10

Metalearning in Ensemble Methods

Summary. This chapter discusses some approaches that exploit metalearning methods in ensemble learning. It starts by presenting a set of issues, such as the ensemble method used, which affect the process of ensemble learning and the resulting ensemble. In this chapter we discuss various lines of research that were followed. Some approaches seek an ensemble-based solution for the whole dataset, others for individual instances. Regarding the first group, we focus on metalearning in the construction, pruning and integration phase. Modeling the interdependence of models plays an important part in this process. In the second group, the dynamic selection of models is carried out for each instance. A separate section is dedicated to hierarchical ensembles and some methods used in their design. As this area involves potentially very large configuration spaces, recourse to advanced methods, including metalearning, is advantageous. It can be exploited to define the competence regions of different models and the dependencies between them.

10.1 Introduction

In the previous chapter (Chapter 9) we introduced several methods that combine base-level classifiers into *ensemble models*, or simply *ensembles*. Some researchers refer to these methods as *ensemble learning*. Most ensembles focus on classification only, hence the term *multiple classifier systems* is also used (Mendes-Moreira et al., 2012; Cruz et al., 2018). Ensemble models have become very popular, as they often obtain superior performance to the best base-level model that can be identified. Typically, the prediction of an ensemble is generated by combining the predictions of multiple and diverse models.

A different perspective on this issue can be obtained by generalizing the scope of the *no free lunch* (NFL) theorem (Wolpert, 1996) from tasks to subtasks, as defined by subspaces of the data. In other words, as the NFL theorem suggests that the best algorithm for different problems may vary, we can assume that the best algorithm for different subspaces of the data could vary too. Thus, one can say that the goal of ensemble learning approaches is to reduce the probability of misclassification based on any single induced model, by increasing the system's area of expertise through combination of the (typically more localized) expertise of multiple models.

The process of learning and using an ensemble clearly involves two levels. Base-level models are obtained by analyzing the data from the ML tasks at hand. On the other hand, the combination of those models is a meta-level operation, even if, in some cases,

P. Brazdil et al., *Metalearning*, Cognitive Technologies,
https://doi.org/10.1007/978-3-030-67024-5_10

it may be rather simple (i.e., voting or averaging). For this reason, the term *metalearning* has sometimes been used to characterize ensemble learning methods. However, we note that the definition of metalearning used in this book is more restrictive (see the *Preface* to this book). Consequently, if we used our definition, not all ensemble systems would be characterized as metalearning systems.

Nevertheless, given the relevance of ensemble learning, a question arises as to what the role of metalearning/AutoML approaches is in this process. Different answers can be obtained depending on which perspective is taken. If we consider an ensemble as an algorithm, then the general aim of metalearning/AutoML methods is to recommend the most suitable algorithm (i.e., an ensemble) for the given task. However, as ensemble learning is a complex process involving multiple steps and models, metalearning can play an important role in this process too. So, from this perspective, metalearning can be used (1) to select a subset of models to make a prediction for a particular observation, (2) to estimate how accurate the prediction of a base-level model for a given observation is and use this information in the process of ensemble learning, or (3) to recommend the best possible method at each step of ensemble learning.

In this chapter we provide more details on different approaches that have been proposed on this topic. But first, let us review some basic characteristics that can be used to categorize different ensemble learning systems.

10.2 Basic Characteristics of Ensemble Systems

Do we want to exploit an existing portfolio of ensembles?

Many users often deal with multiple tasks, which may be rather similar. If they opt for a solution that involves ensembles, they may have a portfolio of various methods already available when a new task is encountered. The user then has a choice of two possible approaches. One involves a search through the existing portfolio for the best possible solution. More details on this are given in Section 10.3. The other involves ensemble learning with the objective of designing the best possible ensemble for the current task. This issue is discussed in Section 10.4.

Are predictions for the whole dataset or for each instance?

Some ensemble learning systems come up with a solution (i.e., an ensemble) for the whole dataset. Others take into account the characteristics of each instance and tailor the ensemble to this. This process is often referred to as *dynamic selection*, or *dynamic classifier selection* if the underlying task is classification. More details on this are given in Section 10.5.

Which ensemble method is used?

