2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
October 11-14, 2020. Toronto, Canada

Improving Multiple Time Series Forecasting with Data Stream Mining Algorithms

Marcos Alberto Mochinski
Graduate Program on Informatics, PPGIa,
Escola Politécnica, Pontifícia Universidade
Católica do Paraná, PUCPR
Curitiba, Brazil
mmochinski@ppgia.pucpr.br

Jean Paul Barddal, PhD
Graduate Program on Informatics, PPGIa,
Escola Politécnica, Pontifícia Universidade
Católica do Paraná, PUCPR
Curitiba, Brazil
jean.barddal@ppgia.pucpr.br

Fabrício Enembreck, PhD
Graduate Program on Informatics, PPGIa,
Escola Politécnica, Pontifícia Universidade
Católica do Paraná, PUCPR
Curitiba, Brazil
fabricio@ppgia.pucpr.br

***Abstract*— This paper proposes a hybrid ensemble learning approach that combines statistical and data stream mining algorithms to obtain better forecasting performance in multiple time series prediction problems. Although some multiple time series algorithms perform surprisingly well in a variety of domains, it is well-known that no one is dominant for every existent domain. Therefore, we developed a meta-technique based on data stream mining and static ensemble selection strategy and evaluated its forecasting goodness-of-fit in time series datasets from M3 and M4 competitions. After training different regression models, we show how the combination of auto.arima and AdaGrad leads to improved forecasting rates, thus surpassing the results of state-of-art algorithms.**

Keywords*— *Time series forecasting, data stream mining algorithms, multiple time series, hybrid ensemble.

I. Introduction

Time series are present in our daily life in the economic indicators that are showed in the newspapers, in the sales data charts of different products, in the displayed data of an electroencephalogram, in the price fluctuation in stock exchange, in the census data of a population. Therefore, they can be found in the most different domains of our society. And with an increasing search for more accurate information, machine learning algorithms are put to the test to forecast future events in such series. Classic statistical techniques reach interesting performance in several scenarios of time series forecasting, mainly on those with low dimensionality. However, hybrid models and machine learning solutions also emerge as viable options at an ever-increasing incidence.

Despite their age, statistical models such as ARIMA (Autoregressive Integrated Moving Average) [1], Exponential Smoothing method [2], Theta method [3], among others, are able to achieve performances that surpass more complex and recent techniques, and thus, may also be considered as state-of-the-art approaches.

In this work, we selected time series datasets, extracted from the M3 [4] and M4 [5] competitions, which have forecasting results obtained from the use of statistical techniques, machine learning techniques, and hybrid techniques. We propose a hybrid model that combines statistical learning and data stream mining (DSM) algorithms and show how it can be an alternative to more widespread methods. Data stream mining algorithms are scalable and perform concept drift detection, thus adapting to changes to data distribution on the fly. The aforementioned traits are of utter importance in large-scale multiple time series, and thus, we argue that these should also be recalled in such scenarios. The rationale behind our technique is that data stream mining algorithms can boost the performance of state-of-the-art statistic models because, even though they do not perform very well in some series, they can perform better than statistical methods in other series due to their inherent adaptability. Therefore, if such techniques are correctly combined and selected, one would reach higher forecasting rates.

First, we discuss related works on time series forecasting and data stream mining. Next, we introduce our proposal, which is later analyzed in the following section. Finally, we conclude this paper and list future works.

II. Related Work

A time series is defined as a set of events observed in time at a constant frequency [6]. The record of a store's monthly sales, a city's hourly electricity demand, and the volume of daily access to a particular website, are examples of events that can be measured at constant time intervals and they are of interest for demand prediction.

In this work, we are particularly interested in scenarios where a set of time series is available, regardless of whether these are inter-correlated or not. In such cases, when estimating the demand for a particular resource for a given period in the future, the amount of time series available poses a challenge, as these may scale to thousands of even millions of series, as well as each may have different trends, degrees of seasonality, autocorrelation, spectral entropy [7], and so forth.

Despite the interesting results obtained by classical statistical methods, it is increasingly common to find hybrid approaches and machine learning as alternatives to these methods, or even as approaches that can be combined.

With an increasing volume of data generated by social networks, by the use of sensors, by the diffusion of concepts such as IoT, and Big Data, the use of traditional techniques based on batch information processing may not be efficient for

978-1-7281-8526-2/20/$31.00 ©2020 IEEE

all application areas. It is in this scenario, with even bigger datasets, that Data Stream Mining (DSM) techniques and tools have been developed. The authors in [8] enumerate, among other characteristics, that a stream mining algorithm must process one instance at a time using a limited amount of memory and time for processing, and that it must be able to give a response (as a prediction, or the identification of a pattern) at any time and be adaptable to temporal changes. Therefore, applications demand faster responses and innovative techniques that adapt to the increasingly overloaded world of information in which we live.

According to [9], learning should take place in an incremental and adaptive fashion, thus allowing the reaction to variations in data behavior (concept drifts) and to predict data in an increasingly precise way. It is essential that the algorithms used in time series analysis are able to identify variations of data behavior with greater accuracy so that the forecasting process is more precise. For this characteristic, it is justifiable to seek to apply data stream mining algorithms, which allow gradual, incremental processing of the observations, and which are highly adaptive in the processing of data of this nature (time series). AdaGrad [10] is an example of an adaptive data stream mining algorithm, capable of dealing with very sparse and non-sparse data. According to the authors in [10] AdaGrad has two goals: to generalize the online learning paradigm of specializing an algorithm to fit a particular dataset and to automatically adjust the learning rates for online learning and stochastic gradient descent on a per-feature basis. It is used in this study to check its applicability in time series forecasting problems.

