$, where b_i is a conjunction of up to k literals and c_i is a class label. The way that a decision list assigns a class label to an instance x is the following. Starting from the beginning of the list, the instance is passed down. The first rule $b_i \rightarrow c_i$ the body b_i of which is satisfied by x "fires" and determines the class label of x. At this point, the instance is not passed down further. When no rule fires, the instance is not assigned a valid class label.

I sorted the attributes according to their information gain (in accordance with the results of Yang & Petersen, 1997). I used two different stratifications. The first stratification consists of four models. These models included all 2-decision lists over the first 250, 500, 750, and 1000 attributes, respectively. The second stratification is $\langle H_1, \ldots, H_7 \rangle$ where H_i contains decision lists the monomials of which cover at least $1, 10, 20, 30, 50, 70,$ and 100 examples, respectively (note that a decision list is a sequence of monomials and each monomial covers some instances). The parameter which is adapted here influences the model by only allowing monomials which are supported by a certain sample size. One could think of this number of examples which have to be covered by each monomial as a pruning threshold which is adapted.

The size of 2-DL(n) is known to be $\log |2 - DL(n)| = O(n^2)$ (Rivest, 1987). I adjusted the multiplicative factor after a few trials in which I compared the predicted generalization error to the hold-out error for some models. In order not to bias the results, I used different models and only one category for this adaptation. The factor influences the predicted error only slightly, so a wide range of factors leads to acceptable results (see Section 3.6).

Defining a learning algorithm. The learner has to minimize the empirical error within the model.The decision list learner of Rivest (1987) runs in $O(n^k \times m)$ (n is the number of attributes, k the number of literals per monomial, and m is the sample size), but after few trials this learner turned out to be too slow for the large given sample. Furthermore, the Rivest algorithm finds a decision list that is consistent with the sample if one exists and fails otherwise. Unfortunately, minimizing the empirical error is much more difficult than finding a consistent hypothesis. The Rivest algorithm is a greedy coverage algorithm which does not back up, once it has committed to a rule; in order to find the minimizing decision list, conducting back-tracking over the rules is necessary. However, back-tracking over the rules is not feasible because this would impose a computational effort of approximately $O((n^2)^{50})$ to $O((n^2)^{100})$ (this is the average length of the observed lists). Therefore, I settled for a greedy algorithm and accepted the drawback of not being guaranteed to find an ERM hypothesis. I used the following modified algorithm which furthermore reduces the cost from $O(n^2 \times m)$ to $O(n^2 \times c)$ where c is a parameter and determines how many different values of empirical error rates the algorithm distinguishes. In order to determine the empirical error rates more efficiently, the algorithm uses an inverse

indexing technique (*i.e.*, there is an array over all words which points at those examples in which the word occurs).

Algorithm DL_1**.** *Input:* sample S, parameters i, p, and c (i is the number of attributes, p is a pruning threshold, and c is a parameter which controls the discretization of empirical error rates). *Output:* 2-DL(i) (approximation of the ERM hypothesis).

1. Start with the empty list $\mathtt{L} = \lambda$.
2. **For** $\mathtt{error\_bound} = 0 \ldots 1$ in steps of $\frac{1}{c}$
   (a) **For** $\mathtt{lit}_1 = 1 \ldots \mathtt{i}$, **For** $\mathtt{lit}_2 = 1 \ldots \mathtt{i}$
      i. **If** the rule $\mathtt{lit}_1, \mathtt{lit}_2 \rightarrow 0$ or the complementary rule $\mathtt{lit}_1, \mathtt{lit}_2 \rightarrow 1$ incurs an empirical error of $\mathtt{error\_bound}$ or less and covers at least p examples **then** commit to the rule, append the rule to L, and remove the covered examples from S.
   (b) (At this point, all instances which can be covered by a rule which cover at least p examples and incur an empirical error of at most $\mathtt{error\_bound}$ have been eliminated from the sample.)
3. (At this point, all instances which can be covered by a rule which cover at least p examples and incur an empirical error of at most 1 – *i.e.*, all instances – have been removed from the sample.)
4. **Return** L.

Estimating $P_{\{h\}}(E_D(h)|H_i, D_{XY})$**.** Straightforward estimation of the prior on error values by drawing c hypotheses would require $O(c \times m \times len(h))$. But since the length of the lists is only bounded on the number of distinct rules (which is outrageous), drawing even a single decision list at random would not be feasible. The latter problem can be solved with the consideration that the prior is estimated by the distribution $P_{\{h\}}(E_S(h)|H_i, S)$ and rules which do not cover at least one instance do not influence the empirical error. The following algorithm runs in $O(c \times m)$ and estimates the prior using up to $c \times 2^m$ randomly drawn hypotheses. This is done by drawing the bodies of c lists of length $\mathtt{len(h)}$ and recording the error values of all $2^{\mathtt{len(h)}}$ possible labelings of class values 0 and 1 to the rules.

Algorithm Estimate-$P_{\{h\}}(E_D(h)|H_i, D_{XY})$**.**

1. **For** $\mathtt{e} = 0 \ldots m$, **Initialize** $\mathtt{P\_E_D[e]}$ to 0. ($\mathtt{P\_E_D[e]}$ will contain the estimate of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$.)
2. **Repeat** c **times:**
   (a) Start with the empty list L and the full sample S.
   (b) **While** S is not empty
      i. Draw an example from S at random and draw two attributes $\mathtt{lit}_1$ and $\mathtt{lit}_2$ which occur in the drawn example.
      ii. **If** the rule $\mathtt{lit}_1, \mathtt{lit}_2 \rightarrow 0$ or $\mathtt{lit}_1, \mathtt{lit}_2 \rightarrow 1$ is covered by at least p examples, add the rule to L and remove the covered examples from S.
   (c) (Now we have a randomly drawn decision list which covers the whole sample; each monomial covers at least p examples.)

(d) **Initialize** newcnt[e] and oldcnt[e] to 0.

(e) (pos(i) and neg(i) refer to the number of positive and negative examples covered by the ith rule of L.)

(f) **Increment** oldcnt[neg(1)] and oldcnt[pos(1)] by 1.

(g) (At this point, oldcnt[e] gives the number of hypotheses which consist of only the body of of the first rule and incur an error of e.)

(h) **For** i $= 2 \ldots$ len(L)

i. **For** e $= 1 \ldots m$

A. **Increment** newcnt[e + pos(i)] by oldcnt[e].

B. **Increment** newcnt[e + neg(i)] by oldcnt[e].

ii. (Now, newcnt[e] gives the number of hypotheses consisting of the bodies of the first i rules with all possible assignments of class labels to the bodies which incur an empirical error of e.)

iii. **For** all e, **set** oldcound[e] to newcount[e].

(i) **For** e $= 0 \ldots m$, **Increment** $\mathtt{P\_E_D}$[e] by newcount[e]$/(2^{\mathtt{len(L)}} \times c)$.

3. **Return** $\mathtt{P\_E_D}$.

The "trick" here is that the algorithm counts the number of hypotheses with the fixed bodies L but all possible combinations of class labels in $O(\mathtt{len(L)})$ although there are $2^{\mathtt{len(L)}}$ such hypotheses. This is done in steps 2(h)iA and 2(h)iB. In step 2i the number of hypotheses is divided by the total number of hypotheses to determine the distribution.

Results. Here, assessing a model by means of the expected error analysis requires approximately 2 minutes on a PC while running even one-fold cross validation for one fixed model takes about 2-3 hours. In order to be able to compare the predicted error rates to hold-out error rates, I had to commit to a relatively small set of models. In a first set of experiments, I wanted to determine how many of the attributes are relevant. I compared 4 models, H_1 through H_4, containing 250, 500, 750, and 1000 attributes, respectively. I measured the hold-out error of the hypotheses returned by the learner on 3472 examples and determined the error rates predicted by Theorem 7. Surprisingly, both learning curves (the curves of the hold-out errors and the predicted errors averaged over the ten most frequent categories) are almost flat except for some noise. The hold-out error is about 0.02 while the predicted error is about 0.05 (the pessimistic bias of about 0.03 has already been observed earlier). When 500 or more attributes are used, the generalization error is significantly lower than when only 250 attributes are used ($p < 0.05$, paired t-test). The predicted error of the model with 500 attributes is lower than the predicted error for 250 models, too. However, there are no significant differences between the error rates for 500, 750, and 1000 attributes ($p > 0.3$, paired t-test). The expected error analysis prefers 1000 attributes but this is neither significantly better nor worse than 500, or 750 attributes. This indicates that (almost) all attributes are relevant but each single attribute carries only little information by itself.

In the next set of experiments, I used the second stratification. The 8 models consist of decision lists the monomials of which cover at least a certain number of examples (*i.e.,* the pruning threshold is adapted). Table 3.5 shows the hold-out error incurred for the ten most frequent concepts, and Table 3.6 shows the error rates predicted by the expected error analysis. Since the hold-out set is quite large (3400 examples), most differences are significant. Figure 3.8 shows the predicted error rates and the error rates estimated by one-fold cross validation, averaged over the ten most frequent categories. The

category	1	10	20	30	50	70	100
acq	.0907258	.0601959	.055011	.0596198	.0561636	.0648041	.0829493
corn	.0034562	.0014400	.0011520	.0011520	.0011520	.0011520	.0011520
earn	.0578917	.0417627	.0345622	.0406106	.0455069	.0486751	.0538594
crude	.0290899	.0129608	.015265	.014400	.0144009	.0172811	.0221774
grain	.0086405	.0089285	.0095046	.0086405	.0080645	.0103687	.0106567
interest	.0241935	.0253456	.0187212	.0207373	.0213134	.0207373	.0207373
money-fx	.0288018	.0218894	.0207373	.0236175	.0250576	.0256336	.0250576
ship	.015841	.0097926	.0086405	.0080645	.0106567	.0092165	.0112327
trade	.0328341	.0213134	.0244816	.0207373	.0239055	.0256336	.0250576
wheat	.0057603	.0011520	.0023041	.0014400	.0023041	.0025921	.0025921
average	.0297	.0204	.01904	.0199	.0208	.0226	.0255

Table 3.5: Hold-out error rates depending on the pruning threshold for 500 attributes.

category	1	10	20	30	50	70	100
acq	.169624	.164426	.136109	.14765	.146808	.157075	.15685
corn	.0254748	.0199941	.0199937	.0198764	.0194109	.0293709	.0285292
earn	.0800091	.0701208	.080318	.0694885	.0787617	.0781107	.0768116
crude	.0500038	.0354245	.0453292	.0426815	.0410724	.0405766	.0400437
grain	.050358	.0486773	.0462126	.0431106	.0408503	.0405403	.050588
interest	.0504822	.0494526	.0436008	.0447569	.0558017	.0403235	.0524012
money-fx	.0678693	.0599967	.0598764	.0694562	.0682563	.0670629	.0659219
ship	.0299725	.0303424	.030023	.0199194	.0195121	.0194726	.0285667
trade	.0541985	.0499002	.0482594	.0428654	.0411051	.0403724	.0401568
wheat	.0200082	.0199976	.0198373	.0199312	.0188736	.0187006	.017471
average	.059797	.054837	.05295	.05196	.053044	.05316	.05573

Table 3.6: Predicted error rates depending on the pruning threshold for 500 attributes.

predicted values are subject to a pessimistic bias of $.03$ (which we already observed in the previous experiments), but the shapes of the learning curves are fairly similar. Note that a constant bias is undesirable when one wants to know just how accurate a particular hypothesis is; but it is harmless when one wants to know which of several hypotheses is the best one. At first blush, model 20 appears to incur a lower averaged hold-out error while model 30 is, on average, predicted to incur a lower generalization error by the expected error analysis. At a closer look, it turns out that, for 8 of 10 categories, the expected error analysis and hold-out testing agree on which of the two models incurs a smaller error. Furthermore, a paired t-test shows that, averaged over all categories, model 20 is unlikely to be superior to model 30 (p=.176) although the differences between the hold-out error rates of these models are significant for the individual categories. These results are very promising; the expected error analysis provides a good estimate of the generalization error. The estimate is obtained extremely efficiently. Assessing a large number of models for this problem would not be feasible by means of hold-out testing, whereas, by the expected error analysis, hundreds of models could be assessed.

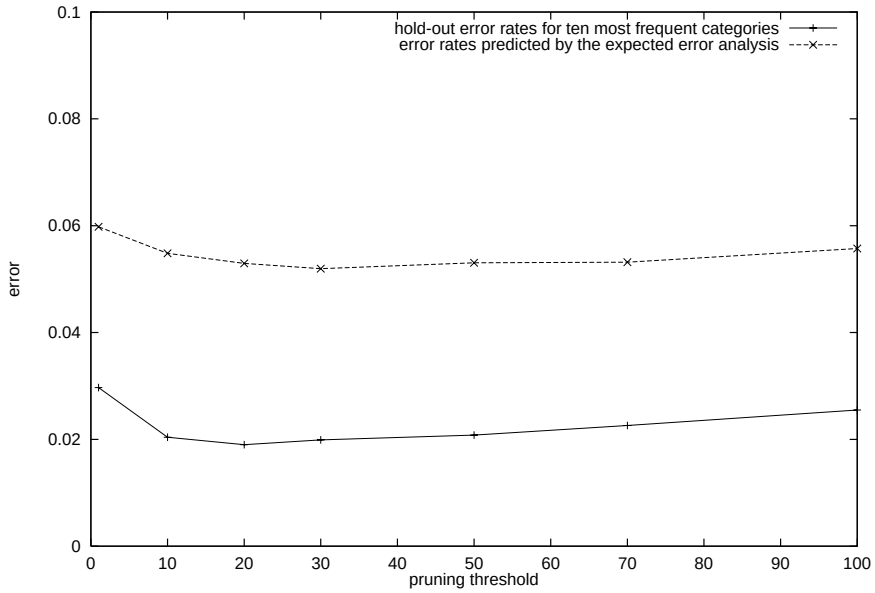

Figure 3.8: Learning curves (measured by one-fold cross validation and the expected error analysis), averaged over the ten most frequent categories.

3.9 Discussion

I showed that the expected true error of an ERM hypothesis can be characterized as a function of the prior distribution of error rates of uniformly drawn hypotheses. This shows that the prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ of a learning problem is extremely elaborate as it (together with the sample size and the model size) contains all information necessary to determine the *exact* probability distribution of the generalization error of h_L – which yields tight error bounds ($P_{\{S,h_L\}}(E_D(h_L) \leq \epsilon) = \int_{e_D=0}^{\epsilon} dP_{\{h_L,S\}}(E_D(h_L) = e_D|m, H_i, D_{XY}, h_L \in H_i^*(S))$) or the expected error $\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, D_{XY}, m, h_L \in H_i^*(S))$ for *a particular learning problem*, rather than PAC or VC-style bounds which hold for the worst possible target distribution. The difference between the classical Bayesian analysis and the expected error analysis should be clearly kept in mind. In the classical Bayesian analysis, the posterior is the chance $P_{\{D_{XY}\}}(D_{XY}|S)$ (*i.e.*, the chance that D_{XY} is the sought target distribution, given the sample S) and the prior is $P(D_{XY})$, the unconditioned chance that D_{XY} is being drawn as target distribution. By contrast, the expected error analysis considers the chance that a hypothesis has a particular error value. In order to estimate the prior $P(D_{XY})$ one would have to observe several learning problems (this is referred to as empirical Bayesian analysis). By contrast, each target distribution D_{XY} induces an individual prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ which can be estimated from one sample.

Learning curves. The presented experiments show that, when the prior distribution of error values $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is known (as is the case when the target is a Boolean function and the instances are uniformly distributed), the analysis predicts and thus explains learning curves with a precision that has not been achieved by PAC theory (*e.g.,* Valiant, 1984), statistical physics (*e.g.,* Tishby, 1995), or other frameworks. One reason for that is that the analysis is not a worst-case anal-

ysis; instead, it considers the distribution of error values which characterizes one particular learning problem, or a distribution of learning problems. The explanation for over-fitting which the expected error analysis provides is substantially different from the explanation provided by the PAC framework. The largest difference between true and empirical error of any hypothesis in the hypothesis language grows necessarily when the hypothesis language grows. Therefore, PAC theory gives weaker guarantees for hypotheses which have been learned from larger models which is sometimes considered to be a justification of the Occam's Razor principle (Blumer *et al.*, 1987). However, learning algorithms have in practice shown patterns of behavior which deviate considerably from the PAC results; for instance, the generalization error of boosting algorithms (Freund & Schapire, 1996) has been observed to continuously decrease when the number of elementary hypotheses (and thus the hypothesis complexity) increases (Schapire *et al.*, 1997; Quinlan, 1996a), similar behavior has been observed for other hypothesis languages as well (Fisher & Schlimmer, 1988; Schaffer, 1993a). The expected error analysis shows that the generalization error primarily depends on the prior distribution of error values in the hypothesis language. When this distribution stays constant while the hypothesis space grows, the generalization error will decrease. But when the variance of this distribution decreases (*i.e.*, the ratio of hypotheses with extremal error values decreases), this causes an elevated generalization error. The variance of the error prior decreases, for instance, when irrelevant attributes are added.

Model selection. When the prior is estimated from the sample, the expected error estimate is poor (optimistically biased) when the sample is too small to estimate the prior distribution well. For sample sizes of 50 or above, the expected error analysis based estimates become reasonably accurate. For large sample sizes, the error estimate has turned out to be pessimistically biased (because we assumed that the empirical errors of distinct hypotheses are independent estimates of the corresponding true error); but the pessimistic bias has been almost at a constant level of 0.03. This makes n-fold cross validation appear preferable when the task is to determine the precise accuracy of some hypothesis. For large samples, cross validation is almost unbiased and comes with (approximate) confidence bounds. However, when the task is to determine which of several hypotheses (learned from distinct models) is better, an almost constant bias is not harmful. Therefore, in the experiments on randomly drawn Boolean function, expected error analysis estimation based model selection has turned out to be at least as accurate as 10-fold cross validation based model selection – even for small samples. However, the expected error analysis based algorithm is much more efficient than cross validation because no learning has to be done. For the text categorization problem with 1000 relevant attributes and 12,000 examples, the estimate of the learning curve was quite accurate and has been obtained in an extremely efficient manner while even one-fold cross validatory assessment of a single model for a single category required hours. Therefore this approach to model selection seems to be particularly interesting for large scale model selection problems (such as text categorization or knowledge discovery in databases) with many hundreds (or thousands) of relevant attributes and many (tens of) thousands of examples. The advantage of the expected error analysis over cross validation increases when the number of attributes and the available sample size increases further.

Limitations. The analysis is subject to two primary restrictions. First, the model size is required to be finite. Decision trees, decision lists, k-DNF, and similar languages lie within the scope of the approach but, as yet, neural networks lie outside. Second, the loss function is restricted to the zero-one loss (or generalization error). I have found a solution for linear cost functions but the corresponding algorithm is too slow for practical purposes. So far, I do not have solutions for further loss functions, such as the quadratic loss. Besides these cases in which the analysis cannot be applied, there are situations in which it should not be applied. Primarily, this is the case when additional background knowledge makes automatic model selection unnecessary. When the prior distribution of target functions or

distributions $P(f)$ is known, choosing any other than the Bayes hypothesis would be sub-optimal. Even when the Bayes hypothesis is computationally intractable, heuristics like the MAP or MDL hypothesis exploit this additional knowledge and promise better results. When the task is to determine the error rate of a particular hypothesis as accurately as possible (as opposed to the model selection task in which one wants to determine which of several hypotheses is the best one) leave-one-out cross validation is able to obtain an estimate with only a small bias while the expected error analysis has often shown a bias of as much as 0.03.

Related results. Many attempts have been made to find an efficient means of assessing models. The potential benefits of analysis of $P_{\{h_L\}}(E_D(h_L)|H_i, D_{XY}, m, L)$ has been discussed by Wolpert (1995). "Bias-variance decompositions" are analyses of the error of hypothesis h_L (returned by learner L) which split the expected loss into two or three intuitively meaningful terms; the actual terms differ slightly: Breiman *et al.* (1984) use the term *bias* for the error rate of the optimal hypothesis of some model whereas the variance term of the error is the error caused by choice of a sub-optimal hypothesis, due to the limitedness of the sample. In the analysis of Kohavi and Wolpert (1996), the intrinsic target noise ("σ^2"), is the expected loss of the Bayes optimal hypothesis; the bias quantifies how well the learner "guesses" the target on average; the variance measures how much the returned hypotheses h_L differ over all possible samples when the learner is invoked repeatedly. Bias-variance decompositions are known for the quadratic loss function (Geman *et al.*, 1992) and the zero-one loss (Kong & Dietterich, 1995; Dietterich & Kong, 1995; Kohavi & Wolpert, 1996). The bias-variance decomposition serves as a tool for the analysis of learning algorithms rather than an efficient means of estimating the error incurred by a learner for a particular problem. Estimating the terms of the decomposition requires to run the learner repeatedly. By contrast, the analysis presented in this Chapter leads to a solution which can be evaluated very efficiently.

Domingos (1998) has found a solution to the expected error of a hypothesis with least empirical error under several additional assumptions. He assumes that all hypotheses in each model H_i have an equal true error (although hypotheses in distinct models may have distinct errors). This additional assumption simplifies the solution considerably which now only depends on the number of hypotheses and the least observed empirical error. The only input to this solution are the empirical error of the returned hypothesis and the number of considered hypotheses. Domingos idea of "process based model selection", as he calls this approach, is to characterize the expected error of the returned hypothesis h_L which has been generated by learning process L which does not necessarily have to be a simple error minimization process. Certainly it would be of considerable practical importance to find solutions for the error of hypotheses which have been generated, for instance, by a greedy algorithm such as C4.5, or by back-propagation. But so far, the only processes for which a solution is known are error minimization among hypotheses with equal true and independent empirical error error (Domingos, 1998), error minimization among hypotheses with independent true and empirical error (Scheffer & Joachims, 1998b) and error minimization among hypotheses with arbitrary true error and independent empirical error values (this chapter).

3.10 Summary

- The expected generalization error of the hypothesis which minimizes the empirical error can be expressed as a function of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$, the distribution of error values of hypotheses in the model.

- An estimate of the error prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ can be obtained by recording

$P_{\{h\}}(E_S(h)|H_i, S, m)$, its empirical counterpart. This estimate is asymptotically consistent. Thus, the resulting generalization errors of several models can be compared and a good model can be chosen efficiently.

- The resulting model selection algorithm is often comparable to 10-fold cross validation in terms of accuracy (sometimes even superior to cross validation as experiments on Boolean decision trees show) but, as experiments on text categorization show, easily scales up to problems with as many as 1000 attributes and 12,000 examples.

- Increasing the size of the hypothesis space does not *per se* cause over-fitting. In fact, growing the hypothesis language while the distribution of error rates stays constant leads to a *decrease* of error. The increase of error referred to as over-fitting occurs when the distribution of error values changes such that the frequency of hypotheses with extremely low (and high) error decreases (*i.e.*, the variance of $P_{\{h\}}(E_D(h)|H_i)$ decreases). This can, for instance, occur when irrelevant attributes are added to the hypothesis language.

Chapter 4

Assumptions that Justify Model Selection

PAC and VC theory quantify worst-case bounds on the difference between true and empirical error of any hypothesis in terms of the size (or the VC dimension) of the hypothesis space and the sample size (*e.g.,* Vapnik & Chervonenkis, 1971; Valiant, 1984; Haussler, 1992). The larger the hypothesis language is, the greater the largest difference between true and empirical error of any hypothesis in this language will be. This motivates "practical" learners to give preference to restricted subspaces of their hypothesis space. Occam algorithms are weak implementations of this idea: From all hypotheses that are consistent with the sample they return the one which is least with respect to a particular ordering (or stratification) of the hypotheses. Thus, Occam algorithms obtain a better estimate of the returned hypothesis' true error, provided that a consistent hypothesis can be found in a model with small index. Most practical learners even accept a higher empirical error if this reduces the complexity of the hypothesis and leads to a better estimated generalization error. Technically, this is done by employing some pruning/regularization techniques (*e.g.*, neural weight decay; Mingers, 1989; Cun *et al.*, 1989), or by conducting cross validation (Stone, 1974; Kohavi & John, 1997). While these model selection approaches usually lead to tighter bounds on the error, they also impose the risk of "missing" some good hypotheses by excluding them from the hypothesis space. Some positive results on the power of model selection are known (*e.g.,* Kearns, 1996; Ng, 1998) but, unfortunately, none of them is strong enough to prove that conducting model selection (as opposed to simply minimizing the empirical error) is actually beneficial. Empirical results (*e.g.,* Kohavi & John, 1997) show that cross validation improves the performance of learners for many of the studied problems, but according to experiments of Schaffer (1993a) and the theoretical analysis of (Wolpert, 1993), this is a property of the particular problems that are studied rather than a property of model selection. In this Chapter, I will study whether at all, or why and when, Occam algorithms and cross validation are beneficial and I will quantify the performance gains, or losses, obtained by cross validation. Most results presented in this Chapter are based on (Hoffmann & Scheffer, 1998).

4.1 Bounds on the Performance of Model Selection

This Section presents results on the limitations of the power of model selection strategies. These results contradict the general belief in the usefulness of model selection that often guides the construction of "practical" learners. The first two claims are based on the first No-Free-Lunch Theorem (see also Section 2.3; Wolpert, 1992). The intuition of this Theorem is that, on off-sample instances, there are just as many concepts which classify the instance as "0" as there are concepts which classify the instance as "1". Therefore, averaged over all possible target concepts, two predictions about

the class of an instance which is not included in the sample are equally bad. This (together with the remaining No-Free-Lunch Theorems) implies that any kind of generalization relies heavily on meta-physical assumptions about the physical reality: It is assumed that some patterns of behavior of "natural" concepts on unobserved samples are less likely than others – which corresponds to a partial alignment between an assumed and the actual prior $P(f)$. As an immediate corollary of Theorem 4, the true errors of hypotheses which are consistent with the sample are equally good or bad, averaged over all target concepts. Corollary 3 rewrites the first claim of the No-Free-Lunch Theorems for model selection algorithms and the generalization error.

