

Ensembles for Predicting Structured Outputs

Dragi Kocev · Celine Vens · Jan Struyf ·
Sašo Džeroski

Received: date / Accepted: date

Abstract While ensembles have been used for structured output learning, the literature lacks an extensive study of different strategies to construct ensembles in this context. In this work, we fill this gap by presenting a thorough empirical comparison of ensembles that predict the complete output structure at once, versus a combination of ensembles that each predicts a single component of the structure. We present results in two structured output learning tasks, using predictive clustering trees as base learners. The main results are that the difference in predictive performance is not significantly different for both approaches. However, in terms of total model size and induction times, ensembles that exploit the output structure are significantly more efficient.

Keywords ensembles · predictive clustering trees · structured outputs

Dragi Kocev

Department of Knowledge technologies, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia

Tel.: +386-1-4773639

Fax: +386-1-4773315

E-mail: Dragi.Kocev@ijs.si

Celine Vens

Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3001 Leuven, Belgium

E-mail: Celine.Vens@cs.kuleuven.be

Jan Struyf

Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3001 Leuven, Belgium

E-mail: Jan.Struyf@cs.kuleuven.be

Sašo Džeroski

Department of Knowledge technologies, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia

E-mail: Saso.Dzeroski@ijs.si

1 Introduction

Supervised learning is one of the most widely researched and investigated areas of machine learning. The goal in supervised learning is to learn, from a set of examples with known class, a function that outputs a prediction for the class of a previously unseen example. If the examples belong to two classes (e.g., the example has some property or not) the task is called binary classification. The task where the examples can belong to a single class from a given set of m classes ($m \geq 3$) is known as multi-class classification. The case where the output is a real value, is called regression.

However, in many real life problems of predictive modelling, the output (i.e., the target property) is structured, meaning that there can be dependencies between classes (e.g., classes are organized into a tree-shaped hierarchy or a directed acyclic graph) or some internal relations between the classes (e.g., sequences). These types of problems occur in domains such as life sciences (predicting the gene function, selecting the most important genes for a given disease, detecting toxic molecules, etc), ecology (analysis of remotely sensed data, habitat modelling), multimedia (annotation and retrieval of images and videos) and the semantic web (categorization and analysis of text and web). Having in mind the needs of the application domains and the increasing quantities of structured data, Yang and Wu (2006) and Krieger et al (2007) listed the task of “mining complex knowledge from complex data” as one of the most challenging problems in machine learning.

A variety of methods, specialized in predicting a given type of structured output (e.g., a hierarchy of classes (Silla and Freitas, 2011)), have been proposed (Bakır et al, 2007). These methods can be categorized into two groups of methods for solving the problem of predicting structured outputs (Bakır et al, 2007; Silla and Freitas, 2011): (1) methods that predict component(s) of the output and then combine the individual models to get the global model and (2) methods that predict the complete structure as a whole (also known as ‘big-bang’ approach). The latter group of methods has several advantages over the former. First, they exploit and use the dependencies that exist between the components of the structured output in the model learning phase, and result in better predictive performance. Next, they are more efficient: it can easily happen that the number of components in the output is very large (e.g., hierarchies in functional genomics can have several thousands components), in which case executing a basic method for each component is not feasible. Furthermore, they produce models that are typically smaller than the sum of the sizes of the models for the components.

The predictive models that we consider in this article are predictive clustering trees (PCTs). PCTs belong to the group of global methods. PCTs offer a unifying approach for dealing with different types of structured outputs and construct the predictive models very efficiently. They are able to make predictions for several types of structured outputs: tuples of continuous/discrete variables, hierarchies of classes, and time series. More details about the PCT framework can be found in (Blockeel et al, 1998; Struyf and Džeroski, 2006; Kocev et al, 2007; Vens et al, 2008; Slavkov et al, 2010).

To further increase the predictive performance of a single predictive model, one can construct an ensemble of predictive models. An ensemble is a set of (base) predictive models. For basic classification and regression tasks, it is widely accepted that an ensemble of predictive models lifts the predictive performance of

its base predictive models (Seni and Elder, 2010). However, for the task of predicting structured outputs, this has never been thoroughly investigated. Moreover, in the case where the base predictive models are decision trees, Bauer and Kohavi (1999) conclude that the ensemble’s increase in performance is stronger if the trees are unpruned, i.e., allowed to overfit. On the other hand, Blockeel et al (2006) state that PCTs for structured outputs show less overfitting than the trees for classification of a single target variable. Having in mind these two conflicting influences, it is not obvious whether an ensemble of predictive clustering trees can significantly increase the predictive performance over that of a single predictive clustering tree.

Furthermore, in an ensemble learning setting, it is not clear if the predictive performance of an ensemble of global predictive models will be better or worse than the predictive performance of a combination of ensembles of local predictive models. Generally, an ensemble is known to perform better than its base learner if the base learner is accurate and diverse (Hansen and Salamon, 1990). While the superior predictive performance of global models have been shown before (Vens et al, 2008), less is known about their diversity or instability (i.e. whether they produce different errors with small changes to the training data). It is expected that a PCT for predicting structured outputs, especially in the case of hierarchical classification, is less unstable than a PCT for predicting components of the output. It is not also clear which approach will be more efficient, both in terms of running time and size of the predictive models.

In this article, we investigate the aforementioned questions. We use bagging and random forests as ensemble learning methods, since they are the two most widely used ensemble learning methods in the context of decision trees. We consider two structured output machine learning tasks: predicting multiple target variables and hierarchical multi-label classification. We perform an extensive empirical evaluation of the proposed methods over a variety of benchmark datasets.

In (Kocev et al, 2007), we have conducted an initial comparison of different ensemble schemes using predictive clustering trees in the context of predicting multiple targets. This work extends (Kocev et al, 2007) in the following directions: (1) we also consider hierarchical classification, (2) we use more datasets, from various domains, (3) we present a more detailed discussion of the results, including saturation curves and the use of Friedman tests combined with a Nemenyi post-hoc test to compare different ensemble schemes instead of performing pairwise comparisons.

The remainder of this paper is organized as follows. In Section 2, the considered machine learning tasks are formally defined and the related work is presented. Section 3 explains first the predictive clustering trees framework and the extensions for predicting multiple targets and hierarchical multi-label classification and then the ensemble methods and their extension for predicting structured outputs. The design of the experiments, the descriptions of the datasets, the evaluation measures and the parameter instantiations for the algorithms are given in Section 4. Section 5 presents and discusses the obtained results. Finally, the conclusions are stated in Section 6.

2 Background

The work presented in this article concerns the learning of ensembles for predicting structured outputs. In this section, we first define the machine learning tasks that we consider: the tasks of predicting multiple targets and hierarchical multi-label classification. We then give a brief overview of methods for predicting structured outputs.

2.1 Machine learning tasks

First, we formally describe the machine learning tasks that we consider here. We follow the suggestions by Džeroski (2007), where predictive modelling is defined for arbitrary types of input and output data. In particular, we describe the tasks of predicting multiple targets and hierarchical multi-label classification.

2.1.1 Predicting multiple targets task

The task of predicting multiple targets was previously referred to as multi-objective prediction (Struyf and Džeroski, 2006; Kocev et al, 2007; Demšar et al, 2006). However, the term ‘multi-objective’ is already established in the area of optimization. We will thus use the term ‘predicting multiple targets’ or multi-target prediction (resp. multi-target classification and regression). We define the task of predicting multiple targets as follows.

Given:

- A description space X that consists of tuples of values of primitive data types (boolean, discrete or continuous), i.e., $\forall X_i \in X, X_i = (x_{i_1}, x_{i_2}, \dots, x_{i_D})$, where D is the size of the tuple (or number of descriptive variables),
- a target space Y which consists of a tuple of several variables that can be either continuous or discrete, i.e., $\forall Y_i \in Y, Y_i = (y_{i_1}, y_{i_2}, \dots, y_{i_T})$, where T is the size of the tuple (i.e., number of target variables),
- a set of examples E , where each example is a pair of tuples from the description and target space, respectively, i.e., $E = \{(X_i, Y_i) | X_i \in X, Y_i \in Y, 1 \leq i \leq N\}$ and N is the number of examples of E ($N = |E|$), and
- a quality criterion q , which rewards models with high predictive accuracy and low complexity.

Find: a function $f : X \rightarrow Y$ such that f maximizes q .

Here, the function f is represented with decision trees, i.e., predictive clustering trees or ensembles thereof. If the tuples from Y (the target space) consist of continuous/numeric variables then the task at hand is multiple targets regression. Likewise, if the tuples from Y consist of discrete/nominal variables then the task is called multiple targets classification.

2.1.2 Hierarchical classification task

Classification is defined as the task of learning a model using a set of previously classified instances and applying the obtained model to a set of previously unseen

examples (Breiman et al, 1984; Langley, 1996). The unseen examples are classified into a single class from a set of possible classes.

Hierarchical classification differs from the traditional classification in the following: the classes are organized in a hierarchy. An example that belongs to a given class automatically belongs to all its super-classes (this is known as the *hierarchy constraint*). Furthermore, if an example can belong simultaneously to multiple classes that can follow multiple paths from the root class, then the task is called hierarchical multi-label classification (HMC) (Silla and Freitas, 2011; Vens et al, 2008). This is the setting we use in this article.

We formally define the task of hierarchical multi-label classification as follows:

Given:

- A description space X that consists of tuples of values of primitive data types (boolean, discrete or continuous), i.e., $\forall X_i \in X, X_i = (x_{i_1}, x_{i_2}, \dots, x_{i_D})$, where D is the size of the tuple (or number of descriptive variables),
- a target space S , defined with a class hierarchy (C, \leq_h) , where C is a set of classes and \leq_h is a partial order (e.g., structured as a rooted tree) representing the superclass relationship ($\forall c_1, c_2 \in C : c_1 \leq_h c_2$ if and only if c_1 is a superclass of c_2),
- a set E , where each example is a pair of a tuple and a set from the descriptive and target space respectively, and each set satisfies the hierarchy constraint, i.e., $E = \{(X_i, S_i) | X_i \in X, S_i \subseteq C, c \in S_i \Rightarrow \forall c' \leq_h c : c' \in S_i, 1 \leq i \leq N\}$ and N is the number of examples of E ($N = |E|$), and
- a quality criterion q , which rewards models with high predictive accuracy and low complexity.

