Machine Learning (2020) 109:2121–2139
https://doi.org/10.1007/s10994-020-05918-z

Incremental predictive clustering trees for online semi-supervised multi-target regression

Aljaž Osojnik[1] · **Panče Panov**[2] · **Sašo Džeroski**[1]

Received: 16 July 2019 / Revised: 8 July 2020 / Accepted: 19 September 2020 / Published online: 28 October 2020

Abstract
In many application settings, labeling data examples is a costly endeavor, while unlabeled examples are abundant and cheap to produce. Labeling examples can be particularly problematic in an online setting, where there can be arbitrarily many examples that arrive at high frequencies. It is also problematic when we need to predict complex values (e.g., multiple real values), a task that has started receiving considerable attention, but mostly in the batch setting. In this paper, we propose a method for online semi-supervised multi-target regression. It is based on incremental trees for multi-target regression and the predictive clustering framework. Furthermore, it utilizes unlabeled examples to improve its predictive performance as compared to using just the labeled examples. We compare the proposed iSOUP-PCT method with supervised tree methods, which do not use unlabeled examples, and to an oracle method, which uses unlabeled examples as though they were labeled. Additionally, we compare the proposed method to the available state-of-the-art methods. The method achieves good predictive performance on account of increased consumption of computational resources as compared to its supervised variant. The proposed method also beats the state-of-the-art in the case of very few labeled examples in terms of performance, while achieving comparable performance when the labeled examples are more common.

Keywords Multi-target regression · Data stream mining · Semi-supervised learning · Predictive clustering

Editors: Larisa Soldatova, Joaquin Vanschoren.

✉ Aljaž Osojnik
aljaz.osojnik@ijs.si

Panče Panov
pance.panov@ijs.si

Sašo Džeroski
saso.dzeroski@ijs.si

[1] Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia

[2] Jožef Stefan International Postgraduate School, Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia

1 Introduction

Recently, there has been lot of interest in the research community to develop methods for prediction of complex values. One such predictive learning task is the task of multi-target regression (MTR), where we want to predict multiple continuous values, called targets, at the same time. The targets are assumed to be related, but equally important. Methods for MTR can be used directly to produce predictive models or they can be utilized by more complex systems, e.g., in recommender systems. Methods for MTR are fairly common in the regular batch learning setting, but rarer in the online learning setting.

In the batch learning setting, the entire dataset is available at the start of the learning process and the order of the examples in the dataset is generally assumed not to have an impact on the learning process. In online learning, the entire dataset is not available at the start of the learning process. Instead, we are learning from a *data stream*, an ordered sequence of examples which become available throughout the learning process. This also means that there is an inherent time component to online learning, i.e., one example arrives either before or after another example.

The datasets in online learning are naturally ordered sequences of examples. To avoid the need for arbitrarily large memory storage, each example is processed only once at the time of its arrival. Afterwards, it is generally discarded, though, a select number of examples may be archived for further use. We expect any predictive model to be applicable in real-time or near real-time, i.e., at any point throughout the learning process the current model can be applied to calculate a prediction. Furthermore, the predictive model is expected to be as up to date as possible, i.e., it should incorporate information from all of the examples available up to this point in time. To allow the most current predictive model to be kept updated, the processing of each individual example should therefore be quick. Additionally, in online learning the underlying distribution governing the examples can change. When it changes, we talk about concept drift (Gama et al. 2014).

To address several of the above constraints most online learning methods utilize incremental as well as decremental model updating (Cauwenberghs and Poggio 2001; Gama 2010). Incremental updating includes updating statistics, as well as growing the current predictive model. Decremental updating, on the other hand, allows for forgetting, i.e., discarding of information, when updating the current predictive model.

Orthogonally to the learning setting, in data mining tasks we also consider the level of supervision. It defines to what degree can a method receive feedback, in terms of which components of the data examples are available to the method. In *supervised learning*, all data examples are *labeled*, i.e., all of their target values are available in the learning process. Thus, a method can use these examples to compare its predictions to the true values and use this feedback to guide learning. On the other hand, in *unsupervised learning*, e.g., in clustering, there is no feedback whether the produced clusters are better than some other potential clusters.

Semi-supervised learning (Chapelle et al. 2006) lies at the midpoint between supervised and unsupervised learning. In semi-supervised learning, feedback can be obtained for some examples but not for others. In the case of MTR, this means that for some examples, we do not have access to the actual target values. These examples are called *unlabeled* examples. In many application domains, e.g., quantitive structure-activity relationship (QSAR) modeling, unlabeled examples are abundant and cheap to produce, while labeling examples can incur significant costs. For example, labeled QSAR datasets contain hundreds of compounds, while unlabeled data are easily reachable in public databases containing thousands

of chemical compounds (Levatić et al. 2013). Finally, utilizing the unlabeled examples to achieve better performance over just using labeled examples is the primary goal in semi-supervised learning.

In this paper, we present a method for online semi-supervised multi-target regression based on an incremental tree-based method for MTR (Osojnik et al. 2018), named iSOUP-Tree, and the predictive clustering framework (Blockeel and De Raedt 1998) and its applications to semi-supervised learning tasks in the batch setting (Levatić et al. 2017a, b, 2018). To the best of our knowledge, the task of online semi-supervised multi-target regression is currently largely unaddressed by the research community, with the exception of the work of Sousa and Gama (2016). They use the self-training approach to semi-supervised learning, where the method uses its own predictions in place of the missing target values for learning.

The remainder of the paper is structured as follows. In Sect. 2, we briefly present the background of this work and focus on the iSOUP-Tree method and the predictive clustering framework. Next, in Sect. 3, we present the related work in the field of online semi-supervised learning for classification and regression, and batch semi-supervised learning for multi-target regression. We continue by introducing the iSOUP-PCT and iSOUP-PCT$^{\text{SSL}}$ methods in Sect. 4. In Sect. 5, we describe our experimental design and in Sect. 6 we present the experimental results. Finally, in Sect. 7, we conclude the paper with a brief summary of contributions and directions for further work.

2 Background

In this section, we focus on a method for online incremental tree induction for the task of multi-target regression and the predictive clustering framework. Both are used as basis of our proposed tree-based method for online semi-supervised multi-target regression.

2.1 Incremental trees for MTR–iSOUP-tree

iSOUP-Tree (Osojnik et al. 2018) is a supervised incremental tree-based learner that utilizes the Hoeffding inequality (Hoeffding 1963) and a variance-reduction-based splitting heuristic. iSOUP-Trees have been used to address the MTR task (Osojnik et al. 2018), as well as the multi-label classification task (Osojnik et al. 2017), in the online learning setting.

When growing an iSOUP-Tree, we must consider several split candidates. When dealing with numeric attributes, a split $\mathcal{S}$ takes the form of a test $\mathcal{S} : x_i \leq v$, where x_i is the value of the i-th attribute and v is one of the possible values of the attribute. Each split $\mathcal{S}$ separates the set of accumulated examples S into $S_\top$ and $S_\perp$, which contain the examples for which the test is either true or false, respectively.

