

# Predictive Clustering Trees for Hierarchical Multi-Target Regression

Vanja Mileski<sup>1,2(✉)</sup>, Sašo Džeroski<sup>1,2</sup>, and Dragi Kocev<sup>1,2</sup>

<sup>1</sup> Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia  
{vanja.mileski,saso.dzeroski,dragi.kocev}@ijs.si

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

**Abstract.** Multi-target regression (MTR) is the task of learning predictive models for problems with multiple continuous target variables. In this work, we introduce the task of hierarchical multi-target regression (HMTR), where these target variables are organized in a hierarchy. The hierarchy contains the target variables and has an aggregation function that defines the parent child relationships in the hierarchy. This information can be used by learning methods to obtain better predictive models. We then propose to extend the approach of predictive clustering trees for MTR towards addressing the task of HMTR. The information from the hierarchy is exploited by defining the variance function through a weighted Euclidean distance. We evaluate the proposed method on 4 practically relevant HMTR datasets. The results show that HMTR performs better than standard MTR. Finally, we illustrate the enhanced interpretability potential of PCTs for HMTR.

**Keywords:** Multi-target regression · Hierarchical multi-target regression · Interpretable models · Predictive clustering trees

## 1 Introduction

The task of building a model that is capable of making predictions is called predictive modeling. Supervised learning is the machine learning task of inferring a function from given training data. It is an area of machine learning that has been extensively researched. The goal in supervised learning is to learn, from a set of examples with known class, a function that outputs a prediction for the class of a previously unseen example. Regression models make predictions for continuous variables, e.g. housing prices, weather temperature and similar.

In this work, we are interested in predicting multiple continuous variables since many real-life problems have multiple outputs. Multi-target regression (MTR), also known in the literature as multi-output, multi-response or multivariate regression, tries to simultaneously predict multiple continuous target variables based on a set of input variables [1]. The methods addressing the MTR task can be categorized into two groups: local and global methods [2]. Local methods construct multiple models for each target variable separately and then

combine the predictions from each model, whereas global models construct only one model that outputs the predictions for all of the target variables. If a problem has multiple outputs, it is generally better to build a model that gives a prediction for all of the outputs, rather than one by one [1]. It has been shown that global methods perform better than local methods [3]. They have several advantages over local methods: (a) they can achieve higher predictive performance since they exploit the dependencies that exist between the output variables; (b) they can be more efficient if there are many outputs since building a separate model for each output will be slower, and (c) can produce smaller models than the combined size of the models created with local methods.

Many applications for MTR have been studied due to its applicability to a wide range of domains, including the assessment of vegetation condition, water quality, stock market selection, in chemometrics, to predict wind noise of vehicle components and gas tanks level prediction, to predict biophysical parameters, to perform channel estimation, etc. These applications of MTR also inherit the many challenges present when working with real-world applications, such as missing data and noise, but also provide the means to create the model considering the underlying relationships between the targets, and not only the relationships between the inputs. Global multi-target approaches therefore give a better representation and interpretability of the given problems, as well as simpler models with higher computational efficiency [3]. For a survey on the MTR task, we refer the reader to [4].

Many real-life objects tend to exist within organizational structures. For example, in education, students exist within a hierarchical social structure that can include family, peer group, classroom, grade level, school, school district, state and country [5]. Data collected about an individual is hierarchical, as all the observations are nested within individuals. While there are other methods to deal with this type of data, the assumptions relating to them are rigorous, whereas procedures relating to hierarchical modeling require fewer assumptions. The individuals in the hierarchy follow the hierarchy constraint - an individual that belongs to a given hierarchy node also belongs to all its supernodes [1]. If an example has multiple continuous outputs in a hierarchy that we want to predict, then the task is called hierarchical multi-target regression (HMTR).

In this work, we extend the predictive clustering trees (PCTs) [1,6] towards the HMTR task. They are a state-of-the-art interpretable global approach for different types of outputs and can efficiently learn models valid for the output structure as a whole [1,7,8]. One major reason why we consider PCTs beside their good predictive performance is their interpretability since the model is a tree. This representation is easily understood by people from different backgrounds and expertise [9].