In Chapter 9 we described various types such as bagging, where the votes are weighed, boosting and stacking, among other approaches. Metalearning approaches for algorithm selection include many of these methods from the given pool of alternatives.

In approaches where metalearning is used as part of the ensemble learning process, the most popular method is stacking (Pinto et al., 2016; Narassiguin et al., 2017; Wistuba

et al., 2017; Khiari et al., 2019). However, other methods, such as bagging (e.g., Pinto et al. (2014)), have also been used.

Ensemble systems, such as ALMA (Houeland and Aamodt, 2018) and metalearning algorithm templates (Kordík et al., 2018), are more general, as they can involve many different ensemble methods and can also exploit metalearning.

Are the models generated with a single or different algorithms?

Heterogeneous ensembles are often preferred, as in principle, they promote more diverse components (e.g., classifiers) (Kuncheva and Whitaker, 2003).

Is the metadata from the current or past datasets?

We can distinguish two kinds of systems: some use just the metadata obtained on the current dataset, while others also explore the metaknowledge obtained on other datasets in previous experiments. Many of the metalearning approaches in ensemble learning learn from the current dataset only.

What is the base-level learning task?

The area of machine learning includes different tasks, such as, classification, multi-label classification and regression, among others. Some ensemble learning approaches are specific to tasks of one type, while others are more general and can deal with different task types (i.e., regression and classification).

10.3 Selection-Based Approaches for Ensemble Generation

Selection-based approaches rely on a good portfolio which can include good representatives of both base-level and ensemble-based methods. Methods exist that enable to explore vast configuration spaces and identify a useful subset of models for a given set of tasks (see Chapter 8).

Once a portfolio has been defined, it can be reused for new tasks. Various methods described in this book allow us to identify the best possible model (here an ensemble of models). One of these is the simple ranking approach discussed in Chapter 2. The approach described in Chapter 5 allows to identify the best possible algorithm (here a particular ensemble method) by also taking into account the existing metaknowledge associated with the current dataset and the past datasets (e.g., dataset characteristics, etc.). More details about this can be found in Chapter 4.

One disadvantage of this approach is that it requires the existence of a portfolio, which could be very large if we were to include all possible useful variants. This problem can be minimized by applying a portfolio reduction technique described in Chapter 8 (Section 8.5), which works like pruning. It eliminates sub-standard and redundant algorithms (here ensemble methods). The experiments of Cachada et al. (2017) described briefly in Chapter 7 (Section 7.4) have shown that the average ranking method AR* can outperform AutoWeka when the time budget is low. The portfolio used in these experiments included various algorithms, while about a half were different ensemble models.

This approach has the limitation that the configuration space is finite and hence may have a difficulty in keeping up with other methods that explore much larger configuration spaces.

10.4 Ensemble Learning (per Dataset)

Ensemble learning approaches build the best possible ensemble from given base-level models. This process typically involves various phases and can be iterative. More details on this are given in the next subsection.

Phases of ensemble learning

The phases of ensemble learning can be divided into three parts: generation, pruning, and integration (Mendes-Moreira et al., 2012) (Figure 10.1). More details on each are given in the following subsections.

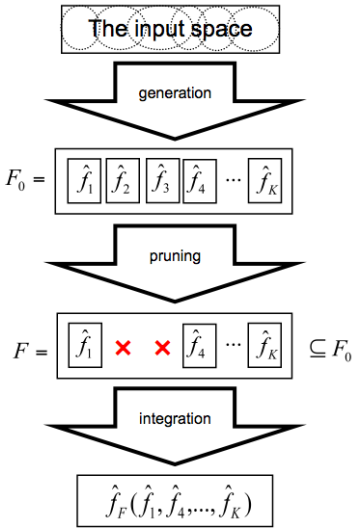


Fig. 10.1: The ensemble learning process (reproduced from Mendes-Moreira et al. (2012)

10.4.1 Metalearning in construction and pruning phases

Generation and pruning

Generation is concerned with obtaining a pool of diverse and sufficiently accurate models (Dietterich, 2000). This involves selecting the data (e.g., a dataset or some part), selecting an appropriate machine learning (ML) algorithm, and conducting training. The selection of the ML algorithm needs to obey various criteria. First, it needs to consider the given task. If, for instance, the aim is to obtain an ensemble for classification tasks, we need to consider a pool of classification algorithms. Pruning removes some of the models which are not considered useful (Mendes-Moreira et al., 2012).