Definition 4.1.1 (Least consistent hypothesis) *Let $\langle H_1, \ldots, H_k \rangle$ be a stratification and let S be a sample. Then h is a least consistent hypothesis if h is consistent with S (i.e., $E_S(h) = 0$), H_i is the first model in which h occurs and no H_j, $j < i$, contains a consistent hypothesis.*

Corollary 3 *Let $H_1, \ldots, H_k$ and $H'_1, \ldots, H'_{k'}$ be stratifications. Let h_L and h'_L be least consistent hypotheses with respect to the orderings $\langle H_1, \ldots, H_k \rangle$ and $\langle H'_1, \ldots, H'_{k'} \rangle$, respectively, and let $\bigcup H_i = \bigcup H'_i$. Uniformly averaged over all target concepts $f \in F$, the generalization error rates of h_L and $h_L{}'$ are equal: $\frac{1}{|F|} \sum_f E_{D_X,f}(h_L) = \frac{1}{|F|} \sum_f E_{D_X,f}(h'_L)$.*

Proof. Consistent hypotheses are equally accurate on in-sample instances. The first claim of Theorem 4 explains that any two hypotheses are equally accurate, averaged over all target concepts. ∎

If we choose k' to equal one (the potential hypothesis language is stuck into one single model), Corollary 3 claims that conducting no model selection (but, instead, learning within the whole available hypothesis language) is just as good as any Occam algorithm. While it appears to be a good idea to return a hypothesis which is consistent with the sample and originates from a "restricted subset" (*i.e.*, early in the stratification) of the hypothesis space (because the error estimate is better for small sets of hypotheses), this intuition is misleading – because "restricted" is a completely relative term. Whichever hypotheses we chose to inhabit the models with small indexes – it does not make any difference if no information about the prior $P(f)$ is given. Note also that, in this situation, the description length is not well defined as, without prior, we can chose an arbitrary encoding scheme. This Corollary has a major impact on consistent learning algorithms, such as the Support Vector Machine: The Support Vector Machine uses a stratification which is defined in terms of the between a hyper-plane and positive and negative examples. The VC-dimension in the structural risk minimization framework corresponds to the model index in this Chapter. From all consistent hyper-planes in the inflated feature space, the SVM returns the one which maximizes the margin between positive and negative instances. Corollary 3, however, claims that all consistent hypotheses are just equally good, on average. This implies that the following propositions are equivalent: (a) The SVM is a better-than-average learner, and (b) some concepts (which can be characterized in terms of a hyper-plane in a space of polynomials which has a wide margin between the distinct centers of positive and negative instances) are more frequent in nature than others.

Now, I will study the general case of learners which are not restricted to hypothesis that are consistent with the sample, such as learners which employ cross-validation based pruning-techniques. Theorem 9 claims that, when no prior on target concepts is known, it is better to just minimize the error within the whole hypothesis space.

Theorem 9 (Hoffmann & Scheffer, 1998) *Let L_1 return an arbitrary hypothesis h_{L_1} from H with least empirical error $E_S(h_{L_1})$. Let L_2 be an* arbitrary *learner (e.g., a cross validation based learner) which returns a hypothesis h_{L_2} with $E_S(h_{L_2}) \geq E_S(h_{L_1})$. Uniformly averaged over all target functions f, $\mathbf{E}_{\{f,S\}}(E_D(h_{L_1})|m, L_1) \leq \mathbf{E}_{\{f,S\}}(E_D(h_{L_2})|m, L_2)$.*

Proof. As the off-sample errors of h_{L_1} and h_{L_2} are, on average, equal (follows from Theorem 4), the generalization error of h_{L_1} is at most equal to the true error of h_{L_2} as the on-sample error of h_{L_1} is minimal. The second statement of Theorem 9 follows from Corollary 3. ∎

4.2 Occam Algorithms

Corollary 3 claims that, when no Bayesian prior is known, we can return *any* hypothesis that is consistent with the sample. On the other hand, if the prior *is* known, we can use Bayes rule to find the *Bayes* hypothesis which is guaranteed to have the least generalization error. If, however, there is only partial knowledge on the prior available (this situation is studied as *robust Bayesian learning;* Berger, 1993), I can prove that Occam algorithms will do much better than PAC learners which return arbitrary consistent hypotheses. In fact, it follows from the No-Free-Lunch Theorems that *only* when at least some knowledge on the prior is given, any learner can perform better than an algorithm which returns a random consistent hypothesis. In order to prove this, I need to tweak the definition of PAC learning to include *distributions* over target concepts. The learner is now required to produce, with high probability, a "good guess" of f when f is drawn with respect to a distribution $P(f)$. I do, however, *not* require the learner to find a good hypothesis with high probability for *any* f. The learner is given *some* prior information: In addition to the concept class F (which every PAC-learner is given) the learner is provided with a *class of distributions*, $\mathcal{P}_F$. This class may contain only one element (in this case, we have a Bayesian scenario with known prior) or potentially infinitely many. The learner is then required to perform well with respect to any D_X and any $P(f) \in \mathcal{P}_F$.

Definition 4.2.1 *Let F be a class of target functions and $\mathcal{P}_F$ a class of distributions over F. A DPAC learner L accepts values ε and δ and a sample S. The learner DPAC-learns $(F, \mathcal{P}_F)$ with sample size m iff, for any distribution D_X and any $P(f) \in \mathcal{P}_F$, $P_{\{S,f\}}(E_{D_X,f}(L(S)) > \varepsilon | m, D_X) \leq \delta$ when f is drawn according to $P(f)$ and S according to D_X and f. L DPAC-learns $(F, \mathcal{P}_F)$ polynomially if the sample size m and the runtime are bounded polynomially in n (the size parameter of F),$\frac{1}{\varepsilon}$, and $\frac{1}{\delta}$.*

What is the relationship between PAC and DPAC learning?

Definition 4.2.2 (Valiant, 1984; Haussler, 1988) *Let F be a class of target functions. A PAC learner L accepts values ε and δ and a sample S of size m. L learns F with sample size m iff, for any $f \in F$ and any D_X, $P_{\{S\}}(E_{D_X,f}(L(S)) > \varepsilon | m, D_X, f) \leq \delta$ when S is drawn according to D_X and f. L PAC-learns F polynomially if the runtime of L and the sample size m can be bounded polynomially in n (the size parameter of F), $\frac{1}{\varepsilon}$, and $\frac{1}{\delta}$.*

Theorem 10 (Hoffmann & Scheffer, 1998) *Let L be a PAC learner for a class of target functions F. Then L is a DPAC-Learner for any class of distributions $\mathcal{P}_F$ over functions in F.*

Proof. Any distribution $P(f) \in \mathcal{P}_F$ will produce functions from F. A PAC-learner for F is guaranteed to learn *any* function in F with high probability up to an error of ε.∎

Throughout this Section, I assume that $F \subseteq H$. This implies that there is at least one $h \in H$ which is consistent with the sample S. All the positive results apply when the class of target functions can be stratified into chunks F_i such that the chance of a target function originating from a chunk with small index is higher than the chance of it lying in a chunk with high index. The actual prior of the

concepts in these chunks is not constrained (and need not be known to the learner). All results refer to a learner that produces a *least consistent hypothesis* with respect to the stratification (*i.e.,* a hypothesis that is consistent with the sample and comes from the model with the smallest index that contains a consistent hypothesis).

Theorem 11 (Hoffmann & Scheffer, 1998) *If the stratification $H_1, \ldots, H_u$ is such that $\forall P(f) \in \mathcal{P}_F$, $P_{\{f\}}(f \in H_i) \geq \frac{1}{2^i}$ and $|H_i| \leq k$, a learner which produces a least consistent hypothesis will DPAC-learn $(F, \mathcal{P}_F)$ ($P_{\{S,f\}}(E_{D_X,f}(h_L) > \varepsilon | m, D_X) \leq \delta$) if the sample size is at least $m \geq \frac{1}{\varepsilon} \log \frac{2k}{\delta}$.*

Proof. First, I shall prove that in the situation given in Theorem 11,

$$P(E_{D_X,f}(h_L) > \varepsilon | m, D_X) \leq 2k(1-\varepsilon)^m$$

where h_L is a least consistent hypothesis of H with respect to the stratification and m is the sample size. The chance that there is a hypothesis that is consistent with the sample in the first model H_1 is obviously at most 1; the chance that h_L is from H_2 is at most $\frac{1}{2}$ (because with probability at least $\frac{1}{2}$ the target concept is from H_1 which implies that there is a hypothesis in H_1 which is consistent with the sample). Generally, the chance that H_i (and no earlier model) contains a consistent hypothesis is at most $\frac{1}{2^{i-1}}$.

$$\begin{aligned} P(E_{D_X,f}(h_L) > \varepsilon | m, D_X) &\leq \sum_{i=1}^{u} \frac{1}{2^{i-1}} |H_i| (1-\varepsilon)^m & (4.1) \\ &\leq 2 \max\{|H_i|\}(1-\varepsilon)^m = 2k(1-\varepsilon)^m & (4.2) \end{aligned}$$

This proves that $P(E_{D_X,f}(h_L) > \varepsilon | m, D_X) \leq 2k(1-\varepsilon)^m$. In order to find a sample size bound I have to choose an m such that $P(E_{D_X,f}(h^*) > \varepsilon | m, D_X)$ in Equation 4.2 becomes less than δ. The bound given in Theorem 11 suffices since

$$2k(1-\varepsilon)^{\left(\frac{1}{\varepsilon} \log \frac{2k}{\delta}\right)} \leq \delta.$$

∎

In Theorem 11, the error (and hence the required sample size) is bounded by the size of each model; the number of models (and hence the actual size of the hypothesis space) can be infinite (and even have an infinite VC dimension) and is irrelevant to the error.

Is the decrease in the required sample size due to the stratification or simply due to the fact that the learner is allowed to fail on δ of all target functions (whereas the PAC learner has to work reliably on *all* target functions? Let us consider the following naive learner: We are allowed to incur a high error with a chance of δ, so one could argue that we could ignore $\frac{\delta}{2}$ of the functions and learn the remaining functions with confidence $1 - \frac{\delta}{2}$. Since we know that $P_{\{f\}}(f \in H_i) \geq \frac{1}{2^i}$ we can restrict ourself to learning only the first $\log_2(\frac{2}{\delta}) + 1$ models. This means that the learner picks an arbitrary hypothesis that is consistent with the sample from a set of $k \times (\log_2(\frac{2}{\delta} + 1))$ hypotheses. If the learner is constrained to the first M models, we obtain an error bound of $\frac{1}{2^M} + (k \times M)(1-\varepsilon)^m$ – *i.e.,* the learner incurs an error of more than ε with chance at most $\frac{1}{2^M}$ due to ignoring all functions outside of these models and with chance $(k \times M)(1-\varepsilon)^m$ due to choosing a suboptimal hypothesis (according

to standard PAC theory) which is consistent with the sample. In order to incur an error of at most ε with probability $\frac{\delta}{2}$ we need a sample size of

$$m \geq \frac{1}{\epsilon} \log \frac{2k(\log_2(\frac{2}{\delta}) + 1)}{\delta}$$

Note that the bound of Theorem 11 is lower since the size of the hypothesis space does not depend on δ. By choosing an arbitrarily small δ we can make the implied upper bound on the sample size (Ehrenfeucht *et al.*, 1989) exceed the lower bound of Theorem 11. This shows that at least a significant part of the "saving" in the required sample size is due to the partial alignment between the stratification and the prior. While Theorem 11 assures us good learning when the models are about equally small, I will now focus on a setting in which the models are growing – *i.e.*, the ith model is of size 2^i – but in which there are only finitely many different models. This resembles a typical model selection situation in which increasing the description length of the hypotheses by adding another attribute doubles the cardinality of the hypothesis language but there are only finitely many attributes.

Theorem 12 (Hoffmann & Scheffer, 1998) *If the stratification $H_1, \ldots, H_M$ is such that $\forall P_{\{f\}} \in \mathcal{P}_F \; P_{\{f\}}(H_i) \geq \frac{1}{2^i}$ and $|H_i| \leq k2^i$, a learner which produces a least consistent hypothesis will DPAC-learn $(F, \mathcal{P}_F)$ $P(E_{D_X,f}(h^*) > \varepsilon | m, D_X) \leq \delta$ if $m \geq \frac{1}{\varepsilon} \log \frac{Mk}{\delta}$.*

Proof. I shall first prove that $P(E_D(h^*) > \varepsilon | m, D_X) \leq (M+1)k(1-\varepsilon)^m$. Analogously to the proof of Theorem 11, I argue that the chance of H_i but no H_j, $j < i$, containing a hypothesis that is consistent with the sample is at most $\frac{1}{2^{i-1}}$. Hence,

$$\begin{aligned} P(E_{D_X,f}(h^*) > \varepsilon | m, D_X) &\leq \sum_{i=1}^{M} \frac{1}{2^{i-1}} k 2^{i-1} (1-\varepsilon)^m && (4.3) \\ &= \sum_{i=1}^{M} k(1-\varepsilon)^m && (4.4) \\ &= Mk(1-\varepsilon)^m && (4.5) \end{aligned}$$

A sample size of $m \geq \frac{1}{\varepsilon} \log Mk\delta$ suffices since

$$Mk(1-\varepsilon)^{\left(\frac{1}{\varepsilon} \log \frac{MK}{\delta}\right)} \leq \delta.$$

∎

Finally, I study a setting in which the models grow exponentially and there are infinitely many models. This is the case with, for instance, pattern languages (Angluin, 1980a). The idea here is that the learner only cares about the first $\log_2(\frac{2}{\delta}) + 1$ models. When the target concept does not fall into one of these models (the chance of this happening is $\frac{\delta}{2}$) the learner returns an arbitrary hypothesis. This incurs a certain error which, together with all other potential sources of errors, must not exceed ε with probability $1 - \delta$.

Theorem 13 (Hoffmann & Scheffer, 1998) *If the stratification $H_1, \ldots$ is such that $\forall P_{\{f\}} \in \mathcal{P}_F$ $P_{\{f\}}(H_i) \geq \frac{1}{2^{i-1}}$ and $|H_i| \leq k2^i$, a learner which produces a least consistent hypothesis if there is a hypothesis that is consistent with the sample in the first $\log_2(\frac{2}{\delta})$ models (and an arbitrary hypothesis otherwise) will DPAC-learn (F, dc) if $m \geq \frac{1}{\varepsilon} \log \frac{2\left(\log_2\left(\frac{2}{\delta}\right)+2\right)k}{\delta}$.*

Proof. Intuitively, we learn the concept class of size $(\log_2(\frac{2}{\delta})+2)k$ with confidence $\frac{\delta}{2}$ and cause an error of another $\frac{\delta}{2}$ by ignoring all concepts outside this restricted class (which occur with probability $\frac{\delta}{2}$). Formally, I have to show that the chance of exceeding an error of ε resolves to at most $\frac{\delta}{2}$ when we assign m as specified in Theorem 13 and set M to $\log_2(\frac{\delta}{2})$, and that ignoring functions from models above M causes a large error with probability at most $\frac{\delta}{2}$.

$$P(E_{D_X,c}(h^*) > \varepsilon | m, D_X) \tag{4.6}$$

$$\leq \frac{1}{2^M} + \left(\log_2\left(\frac{2}{\delta}\right)+2\right) k(1-\varepsilon)^{\left(\frac{1}{\varepsilon}\log\frac{2\left(\log_2 ml\left(\frac{2}{\delta}\right)+2\right)k}{\delta}\right)}$$

$$\leq \frac{\delta}{2} + \frac{\delta}{2} \tag{4.7}$$

This completes the proof. ∎

4.3 Cross Validation

The theorems of Section 4.1 claim that when the prior on target function is not in alignment with the stratification, doing anything else than error minimization within the whole hypothesis language is harmful, and Occam algorithms are just as good as any other algorithm. The theorems of Section 4.2 show that when there is a partial alignment between the stratification and the partially known prior, Occam algorithms perform considerably better than error minimization algorithms without such a bias. So far, I have not studied in which cases it can be beneficial to accept a higher empirical error if this can be traded off for a smaller model index. In this Section, I will quantify the expected error of one-fold cross validation based model selection (also referred to as training and test, or hold-out testing) for ERM learners and thus identify cases in which cross validation is beneficial. In the previous Section, I showed that an alignment between the prior $P(f)$ and the preference for a particular hypothesis guarantees good generalization. However, in the general cross validation setting, the target function cannot be guaranteed to be a member of any model (*i.e.*, it may lie outside $H = \bigcup H_i$). It may therefore occur that even a good hypothesis (*i.e.*, one with a low generalization error) has a posterior probability of being the sought target function of zero (because it is inconsistent with the data, and therefore $P_{\{S\}}(S|f) = 0$). Central to the analysis in this Section is the prior distribution of *error values* in model H_i, $P_{\{h\}}(E_D(h)|H_i, D_{XY})$.

First, the sample S is split into training set S' and hold-out set S''. I assume that, while considering model H_i, the learner perceives $|H_i|$ different empirical error values $E_{S'}(h)$, and, among the hypotheses with least empirical errors, $H_i^*(S')$, the learner draws one uniformly (written h_i^*). Analogously to Chapter 3, models are assumed to be finite and the learner is assumed to minimize the empirical error. After determining the hypotheses h_1^* through h_k^* with least empirical error on training Sample S', the learner determines their hold-out errors $E_{S''}(h_i^*)$. The model H_* which incurred the least hold-out error is selected and, within H_*, the empirical error is minimized by the learner using the whole sample; thus, h_L is determined.

Let some arbitrary distribution $D_{XY} = D_{Y|X}D_X$, a training sample size m', hold-out sample size m'' and a stratification $\langle H_1, \ldots, H_k\rangle$ of models H_i be fixed. D_{XY} and H_i together define the distribution $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ of true error values of hypotheses in H_i. D_{XY} also induces a distribution on samples S' and S'', drawn according to D_{XY}. $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and the distribution of samples lead to a distribution of em-

pirical errors $P_{\{S',h\}}(E_{S'}(h)|m', H_i, D_{XY})$ of hypotheses in the model. For a given true error e_D, the corresponding empirical error is binomially distributed; hence, the distribution of empirical errors can be determined by integrating over the true errors: $P_{\{S',h\}}(E_{S'}(h) = e_S|m', H_i, D_{XY}) = \int_{e_D} B[e_D, m'](e_S) dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$. The learner draws a hypothesis h_i^* uniformly from the set of ERM hypotheses $H_i^*(S')$. This together defines a distribution on error rates of the ERM hypothesis which I write as $P_{\{S',h_i^*\}}(E_D(h_i^*)|H_i, D_{XY}, m', h_i^* \in H_i^*(S'))$. The expectation of this error is $\mathbf{E}_{\{S',h_i^*\}}(E_D(h_i^*)|H_i, D_{XY}, m', h_i^* \in H_i^*(S')) = \int_e e \; dP_{\{S',h_i^*\}}(E_D(h_i^*) = e|D_{XY}, H_i, m' h_i^* \in H_i^*(S'))$. At this point, the learner has determined k hypotheses h_i^* (one for each model), the true errors of which are distributed according to $P_{\{S',h_1^*\}}(E_D(h_1^*)|H_i, D_{XY}, m', h_1^* \in H_1^*(S'))$ through $P_{\{S',h_k^*\}}(E_D(h_k^*)|H_i, D_{XY}, m', h_k^* \in H_k^*(S'))$. These k distributions are distinct. The distribution of hold-out examples S'' of size m'' induces a distribution of hold-out errors $P_{\{S',S'',h_i^*\}}(E_{S''}(h_i^*) = e_H|H_i, D_{XY}, m', m'', h_i^* \in H_i^*(S')) = \int_{e_D} B[e_D, m''](e_H) dP_{\{S',h_i^*\}}(E_D(h_i^*) = e_D|H_i, D_{XY}, m', h_i^* \in H_i^*(S'))$. The learner now selects the model H_* the hypothesis h_*^* of which incurred the least hold-out error (breaking ties in favor of small indexes). It returns the hypothesis h_L which minimizes the error on the whole sample S' and S'' within the selected model H_*. This, finally, induces a distribution of true error values of the hypothesis h_L. $P_{\{S',S'',h_1^*,\ldots,h_k^*,h_L\}}(E_D(h_L)|H_1, \ldots, H_k, D_{XY}, m', m'', h_i^* \in H_i^*(S'), h_L \in h_*^*(S' \cup S''))$. The expected error of the finally returned hypothesis h_L is $\mathbf{E}_{\{S',S'',h_1^*,\ldots,h_k^*,h_L\}}(E_D(h_L)|H_1, \ldots, H_k, D_{XY}, m', m'', h_i^* \in H_i^*(S'), h_L \in H_*^*(S' \cup S''))$. I shall now quantify the abovementioned distributions. Theorem 6 (Chapter 3) quantifies the true error of the ERM hypotheses h_i^* and Theorem 14 quantifies the error of the returned hypothesis h_L.

Theorem 14 (Hoffmann & Scheffer, 1998) *The distribution of true error rates of the hypothesis h_L which minimizes the sample error on $S = S' \cup S''$ (of size m' and m'') within model H_* which is selected by hold-out testing (using S' as training and S'' as hold-out set) is a function of the distributions of errors $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ of the models $H_i \in \langle H_1, \ldots, H_k \rangle$, m', m'', and the $|H_i|$ (under Assumptions 1, 2, and 3).*

$$P_{\{S',S'',h_1^*,\ldots,h_k^*,h_L\}}(E_D(h_L) = e_D|H_1, \ldots, H_k, D_{XY}, m', m'', h_i^* \in H_i^*(S'), h_L \in H_*^*(S))$$
$$= \sum_{i=1}^{k} P_{\{S,h_i\}}(E_D(h_i) = e_D|H_i, D_{XY}, m, h_i \in H_i^*(S)) P_{\{S',S'',h_1^*,\ldots,h_k^*\}}(H_i = H_*) \quad (4.8)$$

where m' and m'' are fixed but S' and S'' are random variables, drawn according to $(D_{XY})^{m'}$ and $(D_{XY})^{m''}$, and the $P_{\{S,h_i\}}(E_D(h_i) = e_D|H_i, D_{XY}, m, h_i \in H_i^*(S))$ are distinct distributions for all i, as defined by Theorem 6. Furthermore,

$$P_{\{S',S'',h_1^*,h_k^*\}}(H_* = H_i)$$
$$= \sum_{e_H} P_{\{S',S'',h_i^*\}}(E_{S''}(h_i^*) = e_H|H_1, \ldots, H_k, D_{XY}, m', m'', h_i^* \in H_i^*(S')) \quad (4.9)$$
$$\times \prod_{j<i} P_{\{S',S'',h_j^*\}}(E_{S''}(h_j^*) > e_H|H_i, D_{XY}, m', m'', h_j^* \in H_j^*(S'))$$
$$\times \prod_{j>i} P_{\{S',S'',h_j^*\}}(E_{S''}(h_j^*) \geq e_H|H_i, D_{XY}, m', m'', h_j^* \in H_j^*(S'))$$

where $P_{\{S',S'',h_i^*\}}(E_{S''}(h_i^*) = e_H|H_i, D_{XY}, m', m'', h_i^* \in H_i^*(S')) = \int_{e_D} B[e_D, m''](e_H)$ $dP_{\{S',h_i^*\}}(E_D(h_i^*) = e_D|H_i, D_{XY}, m', h_i^* \in H_i^*(S'))$ and the $P_{\{S',h_i^*\}}(E_D(h_i^*) = e_D|H_i, D_{XY}, m', h_i^* \in H_i^*(S'))$ are given by Theorem 6.

Proof. Equation 4.8 follows from the training and test learning procedure and sais that the true error of h_L is equal to the error of h_i – which minimizes the error on the sample $S' \cup S''$ within model H_i – times the chance that model H_i is selected. Equation 4.9 quantifies the chance of model H_i being selected. This happens when the hold-out error of H_i (learned on sample S' within model H_i) is e_H and the hold-out error of all models j, $j < i$, is strictly greater, and no hold-out error of any model j, $j > i$, is greater than e_H. Equation 4.9 incorporates the assumption that the hold-out errors of the ERM hypotheses h_i^* are independent (*i.e.*, they are depending on their corresponding true error but not on each other. ∎

What is the practical benefit of Theorem 14? For a given learning problem, this Theorem provides answers to the following questions.

1. Given a potential hypothesis language H, how should this language be stratified into k models $\langle H_1, \ldots, \rangle$ such that the expected error rate is minimized? We can obtain an answer by "trying out" different stratifications. For each stratification, we have to estimate the distribution of error values for each model from the sample, and input these distributions into Theorem 14. A decision can be based on the estimates of the generalization error rate of h_L returned by the Theorem.

2. Given a stratification and a learning problem, how should we split the sample into training and hold-out set? A decision can be based on the generalization error rates estimated by Theorem 14 for each split.

3. Given a stratification $\langle H_1, \ldots, H_k \rangle$, should we conduct hold-out testing based model selection or should we rather simply minimize the empirical error rate in the union of models $H_1 \cup \ldots \cup H_k$? By comparing the outcomes of Theorems 7 and 14, we can estimate which option will lead to a lower error rate.

