Metalearning for Data Mining and KDD

Rostislav Stríž
istriz@fit.vutbr.cz

Department of Information Systems
Faculty of Information Technology
Brno University of Technology

Abstract. Data mining and machine learning techniques help us to better and deeper understanding of collected data. Metalearning techniques extend this concept by providing methods for knowledge discovery process automatization. Metalearning introduces various interesting concepts, including data metafeatures, metaknowledge, algorithm recomendation systems, autonomous process builders, etc. All these techniques aim to improve usually expensive and demanding data mining analysis. This report focuses on general overview of basic data mining, machine learning and metalearning techniques, while focusing on state-of-the-art, basic formalisms and principles, interesting applications and possible future developement in the field of metalearning.

Keywords: data mining, machine learning, metalearning, KDD, IDA, DMA

1 Introduction

The amount of collected data is growing persistently. What we are missing is a sufficient amount of adequately deep and interesting knowledge about that data. Data mining and machine learning provide methods for discovering such knowledge; unfortunattely, both fields of research are so wide and constantly growing that only specialized experts are capable of developing complex data mining processes and extracting required knowledge from the different kinds of data. Even then, there are many obstacles to be tackled in order to produce satisfactory results – analysts have to perform various experiments, which are often led by trial-and-error and subsequently lead to repetitive processing. It is important to note that such processes may be very time/space-consuming making this approach undesirable. *Metalearning* presents several principles that should enable the data mining system to be able to automatically learn from its previous experience when facing a new (although, similar to some previously solved) data mining task. Metalearning system should be able to change the *bias* of a given data mining process, learn and gain new experience during execution of the process with ideally only small amount of user's input. The main advantage of this approach is that the user is not required to have too much data mining experience; instead, the system itself is able to make the key decisions.

The main purpose of this report is to briefly review the state-of-the-art in this field of research. Chapter 2 presents a basic introduction into the data mining enviroment, chapter 3 focuses on metalearning, its possible applications, basic formalisms, principles and history. In chapter 4 we mention the most significant metalearning system implementations. Chapter 5 concludes the report and briefly discusses current trends and possible future developement in the field of metalearning.

2 Data Mining and Machine Learning

As mentioned in previous section, our society is overwhelmed with huge amounts of collected data. Data collecting and storaging is a regular practice for most of today's companies and businesses. Data mining and machine learning offer methods and techniques focused on obtaining information from the data and thus providing us with their deeper understanding.

Following sections describe some basic principles of data mining and machine learning approaches.

2.1 Data Mining Process

Nowadays, the term *data mining* is often used as a synonym for **knowledge discovery from data** (KDD), which is a process that can provide *new*, *interesting*, *non-trivial*, *hidden* and *potentially useful* knowledge about collected data. In practice, KDD is used on very large datasets where there is no easy way to get such knowledge otherwise.

Data mining is technically only a small part of the whole *search-for-knowledge* process. The process itself contains these following parts from [1]:

1. *data cleaning* – removes inconsistencies in the source datasets,
2. *data integration* – data from different sources have to be combined properly,
3. *data selection* – task-relevant data are retrieved from the source,
4. *data transformation* – data have to be transformed to appropriate task-specific form,
5. *data mining* – appropriate algorithms extract data patterns,
6. *pattern evaluation* – interesting patterns are extracted based on different measures,
7. *knowledge presentation* – visualization and knowledge representation to users.

The first 4 steps are often refered to as *data preprocessing* (where the data are prepared for mining). This part may often be one of the most important and demanding ones as the result of the data mining task is directly affected by the quality of the preprocessed source data. At the end of the whole process, the interesting patterns are sometimes being stored in a system **knowledge base** in a form of a new knowledge. This technique presents an easy and convenient way for storing and subsequently browsing, comparing and visualizing obtained knowledge or even groups of related knowledge.

2.2 Data Mining Results

There is a huge variety of different results we can obtain using data mining and machine learning. In the scope of this report, we won't go into many details about different data mining techniques since it is not our main focus; however, we are going to note, that there are two main types of data mining tasks:

- **descriptive** – the result of the task are patterns describing the source data,
- **predictive** – the result of the task is an applicable model (or models) with predictive abilities.

To give an example, online retailer has a database with all his customers and their previous orders. With the use of *descriptive* data mining techniques it is possible to associate which merchandise is selling together (using so-called *association analysis*). But we could also use certain *predictive* methods to categorize the retailer's customers into certain groups (e.g., with regards to monthly spendings) so we would be able to predict a new customer's behaviour (in this case how much he will spend in the store monthly) based on, e.g., his age, location, first few purchases, etc. More on this subject can be found in [1].

2.3 Machine Learning

Machine learning is one of many domains data mining derive its techniques from. Machine learning focuses on *automatic computer learning* that is capable of making own decisions based on data. There are several types of machine learning tasks:

- *supervised learning* – system is learning from the labeled examples in the training dataset,
- *unsupervised learning* – system is learning from unlabeled set of training data discovering target classes on the fly,
- *semi-supervised learning* – system uses both labeled and unlabeled examples learning the model from the labeled data and using unlabeled examples to refine class boundaries,
- *active learning* – user is actively participating in the learning process by, e.g., labeling unlabeled example on demand.

Term machine learning is sometimes used to address a subset of data mining methods as , e.g., classification may be described as supervised learning and clustering as unsupervised learning.

2.4 Data Mining Data Sources

The most common data source for data mining application is a relational database. Another common sources are *transactional databases* that capture transactions, such as customer's purchases, etc., which are indetified by transaction identity number and include a list of transaction items. Besides relational and transaction databases, there are many other forms of databases differing mainly in their semantics. As an example, we can mention temporal databases, spatial and spatio-temporal databases.