Regarding statistical algorithms for time series forecasting, this work presents the definition of the ARIMA model, and focuses in the use of auto.arima for the experiment. ARIMA is a classic statistical algorithm and is also known as a Box-Jenkins model [1]. According to [11], ARIMA is applicable for short-term forecasting in stationary time series, or series that can be converted into one, and the behavior of their explanatory variables do not change significantly from the past. In ARIMA, the "AR" part (from **A**uto**r**egressive) means that the dependent variable is regressed on data obtained from the past information from the series. The "I" part (from **I**ntegrated) is related to the fact that the data is not stationary, and it is possible to transform them by differencing. Moreover, the final "MA" part of the model (from **M**oving **A**verage) assumes that the dependent variable depends on past errors. Therefore, the use of a weighted moving average expects to identify the errors present in the past data and use them to help in future data.

auto.arima [12][13] is an ARIMA implementation available for the R in the *forecast* package, *auto.arima()* function. It is an implementation that seeks to automatically select the best ARIMA model for a given time series under based on AIC [14], AICc [15], or BIC [16] values calculated during the analysis. In a problem with multiple time series, and to avoid demanding an individual analysis of each series, a solution that automates the selection of the best ARIMA parameters can certainly be of great help in the process. Therefore, this study also seeks to predict multiple time series without the need for a meticulous analysis of each series, especially to make the process more user-independent.

The auto.arima function was selected to this study not only because of its automation characteristic, but also because it presented best overall results specially as the algorithm to create the features used to help in the prediction process by data stream mining methods analysed by the study. The following methods were also considered in preliminary analyses: SES (Simple Exponential Smoothing) [17][18][19], Damped (Holt's linear method with damped trend) [18][20], Random Walk forecast [21], Seasonal Naïve [21], Theta method [3], ETS (Exponential Smoothing State Space Model) [22], STLM (STL decomposition) [23], most of them using their implementation as functions of *forecast* package [12] available for R. *Prophet* [24], a forecasting procedure that proposes also an automatic approach for large-scale forecasting of time series was tested too. However, we opted for auto.arima because it presented better prediction results in preliminary studies than the values calculated by Prophet, using default settings for both algorithms.

Time series forecasting problems are often very well solved by statistical models specialized in forecasting this type of data. However, there is an increasing interest in the application of machine learning techniques and other artificial intelligence techniques in solving this type of problem.

Competitions are frequent in this area, and some well-known examples are the competitions proposed by S. Makridakis as M1, M2, M3 [4], and more recently, M4 [5]. Statistical techniques usually achieve excellent performance in the prediction process, but hybrid techniques have emerged as alternatives to classic methods. Following this rationale, the purpose of this paper is to verify whether data stream mining algorithms can also be an alternative to, or combined with, traditional statistical models in time series forecasting. The use of M3 and M4 datasets was chosen mainly because, they contain a great variety of series, selected from different domains (microeconomy, macroeconomy, industry, finance, demographic). M3 dataset was also analysed in an interesting study by Ahmed et al. [25] using a subset of 1045 series in a machine learning forecasting approach. M4 competition, in turn, offered a dataset with 100,000 time series, and in this study different sets of its 48,000 monthly series were considered. Besides that, in the M3 and M4 datasets each series has a small set of observations, and in spite of classic DSM applications usually are more focused in dealing with problems that involve great number of records, this work is interested in evaluate its applicability in the prediction of typical time series scenarios using the monthly series of both of the competitions.

III. Method

Studies on the use of machine learning techniques, statistical models and hybrid techniques for the forecasting of time series are common but the application of data stream mining algorithms in solving this problem is practically unobserved. As multiple time series problems usually involve the use of large datasets, using DSM techniques seems reasonable. It would be possible to consider working with multiple series grouped into a single dataset [26]. For this study, however, to make a more didactic comparison of models, we opted for an individual analysis approach of each series.