Find: a function $f : X \rightarrow 2^C$ (where 2^C is the power set of C) such that f maximizes q and $c \in f(x) \Rightarrow \forall c' \leq_h c : c' \in f(x)$, i.e., predictions made by the model satisfy the *hierarchy constraint*. In our case, the function f is represented with decision trees, i.e., predictive clustering trees or ensembles thereof.

2.2 Related work

The task of predicting structured outputs is gaining more and more attention within the machine learning research community (Bakır et al, 2007; Silla and Freitas, 2011). The community has proposed a number of different methods for addressing this task. However, they are typically “computationally demanding and ill-suited for dealing with large datasets” (Bakır et al, 2007). In this article, we propose a global method for predicting structured outputs that has good predictive performance and is very efficient. We use the predictive clustering framework both for predicting multiple targets and for hierarchical multi-label classification. In the literature, there are mostly methods that solve one of these two tasks. In the remainder of this section, we first present the methods that predict multiple targets and then the methods for hierarchical multi-label classification.

2.2.1 Methods for multi-target prediction

The task of predicting multiple targets is connected with the *multi-task learning* (Caruana, 1997) and *learning to learn* (Thrun and Pratt, 1998) paradigms. These

paradigms include the task of predicting a variable (continuous or discrete) using multiple input spaces (i.e., biological data for a disease obtained using different technologies); predicting multiple variables from multiple input spaces, and predicting multiple variables from a single input space. Here, we consider the last task. The methods we propose can handle two types of outputs: multiple discrete variables (multi-target classification) and multiple continuous variables (multi-target regression), while most of the approaches from the literature can handle only one type of output.

There is extensive empirical work showing an increase in predictive performance when multiple tasks are learned simultaneously as compared to learning each task separately (for example, see (Baxter, 2000; Evgeniou et al, 2005; Caponnetto et al, 2008; Ben-David and Borbely, 2008) and the references therein).

The key for the success of multi-task learning is the *relatedness* between the multiple tasks. The notion of relatedness is differently perceived and defined by different researchers. For example, Ando et al (2005) assume that all related tasks have some common hidden structure. Greene (2007) models the relatedness under the assumption of correlation between the noise for the different regression estimates. Baxter (2000) views the similarity through a model selection criterion, i.e., learning multiple tasks simultaneously is beneficial if the tasks share a common optimal hypothesis space. To this end, a generalized VC-dimension is used for bounding the average empirical error of a set of predictive models over a class of tasks.

We present and categorize the related work in four groups: statistics, statistical learning theory, Bayesian theory and kernel learning. In statistics, Brown and Zidek (1980) extend the standard ridge regression to multivariate ridge regression, while Breiman and Friedman (1997) propose the CURDS&WHEY method, where the relations between the task are modeled in a post-processing phase. In statistical learning theory, for handling multiple tasks, an extension of the VC-dimension and the basic generalization bounds for single task learning are proposed by Baxter (2000) and Ben-David and Borbely (2008).

Most of the work in multi-task learning is done using Bayesian theory (Thrun and Pratt, 1998; Bakker and Heskes, 2003; Wilson et al, 2007). In this case, simultaneously with the parameters of the models for each of the tasks, a probabilistic model that captures the relations between the various tasks is being calculated. Most of these approaches use hierarchical Bayesian models.

Finally, there are many approaches for multi-task learning using kernel methods. For example, Evgeniou et al (2005) extend the kernel methods to the case of multi-task learning by using a particular type of kernel (multi-task kernel). The regularized multi-task learning then becomes equivalent to single-task learning when such a kernel is used. They show experimentally that the support vector machines (SVMs) with multi-task kernels have significantly better performance than the ones with single-task kernels. For more details on kernel methods and SVMs for multi-task learning, we refer the reader to (Caponnetto et al, 2008; Argyriou et al, 2008; Micchelli and Pontil, 2004; Cai and Cherkassky, 2009) and the references therein.

2.2.2 Methods for hierarchical multi-label classification

A number of approaches have been proposed for the task of hierarchical multi-label classification (Bakır et al, 2007). Silla and Freitas (2011) survey and categorize the HMC methods based on some characteristics and the application domains. The characteristics of the methods they consider as most important are: prediction of single or multiple paths from the hierarchy, the depth of the predicted class, the type of the taxonomy that can be handled (tree or directed acyclic graph) and whether the method is local (constructs a model for each part of the taxonomy) or global (constructs a model for the whole taxonomy). The most prominent application domains for these methods are functional genomics (biology), image classification, text categorization, and genre classification.

Here, we present and group some existing methods based on the learning technique they use. We group the methods as follows: network based methods, kernel based methods and decision tree based methods.

Network based methods. The network based approaches predict functions of unannotated genes based on known functions of genes that are nearby in a functional association network or protein-protein interaction network (Chen and Xu, 2004). Since the network based approaches are based on label propagation, a number of approaches were proposed to combine predictions of functional networks with those of a predictive model. Tian et al (2008), for instance, use logistic regression to combine predictions made by a functional association network with predictions from a random forest.

Kernel based methods. Obozinski et al (2008) present a two-step approach in which SVMs are first learned independently for each class separately (allowing violations of the hierarchy constraint) and are then reconciliated to enforce the hierarchy constraint. Similarly, Barutcuoglu et al (2006) use un-thresholded SVMs learned for each class separately and then combine the SVMs by using a Bayesian network so that the predictions are consistent with the hierarchical relationships. Guan et al (2008) extend the method by Barutcuoglu et al (2006) to an ensemble framework. Valentini and Re (2009) also propose a hierarchical ensemble method that uses probabilistic SVMs as base learners. The method combines the predictions by propagating the weighted true path rule both top-down and bottom-up through the hierarchy, which ensures consistency with the hierarchy constraint.

Rousu et al (2006) present a more direct method that does not require a second step to make sure that the hierarchy constraint is satisfied. Their approach is based on a large margin method for structured output prediction which defines a joint feature map over the input and the output space. Next, it applies SVM based techniques to learn the weights of a discriminant function (defined as the dot product of the weights and the joint feature map). Rousu et al (2006) propose a suitable joint feature map and an efficient way for computing the argmax of the discriminant function (which is the prediction for a new instance). Furthermore, Gärtner and Vembu (2009) propose to use counting of super-structures from the output to efficiently calculate (in polynomial time) the argmax of the discriminant function.

Decision tree based methods. Clare (2003) adapts the well-known decision tree algorithm C4.5 (Quinlan, 1993) to cope with the issues introduced by the HMC task. This version of C4.5 (called C4.5H) uses the sum of entropies of the class variables to select the best split. C4.5H predicts classes on several levels of the

hierarchy, assigning a larger cost to misclassifications higher up in the hierarchy. The resulting tree is then transformed into a set of rules, and the best rules are selected, based on a significance test on a validation set.

Geurts et al (2006) present a decision tree based approach related to predictive clustering trees. They start from a different definition of variance and then kernelize this variance function. The result is a decision tree induction system that can be applied to structured output prediction using a method similar to the large margin methods mentioned above. Therefore, this system could also be used for HMC after defining a suitable kernel. To this end, an approach similar to that of Rousu et al (2006) could be used.

Blockeel et al (2002, 2006) proposed the idea of using predictive clustering trees (Blockeel et al, 1998) for HMC tasks (PCTs for HMC). This work (Blockeel et al, 2006) presents the first thorough empirical comparison between an HMC decision tree method in the context of tree shaped class hierarchies. Vens et al (2008) extend the algorithm towards hierarchies structured as DAGs and show that learning one decision tree for predicting all classes simultaneously outperforms learning one tree per class (even if those trees are built by taking into account the hierarchy, i.e., hierarchical single-label classification - HSC). Schietgat et al (2010) introduced ensembles (i.e., bagging) of PCTs for HMC in the context of functional genomics. They report superior predictive performance of the ensemble over a single HMC tree. However, only ensembles with 50 base PCTs were considered and no comparison of different ensemble schemes was performed.

3 Ensembles for predicting structured outputs

In this section, we present the ensemble methods for predicting structured outputs. We begin by presenting the predictive clustering trees and their instantiations for predicting multiple continuous variables, predicting multiple discrete variables, and hierarchical multi-label classification. Next, we describe how ensemble learning methods can be adapted to use predictive clustering trees as base predictive models. Finally, we describe an approach to the prediction of structured outputs that use local predictive models.

3.1 PCTs for structured outputs

The Predictive Clustering Trees (PCTs) framework sees a decision tree as a hierarchy of clusters: the top-node corresponds to one cluster containing all data, which is recursively partitioned into smaller clusters while moving down the tree. The PCT framework is implemented in the CLUS system (Blockeel and Struyf, 2002), which is available for download at <http://www.cs.kuleuven.be/~dtai/clus>.

CLUS takes as input a set of examples $E = \{(x_i, y_i) | i = 1, \dots, N\}$, where each x_i is a vector of attribute values and y_i are values of a structured (output) datatype T_Y . In this article, we consider three different classes of datatypes T_Y : tuples of discrete values, tuples of real values, and hierarchies. For each type T_Y , CLUS needs two functions to be defined. The prototype function returns a representative structured value given a set of such values. The variance function describes how

homogeneous a set of structured values is: It is typically based on a distance function on the space of structured values.