Aside from basic database structures, many companies store their big data in so-called *data warehouses.* Data warehouses are esentially a repositaries of information collected from many different sources under the same schema [1]. A data warehouse is usually presented in a form of a *mulditimensional data cube*, where each *dimension* represents an attribute (or a set of attributes), while the cells themselves store a value of some aggregate measure over chosen dimensions. Data warehouse systems provide tools for *online analytical processing* (OLAP) for interactive analysis of multidimensional data. OLAP enables analysts to view and change a level of abstraction and granularity of displayed measures, as well as arbitrary combine different data dimensions.

Another sources usable for data mining are data streams (infinite continuous streams of data without possibility to rewind or save all records), graph data, hypertext and multimedia data, and the Web. In regards to Web data sources, in recent years, *cloud systems* are rapidly gaining popularity, while the word *cloud* became a huge *buzzword.* The main principle of cloud computing is an idea that everything is stored and performed on external servers that are always accessable over the network. Such services are usually outsourced and provided to users with seemingly unrestricted access to their content. The number of cloud storages and their users grow every year. More information about cloud computing can be found in [2].

To extend on the concept of cloud computing, because of the amount of data that is processed by such systems, it is impossible to store the data in convetional manner. These so-called *big data* (more on this phenomenon in [3]) are often stored in *distributed data storages* accross many storage units. It is obvious, that all operations performed over such data need to be optimized for distributed architecture. To achieve required functionality, *Google* came up with solution in a form of *MapReduce* model. This model automatically parallelizes the computation across large-scale clusters of machines, handles machine failures, and schedules inter-machine communication to make efficient use of the network and disks [4]. There are many concrete system implementation using this principle, one of the most known and used is *Apache Hadoop.* Hadoop is basically an open-source framework that supports large cluster applications by using its own distributed file system (*HDFS*). There are also many Hadoop extensions, one of them being *Apache Hive*, which adds data warehouse infrastructure to the Hadoop system, allowing users to query, summarize, and analyze saved data. Regarding data mining, *Apache Mahout* is a scalable machine learning library that can work together with Hive and Hadoop to perform some basic data mining tasks. Complete overview of Hadoop and related technologies can be found in [5].

2.5 Data Mining Issues

We may address many issues connected to data mining. One of them may be that data mining as a scientific field is very large already and growing every moment.

There is countless amount of specific applications and techniques involved, while new kinds of knowledge and algorithms are being discovered constantly.

Another issue is related to data mining performance - algorithms should be as efficient and scalable as possible. Due to the existence of many types of data sources, mentioned in section 2.4, it may also be difficult (sometimes even impossible) to transfer newly discovered methods to another data source architectures.

Other issues involve directly the data themselves – either regarding their structure (complex data types), or their content (missing values, noise, imbalance). All the main issues are described in [1] in much greater detail.

The last issue we will mention in this report is **user interaction** during the data mining process. User interaction was already mentioned in section 2.3, dividing machine learning tasks into groups based on the type and the amount of interaction. Generally, data mining is highly interactive by its nature. This approach allows users to use their past experience, understanding of the examined domain and any type of additional background knowledge. A good domain knowledge, as well as sufficient knowledge of different data mining techniques and methods are necessary requirements for possibly successfull process result (Fig. 1). Current machine learning/data mining tools are only as powerful/useful as their users.

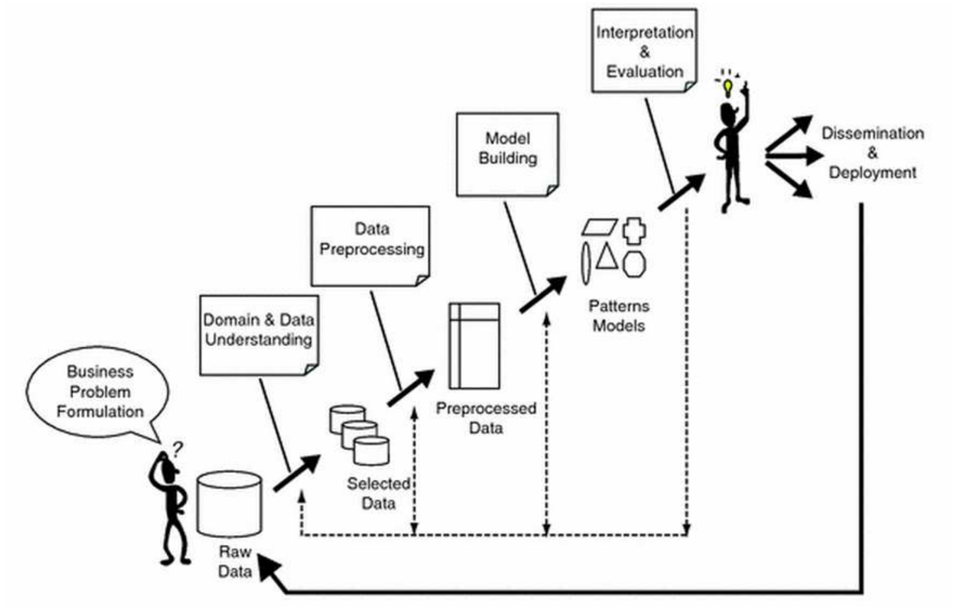

Interpretation & Evaluation
Model Building
Data Preprocessing
Domain & Data Understanding
Business Problem Formulation
?
Dissemination & Deployment
Patterns Models
Preprocessed Data
Selected Data
Raw Data

Fig. 1. The data mining process – source [6]

3 Metalearning

In the previous subsection 2.5, we stated that user interactivity in data mining process is, to some extent, necessary to achieve desired results. On the other hand, one of the mentioned issues regarding data mining is a huge variety of different evolving techniques and methods. Moreover, most of the time, ideal structure of KDD process is not clear from the beginning of the analysis. This forces analysts to constantly reconfigure and alter the process itself either during its course or (probably more often) at the end after viewing and analyzing its results – this is often very much a *trial-and-error*-based procedure. When taking into consideration the time and the computation complexity most of the complicated data mining tasks may introduce, to aquire desired results may be very time and resource-consuming process, demanding repeated user input.

