

Feature Ranking for Multi-target Regression with Tree Ensemble Methods

Matej Petković^{1,2(✉)}, Sašo Džeroski^{1,2}, and Dragi Kocev^{1,2}

¹ Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia

² Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
{matej.petkovic,saso.dzeroski,dragi.kocev}@ijs.si

Abstract. In this work, we address the task of feature ranking for multi-target regression (MTR). The task of MTR concerns problems where there are multiple continuous dependent variables and the goal is to learn a model for predicting all of the targets simultaneously. This task is receiving an increasing attention from the research community. However, performing feature ranking in the context of MTR has not been studied. Here, we propose three feature ranking methods for MTR: Symbolic, Genie3 and Random Forest. These methods are then coupled with three types of ensemble methods: Bagging, Random Forest, and Extremely Randomized Trees. All of the ensemble methods use predictive clustering trees (PCTs) as base predictive models. PCTs are a generalization of decision trees capable of MTR. In total, we consider eight different ensemble-ranking pairs. We extensively evaluate these pairs on 26 benchmark MTR datasets. The results reveal that all of the methods produce relevant feature rankings and that the best performing method is Genie3 ranking used with Random Forests of PCTs.

Keywords: Multi-target Regression · Feature ranking · Feature importance · Ensembles · Predictive Clustering Trees

1 Introduction

Single target regression (STR) is a subfield of predictive modelling, where the goal is to learn a model able to predict the values of a single numeric target variable. STR can be generalized to multi-target regression (MTR), where the goal is to learn a model that predicts $T \geq 2$ targets. The STR and MTR tasks can be formalized as described below.

We are given a set of examples \mathbf{x} from the input domain $\mathcal{X} \subseteq \mathcal{X}_1 \times \dots \times \mathcal{X}_D$, $D \geq 1$ being the number of descriptive attributes (features). We assume that the domain \mathcal{X}_i of the i -th descriptive attribute x_i is either a subset of \mathbb{R} or an arbitrary finite set, i.e., x_i is either numeric or nominal. Each example \mathbf{x} is associated with a target value $\mathbf{y}(\mathbf{x})$ from the target domain $\mathcal{Y} \subseteq \mathcal{Y}_1 \times \dots \times \mathcal{Y}_T \subseteq \mathbb{R}^T$, T being the number of target attributes (targets). STR considers domains where $T = 1$, while MTR considers domains with $T \geq 2$. In the latter case, the j -th component of the target vector $\mathbf{y}(\mathbf{x})$ is denoted by $y_j(\mathbf{x})$. The true

mapping $y : \mathbf{x} \mapsto y(\mathbf{x})$ (STR) or $\mathbf{y} : \mathbf{x} \mapsto \mathbf{y}(\mathbf{x})$ (MTR) is unknown and the goal of regression is to find its approximation, given a dataset $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$.

STR is a well established research topic, while MTR is recently attracting interest in the research community [22, 23]. MTR is a structured output prediction task with applications in a wide range of real life problems where we are interested in simultaneously predicting multiple continuous variables. Prominent examples come from ecology and include predicting the abundance of different species living in the same habitat [12] and predicting properties of forests [21].

A possible way to approach a MTR problem is problem transformation, which transforms one MTR problem to several STR problems and build one predictive model for each target separately. Another way to approach the problem is by algorithm adaptation, i.e., to change STR methods in such a way they are able to exploit the potential relatedness between the multiple targets. For example, regression trees can be generalized so that the heuristic function considers the multiple targets and the leaves make predictions for all targets. For an overview of the different MTR, we refer the reader to Borchani et al. [5].

Another important task in machine learning is feature ranking, which is typically seen as a data preprocessing step. Here, the importances of descriptive attributes (features) are estimated and an ordering (or ranking) of the features is made, based on the estimated importances. There are two main reasons for doing this. First, we may want to reduce the dimensionality of the input space, so that only the features that contain the most information about target(s) are kept in the dataset. By doing this, we decrease the amount of memory/time needed to build a predictive model, while the performance of the model is not degraded. Second, dimensionality reduction typically results in models that are easier to understand, which comes in handy when a machine learning expert works in collaboration with a domain expert. Predictive models, such as decision trees, are easier to interpret when a small number of the most relevant features are used to learn them.

There is a plethora of feature ranking methods for the machine learning tasks of single target regression and classification. For an overview, see Stanczyk and Jain [24]. However, in the case of MTR, the task of feature ranking has not been studied to a great extent. To the best of our knowledge, there is no previous work from the machine learning community.

In the field of statistics, a few such methods can be found. Their main drawback is that they allow only for numeric features, since they typically assume a (generalized) linear model $\mathbf{y} = A\mathbf{x} + \mathbf{e}$, where A is a $T \times D$ matrix and \mathbf{e} is a random noise vector. One such method is forward selection. It starts with a constant model $\mathbf{y}(\mathbf{x}) = c \in \mathbb{R}$, and repeatedly adds the most significant feature that improves the model. The sooner the feature is included in the model, the greater the importance. For an overview of these methods, see Brobbey [9].