Pinto et al. (2015) proposed a variant of bagging, in which a kind of pre-pruning is integrated with model generation. This is done by eliminating bootstrap samples that are not expected to generate useful models. Metalearning was used to predict whether a sample would generate a model that could improve the accuracy of the ensemble or not. This approach aims to reduce the computational cost of generating useless models.

Automatic Frankensteining of Wistuba et al. (2017) employs a multi-layer stacking of tuned models with weighting. In the generation phase, meta-level models are used to predict the execution time of runs of hyperparameter optimization. This permits to decide which sets of hyperparameters to test. The technique of *sequential model-based optimization* (SMBO), discussed in Chapter 6 (Section 6.4), is used in this process.

In general, metalearning can be used to identify which models should be generated. In order for the ensemble to be better than the models included in it, the models need to be diverse (Kuncheva and Whitaker, 2003). So we need measures of the diversity of two (or more) elements. Various measures have been proposed in the past. One is so-called *classifier output difference* (COD) (Peterson and Martinez, 2005). More details about this measure can be found in Chapter 8 (Section 8.5). Another possible measure is the Q-statistic (Kuncheva and Whitaker, 2003).

In some approaches, such as *boosting* (Chapter 9, Subsection 9.1.2), generation is done in an iterative fashion, similar to the process of building a decision tree, where a partially expanded tree is used as the basis for further extensions. The current model (a partially constructed ensemble) is used as the basis for various extensions, which are evaluated with respect to a given measure (or measures), and the best option is evaluated. So the metaknowledge acquired in this process is used to direct the search towards the most promising regions of the whole space. This has the advantage that the search is limited to only a small portion of the whole space.

A good solution to the problem of generating diverse models would possibly eliminate the need for the pruning phase, as was done in the approach proposed by Cruz et al. (2018).

Reusing the selection-based approaches for ensemble learning

An important decision concerning the design of ensemble models is which base-level learning algorithm to consider for the generation of models. The type of base-level task significantly reduces the choices available for selection of ensemble elements. If the base-level task is classification (regression), we need to consider solely classification (regression) algorithms as potential members of the ensemble. The method described in Section 10.3 can be adapted to generate candidates for an ensemble.

As we have mentioned, earlier selection-based approaches rely on a portfolio which should include both well-performing and diverse algorithms. More details about how this portfolio can be generated on the basis of past experiments are given in Chapter 8. Once a portfolio has been defined, it can be reused in new tasks. Various methods described in this book can be used to identify a subset of algorithms that is well suited for a given task. Typically, we would want to keep including algorithms in an iterative manner until some stopping condition has been satisfied. One possibility is to require that the ensemble includes, at most, n members. Another, probably better option is to require that each new member should contribute in some way to the ensemble. One possible criterion to use here is the estimate of *expected performance gain* discussed in Chapter 5 (Section 5.8).

We note that this approach can take into account also the existing dataset characteristics of the current task (dataset) and this way reuse meta-level information regarding

which base-level algorithms were useful on similar datasets used in the past (Pinto et al., 2014; Wistuba et al., 2017; Kordík et al., 2018; Houeland and Aamodt, 2018).

Choice of ML algorithm at the meta-level

When used in the ensemble generation/pruning phases, metalearning usually involves a standard machine learning task, such as classification or regression. When the process of generating models is sequential, then label ranking could also be used, although, to the best of our knowledge, this has never been done.

Since the metalearning task is, most of the time, a standard one, this means that off-the-shelf algorithms are used, such as decision trees, support vector machines (SVMs), random forest, and lazy learning.

Modeling interdependence of models

As explained earlier, the contribution of a single model to an ensemble depends on the remaining models. This represents an opportunity for metalearning, as it can be used to characterize the relation between models and/or data. It is possible to take into account characteristics of the current task (dataset) and this way reuse meta-level information regarding relations between models. In Pinto et al. (2014), a variant of bagging is proposed where this is done in an indirect way. Rather than comparing two models, they compare each sample with the original dataset. Metalearning is then used to identify situations guaranteeing that learning a new model from a sample is worthwhile.