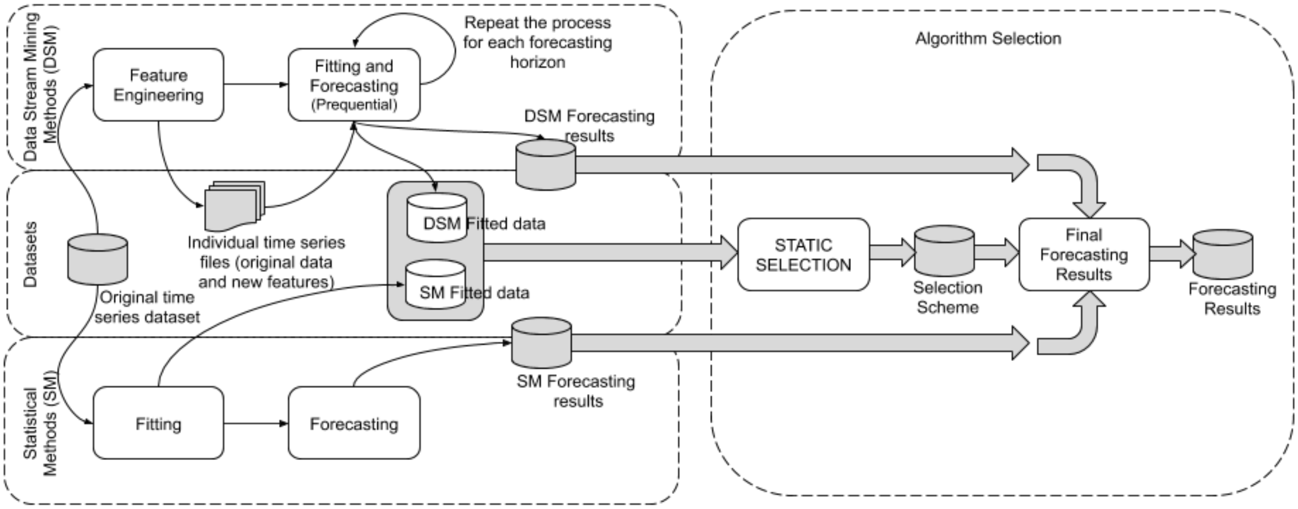

Data Stream Mining Methods (DSM)
Feature Engineering
Fitting and Forecasting (Prequential)
Repeat the process for each forecasting horizon
DSM Forecasting results
Datasets
Individual time series files (original data and new features)
DSM Fitted data
SM Fitted data
Original time series dataset
Statistical Methods (SM)
Fitting
Forecasting
SM Forecasting results
Algorithm Selection
STATIC SELECTION
Selection Scheme
Final Forecasting Results
Forecasting Results

Fig. 1 - Hybrid ensemble learning approach.

In summary, the method proposed in this paper suggests the use of information obtained in the training of different algorithms in a selection procedure of the algorithm that will probably present the best result in the forecast of each time series of the set. Fig. 1 presents a diagram of the method proposed in this study, and in this section the description for each step of the process is detailed.

Fig. 2 represents an example of the proposed method applied to a specific time series from the M3 dataset selected for the study. Its main purpose is to present the simplicity of the method and to show how the behavior of fitting and testing data of auto.arima and AdaGrad (the algorithms of the ensemble) differ from each other. The figure shows a curve with the original data of one series of the M3 dataset, the N1738 series, and the values obtained in the training and forecasting with the methods AdaGrad and auto.arima. The adaptive behavior of AdaGrad is observed, adjusting to the original data of the series as observations are processed.

When we calculate the sMAPE (Symmetric Mean Average Percentage Error) [27] for the 40% final observations of the traing data of the series with each algorithm we obtain: $\text{sMAPE}_{\text{auto.arima(FIT)}} = 0.2618$, e $\text{sMAPE}_{\text{AdaGrad(FIT)}} = 0.2581$. The method proposes that, for the N1738 series, the algorithm to be used for the final forecasting is the AdaGrad. The last 18 points of Fig. 2 shows the forecasts of both algorithms confirming that the choice (selection) based on the training error was a good option for the series.

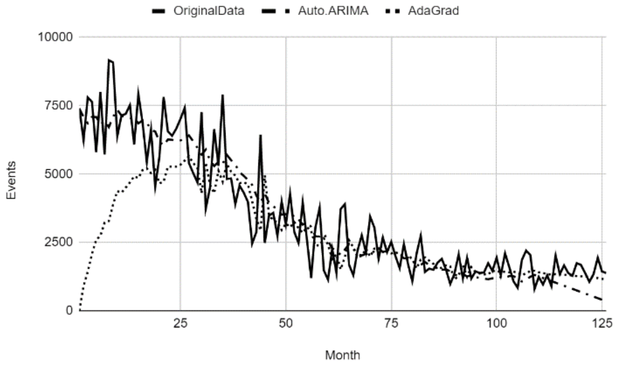
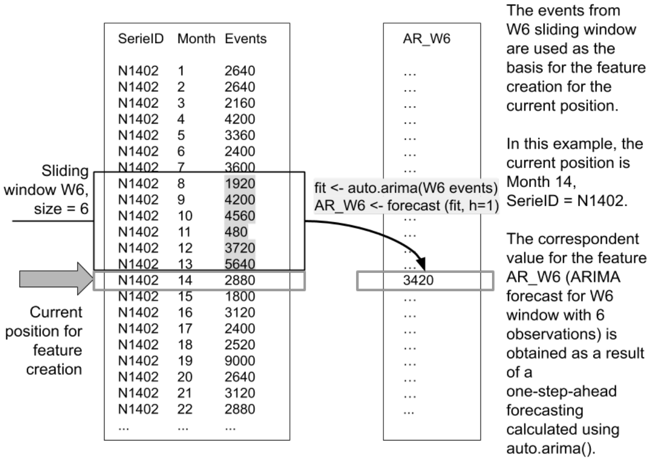


Fig. 2 - Original data from the N1738 time series and values obtained in the training and tests by AdaGrad and auto.arima.

A. Steps for Data Stream Mining Methods (DSM)

Feature Engineering:

In this step, features that assist the regression process with data stream algorithms are created. Initially, the dataset of each time series contains original series information as the series identifier and the total events for each month. Upon this information several features were derived. In practice, lags, differences (diffs) and moving averages were extracted from the time series original data. Considering that DSM algorithms can continuously perform the test and training steps, the central idea in this stage was to use only data from events before the observation for which the features are being created to avoid



SerieID	Month	Events	AR_W6
N1402	1	2640	…
N1402	2	2640	…
N1402	3	2160	…
N1402	4	4200	…
N1402	5	3360	…
N1402	6	2400	…
N1402	7	3600	…
N1402	8	1920	
N1402	9	4200	
N1402	10	4560	…
N1402	11	480	…
N1402	12	3720	…
N1402	13	5640	…
N1402	14	2880	3420
N1402	15	1800	…
N1402	16	3120	…
N1402	17	2400	…
N1402	18	2520	…
N1402	19	9000	…
N1402	20	2640	…
N1402	21	3120	…
N1402	22	2880	…
…	…	…	

Sliding window W6, size = 6

Current position for feature creation

```
fit <- auto.arima(W6 events)
AR_W6 <- forecast (fit, h=1)
```

The events from W6 sliding window are used as the basis for the feature creation for the current position.