We evaluate the proposed PCTs for HMTR on 4 practically relevant domains. The major goal of the comparison is to check whether the introduction of a hierarchical structure in the output space can improve the predictive power as well as the interpretability of the predictive models.

The remainder of this paper is organized as follows. First, we introduce the task of hierarchical multi-target regression (HMTR). We then propose a method for addressing the HMTR task, i.e., the predictive clustering framework. Next, we outline the experimental design used to evaluate the proposed method. Finally, we conclude and give directions for further work.

## 2 The Hierarchical Multi-Target Regression Task

The work presented in this paper concerns the learning of a model for hierarchical multi-target regression (HMTR). In accordance with Džeroski [10], where predictive modeling is defined for arbitrary types of input and output data, we define the HMTR task as follows:

**Given:**

- A description space  $X$  that consists of tuples of values of continuous (or discrete) primitive data types, i.e.  $\forall X_i \in X, X_i = (x_{i_1}, x_{i_2}, \dots, x_{i_{N_d}})$ , where  $N_d$  is the number of descriptive variables,
- A target space  $Z$ , defined with a numeric variable hierarchy  $(H, \leq_p)$ , where  $H$  is a set of numeric variables and  $\leq_p$  is structured as a rooted tree representing the supervariable relationship ( $\forall h_1, h_2 \in H : h_1 \leq_p h_2$  if and only if  $h_1$  is a supervariable of  $h_2$ ). The supervariables are the products of aggregate functions on their respective children (for example, sum, minimum, maximum, average...) i.e.  $h_j = \text{agg}(h_i) \forall h_i \leq_p h_j$ ,
- A set of examples  $E$ , where each example is a pair of a tuple and a set, from the descriptive and target space, respectively, and each set satisfies the hierarchy constraint, i.e.,  $E = \{(X_i, Z_i) | X_i \in X, Z_i \subseteq H, h \in Z_i \implies \forall h' \leq_p h : h' \in Z_i, 1 \leq i \leq N_e\}$  and  $N_e$  is the number of examples in  $E$  ( $N_e = |E|$ ), and
- A quality criterion  $q$ , which rewards models with high predictive accuracy and low complexity.

**Find:** A function  $f : X \rightarrow R^H$  where  $R^H$  are all of the variables from  $H$  such that  $f$  maximizes  $q$  and  $h \in f(x) \implies \forall h' \leq_p h : h' \in f(x)$  in order to satisfy the hierarchy constraint.

To the best of our knowledge, the task of HMTR as defined above has not been treated before by the research community. The HMTR task applies to problems whose target variables have hierarchical dependencies. The HMTR learning algorithm can exploit this information, which means that it can help in building a better predictive model. Such problems up until now could not have been solved, at least not in such a way that the hierarchy would be taken into account when learning the predictive model. The only solution for solving such problems until now was by using a MTR algorithm and simply ignoring the hierarchy of the output space.

The major advantage of HMTR over MTR is that HMTR is a broader task that encapsulates MTR, i.e., if we instantiate HMTR without a hierarchy weighting scheme, the results will be the same as using a MTR algorithm. Even further,

one of the major gains of this design of the learning algorithm is the possibility to define the hierarchy aggregation function based on the task at hand. Most commonly the aggregate function of the supervariable is the sum of its children, however it can also be any other function, for example minimum, maximum, average and similar. In other words, this design provides an additional degree of freedom for the tree induction algorithm and facilitates its application to a wider range of practical tasks.

Let us illustrate the generality of the HMTR definition by deriving the task of hierarchical multi-label classification (HMLC) from the definition above. HMLC is a variant of classification where an example can have multiple labels at the same time and the labels are organized in a form of hierarchy. The presence or absence of a label for a given example can be presented as a boolean variable or as a binary 0/1 variable. Then, by instantiating the aggregation function as logical OR, we obtain the hierarchy constraint defined in [9], i.e., we define the task of HMLC.

