Metalearning - A Tutorial

Christophe Giraud-Carrier

December 2008

Contents

Chapter 1

Introduction

This tutorial is about metalearning. The primary goal of metalearning is the understanding of the interaction between the mechanism of learning and the concrete contexts in which that mechanism is applicable. Metalearning differs from base-learning in the scope of the level of adaptation. Whereas learning at the base level focuses on accumulating experience on a specific learning task (e.g., credit rating, medical diagnosis, mine-rock discrimination, fraud detection, etc.), learning at the metalevel is concerned with accumulating experience on the performance of multiple applications of a learning system.

In order to "cross the chiasm" and allow machine learning and data mining algorithms to be more widely used outside research labs, our community must be able to design robust systems that offer support to practitioners. Current machine learning/data mining tools are only as powerful/useful as their users. One main aim of current research in the community is to develop metalearning assistants, able to deal with the increasing number of models and techniques, and give advice dynamically on such issues as model selection and method combination.

Over the past few years, much work has been done in the area, scattered across journals and conferences. This tutorial is an attempt at describing the state-of-the-art and highlighting avenues of future work in metalearning. We recently published a book on metalearning that participants interested in pursuing research in this exciting area may consider acquiring [16].

1.1 Working Definition

Over the past couple of decades, we (the machine learning community) have been relatively successful at prescribing our medicine (machine learning) to an ever increasing number of professionals across a wide range of industries. However, somewhat paradoxically, we have been suffering from "The Shoemaker's Children" syndrome. In other words, we have failed to take advantage of the medicine ourselves; we have not generally considered the possibility of using machine learning to address some of the problems we face in machine learning.

As a result, we run our business (machine learning) in the exact way we tell our "customers" (industry professionals) NOT to run theirs, i.e., by the "seat of the pants!" We go to great lengths to convince them that they should "listen" to what their data tell them, that they should build and use models that are well-grounded in the information embedded in the data generated by their business. We write best-sellers that describe examples upon examples of real situations where number-crunching, business analytics, data mining —call it what you may, it has some machine learning under the hood— challenges and outperforms human ability and wisdom [3, 20].

Is it not ironical then that applied machine learning is guided mostly by hunches, anecdotal evidence, and individual experience? Should we not look to the data our various applications generate to gain better insight into how to do machine learning? Should we not learn to learn? If we are not quack doctors, but truly believe in our medicine, then the answer should be a resounding "Yes!" Not only would such an approach be more scientific, it would make expertise generalizable to the entire community of researchers and practitioners.

We shall call metadata the type of data that may be viewed as being generated through the application of machine learning and metalearning the use of machine learning techniques to build models from metadata. We will be particularly interested in the important problem of algorithm selection.

1.2 Brief History

While it is impossible to give an exhaustive account of the work relevant to metalearning, the following lists in roughly chronological order several of the most representative contributions.[1]

STABB can be viewed as an early precursor of metalearning systems, since it was the first to show that a learner's bias could be adjusted dynamically [54].

VBMS is a relatively simple metalearning system that learns to select the best among three symbolic learning algorithms as a function of only two dataset characteristics: the number of training instances and the number of features [47].

Further work characterized and investigated the extensive role of data character as a determiner of system behavior in empirical concept learning [48]. The main contribution was the consideration of concepts as functions or surfaces over the instance space, which 1) leads to an useful characterization of complexity based on shape, size and concentration, and 2) allows systematic artificial data generation. The results, which focused on measures of complexity, showed that shape and especially concentration have significant effects. The results, however, are based on only two (now somewhat outdated) learning algorithms: ID3 and PLS1.

The MLT project (ESPRIT Nr. 2154) was one of the first formal attempts at addressing the *practice* of machine learning [34, 19]. To facilitate such practice, MLT produced a rich toolbox consisting of a number (10) of strictly symbolic learning algorithms for classification, datasets, standards and know-how. Considerable insight into many important machine learning issues was gained during the project, much of which was translated into rules that form the basis of Consultant-2, the user guidance system of MLT. Consultant-2 is a kind of expert system for algorithm selection. It functions by means of interactive question-answer sessions with the user. Its questions are intended to elicit information about the data, the domain and user preferences. Answers provided by the user then serve to fire applicable rules that lead to either additional questions or, eventually, a classification algorithm recommendation. Although its knowledge base is built through expert-driven knowledge engineering rather than via metalearning, Consultant-2 stands out as the first automatic tool that systematically relates application and data characteristics to classification learning algorithms.

The StatLog project (ESPRIT Nr. 5170) extended VBMS by considering a larger number of dataset characteristics, together with a broad class

[1] A thorough, although not completely up-to-date, bibliography is at http://groups.google.com/group/meta-learning.

of candidate models and algorithms for selection [13, 14, 26]. The aim was to characterize the space in which a given algorithm achieves positive generalization performance. StatLog produced a thorough empirical analysis of machine learning algorithms and models. Some metalearning for model selection was also attempted with performance restricted to accuracy only.

A statistical metamodel to predict the expected classification performance of 11 learning algorithms as a function of 13 data characteristics was later designed [53]. The 13 data characteristics include basic descriptive statistics (e.g., number of classes) as well as multivariate statistics (e.g., mean skewness) and derived statistics (e.g., square root of the ratio of the number of feature variables to the number of training examples). 17 to 18 of the 22 datasets used in StatLog are used in this work to fit the statistical metamodel. A number of useful conclusions regarding the impact of certain data characteristics on performance are drawn.

The CRISP-DM project (ESPRIT Nr. 24.959) aimed at developing a cross-industry standard process for data mining [17] and to embody this in a support tool which provides user guidance based on the process model. CRISP-DM covers the entire data mining process, from definition of business objectives and their translation into data mining goals, through data access, preparation, exploration and modelling, to results interpretation, assessment against objectives, deployment and documentation. Although not addressing metalearning directly, this project is clearly relevant as it formalizes the data mining process and hence the various decision points where metalearning may assist.

The METAL project (ESPRIT Nr. 26.357) focused on 1) discovering new and relevant data/task characteristics, and 2) using metalearning to select either one in many or a ranking of classifiers. The project's main deliverable is the Data Mining Advisor (DMA), a Web-based metalearning system for the automatic selection of learning algorithms in the context of classification tasks. Given a dataset and goals defined by the user in terms of accuracy and training time, the DMA returns a list of algorithms that are ranked according to how well they meet the stated goals. The DMA implements two ranking mechanisms, one based on exploiting the ratio of accuracy and training time [15] and the other based on the idea of Data Envelopment Analysis [2, 9]. The DMA's choice of providing rankings rather than "best-in-class" is motivated by a desire to give as much information as possible to the user. In a "best-in-class" approach, the user is left with accepting the system's prediction or rejecting it, without being given any alternative. Hence, there is no recourse for an incorrect prediction by the system. Since ranking shows all algorithms, it is much less brittle as the user can always select the next best algorithm if the current one does not appear satisfactory. In some sense, the ranking approach subsumes the "best-in-class" approach.

The notion of Intelligent Discovery Assistant (IDA) provides a template for building ontology-driven, process-oriented assistants for KDD [7, 8]. IDAs encompass the three main algorithmic steps of the KDD process, namely, preprocessing, model building and post-processing. In IDAs, any chain of operations consisting of one or more operations from each of these steps is called a *Data Mining (DM) process*. The goal of an IDA is to propose to the user a list of ranked DM processes that are both valid and congruent with user-defined preferences (e.g., speed, accuracy).

Recent research has focused on extending the IDA approach by leveraging the synergy between ontology (for deep knowledge) and Case-Based Reasoning (for advice and (meta)learning) [18]. The system uses both declarative information (in the ontology and case base) as well as procedural information in the form of rules fired by an expert system. The case base is built around 53 features to describe cases; the expert system's rules are extracted from introductory Data Mining texts; and the ontology comes from human experts. The system is still in the early stages of implementation.

Theoretical results, such as the NFL theorem and its consequences on metalearning (e.g., see [50, 59, 60]), help in identifying limitations and opportunities for metalearning. For example, one of the latest results shows that for almost every single target, the generalization error of any two learning algorithms is almost exactly identical (for zero-one loss) [61]. This emphasizes just how important prior information about the target is and that unless the set of allowed targets is restricted to an extremely small set, any pair of algorithms considered will perform essentially the same. Such results are clearly relevant to metalearning work on model selection.

Extensive empirical studies confirm the theory and provide much insight into learning both as sources of direct meta-knowledge and as input to metalearning (e.g., see [1, 32, 37, 57]). For example, 1-rules (rules that classify an object on the basis of a single attribute) have been shown to achieve surprisingly high accuracy on many datasets [32]. In particular, they almost match C4.5 results. Additionally, The results of Lim et al. (2000) show that most algorithms adapt to noise quite well and the mean error rates of many algorithms are sufficiently similar that their differences are statistically insignificant. This suggests that the differences are also probably insignificant in practical terms, that is, criteria other than predictive accuracy should be considered in selecting algorithms.