PCTs can be induced with a standard *top-down induction of decision trees* (TDIDT) algorithm (Breiman et al, 1984). The algorithm is presented in Table 1. It takes as input a set of examples (E) and outputs a tree. The heuristic (h) that is used for selecting the tests (t) is the reduction in variance caused by partitioning (\mathcal{P}) the instances (see line 4 of BestTest procedure in Table 1). By maximizing the variance reduction the cluster homogeneity is maximized and it improves the predictive performance.

Table 1 The top-down induction algorithm for PCTs.

procedure PCT(E) returns tree	procedure BestTest(E)
1: $(t^*, h^*, \mathcal{P}^*) = \text{BestTest}(E)$	1: $(t^*, h^*, \mathcal{P}^*) = (\text{none}, 0, \emptyset)$
2: if $t^* \neq \text{none}$ then	2: for each possible test t do
3: for each $E_i \in \mathcal{P}^*$ do	3: $\mathcal{P} =$ partition induced by t on E
4: $tree_i = \text{PCT}(E_i)$	4: $h = \text{Var}(E) - \sum_{E_i \in \mathcal{P}} \frac{ E_i }{ E } \text{Var}(E_i)$
5: return $\text{node}(t^*, \bigcup_i \{tree_i\})$	5: if $(h > h^*) \wedge \text{Acceptable}(t, \mathcal{P})$ then
6: else	6: $(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
7: return $\text{leaf}(\text{Prototype}(E))$	7: return $(t^*, h^*, \mathcal{P}^*)$

The main difference between the algorithm for learning PCTs and a standard decision tree learner (for example, see the C4.5 algorithm proposed by Quinlan (1993)) is that the former considers the variance function and the prototype function, that computes a label for each leaf, as parameters that can be instantiated for a given learning task. So far, the PCTs have been instantiated for the following tasks: multiple targets prediction (Struyf and Džeroski, 2006; Kocev et al, 2007), hierarchical-multi label classification (Vens et al, 2008) and prediction of time-series (Slavkov et al, 2010). In this article, we focus on the first two tasks.

3.1.1 PCTs for predicting multiple target variables

PCTs that are able to predict multiple targets simultaneously are called multi-target decision trees (MTDTs). The MTDTs that predict a tuple of continuous variables (regression tasks) are called multi-target regression trees (MTRTs), while the MTDTs that predict a tuple of discrete variables are called multi-target classification trees (MTCTs). The instantiation of the CLUS system that learns multi-target trees is called CLUS-MTDT.

The variance and prototype functions for MTRTs are instantiated as follows. The variance is calculated as the sum of the variances of the target variables, i.e., $\text{Var}(E) = \sum_{i=1}^T \text{Var}(Y_i)$. The variances of the targets are normalized, so each target contributes equally to the overall variance. The prototype function (calculated at each leaf) returns as a prediction the tuple with the mean values of the target variables, calculated by using the training instances that belong to the given leaf.

The variance function for the MTCTs is computed as the sum of the Gini indices of the target variables, i.e., $\text{Var}(E) = \sum_{i=1}^T \text{Gini}(E, Y_i)$. Furthermore, one can also use the sum of the entropies of class variables as a variance function, i.e., $\text{Var}(E) = \sum_{i=1}^T \text{Entropy}(E, Y_i)$ (this definition has also been used in the context

of multi-label prediction (Clare, 2003)). The CLUS system also implements other variance functions, such as reduced error, information gain, gain ratio and the m -estimate.

The prototype function returns a vector of probabilities that an instance belongs to a given class for each target variable. Using these probabilities, the most probable (majority) class for each target attribute can be calculated.

3.1.2 PCTs for hierarchical multi-label classification

Hierarchical multi-label classification is a variant of classification where a single example may belong to multiple classes at the same time and the classes are organized in a form of hierarchy. An example that belongs to some class c automatically belongs to all super-classes of c : This is called the hierarchical constraint. Problems of this kind can be found in many domains including text classification, functional genomics, and object/scene classification. Silla and Freitas (2011) give a detailed overview of the possible application areas and the available approaches to HMC.

Silla and Freitas (2011) describe the algorithms for hierarchical classification with a 4-tuple $\langle \Delta, \Sigma, \Omega, \Theta \rangle$. In this 4-tuple, Δ indicates whether the algorithm makes predictions for a single or multiple paths in the hierarchy, Σ is the depth of the predicted classes, Ω is the taxonomy structure of the classes that the algorithm can handle, and Θ is the type of the algorithm (local or global). Using this categorization, the algorithm we present here can be described as follows:

- Δ = multiple path prediction: the algorithm can assign single or multiple paths or predicted classes to each instance.
- Σ = non-mandatory leaf-node prediction: an instance can be labeled with a label at any level of the taxonomy.
- Ω = tree or directed acyclic graph: the algorithm can handle both tree-shaped or DAG hierarchies of classes.
- Θ = global classifier: the algorithm constructs a single model valid for all classes.

CLUS-HMC is the instantiation (with the distances and prototypes as defined below) of the PCT algorithm for hierarchical classification implemented in the CLUS system. The variance and prototype are defined as follows (Vens et al, 2008). First, the set of labels of each example is represented as a vector with binary components; the i 'th component of the vector is 1 if the example belongs to class c_i and 0 otherwise. It is easily checked that the arithmetic mean of a set of such vectors contains as i 'th component the proportion of examples of the set belonging to class c_i .

The variance of a set of examples E is defined as the average squared distance between each example's class vector (L_i) and the set's mean class vector (\bar{L}), i.e.,

$$Var(E) = \frac{1}{|E|} \cdot \sum_{E_i \in E} d(L_i, \bar{L})^2.$$

In the HMC context, the similarity at higher levels of the hierarchy is more important than the similarity at lower levels. This is reflected in the distance

measure used in the above formula, which is a weighted Euclidean distance:

$$d(L_1, L_2) = \sqrt{\sum_{l=1}^{|L|} w(c_l) \cdot (L_{1,l} - L_{2,l})^2},$$

where $L_{i,l}$ is the l 'th component of the class vector L_i of an instance E_i , $|L|$ is the size of the class vector, and the class weights $w(c)$ decrease with the depth of the class in the hierarchy. More precisely, $w(c) = w_0 \cdot \text{avg}_j \{w(p_j(c))\}$, where $p_j(c)$ denotes the j 'th parent of class c and $0 < w_0 < 1$.

For example, consider the toy class hierarchy shown in Figure 1(a,b), and two data examples: (X_1, S_1) and (X_2, S_2) that belong to the classes $S_1 = \{c_1, c_2, c_{2.2}\}$ (boldface in Figure 1(b)) and $S_2 = \{c_2\}$, respectively. We use a vector representation with consecutive components representing membership of class c_1 , c_2 , $c_{2.1}$, $c_{2.2}$ and c_3 , in that order (preorder traversal of the tree). The distance is then calculated as follows:

$$d(S_1, S_2) = d([1, 1, 0, 1, 0], [0, 1, 0, 0, 0]) = \sqrt{w_0 + w_0^2}.$$

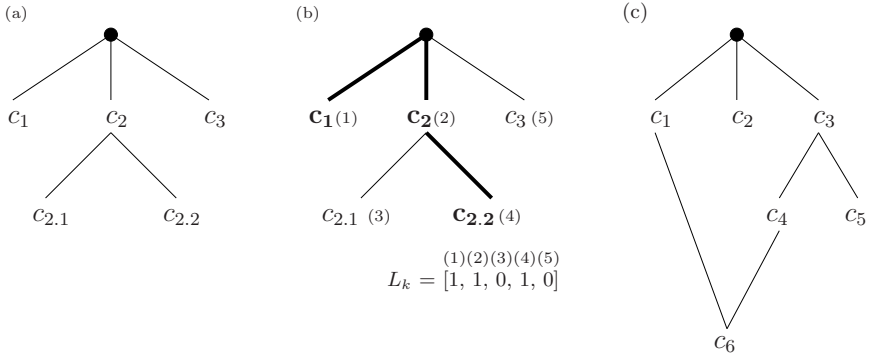


Fig. 1 Toy examples of hierarchies structured as a tree and a DAG. (a) Class label names contain information about the position in the hierarchy, e.g., $c_{2.1}$ is a subclass of c_2 . (b) The set of classes $\{c_1, c_2, c_{2.2}\}$, shown in bold in the hierarchy, represented as a vector. (c) A class hierarchy structured as a DAG. The class c_6 has two parents: c_1 and c_4 .

Note that our definition of $w(c)$ allows the classes to be structured in the form of directed acyclic graph (DAG). Figure 1(c) depicts an example of a DAG structured hierarchy. When the hierarchy is a DAG, then the depth of a class is not unique: classes do not have a single path from the top-node (for example see class c_6 in Figure 1(c)). After an extensive experimental evaluation, Vens et al (2008) recommend to use as a weight of a given class the average over the depths of all its parents ($w(c) = w_0 \cdot \text{avg}_j \{w(\text{par}_j(c))\}$).

A classification tree stores in a leaf the majority class for that leaf, which will be the tree's prediction for all examples that will arrive in the leaf. In the case of HMC, an example may have multiple classes, thus the notion of *majority class* does not apply in a straightforward manner. Instead, the mean \bar{L} of the class vectors of the examples in the leaf is stored as a prediction. Note that the value

for the i -th component of \bar{L} can be interpreted as the probability that an example arriving at the given leaf belongs to class c_i .

The prediction for an example that arrives in the leaf can be obtained by applying a user defined threshold τ on the probability; if the i -th component of \bar{L} is above τ then the examples belong to the class c_i . When a PCT is making a prediction it preserves the hierarchy constraint (the predictions comply to the parent child relationships from the hierarchy) if the values for the thresholds τ are chosen as follows: $\tau_i \leq \tau_j$ whenever $c_i \leq_h c_j$ (c_i is ancestor of c_j). The threshold τ is selected depending on the context. The user may set the threshold such that the resulting classifier has high precision at the cost of lower recall or vice versa, to maximize the F-score, to maximize the interpretability or plausibility of the resulting model etc. In this work, we use a threshold-independent measure (precision-recall curves) to evaluate the performance of the HMC models.

