Towards meta-learning for multi-target regression problems

Gabriel J. Aguiar *, Everton J. Santana *, Saulo M. Mastelini †, Rafael G. Mantovani ‡, Sylvio Barbon Jr *
* Computer Science Department, Londrina State University, Londrina - PR, Brazil
† Institute of Mathematics and Computer Sciences, University of São Paulo (ICMC/USP), São Carlos - SP, Brazil
‡ Computer Engineering Department, Federal Technology University, Campus of Apucarana - PR, Brazil
E-mail: gjonas@uel.br, evertonsantana@uel.br, mastelini@usp.br, rafaelmantovani@utfpr.edu.br, barbon@uel.br

***Abstract*—Several multi-target regression methods were developed in the last years aiming at improving predictive performance by exploring inter-target correlation within the problem. However, none of these methods outperforms the others for all problems. This motivates the development of automatic approaches to recommend the most suitable multi-target regression method. In this paper, we propose a meta-learning system to recommend the best predictive method for a given multi-target regression problem. We performed experiments with a meta-dataset generated by a total of 648 synthetic datasets. These datasets were created to explore distinct inter-targets characteristics toward recommending the most promising method. In experiments, we evaluated four different algorithms with different biases as meta-learners. Our meta-dataset is composed of 58 meta-features, based on: statistical information, correlation characteristics, linear landmarking, from the distribution and smoothness of the data, and has four different meta-labels. Results showed that induced meta-models were able to recommend the best method for different base level datasets with a balanced accuracy superior to 70% using a Random Forest meta-model, which statistically outperformed the meta-learning baselines.**

***Index Terms*—Multi-target, Regression, Meta-learning**

I. Introduction

Machine Learning (ML) approaches have been providing significant advances in understanding and modeling problems from the broadest knowledge fields. A considerable part of the ML solutions takes advantage of supervised learning algorithms which explore the information, i.e., input and prediction target, from the problem data to learn a pattern. However, data from several real problems present more than one target. In this case, when a dataset presents multiple continuous targets, we call it a *multi-target regression problem*.

Currently, there are several methods in the literature addressing this type of problem. The most straightforward approach, referred to as Single-target (ST) regression, is to create a single model for each target disregarding the possible inter-target correlation. Multi-target Regression (MTR) is an alternative approach that, besides using the original input features, exploits the statistical correlation among the outputs. The MTR methods have been applied to solve many problems [1]–[5], leading to improvement in the predictive performance over ST methods. However, each method has specific characteristics and has been effective for different problems.

Selecting the most suitable algorithm for a given problem requires extensive experimental evaluation, which demands massive computational resources (particularly processing time) and specialists [6], [7]. On the other hand, a MTR method could be automatically selected when addressed as an output in an algorithm selection (or recommendation) problem by Meta-learning (MtL) [8].

The MtL core concept is to use the knowledge acquired from previous similar problems to recommend the most suitable algorithm, for a new unseen dataset. In the last years, MtL has been employed in different contexts, such as tasks to select [9], rank [10] and predict [11] the performance of ML algorithms and employing them on a new dataset.

Our hypothesis holds that MtL can be applied to MTR problems and recommend the most suitable method for new unseen problems. Thus, in this study, we propose a recommendation system able to predict the best MTR method for a new dataset. For such, experiments were carried out with meta-datasets generated with a total of 648 synthetic regression problems, also generated to explore the different inter-targets characteristics. In the experiments, the ST approach and three MTR methods were evaluated: Stacking Single Target (SST) [12], Multi-output Tree Chaining (MOTC) [4] and Ensemble of Regressor Chains (ERC) [12]. Thus, the meta-knowledge was generated with different datasets, with different biases, often used for multi-target benchmarking [13]. In the experiments, we compared Naive Bayes (NB), Random Forest (RF), Extreme Gradient Boosting (XGBoost) and Support Vector Machine (SVM) as meta-learners using their default hyperparameter values.