Finally, a most insightful account of the issues surrounding model class selection (that should be considered by anyone who is serious about metalearning) can be found in [56]. A number of methods for model class selection are outlined and a taxonomy of metalearning study types is proposed.

Chapter 2

Metalearning for Algorithm Selection

Before we proceed, we briefly discuss both theoretical and practical considerations for the use of metalearning in algorithm selection, and outline Rice's general framework.

2.1 Theoretical Considerations

Originally introduced to the Neural Network Community, the No Free Lunch (NFL) theorem [59, 60] was contextualized and brought to the attention of the Machine Learning community in the form of a Law of Conservation for Generalization Performance (LCG): When taken across all learning tasks, the generalization performance of any learner sums to 0 [50]. As some researchers have begun to explore metalearning as a means of designing robust learning systems, others have been quick to point to the NFL theorem as the sure show-stopper. However, by making explicit the assumptions that underlie Machine Learning research, it is possible to show that metalearning offers a viable mechanism to build a "general-purpose" learning algorithm. Further details are in [29].

2.1.1 NFL Revisited

As a simple illustration of the NFL theorem, consider the simple space, $\mathcal{F}$, of binary functions defined over $\mathbb{B}^3 = \{0, 1\}^3$, and assume that the instances

	Inputs			f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	...
Training Set	0	0	0	0	0	0	0	0	0	0	0	0	0	...
	0	0	1	0	0	0	0	0	0	0	0	0	0	...
	0	1	0	0	0	0	0	0	0	0	0	0	0	...
	0	1	1	0	0	0	0	0	0	0	0	0	0	...
	1	0	0	0	0	0	0	0	0	0	0	1	1	...
	1	0	1	0	0	0	0	1	1	1	1	0	0	...
Test Set	1	1	0	0	0	1	1	0	0	1	1	0	0	...
	1	1	1	0	1	0	1	0	1	0	1	0	1	...

Figure 2.1: Sample Train/Test Setting for Binary Functions over 3 Boolean Variables

of set $Tr = \{000, 001, \ldots, 101\}$ are observed, whilst the instances of set $Te = \mathbb{B}^3 - Tr = \{110, 111\}$ constitute the off-training set (OTS) test set, as depicted in Figure 2.1. The NFL theorem, or LCG, in this setting shows that, averaged across all functions, $f_1, f_2, \ldots, f_{256} \in \mathcal{F}$ (i.e., all labelings of instances), the behavior on Te of any learner trained on Tr is that of a random guesser.[1]

A quick examination of Figure 2.1 makes this result intuitive and obvious. Consider functions f_1 through f_4 in Figure 2.1. For all 4 functions, Tr is the same. Hence, provided any deterministic learner L, the model induced by L from Tr is the same in all 4 cases. It follows immediately that, since the associated Te's span all possible labelings of the OTS instances, for any OTS instance any model will be correct for half the functions and incorrect for the other half. Indeed, every classifier will have accuracies of 100%, 50%, 50%, and 0% across the four functions. The argument is easily repeated across all such subsets of 4 functions, giving the overall result.[2]

It then becomes apparent that the NFL theorem in essence simply restates Hume's famous conclusion about induction having no rational basis:

[1] Note here that generalization performance consists of two components: expected performance over instances that have already been encountered and expected performance over instances that have not yet been encountered (i.e., OTS). The former is not trivial, since previously encountered instances are not guaranteed to have the same label as in the training set—so there still is an important statistical estimation task. However, we restrict attention here to the OTS performance as that is the focus of the NFL theorem.

[2] A similar intuitive argument is presented in [21].

> There can be no *demonstrative* arguments to prove, *that those instances, of which we have had no experience, resemble those, of which we have had experience*....Thus not only our reason fails us in the discovery of the *ultimate connexion* of causes and effects, but even after experience has inform'd us of their *constant conjunction*, 'tis impossible for us to satisfy ourselves by our reason, why we shou'd extend that experience beyond those particular instances, which have fallen under our observation. We suppose, but are never able to prove, that there must be a resemblance betwixt those objects, of which we have had experience, and those which lie beyond the reach of our discovery. [33]

All other things being equal, given that all one has seen is Tr and its labeling, there is no rational reason to prefer one labeling of Te over another.

The crucial and most powerful contribution of the NFL theorem, is pointing out that whenever a learning algorithm performs well on some function, as measured by OTS generalization, it must perform poorly on some other(s). Hence, building decision support systems for what learning algorithm works well where becomes a valuable endeavor.

2.1.2 Building an Ultimate Learning Algorithm

Let p^Ω be the non-uniform probability distribution over the f_i's induced by some process Ω that presents learning problems. Given a training set, a learning algorithm, L, induces a model, M, which defines a class probability distribution, p, over the instance space. An *Ultimate Learning Algorithm (ULA)* is a learning algorithm that induces a model $M^\star$, such that:

$$\forall M' \neq M^\star \;\; E(\delta(p^\star, p^\Omega)) \leq E(\delta(p', p^\Omega))$$

where the expectation is computed for a given training/test set partition of the instance space, over the entire function space, and *delta* is some appropriate distance measure.

Finding a ULA thus consists of finding a learning algorithm whose induced models closely match our world's underlying distribution of functions. Note that the choice of the word "ultimate" deliberately avoids the ambiguity of the word "universal." Universal is generally defined as either (1) applicable independent of any assumptions, or (2) applicable throughout the entire universe. As pointed out by Hartley: "the first is what most mathematicians mean by universal, and by this definition the conservation law rules out any useful generalization. However when asking about the real world it is the second definition that is important. What could happen in other conceivable universes is of no possible interest to us." [39]. In "our" universe, we embrace the *weak assumption* of Machine Learning, namely that the process that presents us with learning problems induces a non-uniform probability distribution over the possible functions [29]. Hence, the definition of ULA is not in contradiction with the NFL theorem.

Cross-validation is regularly used as a mechanism to select among competing learning algorithms, as illustrated in Figure 2.2. As pointed out by Wolpert [60], however, cross-validation is also subject to the NFL theorem. This is easily seen again from Figure 2.1. Since Tr does not change over f_1 through f_4, cross-validation always selects the same best learner in each case and the original NFL theorem applies.

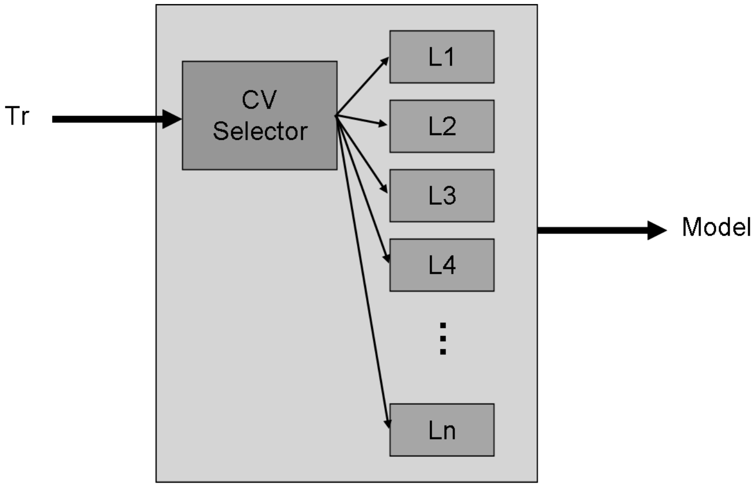


Figure 2.2: CV Meta-selector

In fact, it is possible to show further that a No Free Lunch result holds for cross-validation, even under the weak assumption of Machine Learning [29]. It follows that cross-validation cannot generalize and thus cannot be used as a viable way of building an ultimate learning algorithm.

At the other end of the spectrum, the traditional approach in Machine Learning research is to design one's own algorithm. This is generally motivated by the presence of one or more tasks that one wishes to learn, and for which no existing algorithm seems to perform to the desired level. The researcher then sets out to design —generally with much (unreported) trial-and-error— a new learning algorithm that performs well on the target tasks and others that are available for evaluation. This approach depends upon the *Strong Assumption* of Machine Learning, that the distribution of functions is known well enough to be incorporated by the algorithm's inductive bias. More specifically, it assumes that the universe is such that the tasks likely to occur are exactly those that I am interested in solving, whilst all other tasks are so unlikely —or so irrelevant to me— that I need not worry about them (see Hartley's argument above).

This expertise-driven, knowledge-based strategy results in implicit metalearning (see below) by the research community and is an essential part of the science of Machine Learning. Historically, new algorithms tend to be designed specifically to overcome limitations of known algorithms as they are discovered (e.g., new tasks arise for which no existing algorithm seems to be suitable). In the process, the community's knowledge about learning increases. Although this manual approach is a possibility, we argue that it makes stronger assumptions than we might like, is laborious, and also is somewhat at odds with the philosophy of machine learning. We propose metalearning as a viable alternative.