### 3 Predictive Clustering Trees for Hierarchical Multi-Target Regression

The PCT framework views decision trees as a hierarchy of clusters where the root node corresponds to a cluster that contains all of the examples: The other nodes at the lower levels of the tree are smaller clusters partitioned from the nodes that are above them (parent nodes). The predictive clustering framework is implemented in the system CLUS [11].

The TDIDT algorithm (top-down induction of decision trees) is used for inducing the PCTs [12] as presented in Algorithm 1. It takes a set of examples  $E$  as input and produces a tree as an output. The heuristic score  $s$  that is used for selecting the best tests  $b$  is the reduction of variance that is caused by the partitioning  $p$  of the examples. With maximization of the variance reduction, we maximize the cluster homogeneity and improve the predictive performance of the model.

The main difference between the algorithm for learning PCTs and an algorithm for learning decision trees is that the former considers the variance function and the prototype function (that computes a label for each leaf) as parameters that can be instantiated for a given learning task. The PCTs have been instantiated for multi-target prediction [6, 13], prediction of time series [14] and hierarchical multi-label classification (HMLC) [9]. We instantiate these two functions for the HMTR task as follows.

The variance is calculated using a distance function among the values of the variables. The proposed variance for HMTR of a set of examples  $E$  is defined as the average squared distance between each node vector  $L_i$  of the examples and the mean node vector  $\bar{L}$  [9]:

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

---

**Algorithm 1.** The top-down induction algorithm for learning PCTs.
 

---

**Procedure** PCT

**Input:** A dataset  $E$ 
**Output:** A predictive clustering tree

 $(b^*, s^*, p^*) = \text{BestTest}(E)$ 
**if**  $t^* \neq \text{none}$  **then**

     **for each**  $E_i \in p^*$  **do**

          $\text{tree}_i = \text{PCT}(E_i)$ 

     **return**  $\text{node}(b^*, \cup_i \{\text{tree}_i\})$ 
**else**

     **return**  $\text{leaf}(\text{Prototype}(E))$ 
**Procedure** BestTest

**Input:** A dataset  $E$ 
**Output:** best test  $b^*$ , heuristic score  $s^*$ , induced partition,  $p^*$  on the dataset  $E$ 
 $(b^*, s^*, p^*) = (\text{none}, 0, \emptyset)$ 

     **for each** test  $b$  **do**

          $p = \text{induced partition by } b \text{ on } E$ 

          $s = \text{Var}(E) - \sum_{E_i \in p} \frac{|E_i|}{|E|} \cdot \text{Var}(E_i)$ 

     **if**  $(s > s^*) \& \text{Acceptable}(b, p)$  **then**

          $(b^*, s^*, p^*) = (b, s, p)$ 

     **return**  $(b^*, s^*, p^*)$ 


---

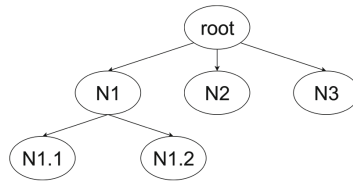
Note that we represent the output hierarchy as a vector of variables by traversing the hierarchy in *preorder* mode.

The target variables are normalized so that they would contribute equally to the overall score. As for the distance  $d$ , any distance can be essentially used. For the task of HMTR, we propose to use a weighted Euclidean distance:

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

where  $L_{i,l}$  is the  $l$ -th component of the node vector  $L_i$  of an instance  $E_i$ ,  $|L|$  is the node's vector size, and  $w(n)$  are the node weights which decrease exponentially with the depth of the node in the hierarchy, e.g., as  $w(n) = w_0^{\text{depth}(n)}$ .