In this example, the current position is Month 14, SerieID = N1402.

The correspondent value for the feature AR_W6 (ARIMA forecast for W6 window with 6 observations) is obtained as a result of a one-step-ahead forecasting calculated using auto.arima().

Fig. 3 - Example of a 6-instance sliding window to select data for creating features.

data leakage. For instance, given a series with 100 observations, to create the additional features of the series for the 14th month, only data from the previous 13 months were used.

Additional features were created using a sliding window (Fig. 3), so there is no need for a complete analysis of all the data from each time series since features are created incrementally, instance to instance, without having to work with a significant amount of past data. These features include:

- auto.arima forecasts for windows with size 6 (AR_W6), 12 (AR_W12), 18 (AR_W18), and 24 (AR_W24).
- Errors calculated from predictions made with auto.arima in sliding windows from 6 to 24 months.
- Values adjusted based on calculated errors.
- auto.arima forecasts for 24-instance windows for horizons from 1 to 18 months (AR_W24_fc1, AR_W24_fc2…AR_W24_fc18).
- original events and date attributes (year, month, bimester, quarter, semester).

After the creation of features, individual files with data from each of the series were generated to enable their individual processing. At this stage, which can be considered as a stage of data preprocessing, it is essential to note that no additional normalization, deseasonalization, nor trend analysis was used. Therefore, it is expected to verify if the problem with multiple time series can be solved by giving up those steps that are typically used for the processing of this type of series. The main reason for opting for this approach is to avoid (or minimize) batch analysis of the entire set of data in stages where this is not essential. It is an important decision to make viable the future use of the online data analysis approach.

Fitting and Forecasting (Prequential):

In this step, MOA [28] was called iteratively (using an R-language program) to perform the training and one-step-ahead forecast of the individual files created for each series to obtain forecasts for the horizon of 1 to 18 months (H1 to H18). For this stage, we used the Prequential process that implements the test-then-train validation scheme supported by the regression algorithms provided in MOA. In this model, for each record, the software makes a prediction, compares it with the actual value, and adapts the behavior of the algorithm based on the prediction error.

After processing the training output logs of each series (fitting data), the algorithm predicts a value for the next month (one-step-ahead forecasting). Predictions calculated by AdaGrad are stored in output files created by MOA. Next, a program in R reads the output file to obtain the calculated forecast value, reinserts this predicted value as a data point of the series, makes the generation of features for the next month in question, and processes the file again in the MOA, to obtain the 18 forecasts expected by the process. It is an iterative process of generating one-step-ahead forecasts.

Training sMAPE values (based on fitting values of the 40% final training records of each series) and forecasting are calculated and stored in a database for later use in the process of static selection and definition of final forecasting. The fitting process was performed with several regression algorithms available in MOA, such as AdaGrad [10], AmRules Regressor [29], ARF-REG [30], FIMT-DD [31], ORTO [32], and RandomRules [28], RandomAMRules [33]. However, this study presents only the values obtained with AdaGrad, which refers to the algorithm that presented a better performance in previous stages of the study.

B. *Steps for Statistical Methods (SM)*

Fitting:

In this step, auto.arima training values are obtained for each time series. For model fitting with auto.arima, no additional features were required, since only the original data of the series (Month and Events) were used. Once the sMAPE training calculations have been performed (based on fitting values of the 40% final training records of each series), the data is stored in a database for use in the static selection process.

Forecasting:

After performing the training with the *auto.arima()* method, the *forecast()* method was used to generate the prediction of 18 months of each time series. The forecast values and sMAPE values obtained in this step are stored in a database for use in the final forecasting selection process.

C. *Static Selection*

The static selection process considers the sMAPE values obtained during the training phase of each series for the AdaGrad and auto.arima algorithms and identifies for each series which algorithm obtained the least error. From this analysis, a Selection Scheme is generated, and it stores data of the series and the respective values used in the selection, among them, the value of sMAPE taken as reference. Thus, the selection is based on simple comparison of errors (sMAPE) calculated after the fitting process. The algorithm that generates the least error will be considered in the final forecasting results for the series.

D. *Final Forecasting Results*

In this step, the forecast values for the horizons of 1 to 18 months are defined based on the algorithms established for each series by the static selection process. Based on the sMAPE values obtained in the training phase, we select the method that will define the final forecasting values assigned to each series. This step concludes the process, and the forecasting of all the series from the dataset is defined based on the algorithms that presented the least error during the training process.

IV. Experiment

The experiment described in this section consists of verifying whether the combined use of statistical forecasting techniques and data stream mining algorithms yields better forecasting values for an 18 months horizon than those achieved by isolated use of state-of-the-art algorithms.

For a proof of concept, we chose to use, for the main experiment, a dataset of time series used by Ahmed et al. [25]

to validate the performance of machine learning algorithms in the prediction of monthly series of the M3 competition. This set represents a subset of time series proposed by S. Makridakis in its M3 competition[1] [4], and was also evaluated by [34] in a comparison between statistical methods and machine learning algorithms evaluated by Ahmed et al. The original set of monthly time series in M3 competition includes 1428 series. However, in a similar way to that used by [25] and [34], only the series with more than 80 observations for training were selected, totalizing 1045 series.

The authors in [25] bring forward a comparative study of the performance of different machine learning methods (Multilayer Perceptron, Bayesian Neural Network, Generalized Regression Neural Network, K-Nearest Neighbor Regression, Classification & Regression Trees, Support Vector Regression, Gaussian Process) in the processing of these series, and [34] makes an additional analysis comparing the results obtained by the methods of machine learning in comparison to the statistical methods.