This paper is structured as follows. Section II presents the background on using MtL for MTR; section III describes the experimental methodology; the results are discussed in section IV; finally, the conclusions and future work are presented.

II. Background

Many ML algorithms have been proposed for different prediction tasks. However, the '*No free lunch*' theorem [14] states there is no one algorithm suitable for every dataset. A possible solution is to recommend the best algorithm for each problem.

The notion of algorithm recommendation problems was introduced in [15], grounded on selecting one algorithm from a portfolio of options. Given a set of datasets $\mathcal{P}$ composed of instances from a distribution $\mathcal{Q}$; a set of algorithms $\mathcal{A}$;

and a performance measure $\mathcal{M} : \mathcal{P} \times \mathcal{A} \rightarrow \mathbb{R}$; the algorithm recommendation problem is to find a mapping $m : \mathcal{P} \rightarrow \mathcal{A}$ that optimizes the expected performance measure for the instance problems described in $\mathcal{Q}$.

In practice, there are some alternatives to induce this mapping between algorithms and datasets/problems: one of them is through the Meta-learning (MtL) [8]. The core concept of MtL is to exploit past learning experiences in a particular type of task and solutions by adapting learning algorithms and data mining processes. This is done by extracting features from a dataset, named as meta-features, to represent a dataset and the performance of the ML algorithms when applied on it. The relation between *meta-features* and the ML performance provides information to select the most suitable algorithm for new datasets. Thus, ML algorithms are applied to a meta-dataset, whose examples are described in terms of meta-features, to induce a meta-model.

In the last years, MtL has been used for: algorithm selection [16], segmentation algorithm recommendation [17], and hyperparameter tuning [18].

A. Multi-target regression

Multi-target Regression (MTR) is related to the problems with multiple continuous outputs. In this way, to solve these problems a function or a collection of functions $\mathcal{H}$ that models the relationship from input ($\mathcal{X}$) to output ($\mathcal{Y}$) is created. If $\mathcal{X}$ is composed of m input variables and $\mathcal{Y}$ has d targets, the prediction problem can be stated as:

$$\mathcal{H} : \mathcal{X}_{1...m} \rightarrow \mathcal{Y}_{1...d} \tag{1}$$

Then, for each vector that belongs to $\mathcal{X}$, $\mathcal{H}$ is capable of predicting an output vector that is the best approximation of the true output vector [12].

MTR methods might use one of two main procedures: Algorithm Adaptation or Problem Transformation [19]. The first one adapts well-known algorithms, such as: Artificial Neural Networks (ANNs); Random Forest (RF) and Support Vector Machines (SVMs), to deal with multiple outputs, modeling the problem at once. On the other hand, problem transformation methods modify the original input task aiming at exploring the correlation among the targets. Spyromitros-Xioufis *et al.* [12] proposed two problem transformation methods that contributed notably to the area: Stacking Single Target (SST) and Ensemble of Regressor Chains (ERC). The SST method builds one model for each target d, which are iteratively stacked to the input, and induced new d models over the augmented input. The prediction of these last models are the final predictions.

Differently, the ERC method creates regressors based on a different order of the targets. For each order, models are trained sequentially: the model that is trained for the second response considers the model trained for the first one. Both models are used in the induction of the third regressor, and so forth. In the end, for each target, the prediction is the average of the predictions of the trained regressors.

These both methods inspired the development of new MTR methods [20]–[22]. One of them, the Multi-output Tree Chaining (MOTC) [4], is a method that requires less memory and training time than ERC, besides generating an interpretation of the targets' dependencies. It creates regressors from a tree built based on correlation assessment of the targets. The training of the models is performed from the leaves to the root, stacking the models' predictions as new inputs.

B. Meta-learning for Multi-target regression

During the literature research, we did not find any papers employing MtL for MTR. However, in some studies, the authors investigated similar problems, such as Multi-label Classification (MLC) problems.

Considering $\mathcal{L}$ the set of labels, differently from Single-label classification task, which there is just one label $L_i \in \mathcal{L}$ to predict for each dataset's example, in MLC tasks the examples are associated with more than one label, i.e., it is necessary to learn how to associate the example with a subset of $\mathcal{L}$.