In particular, when we are evaluating a split, we calculate its relative value using the *intra-cluster variance reduction* (ICVR) heuristic

$$\text{ICVR}(\mathcal{S}) = \frac{1}{M}\sum_{j=1}^{M}\left(1 - \frac{|S_\top|}{|S|}\frac{\text{Var}^j(S_\top)}{\text{Var}^j(S)} - \frac{|S_\perp|}{|S|}\frac{\text{Var}^j(S_\perp)}{\text{Var}^j(S)}\right), \tag{1}$$

where j indexes the target variables, and Var^j is the variance of to the j-th target, i.e.,

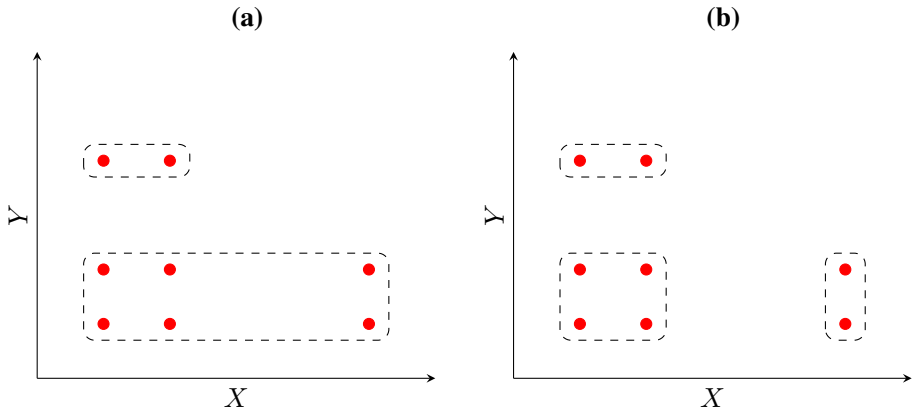

Fig. 1 The regular splitting heuristic seeks to improve the homogeneity in the target space Y (**a**). The predictive clustering heuristic additionally seeks to improve homogeneity in the input space X (**b**). Figure adapted from (Blockeel 1998)

$$\mathrm{Var}^j(S) = \frac{1}{|S|}\sum_{i=1}^{|S|}\left(y_i^j - \overline{y}^j\right)^2, \tag{2}$$

where y_i^j is the value of j-th target of i-th example and $\overline{y}^j$ is the average value of the j-th target on S. Using the Hoeffding inequality we determine whether there is sufficient probabilistic support for choosing the best split based on the ratio of the heuristics of two best splits.

Furthermore, iSOUP-Trees can learn regression trees, in which each leaf produces a prediction based on the average values of the targets over the accumulated examples, or model trees that utilize leaf models, such as perceptrons (Rosenblatt 1958).

2.2 Predictive clustering trees

Predictive clustering trees (PCTs) are a state-of-the-art method for structured output prediction in the batch setting for a wide selection of output structures (Struyf and Džeroski 2006; Vens et al. 2008; Slavkov and Džeroski 2010; Kocev et al. 2013). They utilize the predictive clustering framework (Blockeel and De Raedt 1998), which connects the tasks of predictive modeling and clustering. Briefly said, in predictive clustering all attributes are seen as part of the domain for clustering and targets for predictive modeling, as well. PCTs work under the assumption that grouping similar examples together, i.e., clustering, can improve the predictive performance of the model. They use a modified splitting heuristic that takes into account not only the homogeneity of the targets, but also the homogeneity of input attributes. An example can be seen in Fig. 1.

3 Related work

In this section, we present the related work connected to our proposed method. Here, we focus on an overview of methods for online semi-supervised learning (for the tasks of classification and regression) and methods for semi-supervised structured output prediction (SOP) in the batch case. Notably, there are currently no methods for online semi-supervised structured output prediction tasks, in particular, for multi-target regression.

Online semi-supervised learning The first incremental method that is able to utilize unlabeled examples is the multi-view hidden Markov perceptron (Brefeld et al. 2005). Furao et al. (2007) presented a method for semi-supervised learning based on a self-organizing incremental neural network, which was later extended by Shen et al. (2011).

Additionally, Goldberg et al. (2008) proposed a manifold regularization method for semi-supervised learning that is based on a combination of convex programming with stochastic gradient descent. Later, Goldberg et al. (2011) introduced OASIS, a Bayesian learning framework for semi-supervised learning.

Sousa and Gama (2016) extended the AMRules method (Duarte et al. 2016) for multi-target regression to the semi-suprevised learning setting by using the self-training approach, i.e., using a model's own predictions as proxies for the missing target values. Later, Sousa and Gama (2017) used AMRules in a similar, but slightly different, co-training approach. In co-training, multiple models are learned and examples which are used for the training of a model are labeled with the predictions of one or more of the other learned models.

Online semi-supervised classification has also been used to tackle image tracking, e.g., by Grabner et al. (2008) and Zeisl et al. (2010).

Batch semi-supervised methods for specific tasks Self-training methods use their own predictions of the unlabeled examples as training values in specific circumstances. Zhang and Yeung (2009) and Cardona et al. (2015) introduced methods based on Gaussian processes for semi-supervised multi-task learning, a task related to multi-target regression. Navaratnam et al. (2007) also introduced a Gaussian process based method for semi-supervised multi-target regression, though their approach is aimed specifically at applications in computer vision. Levatić et al. (2017b) introduced a semi-supervised method for multi-target regression based on self-training of random forests of predictive clustering trees.

Generally applicable semi-supervised methods Altun et al. (2006) and Li and Zemel (2014) introduced SVM-like methods based on the maximum-margin approach that can address multiple SOP tasks. Brefeld and Scheffer (2006) used support vector machines (SVMs) in a co-training approach and applied the principle of maximal consensus between independent hypotheses. Zien et al. (2007) introduced transductive SVMs for structured output prediction tasks. Notably, their approach addresses transductive semi-supervised learning, where the goal is not to produce a reusable predictive model, but instead to provide the predictions for specific unlabeled examples.

Gönen and Kaski (2014) and Brouard et al. (2016) proposed methods that can be applied to both nominal and continuous structured output prediction tasks, with kernelized Bayesian matrix factorization and input output kernel regression, respectively.

Finally, Levatić et al. (2017a, 2018) introduced an inductive method for semi-supervised structured output prediction based on predictive clustering trees. The unlabeled examples are considered when the tree is grown by impacting the splitting heuristic which takes into account the homogeneity of the input attributes in addition to the homogeneity of the output attributes.

4 Incremental predictive clustering trees—iSOUP-PCT and iSOUP-PCT^{SSL}

To address online semi-supervised SOP tasks, we utilize the predictive clustering framework (Blockeel and De Raedt 1998; Struyf and Džeroski 2006). The predictive clustering frameworks was successfully used by Levatić et al. (2018) to address the semi-supervised structured output prediction in the batch setting. To achieve this goal, we extended the iSOUP-Trees (Osojnik et al. 2018) towards predictive clustering trees in an online setting and adapted them to handle both labeled and unlabeled data.

