International Journal of Machine Learning and Cybernetics
https://doi.org/10.1007/s13042-021-01329-1

ORIGINAL ARTICLE

Rotation Forest for multi-target regression

Juan J. Rodríguez[1] · Mario Juez-Gil[1] · Carlos López-Nozal[1] · Álvar Arnaiz-González[1]

Received: 31 January 2020 / Accepted: 7 April 2021

Abstract
The prediction of multiple numeric outputs at the same time is called multi-target regression (MTR), and it has gained attention during the last decades. This task is a challenging research topic in supervised learning because it poses additional difficulties to traditional single-target regression (STR), and many real-world problems involve the prediction of multiple targets at once. One of the most successful approaches to deal with MTR, although not the only one, consists in transforming the problem in several STR problems, whose outputs will be combined building up the MTR output. In this paper, the Rotation Forest ensemble method, previously proposed for single-label classification and single-target regression, is adapted to MTR tasks and tested with several regressors and data sets. Our proposal rotates the input space in an efficient and novel fashion, avoiding extra rotations forced by MTR problem decomposition. Four approaches for MTR are used: single-target (ST), stacked-single target (SST), Ensembles of Regressor Chains (ERC), and Multi-target Regression via Quantization (MRQ). For assessing the benefits of the proposal, a thorough experimentation with 28 MTR data sets and statistical tests are used, concluding that Rotation Forest, adapted by means of these approaches, outperforms other popular ensembles, such as Bagging and Random Forest.

Keywords Multi-target regression · Ensemble · Rotation Forest

1 Introduction

In supervised learning, the task is to operate on the values of the input variables, in order to predict the output-variable values. Conventional classification and regression tasks only have one output, although some applications will generate several outputs. If the outputs are binary, the task is referred to as multi-label classification [31], and if nominal, it is a multi-dimensional classification task. When the outputs are real values, the task is known as multi-target regression [13] or multi-output, multi-variate, and multi-response regression.

Multi-target regression (MTR) applications include time-series predictions of drug efficacy [46], predicting poultry meat characteristics [60], predicting shape deformation after breast conserving surgery [84], yield curve forecasting [50], and water quality monitoring [49], among others.

The capabilities of ensemble methods to improve the prediction of single models are widely acknowledged. A popular ensemble is Rotation Forest, which was initially proposed for classification [58] and later adapted to regression [52] but, until now it has not been adapted and validated for MTR. The keystone of Rotation Forest relies on transforming the input space by means of Principal Component Analysis (PCA), what makes it more powerful than other ensemble approaches such as Bagging or Random Forest. Rotation Forest for STR can be used for MTR using problem transformation methods. These methods convert original MTR problems into several STR problems. The drawback of using MTR transformation methods with Rotation Forest for STR is that as many rotations of the data sets are required as the number of STR tasks multiplied by the ensemble size. Let us consider that a single-target task for each target is used and that the number of targets is 16 and the ensemble size is 100; then, the number of rotations that will be performed on the data set will be 1600 (see Fig. 1). Nevertheless, the rotated data sets for one target are also valid for the other targets, implying some wasted effort.

✉ Álvar Arnaiz-González
alvarag@ubu.es

Juan J. Rodríguez
jjrodriguez@ubu.es

[1] Department of Computer Science, Universidad de Burgos, Avda. Cantabria s/n, 09006 Burgos, Spain

Published online: 22 April 2021

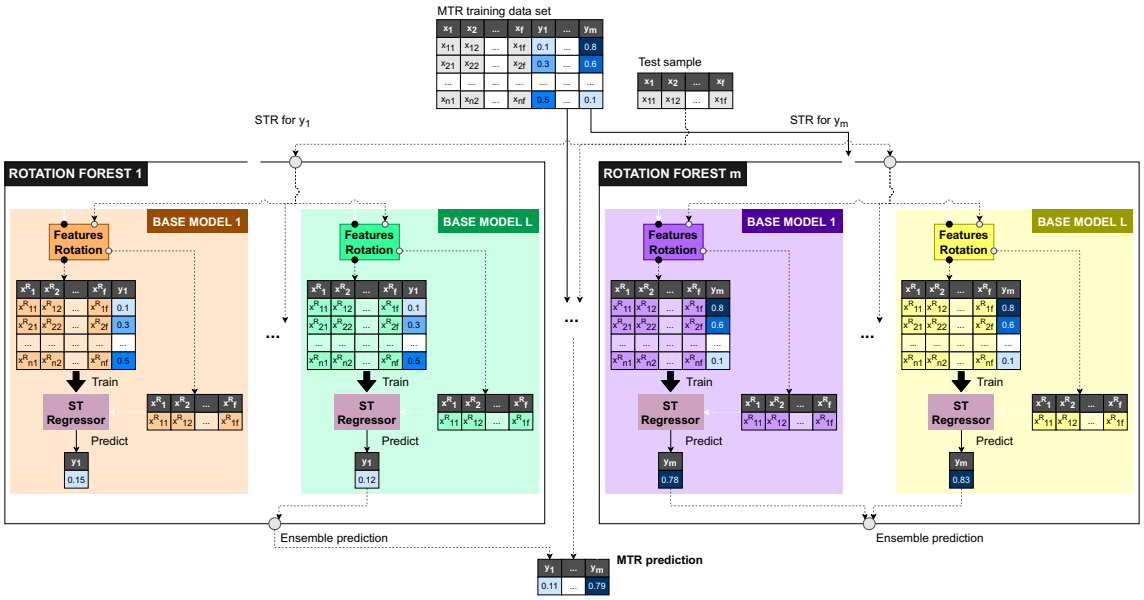

Fig. 1 An example of using Rotation Forest for MTR by means of ST. As many rotations (each one depicted with a different color) as the number of single-target tasks multiplied by the ensemble size ($m \times L$), are performed

The alternative approach considered in this paper is to use Rotation Forest as an ensemble method where each base model is a MTR model. In this way, the number of rotations of the data set will be limited to the size of the ensemble (e.g., 100), regardless of the number of targets and the approach used to deal with MTR (see Fig. 2).

The rest of the paper will be organized as follows. Some of the approaches for MTR will be presented in Sect. 2. In Sect. 3 the Rotation Forest method and the proposed for MTR will be introduced. The experimentation, the results and their discussion will be presented in Sect. 4 and, finally, the conclusions will be advanced in Sect. 5.

2 Multi-target regression

Multi-target regression methods can be categorized in two approaches [13]: *algorithm adaptation* and *problem transformation*. Each one is thoroughly explained below.

2.1 Algorithm adaptation

In these methods, which are extensions of STR algorithms to MTR, a single model (trained with the original multi-output data set) generates predictions for all outputs.

The first approaches for MTR were statistical [13], taking advantage of correlations between the targets [1, 16, 34, 70].

Neural networks can easily be extended to MTR, by for instance using one neuron for each target in the output layer. These MTR methods include [19, 20, 27].

Support vector regression (SVR) has also been extended to MTR [59, 71, 76, 83]; as well as other kernel methods and Gaussian processes [5].

Several MTR approaches are based on regularization methods [62] where the regularization term depends on all the targets and their relationships. Some of these approaches are [8, 35, 51, 81].

In MTR trees, all the targets are predicted with a single tree where each leaf generates predictions for each target. An extension of CART was proposed in [21], redefining the impurity measure of a node, so that all the targets were taken into account. In [65], the trees were constructed according to size and accuracy constraints. A stepwise approach for building model trees was proposed in [6], where the leaves of the tree were linear regression models. Another method for building model trees based on regularization was proposed in [36]. Moreover, MTR trees were also used in ensembles such as bagging and random forest [41]. A random forest method based on multi-objective optimization and alternative splitting strategies was proposed in [2].

Regression rules were also adapted to MTR, and ensembles composed of regression rules were also proposed in [3, 4].

2.2 Problem transformation

In problem transformation methods, the multi-target task is reduced to several single-target tasks [18, 43]. This reduction consists of a transformation of the original multi-output data set into several single-target data sets. Once the data set has

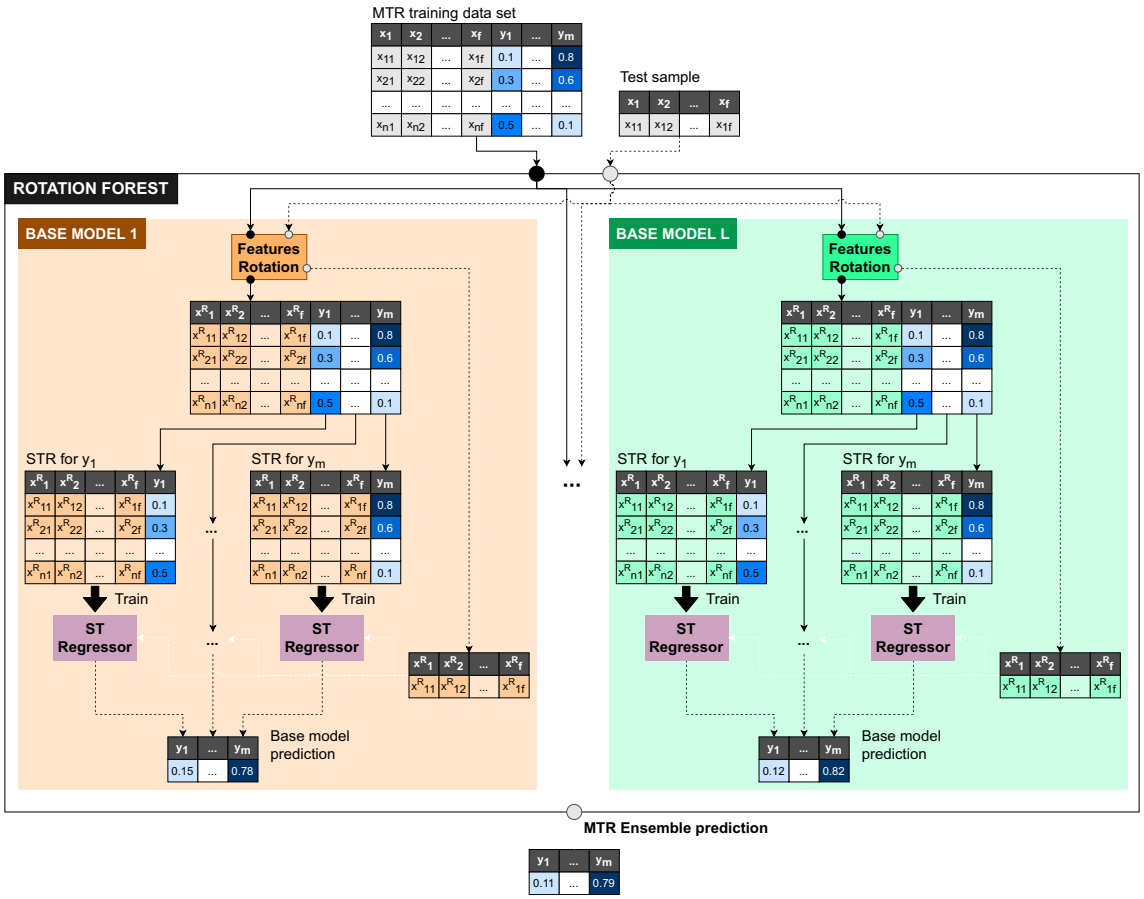

Fig. 2 An example of the Rotation Forest ensemble proposed in this paper, where data set rotation is performed before applying the MTR approach. All single-target tasks of a specific base model share the same rotation of the data (depicted using the same color)

been transformed, any single-label or single-target algorithm can be used without any change.

The most straightforward method is to have a single-target task for each of the m targets. This approach is known as *single-target* (ST) [62] and is equivalent to the *binary relevance* method for multi-label classification. The reason for considering other approaches is that the relations between the targets can provide useful prediction-related information; relations that are completely ignored with the ST method.

Stacked single-target (SST) [62] is an application of stacked generalization [75] to MTR, based on a multi-label method [28]. Training is performed in two stages: in the first, a set of m single-target models is built, one for each target, as in ST; in the second stage, another set of m models is built. The training data for the second stage models are the original training data augmented with m features: the estimates of the targets from the first set of models. In prediction, firstly the models from the first stage are used to produce initial predictions for all m targets. Then, those predictions are used to augment the input that is, in turn, applied to the second stage models, which are used to produce the final predictions.

In SST, the training data for the second stage models have m additional features, one for each output. One issue is which values use for this features in the training data. One option is to use the actual output values, this variant is named SST-true. A possible problem with these approach is that in testing time the actual values are unknown and the output of the first stage models will be used. Another option, SST-train, is to use the values predicted by the first stage models for the training data for the second stage models. This second approach could overfit. Hence, a third approach, named SST-cv, is to use an internal cross validation; as it is done in stacking for singe outputs.

Ensemble of regressor chains (ERC) [62] is the adaptation of the ensemble of classifier chains method [56] to MTR. A regressor chain is formed of m STR models and has an associated random permutation of the targets. The first model predicts the first target in the permutation using the original features. The second model predicts the second

target, but using the original features as inputs, augmented with the prediction for the first target. In general, the i-th model predicts the i-th target in the permutation, using the original features and the predictions given by the previous $i-1$ targets as inputs. Thus, the predictions of a random chain will depend on the order of the targets in the permutation. An ERC is composed of several regressor chains, each one with its random permutation, but also built using bootstrap samples of the training set. The predictions of an ERC are the averages of the predictions of its members. As for SST, there are three variants: ERC-true, ERC-train, and ERC-cv.