Metafeatures

Most metalearning approaches in ensemble learning use many of the metafeatures discussed in Chapter 4. The common ones are the simple, statistical, and information-theoretic metafeatures (Pinto et al., 2014; Wistuba et al., 2017), as well as landmarkers (Pinto et al., 2014).

As mentioned above, as ensembles involve multiple models, metafeatures that quantify relations between pairs of models should be considered, such as COD (Peterson and Martinez, 2005) and the Q-statistic (Kuncheva and Whitaker, 2003), discussed earlier in Subsection 10.4.1. They were used on landmarking models obtained with different bootstrap samples to estimate their redundancy (Pinto et al., 2014).

One approach to generate diversity of the models is by altering the data used to learn them (e.g., resampling the original dataset). In these cases, and for metalearning purposes, the diversity between those models can be estimated by quantifying the differences between those samples. This can be done by measuring the difference between data distributions using, for instance, the Kullback–Leibler divergence (Cruz et al., 2018), or between metafeature values (Pinto et al., 2014). The distance between the corresponding sample and the original dataset has been used to select the bootstrap samples that generate useful models (Pinto et al., 2014).

An alternative approach to algorithm/model selection is based on the prediction of performance or execution time (Wistuba et al., 2017). In these approaches, the configurations of the learning process represented by the meta-examples can also be used as metafeatures. So these can include the values of specific hyperparameter settings used in the experiment (Wistuba et al., 2017).

Strategy employed in Auto-sklearn

The strategy employed in Auto-sklearn involves two phases. In the first one, the system seeks not just one, but a set of good base-level solutions. In the second phase some of these solutions are selected to form an ensemble. More details on this approach can be found in Feurer et al. (2015a) and Feurer et al. (2019).

10.4.2 Metalearning in the integration phase

Integration

Given the set of models that result from the generation and pruning phases of the ensemble learning process and a new observation, a prediction is obtained by combining the individual prediction of each of the models. The question is how to combine the predictions of the various models into a single one. Existing approaches range from simple approaches, such as voting, to complex ones that adjust the combination method to the specific observation at hand. The latter is discussed in Section 10.5.

As pointed out earlier, the contribution of a single model to the ensemble depends on the other models in the ensemble (Pinto et al., 2016). Therefore, every model that is integrated into the ensemble, or eliminated from it, affects not only the performance of the ensemble directly but also the contribution of all the other models in the ensemble. Besides, it also affects the contribution of other candidate models that could be considered in the future. More details on the so-called *marginal contribution* of algorithms can be found in Chapter 8 (Section 8.4).

The marginal contribution of individual elements is related to the so-called *competence region* of each model. In other words, it is necessary to determine subspaces of the given set of tasks and the associated datasets in which the model has good performance (e.g., a high mean and a low variance). This subspace is usually referred to as the *competence region* and is usually associated with a particular algorithm or the trained model.

The competence region of a particular algorithm may include, say, classification of a certain type of datasets (e.g., image datasets or datasets with correlated features), or just a certain type of instances of a given dataset. The second option is explored in the approaches that involve so-called *dynamic selection of models*, discussed in Section 10.5. The aim of these approaches is to identify a set of competent models for each example.

Metalearning method

Metalearning can also play an important role in the integration step. In fact, in its simplest form, it is clearly a metalearning problem: given an observation, which subset of the generated models should be used to make the prediction? We also discuss the metalearning method. The metafeatures, which are particularly challenging in this case, are discussed further on.

10.5 Dynamic Selection of Models (per Instance)

Dynamic selection approaches generate a large number of models and, given a new instance, combine them (e.g., assign weights or select a subset). The term *dynamic classifier selection* (DCS) is used when the approach is applied to classification settings.

Reusing the selection-based approaches for ensemble learning

As explained earlier, DCS can be addressed as a classification/recommendation problem: given an observation, the meta-decision tree selects the most appropriate model to make a prediction. System MetaBags follows this approach, using *meta-decision trees*. Meta-decision trees were proposed by Todorovski and Džeroski (2000) and are described in Chapter 9 (Section 9.5). Curiously, MetaBags uses an ensemble method at the meta-level as well, by bagging the meta-decision trees.