We provide the following example to better illustrate the calculation of the distance for HMTR. We consider the toy hierarchy depicted in Fig. 1 and two



**Fig. 1.** An example of a toy hierarchy.

data examples  $(X_1, Z_1)$  and  $(X_2, Z_2)$  that use a vector representation for the numeric values of the nodes  $\{\text{root}, N1, N1.1, N1.2, N2, N3\}$  in that order. The aggregate function for the supervariables used in this example is *sum*. The examples have the following numeric variables for their outputs:  $Z_1 = \{1.7, 1.1, 0.5, 0.6, 0.1, 0.5\}$  and  $Z_2 = \{1.1, 0.3, 0.1, 0.2, 0.1, 0.7\}$ . The weighted Euclidean distance is then calculated as follows:

$$\begin{aligned} d_{w_0}(Z_1, Z_2) &= d_{w_0}(\{1.7, 1.1, 0.5, 0.6, 0.1, 0.5\}, \{1.1, 0.3, 0.1, 0.2, 0.1, 0.7\}) \\ &= \sqrt{w_0^0(0.6)^2 + w_0^1(0.8)^2 + w_0^2(0.4)^2 + w_0^3(0.4)^2 + w_0^4(0)^2 + w_0^5(-0.2)^2} \quad (3) \\ &= \sqrt{0.36w_0^0 + 0.68w_0^1 + 0.32w_0^2} \end{aligned}$$

Setting a specific value for  $w_0$ ,  $\frac{3}{4}$  for example, yields the following distance:

$$d_{\frac{3}{4}}(Z_1, Z_2) = \sqrt{0.36 \left(\frac{3}{4}\right)^0 + 0.68 \left(\frac{3}{4}\right)^1 + 0.32 \left(\frac{3}{4}\right)^2} = \sqrt{1.05} \approx 1.025 \quad (4)$$

The prototype function used is averaging the values of the examples belonging to a given leaf. One needs to be careful when defining this prototype function, since it can break the hierarchy constraint if it is not compatible with the used hierarchy aggregation function. For example, averaging works fine when the aggregation function is *sum* or *avg*, while it may break the hierarchy constraint if the aggregation function is *min* or *max* (the value calculated by averaging the parents may not be the *min* or *max* of the averaged children).

Next, we analyse the computational complexity of PCTs for HMTR and compare it with the complexity of PCTs for MTR. Let us assume that the size of the training set is  $N_e$ , the number of descriptive attributes is  $N_d$  out of which  $N_c$  are continuous, the number of target attributes is  $N_t$  and the number of supervariables is  $N_s$ . From the algorithm for induction of PCTs, we can note that sorting the  $N_c$  numeric attributes is of the order of  $\mathcal{O}(N_c N_e \log N_e)$  and  $N_c = \mathcal{O}(N_d)$ . Calculating the best split for multiple variables has the complexity order of  $\mathcal{O}(N_t N_d N_e)$  and applying the split to the examples has a linear complexity, i.e.  $\mathcal{O}(N_e)$ . We assume that the tree is balanced, which means that the depth of the tree is the logarithm of the number of examples, i.e.  $\log N_e$ . With these calculations, the computational cost of inducing a single MTR tree is:

$$\mathcal{O}(MTR) = \mathcal{O}(N_d N_e \log^2 N_e) + \mathcal{O}(N_t N_d N_e \log N_e) + \mathcal{O}(N_e \log N_e) \quad (5)$$

For the HMTR algorithm, we also have the supervariables which in this case act like targets. This changes only the cost of calculating the best split to  $\mathcal{O}((N_t + N_s) N_d N_e \log N_e)$ . With this in mind, the order of complexity for the HMTR tree is very similar to the MTR and is given as:

$$\mathcal{O}(HMTR) = \mathcal{O}(N_d N_e \log^2 N_e) + \mathcal{O}((N_t + N_s) N_d N_e \log N_e) + \mathcal{O}(N_e \log N_e) \quad (6)$$

From the complexity analysis of single PCTs for HMTR, we can see that the HMTR algorithm has a higher computational complexity than the MTR algorithm. The increase, however, is linear with the number of targets from the introduced supervariables. We can see that the dominant elements in the computational costs are the first two in the parentheses, i.e. the one containing the second logarithmic power of the number of examples, and the one that is multiplied with the number of targets. The first element is  $\mathcal{O}(N_d N_e \log^2 N_e)$ , and the second is  $\mathcal{O}(N_t N_d N_e \log N_e)$  or  $\mathcal{O}((N_t + N_s) N_d N_e \log N_e)$  for MTR and HMTR respectively. If we compare the two terms, we can see that the first term is bigger than the second when  $\log N_e > N_t$  for MTR and  $\log N_e > (N_t + N_s)$  for HMTR. Let us explore the first case where the first term is smaller. This means that when comparing MTR and HMTR, HMTR will have greater computational cost, due to the addition of  $N_s$ . Let us now explore the second case where  $\log N_e$  is greater. This will make the first term of the equation the major contributor to the cost of the algorithm. With this, the linear increase in the second terms of the equations for the HMTR task (i.e. the addition of  $N_s$  in  $\mathcal{O}((N_t + N_s) N_d N_e \log N_e)$ ) will be insignificant in this case, resulting in comparable performance between MTR and HMTR, on datasets with sufficiently large number of examples.