Since some permutations can be more accurate than others, instead of using random permutations for ERC, a permutation based on some more sophisticated criterion may be selected. In [48], for example, the permutation is selected according to the correlations between targets. In [2] a model is built for each of the remaining targets and the one with the smallest error is selected for the chain.

As has been explained, the targets within a regressor chain are organized in a sequence. Alternatively, the targets within chaining trees [47] are organized in a tree, in which the targets appear as nodes. In each node, an ST regressor predicts the corresponding target, using the input features augmented with the predictions in the offspring.

In [80], the MTR problem is reduced to a single STR problem. The features are augmented with m binary features, one for each label. From among the m features, only one will be 1 and the others will be 0. These m features indicate the label to which the output corresponds. For each original training instance, there will be m instances within the resulting data set, one for each target.

In Random Linear Target Combinations [69], new targets are constructed using random linear combinations of the original targets, and an STR model is built for each one. The predicted values for the original targets are obtained by inverting the linear combinations.

In multi-target regression via target specific features [73], an STR model is built for each target. Additional features are included in the data sets used for training the models. Hierarchical clustering is used to group the targets, and a categorical feature with the cluster assignment is included in the data sets for all the targets. For each target, specific features are included, which are distances to cluster centers. These clusters group the instances according to instance similarity matrices obtained for each target. An initial similarity matrix is obtained with boosting for each target and its final similarity matrix combines the matrices of all the targets taking into account the similarity between the targets.

It is also possible to reduce MTR to classification problems. In [63], *Multi-target regression via quantization* (MRQ) was proposed. The k-means method is used to divide the output space in clusters. The MTR problem is then reduced to multi-class classification, where the classes are the obtained clusters. Given the predicted class, the predictions for the original outputs are the values of the centroid of the corresponding cluster. In the ensemble version (eMRQ) multiple quantizers are used, each one encoding a random subset of the outputs.

2.3 Data reduction

As with other supervised learning tasks, pre-processing methods for data reduction are useful in MTR. First, the results from the pre-processed data sets can outperform the results from the original data sets. Second, the computing time and the memory necessary for building and using the regressors can be reduced.

The feature selection method proposed in [45] selects features by applying a hierarchical clustering strategy based on information measures. Two groups of feature ranking methods were proposed in [53], one based on scores calculated from ensembles of predictive clustering trees and the other on the RReliefF method. Dimensionality reduction methods decrease the number of features, but instead of selecting a subset of features, these methods create new features. Methods for MTR include [78, 82], the method based on regularization proposed in [8] can be used for both selecting and learning features.

In [57] the authors adapted DROP [74] to MTR and proposed an ensemble of instance selectors for prototype or instance selection. The ensemble was composed of $m+1$ selectors, one of which used the original data set, while one of the outputs was used as an additional input rather than an output in the others. In [43], a multi-objective evolutionary algorithm was used for instance selection.

2.4 Ensemble methods

When we, as humans, have to make important decisions, we usually ask for different opinions. We do so, because we want to reduce the probability of an erroneous decision. Ensemble methods follow the same idea. An ensemble is formed by multiple base models, the predictions of which are combined. It has been demonstrated that the generalization performance of the combined decision outperforms, at least, all but one of the base models [44, 55]. The performance of the ensembles rely on the diversity of the base models. There is no point in using several models, if all of them predict the same output.

Problem transformation approaches can be considered ensemble methods, because the final model is composed of several models. Nevertheless, the ensemble and the base models solve different tasks: respectively, MTR and STR.

Ensemble methods composed of MTR base models are also possible. For instance, Bagging [14] and Random

Subspaces [32] can be used directly for MTR, although they were not originally proposed for multiple outputs. The method proposed in this paper, Rotation Forest for MTR, also belongs to that group. The base models for an MTR ensemble can be obtained with any MTR method, including problem reduction and algorithm adaptation methods.

One example is Ensemble of Regressor Chains (ERC), which is bagging using a problem transformation method (regressor chains) as the base model.

Stacked single-target (SST) is a problem transformation method, but it is also an application of stacked generalization combining two ST models: one composed by the regressors built on the original features, and another built on the original features augmented with the predictions of the previous model. The same approach can be used with any other MTR method instead of ST.

Ensembles with random output selections were proposed in [17]. MTR models form the ensemble, but they only predict a randomly selected subset of the targets, except that the ensemble also includes a model that predicts all the targets. These random output selections when are combined with bagging or random forest increase the diversity of the results.

Another method where the models in the ensemble predict a subset of the targets was proposed in [66]. In that case, the targets were partitioned in non-overlapping subsets. Hence, each target was predicted by only one model, but one model could predict several targets.

3 Rotation Forest

The main idea of Rotation Forest [58] is to train each model in the ensemble using a different rotation of the training data. If the method used for training the base models is sensitive to rotations, as is the case for decision or regression trees, the base models can be very different. Moreover, these models can also be accurate because no information is lost with a rotation.

3.1 Related methods

In the original paper where Rotation Forest is presented [58], the rotations were obtained using multiple PCA over subsets of features and instances and the base classifiers were decision trees. Nevertheless, similar approaches can be used with other methods for building the base models and obtaining the rotations. In fact, the main idea can be applied with any transformation of the data set, not just with rotations.

In Random Rotation Ensembles [12] the rotation matrices are random. Supervised subspace projections are used in [26]. A Feature Weighted Rotation Forest is proposed in [72]. Kernel Rotation Forest [61] are based on kernel PCA. RotBoost [79] combines Rotation Forest with AdaBoost. It is also possible to combine Rotation and Random Forest [64]. Hybrid Rotation Forest [9] uses Extreme Learning Machines for building the base classifiers. Rotation Forest fuzzy rule-based Classifier Ensemble is proposed in [54]. As can be noted, several variations and many proposals have been presented since the original Rotation Forest was published, but this paper focuses on the main idea of using PCA, for transforming the input space, and using trees as base models.

3.2 Rotation Forest for multi-target regression

Rotation Forest for MTR can be used for combining models obtained with any MTR algorithm, including problem transformation and algorithm adaptation methods. As was presented before, the drawback of using Rotation Forest directly for MTR is the waste of effort by performing many rotations (once for each single-target task times the ensemble size). For avoiding this, the proposal presented below performs as many rotations, of the input space, as the size of the ensemble. Therefore, only one rotation is performed for all STR tasks of a single base regressor from the MTR algorithm chosen. The Rotation Forest for MTR is presented in Algorithm 1.

Algorithm 1: Rotation Forest for MTR.

Input: A training set $(\mathbf{X}, \mathbf{Y})$ where $\mathbf{X} = \{\mathbf{x}_1, \ldots \mathbf{x}_n\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \ldots \mathbf{y}_n\}$, ensemble size L, number of feature subsets K, base learner.
Output: Ensemble E

for $t \leftarrow 1, \ldots, L$ **do**
  Prepare the rotation matrix $\mathbf{R}_t^a$:
  begin
    Randomly split $\mathbf{F}$ into K subsets $\mathbf{F}_{t,k}$
    for $k \leftarrow 1, \ldots, K$ **do**
      $\mathbf{X}_{t,k} \leftarrow$ submatrix of $\mathbf{X}$ for the features in $\mathbf{F}_{t,k}$
      $\mathbf{X}'_{t,k} \leftarrow$ submatrix of $\mathbf{X}_{t,k}$ with the instances with smallest values in a random projection of $\mathbf{Y}$
      $\mathbf{X}''_{t,k} \leftarrow$ bootstrap sample from $\mathbf{X}'_{t,k}$
      $\mathbf{C}_{t,k} \leftarrow$ rotation matrix from $\texttt{PCA}(\mathbf{X}''_{t,k})$
    $\mathbf{R}_t \leftarrow$ arrangement of the $\mathbf{C}_{t,k}$ matrices in a single rotation matrix
    $\mathbf{P}_t \leftarrow$ permutation matrix, matching the order of the features in $\mathbf{F}$
    $\mathbf{R}_t^a \leftarrow \mathbf{P}_t \mathbf{R}_t$ `// Rearrangement of` $\mathbf{R}_t$
  $D_t \leftarrow \texttt{build-multi-target-regressor}(\mathbf{X}\mathbf{R}_t^a, \mathbf{Y})$
$E \leftarrow \bigcup_{t=1}^{L} D_t$ `// For` $\mathbf{x}$, E `predicts` $E(\mathbf{x}) = \frac{1}{L}\sum_{t=1}^{L} D_t(\mathbf{x})$

Each of the L models in the ensemble, D_t, is built with a different rotation of the training data with each rotation defined by a rotation matrix. The procedure for obtaining each rotation matrix, $\mathbf{R}_t^a$, is as follows. The features are divided into groups where the number of groups, K, is an argument of the method. By default, K is selected so that the number of features in each group is three[1].

A data set, $\mathbf{X}_{t,k}$, is considered for each group, formed only of the features of each group, $\mathbf{F}_{t,k}$. In the Rotation Forest method for classification [58], this data set is filtered, removing all the instances of a proper subset of the classes. For MTR it is also desirable to select a subset of the instances according to the output values, a random projection of the outputs is taken and the selected instances are those with the lowest values in this projection. The number of selected instances is randomly selected, by default between 10 and 50% of the number of training instances.

A bootstrap sample, with a default sample size of 50% is taken from those selected instances. Then, Principal Component Analysis (PCA) is applied to the sample, generating a rotation matrix, $C_{t,k}$.

The removal of instances with the lowest values from a random projection and in the bootstrap sample is for the sake of diversity. The same subset of features can be selected for several ensemble models (especially if the number of features is low). If the rotations are obtained by means of PCA using all the training instances, then the rotated features should be the same.

We therefore have a rotation matrix for each group of features that defines a new set of features on the basis of the PCA components. All the components of all the groups are gathered together in a single rotation matrix, $\mathbf{R}_t$. This matrix could not be used directly on the training data, because the features are not in the same order, so the rotation matrix is rearranged using a permutation matrix, $\mathbf{P}_t$, that simply reorders the features, yielding the final rotation matrix, $\mathbf{R}_t^a$.

Finally, the training data set is rotated using the rotation matrix, then the rotated data set is used to build the base MTR model.

In the prediction stage, the instance $\mathbf{x}$ to be predicted, is rotated as many times as the ensemble size, using the corresponding rotation matrices. For each rotated instance, the corresponding base model generates a prediction and the predictions of the base models are then combined in an average value.

4 Experiments and results

The objective of the experimentation is to validate the Rotation Forest for MTR proposal, and to compare its performance with other MTR ensembles. Regarding the different problem transformation approaches, other objective of this experimentation is to assess which performs better used along Rotation Forest.

[1] If the number of features is not a multiple of 3, the last group is completed with previously selected features.

Table 1 Data sets characteristics

Data set	Examples	Features		Targets	References
		Num.	Nom.		
andro (Andromeda)	49	30	0	6	[30, 62]
atp1d (airline ticket price)	337	411	0	6	[62]
atp7d (airline ticket price)	296	411	0	6	[62]
cal-housing	1 032	7	0	2	[48]
edm (electrical discharge machining)	154	16	0	2	[40]
enb (energy efficiency building)	768	8	0	2	[62, 67]
fri-c0-500-25 (Friedman)	500	20	0	6	[48]
jura	359	15	0	3	[29, 62]
m5spec	699	80	0	3	[48]
mp5spec	699	80	0	4	[48]
mp6spec	699	80	0	4	[48]
oes10 (occupational employment survey)	403	298	0	16	[62]
oes97 (occupational employment survey)	334	263	0	16	[62]
osales (online sales)	639	401	0	12	[38, 62]
polymer	61	10	0	4	[48]
puma32H	819	27	0	6	[48]
puma8NH	2 457	6	0	3	[48]
rf1 (river flow)	9 125	64	0	8	[62]
rf2 (river flow)	9 125	576	0	8	[62]
scm1d (supply chain management)	9 803	280	0	16	[62]
scm20d (supply chain management)	8 966	61	0	16	[62]
scpf (see click predict fix)	1 137	23	0	3	[39, 62]
sf1 (solar-flare)	323	0	10	3	[23]
sf2 (solar-flare)	1 066	0	10	3	[23]
slump	103	7	0	3	[62, 77]
stock	950	7	0	3	[48]
Wisconsin (breast cancer)	194	31	0	2	[48]
wq (water quality)	1 060	16	0	14	[24]

Firstly, the experimental setup is presented in Sect. 4.1. Secondly, the results and their discussion are reported in Sect. 4.2.

4.1 Experimental setup

The description of the data sets, the algorithms used in the experimentation, and the evaluation measures used for assessing the performance of the methods are presented below.

4.1.1 Data sets

Table 1 shows the data sets characteristics, obtained from [62][2] and [48][3]. The data sets are very varied in nature (management, forecast, medicine, astronomy...), in size (the number of examples range between 49 and 9 803), in the number of features (between 6 and 576), and in the number of targets (between 2 and 16).