Layer structure in system ALMA

System ALMA (Houeland and Aamodt, 2018) is an abstract ML framework consisting of a hierarchical structure of components of learning systems and includes layers representing algorithms, , and meta-algorithms for dynamic classifier selection. This approach can be used in conjunction with voting and weighting. It involves a lazy metalearning approach for algorithm selection. The authors report that the system can use metaknowledge from both the current dataset and other datasets. The experiments presented, however, focused on metalearning from the current dataset.

As DCS is concerned with a choice of models for each example, the extension to streaming scenarios is easier than for other systems that exploit batch data.

Modeling interdependence of models

As explained earlier, the contribution of a single model to an ensemble depends on the remaining models. This is true for the combination of models to make the prediction for a single instance, as well as for the construction and pruning of models.

DCS can also be addressed as a multi-target prediction problem: given an observation and a set of models, the question is which ones to use. The term *multi-target prediction* is a broad name for ML tasks that take into account the interdependence between multiple decisions (Waegeman et al., 2019). *Multi-label classification* (MLC) is a multi-target prediction task in which the goal is to predict which of the labels from a given set should be assigned to an observation (Read et al., 2019).

Therefore, dynamic classifier selection (DCS) can be addressed as a MLC problem, where the labels are the models (Pinto et al., 2016; Narassiguin et al., 2017). That is, given the set of N models $H = \{h_1, \ldots, h_N\}$ resulting from the generation and pruning phases, the goal is to select the right subset of those models $H_i \subseteq H$ to make predictions for observation i.

The systems CHADE (Pinto et al., 2016) and PCC-DES (Narassiguin et al., 2017) use a multi-label classification-based metalearning approach to address the problem of dependency between the models. A stacking-inspired approach is used to predict whether a model in the ensemble will accurately predict a new example or not. The algorithms used were classifier chains (Pinto et al., 2016) and probabilistic classifier chains (Narassiguin et al., 2017).

10.5.1 Metafeatures

In the case of metalearning for dynamic classifier selection, where the target is a single example, computing traditional metafeatures is a challenge. The reason for this is that

traditional metafeatures characterize sets of examples, so they cannot be applied directly to a single observation.

One approach is to compute statistics in the neighborhood of the target example (Khiari et al., 2019). Another possibility is to rely on a prior clustering of examples. Classifiers are not selected for an example, but rather for a group of examples that appear to be similar. These approaches enable the use of standard data characterization measures, such as the ones discussed in Chapter 4. Britto et al. (2014) also used a *problem complexity* metafeature.

Model-based metafeatures are not as common. This is not surprising, as this type of metafeatures is also not very common in traditional metalearning scenarios. An exception is in MetaBags (Khiari et al., 2019), which includes a new type of model-based metafeatures, referred to as *local landmarkers*. That is: given an example, they characterize the leaf of a tree where it falls, namely its *depth* and *the number of examples in that leaf*. We note that, despite the name, these are actually model-based metafeatures.

Using base-level features and predictions as metafeatures

The challenge of computing metafeatures on individual observations also creates an opportunity. Since each meta-example represents a single example, the original attributes can also be used as metafeatures (Pinto et al., 2016; Khiari et al., 2019). Furthermore, we note that in stacking approaches the predictions of models are also used as metafeatures (Narassiguin et al., 2017; Khiari et al., 2019).

But these metafeatures can be generalized as a different type of landmarkers that characterize the behaviour of models for a particular example. For instance, a stacking-inspired set of metafeatures has been used by Pinto et al. (2016), consisting of predictions of whether each candidate model would make correct predictions or not.

10.6 Generation of Hierarchical Ensembles

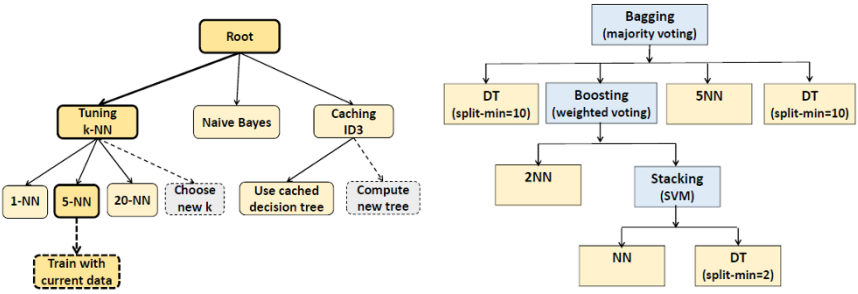
Metalearning methods have also been integrated into two general machine learning frameworks: *metalearning algorithm templates* (MAT) (Kordík et al., 2018) and ALMA (Houeland and Aamodt, 2018). Both approaches represent ensembles with a hierarchical structure (Subsection 10.6.1), but employ metalearning in different ways: MAT is a search-based ensemble construction method that uses metalearning to initialize the search with promising solutions (Section 10.6.2). On the other hand, in ALMA, metalearning is at the core of the ensemble construction method (Section 10.6.3).

