Neurocomputing 127 (2014) 52–64

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

MetaStream: A meta-learning based method for periodic algorithm selection in time-changing data

André Luis Debiaso Rossi [a,*], André Carlos Ponce de Leon Ferreira de Carvalho [a], Carlos Soares [b], Bruno Feres de Souza [a]

[a] Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, Brazil
[b] INESC TEC, Faculdade de Engenharia da Universidade do Porto, Universidade do Porto, Porto, Portugal

ARTICLE INFO

Article history:
Received 10 January 2013
Received in revised form
13 May 2013
Accepted 18 May 2013
Available online 17 August 2013

Keywords:
Algorithm selection
Meta-learning
Data streams

ABSTRACT

Dynamic real-world applications that generate data continuously have introduced new challenges for the machine learning community, since the concepts to be learned are likely to change over time. In such scenarios, an appropriate model at a time point may rapidly become obsolete, requiring updating or replacement. As there are several learning algorithms available, choosing one whose bias suits the current data best is not a trivial task. In this paper, we present a meta-learning based method for periodic algorithm selection in time-changing environments, named MetaStream. It works by mapping the characteristics extracted from the past and incoming data to the performance of regression models in order to choose between single learning algorithms or their combination. Experimental results for two real regression problems showed that MetaStream is able to improve the general performance of the learning system compared to a baseline method and an ensemble-based approach.

1. Introduction

Many machine learning (ML) algorithms that deal with supervised problems are currently available [1]. In ML applications, users tend to test a small set of these algorithms hoping that the assumptions made by one or more of them are appropriate for the characteristics of the data under analysis. Traditional learning algorithms are designed to handle problems that have a limited number of examples available, which are generated according to some stationary probability distribution [2]. However, many dynamic real-world environments generate data continuously and with varying arrival rates, which are known as data streams [3,4]. A few examples of sources of data streams are computer networks, sensor networks, web applications and transport monitoring. In these applications, it is likely that the underlying distribution that generates data naturally changes over time [5]. As a result, the assumptions embedded in the learning algorithm that are suitable for the current data might not be appropriate for future data.

Methods able to explicitly detect changes can be employed to decide when a model or a learning algorithm should be updated or replaced. The most common methods for monitoring model performance are the Page-Hinkley Test and the Statistical Process Control [2]. When the difference between the performance of the current model and the best model learned so far is larger than a given threshold, these tests raise an alarm [2]. However, they have the drawback of not detecting changes that do not cause a reduction of model performance. Therefore, it is not possible to identify other algorithms that have become more appropriate than the current algorithm for the data that is being generated. An alternative to these tests are the change detectors based on data characteristics [6]. Nevertheless, this approach is effective only on one-dimensional data and does not trivially extend to higher dimensions [7].

The management of decision models without explicitly detecting changes is an alternative method for changing environments. Dynamic ensembles of models are able to manage different biases by assigning a higher weight to the most suitable model and decreasing the importance of the others over time [8,5,9]. However, given the number of models, ensembles may be computationally expensive and difficult to interpret [10]. An alternative approach to ensembles is to select only the most suitable model or algorithm. Experimental results from some studies [11–14] suggest that ensembles perform better than individual models. On the other hand, other studies show that an ensemble performed comparably to the best of the individual model, but not better [15,16]. However, identifying the best algorithm is a very challenging task. Meta-learning based methods have presented promising results on selecting the most appropriate model for a given data

* Corresponding author. Tel.: +55 16 3373 8161; fax: +55 16 3373 9633.
E-mail addresses: alrossi@icmc.usp.br, andrerossidc@gmail.com (A.L.D. Rossi), andre@icmc.usp.br (A.C.P.d.L.F. de Carvalho), csoares@fe.up.pt (C. Soares), bferes@icmc.usp.br (B.F. de Souza).

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.05.048

set according to the characteristics extracted from this data set [17,18].

In this paper, we present a meta-learning based method, referred to as MetaStream, that periodically and automatically attempts to select the best regression algorithm for time-changing environments. For such an environment, MetaStream regularly induces a meta-classifier that is able to map the characteristics extracted from past and incoming data to the performance of regression models on these data. The first ideas of the MetaStream method were proposed in [19]. The present study extends that paper in three main directions. Firstly, instead of restricting MetaStream to pairs of algorithms, we extended this method to select an algorithm from a set of more than two regressors (referred to as multi-regressors). Secondly, in addition to selecting only a single algorithm from a pool of algorithms, we explicitly consider the possibility of selecting the combination of these regressors. In this case, the output is the average of their predictions. Thirdly, a more comprehensive experimental evaluation is performed, including new data and a more detailed analysis of the experimental results. These results show that MetaStream is able to predict the most suitable algorithm for incoming examples more accurately than a baseline method and improve the general performance of the learning system compared to an ensemble-based approach.

The next sections of this paper are organized as follows. Section 2 gives a brief introduction to meta-learning and presents relevant similar studies. Section 3 describes the MetaStream method. Section 4 details the experimental evaluation of this method and analyzes the experimental results. Finally, Section 5 presents the main conclusions from this study and points out future research directions.

2. Meta-learning and algorithm selection

According to the No Free Lunch theorem for machine learning [20], no algorithm performs better than any other when their performances are averaged uniformly over all possible problems. In time-changing environments, an algorithm that is suitable at the current moment may become inappropriate when the data characteristics change [2]. Meta-learning can provide automatic and systematic guidance on algorithm selection based on the knowledge acquired from the application of a set of algorithms on different problems. More generally, meta-learning is concerned with understanding which are the conditions that define whether a learning system is inadequate for a particular task in order to improve its performance in future applications [18,21]. Meta-learning is different from base-learning (traditional learning) in the level of adaptation. In the base-level, the bias is fixed a priori (by the choice of the algorithm and the values for its parameters) whereas in meta-learning the most suitable bias is dynamically defined according to the characteristics of the data [22].

Several studies have investigated the use of meta-learning to select the most appropriate model in data changing environments, such as [23–28]. Among them, the method proposed in [23] is the most related to MetaStream. It uses meta-learning to select an algorithm (and its parameterization) from a pool of learning algorithms. In [23], the data are assumed to arrive in batches of examples, and, for each new batch, the proposed method creates a meta-example. A set of these meta-examples is then used to induce a meta-learner, which guides the search for the best base-learning algorithm and the best set of examples at a time point. The meta-learner is induced whenever a new meta-example becomes available, i.e., after each batch. The main difference between [23] and our study is the set of characteristics that constitute a meta-example. In [23], a meta-example consists of information about the learning process related to the batches of examples, such as the number of batches used for training, the most successful algorithm on the previous batch and the most successful learner over all the batches seen so far. Here, we extract characteristics directly from the raw data used in the base-learning. We believe that these characteristics provide a more detailed description of the problem under analysis than the high level characteristics used in [23].

Another meta-learning approach for a non-stationary environment is presented in [24]. In this approach, a system for concept drift (represented as a rise in the classification error rate) detection in data streams dynamically adjusts the size of a training window according to the severity of the concept drift. A meta-learning module calculates the size of the next training window and the number of validation examples to be classified by the new model based on information-theoretic and statistical measures [18], gathered from the base-learning process.

One of the earliest studies employing a meta-learning based approach in the data streams scenario was presented in [25]. In this study, the author defines two types of attributes: predictive and contextual attributes. Predictive attributes are correlated with class labels, and, thus, are used to induce predictive models. Contextual attributes are employed to identify the current concept associated with the data, and systematic changes in their values might indicate a concept drift. Examples of context-defining factors are climate or season in weather prediction and speaker nationality and sex in speech processing. A meta-learning based method was proposed to identify attributes that might provide contextual clues. As the system processes the stream of input data, it stores concept descriptions that seem to be stable over some period. These stable concepts might be informed with the goal of adapting the systems faster when the observed domain changes to a context previously found. When a new example arrives, only examples that have the same context of this new example are used as training data.

A common use of meta-learning in changing environments is to reuse models learned from the past data that are similar to current data. This is the case in [26], where the authors introduced the method *Splice*, which uses hidden contexts to identify subsets of data that are associated with stable concepts. Since contexts can recur, several disjoint intervals from the data may be associated with the same concept. The identified intervals are clustered by Splice, considering that they are similar if they are well classified by the same concept. Next, a batch decision tree induction algorithm induces models for each of these concepts. The problem of recurring concepts is also investigated in [27]. Whenever a concept drift is detected, the base model and its associated meta-model are stored in a pool for possible future use. Every time the performance of the current base model drops below a given threshold, the meta-models in the pool are retrieved and used to predict the most appropriate base model for the current data. A model is reused when the percentage of votes of its meta-model exceeds a given threshold; otherwise a new classifier is induced. The same problem of reusing models is studied in [28], where an ensemble is built based on the context information. Similar to [27], this approach maintains a pool of classifiers learned from the previous concepts using an incremental algorithm. When a concept is identified as recurrent, a weighting function is used to determine which classifiers should be retrieved from the pool and assigns weights to them based on their estimated error and context information.

3. The MetaStream method

MetaStream is a meta-learning based method that periodically selects either one regression algorithm or the combination of

regression algorithms in non-stationary environments. This selection occurs for a set of pre-defined regression algorithms and aims at selecting the most adequate algorithm or combination of algorithms for the prediction of a target value for the incoming data. This method comprises a base-level and a meta-level. At the base-level, a set of learning algorithms uses training data to induce regression models, which are evaluated on incoming data. At the meta-level, a ML algorithm (i.e., a meta-learner) is used to relate the characteristics of the base-level data to the performance of the regressors on these data. Then, the induced meta-model is employed to predict the most appropriate base learning algorithm for new arriving data. We believe that MetaStream is a promising approach mainly when the average performance of the algorithms is similar for a large amount of data but different when analyzed at a fine enough level of granularity. In this study, we use batch learning algorithms at the base-level, once, as pointed in [29], there are just a few incremental regression algorithms available, such as [30–32]. However, it is important to stress that MetaStream is independent of the learning algorithm used in each level. Thus, even incremental algorithms could be used reducing the computational time in the base-level. Because regressors are periodically built from scratch over time, our aim is to predict the most appropriate regression algorithm, not the model (as in [27], for instance), for new incoming examples. In the next subsections, we describe the base-level and the meta-level of MetaStream in more detail.

