



# Ensembles for multi-target regression with random output selections

Martin Breskvar<sup>1,2</sup> · Dragi Kocev<sup>1,2</sup> · Sašo Džeroski<sup>1,2</sup>

Received: 10 April 2017 / Accepted: 3 July 2018 / Published online: 11 July 2018  
© The Author(s) 2018

## Abstract

We address the task of multi-target regression, where we generate global models that simultaneously predict multiple continuous variables. We use ensembles of generalized decision trees, called predictive clustering trees (PCTs), in particular bagging and random forests (RF) of PCTs and extremely randomized PCTs (extra PCTs). We add another dimension of randomization to these ensemble methods by learning individual base models that consider random subsets of target variables, while leaving the input space randomizations (in RF PCTs and extra PCTs) intact. Moreover, we propose a new ensemble prediction aggregation function, where the final ensemble prediction for a given target is influenced only by those base models that considered it during learning. An extensive experimental evaluation on a range of benchmark datasets has been conducted, where the extended ensemble methods were compared to the original ensemble methods, individual multi-target regression trees, and ensembles of single-target regression trees in terms of predictive performance, running times and model sizes. The results show that the proposed ensemble extension can yield better predictive performance, reduce learning time or both, without a considerable change in model size. The newly proposed aggregation function gives best results when used with extremely randomized PCTs. We also include a comparison with three competing methods, namely random linear target combinations and two variants of random projections.

**Keywords** Predictive clustering trees · Multi-target regression · Output space decomposition · Structured outputs · Ensemble methods

## 1 Introduction

Supervised learning is a highly active and researched area of machine learning. Its goal is to produce a model, that can take a previously unseen example and predict the value of a

---

Editors: Michelangelo Ceci and Toon Calders..

---

✉ Martin Breskvar  
[Martin.Breskvar@ijs.si](mailto:Martin.Breskvar@ijs.si)

<sup>1</sup> Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia

<sup>2</sup> Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

variable of interest, typically called a target variable. If the target variable is of a discrete type, the task at hand is classification. If the target variable is of a numeric data type, the task is called regression. Such single-target (ST) prediction scenarios are very common.

A number of challenges from various domains require a more complex representation of the data. In those cases, we need to move away from generating models that make predictions for one target variable to models that make predictions for multiple targets simultaneously, i.e., address the task of multi-target (MT) prediction. In general, MT prediction falls under the scope of structured output prediction (SOP). SOP, as the name suggests, is concerned with predicting values of structured data types, which are composed of values of primitive data types, e.g., boolean, real numbers, or discrete values (Panov et al. 2016). Examples of structured data types are tuples, sequences, sets, tree-shaped hierarchies, directed acyclic graphs, etc. (Džeroski 2007). Examples of SOP tasks are multi-target regression (MTR), (hierarchical) multi-label classification (MLC) and time series prediction. Solving SOP tasks has great potential and importance in many domains, and has been listed as one of the most challenging problems in machine learning by Yang and Wu (2006) and Kriegel et al. (2007).

This work considers the task of MTR—predicting multiple continuous variables. Many real life scenarios exist where one is interested in predicting multiple numerical values, e.g., in ecology (Demšar et al. 2006; Stojanova et al. 2010) and life sciences (Jančič et al. 2016). MT prediction methods differ mostly in the way they exploit the target space structure during learning a predictive model. The most natural and simple starting point is to make a model for each component of the structure separately. The models predicting the individual components are then combined to make predictions for the whole structure. Such methods are called local, because they learn a local model for only one component at a time, whilst ignoring the other components (i.e., global context). Hence, local methods cannot exploit information hidden in the combination of multiple components and relationships between them. In contrast to local methods, global methods take into account all the structural components and their relations and then make predictions for all of them simultaneously. In general, this makes global models more interpretable. Global models (as well as the process of learning them) are also more computationally efficient as compared to local ones. This becomes especially evident when the predicted structure consists of many components. Learning global models can therefore yield better predictive performance while consuming less resources.

In this paper, we propose a new ensemble extension method for the task of MTR, called Random Output Selections (ROS). The method uses predictive clustering trees (PCTs) as base models in the ensembles. PCTs, a generalization of decision trees, are global models for SOP able to solve MTR and MLC tasks, among other (Blockeel et al. 1998). The proposed method can be coupled with any ensemble learning method that employs global decision trees as base learners. The method learns each base model on a random subset of all target variables (each model has its own subset of variables). In this work we apply ROS on three different ensemble learning methods: bagging (Bag) (Breiman 1996; Kocev et al. 2013), random forests (RF) (Breiman 2001; Kocev et al. 2013) and extremely randomized trees (ET) (Geurts et al. 2006; Kocev and Ceci 2015). Analogously, we refer to extended methods as Bag-ROS, RF-ROS and ET-ROS.

The main focus of this study is to determine whether the proposed method can improve the predictive performance and shorten the learning times of the considered ensemble methods. An extensive empirical evaluation over a variety of benchmark datasets is performed in order to determine the effects of using ROS on predictive performance. In addition we perform an analysis with respect to time and space complexity.

We summarize the main contributions of this work as follows:

- A novel global ensemble extension approach for addressing the MTR task. It randomly selects subsets of targets for learning individual base models. This can yield better predictive performance and shorter learning times.
- A novel aggregation function for the prediction of base models in the extended ensembles. By default, each tree in the ensemble predicts all targets. Here, only targets that were considered during the learning of an individual tree contribute to the final predictions.
- An extensive empirical evaluation of the three different ensemble methods on 17 benchmark datasets, which provide performance assessment for the original and extended ensemble methods, as well as individual multi-target regression trees and ensembles of single-target regression trees (all over a range of ensemble sizes). The study also includes parameter setting recommendations for the proposed method. Moreover, we compare the performance to other competing methods that also transform the output space.
- Theoretical computational complexity analysis of the proposed ensemble extension method, linked to the empirical evidence of the aforementioned evaluation.

The remainder of this paper is organized as follows. Section 2 outlines the task definition and related work. Next, Sect. 3 presents the proposed method ROS. Section 4 then provides details about the experimental design and evaluation, the results of which are presented and discussed in Sect. 5. Finally, we conclude the paper and provide directions for further work.

## 2 Background and related work

### 2.1 Task definition

In this section, we formally describe the machine learning task of MTR. **Given:**

- An input space  $X$ , with tuples of dimension  $d$ , containing values of primitive data types, i.e.,  $\forall x_i \in X, x_i = (x_{i1}, x_{i2}, \dots, x_{id})$ ,
- An output (target) space  $Y$ , with tuples of dimension  $t$ , containing real values, i.e.,  $\forall y_i \in Y, y_i = (y_{i1}, y_{i2}, \dots, y_{it})$ , where  $y_{ik} \in \mathbb{R}$  and  $1 \leq k \leq t$
- A set of examples  $S$ , where each example is a pair of tuples from the input and the output space, i.e.,  $S = \{(x_i, y_i) | x_i \in X, y_i \in Y, 1 \leq i \leq N\}$  and  $N$  is the number of examples in  $S$  ( $N = |S|$ ),
- A quality criterion  $c$ , which rewards models with high predictive accuracy and low complexity.

**Find:** A function  $f : X \rightarrow Y$  such that  $f$  maximizes  $c$ .

In this work, the function  $f$  is represented by an ensemble (set) of PCTs. It will be learned by the approaches of bagging (Bag), Random Forests (RF) and extra-PCTs (ET) and their ROS counterparts Bag-ROS, RF-ROS and ET-ROS.

### 2.2 Related work

Our work relates to three main areas: solving the task of MTR, ensemble learning and output space decomposition. Multi-target regression, also referred to as multi-target, multi-variate or multi-response regression, is a machine learning task, where the goal is to predict multiple real-valued variables simultaneously. Borchani et al. (2015) divide multi-target regression

methods into two categories: *problem transformation methods* and *algorithm adaptation methods*.

Problem transformation methods include the work of Spyromitros-Xioufis et al. (2016) on single-target approaches, multi-target regressor stacking and regressor chains, and the work of Zhang et al. (2012) on multi-output support vector regression, and the work of Tsoumakas et al. (2014) on random linear target combinations. These methods transform the output space in such a way, that it is possible to apply existing methods to solve the task at hand. The transformation process usually converts a multi-target problem into several single-target ones, thus approaching the MTR problem locally. However, transformation methods exist, that solve the MTR problem locally but, on the other side, include multiple targets into the learning process thus making them not fully-local nor fully-global, e.g., Tsoumakas et al. (2014).

Algorithm adaptation methods, on the other hand, have the capability to handle multi-target tasks naturally, i.e., no transformation of the data is needed. Such methods are global. They can exploit the potential relatedness of the targets to learn models with better predictive performance faster as compared to problem transformation methods. Algorithm adaptation methods include statistical methods, such as those by Abraham et al. (2013), Breiman and Friedman (1997), Izenman (1975), multi-output support vector regression work by Xu et al. (2013), Han et al. (2012), Deger et al. (2012), kernel methods by Alvarez et al. (2012), Micchelli and Pontil (2004), multi-target regression trees (Kocev and Ceci 2015; Levatić et al. 2014; Appice and Malerba 2014; Kocev et al. 2013; Stojanova et al. 2012; Ikonmovska et al. 2011; Appice and Džeroski 2007) and rule based methods for MTR by Aho et al. (2012). Due to the plethora of existing methods, we will not discuss all of them here but rather briefly describe the ones most closely related to our work.

Predictive clustering trees (PCTs) have been introduced by Blockeel et al. (1998). A PCT is a generalization of a standard decision tree, which can be instantiated to support different tasks of SOP, one of which is the task of MTR. PCTs are decision tree based models that belong to the algorithm adaptation methods group, because they handle the task without transforming the instance space. PCTs are global models, since they give predictions for all targets simultaneously. PCTs are instantiated by two required parameters: the variance and prototype functions. Technically, PCTs perform divisive hierarchical clustering by using the provided variance function. The variance function is used to calculate heuristic scores that guide the learning process until a stopping criterion is met. This eliminates the need for arbitrarily selecting the number of clusters beforehand, as required by traditional clustering methods. When instances are clustered, the prototype function is used to calculate the predictions on all leaf nodes (terminal clusters of the hierarchy). A detailed explanation of PCTs can be found in Sects. 3.1 and 3.2.

Ensembles of PCTs were introduced by Kocev et al. (2007, 2013), specifically bagging and random forests. Kocev and Ceci (2015) later extended extremely randomized trees, initially introduced by Geurts et al. (2006), to structured outputs. They called them extra-PCTs, because they based the implementation on PCTs. Extremely randomized trees select one random split point for each of  $k$  predictive attributes at each split. The best performing split point is selected and the process recursively continues. Extremely randomized trees and their multi-output variant (extra-PCTs) are very unstable models, so it only makes sense to use them in an ensemble setting.

Several multi-target prediction approaches that transform the output space do exist, but they mainly focus on the task of MLC. Joly et al. (2014) reduce the dimensionality of the output space by making random projections of it. They make the projections in such a way, that they preserve the original distances in the projected space. Their approach uses Johnson-Lindenstrauss lemma. If the output space projection matrix satisfies the lemma, the variance

computations in the projected space will be  $\epsilon$ -approximations of the variance in the original output space. They employ Gaussian, Rademacher, Hadamard and Achlioptas projections to compress the output space. Only the variance calculations are made in the projected space while the predictions are made directly in the original output space (i.e., no decoding needed). They use multi-output regression trees to calculate the variances in the projected space and then apply thresholding to obtain predictions for labels (i.e., MLC setting). Our approach is simpler, as it does not perform a transformation of the output space but only takes a subset of it, which makes it more straightforward. They do not report any results for the MTR setting. Joly (2017) proposes a gradient boosting method for MTR that uses random projections of the output space to automatically adapt to the output correlation structure.

Tsoumakas and Vlahavas (2007) propose an ensemble method RAKEL (Random  $k$ -labelsets) for the task of MLC, which is also transformation-based. RAKEL is an ensemble-like wrapper method for solving multi-label classification tasks with existing algorithms for multi-class classification. They construct the ensemble by providing a small random subset of  $k$  labels (organized as a label powerset) to each base model, learned by a multi-class classifier. This results in an additional step in the prediction phase because predictions need to be decoded. In addition to this, RAKEL's computational complexity is high because the generated output spaces are label powersets and the underlying classification algorithm is a parameter, which can considerably change/worsen the training times (e.g., if we use SVMs instead of ordinary decision trees). This approach has been extended by Szymański et al. (2016), where the authors propose not to use the original random partitioning of subsets as performed by RAKEL, but rather a data-driven approach. They propose to use community detection approaches from social networks to partition the label space, which can find better subspaces than random search.

Madjarov et al. (2016) also use a data driven approach to solve the task of MLC. They use label hierarchies which they obtain from hierarchical clustering of flat label sets by using annotations that appear in the training data. Finally, the work of Tsoumakas et al. (2014) considers the MTR task. They use random linear target combinations to enrich the output space by constructing many new target variables. They use a predefined number of original target variables in each random combination and then transform the original output space matrix by multiplying it with the coefficient matrix consisting of new combinations.

### 3 Ensembles for multi-target regression with random output selections

The following section introduces the ROS ensemble extension method. We consider the proposed approach as a global method that belongs to the algorithm adaptation group of methods. Although ROS uses subsets of the output space during learning, the learned ensemble provides predictions for all target variables simultaneously. We first describe the predictive clustering paradigm and then explain the process of learning a single predictive clustering tree (PCT). Next, we present the proposed method for learning ROS tree ensembles. Finally, we provide a computational complexity analysis of the proposed approach.

#### 3.1 Predictive clustering

The predictive clustering (PC) framework has been introduced by Blockeel (1998). It can be seen as a generalization of supervised and unsupervised learning. The two learning approaches are traditionally considered as two separate machine learning tasks. However,

there are supervised methods (e.g., decision trees and rules) that partition the instance space into subsets, which makes it possible, to interpret them as clustering approaches. Unsupervised learning groups/clusters examples that are similar according to some distance measure. In supervised learning, the primary goal is to make predictions. The PC framework combines these two approaches.

The PC framework is implemented in the context of decision trees. From the PC point of view, each decision tree is a hierarchy of clusters. The root node of the tree holds all the examples. When traversing the tree (from the root to a leaf), each intermediate node contains less examples than its parent nodes. The connections between nodes represent available paths that each example can take. The decision which path a new example should take is made at the time of traversal and is based on the example's values in the tuple of predictive variables. The bottom-most nodes of the decision tree are called leaf nodes and hold examples most similar to each other. The examples in a leaf are used for calculating the prediction of the leaf.

A decision tree within the PC framework is called a predictive clustering tree (PCT). A PCT is predictive as it is able to make predictions. A PCT is a clustering, i.e., a hierarchy of clusters, represented by the tree's structure. Each node in the tree represents a cluster, which can be explained/described by the conditions/tests that appear in the tree. Each node holds a test and if we combine all the tests from the root node to the selected node, we get the description of the cluster at the selected node. Several different predictive clustering methods (Blockeel and Struyf 2002; Struyf and Džeroski 2006; Kocev 2011; Ženko 2007; Vens et al. 2008; Slavkov et al. 2010) are already implemented in the CLUS software package and are available at <http://sourceforge.net/projects/clus/>.