4.4 Case study: Boolean Functions

Theorem 14 quantifies the error rate of a learner which uses training and test based model selection. Unfortunately, the solution is relatively complicated and does not provide simple answers to questions regarding the construction of a good learner. However, when the error priors are known, Theorem 14 predicts the resulting error rate. For a particular learning problem, the error priors can be estimated from data. For certain problems, such as the problem of learning a randomly drawn Boolean function, the error priors can be determined analytically. In this Section, I will study the problem of learning randomly drawn Boolean functions. Here, the model selection task is to "guess" the number of relevant attributes. The problem is very similar to the situation studied in Section 3.5. Each target distribution D_{XY} induces a set of error priors $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ – one for each model H_i. Let $P((p_1, \ldots, p_k)|P(D_{XY}))$ be the chance of observing error priors p_1 through p_k when $P(D_{XY})$ is, in our case, the uniform distribution over all Boolean functions with n attributes.

$$\begin{aligned} &\mathbf{E}_{\{S', S'', h_1^*, \ldots, h_k^*, h_L, D_{XY}\}}(E_D(h_L)|H_1, \ldots, H_k, m', m'', h_i^* \in H_i^*(S'), h_L \in H_*^*(S)) \qquad (4.10) \\ &= \int_{(p_1, \ldots, p_k)} \mathbf{E}_{\{S', S'', h_1^*, \ldots, h_k^*, h_L\}}(E_D(h_L)|H_1, \ldots, H_k, m', m'', (p_1, \ldots, p_k), h_i^* \in H_i^*(S'), h_L \in H_*^*(S)) \\ &\quad dP((p_1, \ldots, p_k)|P(D_{XY})) \end{aligned}$$

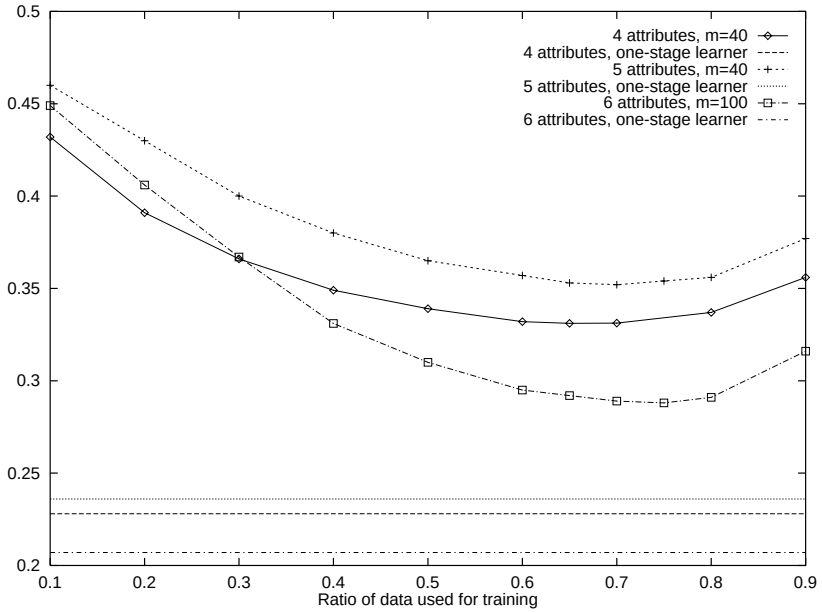

Figure 4.1: Expected error of cross validation based model selection and learning without model selection depending on the training/hold-out split.

For uniformly drawn Boolean functions, $P(p|P(D_{XY}))$ as well as $\mathbf{E}_{\{S,h_L,D_{XY}\}}\left(E_D(h_L)|\, H_i, m, h_L \in H_i^*(S)\right)$ has been determined in Appendix G. By plugging this into Theorem 14, I can now determine $\mathbf{E}_{\{D_{XY},S',S'',h_1^*,\ldots,h_k^*\}}(E_D(h_L) = e_D|m', m'', h_i^* \in H_i^*(S'), h_L \in H_*^*(S))$, the expected error over all possible targets which are distributed according to $P(D_{XY})$. Figure 4.1 shows expected error values for various sample sizes, and training/test splits. As we can clearly see, the optimal training/test split ratio is not fixed but varies with the number of attributes and the sample size (in accordance with the results of Kearns, 1996). We can also see that the error rates of model selection based learning are uniformly greater than the error rates obtained by a learner that uses the whole potentially available hypothesis space. As there are no irrelevant attributes in this setting, this is not surprising. The best that cross validation can do is to select all n attributes – which the learner that does not conduct model selection does in the first place. However, this picture changes when some of the attributes are irrelevant (see Figure 4.2). The error rates of model selection based learning are almost independent of the number of attributes while the error of the learner that does not conduct model selection increases dramatically as the number of irrelevant attributes grows. This demonstrates how well model selection based learners can do in the presence of many irrelevant attributes. Similar observations have been made, for instance, by Ng (1998). Ng gives an error bound for cross validation based model selection (when H_i contains all hypotheses over i attributes) in which the number of irrelevant attributes occurs logarithmically.

4.5 Discussion and Related Results

Model selection techniques such as pruning/regularization (Moody, 1992), cross validation, or structural risk minimization (Vapnik, 1982) are generally considered to lead to generalization errors. This

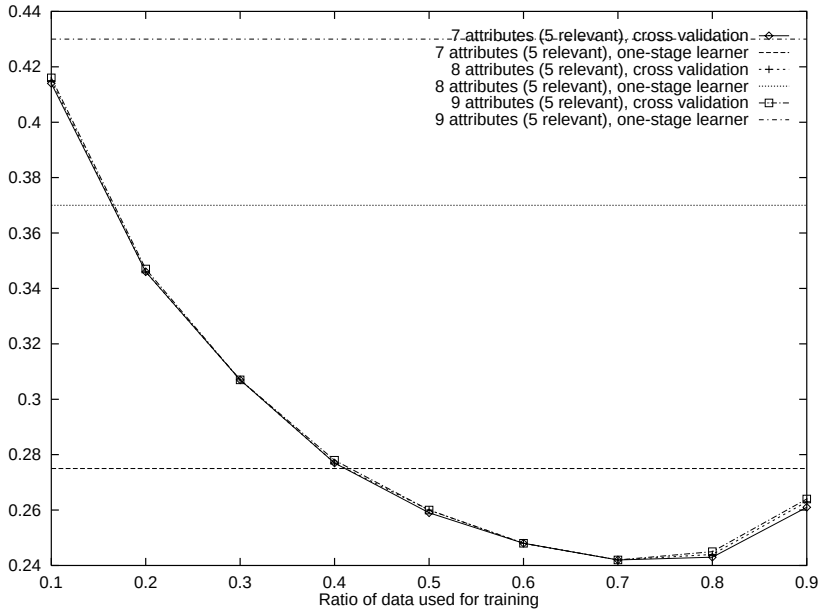

Figure 4.2: Expected error when some of the attributes are irrelevant ($m = 100$).

general belief is based on many empirically supported claims – (*e.g.*, Kohavi & John, 1995; Cun *et al.*, 1989; Mingers, 1989). However, empirical studies by Schaffer (1993a) and a theoretical analysis of Wolpert (1993) contradict this general belief and render model selection as a particular learning bias which will fail for as many problems as it will perform well for. In this Chapter, I picked up on these qualitative results, specified the learning problems for which model selection is an appropriate bias, and quantified the gain which model selection yields in these situations. I characterized distributional assumptions on the target problems which make Occam algorithms and cross validation an appropriate bias and quantified the performance gains, and losses, of Occam algorithms and cross validation. It turns out that Occam algorithms perform much better than PAC learners when *some* knowledge on the prior distribution on target functions is given and is encoded in the stratification. In the more general cross validation setting, the target function is not assumed to be in any model (*i.e.*, even the best hypothesis may not be consistent with the sample). The Bayesian prior $P(f)$ and posterior $P_{\{S\}}(S|f)$ are not helpful in this situation as even a hypothesis which is inconsistent with the sample and is therefore certainly not the sought target function can have a small true error and may be a good solution to the learning problem. However, the prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ on error values in the model is extremely elaborate in this situation. Based on this prior, Theorem 14 quantifies the expected error of one-fold cross validation based model selection and thus compares it to the error of learning without model selection. If the distribution of error values is equal in all models, a learner which does simple error minimization is superior to a learner which conducts cross validation based model selection. If, however, the "ratio of good hypotheses" is higher in few models (as is the case when many of the attributes are irrelevant), model selection performs well. Note that, by contrast, the known positive results on model selection (Kearns, 1996; Kearns *et al.*, 1997; Ng, 1998) do not prove that model selection does better than simple error minimization. The expected error analysis quantifies the expected error of the hypothesis returned by cross validation as a function of the error

distributions in the model. Theorem 14 also makes a claim on the optimal training/test split for a particular problem specified in terms of the distributions $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. The problem of finding an optimal training/test split has been studied intensely. Unfortunately, the answer for the optimal training/test split is not simple. But this is not surprising as previous results showed that the resulting error rate depends on all factors of the learning problem (Kearns, 1996; Müller *et al.*, 1995).

Besides contributing to a better understanding of model selection and its benefits, these results have two practical implications. First, Theorem 14 can be used to decide, in a learning scenario described in terms of the distributions $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ for each H_i, whether one-fold cross validation or learning without model selection is preferable. Second, when a decision has been made for model selection, the theorem can be used to determine the optimal training/test split.

4.6 Summary

- Occam algorithms require an ordering on hypotheses. Among the hypotheses which are consistent with a given sample, they prefer the one which is least with respect to this order. This enables Occam algorithms to give better bounds on the largest difference between true and empirical error of any hypothesis in the model. However, this order can be chosen arbitrarily and, when the order is not aligned with the prior distribution on target functions $P(f)$, all hypotheses that are consistent with the sample incur an equal error on average which renders Occam algorithms useless. Several similar theorems can be proven which provide further evidence of the uselessness of model selection in the absence of additional distributional assumptions.

- When partial knowledge on the distribution on target functions is available (*i.e.*, when $P(f)$ is known to be a member of a particular class), Occam algorithms which give preference to more likely hypotheses perform far better than learners which do not conduct model selection.

- I conducted an analysis of the expected error of the hypothesis which is returned by a learner which conducts one-fold cross validation (also called training and test or hold-out testing). The solution provides the optimal training/test split for a given problem and allows to determine whether, for a given learning problem, one is better off not conducting model selection. The input to the analysis is the set of distributions $P_{\{h\}}(E_D(h)|H_i, D_{XY}, m)$ of error rates of hypotheses in models $H_i \in \langle H_1, \ldots, H_k \rangle$ which can be estimated from the sample or can sometimes be determined analytically.

Chapter 5

Assessment of Learning Algorithms

Empirical assessment of the performance of learning techniques with respect to sets of benchmark problems is a topic that has experienced much attention in the ML community. "Performance" most often translates to expected generalization accuracy. Estimating the generalization error with respect to a particular set of benchmark problems (such as the collection of StatLog data sets or the UCI repository of machine learning data sets; Murphy & Aha, 1998) implies that the assumption is made that the studied learning algorithms are likely to be applied to these (or similar) problems. Remember that the No-Free-Lunch Theorems (see Section 1.2) imply that, under uniform distribution over the target function, two learning algorithms with distinct learning bias perform just equally well. This means that, when learner A does better than learner B for some problem, there has to be another problem for which B does better than A. Empirical studies can help to reveal for which learning problems a particular learning technique is suited.

Learning techniques are usually assessed empirically be means of hold-out testing or n-fold cross validation (Stone, 1974; Toussaint, 1974). I discussed cross validation in Section 2.4.3. When the available data set is too small for the error to be estimated reliably, n-fold cross validation is often used instead of hold-out testing.

Quantifying the performance of the "best of several learners". Virtually any practical learning algorithm possesses a number of parameters (*e.g.*, learning rates, number of learning steps, pruning thresholds, etc.). Selecting values for these parameters is the model selection task, which has to be considered a part of the training process. Unfortunately, sometimes these parameters are adjusted such that the error on either the hold-out set, or, in case of n-fold cross validation, the averaged error on the n hold-out sets, is minimized. Since the error on the test set is used as the quality criterion for the model selection task, the hold-out set influences the training process. Hence, the assumption that the hold-out sets are not used for learning, which is essential to the result that n-fold cross validation is bias-free, is violated. What happens here is very similar to what happens when a learning algorithm selects the hypothesis (from a set of potential hypotheses H) that minimizes the empirical error on a sample. The hypotheses which resulted from some parameter settings will be assessed optimistically while the hypotheses from certain other parameter settings will be assessed pessimistically. The parameter setting which minimized the hold-out error (or cross-validation error) is likely to be an optimistically assessed parameter setting. Taking this hold-out (cross validation) error for an estimate of the learner's expected generalization error would incur an optimistic bias. In this Chapter, I will quantify this bias. This quantification is of practical relevance: If the bias turns out to be negligibly small, then many results which have been obtained with this particular experimental setting are reliable. Otherwise, a more expensive experimental setting which uses nested cross validation

(discussed in Section 5.6) has to be used.

Throughout this Chapter I will assume that the generalization error of a learner with p binary parameters is estimated while the p parameter are adapted at the same time. Note that all results can easily adapted to a learner with p' parameters that can take v values by setting $p = \log_2 v^{p'}$. Note that, although a learner may have continuous parameters, only finitely many settings are tried out.

5.1 Chernoff Bounds

In this Section, I use simple Chernoff bounds to bound the true generalization error of the best of several parameterizations that were assessed on the same hold-out set. This simple result resembles PAC style bounds on the error of the best of several hypotheses.

Theorem 15 *Let learner L possess p binary parameters. Let $p*$ be the parameter setting which leads to the least error on the hold-out set S'' of size m''. The error of the "best of several learners" L_{p*} can be bounded as follows:*

$$P(E_{S''}(L_{p*}(S)) < E_D(L_{p*}(S)) - \varepsilon | m, p, D_{XY}) \leq 2^p \times e^{-2m''\varepsilon^2}. \tag{5.1}$$

Consequently, with probability at least $1-\delta$, the hold-out error can be guaranteed to be "off" by no more than ε – $P(E_{S''}(L_{p*}(S)) < E_D(L_{p*}(S)) - \varepsilon | m, p, D_{XY}) \leq \delta$ – if the hold-out sample is of size at least

$$m'' \geq \frac{1}{2\varepsilon^2}\left(p \log 2 + \log \frac{1}{\delta}\right). \tag{5.2}$$

Proof. Equation 5.1 follows from the Chernoff inequality. Regarding Equation 5.2,

$$\begin{aligned} 2^p \times \exp\left\{-2\frac{1}{2\varepsilon^2}\left(p\log 2 + \log\frac{1}{\delta}\right)\varepsilon^2\right\} &= 2^p \times \exp\left\{-2\left(\frac{1}{2\varepsilon^2}\log\frac{2^p}{\delta}\right)\varepsilon^2\right\} && (5.3)\\ &= 2^p \times \exp\left\{-\log\frac{2^p}{\delta}\right\} && (5.4)\\ &\leq \delta && (5.5) \end{aligned}$$

This completes the proof. ∎

It suffices if the hold-out sample grows linearly in the number of parameters. However, the required hold-out samples are still relatively large for the estimates to be reliable. Consider the following examples.

$$\begin{aligned} \delta = .05, \varepsilon = .05 &\rightarrow m'' \geq p \times 139 + 600\\ \delta = .05, \varepsilon = .02 &\rightarrow m'' \geq p \times 867 + 3745\\ \delta = .05, \varepsilon = .01 &\rightarrow m'' \geq p \times 3466 + 14978\\ \delta = .02, \varepsilon = .05 &\rightarrow m'' \geq p \times 139 + 783\\ \delta = .02, \varepsilon = .02 &\rightarrow m'' \geq p \times 867 + 4890\\ \delta = .02, \varepsilon = .01 &\rightarrow m'' \geq p \times 3466 + 19560 \end{aligned}$$

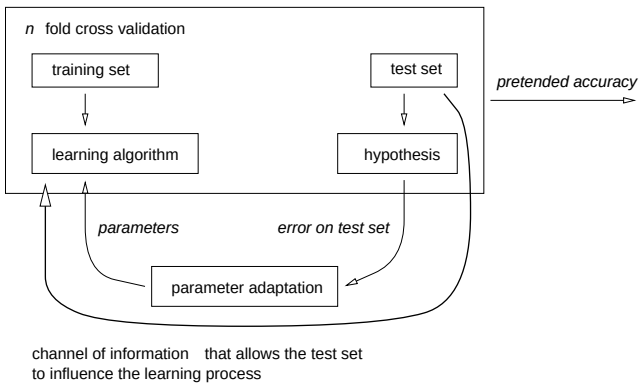

Figure 5.1: If model selection and accuracy estimation are mixed, the parameters for a communication channel, delivering information about the hold-out set to the learning algorithm.

Such large hold-out samples are usually only available for data mining or text categorization problems. These very discouraging results can be criticized for two reasons. First, the results do not cover the advantage of n-fold cross validation over hold-out testing. This difference may be significant; in particular, when the sample is small. Second, Theorem 15 is a worst case result. For a particular given problem, the bias may be considerably smaller. In order to overcome both problems, I will pursue an information theoretic approach in which I will treat the dependencies between the results of distinct cross validation folds explicitly and in which I will refer to properties of the given data set, rather than to the worst case.

5.2 Information-Theoretic Approach

When parameter adaptation is performed on the hold-out sample that is used for error estimation, then the parameters form a communication channel from the hold-out set to the learning algorithm (see Figure 5.1) and the resulting bias depends on the capacity of this channel, *i.e.,* the number of parameter settings, and the number of different parameter settings which are tested.

In the following sections, I will consider different experimental settings: Section 5.3 quantifies the bias of one-shot training and test, Section 5.4 quantifies the bias of n-fold cross validation when different parameter settings are used for each of the n trials, and Section 5.5 is dedicated to cross validation with equal parameter settings for each of the n trials. In Section 5.6, I discuss an experimental setting which yields almost unbiased ranking experiments. The results of this Chapter are based on (Scheffer & Herbrich, 1997).

In this section, I assume the following setting. A learning algorithm L accepts a set of parameters and there are 2^p distinct parameter settings (*i.e.,* the algorithm possesses $log_b(2^p)$ parameters with b possible values each). This set of parameters can be viewed as a communication channel from the parameter optimizer to the learner with a capacity of p bits.

As Figure 5.1 illustrates, when the learner is presented a set of parameters and a training set, it generates a hypothesis h which is used to determine the accuracy of h on the hold-out set of size m. The parameter optimizer is told this accuracy and responds with a new set of parameters, which are again used for training. This cycle is repeated t times, and the best observed accuracy on the hold-

out set is then submitted for publication. Based on the accuracy measured on the hold-out set, the parameter optimizer can send p bits of information on the hold-out set to the learning algorithm. If H is the entropy of the hold-out set (do not confuse entropy H with the hypothesis space which was abbreviated H as well in the preceding chapters) then the capacity of the parameter channel allows to transmit information about the class labels of $c = \frac{p}{H}$ hold-out examples (because H bits are required to encode the class label of one example); *e.g.*, if the hold-out set contains 4 uniformly-distributed class labels, the entropy will be 2, allowing a parameter channel of width 4 (16 distinct settings) to transmit the class labels of $4/2 = 2$ hold-out objects. Intuitively, a parameter channel of p bits accounts for a bias which is as strong as if someone had told the learner the class labels of the first $\frac{p}{H}$ hold-out examples.

However, this knowledge will not actually improve the result by another $\frac{p}{H}$ hits because without this knowledge the learner would not necessarily have failed on all those examples. I will quantify the actual gain in hits on the hold-out examples separately for hold-out testing and n-fold cross validation. First, I will describe a theoretic algorithm that actually does achieve such a difference between true and hold-out error rate.

5.2.1 Parameter Adjustment

The parameter optimizer guesses parameters and obtains the accuracy on the hold-out set in return. I assume that the learning algorithm passes these parameters on to the hypothesis, which uses the information to classify the first $\frac{p}{H}$ hold-out objects that it encounters in some particular way, *i.e.*, the parameters encode class labels for the first $\frac{p}{H}$ hold-out objects. Then the parameter optimizer can use the following strategy:

1. for all objects i, 1 through $\frac{p}{H}$
   (a) for all labels y, 1 through $|Y|$
      – tell the learner to classify the ith example as class y and determine the empirical error on the hold-out set.
   (b) keep the label of i to the class value $y*$ that resulted in the lowest hold-out error.

This algorithm tries the assignment of every possible class label to each of $\frac{p}{H}$ hold-out examples (and leaves the class labels of the remaining examples fixed according to the underlying hypothesis), resulting in a complexity of $\frac{p}{H} \cdot |Y|$ learning trials (where Y is the set of class labels), and finds the parameter setting that encodes correct class labels for $\frac{p}{H}$ samples. We can view this algorithm as greedy search for optimal learning parameters, but since the correct class label of example i is independent of the assigned class label of any other sample, the greedy algorithm will find an optimal assignment after $\frac{p}{H} \cdot |Y|$ trials.[1]

5.3 One-Shot Training and Test

Now, after being told the class labels of the first $\frac{p}{H}$ hold-out objects, how much can this knowledge improve the accuracy on the hold-out set? Assume that there is an initial classifier h, learned by C4.5

[1] Note that this is not a lower bound on the number of trials, as there is an algorithm that needs only $\frac{p}{H}(\frac{3}{2} - \frac{1}{2}\frac{p}{H} + n) - n$ trials in the worst case, but while the algorithm given above essentially performs a gradient search in parameter space, the faster algorithm behaves unlike one would expect a parameter optimizer to behave, so the first result should be somewhat closer to the behavior of a "real" learning algorithm.

(Quinlan, 1993) say, which classifies p_h objects of the hold-out set (of size m'') correctly. Instead of classifying the first $\frac{p}{H}$ instances according to h, we reflect the correct class labels which were determined by the parameter adaptation procedure. We classify the remaining $m'' - \frac{p}{H}$ instances using h. Hence the number of hits is $p_h - x + \frac{p}{H}$, where x is the number of hits that h would have obtained on the first $\frac{p}{H}$ hold-out instances (which are now classified correctly). What is the probability $P(\frac{p}{H} - X \geq z)$ that this procedure increases the number of hits by z? Drawing $c = \frac{p}{H}$ examples from a total of m'', we know that p_h of the m'' are "hits" (classified correctly by h). Hence, the number of hits within the c drawn examples follows a hyper-geometric distribution (note that m'' is finite). The c drawn examples can be replaced with c "hits" (because the parameter optimizer discloses their class labels via the parameter channel).

$$
\begin{aligned}
P(c - X \geq z) &= \sum_{k=z}^{c} P(c - X = k) && (5.6)\\
&= \sum_{k=z}^{c} P(X = c - k) && (5.7)\\
&= \sum_{k=z}^{c} H[m'', p_h, c](c - k) && (5.8)\\
&= \sum_{k=z}^{c} \frac{\binom{p_h}{\frac{p}{H}-k}\binom{m''-p_h}{k}}{\binom{m''}{\frac{p}{H}}} && (5.9)
\end{aligned}
$$

This leads to the bias and computational effort (explained in Section 5.2.1), required to achieve this bias for one-shot training and test. t is the expected number of learning experiments that need to be conducted in order to obtain $p_h + z$ hits on the m'' hold-out examples with probability $P(c - X \geq z)$, when p_h is the true hit rate of h, provided a parameter channel with p bits of capacity is used.

$$
P(c - X \geq z) = \sum_{k=z}^{\frac{p}{H}} \frac{\binom{p_h}{\frac{p}{H}-k}\binom{m-''p_h}{k}}{\binom{m''}{\frac{p}{H}}} \tag{5.10}
$$

$$
t = |Y| \cdot \frac{p}{H} \tag{5.11}
$$

5.3.1 Affected Benchmark Problems

In this Section, as well as in Sections 5.4.1 and 5.5.1, I will quantify the bias on concrete data sets. I assume that one uses a real learning algorithm, C4.5 in most cases, which is "tuned" with additional parameters in order to pretend to outperform the learning algorithm one rank higher than the initial learner in the StatLog performance chart (Michie *et al.*, 1994). I will answer two questions: How many parameters are needed to succeed with probability $> 90\%$ after performing sufficiently many trials and how many trials are needed, on average, to obtain this result, provided the parameter optimizer performs gradient search in the parameter space, which may be an inexact result since it depends on the actual optimizer as well as how strong the parameters influence the result.

Land-sat satellite images: This data set contains 4435 training and 2000 hold-out instances. The default error rate is .231. Based on (Michie *et al.*, 1994), C4.5 is ranked 10th (error .150, *i.e.,* 1700 hits on the hold-out set). To be ranked 9th it would have to outperform Bay-tree (error .147) for which C4.5 needs only $z = 6$ extra hits on the hold-out set. When the class labels of $c = 60$ instances are

known, then $P(c - X \geq z) > 90\%$. Hence we need $t = 6 \cdot 60 = 360$ trials with different parameter settings and since $H = 2.5$ we need a parameter channel of 150 bits. Although an automatic parameter adjustment system may very well run 360 trials, a parameter channel of 150 bits is fairly uncommon (*e.g.*, achieved by 45 parameters with 10 possible values each). Using $c = 16$ examples (parameter channel of 40 bits) and 96 experiments, we still have a 2% chance of being over-ranked.

DNA: This data set, also described in (Michie *et al.*, 1994), possesses 2000 training and 1186 hold-out instances. C4.5 is ranked 10th (1096 hits). To be ranked 9th it would have to outperform INDCart, requiring $z = 4$ extra hits. We need to be told $c = \frac{p}{H} = 85$ class labels to achieve this with 90%: $P(c - X \geq z) > .9$. Since we have 3 classes, we need $t = 3 \cdot 85 = 255$ trials. Since $H = 1.4908$, we need $p = 126$ bit of parameters, *e.g.*, 37 parameters with 10 possible values each.

For both data sets only a very eager scientist may obtain a modest over-ranking of his algorithm by using an incremental algorithm and an automatic parameter-adjustment procedure.

5.4 n-Fold Cross Validation with Parameter Adjustment

In this section I study the bias caused by parameter adaptation when n-fold cross validation is conducted with parameter values chosen differently for the n runs of the learning algorithm. As an example, the number of learning steps is crucial to the performance of back propagation (*e.g.*, Rummelhart & Hinton, 1986). Often, the optimal number of learning steps is determined by observing the error on the hold-out set and selecting the point at which the error rate starts increasing again. The minimum errors that occurred in the n learning curves are then averaged and published, *i.e.*, the number of learning steps may not be fixed to one value within the n folds. Also, this setting is often used when the splits of the training set are explicitly stated (*e.g.*, mesh or vehicle silhouettes).