10.6.1 Hierarchical Ensembles

Hierarchical ensembles are, as the name suggests, a hierarchical structure in the form of a tree. Examples are shown in Figure 10.2. The internal nodes can include ensembles, which in turn can include either base-level algorithms or other ensembles as members. The leaf nodes include just base-level algorithms.

10.6.2 Evolving hierarchical ensembles with evolutionary computing

Kordík et al. (2018) have developed a system that aims to generate the best possible hierarchical ensemble for the target dataset with recourse to *evolutionary computation*



(a) (reproduced from Kordík et al. (2018)) (b) (reproduced from Houeland and Aamodt (2018))

Fig. 10.2: Examples of hierarchical combinations of algorithms

(EC). This system can include different types of ensembles, including bagging, boosting, cascading, and arbitrating.

The process starts with a set of simple solutions, which are evolved into more complex ones. This process is controlled both by fitness and *templates* that embody the authors' knowledge of how a given structure could be extended.

The templates can be represented in a form similar to ontologies or context-free grammar (CFG) rules discussed in Chapter 7 (Section 7.2). As pointed out in that chapter, they embody a certain declarative and procedural bias that constrains the search. The EC algorithm searches for the optimal hierarchical ensemble for the given task.

The system uses the most promising workflows identified on the past problems to initialize the search. These workflows can be regarded as metaknowledge acquired on past tasks. This strategy can be related to the process of initializing the search for the best parameter settings described by Feurer et al. (2015b). Chapter 6 (Section 6.8) discusses this in detail.

As this process can be rather slow when the dataset is large, the authors develop their solutions on smaller data samples and then apply them to full data.

The system has been applied to several concrete tasks. The authors have shown how one particular evolved template (simple ensemble of fast sigmoidal regression models) outperformed the state-of-the-art approaches on a rather large Airline dataset.

10.6.3 Metalearning in hierarchical ensemble methods

In ALMA (Houeland and Aamodt, 2018), metalearning is at the core of the ensemble generation process. It organizes this process into three layers, representing models, algorithms, and meta-algorithms, respectively. In the metalearning layer, a lazy learning algorithm is used for algorithm selection.

The selection-based approach described in Section 10.3 could be extended to generate hierarchical ensembles using a phased approach that would proceed in layers. In the first phase, it would generate a larger set of potentially useful ensembles. Caution could be taken to avoid branches in the search space that are not likely to lead to a useful outcome (e.g., not allow bagging ensembles of base-level algorithms with low variance, etc.). A smaller subset of this set would be identified using the reduction technique described in Section 10.3. This set would then be added to the existing portfolio, and the

process would then be repeated. It would be interesting to see how this approach would work in practice.

10.7 Conclusions and Future Research

Ensemble learning consists of learning systems that combine multiple diverse models to obtain more accurate predictions. This approach is based on the idea that different models specialize in different subspaces of data of the given task. In this chapter we have discussed various lines of research that were followed. As we have shown, some approaches seek an ensemble-based solution for the whole dataset, others for individual instances. We have used this criterion and discussed each group of approaches in a separate section. Hierarchical ensembles were also separated out, although the methods used to construct them are not so different from the simpler counterparts, although, of course, the configuration space is much larger.

So our aim was to clarify the role of metalearning in ensemble learning and how it can be integrated into the whole process. As this area involves potentially very large configuration spaces, recourse to advanced methods, including metalearning, is really a *must*. Metalearning can be exploited to define the competence regions of different models and the dependencies between them. The challenge is how to find a good way of doing this, so that the search for new and useful ensemble-based solutions would become easier.