### 3.2 Learning a single PCT

The induction of a PCT is similar to the induction of a standard decision tree and follows the TDIDT (top down induction of decision tree) algorithm. Algorithm 1 shows the pseudo code for **PCT induction**. Considering the MTR task in the context of ROS tree ensembles, the PCT induction algorithm takes three inputs:

(i) a dataset  $S$ , (ii) a function  $\delta_c(X)$  that randomly samples  $c$  predictive variables from dataset  $X$  without replacement and (iii) a set of target variables  $R_t$ , that the learning process should consider.

---

#### Algorithm 1 TDI of a PCT

---

**Func**  $PCT(S, \delta_c, R_t)$   
**Out:** A predictive clustering tree  
1:  $R_d \leftarrow \delta_c(S)$   
2:  $(t^*, h^*, \mathcal{P}^*) \leftarrow \text{BestTest}(S, R_d, R_t)$   
3: **if**  $t^* \neq \text{none}$  **then**  
4:   **for each**  $S_i \in \mathcal{P}^*$  **do**  
5:      $tree_i \leftarrow PCT(S_i, \delta_c, R_t)$   
6:   **end for**  
7:   **return**  $\text{node}(t^*, \bigcup_i \{tree_i\})$   
8: **else**  
9:   **return**  $\text{leaf}(\text{Prototype}(S))$   
10: **end if**

---



---

#### Algorithm 2 Best test

---

**Func**  $\text{BestTest}(S, R_d, R_t)$   
**Out:** The selected test  $t^*$   
**Out:** The heuristic score  $h^*$  of test  $t^*$   
**Out:** The partitioning  $\mathcal{P}^*$  induced by  $t^*$  on  $S$   
1:  $(t^*, h^*, \mathcal{P}^*) \leftarrow (\text{none}, 0, \emptyset)$   
2:  $S_R \leftarrow \Pi(S, R_d, R_t)$   
3: **for each** possible test  $t$  in  $S_R$  **do**  
4:    $\mathcal{P} \leftarrow$  partitioning induced by  $t$  on  $S_R$   
5:    $h \leftarrow \text{Var}(S_R) - \sum_{S_i \in \mathcal{P}} \frac{|S_i|}{|S_R|} \text{Var}(S_i)$   
6:   **if**  $(h > h^*)$  **then**  
7:      $(t^*, h^*, \mathcal{P}^*) \leftarrow (t, h, \mathcal{P})$   
8:   **end if**  
9: **end for**  
10: **return**  $(t^*, h^*, \mathcal{P}^*)$

---

**Algorithm 3** Best test (extra-PCTs)

---

**Func** *BestTest*( $S, R_d, R_t$ )  
**Out:** The selected test  $t^*$   
**Out:** The heuristic score  $h^*$  of test  $t^*$   
**Out:** The partitioning  $\mathcal{P}^*$  induced by  $t^*$  on  $S$

- 1:  $(t^*, h^*, \mathcal{P}^*) \leftarrow (none, 0, \emptyset)$
- 2:  $S_R \leftarrow \Pi(S, R_d, R_t)$
- 3: **for each** attribute  $a$  in  $R_d$  **do**
- 4:    $t \leftarrow \text{SelectRandomTest}(a, S_R)$
- 5:    $\mathcal{P} \leftarrow$  partitioning induced by  $t$  on  $S_R$
- 6:    $h \leftarrow \text{Var}(S_R) - \sum_{S_i \in \mathcal{P}} \frac{|S_i|}{|S_R|} \text{Var}(S_i)$
- 7:   **if**  $(h > h^*)$  **then**
- 8:      $(t^*, h^*, \mathcal{P}^*) \leftarrow (t, h, \mathcal{P})$
- 9:   **end if**
- 10: **end for**
- 11: **return**  $(t^*, h^*, \mathcal{P}^*)$

---

**Algorithm 4** Random test selection

---

**Func** *SelectRandomTest*( $a, S_R$ )  
**In:** Attribute  $a$   
**In:** Dataset  $S_R$   
**Out:** A test  $t$

- 1:  $t \leftarrow none$
- 2:  $A_v \leftarrow \text{getAttributeValues}(a, S_R)$
- 3: **if**  $a$  is numerical **then**
- 4:    $a_M \leftarrow \text{getMaxValue}(A_v)$
- 5:    $a_m \leftarrow \text{getMinValue}(A_v)$
- 6:    $a_c \leftarrow \text{rndCutPoint}(a_m, a_M)$
- 7:    $t \leftarrow \text{createTest}(a < a_c)$
- 8: **end if**
- 9: **if**  $a$  is categorical **then**
- 10:    $A_s \leftarrow \text{rndNonEmptySet}(A_v, S_R)$
- 11:    $t \leftarrow \text{createTest}(a \in A_s)$
- 12: **end if**
- 13: **return**  $t$

---

The typical PCT considers all predictive and target attributes and is induced by selecting  $\delta_c(X) = \delta_{|D|}(X) = D$  and  $R_t = T$ , where  $D$  and  $T$  are sets of predictive and target variables respectively. The  $\delta_c$  and  $R_t$  parameters are needed when inducing PCTs in the scope of ensemble learning with ROS, which we describe in detail in Sect. 3.3.

There are many ways (i.e., heuristics) to select the **best possible split** in a decision tree. PCT uses the reduction of variance as a measure for the quality of a split. The heuristic function, intuitively, guides the data partitioning in such a way, that the homogeneity of the clusters, from the root to the leaf nodes, increases and the resulting tree model contains the most similar examples in the smallest clusters (leaf nodes of the tree). The reduction in cluster variance is a direct result of partitioning  $\mathcal{P}$ , according to test  $t$  (see line 5 of Algorithm 2). A part of the proposed extension ROS is visible in line 2 of the same algorithm.  $\Pi(S, R_d, R_t)$  is a projection function that reduces the original dataset  $S$  by only considering predictive attributes from the set  $R_d$  and target variables from the set  $R_t$ .

$$\text{Var}(S) = \sum_{i=1}^{|T|} \text{Var}(Y_i) \quad (1)$$

The reduction of variance calculation is instantiated based on the type of machine learning task addressed. In this paper, we focus on multi-target regression and the variance (Eq. 1) is calculated as the sum of the variances of the individual target variables.  $Y_i$  is the vector of continuous values of the  $i$ th target variable in the set of examples  $S$ . The variance is calculated using the standard deviation of the values in each vector  $Y_i$ . The variances of individual target attributes are normalized in order to have them on the same scale. When different target attributes span different scales that are not in the same range, the effect of variance of one variable could be much greater than the effect of another variable with a smaller range. Normalization is needed so that each target attribute contributes equally to the heuristic score.

Regular PCTs make predictions based on the examples in the leaf nodes. Specifically, the *Prototype* function (line 9 of the Algorithm 1) calculates the arithmetic mean of every target variable over all the examples in a given node. If needed, the prototype calculation function can be easily adapted to better address a specific task. The *BestTest* function only

calculates the heuristic value on  $S_R$  (the reduced subset of  $S$ ). This, however, does not restrict the *Prototype* function, which can make predictions for all output variables, even if some of them did not contribute to the calculation of the heuristic value  $h^*$ . We will discuss this further later in Sect. 3.3.3.

In addition to regular PCTs, we also consider extra-PCTs (Kocev and Ceci 2015). These PCTs are induced exactly the same as described in Algorithm 1. However, the extra-PCT finds the split points in a different manner (see Algorithms 3 and 4). The split point is randomly selected for each considered predictive attribute. The evaluation of splits with random split points is performed using the same procedure as for regular PCTs.

### 3.3 Ensembles with ROS

An ensemble is a set of models, called base predictive models. Ensemble models are not considered interpretable, but they generally achieve better predictive performance than individual models, which is usually the reason for using them. The downside of using ensembles is their computational complexity: The cost of learning and using an ensemble model is the sum of the corresponding costs for all of its base models. Predictions for new examples are made by querying base models and combining their predictions.

#### 3.3.1 Generating output space partitions

The proposed ensemble approach introduces randomization in the output space. Whereas regular PCTs simultaneously consider the whole target space in the heuristic used for tree construction, ROS considers a different random subset of it for each base model in the ensemble. Each base model is consequently learned by considering only those targets that are included in the randomly generated partition provided to it (see the call of function  $\Pi$  in Algorithm 2, line 2).

ROS creates the output space partitions (subspaces) in advance, i.e., the partitions are independent of the algorithm for learning a single model. ROS generates a different subspace for every ensemble constituent. Thus, the number of generated subspaces (base models) equals  $b$ . Algorithm 5 constructs the subspaces. The algorithm has the following parameters: (i) number of subspaces  $b$  to generate, (ii) function  $\theta(X, v)$  that uniformly at random samples without replacement of subset from the set  $X$  and (iii) set/space of target attributes  $T$ , from which subspaces are created.

---

#### Algorithm 5 Generating subspaces

---

**Func** GenSubspaces( $b, T, \theta, v$ )

**Out:** list of target subspaces

```

1:  $G \leftarrow \text{emptyList}()$ 
2:  $G_1 \leftarrow T$ 
3: for  $i \in \{2, \dots, b\}$  do
4:    $G_i \leftarrow \theta(T, v)$ 
5: end for
6: return  $G$ 
```

---

In the first step, we create an empty list that will contain all the subspaces, i.e.,  $G = [G_1, G_2, \dots, G_b]$ . The first generated subspace is  $T$  and includes all target variables, i.e.,

the corresponding PCT considers all target attributes. This is needed to ensure that all targets are being considered at least once. We generate the remaining  $b - 1$  subspaces with the  $\theta$  function, which has a parameter  $v$ . An example of a  $\theta$  function could return a random selection of 25% ( $v = \frac{1}{4}$ ) of items in the set provided as input. If one defines  $\theta(X) = X$ , then all ensemble constituents will always consider all targets, which is what a regular ensemble of PCTs does. This function is a parameter of our overall ensemble learning algorithm and we investigate its influence in Sect. 5.

Algorithm 6 describes the random sampling  $\theta$  function used in our experiments. The function  $\theta(X, v)$  uniformly at random samples  $\lceil v \cdot |X| \rceil$  items from the set  $X$ , where  $v$  represents the percentage of  $X$  we want to sample. This algorithm always samples a fixed number of attributes.

---

**Algorithm 6** Sample  $v$ -% of target variables
 

---

**Func**  $\theta(X, v)$   
**In:** A set of variables  $X$   
**In:** Sample percentage  $v \in (0.0, 1.0)$   
**Out:** A subset of variables  $Q$  ( $Q \subseteq X$ )

```

1:  $Q \leftarrow \emptyset$ 
2: while  $|Q| < \lceil v \cdot |X| \rceil$  do
3:    $a \leftarrow \text{randItemFromSet}(X \setminus Q)$ 
4:    $Q \leftarrow Q \cup \{a\}$ 
5: end while
6: return  $Q$ 

```

---

### 3.3.2 Building the ensembles

With all preliminaries laid out, we can now describe our overall process for learning an ensemble of PCTs for MTR. We use three ensemble building methods (bagging, random forests and extremely randomized trees) that have been extended to support multi-target outputs and use PCTs (Bag and RF) and extra-PCTs (ET) as base models. Algorithm 7 is generic and shows how the ensembles are built. We use the values in Table 1 to describe its custom initialization for the ensembles for multi-target regression with random output selections. All methods we consider use the same input parameters: (i)  $S$  is the dataset, (ii) the  $\gamma(X)$  function samples the dataset  $X$ , (iii) the  $\delta_X(D)$  function randomly selects  $X$  predictive attributes, considered at each node during learning and (iv)  $G$  is the list of subspaces generated by the *GenSubspaces* function and  $|G|$  is the number of ensemble constituents.

**Bagging** (Breiman 1996) is short for bootstrap aggregating. It is an ensemble method that uses bootstrap replication of the training data to introduce randomization in the learning dataset. Such perturbations of the learning set have proven useful for unstable base models, such as decision trees, but can generally be used by any model type. A bootstrap replicate  $S^*$  of a dataset  $S$ , is again a dataset, that has been randomly sampled from  $S$ . Sampling with replacement is repeated until both datasets are of equal size (i.e.,  $|S| = |S^*|$ ).

**Random forests** (Breiman 2001) work in a similar fashion to bagging. This ensemble method also starts with bootstrap replicates, that introduce randomization in the instance space. However, it additionally introduces randomization in the predictive attribute space by randomizing

**Algorithm 7** Building the ensemble

---

**Func** BuildEnsemble( $S, \gamma, \delta_c, G$ )  
**In:** Dataset  $S$   
**In:**  $\gamma$  function for sampling data instances  
**In:**  $\delta_c$  function for sampling predictive variables  
**In:** List of generated subspaces  $G$   
**Out:** Ensemble

- 1:  $F \leftarrow \emptyset$
- 2: **for**  $i \in \{1, \dots, |G|\}$  **do**
- 3:    $S_i \leftarrow \gamma(S)$
- 4:    $pct_i \leftarrow PCT(S_i, \delta_c, G_i)$
- 5:    $F \leftarrow F \cup \{pct_i\}$
- 6: **end for**
- 7: **return**  $F$

---

**Table 1** Ensemble building parameters

| Ensemble method | $\delta_c(D)$                                | $\gamma(X)$              | <i>BestTest</i> |
|-----------------|--|--------------------------|-----------------|
| Bag of PCTs     | $D$  | <i>bootstrap</i> ( $X$ ) | Algorithm 2     |
| RF of PCTs      | $\theta\left(D, \frac{1}{\sqrt{ D }}\right)$ | <i>bootstrap</i> ( $X$ ) | Algorithm 2     |
| Extra-PCTs      | $D$  | $X$                      | Algorithm 3     |

the algorithm for the base predictive models. The  $\delta_c$  parameter of the PCT induction algorithm (see Algorithm 1) is instantiated as shown in Table 1. This causes the random forest ensemble method to only consider a subset of randomly selected predictive attributes from the set  $D$  of all predictive attributes, while searching for the best split for a node. This process of random selection of predictive attributes is then repeated afresh at each node, yielding different subsets of predictive attributes. The function  $\delta_c(D)$  can be defined to return any number of items from the set  $D$  between 1 and  $|D|$ , but the recommended setting by Breiman (2001) is  $\sqrt{|D|}$ , which is what we use.

**Extremely randomized trees** (Geurts et al. 2006) are very unstable decision trees. It therefore only makes sense to use them in the ensemble setting. It has two distinctive properties with respect to the other two methods: (i) the dataset is not perturbed by applying bootstrapping and (ii) the *BestSplit* method used by extra-PCTs is shown in Algorithm 3. Extra trees select at random  $k$  predictive attributes and for each of them, randomly select a split point.<sup>1</sup> Each split is then evaluated and the one with the best heuristic value  $h^*$  is selected. Algorithm 4 shows how the random split points are determined. The recommended number (Geurts et al. 2006) of predictive attributes to be considered at every split is  $|D|$ , which is reflected in our ensemble initialization for extra trees (see Table 1).