Similarly to the problem investigated in this paper, many MLC methods [23] were proposed, but there is few research concerning when each method is more efficient.

To select a MLC method and configure their hyperparameters for a given dataset, de Sá *et al.* [24] applied Evolutionary Algorithms (EA). This study was carried using 31 MLC methods, in 3 different datasets. The EA selection outperformed or at least draw the baselines in most of the cases. Also in this direction, the pioneering research based on MtL was done by Chekina *et al.* [25]. They evaluated 11 different multi-label methods, grouping them into: Single-Classifier Algorithms and Ensemble-Classifier Algorithms. They performed experiments in 12 datasets of MLC from the literature. The results showed that employing MtL to select one method in MLC tasks is promising, since in most of the experimented cases, to apply the recommendation through MtL was better than selecting one method for all tasks or selecting it randomly.

MLC tasks are similar to MTR tasks, since both deal with the prediction of multiple targets using a common set of features. The main difference is the type of the predicted variable: while in MLC the targets are binary, in MTR the outputs are continuous. Indeed, both tasks can be seen as a more general learning task of multi-target prediction with different types of variables to predict [12]. Therefore, given that MtL was successfully applied to select MLC methods, it is significant to experiment MtL to select MTR methods.

III. Material and Methods

Fig 1 provides an overview of the adopted experimental methodology. First, we performed exhaustive experiments evaluating all the MTR methods in all available datasets. We also identified the best method for each dataset, selecting the one with the smallest Average Relative Root Mean Square Error (aRRMSE). This information is used to define the meta-label. At the same time, a set of measures, named meta-features, are also extracted to describe each dataset. We then unify the meta-feature values with the meta-labels to compose

our meta-dataset. Then, we can employ ML algorithms to predict the best MTR method for a new unseen dataset. The next subsections describe each one of these processes with details.

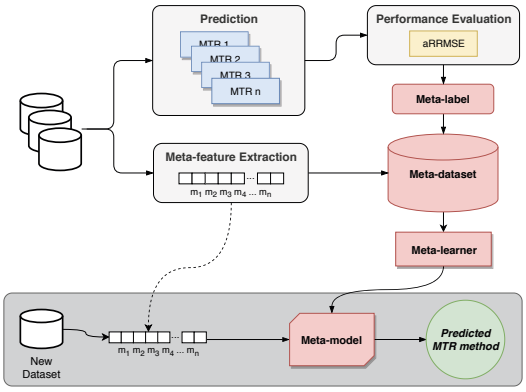

Fig. 1. Overview of the procedure to select a Multi-target Regression method through Meta-learning.

A. Datasets

In the experiments, the meta-dataset was composed of 648 benchmarking synthetic datasets[1], generated by following the procedure described in [13]. We used synthetic datasets to overcome the lack of real datasets that meet specifics scenarios of inter-targets dependencies, complexity levels from the input to output relations, and cover a different number of input features and targets. To create a wide possibility of datasets, the parameters of the dataset generator assumed the values presented in Table I. The numeric targets were built upon math expressions of identity, quadratic, and cubic functions, or their combination.

TABLE I
PARAMETERS USED TO GENERATE SYNTHETIC BASE LEVEL DATASETS.

Symbol	Hyperparameter	Values
N	Number of instances	{500, 1000}
m	Number of features	{15, 30, 45, 60, 75, 90}
d	Number of targets	{3, 6}
g	Generating groups	{1, 2}
η	% Instances affected by noise	{1, 5, 10}

B. Meta-features

Each base-level dataset is represented by a vector of characteristics, the meta-features. In [8] the authors list some requirements that a meta-features must follow: they need to have good discriminative power, their extraction should not be computational complex and the number of meta-features should not be large to avoid overfitting.