In contrast to that, many complicated data mining applications would benefit if they could be performed somewhat automatically with only limited amount of user input. This is one of the main ideas of *metalearning*. Metalearning introduces intelligent data mining processes with the ability to learn and adapt based on previously aquired experience. This limits the amount of user input necessary to perform informed data analyzation task, which may be good either for runing multiple tasks at once without overwhelming the analyst, or for automatic decision making without any need for user intervention when the user himself may lack the expertise. Moreover, such system can learn from every new task, thus being more experienced and informed over time, providing new levels of adaptation to newly introduced obstacles. This area of research is also refered to as *learning to learn*.

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. Learning at the metalevel is concerned with accumulating experience on the performance of multiple applications of a learning system. The main aim of current research is to develop metalearning assistant,which are able to deal with the increasing number of models and techniques, and give advice dynamically on such issues as model selection and method combination.

More about the basics purpose of metalearningcan be found in [7], [8] and [9].

3.1 Metalearning vs. Base-Learning

There are severe differences between metalearning and traditional view of learning (also called base-learning). We already briefly discussed a structure of common data mining task in section 2.1 – to sum up: we apply a base-learner (e.g., decision tree) on source data and it produces a predictive model/function that depends on the *fixed* assumptions embedded in the learner. The predictive function/model usually improves with an increasing number of examples; however, successive applications of the learner on the same data will produce the same hypothesis [6]. Also, no knowledge is extracted from the learning process itself to be transportable across tasks or domains.

Based on previous paragraph, we can determine the two major limitations to base-learning approach:

- Data patterns are usually embedded in the predictive model itself – successive training of the same learner over the same data fails to aquire any additional knowledge.
- There is no easy way of extracting, sharing and reusing aquired knowledge among different domains or analysts.

A key to solving these issues is gathering knowledge about the learning process in form of so-called *metaknowledge*. The main purpose of metalearning is then to find a best way to extract and describe such metaknowledge to then use it to alter and improve the learning process.

3.2 Basic Areas of Metalearning Application

According to [6], there are several basic applications of metalearning:

- selecting and recommending machine learning algorithms,
- employing metalearning in KDD,
- employing metalearning to combine base-level machine learning systems,
- control of the learning process and bias management,
- transfer of metaknowledge across domains

More details on mentioned applications are provided in following sections.

3.3 History of metalearning

As an early precursor of metalearning, STABB system may be introduced, since it was the first to show that a learner's bias could be dynamically adjusted [10]. Next, VBMS (*variable-bias management system*) was developed as 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 [11].

The first formal attempts at addressing the practice of machine learning by producing rich toolbox consisting of a number of symbolic learning algorithms for classification, datasets, standards and know-how were introduced in [12] in a form of the MLT project. During this project, considerable insight into many important machine learning issues was gained. Based on that, the user guidance system Consultant-2 was developed. Consultant-2 is a kind of expert system for algorithm selection - it provides the user with interactive question-answer sessions that are intended to collect information about the data, the domain and user preferences. Consultant-2, presented in [13], stands out as the first automatic tool that systematically relates application and data characteristics to classification learning algorithms.

The StatLog project extended VBMS by considering a larger number of dataset characteristics, together with a broad class of candidate models and algorithms for selection [14]. Later, in [15], a statistical metamodel to predict the expected classification performance of 11 learning algorithms as a function of 13 data characteristics was designed.

The CRISP-DM project aimed at developing a cross-industry standard process for data mining. It covers the entire data mining process, from definition of business objectives, data mining goals, data access, preparation, exploration and modelling, to results interpretation, assessment, deployment and documentation. Although, this project is not addressing metalearning directly, it is relevant as it formalizes the data mining process and hence the various decision points where metalearning may assist. More about CRISP-DM may be found in [16].

Later, a Web-based metalearning system for the automatic selection of classification algrotihms, named **DMA** (*Data Mining Advisor*), was developed as the main deliverable of the METAL project. This project focused on discovering new and relevant data/task characteristics, and using metalearning to select best suitable classifiers for a given task. 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.

Other system, called **IDA** (*Intelligent Discovery Assistant*), provides a template for building ontology-driven, process-oriented assistant for the KDD process. It includes operations from the three basic steps of KDD - preprocessing, model building and post-processing. The main goal of IDA is to generate a list of ranked DM processes that are congruent with user-defined preferences by combining possible operations accordingly. This aproach was presented in [17] and [18]. [19] then extends described concept by using both declarative information (ontology) as well as procedural information (system rules).

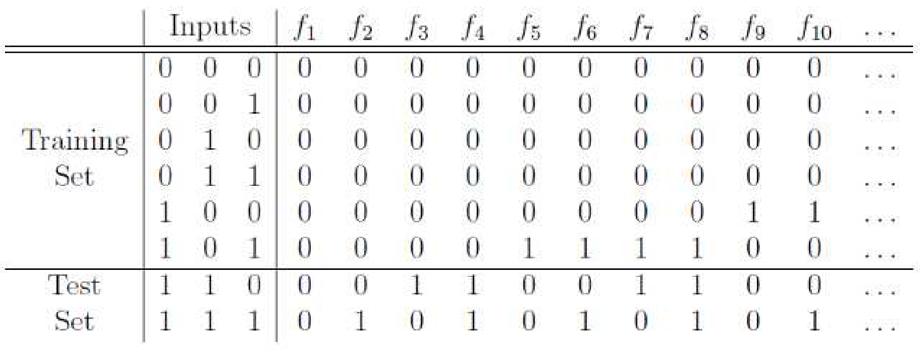
Finally, in [20], most of the issuses surrounding model class selection are adressed as well as a number of methods for the selection itself. Authors also propose a taxonomy of metalearning study types.