### 3.3.3 Making predictions

An ensemble makes predictions by combining the predictions of its base models. Each base predictive model gives its predictions to the aggregation function, which takes all the votes

<sup>1</sup> This is in contrast to the normal procedure of considering multiple/all possible split points for each attribute, before selecting the best one.

and decides on the final prediction of the ensemble. In general, the aggregation function is a parameter and there are many ways to combine the votes of the base predictive models: averaging the predictions, majority vote, introducing weights for individual models, introducing preferences based on domain knowledge, and so on. In this paper, we propose two different aggregation functions used in conjunction with the proposed method: (i) total averaging and (ii) subspace averaging.

**Total averaging** takes all the predictions of the base models and averages them. Each base model gives predictions for all of the  $|T|$  targets. We calculate the average as the arithmetic mean: Final predictions for the  $i$ th target attribute ( $\hat{y}_i$ ) are computed as  $\hat{y}_i = \frac{1}{b} \sum_{j=1}^b y_i^j$ , where  $y_i^j$  represents the prediction of  $j$ th base model for the  $i$ -th target attribute.

**Subspace averaging** considers only the predictions made by the base predictive models for the targets used to learn them. In other words, the prediction for a given target is averaged over only those base models, for which that target was considered in the heuristic during learning (see the input parameter  $R_t$  in Algorithm 2). The final predictions for the  $i$ th target attribute are computed as

$$\hat{y}_i = \frac{\sum_{j=1}^b \mathbb{1}(a_i \in G_j) \cdot y_i^j}{\sum_{j=1}^b \mathbb{1}(a_i \in G_j)} \quad (2)$$

where  $\mathbb{1}(X)$  is an indicator function, which returns 1, if the argument  $X$  is true and 0 otherwise;  $G_j$  (see Sect. 3.3.1) is the subspace of target attributes, considered when learning the  $j$ th ensemble constituent, and  $a_i$  denotes the  $i$ th target. The denominator is the number of ensemble constituents for which  $a_i$  was considered during learning and is non-zero because every target attribute is considered by at least subspace  $G_1$ .

### 3.4 Computational complexity analysis

From the work of Kocev et al. (2013) and the assumption that the decision tree is balanced and bushy (Witten and Frank 2005), it follows that the computational complexity of learning a **single multi-target PCT** is  $\mathcal{O}(dN \log^2 N) + \mathcal{O}(dtN \log N) + \mathcal{O}(N \log N)$ , where  $N$  is the number of instances,  $d$  is the number of predictive attributes and  $t$  is the number of target attributes in the dataset. Similarly, from the work of Kocev and Ceci (2015) and with the same assumption, it follows that the computational complexity of learning a **single multi-target extra-PCT** is  $\mathcal{O}(ktN \log N) + \mathcal{O}(N \log N)$ , where  $k$  is the number of randomly sampled predictive attributes at each split. In general, learning an **ensemble** of  $b$  base models has the complexity of learning all of its constituents. In our case, that amounts to  $b(\mathcal{O}(dN \log^2 N) + \mathcal{O}(dtN \log N) + \mathcal{O}(N \log N))$  for bagging and random forests of PCTs and  $b(\mathcal{O}(ktN \log N) + \mathcal{O}(N \log N))$  for random forests of extra-PCTs.

The computational complexity also depends on the use **bootstrapping** and the amount of predictive and/or target attributes considered for each base model. Computational cost of bootstrapping is  $\mathcal{O}(N)$  and the number of instances considered in that case equals  $N' = 0.632 \cdot N$  (Breiman 1996). Bootstrapping is not used for learning extra-PCTs.

Taking into account the fact that random forests also **sample the input space** (through the sampling function  $\delta_c(D)$ ), the number of predictive variables actually considered by the base models is  $d' = c$  (see the definition of  $\delta_c$  in Sect. 3.2). The sampling of predictive variables happens at every node split, so the complexity of data subsampling is  $\mathcal{O}(d' \log N')$ .

ROS uses additional **sampling of the target space**. The function  $\theta(X, v)$  (see Algorithm 6) is used to sample from the target space. The sampled subsets are always of equal cardinality, which is controlled by the parameter  $v \in (0.0, 1.0)$ . However, the first subset always includes all target attributes (see line 2 in Algorithm 5). We define the variable  $t'$  as the average target subspace cardinality considered by the base models as:  $t' = \frac{1}{b}((b-1) \cdot \lceil v \cdot t \rceil + t)$ . The complexity of the sampling function  $\theta(X, v)$  (see Algorithm 6) is low. All operations of the sampling algorithm have complexity of  $\mathcal{O}(1)$ . The while loop in line 2 is executed  $\lceil v \cdot |X| \rceil$ -times. Each randomly sampled attribute  $a$  has equal probability of being included in the resulting set  $Q$ . Thus, the complexity of the sampling algorithm, which samples  $b-1$  times, is linear and proportionate to  $(b-1) \cdot \mathcal{O}(v \cdot t)$ . Considering all of the above, the complexity of the ROS ensembles is

$$\begin{aligned} \text{Bag-ROS or RF-ROS} &= b \cdot [\mathcal{O}(d'N' \log^2 N') + \mathcal{O}(d't'N' \log N') \\ &\quad + \mathcal{O}(N' \log N') + \mathcal{O}(d' \log N') + \mathcal{O}(N)] \\ &\quad + (b-1) \cdot \mathcal{O}(v \cdot t) \end{aligned} \quad (3)$$

$$\begin{aligned} \text{ET-ROS} &= b \cdot [\mathcal{O}(d't'N \log N) + \mathcal{O}(N \log N) \\ &\quad + \mathcal{O}(d' \log N')] + (b-1) \cdot \mathcal{O}(v \cdot t) \end{aligned} \quad (4)$$

The ratio between the full output space size and the one considered by ROS is constant and is proportionate to  $\frac{t'}{t} = \frac{(b-1) \cdot v \cdot t + t}{b \cdot t} = \lim_{b \rightarrow \infty} \frac{(b-1) \cdot v + 1}{b} = v$ . The overall complexity of ROS is consequently reduced in the parts that correspond to the selection of the best split. We expect a linear decrease in complexity in those terms. Otherwise, the overall complexity is still as described by Kocev et al. (2013), Kocev and Ceci (2015).

Ensembles usually contain many base models which results in longer times to make a prediction. Therefore, we also address the complexity of **making predictions**. Under the previously mentioned assumption about decision trees being balanced and bushy, the average depth of a decision tree is actually the average length of the path that has to be traversed by an instance in order to get to the prediction. The complexity of making a prediction with a single-target decision tree is therefore  $\mathcal{O}(\log(N))$ . In a global MTR scenario, all target variables are predicted simultaneously with the same complexity as that of making a prediction with a single-target tree. When we switch to the ensemble setting, the complexity increases linearly with the number of base models in the ensemble:  $b \cdot \mathcal{O}(\log(N))$ . If we are approaching the problem of MTR locally, each target is predicted with its own ensemble and that additionally increases the complexity in proportion to the number of target variables:  $bT \cdot \mathcal{O}(\log(N))$ .

## 4 Experimental design

To evaluate the performance of the ROS ensembles for MTR, we performed extensive experiments on benchmark datasets. This section presents: (i) the experimental questions addressed, (ii) the evaluation measures used, (iii) the benchmark datasets and (iv) the experimental setup (including the parameter instantiations for the methods used in the experiments).

### 4.1 Experimental questions

In our experiments, we construct PCT ensembles for MTR by using the described ensemble extension method ROS. In order to better understand the effects of ROS, we investigate the resulting ensemble models across three dimensions.

First, we are interested in the **convergence** of their predictive performance as we increase the number of PCTs in the ensemble. We want to establish the number of base models needed in an ensemble to reach the point of performance saturation. We consider an ensemble saturated, when adding additional base models to it would not bring statistically significant improvement in terms of predictive power.

Next, we are interested, whether the proposed extension can improve the **predictive performance** over the performance of the original ensembles. Learning on subsets of targets could exploit additional structural relations that may be overlooked by the original ensemble approaches.

Finally, as we have theoretically derived in Sect. 3.4, we expect that the dimensionality reduction of the output space will yield improvements in terms of **computational efficiency**. Specifically, we are interested in the running times of the ROS ensemble approaches and the sizes of the resulting models.

The specific experimental questions we pose relate to the above three dimensions we are interested in. The experiments and their evaluation have been designed with the following research questions in mind:

1. How many base models do we need in ROS ensembles in order to reach the point of performance saturation?
2. What is the best value for the portion of target space to be used within such ensembles? Is this portion equal for all evaluated ensemble methods?
3. Does it make sense to change the default aggregation function of the ensemble that uses the prediction for all targets? Can this improve predictive performance?
4. Considering predictive performance, how do ROS ensemble methods compare to the original ensemble methods?
5. Is ROS helpful in terms of time efficiency?
6. Do ROS models use less memory than the models trained with the original ensemble methods?
7. How ROS models compare to other output transformation methods?

## 4.2 Evaluation measures

In order to understand the effects that ROS has on the learning process, we first need to evaluate the models induced by the ROS ensemble approaches. In machine learning, empirical evaluation is most commonly used to achieve this goal, that assesses the performance of a given model in terms of evaluation measures. Below we describe the measures that we use for assessing predictive power, time and space complexity.

The **predictive performance** of a MTR model is assessed by using the average relative root mean squared error (aRRMSE), which averages the relative root mean squared errors (RRMSE) for the individual target variables. RRMSE is a relative measure calculated against the baseline model that predicts the arithmetic mean of all values of a given target in the learning set. Specifically, the value  $\bar{y}_i$  in Eq. 5 is the prediction of the baseline model for the  $i$ th target variable, while the value  $\hat{y}_i^{(e)}$  represents the predicted value for the  $i$ th target variable of the example  $e$ .

$$aRRMSE = \frac{1}{t} \sum_{i=1}^t RRMSE_i = \frac{1}{t} \sum_{i=1}^t \sqrt{\frac{\sum_{e=1}^{N_{test}} (y_i^{(e)} - \hat{y}_i^{(e)})^2}{\sum_{e=1}^{N_{test}} (y_i^{(e)} - \bar{y}_i)^2}} \quad (5)$$

**Table 2** Properties of the considered MTR datasets with multiple continuous targets: number of examples ( $N$ ), number of predictive attributes (discrete/continuous,  $d/c$ ), and number of target attributes ( $t$ )

| No. | Name of dataset  | $N$    | $d/c$ | $t$ |
|-----|--|--------|-------|-----|
| 1   | Forestry-Kras (Džeroski et al. 2006)                       | 60,607 | 0/160 | 11  |
| 2   | Vegetation clustering (Gjorgjioski et al. 2008)            | 29,679 | 0/65  | 11  |
| 3   | Vegetation condition (Kocev et al. 2009)                   | 16,967 | 1/39  | 7   |
| 4   | Water quality (Blockeel et al. 1999; Džeroski et al. 2000) | 106    | 0/16  | 14  |
| 5   | ATP 1D (Spyromitros-Xioufis et al. 2016)                   | 337    | 0/441 | 6   |
| 6   | ATP 7D (Spyromitros-Xioufis et al. 2016)                   | 296    | 0/441 | 6   |
| 7   | RF1 (Spyromitros-Xioufis et al. 2016)                      | 9125   | 0/64  | 8   |
| 8   | RF2 (Spyromitros-Xioufis et al. 2016)                      | 9125   | 0/576 | 8   |
| 9   | Sales (Kaggle 2008; Spyromitros-Xioufis et al. 2016)       | 639    | 0/401 | 12  |
| 10  | SCM 1D (Spyromitros-Xioufis et al. 2016)                   | 9893   | 0/280 | 16  |
| 11  | SCM 20D (Spyromitros-Xioufis et al. 2016)                  | 8966   | 0/61  | 16  |
| 12  | OES 10 (Spyromitros-Xioufis et al. 2016)                   | 403    | 0/298 | 16  |
| 13  | OES 97 (Spyromitros-Xioufis et al. 2016)                   | 334    | 0/263 | 16  |
| 14  | Soil resilience (Debeljak et al. 2009)                     | 26     | 1/7   | 8   |
| 15  | Prespa diatoms lake top 10 (Kocev et al. 2010)             | 218    | 0/16  | 10  |
| 16  | PPMI (Marek et al. 2011)                                   | 713    | 0/148 | 35  |
| 17  | ADNI (Gamberger et al. 2016)                               | 659    | 0/232 | 14  |

We also monitor how much our models **overfit** the training data by calculating their relative decrease of performance on the testing data with respect to that on the training data. Smaller values mean less overfitting, with zero being the ideal score. We calculate the overfitting score with Eq. 6.

$$OS = \frac{aRRMSE_{test} - aRRMSE_{train}}{aRRMSE_{train}} \quad (6)$$

The **efficiency** is measured in terms of execution times and sizes of the induced models. Time efficiency is measured with the CPU time needed to induce (train) the model (i.e., learning time). For ROS ensembles, this includes the target space decomposition. We also measure the average time needed to make a prediction (i.e., prediction time). Space efficiency is measured with the total number of nodes in the tree model (intermediate and leaf nodes): the smaller the better. Induction times and model sizes are summed over all ensemble constituents.

### 4.3 Data description

To evaluate the proposed method, we use 17 benchmark datasets that contain multiple continuous target attributes and are mainly from the domain of ecological modeling. Table 2 shows the main characteristics of the considered datasets. In order to have as general evaluation as possible, we use datasets of different sizes in terms of number of instances, number of predictive and number of target attributes.

**Table 3** Parameter values used to build ensembles with ROS

| Location of use                         | Parameter          | Used values   |
|---|--------------------|---|
| $BuildEnsemble(S, \gamma, \delta_c, G)$ | $\gamma, \delta_c$ | See Table 1   |
| $GenSubspaces(b, T, \theta)$            | $b$                | 10, 25, 50, 75<br>100, 150, 250                               |
| $\theta(X, v)$                          | $v$                | $\frac{1}{4}, \frac{1}{2}, \frac{3}{4}, \frac{1}{\sqrt{ T }}$ |
| Making predictions                      | Averaging function | Total, subspace   |

#### 4.4 Experimental setup

We designed the experimental setup according to the experimental questions posed in Sect. 4.1. First, we describe all parameter settings of the ROS ensemble methods. We then outline the procedures for statistical analysis of the results.

We consider three **types of ensembles**: bagging and random forests of PCTs and extra-PCTs. In order for Algorithm 7 to simulate these three ensemble methods, we set its parameters to the values given in Table 1. Following the recommendations from Bauer and Kohavi (1999), the trees in the ensembles are unpruned. Our experimental study considers different **ensemble sizes**, i.e., different numbers of base models (PCTs) in the ensemble, in order to investigate the saturation of ensembles and to select the saturation point.

First, we construct **ensembles without ROS** (Bag, RF, ET) that use the full output space for learning the base predictive models. This means that the list  $G$  contains  $b$  sets, where each set contains all the attributes from the set  $T$  (target attributes), i.e.,  $G = \{T, T, \dots, T\}$ , where  $|G| = b$ .