[1]The generated datasets are available for download in: http://www.uel.br/grupo-pesquisa/remid/?page_id=145

In our meta-level experiments, a set of 58 meta-features were explored. They included measures from different categories: statistical information about the dataset (STAT), correlation between attributes and targets (COR), performance metrics related to a linear regression (LIN), distribution of the dataset (DIM) and smoothness of the data (SMO) [18], [26].

It is important to mention that some of these meta-features were designed for problems with one single output. Since we are dealing with multi-target problems, the real value of the meta-features were aggregated, given that a meta-feature is extracted for each target. To overcome this problem, the meta-feature was extracted for each target, then the average, standard deviation, maximum and minimum was added to the set of meta-features [27]. Most of the meta-features values were extracted using the R package ECoL [26]. A complete list of the meta-features used in the experiments is presented in Table II.

TABLE II
TYPE, ACRONYM, AGGREGATION FUNCTION (WHEN APPLIED) AND DESCRIPTION OF META-FEATURES USED IN THE EXPERIMENTS.

Type	Acronym	Aggregation Functions	Description
STAT	n.samples	-	Number of samples
	n.attributes	-	Number of attributes
	n.targets	-	Number of targets
	target.ratio	-	Ratio between targets and attributes
	pc[1-3]	-	First three components of the Principal Components Analysis
DIM	T2	-	Average number of samples per dimension
	T3	-	Average intrinsic dimensionality per number of examples
	T4	-	Intrinsic dimensionality proportion
COR	cor.targets	{avg,max,min,sd}	Correlation between targets
	C1	{avg,max,min,sd}	Maximum feature correlation to the output
	C2	{avg,max,min,sd}	Average feature correlation to the output
	C3	{avg,max,min,sd}	Individual feature efficiency
	C4	{avg,max,min,sd}	Collective feature efficiency
LIN	regr.L1	{avg,max,min,sd}	Distance of erroneous instances to a linear classifier
	regr.L2	{avg,max,min,sd}	Training error of a linear classifier
	regr.L3	{avg,max,min,sd}	Nonlinearity of a linear classifier
SMO	S1	{avg,max,min,sd}	Smoothness of the output distribution
	S2	{avg,max,min,sd}	Smoothness of the input distribution
	S3	{avg,max,min,sd}	Error of a k-nearest neighbor regressor
	S4	{avg,max,min,sd}	Non-linearity of nearest neighbor regressor

C. Meta-labels

ST approach and three MTR methods were explored in experiment: SST, ERC [12] and MOTC [4]. Even being the most simple, the ST approach was included in the experimental setup because it can perform better than MTR methods in problems with limited inter-target dependency. On the other hand, the other three MTR methods were selected because they offer a proper trade-off between performance and time complexity, as concluded from [13].

These four different methods mentioned above were executed for every single base-level dataset. Their induced models were assessed in terms of Average Relative Root Mean Square Error (aRRMSE) evaluation measure defined in Equation 2, where N represents the number of instances, and y, $\hat{y}$ and $\overline{y}$ represent, respectively, the true, predicted and mean values of the target.

Support Vector Machine was used as base regressor, performing a k-Fold Cross-Validation (CV) resampling strategy, with $k = 10$. SVM was chosen as base regressor due to its usage in the most of MTR Problem transformation literature [1], [3], [4], [21], [28]. The method with the smallest aRRMSE [19] was chosen as the best multi-target method for every dataset. The experiments were performed using the `mtr-toolkit`[2], implemented in R. Thus, our meta-dataset was a multi-class meta-label with four different levels indicating the best MTR method or ST regression. The class distribution (%) in the meta-dataset is also presented in Table III.

$$\text{aRRMSE} = \frac{1}{d}\sum_{t=1}^{d}\sqrt{\frac{\sum_{i=1}^{N}(y_t^i - \hat{y}_t^i)^2}{\sum_{i=1}^{N}(y_t^i - \overline{y})^2}} \quad (2)$$