In our framework, metalearning is cast as the task of learning an estimate of the (universe's) true function distribution. It assumes that it is possible to gather training data at the metalevel to learn that estimate, and use it to select among base-level learners, as depicted in Figure 2.3.

Metalearning assumes only a bias for learning the estimate of the function distribution. We should be clear, however, that even with metalearning our lunch is not free. NFL results apply at the metalevel just as they apply at the level of particular learning tasks. Metalearning does depart from the setting of the NFL theorems, in that it takes into account prior observations from the function distribution. Yet, it is reasonable to conjecture that there is a direct analog of the basic NFL theorem to the meta-level —showing that all metalearners have equivalent performance given some (probably debatable) averaging across universes.

The assumptions that must be made for metalearning are considerably more natural than those that must be made for manual algorithm design. We all hold deep-rooted, intuitive notions of bizarre functions. For example, (among many things) we believe that variables that never have exhibited any relevance are more likely than not to continue to be irrelevant. After many

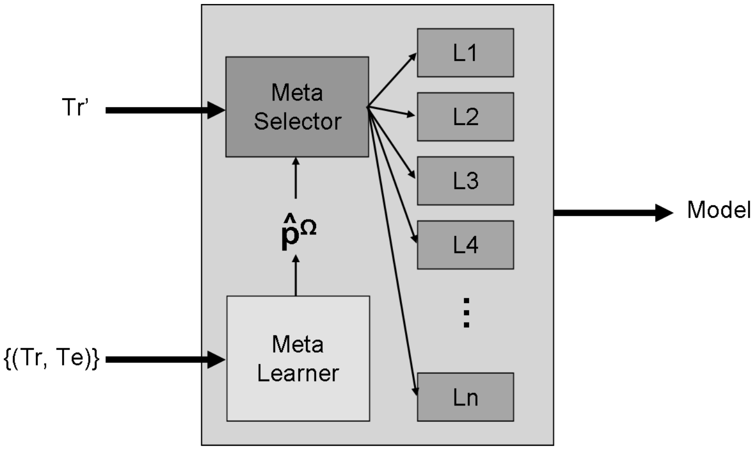


Figure 2.3: Meta-learning Selector

years of experience, we would find it bizarre for the date to play a role in whether crows are black or whether gravity will be attractive or repulsive. Harkening back to Hume, there is no rational reason for these beliefs, which of course is the “riddle” of induction. However, implicit in Western thinking is that if we were to make only one assumption, it would have to be that induction is valid —that we can generalize from what we have seen to things we have yet to encounter. This is the fundamental assumption of science as we practice it. It also is fundamental to our being able to live at ease in the world, not constantly worrying for example that the next time we step on a bridge it will not support our weight.

2.2 Practical Considerations

Although rather theoretical, since it applies to the universe of all tasks, the NFL theorem has two important practical implications for both algorithm designers and practitioners. When a designer introduces a novel classification algorithm, how does she position it in the exisiting algorithm landscape? When a practitioner is faced with a new classification task for which she seeks a model with high accuracy, how does she know which algorithm to use? In either case, the only two viable alternatives in light of the NFL theorem are:

1. *Closed Classification World Assumption (CCWA)*: She assumes that all classification tasks likely to occur in real applications form some well-defined subset of the universe. As a designer, she shows that her novel algorithm performs better than others on that set. As a practitioner, she picks any of the algorithms that performs well on that set.

2. *Open Classification World Assumption (OCWA)*: She assumes no structure on the set of classification tasks likely to occur in real applications. As a designer, she characterizes as precisely as possible the class of tasks on which her novel algorithm outperforms others. As a practitioner, she has some way of determining which algorithm(s) will perform well on her specific task(s).

Interestingly, the widely-used approach consisting in benchmarking algorithm against well-known repositories (e.g., UCI) tends to implicitly favor the CCWA. Yet, this remains at best a conjecture. To date, no one has offered a characterization of "real-life" classification tasks. And evidence suggests that this is by no means a trivial task. To compound the problem for practitioners, most efforts in algorithm design seem to share the same oblivious pattern:

1. They propose new algorithms that overcome known limitations. Yet, unless one accepts the CCWA, this simply shifts the original question of how to overcome the targeted limitations to the equally difficult question of determining what applications the proposed approach works well on.

2. They "promote" new algorithms on the basis of limited empirical results, leaving the burden of proof to the users. It is not trivial to know how well any new approach will generalize beyond the problems it has been tested against so far.

Hence, over the years, we have witnessed the birth of a large number of learning algorithms, with comparatively little insight gained in their individual applicability. As a result, users are faced with a plethora of algorithms, and without some kind of assistance, algorithm selection can turn into a serious road-block for those who wish to access the technology more directly and cost-effectively.

End-users often lack not only the expertise necessary to select a suitable algorithm, but also the availability of many algorithms to proceed on a trial-and-error basis (e.g., by measuring accuracy via some re-sampling technique such as n-fold cross-validation). And even then, trying all possible options is impractical, and choosing the option that "appears" most promising is likely to yield a sub-optimal solution.

Most general-purpose commercial Data Mining packages (e.g., Enterprise Miner,[3] Clementine,[4] Insightful Miner,[5] PolyAnalyst,[6] KnowledgeStudio,[7] Weka,[8] RapidMiner,[9] Xelopes[10], etc.) consist of collections of algorithms —often originating in the public domain and re-implemented with rather minor, if any, proprietary extensions— wrapped in a user-friendly graphical interface. While such tools facilitate access to algorithms, they generally offer no real decision support to non-expert end-users.

What is needed is an informed search process to reduce the amount of experimentation while avoiding the pitfalls of local optima. Informed search requires meta-knowledge about the precise conditions under which a given algorithm is better than others for a given task. In turn, such meta-knowledge may be usefully incorporated into data mining assistants that offer support to non-expert practitioners. Metalearning offers a robust mechanism to build meta-knowledge about algorithm selection in classification. Hence, in a very practical way, metalearning contributes to the successful use of Data Mining tools outside the research arena, in industry, commerce, and government.

2.3 Rice's Framework

Back in 1976, Rice proposed a formalization of the algorithm selection problem [49]. Although the paper gives no specific methods to solve the algorithm selection problem, it describes a few examples and discusses approaches such as linear forms, in general terms. The paper clearly makes no reference to metalearning either, but does recognize that "the selection procedure itself is

[3] http://www.sas.com/technologies/analytics/datamining/miner/

[4] http://www.spss.com/clementine/

[5] http://www.insightful.com/products/iminer/default.asp

[6] http://www.megaputer.com/products/pa/index.php3

[7] http://www.angoss.com/products/studio.php

[8] http://www.cs.waikato.ac.nz/ml/weka/

[9] http://rapid-i.com/content/blogcategory/10/69/ (formerly known as Yale)

[10] http://www.prudsys.com/Produkte/Algorithmen/Xelopes/

an algorithm," making metalearning one rather natural approach to solving the algorithm selection problem.[11] Readers may be interested in a recent, very insightful paper that surveys and unifies a number of past metalearning approaches within Rice's framework [51].

Metalearning for algorithm selection, as we discuss it here, corresponds to Rice's model depicted in Figure 2.4 (adapted from Figure 3 in [49], where a problem x in problem space P is mapped via some feature extraction process to $f(x)$ in some feature space F, and the selection algorithm S maps $f(x)$ to some algorithm a in algorithm space A, so that some selected performance measure (e.g., accuracy), p, of a on x is optimal.

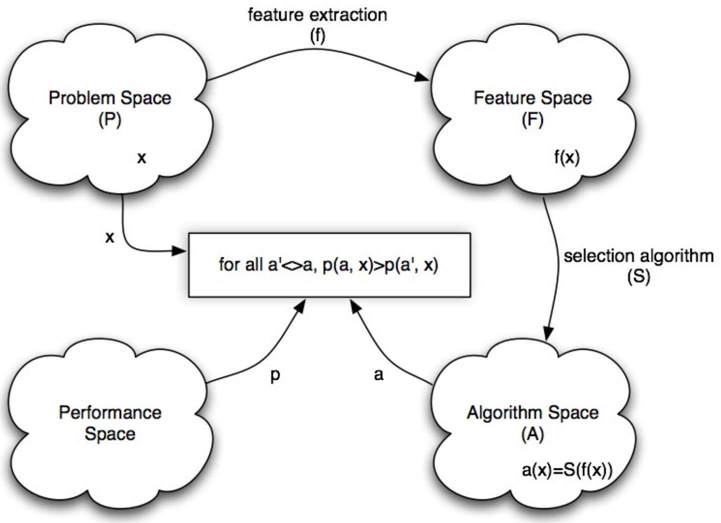


Figure 2.4: Rice's Framework for Algorithm Selection

Interestingly, the selection algorithm works exclusively on the features $f(x)$ of the problem, but computation of the value of the selected performance measure $p(a, x)$ does depend on the actual problem x. Furthermore, one

[11] Other important issues that cut across domains are highlighted. In particular, the Strong Assumption of Machine Learning is one instantiation of Rice's statement that "most algorithms are developed for a particular class of problems even though the class is never explicitly defined. Thus the performance of algorithms is unlikely to be understood without some idea of the problem class associated with their development."

would hope that problems characterized by similar features map to the same algorithm and exhibit similar performance.