4.1.2 Methods

Nine approaches for MTR were used:

- Single Target (ST).
- Three variants of Stacked Single Target (SST): SST-true, SST-train, and SST-cv.
- Three variants of Ensembles of Regressor Chains (ERC): ERC-true, ERC-train, and ERC-cv.
- Multi-target Regression via Quantization (MRQ) and the ensemble version of it (eMRQ).

In SST-true and ERC-true, when targets are used as inputs, their values are the true values. The values in SST-train and ERC-train are the predictions given by the regressors. 10 fold cross validation (without repetition) was used to obtain

[2] http://mulan.sourceforge.net/datasets-mtr.html.

[3] http://people.vcu.edu/~acano/MTR-SVRCC/datasets.zip.

the values for SST-cv and ERC-cv. These approaches were taken from [62], where SST and ERC had the best results in a comparison that also included Multi-Objective Random Forest [42], Trace Norm Regularization multi-task learning [7], a Dirty model for multi-task learning [35], and Random Linear target Combinations [69].

ST, SST, and ERC combine single target regressors. In [62], the methods that were considered for STR were ridge regression, regression tree, support vector regression, and bagging (with regression trees) and stochastic gradient boosting. The best results were achieved by bagging. MRQ and eMRQ use a method for classifier construction. In [63] bagging was also used, but with classification (instead of regression) trees.

There are five variants for each of the nine MTR approaches, depending on the STR method: BagP, BagU, RanF,RotF, and RotRanF. Bagging with pruned trees (BagP) was included, because it was the method used in [62] and [63]. Bagging with unpruned trees (BagU) was included, because Bagging usually works better with more unstable base classifiers, and pruning makes decision trees less diverse. Random Forest [15] (RanF) is a variant of Bagging that uses more unstable trees as base classifiers (Random trees). The fourth variant, Rotation Forest (RotF), was used with unpruned trees. Finally, RotRanF also used the Rotation Forest method but using random trees (i.e., the trees used in Random Forest).

The order of the multi-target approaches and the regression methods associated with each name indicates the order in which they were applied, i.e. which one is the argument. Hence, BagP, BagU, and RanF appear as suffixes, while RotF and RotRanF appear as prefixes. For instance, in SST-true-RanF, the SST-true approach was used with RanF as an argument. In RotF-SST-true, RotF was used with SST-true as argument. The base regressor for this SST-true was a single unpruned regression tree and, likewise, for all the cases where RotF was used. In all the cases where RotRanF was used, such as RotRanF-SST-true, the base regressor was a single random tree.

4.1.3 Settings

The performance was measured using aRRMSE [13, 62], the average Relative Root Mean Squared Error. The RRMSE was calculated for each of the, m, targets and the values were averaged:

$$\frac{1}{m}\sum_{j=1}^{m}\sqrt{\frac{\sum_{\mathbf{y}\in\mathbf{Y}_{test}}(\mathbf{y}_j-\hat{\mathbf{y}}_j)^2}{\sum_{\mathbf{y}\in\mathbf{Y}_{test}}(\mathbf{y}_j-\overline{\mathbf{y}}_j)^2}}$$

where, $\mathbf{Y}_{test}$ are the outputs for the test set and, for target j, $\mathbf{y}_j$ is the actual value, $\hat{\mathbf{y}}_j$ is the prediction given by the model, and $\overline{\mathbf{y}}_j$ is the average value in the *training* data.

Table 2 Results for methods with ST

	ST-BagP	ST-BagU	ST-RanF	RotF-ST	RotRanF-ST
andro	0.6016	0.5281	0.5142	*0.4906*	0.5722
atp1d	0.3735	*0.3692*	0.4021	0.3694	0.4133
atp7d	0.5248	0.4978	0.5297	*0.4807*	0.5665
cal-housing	0.6441	0.6207	0.6398	*0.5708*	0.5811
edm	0.7421	0.7295	0.7319	*0.7041*	0.7076
enb	0.1166	0.1140	0.1082	0.1080	*0.1038*
fri-c0-500-25	*0.9214*	0.9279	0.9335	0.9387	0.9484
jura	0.5891	0.5882	0.5768	*0.5698*	0.5701
m5spec	0.5503	0.5787	0.6122	*0.0525*	0.0725
mp5spec	0.5112	0.5344	0.5728	*0.0521*	0.0759
mp6spec	0.5167	0.5405	0.5764	*0.0478*	0.0791
oes10	0.4200	*0.4084*	0.4101	0.4387	0.4175
oes97	0.5248	0.5159	0.5095	*0.5043*	0.5179
osales	0.7479	0.7204	0.7107	0.7278	*0.6850*
polymer	0.6186	0.5673	*0.5024*	0.5580	0.6426
puma32H	0.8716	*0.8627*	0.9340	0.9129	0.9485
puma8NH	0.8140	0.7913	0.7741	*0.7706*	0.7715
rf1	0.0796	0.0882	0.0735	*0.0533*	0.0721
rf2	0.0866	0.0910	0.0849	0.0841	*0.0703*
scm1d	0.3189	0.3016	0.2913	*0.2820*	0.3010
scm20d	0.4066	0.3676	0.3672	*0.3483*	0.3820
scpf	0.8371	0.9504	0.8260	*0.8166*	0.8521
sf1	*1.1354*	1.4839	1.6386	1.3705	1.5562
sf2	*1.1494*	1.4487	1.4231	1.4105	1.4529
slump	0.6878	0.6699	0.6717	*0.6098*	0.6466
stock	0.1840	0.1667	0.1462	*0.1387*	0.1443
Wisconsin	*0.9314*	0.9582	0.9564	0.9349	0.9385
wq	0.9083	0.9111	0.9073	*0.8988*	0.9071
Mean	0.6005	0.6190	0.6223	*0.5444*	0.5713
Average rank	3.7143	3.3929	3.2143	*1.6786*	3.0000

The best result for each dataset is highlighted in italics

The experiments were performed using Mulan [68]. The results were obtained using a tenfold cross validation. There is an inner loop that only uses the training data for the cross-validation of SST-cv and ERC-cv.

The ensemble size for BagP, BagU, RanF, RotF, and RotRanF was set at 100. The ensemble size (the number of chains) for ERC with BagP, BagU, and RanF was set at 10. As ERC with these methods is an ensemble of ensembles, the number of regressors for each target was set at 10×100 (as in [62]). The ensemble size of ERC was set at 1, for the methods with RotF-ERC (e.g. RotF-ERC-true) and RotRanF, but there were 100 chains[4], because 100 was the ensemble

[4] There can be repetitions among these chains, specially if the number of targets is low.

Table 3 Results for methods with SST-true

	SST-true-BagP	SST-true-BagU	SST-true-RanF	RotF-SST-true	RotRanF-SST-true
andro	0.6029	0.5430	0.5212	*0.5029*	0.5844
atp1d	0.3757	0.3722	0.4008	*0.3684*	0.4088
atp7d	0.5610	0.5266	0.5270	*0.4964*	0.5701
cal-housing	0.7642	0.7201	0.6676	0.6105	*0.5953*
edm	0.7471	0.7285	0.7209	*0.7044*	0.7045
enb	0.1448	0.1501	0.1261	0.1259	*0.1113*
fri-c0-500-25	*0.9218*	0.9296	0.9385	0.9358	0.9473
jura	0.5942	0.5905	0.5787	0.5707	*0.5698*
m5spec	0.5526	0.5806	0.6255	*0.0536*	0.0727
mp5spec	0.5145	0.5376	0.5852	*0.0531*	0.0773
mp6spec	0.5210	0.5433	0.5862	*0.0492*	0.0793
oes10	0.4206	*0.4087*	0.4121	0.4383	0.4160
oes97	0.5262	0.5184	0.5092	*0.5071*	0.5140
osales	0.7514	0.7220	0.7043	0.7101	*0.6883*
polymer	0.7059	0.6749	*0.5594*	0.5930	0.6393
puma32H	0.8757	*0.8675*	0.9382	0.9151	0.9482
puma8NH	0.8292	0.8080	0.7925	0.7854	*0.7719*
rf1	0.0973	0.1078	0.0772	*0.0591*	0.0768
rf2	0.1062	0.1115	0.0836	0.0904	*0.0710*
scm1d	0.3336	0.3141	*0.2908*	0.3016	0.3108
scm20d	0.4268	0.3847	*0.3662*	0.3799	0.4146
scpf	0.8299	0.8398	*0.8051*	0.9126	0.8566
sf1	*0.9975*	1.1773	1.5609	1.1758	1.5243
sf2	*0.9798*	1.3365	1.6409	1.3159	1.6774
slump	0.7216	0.6844	0.6845	*0.6108*	0.6538
stock	0.1837	0.1670	0.1458	*0.1428*	0.1482
wisconsin	0.9307	0.9569	0.9465	0.9340	*0.9258*
wq	0.9141	0.9148	0.9076	*0.9028*	0.9083
Mean	0.6046	0.6149	0.6322	*0.5445*	0.5809
Average rank	3.7500	3.5357	2.9643	*1.9643*	2.7857

The best result for each dataset is highlighted in italics

size for RotF. In this case, the number of regressors for each target was only 100×1.

For MRQ and eMRQ, the settings were taken from [63]. The ensemble size was 100. For MRQ, the number of centroids was 50. For eMRQ, the total number of quantizers was the number of targets multiplied by 3. In each quantizer the number of centroids was selected randomly in [50, 100] and the number of targets in each subquantizer was selected randomly in [1, 2]. The ensemble sizes for MRQ and eMRQ are reduced to 1 when used as based methods for RotF and RotRanF, because for these methods the ensemble size is 100.

Average ranks [22] were used to compare several methods on several data sets. For each data set the methods are sorted from best to worst and assigned a ranking value between 1 and the total number of methods. In the case of ties, the methods are assigned an average value. For instance, if four methods share the best result, they are all assigned a rank of 2.5. The average rank of each method is the average of the ranks from all the data sets.

For comparing multiple classifiers over multiple data sets, we followed [33][5], based on [10, 22, 25]. The Friedman test was used to reject the null hypothesis. A pairwise post-hoc analysis was performed with the Wilcoxon signed-rank test with Holm's alpha correction.

The methods were also compared using the Bayesian Signed-Rank Test [11], the Bayesian equivalent of the Wilcoxon signed-rank test. In that test, the value of the *Region of Practical Equivalence* (ROPE) was set to 0.01 for aRRMSE. Two methods were considered equivalent when the difference in their performance was smaller than this "ROPE". The test yielded three probabilities: (1) one method is better than the other, (2) *vice-versa*, or (3) they are in the "ROPE".

[5] Code available at https://github.com/hfawaz/cd-diagram.

Table 4 Results for methods with SST-train

	SST-train-BagP	SST-train-BagU	SST-train-RanF	RotF-SST-train	RotRanF-SST-train
andro	0.5400	0.4970	0.4951	*0.4944*	0.5797
atp1d	0.3716	*0.3635*	0.3981	0.3651	0.4094
atp7d	0.5143	0.5019	0.5236	*0.4746*	0.5685
cal-housing	0.6939	0.6803	0.6514	*0.5758*	0.5990
edm	0.7430	0.7182	0.7004	*0.7003*	0.7045
enb	0.1229	0.1307	0.1216	0.1133	*0.1105*
fri-c0-500-25	*0.9249*	0.9278	0.9376	0.9365	0.9500
jura	0.5918	0.5896	0.5766	*0.5682*	0.5721
m5spec	0.5931	0.6395	0.6516	*0.0510*	0.0734
mp5spec	0.5645	0.5968	0.6134	*0.0486*	0.0780
mp6spec	0.5687	0.6036	0.6186	*0.0442*	0.0782
oes10	0.4201	*0.4079*	0.4097	0.4390	0.4175
oes97	0.5259	0.5179	0.5102	*0.5053*	0.5177
osales	0.7093	0.7081	0.6929	*0.6633*	0.6899
polymer	0.5701	0.5340	*0.4794*	0.5193	0.6454
puma32H	0.8719	*0.8650*	0.9346	0.9121	0.9468
puma8NH	0.8111	0.7836	0.7713	*0.7676*	0.7713
rf1	0.0775	0.0901	0.0737	*0.0536*	0.0792
rf2	0.0837	0.0903	0.0722	0.0835	*0.0695*
scm1d	0.3119	0.3016	*0.2859*	0.2902	0.3105
scm20d	0.3709	0.3541	*0.3400*	0.3560	0.4146
scpf	0.8553	0.8616	0.8073	*0.7972*	0.8407
sf1	*1.1270*	1.4447	1.5905	1.3628	1.5401
sf2	*0.9448*	1.0888	1.2130	1.4414	1.4458
slump	0.6664	0.7032	0.6907	*0.6125*	0.6554
stock	0.1771	0.1633	0.1447	*0.1398*	0.1468
wisconsin	*0.9287*	0.9537	0.9371	0.9354	0.9321
wq	0.9110	0.9139	0.9095	*0.9009*	0.9083
Mean	0.5926	0.6082	0.6125	*0.5411*	0.5734
Average rank	3.5357	3.5000	2.9643	*1.7857*	3.2143

The best result for each dataset is highlighted in italics

4.2 Results and discussion

Table 2 shows the results (aRRMSE) for the methods using the Single Target (ST) approach: with BagP, BagU, RanF, RotF, and RotRanF. The best result for each data set appears in italics. The table also shows the mean aRRMSE values across all the data sets and the average ranks.