In this work, we propose three feature ranking methods based on ensembles of predictive clustering trees (PCTs) [4, 22]. PCTs are generalization of decision trees able to handle various types of structured output prediction tasks, including MTR. The proposed feature ranking methods can handle both numeric and nominal features.

The proposed methods exploit different properties of the ensemble learning mechanism to estimate feature importances. More specifically, two of the methods are adaptations of the feature importance measures already known from the single target regression task: Genie3 [18] and Random Forest ranking [7]. Genie3 uses the variance reduction at each tree node as a proxy for the importance of the feature that is used in the test at a given node. Random Forest ranking permutes the values of a feature on the out-of-bag set of data to estimate how much worse performance this will yield as compared to the original data. This decrease of predictive performance is then taken as a proxy for the feature’s importance. Finally, the third method named Symbolic counts how often a feature appears in the nodes of the trees from an ensemble. These appearances can be also weighted with the nodes’ depth at which a given feature appears. This is a general method that is applicable to an arbitrary machine learning task for which tree-based models can be learned.

Furthermore, these three ranking methods can be coupled with three ensemble learning methods: Bagging [6], Random Forests [7] and Extremely randomized trees [14]. Note that Random Forests ranking cannot be coupled with Extremely randomized trees because the latter do not perform bootstrapping of the examples. This yields in total 8 pairs of ensemble learning method and feature ranking method.

We extensively evaluate the proposed methods on 26 benchmark MTR datasets. The evaluation is performed by comparing the performance of the standard 5NN (5 nearest neighbors) prediction method with a 5NN prediction method that uses the obtained feature importances as weights during distance calculation. The experiments investigate the relevance of the obtained feature rankings and look for the optimal combination of ensemble learning and feature ranking method.

The remainder of this paper is organized as follows. Section 2 presents the PCT approach to MTR. Ensembles of PCTs for MTR and feature ranking methods based on these are described in Sect. 3. Section 4 outlines the experimental design, while Sect. 5 discusses the results of the experimental evaluation. Finally, the conclusions and a summary are given in Sect. 6.

2 Predictive Clustering Trees for Multi-target Regression

PCTs generalize decision trees and can be used for a variety of learning tasks, including clustering and different types of prediction. The PCT framework views a decision tree as a hierarchy of clusters: The root of a PCT corresponds to one cluster containing all data, which is recursively partitioned into smaller clusters while moving down the tree. The leaves represent the clusters at the lowest level of the hierarchy and each leaf is labeled with its cluster’s prototype (prediction).

PCTs are induced with the standard top-down induction of decision trees algorithm presented in Table 1 [8]. It takes as input a set of examples E and outputs a tree. The heuristic h that is used for selecting the tests is the reduction of variance caused by partitioning the instances (see line 4 of the *BestTest*

Table 1. The top-down induction algorithm for PCTs.

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

procedure in Table 1). By maximizing the variance reduction, the cluster homogeneity is maximized: The algorithm is thus guided towards small trees with good predictive performance. If no acceptable test can be found (line 6 of the PCT procedure), i.e., no test reduces the variance significantly, then a leaf is created and the prototype of the instances belonging to that leaf is computed.

The main difference between the algorithm for learning PCTs and other algorithms for learning decision trees is that the former considers the variance function and the prototype function (that computes predictions in leaves) as parameters that can be instantiated for a given learning task. In this work, we focus on the task of MTR and define the variance function as follows. First, we define the average \bar{y}_j and variance of the target y_j over subset $E \subseteq \mathcal{D}_{\text{TRAIN}}$ as

$$\bar{y}_j(E) = \frac{1}{|E|} \sum_{\mathbf{x} \in E} y_j(\mathbf{x}) \quad \text{and} \quad \text{Var}_j(E) = \frac{1}{|E|} \sum_{\mathbf{x} \in E} (y_j(\mathbf{x}) - \bar{y}_j(E))^2. \quad (1)$$

We then compute the weights $w_j = \text{Var}_j(\mathcal{D}_{\text{TRAIN}})$ and use them as normalization factors in the definition of variance function:

$$\text{Var}(E) = \frac{1}{T} \sum_{j=1}^T \frac{1}{w_j} \text{Var}_j(E).$$

In a leaf L , the prototype function returns a vector $(\bar{y}_1(E_L), \dots, \bar{y}_T(E_L))$, where E_L denotes the set of all examples that fall into the leaf L . For a detailed description of PCTs for MTR, we refer the reader to Blockeel [4] and Kocev [22]. The PCT framework is implemented in the CLUS system (available at <http://clus.sourceforge.net>).

3 Feature Ranking via Ensembles of PCTs

We use PCTs as the base models in three types of ensembles [22] that are constructed to calculate the variable importance, i.e., the feature ranking. In the following, we first present the ensemble methods and then describe the feature ranking methods.