To achieve an average of z extra hits per fold this way, $n \times z$ has to equal at least $n\frac{p}{H} - \sum_i X_i$, where X_i is the number of hits lost by not using h on $c = \frac{p}{H}$ examples of fold i.

$$P\left(nc - \sum_{i=1}^{n} X_i \geq nz\right) = \sum_{k=nz}^{nc} P\left(nc - \sum_{i=1}^{n} X_i = k\right) \tag{5.12}$$

$$= \sum_{k=nz}^{nc} P\left(\sum_{i=1}^{n} X_i = nc - k\right) \tag{5.13}$$

Unfortunately, there is no explicit formula for the distribution of a sum of hyper-geometric random numbers. However, the probability of a sum of random numbers can be split, considering every possible combination that yields this sum: $P(X + Y = z) = \sum_j P(X = j)P(Y = z - j)$. This equation can be used to decompose the last summand, resulting in the following recursive equation: $P(\sum_{i=1}^{m} X_i = x) = \sum_{j=0}^{x} P(\sum_{i=1}^{m-1} X_i = x - j)P(X_m = j)$. Instantiated to the situation this yields

$$P\left(\sum_{i=1}^{n} X_i = nc - k\right) = \sum_{l=0}^{nc-k} P\left(\sum_{i=1}^{n-1} X_i = nc - k - l\right) P(X_n = l) \tag{5.14}$$

$$= \sum_{l=0}^{nc-k} P\left(\sum_{i=1}^{n-1} X_i = nc - k - l\right) H[m, p_h, c](l) \tag{5.15}$$

$$= \sum_{l=0}^{nc-k} P\left(\sum_{i=1}^{n-1} X_i = nc - k - l\right) \frac{\binom{p_h}{l}\binom{m-p_h}{c-l}}{\binom{m}{c}} \tag{5.16}$$

Straightforward evaluation of this recursive equation for an instantiation is very expensive, since for each evaluation of $P(\sum_{i=1}^{n} X_i = k)$ for every $k' < k$ $P(\sum_{i=1}^{n-1} X_i = k')$ has to be calculated. By iteratively filling an array that is indexed n and k with $P(\sum_{i=1}^{n} X_i = k)$ the formula can be evaluated quickly.

5.4.1 Affected Benchmark Problems

FEM mesh design: (Dolsak & Muggleton, 1992), a relational problem popular in inductive logic programming (*e.g.,* Lavrac & Džeroski, 1994). It is explicitly split into five learning problems. There are 277 examples and 13 classes and the entropy is $H = 2.87$. FOIL (Quinlan, 1990) achieves an accuracy of 21% (59 hits). To achieve 26% accuracy with a probability of 99%, FOIL needs $c = \frac{p}{H} = 5$ class labels, while to achieve 31% with a probability of 93% FOIL would need $c = 8$ class labels. To achieve 26%, we need to conduct 65 trials and a parameter channel of 14 bits, while to achieve 31% we require 104 trials and a parameter channel of 23 bits. The parameter adaptation bias is so strong, that accuracy results can easily be pushed from 21% to 31% on this data set. Although this does *not* prove that the actual accuracy of any learning algorithm *actually is* 10% lower than claimed, it clearly shows, that no accuracy claim is validly empirically supported by such an experiment. Results that were achieved with parameter optimization and different parameter settings in the folds are strongly biased.

Diabetes: There are 768 examples with 2 classes, $H = .9331$. C4.5 achieves 561 hits (rank 13), we need to outperform Quadisc (.5 additional hits per fold). We only need $c = 4$ class labels to succeed with probability 92%, *i.e.,* we need a parameter channel of 3.7 bits and have to conduct 8 trials.

In this experimental setting (parameter optimization in each of the n folds) almost arbitrarily good results are easily achievable. Ranking results achieved with this setting are distorted with high probability.

5.5 n-Fold Cross Validation with Fixed Parameters

While in the last section the parameter optimizer was able to communicate the class labels of the first $\frac{p}{H}$ hold-out objects to the learner, the best the parameter optimizer can do here is to communicate the most frequently observed class label of hold-out object i in all folds for the first $\frac{p}{H}$ objects. If, for example, 3 folds are conducted and in two of them the first presented hold-out object is of class A, the parameter optimizer may tell the learner that the first presented hold-out object is of class A, which results in $2/3$ of an extra hit, averaged over the three folds.

Now what is the probability that the number of extra hits Y gained this way takes value y? I study this problem for two class labels. In this case, for each position in the hold-out set j, $1 \leq j \leq \frac{p}{H}$, I choose the class that the majority of the n examples (drawn from position j of the n folds) belongs to. There will be $\max\{y, n-y\}$ representatives of this class. Assuming that $c = \frac{p}{H}$ is significantly less than m and the probability of choosing y examples is binomially distributed $B[n, p_0](y)$, where p_0 is the probability of the default class, *i.e.,* the most frequent class in the data set[2]. Hence:

$$P(Y = y) = \begin{cases} 0 & y < \frac{n}{2} \\ B[n, p_0](y) + B[n, 1-p_0](y) & y > \frac{n}{2} \\ B[n, p_0](y) & y = \frac{n}{2} \end{cases} \tag{5.17}$$

[2]Without this assumption the probability would be hyper-geometrically distributed, but the number of parameters needed to determine the class labels of a number of examples that gets close to the size of the hold-out set would be outrageous.

The total number of nz additional hits on n folds is the difference between the number of extra hits as explained in the last paragraph and the number of lost hits by not using the hypothesis h calculated in the last section:

$$P\left(\sum_{j=1}^{c} Y_j - \sum_{i=1}^{n} X_i \geq nz\right) = \sum_{k=nz}^{nc} P\left(\sum_{j=1}^{c} Y_j - \sum_{i=1}^{n} X_i = k\right) \quad (5.18)$$

$$= \sum_{k=nz}^{nc} \sum_{l=k}^{nc} P\left(\sum_{j=1}^{c} Y_j = l\right) P\left(\sum_{i=1}^{n} X_i = l - k\right) \quad (5.19)$$

$P(\sum_{j=1}^{c} Y_j = l)$ and $P(\sum_{i=1}^{n} X_i = l - k)$ can be determined as follows:

$$P\left(\sum_{j=1}^{c} Y_j = l\right) = \sum_{r=0}^{l} P\left(\sum_{j=1}^{c-1} Y_j = l - r\right) P(Y_c = l) \quad (5.20)$$

$$P\left(\sum_{i=1}^{n} X_i = l - k\right) = \sum_{q=0}^{l-k} P\left(\sum_{i=1}^{n-1} X_i = l - k - q\right) P(X_n = q) \quad (5.21)$$

$$= \sum_{q=0}^{l-k} P\left(\sum_{i=1}^{n-1} X_i = l - k - q\right) H[m, p_h, c](q) \quad (5.22)$$

$$= \sum_{q=0}^{l-k} P\left(\sum_{i=1}^{n-1} X_i = l - k - q\right) \frac{\binom{p_h}{q}\binom{m-p_h}{c-q}}{\binom{m}{c}} \quad (5.23)$$

Note that in this situation the probability of being ranked too high depends on the hit rate of a default classifier and the hit rate of the initial hypothesis: if the default hit rate is high and the initial classifier performs poorly, the probability of being over-ranked is high.

5.5.1 Affected Benchmark Problems

Diabetes: In this setting, we need $c = \frac{p}{H} = 7$ parameters to achieve .5 extra hits per fold with a probability of 2%. Increasing the number of parameters further decreases the probability, since the expected number of examples with equal class labels at some position in n folds, divided by the number of folds, is small compared to the hit rate of the initial hypothesis. In this experimental setting, diabetes is “safe”.

Heart disease: In this data set (Michie *et al.*, 1994) there are 270 examples, 2 classes, $H = .991$. Since C4.5 performs poorly for this problem we only need $c = 2$ extra hits, a parameter channel of 2 bits and 4 trials are sufficient to succeed with $> 90\%$. If we use k-NN as true learning algorithm (16 hits per fold), we need 6 extra hits to be ranked one rank better. With $c = 9$ examples, we succeed with probability of 90%. This is due to the fact that the default probability is not much worse than our initial hypothesis. A parameter channel of 8 bits is required and 18 trials have to be conducted. There is a strong bias on this data set.

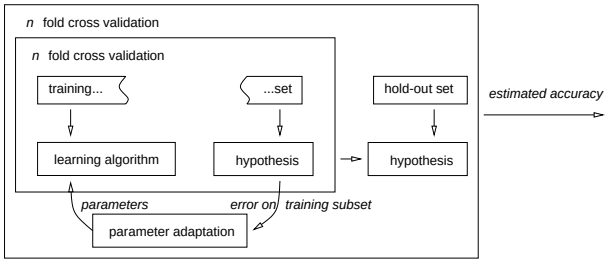

Figure 5.2: n^2-fold cross validation yields an almost unbiased estimate of the resulting generalization error because each outer hold-out set is used for only one single hypothesis.

5.6 Almost Unbiased Assessment

In this Section I want to review how almost unbiased ranking experiments can be conducted. The key is not to confuse parameter adaptation and error estimation. One possible way to obtain an estimate with only a small bias is n-fold triple cross validation (Norman, 1965). In this setting, we have to split the available data into n chunks and repeat the following procedure n times. For all parameter settings which we wish to try out: use $n-2$ chunks to generate a hypothesis and use one chunk (the first hold-out chunk) to obtain an almost unbiased estimate of the generalization error. Then select the hypothesis obtained with the parameter setting that imposed the least error rate on the first hold-out chunk and assess the hypothesis on the second hold-out chunk, the last available chunk. Repeat this procedure n times and average the error rates incurred on the second hold-out chunks. This procedure estimates the generalization error of the hypothesis which is learned when we use hold-out testing to adapt the parameters. It runs in $O(t \times n)$ when we try out t parameter settings. The estimate is only subject to a small pessimistic bias because only $(n-2)/n$ of the training data has been used for learning and $\frac{1}{n}$ of the data has been used for hold-out testing whereas, when we learn and adapt the parameters on the whole data set, we could use slightly larger sets because the second hold-out set is no longer needed. However, when we adapt the parameters using n-fold cross validation we can expect to find better parameter settings than by hold-out testing. The following algorithm which I shall refer to as n^2-fold cross validation (which has, for instance, been applied by Kohavi & John, 1997) obtains an almost unbiased estimate of the expected generalization rate of learner L_{p*} for the given problem and sample size where $p*$ is the parameter setting which minimizes the n-fold cross validation error rate over all parameter settings.

Algorithm n^2-fold cross validation.

1. Split the sample S into n chunks.
2. Repeat n times,
   (a) Let $S_i = S$ minus the ith chunk.
   (b) Split S_i into n chunks and repeat n times
      i. Use $n-1$ chunks of S_i.
      ii. For all possible parameter settings which have to be tried out, invoke the learner, and evaluate the resulting hypothesis on the remaining chunk of S_i which has not been used for training. Thus, determine the parameter setting which leads to the least error on the hold-out chunk.

(c) Use the parameter setting determined by the parameter optimizer and invoke the learner, providing it with S_i.

(d) Determine the ith hold-out error of the resulting hypothesis on $S \setminus S_i$.

3. Return the average of the n measured hold-out error rates.

As Figure 5.2 illustrates, parameter adaptation needs to be performed without considering the hold-out set. In order to get a reliable estimate of the optimal parameter settings, an outer loop of n-fold cross validation evaluates only one hypothesis in each fold. In an inner cross validation loop, the parameters are optimized.

5.7 Discussion

The results presented in this Chapter clearly show that adapting the parameters such that the accuracy on the hold-out set is optimized causes an optimistic bias that depends on the number of parameters and the number of trials. I quantified the bias that will be observed when sufficiently many trials are conducted and the learner makes optimal use of the available parameters, and I calculated the expected number of trials needed, assuming the parameter optimizer follows a gradient descent-like search.

It shows that it is hard to be over-ranked in the one-shot training and test situation if the hold-out set is large. In the n-fold cross validation situation, the probability of being over-ranked is high if the difference between default hit rate and true accuracy is low. If in the n-fold cross validation setting different parameter values are used (*i.e.*, the parameters are optimized locally) a highly over-ranked result can be achieved, so experiments conducted this way do not yield valid results.

These considerations do not prove that any learning algorithm actually *does* perform much worse than claimed, but they do show that such claims are not validly supported by experiments in the naive setting. The results are constructive in some sense: using the provided equations it can easily be proven that in some situations – depending on the properties of the data set – the naive but inexpensive experimental setting yields perfectly valid results.

Based on these results, the validation of performance evaluations for new learning algorithms seems to be difficult: Empirical results that are based on the naive setting are likely to be distorted and cannot be compared to those obtained with unbiased experiments. Heuristic modifications that add new parameters may easily be over-estimated. Even if the modification does not improve the true accuracy of the hypothesis, a new parameter may improve the ranking results.

5.8 Summary

- The averaged hold-out error (or cross validation error) obtained by a particular learner on a sample is a slightly pessimistically biased estimate on the expected error of that learner for the given problem. It is subject to a small pessimistic bias because not the whole sample is used for training when cross validation is conducted. This bias can be minimized by choosing $n = m$ (leave-one-out cross validation).

- However, when a learner is started with several distinct parameter setting, the least observed cross validation error is an optimistically biased estimate of the corresponding learner's performance (just like the training error is an optimistically biased estimate of the returned hypothesis' true error).

- The parameters can be seen as a communication channel between the hold-out set and the learning process. An information theoretic approach can then be used to quantify the optimistic bias of the cross validation error.
- Whether this bias has to be considered depends on several factors, which can be determined from the data set. In some cases, the bias can be neglected – *e.g.,* in one-shot training and test situations when the hold-out sample is large. In the n-fold cross validation setting, the bias becomes considerable when the default error rate is not much higher than the error of the returned hypothesis. When each cross validation fold is allowed to have distinct parameter settings, the bias is extremely strong. Many earlier empirical results appear questionable given these results.
- In order to obtain an unbiased estimate of the generalization error of a learner (the parameters of which have been optimized on the sample) one has to conduct two nested loops of cross validation. The parameters have to be optimized in the inner loop and the error rate of the resulting hypothesis are estimated in the outer loop. In some cases, however, this expensive procedure is not necessary. The results presented here show whether a single loop of cross validation can yield a reliable result.

Chapter 6
Complexity Issues

In model selection based learning, the learner is constrained to a fixed model, determined by the model selection strategy. The complexity of the process of finding a minimizing element in a set of hypotheses has been studied intensely (see, *e.g.*, , Nilsson, 1998). Minimizing the empirical error in some hypothesis language H_i usually requires an effort of $O(|H_i|)$. There are only very few hypothesis languages for which $|H_i|$ is polynomial in the size parameter n (*e.g.*, the number of attributes). This is only the case for shallow decision trees (Dobkin *et al.*, 1997; Auer *et al.*, 1995), or pattern languages with bounded length (Mitchell *et al.*, 1998). Finding a hypothesis that is consistent with the sample in some language H_i (if one exists) is often less difficult. Although the general worst case complexity is $O(|H_i|)$ as well, there are algorithms which run in $O(poly(\log|H_i|))$ for many languages, such as conjunctive concepts (Valiant, 1985), k-DNF (Valiant, 1984), k-DL (Valiant, 1985), or linear threshold units (Blumer *et al.*, 1989).

At first blush, these results seem to show that learning in a large model is easier than learning in a small model which would vindicate the use of model selection based learning algorithms from a complexity point of view. In this Chapter, I will discuss that this is often not the case. I will demonstrate this referring to multilayer perceptrons with a fixed number of hidden units (as an example of model selection based learning) and to boosting. Boosting is a technique of growing hypotheses dynamically by compounding elementary hypotheses by majority voting. I will show that it is primarily the restriction of the hypothesis space to a small model that makes learning difficult. I will support this claim by showing a worst-case time bound of $O(m^2 \log m)$ for the AdaBoost algorithm while solving the same problem with a static hypothesis space is NP-complete. The results of this Chapter are based on (Scheffer & Stephan, 1998).

6.1 Boosting

Suppose that, for some learning problem, we have several strategies available that can be guaranteed to perform just slightly better than randomly guessing. In this situation, majority voting algorithms provide a scheme to combine these weak learners into a powerful system which can achieve arbitrarily high accuracies. The intuition of majority voting is that, for the whole system to give an incorrect answer to some query, at least half of the weak hypotheses have to be wrong for this query. However, as the weak learners perform at least *slightly better* than random guessing, the chance of at least half of them being wrong at the same time vanishes as the number of weak hypotheses grows. AdaBoost (Freund & Schapire, 1996, 1997) is an implementation of this idea into an algorithm which subsequently trains weak hypotheses such that the combined hypothesis can be guaranteed to produce

an arbitrarily low (empirical) error. At stage t, AdaBoost gives high weight to examples which are misclassified by the majority of hypotheses 1 through $t-1$ and trains hypothesis t on the weighted sample. Thus, the overall error incurred by the majority of all hypotheses (error with respect to the sample) drops exponentially fast.

In many empirical studies (many of which are based on the "UCI" repository of machine learning data sets) boosting algorithms (as well as the related Bagging algorithms; Breiman, 1996) have proven to "boost" the generalization accuracy of decision tree learners (Quinlan, 1996a), first order learners (Quinlan, 1996b), and neural networks (Drucker *et al.*, 1993). The empirical success of boosting algorithms can be considered to be due to the particular "geometrical" bias which is induced by majority voting: By adding further weak hypotheses (even after the error on the training sample is zero) boosting maximizes the margin between positive and negative instances in a space which is being "inflated" as new weak hypotheses are added (Schapire *et al.*, 1997). This resembles the learning bias of Support Vector Machines (Vapnik, 1982; Cortes & Vapnik, 1995) or the DIPOL algorithm (Schulmeister & Wysotzki, 1994) or other "wide margin" classifiers. Support Vector Machines stratify the hypothesis space such that the hypotheses are "centered" around the hypothesis which maximizes the margin (in a polynomially inflated instance space) between positive and negative instances. From all hypotheses which are consistent with the sample, the SVM returns the one which is (approximately) closest to this center, as measured in terms of the VC-dimension.

Unfortunately, the analysis of AdaBoost depends on *how much* better than random guessing the weak hypotheses are. Consequently, the known error bounds are in terms of the difference between the accuracy of the weak hypotheses and the accuracy of random guessing. In this paper, I present a *worst-case* analysis for AdaBoost when the weak hypotheses are perceptrons or decision trees. The main result is that the "boosted perceptrons" can be learned in $O(m^2 \log m)$ (where m is the sample size) whereas the "boosted decision tree" cannot be guaranteed to converge unless the tree depth is at least $\log m$. Considering the result that training a neural network of two hidden and one output perceptron is NP-complete (Blum & Rivest, 1992) this clearly points out an unnoticed benefit of boosting: By growing the hypothesis space dynamically, AdaBoost maps the search space into a larger space in which a greedy algorithm exists that can decide the existence of a hypothesis which is consistent with the sample (and, consequently, find that hypothesis) in polynomial time.

Analogous results have been obtained, for instance, for the learnability of Boolean functions: While k-term CNF is not polynomially PAC learnable because no polynomial algorithm can find a hypothesis that is consistent with the sample , k-DNF *is* learnable although k-term CNF $\subset$ k-DNF (Pitt & Valiant, 1988). The reason is that a greedy algorithm exists which decides the existence of a consistent hypothesis in k-DNF.

6.2 Further Definitions

Learning problem. In this Chapter, I assume that there is an unknown *target function* $f : X \rightarrow \{0,1\}$. The *learner* perceives a sample S consisting of m points $x_i \in X_S$ and the corresponding $f(x_i)$ and returns a *hypothesis* $h : X \rightarrow \{0,1\}$. The learner may also have access to a distribution π on sample points. This distribution must not be confused with the underlying (but unknown) distribution D_X. π is a somewhat artificial distribution the objective of which is to put higher weight on those examples which are misclassified by previous elementary hypotheses. By putting a higher weight on these instances, we can maximize the chance that they will be classified correctly by newly learned elementary hypotheses and, eventually, by the majority of elementary hypotheses. The (empirical) *error* of h with respect to π is then defined as $E_{f,X_S,\pi}(h) = \frac{1}{m} \sum_{x_i \in X_S} \ell(h(x_i), f(x_i))\pi(x_i)$, where

ℓ is the zero-one loss function.

AdaBoost. The algorithm AdaBoost (Freund & Schapire, 1997) receives a sample S, a distribution π on points, and an integer T which indicates the desired number of iterations. The algorithm initializes the weight vector w_i^1 to $\pi(x_i)$ and, at step t, proceeds as follows:

1. $\pi^t(x_i) = \frac{w_i^t}{w_1^t + w_2^t + \ldots + w_m^t}$.
2. The weak learner is invoked with distribution π^t and returns hypothesis h_t.
3. $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$ where ϵ_t is the error of h_t with respect to π^t.
4. The new weight vector is set to $w_i^{t+1} = w_i^t \beta_t^{1-|h_t(x_i)-f(x_i)|}$. The final hypothesis consists of a final weight vector β and the weak hypotheses $h_1, \ldots, h_T$.

The compound hypothesis is then the weighted majority (weighted according to β) of the individual hypothesis, more formally:

$$h(x_i) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(x_i) \geq \frac{1}{2} \sum_{t=1}^{T} \log \frac{1}{\beta_t} \\ 0, & \text{otherwise} \end{cases}$$

Perceptrons. A perceptron is a discriminating hyperplane. A perceptron is specified by vectors a and b which define the function

$$h_{\{a,b\}}(x) = \begin{cases} 1, & \text{if } ax + b \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

Accordingly, a *boosted perceptron* consists of a collection of T perceptrons together with a weight vector (the weights may be equal for all perceptrons). The outcome of a boosted perceptron for a point x_i is 1, if the weighted majority of the T perceptrons outputs 1, and 0 otherwise.

Shallow decision trees. The decision tree of fixed depth k which minimizes the error on some sample S can be found in polynomial time (decision trees of depth 2 can be learned in $O(n^2 m \log m)$ Auer *et al.*, 1995; Dobkin *et al.*, 1996). This is a surprisingly positive finding as the error minimizing hypotheses for half-spaces (Höffgen *et al.*, 1995) or conjunctive concepts (Kearns *et al.*, 1992) cannot be found in polynomial time. This makes shallow decision trees particularly interesting weak hypotheses for boosting algorithms. In some empirical studies, even decision trees of depth 1 ("decision stumps") are studied (Freund & Schapire, 1996; Breiman, 1996).

6.3 A Worst-Case Bound for AdaBoost with Perceptrons

In this Section, I will study the worst-case behavior of boosted perceptrons. First, I have to discuss the worst-case behavior of the weak learner, *i.e.*, of a perceptron. When the examples are linearly separable, a simple, well-known greedy algorithm can construct the separating perceptron (Rosenblatt, 1958). Unfortunately, in the general case, finding a half-space which minimizes the error rate on a sample is NP-complete (Höffgen *et al.*, 1995)[1]. However, this does not necessarily mean that learning a boosted perceptron is not possible in polynomial time. It suffices when a weak hypothesis with an

[1] Not that, while finding a hyper-plane that minimizes the zero-one loss is NP-complete, finding a plane that maximizes the squared distance between positive and negative examples is much easier (Unger & Wysotzki, 1981) which gives rise to more efficient piecewise linear classifiers (Schulmeister & Wysotzki, 1994)

error of slightly less than $\frac{1}{2}$ can be constructed in polynomial time. As we will see, this can easily be done in an efficient manner. I prove this by demonstrating such an algorithm. The algorithm picks an arbitrary example point x and constructs a hypothesis which classifies at least half of the sample without x correctly *and* classifies x correctly. Hence, the number of correctly classified points exceeds the number of mis-classified sample points by at least 1. In Lemma 1, I show that a hyper-plane which touches x but no other point can be constructed efficiently from the sample. This is always possible because the finitely many examples cannot "fill" the space.

Lemma 1 *Let x be a point in $\mathbb{R}^n$ and let F be a finite set of points not containing x. Then one can efficiently find a hyper-plane $p \in \mathbb{R}^{n-1}$ containing x but no point from F.*

Proof of Lemma 1. Without loss of generality, we can assume that all coordinates of x are 0 — otherwise this could be obtained easily by a coordinate transformation. Now p is constructed as a linear mapping f on the subspace generated by the coordinates $x_2, \ldots, x_n$ to the first one such that the plane contains the points $(f(x_2, \ldots, x_n), x_2, \ldots, x_n)$. Since F does not contain x, the definition $f(0, \ldots, 0) = 0$ guarantees that the plane is disjoint to all points in F of the form $(a, 0, \ldots, 0)$. Now the linear function f is defined inductively on the values z_k which are 0 for the all coordinates except the k-th one where z_k takes 1. The goal of the construction is to define $f(z_k)$ such that p is forced not to contain any points from F_k in the k-th step where F_k contains those points in F for which the k-th coordinate is the last non-zero one, that is, F_k is in the linear hull of $\{z_1, z_2, \ldots, z_k\}$ but not in the linear hull of $\{z_1, z_2, \ldots z_{k-1}\}$. For each number in $y_i \in F_k$ there is a unique f_i such that $y \in p$ iff $f(z_k) = f_i$. One can compute these finite numbers and define that $f(z_k) = 1 + |f_1| + \ldots + |f_{m'}|$ for the m' numbers f_j. It follows that none of the elements of F_k is on p and, by induction, that p is disjoint from F. ∎

I now use Lemma 1 to show that we can efficiently construct a perceptron which achieves a hit rate of at least $\frac{m+1}{2m}$. Essentially, we pick a point x with maximal weight $\pi(x)$ from the sample and construct a hyper-plane which touches x but no other example. By switching the sign of the perceptron, we can always achieve an error rate of at most $\frac{1}{2}$ on the sample *without* x. After fixing the sign, we can "shift" the plane slightly "forward" or "backward", such that x falls on the "right" side of the plane but no other sample point changes the side. Hence we obtain at least one extra hit.