The results for methods with Stacked Single Target (SST) are shown in Tables 3 (SST-true), 4 (SST-train) and 5 (SST-cv). For Ensembles of Regressor Chains (ERC), the results are shown in Tables 6 (ERC-true), 7 (ERC-train), and 8 (ERC-cv). The results for Multi-target Regression via Quantization are in Tables 9 (MRQ) and 10 (eMRQ).

In Tables 2, 3, 4,5,6,7,8,9 and 10, the method with RotF has by far the best mean aRRMSE, the top average rank, and is the best method for most data sets.

Table 11 shows the results for the nine methods that use RotF. None of the methods show a clear advantage. The top average rank is for RotF-ERC-train, while the best mean aRRMSE is for RotF-eMRQ. The best results for the data sets are fairly spread out across the methods, there are three methods that are the best for six data sets: RotF-ST, RotF-SST-train, and RotF-eMRQ.

4.2.1 aRRMSE differences

Figure 3 shows the boxplots of the differences in aRRMSE for the nine MTR approaches (ST, SST-true, SST-train, SST-cv, ERC-true, ERC-train, ERC-cv, MRQ, and eMRQ) with BagP, BagU , RanF and RotRanF against their corresponding RotF variant. The positive values indicate that the variant with RotF is better. Most of the differences represented in

Table 5 Results for methods with SST-cv

	SST-cv-BagP	SST-cv-BagU	SST-cv-RanF	RotF-SST-cv	RotRanF-SST-cv
andro	0.5793	0.5035	*0.4795*	0.4892	0.5966
atp1d	0.3717	*0.3669*	0.4005	0.3693	0.4133
atp7d	0.5074	0.4936	0.5278	*0.4779*	0.5650
cal-housing	0.6592	0.6328	0.6234	*0.5733*	0.5932
edm	0.7396	0.7271	0.7104	*0.7040*	0.7107
enb	0.1205	0.1218	0.1218	*0.1093*	0.1112
fri-c0-500-25	*0.9203*	0.9286	0.9375	0.9347	0.9540
jura	0.5906	0.5912	0.5938	*0.5713*	0.5716
m5spec	0.5638	0.5531	0.5789	*0.0550*	0.0728
mp5spec	0.5165	0.5086	0.5361	*0.0517*	0.0757
mp6spec	0.5129	0.5096	0.5362	*0.0473*	0.0762
oes10	0.4205	*0.4077*	0.4140	0.4389	0.4205
oes97	0.5243	0.5159	0.5060	*0.5051*	0.5203
osales	0.7260	0.7303	*0.6613*	0.7044	0.6884
polymer	0.6605	0.6357	0.6398	*0.5664*	0.6771
puma32H	*0.8775*	0.8776	0.9501	0.9202	0.9577
puma8NH	0.8325	0.8232	0.8194	*0.7942*	0.8074
rf1	0.0775	0.0886	0.0727	*0.0546*	0.0792
rf2	0.0836	0.0897	0.0757	0.0850	*0.0714*
scm1d	0.3077	0.2942	*0.2794*	0.2949	0.3095
scm20d	0.3539	0.3326	*0.3031*	0.3726	0.4165
scpf	0.8310	0.9568	0.9206	*0.8144*	0.8522
sf1	*1.0680*	1.2805	1.2634	1.2528	1.4567
sf2	*1.0553*	1.7773	1.6863	1.3263	1.5541
slump	0.6953	0.6840	0.6998	*0.6115*	0.6657
stock	0.1784	0.1624	0.1482	*0.1415*	0.1517
wisconsin	0.9333	0.9584	0.9474	0.9331	*0.9232*
wq	0.9095	0.9104	0.9076	*0.9023*	0.9085
Mean	0.5934	0.6236	0.6193	*0.5393*	0.5786
Average rank	3.3571	3.4286	3.1071	*1.8214*	3.2857

The best result for each dataset is highlighted in italics

the boxplots and, in all the cases, the average and median differences, are all positive. This remarks that the use of RotF is always improving the performance.

Figure 4 shows the boxplots of the differences in aRRMSE of the methods with RotF and RotF-ST. Positive values indicate that RotF-ST is a better option. The average difference is negative, with the only exceptions of RotF-SST-true and RotF-MRQ, but the median difference is only favorable (negative) for RotF-SST-train and RotF-ERC-train.

4.2.2 Ranks

Table 12 shows the average rankings of the 45 methods tested. The eight top positions are for methods with RotF. The two top methods are RotF-ERC-train and RotF-SST-train, while the third method is RotF-ST.

Figure 5 shows boxplots for the ranks, from the 45 methods under consideration. For each data set, every method is assigned a rank between 1 and 45. For each method its boxplot is from the ranks across all the data sets. Smaller values, i.e., at the right, indicate better ranks. The methods with better ranks include RotF.

Figure 6 also shows boxplots for the ranks, but grouped by family methods: the nine multi-target approaches and methods with RotF (Fig. 6b). The corresponding RotF variant shows better ranks from among the nine multi-target approaches. Among the RotF variants, the best ranks are for RotF-ERC-train and RotF-SST-train, while the third method is RotF-ST.

Figure 7 shows the average ranks and critical difference diagrams [22] for the Wilcoxon signed-rank test with Holm's correction [10, 33]. A thick horizontal line shows a group of classifiers that are not significantly different.

In some cases, it is possible that the horizontal line includes a pair of classifiers that are significantly different, because the test is performed for pairs of classifiers without

Table 6 Results for methods with ERC-true

	ERC-true-BagP	ERC-true-BagU	ERC-true-RanF	RotF-ERC-true	RotRanF-ERC-true
andro	0.5961	0.5257	0.5119	*0.5012*	0.5844
atp1d	0.3710	*0.3675*	0.3990	0.3692	0.4069
atp7d	0.5343	0.5035	0.5281	*0.4870*	0.5684
cal-housing	0.6682	0.6252	0.6442	*0.5857*	0.5860
edm	0.7435	0.7272	0.7205	*0.7010*	0.7168
enb	0.1253	0.1258	0.1147	0.1151	*0.1080*
fri-c0-500-25	*0.9206*	0.9267	0.9306	0.9384	0.9516
jura	0.5906	0.5871	0.5731	*0.5699*	0.5722
m5spec	0.5522	0.5797	0.6206	*0.0528*	0.0707
mp5spec	0.5145	0.5375	0.5826	*0.0523*	0.0770
mp6spec	0.5203	0.5433	0.5842	*0.0491*	0.0763
oes10	0.4202	0.4084	*0.4057*	0.4375	0.4186
oes97	0.5254	0.5168	*0.5058*	0.5094	0.5211
osales	0.7280	0.6901	0.6996	0.6934	*0.6846*
polymer	0.6568	0.6022	*0.5090*	0.5763	0.6292
puma32H	0.8729	*0.8637*	0.9339	0.9139	0.9493
puma8NH	0.8204	0.7964	0.7802	0.7765	*0.7752*
rf1	0.0843	0.0918	0.0698	*0.0564*	0.0890
rf2	0.0935	0.0954	0.0767	0.0865	*0.0756*
scm1d	0.3257	0.3049	*0.2882*	0.2978	0.3065
scm20d	0.4272	0.3784	*0.3560*	0.3989	0.4178
scpf	0.8120	0.8312	*0.8022*	0.8235	0.8440
sf1	*1.0501*	1.2819	1.5511	1.2286	1.5391
sf2	*1.0532*	1.3690	1.4806	1.3535	1.5746
slump	0.7006	0.6704	0.6699	*0.6182*	0.6530
stock	0.1779	0.1602	0.1442	*0.1409*	0.1463
wisconsin	0.9310	0.9573	0.9427	0.9316	*0.9254*
wq	0.9097	0.9098	0.9025	*0.9013*	0.9072
Mean	0.5973	0.6063	0.6188	*0.5416*	0.5777
Average rank	3.7143	3.3571	2.8571	*2.0000*	3.0714

The best result for each dataset is highlighted in italics

taking into account the rest of classifiers and the average ranks. For instance, for SST-true, RotF-SST-true (the method with best rank) is not significantly different from SST-true-BagP (worst rank) but it is significantly different from RotRanF-SST-true (second best rank).

RotF-ST, RotF-MRQ, and RotF-eMRQ are significantly different from all the other methods in their group. The critical difference diagram for methods with RotF shows that many variants are not significantly different.

4.2.3 Bayesian tests

Figure 8 shows the posteriors for the Bayesian sign-rank tests. In these triangles [11, 37], the bottom-left and bottom-right regions correspond to the case where one method is better than the other or *vice-versa*. The top region represents the case where the "ROPE" is more probable. The corner triangles show the probability of each region. For each of the seven multi-target methods under consideration, four triangles are shown, comparing BagP, BagU, RanF, and RotRanF with RotF. The bottom right region of the triangles, where the point clouds are mainly concentrated, signal where the method with Rotation Forest is better. The numeric values in the triangles corners indicate the proportion of points in each region. As can be noted, all the values at the bottom right of the triangles are greater than 0.9. This remarks the superiority of the RotF according to Bayesian test.

Figure 9 shows the posteriors for the Bayesian sign-rank tests, comparing RotF-ST with the rest of multi-target methods also using RotF. In this case, the points clouds are mostly in the "ROPE" region, with the exceptions of MRQ

Table 7 Results for methods with ERC-train

	ERC-train-BagP	ERC-train-BagU	ERC-train-RanF	RotF-ERC-train	RotRanF-ERC-train
andro	0.5384	*0.4898*	0.4998	0.4923	0.5783
atp1d	0.3675	*0.3578*	0.3983	0.3669	0.4107
atp7d	0.5094	0.4859	0.5264	*0.4789*	0.5664
cal-housing	0.6274	0.6008	0.6326	*0.5692*	0.5859
edm	0.7418	0.7223	0.7192	*0.6984*	0.7168
enb	0.1169	0.1180	0.1128	0.1106	*0.1073*
fri-c0-500-25	*0.9218*	0.9258	0.9305	0.9374	0.9501
jura	0.5899	0.5870	0.5731	*0.5692*	0.5711
m5spec	0.5815	0.6222	0.6324	*0.0508*	0.0707
mp5spec	0.5605	0.5802	0.5948	*0.0488*	0.0785
mp6spec	0.5636	0.5945	0.5992	*0.0464*	0.0742
oes10	0.4198	0.4080	*0.4062*	0.4377	0.4189
oes97	0.5250	0.5163	*0.5062*	0.5088	0.5198
osales	0.6985	0.6838	0.6881	*0.6566*	0.6867
polymer	0.5771	0.5266	*0.4778*	0.5144	0.6045
puma32H	0.8710	*0.8617*	0.9314	0.9129	0.9494
puma8NH	0.8108	0.7837	0.7677	*0.7668*	0.7725
rf1	0.0743	0.0852	0.0675	*0.0534*	0.0747
rf2	0.0820	0.0871	*0.0756*	0.0826	0.0796
scm1d	0.3043	0.2918	*0.2851*	0.2890	0.3067
scm20d	0.3549	0.3381	*0.3357*	0.3720	0.4171
scpf	0.8212	0.8628	*0.8073*	0.8142	0.8361
sf1	*1.1318*	1.4440	1.6009	1.3549	1.5425
sf2	*1.0869*	1.2284	1.2833	1.4144	1.5253
slump	0.6692	0.6744	0.6693	*0.6201*	0.6520
stock	0.1738	0.1579	0.1432	*0.1388*	0.1464
wisconsin	0.9298	0.9555	0.9499	0.9313	*0.9237*
wq	0.9053	0.9041	0.9014	*0.8991*	0.9072
Mean	0.5912	0.6033	0.6113	*0.5406*	0.5740
Average rank	3.3929	3.2500	2.9286	*2.0000*	3.4286

The best result for each dataset is highlighted in italics

and eMRQ. Therefore none of the multi-target methods show a clear advantage over ST when using RotF.

5 Conclusions

In this paper, the Rotation Forest ensemble method has been adapted and tested for multi-target regression. Rotation Forest (RotF) has been compared with Bagging and Random Forest in several MTR approaches, based on reducing the problem to several STR problems: single target (ST), stacked single target (SST), Ensembles of Regressor Chains (ERC), and Multi-target Regression via Quantization (MRQ and eMRQ). Both SST and ERC with three variants: true, train, and cv (cross validation), depending on which values are used when other outputs are used as inputs. Rotation Forest showed a clear advantage for the nine resulting multi-target approaches.

When comparing the nine multi-target approaches using Rotation Forest, the best methods were RotF-SST-train and RotF-ERC-train. Fortunately, the most expensive methods with internal cross validation hardly appeared necessary.

In contrast, none of the methods showed a clear advantage[6] over the most straightforward approach: RotF-ST. As it was noted, this method treats each output independently. Hence, it raises the question of whether the results

[6] There is neither advantage according to the average ranks nor Bayesian tests from the results of all the data sets. But for particular data sets the results were improved with other RotF methods.

