Submission date: 18/06/2018
Resubmission date: 06/04/2019
Camera ready submission date: 13/04/2019

1st round notification: 16/03/2019
2nd round notification: 11/04/2019
Available online: 13/04/2019
Publication date: 16/04/2019

Section: extended version
Best papers from the XIV Brazilian Symposium on Information Systems (SBSI 2018)

Evaluation of Multi-Target Regression to Support Decision on Stock Portfolio Investment

Everton Jose Santana[1], João Augusto Provin Ribeiro da Silva[1], Saulo Martiello Mastelini[2], Sylvio Barbon Jr[1]

[1] Computer Science Department – State University of Londrina
Londrina, Paraná – Brazil

[2]Institute of Mathematics and Computer Sciences – University of São Paulo
São Carlos, São Paulo – Brazil

evertonsantana@uel.br, joaoaugustoprovin@gmail.com, mastelini@usp.br, barbon@uel.br

Abstract. *Investing in the stock market is a complex process due to its high volatility caused by factors as exchange rates, political events, inflation and the market history. To support investor's decisions, the prediction of future stock price and economic metrics is valuable. With the hypothesis that there is a relation among investment performance indicators, the goal of this paper was exploring multi-target regression (MTR) methods to estimate 6 different indicators and finding out the method that would best suit in an automated prediction tool for decision support regarding predictive performance. The experiments were based on 4 datasets, corresponding to 4 different time periods, composed of 63 combinations of weights of stock-picking concepts each, simulated in the US stock market. We compared traditional machine learning approaches with seven state-of-the-art MTR solutions: Stacked Single Target, Ensemble of Regressor Chains, Deep Structure for Tracking Asynchronous Regressor Stacking, Deep Regressor Stacking, Multi-output Tree Chaining, Multi-target Augment Stacking and Multi-output Random Forest (MORF). With the exception of MORF, traditional approaches and the MTR methods were evaluated with Extreme Gradient Boosting, Random Forest and Support Vector Machine regressors. By means of extensive experimental evaluation, our results showed that the most recent MTR solutions can achieve suitable predictive performance, improving all the scenarios (14.70% in the best one, considering all target variables and periods). In this sense, MTR is a proper strategy for building stock market decision support system based on prediction models.*

Keywords. *Stock market; Multi-target regression; Decision support system; Machine Learning*

1. Introduction

The prediction of stock market is a very challenging task because it is affected by several macro-economic factors, for instance exchange rates, political events, recession or expansion periods, and investor's expectations [Atsalakis and Valavanis 2009, Hu et al. 2017].

Cite as: Santana, E.J., Silva, J. A. P. R. da, Mastelini, S. M. & Barbon Jr., S. (2019). Stock Portfolio Prediction by Multi-Target Decision Support. iSys: Revista Brasileira de Sistemas de Informação (Brazilian Journal of Information Systems), 12(1), 05-27.

Other factors that usually influence this volatility are inflation, interest rates, rising bond yields and the stock market itself, since it can be overheated[1].

When making a decision, the stock price is not the main information source, so that investors consider parameters that give information on return rates and the associated risks, such as price to book value ratio, dividend-price ratio, return on investment, return on equities and systematic risk [Roko and Gilli 2008, Basu 1977, Liu and Yeh 2017, Bruni 2017]. The use of decision support tools in stock trading is also helpful. These tools, as Kvout[2] and Trade Ideas[3], are becoming more sophisticated as they use artificial intelligence to improve the prediction of investments performance.

In fact, many computational intelligence (CI) and machine learning (ML) algorithms tried to address stock forecast problem: artificial neural networks (ANNs), linear and multi-linear regression (LR, MLR), support vector machine (SVM), autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) models, genetic algorithms (GAs), random forest (RF) and random walk (RW) to name some [Atsalakis and Valavanis 2009, Bahrammirzaee 2010, Khaidem et al. 2016, Hu et al. 2017]. In these works, the prediction of the stock market price or economic properties were done by building ML solutions which deals with a single response or output, i.e., a single-target (ST) approach. Besides, as the stock market outcomes are continuous values, the related ML problems are called regression tasks [Kocev et al. 2013].

Until now, most of the existing decision support systems modelled the stock market performance indicators as separated ST problems. Multi-target regression (MTR), however, is a research field of ML that deals with predictive problems which present multiple continuous outputs or responses, and could be better explored in stock prediction problem. In these tasks, the targets may present underlying inter-dependencies, influencing and being influenced among themselves [Kocev et al. 2013, Borchani et al. 2015, Spyromitros-Xioufis et al. 2016]. Therefore, MTR aims at modelling not only the input to output relationships in a predictive problem, but also the inter-output relationships.

Many MTR methods with noticeable performance were created in last few years and have not been used yet to predict performance indicators of a stock portfolio. Our hypothesis is that the multiple stock market performance indicators may present underlying relationships among themselves. Since different performance indicators share the same explaining features, they can be modelled as an MTR problem. In this sense, a regression model for a performance indicator could use information of other outcomes to yield better predictions and compose a more reliable decision support tool.

Figure 1 presents an overview of two prediction methods: single-target and multi-target. This kind of prediction method composes a decision support tool in extracting knowledge from stock-picking concepts as a potential aid to the decision making of an

[1]CNNMoney (New York) (2018, February 12). How to handle stock market volatility and keep your retirement plan in check. *CNN Money*. Retrieved from `http://money.cnn.com/2018/02/12/pf/applenews-stock-market-dow-down/index.html`. [Accessed: 25th February 2018]

[2]www.kavout.com

[3]www.trade-ideas.com

investor.

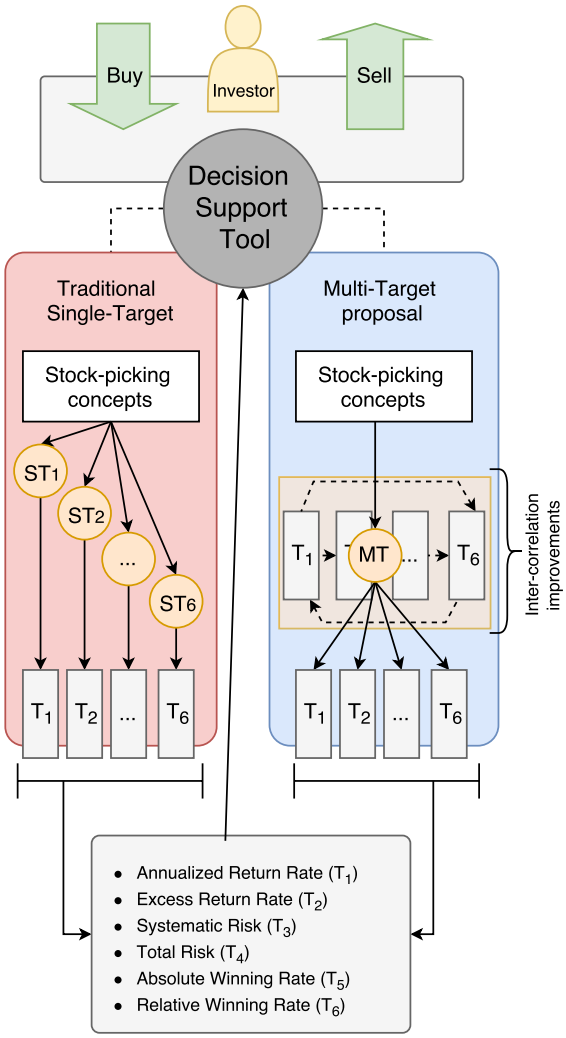


Figure 1. Decision Support Tool from stock market investor, comparing single-target and multi-target prediction kernels.

This paper is an extension of the work of [Silva et al. 2018], whose goal was proposing a kernel to decision support tool based on MTR to predict performance indicators of stock market, showing that this approach can generate an accurate model due to the inter-target influence in this model. This work aims at obtaining the method that would result in the best prediction performance for the given problem. In this way, we extend the previous work in the following main aspects:

- Three other MTR methods (DRS, MOTC and MTAS) were added in the discussion, as well as another regressor (XGBoost);
- Other metrics and methods were adopted to investigate inter-target correlation and to analyse the results;
- The percentage of error reduction and critical difference diagrams were extended for all regressors instead of showing only for SVM;

- Discussion on complexity was also added.