The above framework makes explicit the issues that must be addressed in automatic algorithm selection, namely:

1. The choice of f,
2. The choice of S, and
3. The choice of p.

In the context of metalearning, A is a set of base-level learning algorithms and S is itself also a learning algorithm; the choice of S, here, amounts to deciding which learning algorithm to use at the metalevel. Making S a learning algorithm, i.e., using metalearning, has further important practical implications about:

1. The construction of the training metadata set, i.e., problems in P that feed into F through the characterization function f,
2. The content of A, i.e., the set of base-level learning algorithms that the metalearner can choose from, and
3. The computational cost of f and S.

In an attempt at slightly generalizing Rice's framework, we will also consider the choice of the form of the output of S, in particular the possibility of ranking n algorithms rather than selecting 1 of n, as a parameter of the framework.

We address each of the aforementioned issues in the next chapter.

2.4 Algorithm Selection vs. Model Combination

Before we continue on, it is worthwhile to draw a distinction between metalearning for algorithm selection and what has become known as model combination in machine learning, which is often labeled metalearning by its authors.

Model combination consists of creating a single learning system from a collection of learning algorithms. There are two basic approaches to model combination. The first one exploits variability in the application's data and combines multiple copies of a single learning algorithm applied to different subsets of that data. The second one exploits variability among learning algorithms and combines several learning algorithms applied to the same application's data. The main motivation for combining models is to reduce the probability of misclassification based on any single induced model by increasing the system's area of expertise through combination. Indeed, one of the implicit assumptions of metalearning for algorithm selection is that there exists an optimal learning algorithm for each task. Although this clearly holds in the sense that, given a problem p and a set of learning algorithms A, there is a learning algorithm $a \in A$ that performs better than all of the others on p, the actual performance of a may still be poor. In some cases, one may mitigate the risk of settling for a suboptimal learning algorithm by replacing single model selection with model combination.

Because it draws on information about base-level learning —in terms of either the characteristics of various subsets of data or the characteristics of various learning algorithms— model combination is often considered a form of metalearning. We purposely exclude model combination from our presentation of metalearning and refer the interested reader to Chapter 4 of [16] for a survey of extant model combination techniques.

Chapter 3

The Practice of Metalearning

Using Rice's framework as a reference point, we discuss in detail the different aspects of the practice of metalearning for effective base-level learning algorithm selection. Some of the material presented here is adapted from [28].

3.1 Choosing the content of A

As demonstrated by the NFL theorem, no learner is universal. However, one may naturally consider that each learner has its own area of expertise, defined as the set of learning tasks on which it performs well [5]. Since the role of the metamodel is to predict which algorithm is most likely to perform best on each new task, one should select base learners with complementary areas of expertise. The two issues in this choice are coverage and size.

In principle, one should seek the smallest set of learners that is most likely to ensure a reasonable coverage of the base-level learning space. Minimizing the size ensures efficiency (see computational cost below), whilst maximizing coverage increases effectiveness. In practice, it is non-trivial to ensure both coverage and a minimal set of learners. In fact, our experiments on the space of binary classification tasks of three Boolean variables (i.e., the complete space of 256 functions from $\{0, 1\}^3$ to $\{0, 1\}$) suggest that, from 26 applicable algorithms from Weka [58], a subset of 7 is sufficient to obtain maximal coverage, but 9 tasks still remain uncovered (i.e., none of the learners is better than chance on these). Short of designing effective learners for these tasks, one may hope that the performance gain obtained from the metamodel's correct predictions on covered tasks exceeds the loss incurred on uncovered tasks. Given that one is rarely interested in solving all possible learning tasks, one may further hope that the uncovered tasks are unlikely to arise in practice.

In order to approximate the best situation, we recommend that base learners have different biases by choosing representatives from varied model classes. This is consistent with the approach discussed in [56], where it is argued that experts generally first select a model class (e.g., decision trees or neural networks) on the basis of high-level characteristics of the task (e.g., noise level, attribute type, etc.), and then select the most promising algorithm(s) in that class. The more varied the biases, the greater the coverage.

3.2 Constructing the Training Metadata

Metalearning requires training metadata, i.e., data about base level learning problems or tasks. Unfortunately, the number of accessible, documented, real-world classification tasks is less than a couple hundreds today. Such a small sample poses a challenge for learning, which may be addressed either by augmenting the training set through systematic generation of synthetic base level tasks, or by taking the view that the algorithm selection task is inherently incremental and treating it as such.

The first approach assumes that we know something about the kinds of tasks that occur in nature so that we are able to generate tasks with adequate characteristics. The second approach makes no such assumption but results in slower learning since classification tasks become available over time. On the other hand, it naturally adapts to reality, extending to new areas of the base level learning space only when tasks from these areas actually arise.

In all cases, the training metadata consists of meta-examples of the form $< f(x), t(x) >$, where $t(x)$ represents some target value for x. By definition, $t(x)$ is predicated upon the choice of performance measure p, as well as the choice of the form of the output of S. Focusing on the case of selection of 1 of n, the target value corresponds to the algorithm that performs best (according to p) on problem x, i.e.,

$$t(x) = \text{argmax}_{a \in A} \ p(a, x)$$

Metalearning then takes the set $\{< f(x), t((x) >: x \in P' \subseteq P\}$ as a training set and induces a metamodel that, for each new classification problem, predicts the algorithm from A that will perform best. It is clear from this that constructing meta-examples is computationally intensive.

3.3 Choosing f

As in any learning task, the characterization of the examples plays a crucial role in enabling learning. In particular, the features used must have some predictive power. Selecting appropriate features, i.e., choosing the function f, is by no means trivial. As pointed out by Rice, "the determination of the best (or even good) features is one of the most important, yet nebulous, aspects of the algorithm selection problem." [49].

To date, four main classes of characterization have been proposed, as described below.

3.3.1 Statistical and Information-theoretic Characterization

The simplest and most widely used function f takes classification learning problems, described as datasets of labeled instances, and extracts a number of statistical and information-theoretic measures (e.g., see [47, 1, 38, 26, 22, 53]). Typical measures include number of features, number of classes, ratio of examples to features, degree of correlation between features and target concept, average class entropy and class-conditional entropy, skewness, kurtosis, and signal to noise ratio.

The assumption made here is that learning algorithms are sensitive to the underlying structure of the data on which they operate, so that one may hope that it may be possible to map structures to algorithms. Interestingly, the results of [61] suggest that the size of the training set and the size of the input space play a crucial role in determining the difference between algorithms. Since in practice, these are usually different, one may expect to capture sufficient information from these and other measures to discriminate among learners. Empirical results do seem to confirm this intuition.

3.3.2 Model-based Characterization

A different form of f exploits properties of a hypothesis induced on problem x as an indirect form of characterization of x [4, 6, 42]. This approach has a couple of advantages:

1. The dataset is summarized into a data structure that can embed the complexity and performance of the induced hypothesis, and thus is not limited to the example distribution.

2. The resulting representation can serve as a basis to explain the reasons behind the performance of the learning algorithm.

To date, only decision trees have been considered, where a decision tree is built on x and $f(x)$ consists of either the tree itself, if the metalearning algorithm can manipulate it directly, or properties extracted from the tree, such as nodes per feature, maximum tree depth, shape, and tree imbalance.

Although (higher-order) learning algorithms exist that can in principle handle the induced hypothesis directly at the metalevel, the only results reported with this approach to characterization have been obtained with more traditional metalearning algorithms and features extracted from the induced trees.

3.3.3 Landmarking

Another source of characterization falls within the concept of landmarking (e.g., see [5, 45]). Each learner has a class of tasks on which it performs particularly well, under a reasonable measure of performance. We call this class the *area of expertise* of a learner. The basic idea of the landmarking approach is that the performance of a learner on a task uncovers information about the nature of the task. Hence, a task can be described by the collection of areas of expertise to which it belongs.

We call a *landmark learner*, or simply a *landmarker*, a learning mechanism whose performance is used to describe a task. Landmarking is the use of these learners to locate the task in the *expertise space*, the space of all areas of expertise. Landmarking thus views metalearning as intending to find locations of tasks in the expertise space. While other approaches rely on peripheral information (e.g., statistical characteristics, model-based properties, etc.), landmarking uses an expertise map as its main source of information.