Table 8 Results for methods with ERC-cv

	ERC-cv-BagP	ERC-cv-BagU	ERC-cv-RanF	RotF-ERC-cv	RotRanF-ERC-cv
andro	0.5668	0.4931	*0.4838*	0.4862	0.5865
atp1d	0.3724	*0.3668*	0.3999	0.3702	0.4096
atp7d	0.5124	0.4893	0.5237	*0.4857*	0.5726
cal-housing	0.6133	0.5845	0.6124	*0.5729*	0.5855
edm	0.7407	0.7280	0.7185	*0.6976*	0.7167
enb	0.1136	0.1111	0.1124	0.1075	*0.1063*
fri-c0-500-25	*0.9199*	0.9254	0.9307	0.9407	0.9514
jura	0.5896	0.5889	0.5780	*0.5689*	0.5736
m5spec	0.5527	0.5483	0.5922	*0.0535*	0.0713
mp5spec	0.5156	0.5060	0.5516	*0.0525*	0.0765
mp6spec	0.5101	0.5072	0.5541	*0.0502*	0.0761
oes10	0.4199	0.4074	*0.4059*	0.4375	0.4161
oes97	0.5239	0.5153	*0.5063*	0.5082	0.5221
osales	0.7131	0.7005	*0.6673*	0.7114	0.6846
polymer	0.6348	0.6044	0.5799	*0.5488*	0.6499
puma32H	0.8738	*0.8659*	0.9387	0.9159	0.9546
puma8NH	0.8209	0.8013	0.7890	*0.7799*	0.7897
rf1	0.0745	0.0844	0.0692	*0.0542*	0.0826
rf2	0.0814	0.0870	0.0762	0.0833	*0.0741*
scm1d	0.3008	0.2849	*0.2810*	0.2913	0.3054
scm20d	0.3377	0.3155	*0.3063*	0.3917	0.4130
scpf	0.8299	0.9310	0.8536	*0.8290*	0.8503
sf1	*1.0887*	1.3001	1.3977	1.2729	1.5100
sf2	*1.0879*	1.3811	1.4931	1.3265	1.5076
slump	0.6888	0.6749	0.6786	*0.6235*	0.6593
stock	0.1743	0.1577	0.1431	*0.1404*	0.1476
wisconsin	0.9321	0.9578	0.9522	0.9329	*0.9203*
wq	0.9059	0.9038	*0.9004*	0.9005	0.9054
Mean	0.5891	0.6008	0.6106	*0.5405*	0.5757
Average rank	3.6429	3.0714	2.8929	*2.0357*	3.3571

The best result for each dataset is highlighted in italics

of RotF-ST can be improved using other MTR approaches, considering output dependencies for building the models that are combined in Rotation Forest. Moreover, Rotation Forest can also be used with other STR methods instead of regression trees.

In the proposed method, rotation matrices are obtained using PCA, an unsupervised method. The use of supervised projection methods for MTR, instead of PCA, could improve the performance of this method and represents an area for future research.

Table 9 Results for methods with MRQ

	MRQ-BagP	MRQ-BagU	MRQ-RanF	RotF-MRQ	RotRanF-MRQ
andro	0.9541	0.7702	*0.5442*	0.7519	0.6608
atp1d	0.4742	0.4792	0.4977	*0.4060*	0.4326
atp7d	0.5830	*0.4541*	0.5247	0.5202	0.5927
cal-housing	0.8333	0.8380	0.8398	*0.5943*	0.6203
edm	0.8895	0.8075	0.8177	*0.7144*	0.7219
enb	0.1891	0.1664	0.1519	0.1240	*0.1174*
fri-c0-500-25	1.2028	1.2031	1.2289	*0.9600*	0.9748
jura	0.7623	0.7282	0.7683	*0.5878*	0.6100
m5spec	0.7290	0.7414	0.7298	*0.0660*	0.0813
mp5spec	0.6626	0.6717	0.6829	*0.0611*	0.0816
mp6spec	0.6691	0.6868	0.6971	*0.0573*	0.0779
oes10	0.5080	0.5228	0.4834	*0.4351*	0.4429
oes97	0.6423	0.6196	0.5947	*0.5316*	0.5416
osales	0.7617	0.7603	0.8614	*0.6685*	0.7324
polymer	0.9582	0.8444	0.8764	0.7211	*0.6547*
puma32H	1.1513	1.1537	1.2022	*0.9521*	0.9629
puma8NH	1.0096	1.0233	1.0274	0.7890	*0.7846*
rf1	0.2014	0.2007	0.2003	*0.1693*	0.1961
rf2	0.2014	0.2008	0.2002	*0.1709*	0.1851
scm1d	0.4507	0.4408	0.4349	*0.3905*	0.4061
scm20d	0.4777	0.4576	0.4548	*0.4274*	0.4580
scpf	0.9565	1.2044	0.9920	*0.7923*	0.8155
sf1	*0.8251*	1.0590	1.6798	0.8873	1.4494
sf2	*0.8217*	0.8373	0.9071	0.9677	1.3620
slump	0.9267	0.9301	0.8872	*0.6770*	0.6968
stock	0.2132	0.2056	0.1980	*0.1611*	0.1699
wisconsin	1.2824	1.2974	1.3553	*0.9282*	0.9440
wq	1.0659	1.0837	1.0744	*0.9117*	0.9189
Mean	0.7287	0.7282	0.7469	*0.5508*	0.5961
Average rank	3.7500	3.7857	3.8571	*1.3571*	2.2500

The best result for each dataset is highlighted in italics

Table 10 Results for methods with eMRQ

	eMRQ-BagP	eMRQ-BagU	eMRQ-RanF	RotF-eMRQ	RotRanF-eMRQ
andro	0.7856	0.6582	*0.5748*	0.7117	0.6633
atp1d	0.3850	*0.3778*	0.4589	0.3883	0.4276
atp7d	0.5112	*0.3933*	0.4815	0.5162	0.5771
cal-housing	0.7151	0.7023	0.7164	*0.5869*	0.6217
edm	0.8195	0.7694	0.7713	*0.7120*	0.7162
enb	0.1693	0.1537	0.1367	0.1202	*0.1155*
fri-c0-500-25	1.0614	1.0562	1.0632	*0.9455*	0.9613
jura	0.6592	0.6461	0.6439	*0.5719*	0.5991
m5spec	0.6901	0.6938	0.6983	*0.0623*	0.0733
mp5spec	0.6315	0.6217	0.6454	*0.0594*	0.0761
mp6spec	0.6384	0.6543	0.6708	*0.0538*	0.0748
oes10	0.4859	0.4668	0.4589	*0.4144*	0.4293
oes97	0.6077	0.6008	0.6022	*0.5172*	0.5379
osales	0.8022	0.7461	0.8095	*0.6708*	0.7046
polymer	0.7140	*0.6132*	0.7488	0.6534	0.6187
puma32H	0.9775	0.9697	1.0210	*0.9373*	0.9494
puma8NH	0.8732	0.8818	0.8948	0.7792	*0.7746*
rf1	0.0606	*0.0582*	0.0743	0.0634	0.1018
rf2	0.0601	0.0582	0.0577	*0.0551*	0.0805
scm1d	0.3103	0.2932	*0.2834*	0.2923	0.3155
scm20d	0.3289	0.3062	*0.3028*	0.3534	0.3915
scpf	0.8411	0.9862	0.9290	*0.7642*	0.8218
sf1	*0.8251*	0.9357	1.6814	0.8943	1.4568
sf2	*0.8186*	0.8477	0.8772	0.8326	1.3811
slump	0.8562	0.8643	0.8260	0.7283	*0.7100*
stock	0.1591	0.1475	*0.1358*	0.1403	0.1481
wisconsin	1.0857	1.1069	1.0932	*0.9246*	0.9363
wq	1.0440	1.0121	1.0193	*0.8907*	0.8972
Mean	0.6399	0.6293	0.6670	*0.5228*	0.5772
Average rank	3.6786	3.0714	3.6786	*1.7857*	2.7857

The best result for each dataset is highlighted in italics

Table 11 Results for methods with RotF

	RotF-ST	RotF-SST-true	RotF-SST-train	RotF-SST-cv	RotF-ERC-true	RotF-ERC-train	RotF-ERC-cv	RotF-MRQ	RotF-eMRQ
andro	0.4906	0.5029	0.4944	0.4892	0.5012	0.4923	*0.4862*	0.7519	0.7117
atp1d	0.3694	0.3684	*0.3651*	0.3693	0.3692	0.3669	0.3702	0.4060	0.3883
atp7d	0.4807	0.4964	*0.4746*	0.4779	0.4870	0.4789	0.4857	0.5202	0.5162
cal-housing	0.5708	0.6105	0.5758	0.5733	0.5857	*0.5692*	0.5729	0.5943	0.5869
edm	0.7041	0.7044	0.7003	0.7040	0.7010	0.6984	*0.6976*	0.7144	0.7120
enb	0.1080	0.1259	0.1133	0.1093	0.1151	0.1106	*0.1075*	0.1240	0.1202
fri-c0-500-25	0.9387	0.9358	0.9365	*0.9347*	0.9384	0.9374	0.9407	0.9600	0.9455
jura	0.5698	0.5707	*0.5682*	0.5713	0.5699	0.5692	0.5689	0.5878	0.5719
m5spec	0.0525	0.0536	0.0510	0.0550	0.0528	*0.0508*	0.0535	0.0660	0.0623
mp5spec	0.0521	0.0531	*0.0486*	0.0517	0.0523	0.0488	0.0525	0.0611	0.0594
mp6spec	0.0478	0.0492	*0.0442*	0.0473	0.0491	0.0464	0.0502	0.0573	0.0538
oes10	0.4387	0.4383	0.4390	0.4389	0.4375	0.4377	0.4375	0.4351	*0.4144*
oes97	*0.5043*	0.5071	0.5053	0.5051	0.5094	0.5088	0.5082	0.5316	0.5172
osales	0.7278	0.7101	0.6633	0.7044	0.6934	*0.6566*	0.7114	0.6685	0.6708
polymer	0.5580	0.5930	0.5193	0.5664	0.5763	*0.5144*	0.5488	0.7211	0.6534
puma32H	0.9129	0.9151	*0.9121*	0.9202	0.9139	0.9129	0.9159	0.9521	0.9373
puma8NH	0.7706	0.7854	0.7676	0.7942	0.7765	*0.7668*	0.7799	0.7890	0.7792
rf1	*0.0533*	0.0591	0.0536	0.0546	0.0564	0.0534	0.0542	0.1693	0.0634
rf2	0.0841	0.0904	0.0835	0.0850	0.0865	0.0826	0.0833	0.1709	*0.0551*
scm1d	*0.2820*	0.3016	0.2902	0.2949	0.2978	0.2890	0.2913	0.3905	0.2923
scm20d	*0.3483*	0.3799	0.3560	0.3726	0.3989	0.3720	0.3917	0.4274	0.3534
scpf	0.8166	0.9126	0.7972	0.8144	0.8235	0.8142	0.8290	0.7923	*0.7642*
sf1	1.3705	1.1758	1.3628	1.2528	1.2286	1.3549	1.2729	*0.8873*	0.8943
sf2	1.4105	1.3159	1.4414	1.3263	1.3535	1.4144	1.3265	0.9677	*0.8326*
slump	*0.6098*	0.6108	0.6125	0.6115	0.6182	0.6201	0.6235	0.6770	0.7283
stock	*0.1387*	0.1428	0.1398	0.1415	0.1409	0.1388	0.1404	0.1611	0.1403
wisconsin	0.9349	0.9340	0.9354	0.9331	0.9316	0.9313	0.9329	0.9282	*0.9246*
wq	0.8988	0.9028	0.9009	0.9023	0.9013	0.8991	0.9005	0.9117	*0.8907*
Mean	0.5444	0.5445	0.5411	0.5393	0.5416	0.5406	0.5405	0.5508	*0.5228*
Average rank	4.0000	6.2143	3.6071	4.9286	5.4286	*3.1429*	4.7857	7.3571	5.5357