3.1. Base-level

The base-level receives a stream of examples from the problem under analysis. Each example $e=(\mathbf{x},y_b)$ consists of a tuple of p predictive attribute values $\mathbf{x}=(x_1,\ldots,x_p)$ and a target attribute $y_b\in\mathbb{R}$. The learning and evaluation processes of the regression models are performed using the interleaved test-then-train or prequential method [33] with a sliding window. In this method, each example can be used to test the model before being used for training, assuring that the model is always tested on unseen data. The sliding window is a well-known forgetting mechanism in environments where data arrive continuously, discarding old examples that do not fit in the memory or are out-of-date. Fig. 1 illustrates a flow of data at the base-level. At time t, the current sliding window contains the ω_b which are the most recent examples (training data), whose true target values are known. These examples are used to induce a regression model, which is applied to predict the target value of a batch of examples arriving from time $t+\eta_b+1$ up to $t+\eta_b+\lambda_b$, where $\eta_b\in\mathbb{N}$ is a delay (in the number of examples) between obtaining an example and observing its true target value, and $\lambda_b\in\mathbb{N}^*$ is the size of the evaluation window, which contains examples whose attribute values are known in advance, but not the true target, which are to be predicted by the induced model. The predictive attributes of the examples in the interval from $t+1$ to $t+\eta_b$ were also known at time t, but their true target are still unknown. The step of the training and evaluation windows is equal to the size of the latter, i.e., the window slides for each λ_b examples.

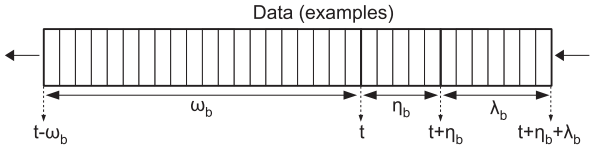

Fig. 1. Flow of data in the base-level using a sliding window.

3.2. Meta-level

The meta-learning phase takes place in the second level of MetaStream and can be divided into three main steps: (i) the generation of the meta-data; (ii) the induction of a meta-learner, and (iii) the deployment of the meta-learner for algorithm selection. These three steps are illustrated by Fig. 2. In this illustrative diagram, we consider a training window of $\omega_b=100$ examples, an evaluation window of $\lambda_b=1$, no delay for observing the target, $\eta_b=0$, and that 500 examples have already been processed. Thus, the current sliding window contains examples in the interval from 501 to 600. In the following, we describe how each step of the meta-level of MetaStream works.

3.2.1. Meta-data

The term meta-data refers to the data extracted from the learning process. These data can be used to improve the performance of the learning mechanism itself [18]. In this context, meta-data in a data stream scenario is a stream of meta-examples, where each meta-example $z=(\mathbf{x},y_m)$ is a tuple of q meta-attribute values $\mathbf{x}=(x_1,\ldots,x_q)$ and a label y_m. The label of a meta-example represents either the best learning algorithm on the set of base-data associated with the meta-example; a set of algorithms that are not considerably different among them; or a set of algorithms that generated an ensemble of models that is better than any of the models separately. From now on, we will refer to the predictive meta-attributes simply as meta-features. The meta-features of a meta-example are generated by extracting characteristics from a set of base-level examples. It should be noted that this set of examples includes both the training and the algorithm selection windows. The latter (henceforth referred to simply as selection window) is related to the test set at base-level (evaluation window), but it is not necessarily the same. It contains the batch of new γ examples, for which MetaStream selects an algorithm (or a combination of the algorithms). The size of this batch of examples may be smaller than λ_b, which allows the selection of a different algorithm for each subset of λ_b, or larger than λ_b, which allows the selection of an algorithm for a set of base-level examples whose target will be predicted. For the sake of simplicity, we restricted the size of γ to be multiple of λ_b, $\lambda_b\equiv 0\ (\mathrm{mod}\ \gamma)$ when γ is smaller than λ_b. The top part of Fig. 2 illustrates the

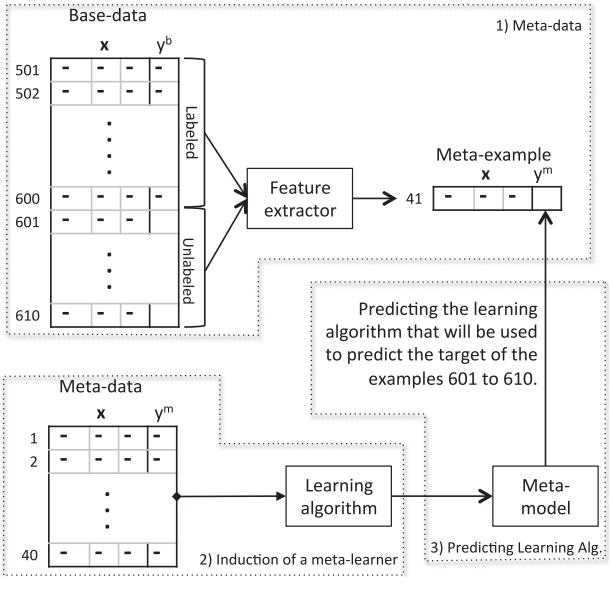

Fig. 2. An illustrative example of the generation of one test meta-example and the prediction of its class using a meta-learner.

Table 1
Meta-features: measures used to characterize the data from the training (Train.) and selection (Sel.) windows.

Measure	Train.	Sel.
Average, variance, minimum, maximum and median of continuous attributes	✓	✓
Average, variance, minimum, maximum and median of the target	✓	
Correlation between numeric attributes		✓
Correlation of numeric attributes to the target	✓	
Possibility of existence of outliers in numeric attributes		✓
Possibility of existence of outliers in the target	✓	
Dispersion gain	✓	
Skewness of numeric attributes		✓
Kurtosis of numeric attributes		✓

process of characterization of the base-data, creating a new unlabeled meta-example. The selection window with $\gamma = 10$ contains the examples 601–610.

In this study, we employed the measures listed in Table 1, which are commonly used in the meta-learning research area [34,18] to extract data characteristics from the base examples in the training (Train.) and selection (Sel.) windows. From the data in the training window, it is possible to extract supervised characteristics, i.e., information about the relationship between predictive and target attributes, because the target was already observed for these examples. On the other hand, only unsupervised characteristics can be extracted from examples in the selection window, since the target is unknown. Additionally, the characterization of the data from each attribute is possible here, because the same set of attributes is used on the data associated with each meta-example. In traditional meta-learning studies [35,36], on the other hand, data sets with different sets of attributes (e.g., from the marketing and from the medical domains) are considered and, thus, the measures used for the meta-features typically have to be aggregated (e.g., averaged). The measures listed in Table 1 generate meta-features that are referred to in the meta-learning literature as general, statistical and information-theoretic [18]. The confidence that the extracted characteristics correctly represent the data is associated with the number of examples that are used. Thus, the larger the number of examples, the higher the reliability on the characteristics that describe the data.

When the target of the examples in the selection window are made available, the corresponding meta-example is labeled according to the average performance of the base level models (or ensemble) on these base-examples. The reliability of both the characteristics extracted from the data and the estimated predictive performance of the induced models depends on the selection window size (γ). The smaller the window, the lower the confidence on both the characteristics extracted from the data in this window and the predictive performance of the algorithms to these examples. It is due to the high variance of the estimate, which may increase the noise in the meta-level target, leading to a more difficult algorithm selection problem. In contrast, if γ is very large, the predictive performance of the different models and the characteristics extracted from these data will likely become similar on average, even if they are clearly distinct for different subsets of the selection window.

In this paper, we investigate two different strategies to label meta-examples. The first strategy, named Tie, labels a meta-example as a categorical value that represents the model with the best predictive performance if the difference between the predictive performance of the two models is larger than or equal to a predefined threshold δ. When the performance obtained by the models is not sufficiently different, the meta-example is labeled as a *tie*. The threshold δ is a free parameter, whose value is set by the user. The present paper investigates this strategy only when selecting between pair of algorithms because its generalization to more than two regressors (multi-regressors) is not trivial. A multi-label classification method [37] could be employed to deal with more than two algorithms using a similar approach to that used for the tie between two regressors, but it would demand a new set of experiments that we believe should be left for future works, since it is beyond the scope of this study. The second strategy, referred to as Combination, explicitly takes into account the predictive performance obtained by averaging the predictions from the regressors in addition to considering them separately. Thus, a meta-example is labeled as either one of the regressors or as the *combination* of the regressors. This strategy is investigated to select a regressor from a pool of two (pairs) or more (multi) regressors.

The Tie and Combination strategies were motivated by their potential strength. For example, the use of a threshold to label meta-examples may be a strength of the Tie strategy, since it creates a larger margin of separation between the two most important classes (i.e., the case when either one of the algorithms clearly wins), while separating the harder cases to discriminate. However, the choice of any of these cases does not really affect the base-level performance. On the other hand, a possible strength of the Combination strategy is that it labels a meta-example as the combination of models prediction only when it, in fact, is better than using the best model. As a result of the labeling strategies, two meta-data sets were created, differing only on the meta-target value. The Tie strategy generated a meta-data where each meta-example is labeled as either the best regressor algorithm or a *tie* (the predictive performance of the two models are not considerably different) whereas the Combination strategy generates a meta-data where each example is labeled as either the best regressor algorithm or a *combination* (integration of the predictions of the models).