The work is organized as follows: Section 2 presents related works, Section 3 describes the experimental setup, Section 4 reports the results and their analysis, followed by Section 5, that concludes the work.

2. Related Works

Machine learning algorithms have been vastly used in the literature to predict stock market characteristics. [Roko and Gilli 2008], for example, used classification trees improved by bootstrap aggregation to predict assets which achieve future returns above the average. According to the authors, the performance of these portfolios are significantly superior to recorded indexes.

However, the most common problems deal with continuous responses, implying in regression models. [Patel et al. 2015] used ANNs, SVM, RF and Naive-Bayes to predict the direction of movement of stock and stock price index for Indian stock markets and compared technical parameters such as Relative Strength Index, Momentum, Accumulation/Distribution Oscillator and Commodity Channel Index with a discretization as input data resulted in higher accuracy than using open, high, low and close prices continuous parameters.

[Hu et al. 2017] applied sharp ratio, a profit metric, to tune support vector regression models in stock index forecasting. The results showed that profit guided stock index forecasting is competitive and is able to produce significantly better performances than statistical error guided models.

[Liu and Yeh 2017] proposed the use of mixture design, a kind of design of experiment which the independent factors are used in different proportions, and ANN to build models in order to optimize weighted scoring stock selection. They found out useful models for investors to search for the optimal investment strategies to meet their specific preferences.

When it comes to MTR applied to stock forecasting, [Xiong et al. 2014] used Multiple-output support vector regression with a firefly algorithm to estimate the lower and upper values of stock price index, resulting in a promising alternative for forecasting interval-valued financial time series.

Multi-target regression intends to simultaneously predict multiple continuous variables in a common set of inputs. Furthermore, MTR provides a more appropriate interpretability of real life problems since it takes the relationship between the targets into consideration. The prediction of MT tasks had, commonly, been made through two base approaches: algorithm adaptation and problem transformation [Borchani et al. 2015]. [Kocev et al. 2013] denotes the same strategies as global and local approaches, respectively.

Algorithm adaptation (global) approach provides challenge, since it does not only aim at dealing with multiple targets by changing a well known ST regression technique, but also the investigation, modelling and interpretation of the possible relations between the problem output variables. Through refinement methods as node splitting regression

trees [Kocev et al. 2007] and optimization functions (SVMs) [Borchani et al. 2015], this approach leads to alteration in the original technique. Although complex, algorithm adaptation methods have reached satisfactory prediction performances, along with the generation of unique models and target correlation exploration [Aho et al. 2012, Kocev et al. 2007, Kocev et al. 2013, Borchani et al. 2015].

Problem transformation (local), the latter approach, consists of data manipulation and regression techniques as a means of simultaneously predicting separated ST problems. Even though building separate models for each instance leads to ignoring the relationship between targets, this method may, sometimes, provide superior predictive performance and it was used as a baseline method in multiple MTR works [Aho et al. 2012, Borchani et al. 2015, Spyromitros-Xioufis et al. 2016, Mastelini et al. 2017]. Nevertheless, an MTR approach carries more potential on the quality of predictions due to target dependence exploration.

Throughout the years, some problem transformation methods were created, aiming at exploring inter-target dependencies through the employment of multiple ST regressors [Tsoumakas et al. 2014, Borchani et al. 2015, Spyromitros-Xioufis et al. 2016, Santana et al. 2017, Moyano et al. 2017, Melki et al. 2017, Mastelini et al. 2017]. Among them, some multi-label classification (MLC) methods were adapted to MTR. As proposed in [Spyromitros-Xioufis et al. 2016], two relevant methods, inspired by the MLC research area, came to fruition: SST (Stacked Single-Target) and ERC (Ensemble of Regressor Chains). They also influenced some posterior researches in MTR [Santana et al. 2017, Moyano et al. 2017, Melki et al. 2017, Mastelini et al. 2017]. To the best of our knowledge, problem transformation multi-target methods were never used to predict performance indicators of a stock portfolio.

The SST method consists of two major steps. It begins by training separate d base ST models, where d represents the number of targets. However, the important part lies within the second stage, where instead of directly using these models for prediction, the method employs their predictions as new inputs and performs an additional training step for each target, thus generating d meta-models. The employment of predictions from base models as new inputs for meta models is called stacking, which is also employed in ST problems as a strategy for combining different predictors in ensembles [Wolpert 1992]. In essence, SST implements the idea of correcting the predictions acquired throughout the first stage, thus increasing the task's description capability and the prediction performance with the insertion and exploration of target correlations.

The ERC method makes use of target chains, which are randomly chosen to form a set. By following the chain sequence, ST regression models are formed for each target, and then trained following the order of the sequence. The method creates new datasets with the combination of the native variables and the predictions of the last models. After repeating this process for the whole chain sequence, the training is done. New instances shall be directed to the set of chains. Then, generating the average result of the predicted y values should return the final prediction for the y target. Permutation is an important part of the whole process. ERC makes use of all target combinations, as long as the number of total permutations is equal to or less than 10. If that is not the case, then 10 combinations

are selected.

Deep Structure for Tracking Asynchronous Regressor Stacking (DSTARS) was recently proposed by [Mastelini et al. 2017] as an extension to the original SST idea. The authors proposed the employment of multiple steps of regressor stacking for each target in a dynamic way. Their hypothesis is that the addition of more stacked regressors could decrease the prediction error of the most dependent targets. In this sense, differently from the previous methods, DSTARS considers multiple levels of inter-target dependencies explicitly. The authors reported superior results than the SST and ERC methods in MTR benchmarking problems.

Another extension to the SST approach was proposed by [Santana et al. 2017], named Deep Regressor Stacking (DRS). The authors employed a deep learning strategy to MTR problems by implementing multiple iterations of base and meta models training. In their approach, the meta models act as base models for the next stacking stage. In this sense, at each stacking round d features are added to the original input set, being used along with the previously created ones for training new meta models. This process repeats until no error decrease is observed in a validation set or a maximum iteration criteria is reached. DRS calculates which target generated the smallest errors among all outputs, and removes this response from the deep stacking procedure, adding it as a fixed element of the input set. The final responses for the chosen target comprehend the outputs of the last trained meta model, using stacked features from multiple base models. The deep procedure repeats for the remaining $d - 1$ targets, until all responses are assessed. DRS was evaluated in some scenarios achieving the smallest errors in multiple cases [Santana et al. 2017, Santana et al. 2018], at the cost of being a very computationally expensive method.

The Multi-output Tree Chaining (MOTC) [Mastelini et al. 2018] method was proposed as an extension to the ERC approach, which, instead of randomly defining target orders (as in ERC), employs heuristics (a correlation measurement metric and a statistical bound to rank target relevance) to build a tree structure to represent inter-target dependencies. These trees are called Chaining Trees (CT) and they also represent the strategy in which the regressors are going to be created: the process start from the leaves up to the root; the inner nodes employ the predictions of their descendent's regressors as additional features. As stated by the authors, by using the mentioned strategies, MOTC reaches prediction errors very similar to ERC, whereas being more time and memory conservative. In addition, the CTs offer clues about how the targets interact and influence each other. In their work the authors also reported an unified graph compression of the obtained CTs as means of viewing the chains of dependence among all targets. This strategy is also going to be used in this work.

[Santana et al. 2018] proposed the Multi-target Augment Stacking method as an application of the traditional ST stacking ensemble approach in MTR problems. In their method, an arbitrary number of regression techniques are used to generate the base models, instead of just one as in SST. After, predictions from all base models are added as new inputs, aiming at exploring increased variability and bias for the new features and, thus, decrease prediction error. Additionally, MTAS applies a filtering step where

the inter-dependencies among the outputs are measured via a correlation metric (similar to MOTC). Based on this measurement, only the predictions of outputs that influence the current target are added as new features. The authors applied their method in a food quality evaluation problem, but also validated their approach in benchmarking datasets, achieving competitive results to other methods.