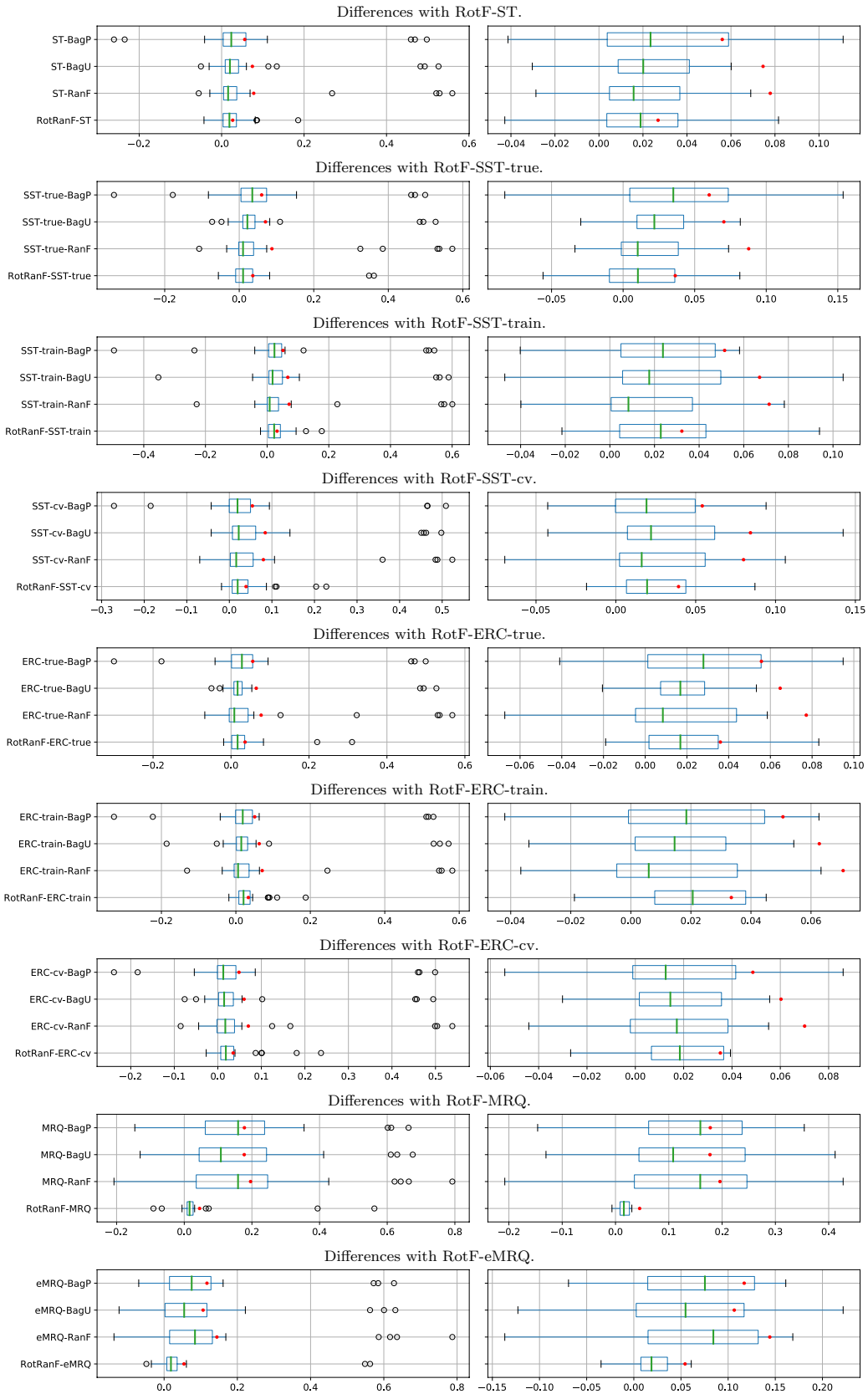

Fig. 3 Boxplots of the error differences. The boxplots on the right do not include the outliers. The average differences are marked with a red dot (•) (color figure online)

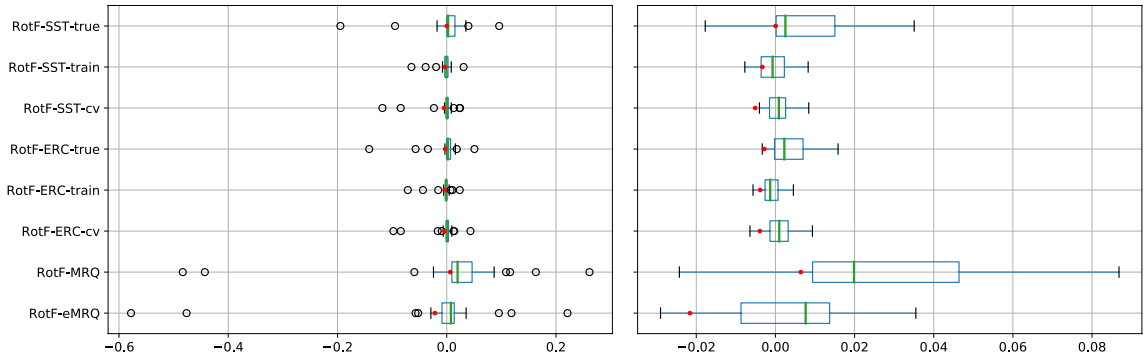

Fig. 4 Boxplots of the error differences between methods with RotF and RotF-ST. The boxplots on the right do not include the outliers. The average differences are marked with a red dot (•) (color figure online)

Table 12 Average ranks from all the considered methods

Method	Rank	Method	Rank	Method	Rank
RotF-ERC-train	9.6429	ERC-train-BagP	20.6786	SST-true-RanF	24.7143
RotF-SST-train	10.0000	RotRanF-ERC-train	20.8571	SST-cv-BagP	24.7500
RotF-ST	11.3571	ERC-cv-BagP	21.3214	SST-train-BagP	24.8214
RotF-ERC-cv	12.7500	RotRanF-SST-true	21.7857	ST-BagP	25.4643
RotF-SST-cv	12.9643	RotRanF-ERC-true	22.0357	RotRanF-eMRQ	26.3571
RotF-ERC-true	14.1071	RotRanF-SST-train	22.1071	ERC-true-BagP	26.9286
RotF-eMRQ	14.8214	RotRanF-ERC-cv	22.2143	SST-true-BagU	28.2857
RotF-SST-true	17.0357	SST-cv-RanF	22.2857	eMRQ-BagU	29.7500
ERC-train-RanF	17.5000	ST-RanF	22.5714	SST-true-BagP	29.8214
ERC-cv-RanF	18.3214	ERC-true-BagU	23.2500	eMRQ-RanF	31.9286
ERC-cv-BagU	18.3571	ST-BagU	23.8929	eMRQ-BagP	32.0714
ERC-train-BagU	19.3571	RotRanF-SST-cv	24.1429	RotRanF-MRQ	32.1429
ERC-true-RanF	19.6071	SST-train-BagU	24.2500	MRQ-BagU	39.6071
RotRanF-ST	19.8214	SST-cv-BagU	24.6786	MRQ-BagP	40.7500
SST-train-RanF	20.1071	RotF-MRQ	24.7143	MRQ-RanF	41.0714

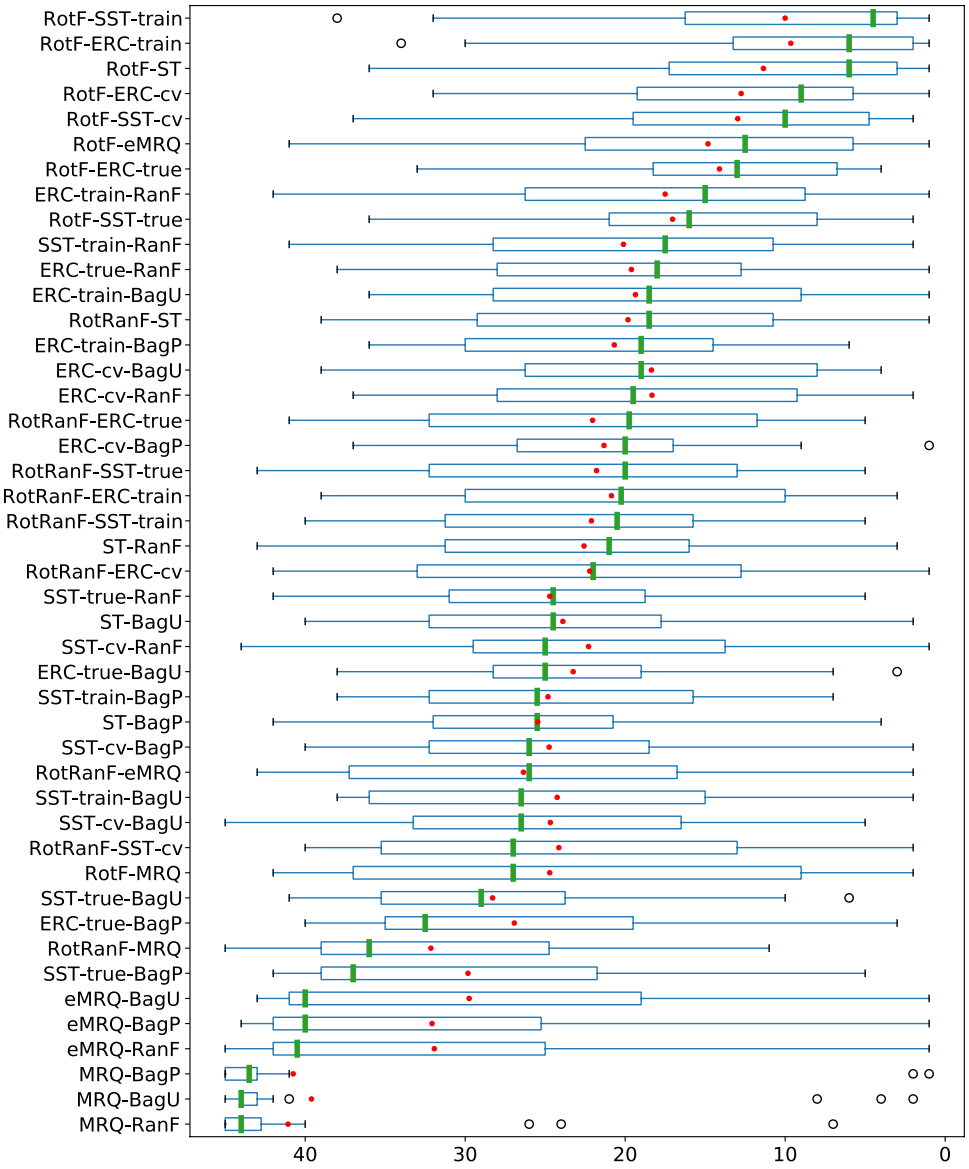

Fig. 5 Boxplots for the ranks, from all the considered methods. The methods are sorted by their median value. The average ranks are marked with a red dot (•) (color figure online)

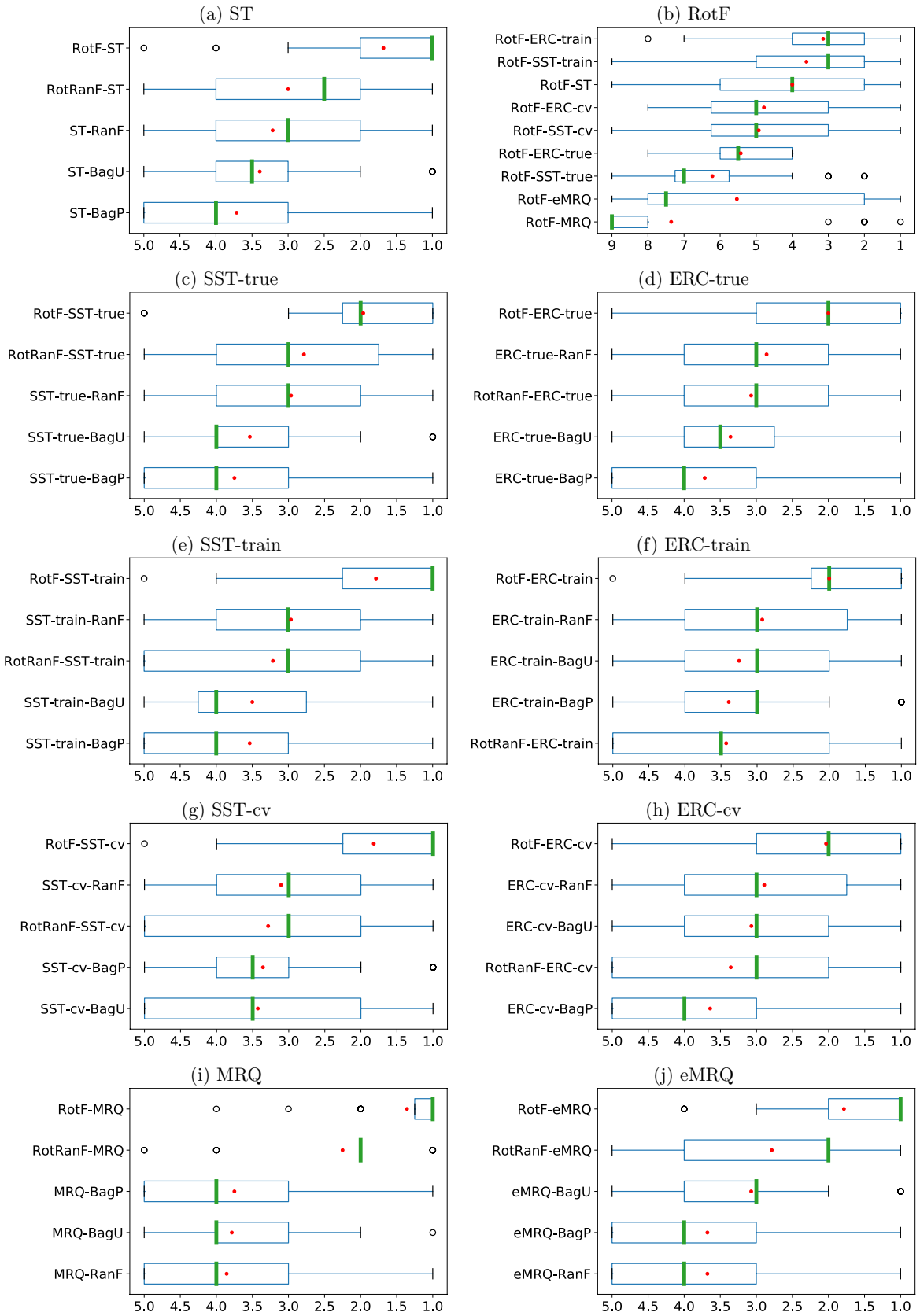

Fig. 6 Boxplots for the ranks, from different groups of methods. The average ranks are marked with a red dot (•) (color figure online)