TABLE III
SPECIFICATION OF THE META-DATASET USED IN EXPERIMENTS

	ERC	MOTC	SST	ST	**Total**
examples	166	89	362	31	648
%	25.6	13.7	55.8	4.9	100

D. Meta-learners

Four ML algorithms, with different learning biases, were used as meta-learners: Naive Bayes (NB) [29], Random Forest (RF) [30], Support Vector Machine (SVM) [31] and Extreme Gradient Boosting (XGBoost) [32]. These algorithms were selected due to their widespread use and capacity of high-performance models induction. The k-Fold CV resampling methodology was also adopted in the meta-level of the experiments to assess the predictive performance of the meta-learners, with $k = 10$ folds. All the ML algorithms were implemented in R, using the `mlr` package and their correspondent default hyperparameters.

E. Evaluation measures and baselines

Seven evaluation metrics were used to assess the predictive performance of the induced models: Accuracy, Balanced per class accuracy, Precision, Recall, F-score (f1), Sensitivity and Specificity.

Besides, we used two different baselines from the MtL literature for comparisons: a model that always recommends the majority class for the whole dataset (`Majority`) and a model that provides random recommendations (`Random`). These baselines are widely used to endorse the need for a recommendation system [8]. Also, we used an upper-bound as the ground-truth (`Truth`).

IV. Results and Discussion

The results were organized starting by exposing the results regarding the predictive performance of meta-models from different ML algorithms. Afterward, based on the RF meta-model performance, the meta-features were compared and discussed. Finally, some contributions and open issues related to MtL and MTR were presented.

A. Predictive Performance

The predictive performance obtained by the four meta-learners and the baselines are presented as a radar chart in Fig. 2. In this figure, each line represents a meta-model and each vertex its related to a different performance measure. The larger the area in the radar chart, the better the meta-model.

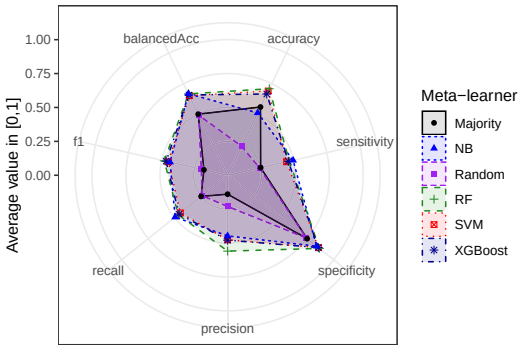

Fig. 2. Performance of the meta-models

Looking at the radar chart, it is possible to observe that all meta-models had a superior performance than `Random` baseline for all metrics. The same occurs for `Majority`, except for accuracy with NB, since `Majority` has 55.8% of accuracy, whereas the NB meta-model achieved 51.08%. Still for this metric, RF obtained the best results with 70.83% of accuracy. Following the RF, the SVM achieved 68.9% and XGBoost was the third, with 66.51% of accuracy. The only metric that RF meta-model did not obtain the higher value was Sensitivity, when NB was the best with 0.49. Regarding the other evaluation metrics, RF achieved the best results, with 0.86 of Specificity, 0.55 of Precision, 0.46 of Recall, 0.47 of F1 and 66.59% of Balanced per class accuracy.

Although three of four meta-models overcame the baselines for all metrics, the predictive performance did not achieve high values, which might be related to the meta-dataset imbalance problem. However, the superiority of the MtL recommending system regarding the baselines was confirmed by statistical tests. We used the Friedman test, with a significance level of $\alpha = 0.05$. The null hypothesis is that the recommendation by the meta-models and by the baselines are similar. Anytime the null hypothesis is rejected, the Nemenyi post hoc test can be applied, stating that the performance of the two approaches are significantly different if their corresponding average ranks differ by at least a Critical Difference (CD) value. When multiple algorithms are compared in this way, a graphic representation can be used to represent the results with the CD diagram, as proposed by Demšar [33].

[2]https://github.com/smastelini/mtr-toolkit

The meta-models (RF, SVM, XGBoost, NB) were compared with `Truth` (expected method), the `Majority` (which always predicts the SST) and `Random` (the random selection of a method for each dataset), using the aRRMSE of the prediction as performance metric. This analysis is shown in Fig. 3, using the results from the Nemenyi test.