3.2.2. Induction of a meta-learner

After the generation of a sufficient number of meta-examples, the MetaStream method can proceed to the next step. In order to map the characteristics of the base-level data to the performance of the regressors for these data, MetaStream employs a learning algorithm that uses the meta-data generated in the previous step to induce a meta-classifier. A sliding window mechanism is used by MetaStream at the meta-level, similar to the one used at the base-level, to induce a meta-classifier for each meta-data set that is created. By looking back at Fig. 2, one can see 40 meta-examples in the meta-data set (bottom-left part of Fig. 2) that were generated earlier. The number of meta-examples is defined by the meta-level parameters as well as the base-level experimental setup. In our illustrative example, $\omega_b = 100$, $\lambda_b = 1$ and $\gamma = 10$. As 500 examples were already processed, the first 100 examples in the base-level were used only for training. Thus, considering a training window size of $\omega_m = 40$ at the meta-level, all the

examples in the meta-data shown in Fig. 2 are used by a learning algorithm to induce the meta-classifier.

3.2.3. Recommending learning algorithms with the meta-models

In this step, the meta-model produced in the previous step predicts the class for the new unlabeled meta-example. If this label represents a single learning algorithm, this algorithm is used to predict the target of the base-level data. Otherwise, if it is predicted as *tie* or as *combination*, MetaStream uses the average prediction of the regression algorithms at the base-level. In Fig. 2, the meta-learner predicts the label for the meta-example 41. This label corresponds to the algorithm (or the combination of algorithms) that will be employed to predict the target of examples 601–610 at the base-level.

4. Experiments

In this section, we present the experiments carried out to evaluate the MetaStream predictive performance on selecting the most promising algorithm over time from a pool of either pairs of regressors, using both the Tie and Combination strategies, or multi-regressors using the Combination strategy. MetaStream is compared at the meta-level to a baseline method, named Default, which predicts the label of the test meta-examples as the majority class in the training data. At the base-level, besides Default, we also compare MetaStream to a simple ensemble approach, referred to as Ensemble, which predicts the target value of an example by averaging the predictions of the regressors under analysis. The remainder of this section presents the experimental setup for the evaluation of the base and the meta levels (Sections 4.1 and 4.2, respectively), the experimental results (Section 4.3), and the Discussion of the main results (Section 4.4).

4.1. Base-level setup

We use two real-world problems involving streaming data. The first data set is the Travel Time Prediction (TTP) [38], a regression problem in public transportation. The second data set is the Electricity Demand Prediction (EDP) [39], which is used for systems operation and planning in the energy sector.

Predicting the travel time of public transports may be useful for different tasks, like definition of crew's duties, helping users to decide the best route as well as the right departure time in order to arrive on time at the destination, and for real time adjustments of schedules [40]. Usually, TTP needs to be performed some days in advance for the first task, some minutes or hours in advance for the second, and online for the last task. Here, we are concerned with using regression models to predict travel time some minutes or hours in advance. For the purpose of this study, travel time is defined as the time a vehicle takes to travel from one terminal to the other. In urban networks, travel time of public transport is stochastic because delays in intersections and dwell time at stops, for instance, oscillate spatially and temporally [41]. Therefore, a mechanism to constantly adapt predictive models to changes in the TTP data set is necessary. In [38], the author investigated the problem of planning for passenger transport companies using data from the Sociedade de Transportes Colectivos do Porto SA (STCP[1]), the public bus operator company from the city of Porto, Portugal. The same data for the route 205-1-1, that contains 24,975 examples, are analyzed in this paper. This application does not generate data in equally spaced time, and periodicity may change arbitrarily, according to business strategies. Each example is described by 5 attributes: day of the week, day of the year, type of the day (holiday, bridge, tolerance and normal), departure time and travel time. The task is to predict the last attribute.

The EDP problem investigated here, known as Elec2, was collected from the Australian New South Wales Electricity Market and was first described in [39]. It contains 45,312 examples dated from 7 May 1996 to 5 December 1998, which were collected at 30 min intervals. Thus, there are 48 examples for each day. These data are widely used in the data streams literature [5,9,32] to investigate the classification task of predicting a rise or reduction in the electricity price according to a moving average of the last 24 h. However, in this paper, we investigate the task of predicting the New South Wales electricity demand for the next week (next 336 examples) as in [32]. Each example from this data set is described by 6 attributes: the day of the week, the time stamp related to the 30 min intervals (from 1 to 48), the NSW electricity demand, the Victoria electricity demand, the scheduled electricity transfer between states, and the change of the price using a moving average of the last 24 h (up or down). The target attribute is the NSW electricity demand. Here, we removed the first 17,424 examples concerning to the period up to 4 May 1997 because they do not have the attribute values for the Victoria electricity demand and the scheduled electricity transfer between states, which were included after that date.

Five regression algorithms were applied to these regression problems: Random Forests (RF) [42], Support Vector Machines (SVM) [43], Classification and Regression Trees (CART) [44], Project Pursuit Regression (PPR) [45] and Multivariate Adaptive Regression Splines (MARS) [46]. They all used the default parameter values suggested in the corresponding R packages [47]. For the TTP data set, we have used a training sliding window of $\omega_b = 1000$ examples, an evaluation window of $\lambda_b = 1$ and a delay of $\eta_b = 2$ examples whereas for the EDP data set, $\omega_b = 672$, $\lambda_b = 336$ and $\eta_b = 0$. These parameter values were selected based on the studies of [38,32].

4.2. Meta-level setup

In order to gain some insights on the robustness of MetaStream to parameter settings, we performed experiments using the interleaved test-then-train method with training windows of $\omega_m = \{200, 300\}$ meta-examples, an evaluation window of $\lambda_m = 1$ and three classification algorithms. Larger training windows were not examined because they would frequently include outdated data and reduce the amount of test meta-examples, decreasing the confidence on the evaluation of the methods. The size of the evaluation window was set to one because we wanted to make predictions based on meta-models which are up-to-date (i.e., they were induced by using the most recent training examples). Given that the size of the meta-data is relatively small, the cost of updating the meta-model is acceptable. The classification algorithms investigated as meta-learners were as follows: RF, known for its high predictive performance in many studies [48,49]; k-NN, commonly employed in other meta-learning problems [36,50]; and Naive Bayes (NB) [51], used for model selection in a data stream scenario [27]. As in the base-level, the parameters were set according to the default values on their R implementation.

These experiments indicated that MetaStream is more sensitive to the choice of the meta-learning algorithm than to the window size. In the interest of space, we only present the best results in Sections 4.3.1 and 4.3.2, which were obtained by MetaStream using the RF algorithm as the meta-learner and with a sliding window of 300 meta-examples for both MetaStream and Default methods. The influence of these parameters are briefly analyzed in Section 4.3.3, where we present the multi-regressors' experiments. MetaStream and Default are applied to select a learning algorithm

[1] www.stcp.pt.

periodically for each $\gamma = 25$ base examples (selection window) for the TTP data set and $\gamma = 24$ (corresponding to half-day) for the EDP data set. These values were defined as a trade-off between the quality of the estimates of the values of the meta-features (i.e., the larger the window, the better) and the variance of the performance of the algorithms (i.e., the smaller the window, the more clearly the winner typically is).

In addition to these crucial MetaStream parameters, the threshold δ of the Tie strategy also has to be set. It defines whether the performance of two regressors can be considered as different. A statistical test could be used for this purpose. However, the outcomes of the tests are expected to be very conservative because of the small number of examples in the selection window. On the other hand, as smaller sets are used at the base-level, the number of meta-level examples increases. Thus, any systematic difference in the performance of the meta-level models is expected to be significant. Since we are using a relative measure to assess base-level, the δ value represents the percentage over the target deviation. Based on the difference between regressors performance for the first training window, we set this parameter to 0.1. Larger values could be as conservative as statistical tests and would prevent finding differences for a small number of examples (γ), whereas smaller values may define the performance of two regressors as different which is really very similar.

The amount of meta-examples generated by MetaStream for the TTP and EDP data sets using this experimental setting was 958 and 1120, respectively. From this total, the first 300 meta-examples were used only for training (the first 100 meta-examples were discarded for $\omega_m = 200$) whereas the last 657 and 812 meta-examples were used to evaluate the methods of algorithm selection at the meta-level for TTP and EDP, respectively. The remainder to the total number of meta-examples is due to the delay between predicting the meta-label and observing its true value. These methods and the Ensemble approach are evaluated at the base-level for the corresponding base-level examples. The class distribution of these meta-examples, considering pairs of regressors, is depicted in Fig. 3. It is averaged over all training windows at the meta-level for each labeling strategy (Tie and Combination) and pair of regressors. The figure shows that the meta-data sets for TTP (Fig. 3(a)) generally present a more imbalanced class distribution in comparison to EDP (Fig. 3(b)). This means that one regressor outperformed the other for the majority of the meta-examples on TTP. On the other hand, a balanced class distribution for the EDP data set occurred because each regressor outperformed the other on nearly the same number of meta-examples. Regarding the multi-regressors problem, we analyze the frequency of each class (each regressor or *combination*) for each training window over time. In Fig. 4, we observe that the frequency of classes for both

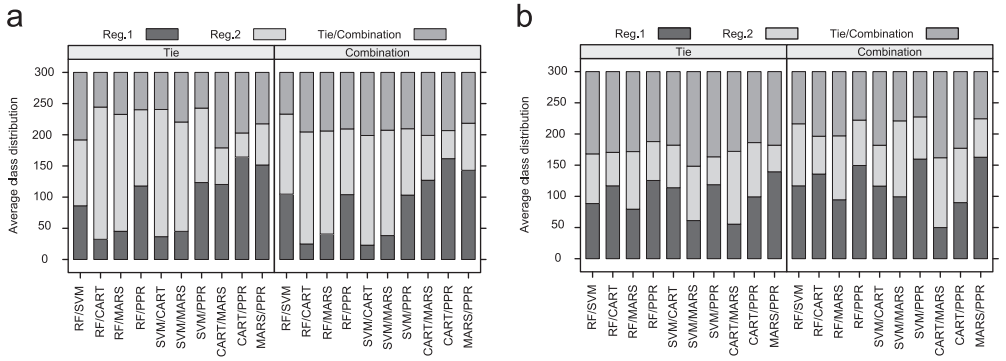

Fig. 3. Average distribution of classes over all training windows for each pair of algorithms and meta-data set. (a) TTP, (b) EDP.