3.1 Ensembles of PCTs

An ensemble is a set of base predictive models constructed with a given algorithm. The prediction for each new example is made by combining the predictions of every model from the ensemble. This can be done by taking the average in regression tasks, and the majority or probability distribution vote in classification tasks [6]. For the task of MTR, we consider ensembles of PCTs [22], where the predictions are the average values for each target.

A necessary condition for an ensemble to be more accurate than any of its individual members, is that the members are accurate and diverse models [16]. This means that they perform better than random guessing. On the other hand, it means that they make different errors on new examples.

There are several ways to introduce diversity among the base predictive models in an ensemble. We describe how this is done in Random Forests [7], Bagging [6] and Extra Trees ensembles [14].

Random Forest (RF) and Bagging. A Random Forest is an ensemble of trees where diversity among the predictive models is obtained in two ways. First, instead of being learned from the original dataset $\mathcal{D}_{\text{TRAIN}}$, each tree is built from a different bootstrap replicate \mathcal{B} . The chosen examples from such a replicate form a so called bag \mathcal{B} , while the rest are called out-of-bag examples (OOB). Hence, we perform a call $\text{PCT}(\mathcal{B})$ rather than $\text{PCT}(\mathcal{D}_{\text{TRAIN}})$ as we would do in the case when a single PCT is to be grown.

Additionally, we modify the line 2 of the *BestTest* procedure (see Table 1), to change the feature set during learning. More precisely, at each node in a decision tree, a random subset of the input attributes is taken, and the best test is selected from the splits defined on these. The number of attributes that are retained is given as a function of the total number of descriptive attributes D , e.g., $\lceil \sqrt{D} \rceil$, $\lceil \log_2(D) \rceil$, $D/4$, etc. In the special case when we keep all attributes, we obtain the Bagging procedure, where the only source of diversity is the difference in the bootstrap replicates of the data.

Extra trees ensemble (ET). The source of diversity in ET comes from the extreme randomization of the tree learning procedure. Here, at each node all attributes are considered (as in Bagging), but we do not evaluate all tests that the attributes yield. Rather, we choose randomly only one per attribute. Among these D tests, we choose the best one, hence the only difference compared to standard top-down PCT induction, is that a modified line 2 of the *BestTest* procedure is used. Note that ET uses the initial dataset $\mathcal{D}_{\text{TRAIN}}$ for learning the base predictive models and does not make bootstrap replicates.

3.2 Ensemble Feature Ranking Methods

Feature ranking of the descriptive variables can be obtained either by exploiting the ensemble structure of the learning algorithm or the mechanism of Random Forests. For its simplicity, we first describe symbolic ranking. Then, we discuss Genie3 and Random Forest ranking.

In the following, we denote a tree by \mathcal{T} , whereas $\mathcal{N} \in \mathcal{T}$ denotes a node. Trees form a forest \mathcal{F} . Its size (the number of trees in the forest) is denoted by $|\mathcal{F}|$. The set of all internal nodes of a tree \mathcal{T} in which the attribute x_i appears as part of the test, is denoted by $\mathcal{T}(x_i)$.

Symbolic ranking (Symb). Let $d(\mathcal{N})$ denote the *depth* of $\mathcal{N} \in \mathcal{T}$. The depth is defined recursively: if \mathcal{N} is the root of \mathcal{T} , then $d(\mathcal{N}) = 0$. Otherwise, $d(\mathcal{N}) = 1 + d(\text{parent}(\mathcal{N}))$. In the basic version of symbolic ranking, we simply count how many times a given attribute occurs in the tests in the internal nodes of the trees in the forest. Since the attributes that appear at lower depths (i.e., closer to the root) are intuitively more important than those that appear deeper in the trees, we introduce the parameter $w \in (0, 1]$ and define the importance of the attribute x_i as

$$\text{importance}_{\text{SYMB}}(x_i) = \frac{1}{|\mathcal{F}|} \sum_{\mathcal{T} \in \mathcal{F}} \sum_{\mathcal{N} \in \mathcal{T}(x_i)} w^{d(\mathcal{N})}. \quad (2)$$

Note that symbolic ranking is applicable to all three ensemble methods that we use, and that the basic version of the ranking corresponds to the choice $w = 1$.

Genie3. The main motivation for Genie3 ranking is that splitting the current subset $E \subseteq \mathcal{D}_{\text{TRAIN}}$, according to a test where an important attribute appears, should result in high variance reduction. As in the symbolic ranking case, greater emphasis is put on the attributes higher in the tree, i.e., on the splits where $|E|$ is larger. The Genie3 importance of the attribute x_i is defined as

$$\text{importance}_{\text{GENIE3}}(x_i) = \frac{1}{|\mathcal{F}|} \sum_{\mathcal{T} \in \mathcal{F}} \sum_{\mathcal{N} \in \mathcal{T}(x_i)} |E(\mathcal{N})| h^*(\mathcal{N}),$$

where $E(\mathcal{N})$ is the set of examples that come to the node \mathcal{N} , and $h^*(\mathcal{N})$ is the value of the variance reduction function described in the *BestTest* procedure. Genie3 ranking is applicable to all three ensemble methods that we use.