The kind of inference on which landmarking relies can be illustrated with the help of Figure 3.1. The outer rectangle represents an expertise space and the labelled areas within it are areas of expertise of learners $i1, \ldots, i7$. Assume that $i1$, $i2$, and $i3$ are taken as landmarkers. In this case, landmarking may for example conclude that problems on which both $i1$ and $i3$ perform well, but on which $i2$ performs poorly, are likely to be in $i4$'s area of expertise. Of course, the proximity of the areas of expertise of two learners indicates some similarity in the mechanism behind them. For landmarking purposes, however, learners are considered as black boxes and we are interested only in correlations between their areas of expertise that can be found empirically.

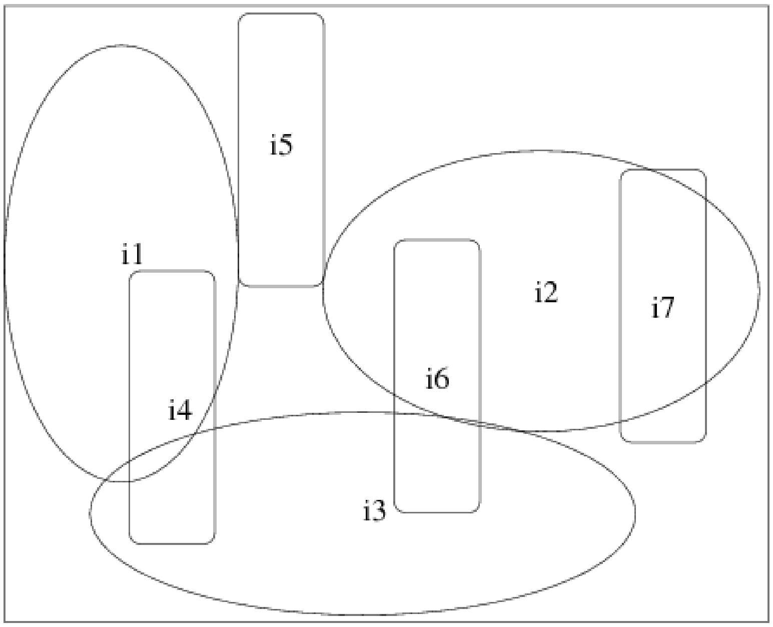


Figure 3.1: Illustrative Map of Areas of Expertise

We can say that landmarking concentrates solely on cartographic considerations. Tasks are described only by how landmarkers fare on them. Exploring the metalearning potential of landmarking amounts to investigating how well a landmark learner's performance hints at the location of the learning tasks in the expertise map.[1]

In principle, every learner's performance can signpost the location of a problem with respect to other learner's expertise. In practice, however, we want landmark learners to be efficient. The *prima facie* advantage of landmarking resides in its simplicity: learners are used to signpost learners. To

[1] Arguably, different performance measures have different expertise maps. Landmarking can, in principle, be applied to expertise spaces based on any performance measure.

preserve this simplicity, it should use simple and efficient learning strategies as landmarkers. The cost of running the chosen set of landmark learners must be smaller than the cost of running the learners we are focusing on (i.e., the set A); otherwise, it would be better to run all the learners to find out which one performs best. Nevertheless, if some of the target learners are themselves computationally efficient, they can readily be used as landmarkers. In this case, if one of these efficient learners is itself the selected model, the overall efficiency of the system is optimal.

Note that the idea of landmarking can be compared with the cross-validation approach to model selection. Instead of running the "full-blown" learning mechanism on parts of the task, we run simplified versions of the mechanism, as landmarkers, on the whole task. Subsequently, the landmarkers' performance values are used as task descriptors or meta-attributes. An idea building on both landmarking and cross-validation is to exploit information obtained on simplified versions of the data (e.g., small samples) [25, 52]. Performance results on these samples serve to characterize tasks and are referred to as sampling landmarks. This information is subsequently used to select a learning algorithm.

The main issue when using landmarking for metalearning is the selection of the set of landmarkers. Often this are naive learning algorithms (e.g., OneR, Naive Bayes) or "scaled-down" versions of more complex algorithms (e.g., DecisionStump). Results with landmarking have been promising.

3.3.4 Partial Learning Curves

Most recently, partial learning curves were suggested as an indirect kind of characterization to select between two algorithms [36]. A learning curve depicts the evolution of some learning performance measure, such as accuracy, as a function of the size of the training set. Popular in the early days of Machine Learning research, learning curves have mostly been ignored in recent years, where single aggregate measures, such as n-fold cross-validation accuracy, or other curves, such as the ROC curve, have been favored. Recent research suggests, however, that training set size should be taken into account, as conclusions regarding relative predictive performance for small training set sizes may be reversed as training set size increases [43]. The argument is further made there that "reviving the learning curve as an analytical tool in machine learning research can lead to other important, perhaps surprising insights." (p. 239).

Leite and Brazdil's method predicts the relative performance of a pair of learning algorithms, say A_1 and A_2, using partial learning curves as follows. The training metadata may be regarded as consisting of triplets $< D, lc_{A_1,D}, lc_{A_2,D} >$, where D is a (base-level) dataset and $lc_{A_1,D}$ (resp., $lc_{A_2,D}$) is the learning curve for A_1 (resp., A_2) on D, computed with progressive sampling (sample m has size $2^{6+0.5m}$) [46, 35]. Each learning curve is in turn represented as a vector $< a_{A_k,D,1}, a_{A_k,D,2}, \ldots, a_{A_k,D,\#S} >$, where $a_{A_k,D,r}$ is the accuracy of algorithm A_k on dataset D for the r-th sample. The distance between two datasets D_i and D_j, in the context of discriminating between the predictive performances of algorithms A_1 and A_2, is then given by the function:

$$\begin{aligned} d_{A_1,A_2}(D_i, D_j) &= \sum_{m=1}^{\#S} [(a_{A_1,D_i,m} - a_{A_1,D_j,m})^2 + (a_{A_2,D_i,m} - a_{A_2,D_j,m})^2] \\ &= \sum_{k=1}^{2} \sum_{m=1}^{\#S} (a_{A_k,D_i,m} - a_{A_k,D_j,m})^2 \end{aligned}$$

Generalizing to n learning algorithms and partial curves of varying lengths, the distance between two datasets D_i and D_j, in the context of discriminating among the predictive performances of algorithms A_1 through A_n, is given by:

$$d_{A_1,\ldots,A_n}(D_i, D_j) = \sum_{k=1}^{n} \sum_{m=1}^{\#S_k} (a_{A_k,D_i,m} - a_{A_k,D_j,m})^2$$

which is simply the (square of the) Euclidean distance between two dataset "vectors" from the meta-database.

When a new target dataset T is presented, algorithms A_1 and A_2 are executed to compute their partial learning curves on it, up to some predefined sample size, $\#S$. The 3 nearest neighbors of T are then identified using the distance function d. The accuracies of A_1 and A_2 on these neighbors for sample size $|T|$ are retrieved (or computed), from the database. Each neighbor votes for either A_1, if A_1 has higher accuracy than A_2 at the target size, or A_2, if the opposite is true. The "best" algorithm predicted for T corresponds to the majority vote.

3.4 Computational Cost of f and S

The issue of computational cost is a consequence of the others and the price to pay to be able to perform algorithm selection learning at the metalevel. We argue however that, in order to be justifiable, the cost of computing $f(x)$ should be significantly lower than the cost of computing $t(x)$. Otherwise, even if the metamodel is very accurate, it has little value as the user would be better off trying all algorithms in A and selecting the best one, which clearly defeats the purpose.

The larger the set A and the more computationally intensive the algorithms in A, the more likely it is that the above condition holds. In all implementations of the aforementioned characterization approaches, that condition has been satisfied.

As far as S is concerned, there are two costs to be considered, namely, the cost of inducing the metamodel and the cost of making prediction with the metamodel. For most (meta-)learning algorithms, the latter is typically not a concern.[2] The former is dependent on the learning regime chosen. If metalearning is performed in batch mode, the cost of building the metamodel may be relatively large, but may also be considered as an upfront cost, incurred offline and with no direct impact on the overall performance of the decision system. If, on the other hand, metalearning is performed incrementally, the cost of building the model, which is updated after each new piece of training metadata becomes available, must be taken into account.

3.5 Choosing p

Since performance is generally paramount, predictive accuracy has become the *de facto* criterion, or performance measure, for algorithm selection. That is, metalearning has focused on the induction of models that match problems to learning algorithms using accuracy only. These metamodels are then used to choose, for a given problem, the learning algorithm that will produce a hypothesis with the highest predictive accuracy.

This bias on predictive accuracy in metalearning is largely justified by both theoretical results and pragmatic considerations, including:

[2] With the possible exception of instance-based learners when both the size of the training data and the number of neighbors to consider are large.

- The NFL theorem, as pointed out earlier, shows that no single learning algorithm will construct hypotheses of high accuracy on all problems. Hence, good performance on a given set of problems cannot be taken as guarantee of good performance on applications outside of that set.