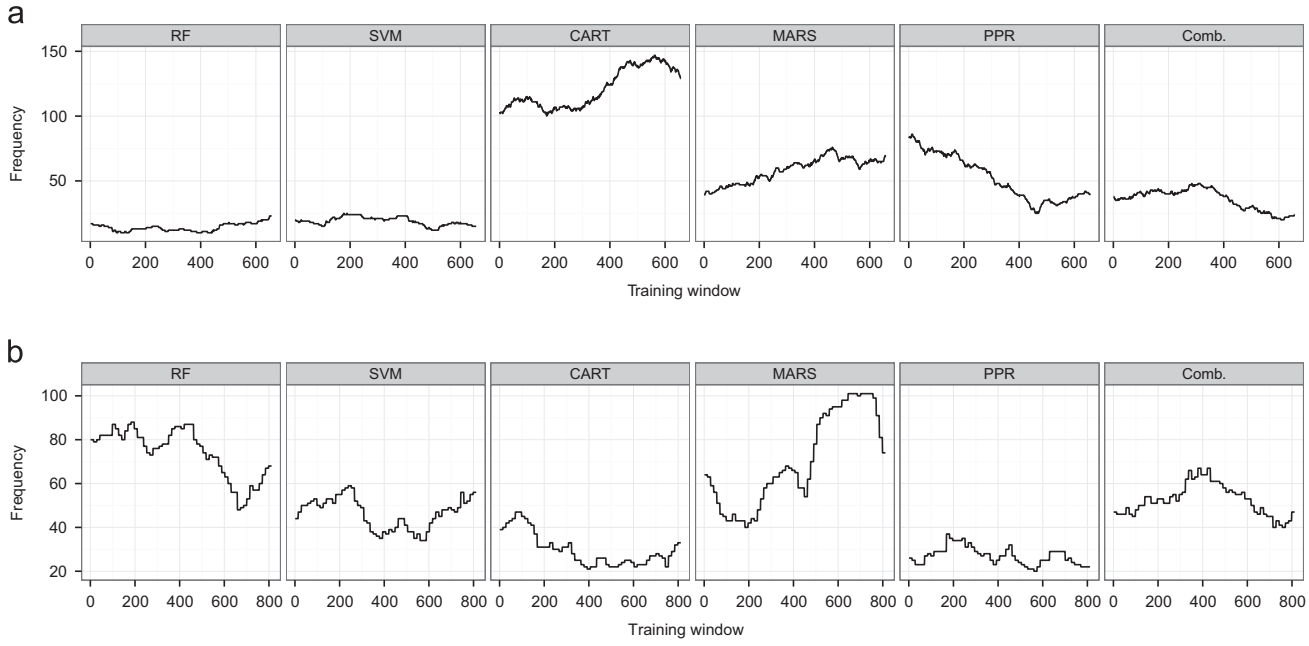

Fig. 4. Frequency of classes over time for each training window of 300 meta-examples for the (a) TTP and (b) EDP data sets.

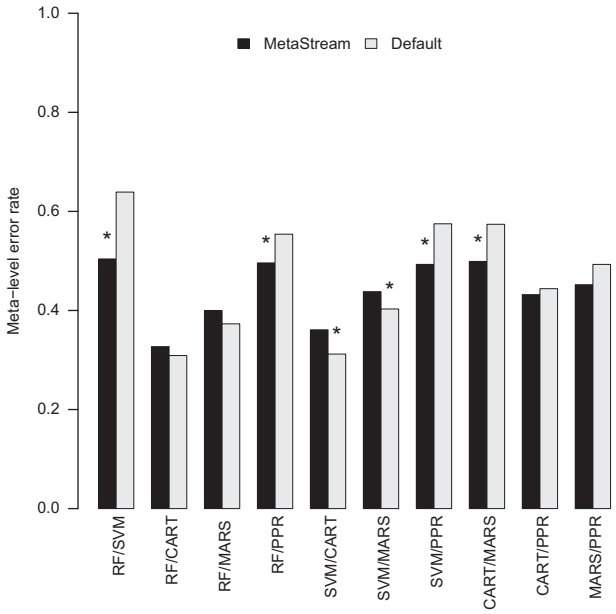

Fig. 5. Meta-level error rates for the TTP data set using the Tie strategy. The star represents a significantly smaller error.

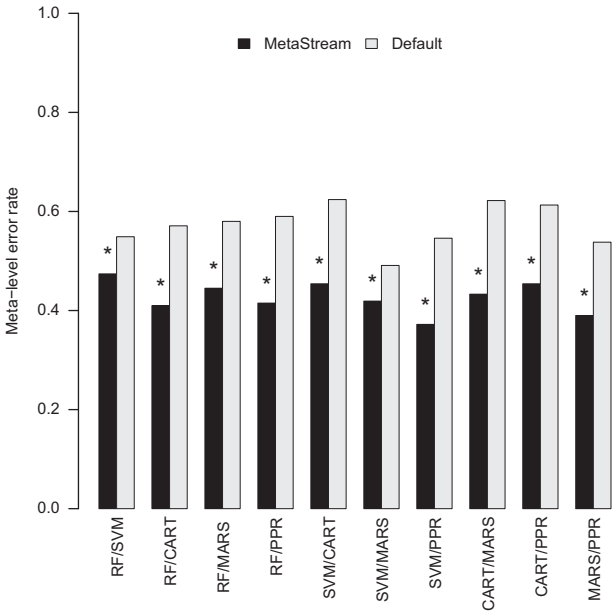

Fig. 6. Meta-error rates for the EDP data set using the Tie strategy. The star represents a significantly smaller error.

data sets changes over time. This evidences that it is necessary to select an algorithm dynamically for these data sets. At the beginning, there was a larger number of meta-examples labeled as PPR than as MARS on TTP (Fig. 4(a)). After the 250th training window, there was a greater number of meta-examples labeled as the latter class. This is due either to the improvement of the results of MARS or to the decline of PPR. Something similar happened with the other classes for EDP (Fig. 4(b)), mainly for MARS and RF. Furthermore, the majority class for TTP (CART) is one of the minority classes for EDP. The opposite is also true for the RF class.

4.3. Experimental results

In this section, we analyze the MetaStream results for the TTP and EDP data sets and compare them to those obtained by the Default method and the Ensemble approach. The performance of MetaStream and Default at the meta-level is assessed by the predictive error rate. In order to test whether the difference in errors is significantly different, we applied the McNemar test for each comparison, as suggested by [33], with a significance level of $\alpha = 0.05$. The null hypothesis to be tested is the equality between their error rates. The significant differences are marked with a star at Figs. 5–8.

In order to evaluate the performance of MetaStream and Default at the base-level, the regressors selected by both methods are applied to the TTP and EDP problems and the results are compared using the Normalized Mean Squared Error (NMSE) statistic [36]. Moreover, MetaStream and Default are compared to the Ensemble approach at the base-level. The smallest value of NMSE for each pair of regressors is presented in bold in Tables 2–5. In addition to the results achieved by the algorithm selection methods and the Ensemble approach, we also present the performance of each individual regressor (columns *Reg. 1* and *Reg. 2*) for the experiments with pairs of regressors, and the theoretical minimum NMSE (column *Min.*), which is computed by choosing the best regressor for each selection window. The meta-level and base-level results for the multi-regressors problem are exhibited in Tables 6 and 7. The larger the number of test examples considered in the comparison, the higher the confidence that even a small difference between two methods is significant [33]. Thus,

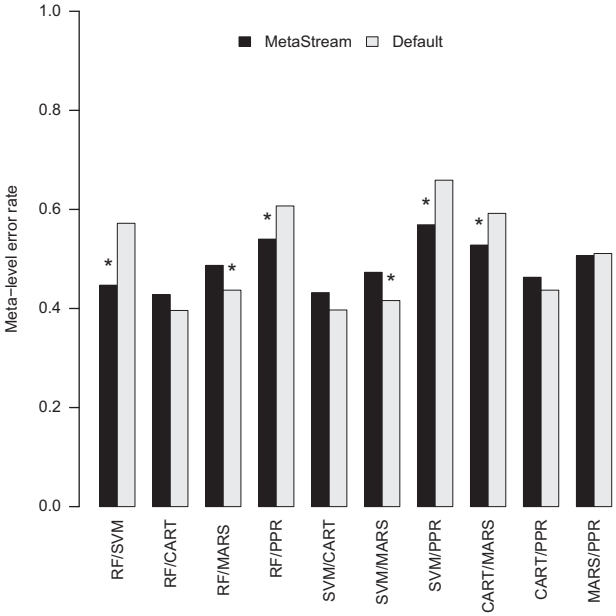

Fig. 7. Meta-level error rates for the TTP data set using the Combination strategy for pairs of regressors. The star represents a significantly smaller error.

because of the large number of test examples used at the base-level,[2] we did not apply any statistical test to assess the significance of the difference in errors.