## 4 Experimental Design

We assess the effect of introducing a hierarchy in the output space by comparing the performance of PCTs for HMTR with the performance of PCTs for MTR. We estimate the predictive performance by using 10-fold cross-validation. We follow the recommendations from Borchani et al. [4] and adopt the average correlation coefficient (*aCC*) and average relative root mean squared error (*aRRMSE*) as evaluation measures. For a fair comparison, we calculate these errors only for the variables at the leafs of the hierarchy.

The parametrization of HMTR sets the weight parameter  $w(n)$  for a node  $n$ . From the weights of the Euclidean distance, one might have already concluded that if we instantiate the algorithm such that  $w = 1$ , all of the nodes will have the same weight regardless of their depth in the hierarchy, thus giving little weight to the hierarchy. Lowering the weight  $w$  from 1 downwards, we place increasingly greater importance on the hierarchy, meaning that a smaller weight  $w$  assigns larger relative weights to the higher levels of the hierarchy in comparison to the deepest nodes in the hierarchy, which get the smallest weights.

For each application domain, the user needs to define the hierarchy for the output space of the problem (if it is not readily available) and an adequate aggregation function that will be applied in order to calculate the values for the parent nodes. The user also needs to choose the weighting parameter.

We perform the experimental evaluation of our method on 4 practically relevant datasets: **OSALES**, **MARSEXPRESS**, **ADNI** and **SIS**.

The task for the **OSALES** dataset is the prediction of online sales of products described with various product features. The dataset is from the Kaggle's Online



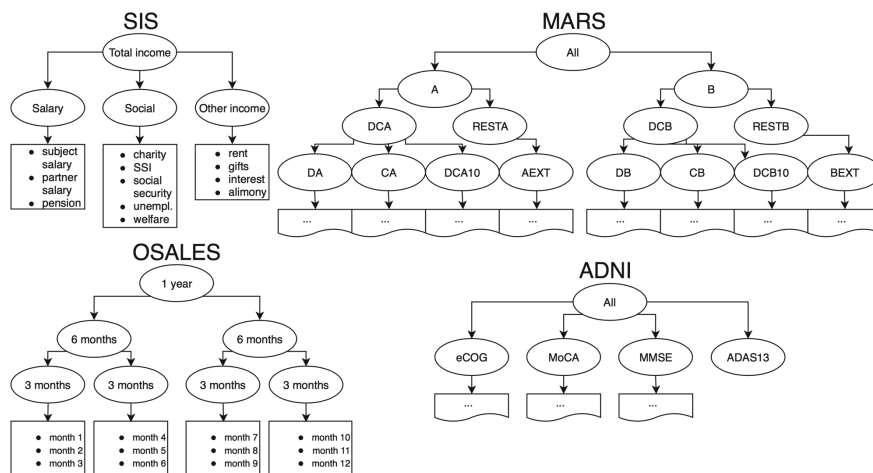
Product Sales competition in 2012 and the goal is to predict the monthly sales for 12 months of products. It has 639 examples, 413 features and 12 target variables that need to be predicted. The data and additional information are available at <https://www.kaggle.com/c/online-sales>.

The **MARSEXPRESS** dataset concerns the power consumption of the the different systems of the European Space Agency’s MARS Express spacecraft orbiting Mars. It includes context data as descriptive variables and 33 thermal power lines read-outs of the electric current (power consumption) in each thermal subsystem node. The data consists of 464 features describing the current and past operation of the satellite and 33 target features referring to the thermal power lines. Here, we consider a subsample of 20000 examples out of the over 2.6 million examples in the dataset. More information about the data can be found at <https://kelvins.esa.int/mars-express-power-challenge/home/>.