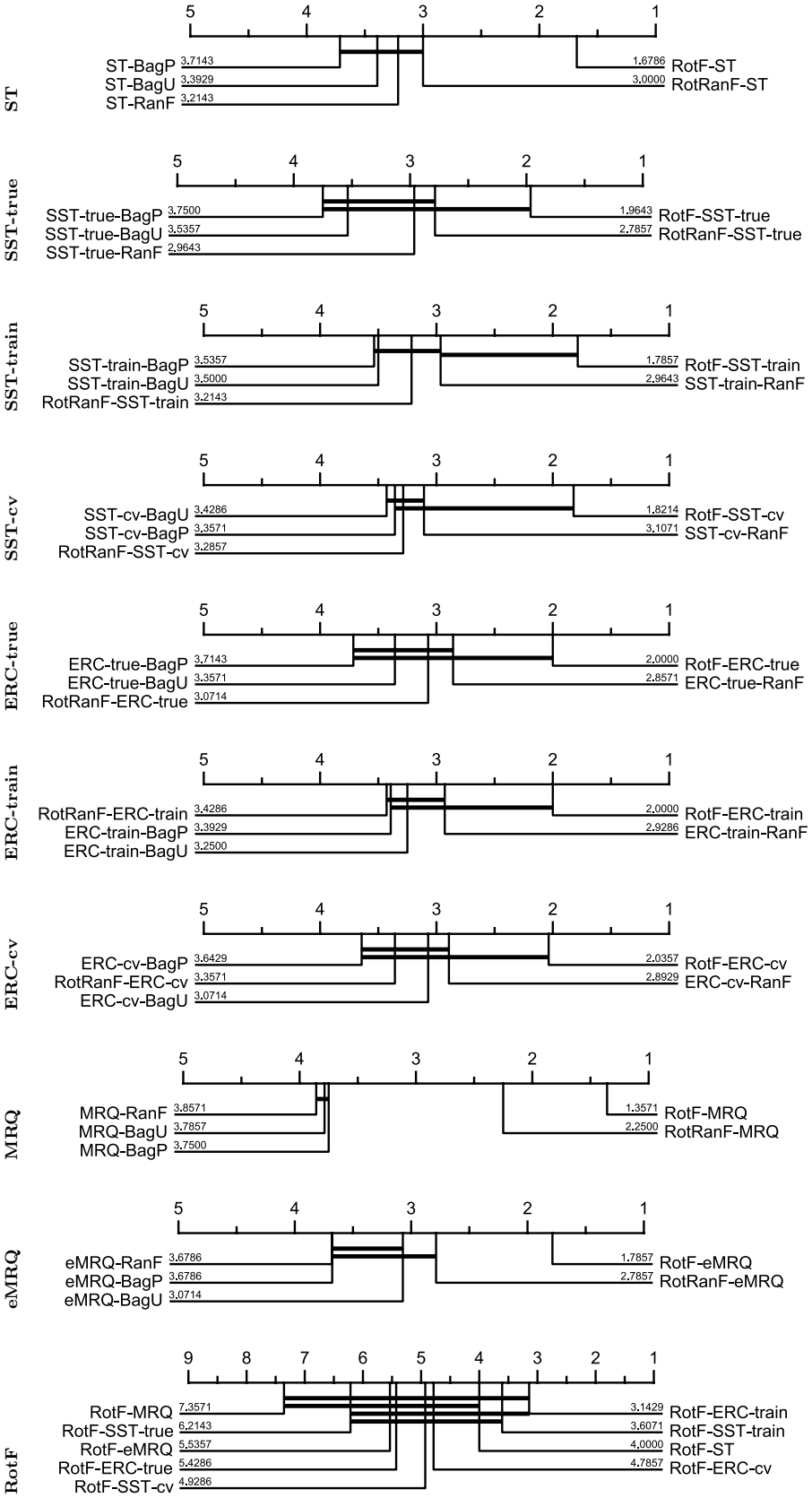

Fig. 7 Critical difference diagrams

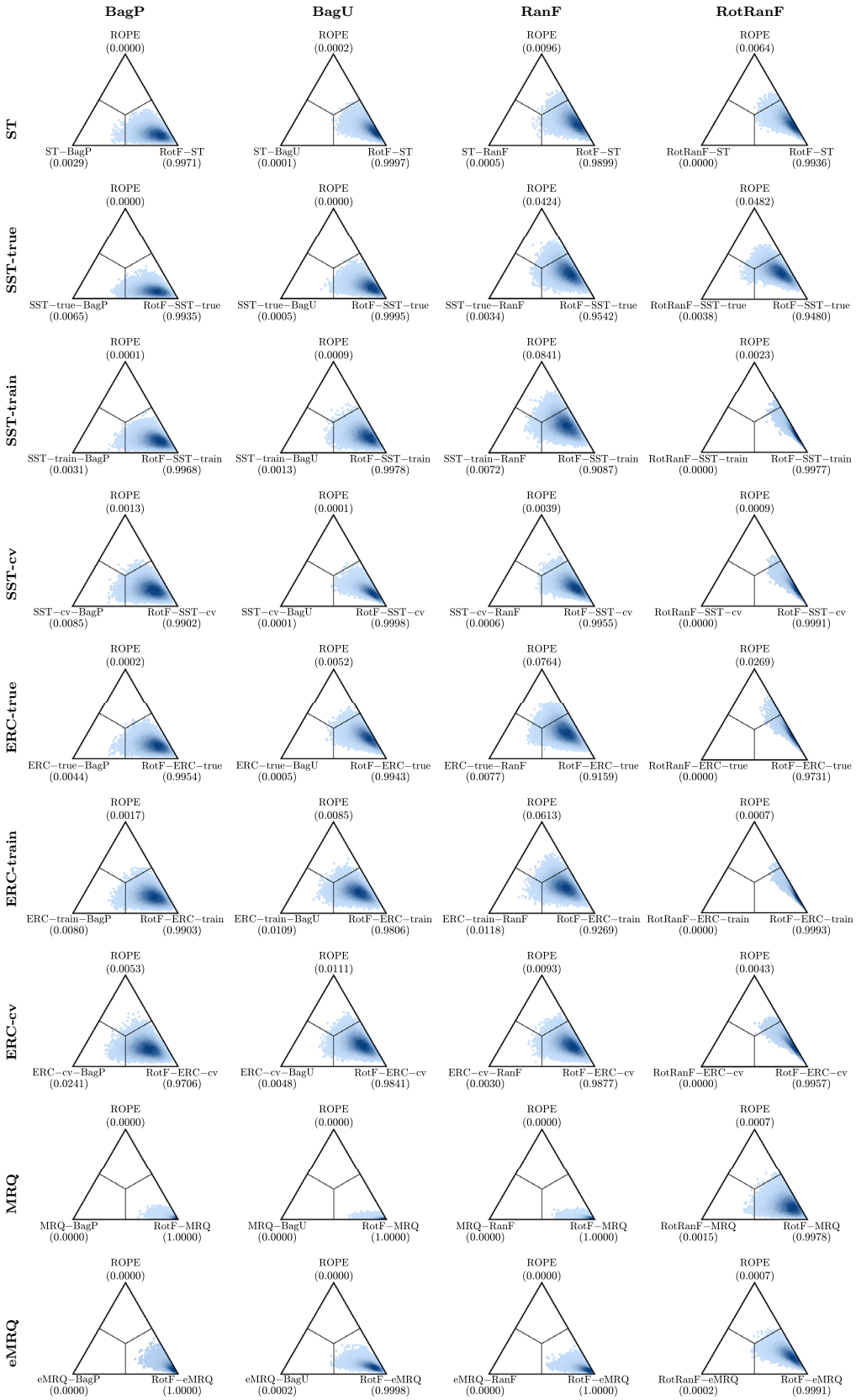

Fig. 8 Posteriors for the Bayesian sign-rank tests. Each row is for a multi-target method while the columns are for ensemble methods (BagP, BagU, RanF, and RotRanF). Each triangle compares the ensemble method of the column with RotF, for the multi-target method of the row

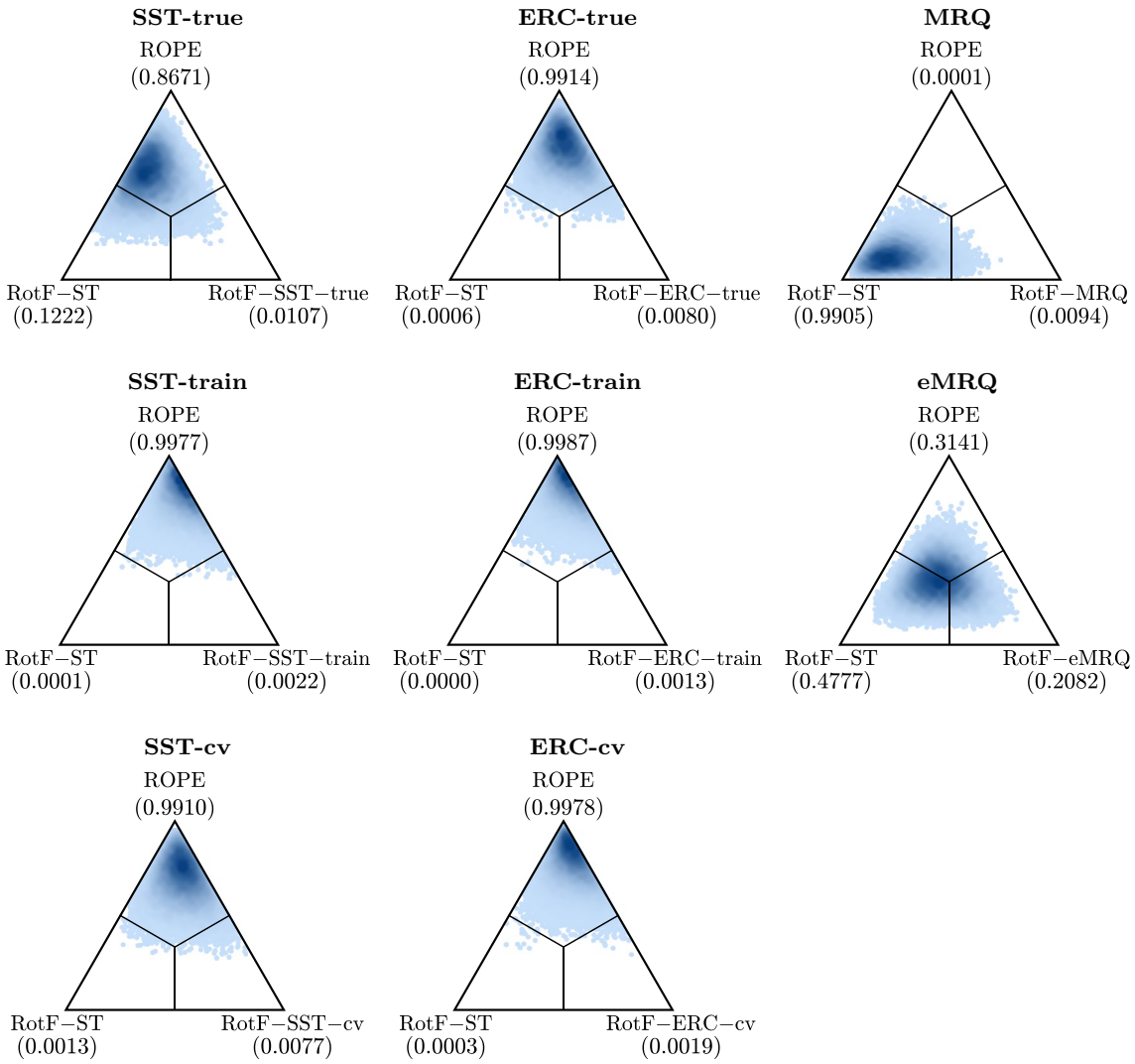

Fig. 9 Posteriors for the Bayesian sign-rank tests. Each triangle shows the comparison of RotF-ST with other method using RotF

Acknowledgements We thank Eleftherios Spyromitros-Xioufis and Esra Adıyeke for their help with the implementations [2, 63]. This work was supported by the *Ministerio de Economía y Competitividad* of the Spanish Government under project TIN2015-67534-P (MINECO-FEDER, UE), by the *Junta de Castilla y León* under project BU085P17 (JCyL/FEDER, UE) (both projects co-financed through European Union FEDER funds), and by the *Consejería de Educación of the Junta de Castilla y León* and the European Social Fund with the EDU/1100/2017 pre-doctoral grant. The authors gratefully acknowledge the support of the NVIDIA Corporation and its donation of the TITAN Xp GPUs used in this research.