The Multi-output Random Forest (MORF) method [Kocev et al. 2007, Kocev et al. 2013] is an extension to the original RF formulation and represents an algorithm adaptation (global) method. Instead of using regular regression tree algorithms [Breiman et al. 1984, Breiman 2001], MORF employs Predictive Clustering Trees (PCT), which were proposed by [Blockeel et al. 1998]. PCTs create at each data partition clusters containing all instances which match the evaluated decisions made. The root node corresponds to the cluster containing all the training instances. PCT can be seen as a generalisation of traditional decision tree algorithms. The PCTs which compose MORF are grown by creating clusters which aim at minimising the intra-cluster variances, while increasing the inter-cluster variance. The predictions made for each target correspond to the average of all instances which lie in the considered leaf node. MORF was employed in multiple MTR works [Aho et al. 2012, Borchani et al. 2015, Spyromitros-Xioufis et al. 2016] as a comparison method in past years, and it was also used in our experiments.

The asymptotic time complexity of the methods are summarised in Table 1, where d represents the number of targets, m the number of features, N the number of training instances, b the time complexity of the chosen regression algorithm (which depends on N and tends to vary in DRS due to increases in m), r the cost to calculate the chosen importance metric for one of the responses, ζ the total number of nodes in the chaining tree, k the number of partitions of cross-validation, L the number of layers, and t the number of trees in MORF. In the case of MTAS, the complexity of the multiple base regressors employed are represented by $b_1, b_2, ..., b_z$.

Table 1. Asymptotic time complexity of the methods.

Method	**Complexity**	**Author**
ST	$O(d\,b)$	-
SST	$O(2\,d\,b)$	[Spyromitros-Xioufis et al. 2016]
ERC	$O(10\,d\,b)$	[Spyromitros-Xioufis et al. 2016]
DRS	$O((k+1)\,L\,d\,b)$	[Santana et al. 2017]
DSTARS	$O(d\,[r+2\,L\,b])$	[Mastelini et al. 2017]
MOTC	$O(\zeta\,b)$	[Mastelini et al. 2018]
MTAS	$O(d\,(b_1+b_2+...+b_z+2\,b))$	[Santana et al. 2018]
MORF	$O(t\,m\,N(\log^2 N + d\,\log N))$	[Kocev et al. 2013]

Among the evaluated methods, ST, SST, MORF and MOTC can be considered the most lightweight ones. The global method, MORF, indeed is less costly than the local ones. The local methods depend on the choice of the regression techniques to define their definitive costs. In the case of MTAS, multiple techniques can be used, which increases its costs when compared with SST. ERC and MOTC do not differ in their complexity mag-

nitude order, but as empirically demonstrated by MOTC's authors [Mastelini et al. 2018], their method expends less regressors than ERC, being consequently, lighter than the latter. DRS was the most costly evaluated method, since it uses a deep learning strategy, as previously mentioned. DSTARS, by using the chosen set of hyperparameters, presents a cost superior to SST but lesser than DRS's, representing a balanced option to deep exploration of stacked regressors and the employment of a single round of stacking.

3. Experimental Setup

This section presents the datasets used in this work, as well as the chosen regression techniques and evaluation metrics.

3.1. Datasets

The datasets adopted in this work were based on simulations with Standard and Poor's Compustat US database in four periods from 1990 to 2010: the first from September of 1990 to June of 1995, the second from September of 1995 to June of 2000, the third from September of 2000 to June of 2005 and the fourth from September of 2005 to June of 2010. Each of these periods consists of 63 different weighting combinations, selected from the top 10% overall weighted scores. Detailed explanation of the dataset can be consulted in [Liu and Yeh 2017].

It has as feature inputs six different weights related to stock-picking concepts:

1. Book value-to-price ratio (B/P), used to evaluate whether the firm's stock is undervalued or not. The larger the ratios are, the higher the possibility that the stock is undervalued;
2. Sales-to-price ratio (S/P), also used to evaluate whether the firm's stock is undervalued or not. The larger the ratios are, the higher the possibility that the stock is undervalued;
3. Return on equity (ROE), used to evaluate a firm's profitability. The larger the ROE is, the more profitable the firm is;
4. Return rate in the last quarter, used to select stocks with high return rates. Considering that the rebalance period of the portfolio in this study is one-quarter, momentum concept can be adopted, indicating that if the return rate of the stock is currently high, it will continue to go higher in the future;
5. Market capitalization, used to select stocks with high liquidity and low risk;
6. Systematic risk, used to select stocks with low risk in the next holding period. If the systematic risk is greater than 1, its fluctuation in return is greater than the benchmark, and vice versa.

The outputs are 6 investment performance indicators:

1. Annualized Return Rate (ARR), that can be calculated by $((1 + R)^{1/t}) - 1$, with t corresponding to the period (in years) and R to the accumulated return rates;
2. Excess Return Rate (α),
3. Systematic Risk (β), that are coefficients of the regression $R_i - R_f = \alpha + \beta(R_m - R_f)$, where R_i is the investment portfolio return rate, R_f the risk-free return rate and R_m the market return rate;

4. Total Risk (TR), corresponding to the standard deviation of the return rate of the portfolio during a certain period;
5. Absolute Winning Rate (AWR), that is the ratio between the number of portfolio holding periods with positive return rate and the total number of portfolio holding periods;
6. Relative Winning Rate (RWR), that is the ratio between the number of portfolio holding periods with a return rate greater than the market return rate and the total number of portfolio holding periods.

The existence of dependencies among the targets should lead the MTR methods to increase their predictive performance when compared to the ST strategy. To verify this, we have comprised all the considered periods within a single set to measure how the targets relate to each other.

Figure 2 presents the linear correlation coefficients (Pearson coefficient) calculated over all the targets. The closer to one (or negative one) the coefficient, the more correlated (or inversely correlated) are the compared targets.

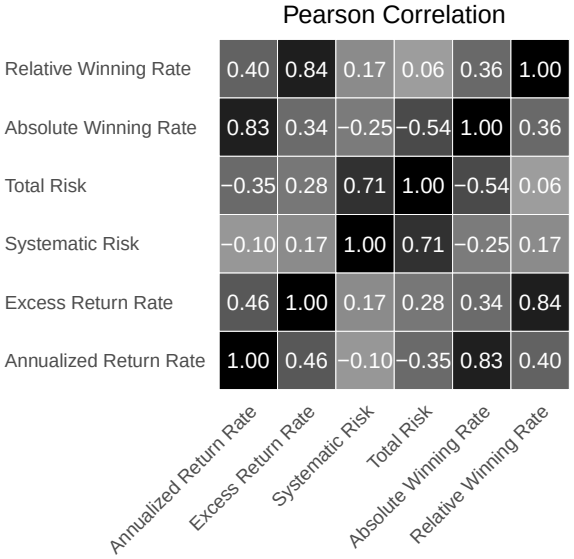


Figure 2. Pearson coefficient calculation for all the evaluated targets.

As shown in Figure 2, prominent relationships were observed for some targets). In Figure 2, for instance, the Annualized Return Rate is highly correlated with the Absolute Winning Rate, as well as the Excess Return Rate is related to the Relative Winning Rate. Some inverse relationships were also observed, such as the Total Risk and the Absolute Winning Rate.

Although the observed correlations correspond only to linear relationships, additional nonlinear inter-dependencies may be explored by MTR methods. In this sense, Random Forest Importance [Breiman 2001, Grömping 2009] measure was also calculated, with the goal to investigate nonlinear relations between target variables. This metric is calculated by the evaluation of the error increase when permuting a feature's values

before Random Forest prediction. A positive value means that the evaluated feature contribute to the explanation of the task. Null or negative results mean that the feature do not contribute and disturb the problem comprehension, respectively. They can be found in Figure 3.

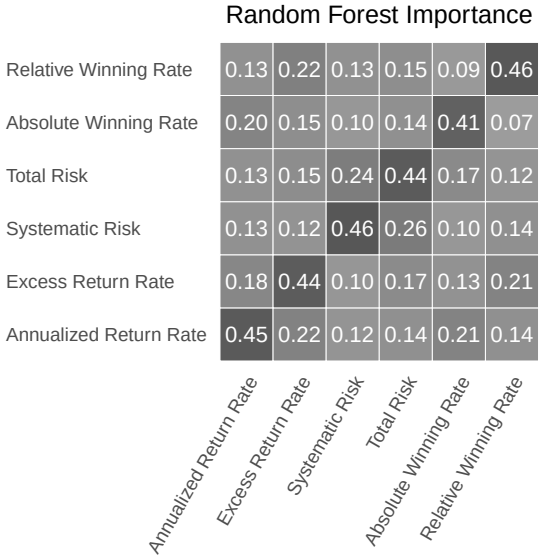


Figure 3. Random Forest Importance for all the evaluated targets.