The **ADNI** dataset contains data from the Alzheimer’s Disease Neuroimaging Initiative (ADNI) – a longitudinal study for developing clinical, genetic, imaging and biochemical biomarkers for the Alzheimer’s disease (AD). The ADNI dataset consists of subjects with AD, as well as elderly controls and patients with mild cognitive impairment. The dataset consists of 659 examples [15]. We consider 10 descriptive features (APOE4 and PET imaging data) and 27 targets consisting of clinical scores that describe everyday cognition (eCOG), Montreal Cognitive Assessment (MoCA), Mini-Mental State Exam (MMSE) and Alzheimer’s Disease Assessment Scale (ADAS13).

The **SIS** dataset refers to the New York City Social Indicators Survey (SIS) – a study on social problems and inequality in New York City for the year 2001. For the 229 descriptive features, we consider questionnaire answers on a variety of topics, while the 12 targets are the subjects’ income from various sources. We consider data from 1501 subjects. More information is available at <http://cupop.columbia.edu/research/research-areas/social-indicators-survey-sis>.

For the four datasets, we consider the hierarchies presented in Fig. 2. The OSALES hierarchy combines the months into quarters (a period of 3 months),



**Fig. 2.** The hierarchies for the output spaces of the four HMTR datasets.



then groups the quarters into semesters, and then to an entire year. In the MARSEXPRESS dataset, the thermal power lines are grouped depending on their location and subsystem, resulting in a hierarchy of depth 5. The ADNI dataset before combining all of the target variables in the root node separates them according to the four different assessment examinations. The SIS dataset groups the target variables depending on the type of income: salary (and pension), social income and income from other sources.

## 5 Results and Discussion

In this section, we present and discuss the results obtained from the experimental evaluation. We first compare the predictive performance of the proposed PCTs for HMTR with the predictive performance of PCTs for MTR. We next illustrate the enhanced interpretability of the PCTs for HMTR.

Table 1 gives a detailed overview of the performance of PCTs for HMTR and PCTs for MTR as measured with  $aRRMSE$  on the training set and on unseen data (estimated by 10-fold cross-validation). First of all, it shows the performance of PCTs for HMTR for different values of the parameter  $w_0$ . We can note that the training error generally increases when decreasing the value of  $w_0$  (an exception is the SIS dataset for  $w_0 = 0.25$ ). This means that putting less weight on the more general concepts in the hierarchy guides the learning algorithm towards learning better fitted descriptions of the data. However, we

**Table 1.** Train and test  $aRRMSE$  errors of PCTs for HMTR and PCTs for MTR with different weights  $w_0$ .

Dataset	HMTR			MTR	
	Weight	Train	Test	Train	Test
OSALES	0.75	0.445	0.840	0.422	0.878
	0.5	0.450	0.844		
	0.25	0.476	0.842		
MARS	0.75	0.453	0.836	0.449	0.839
	0.5	0.456	0.836		
	0.25	0.472	0.841		
ADNI	0.75	0.640	1.055	0.632	1.074
	0.5	0.648	1.051		
	0.25	0.652	1.063		
SIS	0.75	0.704	1.155	0.705	1.175
	0.5	0.720	1.124		
	0.25	0.672	1.112		

cannot make the same statement about the errors as estimated with 10-fold cross validation (the column test in the Table).