3.4 Algorithm Recommendation

Probably the most straight-forward application of metalearning is the automatic algorithm recommendation. Almost every machine learning task can be performed via using different methods and algorithms. Our goal is to achieve the best results possible, no matter what the main quality indicator may be (performance, confidence, or other). However, to perform the same machine learning task over and over again while substituting algorithms may be enormously time-consuming.

NFL theorem

Before we procees, we briefly discuss some theoretical consideration for the use of metalearning in algorithm selection. The *No Free Lunch* (NFL) theorem was introduced in [21] and [22]. In the context of machine learning, it is also known as a *Law of Conservation for Generalization Performance* (LGC). As a simple illustration of the NFL theorem, consider the simple space, Γ, of binary functions defined over $\mathbb{B}^3 = \{0,1\}^3$, and assume that the instaces of set $T_r = \{000, 001, ..., 101\}$ are observed. The instances of set $T_e = \mathbb{B}^3 - T_r = \{110, 111\}$ constitute the off-training test set. The situation is shown in Fig. 2.



	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	...

Fig. 2. Sample train/test setting for binary functions over 3 boolean variables – source [9]

The NFL theorem in this setting shows that the behavior on T_e of any learner $(f_1, f_2, ..., f_{256})$ trained on T_r is that of a random guesser (can be seen, e.g., while considering functions f_1 through f_4 in Fig. 2). It is apparent that the NFL theorem in essence simply restates Hume's conslusion about induction having no rational basis [23]: "There can be no demonstrative argument to prove, that those instances, of which we have had no experience, resemble those, of which we have experience. . . "

The crucial contribution of the NFL theorem, is poiting out that whenever a learning algorithm performs well on some function it must perform poorly on some others. Hence, builidng decision support system for what learning algorithm works well where becomes a valuable effort.

Ultimate Learning Algorithm

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* is a learning algorithm that induces a model M^*, such that $\forall M^{'} \neq M^* E(\delta(p^*, p^{\Omega})) \leq E(\delta(p^{'}, p^{\Omega}))$, where the expectation is computed for a given training set partition of the instance space, over the entire function space, and δ is some appropriate distance measure.

Problem formalization

Back in 1976, Rice proposed a formalization of the algorithm selection problem in [24]. From there, Rice's framework for algorithm selection emerged. A survey or many metalearning approaches within Rice's framework can be found in [25]. Metalearning for algorithm selection corresponds to Rice's model depiced in Fig. 3, 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 (like, e.g., accuracy), p, of a on x is optimal. The selection

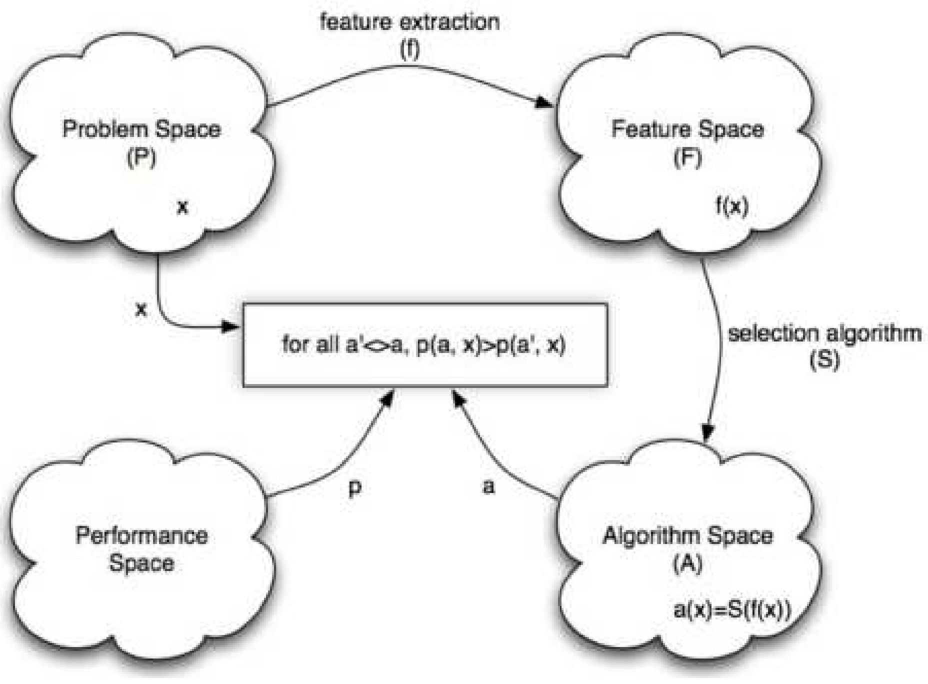


Fig. 3. Rice's framework for algorithm selection – adapted from [24]

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.

Simplified basic approach

The basic approach, presented by metalearning (Fig. 4), is to identify a suitable subset of learning algorithms given a training dataset, using avabile metaknowledge. The result of described process will be a set of ranked algorithms. The next phase is to evaluate each selected algorithm using various performacnce criteria to identify the best alternative. This method alters from traditional approaches in exploiting a metaknowledge base, which may, in this case, contain a set of learning algorithms that have shown good performance on similar datasets in prior ML tasks. It is important to note, that this approach won't generally completely eliminate the need for search, but rather provides a more effective way of searching through the space of alternatives. It is also clear that the effectiveness of the search process depends on the quality of the avabile metaknowledge [6].