The second part of our experiments is concerned with the proposed extension—**Random Output Selections**. We start with the parametrization of the  $GenSubspaces$  function (Algorithm 5), which takes as input  $b, T$  and the sampling function  $\theta(X, v)$  (see Algorithm 6). We consider four values for  $v$  in the allowed range (0.0, 1.0), namely  $\frac{1}{\sqrt{|T|}}, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$ . Additionally, we use two **ensemble predictions aggregation functions**: total averaging and subspace averaging. Table 3 summarizes the parameter values considered in our experiments.

The third part of our study focuses on the comparison of our ROS ensemble methods to baseline methods. To that end, we also train multi-target PCTs and ensembles of single-target PCTs on each of the 17 benchmark datasets. F-test pruning is applied to single multi-target PCTs. The F value is selected using internal 3-fold cross-validation. We build one ensemble for each target variable. Ensembles contain 100 base models and are built by using the same parameters as the original ensembles.

We **estimate the predictive performance** of the considered methods by using 10-fold cross-validation. All methods use the same folds. For **statistical evaluation** of the obtained results, we follow the recommendations from Demšar (2006). The Friedman test (Friedman 1940), with the correction by Iman and Davenport (1980), is used to determine statistical significance. In order to detect statistically significant differences, we calculate the critical distances (CD) by applying the Nemenyi (1963) or Dunn (1961) post-hoc statistical tests. Both post-hoc tests compute critical distance between the ranks of considered algorithms. The difference is that Nemenyi post-hoc test compares the relative performance of all considered methods (all vs. all), whereas Bonferroni–Dunn post-hoc test compares the performance of a single method to other methods (one vs. all). The results of these tests are presented with average rank diagrams (Demšar 2006), where methods connected with a line have results

that are not statistically significantly different. All statistical tests were conducted at the significance level  $\alpha = 0.05$ . Statistical tests have been calculated for two variants of the results: per dataset (using aRRMSE value for each dataset) and per target (using the RRMSE values for all targets of all datasets). We used the Bonferroni–Dunn (CD is shown as a dotted blue line) post-hoc test to present results in Sect. 5.4 and Nemenyi post-hoc test otherwise (CD is shown as a solid red line).

The experiments were executed on a heterogeneous computing infrastructure, i.e., the SLING grid, which can affect time-sensitive evaluations. To avoid having incomparable measurements of running times, we run time-sensitive experiments separately by using a single computational node.

## 5 Results and discussion

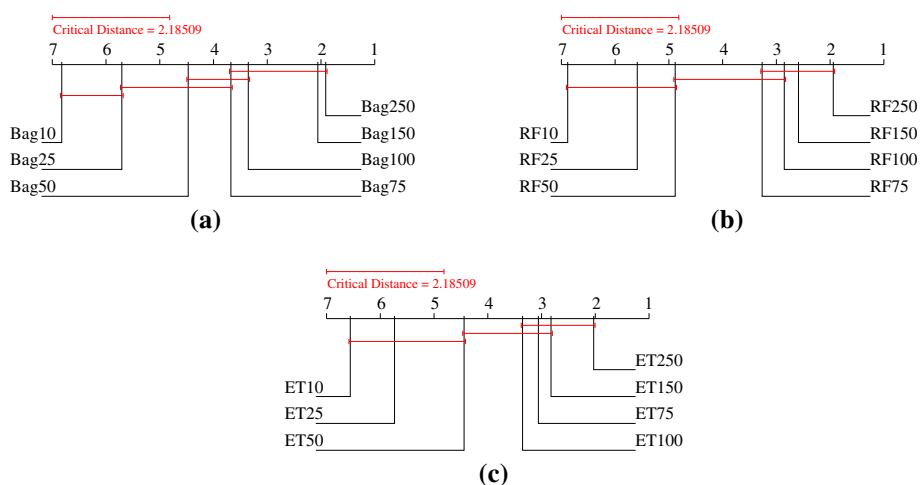
Here we present the results of our comprehensive experimental study. Considering a large number of datasets (17) and several ensemble methods, we present the results in terms of predictive performance (aRRMSE, overfitting score—OS), time complexity (learning and prediction time) and space complexity (model size). In the presentation of time complexity results, we focus on two datasets *Forestry Kras* and *OES 10*, that have relatively large output spaces (11 and 16 targets, respectively). The selected datasets also differ in the number of examples: *Forestry Kras* has many whereas *OES 10* has few. For reference, all other results are available in “Appendix”.

The presentation and discussion of the results follows the experimental questions from Sect. 4.1. First, we examine the convergence of original and ROS ensembles. Next, we focus on selecting the output space size. We experiment with four different output space sizes (see Table 3), that have consequently been used. This parameter is crucial because it introduces additional point of randomization in all three considered ensemble methods. In that sense, ROS can also be seen as a localization process: the constructed base models are tailored to a specific output subspace. We recommend values for this parameter for each ensemble learning method. Furthermore, we show the effects of changing the aggregation functions in our ensembles. Finally, we use the recommended parameters for ROS and provide an overall evaluation, by comparing the extended ensembles to the original ones. We compare the ROS methods to the baseline methods in terms of predictive power, running times and model sizes.

### 5.1 Ensemble convergence

The saturation points of the original ensembles (Bag, RF and ET) are located between 50 and 75 base models (Fig. 1). These findings are in line with the work of Kocev et al. (2013), where Bag and RF saturate with 50 base models and Kocev and Ceci (2015), RF and ET saturate with 100 and 75 base models respectively. We attribute this difference to two factors: (i) we use a different and larger number of datasets and (ii) the number of target attributes per dataset in our study is considerably larger. All in all, we consider the original ensembles with 75 base models saturated.

We next investigate the saturation of the ROS ensembles. A subset of the results for ensembles with 50, 100, 150 and 250 models are reported in Fig. 2 and illustrate the saturation of ROS ensembles for all three considered ensemble methods. Lines on the plots represent different output space sizes. Values in brackets indicate the value for the  $v$  parameter. Left and right sides of the plots depict voting with total averaging and subspace averaging respectively.



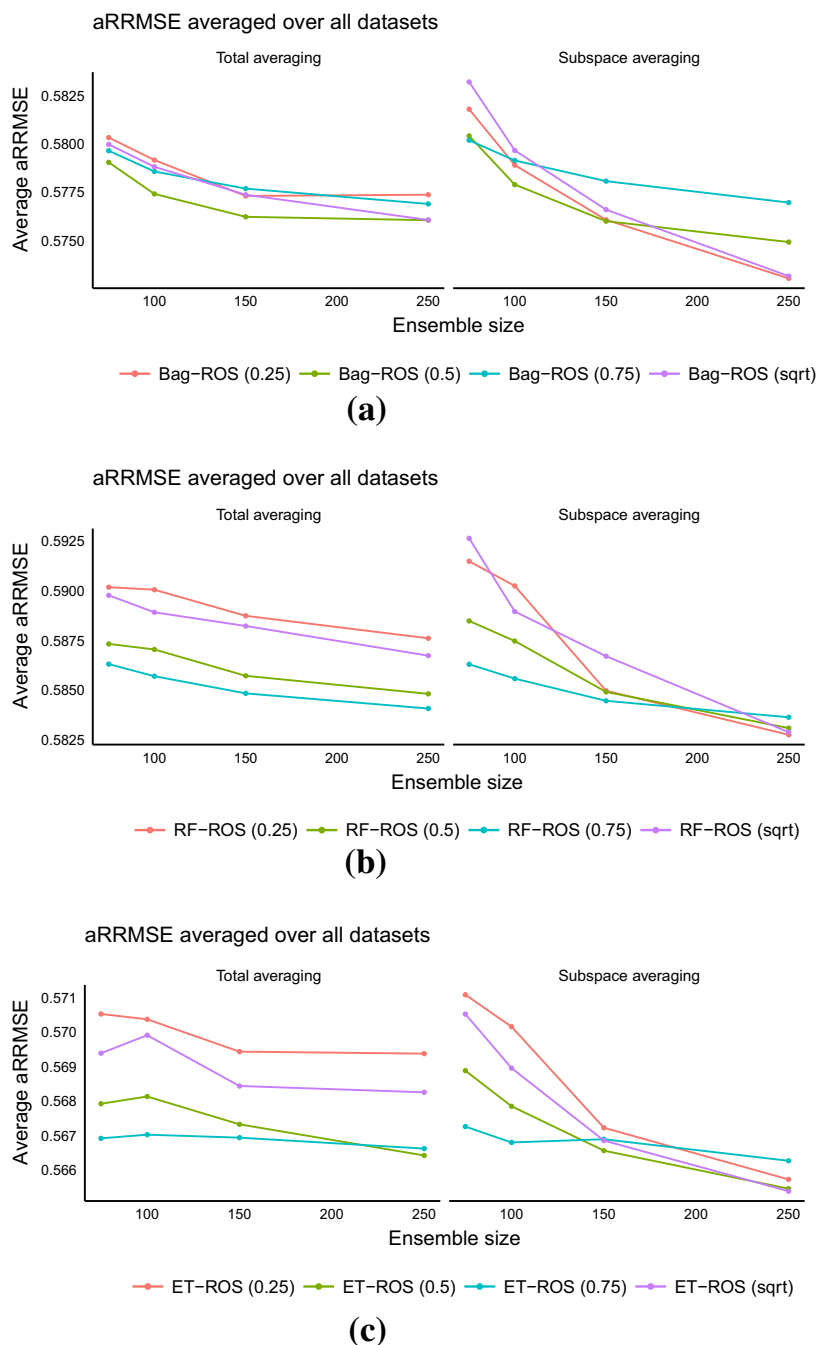
**Fig. 1** Saturation of original ensemble methods. The average rank diagrams compare the performance (aRRMSE) of ensembles with different size. The saturation point is the lowest number of trees in the ensemble for which the performance is not significantly different than the best: this is 75 for Bag and RF and 50 for ET. **a** Saturation of Bag, **b** saturation of RF, **c** saturation of ET

The y axis shows aRRMSE values averaged over all considered datasets. The results show that Bag-ROS and ET-ROS ensembles saturate between 50 and 100 base models while RF-ROS ensembles saturate a bit later, between 75 and 100 base models. Figure 2 suggests that ROS ensembles saturate at a large number of base models when subspace averaging is used to aggregate the predictions of the base models. Performance in terms of aRRMSE and overfitting scores of all discussed ensembles with 100 trees (multi-target and single-target variants) and single multi-target regression trees is presented in “Appendix B”.

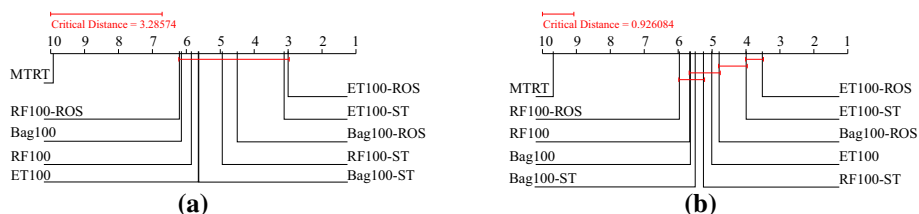
## 5.2 ROS parameter selection

This section describes the selection of the best performing output subspace size and aggregation function. The considered ensemble methods introduce different randomizations in their learning processes, so we cannot assume that ROS has the same influence on all three types of ensemble methods. Figure 2 also suggests that the choice of the aggregation function has a direct effect on performance. We therefore analyze the effect of output subspace size and aggregation function for each ensemble type separately.

We selected candidate values for the ROS parameters based on the curves given in Fig. 2. With both aggregation functions, candidate parameter values for subspace sizes are  $v = \frac{1}{2}$  for Bag-ROS and  $v = \frac{3}{4}$  for RF-ROS and ET-ROS. We selected these values because they exhibit the lowest aRRMSE averaged over all datasets used in this study. The averaged saturation curves in Fig. 2 sometimes intertwine and make it difficult to make this decision. In those cases, we selected the parameter values based on the averaged performance of ensembles with 100 trees. Next, we performed a simple analysis by comparing the wins of the two considered aggregation functions using the candidate output space size. For Bag-ROS, it turned out that total averaging had most wins, whereas subspace averaging was dominant for RF-ROS and ET-ROS. Our final parameter recommendation is therefore to use total averaging with  $v = \frac{1}{2}$  for Bag-ROS and subspace averaging with  $v = \frac{3}{4}$  for RF-ROS and ET-ROS.



**Fig. 2** Saturation of all three ensemble methods extended with ROS. The different (color) lines on the plots represent different output space sizes. The values in the brackets after an ensemble method name indicate the value for the  $v$  parameter. The left and right panels of the plots show results for voting with total averaging and subspace averaging respectively. The y axis shows aRRMSE values averaged over all 17 considered datasets. **a** Bagging, **b** random forests, **c** extra PCTs (Color figure online)



**Fig. 3** Overall average rank diagrams for predictive performance. **a** Per dataset (aRRMSE), **b** per target (RRMSE)

### 5.3 Predictive performance and computational efficiency

Here, we compare the original ensemble methods (Bag, RF, ET) to the ones that use the ROS extension. In addition, ROS is also compared to multi-target PCTs and ensembles of single-target PCTs. We show the relative performance of the different methods by using the average rank diagrams shown in Fig. 3.

Figure 3 depicts two average rank diagrams: one per dataset and one per target. The per dataset diagram is based on aRRMSE value, one per dataset. Both analyses show that ensembles statistically significantly outperform individual multi-target PCTs, i.e., multi-target PCTs perform significantly worse than the ensemble methods. The *per dataset analysis* shows no statistically significant difference in terms of predictive performance among the other methods. We can however note that Bag-ROS and ET-ROS outperform their original counterparts and ensembles of single-target PCTs. RF-ROS performs on par with the original bagging and random forest ensembles, but worse than the other ROS ensembles (Bag-ROS and ET-ROS). The best performing of all methods is ET-ROS.

The *per target analysis* detects two statistically significant differences in performance. First, with the exception of ET-ST, ET-ROS outperforms all other methods with statistical significance. Second, Bag-ROS outperforms RF-ROS, which performs worst of all ensemble methods. All original ensembles (Bag, RF and ET) show no statistically significant differences in performance. All in all, looking at the big picture, ROS ensembles generally perform better than their original counterparts, with the exception of random forests.

In Table 4, we show the predictive performance (aRRMSE) for two highlighted datasets: *Forestry Kras* and *OES 10*. The table contains results for the baseline ensembles (Bag, RF and ET) as well as the extended ROS ensembles, individual multi-target PCTs and ensembles of single-target PCTs. All ensembles contain 100 base models. The ensembles of single-target PCTs contain 100 base models per target.