Theorem 16 (Scheffer & Stephan, 1998) *Let $X_S = \{x_1, x_2, \ldots, x_m\}$ be a set of data points with class labels $f(x_i)$ and let π be a distribution on X_S. Then one can efficiently find a perceptron h with*

$$Err_{f,X_S,\pi}(h) \leq \frac{m-1}{2m}.$$

Proof of Theorem 16. Let $x_p \in X_S$ be a point with maximally large $\pi(x_p)$ (if there are several points which maximize π, x_p may be either of them) and let $X_S^{-p} = X_S \setminus \{x_p\}$. We know that $\pi(x_p)$ is at least $\frac{1}{m}$. Lemma 1 implies that we can select a hyper-plane $ax + b = 0$ efficiently which touches x_p but none of $X_S \setminus \{x_p\}$. Let, without loss of generality, $|a| = 1$ (otherwise, we can normalize a and b). Then we can define the following two perceptrons: $h_1 = ax + b \leq 0$, and $h_2 = -ax - b \leq 0$ – *i.e.*, h_1 and h_2 are complementary for all sample points other than x_p. Let us look at the error of h_1 and h_2 on X_S^{-p} (with respect to π). It follows from the complementarity of h_1 and h_2 that $\pi(x_p) + \sum_{x_i \in X_S^{-p}} \pi(x_i)(|f(x_i) - h_1(x_i)| + |f(x_i) - h_2(x_i)|) = 1$. x_p was chosen such that $\pi(x_i) \geq \frac{1}{m}$, hence $\sum_{x_i \in X_S^{-p}} \pi(x_i)|f(x_i) - h_1(x_i)| + \sum_{x_i \in X_S^{-p}} \pi(x_i)|f(x_i) - h_2(x_i)| \leq 1 - \frac{1}{m}$. This

implies $\min\{\sum_{x_i \in X_S^{-p}} \pi(x_i)|f(x_i) - h_1(x_i)|, \sum_{x_i \in X_S^{-p}} \pi(x_i)|f(x_i) - h_2(x_i)|\} \leq \frac{1-1/m}{2} = \frac{m-1}{2m}$. Hence, at least one hypothesis from $\{h_1, h_2\}$ incurs an error of no more than $\frac{m-1}{2m}$ on all points except x_p. Let this hypothesis be $a^*x + b^* \leq 0$. By now, I disregarded a possible misclassification of x_p. Let the smallest distance the plane and any data point (except x_p) be 2ϵ (Lemma 1 implies that the distance is non-zero). Then we can define two hypotheses: $h_*^1 = a^*x^* + b + \epsilon \leq 0$ and $h_*^2 = a^*x^* + b - \epsilon \leq 0$. $|a| = 1$ and the definition of ϵ imply that these hypotheses behave equally for all points except for x_p but are complementary for x_p. Hence, one of them, h_*^*, will classify x_p correctly. Therefore, h_*^* has a total error (with respect to π) of no more than $\frac{m-1}{2m}$. ∎

Unfortunately, the above bound cannot be improved.

Observation 1 *No learning algorithm exists which can be guaranteed to find a boosted perceptron with error less than $\frac{m-1}{2m}$ on m data points x_S with respect to distribution π. In particular, the worst case complexity of a boosting learner cannot be below $m-1$ invocations of the weak learning algorithm.*

Proof of Observation 1. Assume $\pi(x_i) = \frac{1}{m}$ for all x_i. Assume further that m is odd and that $X_S = \{1, \ldots, m\}$. We define f such that $f(x) = 1$ if x is even, 0 otherwise. The following hypotheses are possible: (a) $x \leq -1$ (effectively no split). There are $\lceil \frac{m}{2} \rceil$ odd numbers and $\lfloor \frac{m}{2} \rfloor = \lceil \frac{m}{2} \rceil - 1$ even numbers; hence, the error is $\frac{m-1}{2m}$. (b) $x \leq c$, where c is even and $0 \leq c \leq m-1$. There are equally many odd and even numbers between 1 and c (for even c) but there are $\lceil \frac{m-c}{2} \rceil$ odd numbers and only $\lfloor \frac{m-c}{2} \rfloor$ even numbers between $c+1$ and m. Hence, this hypothesis incurs error $\frac{m+1}{2m}$. (c) $x \leq c$ for odd c ($1 \leq c \leq m$). There are $\lceil \frac{c}{2} \rceil$ even and $\lfloor \frac{c}{2} \rfloor$ odd numbers between 1 and c and equally many odd and even numbers between $c+1$ and m. This hypothesis incurs an error of $\frac{m-1}{2m}$. These are all affine hypotheses in the given space.

The second statement is due to the fact that a majority vote on some perceptrons can identify X_S with error 0 only if there is, for each $i < m$, a perceptron which outputs different values for x_i and x_{i+1}. Therefore, the final hypothesis needs at least $m-1$ perceptrons and the boosting algorithm has to invoke any given weak learner at least $m-1$ times. ∎

Now that it has become clear *how much* better an accuracy than randomly guessing the weak learners can be guaranteed to achieve, I can give *worst-case* bounds on the error of the combined hypothesis with respect to the sample.

Theorem 17 (Scheffer & Stephan, 1998) *AdaBoost, using perceptrons as weak hypotheses, requires at most $\lceil 2m^2 \log \frac{1}{\varepsilon} \rceil$ iterations in order to produce a hypothesis with error at most ε.*

Proof of Theorem 17. In Theorem 16, I showed that each perceptron incurs an error of at most $\frac{m-1}{2m}$. Hence, I can bound the difference between the weak learner and "randomly guessing" $\gamma \geq \frac{1}{2m}$. Equation 23 of (Freund & Schapire, 1997) bounds the number of invocations of the weak learner by AdaBoost, required to reach an error of at most ε to $\lceil \frac{1}{2\gamma^2} \log \frac{1}{\varepsilon} \rceil$. This yields $\lceil 2m^2 \log \frac{1}{\varepsilon} \rceil$ when $\gamma \geq \frac{1}{2m}$. ∎

This shows that AdaBoost converges well, reaching any given error bound in polynomially few steps.

Corollary 4 *AdaBoost will produce an error (with respect to the sample) of zero after at most $2m^2 \log m$ iterations.*

Proof of Corollary 4. Follows from Theorem 17 using $\varepsilon < \frac{1}{m}$. ∎

The bounds given in Theorem 17 and Corollary 4 are based on the assumption that the number of hits of the weak hypotheses exceeds the number of failures by just one sample point. Hence, these bounds give the borderline for any weak learner under which boosting still converges.

Corollary 4 sais that AdaBoost will always find a hypothesis which is consistent with the sample in polynomial time, but the hypothesis may consist of up to $m^2 \log m$ elementary hypotheses. On the other hand, two-layer neural networks have a fixed number of hidden units. But, unfortunately, there is *no known algorithm* for the training problem of neural networks with sub-exponential time worst case complexity.

Theorem 18 (Blum and Rivest, 1992) *Deciding whether a neural network with n input, two hidden and one output neuron exists that is consistent with a given sample (or even finding the neural network with two hidden unit that minimizes the empirical error) is NP-complete.*

In fact, training a neural network with a number of hidden units which is bounded polynomially in the number of input terminals n is NP-complete.

Observation 2 *While being more expressive than neural networks with two hidden and one output units, boosted perceptrons can be learned in polynomial time while neural networks cannot (unless $P = NP$).*

Proof. By more expressive, I mean that there is a boosted perceptron which is consistent with an arbitrary f on an arbitrarily large set of points X_S (this follows from Theorem 17; this boosted hypothesis will consist of at most $m^2 \log m$ perceptrons). By contrast, the VC-dimension of a neural network with a fixed number of units is a fixed number d which means that it cannot be consistent with every possible function f on a set of at least $d+1$ points (otherwise its VC-dimension would be d+1). While Theorem 17 claims that the boosted perceptron can be learned in P, Theorem 18 shows that training a neural network is NP-complete. ∎

6.4 Boosting Decision Stumps

The empirical error of shallow decision trees can be minimized very efficiently: when the depth is 2, the T2 algorithm finds the minimizing tree over n variables in $O(n^2 m \log m)$ (Auer *et al.*, 1995) – *i.e.*, finding an optimal shallow decision tree is much easier than finding an optimal hyper-plane. This raises the question whether shallow decision trees ("decision stumps") can be used as weak hypotheses for boosting algorithms. Unfortunately, shallow decision trees cannot be guaranteed to drop the error below $\frac{1}{2}$.

Theorem 19 (Scheffer & Stephan, 1998) *Given m examples, AdaBoost does not, in general, terminate when the weak hypotheses are decision trees of depth less than $\log m$.*

Proof of Theorem 19. Consider a Boolean space with n dimensions. Let $\pi(x_i)$ give equal weight $\frac{1}{2^n}$ to all points. Let the x_i ($1 \leq i \leq 2^n$) be all possible data points and let $\#1(x_i)$ be the number of 1's occurring in x_i (*e.g.*, $\#1((0,0,1,0,1))$ equals 2). I then define f as $f(x_i) = 1$ if $\#1(x_i)$ is odd, 0 otherwise (intuitively, f resembles an n dimensional chess board). I claim that no decision tree

of depth less than $\log m$ can achieve an error rate of less than $\frac{1}{2}$. Let X_l be the points which reach an arbitrary leaf. Let the corresponding branch perform a sequence of tests $(v_1 = a_1, \ldots, v_k = a_k)$, where k is at most $\log m - 1$, v_k are Boolean variables and a_k are 0 or 1. All points which "reach" this leaf have defined values for k attributes and arbitrary values on the remaining $n - k$ attributes – hence, there are 2^{n-k} points in any leaf which differ in their value of $n - k$ attributes. Each of these points corresponds to a binary number with $n - k$ bits. There are equally many binary numbers with an odd number of 1's as there are with an even number; hence, at any leaf the number of points x with $f(x) = 1$ is equal to the number of points with $f(x) = 0$. Therefore, no decision tree can achieve more than exactly $\frac{m}{2}$ hits. Voting among hypotheses with error $\frac{1}{2}$ yields a total error chance of $\frac{1}{2}$ again. ∎

AdaBoost with decision trees can, however, be guaranteed to terminate yielding an arbitrarily small error when the trees is of depth at least n (the number of attributes). However, such maximally deep trees do not match the intuition of boosted decision stumps (note that one single tree of depth n already is a consistent hypothesis). Of course, Theorem 19 refers to the worst case. Boosting with shallow decision trees *may* terminate when the problem is less malicious.

Intermediate decision trees are those which branch at most $\log(m) + 1$ times. Such programs try to combine learnability with the intuitive idea of small concepts. They allow AdaBoost to learn decision trees although the boosting algorithm needs a slightly super-polynomial but not exponential number of iterations.

Theorem 20 (Scheffer & Stephan, 1998) *Given m examples, AdaBoost learns the hypothesis in $m^{2\log(m)+1}$ iterations from weak hypotheses which are decision trees of depth $\log(m)$.*

The proof can be found in (Scheffer & Stephan, 1998).
The lower bound of $m - 1$ iterations for a weak learner based on perceptrons was established by showing that the full interpolation of the problems needs at least $m - 1$ perceptrons. In the present case, this lower bound can not be improved to a super-linear value, thus there is a gap between the lower and upper bound for the worst case complexity of AdaBoost when learning decision trees with logarithmic depth or even only logarithmically many branching points.

6.5 Discussion and Related Work

I showed that a boosted perceptron can be learned in $O(m^2 \log m)$. By contrast, a two-layer neural network with two hidden and one output neuron cannot be learned by any polynomial algorithm unless $P = NP$. However, for any function there is a boosted perceptron which is consistent with that function on any finite sample of points – *i.e.,* the expressive power of boosted perceptrons is greater than the expressive power of a fixed sized neural network. This shows that *increasing* the expressiveness of the hypothesis eases the difficulty of finding a consistent hypothesis because consistent boosted perceptrons can be found by a greedy algorithm whereas neural networks cannot. There are two known analogous results in different domains. An element of k-term CNF(n) is a conjunction of at most k Boolean disjunctions over n Boolean variables. k-DNF(n) contains all disjunctions of conjunction with at most k literals over n Boolean variables. Every k-term CNF can be turned into a k-DNF but not vice versa – *i.e.,* k-term CNF$(n) \subset k$-DNF(n). The sample complexity required to PAC-learn both k-term CNF and k-DNF is polynomial in n (for fixed k). However, in order to decide the existence of a hypothesis that is consistent with the sample in k-DNF one essentially has to exhaust the whole hypothesis space which is exponential in n. Using k-DNF as hypothesis language

one can greedily search for a conjunction of up to k literals which covers at least one positive example and is consistent with the negative examples, then remove the covered positive instances and recur. This greedy algorithm will find a consistent hypothesis if one exists and runs in time polynomial in n. The returned DNF will not necessarily contain the least number of conjunctions but note that, in contrast to k-term CNF, the number of terms (*i.e.*, conjunctions) is not restricted for k-DNF. A similar result holds for the Support Vector Machine. While it is NP-complete to find the hyper-plane which minimizes the mis-classification rate (Höffgen *et al.*, 1995) the Support Vector Machine inflates the space by adding polynomials of the original attributes (the "kernel trick" assures that these added attributes need not be represented explicitly). In the inflated space, a hyperplane which is consistent with the sample can be found by a greedy algorithm in polynomial time.

In all three cases, the drawback of inflating the hypothesis space is that a larger sample is required for good generalization. But in many situations this may be a reasonable trade-off. The sample required for k-DNF(n) is still polynomial and for many "practical problems" the appropriateness of the maximally wide margin bias compensates for the more complex hypothesis space.

6.6 Summary

- In the model selection framework that has been studied in the previous chapter, the model selection algorithm pre-selects a model and invokes the learner which is then required to minimize the error within the given model. By contrast, in the boosting setting the learner constructs a hypothesis and may grow the hypothesis language dynamically until a hypothesis which is consistent with the sample is found.

- Growing the hypothesis dynamically, as is done by the AdaBoost algorithm, is very efficient. AdaBoost with perceptrons operates in $O(m^2 \log m)$ while a comparable task with a "static" hypothesis language (multilayer network with a fixed number of units) is NP-complete.

- The hypotheses learned by AdaBoost are more complex. They consist of up to $m^2 \log m$ elementary hypotheses. This makes proving meaningful bounds on the largest difference between the true and empirical error of a hypothesis impossible. However, it does not imply that the boosted hypotheses are worse than the static hypotheses. According to Chapter 3, this depends only on how adequate the hypothesis language is for the problem, which can be measured in terms of the distribution $P_{\{h\}}(E_D(h)|H_i, D_{XY})$.

Chapter 7
Conclusion

The problem of error minimization revolves around the following problem. A hypothesis h incurs an unknown error; the error can be estimated using a sample which usually yields an unbiased estimate of the true error. When there are many hypotheses h_j, then the error rate of each of them can be estimated unbiasedly on the sample. Unbiased means that the expected empirical error of a hypothesis h_j with true error $E_D(h)$ is just the true error $E_D(h)$. The empirical error is linked to the true error by the binomial distribution. However, as we want to minimize the error, we now focus on the hypothesis which minimizes the observed, empirical error. But the expected empirical loss of the hypothesis h_L which minimizes the empirical error is, unfortunately, distinct from its true error $E_D(h)$. Some empirical error rates are optimistic estimates while others are pessimistic estimates of the corresponding true error rates and, going for the smallest observed loss, one is likely to choose a hypothesis the true loss of which has been estimated rather optimistically. In order to know how well a particular learning bias performs for a particular problem we would like to find out *just how* optimistically biased the empirical loss of h_L is.

One possible answer to this problem is provided by cross-validation. When a small part of the sample has been held back and is now used to asses only this one single hypothesis, the estimate is only subject to a small pessimistic bias. PAC and VC theory provide different answers by bounding the largest difference between the empirical and true loss of any hypothesis by Chernoff bounds. But the PAC bounds are overly general as they hold for *any* problem (not just some particular, given problem which might be less difficult) and bound the error of all hypotheses (not just h_L). The expected error analysis provides a very different answer for finite models. It quantifies the expected error of the hypothesis hypothesis which minimizes the empirical error for the given fixed sample size (which is a very natural and widely used target criterion) in terms of the prior distribution of error rates in the model $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. The error prior can be estimated efficiently from a sample by recording its empirical counterpart $P_{\{h\}}(E_S(h)|H_i, S)$ which leads to an efficient way to estimate the error rate of the hypothesis which would be the result of an error minimization process, without this process having to be carried out.

When the hypotheses h_j are the output of a parameterized learner L_p which are assessed on the same hold-out sample, a similar situation arises. The lowest observed hold-out error of several learners is an optimistic estimate of this learner's true error and it would be interesting to know *just how* optimistically biased the hold-out error rate is. When the hypotheses are assessed by means of training and test, PAC-style results and elementary statistic results on the confidence of estimates can be applied. But when the hypotheses are evaluated by n-fold cross validation such approaches would require the additional assumption that estimates obtained in distinct folds are independent. This assumption is not reasonable because it would imply that it is possible to increase the available information. In Chapter

5, I presented an information theoretic approach in which I treated the dependencies between distinct cross validation folds explicitly and quantified the bias. The solution depends on several elementary properties of the data set which can easily be measured (such as the entropy and the hit rate of the default classifier) and is therefore not a worst-case result.

7.1 Expected Error Analysis

One of the main contributions of this thesis is an analysis of the expected error of h_L, where h_L is a hypothesis which has been drawn at random under uniform distribution from $H_i^*(S)$, the hypotheses with least empirical error within model H_i. The posterior distribution of error values of h_L is reduced to the prior distribution $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ of error values of hypotheses in H_i. The practical use of this analysis is that the prior can be estimated by recording $P_{\{h\}}(E_D(h)|H_i, S)$, the empirical errors of randomly drawn hypotheses which gives an efficient means of estimating the error of the hypothesis which an error minimizing learner that operates on H_i will return. No learning has to be conducted in order to obtain this estimate (as opposed to cross validation).

Understanding Learning Curves. Learning curves express the relationship between the hypothesis language and the generalization error. In order to construct learners which incur a low generalization error, it is important to understand learning curves. PAC/VC theory provides an explanation to the shape of learning curves which, when taken as a guideline for the construction of learners, would sometimes leads to sub-optimal learning algorithms. The PAC model of learning curves is coupled to the largest difference between true and empirical error of any hypothesis which is inevitably increasing (and makes PAC theory predict greater errors) when the number of hypotheses grows. However, in experiments learning curves often behave complementary. This has, for instance, been observed in the context of boosting (*e.g.,* Schapire *et al.*, 1997), but also with decision trees or other hypothesis languages (Fisher & Schlimmer, 1988; Schaffer, 1993a). A boosted hypothesis is a weighted majority of several elementary hypotheses. An increase in the number of elementary hypotheses often causes the learning curve to decrease which contradicts PAC theory. The expected error analysis explains that this can happen when the error prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ stays constant while $|H_i|$ grows. This happens, for instance, when further relevant attributes are added to H_i. On the other hand, when irrelevant attributes are added, the variance of the prior decreases which causes the predicted error to increase. The predicted learning curves match those measured in simulations nicely, although there is a small (but fairly constant) bias caused by the assumed independence of empirical errors of distinct hypotheses.

Empirical Results. I conducted experiments on an artificial problem, on many randomly drawn Boolean functions, and on a large-scale text categorization problem. Unfortunately, the error estimates did not turn out to be unbiased. Two distinct types of bias interfere. The assumed independence assumption caused a small pessimistic bias. However, the bias did not distort the shape of the learning curves considerably. More importantly, the location of the optimal model was predicted quite accurately. A second (optimistic) bias was due to the way in which the prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ was estimated. For small sample sizes, the true prior and the distribution $P_{\{h\}}(E_S(h)|H_i, S)$ differ considerably. Note that $P_{\{S,h\}}(E_S(h)|H_i, m) = \int_{e_D} B[e_D, m] dP_{\{h\}}(E_D(h)|H_i, D_{XY})$. This means that, even if $P_{\{h\}}(E_D(h) = 0|H_i, D_{XY})$ is zero, the chance of an empirical error of zero is strictly greater than zero. This in turn means that the ratio of very good hypothesis is over-estimated. But note further that $P_{\{S,h\}}(E_S(h)|H_i, m)$ converges towards $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ as m grows – *i.e.*, the estimate is asymptotically consistent. The empirical comparison of expected error analysis based model selection with 10-fold cross validation showed that, even for small sample sizes, the error rates obtained by

the expected error analysis based model selection are about equal to the error rates obtained by means of cross validation; in some cases even better. The experiments on the text categorization problem demonstrated that the expected error analysis scales well to large problems. Here, assessing a model required about two minutes, while it would have required about 80 hours for 500 attributes. An excerpt of the learning curve which has been estimated by hold-out testing was matched reasonably well by the predicted learning curve (up to the usual pessimistic bias). The expected error analysis selected models for the ten most frequent categories which lead to hypotheses with an average hold-out error that was close to the least hold-out error that was observed for any model.

Limitations. The presented framework is subject to some constraints. Perhaps the most important one is the assumed finiteness of the H_i. As yet, the analysis cannot be applied to regression problems, or to classification problems when the hypothesis language consists of, for instance, linear units. Also, the number of models has to be finite. This, however, is not usually a problem as the number of attributes is most often finite. When the learning curve is assumed to be "U"-shaped (*i.e.,* convex), infinite stratifications can be processed. In this case, the search can be stopped when the minimum has been passed. Another important constraint is that only the expected generalization error is minimized. So far, no efficient and practical solutions have been found for other loss functions. The assumed independence of the empirical errors of distinct hypotheses should be thought of as a source of a modest bias, rather than a constraint on the applicability. The experiments have shown that this assumption accounts for a bias, but this bias is not very strong and, more importantly, has not been observed to distort the position of the minimum strongly.

Extensions: Linear cost models and continuous models. So far, the expected zero-one loss has been the only loss function for which the expected error analysis has been conducted. This raises the question whether there is a solution for other loss functions, such as linear cost functions, or the quadratic loss used for regression. Where is the loss function "plugged" into the derivation? In the proof of Theorem 6, the empirical error (or zero-one loss) under given true error is given as $P_{\{S\}}(E_S(h)|E_D(h), m) = B[E_D(h), m]$. When a distinct cost function is to be used, the distribution of empirical loss values under given true loss has to be plugged in at this point. It turns out that there is a similar solution for linear cost functions. For some learning problems, there is a cost matrix $c : Y \times Y \to \mathbb{R}$ available. $c(y_1, y_2)$ specifies the costs which are imposed when a hypothesis assigns a class label y_2 to an instance, when the instance really belongs to class y_1. In many medical diagnosis applications, such a cost function, rather than the zero-one loss, has to be minimized. Missing a serious disease is considerably worse than making an overly pessimistic diagnosis. Unfortunately, the solution is based on the prior distribution $P_{\{h\}}(h(x) = y_1|f(x) = y_2, H_i, f)$ instead of $P_{\{h\}}(E_D(h)|D_{XY}, H_i)$. This distribution has $|Y|^2$ dimensions – *i.e.,* when k hypotheses are enough to estimate $P_{\{h\}}(E_D(h)|D_{XY}, H_i)$, then $k^{|Y|^k}$ hypotheses are required in the linear cost case. This will usually be too expensive. Extending the analysis to cover regression turns out to be difficult, too. There is no meaningful equivalent to $P_{\{h\}}(E_D(h)|D_{XY}, H_i)$ for infinite spaces H_i because this would require a uniform distribution over an infinite H_i. One approach that might perhaps lead to a solution could be to consider only the error values of those hypotheses which are considered by a greedy learner. But such an approach faces the problem that the recorded empirical errors would not be an unbiased estimate of the true errors which occurred during the search (because the greedy search is guided by low empirical errors).

When the sample size is small, $P_{\{h\}}(E_S(h)|H_i, S)$ is not a good estimate of $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ (*i.e.,* the estimate is biased). Nevertheless, the error rate of expected error analysis based model selection was not worse than the error rate of cross validation based model selection, even for small sample sizes. However,

for domains in which the sample size is small it might be interesting to obtain a better solution. The problem of obtaining $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ from $P_{\{h\}}(E_S(h)|H_i, S)$ is equivalent to the problem of estimating a mixing density (*e.g.,* Goutis, 1997). Many solutions to this intensely studied problem exist: Several non-parametric approaches are based on the deconvolution of a kernel estimator of the observed data (*e.g.,* Goutis, 1997; Carroll & Hall, 1988; Liu & Taylor, 1989); furthermore, the discrete maximum likelihood estimator has at most $|S|$ mass points (Laird, 1978). Parametric approaches assume a model for the mixing density (here, $P(E_D(h))$) and adapt the model parameter such that the error between the observed and the predicted $P(E_S(h))$ is minimized – using, for instance, the EM algorithm (*e.g.,* Vardi & Lee, 1993; Eggermont & LaRicca, 1995). These results can be applied to the expected error analysis and promise more accurate results for very small sample sizes (see also Section 4.2 of Scheffer & Joachims, 1998b).