In this paper, we focus on online semi-supervised multi-target regression, as an instance of online semi-supervised SOP task. Specifically, we adapt the iSOUP-Tree method for online MTR toward semi-supervised MTR in three ways. First, we modify the splitting heuristic to take into account the homogeneity of the input attributes in addition to the homogeneity of the targets. Second, we adapt the extended binary search trees (E-BST, (Ikonomovska et al. 2011)) that collect the statistics required to calculate the ICVRheuristics to also record the values of the input attributes. Finally, we modify the initialization of the leaf averages after splitting to use as much information as possible from the observed labeled examples. The resulting method is called iSOUP-PCT and is implemented in the Massive Online Analysis (MOA, (Bifet et al. 2010)) framework.[1]

4.1 Predictive clustering splitting heuristic

To take into account the homogeneity of the input attributes in the splitting heuristic, we modify the standard intra-cluster variance reduction definition (Levatić et al. 2018) to include the variance of the input attributes

$$\begin{aligned}\mathrm{ICVR}(\mathcal{S}) = w \cdot \frac{1}{M}\sum_{j=1}^{M}\left(1 - \frac{|S_\top|}{|S|}\frac{\mathrm{Var}^j(S_\top)}{\mathrm{Var}^j(S)} - \frac{|S_\perp|}{|S|}\frac{\mathrm{Var}^j(S_\perp)}{\mathrm{Var}^j(S)}\right)\\ + (1-w)\cdot\frac{1}{N}\sum_{i=1}^{N}\left(1 - \frac{|S_\top|}{|S|}\frac{\mathrm{Var}_i(S_\top)}{\mathrm{Var}_i(S)} - \frac{|S_\perp|}{|S|}\frac{\mathrm{Var}_i(S_\perp)}{\mathrm{Var}_i(S)}\right),\end{aligned}$$

where $w \in [0, 1]$ is the *level of supervision* and $\mathrm{Var}_i(S)$ is the variance of the values of the i-th input attribute over set S. To the best of our knowledge no other methods for online prediction (let alone online multi-traget regression) evaluate both the homogeneity of the targets and of the input attributes when learning.

When $w = 1$ the above definition coincides with the regular definition of intra-cluster variance reduction. When $w = 0$, we consider the input attributes exclusively, when estimating the homogeneity of the different split subsets. In essence instead of trying to minimize an evaluation measure, we are grouping the examples according to their similarity, which is equivalent to solving the clustering data mining tasks. In the batch learning approach, the w parameter is chosen based on internal cross-validation that determines the appropriate w for each individual dataset. In the online scenario, this procedure is unfeasible. Thus, we select a midpoint value of $w = 0.5$.

[1] The particular implementation is available at https://github.com/aosojnik/moa.

4.2 Maintaining the required statistics

To calculate the variance of an input attribute/target, we require the number of observed examples k, the sum of values of the attribute/target Σ, and the sum of squared values of the attribute/target Σ^2. Variance is then calculated as

$$\text{Var} = \frac{1}{k}\left(\Sigma^2 - \frac{1}{k}(\Sigma)^2\right).$$

In regular iSOUP-Trees, we keep the k, Σ and Σ^2 statistics only for each target. They are stored in an extended binary search tree (E-BST), which holds the statistics given different potential split values. One E-BST is kept for each input attribute.

One way of understanding the predictive clustering framework, is that all of the attributes, input attributes and targets alike, act as targets. In this way, we extend the E-BST tree structure to hold the statistics k, Σ and Σ^2 for input attributes as well as targets. As splits can still only occur on input attributes the number of E-BST structures we keep remains the same.

This modification, however, incurs a heavy cost on the consumption of resources. In a regular E-BST, we use $\mathcal{O}(n \cdot M)$ units of memory to record the statistics, where n is the number of unique attribute values recorded in the tree and M is the number of targets. In the modified E-BST, however, we use $\mathcal{O}(n(N+M))$ units of memory, where N is the number of input attributes, since we also store the statisics of the input attributes in addition to those of the targets.

Given that we keep one E-BST for each input attribute, this increases the total memory complexity of the method from $\mathcal{O}(n_{\max}MN)$ to $\mathcal{O}(n_{\max}(N+M)N)$ where $n_{\max} = \max\{\text{number of distinct values of attribute } A\}$ over all attributes A. Therefore, in problems where there are strict constraints on memory consumption, we must be careful in applying iSOUP-PCTs, especially, when the number of input attributes is large, as the complexity is quadratic in the number of attributes.

Notably, this issue does not appear in the batch case. As all examples are available throughout the learning process, there is no additional need to record any kind of statistics as they can be calculated on the fly. This has propelled semi-supervised PCTs to great performances for SSL data mining tasks (Levatić et al. 2017a).

4.3 Initializing leaf models

We apply another adaptation in the initialization of new leaf models, when splitting a leaf. In particular for semi-supervised learning, we do not recommend the use of leaf models such as the perceptron. Usually, when a leaf node is split, the linear coefficients of the perceptron in the split leaf are copied to the initial perceptrons of the new leaves. However, the perceptron updating procedure only works on labeled examples and in streams with a few labeled examples, it takes a while for the two new perceptrons to produce substantially different predictions.

To this end, we use the mean regressor in the leaves, which predicts for each target the average value of the corresponding observed target values of the accumulate examples. It too, however, requires labeled examples to start accurately modeling the different subspaces resulting from the split. To facilitate faster learning, we harness the splitting procedure. To evaluate various possible splits, we calculate the pre- and post-split variances.

In particular we have access to the splitting heuristics of both splitting subsets $S_\top$ and $S_\perp$. The split we select, is by definition the one that produces the most homogeneous splitting subsets. For each leaf corresponding to a splitting subset, we set the appropriate counts and sums used by the mean predictor to k and Σ. These mean predictors utilize the past (labeled) examples to the fullest extent.

4.4 iSOUP-PCT$^{\text{SSL}}$

The iSOUP-PCT method can be used both in a supervised and semi-supervised fashion. When we are using iSOUP-PCTs as a semi-supervised method, i.e., when we learn from unlabeled examples in addition to the labeled ones, we name the method $iSOUP-PCT^{\text{SSL}}$. iSOUP-PCT is then the variant of the method that learns only from the labeled examples, while ignoring the unlabeled ones.

5 Experimental design

The main goal of this paper is to determine whether and by how much does the use of incremental predictive clustering trees increase the performance in the online semi-supervised multi-target learning setting. In this section, we present the experimental questions, the datasets and evaluation measures used in the experiments and the procedure for validating the obtained models.

5.1 Methods

We compare the predictive performance of three methods, iSOUP-Tree, iSOUP-PCT and $iSOUP-PCT^{\text{SSL}}$, to an oracle method $iSOUP-Tree^{\text{Oracle}}$ (or, for brevity, Tree, PCT, PCT^{SSL} and Oracle). iSOUP-PCT is a supervised learner, i.e., it learns only from labeled examples. On the other hand, $iSOUP-PCT^{\text{SSL}}$ is a semi-supervised learner that learns from all examples, both labeled and unlabeled. The $iSOUP-Tree^{\text{Oracle}}$ method is an "oracle" method, which means that it learns from all examples as though they were labeled, i.e., it also has access to the target values even for the unlabeled examples. The oracle trees show a practical limit of how much can be learned in a semi-supervised scenario with this kind of method. $AMRules^{\text{SSL}}$ (Sousa and Gama 2016) is the semi-supervised implementation of AMRules which uses self-training to utilize unlabeled examples, i.e., the method's predictions are used as proxies for the missing target values.

The major difference between $iSOUP-PCT^{\text{SSL}}$ and $AMRules^{\text{SSL}}$, other than the obvious use of a different class of model (trees vs. rules), is the approach taken for processing of the unlabeled examples. $AMRules^{\text{SSL}}$ uses the *self-training approach*, i.e., the method predicts the target values of the unlabeled examples, which are then artificially labeled with these predictions. These self-labeled examples are then fed back into the method's learning process.