For the *Forestry Kras* dataset, the proposed ROS methods do not have a notable effect on the predictive performance (aRRMSE) of the three ensemble methods. Similar findings are observed when calculating the overfitting score (OS): the ROS ensembles overfit the training data to the same extent as their original counterparts. Next, multi-target PCTs and ensembles of single-target PCTs have the worst predictive performance. The difference in predictive performance between ensembles of single-target PCTs and other ensembles is minimal. However, notable differences exist in terms of time needed for learning the model and making predictions. Namely, the multi-target ensembles have significantly lower learning and prediction times than the single-target ensembles. The ROS ensembles train the ensembles faster (for Bagging and Extra trees) but still in the same order of magnitude as the original methods. Not surprisingly, single multi-target PCTs have the shortest learning times at the cost of lowest predictive performance. Similar findings are observed when considering model

**Table 4** Performance of ensembles and single trees on two datasets (*Forestry Kras* and *OES 10*) measured in terms of aRRMSE, overfitting score, average learning times, average per-instance prediction time and model complexity (total number of nodes)

| Dataset              | Method           | aRRMSE | OS    | $\overline{LT}$ (s) | $\overline{PT}$ ( $\mu$ s) | Complexity         |
|----------------------|------------------|--------|-------|---------------------|----------------------------|--------------------|
| <i>Forestry Kras</i> | Bag              | 0.55   | 0.476 | 394                 | 272                        | $3.22 \cdot 10^6$  |
|                      | Bag-ROS          | 0.548  | 0.471 | 267                 | 274                        | $3.20 \cdot 10^6$  |
|                      | Bag-ST           | 0.551  | 0.494 | 34,500              | 3960                       | $34.61 \cdot 10^6$ |
|                      | RF               | 0.545  | 0.44  | 34.15               | 259                        | $3.16 \cdot 10^6$  |
|                      | RF-ROS           | 0.546  | 0.44  | 36.87               | 273                        | $3.15 \cdot 10^6$  |
|                      | RF-ST            | 0.546  | 0.457 | 2250                | 2300                       | $17.13 \cdot 10^6$ |
|                      | ET               | 0.557  | 0.575 | 450                 | 264                        | $3.67 \cdot 10^6$  |
|                      | ET-ROS           | 0.557  | 0.579 | 281                 | 274                        | $3.67 \cdot 10^6$  |
|                      | ET-ST            | 0.56   | 0.611 | 73,780              | 3450                       | $39.96 \cdot 10^6$ |
|                      | Multi-target PCT | 0.61   | 0.169 | 76.68               | 2.39                       | $2.59 \cdot 10^3$  |
| <i>OES 10</i>        | Bag              | 0.531  | 1.114 | 19                  | 157                        | $3.15 \cdot 10^4$  |
|                      | Bag-ROS          | 0.527  | 1.052 | 17                  | 159                        | $3.14 \cdot 10^4$  |
|                      | Bag-ST           | 0.487  | 1.556 | 412                 | 3960                       | $21.5 \cdot 10^4$  |
|                      | RF               | 0.517  | 1.093 | 1.59                | 189                        | $3.15 \cdot 10^4$  |
|                      | RF-ROS           | 0.518  | 1.132 | 1.49                | 208                        | $3.16 \cdot 10^4$  |
|                      | RF-ST            | 0.492  | 1.525 | 69                  | 4760                       | $43.51 \cdot 10^4$ |
|                      | ET               | 0.514  | 0.986 | 18.27               | 180                        | $3.48 \cdot 10^4$  |
|                      | ET-ROS           | 0.496  | 1.156 | 18.57               | 201                        | $3.50 \cdot 10^4$  |
|                      | ET-ST            | 0.467  | 2.654 | 480                 | 4410                       | $51.29 \cdot 10^4$ |
|                      | Multi-target PCT | 0.616  | 0.704 | 0.80                | 2.81                       | $0.45 \cdot 10^3$  |

complexity (measured as total number of nodes in all of the trees in an ensemble or in a multi-target PCT). Average prediction times per instance do not differ across the different approaches. This is expected since all base models are trees and no additional computation overhead is needed to calculate the predictions. Ensembles of single-target PCTs always have an order of magnitude higher learning and prediction times, as well as model complexity as a separate ensemble is learned for predicting each target.

For the *OES 10* dataset, improvements in predictive performance are present. The proposed ROS ensembles outperform their original counterparts. Furthermore, the original ensembles were outperformed by the ensembles of single-target PCTs. The predictive performance gain with ET-ROS w.r.t. ET is substantial. This is an interesting observation and suggests that ROS could lift predictive performance on smaller datasets with larger output spaces, especially for heavily randomized methods such as extra trees. One possible explanation is that the sampling of input variables in ET, coupled with the small number of examples in the dataset and absence of bootstrapping, introduces a relatively high level of noise in the learning process. The ROS ensemble then actually reduces the effect of this noise at the level of individual base models by specializing them for a smaller output space. This can also explain the small gains for bagging and random forests with the ROS extension on this dataset, because the bootstrapping actually negatively impacts the overall ensemble performance. By inspecting the overfitting score, we note that ROS ensembles consistently exhibit a decreased score w.r.t. ensembles of

single-target PCTs and perform comparably w.r.t. ensembles of multi-target PCTs. Learning and prediction times, as well as model complexity, follow similar patterns as for the *Forestry Kras* dataset.

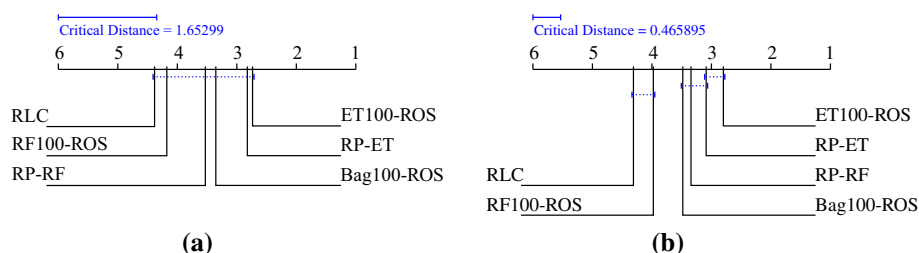
## 5.4 Comparison with other output space transformation methods

In order to put ROS in the broader context of MTR methods with output space transformations, we compare the predictive performance of ROS ensembles and ensembles built with the competing methods proposed in Joly et al. (2014) and Tsoumakas et al. (2014). We have selected these specific methods because they all specialize individual models in the ensemble to a subset of target variables.

Joly et al. (2014) propose ensembles of multi-output regression trees, where each individual tree is built by using a projected output space. Gaussian, Rademacher, Hadamard and Achlioptas projections are used. The goal is to truncate the output space in order to reduce the number of calculations needed to find the best split, which is the main computational burden while building a decision tree. While learning the ensemble, each tree is given a different output space projection. They use two different ensemble methods: Random forests and Extra trees. We dubbed their method Random projections and its two variants as RP-RF and RP-ET. Note that Random projections can not handle nominal attributes and missing values. Hence, the nominal attributes have been converted to numeric scales and missing values have been imputed with the arithmetic mean of that feature.

Tsoumakas et al. (2014) propose an ensemble method called Random Linear Target Combinations for MTR (RLC). They construct new target variables via random linear combinations of existing ones. The data must be normalized in order for the linear combinations to make sense, i.e., to prevent targets on larger scales dominating over the ones with lower scales and thus deteriorating the learning process. The output space is transformed in such a way that each linear combination consists of  $k$  original output features. Each combination is then considered for learning one ensemble member. The transformation of the output space matrix  $\mathbf{Y}$  ( $m \times q$ ) is achieved via a coefficient matrix  $\mathbf{C}$  of size  $q \times r$  filled with random values uniformly chosen from  $[0, 1]$ . Columns of the matrix  $\mathbf{C}$  represent coefficients of the linear combination of the target variables. By multiplying the two matrices, we get the transformed output space  $\mathbf{Y}' = \mathbf{Y}\mathbf{C}$  ( $m \times r$ ) that is then used for training. A user-selected regression algorithm can then be applied on the transformed data.

We present the results using average rank diagrams in Fig. 4, while the complete experimental results are available in “Appendix C” (Table 11). Figure 4 depicts two average rank diagrams: one per dataset and one per target. The per dataset diagram is based on the aRRMSE values, one per dataset. The per target diagram is based on RRMSE values with multiple targets per dataset. The *per dataset analysis* shows no statistically significant differences between the predictive performances of the considered ensemble methods. We can however note, that performances of ET-ROS and RP-ET ensembles seem to be on par (with a minimal advantage of the ET-ROS ensemble). The *per target analysis* detects two statistically significant differences. First, ET-ROS statistically significantly outperforms all other methods with the exception of RP-ET. Second, RLC and RF-ROS ensembles are on par and both are statistically significantly outperformed by the other methods. Third, Bag-ROS, RP-RF and RP-ET perform equally well. All in all, ET-ROS ensembles generally perform better than the other considered ensemble methods.



**Fig. 4** Average rank diagrams showing the predictive performance of ROS (ET-ROS, Bag-ROS and RF-ROS), Random projections (RP-ET and RP-RF) and RLC ensembles. **a** Per dataset (aRRMSE), **b** per target (RRMSE)

## 5.5 Summary of the results

We summarize the main findings of the extensive experimental work presented in the paper by answering the experimental questions posed in Sect. 4.1.

### 1. How many base models do we need in ROS ensembles in order to reach the point of performance saturation?

The saturation point of the original PCT ensembles is between 50 and 75 base models. Bag-ROS and ET-ROS ensembles saturate between 50 and 100 base models. Especially RF-ROS ensembles saturate a bit later, at 75 to 100 base models learned. In the comparative analysis of performance, we consider ensembles with 100 base models (in order to make the comparison fair for all considered methods).

### 2. What is the best value for the portion of target space to be used within such ensembles? Is this portion equal for all evaluated ensemble methods?

The most appropriate size of the portion of target space to be used varies with the ensemble method. The results suggest to use  $v = \frac{1}{2}$  for Bag-ROS and  $v = \frac{3}{4}$  for RF-ROS and ET-ROS.

### 3. Does it make sense to change the default aggregation function of the ensemble that uses the prediction for all targets? Can this improve predictive performance?

Changing the aggregation function changes the behaviour of the ROS ensembles. For Bag-ROS, it can even decrease the predictive performance, so we recommend using the standard aggregation function, i.e., total averaging. For RF-ROS and ET-ROS we recommend making predictions with subspace averaging.

### 4. Considering predictive performance, how do ROS ensemble methods compare to the original ensemble methods?

Using ROS can improve the predictive performance of PCT ensembles. This is especially notable when using ET-ROS with small datasets with larger output spaces.

### 5. Is ROS helpful in terms of time efficiency?

The observed learning times for ROS methods can be substantially lower than the ones of their original counterparts. This especially holds for large datasets. Prediction times, however, do not change.

### 6. Do ROS models use less memory than the models trained with the original ensemble methods?

Ensemble models obtained with ROS have sizes comparable to the ensemble models produced by the original ensemble models.

## 7. How ROS models compare to other output transformation methods?

ET-ROS ensembles generally perform better than ensembles of other competing output transformation methods.

## 6 Conclusions

This work has addressed the task of learning predictive models that can predict the values of multiple continuous variables for a given input tuple, referred to as multi-target regression (MTR): MTR is a task of predicting structured outputs, i.e., structured output prediction. There are two general approaches to solving tasks of such nature. The first, local approach, learns separate models for every component of the predicted structure, whereas the second, global approach, learns one model capable of predicting all components of the structure simultaneously.

We have proposed novel ensemble methods for MTR. An ensemble is a set of predictive models whose predictions are combined and yield the model output. The proposed methods build further on of well known methods for learning ensembles that have been extended to structured outputs. The base models we have considered are predictive clustering trees (PCTs) for MTR. The methods we have proposed are based on the ensemble extension method, i.e., ROS—Random Output Selections. For each ensemble constituent (PCT), the proposed extension randomly selects targets that are considered while learning that particular base model. We perform an extensive experimental evaluation of three ensemble methods extended with ROS, i.e., bagging, random forests and extra-PCTs. The performance has been evaluated on 17 benchmark datasets of varying sizes in terms of number of examples, number of predictive attributes and number of target attributes.

The results show that the proposed extension has a favorable effect, yielding lower error rates and shorter running times. ROS coupled with bagging and extra trees can outperform the original ensemble methods. Random forests do not benefit from ROS in terms of predictive power, but do benefit in terms of shorter learning time. ET-ROS (extra trees with ROS) statistically significantly outperform all original ensemble methods and their ROS (when analyzing predictive performance on a per target basis). We also conducted experiments with three competing methods showing that the proposed method yields the best performance.

We have also provided a computational complexity analysis for the proposed ensemble extension. Our experiments confirm the results of the theoretical analysis. Ensembles with ROS can yield better predictive performance, as well as reduce learning times, whereas the sizes of the induced models do not change notably.

We plan **future work** along several possible directions. To begin with, the aggregation function has an effect on the ensemble predictive performance, as the present work demonstrates. We plan to design a new aggregation function by combining total averaging and subspace averaging, hoping to achieve better performance and better understanding of the effect of subspace averaging. Additionally, we can use out-of-bag errors to derive aggregation weights, such that ensemble constituents with higher error rates would make a smaller contribution to the final prediction. Furthermore, we could perform bias-variance decomposition of the error of all the investigated methods and investigate the sources of errors.

Following an alternative direction, the process of generating target subspaces could also be adapted. The current approach generates target subspaces at random, which is not necessarily the best approach. The relations between target variables could be exploited in order to generate a smaller set of more sensible subspaces.

The final direction we intend to follow is the adaptation of the proposed approaches to other structured output prediction tasks, such as multi-target classification, (hierarchical) multi-label classification, and time-series prediction. For all of these tasks, global random forests are already being used to obtain feature rankings (in the context of predicting structured outputs). ROS could improve this approach/ranking, by considering subsets of the set of target attributes in the process of producing ranks.

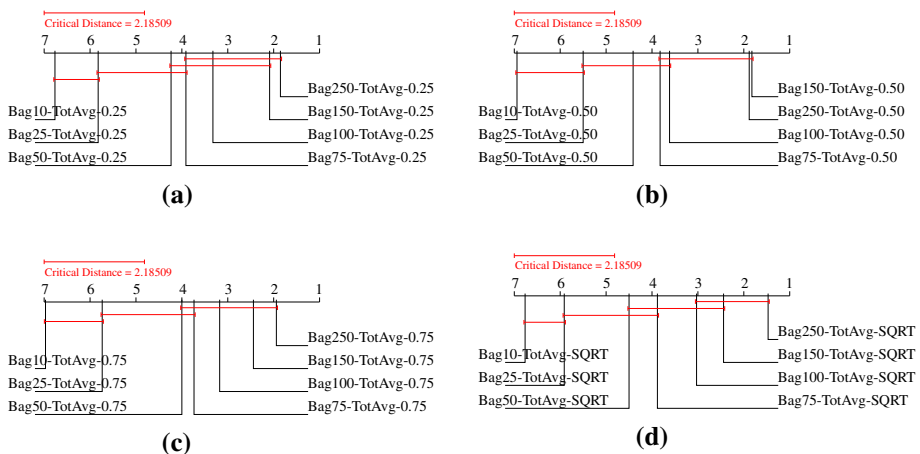
**Acknowledgements** We acknowledge the financial support of the Slovenian Research Agency via the grants P2-0103 and a young researcher grant to MB, as well as the European Commission, through the grants MAESTRA (Learning from Massive, Incompletely annotated, and Structured Data) and HBP (The Human Brain Project), SGA1 and SGA2. SD also acknowledges support by Slovenian Research Agency (via grants J4-7362, L2-7509, and N2-0056), the European Commission (project LANDMARK) and ARVALIS (project BIODIV). The computational experiments presented here were executed on a computing infrastructure from the Slovenian Grid (SLING) initiative.