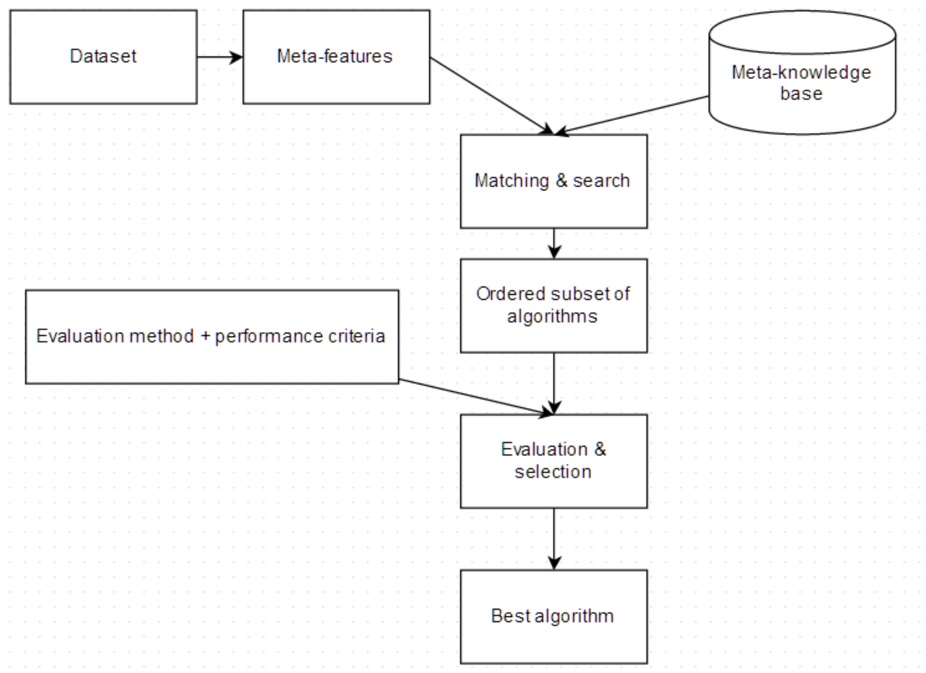


Fig. 4. Finding a reduced space and selecting the best learning algorithm – revorked from [6]

Metafeatures

To be able to implement the approach mentioned above, it is necessary to have the ability to characterize the source dataset. This can be resolved by using *metafeatures*. The idea is to gather descriptors about the data distribution that correlate well with the performance of learned models. There are three main classes of metafeatures so far:

– *simple statistical and information-theoretic metafeatures* – e.g., number of classes, number of features, ratio of examples to features, correlation between features, entropy . . . (more in [26], [27]),
– *model-based metafeatures* – exploiting properities of some induced hypothesis (e.g., building decision tree model from the data and collect its properities – depth, shape, imbalance . . . – more in [28]),

– *landmarkers* – exploiting dataset properties from the performance of a set of simple and fast learners with significantly different learning mechanisms (more in [29]). Various landmarking methods are evaluated in [30].

A schema of obtaining metaknowledge for algorithm selection is shown in Fig. 5.

After obtaining dataset metafeatures, we are able to identify a set of the most similar datasets for our given input dataset by using proper method, e.g., k-Nearest Neighbor method (details in [31]). Subsequently, we can use the metaknowledge we gathered while analyzing the selected (most similar) datasets to adjust our data mining process – in this case, to select the best algorithm for the given task (with the highest possible probability) by using data mining (in this case classification where the target class is the best algorithm to use) over our *metadatabase* (features being datasets' metafeatures).

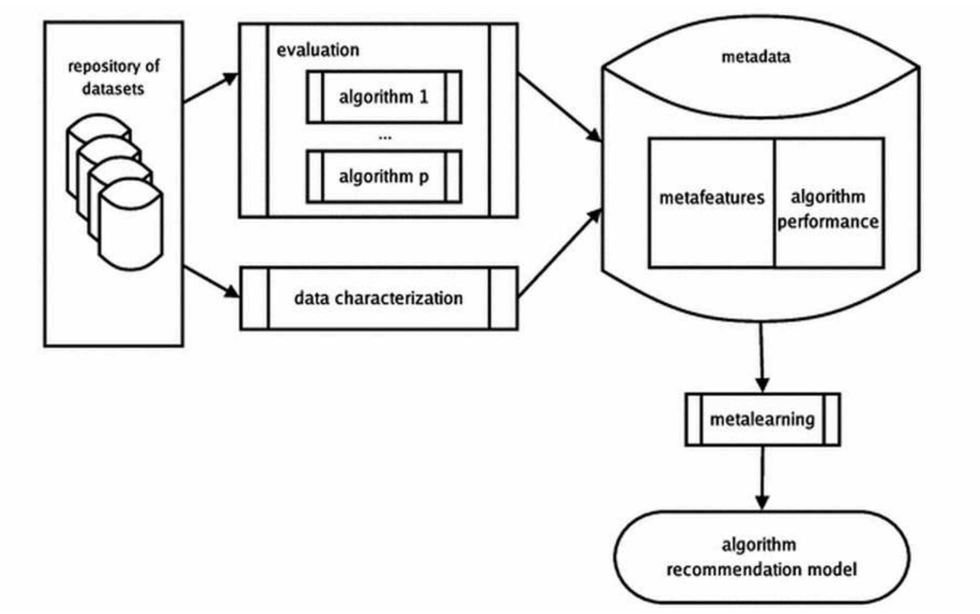


Fig. 5. Metalearning to obtain metaknowledge for algorithm selection – source [6]

3.5 Employing Metalearning in KDD

We can extend the concept mentioned in previous section 3.4 not only to select a ML/DM algorithm, but to build the whole KDD process. The KDD process can be viewed as a set of simple subsequent operations that can be further decomposed into smaller operations. These sequences can be characterized as partially ordered acyclic graphs and each partial order of operations can be regarded as an *executable plan* that produces certain effect. Examples can be found in [32].

The main goal, under this framework, is to automatically compose suitable executable plan with regard to the source data and, again, previous system experience. The problem of generating a plan may be formulated as identifying a partial order of operations to satisfy certain criteria or maximize certain evaluation measures [6]. Naturally, the difficulty of this optimization process raises with the rising number of possible operations.