3.2 Ensembles of PCTs for predicting structured outputs

An ensemble is a set of predictive models (called base predictive models). In homogeneous ensembles, such as the ones we consider here, the base predictive models are constructed by using the same algorithm. The prediction of an ensemble for a new instance is obtained by combining the predictions of all base predictive models from the ensemble. In this article, we consider ensembles of PCTs for structured prediction. The PCTs in the ensembles are constructed by using bagging and random forests methods that are often used in the context of decision trees. We have adapted these methods to use PCTs.

3.2.1 Constructing ensembles of PCTs

A necessary condition for an ensemble to have better predictive performance than any of its individual members, is that the base predictive models are accurate and diverse (Hansen and Salamon, 1990). An accurate predictive model does better than random guessing on new examples. Two predictive models are diverse if they make different errors on new examples. There are several ways to introduce diversity in a set of base predictive models: by manipulating the training set (by changing the weight of the examples (Breiman, 1996; Freund and Schapire, 1996), by changing the attribute values of the examples (Breiman, 2001b), by manipulating the feature space (Breiman, 2001a; Ho, 1998)) and by manipulating the learning algorithm itself (Breiman, 2001a; Dietterich, 2000).

We have implemented the bagging and random forests methods within the CLUS system. These two ensemble learning techniques are most widely known and have primarily been used in the context of decision trees. The algorithms of these ensemble learning methods are presented in Table 2. For the random forests method (right in Table 2), the PCT algorithm for structured prediction needed changes: A randomized version of the selection of attributes was implemented, which replaced the standard selection of attributes.

Table 2 The ensemble learning algorithms: bagging and random forests. Here, E is the set of the training examples, k is the number of trees in the forest, and $f(D)$ is the size of the feature subset that considered at each node during tree construction for random forests.

procedure Bagging(E, k)	procedure RForest($E, k, f(D)$)
returns Forest	returns Forest
1: $F = \emptyset$	1: $F = \emptyset$
2: for $i = 1$ to k do	2: for $i = 1$ to k do
3: $E_i = \text{bootstrap}(E)$	3: $E_i = \text{bootstrap}(E)$
4: $T_i = \text{PCT}(E_i)$	4: $T_i = \text{PCT_rnd}(E_i, f(D))$
5: $F = F \cup T_i$	5: $F = F \cup T_i$
6: return F	6: return F

3.2.2 Bagging

Bagging (Breiman, 1996) is an ensemble method that constructs the different classifiers by making bootstrap replicates of the training set and using each of these replicates to construct a predictive model (Table 2, left). Each bootstrap sample is obtained by randomly sampling training instances, with replacement, from the original training set, until an equal number of instances as in the training set is obtained. Breiman (1996) showed that bagging can give substantial gains in predictive performance, when applied to an unstable learner (i.e., a learner for which small changes in the training set result in large changes in the predictions), such as classification and regression tree learners.

3.2.3 Random forests

A random forest (Breiman, 2001a) is an ensemble of trees, where diversity among the predictors is obtained by using bootstrap replicates as in bagging, and additionally by changing the set of descriptive attributes during learning (Table 2, right). More precisely, at each node in the decision trees, a random subset of the descriptive attributes is taken, and the best attribute is selected from this subset. The number of attributes that are retained is given by a function f of the total number of descriptive attributes D (e.g., $f(D) = 1$, $f(D) = \lfloor \sqrt{D} + 1 \rfloor$, $f(D) = \lfloor \log_2(D) + 1 \rfloor \dots$). By setting $f(D) = D$, we obtain the bagging procedure. The algorithm for learning a random forest using PCTs as base classifiers is presented in Table 2.

3.2.4 Combining the predictions of individual PCTs

The prediction of an ensemble for a new instance is obtained by combining the predictions of all the base predictive models from the ensemble. The predictions from the models can be combined by taking the average (for regression tasks) and the majority or probability distribution vote (for classification tasks), as described in (Bauer and Kohavi, 1999; Breiman, 1996), or by taking more complex aggregation schemes (Kuncheva, 2004).

We use PCTs as base predictive models for the ensembles for structured outputs. To obtain a prediction from an ensemble for predicting structured outputs, we accordingly extend the voting schemes. For the datasets with multiple continuous targets, as prediction of the ensemble, we take average per target of the

predictions of the base classifiers. For the datasets for hierarchical classification we also use the average of the predictions and apply the thresholding described in Section 3.1.2. We obtain the ensemble predictions for the datasets with multiple discrete targets using probability distribution voting (as suggested by Bauer and Kohavi (1999)) per target.

3.3 Local prediction of structured outputs with PCTs and ensembles

The presented structured output learning algorithms (CLUS-MTDT and CLUS-HMC) belong to the group of methods known as ‘big-bang’ or global predictive models (Silla and Freitas, 2011; Bakır et al, 2007). Global predictive models make a single prediction for the entire structured output, i.e., simultaneously predict all of its components. Local predictive models of structured outputs, on the other hand, use a collection of predictive models, each predicting a component of the overall structure that needs to be predicted.

The local predictive models for the task of predicting multiple targets are constructed by learning a predictive model for each of the targets separately. In the task of hierarchical multi-label classification, however, there are four different approaches that can be used: flat classification, local classifiers per level, local classifiers per node, and local classifiers per parent node (see (Silla and Freitas, 2011) for details).

Vens et al (2008) investigated the performance of the last two approaches with local classifiers over a large collection of datasets from functional genomics. The conclusion of the study was that the last approach (called hierarchical single-label classification - HSC) performs better in terms of predictive performance, smaller total model size and faster induction times.

In particular, the CLUS-HSC algorithm by Vens et al (2008), presented in Figure 2(a), constructs a decision tree classifier for each edge (connecting a class c with a parent class $par(c)$) in the hierarchy, thus creating an architecture of classifiers. The corresponding tree predicts membership to class c , using the instances that belong to $par(c)$. The construction of this type of tree uses few instances: only instances labeled with $par(c)$ are used for training. The instances labeled with class c are positive instances, while the ones that are labeled with $par(c)$, but not with c are negative.

The resulting HSC tree predicts the conditional probability $P(c|par(c))$. A new instance is predicted by recursive application of the product rule $P(c) = \min_j P(c | par_j(c)) \cdot P(par_j(c))$ (with $par_j(c)$ denoting the j -th parent of c in the case of a DAG), starting from the tree for the top-level class. Again, the probabilities are thresholded to obtain the set of predicted classes. To satisfy the hierarchy constraint, the threshold τ should be chosen as in the case of CLUS-HMC.

In this article, we extend the approach of Vens et al (2008) by applying ensembles as local classifiers, instead of single decision trees. The CLUS-HSC algorithm can be applied to ensemble learning in two ways: by constructing an ensemble of architectures or an architecture of ensembles. The first approach creates the ensemble by creating multiple architectures (similar to the one shown in Figure 2(a)). These multiple architectures can be created on different bootstrap replicates, on different feature spaces, by different local classifiers etc. The second approach is

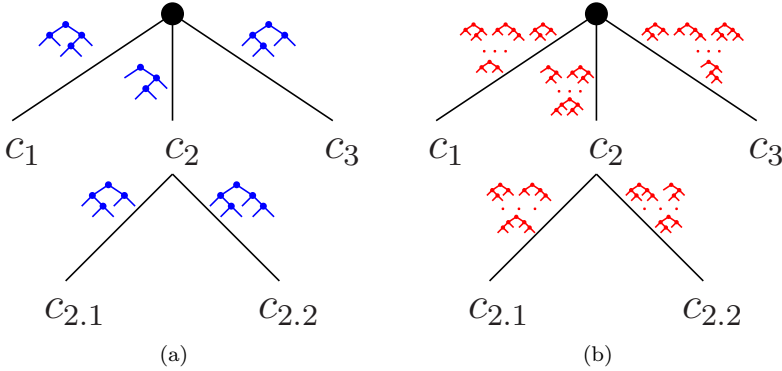


Fig. 2 An illustration of the hierarchical single-label classification approach used by Vens et al (2008). The local classifiers at each branch from the hierarchy are: (a) decision trees and (b) ensembles of decision trees.

simpler and, instead of a single local classifier (for example a decision tree), uses an ensemble as a classifier at each branch (depicted in Figure 2(b)). We prefer here the second approach since it is closer to the learning of local classifiers for predicting multiple target variables.

4 Experimental design

In this section, we describe the procedure for experimental evaluation of the proposed ensemble methods for predicting structured outputs. First, we state the questions we consider. Next, we present the datasets we use to evaluate the algorithms, and then the evaluation measures we applied. In the last subsection, we give the parameter values used in the algorithms and the statistical tests that we used.

4.1 Experimental questions

Given the methodology from Section 3, we construct several types of trees and ensembles. First, we construct PCTs that predict components of the structured output: a separate tree for each variable from the target tuple (predicting multiple targets) and a separate tree for each hierarchy edge (hierarchical classification). Second, we learn PCTs that predict the entire structured output simultaneously: a tree for the complete target tuple and a tree for the complete hierarchy, respectively. Finally, we construct the ensemble classifiers in the same manner by using both bagging and random forests.

We consider the following questions:

- *Predictive performance*: Does the performance of PCTs for structured outputs increases when combined into an ensemble? Can exploitation of the structure of the output lift the predictive performance of an ensemble?
- *Convergence*: Does the performance of the ensembles for structured outputs converge/saturate faster than for ensembles that predict components of the output?

- *Efficiency*: How much can the learning process benefit, in terms of time and memory consumption, from the ensembles for structured outputs as compared to the sets of ensembles that predict components of the structured output?