## Appendix A: Average rank diagrams for ROS variants

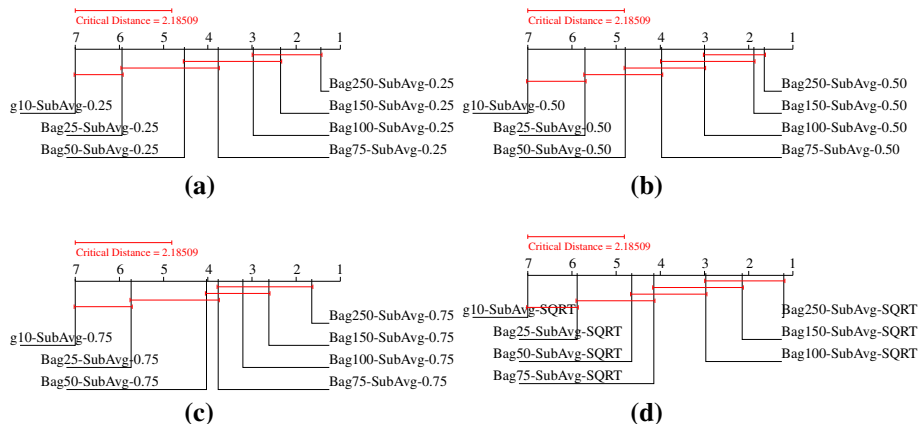
Below we provide average rank diagrams for all considered ROS variants (Bag-ROS, RF-ROS, ET-ROS) for all considered values of the parameter  $v \in \left\{ \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, \frac{1}{\sqrt{T}} \right\}$  and the two types of prediction averaging functions. One average rank diagram corresponds to one combination of values of the above parameters, for which it compares ensembles of different sizes (10, 25, 50, 75, 100, 150 and 250 base models). The saturation point is the lowest number of trees in the ensemble for which the performance is not significantly different than the best. Saturation points are shown in brackets next to value for parameter  $v$ .

### A.1 Bag-ROS variants saturation

The average rank diagrams for the Bag-ROS variants are given in Figs. 5 and 6.



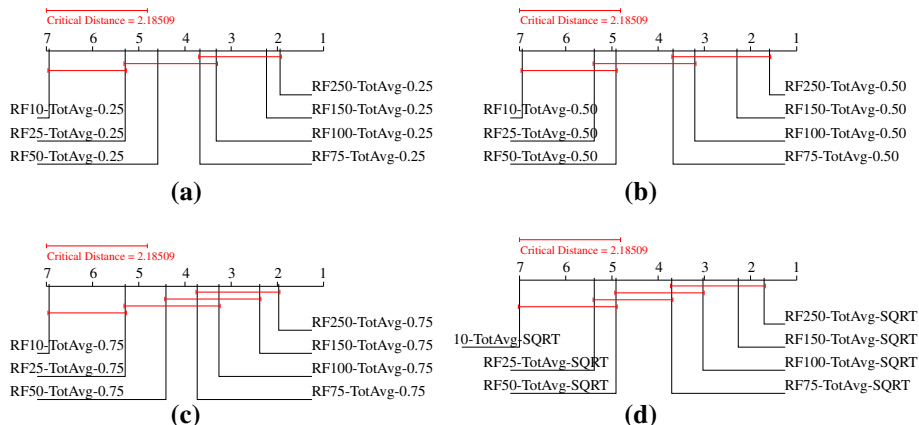
**Fig. 5** Bag-ROS saturation with total averaging. **a** With  $v = \frac{1}{4}$  (50), **b** with  $v = \frac{1}{2}$  (75), **c** with  $v = \frac{3}{4}$  (50), **d** with  $v = \frac{1}{\sqrt{T}}$  (75)



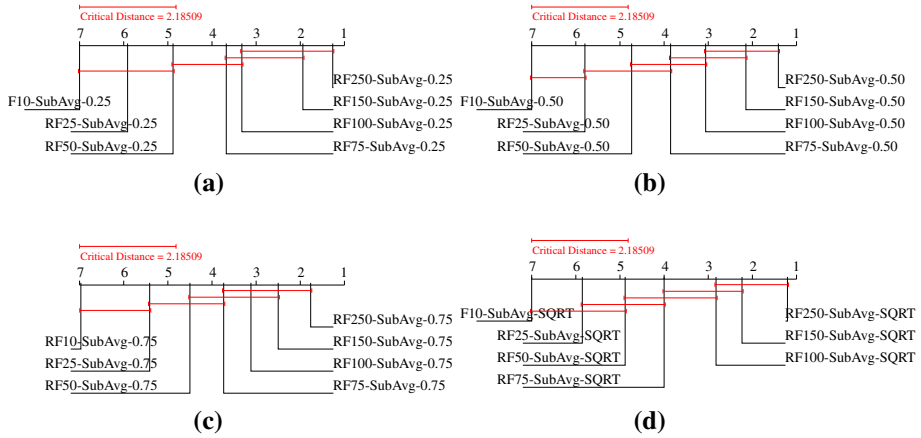
**Fig. 6** Bag-ROS saturation with subset averaging. **a** With  $v = \frac{1}{4}$  (50), **b** with  $v = \frac{1}{2}$  (75), **c** with  $v = \frac{3}{4}$  (50), **d** with  $v = \frac{1}{\sqrt{|T|}}$  (100)

## A.2 RF-ROS variants saturation

The average rank diagrams for the RF-ROS variants are given in Figs. 7 and 8.



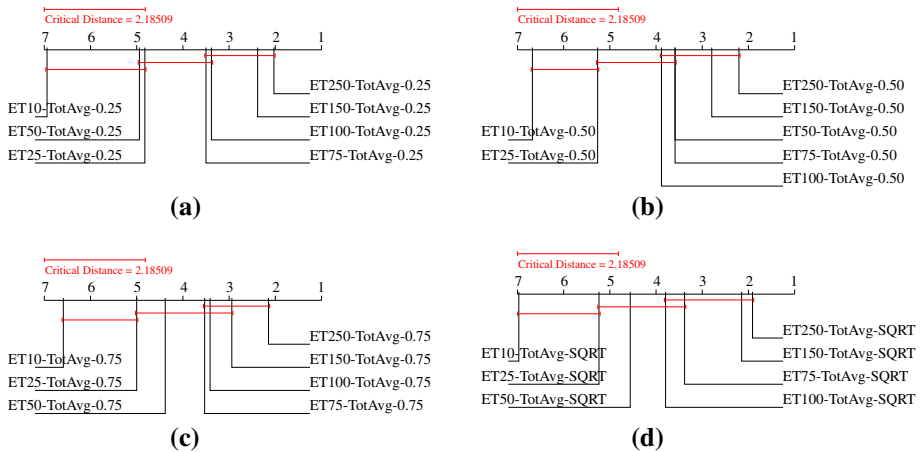
**Fig. 7** RF-ROS saturation with total averaging. **a** With  $v = \frac{1}{4}$  (75), **b** with  $v = \frac{1}{2}$  (75), **c** with  $v = \frac{3}{4}$  (75), **d** with  $v = \frac{1}{\sqrt{|T|}}$  (75)



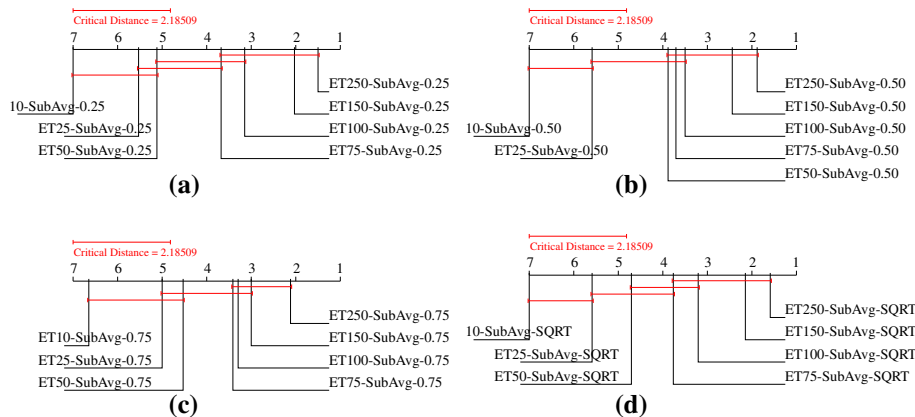
**Fig. 8** RF-ROS saturation with subset averaging. **a** With  $v = \frac{1}{4}$  (100), **b** with  $v = \frac{1}{2}$  (100), **c** with  $v = \frac{3}{4}$  (75), **d** with  $v = \frac{1}{\sqrt{|T|}}$  (100)

### A.3 ET-ROS variants saturation

The average rank diagrams for the ET-ROS variants are given in Figs. 9 and 10.



**Fig. 9** ET-ROS saturation with total averaging. **a** With  $v = \frac{1}{4}$  (75), **b** with  $v = \frac{1}{2}$  (50), **c** with  $v = \frac{3}{4}$  (75), **d** with  $v = \frac{1}{\sqrt{|T|}}$  (75)



**Fig. 10** ET-ROS saturation with subset averaging. **a** With  $v = \frac{1}{4}$  (75), **b** with  $v = \frac{1}{2}$  (50), **c** with  $v = \frac{3}{4}$  (75), **d** with  $v = \frac{1}{\sqrt{7}}$  (100)

## Appendix B: Performance results

The complete performance results for the different ensemble types (bagging, RF, ET) and variants (ST, MT, ROS) are given below. Tables 5, 6 and 7 contain results in terms of aRRMSE, while Tables 8, 9 and 10 contain results in terms of overfitting scores.

**Table 5** Predictive performance of bagging ensembles in terms of aRRMSE

| Dataset                    | MTRT  | Bag-ST | Bag   | Bag-ROS | Subspace averaging |                   |                   |                           |                           |
|----------------------------|-------|--------|-------|---------|--------------------|-------------------|-------------------|---------------------------|---------------------------|
|                            |       |        |       |         | Total averaging    |                   |                   |                           |                           |
|                            |       |        |       |         | $v = \frac{1}{4}$  | $v = \frac{1}{2}$ | $v = \frac{3}{4}$ | $v = \frac{1}{\sqrt{17}}$ | $v = \frac{1}{\sqrt{17}}$ |
| ADNI                       | 1.086 | 0.923  | 0.917 | 0.919   | 0.919              | 0.923             | 0.921             | 0.919                     | 0.926                     |
| ATP ID                     | 0.515 | 0.369  | 0.385 | 0.378   | 0.378              | 0.38              | 0.382             | 0.38                      | 0.374                     |
| ATP 7D                     | 0.585 | 0.486  | 0.473 | 0.474   | 0.474              | 0.467             | 0.461             | 0.467                     | 0.464                     |
| Forestry Kras              | 0.61  | 0.551  | 0.55  | 0.548   | 0.548              | 0.548             | 0.549             | 0.548                     | 0.551                     |
| OES 10                     | 0.616 | 0.487  | 0.531 | 0.531   | 0.531              | 0.527             | 0.53              | 0.531                     | 0.522                     |
| OES 97                     | 0.704 | 0.52   | 0.585 | 0.565   | 0.565              | 0.572             | 0.578             | 0.565                     | 0.566                     |
| Sales                      | 0.867 | 0.681  | 0.691 | 0.687   | 0.687              | 0.69              | 0.691             | 0.689                     | 0.69                      |
| PPMI                       | 0.868 | 0.751  | 0.739 | 0.738   | 0.738              | 0.739             | 0.738             | 0.741                     | 0.749                     |
| Prespa diatoms lake top 10 | 0.995 | 0.968  | 0.946 | 0.935   | 0.935              | 0.937             | 0.944             | 0.937                     | 0.946                     |
| RF 1                       | 0.19  | 0.153  | 0.156 | 0.148   | 0.148              | 0.145             | 0.144             | 0.143                     | 0.141                     |
| RF 2                       | 0.199 | 0.157  | 0.16  | 0.15    | 0.15               | 0.147             | 0.147             | 0.147                     | 0.148                     |
| SCM 1D                     | 0.433 | 0.298  | 0.307 | 0.318   | 0.318              | 0.305             | 0.304             | 0.318                     | 0.303                     |
| SCM 20D                    | 0.527 | 0.363  | 0.359 | 0.363   | 0.363              | 0.354             | 0.353             | 0.363                     | 0.36                      |
| Soil resilience            | 0.926 | 0.889  | 0.883 | 0.874   | 0.874              | 0.872             | 0.88              | 0.88                      | 0.893                     |
| Vegetation condition       | 0.658 | 0.602  | 0.603 | 0.608   | 0.608              | 0.603             | 0.602             | 0.605                     | 0.604                     |
| Vegetation clustering      | 0.809 | 0.699  | 0.706 | 0.702   | 0.702              | 0.699             | 0.702             | 0.699                     | 0.7                       |
| Water quality              | 0.952 | 0.913  | 0.901 | 0.901   | 0.901              | 0.899             | 0.9               | 0.901                     | 0.908                     |

MTRT represents multi-target PCTs. Bag-ST represents ensembles of single-target regression trees. Bag are original bagging ensembles. Bag-ROS are the ROS variants. Ensembles contain 100 trees (ST ensembles contain 100 trees/target)