When observing RF importance (Figure 3), all values were positive, meaning that all targets led to the explanation of others, even that sometimes it was weak, for example 0.07. The strongest relations occur between Excess Return Rate and Relative Winning Rate, and Absolute Winning Rate and Annualized Return Rate.

3.2. Regression Methods

In this work, Single-Target regression was compared to one problem transformation MTR method, Multi-output Random Forest, and six problem transformation methods:

- Stacked Single Target;
- Ensemble of Regressor Chains;
- Deep Structure for Tracking Asynchronous Regressor Stacking;
- Deep Regressor Stacking;
- Multi-output Tree Chaining;
- Multi-target Augment Stacking.

The explanation of these methods can be consulted in Section 2.

3.3. Regression Techniques

As problem transformation approach requires base-learners, the experiments made use of three machine learning regression algorithms: Random Forest (RF), Support Vector Machine (SVM) and Extreme Gradient Boosting (XGBoost). We aimed at evaluating

techniques belonging to different ML paradigms. All of the algorithms had their implementations[4] in the R programming language [Ihaka and Gentleman 1996] with a fixed seed in zero and used standard parameter settings.

Random Forest: As the name suggests, the RF [Breiman 2001] algorithm makes use of multiple decision trees [Breiman et al. 1984]. When applied to regression, the tree predictors do not take on class labels, but actually continuous numerical values. The bagging meta-algorithm [Breiman 2001] is used to grow trees, making different training datasets be formed by bootstrap sampling. By taking the average results of all trees in the Forest, the RF predictor is formed. In our experiments we used the `ranger` R package.

Support Vector Machine: Through the use of support vectors, SVM is a kernel-based technique for classification and regression. Its main idea is to fit a decision hyperplane which adapts to the dealt problem characteristics [Ben-Hur and Weston 2010]. The modeling of nonlinear tasks is done by transforming the input space. To do so, a kernel function is applied to the input variables, possibly increasing their dimension. By changing the characteristics of the input data, data separability also increases. This technique was executed in R through the `e1071` package.

Extreme Gradient Boosting: This framework implements a tree boosting ensemble strategy by sequentially creating new regressors that aim at minimising the prediction error of the previous induced ones. XGBoost has been presenting state-of-the-art results in many prediction tasks [Chen and Guestrin 2016]. This specific implementation of tree boosting got increased attention by implementing optimisation strategies that reduce the time for fitting the regressors, as well as constraints and regularisation terms to avoid overfitting. The experiments were performed using the `xgboost` R package.

3.4. Evaluation metrics

The evaluation was conducted performing a 10-folds cross-validation strategy. We chose three metrics to evaluate the performance of the compared MTR methods and regression techniques on the presented datasets. The mentioned metrics were the Coefficient of Determination (R^2), the average Relative Root Mean Square Error (aRRMSE) and the Relative Performance (RP) [Borchani et al. 2015, Spyromitros-Xioufis et al. 2016]. Besides, we also reported the percentage of error reduction of the local MTR methods over the ST strategy, considering all targets.

The R^2 can be used to evaluate the adequacy of a regression model, related to the amount of variability in the data explained or accounted for by the regression model. It assumes values between 0 and 1. If equal to 1, the model accounts for 100% of the variability in the data [Montgomery and Runger 2014].

The RRMSE compares the predictive error obtained by a regressor when compared with the performance of a simple predictor which always outputs the target mean. In this sense, this metric measures how a predictor was capable of learning the distribution of the evaluated data by comparing it to a baseline regressor. Every time the error obtained by a regressor is very close to the referred mean predictor, the resulting RRMSE

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

will tend to one. If the evaluated regressor performs worse than the mean predictor, the resulting RRMSE will be greater than one. The RRMSE calculation is given by:

$$\text{RRMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2}},$$

where y_i and $\hat{y}_i$ represent, respectively, the true and the predicted values for the ith instance of the target y. Besides, $\overline{y}$ represents mean value of the target y, and N represents the number of evaluated problem's instances.

The aRRMSE is calculated by averaging the RRMSE of all targets Y. The aRRMSE calculation is given by:

$$\text{aRRMSE}(Y, \hat{Y}) = \frac{1}{d}\sum_{y,\hat{y}}^{Y,\hat{Y}} \text{RRMSE}(y, \hat{y}),$$

where $\hat{Y}$ represents the predictions obtained for Y, and d is the number of targets.

Next, RP is obtained by dividing the aRRMSE of the ST approach by the corresponding error metric of another MTR method, as chosen in the next expression:

$$\text{RP}_{\text{M}} = \frac{\text{aRRMSE}_{\text{ST}}}{\text{aRRMSE}_{\text{M}}},$$

where M represents an MTR method. In our case, we compared the performance of SST, ERC and DSTARS with the ST approach to verify whether the use of an MTR method could reduce the error when forecasting stock indexes. A RP greater than one implies that the chosen MTR method surpassed the ST approach, whereas outcomes smaller than one indicate that ST approach was the better choice.

Last, we employed the Friedman test to verify whether an MTR method was statistically better than the others, using a confidence level, α, of 0.05. Anytime the obtained p-values were smaller than α, we performed the post hoc Nemenyi test to rank the compared MTR methods. We graphically represented the obtained ranks, as proposed by Demšar (2006).

In this representation, MTR approaches connected by a Critical Distance (CD) value are statistically equivalent, with respect to the chosen confidence level.

4. Results and Discussion

We performed all the mentioned MTR methods over the four stock index datasets using the chosen regression techniques. Our goal was to evaluate whether the MTR solutions would achieve satisfactory predictions results, enabling the creation of a decision support system to help in predicting the tendencies in the action market.

The obtained aRRMSE values are reported in Table 2. In this table, the smallest errors obtained per regression technique are in bold, whereas the smallest ones per dataset

are underlined. MORF presents just a single result per dataset since it is a global MTR method.

Table 2. The aRRMSE results obtained from all the compared MTR approaches and regression techniques. The smallest errors obtained per regression technique are in bold, whereas the smallest ones per dataset are underlined.

Period	Regressor	ST*	ERC	SST	MOTC	MTAS	DRS	DSTARS	MORF
1st	XGBoost	0.80783	**0.74113**	0.82046	0.84722	0.81423	0.80784	0.82036	
	RF	0.80327	0.77498	0.76629	0.76594	**0.69102**	0.78617	0.78884	0.97860
	SVM	0.71040	0.68019	**0.67601**	0.68721	0.68332	0.69137	0.71903	
2nd	XGBoost	0.85737	**0.77782**	0.89162	0.87507	0.88812	0.86908	0.89156	
	RF	0.82412	0.80428	0.78429	0.80311	**0.74283**	0.79985	0.79903	0.97678
	SVM	0.75729	**0.74028**	0.74513	0.74475	0.74985	0.74752	0.77315	
3rd	XGBoost	0.66846	**0.60054**	0.65769	0.67880	0.65855	0.66910	0.65762	
	RF	0.67660	0.64860	0.63001	0.65569	**0.55019**	0.64673	0.62814	0.88958
	SVM	0.56950	0.52853	0.51685	0.55356	0.55164	0.52130	**0.50604**	
4th	XGBoost	0.75891	**0.74076**	0.79710	0.78251	0.78598	0.76345	0.75736	
	RF	0.74849	0.73515	0.70996	0.72986	**0.67629**	0.71990	0.72607	0.91661
	SVM	0.68315	0.65846	0.65306	0.68651	0.68301	0.67885	**0.65152**	

* ST refers the traditional machine learning

As it can be observed, SVM obtained the best result overall when compared with the regressors XGBoost and RF, which is also the base of the MORF algorithm. Despite no fine tuning was performed, SVM achieved the smallest error rates for all evaluated datasets. When developing a final product with the best set of MTR tools, further adjustments could be made to improve even more the performance of the mentioned regressor.

Regarding the MTR methods, differently from what found out in [Silva et al. 2018], ERC presented the smallest errors in 5 out 12 times, followed by MTAS (the best 4 times), DSTARS (the best 2 times) and SST (the best once). Though ERC presented the smallest aRRMSE most of times, only in the second period it was the smallest error for the dataset. SST had the smallest error per dataset in the first period.