We compare the algorithms that predict the complete structured output with the algorithms that predict the components of the structured outputs separately. First, we inspect the predictive performance of all algorithms. Next, we focus only on the ensembles and examine their predictive performance at different ensemble sizes (i.e., we construct saturation curves). Our intention is to investigate whether the performance of the ensembles for structured outputs saturates at a smaller number of trees as compared to the saturation of ensembles predicting the components of the structured output. At the end, we compare the running times and the sizes of the obtained models.

4.2 Descriptions of the datasets

In this section, we present the datasets that were used to evaluate the performance of the ensembles. The datasets are divided into three groups based on the type of their output: multiple continuous targets datasets (regression), multiple discrete targets datasets (classification) and hierarchical multi-label classification datasets (HMC). Statistics about the used datasets are presented in Tables 3, 4, and 5, respectively.

Table 3 Properties of the datasets with multiple continuous targets (regression datasets); N is the number of instances, D/C the number of descriptive attributes (discrete/continuous), and T the number of target attributes.

Name of dataset	N	$ D / C $	T
Collembola (Kampichler et al, 2000)	393	8/39	3
EDM (Karalič, 1995)	154	0/16	2
Forestry-Kras (Stojanova et al, 2010)	60607	0/160	11
Forestry-Slivnica-LandSat (Stojanova, 2009)	6218	0/150	2
Forestry-Slivnica-IRS (Stojanova, 2009)	2731	0/29	2
Forestry-Slivnica-SPOT (Stojanova, 2009)	2731	0/49	2
Sigma real (Demšar et al, 2005)	817	0/4	2
Soil quality (Demšar et al, 2006)	1944	0/142	3
Solar-flare 1 (Asuncion and Newman, 2007)	323	10/0	3
Solar-flare 2 (Asuncion and Newman, 2007)	1066	10/0	3
Vegetation Clustering (Gjorgjioski et al, 2008)	29679	0/65	11
Vegetation Condition (Kocev et al, 2009)	16967	1/39	7
Water quality (Blockeel et al, 1999; Džeroski et al, 2000)	1060	0/16	14

The datasets with multiple continuous targets (13 in total, see Table 3) are mainly from the domain of ecological modelling. The datasets with multiple discrete targets (9 in total, see Table 4) are from various domains: ecological modelling (*Sigma Real* and *Water Quality*), biology (*Yeast*), multimedia (*Scene* and *Emotions*) and media space analysis (*Mediana*). The datasets that have classes organized in a hierarchy come from various domains, such as: biology (*Expression-FunCat*, *SCOP-GO*, *Yeast-GO* and *Sequence-FunCat*), text classification (*En-*

Table 4 Properties of the datasets with multiple discrete targets (classification datasets); N is the number of instances, D/C the number of descriptive attributes (discrete/continuous), and T the number of target attributes.

Name of dataset	N	$ D / C $	T
EDM (Karalič, 1995)	154	0/16	2
Emotions (Trohidis et al, 2008)	593	0/72	6
Mediana (Skrjanc et al, 2001)	7953	21/58	5
Scene (Boutell et al, 2004)	2407	0/294	6
Sigma real (Demšar et al, 2005)	817	0/4	2
Solar-flare 1 (Asuncion and Newman, 2007)	323	10/0	3
Thyroid (Asuncion and Newman, 2007)	9172	22/7	7
Water quality (Blockeel et al, 1999; Džeroski et al, 2000)	1060	0/16	14
Yeast (Elisseff and Weston, 2001)	2417	0/103	14

ron, *Reuters* and *WIPO*) and image annotation/classification (*ImCLEF07D*, *ImCLEF07A* and *Diatoms*). Hence, we use 10 datasets from 3 domains (see Table 5). Note that two datasets from the biological domain have a hierarchy organized as a DAG (they have GO in the dataset name), and the remaining datasets have tree-shaped hierarchies. For more details on the datasets, we refer the reader to the referenced literature.

Table 5 Properties of the datasets with hierarchical targets; N_{tr}/N_{te} is the number of instances in the training/testing dataset, D/C is the number of descriptive attributes (discrete/continuous), $|\mathcal{H}|$ is the number of classes in the hierarchy, \mathcal{H}_d is the maximal depth of the classes in the hierarchy, $\bar{\mathcal{L}}$ is the average number of labels per example, and $\bar{\mathcal{L}}_L$ is the average number of leaf labels per example.

Domain	N_{tr}/N_{te}	$ D / C $	$ \mathcal{H} $	\mathcal{H}_d	$\bar{\mathcal{L}}$	$\bar{\mathcal{L}}_L$
ImCLEF07D(Dimitrovski et al, 2008)	10000/1006	0/80	46	3.0	3.0	1.0
ImCLEF07A(Dimitrovski et al, 2008)	10000/1006	0/80	96	3.0	3.0	1.0
Diatoms (ADIAC, 2008)	2065/1054	0/371	377	3.0	1.95	0.94
Enron (Klimt and Yang, 2004)	988/660	0/1001	54	3.0	5.30	2.84
Reuters (Lewis et al, 2004)	3000/3000	0/47236	100	4.0	3.20	1.20
WIPO (Rousu et al, 2006)	1352/358	0/74435	183	4.0	4.0	1.0
Expression-FunCat (Clare, 2003)	2494/1291	4/547	475	4.0	8.87	2.29
SCOPE-GO (Clare, 2003)	6507/3336	0/2003	523	5.5	6.26	0.95
Sequence-FunCat (Clare, 2003)	2455/1264	2/4448	244	4.0	3.35	0.94
Yeast-GO (Barutcuoglu et al, 2006)	2310/1155	5588/342	133	6.3	5.74	0.66

4.3 Evaluation measures

Empirical evaluation is the most widely used approach for assessing the performance of machine learning algorithms. The performance of a machine learning algorithm is assessed using some evaluation measure. The different machine learning tasks, described in Section 2.1, use ‘task-specific’ evaluation measures. We first describe the evaluation measures for multiple continuous targets (regression), then for multiple discrete targets (classification) and at the end for hierarchical classification.

For the task of predicting multiple continuous targets (regression), we employed three well known measures: the correlation coefficient (*CC*), root mean squared error (*RMSE*) and relative root mean squared error (*RRMSE*). For each of these measures, we performed tests for statistical significance and constructed saturation curves. We present here only the results in terms of *RRMSE*, but same conclusions hold for the other two measures.

What evaluation measure to use in the case of classification algorithms is not as clear as in the case of regression. Sokolova and Lapalme (2009) conducted a systematic analysis of twenty four performance measures that can be used in a classification context. They conclude that evaluation measures for classification algorithms should be chosen based on the application domain.

In our study, we used seven evaluation measures for classification: accuracy, precision, recall, F-score, the Matthews correlation coefficient, balanced accuracy (also known as Area Under the Curve) and discriminant power. We used two averaging approaches to adapt these measures for multi-class problems: micro and macro averaging (note that averaging is not needed for accuracy). Since the goal of this study is not to assess the evaluation measures themselves, we present here only the results in terms of the micro average F-score ($F = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$). However, the conclusions drawn from the evaluation of the performance of the algorithms using the other measures concur with the ones presented here.

In the case of hierarchical classification, we evaluate the algorithms using the Area Under the Precision-Recall Curve (*AUPRC*), and in particular, the Area Under the Average Precision-Recall Curve (*AUPRC*) as suggested by Vens et al (2008). A Precision-Recall curve plots the precision of a classifier as a function of its recall. The points in the *PR* space are obtained by varying the value for the threshold τ from 0 to 1 with step 0.02. The precision and recall are micro averaged for all classes from the hierarchy.

Finally, we compare the algorithms by measuring their efficiency in terms of time consumption and size of the models. We measure the processor time needed to construct the models: in the case of predicting the components of the structure, we sum the times needed to construct the separate models. In a similar way, we calculated the sizes of the models as the total number of nodes (internal nodes and leafs). The experiments for predicting multiple targets were performed on a server running Linux, with two Intel Quad-Core Processors@2.5GHz and 64GB of RAM. The experiments for the hierarchical classification were run on a cluster of AMD Opteron processors (1.8 – 2.4GHz, \geq 2GB RAM).

4.4 Experimental setup

Here, we first state the parameter values used in the algorithms for constructing the single trees and the ensembles for all types of targets. We then describe how we assessed the statistical significance of the differences in performance of the studied algorithms.

The single trees for all types of outputs are obtained using F-test pruning. This pruning procedure uses the exact Fisher test to check whether a given split/test in an internal node of the tree results in a reduction in variance that is statistically significant at a given significance level. If there is no split/test that can satisfy this, then the node is converted to a leaf. An optimal significance level was selected by

using internal 3-fold cross validation, from the following values: 0.125, 0.1, 0.05, 0.01, 0.005 and 0.001.

The construction of an ensemble takes, as an input parameter, the size of the ensemble, i.e., number of base predictive models to be constructed. We constructed ensembles with 10, 25, 50, 75 and 100 base predictive models for all types of outputs and all datasets. In addition, for the datasets with multiple continuous targets we constructed ensembles with 150 and 250 base predictive models, and for the datasets with multiple discrete targets ensembles with 250, 500 and 1000 base predictive models. Following the findings from the study conducted by Bauer and Kohavi (1999), the trees in the ensembles were not pruned.

The random forests algorithm takes as input the size of the feature subset that is randomly selected at each node. For the multiple targets datasets, we apply the logarithmic function of the descriptive attributes $\lfloor \log_2 |D| \rfloor + 1$, which is recommended by Breiman (2001a). For the hierarchical classification datasets, we used $\lfloor 0.1 \cdot |D| \rfloor + 1$, since the feature space of some of these datasets is large (several thousands of features, see Table 5) and the logarithmic function is under-sampling the feature space.