On the other hand, $iSOUP-PCT^{\text{SSL}}$ uses the *predictive clustering approach* of maximizing the homogeneity of the input space in addition to that of the target space. In our case, unlabeled examples contribute statistics which impact the estimate of homogeneity of the input space while having no impact on the estimate of homogeneity of the target space, since the target values are unknown, whereas labeled examples contribute to both estimates of homogeneity.

Table 1 Datasets used in the online multi-target regression experiments

Dataset	No. of examples	Attributes	T
Bicycles	17,379	12 numeric	3
EUNITE03	8064	29 numeric	5
Forestry Kras	60,607	160 numeric	11
Forestry Slivnica	6218	150 numeric	2
RF1	9005	64 numeric	8
RF2	7679	576 numeric	8
SCM1d	9803	280 numeric	16
SCM20d	8966	61 numeric	16

T number of targets

5.2 Experimental questions

In this paper, we are interested in how the predictive performance depends on the ratio of labeled examples. This shows us whether, at some ratios of labeled to unlabeled examples, it is even advisable to use semi-supervised methods. In the online setting, in particular, it might not be problematic if only each second example is labeled as we can expect a large number of examples. When only each tenth example is labeled, however, the learning rate, i.e., tree growth, could be too slow to accurately model the incoming data.

Additionally, we are interested in how the proposed method compares to the state-of-the-art in this field. Notably, to the best of our knowledge, AMRules (Sousa and Gama 2016) is the only other method that is capable of addressing the online semi-supervised multi-target regression task.

Finally, we are interested in how much memory the compared methods consume. As discussed above in Sect. 4, we expect that incremental predictive clustering trees (iSOUP-PCT and iSOUP-PCT^{SSL}) will consume considerably more memory.

5.3 Datasets

For the online multi-target regression experiments, we have selected 8 datasets, based on their size, looking for diversity in the number of input and target attributes. We consider the datasets under the assumption of no concept drift, given that these datasets are usually considered as benchmark datasets in the batch setting. A summary of the datasets and their properties is shown in Table 1.

The *Bicycles* dataset is concerned with the prediction of demand for rental bicycles on an hourly basis (Fanaee-T and Gama 2013). The 3 targets are the number of casual (non-registered) users, the number of registered users and the total number of users for a given hour, respectively.

The *EUNITE03* dataset was part of the competition at the 3rd European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems.[2] The data describes a complex process of continuous manufacturing of glass products, i.e., the input attributes describe various influences, while the 5 targets describe the glass quality.

[2] http://www.eunite.org/eunite/news/Summary%20Competition.pdf. Accessed 22 Jan 2018.

The data in the *Forestry Kras* dataset was derived from multi-spectral multi-temporal Landsat satellite images and 3D LiDAR recordings of a part of the Kras region in Slovenia (Stojanova et al. 2010). Each example corresponds to a spatial unit, i.e., an area of 25 by 25 meters. For each example, the input attributes and targets were derived from the LandSat and LiDAR recordings of the spatial unit. For specifics on the data preparation procedure, see (Stojanova et al. 2010). The task is to predict 11 targets, which correspond to properties of the vegetation in the observed spatial unit.

The *Forestry Slivnica* dataset was, as in the previous case, constructed from multi-spectral multi-temporal Landsat satellite images and 3D LiDAR recordings of a part of the Slivnica region in Slovenia (Stojanova 2009). In this dataset, the task is to predict only 2 target variables: vegetation height and canopy cover.

The river flow datasets, *RF1* and *RF2*, concern the prediction of river network flows for 48 h at 8 locations on the Mississippi River network (Spyromitros- et al. 2012). Each data example comprises observations for each of the 8 locations at a given time point, as well as time-lagged observations from 6, 12, 18, 24, 36, 48 and 60 h in the past. In RF1, each location contributes 8 input attributes, for a total of 64 input attributes and 8 target variables. The RF2 dataset extends RF1 with the precipitation forecast information for each of the 8 locations and 19 other meteorological sites. Specifically, the precipitation forecast for 6 h windows up to 48 h in the future is added, which nets a total of 280 input attributes.

The *SCM1d* and *SCM20d* are datasets derived form the Trading Agent Competition in Supply Chain Management (TAC SCM) conducted in 2010. The dataset preparation is described by Spyromitros- et al. (2012). The data examples correspond to daily updates in a tournament—there are 220 days in each game and 18 games per tournament. The 16 targets are the predictions of the next day and the 20 day mean price for each of the 16 products in the simulation, for the SCM1d and SCM20d datasets, respectively.

The Bicycles dataset is available at the UCI Machine Learning Repository[3] and the RF1, RF2, SCM1d and SCM20d datasets are available at the Mulan multi-target regression dataset repository.[4] The examples with missing values (on some input attributes) in the RF1 and RF2 datasets were removed, so the resulting datasets were somewhat smaller than reported in the repository.

5.4 Evaluation measures

To evaluate the predictive performance of predictive models for multi-target regression, we define the *average relative mean absolute error* ($\overline{\text{RMAE}}$) measure on an evaluation sample as

$$\overline{\text{RMAE}} = \frac{1}{M}\sum_{j=1}^{M}\frac{\sum_{i=1}^{n}\left|y_i^j - \hat{y}_i^j\right|}{\sum_{i=1}^{n}\left|y_i^j - \overline{y}^j(i)\right|},$$

where y_i^j is a true value of the target j for example i, $\hat{y}_i^j$ are the predictions of the evaluated model and $\overline{y}^j(i)$ is the value predicted by the j-th mean regressor for the i-th example. As we will be using the prequential evaluation approach, each prediction $\hat{y}_i$ will have been

[3] https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset. Accessed 22 Jan 2018.
[4] http://mulan.sourceforge.net/datasets-mtr.html. Accessed 22 Jan 2018.

made using knowledge of only prior examples. We use the mean regressor in place of the actual mean of the sample, to approximate the same setting. To avoid the problem for the first example, we take the average to be the prediction of the mean regressor based on all prior examples. Finally, the $\overline{\text{RMAE}}$of a perfect regressor is 0, and lower values of the error are desired (↓).

In this paper, we evaluate the computational efficiency of a method based on how much memory it consumes to learn and make predictions on the provided dataset. We use the tools available in the MOA framework (Bifet et al. 2010) to directly measure the amount of memory in megabytes that the model uses.

5.5 Experimental setup

Prequential evaluation In the online setting, there is no clear distinction between training and testing phases and evaluation approaches are designed to be continuous to keep pace with the online learning procedure. Due to the real-time nature of online learning, we are interested in how the model performs throughout the learning procedure, at each timepoint. Hence, we use each example to both test and train the model. To avoid introducing bias into the evaluation procedure, we always test each model on an example before we learn from it. *Predictive sequential (prequential) evaluation* (Dawid 1984) uses an interleaved test-then-train approach. An example is used for training immediately after recording its prediction.

Artificially unlabeled examples In batch semi-supervised learning, the main goal is to utilize the cheap and abundant collected data, which is not labeled, to improve the predictive performance of a predictive model. As these examples are unlabeled, we can also not calculate any evaluation measures for them. However, when we evaluating methods for semi-supervised tasks, we often artificially unlabel examples. In this case, our main interest is the predictive performance of the model on only the remaining labeled examples, even though, in this setting it is possible to calculate the predictive performance on the unlabeled examples as well.