**Table 6** Predictive performance of random forest ensembles in terms of aRRMSE

| Dataset                    | MTRT  | RF-ST | RF    | RF-ROS | Subspace averaging |                   |                   |                            |                            |
|----------------------------|-------|-------|-------|--------|--------------------|-------------------|-------------------|----------------------------|----------------------------|
|                            |       |       |       |        | Total averaging    |                   |                   |                            |                            |
|                            |       |       |       |        | $v = \frac{1}{4}$  | $v = \frac{1}{2}$ | $v = \frac{3}{4}$ | $v = \frac{1}{\sqrt{ T }}$ | $v = \frac{1}{\sqrt{ T }}$ |
| ADNI                       | 1.086 | 0.92  | 0.919 | 0.916  | 0.916              | 0.916             | 0.917             | 0.916                      | 0.917                      |
| ATP ID                     | 0.515 | 0.398 | 0.41  | 0.413  | 0.418              | 0.418             | 0.414             | 0.418                      | 0.412                      |
| ATP 7D                     | 0.585 | 0.518 | 0.521 | 0.545  | 0.535              | 0.535             | 0.521             | 0.535                      | 0.517                      |
| Forestry Kras              | 0.61  | 0.546 | 0.545 | 0.545  | 0.545              | 0.545             | 0.545             | 0.545                      | 0.546                      |
| OES 10                     | 0.616 | 0.492 | 0.517 | 0.525  | 0.519              | 0.519             | 0.518             | 0.525                      | 0.521                      |
| OES 97                     | 0.704 | 0.491 | 0.519 | 0.52   | 0.532              | 0.532             | 0.526             | 0.52                       | 0.523                      |
| Sales                      | 0.867 | 0.721 | 0.735 | 0.738  | 0.735              | 0.735             | 0.732             | 0.733                      | 0.733                      |
| PPMI                       | 0.868 | 0.759 | 0.748 | 0.751  | 0.749              | 0.749             | 0.749             | 0.757                      | 0.751                      |
| Prespa diatoms lake top 10 | 0.995 | 0.942 | 0.937 | 0.936  | 0.932              | 0.932             | 0.937             | 0.936                      | 0.939                      |
| RF 1                       | 0.19  | 0.144 | 0.154 | 0.157  | 0.155              | 0.155             | 0.154             | 0.154                      | 0.152                      |
| RF 2                       | 0.199 | 0.161 | 0.167 | 0.173  | 0.169              | 0.169             | 0.169             | 0.17                       | 0.168                      |
| SCM 1D                     | 0.433 | 0.292 | 0.31  | 0.323  | 0.314              | 0.314             | 0.312             | 0.323                      | 0.31                       |
| SCM 20D                    | 0.527 | 0.378 | 0.378 | 0.39   | 0.38               | 0.38              | 0.379             | 0.39                       | 0.38                       |
| Soil resilience            | 0.926 | 0.86  | 0.864 | 0.871  | 0.865              | 0.865             | 0.872             | 0.869                      | 0.884                      |
| Vegetation condition       | 0.658 | 0.601 | 0.602 | 0.61   | 0.604              | 0.604             | 0.603             | 0.606                      | 0.603                      |
| Vegetation clustering      | 0.809 | 0.696 | 0.703 | 0.711  | 0.705              | 0.705             | 0.703             | 0.708                      | 0.704                      |
| Water quality              | 0.952 | 0.902 | 0.898 | 0.901  | 0.899              | 0.899             | 0.899             | 0.901                      | 0.901                      |

MTRT represents multi-target PCTs. RF-ST represents ensembles of single-target regression trees. RF are original random forest ensembles. RF-ROS are the ROS variants. Ensembles contain 100 trees (ST ensembles contain 100 trees/target)

**Table 7** Predictive performance of extra tree ensembles in terms of aRRMSE

| Dataset                    | MTRT  | ET-ST | ET    | ET-ROS | Subspace averaging |                   |                   |                            |                   |                            |
|----------------------------|-------|-------|-------|--------|--------------------|-------------------|-------------------|----------------------------|-------------------|----------------------------|
|                            |       |       |       |        | Total averaging    |                   |                   |                            |                   |                            |
|                            |       |       |       |        | $v = \frac{1}{4}$  | $v = \frac{1}{2}$ | $v = \frac{3}{4}$ | $v = \frac{1}{\sqrt{ T }}$ | $v = \frac{1}{4}$ | $v = \frac{1}{\sqrt{ T }}$ |
| ADNI                       | 1.086 | 0.917 | 0.922 | 0.914  | 0.913              | 0.916             | 0.914             | 0.918                      | 0.915             | 0.918                      |
| ATP ID                     | 0.515 | 0.365 | 0.435 | 0.372  | 0.378              | 0.376             | 0.378             | 0.364                      | 0.37              | 0.37                       |
| ATP 7D                     | 0.585 | 0.468 | 0.523 | 0.453  | 0.448              | 0.438             | 0.448             | 0.441                      | 0.442             | 0.442                      |
| Forestry Kras              | 0.61  | 0.56  | 0.557 | 0.556  | 0.557              | 0.557             | 0.556             | 0.56                       | 0.558             | 0.559                      |
| OES 10                     | 0.616 | 0.467 | 0.514 | 0.511  | 0.499              | 0.497             | 0.511             | 0.502                      | 0.495             | 0.502                      |
| OES 97                     | 0.704 | 0.464 | 0.52  | 0.517  | 0.517              | 0.518             | 0.517             | 0.513                      | 0.514             | 0.513                      |
| Sales                      | 0.867 | 0.733 | 0.763 | 0.702  | 0.7                | 0.698             | 0.7               | 0.709                      | 0.704             | 0.701                      |
| PPMI                       | 0.868 | 0.759 | 0.743 | 0.745  | 0.745              | 0.745             | 0.746             | 0.75                       | 0.747             | 0.754                      |
| Prespa diatoms lake top 10 | 0.995 | 0.934 | 0.929 | 0.922  | 0.923              | 0.926             | 0.923             | 0.935                      | 0.932             | 0.931                      |
| RF 1                       | 0.19  | 0.137 | 0.144 | 0.151  | 0.148              | 0.147             | 0.147             | 0.146                      | 0.145             | 0.144                      |
| RF 2                       | 0.199 | 0.147 | 0.149 | 0.154  | 0.151              | 0.152             | 0.152             | 0.149                      | 0.152             | 0.152                      |
| SCM 1D                     | 0.433 | 0.274 | 0.289 | 0.293  | 0.284              | 0.282             | 0.293             | 0.283                      | 0.281             | 0.283                      |
| SCM 20D                    | 0.527 | 0.322 | 0.33  | 0.326  | 0.319              | 0.316             | 0.326             | 0.325                      | 0.319             | 0.325                      |
| Soil resilience            | 0.926 | 0.884 | 0.929 | 0.869  | 0.877              | 0.873             | 0.871             | 0.884                      | 0.88              | 0.87                       |
| Vegetation condition       | 0.658 | 0.6   | 0.605 | 0.606  | 0.601              | 0.599             | 0.602             | 0.606                      | 0.601             | 0.602                      |
| Vegetation clustering      | 0.809 | 0.688 | 0.695 | 0.702  | 0.698              | 0.696             | 0.7               | 0.699                      | 0.696             | 0.697                      |
| Water quality              | 0.952 | 0.896 | 0.9   | 0.895  | 0.893              | 0.894             | 0.895             | 0.9                        | 0.893             | 0.9                        |

MTRT represents multi-target PCTs. ET-ST represents ensembles of single-target regression trees. ET are original extra tree ensembles. ET-ROS are the ROS variants. Ensembles contain 100 trees (ST ensembles contain 100 trees/target)

**Table 8** Predictive performance of bagging ensembles in terms of overfitting score

| Dataset                    | MTRT  | Bag-ST | Bag   | Bag-ROS | Subspace averaging |                   |                   |                          |                          |
|----------------------------|-------|--------|-------|---------|--------------------|-------------------|-------------------|--------------------------|--------------------------|
|                            |       |        |       |         | Total averaging    |                   |                   |                          |                          |
|                            |       |        |       |         | $v = \frac{1}{4}$  | $v = \frac{1}{2}$ | $v = \frac{3}{4}$ | $v = \frac{1}{\sqrt{p}}$ | $v = \frac{1}{\sqrt{p}}$ |
| ADNI                       | 0.959 | 0.832  | 0.593 | 0.555   | 0.587              | 0.601             | 0.555             | 0.659                    | 0.639                    |
| ATP ID                     | 0.368 | 1.464  | 1.216 | 0.933   | 1.069              | 1.181             | 1.069             | 1.255                    | 1.257                    |
| ATP 7D                     | 2.131 | 1.511  | 1.307 | 1.052   | 1.173              | 1.369             | 1.173             | 1.464                    | 1.444                    |
| Forestry Kras              | 0.169 | 0.494  | 0.476 | 0.464   | 0.471              | 0.474             | 0.467             | 0.476                    | 0.475                    |
| OES 10                     | 0.704 | 1.556  | 1.114 | 1.034   | 1.052              | 1.108             | 1.034             | 1.238                    | 1.117                    |
| OES 97                     | 0.745 | 1.427  | 0.973 | 0.839   | 0.933              | 0.958             | 0.839             | 1.055                    | 1.038                    |
| Sales                      | 0.792 | 1.128  | 0.781 | 0.667   | 0.728              | 0.763             | 0.698             | 0.875                    | 0.83                     |
| PPMI                       | 0.383 | 0.529  | 0.467 | 0.449   | 0.46               | 0.462             | 0.438             | 0.48                     | 0.472                    |
| Prespa diatoms lake top 10 | 0.226 | 1.182  | 0.988 | 0.745   | 0.857              | 0.927             | 0.803             | 1.078                    | 1.038                    |
| RF 1                       | 1.085 | 1.034  | 0.748 | 0.545   | 0.65               | 0.685             | 0.584             | 0.539                    | 0.679                    |
| RF 2                       | 1.219 | 1.046  | 0.806 | 0.532   | 0.654              | 0.699             | 0.608             | 0.568                    | 0.683                    |
| SCM 1D                     | 0.681 | 1.482  | 0.975 | 0.846   | 0.923              | 0.958             | 0.846             | 1.105                    | 1.031                    |
| SCM 20D                    | 0.595 | 1.469  | 1.054 | 0.961   | 1.011              | 1.032             | 0.961             | 1.138                    | 1.096                    |
| Soil resilience            | 0.217 | 1.153  | 0.791 | 0.679   | 0.746              | 0.784             | 0.697             | 0.944                    | 0.878                    |
| Vegetation condition       | 0.075 | 1.352  | 1.061 | 0.628   | 0.903              | 1.021             | 0.8               | 1.205                    | 1.136                    |
| Vegetation clustering      | 0.085 | 1.179  | 1.018 | 0.839   | 0.932              | 0.993             | 0.878             | 1.06                     | 1.036                    |
| Water quality              | 0.018 | 1.245  | 0.915 | 0.643   | 0.788              | 0.871             | 0.643             | 1.041                    | 0.976                    |

MTRT represents multi-target PCTs. Bag-ST represents ensembles of single-target regression trees. Bag are original bagging ensembles. Bag-ROS are the ROS variants. Ensembles contain 100 trees (ST ensembles contain 100 trees/target)

**Table 9** Predictive performance of random forest ensembles in terms of overfitting score

| Dataset                    | MTRT  | RF-ST | RF    | RF-ROS | Subspace averaging |                   |                   |                            |                            |
|----------------------------|-------|-------|-------|--------|--------------------|-------------------|-------------------|----------------------------|----------------------------|
|                            |       |       |       |        | Total averaging    |                   |                   |                            |                            |
|                            |       |       |       |        | $v = \frac{1}{4}$  | $v = \frac{1}{2}$ | $v = \frac{3}{4}$ | $v = \frac{1}{\sqrt{ T }}$ | $v = \frac{1}{\sqrt{ T }}$ |
| ADNI                       | 0.959 | 0.645 | 0.481 | 0.447  | 0.447              | 0.462             | 0.479             | 0.447                      | 0.514                      |
| ATP ID                     | 0.368 | 1.31  | 1.115 | 0.982  | 0.982              | 1.127             | 1.186             | 1.127                      | 1.292                      |
| ATP 7D                     | 2.131 | 1.411 | 1.177 | 1.002  | 1.002              | 1.176             | 1.205             | 1.176                      | 1.225                      |
| Forestry Kras              | 0.169 | 0.457 | 0.44  | 0.434  | 0.434              | 0.438             | 0.44              | 0.435                      | 0.441                      |
| OES 10                     | 0.704 | 1.525 | 1.093 | 1.059  | 1.059              | 1.074             | 1.108             | 1.059                      | 1.132                      |
| OES 97                     | 0.745 | 1.291 | 0.88  | 0.804  | 0.804              | 0.877             | 0.871             | 0.804                      | 0.974                      |
| Sales                      | 0.792 | 0.563 | 0.458 | 0.462  | 0.462              | 0.468             | 0.475             | 0.437                      | 0.503                      |
| PPMI                       | 0.383 | 0.531 | 0.471 | 0.464  | 0.464              | 0.47              | 0.475             | 0.462                      | 0.478                      |
| Prespa diatoms lake top 10 | 0.226 | 1.071 | 0.917 | 0.718  | 0.718              | 0.782             | 0.859             | 0.764                      | 0.902                      |
| RF 1                       | 1.085 | 0.678 | 0.511 | 0.493  | 0.493              | 0.521             | 0.552             | 0.479                      | 0.53                       |
| RF 2                       | 1.219 | 0.53  | 0.479 | 0.427  | 0.427              | 0.488             | 0.507             | 0.462                      | 0.501                      |
| SCM 1D                     | 0.681 | 1.366 | 0.955 | 0.87   | 0.87               | 0.925             | 0.951             | 0.87                       | 1.047                      |
| SCM 20D                    | 0.595 | 1.356 | 1.034 | 0.958  | 0.958              | 0.997             | 1.015             | 0.958                      | 1.061                      |
| Soil resilience            | 0.217 | 1.012 | 0.766 | 0.679  | 0.679              | 0.732             | 0.763             | 0.708                      | 0.803                      |
| Vegetation condition       | 0.075 | 1.241 | 0.982 | 0.654  | 0.654              | 0.866             | 0.951             | 0.787                      | 1.036                      |
| Vegetation clustering      | 0.085 | 1.029 | 0.885 | 0.767  | 0.767              | 0.827             | 0.866             | 0.794                      | 0.898                      |
| Water quality              | 0.018 | 1.12  | 0.836 | 0.659  | 0.659              | 0.761             | 0.812             | 0.659                      | 0.857                      |

MTRT represents multi-target PCTs. RF-ST represents ensembles of single-target regression trees. RF are original random forest ensembles. RF-ROS are the ROS variants. Ensembles contain 100 trees (ST ensembles contain 100 trees/target)

**Table 10** Predictive performance of extra tree ensembles in terms of overfitting score

| Dataset                    | MTRT  | ET-ST | ET    | ET-ROS | Subspace averaging |                   |                   |                            |                   |                   |
|----------------------------|-------|-------|-------|--------|--------------------|-------------------|-------------------|----------------------------|-------------------|-------------------|
|                            |       |       |       |        | Total averaging    |                   |                   |                            |                   |                   |
|                            |       |       |       |        | $v = \frac{1}{4}$  | $v = \frac{1}{2}$ | $v = \frac{3}{4}$ | $v = \frac{1}{\sqrt{ T }}$ | $v = \frac{1}{4}$ | $v = \frac{3}{4}$ |
| ADNI                       | 0.959 | 1.274 | 1.022 | 0.708  | 0.708              | 0.746             | 0.765             | 0.708                      | 0.923             | 0.844             |
| ATP ID                     | 0.368 | 4     | 2.616 | 1.694  | 1.694              | 2.147             | 2.504             | 2.147                      | 3.507             | 2.856             |
| ATP 7D                     | 2.131 | 5.397 | 2.766 | 2.186  | 2.186              | 2.679             | 3.176             | 2.679                      | 4.713             | 4.188             |
| Forestry Kras              | 0.169 | 0.611 | 0.575 | 0.567  | 0.567              | 0.575             | 0.576             | 0.57                       | 0.595             | 0.584             |
| OES 10                     | 0.704 | 2.654 | 0.986 | 1.254  | 1.254              | 1.156             | 1.104             | 1.254                      | 1.908             | 1.344             |
| OES 97                     | 0.745 | 3.31  | 0.827 | 1.13   | 1.13               | 1.195             | 1.217             | 1.13                       | 2.028             | 1.609             |
| Sales                      | 0.792 | 1.32  | 1.006 | 0.783  | 0.783              | 0.833             | 0.85              | 0.804                      | 1.047             | 0.948             |
| PPMI                       | 0.383 | 0.614 | 0.517 | 0.531  | 0.531              | 0.535             | 0.538             | 0.525                      | 0.563             | 0.549             |
| Prespa diatoms lake top 10 | 0.226 | 1.99  | 1.419 | 1.018  | 1.018              | 1.167             | 1.356             | 1.095                      | 1.651             | 1.548             |
| RF 1                       | 1.085 | 0.763 | 0.648 | 0.539  | 0.539              | 0.563             | 0.6               | 0.538                      | 0.765             | 0.661             |
| RF 2                       | 1.219 | 0.605 | 0.572 | 0.51   | 0.51               | 0.516             | 0.556             | 0.566                      | 0.554             | 0.541             |
| SCM 1D                     | 0.681 | 5.235 | 1.415 | 1.256  | 1.256              | 1.39              | 1.432             | 1.256                      | 2.443             | 1.836             |
| SCM 20D                    | 0.595 | 4.439 | 1.524 | 1.387  | 1.387              | 1.469             | 1.503             | 1.387                      | 2.242             | 1.807             |
| Soil resilience            | 0.217 | 1.3   | 0.908 | 0.802  | 0.802              | 0.866             | 0.85              | 0.84                       | 1.106             | 0.977             |
| Vegetation condition       | 0.075 | 4.321 | 1.935 | 1.166  | 1.166              | 1.616             | 1.886             | 1.421                      | 3.143             | 2.451             |
| Vegetation clustering      | 0.085 | 2.562 | 1.785 | 1.293  | 1.293              | 1.558             | 1.73              | 1.404                      | 2.159             | 1.953             |
| Water quality              | 0.018 | 2.558 | 1.325 | 0.981  | 0.981              | 1.124             | 1.241             | 0.981                      | 1.785             | 1.555             |

MTRT represents multi-target PCTs. ET-ST represents ensembles of single-target regression trees. ET are original extra tree ensembles. ET-ROS are the ROS variants. Ensembles contain 100 trees (ST ensembles contain 100 trees/target)

## Appendix C: Performance results compared to other output space transformation methods

This section includes the results of the comparison of the performance of ROS ensembles with the performance of three competing methods: two variants of Random projections method which were proposed by Joly (2017) and RLC ensembles, proposed by Tsoumakas et al. (2014). The predictive performance is measured in terms of aRRMSE.