Additional experiments were done using three different subsets of M4 time series (with 5,000, 10,000 and all of the its 48,000 monthly series) in order to validate if the method created based on the 1,045 series of M3 would be applicable to different sets of time series.

In this study, we attempted to add a new method in this forecasting scenario, combining a statistical method with a DSM algorithm to verify the possibility of improving the prediction accuracy of this set of series.

In opposition to what is usually proposed for working with time series, in this work, the observations of each time series were considered without making use of data normalization, neither trend nor seasonality analysis. In this way, we try to eliminate the need for an individualized analysis of each series, especially for the work with DSM algorithms. However, a work with feature engineering was carried out to increase the set of original features of each series and improve the prediction capacity of this type of algorithms.

A. *Algorithms and Tools*

The statistical algorithm selected for the study, *auto.arima* [12][13], available as a function of the *forecast* package of R, was chosen for its approach of working automatically, allowing its use without interference by the user, although it offers the possibility of parameter customization.

Regarding data stream mining algorithms, more specifically about regression algorithms, AdaGrad [10], a gradient descent optimization algorithm, was selected after preliminary studies that included the following methods: AdaGrad [10], AmRules Regressor [29], ARF-REG [30], FIMT-DD [31], ORTO [32], Random AMRules [33], RandomRules (class moa.classifiers.meta.RandomRules in MOA [28]). They are available in software MOA[2] (Massive Online Analysis) [28], a platform for data stream mining. AdaGrad was selected because of its better results in preliminary one-step-ahead forecasts experiments using the cited algorithms.

For the execution of the AdaGrad algorithm in MOA, the following main parameters were used:

- Task: EvaluatePrequentialRegression
- Learner: AdaGrad, default parameters: epsilon = 0; lambdaRegularization = 0; learningRate = 0.01; lossFunction = HINGE.
- Evaluator: BasicRegressionPerformanceEvaluator

The experiments described in this work were performed in a MacOS High Sierra 10.13.2 operating system, using as tools:

- Programming Language: R (version 3.4.4)
- IDE: RStudio (version 1.0.153)
- MOA (Massive Online Analysis) version 2019.04.01
- *Metrics* package [35]: version 0.1.4
- *forecast* package [12]: version 8.5

B. *Evaluation Protocol*

sMAPE, the acronym for Symmetric Mean Average Percentage Error, also known as symmetric MAPE, is the primary metric used in this work and it was earliest presented in [27]. It was also used by the M3 and M4 competitions among other metrics, so this influenced our choice for the use of it in our experiment.

In [34] the authors presented the definition (1) for the sMAPE calculation. The final sMAPE is defined as the average error of all forecasts for all the horizons:

$$sMAPE = \frac{2}{k}\sum_{t=1}^{k}\frac{|Y_t - \hat{Y}_t|}{|Y_t| + |\hat{Y}_t|} * 100\% \qquad (1)$$

where k is the forecasting horizon, Y_t the actual values, and $\hat{Y}_t$ the forecasts for a specific time t.

For this work, the sMAPE calculation was made using the function *smape()* from the package *Metrics* [35] for the R language. After calculating the sMAPE for each time series and for each horizon, an average sMAPE of all the series was obtained, establishing the percentage error for all of the time series set.

V. Static Selection

Analyzing the results obtained during preliminary tests of the algorithms, in which the values of sMAPE obtained for one-step-ahead forecast were evaluated, it was observed that, in 426 series of the total of 1045 series (i.e., in 40.8% of the series used by the main experiment) AdaGrad presented smaller prediction errors compared to auto.arima.

This performance suggested the possibility that by combining auto.arima and AdaGrad, a better overall accuracy could be obtained in terms of forecasting. Given this motivation, it was necessary to establish a way to select the best method for each series, and the analysis of errors obtained in the training phase could be one of the approaches.

Before choosing the static selection based on the training data of 40% of the final observations of each series, other

[1] https://forecasters.org/resources/time-series-data/m3-competition/
[2] https://moa.cms.waikato.ac.nz/

analyses were carried out: 1) Tests using separate datasets for training and validation, in a ratio of 80/20. For this case, forecasts were made and compared against the separate data for validation, resulting in the calculation of sMAPE for these data; 2) Analysis of sMAPE calculated on partitions with different sizes (last 6 events, last 12 events, last 20%, last 80% and 100%) of the training data were also performed. The results of these analysis are presented in Table I.

VI. Results

The proposed method selects the best algorithm for each time series based on the sMAPE value obtained during the training (fitting) step. Before we conclude that the best results for the selection should consider the sMAPE obtained with 40% of the final records of the training data set, other experiments were performed, and their results are described in the Table I, which presents the forecasting values of 1 to 18 months obtained using different selection criteria. The first column of the table presents the results obtained using a validation dataset (a subset of 20% of training dataset). The remaining columns show the results of forecasting considering different parts of the training dataset, among other combinations. It is observed that the use of 40% of the final records in a combination of AdaGrad + auto.arima shows the highest accuracy, with the lowest value of average sMAPE (11.83% = 0.1183). From these results, we made a comparison with the data obtained by the individual use of the methods (auto.arima and AdaGrad). Table II presents the forecasting performance obtained by the methods used in this study compared to the sMAPE values calculated for auto.arima (considered the benchmark). For each method are presented the gain values obtained for the horizons from 1 to 18 months ahead. It can be observed that the ensemble (the combination of auto.arima and AdaGrad) presents positive gain values for the most of the 18 horizons, varying from 0.3 to 3.5% of gain when compared to the use of isolated auto.arima. On the other hand, the isolated use of AdaGrad cannot surpass the performance presented by auto.arima.