7.2 Is the Error Rate an Intrinsic Property of the Hypothesis Language?

PAC and VC results bound the error rate of any hypothesis which has been learned from a hypothesis language H in terms of the hypothesis language H. Either the size of H or, similarly, the VC dimension is used to bound the possible error (or the difference between empirical and generalization error) which a hypothesis that has been learned from this hypothesis language (or model) can impose. Even the best hypothesis within a small hypothesis language is likely to incur a relatively high error rate (the error rate of the best hypothesis in the hypothesis language is often referred to as the bias term in the resulting error). On the other hand, the greatest difference between true and empirical error rate of any hypothesis in the hypothesis language grows with the model size. Hence, we know less about the true error of the returned hypothesis. The difference between the error of the best hypothesis in the hypothesis language and the error rate of the hypothesis which is returned by the learner is sometimes referred to as the variance term of the error rate. When the size (or VC dimension) of the hypothesis language grows, we know less about the error rate which is often misinterpreted as meaning that the variance term of the error increases. This sometimes leads to the false idea that some language biases are intrinsically better than others or that conducting model selection (and thereby trading the bias term against the variance term) is intrinsically beneficial. But experiments of Schaffer (1993a, 1993b) and the theoretical analysis of Wolpert (1992, 1993, 1995) show that all learning biases are *per se* equally good (as long as the empirical error is minimized and as far as the expected generalization error is concerned). A better-than-average result for a set of problems can only be achieved by implementing additional assumptions on the problems into the learner and thereby narrowing the learner to these problems. These considerations provide evidence that the generalization error can only meaningfully be quantified with reference to both the problem and the hypothesis language and provides further motivation for the error analysis presented in Chapter 3. The crucial property of the problem which influences the generalization error is the prior distribution of error rates $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. With reference to this distribution it can be explained in which cases curve simply decreases (although the *bound* on the variance term increases) and in which cases the learning curve increases again and overfitting occurs. The expected error analysis can be taken one step further (see Section 4.3) to explain in which cases conduction (training and test based) model selection is beneficial. Whether this is the case depends on the prior distributions of error rates $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ of all models H_i in the stratification $\langle H_1, \ldots H_k \rangle$. A typical situation in which the learning bias of conducting model selection is beneficial is when it is known that some of the attributes are likely to be irrelevant.

These results have implications on how learning algorithms should be applied to practical prob-

lems. Since it is impossible to construct a learner that is both general and accurate, one has to specialize the learner as strongly as possible to a given problem in order to obtain a good result for that problem. Specializing a learner means to encode all available background knowledge into the learning bias. When nothing is known about a focused problem, the best one can do is to construct a learner which solves all possible problem equally well. On the other hand, when the target distribution can be determined exactly without any learning (this is an extreme special case), one can construct a trivial learner which always returns the function that approximates this distribution optimally without referring to potentially available data. Such a learner would solve this particular problem optimally but would solve no other learning problem reasonably. Typically, some background knowledge is available on the focused learning problem which makes it possible to narrow the hypothesis space to a model which contains a high density of hypotheses which are good for this problem. This can, for instance, be realized by choosing (or constructing) a small set of attributes which carry all necessary information relevant to the classification task.

7.3 When does Model Selection Work?

Model selection is often considered to be a technique which generally improves the performance of learners. In fact, virtually all "practical" learners employ some type of model selection technique, such as pruning, or weight decay. However, it is not known for exactly which learning problems model selection is beneficial (as compared to choosing the maximally complex model in the first place). The results presented in this book indicate that the class of problems which should be solved by means of cross validation based learning is smaller than has generally been assumed. Generally speaking, Theorems 11 through 13 say that Occam algorithms are useful only when the stratification is aligned with the prior distribution of target functions $P(f)$. Theorem 14 quantifies the generalization error of a hypothesis that has been generated by one-fold cross validation based learning which can be compared to the error of a hypothesis generated by simple error minimization with a specific stratification (quantified by Theorems 6, 7). Which of these errors is lower depends on the problem, *i.e.*, on the error priors $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ for all H_i. These can, in principle, be estimated and a decision can be made based on the estimates and Theorems 6,7, and 14. In fact, the prior arrays can be estimated for several distinct stratifications and a decision on the optimal stratification. However, the error priors are only estimated which imposes the risk of mis-estimating the errors and thus making a wrong decision. The greater the number of considered stratifications is, the more optimistically biased the lowest estimated error will be – *i.e.*, a meta-level over-fitting occurs.

Complexity of Model Selection. At first blush, learning from a large hypothesis language appears to be more difficult than learning from a small hypothesis language because error minimization usually requires $O(|H|)$. Surprisingly, though, in many cases at least finding a hypothesis that is consistent with the sample is much easier for larger hypothesis languages. The reason is that for some redundant representation greedy algorithms can be found which can construct consistent hypotheses in time logarithmically in the time which is required to enumerate the hypothesis space. In Chapter 6, I demonstrated this phenomenon referring to AdaBoost and multilayer networks. Learning from hypothesis languages which are larger than necessary can result in a lower generalization performance. However, in contrast to the PAC results, the results of Section 3.5 show that this does not have to be the case. When the learning bias can be chosen very adequately for the problem, then even techniques like boosting can find hypotheses with very low error efficiently.

When Should the Expected Error Analysis be Used? In Section 7.1 and in Chapter 3, I discussed when the expected error analysis *can* be applied, but this still leaves the question open when it *should*

be applied. When the prior distribution $P(f)$ is known, under ideal conditions and when the function class is small, the optimal Bayes hypothesis can be determined. Returning any other hypothesis (like a hypothesis determined by the expected error analysis) would lead to a sub-optimal result. When $P(f)$ is known but the Bayes hypothesis cannot be determined, the MAP or MDL hypotheses are often reasonable; it is not clear whether these hypotheses perform better than other model selection algorithms with a stratification which is aligned with the prior. But as the expected error analysis algorithm exploits the prior $P(f)$ only in a very weak form (only by means of the stratification which can be aligned with the prior) one can expect to most likely be better off using the MAP or MDL hypothesis. When the prior is not known, a decision has to be made between complexity penalization, cross validation, and the expected error analysis. Most complexity penalization algorithms require a parameter that trades empirical error against model complexity. This parameter effectively determines which model is selected. It usually has to be adjusted by means of hold-out testing. Some complexity penalization methods (like Schuurmans, 1997) use a certain heuristic instead of a parameter. On one hand this means that no parameter has to be adjusted but, on the other hand, the heuristic will certainly do well for some problems while it will just as certainly fail for others.

An advantage of cross validation over the expected error analysis is that the cross validation estimates are almost unbiased. By contrast, the expected error analysis yields pessimistically biased estimates (for sufficiently large samples) – although this does not necessarily cause the expected error analysis to chose a sub-optimal model more often than cross validation. However, when the sample size is below 50, the error prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ cannot be estimated with sufficient accuracy which renders cross validation preferable. The principle drawback of cross validation is the high computational complexity. The longer a run of the learner takes, the more preferable the expected error analysis is.

Therefore, the most promising areas of application of the expected error analysis are large-scale learning tasks, such as text categorization, knowledge discovery in databases, or other large-scale classification tasks.

7.4 Applicability of Learning Algorithms

Virtually all loss functions (*e.g.*, the zero-one loss studied throughout this book or the quadratic loss used for regression) assume a distribution on instances relatively to which the loss is defined, and the observations are required to be independent and identically distributed according to this distribution. One should be aware of the restriction which this assumption imposes on the application of learning algorithms. A hypothesis which has a small expected loss can be guaranteed to incur only little loss, on average, in the future, provided that it is fed with input that is drawn with respect to this very same distribution. In some applications, this assumption is met at least to some degree. But in many other domains, these assumptions are not reasonable. Think, for instance, of databases with customer transactions. Let us assume that each transaction is one observation. First, transactions which were conducted by the same customer the same day are by no means independent and, second, the distribution over transactions changes over time as old products vanish, new products appear, and some products get advertised. Or think of share prices. The distribution of up-ticks and down-ticks changes dramatically when the national (or global) economic situation changes, rather than being stationary.

There are also situations in which the assumption of a "natural" distribution appears questionable in principle. This is, for instance, the case in scientific inquiry where often a hypothesis is sought that explains a certain area of discourse completely and exactly – not just with respect to a particular distri-

bution of observations. In these cases, the available data is often generated by conducting experiments; and these experiments are guided by scientists rather than according to a "natural" distribution.

The identification in the limit framework, on the other hand, requires no such distributional assumptions. It guarantees that the target is identified exactly eventually. The unboundedness of the learning time constrains the applicability of some algorithms which emerged from this framework, although there are average case complexity analyses for some algorithms (*e.g.*, Shinohara, 1982; Erlebach *et al.*, 1997).

Appendix

A Proof of Theorem 5

In Equation 7.1, I refer to the definition of $H_i^*(S)$ and replace $h_L \in H_i^*(S)$ by $E_S(h_L) = \inf_{h' \in H_i}\{E_S(h')\}$. Following the notational conventions, I then relabel h^* as h because the distribution of h will change during the next equations. Note again that h^* is an ERM hypothesis – *i.e.,* it incurs the least empirical error on S but not necessarily the least true error. In Equation 7.2 I factorize the empirical error. In Equation 7.3, I use Bayes' equation in order to swap the posterior probabilities. As always, S is distributed according to D_{XY} and h is distributed uniformly over H_i. What happens in Equation 7.4 is the following: $\frac{P(a|b)P(c)}{P(a,b)} = \frac{P(a,b),P(c)}{P(b)P(a,b)} = \frac{P(c)}{P(b)}$ where $a = \{E_S(h) = e\}$, $b = \{E_S(h) = \inf\{E_S(h')\}\}$, $c = \{E_D(h) = e_D\}$. The idea of this Equation 7.5 is $P(a, b|c) = P(a|c)P(b|a, c)$. Finally, in Equation 7.6, I state that the empirical error is binomially distributed (with the true error as mean value).

$$
\begin{aligned}
&dP_{\{S,h_L\}}(E_D(h_L) = e_D|m, H_i, D_{XY}, h_L \in h_i^*(S)) \\
&= dP_{\{S,h_L\}}\left(E_D(h_L) = e_D|, H_i, D_{XY}, E_S(h_L) = \inf_{h' \in H_i}\{E_S(h')\}\right) && (7.1)\\
&= \sum_e dP_{\{S,h\}}\left(E_D(h) = e_D|m, D_{XY}, E_S(h) = e, E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}\right) \\
&\quad P_{\{S,h\}}\left(E_S(h) = e|m, H_i, D_{XY}, E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}\right) && (7.2)\\
&= \sum_e P_{\{S,h\}}\left(E_S(h) = e, E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_D(h) = e_D, H_i, m, D_{XY}\right) && (7.3)\\
&\quad \frac{P_{\{S,h\}}\left(E_S(h) = e|E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}, m, H_i, D_{XY}\right) dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{P_{\{S,h\}}(E_S(h) = e, E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|m, H_i, D_{XY})} \\
&= \sum_e P_{\{S,h\}}\left(E_S(h) = e, E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_D(h) = e_D, H_i, m, D_{XY}\right) \\
&\quad \frac{dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{P_{\{S,h\}}(E_S(h) = \inf\{E_S(h^j)\}|H_i, m - D_{XY})} && (7.4)\\
&= \sum_e P_{\{S,h\}}\left(E_S(h) = e|E_D(h) = e_D, m\right) \frac{dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{P_{\{S,h\}}(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|H_i, m, D_{XY})} \\
&\quad P_{\{S,h\}}\left(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_S(h) = e, E_D(h) = e_D, m, H_i, D_{XY}\right) && (7.5)\\
&= \sum_e B[m, e_D](e) \frac{dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{P_{\{S,h\}}(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|H_i, m, D_{XY})}
\end{aligned}
$$

$$P_{\{S,h\}}\left(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\} | E_S(h) = e, E_D(h) = e_D, H_i, m, D_{XY}\right) \tag{7.6}$$

In order to determine the chance of finding a hypothesis with least empirical error by randomly picking a hypothesis I factorize the empirical error (Equation 7.7).

$$\begin{aligned} & P_{\{S,h\}}(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\} | H_i, m, D_{XY}) \\ = & \sum_e P_{\{S,h\}}\left(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\} | E_S(h) = e, H_i, m, D_{XY}\right) \\ & P_{\{S,h\}}(E_S(h) = e | H_i, m, D_{XY}) \end{aligned} \tag{7.7}$$

Exploiting the assumption that the observed empirical errors are independent random events (Assumptions 1 and 2) in Equation 7.8, I state that the least empirical error of all errors in H_i is e iff $|H_i|$ times an error of at least e is observed (we know that one hypothesis, h, does incur e). This equation reflects my perspective of viewing the process of exhausting an hypothesis space as a stochastic process in which perceived empirical error values are outcomes of a random experiment. Under the assumed independence, I write this conjunction of events as the $|H_i| - 1$st power of the individual event (Equation 7.9).

$$\begin{aligned} & P_{\{S,h\}}\left(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\} | E_S(h) = e_S, E_D(h) = e_D, m, H_i, D_{XY}\right) \\ = & P_{\{S,h\}}\left(\bigwedge_{\substack{j=1\ldots|H_i| \\ h_j \sim U[H_i]}} E_S(h_j) \geq e_S | E_S(h) = e_S, E_D(h) = e_D, m, H_i, D_{XY}\right) && (7.8) \\ = & \left(P_{\{S,h,h'\}}(E_S(h') \geq e_S | m, H_i, D_{XY})\right)^{(|H_i|-1)} && (7.9) \\ = & \left(\sum_{e >= e_S} P_{\{S,h'\}}(E_S(h') = e | m, H_i, D_{XY})\right)^{(|H_i|-1)} && (7.10) \end{aligned}$$

In Equations 7.11 and 7.12 I reduce the prior on empirical errors to the prior on true errors.

$$\begin{aligned} & P_{\{S,h'\}}(E_S(h') = e | m, D_{XY}, H_i) && (7.11) \\ = & \int_{e'_D} P_{\{S,h'\}}(E_S(h') = e | E_D(h') = e'_D, m, H_i, D_{XY}) dP_{\{h'\}}(E_D(h') = e'_D | H_i, D_{XY}) \\ = & \int_{e'_D} B[m, e'_D](e) dP_{\{h'\}}(E_D(h') = e'_D | H_i, D_{XY}) && (7.12) \end{aligned}$$

Finally, I eliminated all unknown terms up to $P_{\{h\}}(E_D(h) = e_D | H_i, D_{XY})$. ∎

B Efficient Implementation of Theorem 5

In this Section, I will assemble Corollary 1 and Theorem 5 into a coherent algorithm. I first present the algorithm itself and subsequently give some documenting remarks. However, in order to gain a substantial understanding of the algorithms' details, the reader will need to study the proof of Theorem

5 given in Appendix A. While a straightforward implementation of Corollary 2 and Theorem 6 would incur a computational effort of $O(m^4)$ the algorithm exploits some intimate details of the derivation and calculates the error estimate in $O(m^2)$.

Algorithm expected error analysis based model selection. *Input*: sample S, stratification $\langle H_1, \ldots, H_k \rangle$. *Output:* number i such that the estimated expected error of the ERM hypothesis of H_i is less than the estimated expected error of the ERM hypotheses of the other models.

1. Initialize all variables to zero.
2. **For** all models $i = 1 \ldots k$
   (a) **Draw** a fixed number c of hypotheses from H_i. Measure their empirical error on S.
   (b) **For** $\mathtt{e} = 0 \ldots m$, **Let** $\mathtt{P\_E_S[e]} = \frac{1}{c} \times$ the number of hypotheses which incurred an empirical error of $\frac{\mathtt{e}}{m}$ (from the drawn hypotheses).
   (c) **For** $\mathtt{e} = 0 \ldots m$, **Let** $\mathtt{P\_E_D[e]} = \mathtt{P\_E_S[e]}$ (identify $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and $P_{\{h\}}(E_S(h)|H_i, S)$).
   (d) **For** $\mathtt{e} = 0 \ldots m$, **Let** $\mathtt{P\_h^*\_E_S[e]} = (\sum_{\mathtt{i}=\mathtt{e}}^{m} \mathtt{P\_E_S[e]})^{|H_i|-1}$ (the chance $P(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_S(h) = e)$ that a hypothesis with empirical error $\mathtt{e}$ is an ERM hypothesis).
   (e) **Let** $\mathtt{P\_h^*} = \sum_{\mathtt{e}=0}^{m} \mathtt{P\_h^*\_E_S[e]} \times \mathtt{P\_E_S[e]}$ (chance $P(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\})$ that a randomly drawn hypothesis is in $H_i^*(S)$).
   (f) **For** $\mathtt{e_D} = 0 \ldots m$
      i. **Increment** $\mathtt{Exp\_E_D[i]}$ **by** $\mathtt{e_D} \times (\sum_{\mathtt{e}=0}^{m} B\left[\frac{\mathtt{e_D}}{m}, m\right](\mathtt{e}) \times \mathtt{P\_h^*\_E_S[e]}) \times \frac{\mathtt{P\_E_D[e_D]}}{\mathtt{P\_h^*}}$
   (g) ($\mathtt{Exp\_E_D[i]}$ is now the estimated expected true error of model i).
3. **Return** the i which minimizes $\mathtt{Exp\_E_D[i]}$.

The algorithm runs in $O(m^2)$. In step 2c the algorithm identifies the estimates of $P_{\{h\}}(E_S(h)|H_i, S)$ and $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ – see Section 3.3 for a discussion. In step 2d, the algorithm calculates the probability $P_{\{S,h\}}(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_S(h) = e_S, E_D(h) = e_D, m, D_{XY})$ as described in Equation 7.9 in the Appendix. Note that $P_{\{S,h\}}(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_S(h) = e_S, E_D(h) = e_D, m, D_{XY}) = P_{\{S,h\}}(E_S(h) = \inf_{h' \in H_i}\{E_S(h')\}|E_S(h) = e_S, m, D_{XY})$ follows from assumption 1. Step 2e is an implementation of Equation 7.7. Step 2(f)i jointly implements the sum of Corollary 2 and Equation 7.6 (note that the Equation of Theorem 6 is just an expansion of Equation 7.6). The implementation of the binomial distribution uses Pascal's triangle for small values of m and refers to the normal distribution for large values.

In the actual implementation, the resolution of the true error values is restricted to an arbitrary constant which reduces the time complexity from $O(m^2)$ to $O(k \times m) = O(m)$.

C Proof of Theorem 6

The expected error $\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, D_{XY}, m, h_L \in H_i^*(S))$ can be expressed as the sum over all errors $E_D(h_j)$ times the chance that h_j is selected by the learner.

$$\begin{aligned}
&\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, D_{XY}, m, h_L \in H_i^*(S)) \\
&= \sum_{j=1}^{|H_i|} E_D(h_j) P_{\{S,h_L\}}(h_L = h_j|H_i, D_{XY}, m, h_L \in H_i^*(S)) \qquad (7.13)
\end{aligned}$$

The chance of a hypothesis being selected which is not in $H_i^*(S)$ (the set of ERM hypotheses) is zero (Equation 7.14). Now I factorize the number of ERM hypotheses n (Equation 7.15). The chance of h_j being chosen as h_L when h_j is a minimum error hypothesis is $\frac{1}{n}$ (Equation 7.17). Now I factorize the empirical error e_S of h_j.

$$\begin{aligned}
&P_{\{S,h_L\}}(h_L = h_j|H_i, D_{XY}, m, h_L \in H_i^*(S)) \\
&= P_{\{S,h_L\}}(h_L = h_j|H_i, D_{XY}, m, h_L \in H_i^*(S), h_j \in H_i^*(S)) \\
&\quad P_{\{S,h_L\}}(h_j \in H_i^*(S)|H_i, D_{XY}, m) \qquad (7.14) \\
&= \sum_{n=1}^{|H_i|} P_{\{S,h_L\}}(h_L = h_j||H_i^*(S)| = n, h_j \in H_i^*(S), H_i, D_{XY}, m, h_L \in H_i^*(S)) \qquad (7.15) \\
&\quad P_{\{S\}}(h_j \in H_i^*(S), |H_i^*(S)| = n|H_i, D_{XY}, m) \qquad (7.16) \\
&= \sum_{n=1}^{|H_i|} \frac{1}{n} P_{\{S\}}(h_j \in H_i^*(S), |H_i^*(S)| = n|H_i, D_{XY}, m) \qquad (7.17) \\
&= \sum_{e_S} P_{\{S\}}(E_S(h_j) = e_S|E_D(h_j), H_i, m) \\
&\quad \sum_{n=1}^{|H_i|} \frac{1}{n} P_{\{S\}}(h_j \in H_i^*(S), |H_i^*(S)| = n|E_S(h_j) = e_S, H_i, D_{XY}, m) \qquad (7.18) \\
&= \sum_{e_S} B[E_D(h_j), m](e_S) \\
&\quad \sum_{n=1}^{|H_i|} \frac{1}{n} P_{\{S\}}(h_j \in H_i^*(S), |H_i^*(S)| = n|E_S(h_j) = e_S, H_i, D_{XY}, m) \qquad (7.19)
\end{aligned}$$

The empirical error (given the true error) is binomially distributed, so $P_{\{S\}}(E_S(h_j) = e_S|E_D(h_j), H_i, m)$ in Equation 7.18 equals $B[E_D(h_j), m](e_S)$ in Equation 7.19. Now I need to determine the unknown term $P_{\{S\}}(h_j \in H_i^*(S), |H_i^*(S)| = n|E_S(h_j) = e_S, H_i, D_{XY}, m)$. In Equation 7.20, I factorize all possible $H_i^*(S)$ of size n. When the hypotheses in $H_i^*(S)$ incur an empirical error of e_S, all other hypotheses have to incur a strictly higher error (according to the definition of $H_i^*(S)$). In Equation 7.21, I exploit the independence assumption to resolve the quantifiers.

$$\begin{aligned}
&P_{\{S\}}(h_j \in H_i^*(S), |H_i^*(S)| = n|E_S(h_j) = e_S, H_i, D_{XY}, m) \qquad (7.20) \\
&= \sum_{\substack{H^* \subseteq H_i \\ |H^*|=n}} P(\forall h^* \in H^* . E_S(h^*) = e_S, \forall h \in H_i \setminus H^* . E_S(h) > e_S|E_S(h_j) = e_S, E_D(h^*), E_D(h), H_i, D_{XY}, m) \\
&= \sum_{\substack{H^* \subseteq H_i \\ |H^*|=n}} \prod_{h^* \in H^*} P(E_S(h^*) = e_S|E_S(h_j) = e_S, E_D(h^*), m) \\
&\quad \prod_{h \in H \setminus H^*} P(E_S(h) > e_S|E_S(h_j) = e_S, E_D(h), m) \qquad (7.21)
\end{aligned}$$

$$= \sum_{\substack{H^* \subseteq H_i \setminus \{h_j\}, \\ |H^*|=n-1}} \prod_{h^* \in H^*} B[E_D(h^*), m](e_S) \prod_{h \in H_i \setminus \{h_j\} \setminus H^*} \sum_{e > e_S} B[E_D(h^*), m](e) \tag{7.22}$$

This completes the proof. ∎

D Proof of Theorem 7

Remember that the learner draws a hypothesis from $H_i^*(S)$ under uniform distribution and that assigning error values to individual hypothesis names was a notational trick in the first place. This means, if $E_D(h_i) = E_D(h_j)$ then $P_{\{S\}}(h_L = h_i|H_i, m) = P_{\{S\}}(h_L = h_j|H_i, m)$. Let h_{e_D} be an arbitrary hypothesis with $E_D(h_{e_D}) = e_D$. Then, Equation 7.13 becomes

$$\begin{aligned} &\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S)) \qquad (7.23)\\ &= \int_{e_D} e_D dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY}) P_{\{S,h_L\}}(h_L = h_{e_D}|H_i, D_{XY}, m, h_L \in H_i^*(S)) \end{aligned}$$

Now I have to take care of $P_{\{S,h_L\}}(h_L = h_{e_D}|H_i, D_{XY}, m, h_L \in H_i^*(S))$. I insert Equation 7.19 into Equation 7.23.

$$\begin{aligned} &\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))\\ &= \int_{e_D} e_D dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})\\ &\quad \sum_{e_{min}} \left(\sum_{n=1}^{|H_i|} \frac{1}{n} P_{\{S\}}(|H_i^*(S)| = n|h_{e_D} \in H_i^*(S), E_S(h) = e_{min}, H_i, D_{XY}, m) \right)\\ &\quad P(h_{e_D} \in H_i^*(S)|E_S(h_{e_D}) = e_{min}, H_i, m) B[e_D, m](e_{min}) \qquad (7.24) \end{aligned}$$

Exploiting assumption 3, I can claim that

$$const = \sum_{n=1}^{|H_i|} \frac{1}{n} P(|H_i^*(S)| = n|h \in H_i^*(S), E_S(h) = e_{min}, H_i, D_{XY}, m) \tag{7.25}$$

is constant for all hypotheses h. $P_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$ should integrate to 1. This fixes the scaling factor $const$ to

$$\begin{aligned} const \quad = \quad &\left(\int_{e_D} dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY}) \sum_{e_{min}} B[e_D, m](e_{min}) \right. \qquad (7.26)\\ &\left. P(h_{e_D} \in H_i^*(S)|E_S(h_{e_D}) = e_{min}, H_i, m) \right)^{-1}. \end{aligned}$$

I abbreviate $\sum_{e_{min}} B[e_D, m](e_{min}) P(h_{e_D} \in H_i^*(S)|E_S(h_{e_D}) = e_{min}, H_i, m)$ as $P_{\{S\}}(h_{e_D} \in H_i^*(S)|H_i, m, E_D(h_{e_D})$ and thus arrive at Equation 3.5. Now I will focus on Equation 3.6. Equation 7.27 follows from the straightforward observation that h_{e_D} is in $H_i^*(S)$ iff h_{e_D} incurs an empirical error of e_S and all other h_j incur at least an error of e_S. Note that this equation exploits assumption 1. In Equation 7.28, I group all hypotheses with equal true error ϵ_D' into one factor and take this factor to

the number of hypotheses with that error. Note that one hypothesis (h_{e_D}) has already been assigned an empirical error and is thus not included in the product.

$$P_{\{S\}}(h_{e_D} \in H_i^*(S)|H_i, m, E_D(h_{e_D}))$$

$$= \sum_{e_S} B[e_D, m](e_S) \prod_{\substack{j=1 \\ h_j \neq h_{e_D}}}^{|H_i|} \left(\sum_{e \geq e_S} B[E_D(h_j), m](e) \right) \tag{7.27}$$

$$= \sum_{e_S} B[e_D, m](e_S) \prod_{e'_D} \left(\sum_{e \geq e_S} B[e'_D, m](e) \right)^{|\{h: h \in H \setminus \{h_{e_D}\}, E_D(h) = e'_D\}|} \tag{7.28}$$

Now Equations 7.24, 7.26, and 7.28 can be rewritten as Equation 3.6. This completes the proof.∎

E Efficient Implementation of Theorem 7

I shall now give an efficient algorithm which evaluates Theorem 6 in $O(m^2)$.