4.3.1. Tie strategy

The results obtained by MetaStream and Default at the meta-level considering the Tie strategy for the TTP data set (Fig. 5) show that the former outperformed the latter for the majority pairs of regressors. In such cases, the McNemar test detected significant differences for four out of six comparisons. For the other four

[2] The number of test examples in the base-level is given by the number of test meta-examples times the size γ of the selection window.

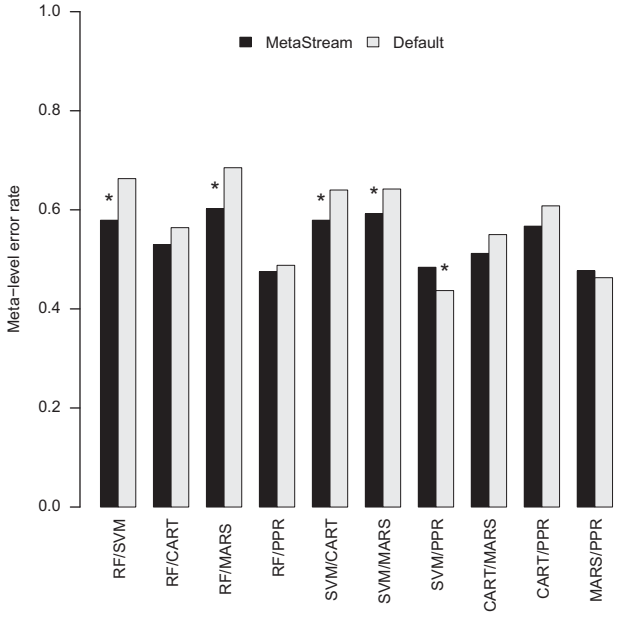

Fig. 8. Meta-level error rates for the EDP data set using the Combination strategy for pairs of regressors. The star represents a significantly smaller error.

Table 2
NMSE values for the TTP data set using the Tie strategy. The values in bold are the best results for the corresponding pair of algorithms.

Regressors	MetaS.	Default	Ens.	Reg. 1	Reg. 2	Min.
RF/SVM	**0.662**	0.676	0.667	0.705	0.688	0.630
RF/CART	0.510	**0.509**	0.529	0.705	0.509	0.481
RF/MARS	1.049	1.048	**0.693**	0.705	1.048	0.535
RF/PPR	0.684	0.726	**0.670**	0.705	0.781	0.595
SVM/CART	0.515	**0.509**	0.520	0.688	0.509	0.474
SVM/MARS	**0.703**	1.048	0.708	0.688	1.048	0.532
SVM/PPR	0.672	0.716	**0.658**	0.688	0.781	0.585
CART/MARS	**0.492**	0.501	0.614	0.509	1.048	0.464
CART/PPR	0.514	**0.511**	0.543	0.509	0.781	0.476
MARS/PPR	1.052	1.069	**0.698**	1.048	0.781	0.518

Table 3
NMSE values for the EDP data set using the Tie strategy. The values in bold are the best results for the corresponding pair of algorithms.

Regressors	MetaS.	Default	Ens.	Reg. 1	Reg. 2	Min.
RF/SVM	0.154	**0.153**	**0.153**	0.167	0.181	0.126
RF/CART	0.160	0.160	**0.158**	0.167	0.193	0.136
RF/MARS	0.154	**0.143**	**0.143**	0.167	0.176	0.117
RF/PPR	0.165	0.167	**0.159**	0.167	0.212	0.128
SVM/CART	0.165	0.171	**0.157**	0.181	0.193	0.130
SVM/MARS	**0.160**	**0.160**	**0.160**	0.181	0.176	0.136
SVM/PPR	**0.176**	0.181	0.177	0.181	0.212	0.148
CART/MARS	0.163	0.152	**0.147**	0.193	0.176	0.125
CART/PPR	0.183	0.179	**0.163**	0.193	0.212	0.141
MARS/PPR	0.173	**0.168**	0.172	0.176	0.212	0.143

cases, Default was significantly better than MetaStream for two pairs. Class distribution plays an important role in the performance of the algorithm selection methods, as can be seen by observing Figs. 5 and 3(a) together. When the problem is very imbalanced, Default achieved significantly smaller NMSE values than MetaStream, such as for SVM/CART. With an average class distribution of 37/204/59, the error rate produced by MetaStream is 0.049 greater than that of Default. This result is expected as we did not use any of the approaches that are commonly used to deal

Table 4
NMSE values for the TTP data set using the Combination strategy for pairs of regressors. The values in bold are the best results for the corresponding pair of algorithms.

Regressors	MetaS.	Default	Ens.	Reg. 1	Reg. 2	Min.
RF/SVM	**0.658**	0.686	0.667	0.705	0.688	0.627
RF/CART	**0.507**	0.509	0.529	0.705	0.509	0.476
RF/MARS	**0.691**	1.048	0.693	0.705	1.048	0.528
RF/PPR	**0.669**	0.723	0.670	0.705	0.781	0.591
SVM/CART	0.511	**0.509**	0.520	0.688	0.509	0.469
SVM/MARS	**0.703**	1.048	0.708	0.688	1.048	0.530
SVM/PPR	0.661	0.729	**0.658**	0.688	0.781	0.581
CART/MARS	0.619	**0.509**	0.614	0.509	1.048	0.460
CART/PPR	0.514	**0.510**	0.543	0.509	0.781	0.470
MARS/PPR	1.054	1.087	**0.698**	1.048	0.781	0.512

Table 5
NMSE values for the EDP data set using the Combination strategy for pairs of regressors. The values in bold are the best results for the corresponding pair of algorithms.

Regressors	MetaS.	Default	Ens.	Reg. 1	Reg. 2	Min.
RF/SVM	0.159	0.172	**0.153**	0.167	0.181	0.124
RF/CART	0.162	0.167	**0.158**	0.167	0.193	0.134
RF/MARS	0.155	0.176	**0.143**	0.167	0.176	0.114
RF/PPR	0.164	0.167	**0.159**	0.167	0.212	0.123
SVM/CART	0.167	0.172	**0.157**	0.181	0.193	0.127
SVM/MARS	0.171	0.187	**0.160**	0.181	0.176	0.133
SVM/PPR	**0.175**	0.181	0.177	0.181	0.212	0.144
CART/MARS	0.160	**0.147**	**0.147**	0.193	0.176	0.121
CART/PPR	0.174	0.165	**0.163**	0.193	0.212	0.136
MARS/PPR	0.173	0.176	**0.172**	0.176	0.212	0.140

Table 6
Meta-level error rates and base-level NMSE values for the TTP data set using the Combination strategy for multi-regressors. The values in bold for each level are the best results and the gray background represents significant differences for the corresponding experimental setting.

Meta-learner	ω_m	Meta-level		Base-level			Min.
		MetaS.	Default	MetaS.	Default	Ens.	
RF	200	**0.577**	0.583	0.521	**0.509**	0.560	0.444
RF	300	**0.569**	0.583	0.519	**0.509**	0.560	0.444
k-NN	200	0.641	**0.583**	0.530	**0.509**	0.560	0.444
k-NN	300	0.642	**0.583**	0.532	**0.509**	0.560	0.444
NB	200	0.717	**0.583**	0.562	**0.509**	0.560	0.444
NB	300	0.712	**0.583**	0.559	**0.509**	0.560	0.444

with imbalanced classes [52]. On the other hand, for problems with more even distributions, MetaStream outperformed the baseline method, because it is able to extract higher-order knowledge from data. Thus, for the pair of algorithms RF/SVM, with an average class distribution of 86/106/108, MetaStream was significantly better than Default (error rates of 0.504 and 0.639, respectively).

The advantages of MetaStream over Default at the meta-level usually led to better performance at the base-level, as can be seen in Table 2. However, it is also possible to observe some disagreements between the meta and base levels. It occurred because learning at the meta-level does not consider the error magnitude of the regressors at the base-level. Thus, the correct or incorrect classification of a meta-example always has the same positive or negative, respectively, weight on the meta-level error. However, the base-level error depends on the magnitude of the difference between the regressors errors. For instance, while MetaStream

Table 7
Meta-level error rates and base-level NMSE values for the EDP data set using the Combination strategy for multi-regressors. The values in bold for each level are the best results and the gray background represents significant differences for the corresponding experimental setting.

Meta-learner	ω_m	Meta-level		Base-level			Min.
		MetaS.	Default	MetaS.	Default	Ens.	
RF	200	**0.707**	0.754	0.163	0.182	**0.143**	0.092
RF	300	**0.687**	0.743	0.155	0.181	**0.143**	0.092
K-NN	200	**0.727**	0.754	0.162	0.182	**0.143**	0.092
K-NN	300	**0.732**	0.743	0.161	0.181	**0.143**	0.092
NB	200	0.755	**0.754**	0.160	0.182	**0.143**	0.092
NB	300	**0.741**	0.743	0.156	0.181	**0.143**	0.092

significantly outperformed Default for RF/SVM at the meta-level, it obtained a small gain at the base-level. This dissonance is more apparent for the SVM/MARS pair because MARS seems to be more prone to make spurious predictions. Thus, if this regressor is erroneously chosen, the NMSE may rise considerably. In summary, the algorithms selected by MetaStream yielded smaller NMSE values than those selected by Default for six out of ten pairs of regressors. For the other pairs, the latter method obtained better results than the former, but by a narrow margin. At the base-level, MetaStream can also be compared with the Ensemble approach. In general, the former presented smaller NMSE values, outperforming the latter for six out of ten pairs. For RF/MARS and MARS/PPR, MetaStream performed substantially worse than Ensemble. This odd behavior arises from the selection of MARS by MetaStream, in cases where the predictions made by this regressor are very inaccurate. Considering all pairs of algorithms, the best results are achieved when MetaStream selects between CART/MARS. It is also better when each algorithm is chosen separately (*Reg. 1* and *Reg. 2*), and there is still margin for improvements, as can been seen in Table 2.