Table III presents the average sMAPE obtained by auto.arima for short-term (average for the first six months), medium-term (average for results from the 7th to 12th months) and long-term (average obtained for the last six months) horizons, and shows the gain of the ensemble compared to the isolated use of auto.arima. It is possible to notice that the ensemble presented positive gain for all of the three intervals. The results suggest that the best results are reached by combining auto.arima and AdaGrad as proposed.

Table I. sMAPE for the Forecasting (1 to 18 months) Calculated by the Use of Different Criteria to Select the Algorithms Based on Fitting and Validation Data (1045 Time Series from M3).

	Selection based on validation data	Selection based on fitting data						
H	AA Minimum	AA Complete Series	AA 40 percent	AA 6 regs	AA 12regs	AA 20 percent	AA Average	AA Minimum
1	8,24	8.09	8.04	8,38	8,29	8,08	8,24	8,21
2	8.97	8.78	8.65	8.79	8.76	8.64	8.66	8.75
3	9.71	9.79	9.57	10.01	9.68	9.67	9.91	9.91
4	10.51	10.59	10.47	10.82	10.61	10.49	10.73	10.69
5	10.21	10.08	9.81	10.07	9.93	9.93	10.05	9.97
6	10.25	9.90	9.76	9.84	9.66	9.88	9.81	9.75
7	10.84	10.54	10.28	10.46	10.35	10.44	10.39	10.37
8	10.92	10.59	10.34	10.39	10.41	10.44	10.39	10.19
9	11.6	11.08	11.04	11.18	11.06	11.09	10.98	10.95
10	12.47	11.46	11.74	11.74	11.86	11.76	11.67	11.65
11	11.64	10.96	10.90	11.09	11.01	11.03	11.04	10.91
12	12.41	11.72	11.69	12.11	11.86	11.94	11.93	12.03
13	13.6	12.48	12.36	12.75	12.42	12.55	12.55	12.56
14	15.01	14.42	14.17	14.81	14.14	14.23	14.55	14.57
15	16.57	16.08	15.55	16.67	15.83	15.70	16.20	16.18
16	17.39	16.56	16.01	17.17	16.22	16.26	16.65	16.84
17	18.02	16.57	16.26	17.71	16.64	16.52	17.07	17.25
18	18.44	16.45	16.25	17.67	16.92	16.67	16.91	17.17
AVG	12,60	12,01	**11,83**	12,31	11,98	11,96	12,10	12,11

Legend: AA=AdaGrad+auto.arima; AVG=Average sMAPE for all the 18 months

In additional experiments with 5,000, 10,000 and 48,000 monthly series of M4 (Table IV), it was observed that for closer horizons (short-term), especially for the 1st to the 4th month interval, the method proved to be valid, since the results present gain when compared to the values obtained by the isolated use of auto.arima (benchmark). This suggests that the diversity of characteristics inherent to each series in the set can influence the gain obtained by the proposed method. An additional technique that can be considered in future studies is to identify features of the series such as linearity, tendency, curvature, autocorrelations, among others, and to use such inherent properties to help the algorithm selection process in the ensemble, such as already explored by Montero-Manso et al. [36][37]. Another interesting perception obtained from the results is that, although the individual results presented by AdaGrad show negative gains in relation to auto.arima, using

Table II. Comparison of Forecasting Performance Obtained by the Use of Different Methods on 1045 Series of M3 Dataset. sMAPE Values for auto.arima are Used as Benchmark in Order to Calculate the Gain Obtained by the Ensemble

	H1	**H2**	**H3**	**H4**	**H5**	**H6**	**H7**	**H8**	**H9**	**H10**	**H11**	**H12**	**H13**	**H14**	**H15**	**H16**	**H17**	**H18**
auto.arima (sMAPE, benchmark)	8.02	8.85	9.85	10.62	10.12	9.97	10.61	10.67	11.08	11.45	10.93	11.65	12.40	14.35	16.00	16.59	16.60	16.53
	Gain Ratio (%)																	
AdaGrad	-21.4	-16.8	-15.3	-20.6	-12.2	-11.9	-16.1	-14.4	-18.7	-37.9	-31.6	-28.6	-41.0	-29.6	-27.3	-32.0	-39.8	-49.2
Ensemble	-0.2	**2.3**	**2.8**	**1.4**	**3.1**	**2.1**	**3.1**	**3.1**	**0.4**	-2.5	**0.3**	-0.3	**0.3**	**1.3**	**2.8**	**3.5**	**2.0**	**1.7**

TABLE III. COMPARISON OF FORECASTING PERFORMANCE FOR SHORT, MEDIUM AND LONG TERM. sMAPE VALUES FOR AUTO.ARIMA ARE USED AS BENCHMARK IN ORDER TO CALCULATE THE GAIN OBTAINED BY THE ENSEMBLE

	Short-term (1 to 6 months)	**Medium-term (7 to 12 months)**	**Long-term (13 to 18 months)**
auto.arima (sMAPE, benchmark)	9.57	11.06	15.41
	Gain Ratio (%)		
AdaGrad	-16.30	-24.86	-36.53
Ensemble	**1.88**	**0.54**	**2.01**

AdaGrad combined to auto.arima (the ensemble) is capable to reach positive gains for the set of series, even in some prediction horizons. This demonstrates that, once the ideal series are assigned to AdaGrad, the ensemble results are better for the series in the set.