In an online semi-supervised learning setting, we view the unlabeled examples differently. The prequential evaluation approach closely follows the natural streaming process with examples arriving unlabeled, after which a predictive model produces its predictions. Later on in the stream, the examples may arrive labeled and get used for the purpose of learning.

To compare machine learning methods for online semi-supervised predictive modeling, we use the same artificial unlabeling procedure as in the batch setting (Levatić et al. 2017a). However, we record the predictions on the unlabeled examples and use both the predictions on labeled and unlabeled examples to calculate the evaluation measures. Thus, we use measures appropriate for the corresponding supervised task.

Labeling ratio In our experiments, we observe three labeling ratios κ, generated by the following probabilities of labeling, 0.1, 0.2 and 0.5. This yields three variants of each of the multi-target regression datasets shown in Table 1. Specifically, for a given *labeling ratio* κ, we unlabel each example, except for the first two, with probability $1 - \kappa$. The first two examples are always labeled, to allow the predictive models to properly initialize. To account for the randomness of the unlabeling procedure, we repeat each experiment for a given dataset and a given labeling ratio 10 times.

Calculation of the performance measures Unlike the batch setting, where we are predominantly interested in the performance on labeled examples, we treat both the

(artificially) unlabeled and the labeled examples equally. Therefore, we calculate the $\overline{\text{RMAE}}$evaluation measure for each example on each of the repeated dataset, and record their average value once every 1000 examples, i.e., we are averaging over the last 1000 examples in each of the 10 repetitions of a dataset. When calculating the $\overline{\text{RMAE}}$, we are comparing to the oracle mean regressor, i.e., the average values we divide by are calculated on averages of all target values, even on the examples that were artificially unlabeled. We present the obtained $\overline{\text{RMAE}}$values as plots to qualitatively compare the observed methods, as well as the final values of the $\overline{\text{RMAE}}$once the methods had processed all of the examples.

We record the final memory consumption of the observed methods, i.e., the consumed memory after all examples have been processed. To show the evolution of the memory consumption of the observed methods, we also record and plot the memory consumption at intervals of 1000 examples for select datasets.

Table 2 The predictive performance in terms of $\overline{\text{RMAE}}$ (↓) of the compared methods

	Tree	PCT	PCT^{SSL}	Oracle	$AMRules^{\text{SSL}}$
Results for $\kappa = 0.1$					
Bicycles	0.7826	0.8836	**0.6713**	0.6024	0.8218
EUNITE03	1.0060	0.9997	**0.9432**	0.9372	0.9465
Forestry Kras	0.8747	0.8883	**0.7960**	0.7649	1.0002
Forestry Slivnica	0.9636	0.9636	**0.9210**	0.8685	1.0069
RF1	0.9958	0.9848	**0.8201**	0.9965	0.9790
RF2	1.0090	1.0090	**0.9495**	1.0551	1.0019
SCM1d	0.9831	0.9855	**0.9480**	0.9050	1.0081
SCM20d	1.0047	1.0136	**0.9701**	0.9786	1.0130
Results for $\kappa = 0.2$					
Bicycles	0.7542	0.7844	**0.6373**	0.6024	0.7173
EUNITE03	0.9967	0.9795	0.9419	0.9372	**0.7259**
Forestry Kras	0.8339	0.8373	**0.7860**	0.7649	0.8911
Forestry Slivnica	0.9371	0.9453	**0.9059**	0.8685	1.0028
RF1	0.9489	0.9444	0.8401	0.9965	**0.8139**
RF2	1.0197	1.0175	**0.9343**	1.0551	0.9983
SCM1d	0.9634	0.9931	**0.9290**	0.9050	1.0041
SCM20d	0.9984	1.0069	0.9701	0.9786	**0.9600**
Results for $\kappa = 0.5$					
Bicycles	0.6624	0.7427	**0.6251**	0.6024	0.6781
EUNITE03	0.9743	0.9699	0.9514	0.9372	**0.6141**
Forestry Kras	0.8079	0.8086	0.7896	0.7649	**0.7656**
Forestry Slivnica	0.9063	0.9230	**0.8994**	0.8685	1.0298
RF1	0.9071	0.9179	0.8663	0.9965	**0.6946**
RF2	0.9626	0.9706	**0.9645**	1.0551	0.9654
SCM1d	0.9472	0.9408	**0.9121**	0.9050	0.9804
SCM20d	1.0099	0.9960	0.9702	0.9786	**0.6532**

The best performance for each dataset is shown in bold

6 Results and discussion

6.1 Predictive performance

The results of the online semi-supervised multi-target regression experiments are shown in Table 2 and Fig. 2.

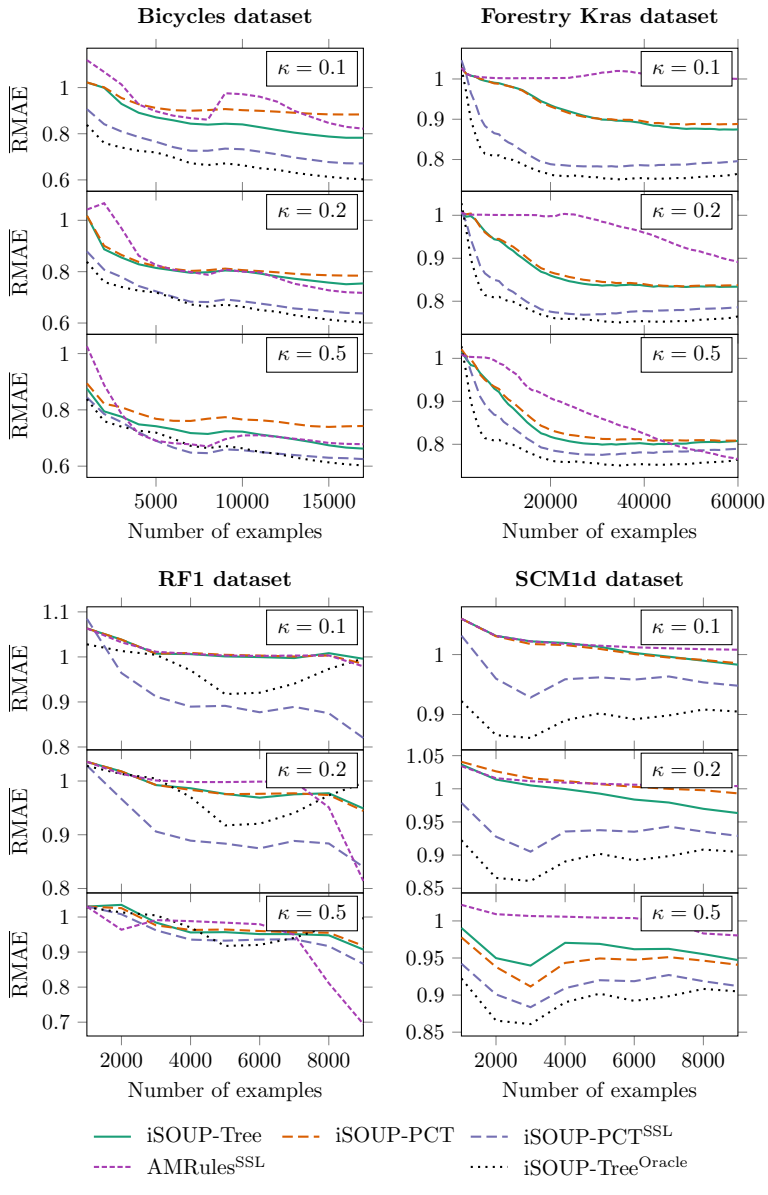

Fig. 2 The evolution of $\overline{\text{RMAE}}(\downarrow)$ as more examples are processed on the Bicycles, Forestry Kras, RF1 and SCM1d datasets