Generally, there can be two ways of how to approach the generation of the new plan:

– The system begins with an empty plan and gradually extends it with the composition of operators. This approach was presented in [32].
– The system starts with a suitable previously used plan and adapts it further to the exact needs of current task. More on this approach can be found in [33].

Although the idea of completely automatic generation of KDD process might be very appealing, it is important to note that this approach is inherently difficult. There needs to be many possibiliries considered, some of them with high computational complexity. In the context of metalearning, metaknowledge can be used to facilitate this task. Past plans may be enriched with additional metainformation and can serve as *procedural metaknowledge.* Other metaknowledge may be captured about the applicability of existing plans to support reuse and on how they can be adapted to new circumstances. More information about this topic can be found in [6].

3.6 Combining Base-Level ML Systems

The approach of model combination is quite common nowadays, although it's not usually associated with the term metalearning. However, its principles correspond with the metalearning philosophy. Model combination consists of creating a single learning system from a collection of learning algorithms [6]. There are two basic approaches to this concept:

– System consists of multiple copies of a *single* algorithm that are applied to different subsets of the source data.
– System consists of a set of *different* mining algorithms that are trained over the same data.

The primary motivation for the model combination is usually to increase the accuracy of the final model; however, because it draws information about base-level learning (e.g., data characterization, algorithm characteristics . . .), methods for model combinations are often considered to be part of metalearning.

Perhaps the most known techiques for exploiting variation in data are *bagging* and *boosting.* They combine multiple models built from a single learning algorithm by systematically varying the training data [6]. *Bagging*, introduced in [34], produces replicate training sets by *sampling with replacement* from the set of training instances. This training set is the same size as the original data, but some tuples may not appear in it while others appear more than once (hence "*with replacement*"). *Boosting* (from [35]), on the other hand, maintains a weight for each training data instance – the higher the weight, the more the instance influences the classifier. At each trial, the vector of weights is adjusted to reflect the performance of the corresponding classifier in such way that the weight of misclassified instances is increased [36]. Bagging and boosting are easily applicable to various base-level learners and are proven to successfully increase the classification accuracy of created result models.

Bagging and boosting exploit variation in the source data, thus they are methods belonging to the first mentioned model combination concept. *Stacking* and *cascade generalization* are then methods belonging to the second mentioned concept – they combine multiple learners to create a new learning method. *Stacking* creates a new learner that builds a metamodel mapping the precitions of the base-level learners to target classes. This method was presented in [37]. *Cascade generalization*, described in [38], also builds a metalearner but rather then building it based on parallel results from base-level learners, it builds it subsequently – results of every base-level learner are enriched of metainformation and given on to the next base-level learner creating a chain-like structure.

More proposed methods for model combination metalearning, including *cascading*, *delegating*, *arbitrating* and *meta-decision trees*, are described in [6].

3.7 Control of the Learning Process

In this section we consider situations where our dataset is potentially infinite (e.g., continuous data streams). The main issue in this case is that we cannot predict what data are we going to get in the future, which is very limiting in the case of selecting the ideal ML algorithm. According to [6], we can distinguish among several different strategies:

- *Active learning*([39]) – data are processed in batches, after the initial model is created from the first batch using selected ML algorithm, the key is to select only informative examples from the next batch while ignoring the rest.
- *Controlling learning* – characterization of a new dataset is done progressively, different algorithms are tested on samples of increasing size, while every result determine what should be done in the next phase. This method is essentially trying to select the most appropriate base-algorithm at each time.
- *Learning from data streams* – the aim of this method is to be able to rebuild the result model if necessary. It is possible to start learning using one ML algorithm and with the addition of new examples to then change this algorithm and ultimately upgrade or recreate the basic result model. The quantity of data and data characteristics are used to determine whether the system should continue with the same model or not.

More details about mentioned approaches can be found in [6].

3.8 Metaknowledge Transfer Across Domains

Accumulating metaknowledge is one of the main targets of metalearning. The amount of aquired metaknowledge has a direct impact on the learning process itself – the methods prosper directly from greater amount of metaknowledge (concrete quantifications of such benefits can be found in [40]). Because of this principle, it would be convenient to be able to transfer aquired metaknowledge accross different domains, potentially accross different metalearning systems. This problem is also known as *inductive transfer*. Methods have been proposed for transporting metaknowledge across domains while preserving the original data mining/machine learning algorithm – there are methods for inductive trasnfer accross neural networks, kernel methods and parametric bayesian models (for more details on each method refer to [41]). There are also other methods of transfer that are not directly connected to concrete models, such as propabilistic transfer, transfer by feature mapping and transfer by clustering. However, the issue of knowledge transfer is quite complicated and to be able to create a method for unlimited inductive transfer one would have to create a standardized high level metaknowledge description language and corresponding ontology. For the complete overview and deeper description of inductive transfer methods and connected issues refer to [6].

4 Metalearning Systems

Metalearning in practice focuses on offering support for data mining. 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. In this chapter, which is based on the information from [6] and [9], we describe some of the most significant attemps at integrating metaknowledge in DM decision support systems. While usual data mining software packages (e.g., Rapid-Miner, Weka) provide user-friendly access to wide collections of algorithms and DM process building, they generally offer no real decision support for nonexpert users.

It is also obvious, that not all phases of the KDD process can/should be automatized. Usually, the early stages (problem formulation, domain understanding) and the late stages (interpretation and evaluation) require significant human input as they depend heavily on business knowledge.

Most of systems from this chapter has been already briefly mentioned in section 3.3; however, in following paragraphs we are going to describe the most interesting ones in greater detail.