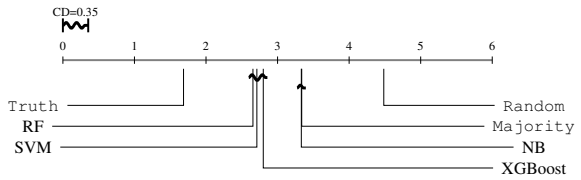

Fig. 3. Comparison of the aRRMSE values obtained by meta-models when recommending MTR methods according to the Nemenyi test. Groups that are not significantly different (α= 0.05 and CD = 0.35) are connected.

As exposed in Fig. 3, no solution was similar to the `Truth`, which was expected due to the predictive performance. However, the RF, SVM, XGBoost are connected, which means they were similar and superior the baselines `Majority` and `Random`. This fact supports the benefit of using MtL recommending system in comparison to select a specific algorithm for every dataset or select it randomly.

B. Relative importance of the meta-features

Random Forest meta-model was used to assess the importance of each meta-feature by using the RF Feature Importance metric. This metric is calculated by permuting the values of a feature in the Out-of-Bag (OOB) samples and recalculating the OOB error in the whole ensemble. In other words, if substituting the values of a meta-feature by random values results in error increase, this meta-feature is considered important. Otherwise, if the error decreases, the resulting importance is negative. Thus, the meta-feature is considered not important and should be removed from modeling. This procedure could be performed for each meta-feature toward explaining its impact [30]. Fig. 4 shows the meta-feature importance for the meta-dataset.

Correlation and Linearity meta-features achieved the higher values of importance, especially the Minimum value of distance of erroneous instances to a linear classifier (12.54), Minimum value of non-linearity of a linear classifier (12.51) and the Standard Deviation of the Maximum feature correlation to the output (11.37). Once the MTR method tries to explore the correlation between the features and the targets in different ways, their selection makes sense. The number of targets, attributes and samples had low importance. This might have occurred because these meta-features did not influence in the predictive performance, showing that the MTR methods used in the experiments can deal with different numbers of targets, attributes and samples in the same way.

C. Insights and open issues

It is important to highlight the meta-label attribution was straightforward related to the highest predictive performance (low aRRMSE) based on the ranking of methods. Differences between the predictive performance of the MTR methods, independent of their magnitude, were not considered while building the meta-dataset.

Alternatively, the meta-label assessment could be performed by indicating two or more methods suitable to solve a given problem in the case of no statistical difference between their performances. However, this scenario poses an additional challenge to deal with a multi-label problem in the meta-level of the recommending system.

Another important issue was the fact of meta-label assessment was made regarding only low predictive error of MTR methods. In some cases, e.g., Online Multi-target Regression [34], the most proper method concerns to address a trade-off among predictive performance, memory, and time cost when predicting the output. This scenario demands additional information, as well as complexity, toward identifying the best MTR method to be learned by the recommending system.

V. Conclusions and Future Work

In this study, a framework for recommending MTR methods using meta-learning was presented. A meta-dataset, composed with 648 datasets used for MTR methods benchmark, was created for the induction of meta-models toward predicting the best one for a given dataset. Experiments performed with the meta-dataset and four meta-learners led to 70.83% of accuracy with RF, the best recommender. Besides, it overcame the baselines, and statistical tests showed that the recommendation system was better than select one for every task or selecting a method randomly. The analysis of meta-feature importance revealed that correlation between targets and error of a linear classifier were the most useful features to predict the performance of a MTR method for a given unseen dataset.

As future work, besides implementing more meta-features, we intend to use more MTR benchmarking datasets, in order to improve the generalization capability of the meta-models. Also, we expect to apply MLC to predict the MTR method and its base regressor. Further information related to the memory and time cost will be used to match the requirement of different scenarios, e.g., Online MTR.