On the datasets with multiple targets, the predictive performance of the algorithms is estimated by 10-fold cross-validation. The hierarchical datasets were previously divided (by the data providers) into train and test sets. Thus, we estimate the predictive performance of the algorithms on the test sets.

We adopt the recommendations by Demšar (2006) for the statistical evaluation of the results. We use the Friedman test (Friedman, 1940) for statistical significance with the correction from Iman and Davenport (1980). Afterwards, to check where the statistically significant differences appear (between which algorithms), we use the Nemenyi post-hoc test (Nemenyi, 1963). We present the results from the statistical analysis with *average ranks diagrams*. The diagrams plot the average ranks of the algorithms and connect the ones whose average ranks are smaller than a given value, called critical distance. The critical distance depends on the level of the statistical significance, in our case 0.05. The difference in the performance of the algorithms connected with a line is not statistically significant at the given significance level.

5 Results and discussion

The results from the experiments can be analyzed along several dimensions. First, we present the saturation curves of the ensemble methods (both for predicting the structured outputs and the components of the outputs). Then, we compare models that predict the complete structured output vs. models that predict components of the structured output. We thereby also compare single trees vs. ensembles of trees. Last, we evaluate the algorithms by their efficiency in terms of running time and model size. We do these comparisons for each task separately: multiple targets regression, multiple targets classification and hierarchical multi-label classification. We conclude the section with a general discussion.

5.1 Multi-target regression

In Figure 3, we present the saturation curves for the ensemble methods for multi-target regression. Although these curves are averaged across all target variables for a given dataset, they still provide useful insight on the performance of the algorithms. Figure 3 (a) and (b) present the saturation curves for two specific datasets, while Figure 3 (c) presents averaged curves for all datasets. We have checked at which ensemble size (saturation point) the RRMSE is not longer statistically significantly changed, i.e., when adding trees in the ensemble does not increase the predictive performance significantly. For all algorithms, the difference is not statistically significant after 50 trees are added. Thus, we compare the performance of the algorithms after 50 trees for the rest of the analysis.

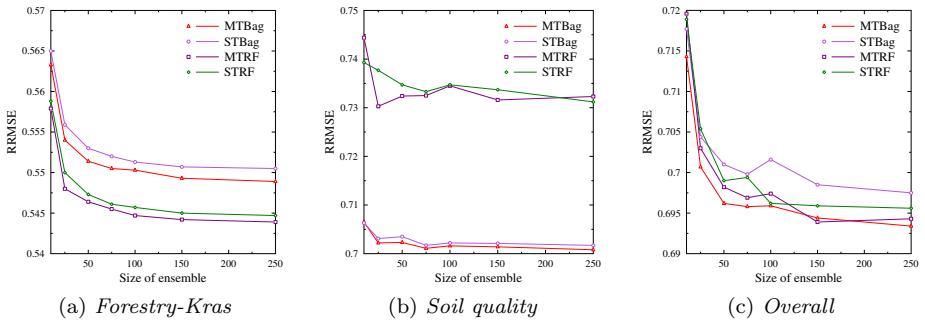


Fig. 3 Saturation curves for multi-target regression. These curves are obtained by averaging the $RRMSE$ values for all of the target variables in a dataset. Smaller $RRMSE$ values mean better predictive performance. The algorithm names are abbreviated as follows: random forests for the prediction of multiple targets – *MTRF*, random forests for the prediction of a single target – *STRF*, bagging for the prediction of multiple targets – *MTBag* and bagging for the prediction of a single target – *STBag*. Note that, in order to increase visibility the range of the y-axis is adapted for each curve.

The statistical test in Figure 4 shows that the difference in the predictive performance among the different ensemble methods is not statistically significant at the level of 0.05. For most of the datasets, the best performing method is random forests for predicting multiple targets. However, if we look at the saturation curves in Figure 3 (c), we note that, on average, multiple targets bagging is best. A closer look at the results learns that, especially for the larger datasets (e.g., *Forestry-Kras* from Figure 3(a)), random forests tend to outperform bagging, both for the local and global algorithms. The difference in performance between ensembles and the single PCTs is statistically significant at 0.05. The PCTs for predicting multiple targets simultaneously are better (though not statistically significant) than the trees for predicting the multiple targets separately.

Finally, we compare the algorithms by their running time and the size of the models for ensembles of 50 trees (see Figure 5). The statistical tests show that, in terms of time efficiency, random forests for multi-target regression significantly outperform ensemble methods for predicting the targets separately. Also, bagging for multiple targets is significantly faster than bagging for separate prediction of the targets. In terms of size of models, both random forests and bagging for

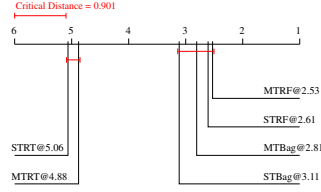


Fig. 4 Average rank diagrams at significance level of 0.05 for multi-target regression. The ensembles contain 50 trees. The difference in the performance of the algorithms connected with a red line are not statistically significant. The numbers after the name of the algorithm indicate its average rank. The abbreviations are same as in Figure 3 with addition of predicting clustering tree for multiple continuous targets – *MTRT* and predictive clustering tree for single continuous target – *STRT*.

predicting multiple targets simultaneously outperform significantly the ensembles that predict multiple targets separately.



Fig. 5 Efficiency (running time and model size) of the ensembles for prediction of multiple continuous targets. The size of the ensembles is 50 trees.

5.2 Multi-target classification

In Figure 6, we present three saturation curves for the ensemble methods for multi-target classification. Same as for multi-target regression, the values for drawing the curves are averaged from all target variables for a given dataset (and in Figure 6(c) averaged across all datasets). In a similar manner as for the multi-target regression, we determined the ensemble size after which the predictive performance is no longer statistically significant. The ensembles for predicting the multiple targets simultaneously saturate with 50 trees added, while the ensembles for separate prediction of the targets require more trees: 75 for the random forests and 250 for bagging. After this, we select ensemble sizes of 50 (Figure 7) to compare the algorithms. This is in line with the results from the saturation curves which show that ensembles for multi-target classification perform better than the ensembles for single-target classification at smaller ensemble sizes (this can be also noticed in the overall saturation curve shown in Figure 6(c)).

The statistical tests reveal that there is no statistically significant difference at the level of 0.05 in the performance of the ensemble methods (Figure 7). Bagging for predicting the multiple targets simultaneously is the best performing method (average rank 2.59) and the remaining methods have larger and very close to each other average ranks (ranging from 3.0 to 3.11) with random forest for separate

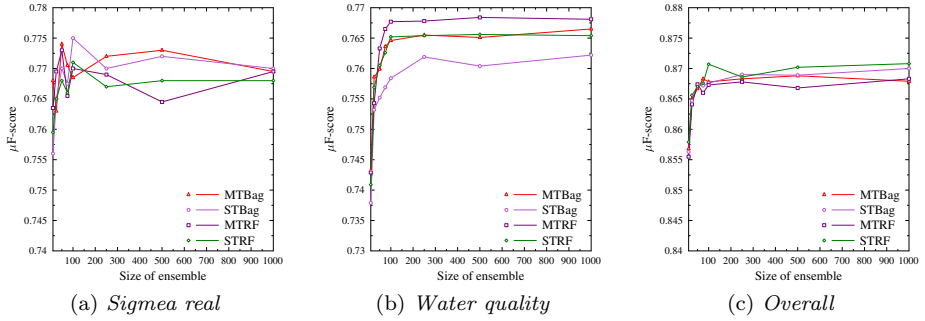


Fig. 6 Saturation curves for the prediction of multiple discrete targets. These curves are obtained by averaging the μF – score values for all of the target variables. Bigger μF – score values mean better predictive performance. The algorithm names are abbreviated as follows: random forests for prediction of multiple targets – *MTRF*, random forests for prediction of single target – *STRF*, bagging for prediction of multiple targets – *MTBag* and bagging for prediction of single target – *STBag*. Note that, in order to increase visibility the range of the y-axis is adapted for each curve.

prediction of the targets having the largest average rank. Similar conclusions can be made if instead of ensembles with 50 trees, we select ensembles with 75 or 250 trees. The only difference is that in these cases random forests for multiple targets have larger average ranks.

The both types of ensembles (multi-target and single-target classification) perform statistically significantly better than the single PCTs. Furthermore, the single PCTs for multi-target classification perform better (although not statistically significantly) than the PCTs for single-target classification.

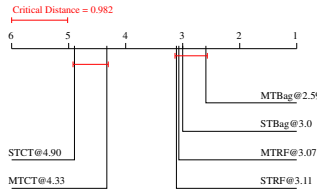


Fig. 7 Average ranks diagrams at significance level of 0.05 for prediction of multiple discrete targets. The ensembles contain 50 trees. The difference in the performance of the algorithms connected with a red line are not statistically significant. The numbers after the name of the algorithm indicate its average rank. The abbreviations are same as in Figure 6 with addition of predicting clustering tree for multiple discrete targets – *MTCT* and predictive clustering tree for single discrete target – *STCT*.

Finally, we compare the ensembles by their efficiency: running times (Figure 8(a)) and size of models (Figure 8(b)). Concerning the running time, we can state that the random forests for predicting multiple targets simultaneously significantly outperform the bagging for predicting the multiple targets separately. As for the size of the models, we can note the following: (1) the bagging for predicting multiple targets simultaneously significantly outperforms both ensemble methods for separate prediction of the targets and (2) random forests for predicting multi-

ple targets simultaneously significantly outperform the random forests for separate prediction of the targets.



Fig. 8 Efficiency of the ensembles for prediction of multiple discrete targets. The size of the ensembles is 50 trees.