VII. DISCUSSION

A. *Impact of the Training Window Size*

Table I shows that the performance of the static selection made based on values obtained by the algorithms during the training phase is highly dependent on the number of records taken into consideration for the calculation of sMAPE. Taking all of the training data into consideration, or a minimal set (such as the 6 or 12 final records) did not have a good result as the option for calculation based on 40% of the records. This suggests that the performance of the algorithms is best represented by that portion of the data, and that the analysis of that region of the time series is enough for a satisfactory prediction performance.

B. *Ensemble versus Isolated Algorithms*

In this paper, we presented the results obtained by the isolated use of AdaGrad and auto.arima, and comparing them with the results achieved by the combination of AdaGrad and auto.arima (ensemble), and it can be observed (for the main experiment) that the combined use of the methods is more accurate in 15 of the 18 observations. For an instant horizon (one-step-ahead forecast, H1), and for H10 and H12, the isolated use of auto.arima presents better results than the hybrid solution. However, by comparing the mean values obtained for short-term, medium-term, and long-term (Table III), it can be observed that the combined use can overcome the use of a classic method.

Based on the results it can be stated that the hypothesis "Data stream mining (DSM) algorithms achieve better results in time series forecasting if used together with state-of-the-art statistical algorithms" was confirmed, and that the combined use may lead to an improvement in the performance obtained since the results obtained after static selection were better than those achieved by the isolated use of the algorithms.

VIII. CONCLUSION

Data stream mining techniques are increasingly widespread in scenarios where information volume is increasing as in areas such as social networks, the universe of IoT and Big Data. Forecasting in this kind of scenario, where it is not feasible to consider the use of batch processes because it is inconceivable to have all the data available for analysis, or where forecasting of future values in real-time is expected, suggests that new technologies must be constantly sought. It is expected that the experiment presented in this study may serve as a basis for future research, since the universe of data has a dynamic behavior and demands constant research in order to present more and more real-time answers, with greater accuracy, without dependence of total availability of historical data for predictions in the most diverse possible scenarios. The datasets used in this experiment were used in a didactic purpose and they were selected to serve as a guide for future works in which massive databases will be selected, as well as different algorithm selection schemes. The use of dynamic selection, for example, suggest more reliable and innovative approaches to a universe in which data series variance is high. The search for a short set of features must also be done to reduce the efforts made during steps such as data preprocessing. That is, it is an area where research can undoubtedly present positive results. For future works, it is reserved to extend the study to all of the 100,000 time series available in the M4 competition, and to use different base forecasting algorithms as well. Besides that, the use of feature extraction and meta-learning techniques [36][37], should be explored to verify how inherent properties can improve the algorithm selection step of the method proposed in this study.

TABLE IV. COMPARISON OF FORECASTING PERFORMANCE OBTAINED BY THE USE OF DIFFERENT METHODS ON THREE DIFFERENT SETS OF M4 DATASET (5,000, 10,000 AND 48,000 SERIES).

5,000 series	**H1**	**H2**	**H3**	**H4**	**H5**	**H6**
auto.arima (sMAPE, benchmark)	6.91	7.79	9.49	10.76	11.39	12.2
	Gain Ratio (%)					
AdaGrad	-15.0	-16.4	-12.4	-16.5	-17.4	-18.6
Ensemble	**1.2**	**2.0**	**1.1**	**0.1**	-0.3	**0.2**

10,000 series	**H1**	**H2**	**H3**	**H4**	**H5**	**H6**
auto.arima (sMAPE, benchmark)	6.82	7.68	9.38	10.96	11.45	11.92
	Gain Ratio (%)					
AdaGrad	-15.6	-18.0	-15.0	-15.1	-18.6	-22.4
Ensemble	**0.7**	**0.3**	**0.2**	**0.2**	-0.2	-0.6

48,000 series	**H1**	**H2**	**H3**	**H4**	**H5**	**H6**
auto.arima (sMAPE, benchmark)	6.62	7.66	9.33	10.68	11.44	11.87
	Gain Ratio (%)					
AdaGrad	-18.1	-19.7	-15.7	-16.1	-17.9	-22.0
Ensemble	**0.5**	**0.4**	**0.1**	0	-0.1	-0.4

REFERENCES

[1] G. E. P. Box and G. M. Jenkins, “Time series analysis: Forecasting and control,” San Francisco: Holden-Day, 1970.

[2] R. G. Brown, "Exponential Smoothing for Predicting Demand," 1956, Cambridge, Massachusetts: Arthur D. Little, Inc. p. 15.

[3] V. Assimakopoulos and K. Nikolopoulos,“The theta model: a decomposition approach to forecasting,” International Journal of Forecasting, 16(4), 521–530, 2000.

[4] S. Makridakis and M. Hibon, “The M3-competition: Results, conclusions and implications,” International Journal of Forecasting 16(4):451-476, 2000. DOI: 10.1016/S0169-2070(00)00057-1

[5] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, “The M4 Competition: Results, findings, conclusion and way forward”. Int. J. Forecast., vol. 34, no. 4, pp. 802–808, Oct. 2018.

[6] S. Makridakis, “A Survey of Time Series”, 1976, Int. Stat. Rev. / Longman Group Ltd. Vol. 44. http://hdl.handle.net/11728/6326.

[7] T. S. Talagala, R. J. Hyndman, and G. Athanasopoulos, “Meta-learning how to forecast time series,” 2018. Department of Econometrics and Business Statistics, Monash University. ISSN 1440-771X

[8] A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer, Machine Learning for Data Streams with Practical Examples in MOA. MIT Press, 2018.