Acknowledgements

The authors would like to thank the financial support of Coordination for the Improvement of Higher Education Personnel (CAPES) - Finance Code 001 -, the National Council for Scientific and Technological Development (CNPq) of Brazil - Grant of Project 420562/2018-4 - and São Paulo Research Foundation (FAPESP) - grant #2018/07319-6.

References

[1] G. Tsoumakas, E. Spyromitros-Xioufis, A. Vrekou, and I. Vlahavas, "Multi-target regression via random linear target combinations," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 225–240.

[2] J. Levatić, M. Ceci, D. Kocev, and S. Džeroski, "Semi-supervised learning for multi-target regression," in *International Workshop on New Frontiers in Mining Complex Patterns*. Springer, 2014, pp. 3–18.

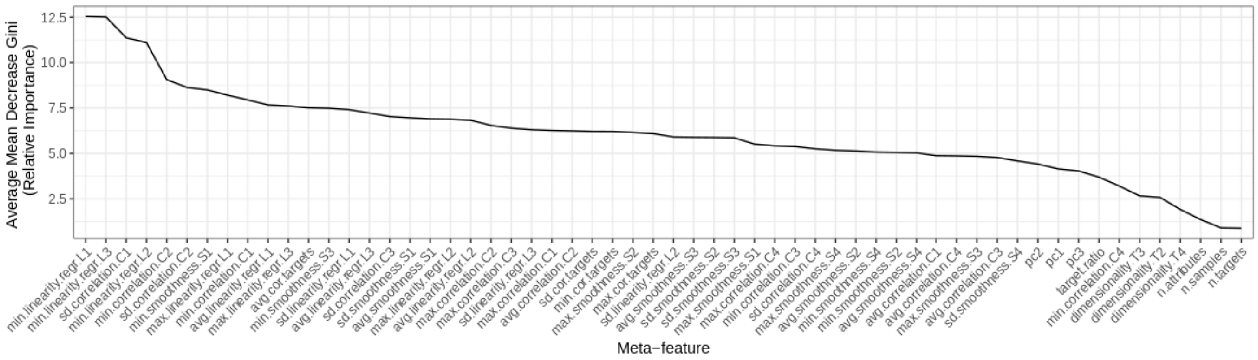

Fig. 4. Average relative importance of the meta-features obtained from RF importance. The names of the meta-features in the x-axis follow the acronyms presented in Table II.

[3] E. J. Santana, B. C. Geronimo, S. M. Mastelini, R. H. Carvalho, D. F. Barbin, E. I. Ida, and S. Barbon, "Predicting poultry meat characteristics using an enhanced multi-target regression method," *Biosystems Engineering*, vol. 171, pp. 193 – 204, 2018.

[4] S. M. Mastelini, V. G. T. da Costa, E. J. Santana, F. K. Nakano, R. C. Guido, R. Cerri, and S. Barbon, "Multi-output tree chaining: An interpretative modelling and lightweight multi-target approach," *Journal of Signal Processing Systems*, pp. 1–25, 2018.

[5] E. J. Santana, J. A. P. R. d. Silva, S. M. Mastelini, and S. Barbon Jr., "Stock portfolio prediction by multi-target decision support," *iSys-Revista Brasileira de Sistemas de Informação*, vol. 12, no. 1, 2019.

[6] B. Bilalli, A. Abelló, and T. Aluja-Banet, "On the predictive power of meta-features in openml," *International Journal of Applied Mathematics and Computer Science*, vol. 27, no. 4, pp. 697–712, 2017.

[7] T. Cunha, C. Soares, and A. C. de Carvalho, "Metalearning and recommender systems: A literature review and empirical study on the algorithm selection problem for collaborative filtering," *Information Sciences*, vol. 423, pp. 128–144, 2018.

[8] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to Data Mining*, 2nd ed. Springer Verlag, 2009.

[9] S. Ali and K. A. Smith-Miles, "A meta-learning approach to automatic kernel selection for support vector machines," *Neurocomputing*, vol. 70, no. 1-3, pp. 173–186, 2006.