References

1. Abraham Z, Tan PN, Winkler J, Zhong S, Liszewska M, et al (2013) Position preserving multi-output prediction. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, pp 320–335
2. Adıyeke E, Baydoğan MG (2020) The benefits of target relations: a comparison of multitask extensions and classifier chains. Pattern Recogn 107:107507
3. Aho T, Ženko B, Džeroski S, Elomaa T (2012) Multi-target regression with rule ensembles. J Mach Learn Res 13(Aug):2367–2407
4. Aho T, Ženko B, Džeroski S (2009) Rule ensembles for multi-target regression. In: 2009 Ninth IEEE International Conference on Data Mining, pp 21–30. IEEE
5. Alvarez MA, Rosasco L, Lawrence ND et al (2012) Kernels for vector-valued functions: a review. Found Trends Mach Learn 4(3):195–266
6. Appice A, Džeroski S (2007) Stepwise induction of multi-target model trees. In: European Conference on Machine Learning. Springer, pp 502–509
7. Argyriou A, Evgeniou T, Pontil M (2008) Convex multi-task feature learning. Mach Learn 73(3):243–272
8. Argyriou A, Evgeniou T, Pontil M (2006) Multi-task feature learning. In: Advances in neural information processing systems. pp 41–48
9. Ayerdi B, Graña M (2014) Hybrid extreme rotation forest. Neural Networks 52:33–42

10. Benavoli A, Corani G, Mangili F (2016) Should we really use post-hoc tests based on mean-ranks? J Mach Learn Res 17(1):152–161
11. Benavoli A, Corani G, Demšar J, Zaffalon M (2017) Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. J Mach Learn Res 18(77): 1–36. http://jmlr.org/papers/v18/16-305.html
12. Blaser R, Fryzlewicz P (2016) Random rotation ensembles. J Mach Learn Res 17(1):126–151
13. Borchani H, Varando G, Bielza C, Larrañaga P (2015) A survey on multi-output regression. Wiley Interdiscip Rev 5(5):216–233
14. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
15. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
16. Breiman L, Friedman JH (1997) Predicting multivariate responses in multiple linear regression. J Roy Stat Soc 59(1):3–54
17. Breskvar M, Kocev D, Džeroski S (2018) Ensembles for multi-target regression with random output selections. Mach Learn 107(11):1673–1709
18. Cai Z, Zhu W (2018) Multi-label feature selection via feature manifold learning and sparsity regularization. Int J Mach Learn Cybernet 9(8):1321–1334. https://doi.org/10.1007/s13042-017-0647-y
19. Caruana R (1994) Learning many related tasks at the same time with backpropagation. In: Advances in neural information processing systems. pp 657–664
20. Collobert R, Weston J (2008) A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning. pp 160–167. ACM
21. De'Ath G (2002) Multivariate regression trees: a new technique for modeling species-environment relationships. Ecology 83(4):1105–1117
22. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(Jan):1–30
23. Dua D, Graff C (2019) UCI machine learning repository. http://archive.ics.uci.edu/ml
24. Džeroski S, Demšar D, Grbović J (2000) Predicting chemical parameters of river water quality from bioindicator data. Appl Intell 13(1):7–17
25. Garcia S, Herrera F (2008) An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. J Mach Learn Res 9(12): 2677–2694
26. García-Pedrajas N, Maudes-Raedo J, García-Osorio C, Rodríguez-Díez JJ (2012) Supervised subspace projections for constructing ensembles of classifiers. Inf Sci 193:1–21
27. Ghosn J, Bengio Y (1997) Multi-task learning for stock selection. Adv Neural Inf Process Syst pp. 946–952
28. Godbole S, Sarawagi S (2004) Discriminative methods for multi-labeled classification. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, pp 22–30
29. Goovaerts P et al (1997) Geostatistics for natural resources evaluation. Oxford University Press on Demand
30. Hatzikos EV, Tsoumakas G, Tzanis G, Bassiliades N, Vlahavas I (2008) An empirical study on sea water quality prediction. Knowl-Based Syst 21(6):471–478
31. Herrera F, Charte F, Rivera AJ, Del Jesus MJ (2016) Multilabel classification. In: Multilabel classification. Springer, pp 17–31
32. Ho TK (1998) The random subspace method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell 20(8):832–844
33. Ismail Fawaz H, Forestier G, Weber J, Idoumghar L, Muller PA (2019) Deep learning for time series classification: a review. Data Min Knowl Disc 33(4):917–963
34. Izenman AJ (1975) Reduced-rank regression for the multivariate linear model. J Multivar Anal 5(2):248–264
35. Jalali A, Sanghavi S, Ruan C, Ravikumar PK (2010) A dirty model for multi-task learning. Adv Neural Inf Process Syste 23:964–972
36. Jeong JY, Kang JS, Jun CH (2020) Regularization-based model tree for multi-output regression. Inf Sci 507:240–255
37. Juez-Gil M (2020) mjuez/baycomp_plotting. https://doi.org/10.5281/zenodo.4244542
38. Kaggle (2012) Kaggle competition: online product sales. https://www.kaggle.com/c/online-sales
39. Kaggle (2013) Kaggle competition: see click predict fix. https://www.kaggle.com/c/see-click-predict-fix
40. Karalič A, Bratko I (1997) First order regression. Mach Learn 26(2–3):147–176
41. Kocev D, Vens C, Struyf J, Džeroski S (2013) Tree ensembles for predicting structured outputs. Pattern Recogn 46(3):817–833
42. Kocev D, Vens C, Struyf J, Džeroski S (2007) Ensembles of multi-objective decision trees. In: European conference on machine learning. Springer, pp 624–631
43. Kordos M, Arnaiz-González Á, García-Osorio C (2019) Evolutionary prototype selection for multi-output regression. Neurocomputing 358:309–320
44. Kuncheva LI (2014) Combining pattern classifiers: methods and algorithms, 2nd edn. Wiley
45. Latorre Carmona P, Sotoca JM, Pla F (2012) Filter-type variable selection based on information measures for regression tasks. Entropy 14(2):323–343
46. Li H, Zhang W, Chen Y, Guo Y, Li GZ, Zhu X (2017) A novel multi-target regression framework for time-series prediction of drug efficacy. Sci Rep 7:40652
47. Mastelini SM, da Costa VGT, Santana EJ, Nakano FK, Guido RC, Cerri R, Barbon S (2019) Multi-output tree chaining: an interpretative modelling and lightweight multi-target approach. J Signal Process Syst 91(2):191–215
48. Melki G, Cano A, Kecman V, Ventura S (2017) Multi-target support vector regression via correlation regressor chains. Inf Sci 415:53–69
49. Mitrović T, Antanasijević D, Lazović S, Perić-Grujić A, Ristić M (2019) Virtual water quality monitoring at inactive monitoring sites using monte carlo optimized artificial neural networks: a case study of danube river (serbia). Sci Total Environ 654:1000–1009
50. Nunes M, Gerding E, McGroarty F, Niranjan M (2019) A comparison of multitask and single task learning with artificial neural networks for yield curve forecasting. Expert Syst Appl 119:362–375
51. Obozinski G, Taskar B, Jordan MI (2010) Joint covariate selection and joint subspace selection for multiple classification problems. Stat Comput 20(2):231–252
52. Pardo C, Diez-Pastor JF, García-Osorio C, Rodríguez JJ (2013) Rotation forests for regression. Appl Math Comput 219(19):9914–9924
53. Petković M, Kocev D, Džeroski S (2020) Feature ranking for multi-target regression. Mach Learn 109(6):1179–1204
54. Pham BT, Bui DT, Prakash I, Dholakia M (2016) Rotation forest fuzzy rule-based classifier ensemble for spatial prediction of landslides using gis. Nat Hazards 83(1):97–127
55. Polikar R (2006) Ensemble based systems in decision making. IEEE Circuits Syst Mag 6(3):21–45. https://doi.org/10.1109/MCAS.2006.1688199
56. Read J, Pfahringer B, Holmes G, Frank E (2011) Classifier chains for multi-label classification. Mach Learn 85(3):333
57. Reyes O, Fardoun HM, Ventura S (2018) An ensemble-based method for the selection of instances in the multi-target regression problem. Integr Comput-Aided Eng 25(4):305–320
58. Rodríguez JJ, Kuncheva LI, Alonso CJ (2006) Rotation forest: a new classifier ensemble method. IEEE Trans Pattern Anal Mach Intell 28(10):1619–1630. https://doi.org/10.1109/TPAMI.2006.211. http://doi.ieeecomputersociety.org/10.1109/TPAMI.2006.211
59. Sánchez-Fernández M, de Prado-Cumplido M, Arenas-García J, Pérez-Cruz F (2004) SVM multiregression for nonlinear channel

estimation in multiple-input multiple-output systems. IEEE Trans Signal Process 52(8):2298–2307
60. Santana EJ, Geronimo BC, Mastelini SM, Carvalho RH, Barbin DF, Ida EI, Barbon S Jr (2018) Predicting poultry meat characteristics using an enhanced multi-target regression method. Biosyst Eng 171:193–204
61. Shim J, Kang S, Cho S (2020) Kernel rotation forests for classification. In: 2020 IEEE International Conference on Big Data and Smart Computing (BigComp). pp 406–409. IEEE
62. Spyromitros-Xioufis E, Tsoumakas G, Groves W, Vlahavas I (2016) Multi-target regression via input space expansion: treating targets as inputs. Mach Learn 104(1):55–98
63. Spyromitros-Xioufis E, Sechidis K, Vlahavas I (2020) Multi-target regression via output space quantization. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp 1–9. IEEE
64. Stiglic G, Rodriguez JJ, Kokol P (2011) Rotation of random forests for genomic and proteomic classification problems. In: Software tools and algorithms for biological systems. Springer, pp 211–221
65. Struyf J, Džeroski S (2005) Constraint based induction of multi-objective regression trees. In: International workshop on knowledge discovery in inductive databases. Springer, pp 222–233
66. Triguero I, Basgalupp M, Cerri R, Schietgat L, Vens C (2016) Partitioning the target space in multi-output learning. In: Proceedings of the 25th Belgian-Dutch Machine Learning Conference (Benelearn)
67. Tsanas A, Xifara A (2012) Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. Energy Build 49:560–567
68. Tsoumakas G, Spyromitros-Xioufis E, Vilcek J, Vlahavas I (2011) Mulan: a java library for multi-label learning. J Mach Learn Res 12:2411–2414
69. Tsoumakas G, Spyromitros-Xioufis E, Vrekou A, Vlahavas I (2014) Multi-target regression via random linear target combinations. In: Joint european conference on machine learning and knowledge discovery in databases. Springer, pp 225–240
70. Van Der Merwe A, Zidek J (1980) Multivariate regression analysis and canonical variates. Can J Stat 8(1):27–39
71. Vazquez E, Walter E (2003) Multi-output suppport vector regression. IFAC Proc Volumes 36(16):1783–1788
72. Wang L, You ZH, Xia SX, Chen X, Yan X, Zhou Y, Liu F (2018) An improved efficient rotation forest algorithm to predict the interactions among proteins. Soft Comput 22(10):3373–3381
73. Wang J, Chen Z, Sun K, Li H, Deng X (2019) Multi-target regression via target specific features. Knowl-Based Syst 170:70–78
74. Wilson DR, Martinez TR (2000) Reduction techniques for instance-based learning algorithms. Mach Learn 38(3):257–286
75. Wolpert DH (1992) Stacked generalization. Neural Networks 5(2):241–259
76. Xu S, An X, Qiao X, Zhu L, Li L (2013) Multi-output least-squares support vector regression machines. Pattern Recogn Lett 34(9):1078–1084
77. Yeh IC (2007) Modeling slump flow of concrete using second-order regressions and artificial neural networks. Cement Concr Compos 29(6):474–480
78. Zeng J, Liu Y, Leng B, Xiong Z, Cheung YM (2017) Dimensionality reduction in multiple ordinal regression. IEEE Trans Neural Networks Learn Syst 29(9):4088–4101
79. Zhang CX, Zhang JS (2008) Rotboost: a technique for combining rotation forest and adaboost. Pattern Recogn Lett 29(10):1524–1536
80. Zhang W, Liu X, Ding Y, Shi D (2012) Multi-output lS-SVR machine in extended feature space. In: 2012 IEEE International conference on computational intelligence for measurement systems and applications (CIMSA) proceedings. pp 130–134. IEEE
81. Zhen X, Yu M, He X, Li S (2017) Multi-target regression via robust low-rank learning. IEEE Trans Pattern Anal Mach Intell 40(2):497–504
82. Zhen X, Wang Z, Yu M, Li S (2015) Supervised descriptor learning for multi-output regression. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 1211–1218
83. Zhu X, Gao Z (2018) An efficient gradient-based model selection algorithm for multi-output least-squares support vector regression machines. Pattern Recogn Lett 111:16–22
84. Zolfagharnasab H, Bessa S, Oliveira S, Faria P, Teixeira J, Cardoso J, Oliveira H (2018) A regression model for predicting shape deformation after breast conserving surgery. Sensors 18(1):167