MTAS was able to significantly decrease the error in relation to ST when considering RF, but as SVM in its single target form had already a smaller error than RF, MTAS was not the the best method for any period.

Nevertheless, the smallest error among all the datasets was obtained by DSTARS in the third observed period. This dataset consists of observations made in the period between 2000 and 2005, and presented prominently smaller aRRMSE than the other periods.

Moreover, it is worth mentioning that DSTARS achieved results very close to the SST ones in almost all cases, which is explained by the nature of this method. In fact, DSTARS is an extension of the original SST idea, being built upon stacking multiple regressors for each target. Potentially DSTARS can mimic the SST behaviour, which explains the very similar results obtained by the two methods. Notwithstanding, the asymp-

totic time complexity of SST is smaller than the DSTARS complexity, so this method ought to present reliable and fast responses in real scenarios. In addition, the global MTR method MORF presented the worst results in all cases.

Besides aRRMSE, the average coefficient of determination was calculated. The results were registered in Table 3.

Table 3. The R^2 results obtained from all the compared MTR approaches and regression techniques. The greatest R^2 obtained per regression technique are in bold, whereas the greatest ones per dataset are underlined.

Dataset	Regressor	ST*	ERC	SST	MOTC	MTAS	DRS	DSTARS	MORF
1st	XGBoost	0.59253	**0.62606**	0.58942	0.58912	0.59535	0.59163	0.58941	0.49240
	RF	0.58404	0.60777	0.60571	0.60493	**0.66596**	0.58402	0.58949	
	SVM	0.63404	0.65683	0.65586	0.65282	**0.65981**	0.63692	0.63951	
2nd	XGBoost	0.59871	**0.62752**	0.59261	0.58779	0.59445	0.60055	0.59262	0.51667
	RF	0.59552	0.59091	0.59341	0.58692	**0.62799**	0.59447	0.57404	
	SVM	0.62621	0.63647	0.63054	0.63517	0.61771	**0.63715**	0.61524	
3rd	XGBoost	0.70714	**0.76928**	0.72479	0.68573	0.72122	0.70615	0.72477	0.62723
	RF	0.71164	0.72848	0.72548	0.69570	**0.80110**	0.70100	0.72047	
	SVM	0.78200	0.81392	0.81450	0.78379	0.80400	0.81899	**0.82092**	
4th	XGBoost	0.59873	**0.60087**	0.59538	0.58670	0.59671	0.59690	0.59545	0.55546
	RF	0.58723	0.59838	0.60882	0.58933	**0.63205**	0.59458	0.60414	
	SVM	0.58427	0.60665	**0.61567**	0.57126	0.60883	0.59139	0.60613	

* ST refers the traditional machine learning

Regarging R^2, MTAS was the best method in 5 out of 12 possibilities. Next, ERC outperformed the other methods 4 times. SST, DRS and DSTARS had the greatest R^2 once, each.

When focusing on the best combination for each dataset, MTAS was the best twice, DRS and DSTARS, once. In addition, RF and SVM were the best 2 times, each. From Table 3, it is also possible to see how MTR can improve the R^2 values: in the first period, for instance, RF in its ST form obtained an R^2 of 0.58404, whereas with MTAS, it increased to 0.66596.

Next, we compared the performance of the local MTR methods against the ST approach. Our objective was to verify whether the exploration of the possible existing inter-dependencies among the stock index outputs led to prediction performance improvements. In this sense, we employed the RP to compare SST, ERC, MOTC, MTAS, DRS and DSTARS to the ST approach. The obtained results are summarized in the Table 4.

As shown in Table 4, except when combined with XGBoost, most of the cases obtained RPs greater than one, meaning that they outperformed the ST approach. Moreover, ERC presented the greatest average RP (1.0529), followed by MTAS (1.0524), SST (1.0298), DRS (1.0214), DSTARS (1.0159) and MOTC (1.0079). In the 3rd period, MTAS coupled with RF had the greatest RP, 1.2298.

Table 4. RP values along the four periods.

Period	Regressor	ERC	SST	MOTC	MTAS	DRS	DSTARS
1st	XGBoost	1.0900	0.9846	0.9535	0.9921	1.0000	0.9847
	RF	1.0365	1.0483	1.0487	1.1624	1.0217	1.0183
	SVM	1.0444	1.0509	1.0337	1.0396	1.0275	0.9880
2nd	XGBoost	1.1023	0.9616	0.9798	0.9654	0.9865	0.9616
	RF	1.0247	1.0508	1.0262	1.1094	1.0303	1.0314
	SVM	1.0230	1.0163	1.0168	1.0099	1.0131	0.9795
3rd	XGBoost	1.1131	1.0164	0.9848	1.0150	0.9990	1.0165
	RF	1.0432	1.0739	1.0319	1.2298	1.0462	1.0771
	SVM	1.0775	1.1019	1.0288	1.0324	1.0925	1.1254
4th	XGBoost	1.0245	0.9521	0.9698	0.9656	0.9940	0.9519
	RF	1.0181	1.0543	1.0255	1.1068	1.0397	1.0296
	SVM	1.0375	1.0461	0.9951	1.0002	1.0063	1.0269

Table 5 presents the percentage of error reduction obtained by the MTR when compared to the ST strategy.

Table 5. Percentage of error reduction achieved by the MTR methods over the ST strategy, when using XGBoost, RF and SVM as regressors. The best gains in prediction performance per dataset are in bold, whereas the worst ones are in italic, and the minus signal indicates an error reduction.