References

Britto, A. S., Sabourin, R., and Oliveira, L. E. (2014). Dynamic selection of classifiers—A comprehensive review. *Pattern Recognition*, 47(11):3665–3680.

Cachada, M., Abdulrahman, S., and Brazdil, P. (2017). Combining feature and algorithm hyperparameter selection using some metalearning methods. In *Proc. of Workshop AutoML 2017, CEUR Proceedings Vol-1998*, pages 75–87.

Cruz, R. M., Sabourin, R., and Cavalcanti, G. D. (2018). Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, 41:195–216.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Multiple Classifier Systems. MCS 2000*, volume 1857 of *LNCS*. Springer.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. (2015a). Efficient and robust automated machine learning. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, NIPS'15, pages 2962–2970. Curran Associates, Inc.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J. T., Blum, M., and Hutter, F. (2019). Auto-sklearn: Efficient and robust automated machine learning. In Hutter, F., Kotthoff, L., and Vanschoren, J., editors, *Automated Machine Learning: Methods, Systems, Challenges*, pages 113–134. Springer.

Feurer, M., Springenberg, J., and Hutter, F. (2015b). Initializing Bayesian hyperparameter optimization via meta-learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 1128–1135.

Houeland, T. G. and Aamodt, A. (2018). A learning system based on lazy metareasoning. *Progress in Artificial Intelligence*, 7(2):129–146.

Khiari, J., Moreira-Matias, L., Shaker, A., Ženko, B., and Džeroski, S. (2019). MetaBags: Bagged meta-decision trees for regression. In *Procedings of ECML/PKDD 2018*, pages 637–652. Springer.

Kordík, P., Černý, J., and Frýda, T. (2018). Discovering predictive ensembles for transfer learning and meta-learning. *Machine Learning*, 107(1):177–207.

Kuncheva, L. I. and Whitaker, C. J. (2003). Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. *Machine Learning*, 51(2):181–207.

Mendes-Moreira, J., Soares, C., Jorge, A. M., and Sousa, J. F. D. (2012). Ensemble approaches for regression. *ACM Computing Surveys*, 45(1):1–40.

Narassiguin, A., Elghazel, H., and Aussem, A. (2017). Dynamic Ensemble Selection with Probabilistic Classifier Chains. In *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2017, Lecture Notes in Computer Science*, volume 10534 LNAI, pages 169–186. Springer, Cham.

Peterson, A. H. and Martinez, T. (2005). Estimating the potential for combining learning models. In *Proc. of the ICML Workshop on Meta-Learning*, pages 68–75.

Pinto, F., Soares, C., and Mendes-Moreira, J. (2014). An empirical methodology to analyze the behavior of bagging. In *Advanced Data Mining and Applications. ADMA 2014. Lecture Notes in Computer Science*, volume 8933. Springer, Cham.

Pinto, F., Soares, C., and Mendes-Moreira, J. (2015). Pruning bagging ensembles with metalearning. In *International Conference on Advanced Data Mining and Applications, Lecture Notes in Computer Science*, volume 9132, pages 64–75. Springer, Cham.

Pinto, F., Soares, C., and Mendes-Moreira, J. (2016). CHADE: Metalearning with classifier chains for dynamic combination of classifiers. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2016, Lecture Notes in Computer Science*, volume 9851 LNAI, pages 410–425. Springer, Cham.

Read, J., Pfahringer, B., Holmes, G., and Frank, E. (2019). Classifier chains: A review and perspectives. *arXiv preprint arXiv:1912.13405*.

Todorovski, L. and Džeroski, S. (2000). Combining multiple models with meta decision trees. In Zighed, D. A., Komorowski, J., and Zytkow, J., editors, *Proc. of the Fourth European Conf. on Principles and Practice of Knowledge Discovery in Databases*, pages 255–264. Springer-Verlag.

Waegeman, W., Dembczyński, K., and Hüllermeier, E. (2019). Multi-target prediction: a unifying view on problems and methods. *Data Mining and Knowledge Discovery*, 33(2):293–324.

Wistuba, M., Schilling, N., and Schmidt-Thieme, L. (2017). Automatic Frankensteining: Creating complex ensembles autonomously. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 741–749. Society for Industrial and Applied Mathematics, Philadelphia, PA.

Wolpert, D. (1996). The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8:1341–1390.