For the EDP data set, MetaStream yielded significantly smaller error rates than Default for all pairs of algorithms at the meta-level (Fig. 6). Once again, these achievements can be partially explained by the balanced average distribution of classes, which makes the Default method less competitive. Regarding the base-level results shown in Table 3, we note that the Ensemble approach produced the smallest NMSE values for the majority of the pairs of regressors. It is due to the fact that averaging the predictions of the two regressors under analysis is a good approach for these data while MetaStream and Default were not able to select the best regressor at the meta-level accurately as would be necessary to outperform Ensemble at the base-level. MetaStream and Default results at the meta and base levels are in strong disagreement. This can be explained by analyzing the average class distribution of the meta-examples in Fig. 3(b). Since *tie* is the majority class for some pairs or regressors, Default usually selects this class. As stated earlier, when the class *tie* is selected, the base-level evaluation uses the average of the regressors prediction. Consequently, the Default error at the base-level did not increase considerably when it selected *tie* instead of the best regressor and the NMSE values obtained by this method were similar to those obtained by Ensemble for most of the comparisons, as can be seen in Table 3.

4.3.2. Combination strategy for pairs of regressors

Here, we analyze the behavior of MetaStream for the meta-data set labeled using the Combination strategy. By looking at Fig. 7, one can observe that MetaStream and Default performed similarly for the TTP data set regarding the number of wins and losses at the meta-level. Statistically, MetaStream was better than Default on four and worse on two comparisons. As in the Tie strategy, these results are strongly related to the average class distribution of the meta-data, presented in Fig. 3(a). For balanced problems, such as SVM/PPR and CART/MARS, MetaStream is a more suitable choice, since it is able to map data characteristics to the target well. For imbalanced problems, Default takes advantage of predicting the majority class and MetaStream usually was not able to do better. For instance, when the pair SVM/MARS is considered, the average number of meta-examples labeled as MARS represents more than half of the training meta-examples. Hence, Default produced a reasonable error rate of 0.416 by always predicting MARS.

Interestingly, the dominance of Default over MetaStream for some pairs did not lead to better results at the base-level, as shown in Table 4. Thus, while Default significantly outperformed MetaStream at the meta-level for RF/MARS and SVM/MARS, its NMSE values were greater than the latter for the same pairs. This occurred because, in spite of being the majority class, MARS can make very poor predictions (see column *Reg. 2* for these pairs of regressors), as discussed earlier. This raises the NMSE for Default. Other disagreements between the results obtained at the two levels, such as for RF/CART, are likely to appear mainly due to the slight differences between MetaStream and Default error rates at the meta-level and to the fact that these methods do not take the magnitude of the error into account at the meta-level. In summary, MetaStream was superior to Default and Ensemble for seven out of ten comparisons. It also achieved the lowest NMSE (0.507) for the pair RF/CART, considering all methods and pairs of regressors.

Analyzing the results for the EDP data set, presented in Fig. 8, one can see that MetaStream yielded significantly smaller error rates for four pairs of algorithms whereas the opposite was observed for only one pair. For the remaining cases, MetaStream performed better than Default (except for one pair), but the statistical test could not reject the null hypothesis. Regarding the base-level, MetaStream maintained its superiority, achieving smaller NMSE values than its contender for eight comparisons, including SVM/PPR, for which the Default obtained an error rate that was significantly lower than the error of MetaStream at the meta-level. On the other hand, MetaStream outperformed the Ensemble approach only for one pair of regressors. This evidences that averaging predictions is a suitable approach for these data, consistently with the results from the previous section. Moreover, the Ensemble approach presented the smallest general NMSE for RF/MARS (0.143), including the comparison with any of the two regressors taken individually. However, there is the possibility of further reduction by algorithm selection methods, as shown in the *Min.* column.

4.3.3. Combination strategy for multi-regressors

As we mentioned before, the Combination strategy can naturally be extended to deal with multi-regressors at a time, instead of pair of regressors. In this section, we investigate this possibility using the same regressors employed at the base-level previously. As in the Combination strategy for pairs of algorithms, a simple ensemble-based approach, referred to again as Ensemble, can be obtained by averaging predictions of all of them. This experimental setup has a lower computational complexity and generates a much fewer number of results than in the case where pairs of algorithms were handled separately. Therefore, it is possible to analyze the behavior of MetaStream for all parameter settings (see Section 4.2). The number of test meta-examples used for all comparisons is equal to the number of test meta-examples available when the largest window size is analyzed. Thus, the same meta-examples are used to evaluate the performance of the methods, regardless of the window

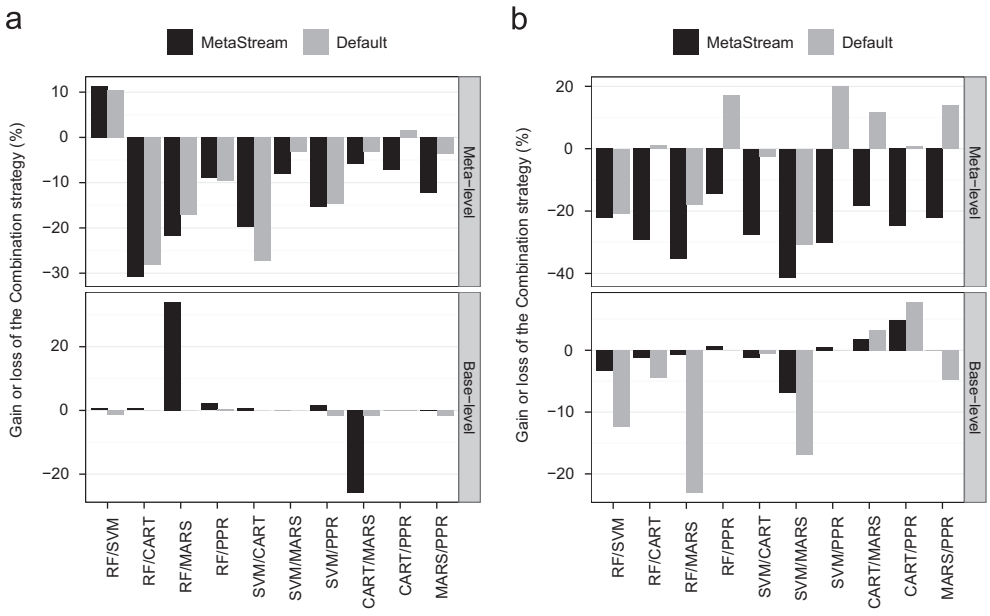

Fig. 9. Gains and losses of using Combination strategy instead of Tie. (a) TTP, (b) EDP.

size. This is necessary because the larger the window size, the fewer the number of meta-examples available for the evaluation of the method. The results at the base-level are also comparable to those for pairs of algorithms, since the largest window size was also used in those experiments. Tables 6 and 7 present the error rates for the meta-level and the NMSE values for the base-level for the TTP and EDP data sets, respectively. The smallest error for each level and parameter setting is in bold and significant differences at the meta-level are highlighted using a gray background.

According to Table 6, MetaStream using RF outperformed Default and the other meta-learners at the meta-level, independent of the window size, ω_m. On the other hand, Default presented significantly better results than MetaStream using k-NN and NB for all window sizes. The superiority of RF at the meta-level did not translate into equally better performance at the base-level. This can be explained by the small differences between these methods at the meta-level. As in the experiments with pairs of algorithms, the imbalanced class distribution (Fig. 3(a)) makes Default more competitive. In fact, this method always selects the CART regressor, since it is the majority class for all training windows over time. Nevertheless, MetaStream was able to obtain lower NMSE values than Ensemble for all meta-learners and window sizes, except for NB with 200 meta-examples. This is due to the spurious predictions made by MARS and the inaccurate predictions made by other regressors that integrate the Ensemble, such as RF and SVM. Column *Min.* suggests that the NMSE can be reduced by improving the MetaStream predictive performance at the meta-level. The Default method and the Ensemble approach, on the other hand, cannot improve their results.

For the EDP data set, the results shown in Table 7 make the superiority of MetaStream clear when compared to Default at the meta-level. Statistically significant differences between these methods occurred when RF was employed as the meta-learner, mainly with a training window of 300 meta-examples, while NB error rates are just slightly lower than the ones achieved by Default. Unlike for the TTP data set, MetaStream also obtained smaller NMSE values than Default at the base-level. Although k-NN and NB have a distinct performance at the meta-level, they were very similar at the base-level. Once again, RF with a sliding window of 300 meta-examples obtained the best solution. Despite these advantages, MetaStream presented worse results than Ensemble, regardless of the meta-learner and the window size. The reasons for these achievements are the difficulty of MetaStream on dealing with this meta-classification problem and the suitability of the Ensemble approach for this data set, such as the absence of spurious predictions, as those made by MARS for TTP, and the complementarity of the predictions made by the regressors.

4.4. Discussion

In this section, we focus on the comparison between the Tie and Combination strategies and briefly discuss the main results obtained at the base-level by MetaStream, Default and Ensemble for both pairs of regressors and multi-regressors.

4.4.1. Labeling strategies

In order to provide a more comprehensive comparison between the labeling strategies, Fig. 9 shows the gains and the losses, in percentage, for the Combination strategy in relation to Tie. A positive bar indicates a better result obtained with Combination (in percentage) whereas a negative bar indicates that better results are obtained with Tie.