Period	Target	XGBoost						RF						SVM					
		SST	MTAS	MOTC	ERC	DSTARS	DRS	SST	MTAS	MOTC	ERC	DSTARS	DRS	SST	MTAS	MOTC	ERC	DSTARS	DRS
1st	ARR	0.49	-0.01	11.07	1.60	0.49	0.15	-11.00	-24.02	1.33	-2.28	-5.39	-11.91	-11.66	-5.96	-5.71	-8.76	-10.19	-8.67
	α	-2.31	-4.04	*16.06*	-5.73	-2.41	-1.30	-3.38	**-25.49**	-11.30	-0.59	2.81	3.57	-11.82	-13.04	-5.78	-5.64	-11.23	-15.31
	β	-0.07	-2.07	10.10	-12.86	-0.07	0.03	-7.19	-20.86	-9.16	0.84	-2.07	-9.97	-12.65	-8.74	-9.87	-8.94	-9.14	-7.60
	TR	-0.73	-1.06	-18.15	-12.84	-0.73	-0.43	-7.11	-11.63	-1.12	-2.89	-7.44	-8.38	-4.77	-2.25	-2.53	-6.00	-3.99	2.15
	AWR	1.82	1.82	3.83	-8.00	1.82	0.00	-5.10	-5.67	-6.54	-8.59	-8.15	-0.12	-2.09	-9.99	-3.83	-3.72	2.45	-0.31
	RWR	3.75	2.49	-0.01	-8.57	3.75	0.84	-1.47	-5.32	-1.45	-2.41	2.70	4.38	-4.52	-1.12	-2.97	-2.85	3.20	0.00
2nd	ARR	0.97	-0.31	-2.14	-18.53	0.97	0.05	-11.06	-14.73	-3.01	-0.44	-6.07	-15.54	-7.40	-8.82	-5.40	-5.58	-6.01	-0.47
	α	-1.20	-1.11	1.29	-9.45	-1.20	-1.37	-3.79	-13.22	-1.74	0.63	-4.08	-6.07	2.39	-2.99	1.13	-1.97	4.84	0.00
	β	-3.12	-5.36	4.36	-3.81	-3.15	-0.08	-7.58	-15.78	-4.54	-4.01	-6.45	-7.05	-10.68	-8.98	-5.99	-5.90	-6.77	-11.82
	TR	5.75	5.69	-15.81	**-15.90**	5.75	4.88	-5.04	-12.44	-6.25	-3.75	-8.30	-2.34	-3.86	-2.85	-2.75	-2.42	-3.52	-4.29
	AWR	11.15	11.15	*21.22*	-1.19	11.15	0.00	-1.80	-5.87	2.58	-0.69	-0.93	-1.04	-0.73	-6.59	-2.70	-2.09	1.97	0.60
	RWR	0.65	0.65	6.77	-4.98	0.65	0.00	-1.44	0.86	0.93	0.32	4.87	2.93	3.23	14.54	3.02	1.54	6.84	2.57
3rd	ARR	-2.20	-2.57	1.91	-3.91	-2.20	-0.29	-12.33	**-24.85**	-8.22	-8.60	-12.94	-8.42	-15.75	-18.10	-0.88	-9.71	-14.51	-23.40
	α	0.34	-0.10	5.58	1.09	0.34	0.00	-12.51	-22.66	-13.01	-1.82	-10.97	-14.65	-15.34	-18.33	-1.26	-9.15	-14.42	-22.61
	β	-0.78	-1.05	5.15	-2.66	-0.84	0.00	-4.78	-20.38	-3.82	-0.81	-6.12	-4.38	-1.61	6.40	3.25	-1.41	-3.43	3.88
	TR	0.00	0.08	*14.50*	-6.44	0.00	0.05	-3.99	-18.87	-4.17	1.29	-3.39	-0.55	-1.61	2.10	4.33	-4.37	-4.24	-1.82
	AWR	-4.89	-4.89	-15.73	-19.53	-4.89	1.16	-13.52	-21.58	-7.85	-11.48	-13.76	-3.32	-3.75	-3.77	-10.16	-5.37	-6.73	5.10
	RWR	-5.13	-3.07	1.59	-19.10	-5.13	0.00	-5.06	-10.34	4.12	-0.50	-7.14	-5.99	-11.16	-2.30	-9.10	-7.67	-15.27	-8.71
4th	ARR	-0.67	-1.18	6.02	-3.00	-0.65	0.03	-9.23	-12.55	-5.86	-1.77	-4.83	-6.85	-3.18	-1.90	0.68	-3.04	-3.36	3.89
	α	7.71	4.02	3.53	0.46	7.83	4.20	-11.34	-14.00	-6.69	-0.76	-12.04	-11.72	-5.78	-9.47	-0.05	-3.27	-5.84	-0.63
	β	2.69	2.50	2.83	1.16	2.65	-0.32	-14.16	**-29.94**	-7.38	-5.53	-14.90	-12.72	-10.99	-14.19	-1.15	-8.30	-9.43	-17.39
	TR	0.68	-1.71	4.18	5.67	0.70	-0.16	-11.37	-26.55	-7.53	-4.84	-12.21	-9.18	-10.99	-14.83	-1.09	-8.65	-8.58	-14.94
	AWR	9.60	9.60	-5.32	-9.06	9.60	0.00	-0.11	8.34	2.09	-0.36	0.69	-0.07	1.82	*14.14*	4.82	2.35	5.58	3.62
	RWR	2.21	1.24	5.70	-0.10	2.21	0.85	-4.02	-5.35	-1.88	-3.14	-1.35	-1.85	-6.31	-7.81	-3.93	-5.34	-9.00	1.43
	Average	1.11	0.45	2.85	-6.49	1.11	0.35	-7.01	-14.70	-4.19	-2.59	-5.73	-5.47	-6.22	-5.20	-2.41	-4.84	-5.03	-4.78

As observed, with exception of XGBoost, there were gains in almost all cases. The greatest gain in prediction performance was obtained by MTAS with RF when predicting the β in the 4th period (29.94% of error reduction). In average, the greatest gains were obtained with XGBoost and ERC (6.49%), RF with MTAS (14.7%) and SST (7.01%) and SVM with SST (6.22%).

Notwithstanding, there were some cases where the MTR methods were surpassed by the ST strategy. At the worst case, the MTR methods were 21.22% worse than ST (MOTC with XGBoost for target AWR, 2nd period).

We also performed statistical tests to verify the possible significant statistical superiority of some MTR method in the evaluated problems. We considered multiple comparison scenarios, using the aRRMSE metric, to evaluate the performance of the MTR methods.

Firstly, we considered the aRRMSE results obtained by all MTR approaches and all evaluated regressors. Figure 4 presents the Nemenyi results observed in this scenario.

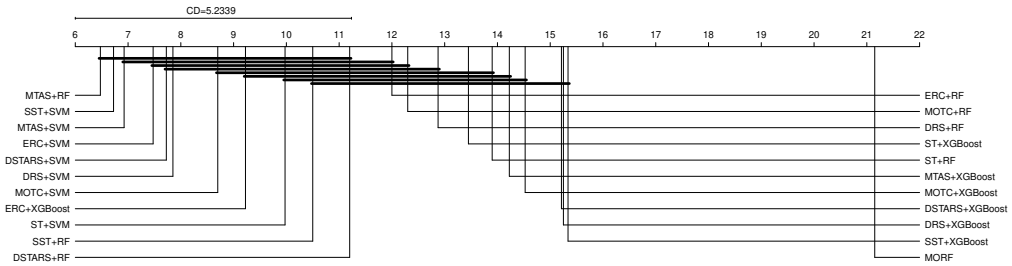


Figure 4. Nemenyi post hoc test aRRMSE results when considering all the MTR approaches and regression techniques.

In this figure, combinations that are connected by a CD present no statistical difference at α. Besides, the closer to 1, the better the performance.

The first positions were filled by the local methods, mainly when coupled with the SVM regressor. MTAS appeared in the first position coupled with RF, being followed by SST, MTAS, ERC, DSTARS, DRS and MOTC coupled with SVM. After that, the rank shows ERC with XGBoost, ST of SVM, SST and DSTARS with RF, finishing the first group. There was no statistically significant difference among the elements of this first group. ST, DRS and MOTC appeared just once in this group, whereas, MTAS, SST, ERC, DSTARS appeared twice, each.

From this figure we concluded that XGBoost, that appeared in the first group only for ERC and in the last group for the other methods, with its default R configuration, was the worst regressor to explain the problem, either in ST or with MTR methods. MORF was significantly worse than the other approaches.

As many MTR methods were evaluated, we analysed each regressor separately. Firstly we compared the local MTR approaches when combined with XGBoost, as shown in Figure 5.

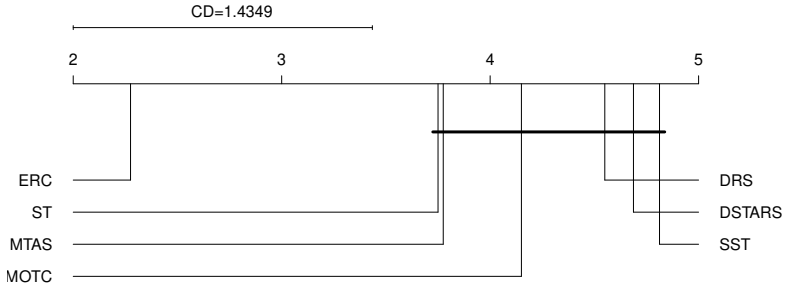


Figure 5. Nemenyi post hoc test aRRMSE results when considering all the local MTR approaches along with XGBoost.

ERC is the first in the rank, isolated from all the other techniques. It justifies its appearance in the first group of the previous figure. The other methods appear in the last group, together with ST.

Figure 6 presents the obtained test results using the regressor RF.

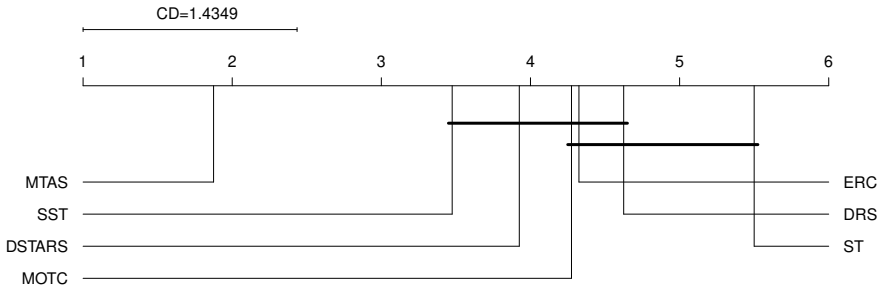


Figure 6. Nemenyi post hoc test aRRMSE results when considering all the local MTR approaches along with RF.

For this regressor, MTAS appeared isolated in the first position, presenting the best result among the compared methods. After that, SST, DSTARS, MOTC, ERC and DRS come in the same group with no statistical difference. Only the last group englobes ST.

Next, Figure 7 presents the obtained test results using SVM.