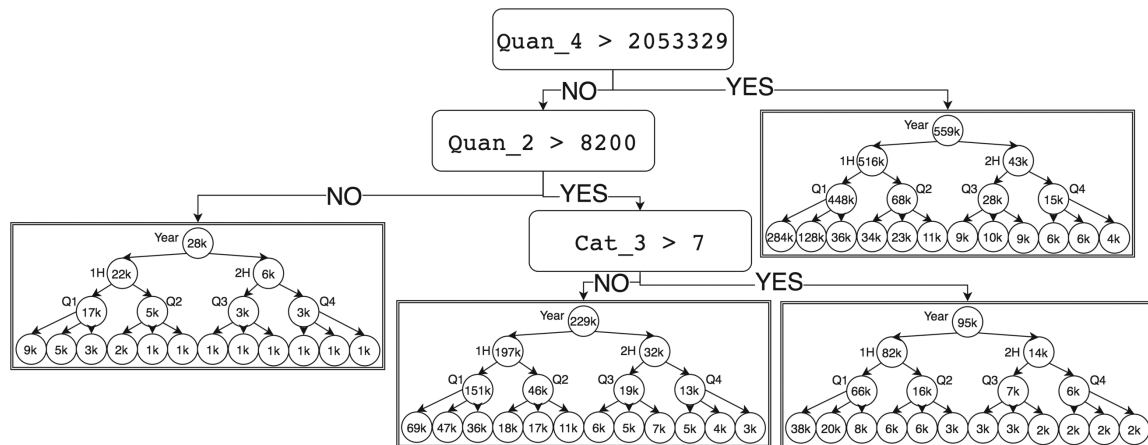
Furthermore, we observe that the PCTs for HMTR overfit less than PCTs for MTR. Namely, the training errors of PCTs for HMTR are always worse than the training errors of the PCTs for MTR, while this is reversed for the error estimated with 10-fold cross validation. This is due to the fact that with the hierarchy, the learning algorithm now is able to generalize better the data. Hence, PCTs for HMTR perform worse on the training set because the hierarchy prevents them from focusing on too specific concepts existing in the data.

Next, we can note that PCTs for HMTR have better predictive performance than PCTs for MTR across all datasets and almost all of the values for  $w_0$  (an exception is the MARSEXPRESS dataset for  $w_0 = 0.25$ ). Based on these results, the best values for the depth parameter  $w_0$  per dataset are as follows. For the OSALES and MARSEXPRESS datasets, the best value for  $w_0$  is 0.75. For the ADNI dataset, the best value is 0.5, while for the SIS dataset is 0.25.

We then compare the performance of PCTs for HMTR with the best values for the parameter  $w_0$  with the performance of PCTs for MTR in Table 2. The performance is here measured with both  $aCC$  and  $aRRMSE$ . For both evaluation

**Table 2.** Evaluation (test) and efficiency comparison of HMTR and MTR

Dataset	HMTR		MTR	
	$aCC$	$aRRMSE$	$aCC$	$aRRMSE$
OSALES	0.361	0.840	0.331	0.878
MARS	0.418	0.836	0.416	0.839
ADNI	0.081	1.051	0.074	1.074
SIS	0.014	1.112	0.010	1.175



**Fig. 3.** A model for the OSALES dataset: A PCT for HMTR predicting the online sales of products by months (lowest part of the hierarchy), quarters, semesters and year (root node of the hierarchy).

measures, the same conclusions can be made: PCTs for HMTR outperform PCTs for MTR on the four datasets.

Finally, we demonstrate the enhanced interpretability of the PCTs by showing a heavily pruned PCT for HMTR learned on the OSALES dataset in Fig. 3. As it can be seen, the leafs of the PCT contain a prediction for the complete hierarchy, thus a domain expert can directly observe and comment on the more specific variables (lower in the hierarchy) and the more global variables (upper in the hierarchy, quarters, semesters and yearly). For example, if the attribute *QUAN\_4* is higher than 2053329, from the predictions from the model, we can see that the sales of these products will be high overall, but will mostly take place in the first quarter of the year.

## 6 Conclusions

In this work, we introduce the task of hierarchical multi-target regression (HMTR). This structured output prediction task considers that a problem has multiple continuous target variables that can have hierarchical relationships among themselves. The formal task definition presented here unites all of the multi-target prediction tasks available.

We also propose a method for learning predictive clustering trees (PCTs) for the task of HMTR. Moreover, we propose to include the information about the hierarchy through adapting the Euclidean distance. The weighted Euclidean distance introduces weights to take into account the depth of the nodes in the hierarchy.