[10] A. L. D. Rossi, A. C. P. de Leon Ferreira, C. Soares, B. F. De Souza *et al.*, "Metastream: A meta-learning based method for periodic algorithm selection in time-changing data," *Neurocomputing*, vol. 127, pp. 52–64, 2014.

[11] M. Reif, F. Shafait, and A. Dengel, "Meta-learning for evolutionary parameter optimization of classifiers," *Machine learning*, vol. 87, no. 3, pp. 357–380, 2012.

[12] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-target regression via input space expansion: treating targets as inputs," *Machine Learning*, vol. 104, no. 1, pp. 55–98, 2016.

[13] S. M. Mastelini, E. J. Santana, V. G. T. da Costa, and S. Barbon, "Benchmarking multi-target regression methods," in *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, 2018, pp. 396–401.

[14] D. H. Wolpert, "The lack of a priori distinctions between learning algorithms," *Neural computation*, vol. 8, no. 7, pp. 1341–1390, 1996.

[15] J. R. Rice, "The algorithm selection problem," *Advances in Computers*, vol. 15, pp. 65–118, 1976.

[16] D. G. Ferrari and L. N. De Castro, "Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods," *Information Sciences*, vol. 301, pp. 181–194, 2015.

[17] G. F. Campos, S. Barbon, and R. G. Mantovani, "A meta-learning approach for recommendation of image segmentation algorithms," in *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2016, pp. 370–377.

[18] R. G. Mantovani, A. L. Rossi, E. Alcobaça, J. Vanschoren, and A. C. de Carvalho, "A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves svm classifiers," *Information Sciences*, vol. 501, pp. 193 – 221, 2019.

[19] H. Borchani, G. Varando, C. Bielza, and P. Larrañaga, "A survey on multi-output regression," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 5, pp. 216–233, 2015.

[20] G. Melki, A. Cano, V. Kecman, and S. Ventura, "Multi-target support vector regression via correlation regressor chains," *Information Sciences*, vol. 415, pp. 53–69, 2017.

[21] S. M. Mastelini, E. J. Santana, R. Cerri, and S. Barbon, "DSTARS: A multi-target deep structure for tracking asynchronous regressor stack," in *2017 Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, oct 2017.

[22] J. M. Moyano, E. L. Gibaja, and S. Ventura, "An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, jun 2017.

[23] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining (IJDWM)*, vol. 3, no. 3, pp. 1–13, 2007.

[24] A. G. de Sá, G. L. Pappa, and A. A. Freitas, "Towards a method for automatically selecting and configuring multi-label classification algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2017, pp. 1125–1132.

[25] L. Chekina, L. Rokach, and B. Shapira, "Meta-learning for selecting a multi-label classification algorithm," in *2011 IEEE 11th International Conference on Data Mining Workshops*. IEEE, 2011, pp. 220–227.

[26] A. C. Lorena, A. I. Maciel, P. B. de Miranda, I. G. Costa, and R. B. Prudêncio, "Data complety meta-features for regression problems," *Machine Learning*, vol. 107, no. 1, pp. 209–246, 2018.

[27] A. Rivolli, L. P. F. Garcia, C. Soares, J. Vanschoren, and A. C. P. L. F. de Carvalho, "Towards reproducible empirical research in meta-learning," *CoRR*, vol. abs/1808.10406, 2018. [Online]. Available: http://arxiv.org/abs/1808.10406

[28] E. J. Santana, S. M. Mastelini, and S. Barbon Jr., "Deep Regressor Stacking for Air Ticket Prices Prediction," in *XIII Brazilian Symposium on Information Systems: Information Systems for Participatory Digital Governance*. Brazilian Computer Society (SBC), 2017, pp. 25–31.

[29] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[30] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[31] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[32] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794.

[33] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[34] A. Osojnik, P. Panov, and S. Džeroski, "Tree-based methods for online multi-target regression," *Journal of Intelligent Information Systems*, vol. 50, no. 2, pp. 315–339, 2018.