Random Forest (RF). This feature ranking method tests how much does noise in a given descriptive attribute decrease the predictive performance of the trees in the forest. The greater the performance degradation, the more important the attribute. This feature ranking algorithm uses the internal out-of-bag estimates of the error, therefore it cannot be used with ensembles of extra trees.

Once a tree \mathcal{T} is grown, the algorithm evaluates the performance of the tree by using the corresponding OOB $_{\mathcal{T}}$ examples. This results in the predictive error $\text{Err}(\text{OOB}_{\mathcal{T}}) \geq 0$. Here, we assume that lower error value corresponds to better predictions. To assess the importance of the attribute x_i for the tree \mathcal{T} , we randomly permute the values of this attribute in the set OOB $_{\mathcal{T}}$ and obtain the set OOB $_{\mathcal{T}}^i$. Then, the error $\text{Err}(\text{OOB}_{\mathcal{T}}^i)$ is computed and the importance of the attribute x_i for the tree \mathcal{T} is defined as the relative increase of error after noising the attribute. The Random Forest importance of the attribute is the average of these values across all trees in the forest, namely

$$\text{importance}_{\text{RF}}(x_i) = \frac{1}{|\mathcal{F}|} \sum_{\mathcal{T} \in \mathcal{F}} \frac{\text{Err}(\text{OOB}_{\mathcal{T}}^i) - \text{Err}(\text{OOB}_{\mathcal{T}})}{\text{Err}(\text{OOB}_{\mathcal{T}})}.$$

Note that $Err(OOB_{\mathcal{T}}^i) = Err(OOB_{\mathcal{T}})$ if the attribute x_i does not appear in \mathcal{T} . This can speed up the computation of $importance_{RF}$, but this feature ranking method is still the most time consuming. While the time complexity of the first two is negligible as compared to the one of growing the forest, this one has an additional linear factor: the number of examples in the dataset.

4 Experimental Design

In this section, we present the experimental design used to evaluate the performance of the proposed feature ranking methods. We begin by stating the main experimental questions and then briefly summarize the MTR datasets used in this study. We next describe the evaluation procedure and give the specific parameter instantiations of the methods.

4.1 Experimental Questions

The main focus of this study is to answer the following questions:

1. Can additional knowledge from feature importances lead to better predictive performance of a regressor, i.e., are the obtained feature rankings relevant?
2. Which ranking method is the most appropriate for a given ensemble method?
3. Which ensemble method is the most appropriate for a given ranking algorithm?
4. Which *ensemble-ranking* pair is the best overall?

For answering these questions, we design several experiments and comparisons of performance. We learn different feature rankings by considering combinations of ensemble learning methods and feature ranking methods. More specifically, we construct 8 different feature rankings: *Random Forest Symb*, *Random Forest Genie3*, *Random Forest RF*, *Bagging Symb*, *Bagging Genie3*, *Bagging RF*, *Extra trees Symb*, and *Extra trees Genie3*. We then use the obtained feature importances as weights in k nearest neighbor predictor (kNN) and compare the different rankings to address the questions outlined above. Finally, based on the obtained results, we identify the method that yields the best feature ranking.

4.2 Data Description

We use 26 MTR benchmark problems. Table 2 presents the basic statistics of the datasets: The number of features per dataset ranges from 6 to 576 and features are mainly numeric. The number of targets ranges from 2 to 16, while the number of examples takes values between 42 and 60607.

The datasets come from different domains: *andro*, *ENB* and *water quality* originate from studies of water quality; the *ATP* datasets concern the prediction of airline tickets prices; *collembola*, the *Forestry* datasets, *soil quality* and *vegetation condition* describe soil and vegetation conditions; *EDM* stands for electrical

discharge machining; *jura* contains measurements of heavy metals concentrations; *metal-data* is about meta learning; *OES* stands for occupational employment survey; *osales* (online product sales) and *scpf* (see-click-predict fix) originate from two Kaggle competitions; *RF1* and *RF2* describe river flows; *SCM1d* and *SCM20d* were derived from a competition in supply chain management; *sigmeareal* and *sigmeasim* deal with cross-pollination between conventional and GM crops, and *slump* concerns the prediction of concrete slump.

Table 2. Description of the benchmark problems in terms of the number of nominal and numeric descriptive attributes, the number of targets, and the number of examples.