[9] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation”, ACM Comput. Surv., vol. 46, no. 4, pp. 1–37, Mar. 2014.

[10] J. Duchi and Y. Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization”. 2011.

[11] C. L. Jain, “Fundamentals of Demand Planning and Forecasting”. St. John’s University. Graceway Publishing Company, Inc. 2017. ISBN 978-0-9839413-2-3

[12] R. J. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang and F. Yasmeen. 2019. “forecast: Forecasting functions for time series and linear models”. R package version 8.7, <URL: http://pkg.robjhyndman.com/forecast>.

[13] R. J. Hyndman and Y. Khandakar, “Automatic time series forecasting: the forecast package for R.”. Journal of Statistical Software,*26*(3), 1-22, 2008, http://www.jstatsoft.org/article/view/v027i03.

[14] H. Akaike, "A new look at the statistical model identification," in IEEE Transactions on Automatic Control, vol. 19, no. 6, pp. 716-723, December 1974.doi: 10.1109/TAC.1974.1100705

[15] C. M. Hurvich and C.L. Tsai, “Regression and time series model selection in small samples”, 1989, Biometrika 76, 297-307

[16] G. E. Schwarz, "Estimating the dimension of a model", 1978, Annals of Statistics, 6 (2): 461–464, doi:10.1214/aos/1176344136, MR 0468014.

[17] R. G. Brown. "Statistical Forecasting for Inventory Control". [S.l.]: McGraw-Hill, 1959.

[18] C. C. Holt. "Forecasting seasonals and trends by exponentially weighted moving averages". Office of Naval Reseach (ONR) Memorandum No. 52, Carnegie Institute of Technology, Graduate school of Industrial Administration, Pittsburgh, 1957.

[19] P. R. Winters. "Forecasting sales by exponentially weighted moving averages". Management Science, v. 6, n. 3, p. 324–342, 1960.

[20] E. S. Gardner and Ed. Mckenzie. "Forecasting trends in time series". Manage. Sci., INFORMS, Linthicum, MD, USA, v. 31, n. 10, p. 1237–1246, out. 1985. ISSN 0025-1909. Available in: <https://doi.org/10.1287/mnsc.31.10.1237>

[21] R. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang and F. Yasmeen. "rwf - Naive and Random Walk Forecasts". 2019.

[22] R. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang and F. Yasmeen. "ets - Exponential Smoothing State Space Model". 2019.

[23] R. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang and F. Yasmeen. "forecast.stl - Forecasting Using Stl Objects". 2019. Available in:<https://www.rdocumentation.org/packages/forecast/versions/8.5/topics/forecast.stl>.

[24] S. J. Taylor and B. Letham, “Forecasting at scale”. PeerJ Preprints 5:e3190v2, 2017, https://doi.org/10.7287/peerj.preprints.3190v2

[25] N. K. Ahmed, A. F. Atiya, N. El Gayar, and H. El-Shishiny. “An Empirical Comparison of Machine Learning Models for Time Series Forecasting”. Econom. Rev., vol. 29, no. 5–6, pp. 594–621, Aug. 2010.

[26] M. Filho, “How to Predict Multiple Time Series with Scikit-Learn (With a Sales Forecasting Example)” [Online]. Available: http://mariofilho.com/how-to-predict-multiple-time-series-with-scikit-learn-with-sales-forecasting-example/. [Accessed: 28-Dec-2018]

[27] J. S. Armstrong, “Long-range Forecasting: From Crystal Ball to Computer”, 1985, 2nd. ed. Wiley. ISBN 978-0-471-82260-8

[28] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, “MOA: Massive Online Analysis”, 2010, Journal of Machine Learning Research 11 (2010) 1601-1604

[29] E. Almeida, C. Ferreira, and J. Gama, “Adaptive Model Rules from Data Streams” in ECML PKDD 2013: Machine Learning and Knowledge Discovery in Databases, Springer, Berlin, Heidelberg, 2013, pp. 480–492.

[30] H. M. Gomes, J. Paul Barddal, L. E. Boiko, and A. Bifet, “Adaptive random forests for data stream regression” in ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 2018.

[31] E. Ikonomovska, J. Gama, S. Džeroski, J. Gama, and E. Ikonomovska, “Learning model trees from evolving data streams”. Data Min Knowl Disc. DOI 10.1007/s10618-010-0201-y, 2010.

[32] E. Ikonomovska, J. Gama, B. Ženko, and S. Džeroski, “Speeding-up Hoeffding-based regression trees with options” in: Proceedings of 28th International Conference on Machine Learning, ACM, 2011, pp. 537–544.

[33] J. Duarte, J. Gama, W. Fan, A. Bifet, Q. Yang, and P. Yu, “Ensembles of Adaptive Model Rules from High-Speed Data Streams”. 2014.

[34] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, “Statistical and Machine Learning forecasting methods: Concerns and ways forward”. 2018. PLOS ONE 13(3): e0194889. https://doi.org/10.1371/journal.pone.0194889

[35] B. Hamner and M. Frasco, “Metrics: Evaluation Metrics for Machine Learning”. R package version 0.1.4. https://CRAN.R-project.org/package=Metrics. 2018.

[36] P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman and T. S. Talagala. "FFORMA: Feature-based forecast model averaging", 2018.

[37] P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman and T. S. Talagala. "M4metalearning". University of A Coruña (Spain) and Monash University (Australia and Sri Lanka), 2018. Available in: <https://github.com/Mcompetitions/M4-methods/blob/master/245-pmontman/M4-Method-Description.pdf>.