Algorithm expected error analysis$_2$. *Input*: sample S, stratification $\langle H_1, \ldots, H_k \rangle$. *Output:* number i such that the estimated expected error of the ERM hypothesis of H_i is less than the estimated expected error of all other ERM hypotheses.

1. Initialize all variables to zero.
2. **For** all models $i = 1 \ldots k$
   (a) **Draw** a fixed number c of hypotheses from H_i. Measure their empirical error on S.
   (b) **For** $\mathtt{e} = 0 \ldots m$, **Let** $\mathtt{P\_E_S[e]} = \frac{1}{c} \times$ the number of hypotheses which incurred an empirical error of $\frac{\mathtt{e}}{m}$ (from the c drawn hypotheses).
   (c) **For** $\mathtt{e} = 0 \ldots m$, **Let** $\mathtt{P\_E_D[e]} = \mathtt{P\_E_S[e]}$ (identify $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ and $P_{\{h\}}(E_S(h)|H_i, S)$).
   (d) **For** $\mathtt{e_D} = 0 \ldots m$, **For** $\mathtt{e_S} = m \ldots 0$, **Let** $\mathtt{P\_E_S\_geq\_e_S\_under\_e_D[eS][eD]} = B[m, \frac{\mathtt{eD}}{\mathtt{m}}](\mathtt{e_S}) + \mathtt{P\_E_S\_geq\_e_S\_under\_e_D[e_S + 1][e_D]}$. (make a table of $P_{\{S\}}(E_S(h) \geq e_S|E_D(h) = e_D)$.)
   (e) **For** $\mathtt{emin} = 0 \ldots m$
      i. **Let** $\mathtt{prod[emin]} = 0$,
      ii. **For** $\mathtt{e_D} = 0 \ldots m$, **Let** $\mathtt{prod[eD]} = \mathtt{prod[e_D]} \times pow(\mathtt{P\_E_S\_geq\_e_S\_under\_e_D[emin][e_D]}, |H_i| \times \mathtt{P\_E_D[e_D]})$.
   (f) **For** $\mathtt{e_D} = 0 \ldots m$
      i. **For** $\mathtt{eS} = 0 \ldots m$, **Increment** $\mathtt{P\_h\_in\_Hstar[eD]}$ by $B[\frac{\mathtt{eD}}{\mathtt{m}}, \mathtt{m}](\mathtt{eS}) \times \mathtt{prod[eS]}$ $(P_{\{S\}}(h \in H_i^*(S)|H_i, m, E_D(h)))$.
   (g) **For** $\mathtt{eD} = 0 \ldots m$
      i. **Increment** numerator by $\frac{\mathtt{e_D}}{\mathtt{m}} \times \mathtt{P\_E_D[e_S]} \times \mathtt{P\_h\_in\_Hstar[e_D]}$.
      ii. **Increment** denominator by $\mathtt{P\_E_D[e_S]} \times \mathtt{P\_h\_in\_Hstar[e_D]}$.
   (h) **Let** $\mathtt{Exp\_E_D[i]} = \frac{\mathtt{numerator}}{\mathtt{denominator}}$ (the estimated expected true error of model i).
3. **Return** the i which minimizes $\mathtt{Exp\_E_D[i]}$.

The algorithm generates some tables to avoid double computations. It runs in $O(m^2)$.

F Proof of Theorem 8

In Equation 7.29, I write the expected error as the integral over all error values; in Equation 7.30, I factorize the empirical error of h_L. Note that $P(a) = \sum_b P(a|b)P(b)$. In Equation 7.31, I apply Bayes Theorem: $P(a|b) = P(b|a)\frac{P(a)}{P(b)}$, where $a = \{E_D(h_L) = e_D\}$, $b = \{E_S(h_L) = e, h_L \in H_i^*(S)\}$, and $c = \{E_S(h) = e\}$. What happens in Equation 7.32 is $\frac{P(a|b)p(c)}{P(a,b)} = \frac{P(a,b),P(c)}{P(b)P(a,b)} = \frac{P(c)}{P(b)}$, where $a = \{E_S(h_L) = e\}$, $b = \{h_L \in H_i^*(S)\}$, and $c = \{E_D(h_L) = e_D\}$. The idea of Equation 7.33 is that $P(a, b|c) = P(a|c)P(b|a, c)$. Here, $a = \{E_S(h_L) = e\}$, $b = \{h_L \in H_i^*(S)\}$, and $c = \{E_D(h_L) = e_D\}$. In Equation 7.34, I state that the empirical error is binomially distributed; I can remove the sum since $P_{\{S,h_L\}}(h_L \in H_i^*(S)|E_S(h_L) = e)$ is zero for $e > 0$ and 1 for $e = 0$ when $|H_i| \to \infty$. In Equation 7.35 I claim that h_L is in $H_i^*(S)$ if and only if it incurs an empirical error rate of zero. Note that every hypothesis with error strictly less than 1 has a nonzero chance of incurring an empirical error rate of 0. As $|H_i|$ approaches infinity, the chance of at least one hypothesis incurring an empirical error of 0 (under assumption 1) approaches 1.

$$
\begin{aligned}
&\lim_{|H_i|\to\infty} \mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S)) \\
&= \lim_{|H_i|\to\infty} \int_{e_D} e_D dP_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, m, , D_{XY} h_L \in H_i^*(S)) && (7.29)\\
&= \lim_{|H_i|\to\infty} \int_{e_D} e_D \sum_e dP_{\{S,h_L\}}(E_D(h_L) = e_D|H_i, m, D_{XY}, h_L \in H_i^*(S), E_S(h_L) = e) \\
&\quad P_{\{S,h_L\}}(E_S(h_L) = e|H_i, D_{XY}, m, h_L \in H_i^*(S)) && (7.30)\\
&= \lim_{|H_i|\to\infty} \int_{e_D} e_D \sum_e P_{\{S,h_L\}}(E_S(h_L) = e, h_L \in H_i^*(S)|E_D(h_L) = e_D, H_i, m, D_{XY}) \\
&\quad \frac{P_{\{S,h_L\}}(E_S(h_L) = e|H_i, D_{XY}, m, h_L \in H_i^*(S)) dP_{\{h_L\}}(E_D(h_L)|H_i, D_{XY})}{P_{\{S,h_L\}}(E_S(h_L) = e, h_L \in H_i^*(S)|H_i, m, D_{XY})} && (7.31)\\
&= \lim_{|H_i|\to\infty} \int_{e_D} e_D \sum_e P_{\{S,h_L\}}(E_S(h_L) = e, h_L \in H_i^*(S)|E_D(h_L) = e_D, H_i, D_{XY}, m) \\
&\quad \frac{dP_{\{h_L\}}(E_D(h_L) = e_D|H_i, D_{XY})}{P_{\{S,h_L\}}(h_L \in H_i^*(S)|H_i, m, D_{XY})} && (7.32)\\
&= \lim_{|H_i|\to\infty} \int_{e_D} e_D \sum_e P_{\{S,h_L\}}(E_S(h_L) = e|H_i, m, D_{XY}, E_D(h_L) = e_D) \\
&\quad \frac{dP_{\{h_L\}}(E_D(h_L) = e_D|H_i, D_{XY})}{P_{\{S,h_L\}}(h_L \in H_i^*(S)|H_i, m, D_{XY})} \\
&\quad P_{\{S,h_L\}}(h_L \in H_i^*(S)|H_i, m, D_{XY}, E_S(h_L) = e, E_D(h_L) = e_D) && (7.33)\\
&= \int_{e_D} e_D B[e_D, m](0) \frac{dP_{\{h_L\}}(E_D(h_L) = e_D|H_i, D_{XY})}{P_{\{S,h_L\}}(h_L \in H_i^*(S)|H_i, m, D_{XY})} && (7.34)\\
&= \frac{\int_{e_D} e_D \times (1 - e_D)^m dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{P_{\{S,h\}}(E_S(h) = 0|H_i, m, D_{XY})} && (7.35)\\
&= \frac{\int_{e_D} e_D \times (1 - e_D)^m dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})}{\int_{e_D} (1 - e_D)^m dP_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})} && (7.36)
\end{aligned}
$$

Note that $B[e_D, m](0) = (1 - e_D)^m$ which completes the proof. ∎

G Expected Learning Curve for Boolean Functions

G.1 Boolean Functions over Attributes $x_1, \ldots, x_i$

In this Section, I focus on $\mathbf{E}_{\{D_{XY},S,h_L\}}(E_D(h_L)|H_i, m, D_{XY}, h_L \in H_i^*(S))$, the expected error of the ERM hypothesis h_L, where D_{XY} is governed by the uniform distribution over Boolean concepts.

Assume that the learning domain is characterized by a prior $P(D_{XY})$ over target distributions. What is now the expected error of h_L? Theorem 6 claims that, when $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is known, knowledge of the exact D_{XY} is not necessary. Each D_{XY} yields some $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. Let $P(P_{\{h\}}(E_D(h)|H_i, D_{XY})|P(D_{XY}))$ be the probability of a particular error prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ under the given distribution of targets $P(D_{XY})$. In order to determine the expected error over all target functions, we integrate over all possible error priors.

$$\begin{aligned}&\mathbf{E}_{\{D_{XY},S,h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S)) \\ &= \int_p \mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, p, m, h_L \in H_i^*(S))dP(p|P(D_{XY}))\end{aligned} \tag{7.37}$$

$\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, p, m, h_L \in H_i^*(S))$ is the expected error of h_L when the error prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is p and is given in Theorem 6. For Boolean functions, this Equation has an explicit solution.

Let $P(D_{XY}) = P(D_{Y|X}D_X)$ be such that D_X is the uniform distribution over all Boolean instances and $D_{Y|X}$ is governed by the uniform distribution over all Boolean concepts over the Boolean variables x_1 through x_n. Let H_i be the set of all Boolean functions over Boolean variables $x_1, \ldots, x_i$. Let S be a sample of size m and let h_L be drawn uniformly from $H_i^*(S)$, the hypotheses with least empirical error. Two cases have to be distinguished.

(1) $i < n$**.** D_{XY} splits the Boolean space into 2^n instances whereas the hypotheses h split the space only into 2^i distinguishable subspaces. Hence, 2^{n-i} Boolean instances with potentially distinct class labels fall into each subspace. However, the hypothesis has to assign one class label to each subspace. Since the target function is uniformly distributed, assigning a class label of 0 will mis-classify a number of instances distributed according to $\epsilon = B[\frac{1}{2}, 2^{n-i}]$ while a class label of 1 will incur an error of $2^{(n-i)} - \epsilon$. Let $\epsilon_1, \ldots, \epsilon_{2^i}$ be the number of instances which are mis-classified when the corresponding subspace (1 through 2^i) is assigned a class label of 0 (the ϵ_j lie between 0 and 2^{n-i}). The ϵ_j are the parameters of the distribution $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. Over all target functions, these parameters are distributed according to

$$P_{\{D_{XY}\}}([\epsilon_1, \ldots, \epsilon_{2^i}]) = B\left[\frac{1}{2}, 2^{n-i}\right](\epsilon_1) \times \ldots \times B\left[\frac{1}{2}, 2^{n-i}\right](\epsilon_{2^i}). \tag{7.38}$$

For a given set of parameters $\epsilon_1, \ldots, \epsilon_{2^i}$, $P_{\{h\}}(E_D(h)|H_i, D_{XY})$, the sum of errors E_j incurred in subspace $j = 1$ through 2^i, is distributed according to

$$P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY}) = P[\epsilon_1, \ldots, \epsilon_{2^i}]\left(\sum_{j=1}^{2^i} E_j = 2^i e_D\right) \tag{7.39}$$

where

$$P[\epsilon_1, \ldots, \epsilon_k]\left(\sum_{j=1}^{k} E_j = e\right) \tag{7.40}$$

$$= \frac{1}{2}P[\epsilon_2,\ldots,\epsilon_{2^i}]\left(\sum_{j=2}^{k} E_j = e - \epsilon_1\right) + \frac{1}{2}P[\epsilon_2,\ldots,\epsilon_{2^i}]\left(\sum_{j=2}^{k} E_j = e - 2^{n-i} + \epsilon_1\right)$$

and

$$P[\epsilon_1](E_1 = e_1) = \begin{cases} 1 & \text{iff } \epsilon_1 = 2^{n-i} - e_1 = e_1 \\ \frac{1}{2} & \text{iff } \epsilon_1 = e_1 \\ \frac{1}{2} & \text{iff } \epsilon_1 = 2^{n-i} - e_1 \\ 0 & \text{otherwise} \end{cases} \tag{7.41}$$

Equation 7.40 is recursive; the intuition of this equation is that an error of e is incurred in subspace 1 through k when *either* an error of ϵ_1 (class label 0) is incurred in subspace 1 and an error of $e - \epsilon_1$ is incurred in subspace 2 through k, *or* an error of $2^{n-i} - \epsilon_1$ is incurred in subspace 1 (class label 1) and the remaining error of $e - (2^{n-i} - \epsilon_1)$ is incurred in subspaces 2 through k.

In this case, $\mathbf{E}_{\{D_{XY},S,h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S))$ takes the form

$$\begin{aligned} &\mathbf{E}_{\{D_{XY},S,h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S)) \\ &= \sum_{(\epsilon_1,\ldots,\epsilon_{2^i})} \mathbf{E}_{\{S,h_L\}}(E_D(h)|H_i, m, P_{\{h\}}(E_D(h)|H_i, D_{XY}), h_L \in H_i^*(S)) \\ &\qquad P_{D_{XY}}([\epsilon_1,\ldots,\epsilon_{2^i}]) \end{aligned} \tag{7.42}$$

where $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ depends on $[\epsilon_1,\ldots,\epsilon_{2^i}]$ and is given by Equation 7.39, $dP_{D_{XY}}([\epsilon_1,\ldots,\epsilon_{2^i}])$ is given by Equation 7.38, and $\mathbf{E}_{\{S,h_L\}}(E_D(h)|H_i, m, P_{\{h\}}(E_D(h)|H_i, D_{XY}), h_L \in H_i^*(S))$ is given by Theorem 6.

$i \geq n$**.** In this case, the target function assigns one class label to 2^{i-n} instances which can be distinguished by the hypothesis. Given a target function D_{XY},

$$P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY}) = B\left[\frac{1}{2}, 2^i\right](2^i e_D). \tag{7.43}$$

What is the distribution of the distributions $P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$ when $P(D_{XY})$ is the uniform distribution over all Boolean functions with n attributes and D_X is the uniform distribution over all Boolean instances? Here, the key observation is that, in this situation, $P_{\{h\}}(E_D(h) = e_D|H_i, D_{XY})$, as given in Equation 7.43, does not depend on the concrete D_{XY}. Hence, $P_{\{h,D_{XY}\}}(E_D(h)|H_i) = P_{\{h\}}(E_D(h)|H_i, D_{XY})$. $\mathbf{E}_{\{D_{XY},S,h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S))$ takes the form

$$\begin{aligned} &\mathbf{E}_{\{D_{XY},S,h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S)) \\ &= \mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, P_{\{h\}}(E_D(h)|H_i, D_{XY}), h_L \in H_i^*(S)) \end{aligned} \tag{7.44}$$

where the error prior $P_{\{h\}}(E_D(h)|H_i, D_{XY})$ is given by Equation 7.43 and $\mathbf{E}_{\{S,h_L\}}(E_D(h_L)|H_i, m, P_{\{h\}}(E_D(h)|H_i, D_{XY}), h_L \in H_i^*(S))$ is given by Theorem 6. Note that the expected error of h_L only depends on n (the number of relevant attributes), i (the number of attributes of model H_i), and the sample size m.

G.2 Expected Learning Curve for Boolean Functions over i Attributes

In this Section, I will discuss the solution to Equation 7.37 when H_i contains all Boolean functions over any i attributes when there are b possible attributes and the target is a Boolean function over

n attributes. By contrast, in the previous section, H_i included only hypotheses over the particular attributes x_1 through x_i. What is the error prior in this situation? Let us first look at how many relevant and how many irrelevant attributes a randomly drawn hypothesis $h \in H_i$ contains. There are n relevant attributes out of a total of b attributes, hence the number of relevant attributes in H_i is hyper-geometrically distributed: $H[b, \frac{n}{b}, i]$. Let us now assume that $i-u$ attributes are relevant and u attributes are irrelevant.

$$P_{\{h,D_{XY}\}}(E_D(h) = e_D|H_i) = \sum_u H\left[b, \frac{n}{b}, i\right](i-u) \times P_{\{h\}}(E_D(h)|n,i,u) \tag{7.45}$$

In the next part, I will have to refer to a particular distribution, which I shall call M_2. $M_2[m; p_1, p_2](x)$ gives the chance of the sum of m random numbers being x, when the chance of each random number being p_1 or p_2 is $\frac{1}{2}$ each. When p_1 equals p_2, the sum $m \times p_1$ occurs with probability 1. Otherwise p_1 has to occur a certain number of times. When p_1 occurs π times (and p_2 occurs $m-\pi$ times), the result is $\pi p_1 + (m-\pi)p_2$. Hence, p_1 has to occur $\frac{x-mp_1}{p_1-p_2}$ times. Therefore, M_2 can then be characterized as follows.

$$M_2[m; p_1, p_1](x) = \begin{cases} 1 & \text{iff } x = m \times p_1 \\ 0 & \text{otherwise} \end{cases} \tag{7.46}$$

$$M_2[m; p_1, p_2](x) = B\left[m, \frac{1}{2}\right]\left(\frac{x-mp_2}{p_1-p_2}\right) \text{ iff } p_1 \neq p_2 \tag{7.47}$$

When $p_1 = 0$ and $p_2 = 1$, M_2 is the binomial distribution with individual probability $\frac{1}{2}$. The relevant attributes split the space of instances into 2^{i-u} subspaces. Each of these subspaces is split into 2^{n-i+u} parts by the target function. A very similar situation has been studied in the previous subsection. The hypothesis will assign one class label to the whole of the 2^{n-i+u} parts. Since the correct class label is uniformly distributed in each part, assigning one class label to all of them will misclassify a number of instances ϵ which is distributed according to $B[2^{n-i+u}, \frac{1}{2}]$. Hence, for a given target function, in each of the 2^{i-u} subspaces only error values of $\frac{\epsilon}{2^{n-i+u}}$ or $\frac{2^{n-i+u}-\epsilon}{2^{n-i+u}}$ are possible. The ϵ_j, $1 \leq j \leq 2^{i-u}$ are, again, the parameters of the prior distribution $P_{\{h\}}(E_D(h)|H_i, D_{XY})$. Each of the 2^{i-u} subspaces (say subspace j) which are generated by the $i-u$ relevant attributes is split into 2^u subspaces by the u irrelevant attributes. Now remember that only error values $\frac{\epsilon_j}{2^{n-i+u}}$ or $\frac{2^{n-i+u}-\epsilon_j}{2^{n-i+u}}$ are possible in subspace j. Assume that whenever the hypothesis assigns class 0 to a subspace, an error of $\frac{\epsilon_j}{2^{n-i+u}}$ occurs, an error of $\frac{2^{n-i+u}-\epsilon_j}{2^{n-i+u}}$ otherwise. How is the number of possible assignments of class labels to the 2^u subspaces which incurs a particular error values distributed? 2^u random experiments with possible outcomes ϵ_j and $2^{n-i+u} - \epsilon_j$ (chance $\frac{1}{2}$ for each possible outcome) are carried out. Hence, the sum of the outputs is governed by the M_2 distribution. $P(E_j = \frac{e}{2^{n-i+2u}}) = M_2[2^u; \epsilon_j, 2^{n-i+u} - \epsilon_j](e)$. Consequently, the chance that the sum of all error values over all 2^{i-u} subspaces generated by the relevant attributes is e_D is

$$P_{\{h\}}(E_D(h) = e_D|n,i,u) = P[\epsilon_1, \ldots, \epsilon_{2^{i-u}}]\left(\sum_{j=1}^{2^{i-u}} E_j = e_D 2^{i-u}\right) \tag{7.48}$$

where

$$P[\epsilon_1, \ldots, \epsilon_k]\left(\sum_{j=1}^{k} E_j = \frac{e}{2^{n-i+2u}}\right)$$

$$= \sum_{e_j} M_2\left[2^u; \epsilon_1, 2^{n-i+u} - \epsilon_1\right](e_j) P[\epsilon_2, \ldots, \epsilon_k]\left(\sum_{j=2}^{k} E_j = \frac{e - e_j}{2^{n-i+2u}}\right) \quad (7.49)$$

and

$$P[\epsilon_1]\left(E_j = \frac{e}{2^{n-i+2u}}\right) = M_2\left[2^u; \epsilon_1, 2^{n-i+u} - \epsilon_1\right](e) \quad (7.50)$$

The distribution which governs the parameters ϵ_1 through $\epsilon_{2^{i-u}}$ of the error prior is analogous to Equation 7.38.

$$P([\epsilon_1, \ldots, \epsilon_{2^{i-u}}]) = B\left[\frac{1}{2}, 2^{n-i+u}\right](\epsilon_1) \times \ldots \times B\left[\frac{1}{2}, 2^{n-i+u}\right](\epsilon_{2^{i-u}}) \quad (7.51)$$

Now the parametric error prior and the prior distribution of the parameters have been determined and can be "plugged" into Theorem 6. When $P(D_{XY})$ is the uniform distribution over Boolean functions with n attributes and H_i contains all Boolean functions over i attributes (when there are b possible attributes), then

$$\begin{aligned}
&\mathbf{E}_{\{S, D_{XY}, h_L\}}(E_D(h_L)|H_i, m, h_L \in H_i^*(S)) \\
&= \sum_{u=0}^{i} \sum_{(\epsilon_1, \ldots, \epsilon_{2^{i-u}})} \mathbf{E}_{\{S, h_L\}}(E_D(h_L)|H_i, m, P_{\{h\}}(E_D(h)|n, i, u), h_L \in H_i^*(S)) \\
&\qquad P([\epsilon_1, \ldots, \epsilon_{2^{i-u}}]) H\left[b, \frac{n}{b}, i\right](i - u).
\end{aligned} \quad (7.52)$$

This completes the derivation.

H Notation

In this Section, I provide a quick reference table for the notation used throughout the book.

$B[p, n](x)$. Binomial distribution density. $B[p, n](x)$ is the chance of achieving x "hits" on n trials when the chance of a hit is p for each trial and when the trials are independent. $B[p, n](x) = \binom{n}{x} p^x (1-p)^{n-x}$.

BC. A learner identifies the target function behaviorently correct (the corresponding learnability class is abbreviated BC) if it makes only finitely many erroneous predictions and, from some finite point on, keeps making true predictions although it may still change its conjecture about the target function.

D_{XY}. Target distribution. $D_{XY} = D_{Y|X} D_X$ where $D_{Y|X}(y|x)$ is the chance of class y for instance x and $D_X(x)$ is the probability (mass) of instance x. See Section 2.1 for the definition.

$E_D(h)$**.** True error (with respect to target distribution D_{XY}) of hypothesis h. See Section 2.1 for the definition.

ERM hypothesis. The set of hypotheses $H_i^*(S)$ with least empirical error (with respect to sample S) are called the ERM hypotheses. A single hypothesis which is drawn from this set under uniform distribution is an ERM hypothesis.

$E_S(h)$. Empirical error of hypothesis h on sample S. See Section 2.1 for the definition.

EX. The class of explanatory learnable languages. A language class is explanatory learnable if there exists a learner that identifies each language of that class with certainty eventually. After each new example the learner outputs a new hypothesis; the learner may change its mind finitely often but no bound on the required number of example is needed.

$\mathbf{E}_{\{x\}}(f(x))$. The expectation of $f(x)$ over all x, distributed according to $P(x)$ (which is usually given in the context).

FIN. The class of finitely identifiable languages. A learner identifies a language class finitely if it outputs one single correct hypothesis after finitely many examples and does not change its mind thereafter.

h. Arbitrary hypothesis. Usually refers to hypotheses which are drawn at random under uniform distribution in Chapter 3.

H_i. i'th model. A model is a set of hypotheses $h \in H_i$. In Chapter 3, these models are assumed to be finite.

$H_i^*(S)$. The set of hypotheses of H_i with least empirical error on S.

h_L. Hypothesis which is returned by learner L.

k-**CNF**(n). The class of all conjunctive normal forms (conjunctions of disjunctions) over n Boolean attributes where each disjunction has up to k literals.

k-**DL(n).** The class of all decision lists over n Boolean attributes where each rule consists of a body of at most k literals and a head which is labeled with a class symbol. The first rule which fires for some instance assigns the class label to the instance. See Section 3.8.

k-**DNF(n).** The class of all disjunctive normal forms (disjunctions of conjunctions) over n Boolean attributes where each conjunction has up to k literals.

$L_{H_i}(S)$. The result of a deterministic learner L which operates on model H_i when given sample S is a particular hypothesis.

$P_L(h_L|H_i, S)$. The chance of learner L returning hypothesis h_L when operating on model H_i and the sample is S.

$P_{\{x\}}(f(x) = y)$. The probability of drawing an x such that $f(x) = y$.

X. The set of instances.

Y. A finite set of class labels.

Bibliography

Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. In *ACM SIGMOD Conference on Management of Data*, pp. 207–216.

Angluin, D. (1980a). Inductive inference of formal languages from positive data. *Inform. Control*, *45*(2), 117–135.

Angluin, D. (1980b). Inductive inference of formal languages from positive data. *Information and Control*, *45*, 117–135.

Angluin, D. (1993). Learning with queries. In Baum, E. (Ed.), *Computational Learning and Cognition*, pp. 1–28. SIAM.

Angluin, D., & Smith, C. (1983). Inductive inference: Theory and methods. *Computing Surveys*, *15*(3), 237–269.