We evaluated the proposed method on 4 datasets and compared its performance to PCTs for MTR that do not exploit the hierarchical structure in the output space. The evaluation revealed that the PCTs for HMTR have better predictive performance than PCTs for MTR across all datasets. Furthermore, the PCTs for HMTR overfit less than PCTs for MTR. All in all, we showed that considering a hierarchical structure in the output space consisting of multiple continuous variables improves the performance.

We plan to extend this work along several directions. First, we will look more closely into the influence of various aggregation measures for different problems. We will also look into the weight parameter that the algorithm uses. As a general rule of thumb,  $w = 3/4$  is a good starting point for most of the data sets, where higher values reduce the influence of the hierarchy and make the problem closer to standard MTR, and low values give more importance to the hierarchy than the target variables in the leaves. Finally, another direction to explore is the one of ensemble methods like random forests and bagging. We will investigate whether the performance improvement in HMTR carries over in the ensemble learning setting.

**Acknowledgments.** We acknowledge the financial support of the European Commission through the grants ICT-2013-612944 MAESTRA and ICT-2013-604102 HBP.

## References

1. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. *Pattern Recogn.* **46**(3), 817–833 (2013)
2. Bakır, G.H., Hofmann, T., Schölkopf, B., Smola, A.J., Taskar, B., Vishwanathan, S.V.N.: *Predicting Structured Data*. Neural Information Processing. The MIT Press, Cambridge (2007)
3. Kocev, D., Džeroski, S., White, M.D., Newell, G.R., Griffioen, P.: Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. *Ecol. Model.* **220**(8), 1159–1168 (2009)
4. Borchani, H., Varando, G., Bielza, C., Larrañaga, P.: A survey on multi-output regression. *Wiley Interdisc. Rev. Data Min. Knowl. Discov.* **5**(5), 216–233 (2015)
5. Osborne, J.W.: The advantages of hierarchical linear modeling (2000)
6. Struyf, J., Džeroski, S.: Constraint based induction of multi-objective regression trees. In: Bonchi, F., Boulicaut, J.-F. (eds.) *KDID 2005*. LNCS, vol. 3933, pp. 222–233. Springer, Heidelberg (2006). doi:[10.1007/11733492\\_13](https://doi.org/10.1007/11733492_13)
7. Tsoumakas, G., Spyromitros-Xioufis, E., Vrekou, A., Vlahavas, I.: Multi-target regression via random linear target combinations. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) *ECML PKDD 2014*. LNCS, vol. 8726, pp. 225–240. Springer, Heidelberg (2014). doi:[10.1007/978-3-662-44845-8\\_15](https://doi.org/10.1007/978-3-662-44845-8_15)
8. Spyromitros-Xioufis, E., Tsoumakas, G., Groves, W., Vlahavas, I.: Multi-target regression via input space expansion: treating targets as inputs. *Mach. Learn.* **104**(1), 55–98 (2016)
9. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. *Mach. Learn.* **73**(2), 185–214 (2008)
10. Džeroski, S.: Towards a general framework for data mining. In: Džeroski, S., Struyf, J. (eds.) *KDID 2006*. LNCS, vol. 4747, pp. 259–300. Springer, Heidelberg (2007). doi:[10.1007/978-3-540-75549-4\\_16](https://doi.org/10.1007/978-3-540-75549-4_16)
11. Blockeel, H., Struyf, J.: Efficient algorithms for decision tree cross-validation. *J. Mach. Learn. Res.* **3**, 621–650 (2002)
12. Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton (1984)
13. Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S.: An extensive experimental comparison of methods for multi-label learning. *Pattern Recogn.* **45**(9), 3084–3104 (2012)
14. Slavkov, I., Gjorgjioski, V., Struyf, J., Džeroski, S.: Finding explained groups of time-course gene expression profiles with predictive clustering trees. *Mol. BioSyst.* **6**(4), 729–740 (2010)
15. Gamberger, D., Ženko, B., Mitelpunkt, A., Shachar, N., Lavrač, N.: Clusters of male and female alzheimers disease patients in the alzheimers disease neuroimaging initiative (adni) database. *Brain Inf.* **3**(3), 169–179 (2016)