It is important to observe in this figure the losses of MetaStream and Default at the meta-level for both data sets. For TTP (Fig. 9(a)), both methods improved their performance only for the pair RF/SVM whereas, for EDP (Fig. 9(b)), there were gains in a few pairs, but only for the Default method. Default generally performed worse using Combination than Tie because the number of meta-examples of the majority class in the former is usually smaller than in the latter. For instance, for the pair SVM/CART regarding the TTP data (Fig. 3(a)), the number of meta-examples of the majority class (CART) is smaller in Combination than in Tie. Thus, Default obtained worse results using Combination for this pair of regressors at the meta-level. The opposite scenario can be seen in the same figure for the pair RF/SVM. In this case, the number of meta-examples of the majority class in Combination is larger than in the Tie strategy. MetaStream can also be favored by using Tie because this strategy creates a margin of separation

between the most important classes, i.e., when one of the regressors clearly outperforms the other. Conversely, the Combination strategy is more likely to have a narrow margin of separation between classes (or among classes for the multi-regressors case). This occurs, for instance, when one regressor just slightly outperforms the other for a set of examples, i.e., the difference in their performance may not be sufficient to state which one is the best.

Another observation that can clearly be made in this figure is the small degree of changes at the base-level, despite the clear differences at the meta-level. Hence, the advantages of using Tie at the meta-level did not lead to better results at the base-level. A possible explanation for this result is that, although Tie creates a large margin of separation between classes when it labels a meta-example as *tie*, it is not guaranteed that averaging the predictions of the regressors when this class is predicted produces a lower NMSE value than selecting any of the regressors. The two exceptions are RF/MARS and CART/MARS, which are caused, as explained earlier, by some very large errors made by MARS. Concerning EDP (Fig. 9(b)), besides the results involving MARS, better results were obtained by MetaStream and Default using the Combination strategy for CART/MARS and CART/PPR while the best results were obtained using the Tie strategy for RF/SVM and RF/CART. In summary, the differences between the Tie and Combination strategies were verified at the meta-level. However, these differences had a minor impact on the base-level error, mainly for MetaStream, which the results suggest to be more robust than Default. The larger differences for Default may be due to the fact that it entirely depends on the class distribution.

4.4.2. Base-level results

Analyzing all the base-level results for pairs of regressors and multi-regressors, we can argue that MetaStream is more robust than both Default and Ensemble. Regarding the TTP data set, MetaStream was the recommended method for most of the pairs of regressors. For the multi-regressors case, MetaStream was slightly worse than Default but always better than Ensemble. This scenario is inverted for the EDP data set. While MetaStream was usually better than Default for pairs of regressors and always better than Default for multi-regressors, it was worse than Ensemble for all comparisons. For the multi-regressors' comparisons, MetaStream was marginally worse than the Ensemble approach, but considerably better than the Default method, regardless of the data set.

5. Conclusion

In this study, we presented a method for periodic algorithm selection in time-changing environments, named MetaStream. This method attempts to predict the most appropriate learning algorithm dynamically for new data. MetaStream follows a meta-learning approach, which uses machine learning to automatically relate characteristics extracted from past and incoming data to the predictive performance of learning algorithms. The present study extended the original paper that proposed MetaStream [19] in three main directions. Firstly, we enhanced MetaStream in order to select a regressor (or the combination of regressors) from a pool of more than two regressors. Secondly, in addition to the Tie strategy, we proposed a new labeling strategy, referred to as Combination. This strategy labels a meta-example as *combination* when the average of the predictions of the regressors is better than any of them separately. Lastly, we comprehensively evaluated MetaStream using the Tie and Combination labeling strategies for the TTP and EDP data sets, which are real-world problems.

Empirical results considering pairs of regressors for the TTP data set showed that MetaStream, regardless of the strategy employed, is superior to Default and Ensemble for the majority of the pairs. With respect to the EDP data set, MetaStream clearly outperformed Default, but presented worse results than Ensemble for both the labeling strategies. In general, the observations made for pairs of regressors are also suitable for multi-regressors. MetaStream using RF as meta-learner achieved NMSE values comparable to Default and smaller than Ensemble for the TTP data set, whereas MetaStream was better than Default but worse than Ensemble for the EDP data set. According to these results, MetaStream is more robust than both Default and Ensemble, since it was the worst method in very few cases, while both Default and Ensemble were frequently the worst. Regarding the labeling strategies, Combination and Tie yielded similar results, but the former has the advantage of not having parameters to be set and its generalization for the multi-regressors problem is straightforward.

In this study we used a set of traditional meta-features to characterize data. However, applying meta-learning on time-changing data is very different from conventional scenarios. For instance, here, the recommendations are made for sets of examples that have exactly the same structure (i.e., the examples are characterized by the same set of variables). Therefore, it makes sense to compute meta-features that characterize individual variables, unlike in the traditional scenario. As future work, we plan to explore this possibility as well as study meta-features from related areas, such as time-series analysis, as investigated in [35,53–55]. Moreover, the results for the EDP data set suggest that averaging the predictions of the regressors may be a promising approach. Therefore, we also plan to investigate how MetaStream can be employed to dynamically integrate these predictions in order to obtain the final ensemble prediction [56]. For instance, rather than using the same weights for all predictions of the regressors, they could be provided by the class probabilities generated by the meta-classifier [11].

Acknowledgments

The authors would like to thank the financial support of the Brazilian funding agencies FAPESP, CAPES and CNPq. This work is also partly funded by National Funds through the FCT - Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project KDUS (PTDC/EIA-EIA/098355/2008) and the ERDF - European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT within project “FCOMP - 01-0124-FEDER-022701”.

Appendix A. Supplementary material

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.neucom.2013.05.048.

References

[1] P. Domingos, A few useful things to know about machine learning, Communications of the ACM 55 (10) (2012) 78–87, http://dx.doi.org/10.1145/2347736.2347755.
[2] J. Gama, Knowledge Discovery from Data Streams, 1st edition, Chapman & Hall/CRC, 2010.
[3] J. Gama, P. Medas, G. Castillo, P.P. Rodrigues, Learning with drift detection, in: A.L.C. Bazzan, S. Labidi (Eds.), Proceedings of the 17th Brazilian Symposium on Artificial Intelligence, vol. 3171, Springer, São Luis, Maranhão, 2004, pp. 286–295.
[4] D. Alberg, M. Last, A. Kandel, Knowledge discovery in data streams with regression tree methods, WIREs: Data Mining and Knowledge Discovery 2 (1) (2012) 69–78, http://dx.doi.org/10.1002/widm.51.
[5] J.Z. Kolter, M.A. Maloof, Dynamic weighted majority: an ensemble method for drifting concepts, Journal of Machine Learning Research 8 (2007) 2755–2790.