Atteson (1991). An overview of the minimum description length principle. In *CSSS: 1989 Lectures in Complex Systems, The Proceedings of the 1989 Complex Systems Summer School*. Lectures Volumes II and III, Santa Fe Institute/Studies in the Sciences of Complexity, Addison-Wesley.

Auer, P., Holte, R. C., & Maass, W. (1995). Theory and applications of agnostic PAC-learning with small decision trees. In *12th ICML*, pp. 21–29.

Barzdin, J. (1974). Two theorems on the limiting synthesis of functions. *Latv. Gos. Univ. Uch. Zapiski*, *210*, 82–88. (in russian).

Baxter, J. (1997a). A Bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, *28*, 7–39.

Baxter, J. (1997b). A model of bias learning. Submitted for JACM.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances.. *Phil. Trans. of the Royal Soc. of London*, *53*, 370–418.

Berger, J. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag.

Berger, J. (1993). An overview of robust Bayesian analysis. Tech. rep. 93-53C, Department of Statistics, Purdue University.

Berger, J., & Berliner, L. (1986). Robust Bayes and empirical Bayes analysis with ε-contaminated priors. *Annals of Statistics*, *14*, 461–486.

Blum, A., & Rivest, R. (1992). Training a 3-node neural network is NP-complete. *Neural Networks*, *5*, 117–127.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. (1987). Occam's razor. *Information processing Letters*, *24*, 377–380.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, *36*(4), 929–965.

Bose, S. (1993). Bayesian robustness with mixture classe of priors. Tech. rep. 93-1, Dept. of Statistics, George Washington University.

Bose, S. (1994). Bayesian robustness with more than one class of contaminations. *Journal of Statistical Planning and Inference*.

Breiman, L. (1996). Bias, variance, and arcing classifiers. Tech. rep. 460, Statistics Department, University of California at Berkeley.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and Regression Trees*. Pacific Grove.

Breiman, L. (1996). Stacked regressions. *Machine Learning*, *24*, 49–64.

Bshouty, N., Cleve, R., Kannon, S., & Tamon, C. (1994). Oracles and queries that are sufficient for exact learning. In *Proc. 7th Annual ACM Workshop on Computational Learning Theory*, pp. 130–139.

Cano, J., Hernandez, A., & Moreno, E. (1985). Posterior measures under partial prior information. *Statistica*, *2*, 219–230.

Carroll, R., & Hall, P. (1988). An alternative method of cross validation for the smoothing of density estimates. *Journal of the American Statistical Association*, *83*, 1184–1186.

Case, J., & Smith, C. (1983). Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, *25*, 193–220.

Cohen, P. R. (1995). *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273–297.

Cun, Y. L., Denker, J., & Solla, S. (1989). Optimal brain damage. In *NIPS-89*, pp. 598–605.

de Finetti, B. (1972). *Probability, Induction, and Statistics*. Wiley, New York.

DeRobertis, L. (1978). *The use of partial prior knowledge in Bayesian inference*. Ph.D. thesis, Yale University, New Haven.

Dietterich, T., & Kong, E. (1995). Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Tech. rep., Department of Computer Science, Oregon State University.

Dietterich, T. (1997). Statistical tests for comparing supervised classification learning algorithms. unpublished manuscript (submitted).

Dobkin, D., Gunopoulos, D., & Kasif, S. (1996). Computing optimal shallow decision trees. In *Proc. International Workshop on Mathematics in Artificial Intelligence*.

Dobkin, D., Gunopoulous, D., Fulton, T., Kasif, S., & Salzberg, S. (1997). Induction of shallow decision trees. *IEEE Trans. on Pattern Analysis and Machine Intelligence*. in review.

Dolsak, B., & Muggleton, S. (1992). The application of inductive logic programming to finite-element mesh design. In Muggleton, S. (Ed.), *Inductive Logic Programming*, pp. 453–472. Academic Press.

Domingos, P. (1998). A process-oriented heuristic for model selection. In *ICML-98*, pp. 127–135.

Drucker, H., Schapire, R., & Simard, P. (1993). Improving performance in neural networks using a boosting algorithm. In Hanson, S. J., Cowan, J. D., & Giles, C. L. (Eds.), *Advances in Neural Information Processing Systems*, Vol. 5, pp. 42–49. Morgan Kaufmann.

Efron, B. (1979). Bootstrap methods: another look at the jacknife. *Annals of Statistics*, *7*(1), 1–26.

Efron, B. (1983). Estimating the error rate of a prediction rule. *Journal of the American Statistical Association*, *68*(382), 316–330.

Eggermont, P., & LaRicca, V. (1995). Maximum smoothed likelihood density estimation for inverse problems. *Annals of Statistics*, *23*, 199–220.

Ehrenfeucht, A., Haussler, D., Kearns, M., & Valiant, L. G. (1989). A general lower bound on the number of examples needed for learning. *Information and Computation*, *82*, 247–261. First appeared in Proc. 1st Annu. Workshop on Comput. Learning Theory, 1988.

Erlebach, T., Rossmanith, P., Stadtherr, H., Steger, A., & Zeugmann, T. (1997). Learning one-variable pattern languages very efficiently on average, in parallel, and by asking queries. In *Algorithmic Learning Theory: Eighth International Workshop (ALT '97)*, Vol. 1316 of *Lecture Notes in Artificial Intelligence*, pp. 260–276.

Fayyad, U., Piatetski-Shapiro, G., & Smyth, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. In *KDD-96*.

Fisher, D., & Schlimmer, J. (1988). Concept simplification and prediction accuracy. In *ICML-88*, pp. 22–28.

Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Ann. Eugenics*, *7*, 111–132.

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*.

Freund, Y., & Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139.

Gaines, B. R., & Compton, P. J. (1992). Induction of ripple down rules. *5th Australian Conference on Artificial Intelligence*, 349–355.

Gasarch, W., & Smith, C. (1988). Learning via queries. In *Proc. 29th Annual Symposium on Foundactions of Computer Science (FOCS)*.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation, 4*, 1–48.

Gold, E. M. (1967). Language identification in the limit. *Information and Control, 10*, 447–474.

Good, I. (1983). *Good Thinking: The Foundations of Probability Theory and its Applications*. University of Minnesota Press, Minneapolis.

Goutis, C. (1991). Ranges of posterior measures for some classes of priors wioth specified moments. Research report 70, Dept. of Statistical Science, University College of London.

Goutis, C. (1997). Nonparametric estimation of a mixing density via the kernel method. *Journal of the American Statistical Association, 92*(440), 1445–1450.

Greiner, R. (1996). PALO: A probabilistic hill-climbing algorithm. *Artificial Intelligence, 83*(1–2).

Grünwald, P. (1996). A minimum description length approach to grammar inference. In Wermter, S., Riloff, E., & Scheler, G. (Eds.), *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, Vol. 1040 of *LNAI*, pp. 203–216. Springer Verlag, Berlin.

Hartigan, J. (1983). *Bayes Theory*. Springer-Verlag, New York.

Haussler, D. (1988). Quantifying inductive bias : AI learning algorithms and valiant's learning framework. *Artificial Intelligence, 36*(2).

Haussler, D. (1989). Learning conjunctive concepts in structural domains. *Machine Learning, 4*(1), 7–40.

Haussler, D., Kearns, M., & Schapire, R. (1994). Bounds on the sample complexity of Bayesian learning using information theory and the VC dimension. *Machine Learning, 14*(1), 83–113.

Haussler, D. (1992). Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation, 100*(1), 78–150.

Highleyman, W. (1962). The design and analysis of pattern recognition experiments. *Bell Syst. Tech. Journal, 41*, 723–744.

Höffgen, K., Simon, H., & Horn, K. V. (1995). Robust trainability of single neurons. *Journal of Computer and System Sciences, 50*(1), 114–125.

Hoffmann, A., & Scheffer, T. (1998). On the hidden assumptions of model selection algorithms. Submitted.

Holsheimer, M., & Siebes, A. (1991). Data mining. the search for knowledge in databases. Tech. rep. CS-R9406, CWI Amsterdam, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands.

Jain, R., Kasturi, R., & Schunck, B. (1997). *Machine Vision*. McGraw Hill.

Jain, S., Osherson, D., Royer, J., Sharma, A., & Weinstein, S. (1999). *Systems That Learn*. MIT Press.

Jain, S., & Sharma, A. (1990). Language learning by a team. In *Automata, Languages, and Programming*, pp. 153–166.

Jeffreys, J. (1961). *Theory of Probability*. Oxford University Press.

Joachims, T. (1998). Text categorization with support vector machines. In *Proceedings of the European Conference on Machine Learning*.

Joachims, T., Freitag, D., & Mitchell, T. (1997). WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pp. 770–777 San Francisco. Morgan Kaufmann Publishers.

Kang, B., Compton, P., & Preston, P. (1995). Multiple classification ripple down rules: evaluation and possibilities. In Gaines, B., & Musen, M. (Eds.), *Proc. 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop*, pp. 17.1–17.20.

Kearns, M., Mansour, Y., Ng, A., & Ron, D. (1997). An experimental and theoretical comparison of model selection methods. *Machine Learning Journal*, *27*, 7–50.

Kearns, M., & Pitt, L. (1989). A polynomial-time algorithm for learning k-variable pattern languages. In *COLT-89*.

Kearns, M., Schapire, R., & Sellie, L. (1992). Towards efficient agnostic learning. In *Int. Conference on Computational Learning Theory*, pp. 341–352.

Kearns, M., & Vazirani, U. (1994). *An Introduction to Computational Learning Theory*. MIT Press.

Kearns, M. (1996). A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split. In *Advances in Neural Information Processing Systems*, Vol. 8, pp. 183–189.

Kietz, J.-U., & Dzeroski, S. (1994). Inductive logic programming and learnability. *SIGART Bulletin*, *5*(1), 22–32.

Kobler, & Lindner (1997). Oracles in sigma$_2^p$ are sufficient for exact learning. In *Proc. International Workshop on Algorithmic Learning Theory*.

Kohavi, R. (1995). *Wrappers for performance enhancement and oblivious decision graphs*. Ph.D. thesis, Stanford University.

Kohavi, R., & John, G. (1995). Automatic parameter selection by minimizing expected error. In *ICML-95*.

Kohavi, R., & John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*(1-2), 245–271.

Kohavi, R., & Wolpert, D. (1996). Bias plus variance decomposition for zero-one loss functions. In *ICML-96*.

Kong, E., & Dietterich, T. (1995). Error-correcting output coding corrects bias and variance. In *ICML-95*, pp. 313–321.

Kononenko, I. (1993). Inductive and Bayesian learning in medical domains. *Applied Artificial Intelligence*, *7*, 740–741.

Koza, J. R., & Rice, J. P. (1991). Genetic generation of both the weights and architecture for a neural network. In *International Joint Conference on Neural Networks, IJCNN-91*, Vol. II, pp. 397–404.

Lachenbruch, P. (1967). An almost unbiased method of obtaining confidence intervals for the probability of misclassification in discriminant analysis. *Biometrics, 23*, 639–645.

Laird, N. (1978). Nonparametric maximum likelihood estimation of a mixing distribution. *Journal of the American Statistical Association, 73*, 805–811.

Lang, K. (1995). Newsweeder: learning to filter netnews. In *ICML-95*, pp. 331–339.

Lange, S., & Zeugmann, T. (1993). Monotonic versus non-monotonic language learning. In *Proceedings of the Second International Workshop on Nonmonotonic and Inductive Logic*, Vol. 659 of *Lecture Notes in Artificial Intelligence*, pp. 254–269. Springer-Verlag.

Lavine, M. (1992). A note on bounding Monte Carlo variances. *Communications of Statistics – Theory and Methods, 21*(10), 2855–2860.

Lavrac, N., & Džeroski, S. (1994). *Inductive Logic Programming*. Ellis Horwood.

Lee, K. (1989). *Automatic Speech Recognition: The Development of the Sphinx System*. Kluwer Academic Publishers.

Lewis, D. (1991). *Representation and learning in information retrieval*. Phd thesis, University of Massachusetts.

Lindner, R. (1972). *Algorithmische Erkennung*. Ph.D. thesis, Friedrich-Schiller-Universität, Jena.

Liu, M., & Taylor, R. (1989). A consistent nonparametric density estimator for the deconvolution problem. *Canadian Journal of Statistics, 17*, 427–438.

McAllester, D. (1998). Some PAC-Bayesian theorems. In *COLT-98*, pp. 230–234. Morgan Kaufmann.

Mehta, M., Rissanen, J., & Agrawal, R. (1995). MDL-based decision tree pruning. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD'95)*, pp. 216–221.

Michalski, R. S., Mozetič, I., Hong, J., & Lavrač, N. (1986). The multi-purpose incremental learning system AQ15 and its testing application on three medical domains. In *Proc. Fifth National Conference on Artificial Intelligence*, pp. 1041–1045 San Mateo, CA. Morgan Kaufmann.

Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood.

Milne, L. (1997). Attribute selection in neural networks used to classify remotely sensed data. In *Proc. Visual Information Processing Workshop* Sydney.

Mingers, J. (1989). An empirical comparison of selection measures for decision-tree induction. *Machine Learning, 3*, 319–342.

Mitchell, A. (1998). Learnability of a subclass of extended pattern languages. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*. ACM Press.

Mitchell, A., Scheffer, T., & Sharma, A. (1998). PAC learnability of subclasses of pattern languages. Unpublished manuscript.

Mitchell, T. (1997). *Machine Learning*. McGraw Hill.

Mitchell, T. M. (1982). Generalization as search. *AI Journal*, *18*, 203–226.

Moody, J. (1992). The effective number of parameters: An analysis of generalization and regularization in non-linear learning systems. In *NIPS-4*.

Moody, J., & Utans, J. (1992). Principled architecture selection for neural networks. In *NIPS-4*.

Moreno, E., & Cano, J. (1991). Robust Bayesian analysis for ε-contaminations partially known. *Journal of the Royal Statistical Society Series B*, *53*, 143–155.

Mosier, C. (1951). Problems and designs of cross validation. *Educational and Psychological Measurement*, *11*, 5–11.

Mosteller, F., & Tukey, J. (1968). Data analysis, including statistics. In Lindzey, G., & Aronson, E. (Eds.), *Revised Handbook of Social Psychology*, Vol. 2, pp. 80–203. Addison Wesley.

Muggleton, S. (1992). *Inductive Logic Programming*. Volume 38 of A.P.I.C. series. Academic Press Ltd., London.

Müller, K.-R., Finke, M., Schulten, K., Murata, N., & Amari, S. (1995). A numerical study on learning curves in stochastic multi-layer feed-forward networks. *Neural Computation*, *8*, 1085–1106.

Müller, W., & Wysotzki, F. (1992). Automatic construction of decision trees for classification. *Annalen für Operations Research*.

Müller, W., & Wysotzki, F. (1995). Automatic synthesis of control programs by combination of learning and problem solving methods (extended abstract). In Lavrač, N., & Wrobel, S. (Eds.), *Machine Learning: ECML-95*, LNAI 912, pp. 323 – 326. Springer Verlag.

Murphy, P., & Aha, D. (1998). The UCI repository of machine learning databases. http://www.ics.uci.edu/˜mlearn/MLRepository.

Musial, M., & Scheffer, T. (1994). A term-based genetic code for artificial neural networks. In Hopf, J. (Ed.), *Genetic Algorithms within the Framework of Evolutionary Computation*. Max-Planck-Institut für Informatik.

Neyman, J. (1967). *A Selection of Early Statistical Papers by J. Neyman*. University of California Press, Berkeley.

Ng, A. (1997). Preventing overfitting of cross validation data. In *Proc. Int. Conference on Machine Learning*, pp. 245–253. Morgan Kaufmann.

Ng, A. (1998). On feature selection: Learning with exponentially many irrelevant features. In *ICML-98*, pp. 404–412.

Nilsson, N. (1998). *Introduction to Machine Learning*. MIT Press (to apprear).

Norman, W. (1965). Double-split cross validation: an extension of Mosier's design, two undesirable alternatives, and some enigmatic results. *Journal of Applied Psychology*, *49*, 248–257.

Oliver, J., & Baxter, R. (1994). MDL and MML: similarities and differences. TR 207, Dept. of Computer Science, Monash University, Clayton, Victoria 3168, Australia. Available on the WWW from `http://www.cs.monash.edu.au/~jono`.

Oliver, J., R.A., B., & C.S., W. (1996). Unsupervised Learning using MML. In *Machine Learning: Proceedings of the Thirteenth International Conference (ICML 96)*, pp. 364–372. Morgan Kaufmann Publishers, San Francisco, CA. Available on the WWW from `http://www.cs.monash.edu.au/~jon o`.

Pitt, L., & Valiant, L. (1988). Computational limitations of learning from examples. *J. ACM*, *35*(4), 965–984.

Pomerleau, D. (1989). ALVIN: An autonomus land vehicle in a neural network. Technical report CMU-CS-89-107, Carnegie Mellon University, Pittsburgh.

Quinlan, J. R. (1990). Learning logical definitions from releations. *Machine Learning*, *5*, 239–266.

Quinlan, J. R. (1993). *C4.5 Programs for Machine Learning*. Morgan Kaufmann Publisher.

Quinlan, J. R., & Rivest, R. L. (1987). Inferring decision trees using the minimum description length principle. Technical memo MIT/LCS/TM-339, Massachusetts Institute of Technology, Laboratory for Computer Science.

Quinlan, R. (Ed.). (1989). *Applications of Expert Systems*. Addison Wesley.

Quinlan, R. (1996a). Bagging, boosting, and C4. 5. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pp. 725–730 Menlo Park. AAAI Press / MIT Press.

Quinlan, R. (1996b). Boosting first-order learning. In *Proc. International Workshop on Algorithmic Learning Theory*, pp. 143–155 Sydney.

Richter, K., & et al., F. W. (1974). Thorax-Schirmbildaufnahmen als Screeningmethode für Herz-Kreislauf-Erkrankungen. *Rad.Diagn.*

Rissanen, J. (1978). Modelling by shortest data descriptions. *Automatica*, *14*, 465–471.

Rissanen, J. (1985). Minimum-description-length principle. *Ann. Statist.*, *6*, 461–464.

Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquery*. World Scientific.

Rivest, R. L. (1987). Learning decision lists. *Machine Learning*, *2*(2), 229–246.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.*, *65*, 386–407. (Reprinted in *Neurocomputing* (MIT Press, 1988).).

Rummelhart, D. E., & Hinton, G. E. . J. W. (1986). Learning internal representations by error propagation. In Rummelhart, D. E., & McClelland, J. L. (Eds.), *Parallel Distributed Processing*, Vol. 1, pp. 318–362. MIT Press Cambridge MA.

Salomaa, A. (1994a). Patterns (The Formal Language Theory Column). *EATCS Bulletin*, *54*, 46–62.

Salomaa, A. (1994b). Return to patterns (The Formal Language Theory Column). *EATCS Bulletin*, *55*, 144–157.

Salton, G., & Ruckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Managament, 24*(5), 513–523.

Sammut, C., Hurst, S., Kedzier, D., & Michie, D. (1992). Learning to fly. In *Proceedings of the Ninth International Conference on Machine Learning* Aberdeen. Morgan Kaufmann.

Schafer, G. (1981). Constructive probability. *Synthese, 48*, 1–60.

Schaffer, C. (1993a). Overfitting avoidance as bias. *Machine Learning, 10*, 153–178.

Schaffer, C. (1993b). Selecting a classification method by cross-validation. *Machine Learning, 13*, 135–143.

Schaffer, C. (1994). A conservation law for generalization performance. In *ICML-94*, pp. 259–265.

Schapire, R., Freund, Y., Bartlett, P., & Lee, W. (1997). Boosting the margin: A new explanation for the effectiveness of voting methods. In *Machine Learning: Proceedings of the Fourteenth International Conference*, pp. 322–330.

Schapire, R. E. (1990). Pattern languages are not learnable. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory*, pp. 122–129 San Mateo, CA. Morgan Kaufmann.

Scheffer, T. (1996). Algebraic foundation and improved methods of induction of ripple down rules. In *Proc. Pacific Knowledge Acquisition Workshop*.

Scheffer, T., Greiner, R., & Darken, C. (1997). Why experimentation can be better than perfect guidance. In *Proc. 14th International Conference on Machine Learning*, pp. 331–339. Morgan Kaufmann.

Scheffer, T., & Herbrich, R. (1997). Unbiased assessment of learning algorithms. In *IJCAI-97*, pp. 798–803.

Scheffer, T., Herbrich, R., & Wysotzki, F. (1996). Efficient theta-subsumption based on graph algorithms. In *Inductive Logic Programming, 6th International Workshop, Selected Papers*. Springer-Verlag.

Scheffer, T., & Joachims, T. (1998a). Estimating the expected error of empirical minimizers for model selection (abstract). AAAI-98.

Scheffer, T., & Joachims, T. (1998b). Estimating the expected error of empirical minimizers for model selection. Tech. rep. TR 98-9, Technische Universitaet Berlin.

Scheffer, T., & Joachims, T. (1999a). Expected error analysis for model selection. In *Proceedings of the International Conference on Machine Learning (ICML-99)*.

Scheffer, T., & Joachims, T. (1999b). Expected error analysis for model selection. Preprint, TU Berlin. Available at http:ki.cs.tu-berlin.de/˜scheffer.

Scheffer, T., & Stephan, F. (1998). A worst-case analysis of boosting. unpublished.

Schulmeister, B., & Wysotzki, F. (1994). The piecewise linear classifier DIPOL92. In Bergadano, F., & Raedt, L. D. (Eds.), *Machine Learning: ECML-94*, LNAI 784. Springer Verlag.

Schuurmans, D. (1997). A new metric-based approach to model selection. In *AAAI-97*.

Schuurmans, D., Ungar, L., & Foster, D. (1997). Characterizing the generalization performance of model selection strategies. In *ICML-97*, pp. 340–348.

Shinohara, T., & Arikawa, A. (1995). Pattern inference. In Jantke, K. P., & Lange, S. (Eds.), *Algorithmic Learning for Knowledge-Based Systems*, Vol. 961 of *Lecture Notes in Artificial Intelligence*, pp. 259–291. Springer-Verlag.

Shinohara, T. (1982). Polynomial time inference of extended regular pattern languages. In Eiichi Goto, Koichi Furukawa, Reiji Nakajima, Ikuo Nakata, A. Y. (Ed.), *Proceedings of the RIMS Symposia on Software Science and Engineering*, Vol. 147 of *LNCS*, pp. 115–127 Kyoto, Japan. Springer.

Simon, H. A. (1983). Why should machines learn?. In Michalski, R., Carbonnel, J., & Mitchell, T. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, pp. 25–37. Tioga, Palo Alto, CA.

Sivaganesan, S. (1994). Bounds on posterior expectations for density bounded classes with constant bandwidth (with discussion). *Journal of Statistical Planning and Inference*.

Sivaganesan, S., & Berger, J. (1993). Robust analysis of the binomial empirical Bayes problem. *Canadian Journal of Statistics, 21*, 107–119.

Smith, C. (1947). Some examples of discrimination. *Ann. Eugen., 18*, 272.

Smith, C. (1994). Three decades of team learning. In Arikawa, S., & Jandtke, K. (Eds.), *Proc. of the 4th International Workshop on Analogical and Inductive Inference and 5th International Workshop on Algorithmic learning Theory*, pp. 211–228.

Stephan, F. (1998). Learning via queries and oracles. *Journal of Pure and Applied Logic*, *94*(1-3).

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, B 36*, 111–147.

Sutton, R., & Barto, A. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, *8*, 257.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, *38*(3), 58–68.

Tishby, N. (1995). Statistical physics models of supervised learning. In Wolpert, D. (Ed.), *The Mathematics of Generalization*. Addison-Wesley.

Toivonen, H. (1996). Sampling large databases for association rules. In *Proc. VLDB Conference*.

Toussaint, G. (1974). Bibliography on estimation of misclassification. *IEEE Transactions on Information Theory IT20, 4*, 472–479.

Ulbrich, P., & Wysotzki, F. (1972). Metaalgoritmen zur Konstruktion von Klassifizierungsalgorithmen und ihre Simulation auf Rechenautomaten. *Kybernetik-Forschung, 1*.

Unger, S., & Wysotzki, F. (1981). *Lernfähige Klassifizierungssysteme*. Akademie Verlag Berlin.

Valiant, L. (1985). Learning disjunctions of conjunctions. In *Proc. 9th IJCAI*, Vol. 1, pp. 560–566. Morgan Kaufmann.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM, 27*.

Vapnik, V. (1982). *Estimation of Dependencies Based on Empirical Data*. Springer.

Vapnik, V. (1996). *The Nature of Statistical Learning Theory*. Springer.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley.

Vapnik, V. N., & Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications, 16*(2), 264–280.

Vardi, Y., & Lee, D. (1993). From image deblurring to optimal investments: Maximum likelihood solutions for positive linear inverse problems. *Journal of the Royal Statistical Society B, 55*, 569–612.

Vidyasagar, M. (1997). *A Theory of Learning and Generalization*. Springer.

Wapnik, W., & Tscherwonenkis, A. (1979). *Theorie der Zeichenerkennung*. Akademie Verlag, Berlin.

Wexler, K., & Culicover, P. (1980). *Formal Principles of Language Acquisition*. MIT Press, Cambridge, MA.

Wolpert, D. (1993). On overfitting avoidance as bias. Working paper 93-03-016, The Santa Fe Institute.

Wolpert, D. H. (1995). The relationship between PAC, the statistical physics framework, the Bayesian framework, and the VC framework. In Wolpert, D. H. (Ed.), *The Mathematics of Generalization*, The SFI Studies in the Sciences of Complexity, pp. 117–214. Addison-Wesley.

Wolpert, D. (1992). On the connection between in-sample testing and generalization error. *Complex Systems, 6*(1), 47.

Yang, Y., & Petersen, J. (1997). A comparative study on feature selection in text categorization. In *ICML-97*.