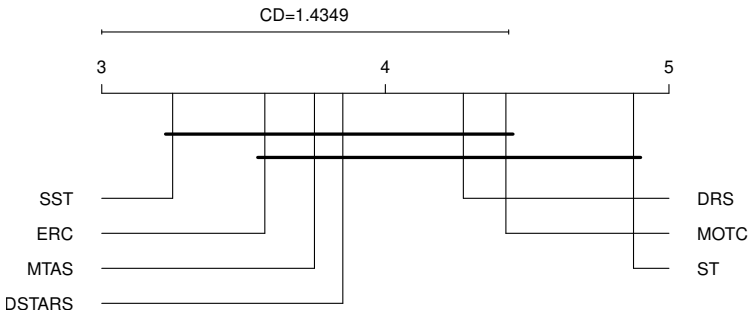


Figure 7. Nemenyi post hoc test aRRMSE results when considering all the local MTR approaches along with SVM.

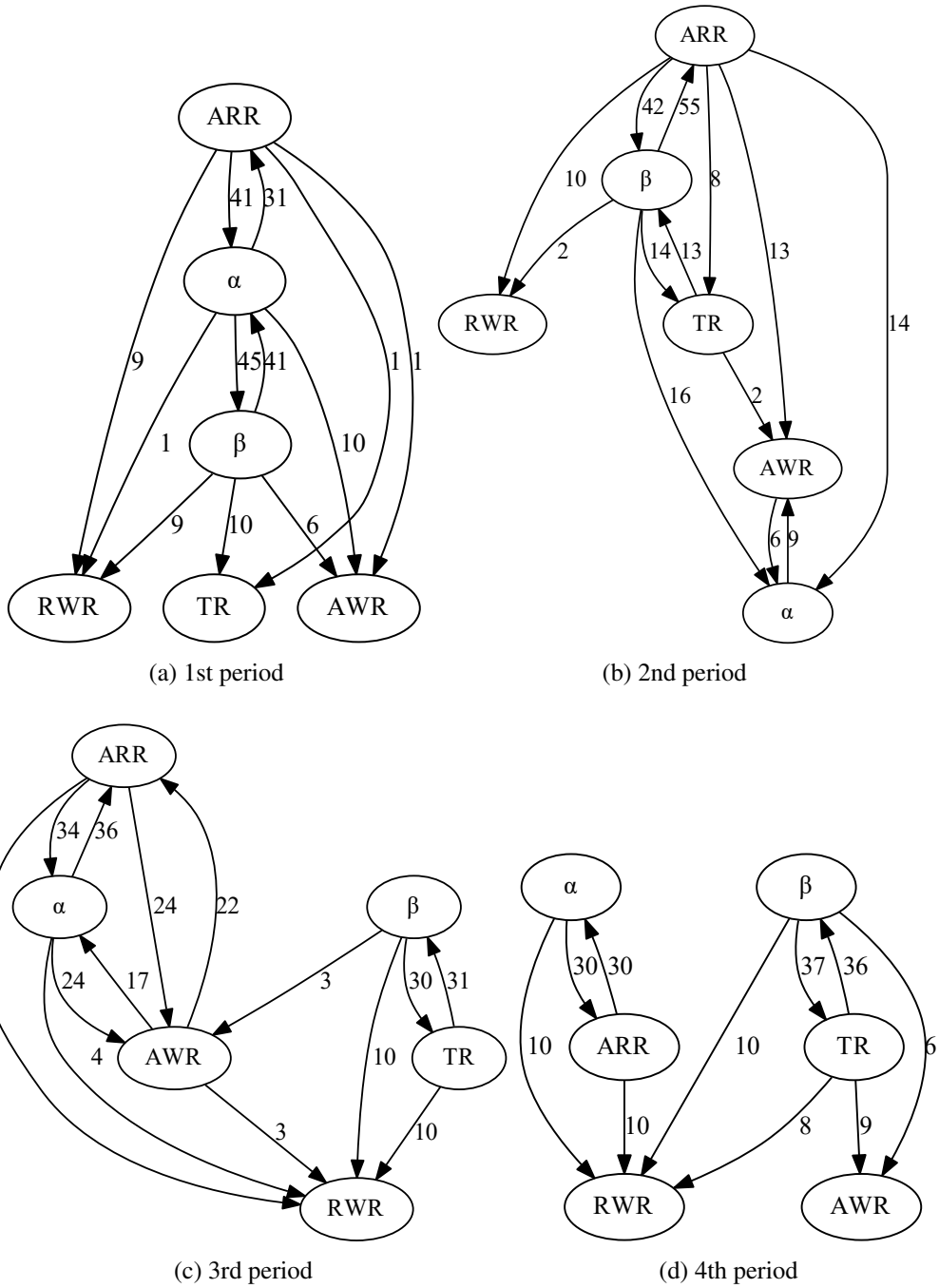
SST appeared in the first position, being statistically equivalent to ERC, MTAS, DSTARS, DRS and MOTC. Only ST did not figure in the first group, showing the superiority of MTR over ST. This reinforces the evidences of statistical dependencies among the stock index output variables. In fact, the obtained results show that evaluated problems are MTR tasks, and thus, they must be modelled in this way to achieve superior performance.

To sum up, the employment of MTR techniques to model the prediction of multiple stock market variables, indeed, obtained superior results than considering each output separately. Among the evaluated MTR methods, SST with SVM would be the best choice for being a simple yet effective solution. However, if more prominent gains are desired, MTAS should be considered.

In our experiments, SVM and RF outperformed XGBoost when both techniques were implemented without further adjustments. In this sense, to compose a final solution, the regressors that are base learners of the MTR local methods might be tunned to achieve even better results.

With enough training instances, a software to offer stock indexes estimates can be created. Besides, as the time evolves and the real outcomes come to knowledge, the prediction models can be updated. A further venue for research would be the comparison of two model update strategies: the selection of training instances with a sliding window of time or aggregating all new cases through in a single knowledge database.

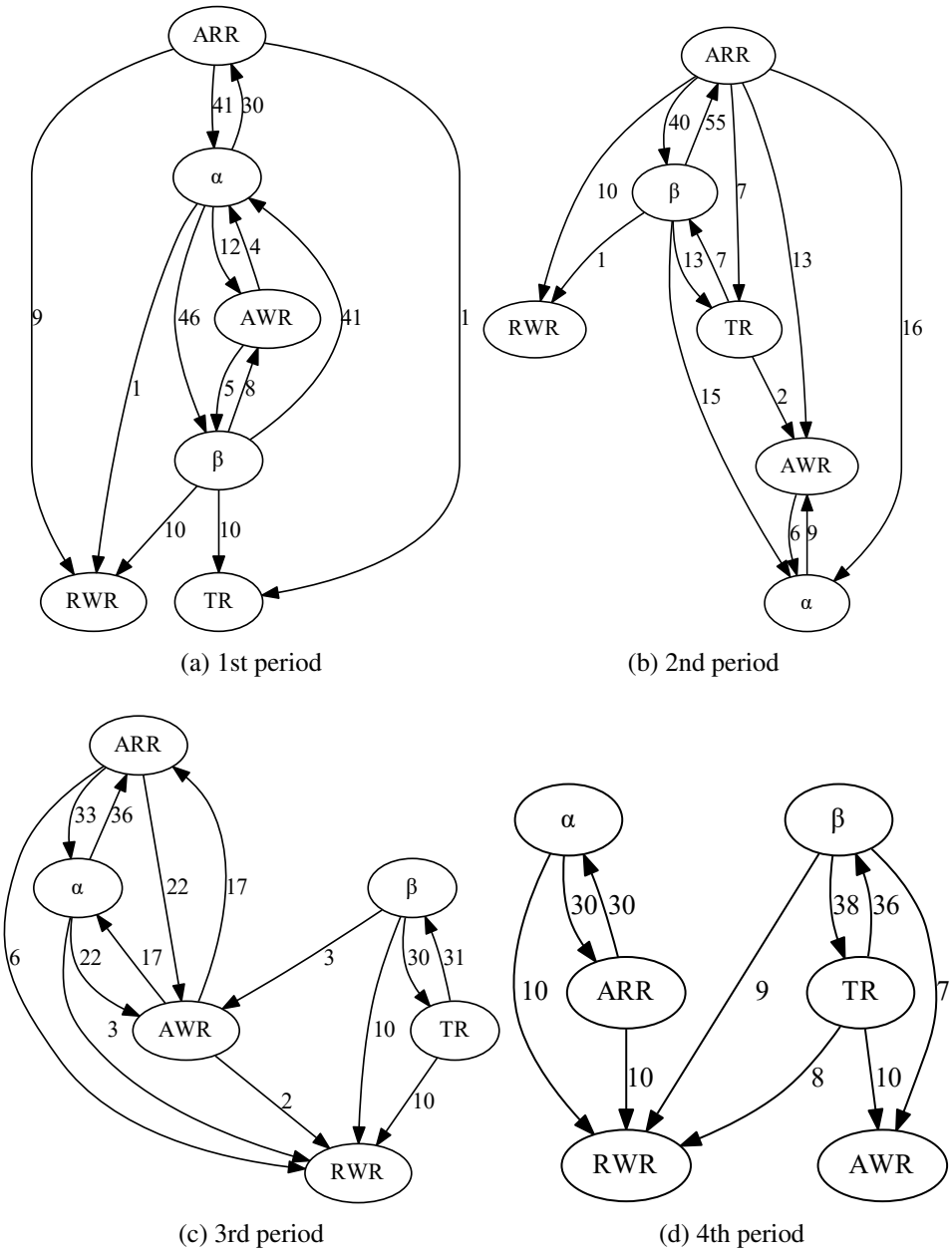
Although MOTC was not the best regressor, for SVM and RF it had an average performance. Since it provides graphs that shows the targets dependency during model generation, we analysed them in order to improve the interpretation. The graphs are in Figures 8 and 9.