5.3 Hierarchical multi-label classification

In this subsection, we compare the performance of ensembles and PCTs for the tasks of HMC and HSC. We begin by presenting the saturation curves of the ensemble methods in Figure 9. Figures 9(a) and 9(b) show the saturation curves for *SCOP-GO* and *ImCLEF07D* domains, respectively, while Figure 9(c) shows the saturation curve averaged across all domains. We determine the ensemble size after which adding trees in the ensemble does not statistically significantly improve ensemble’s performance. Both ensembles for HMC and random forests for HSC saturate after 50 trees are added in the ensemble, while bagging for HSC saturates after only 25 trees. We further compare the performance of the ensembles at 50 trees (Figure 10).

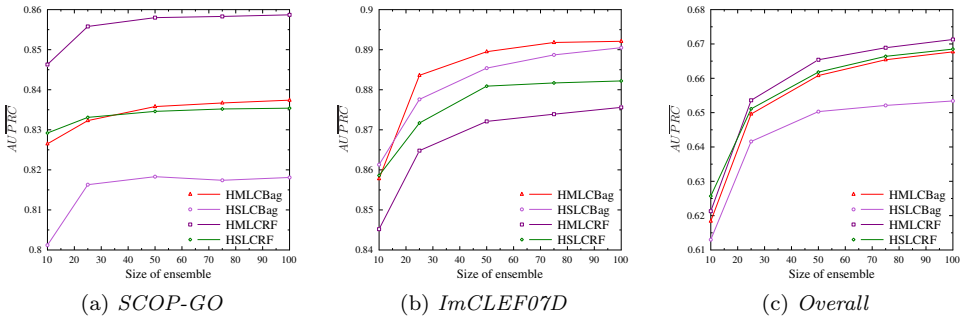


Fig. 9 Saturation curves for hierarchical multi-label classification. These curves are obtained by averaging the $AUPRC$ values for all of the target variables. Bigger $AUPRC$ values mean better predictive performance. The algorithms are abbreviated as follows: random forests for hierarchical multi-label classification – *HMLCRF*, random forests for hierarchical single-label classification – *HSLCRF*, bagging for hierarchical multi-label classification – *HMLCBag* and bagging for hierarchical single-label classification – *HSLCBag*. Note that, in order to increase visibility the range of the y-axis is adapted for each curve.

The average ranks diagram for the ensembles with 50 trees (Figure 10) shows that the performance of the ensembles is not statistically significantly different. Note that the best performing method is random forests for HSC (average rank

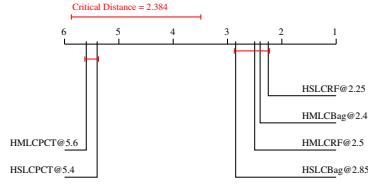


Fig. 10 Average ranks diagrams at significance level of 0.05 for hierarchical multi-label classification. The ensembles contain 50 trees. The difference in the performance of the algorithms connected with a red line are not statistically significant. The numbers after the name of the algorithm indicate its average rank. The abbreviations are same as in Figure 9 with addition of predicting clustering tree for hierarchical multi-label classification – *HMLCPCT* and predictive clustering tree for hierarchical single-label classification – *HSLCPCT*.

2.25) and worst performing method is bagging for HSC (average rank 2.85). Ensembles for HMC and HSC are statistically significantly better than single PCT for HMC and HSC. However, if we focus on the overall saturation curve from Figure 9(c), bagging for HMC is the best performing method. Moreover, ensembles for HMC perform better than ensembles for HSC on the datasets with larger hierarchies (i.e., datasets with $|\mathcal{H}| > 300$).

Finally, we compare the algorithms by their efficiency when they contain 50 trees (running times in Figure 11(a) and size of the models in Figure 11(b)). The random forests for HMC are statistically significantly faster than both bagging for HMC and HSC, while random forests for HSC are significantly faster than bagging for HSC. The models of bagging of HMC are statistically significantly smaller than the models from both ensembles for HSC, while the models of random forests for HMC are statistically significantly smaller than the models from the random forests for HSC.

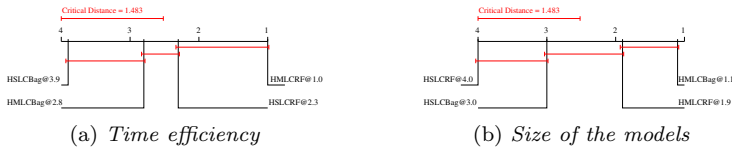


Fig. 11 Efficiency of the ensembles for hierarchical multi-label classifications. The size of the ensembles is 50 trees.

5.4 Summary of the results

To summarize the results from the experiments, we discuss some general conclusions by formulating an answer to each of the questions from Section 4.1.

5.4.1 Predictive performance

The ensembles for predicting structured outputs lift the predictive performance of a single PCT for predicting structured outputs. The difference in performance is

statistically significant at the significance level of 0.05. Previously this was shown only on applications where the target is a single continuous or discrete variable (Seni and Elder, 2010). This finding is valid for all three machine learning tasks that we consider in this article.

The differences in predictive performances between ensembles of PCTs and ensembles of trees predicting components of the output are not statistically significant at 0.05 in any task. Also, none of the four considered ensemble algorithms is consistently performing best. However, the ensembles of PCTs often have better predictive performance (i.e., smaller average ranks) than the ensembles of trees predicting components of the output.

5.4.2 Convergence

We looked at the saturation point of the ensembles' performances with respect to the number of base classifiers. In the majority of the cases, the predictive performance of the ensembles saturates after the 50th tree is added in the ensemble. Exceptions of this are: bagging for HSC that saturates when 25 trees are added and random forests and bagging for predicting multiple discrete targets separately that saturate after 75 and 250 trees, respectively. Furthermore, the saturation curves offer some insight about the application of a given method on a given dataset (summarized by their number of examples and number of descriptive variables). Also, the curves show that on majority of the datasets the ensembles for predicting the structured outputs as a whole have better performance than the ensembles that predict the components. This is especially the case when the ensembles contain fewer trees.

5.4.3 Efficiency

With respect to model size and induction times, the ensembles that exploit the structure do have a significant advantage. The advantage is more pronounced when the datasets have large number of instances and descriptive attributes. Moreover, because of the feature sampling, the random forests for predicting structured outputs benefit even more, in terms of induction time, when the datasets have many descriptive attributes.

Averaged over all datasets considered in this study, we obtain that random forests for predicting the complete structured outputs are 4 times faster to construct and the models are 3.4 times smaller than a collection of random forests for predicting single targets or labels. In addition, they are 5.5 times faster to construct and have models with similar size as bagging for predicting the complete structured output. Furthermore, the latter is 3.9 times faster and yields models that are 2.9 times smaller than a collection of bagged trees for predicting single targets or labels.

6 Conclusions

In this article, we present and compare several approaches for learning ensembles for structured output prediction. We compare approaches that exploit the output

structure to approaches that focus on a component of the structure. All approaches are implemented in the predictive clustering tree framework.

We performed the empirical evaluation over a wide range of datasets. In particular, we used 13 datasets for the task of multi-target regression, 9 datasets for the task of multi-target classification and 10 datasets for the task of hierarchical multi-label classification.

The experimental comparison comprises 32 datasets from three learning tasks: multiple targets regression, multiple targets classification, and hierarchical multi-label classification. To our knowledge, this is the first extensive comparison of ensemble methods for predicting structured outputs.

The main results are summarized as follows. First, we have shown that the fact that ensembles lift the predictive performance of a single classifier carries over to the context of structured outputs. This finding suggests that the non-trivial relations that might exist between the components of the structure are maintained when combining predictions of several classifiers or when injecting some source of randomness in the learning algorithm.

Second, we have presented saturation curves, which show that in the majority of cases, the predictive performance of the ensembles saturates at about 50 trees. Furthermore, the curves suggest that, especially when the ensembles contain smaller number of trees, for the majority of the datasets considered, the global ensemble methods have better predictive performance than the local ensemble methods.

Finally, we have shown that, although the ensembles for predicting structured output often have smaller average ranks than the ensembles for predicting the components of the structure, the differences in predictive performance of all ensemble methods are not statistically significant at the 0.05 level in any of the tasks. Furthermore, ensembles that predict the whole structure at once are preferable when considering efficiency in terms of total model size and induction times.

Acknowledgements The authors would like to thank Ivica Dimitrovski from the Faculty of Electrical Engineering and Information Technologies, Skopje, Macedonia for providing the image annotation datasets. Celine Vens is a postdoctoral fellow of the Research Fund – Flanders (FWO–Vlaanderen).

References

- ADIAC (2008) Automatic diatom identification and classification. <http://rbg-web2.rbge.org.uk/ADIAC/>
- Ando RK, Zhang T, Bartlett P (2005) A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research* 6:1817–1853
- Argyriou A, Evgeniou T, Pontil M (2008) Convex multi-task feature learning. *Machine Learning* 73:243–272
- Asuncion A, Newman D (2007) UCI - machine learning repository. <http://www.ics.uci.edu/~mllearn/MLRepository.html>, URL <http://www.ics.uci.edu/~mllearn/MLRepository.html>