Dataset	Nominal	Numeric	Targets	Examples
andro [17]	0	30	6	49
ATP1d [23]	0	411	6	337
ATP7d [23]	0	411	6	296
collembola [19]	8	39	3	393
EDM [20]	0	16	2	154
ENB [28]	0	8	2	768
Forestry Kras [26]	0	160	11	60607
Forestry LIDAR IRS [25]	0	29	2	2730
Forestry LIDAR Landsat [25]	0	150	2	6218
Forestry LIDAR Spot [25]	0	49	2	2730
jura [15]	0	15	3	359
metal-data [27]	0	53	10	42
OES10 [23]	0	298	16	403
OES97 [23]	0	263	16	334
osales [1]	0	401	12	639
RF1 [23]	0	64	8	9125
RF2 [23]	0	576	8	9125
SCM1d [23]	0	280	16	9803
SCM20d [23]	0	61	16	8966
scpf [2]	0	23	3	1137
sigmeareal [11]	0	6	2	817
sigmeasim [11]	2	9	2	10368
slump [29]	0	7	3	103
soil quality [12]	0	156	3	1944
vegetation condition [21]	1	39	7	16967
water quality [13]	0	16	14	1060

4.3 Evaluation Methodology

We adopted the following evaluation methodology to properly assess the performance of the proposed methods. First, we randomly divide each dataset \mathcal{D} into 2/3 for the training part $\mathcal{D}_{\text{TRAIN}}$ and 1/3 for the testing part $\mathcal{D}_{\text{TEST}}$. A ranking is computed from an ensemble that is built on the training part only. This procedure is repeated 10 times and the performance measures are averaged.

The quality of the ranking is assessed by using the kNN algorithm. Instead of the standard Euclidean distance, its weighted version was used in kNN. For two input vectors \mathbf{x}^1 in \mathbf{x}^2 , the distance d between them is defined as

$$d(\mathbf{x}^1, \mathbf{x}^2) = \sqrt{\sum_{i=1}^D w_i d_i^2(\mathbf{x}_i^1, \mathbf{x}_i^2)}, \quad (3)$$

where the distance $d_i : \mathcal{X}_i \times \mathcal{X}_i \rightarrow [0, 1]$ is defined as

$$d_i(\mathbf{x}^1, \mathbf{x}^2) = \begin{cases} \mathbf{1}[\mathbf{x}_i^1 \neq \mathbf{x}_i^2] & : \mathcal{X}_i \text{ nominal} \\ \frac{|\mathbf{x}_i^1 - \mathbf{x}_i^2|}{\max_{\mathbf{x}} \mathbf{x}_i - \min_{\mathbf{x}} \mathbf{x}_i} & : \mathcal{X}_i \subseteq \mathbb{R} \end{cases},$$

where max and min go over the known examples \mathbf{x} . The weights are set to $w_i = \max\{\text{importance}(x_i), 0\}$ and are equal to the feature importances obtained by Symb and Genie3 ranking. They need to be made non-negative for RF ranking. In this way, we ensure that d is well defined and ignore the attributes that are of lower importance than a randomly generated attribute.

The evaluation through a kNN predictive model was chosen for two main reasons. First, this is a distance based model, which can easily use the information contained in the feature importances in the learning phase. The second reason is kNN's simplicity: its only parameter is the number of neighbors k , which we set to 5. In the prediction stage, the neighbors' contributions to the predicted value are equally weighted, so we do not introduce additional parameters that would influence the performance.

The rationale for using kNN as an evaluation model is as follows. If a feature ranking is meaningful, then the feature importances used as weights in the calculation of distances should yield better predictive power as compared to not using these weights [10].

We assess the predictive performance with the average relative root mean squared error $\overline{\text{RRMSE}}$. If we denote the predicted value of the target y_j by $\hat{y}_j(\mathbf{x})$, the RRMSE for this target is defined as

$$\text{RRMSE}(y_j) = \sqrt{\frac{\frac{1}{|\mathcal{D}_{\text{TEST}}|} \sum_{\mathbf{x} \in \mathcal{D}_{\text{TEST}}} (y_j(\mathbf{x}) - \hat{y}_j(\mathbf{x}))^2}{\text{Var}_j(\mathcal{D}_{\text{TRAIN}})}},$$

and $\overline{\text{RRMSE}}$ can be expressed as $\overline{\text{RRMSE}} = \sqrt{\frac{1}{T} \sum_{j=1}^T \text{RRMSE}^2(y_j)}$.

4.4 Statistical Analysis of the Results

For comparing two algorithms, we use the Wilcoxon’s test, and for comparing more than two algorithms, we use the Friedman’s test [12]. In both cases, the null hypothesis H_0 is that all considered algorithms have the same performance. If H_0 is rejected by the Friedman’s test, we additionally apply Nemenyi’s post-hoc test [12] to investigate where the statistically significant differences occur. Finally, to control the false discovery rate, the Benjamini-Hochberg procedure [3] was applied: let p_i be the i -th smallest among the obtained p -values, and m the number of tests. Let i_0 be the largest i , such that $p_i \leq \frac{i}{m}\alpha =: \hat{\alpha}_i$. Then, we can reject the hypotheses that correspond to p -values p_i , for $1 \leq i \leq i_0$.

The results of the Nemenyi’s tests are presented by average ranks diagrams. Each diagram shows the average rank of each algorithm over the considered datasets, and the critical distance, i.e., the distance by which the average ranks of two algorithms must differ to be considered statistically significantly different. Additionally, the groups of algorithms among which no statistically significant differences occur are connected with a red line. In the analysis, the significance level was set to $\alpha = 0.05$.