(a) 1st period (b) 2nd period

(c) 3rd period (d) 4th period

Figure 8. MOTC's target graphs for RF.



(a) 1st period (b) 2nd period

(c) 3rd period (d) 4th period

Figure 9. MOTC's target graphs for SVM.

As observed, the structure for both regressors are very similar, but every period assumes a particular structure, which follows the difference in the results for each period and ratifies that the behaviour of the data changes over time. To exemplify this, consider the relation of α and ARR. In the first, third and fourth periods, α and ARR explain each other mutually. However, in the second period only ARR explains α directly, with a lower weight when compared to the other periods.

α also explains and is explained by AWR in most of the periods, but in the fourth, they are not correlated. TR and β also follow a mutual explanation, except for the first period, when TR does not explain β. This change in relation over the years shows that

the data pattern can change significantly with time, and thus it is necessary to update the models after some time period.

However, some targets are connected or disconnected in a fixed way. RWR never explains other targets, but is explained by others, specially β and ARR. Except for a weak connection in the third period, AWR and RWR are never connected.

5. Conclusion

Different approaches have been used in stock market to predict economic metrics, including the usage of computational intelligence and machine learning (ML) algorithms. However, the benefits of Multi-Target Regression (MTR) were underexplored. In this paper, we proposed the use of MTR for improving the prediction of stock market indicators. Moreover, it plays like a kernel of a decision support tool to aid the investor's decisions with superior results when compared to traditional solutions based on ML.

Our contribution to stock target prediction outperformed the traditional single target methods in a real-life scenario using mainly Random Forest (RF), Extreme Gradient Boosting (XGBoost) and Support Vector Machine (SVM) learn-based algorithms. In the best cases, the MTR reached an improvement over Single Target Regression, considering the average error of all targets in all periods, of 14.70% (Multi-target Augmented Stacking (MTAS) with RF), 7.01% (Stacked Single Target (SST) with RF), 6.49% (Ensemble of Regressor Chains (ERC) with XGBoost) and 6.22% (SST with SVM). In this way, we affirm that the use of Multi-target improves the capabilities of stock market predictions taking advantage from their inter-correlations to built a predictor.

Some limitations of this work are the restricted amount of weights combinations in the datasets and the not implementation of the methods in a system, which could be done in future work. As a continuity of this work, we aim at dealing with online updating of the model owing to check concept-drifts on a data stream scenario, focusing on standard behaviour changing over the time.

6. Acknowledgement

We would like to thank the Brazilian agencies CAPES (*Coordenação de Aperfeiçoamento de Pessoal de Nível Superior*), CNPQ (*Conselho Nacional de Desenvolvimento Científico e Tecnológico* and FAPESP (*Fundação de Amparo à Pesquisa do Estado de São Paulo*) for scholarships.

References

Aho, T., Zenko, B., Dzeroski, S., and Elomaa, T. (2012). Multi-Target Regression with Rule Ensembles. *J. Mach. Learn. Res.*, 13:2367–2407.

Atsalakis, G. S. and Valavanis, K. P. (2009). Surveying stock market forecasting techniques – Part II: Soft computing methods. *Expert Systems with Applications*, 36(3, Part 2):5932 – 5941.

Bahrammirzaee, A. (2010). A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems. *Neural Computing and Applications*, 19(8):1165–1195.

Basu, S. (1977). Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The Journal of Finance*, 32(3):663–682.

Ben-Hur, A. and Weston, J. (2010). A user's guide to support vector machines. *Data Mining Techniques for the Life Sciences*, pages 223–239.

Blockeel, H., Raedt, L. D., and Ramon, J. (1998). Top-down induction of clustering trees. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 55–63. Morgan Kaufmann Publishers Inc.

Borchani, H., Varando, G., Bielza, C., and Larrañaga, P. (2015). A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):216–233.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Bruni, R. (2017). Stock market index data and indicators for day trading as a binary classification problem. *Data in Brief*, 10:569 – 575.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM.

Grömping, U. (2009). Variable importance assessment in regression: linear regression versus random forest. *The American Statistician*, 63(4):308–319.

Hu, Z., Bao, Y., Chiong, R., and Xiong, T. (2017). Profit guided or statistical error guided? A study of stock index forecasting using support vector regression. *Journal of Systems Science and Complexity*, 30(6):1425–1442.

Ihaka, R. and Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314.

Khaidem, L., Saha, S., and Dey, S. R. (2016). Predicting the direction of stock market prices using random forest. *CoRR*, abs/1605.00003.

Kocev, D., Vens, C., Struyf, J., and Džeroski, S. (2007). Ensembles of multi-objective decision trees. In *European Conference on Machine Learning*, pages 624–631. Springer.

Kocev, D., Vens, C., Struyf, J., and Džeroski, S. (2013). Tree ensembles for predicting structured outputs. *Pattern Recognition*, 46(3):817–833.

Liu, Y.-C. and Yeh, I.-C. (2017). Using mixture design and neural networks to build stock selection decision support systems. *Neural Computing and Applications*, 28(3):521–535.

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

Mastelini, S. M., Santana, E. J., Cerri, R., and Barbon, S. (2017). DSTARS: A multi-target deep structure for tracking asynchronous regressor stack. In *2017 Brazilian Conference on Intelligent Systems (BRACIS)*, pages 19–24. IEEE.

Melki, G., Cano, A., Kecman, V., and Ventura, S. (2017). Multi-target support vector regression via correlation regressor chains. *Information Sciences*, 415:53–69.

Montgomery, D. and Runger, G. (2014). *Applied Statistics and Probability for Engineers*. John Wiley & Sons, Incorporated, 6th edition.

Moyano, J. M., Gibaja, E. L., and Ventura, S. (2017). An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains. In *Evolutionary Computation (CEC), 2017 IEEE Congress on*, pages 2015–2021. IEEE.

Patel, J., Shah, S., Thakkar, P., and Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, 42(1):259 – 268.

Roko, I. and Gilli, M. (2008). Using economic and financial information for stock selection. *Computational Management Science*, 5(4):317–335.

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

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

Silva, J. A. P. R. d., Santana, E. J., Mastelini, S. M., and Barbon Jr., S. (2018). Stock Portfolio Prediction by Multi-Target Decision Support. In *XIV Brazilian Symposium on Information Systems*. Brazilian Computer Society (SBC).

Spyromitros-Xioufis, E., Tsoumakas, G., Groves, W., and Vlahavas, I. (2016). Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98.

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

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241–259.

Xiong, T., Bao, Y., and Hu, Z. (2014). Multiple-output support vector regression with a firefly algorithm for interval-valued stock price index forecasting. *Knowledge-Based Systems*, 55:87–100.