Table 2 shows that on all datasets and for all labeling ratios iSOUP-PCT^{SSL} achieves better predictive performance than both the unsupervised iSOUP-Tree and iSOUP-PCT. As expected, on most datasets iSOUP-PCT^{SSL} is worse than the iSOUP-Tree^{Oracle}, i.e., the regression tree that learns from the entire dataset as though it was labeled. Curiously, on the RF1, RF2 and SCM20d dataset iSOUP-PCT^{SSL} performs even better than the oracle method. On the RF2 dataset, even iSOUP-Tree and iSOUP-PCT outperform the oracle tree. This very likely means that mean regressors are not good as leaf models on the RF1 and RF2 datasets, as learning from more examples decreases the performance.

Figure 2 shows the evolution of $\overline{\text{RMAE}}$as more examples are processed on four of the datasets: Bicycles, Forestry Kras, RF1 and SCM1d. The results on the remaining four datasets are similar and are omitted for brevity. As we expected, the differences between the semi-supervised models and regular, supervised models decrease as we go from low to high labeling ratios. As the supervised models see more and more labeled examples, the benefits of using the semi-supervised iSOUP-PCT method are reduced. In the extreme, where all examples are labeled, the iSOUP-Tree and iSOUP-Tree^{Oracle} would coincide, as would iSOUP-PCT and iSOUP-PCT^{SSL}.

Table 3 The consumption of memory in terms of megabytes of the compared methods

	Tree	PCT	*PCT*^{SSL}	Oracle	*AMRules*^{SSL}
$\kappa = 0.1$					
Bicycles	0.20	0.38	3.49	1.29	0.58
EUNITE03	1.54	6.86	26.71	5.62	1.54
Forestry Kras	51.69	1000.27	2879.21	203.41	30.60
Forestry Slivnica	9.10	265.36	812.26	14.42	12.88
RF1	2.95	24.03	265.00	10.05	5.37
RF2	14.79	1006.47	745.31	13.04	10.10
SCM1d	73.11	1282.54	17,150.37	659.56	46.13
SCM20d	32.16	141.96	518.00	163.20	9.99
$\kappa = 0.2$					
Bicycles	0.22	1.17	2.71	1.29	0.61
EUNITE03	1.82	6.13	19.94	5.62	1.59
Forestry Kras	51.34	900.19	2205.68	203.41	37.22
Forestry Slivnica	12.32	425.25	1222.74	14.42	13.29
RF1	19.48	160.24	282.02	10.05	5.29
RF2	10.08	704.55	2086.02	13.04	10.24
SCM1d	184.95	4471.63	17,526.12	659.56	46.23
SCM20d	23.43	207.23	661.15	163.20	9.99
$\kappa = 0.5$					
Bicycles	0.73	1.69	3.57	1.29	0.72
EUNITE03	1.87	14.03	31.74	5.62	1.62
Forestry Kras	88.25	1644.26	2351.13	203.41	64.60
Forestry Slivnica	27.72	570.69	2285.14	14.42	13.70
RF1	3.92	81.46	81.78	10.05	8.81
RF2	21.23	1478.65	1008.31	13.04	9.65
SCM1d	599.33	7317.71	15,351.75	659.56	91.00
SCM20d	80.57	369.19	626.74	163.20	19.55

The comparison of the results of iSOUP-PCT^{SSL} and AMRules^{SSL} is also shown in Table 2, where the better method is highlighted in strong text. In the case of $\kappa = 0.1$, iSOUP-PCT^{SSL} outperforms AMRules^{SSL} on all datasets. In the remaining cases, i.e., $\kappa = 0.2$ and $\kappa = 0.5$, the results are not as clear cut, with both methods winning out over the other on about half of the datasets. Looking at the evolution of RMAEof AMRules^{SSL} in Fig. 2, we see that its performance is quite inconsistent and very dependent on the dataset.

6.2 Memory consumption

From Table 3, we clearly see that the iSOUP-PCT and iSOUP-PCT^{SSL} methods consume significantly more memory than iSOUP-Tree and iSOUP-Tree^{Oracle}, respectively. This is according to our expectations, based on the earlier analysis of memory complexity, presented in Sect. 4. Furthermore, the relative increases from regular trees to predictive clustering trees are consistent with the theoretical analysis presented earlier, i.e., they are larger for datasets with more input attributes.

In Fig. 3, we show the evolution of memory consumption on the Bicycles and SCM1d datasets, which represent the cases with the lowest and highest memory consumptions at $\kappa = 0.1$, respectively. In the case of the Bicycles dataset, both iSOUP-Tree and iSOUP-PCT receive only around 1600 labeled examples, and thus their memory consumption grows very slowly. On the other hand, the memory consumption of iSOUP-Tree^{Oracle} and

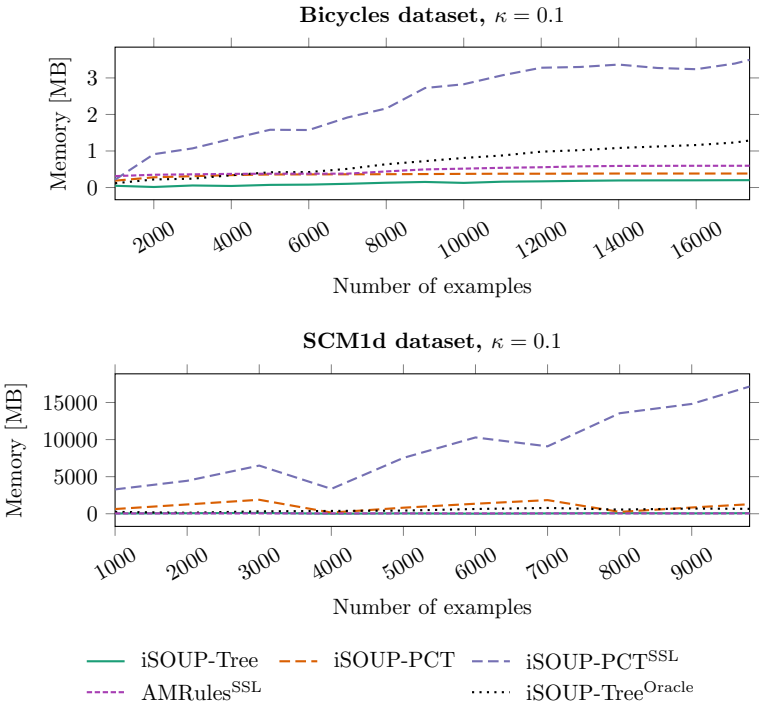

Fig. 3 The evolution of memory consumption as examples are processed for the Bicycles and SCM1d datasets when $\kappa = 0.1$

iSOUP-PCT^{SSL} show a faster rate of growth, given that they process all examples, labeled and unlabeled alike. On the SCM1d dataset, we see large decreases in the memory consumption of both predictive clustering tree methods. These result from the decremental forgetting of superfluous statistics when leaves are split. In particular, all E-BST tree structures are removed once a leaf is split and new, empty ones are initialized in the new leaves. These patterns are also observed for the iSOUP-Tree methods, however, due to the considerably higher memory consumption of the PCT methods, they are not seen at the shown scale.