- Predictive accuracy is virtually impossible to forecast. In other words, one cannot know how accurate a hypothesis will be until that hypothesis has been induced by the selected learning model and tested on unseen data.

- Predictive accuracy is easy to quantify. It is not subjective and induces a total order on the set of all hypotheses. Given an application, it is straightforward, through experimentation, to find which of a number of available models produces the most accurate hypothesis.

Although these considerations clearly show predictive accuracy to be a valid criterion, they ignore the fact that there are a number of other aspects which have an impact on which model to select. In fact, empirical evidence suggests that for large classes of applications, most learning models perform well in terms of accuracy [32]. Yet they often exhibit extreme variance along other interesting dimensions. Furthermore, this (ab)use of accuracy is a result of two of machine learning/data mining's strongest assumptions, namely that data adequately represents what it models and that predicting data values equates to predicting useful business outcomes [40]. Again, evidence suggests that this assumption may not always hold and other factors should be considered.

A number of other factors, including expressiveness, compactness, computational complexity and comprehensibility, are discussed in [27]. These could be handled in isolation or in combination to build multi-criteria performance measures. To the best of our knowledge, only computational complexity, as measured by training time, has been considered in tandem with predictive accuracy.

3.6 Choosing the form of the output of S

The standard form of the output of S, which fits Rice's framework, is a single algorithm selected among n algorithms. In this case, for every new problem, the metamodel simply returns one learning algorithm that it predicts will perform best on that problem.

Alternatively, one may build a metamodel that predicts a ranking of algorithms from A (e.g., [9, 15]). This approach reduces the brittleness of the metamodel. Assume that the algorithm predicted best for some new classification problem results in what appears to be a poor performance. In the single-model prediction approach, the user has no further information as to what other model to try. In the ranking approach, the user may try the second best, third best, and so on, in an attempt to improve performance. Empirical evidence suggests that the best algorithm is generally within the top three in the rankings.

Chapter 4

Metalearning Systems

Although a valid intellectual challenge in its own right, metalearning finds its real raison d'être in the practical support it offers Data Mining practitioners, i.e., those faced with real problems that are well-suited for machine learning techniques, but who have no or only limited technical expertise. The metaknowledge induced by metalearning provides the means to inform decisions about the precise conditions under which a given algorithm, or sequence of algorithms, is better than others for a given task. Here we review a few of the most promising, metaknowledge-based implementations of decision-support systems for algorithm selection in Data Mining.

4.1 MiningMart and Preprocessing

MiningMart, a large European research project, focused its attention on algorithm selection for preprocessing [23, 41, 24]. Preprocessing generally consists of nontrivial sequences of operations or data transformations, and is widely recognized as the most time consuming part of the KDD process, accounting for up to 80% of the overall effort. Hence, automatic guidance in this area can indeed greatly benefit users.

The goal of MiningMart is to enable the reuse of successful preprocessing phases across applications through case-based reasoning. A model for metadata, called M4, is used to capture information about both data and operator chains through a user-friendly computer interface. The complete description of a preprocessing phase in M4 makes up a *case*, which can be added to

MiningMart's case base.[1] "To support the case designer a list of available operators and their overall categories, e.g., feature construction, clustering or sampling is part of the conceptual case model of M4. The idea is to offer a fixed set of powerful pre-processing operators, in order to offer a comfortable way of setting up cases on the one hand, and ensuring re-usability of cases on the other." [41].

Given a new mining task, the user may search through MiningMart's case base for the case that seems most appropriate for the task at hand. M4 supports a kind of business level, at which connections between cases and business goals may be established. Its more informal descriptions are intended to "help decision makers to find a case tailored for their specific domain and problem." Once a useful case has been located, its conceptual data can be downloaded. The local version of the system then generates preprocessing steps that can be executed automatically for the current task. MiningMart's case base is publicly available on the Internet at http://mmart.cs.uni-dortmund.de/end-user/caseBase.html. As of June 2006, it contained five fully specified cases.

4.2 Data Mining Advisor

The Data Mining Advisor (DMA) is the main product of METAL, another European ESPRIT research project. The DMA is a Web-based metalearning system for the automatic selection of model building algorithms in the context of classification tasks. It is available on the INternet at http://www.metal-kdd.org. Given a dataset and goals defined by the user in terms of accuracy and training time, the DMA returns a list of algorithms that are ranked according to how well they meet the stated goals.

The ten algorithms considered by the DMA (the set A in Rice's framework) are: three variants of C5.0 (c50rules, c50tree and c5.0boost), Linear tree (ltree), linear discriminant (lindiscr), MLC++ IB1 (mlcib1) and Naïve Bayes (mlcnb), SPSS Clementine's Multilayer Perceptron (clemMLP) and RBF Networks (clemRBFN), and Ripper.

The DMA guides the user through a wizard-like step-by-step process consisting of the following activities.

[1] Case descriptions are too large to be included here, but MiningMart's case base can be browsed at http://mmart.cs.uni-dortmund.de/caseBase/index.html

1. Upload Dataset. The user is asked to identify the dataset of interest and to upload it into the DMA.

2. Characterize Dataset. Once the dataset is loaded, its characterization, consisting of statistical and information-theoretic measures is computed. This characterization becomes the metalevel instance that serves as input to the DMA's metalearner.

3. Parameter Setting and Ranking. The user chooses the selection criteria and the ranking method, and the DMA returns the corresponding ranking of all available algorithms.

   - Selection criteria: The user chooses among three predefined tradeoff levels between accuracy and training time, corresponding intuitively to main emphasis on accuracy, main emphasis on training time, and compromise between the two.
   - Ranking method: The DMA implements two ranking mechanisms, one based on exploiting the ratio of accuracy and training time and the other based on the notion of Data Envelopment Analysis.

4. Execute. The user may select any number of algorithms to execute on the dataset. Although the induced models themselves are not returned, the DMA reports tenfold cross-validation accuracy, true rank and score, and, when relevant, training time.

Recent site statistics suggest that the DMA has attracted over 325 registered visitors from 48 countries, who have uploaded over 165 datasets. Unfortunately, no data is available about actual uses of the system's recommendations.

4.3 METALA

Botía et al. have developed METALA, an agent-based architecture for distributed Data Mining, supported by metalearning [11, 12, 30].[2] The aim of METALA is to provide a system that:

1. Supports an arbitrary number of algorithms and tasks (i.e., P, and more importantly A may grow over time), and

[2] The architecture was originally known as GEMINIS and later renamed METALA.

2. Automatically selects an algorithm that appears best from the pool of available algorithms, using metalearning.

Each algorithm is characterized by a number of features relevant to its usage, including the type of input data it requires, the type of model it induces, and how well it handles noise. A hierarchical directory structure, based on the X.500 model, provides a physical implementation of the underlying ontology. Each learning algorithm is embedded in an agent that provides clients with a uniform interface to three basic services: configuration, model building and model application. Each agent's behavior is correspondingly governed by a simple state-transition diagram, with the three states *idle*, *configured* and *learned*, and natural transitions among them.

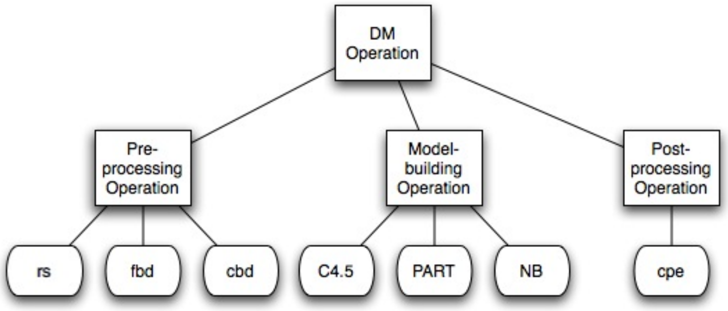
Similarly, each task is characterized by statistical and information-theoretic features, as in the DMA. METALA is designed so as to be able to autonomously and systematically carry out experiments with each task and each learner and, using task features as meta-attributes, induce a meta-model for algorithm selection. As new tasks and algorithms are added to the system, corresponding experiments are performed and the metamodel is updated.

The latest version of METALA is a J2EE implementation on a JBoss application server. Although they were developed independently of each other, METALA may be viewed as a natural extension of the DMA. It provides the architectural mechanisms necessary to scale the DMA up to any arbitrary number of learners and tasks, in a kind of online or incremental manner. The metalearning task is essentially the same (i.e., use task characterizations and induce a metamodel), and some of the functionality of the DMA (e.g., multicriteria ranking) could be added.