4.1 MiningMart and Preprocessing

MiningMart, presented in [42] and [43], is a result of another large European research project focused on algorighm selection for data preprocessiong. As mentioned in section 2.1, preprocessing is generally very time consuming (acording to [6] almost 80% of the overall KDD process time) and it consists of nontrivial sequences of operations or data transformations. Because of that, the advantages of automatic user guidance are greatly appriciated. MiningMart provides a case-based reuse of successfull preprocessing phases across aplications. It uses a metadata model, refered to as M4, to capture information about data and operator chains through a user-friendly interface. MiningMart has its own case base and every new mining task leads to its search through while looking for the most appropriate case for the task at hand. After that, the system generates preprocessing steps that can be executed automatically for the current task. Similar efforts are also described in [44].

4.2 Data Mining Advisor and Ranking Classification Algorithms

The Data Mining Advisor (DMA) serves as a metalearning system for the automatic selection of model building classification algorithms. The user provides the system with a source dataset, specific goals in terms of result model accuracy and process training time; subsequently, DMA returns a list of algorithms that are ranked according to user-defined goals (currently, there are 10 different classification algorithms). The DMA guides the user through a step-by-step process wizard defining the source dataset, computing dataset characteristics, and setting up the ranking method via defining selection criteria and selecting the ranking mechanism.

4.3 METALA and Agent-Based Mining

METALA is an agent-based architecture for distributed data mining, supported by metalearning. It can be viewed as a natural extension of the DMA, mentioned in previous section. METALA provides the architectural mechanisms necessary to scale DMA up to any number of learners and tasks. Each learning algorithm is embedded in an agent that provides clients with a uniform interface so the system is able to autonomously and systematically perform experiments with each task and each learner to induce a metamodel for algorithm selection. When a new algorithm or new task are added to the system, it performs corresponding experiments and the metamodel is updated. More information about METALA can be found in [45] and [46].

4.4 Intelligent Discovery Assistants and Ranking Processes

The idea of Intelligent Discovery Assistant (IDA) (from [17] and [18]) focuses on the automatic construction of the whole KDD process rather than on its specific part – namely on preprocessing, model building and post-processing. IDAs are able to produce a chain of operations consisting of one or more operations from each of these steps. The goal here is to propose a list of ranked DM processes that are valid and congruent with user-defined preferences (e.g., build a fast, comprehensible classifier). The IDA's underlying ontology is essentialy a taxonomy of DM operations or algorithms, where the leaves represent implementations avabile in the corresponding IDA. Operations are characterized by preconditions, post-conditions and heuristic indicators [6]. The versatility of a concrete IDA depends directly on the richness of its ontology (i.e., the number of possible operations/algorithms).

User provides the system with an input dataset, user-defined objectives and information about the data that may not be obtained automatically. System then starts with an empty process while searching for an operation whose preconditions are met and whose indicators are congruent with the process preferences. Finally, all valid DM processes are computed. Subsequently, a heuristic ranker is applied to assist the user with the process ranking leading to the most efficient process selection. However, because the system provides all valid processes, it is possible to uncover novel processes that experts had never thought about before, thus enriching the community's metaknowledge.

Recent research [19] has focused on extending this concept by adding case-base reasoning (known, e.g., from mentioned MiningMart). The system then uses both declarative and procedural information; however, it is still in the early stages of implementation.

5 Conclusion

While data mining and machine learning provide sufficient tools for deep data analysis, a lack of experience or other resources may prolong the pursuit of desired data knowledge. Metalearning presents various techniques within different kinds of application to make the data mining process more autonomous, based on collected metaknowledge. It presents some new concepts, e.g., metaknowledge base, data metafeatures, their extraction, base-learner combinations and even continuous data stream data mining. Metalearning is a very variable field and its applications may severely differ. Many systems have been/are being developed to include different types of metalearning features; however, there is still much room for improvement as well as the developement of new ideas. In [47], the autors claim that the focus should be on trying to determine not so much when certain algorithms work or fail, but rather why they do. They also argue, 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."

Subsequently, in [48], the benefits of building experiment databases and reports on the design and implementation of such database are presented. Described database is currently publicly active and contains over 650000 experiments.

But most of the latest work in metalearning is focused on characterizing problems (designing the f function in context of discussed Rice's framework), when only a small amount of attention has been turned to characterizing learning algorithms and gaining a better understanding of their behavior. According to [9], more work is needed in the defining and effectively operationalizing of multicriteria 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.

References

1. Han, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2011)
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: A view of cloud computing. Commun. ACM **53**(4) (April 2010) 50–58
3. Dumbill, E.: What is big data: An introduction to the big data landscape (2012)
4. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. Commun. ACM **51**(1) (January 2008) 107–113
5. White, T.: Hadoop: The Definitive Guide. 1st edn. O'Reilly Media, Inc. (2009)
6. Soares, C., Giraud-Carrier, C., Brazdil, P., Vilalta, R.: Metalearning : Applications to Data Mining. Springer (2009)
7. Thrun, S.: Learning to learn: Introduction. In: In Learning To Learn, Kluwer Academic Publishers (1996)
8. Thrun, S.: Lifelong learning algorithms. In Thrun, S., Pratt, L., eds.: Learning to Learn. Springer US (1998) 181–209
9. Giraud-Carrier, C.: Metalearning-a tutorial. In: Tutorial at the 2008 International Conference on Machine Learning and Applications, ICMLA. (2008)
10. Utgoff, P.E.: Shift of bias for inductive concept learning. Machine learning: An artificial intelligence approach **2** (1986) 107–148
11. Rendell, L., Seshu, R., Tcheng, D.: Layered concept-learning and dynamically-variable bias management. In: In Proceedings of IJCAI-87, Morgan Kaufmann (1987) 308–314
12. Craw, S., Sleeman, D., Graner, N., Rissakis, M., Sharma, S.: Consultant: Providing advice for the machine learning toolbox. Proceedings of the Research and Development in Expert Systems IX (1992) 5–23
13. Sleeman, D., Rissakis, M., Craw, S., Graner, N., Sharma, S.: Consultant-2: Pre-and post-processing of machine learning applications. (1995)
14. Brazdil, P., Gama, J., Henery, B.: Characterizing the applicability of classification algorithms using meta-level learning. In: Machine Learning: ECML-94, Springer (1994) 83–102
15. Sohn, S.Y.: Meta analysis of classification algorithms for pattern recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on **21**(11) (1999) 1137–1144
16. Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R.: Crisp-dm 1.0. CRISP-DM Consortium (2000)
17. Bernstein, A., Provost, F.: An intelligent assistant for the knowledge discovery process. Information Systems Working Papers Series, Vol (2001)
18. Bernstein, A., Provost, F., Hill, S.: Toward intelligent assistance for a data mining process: An ontology-based approach for cost-sensitive classification. Knowledge and Data Engineering, IEEE Transactions on **17**(4) (2005) 503–518
19. Charest, M., Delisle, S.: Ontology-guided intelligent data mining assistance: Combining declarative and procedural knowledge. In: Artificial Intelligence and Soft Computing. (2006) 9–14
20. van Someren, M.: Model class selection and construction: Beyond the procrustean approach to machine learning applications. In: Machine Learning and Its Applications. Springer (2001) 196–217
21. Wolpert, D.H., Macready, W.G.: No free lunch theorems for search. Technical report, Technical Report SFI-TR-95-02-010, Santa Fe Institute (1995)