The large consumption of memory of iSOUP-PCT^{SSL} is a natural consequence of the space complexity of the statistics storage. As discuessed earlier in Sect. 4.2, the semi-supervised method is quadratic in the number of input attributes, which can result in considerable memory consumption. The extremely large consumption of memory of iSOUP-PCT^{SSL} on the SCM1d dataset is a combination of a large number of input attributes as well as an extremely large number of unique attribute values, i.e., n_{max}, defined above, is very high.

Both Table 3 and Fig. 3 show that AMRules^{SSL} convincingly outperforms iSOUP-PCT^{SSL} in terms of memory consuption. This is expected as AMRules^{SSL} keeps a set of rules of fixed size. On the other hand, the size of the iSOUP-PCT^{SSL} is not constrained resulting in considerably larger consumption of memory.

6.3 Discussion

An important aspect of using iSOUP-PCTs is the trade-off between the use of additional resources and the achieved predictive performance. iSOUP-PCT^{SSL}s utilize the unlabeled examples to achieve better predictive performance, especially when there are few labeled examples. However, the improvement in predictive performance might come at such a high increase in memory consumption to make it unfeasible. In particular, this makes them uncompetitive in the supervised scenario.

iSOUP-PCTs might also suffer from slower growth than regular iSOUP-Trees, as we are averaging additional values in the PCT heuristic, resulting in split candidates that have closer heuristic scores. While the iSOUP-PCT^{SSL}s get more examples to calculate the heuristic values, the supervised iSOUP-PCTs receive the same labeled examples as iSOUP-Trees and generally perform slightly worse. The benefit of the semi-supervised approach thus appears to entirely stem from the fact that it can utilize unlabeled examples.

The comparison of iSOUP-PCT^{SSL} and AMRules^{SSL} is not quite straightforward, with the exception of the $\kappa = 0.1$ scenario. In that case, iSOUP-PCT^{SSL} outperforms AMRules^{SSL} on all of the observed datasets. In the remaining scenarios, it is not possible to determine a clear winner, as the performance of the two methods is comparable. We note, however, that AMRules^{SSL} does use a smaller amount of memory.

7 Conclusions and further work

In this paper, we have extended the incremental tree-based method iSOUP-Tree toward the predictive clustering framework and introduced the iSOUP-PCT method for online semi-supervised multi-target regression. We have evaluated the introduced method in both supervised (iSOUP-PCT) and semi-supervised (iSOUP-PCT^{SSL}) scenarios. We have found that the use of iSOUP-PCTs is *not* warranted when there are no unlabeled

examples, i.e., in the supervised scenario. In that case, iSOUP-PCTs consume more memory, while producing slightly worse results than iSOUP-Trees. On other hand, in the semi-supervised scenario, and in particular, when there is a considerable amount of unlabeled examples, iSOUP-PCT^{SSL} outperforms regular iSOUP-Tree, achieving predictive performance close to that of an oracle method, which has access to the target values of even the unlabeled examples.

However, the consumption of memory of iSOUP-PCTs considerably higher than that of iSOUP-Trees. Thus, we must consider that using iSOUP-PCTs comes at a considerable cost of computational resources. We should evaluate whether their use is warranted for each particular application scenario.

Comparing the proposed method with the current state-of-the-art, shows us that its performace is at least as good as that of the singular state-of-the-art method. In particular, when there are very few labeled examples (which might translate into an application where the labelling is particularly cumbersome or expensive), the propsed method beats the state-of-the-art on all observed datasets. The memory consuption, however, as in the case of the other experimental questions, remains a considerable problem for the proposed method.

We identify several avenues for further work. Given that the main concern related to the iSOUP-PCTs is the consumption of memory, we plan to test different techniques to reduce memory usage. In particular, using iSOUP-PCTs with the online random forest methodology (Oza 2005), where only a subset of input attributes are considered at each tree node, could prove beneficial. Random forests thrive on a large number of input attributes, which is, on the other hand, the major source of computational complexity of iSOUP-PCT. Furthermore, in this paper we have exclusively used regression trees, as leaf models that are commonly used (typically linear) are unable to learn from unlabeled examples. We plan to explore other kinds of leaf models that could utilize the unlabeled examples. Finally, we intend to extend semi-supervised iSOUP-PCTs to other online SOP tasks, such as multi-label classification, using problem transformation approaches as in Osojnik et al. (2017).

Acknowledgements This work is supported by grants funded by the Slovenian Research Agency (P2-0103, J2-2505, J2-9230) and grants funded by the European Commission (H2020 952215 TAILOR, H2020 825619 AI4EU, H2020 769661 SAAM, H2020 833671 RESILOC).

References

Altun, Y., McAllester, D., & Belkin, M. (2006). Maximum margin semi-supervised learning for structured variables. In *Advances in neural information processing systems 18 (NIPS 2005)*, NIPS Foundation, pp. 33–40.

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: Massive online analysis. *Journal of Machine Learning Research*, *11*(May), 1601–1604.

Blockeel, H. (1998). Top-down induction of first-order logical decision trees. Ph.D. thesis, Katholieke Universiteit Leuven, Leuven, Belgium

Blockeel, H., & De Raedt, L. (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*, *101*(1), 285–297. https://doi.org/10.1016/S0004-3702(98)00034-4.

Brefeld, U., & Scheffer, T. (2006). Semi-supervised learning for structured output variables. In *Proceedings of the 23rd international conference on machine learning (ICML 2006)*, ACM, pp. 145–152. https://doi.org/10.1145/1143844.1143863

Brefeld, U., Büscher, C., & Scheffer, T. (2005). Multi-view discriminative sequential learning. In *Machine learning: ECML 2005, LNCS, vol 3720*, Springer, pp. 60–71, https://doi.org/10.1007/11564096_11.

Brouard, C., Szafranski, M., & d'Alché Buc, F. (2016). Input output kernel regression: Supervised and semi-supervised structured output prediction with operator-valued kernels. *Journal of Machine Learning Research*, *17*(176), 1–48.

Cardona, H. D. V., Álvarez, M. A., & Orozco, Á. A. (2015). Convolved multi-output Gaussian processes for semi-supervised learning. In *Image analysis and processing—ICIAP 2015, LNCS, vol 9279*, Springer, pp. 109–118. https://doi.org/10.1007/978-3-319-23231-7_10.

Cauwenberghs, G., & Poggio, T. (2001). Incremental and decremental support vector machine learning. In *Advances in neural information processing systems 13 (NIPS 2000)*, NIPS Foundation, pp. 409–415.

Chapelle, O., Schlkopf, B., & Zien, A. (2006). Semi-supervised learning. MIT Press. https://doi.org/10.7551/mitpress/9780262033589.001.0001.

Dawid, A. P. (1984). Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society Series A (General)*, *147*(2), 278–292. https://doi.org/10.2307/2981683.

Duarte, J., Gama, J., & Bifet, A. (2016). Adaptive model rules from high-speed data streams. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *10*(3), 30. https://doi.org/10.1145/2829955.