4.5 Parameter Instantiation

The algorithm for inducing an ensemble of PCTs for MTR takes as input the following parameters: the number of base predictive models in the forest (all ensemble types), minimal number of examples in a leaf of a tree (all ensemble types), and the feature subset size (Random Forest only). In all cases, we grow 100 trees, whose leaves must contain at least two examples each. Additionally, the feature subset size in the case of Random Forests is set to $\lceil \sqrt{D} \rceil$.

Next, recall that the symbolic ranking requires selecting a value for w . In a preliminary study, we investigate the influence of several values of w , i.e., $w \in \{0.25, 0.5, 0.75, 1\}$, on the performance of feature ranking. We perform Friedman’s test with the null hypothesis H_0 that the four symbolic rankings perform equally well. It turns out that the differences among the rankings are not statistically significant in the case of Bagging (p -value is 0.418) and ET (p -value is 0.230), whereas in the case of RF, they are (p -value is 0.000697). In the RF case, we can proceed to Nemenyi’s test, whose results are shown in Fig. 1.

The diagram reveals that only the symbolic ranking with weight $w = 1.0$ is statistically significantly worse than the rankings with weights $w = 0.5$ and $w = 0.25$. This can be explained by Eq. 2: this value for the weight is the only one where the depth of the node where an attribute appears is not taken into account when computing the relevance.

Since the average ranks of the ranking methods with $w = 0.25$, $w = 0.5$, $w = 0.75$ and $w = 1.0$ are respectively 2.38, 2.25, 2.54 and 2.83 for Bagging, and 2.42, 2.38, 2.25 and 2.94 for ET, the ranking *Symb50* is a reasonable choice for all three ensembles, since it is always ranked at least second. The reason for this is less obvious but we hypothesize that it could be an artifact of the algorithm for inducing ensembles (see Table 1). Namely, splits in the ensemble trees are

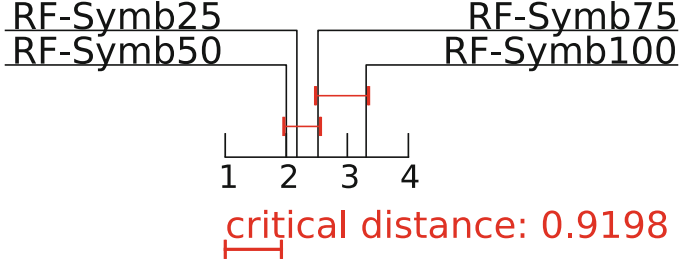


Fig. 1. The average ranks diagram for Nemenyi’s test, performed for the symbolic ranking methods with the Random Forest ensemble at significance level of $\alpha = 0.05$.

binary. If we assume that the best test in an internal node $\mathcal{N} \in \mathcal{T}$ partitions $E(\mathcal{N}) \subseteq \mathcal{D}$ approximately in half, then the attribute in \mathcal{N} ’s test influences one half of the instances that arrive to its parent; hence, the parent should receive twice as large a reward as each of its two children.

5 Results and Discussion

In this section, we present the results from the experimental evaluation, responding to the experimental questions posed above. The baseline kNN is denoted as 5NN, while the weighted 5NN is denoted by the combination of ensemble and ranking method (*ensemble-ranking*).

5.1 Are the Obtained Feature Rankings Relevant?

The investigation of whether a given feature ranking is relevant has a pivotal role in this work. More specifically, we investigate whether 5NN prediction can benefit from using the additional information from the feature importances. To this end, we compare the performance of 5NN without and with feature importances. We use the Wilcoxon’s test to assess the statistical significance of the differences in performance between the two 5NN methods.

Table 3 gives the results of the statistical evaluation. It shows that all hypotheses are rejected, since we have $p_i < \hat{\alpha}_i$ for all i . Therefore, we can conclude that using feature ranking is clearly beneficial, i.e., the obtained feature rankings (more precisely, feature importances) are relevant and meaningful. Considering that the answer to the first question is positive, we now proceed with discussing the remaining experimental questions.

5.2 Comparison of the Different Ranking Methods

We compare the performance of the different ranking methods when coupled with a given ensemble learning method. In other words, we perform three analyses, where the ensemble method is fixed in each analysis. The results of this analysis

Table 3. The results of the Wilcoxon’s tests that compare the performance of standard 5NN to its weighted-distance version. The i -th row contains the name of the *ensemble-ranking* pair that provided the feature importances and was tested against standard 5NN; the p -value p_i ; and the corrected value $\hat{\alpha}_i$.