22. Wolpert, D.H.: The supervised learning no-free-lunch theorems. In: Soft Computing and Industry. Springer (2002) 25–42
23. Hume, D.: A treatise of human nature. Courier Dover Publications (2003)
24. Rice, J.R.: The algorithm selection problem. (1975)
25. Smith-Miles, K.A.: Cross-disciplinary perspectives on meta-learning for algorithm selection. ACM Computing Surveys (CSUR) **41**(1) (2008) 6
26. Engels, R., Theusinger, C.: Using a data metric for preprocessing advice for data mining applications. In: In proceedings of the Thirtheenth European Conference on Artifical Intelligence. (1998)
27. Sohn, S.Y.: Meta analysis of classification algorithms for pattern recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on **21**(11) (1999) 1137–1144
28. Peng, Y., Flach, P., Brazdil, P., Soares, C. In: Decision tree-based characterization for meta-learning. University of Helsinki (2002) 111 – 122 Conference Proceedings/Title of Journal: ECML/PKDD'02 workshop on Integration and Collaboration Aspects of Data Mining, Decision Support and Meta-Learning.
29. Pfahringer, B., Bensusan, H., Giraud-Carrier, C.: Meta-learning by landmarking various learning algorithms. In: In Proceedings of the Seventeenth International Conference on Machine Learning, Morgan Kaufmann (2000) 743–750
30. Fürnkranz, J., Petrak, J.: An evaluation of landmarking variants. In: Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning (IDDM-2001. (2001) 57–68
31. Brazdil, P., Soares, C., da Costa, J.: Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results. Machine Learning **50**(3) (2003) 251–277
32. Bernstein, A., Provost, F., Hill, S.: Toward intelligent assistance for a data mining process: an ontology-based approach for cost-sensitive classification. Knowledge and Data Engineering, IEEE Transactions on **17**(4) (2005) 503–518
33. Morik, K., Scholz, M.: The miningmart approach to knowledge discovery in databases. In: Intelligent Technologies for Information Analysis. Springer Berlin Heidelberg (2004) 47–65
34. Breiman, L.: Bagging predictors. In: Machine Learning. (1996) 123–140
35. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting (1997)
36. Quinlan, J.R.: Bagging, boosting, and c4.5. In: In Proceedings of the Thirteenth National Conference on Artificial Intelligence, AAAI Press (1996) 725–730
37. Wolpert, D.H.: Stacked generalization. Neural Networks **5** (1992) 241–259
38. Gama, J., Brazdil, P.: Cascade generalization. Machine Learning **41**(3) (2000) 315–343
39. Weiss, S., Indurkhya, N., Zhang, T., Damerau, F.: Text Mining: Predictive Methods for Analyzing Unstructured Information. SpringerVerlag (2004)
40. Baxter, J.: A model of inductive bias learning. Journal of Artificial Intelligence Research **12** (2000) 149–198
41. Pratt, L., Jennings, B.: A survey of connectionist network reuse through transfer. In Thrun, S., Pratt, L., eds.: Learning to Learn. Springer US (1998) 19–43
42. Morik, K., Scholz, M.: The miningmart approach to knowledge discovery in databases. In: Intelligent Technologies for Information Analysis. Springer (2004) 47–65
43. Kietz, J.U., Vaduva, A., Zücker, R.: Miningmart: Metadata-driven preprocessing. In: Proceedings of the ECML/PKDD Workshop on Database Support for KDD. Volume 3., Z001 (2001)

44. Phillips, J., Buchanan, B.G.: Ontology-guided knowledge discovery in databases. In: Proceedings of the 1st international conference on Knowledge capture, ACM (2001) 123–130
45. Botía, J.A., Gómez-Skarmeta, A.F., Velasco, J.R., Garijo, M.: A proposal for meta-learning through a mas (multi-agent system). In: Infrastructure for agents, multi-agent systems, and scalable multi-agent systems. Springer (2001) 226–233
46. Botía, J.A., Gómez-Skarmeta, A.F., Valdés, M., Padilla, A.: Metala: A meta-learning architecture. In: Computational Intelligence. Theory and Applications. Springer (2001) 688–698
47. Vanschoren, J., Blockeel, H.: Towards understanding learning behavior. In: Proceedings of the Annual Machine Learning Conference of Belgium and the Netherlands. (2006) 89–96
48. Blockeel, H., Vanschoren, J.: Experiment databases: Towards an improved experimental methodology in machine learning. In: Knowledge Discovery in Databases: PKDD 2007. Springer (2007) 6–17