*Method parameters* All ensembles contain 100 base models. **ROS** ensembles were parametrized as described in Sect. 5.2. **Random projections** variants (RP-RF, RP-ET) use  $m = \log(|T|)$  components in the projected output space where  $T$  is the set of target attributes. In addition, Rademacher random projections were used for output space transformations. RP-RF used  $k = \sqrt{|X|}$  randomly chosen input features to calculate splits where  $X$  is the set of all input features. RP-ET used  $k = |X|$ . Minimal number of allowed instances in a leaf node was set to 1 for both variants. The code for both variants of Random projections is available at <https://github.com/arjoly/random-output-trees>. **RLC** was parametrized to use gradient boosting with 4-terminal node regression tree as the base regressor with learning rate of 0.1 and 100 boosting iterations. Number of targets that participate in the random linear combinations was set to  $k = 2$ . RLC is implemented as part of the MULAN library, available at <http://mulan.sourceforge.net> (Table 11).

**Table 11** Predictive performance of ROS ensembles, ensembles of Random projections variants (RP-RF, RP-ET) and RLC ensembles in terms of aRRMSE. Lower values mean better performance

| Dataset                    | RP-RF        | RP-ET        | RLC          | Bag-ROS | RF-ROS       | ET-ROS       |
|----------------------------|--------------|--------------|--------------|---------|--------------|--------------|
| ADNI                       | 0.941        | 0.936        | 0.94         | 0.923   | 0.918        | <b>0.916</b> |
| ATP 1D                     | 0.435        | 0.427        | 0.415        | 0.38    | 0.412        | <b>0.374</b> |
| ATP 7D                     | 0.413        | <b>0.326</b> | 0.358        | 0.467   | 0.517        | 0.436        |
| Forestry Kras              | 0.558        | 0.583        | 0.629        | 0.548   | <b>0.546</b> | 0.557        |
| OES 10                     | 0.468        | 0.467        | <b>0.447</b> | 0.527   | 0.518        | 0.496        |
| OES 97                     | 0.535        | <b>0.513</b> | <b>0.513</b> | 0.572   | 0.528        | 0.518        |
| Sales                      | <b>0.646</b> | 0.647        | 0.707        | 0.69    | 0.733        | 0.698        |
| PPMI                       | <b>0.73</b>  | 0.757        | 0.767        | 0.739   | 0.75         | 0.746        |
| Prespa diatoms lake top 10 | 0.831        | 0.827        | <b>0.811</b> | 0.937   | 0.939        | 0.928        |
| RF 1                       | 0.14         | <b>0.136</b> | 0.238        | 0.145   | 0.152        | 0.147        |
| RF 2                       | <b>0.145</b> | 0.151        | 0.241        | 0.147   | 0.168        | 0.153        |
| SCM 1D                     | 0.311        | 0.285        | 0.376        | 0.305   | 0.31         | <b>0.281</b> |
| SCM 20D                    | 0.366        | 0.324        | 0.595        | 0.354   | 0.379        | <b>0.316</b> |
| Soil resilience            | 0.847        | 0.845        | <b>0.761</b> | 0.872   | 0.872        | 0.873        |
| Vegetation condition       | 0.605        | 0.603        | 0.647        | 0.603   | 0.603        | <b>0.599</b> |
| Vegetation clustering      | 0.702        | 0.702        | 0.79         | 0.699   | 0.703        | <b>0.696</b> |
| Water quality              | 0.902        | 0.901        | 0.91         | 0.899   | 0.899        | <b>0.894</b> |
| Win count                  | 3            | 3            | 4            | 0       | 1            | 7            |

Bold values indicate the best performing method on a given dataset

## References

- Abraham, Z., Tan, P. N., Winkler, J., Zhong, S., Liszewska, M., et al. (2013). Position preserving multi-output prediction. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 320–335). Springer.
- Aho, T., Ženko, B., Džeroski, S., & Elomaa, T. (2012). Multi-target regression with rule ensembles. *Journal of Machine Learning Research*, 13, 2367–2407.
- Alvarez, M. A., Rosasco, L., Lawrence, N. D., et al. (2012). Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3), 195–266.
- Appice, A., & Džeroski, S. (2007). Stepwise induction of multi-target model trees. In *Machine Learning: ECML 2007, LNCS* (Vol. 4701, pp. 502–509). Springer.
- Appice, A., & Malerba, D. (2014). Leveraging the power of local spatial autocorrelation in geophysical interpolative clustering. *Data Mining and Knowledge Discovery*, 28(5–6), 1266–1313.
- Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1), 105–139.
- Blockeel, H. (1998). *Top-down induction of first order logical decision trees*. Ph.D. thesis, Katholieke Universiteit Leuven, Leuven, Belgium.
- 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* (Vol. 1704, pp. 32–40). Springer.
- Blockeel, H., Raedt, L. D., & Ramon, J. (1998). Top-down induction of clustering trees. In *Proceedings of the 15th international conference on machine learning* (pp. 55–63), Morgan Kaufmann.
- Blockeel, H., & Struyf, J. (2002). Efficient algorithms for decision tree cross-validation. *Journal of Machine Learning Research*, 3, 621–650.
- Borchani, H., Varando, G., Bielza, C., & Larrañaga, P. (2015). A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5), 216–233.
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123–140.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- 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.
- Debeljak, M., Kocev, D., Towers, W., Jones, M., Griffiths, B., & Hallett, P. (2009). Potential of multi-objective models for risk-based mapping of the resilience characteristics of soils: Demonstration at a national level. *Soil Use and Management*, 25(1), 66–77.
- Deger, F., Mansouri, A., Pedersen, M., Hardeberg, J. Y., & Voisin, Y. (2012). Multi-and single-output support vector regression for spectral reflectance recovery. In *2012 eighth international conference on signal image technology and internet based systems (SITIS)* (pp. 805–810). IEEE.
- Demšar, D., Džeroski, S., Larsen, T., Struyf, J., Axelsen, J., Bruns-Pedersen, M., et al. (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.
- Dunn, O. J. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293), 52–64.
- Džeroski, S., Demšar, D., & Grbović, J. (2000). Predicting chemical parameters of river water quality from bioindicator data. *Applied Intelligence*, 13(1), 7–17.
- Džeroski, S., Kobler, A., Gjorgjioski, V., & Panov, P. (2006). Using decision trees to predict forest stand height and canopy cover from LANSAT and LIDAR data. In *Managing environmental knowledge: EnviroInfo 2006: Proceedings of the 20th international conference on informatics for environmental protection* (pp. 125–133). Aachen: Shaker Verlag.
- Džeroski, S. (2007). *Towards a general framework for data mining* (pp. 259–300). Berlin: Springer. [https://doi.org/10.1007/978-3-540-75549-4\\_16](https://doi.org/10.1007/978-3-540-75549-4_16).
- Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, 11, 86–92.
- Gamberger, D., Ženko, B., Mitelpunkt, A., Shachar, N., & Lavrač, N. (2016). Clusters of male and female alzheimers disease patients in the Alzheimers disease neuroimaging initiative (ADNI) database. *Brain Informatics*, 3(3), 169–179.
- Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1), 3–42.
- Gjorgjioski, V., Džeroski, S., & White, M. (2008). *Clustering analysis of vegetation data*. Technical report 10065, Jožef Stefan Institute.
- Han, Z., Liu, Y., Zhao, J., & Wang, W. (2012). Real time prediction for converter gas tank levels based on multi-output least square support vector regressor. *Control Engineering Practice*, 20(12), 1400–1409.

- Ikonovska, E., Gama, J., & Džeroski, S. (2011). Incremental multi-target model trees for data streams. In *Proceedings of the 2011 ACM symposium on applied computing* (pp. 988–993). ACM.
- Iman, R. L., & Davenport, J. M. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics: Theory and Methods*, 9(6), 571–595.
- Izenman, A. J. (1975). Reduced-rank regression for the multivariate linear model. *Journal of multivariate analysis*, 5(2), 248–264.
- Jančič, S., Frisvad, J. C., Kocev, D., Gostinčar, C., Džeroski, S., & Gunde-Cimerman, N. (2016). Production of secondary metabolites in extreme environments: Food- and airborne *Walleimia* spp. produce toxic metabolites at hypersaline conditions. *PLoS ONE*, 11(12), e0169116.
- Joly, A. (2017). Exploiting random projections and sparsity with random forests and gradient boosting methods—Application to multi-label and multi-output learning, random forest model compression and leveraging input sparsity. arXiv preprint [arXiv:1704.08067](https://arxiv.org/abs/1704.08067)
- Joly, A., Geurts, P., Wehenkel, L. (2014). Random forests with random projections of the output space for high dimensional multi-label classification. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 607–622). Springer.
- Kaggle. (2008). *Kaggle competition: Online product sales*. <https://www.kaggle.com/c/online-sales/data>. Accessed July 19, 2017.
- Kocev, D. (2011). *Ensembles for predicting structured outputs*. Ph.D. thesis, Jožef Stefan International Post-graduate School, Ljubljana, Slovenia.
- Kocev, D., & Ceci, M. (2015). Ensembles of extremely randomized trees for multi-target regression. In *Discovery science: 18th international conference (DS 2015), LNCS*, (Vol. 9356, pp. 86–100).
- 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.
- Kocev, D., Naumoski, A., Mitreski, K., Krstić, S., & Džeroski, S. (2010). Learning habitat models for the diatom community in Lake Prespa. *Ecological Modelling*, 221(2), 330–337.
- 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* (Vol. 4701, pp. 624–631). Springer.
- Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2013). Tree ensembles for predicting structured outputs. *Pattern Recognition*, 46(3), 817–833.
- Kriegel, H. P., Borgwardt, K., Kröger, P., Pryakhin, A., Schubert, M., & Zimek, A. (2007). Future trends in data mining. *Data Mining and Knowledge Discovery*, 15, 87–97.
- Levatić, J., Ceci, M., Kocev, D., & Džeroski, S. (2014). Semi-supervised learning for multi-target regression. In *International workshop on new frontiers in mining complex patterns* (pp. 3–18). Springer.
- Madjarov, G., Gjorgjević, D., Dimitrovski, I., & Džeroski, S. (2016). The use of data-derived label hierarchies in multi-label classification. *Journal of Intelligent Information Systems*, 47(1), 57–90.
- Marek, K., Jennings, D., Lasch, S., Siderowf, A., Tanner, C., Simuni, T., et al. (2011). The Parkinson Progression Marker Initiative (PPMI). *Progress in Neurobiology*, 95(4), 629–635.
- Micchelli, C. A., & 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, P. B. (1963). *Distribution-free multiple comparisons*. Ph.D. thesis, Princeton University, Princeton, NY, USA.
- Panov, P., Soldatova, L. N., & Džeroski, S. (2016). Generic ontology of datatypes. *Information Sciences*, 329, 900–920.
- 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.
- Spyromitros-Xioufis, E., Tsoumakas, G., Groves, W., & Vlahavas, I. (2016). Multi-target regression via input space expansion: Treating targets as inputs. *Machine Learning*, 104(1), 55–98.
- Stojanova, D., Ceci, M., Appice, A., & Džeroski, S. (2012). Network regression with predictive clustering trees. In *Data mining and knowledge discovery* (pp. 1–36).
- 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 *Proceedings of the 4th international workshop on knowledge discovery in inductive databases KDID—LNCS* (Vol. 3933, pp. 222–233). Springer.
- Szymański, P., Kajdanowicz, T., & Kersting, K. (2016). How is a data-driven approach better than random choice in label space division for multi-label classification? *Entropy*, 18(8), 282.

- Tsoumakas, G., Spyromitros-Xioufis, E., Vrekou, A., & Vlahavas, I. (2014). Multi-target regression via random linear target combinations. In *Machine learning and knowledge discovery in databases: ECML-PKDD 2014, LNCS* (Vol. 8726, pp. 225–240).
- Tsoumakas, G., & Vlahavas, I. (2007). Random k-labelsets: An ensemble method for multilabel classification. In *Proceedings of the 18th European conference on machine learning* (pp. 406–417).
- Vens, C., Struyf, J., Schietgat, L., Džeroski, S., & Blockeel, H. (2008). Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2), 185–214.
- Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*. Los Altos: Morgan Kaufmann.
- Xu, S., An, X., Qiao, X., Zhu, L., & Li, L. (2013). Multi-output least-squares support vector regression machines. *Pattern Recognition Letters*, 34(9), 1078–1084.
- Yang, Q., & Wu, X. (2006). 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(4), 597–604.
- Ženko, B. (2007). *Learning predictive clustering rules*. Ph.D. thesis, Faculty of Computer Science, University of Ljubljana, Ljubljana, Slovenia.
- Zhang, W., Liu, X., Ding, Y., & Shi, D. (2012). Multi-output LS-SVR machine in extended feature space. In *2012 IEEE international conference on computational intelligence for measurement systems and applications (CIMSAP)* (pp. 130–134). IEEE.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.