[6] D. Kifer, S. Ben-David, J. Gehrke, Detecting change in data streams, in: Proceedings of the 13th International Conference on Very Large Data Bases, VLDB Endowment, 2004, pp. 180–191.
[7] T. Dasu, S. Krishnan, G. Pomann, Robustness of change detection algorithms, in: J.A. Gama, E. Bradley, J. Hollmén (Eds.), Advances in Intelligent Data Analysis, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2011, pp. 125–137.
[8] H. Wang, W. Fan, P.S. Yu, J. Han, Mining concept-drifting data streams using ensemble classifiers, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2003, pp. 226–235. http://dx.doi.acm.org/10.1145/956750.956778.
[9] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, R. Gavaldà, New ensemble methods for evolving data streams, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2009, pp. 139–148. http://dx.doi.acm.org/10.1145/1557019.1557041.
[10] S. Džeroski, P. Panov, B. Ženko, Ensemble methods in machine learning, in: R.A. Meyers (Ed.), Encyclopedia of Complexity and Systems Science, Springer, New York, 2009, pp. 5317–5325.
[11] T.G. Dietterich, Ensemble methods in machine learning, in: Proceedings of the First International Workshop on Multiple Classifier Systems, Springer-Verlag, London, UK, 2000, pp. 1–15.
[12] J. Rodríguez, L. Kuncheva, Combining online classification approaches for changing environments, in: International Workshop on Structural, Syntactic, and Statistical Pattern Recognition, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2008, pp. 520–529.
[13] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: bagging, boosting, and variants, Machine Learning 36 (1–2) (1999) 105–139, http://dx.doi.org/10.1023/A:1007515423169.
[14] N. García-Pedrajas, C. Hervás-Martínez, D. Ortiz-Boyer, Cooperative coevolution of artificial neural network ensembles for pattern classification, IEEE Transactions on Evolutionary Computation 9 (3) (2005) 271–302, http://dx.doi.org/10.1109/TEVC.2005.844158.
[15] M. Hibon, T. Evgeniou, To combine or not to combine: selecting among forecasts and their combinations, International Journal of Forecasting 21 (1) (2005) 15–24, http://dx.doi.org/10.1016/j.ijforecast.2004.05.002.
[16] D. Saso, B. Ženko, Is combining classifiers with stacking better than selecting the best one? Machine Learning 54 (2004) 255–273, http://dx.doi.org/10.1023/B:MACH.0000015881.36452.6e.
[17] K.A. Smith-Miles, Cross-disciplinary perspectives on meta-learning for algorithm selection, ACM Computing Surveys 41 (1) (2008) 1–25 http://dx.doi.acm.org/10.1145/1456650.1456656.
[18] P. Brazdil, C. Giraud-Carrier, C. Soares, R. Vilalta, Metalearning: Applications to Data Mining, Springer Verlag, 2009.
[19] A.L.D. Rossi, A.C.P.L.F. de Carvalho, C. Soares, Meta-learning for periodic algorithms selection in time-changing data, in: Proceedings of the Brazilian Symposium on Neural Networks, IEEE Computer Society, 2012, pp. 7–12. http://dx.doi.org/10.1109/SBRN.2012.50.
[20] D. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
[21] C. Giraud-Carrier, R. Vilalta, P. Brazdil, Introduction to the special issue on meta-learning, Machine Learning 54 (3) (2004) 187–193 http://dx.doi.org/10.1023/B:MACH.0000015878.60765.42.
[22] R. Vilalta, Y. Drissi, A perspective view and survey of meta-learning, Artificial Intelligent Review 18 (2) (2002) 77–95 http://dx.doi.org/10.1023/A:1019956318069.
[23] R. Klinkenberg, Meta-learning, model selection, and example selection in machine learning domains with concept drift, in: J. Furnkranz, G. Grieser (Eds.), Proceedings of the Annual Workshop of the Special Interest Group on Machine Learning, Knowledge Discovery, and Data Mining (FGML-2005) of the German Computer Science Society Learning – Knowledge Discovery – Adaptivity (LWA-2005), 2005, pp. 164–171.
[24] M. Last, Online classification of nonstationary data streams, Intelligent Data Analysis 6 (2) (2002) 129–147.
[25] G. Widmer, Tracking context changes through meta-learning, Machine Learning 27 (3) (1997) 259–286 http://dx.doi.org/10.1023/A:1007365809034.
[26] M.B. Harries, C. Sammut, Extracting hidden context, Machine Learning 32 (1998) 101–126.
[27] J. Gama, P. Kosina, Learning about the learning process, in: Proceedings of the 10th International Conference on Advances in Intelligent Data Analysis, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 162–172.
[28] J.B. Gomes, E. Menasalvas, P.A.C. Sousa, Learning recurring concepts from data streams with a context-aware ensemble, in: Proceedings of the ACM Symposium on Applied Computing, ACM, New York, NY, USA, 2011, pp. 994–999. http://dx.doi.org/10.1145/1982185.1982403.
[29] I. Žliobaite, A. Bifet, M. Gaber, B. Gabrys, J. Gama, L. Minku, K. Musial, Next challenges for adaptive learning systems, SIGKDD Explorations 14 (1) (2012) 48–55, http://dx.doi.org/10.1145/2408736.2408746.
[30] D. Potts, C. Sammut, Incremental learning of linear model trees, Machine Learning 61 (2005) 5–48 http://dx.doi.org/10.1007/s10994-005-1121-8.
[31] E. Ikonomovska, J. Gama, S. Džeroski, Learning model trees from evolving data streams, Data Mining and Knowledge Discovery 23 (1) (2011) 128–168, http://dx.doi.org/10.1007/s10618-010-0201-y.
[32] R. Fidalgo-Merino, M. Nunez, Self-adaptive induction of regression trees, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (2011) 1659–1672, http://dx.doi.ieeecomputersociety.org/10.1109/TPAMI.2011.19.
[33] J. Gama, R. Sebastião, P.P. Rodrigues, Issues in evaluation of stream learning algorithms, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2009, pp. 329–338. http://dx.doi.acm.org/10.1145/1557019.1557060.
[34] C. Soares, Learning Rankings of Learning Algorithms: Recommendation of Algorithms with Meta-learning, Ph.D. Thesis, Faculdade de Ciências da Universidade do Porto, Porto, Portugal, 2004.
[35] R.B.C. Prudêncio, T.B. Ludermir, Meta-learning approaches to selecting time series models, Neurocomputing 61 (2004) 121–137.
[36] C. Soares, P.B. Brazdil, P. Kuba, A meta-learning method to select the kernel width in support vector regression, Machine Learning 54 (2004) 195–209, http://dx.doi.org/10.1023/B:MACH.0000015879.28004.9b.
[37] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Deroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognition 45 (9) (2012) 3084–3104, http://dx.doi.org/10.1016/j.patcog.2012.03.004.
[38] J.P.C.L.M. Moreira, Travel Time Prediction for the Planning of Mass Transit Companies: A Machine Learning Approach, Ph.D. Thesis, Faculty of Engineering of University of Porto, 2008.
[39] M. Harries, Splice-2 Comparative Evaluation: Electricity Pricing, Technical Report 9905, School of Computer Science and Engineering, University of New South Wales, 1999.
[40] S. Bajwa, E. Chung, M. Kuwahara, Performance evaluation of an adaptive travel time prediction model, in: Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE, 2005, pp. 1000–1005. http://dx.doi.org/10.1109/ITSC.2005.1520187.
[41] Y. Bin, Y. Zhongzhen, Y. Baozhen, Bus arrival time prediction using support vector machines, Journal of Intelligent Transportation Systems 10 (4) (2006) 151–158, http://dx.doi.org/10.1080/15472450600981009.
[42] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.
[43] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods, Cambridge University Press, New York, NY, USA, 2000.
[44] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees, Chapman & Hall (Wadsworth, Inc.), 1984.
[45] J.H. Friedman, W. Stuetzle, Projection pursuit regression, Journal of the American Statistical Association 76 (376) (1981) 817–823.
[46] J.H. Friedman, Multivariate adaptive regression splines, Annals of Statistics 19 (1) (1991) 1–67.
[47] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2013. URL ⟨http://www.R-project.org/⟩.
[48] R. Caruana, A. Niculescu-Mizil, An empirical comparison of supervised learning algorithms, in: Proceedings of the 23rd International Conference on Machine Learning, ACM, New York, NY, USA, 2006, pp. 161–168. http://dx.doi.org/10.1145/1143844.1143865.
[49] N. Musliu, M. Schwengerer, Algorithm selection for the graph coloring problem, in: Proceedings of the Learning and Intelligent Optimization Conference, Springer, to appear.
[50] T.A. Gomes, R.B. Prudêncio, C. Soares, A.L. Rossi, A. Carvalho, Combining meta-learning and search techniques to select parameters for support vector machines, Neurocomputing 75 (1) (2012) 3–13, http://dx.doi.org/10.1016/j.neucom.2011.07.005.
[51] I. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, San Francisco, 2000.
[52] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, SIGKDD Explorations Newsletter 6 (1) (2004) 20–29, http://dx.doi.org/10.1145/1007730.1007735.
[53] M. Adya, F. Collopy, J.S. Armstrong, M. Kennedy, Automatic identification of time series features for rule-based forecasting, International Journal of Forecasting 17 (2) (2001) 143–157, http://dx.doi.org/10.1016/S0169-2070(01)00079-6.
[54] X. Wang, K. Smith-Miles, R. Hyndman, Rule induction for forecasting method selection: meta-learning the characteristics of univariate time series, Neurocomputing 72 (10–12) (2009) 2581–2594 http://dx.doi.org/10.1016/j.neucom.2008.10.017.
[55] C. Lemke, B. Gabrys, Meta-learning for time series forecasting and forecast combination, Neurocomputing 73 (10–12) (2010) 2006–2016, http://dx.doi.org/10.1016/j.neucom.2009.09.020.
[56] J. Mendes-Moreira, C. Soares, A.M. Jorge, J.F.D. Sousa, Ensemble approaches for regression: a survey, ACM Computing Surveys 45 (1) (2012), http://dx.doi.org/10.1145/2379776.2379786 10:1–10:40.

André Luis Debiaso Rossi received his B.Sc. degree in Computer Science from the Universidade Estadual de Londrina, Brazil, and his M.Sc. degree in Computer Science from the Universidade de São Paulo (USP), Brazil. André Rossi is currently a Ph.D. student at Universidade de São Paulo, Brazil. His main interests are Machine Learning, Data Mining, Data Streams and Evolutionary Computation.

André Carlos Ponce de Leon Ferreira de Carvalho is a Full Professor in the Department of Computer Science, University of São Paulo, Brazil, where he was head of Department from 2008 to 2010. He received his B.Sc. and M.Sc. degrees in Computer Science from the Universidade Federal de Pernambuco, Brazil. He received his Ph.D. degree in Electronic Engineering from the University of Kent, UK. He has published around 80 Journal and 200 Conference refereed papers. He has been involved in the organization of several conferences and journal special issues. His main interests are Machine Learning, Data Mining and Hybrid Intelligent Systems. He is in the editorial board of several journals and was a member of the Brazilian Computing Society, SBC, Council. He was the editor of the SBC/Elsevier textbook series until July 2011. Prof. de Carvalho is the Director of the Research Center for Machine Learning in Data Analysis of the Universidade de Sao Paulo, Brazil.

Carlos Soares received a B.Sc degree (licenciatura) in Systems Engineering and Informatics from Universidade do Minho, Portugal, an M.Sc. degree in Artificial Intelligence and a Ph.D. degree in Computer Science from Universidade do Porto, Portugal. After 15 years at the Faculty of Economics, he is now an Associate Professor at the Faculty of Engineering of Universidade do Porto. Carlos also teaches at PBS (Porto Business School, formerly EGP-UPBS) and he is a post-graduation supervisor at the Universidade de São Paulo. Carlos is a researcher at INESC TEC, with interests on Data Mining, Business Intelligence and Evolutionary Computation. He has participated in more than 20 national and international R&ID as well as consulting projects. Carlos has published/edited several books and more than 40 papers in journals and conferences with proceedings from major publishing houses, of which more than 30 are indexed by ISI. He has participated in the organization of several events, including KDD 2009 and ECML PKDD 2012. He was awarded the Scientific Merit and Excellence Award of the Portuguese AI Association.

Bruno Feres de Souza received his B.Sc. degree in Computer Science from the Federal University of Maranhao (UFMA), Brazil, and his M.Sc. and Ph.D. degrees in Computer Science from University of Sao Paulo (USP), Brazil. He is currently a Post-doctorate fellow at the Department of Computer Science at USP, working on Meta-learning applied to gene expression data analysis. His main research interests include Machine Learning, Meta-learning, Meta-heuristics and Bioinformatics.