<i>ensemble-ranking</i>	p_i	$\hat{\alpha}_i$
RF-Genie3	0.000146	0.006250
RF-Symb50	0.001938	0.012500
ET-Symb50	0.009776	0.018750
Bagging-RF	0.017592	0.025000
ET-Genie3	0.020120	0.031250
Bagging-Genie3	0.028920	0.037500
Bagging-Symb50	0.031316	0.043750
RF-RF	0.035411	0.050000

for Random Forests and Bagging are given in Fig. 2. For Random Forests, the Friedman test found that there are statistically significant differences among the three ranking methods with $p = 0.00316$. The follow-up Nemenyi test reveals that the performance of a feature ranking obtained with Genie3 is statistically significantly better than the Random Forest ranking.

The differences between the different rankings for the Bagging ensemble method are not statistically significant ($p = 0.347$) and we can note that the three ranking methods have close average ranks. Furthermore, the Wilcoxon test for the Extra Trees ensemble revealed that there is no statistically significant difference ($p = 0.191$) between the two rankings, but Genie3 has a better sum of ranks than Symb50. Finally, the Random Forest ranking should be avoided: it has the worst computational complexity and it is consistently the worst performing ranking method (both for the Bagging and RF ensemble method).

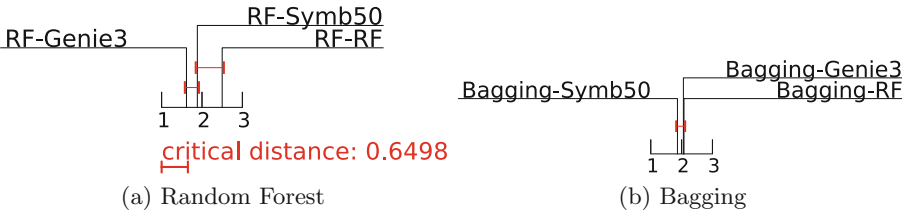


Fig. 2. The average ranks diagrams for Nemenyi’s post-hoc test at a significance level of $\alpha = 0.05$, performed for the rankings RF, Genie3 and Symb50, for (a) RF ensemble method and (b) Bagging ensemble.

5.3 Comparison of the Different Ensemble Methods

We also compare the performance of the different ensemble methods when used together with a given ranking method. In other words, we perform three analyses where the ranking method is fixed in each analysis. The Friedman test for Genie3 and Symb50 and the Wilcoxon test for Random Forest ranking did not reject the null hypotheses with p-values 0.403, 0.346 and 0.176, respectively. Nevertheless, top ranked ensemble methods for the given ranking methods are: Random Forests for Genie3, and Bagging for Symb50 and Random Forests ranking.

5.4 Selecting the Best Ensemble-Ranking Pair

Finally, one of the goals of this paper is to select the best ensemble-ranking pair of methods. For this purpose, we evaluate the performance of the 8 ensemble-ranking pairs by performing a Friedman test. It reveals that there are statistically significant differences in performance among the methods and the results of the post hoc Nemenyi test are shown in Fig. 3. The average ranks diagram shows that the best performing pair is the Random Forest ensemble method coupled with the Genie3 ranking method. Moreover, the best performing method pair is statistically significantly better than the worst performing method pair (Random Forest ensembles coupled with Random Forest ranking).

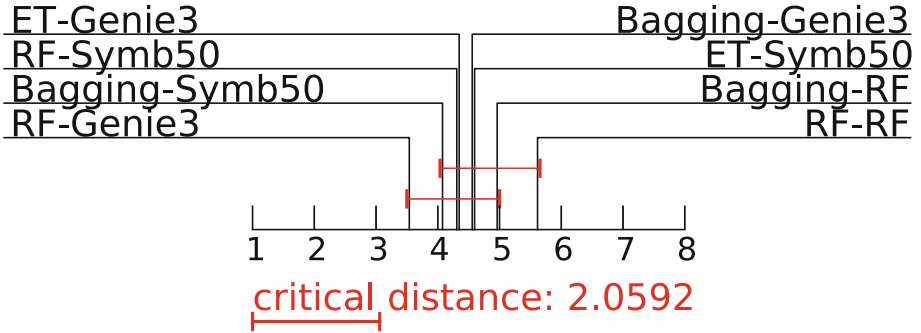


Fig. 3. The average ranks diagram from the Nemenyi's post-hoc test, performed for all *ensemble-ranking* pairs, at a significance level of $\alpha = 0.05$.

6 Conclusions

In this work, we proposed three base feature ranking methods that can be coupled with three ensemble learning methods. We investigated and evaluated eight *ensemble-ranking* options. These are the first methods that can address the task of feature ranking in the case of MTR with numeric and nominal attributes. More specifically, we extend Genie3, Random Forest and Symbolic ranking towards the task of MTR. We then coupled these rankings with the following ensemble

learning methods: Bagging, Random Forests and Extra trees – all of which use predictive clustering trees as base predictive models.

We perform an extensive experimental evaluation of the proposed feature ranking methods using 26 benchmark MTR datasets. The evaluation is based on a 5NN predictive model that uses the obtained feature importances as weights.