4.4 Intelligent Discovery Assistant

Although it does not (yet) implement metalearning, the notion of Intelligent Discovery Assistant (IDA), introduced by Bernstein and Provost, is unique in that, unlike the previous systems that focus on a single aspect of the KDD process, it encompass the three main algorithmic steps of the KDD process, namely, preprocessing, model building and post-processing [7, 8]. In IDAs, any chain of operations consisting of one or more operations from each of these steps is called a *Data Mining (DM) process.* The goal of an IDA is to propose to the user a list of ranked DM processes that are both valid and congruent with user-defined preferences (e.g., speed, accuracy).

The IDA's underlying ontology is essentially a taxonomy of DM operations or algorithms, where the leaves represent implementations available in the corresponding IDA. A simple ontology of only seven operators is depicted in Figure 4.1. Operations are characterized by pre-conditions, post-conditions and heuristic indicators.[3]



rs = random sampling (10%), fbd = fixed-bin discretization (10 bins), cbd = class-based discretization, cpe = CPE-thresholding post-processor

Figure 4.1: Sample IDA Ontology

The typical organization of an IDA consists of two components. The plan generator takes as input a dataset, a user-defined objective (e.g., build a fast, comprehensible classifier) and user-supplied information about the data, information that may not be obtained automatically. Starting with an empty process, it systematically searches for an operation whose pre-conditions are met and whose indicators are congruent with the user-defined preferences. Once an operation has been found, it is added to the current process, and its post-conditions become the system's new conditions from which the search resumes. The search ends once a goal state has been reached or when it is clear that no satisfactory goal state may be reached. The plan generator's search is exhaustive: all valid DM processes are computed. Table 4.1 shows the output of the plan generator for the ontology of Figure 4.1, when the

[3] Heuristic indicators point to the influence of the operation on overall goals such as accuracy, speed, model size, comprehensibility, etc. (e.g., sampling increases speed, pruning decreases speed but increases comprehensibility).

	Steps
Plan #1	C4.5
Plan #2	PART
Plan #3	rs, C4.5
Plan #4	rs, PART
Plan #5	fbd, C4.5
Plan #6	fbd, PART
Plan #7	cbd, C4.5
Plan #8	cbd, PART
Plan #9	rs, fbd, C4.5
Plan #10	rs, fbd, PART
Plan #11	rs, cbd, C4.5
Plan #12	rs, cbd, PART
Plan #13	fbd, NB, cpe
Plan #14	cbd, NB, cpe
Plan #15	rs, fbd, NB, cpe
Plan #16	rs, cbd, NB, cpe

Table 4.1: Sample List of IDA-generated DM Processes

input dataset is continuous-valued and comprehensible classifiers are to be preferred.

Once all valid DM processes have been generated, a heuristic ranker is applied to assist the user further by organizing processes in descending order of “return” on user-specified goals. For example, the processes in Figure 4.1 are ordered from simplest (i.e., least number of steps) to most elaborate. The ranking relies on the knowledge-based heuristic indicators. If speed rather than simplicity were the objective then Plan #3 in Figure 4.1 would be bumped to the top of the list, and all plans involving random sampling (*rs* operation) would also move up. In the current implementation of IDAs, rankings rely on fixed heuristic mechanisms. However, IDAs are independent of the ranking method and, thus, they could possibly be improved by incorporating metalearning to generate rankings based on past performance.

Chapter 5

The Road Ahead

Metalearning is still a relatively young area of research. Although efforts so far have demonstrated promise, there is still much room for improvement as well as the development of new ideas and systems. The purpose of this tutorial was to review the state-of-the-art and spark interest in this exciting field of study.

We conclude with a few thoughts on some interesting directions for further research and suggestions on how to become involved.

5.1 Promising Avenues of Research

A recent paper by Vanschoren and Blockeel outlines most of the issues raised by metalearning, such as the curse of dimensionality (due to the size of the problem space), the impact of parameter settings on various algorithms, the need to better understand learning behavior, and the impact of data transformation on the machine learning process [55]. The focus is on trying to determine not so much when certain algorithms work or fail, but rather why they do. They argue, as in IDA, that “ideally, our advice to the user should be a ranked list of machine learning ’plans,’ stating interesting learning algorithms combined with the preprocessing steps that may increase their performance.” The authors also outline their experiment database framework, whose purpose is to collect and organize all data relevant to metalearning (e.g., data characteristics, algorithm parameter settings, algorithm properties, performance measures, etc.). A subsequent paper further describes the benefits of building experiment databases and report on the design and implementation of one such database (see http://expdb.cs.kuleuven.be/expdb/) [10]. It also shows how the database can be used to produce new information, test hypotheses and verify existing hypotheses or results, through the use of what can be seen as metalevel analyses or experiment mining. The current database has been populated with details of over 650,000 experiments; it is extendible and public. This is a tremendous contribution to the community and a serious boost to metalearning research.

Most of the work in metalearning has focused on characterizing problems, i.e., designing f functions. This is clearly important, and certainly an essential element of Rice's framework. Yet, one may argue that efforts should also be expended on characterizing learning algorithms and gaining a better understanding of their behavior. With the exception of a few attempts (e.g., see [31]), very little work has been done in this area. Recently researchers have replaced accuracy with classifier output distance (a measure similar to error correlation, which measures how often models induced by different learning algorithms differ in their prediction) to capture algorithm behavior, and cluster learning algorithms [44]. Our own current efforts focus on the analysis of learning curves, as we feel these may also reveal intrinsic properties of learning algorithms not readily attainable through other measures.

More work is needed in the defining and effectively operationalizing of multi-criteria performance measures, as well as the design of truly incremental systems, where new problems and new (base-level) algorithms may be continually added without retraining the system.

5.2 Getting Involved

For those new to metalearning, a great place to start is [16]. The book covers the main concepts and techniques of metalearning for model selection. The survey in [51] is a great complement, as it is the first to bring metalearning clearly under Rice's framework, and to recast a number of metalearning studies within that framework.

For those interested in going further with metalearning, there is a standing invitation to join the Google Meta-learning Group at:

http://groups.google.com/group/meta-learning

Bibliography

[1] Aha, D. (1992). Generalizing from Case Studies: A Case Study. In *Proceedings of the Ninth International Conference on Machine Learning*, 1-10.

[2] Andersen, P. and Petersen, N.C. (1993). A Procedure for Ranking Efficient Units in Data Envelopment Analysis. *Management Science*, **39**(10):1261-1264.

[3] Ayres, I. (2007). *Super Crunchers: Why Thinking-by-Numbers is the New Way to be Smart*, Bantam Books.

[4] Bensusan, H. (1998). Odd Bites into Bananas Don't Make You Blind: Learning about Simplicity and Attribute Addition. In *Proceedings of the ECML Workshop on Upgrading Learning to the Meta-level: Model Selection and Data Transformation*, 30-42.

[5] Bensusan, H. and Giraud-Carrier, C. (2000). Discovering Task Neighbourhoods through Landmark Learning Performances. In *Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases*, LNCS 1910, 325-330.

[6] Bensusan, H., Giraud-Carrier, C. and Kennedy, C. (2000). A Higher-order Approach to Meta-learning. In *Proceedings of the ECML Workshop on Meta-learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, 109-118.

[7] Bernstein, A. and Provost, F. (2001). An Intelligent Assistant for the Knowledge Discovery Process. In *Proceedings of the IJCAI Workshop on Wrappers for Performance Enhancement in KDD*.

[8] Bernstein, A., Provost, F. and Hill, S. (2005). Toward Intelligent Assistance for a Data Mining Process: An Ontology-based Approach for Cost-sensitive Classification. *IEEE Transactions on Knowledge and Data Engineering*, **17**(4):503-518.

[9] Berrer, H., Paterson, I. and Keller, J. (2000). Evaluation of Machine-learning Algorithm Ranking Advisors. In *Proceedings of the PKDD Workshop on Data-Mining, Decision Support, Meta-Learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions*.

[10] Blockeel, H. and Vanschoren, J. (2007). Experiment Databases: Towards an Improved Experimental Methodology in Machine Learning. In *Proceedings of the Eleventh European Conference on Principles and Practice of Knowledge Discovery in Databases*, LNCS 4702, 6-17.

[11] Botía, J.A., Gómez-Skarmeta, A.F., Garijo, M. and Velasco, J.R. (2000). A Proposal for Meta-Learning Through a Multi-Agent System. In *Proceedings of the Agents Workshop on Infrastructure for Multi-Agent Systems*, 226-233.

[12] Botía, J.A., Gómez-Skarmeta, A.F., Valdés, M. and Padilla, A. (2001. METALA: A Meta-learning Architecture. In *Proceedings of the International Conference, Seventh Fuzzy Days on Computational Intelligence, Theory and Applications*, LNCS 2206, 688-698.

[13] Brazdil, P. and Henery, B. (1994). Analysis of Results. In Michie, D. et al. (Eds.), *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, Chapter 10.

[14] Brazdil, P., Gama, J. and Henery, B. (1994). Characterizing the Applicability of Classification Algorithms Using Meta-Level Learning. In *Proceedings of the Seventh European Conference on Machine Learning*, 83-102.