- Bakır GH, Hofmann T, Schölkopf B, Smola AJ, Taskar B, Vishwanathan SVN (2007) Predicting structured data. Neural Information Processing, The MIT Press
- Bakker B, Heskes T (2003) Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Research* 4:83–99
- Barutcuoglu Z, Schapire RE, Troyanskaya OG (2006) Hierarchical multi-label prediction of gene function. *Bioinformatics* 22(7):830–836
- Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning* 36(1):105–139
- Baxter J (2000) A model of inductive bias learning. *Journal of Artificial Intelligence Research* 12:149–198
- Ben-David S, Borbely RS (2008) A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning* 73(3):273–287
- Blockeel H, Džeroski S, Grbović J (1999) Simultaneous prediction of multiple chemical parameters of river water quality with tilde. In: *Proceedings of the 3rd European Conference on PKDD - LNAI 1704*, Springer, pp 32–40
- Blockeel H, Struyf J (2002) Efficient algorithms for decision tree cross-validation. *Journal of Machine Learning Research* 3:621–650
- Blockeel H, Raedt LD, Ramon J (1998) Top-down induction of clustering trees. In: *Proceedings of the 15th International Conference on Machine Learning*, Morgan Kaufmann, pp 55–63
- Blockeel H, Bruynooghe M, Džeroski S, Ramon J, Struyf J (2002) Hierarchical multi-classification. In: *KDD-2002 Workshop Notes: MRDM 2002, Workshop on Multi-Relational Data Mining*, pp 21–35
- Blockeel H, Schietgat L, Struyf J, Džeroski S, Clare A (2006) Decision trees for hierarchical multilabel classification: A case study in functional genomics. In: *Knowledge Discovery in Databases: PKDD 2006 - LNCS 4213*, Springer Berlin / Heidelberg, pp 18–29
- Boutell M, Luo J, Shen X, Brown C (2004) Learning multi-label scene classification. *Pattern Recognition* 37(9):1757–1771
- Breiman L (1996) Bagging predictors. *Machine Learning* 24(2):123–140
- Breiman L (2001a) Random forests. *Machine Learning* 45(1):5–32
- Breiman L (2001b) Using iterated bagging to debias regressions. *Machine Learning* 45(3):261–277
- Breiman L, Friedman J (1997) Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 59(1):3–54
- Breiman L, Friedman J, Olshen R, Stone CJ (1984) *Classification and Regression Trees*. Chapman & Hall/CRC
- Brown PJ, Zidek JV (1980) Adaptive multivariate ridge regression. *The Annals of Statistics* 8(1):64–74
- Cai F, Cherkassky V (2009) Svm+ regression and multi-task learning. In: *International Joint Conference on Neural Networks (IJCNN)*, pp 418–424
- Caponnetto A, Micchelli CA, Pontil M, Ying Y (2008) Universal multi-task kernels. *Journal of Machine Learning Research* 9:1615–1646
- Caruana R (1997) Multitask learning. *Machine Learning* 28:41–75
- Chen Y, Xu D (2004) Global protein function annotation through mining genome-scale data in yeast *saccharomyces cerevisiae*. *Nucleic Acids Research* 32(21):6414–6424

- Clare A (2003) Machine learning and data mining for yeast functional genomics. PhD thesis, University of Wales Aberystwyth, Aberystwyth, Wales, UK
- Demšar D, Debeljak M, Džeroski S, Lavigne C (2005) Modelling pollen dispersal of genetically modified oilseed rape within the field. In: The Annual Meeting of the Ecological Society of America
- Demšar D, Džeroski S, Larsen T, Struyf J, Axelsen J, Bruns-Pedersen M, Krogh PH (2006) Using multi-objective classification to model communities of soil. *Ecological Modelling* 191(1):131–143
- Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7:1–30
- Dietterich TG (2000) Ensemble methods in machine learning. In: Proc. of the 1st International Workshop on Multiple Classifier Systems - LNCS 1857, Springer, pp 1–15
- Dimitrovski I, Kocev D, Loskovska S, Džeroski S (2008) Hierchical annotation of medical images. In: Proceedings of the 11th International Multiconference - Information Society IS 2008, IJS, Ljubljana, pp 174–181
- Džeroski S (2007) Towards a general framework for data mining. In: Džeroski S, Struyf J (eds) Knowledge Discovery in Inductive Databases, 5th International Workshop, KDID 2006, Revised Selected and Invited Papers, vol 4747, pp 259–300
- Džeroski S, Demšar D, Grbović J (2000) Predicting chemical parameters of river water quality from bioindicator data. *Applied Intelligence* 13(1):7–17
- Elisseeff A, Weston J (2001) A kernel method for multi-labelled classification. In: In Advances in Neural Information Processing Systems 14, MIT Press, pp 681–687
- Evgeniou T, Micchelli CA, Pontil M (2005) Learning multiple tasks with kernel methods. *Journal of Machine Learning Research* 6:615–637
- Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: Proc. of the Thirteenth International Conference on Machine Learning - ICML, Morgan Kaufman, pp 148–156
- Friedman M (1940) A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics* 11:86–92
- Gärtner T, Vembu S (2009) On structured output training: hard cases and an efficient alternative. *Machine Learning* 76:227–242
- Geurts P, Wehenkel L, D'Alché-Buc F (2006) Kernelizing the output of tree-based methods. In: ICML '06: Proceedings of the 23rd International Conference on Machine Learning, ACM, pp 345–352
- Gjorgjioski V, Džeroski S, White M (2008) Clustering analysis of vegetation data. Tech. Rep. 10065, Jožef Stefan Institute
- Greene WH (2007) Econometric analysis, 6th edn. Prentice Hall
- Guan Y, Myers CL, Hess DC, Barutcuoglu Z, Caudy AA, Troyanskaya OG (2008) Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome biology* 9(S1):S3+
- Hansen LK, Salamon P (1990) Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12(10):993–1001
- Ho TK (1998) The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(8):832–844
- Iman RL, Davenport JM (1980) Approximations of the critical region of the friedman statistic. *Communications in Statistics - Theory and Methods* 9(6):571–595

- Kampichler C, Džeroski S, Wieland R (2000) Application of machine learning techniques to the analysis of soil ecological data bases: relationships between habitat features and collembolan community characteristics. *Soil Biology and Biochemistry* 32(2):197–209
- Karalič A (1995) First order regression. PhD thesis, Faculty of Computer Science, University of Ljubljana, Ljubljana, Slovenia
- Klimt B, Yang Y (2004) The enron corpus: A new dataset for email classification research. In: *ECML '04: Proceedings of the 18th European Conference on Machine Learning – LNCS 3201*, Springer Berlin / Heidelberg, pp 217–226
- Kocev D, Vens C, Struyf J, Džeroski S (2007) Ensembles of multi-objective decision trees. In: *ECML '07: Proceedings of the 18th European Conference on Machine Learning – LNCS 4701*, Springer Berlin / Heidelberg, pp 624–631
- Kocev D, Džeroski S, White M, Newell G, Griffioen P (2009) Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. *Ecological Modelling* 220(8):1159–1168
- Kriegel HP, Borgwardt K, Kröger P, Pryakhin A, Schubert M, Zimek A (2007) Future trends in data mining. *Data Mining and Knowledge Discovery* 15:87–97
- Kuncheva L (2004) *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience
- Langley P (1996) *Elements of machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
- Lewis DD, Yang Y, Rose TG, Li F (2004) Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research* 5:361–397
- Micchelli CA, Pontil M (2004) Kernels for multi-task learning. In: *Advances in Neural Information Processing Systems 17 - Proceedings of the 2004 Conference*, pp 921–928
- Nemenyi PB (1963) Distribution-free multiple comparisons. PhD thesis, Princeton University, Princeton, NY, USA
- Obozinski G, Lanckriet G, Grant C, Jordan MI, Noble WS (2008) Consistent probabilistic outputs for protein function prediction. *Genome Biology* 9(S1):S6+
- Quinlan RJ (1993) *C4.5: Programs for Machine Learning*, 1st edn. Morgan Kaufmann
- Rousu J, Saunders C, Szedmak S, Shawe-Taylor J (2006) Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research* 7:1601–1626
- Schietgat L, Vens C, Struyf J, Blockeel H, Kocev D, Džeroski S (2010) Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinformatics* 11(2):1–14
- Seni G, Elder JF (2010) *Ensemble methods in data mining: Improving accuracy through combining predictions*. Morgan & Claypool Publishers
- Silla C, Freitas A (2011) A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22(1-2):31–72
- Skrjanc M, Grobelnik M, Zupanec D (2001) Insights offered by data-mining when analyzing media space data. *Informatica (Slovenia)* 25(3):357–363
- Slavkov I, Gjorgjioski V, Struyf J, Džeroski S (2010) Finding explained groups of time-course gene expression profiles with predictive clustering trees. *Molecular BioSystems* 6(4):729–740
- Sokolova M, Lapalme G (2009) A systematic analysis of performance measures for classification tasks. *Information Processing & Management* 45(4):427–437

- Stojanova D (2009) Estimating forest properties from remotely sensed data by using machine learning. Master's thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
- Stojanova D, Panov P, Gjorgjioski V, Kobler A, Džeroski S (2010) Estimating vegetation height and canopy cover from remotely sensed data with machine learning. *Ecological Informatics* 5(4):256–266
- Struyf J, Džeroski S (2006) Constraint based induction of multi-objective regression trees. In: Proc. of the 4th International Workshop on Knowledge Discovery in Inductive Databases KDID - LNCS 3933, Springer, pp 222–233
- Thrun S, Pratt L (1998) Learning to learn. Kluwer Academic Publishers
- Tian W, Zhang LV, Taşan M, Gibbons FD, King OD, Park J, Wunderlich Z, Cherry JM, Roth FP (2008) Combining guilt-by-association and guilt-by-profiling to predict *saccharomyces cerevisiae* gene function. *Genome biology* 9(S1):S7+
- Trohidis K, Tsoumakas G, Kalliris G, Vlahavas I (2008) Multilabel classification of music into emotions. In: Proc. 9th International Conference on Music Information Retrieval (ISMIR 2008), pp 325–330
- Valentini G, Re M (2009) Weighted true path rule: a multilabel hierarchical algorithm for gene function prediction. In: Proceedings of the 1st International Workshop on Learning from Multi-Label Data, pp 133–146
- Vens C, Struyf J, Schietgat L, Džeroski S, Blockeel H (2008) Decision trees for hierarchical multi-label classification. *Machine Learning* 73(2):185–214
- Wilson A, Fern A, Ray S, Tadepalli P (2007) Multi-task reinforcement learning: a hierarchical bayesian approach. In: ICML '07: Proceedings of the 24th international conference on Machine learning, ACM, pp 1015–1022
- Yang Q, Wu X (2006) 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making* 5(4):597–604