The results show that all of the proposed eight methods yield a relevant feature ranking, i.e., the 5NN predictive models that use the feature importances as weights statistically significantly outperform the standard 5NN. Next, the best values for the weight parameter of the Symbolic ranking is 0.5. Furthermore, the best performing method is the one that uses Random Forests for learning the ensemble and Genie3 for calculating the feature importances. Moreover, this method is also computationally efficient: Random Forests are among the most efficient ensemble learning methods and Genie3 adds just a small computational cost of a single traversal of each tree in the ensemble.

We plan to extend this work along three major directions. First, we will compare the proposed methods to methods that use the data transformation approach, i.e., transform a MTR problem to a set of STR problems, coupled with a feature ranking algorithm for STR. Second, we will extend the proposed method to other structured output prediction tasks, such as multi-label classification, and hierarchical multi-label classification. Finally, we will investigate the influence of the ensemble size on the produced feature rankings.

Acknowledgments. We would like to acknowledge the support of the European Commission through the project MAESTRA - Learning from Massive, Incompletely annotated, and Structured Data (Grant number ICT-2013-612944), and of the Slovenian Research Agency through a young researcher grant.

References

1. Kaggle: Online product sales. <https://www.kaggle.com/c/online-sales>. Accessed 05 May 2017
2. Kaggle: See click predict fix. <https://www.kaggle.com/c/see-click-predict-fix>. Accessed 05 May 2017
3. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. Ser. B (Methodol.)* **57**(1), 289–300 (1995)
4. Blockeel, H.: Top-down induction of first order logical decision trees. Ph.D. thesis, Katholieke Universiteit Leuven, Leuven, Belgium (1998)
5. Borchani, H., Varando, G., Bielza, C., Larrañaga, P.: A survey on multi-output regression. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **5**(5), 216–233 (2015)
6. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996)
7. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
8. Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton (1984)
9. Brobbey, A.: Variable Selection in Multivariate Multiple Regression. Master's thesis, Department of Mathematics and Statistics, Memorial University, Newfoundland and Labrador, Canada (2015)

10. Cunningham, P., Delany, S.J.: k-Nearest Neighbour Classifiers. Technical report 2, University College Dublin (2007)
11. Demšar, D., Debeljak, M., Džeroski, S., Lavigne, C.: Modelling pollen dispersal of genetically modified oilseed rape within the field. In: Proceedings of the 9th Annual Meeting of the Ecological Society of America. p. 152 (2005)
12. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006)
13. Džeroski, S., Demšar, D., Grbović, J.: Predicting chemical parameters of river water quality from bioindicator data. *Appl. Intell.* **1**(13), 7–17 (2000)
14. Geurts, P., Erns, D., Wehenkel, L.: Extremely randomized trees. *Mach. Learn.* **36**(1), 3–42 (2006)
15. Goovaerts, P.: Geostatistics for Natural Resources Evaluation. Oxford University Press, New York (1997)
16. Hansen, L.K., Salamon, P.: Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.* **12**(10), 993–1001 (1990)
17. Hatzikos, E.V., Tsoumakas, G., Tzanis, G., Nick, B., Vlahavas, I.P.: An empirical study on sea water quality prediction. *Knowl. Based Syst.* **21**(6), 471–478 (2008)
18. Huynh-Thu, V.A., Irrthum, A., Wehenkel, L., Geurts, P.: Inferring regulatory networks from expression data using tree-based methods. *PLoS One* **5**(9), 1–10 (2010)
19. Kampichler, C., Džeroski, S., Wieland, R.: Application of machine learning techniques to the analysis of soil ecological data bases: relationships between habitat features and Collembolan community characteristics. *Soil Biol. Biochem.* **32**(2), 197–209 (2000)
20. Karalič, A., Bratko, I.: First order regression. *Mach. Learn.* **26**(2–3), 147–176 (1997)
21. Kocev, D., Džeroski, S., White, M., Newell, G., 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)
22. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. *Pattern Recognit.* **46**(3), 817–833 (2013)
23. 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)
24. Stańczyk, U., Jain, L.C. (eds.): Feature Selection for Data and Pattern Recognition. Studies in Computational Intelligence. Springer, Heidelberg (2015)
25. Stojanova, D.: Estimating Forest Properties from Remotely Sensed Data by using Machine Learning. Master's thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia (2009)
26. Stojanova, D., Panov, P., Gjorgjioski, V., Kobler, A., Džeroski, S.: Estimating vegetation height and canopy cover from remotely sensed data with machine learning. *Ecol. Inform.* **5**(4), 256–266 (2000)
27. Todorovski, L., Blockeel, H., Džeroski, S.: Ranking with predictive clustering trees. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) *ECML 2002. LNCS (LNAI)*, vol. 2430, pp. 444–455. Springer, Heidelberg (2002). doi:[10.1007/3-540-36755-1_37](https://doi.org/10.1007/3-540-36755-1_37)
28. Tsanas, A., Xifara, A.: Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy Build.* **49**, 560–567 (2012)
29. Yeh, I.C.: Modeling slump flow of concrete using second-order regressions and artificial neural networks. *Cem. Concr. Compos.* **29**, 474–480 (2007)