[15] Brazdil, P., Soares, C. and Pinto da Costa, J. (2003). Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results. *Machine Learning*, **50**(3):251-277.

[16] Brazdil, P., Giraud-Carrier, C., Soares, C. and Vilalta, R. (2009). *Metalearning: Applications to Data Mining*, Springer-Verlag.

[17] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C. and Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step Data Mining Guide*, SPSS, Inc.

[18] Charest, M. and Delisle, S. (2006). Ontology-guided Intelligent Data Mining Assistance: Combining Declarative and Procedural Knowledge. In *Proceedings of the Tenth IASTED International Conference on Artificial Intelligence and Soft Computing*, 9-14.

[19] Craw, S., Sleeman, D., Granger, N., Rissakis, M. and Sharma, S. (1992). Consultant: Providing Advice for the Machine Learning Toolbox. In Bramer, M. and Milne, R. (Eds.), *Research and Development in Expert Systems IX (Proceedings of Expert Systems'92)*, SGES Publications, 5-23.

[20] Davenport, T.H. and Harris, J.G. (2007). *Competing on Analytics*, Harvard Business School Press.

[21] Duda, R.O., Hart, P.E. and Stork, D.G. (2001). *Pattern Classification*, Second Edition, John Wiley & Sons, Inc.

[22] Engels, R. and Theusinger, C. (1998). Using a Data Metric for Offering Preprocessing Advice in Data-mining Applications. In *Proceedings of the Thirteenth European Conference on Artificial Intelligence*, 430-434.

[23] Euler, T. and Scholz, M. (2004). Using Ontologies in a KDD Workbench. In *Proceedings of the ECML/PKDD Workshop on Knowledge Discovery and Ontologies*, 103-108".

[24] Euler, T. (2005). Publishing Operational Models of Data Mining Case Studies. In *Proceedings of the ICDM Workshop on Data Mining Case Studies*, 99-106.

[25] Fuernkranz, J. and Petrak, J. (2001) An Evaluation of Landmarking Variants. In *Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-learning*, 57-68.

[26] Gama, J. and Brazdil, P. (1995). Characterization of Classification Algorithms. In *Proceedings of the Seventh Portuguese Conference on Artificial Intelligence*, 189-200.

[27] Giraud-Carrier, C. (1998). Beyond Predictive Accuracy: What?. In *Proceedings of the ECML Workshop on Upgrading Learning to the Meta-Level: Model Selection and Data Transformation*, 78-85.

[28] Giraud-Carrier, C. (2005). The Data Mining Advisor: Meta-learning at the Service of Practitioners. In *Proceedings of the Fourth International Conference on Machine Learning and Applications*, 113-119.

[29] Giraud-Carrier, C. and Provost, F. (2005). Toward a Justification of Meta-learning: Is the No Free Lunch Theorem a Show-stopper? In *Proceedings of the ICML Workshop on Meta-learning*, 9-16.

[30] Hernansaez, J.M., Botía, J.A. and Gómez-Skarmeta, A.F. (2004). A J2EE Technology Based Distributed Software Architecture for Web Usage Mining. In *Proceedings of the Fifth International Conference on Internet Computing*, 97-101.

[31] Hilario, M. and Kalousis, A. (2000). Quantifying the Resilience of Inductive Classification Algorithms. In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery*, LNCS 1910, 106-115.

[32] Holte, R.C. (1993). Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning*, **11**:63-91.

[33] Hume, D. (1740). *A Treatise of Human Nature.* (Edited by D.F. Norton and M.J. Norton, Oxford University Press, 2000).

[34] Kodratoff, Y., Sleeman, D., Uszynski, M., Causse, K. and Craw, S. (1992). Building a Machine Learning Toolbox. In Steels, L. and Lepape, B. (Eds), *Enhancing the Knowledge Engineering Process*, Elsevier Science Publishers, 81-108.

[35] Leite, R. and Brazdil, P. (2004). Improving Progressive Sampling via Meta-learning on Learning Curves. In *Proceedings of the Fifteenth European Conference on Machine Learning*, 250-261.

[36] Leite, R. and Brazdil, P. (2005). Predicting Relative Performance of Classifiers from Samples. In *Proceedings of the Twenty-second International Conference on Machine Learning*, 497-503.

[37] Lim, T-S., Loh, W-Y. and Shih Y-S. (2000). A Comparison of Prediction Accuracy, Complexity and Training Time of Thirty-Three Old and New Classification Algorithms. *Machine Learning*, **40**:203-228.

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

[39] ML-List (1994). *LCG Discussion Thread*, Machine Learning List, Vol. 6, Nos. 19-27 (online at: ftp://ftp.ics.uci.edu/pub/ml-list/V6).

[40] Montgomery, A. (1998). Data Mining: Business Hunching, Not Just Data Crunching. In *Proceedings of the Second International Conference on the Application of Knowledge Discovery and Data Mining*, 39-48.

[41] Morik, K. and Scholz M. (2004). The MiningMart Approach to Knowledge Discovery in Databases. In Zhong, N. and Liu, J. (Eds.), *Intelligent Technologies for Information Analysis*, Springer, 47-65.

[42] Peng, Y., Flach, P.A., Brazdil, P. and Soares, C. (2002). Improved Data Set Characterisation for Meta-learning. In *Proceedings of the Fifth International Conference on Discovery Science*, 141-152.

[43] Perlich, C., Provost, F. and Simonoff, J.S. (2003). Tree Induction vs. Logistic Regression: A Learning-Curve Analysis. *Journal of Machine Learning Research*, **4**(2):211-255.

[44] Peterson, A.H. and Martinez, T.R. (2005). Estimating the Potential for Combining Learning Models. In *Proceedings of the ICML Workshop on Meta-Learning*, 68-75.

[45] Pfahringer, B., Bensusan, H. and Giraud-Carrier, C. (2000). Meta-learning by Landmarking Various Learning Algorithms. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 743-750.

[46] Provost, F., Jensen, D. and Oates, T. (1999). Efficient Progressive Sampling. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, 23-32.

[47] Rendell, L., Seshu, R. and Tcheng, D. (1987). Layered Concept-Learning and Dynamically-variable Bias Management. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, 308-314.

[48] Rendell, L. and Cho, H. (1990). Empirical Learning as a Function of Concept Character. *Machine Learning*, **5**:267-298.

[49] Rice, J.R. (1976). The Algorithm Selection Problem. *Advances in Computers*, **15**:65-118.

[50] Schaffer, C. (1994). A Conservation Law for Generalization Performance. In *Proceedings of the Eleventh International Conference on Machine Learning*, 259-265.

[51] Smith-Miles, K.A. (2009). Cross-disciplinary Perspectives on Meta-learning for Algorithm Selection. *ACM Computing Surveys*, to appear. (available as a technical report at http://www.deakin.edu.au/∼katesm/techreports/TechreportACMComp Surveys.pdf).

[52] Soares, C., Petrak, J. and Brazdil, P. (2001). Sampling-Based Relative Landmarks: Systematically Test-Driving Algorithms Before Choosing. In *Proceedings of the Tenth Portuguese Conference on Artificial Intelligence*, 13-38.

[53] Sohn, S.Y. (1999). Meta Analysis of Classification Algorithms for Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**(11):1137-1144.

[54] Utgoff, P.E. (1986). Shift of Bias for Inductive Concept Learning. In Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Volume II, Morgan Kaufmann Publishers. Inc., Chapter 5.

[55] Vanschoren, J. and Blockeel, H. (2006). Towards Understanding Learning Behavior. In *Proceedings of the Annual Machine Learning Conference of Belgium and the Netherlands*, 89-96.

[56] van Someren, M. (2000). Model Class Selection and Construction: Beyond the Procustean Approach to Machine Learning Applications. In Paliouras, G., Karkaletsis, V. and Spyropoulos, C.D. (Eds.), *Machine Learning and Its Applications: Advanced Lectures*, LNCS 2049, Springer-Verlag, 196-217.

[57] Vilalta, R. and Oblinger, D. (2000). A Quantification of Distance-Bias Between Evaluation Metrics in Classification. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 1087-1094.

[58] Witten, I.H. and Eibe, F. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*, Second Edition, Morgan Kaufmann, San Francisco, CA.

[59] Wolpert, D.H. and Macready, W.G. (1995). No Free Lunch Theorems for Search. Technical Report SFI-TR-95-02-010, Santa Fe Institute.

[60] Wolpert, D.H. (2001). The Supervised Learning No-Free-Lunch Theorems. In *Proceedings of the Sixth On-line World Conference on Soft Computing in Industrial Applications*, 325-330.

[61] Wolpert, D.H. (2001). Any Two Learning Algorithms Are (Almost) Exactly Identical. Technical Report, NASA Ames Research Center. (A shorter version also appeared in the Proceedings of the ICML-2000 Workshop on What Works Well Where).