Fanaee-T, H., & Gama, J. (2013). Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, *2*(2–3), 113–127. https://doi.org/10.1007/s13748-013-0040-3.

Furao, S., Sakurai, K., Kamiya, Y., Hasegawa, O. (2007). An online semi-supervised active learning algorithm with self-organiing incremental neural network. In *Proceedings of the 2007 international joint conference on neural networks (IJCNN 2007)*, IEEE, pp. 1139–1144. https://doi.org/10.1109/ijcnn.2007.4371118.

Gama, J. (2010). *Knowledge discovery from data streams*. Boca Raton: CRC Press.

Gama, J., Žliobaite, I., Bifet, A., Pechenizkiy, M., Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys (CSUR) 46*(4):44:1–44:37.https://doi.org/10.1145/2523813.

Goldberg, A. B., Li, M., & Zhu, X. (2008). Online manifold regularization: A new learning setting and empirical study. In *Machine learning and knowledge discovery in databases (ECML PKDD 2008)*, Springer, LNCS, vol. 5211, pp. 393–407. https://doi.org/10.1007/978-3-540-87479-9_44.

Goldberg, A. B., Zhu, X., Furger, A., Xu, J. M. (2011). OASIS: Online active semi-supervised learning. In *Proceedings of the twenty-fifth AAAI conference on artificial intelligence*, AAAI, pp. 362–367.

Gönen, M., & Kaski, S. (2014). Kernelized Bayesian matrix factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(10), 2047–2060. https://doi.org/10.1109/tpami.2014.2313125.

Grabner, H., Leistner, C., & Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. In *Computer vision—ECCV 2008*, Springer, LNCS, vol 5302, pp. 234–247. https://doi.org/10.1007/978-3-540-88682-2_19.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, *58*(301), 13–30. https://doi.org/10.2307/2282952.

Ikonomovska, E., Gama, J., & Džeroski, S. (2011). Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery*, *23*(1), 128–168. https://doi.org/10.1007/s10618-010-0201-y.

Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2013). Tree ensembles for predicting structured outputs. *Pattern Recognition*, *46*(3), 817–833. https://doi.org/10.1016/j.patcog.2012.09.023.

Levatić, J., Džeroski, S., Supek, F., & Šmuc, T. (2013). Semi-supervised learning for quantitive structure-activity modeling. *Informatica*, *37*, 173–179.

Levatić, J., Ceci, M., Kocev, D., & Džeroski, S. (2017a). Semi-supervised classification trees. *Journal of Intelligent Information Systems*, *49*(3), 461–486. https://doi.org/10.1007/s10844-017-0457-4.

Levatić, J., Ceci, M., Kocev, D., & Džeroski, S. (2017b). Self-training for multi-target regression with tree ensembles. *Knowledge-Based Systems*, *123*, 41–60. https://doi.org/10.1016/j.knosys.2017.02.014.

Levatić, J., Kocev, D., Ceci, M., & Džeroski, S. (2018). Semi-supervised trees for multi-target regression. *Information Sciences*, *450*, 109–127. https://doi.org/10.1016/j.ins.2018.03.033.

Li, Y., & Zemel, R. (2014). High order regularization for semi-supervised learning of structured output problems. In *Proceedings of the 31st international conference on machine learning (ICML 2014)*, PMLR, vol. 32, pp. 1368–1376.

Navaratnam, R., Fitzgibbon, A. W., & Cipolla, R. (2007). The joint manifold model for semi-supervised multi-valued regression. In *Proceedings of the 11th IEEE international conference on computer vision (ICCV 2007)*, IEEE, pp. 1–8. https://doi.org/10.1109/iccv.2007.4408976.

Osojnik, A., Panov, P., & Džeroski, S. (2017). Multi-label classification via multi-target regression on data streams. *Machine Learning*, *106*(6), 745–770. https://doi.org/10.1007/s10994-016-5613-5.

Osojnik, A., Panov, P., & Džeroski, S. (2018). Tree-based methods for online multi-target regression. *Journal of Intelligent Information Systems*, *50*, 315–339. https://doi.org/10.1007/s10844-017-0462-7.

Oza, N. C. (2005). Online bagging and boosting. In *Proceedings of the 2005 IEEE international conference on systems, man and cybernetics (SMC 2005)*, IEEE, vol. 3, pp. 2340–2345. https://doi.org/10.1109/icsmc.2005.1571498.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519.

Shen, F., Yu, H., Sakurai, K., & Hasegawa, O. (2011). An incremental online semi-supervised active learning algorithm based on self-organizing incremental neural network. *Neural Computing and Applications*, *20*(7), 1061–1074. https://doi.org/10.1007/s00521-010-0428-y.

Slavkov, I., & Džeroski, S. (2010). Analyzing gene expression data with predictive clustering trees. Springer, chap., *16*, 389–406. https://doi.org/10.1007/978-1-4419-7738-0_16.

Sousa, R., & Gama, J. (2016). Online semi-supervised learning for multi-target regression in data streams using AMRules. In *Advances in intelligent data analysis XV*, Springer, Berlin, pp. 123–133. https://doi.org/10.1007/978-3-319-46349-0_11.

Sousa, R., & Gama, J. (2017). Co-training semi-supervised learning for single-target regression in data streams using AMRules. In *Foundations of intelligent systems (ISMIS 2017)*, Springer, Berlin, pp. 499–508. https://doi.org/10.1007/978-3-319-60438-1_49.

Spyromitros-Xioufis, E., Groves, W., Tsoumakas, G., & Vlahavas, I. (2012). Multi-label classification methods for multi-target regression. eprint1211.6581

Stojanova, D. (2009). Estimating forest properties from remotely sensed data by using machine learning. Master's thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia.

Stojanova, D., Panov, P., Gjorgjioski, V., Kobler, A., & Džeroski, S. (2010). Estimating vegetation height and canopy cover from remotely sensed data with machine learning. *Ecological Informatics*, *5*(4), 256–266. https://doi.org/10.1016/j.ecoinf.2010.03.004.

Struyf, J., & Džeroski, S. (2006). Constraint based induction of multi-objective regression trees. In *Knowledge discovery in inductive databases (KDID 2005)*, LNCS, vol 3933, Springer, Berlin, pp. 222–233. https://doi.org/10.1007/11733492_13.

Vens, C., Struyf, J., Schietgat, L., Džeroski, S., & Blockeel, H. (2008). Decision trees for hierarchical multi-label classification. *Machine Learning*, *73*(2), 185–214.

Zeisl, B., Leistner, C., Saffari, A., & Bischof, H. (2010). On-line semi-supervised multiple-instance boosting. In *Proceedings of the 2010 IEEE conference on computer vision and pattern recognition (CVPR 2010)*, IEEE, pp. 1879–1879. https://doi.org/10.1109/cvpr.2010.5539860.

Zhang, Y., & Yeung, D. Y. (2009). Semi-supervised multi-task regression. In *Machine learning and knowledge discovery in databases (ECML PKDD 2009)*, Springer, LNCS, vol. 5782, pp. 617–631. https://doi.org/10.1007/978-3-642-04174-7_40.

Zien, A., Brefeld, U., & Scheffer, T. (2007). Transductive support vector machines for structured variables. In *Proceedings of the 24th international conference on machine learning (ICML 2007)*, ACM, pp. 1183–1190. https://doi.org/10.1145/1273496.1273645.