Adaptive Fast XGBoost for Regression

Fernanda Maria de Souza[1], Julia Grando[1], and Fabiano Baldo[1]

Universidade do Estado de Santa Catarina (UDESC), Joinville - SC, Brazil
{fernandamsouza833, juliagrando}@gmail.com, fabiano.baldo@udesc.br

Abstract. The increasing generation of data by devices, people and systems arises the need for processing non-stationary data streams, which continuously change over time. It was noticed that when compared to data stream classification, there is a lack of data stream regression studies. This work proposes AFXGBReg-D, an Adaptive Fast regression algorithm using XGBoost and active concept drift detectors. AFXGBReg uses an alternate model training strategy to achieve lean models adapted to concept drift, combined with a set of drift detector algorithms: ADWIN, KSWIN and DMM. We compared two AFXGBReg variants with other regressors and data stream regressors, simulating using synthetic datasets with different kinds of concept drifts. We show that AFXGBReg models have similar MSE to ARFReg, with these models achieving the best performance than others as proven statistically. Also AFXGBReg is 33 times faster than ARFReg, meaning that it is able to keep the same MSE level while being much faster. Another improvement is its ability of doing a faster recovery from concept drifts, having a smaller MSE peak.

Keywords: Data stream regression · Concept Drift · XGBoost.

1 Introduction

Currently, with the evolution of cloud storage technologies, the improvement of communications infrastructure and the popularization of mobile devices, data is continuously produced on an increasing scale. With this abundance of data, the concept of *Big Data* emerged, whose data science tries to extract potential benefits from its analysis for society [13]. The name *Big Data* aims to describe the exponential growth of data based on five principles: i) the volume of data, ii) the variability of data types, iii) the speed at which data is generated, captured and processed , iv) its veracity, in order to obtain true data, according to reality and v) the value, the costs involved in this operation and the added value of all this work [12].

Considering the potential benefits of extracting patterns and relevant information from large volumes of data, the concept of data mining was consolidated within data science. In it, using machine learning algorithms, it is possible to explore a set of data in order to establish relationships that are difficult to visualize manually.

In the context of machine learning, the vast majority of research focuses on building static models, which are trained once on a training dataset and then

applied to the analysis of new data [9]. However, the increasing generation of data by sensors, IoT (Internet of Things) devices, computer network traffic, telephone conversations and financial transactions, arises the demand for processing non-stationary data streams, which continuously change over time. Data streams can be defined as sequences of non-stationary data generated constantly and indefinitely [26]. These characteristics mean that mining algorithms cannot store the data and that they have to deal with changes in concepts inherent to the non-stationary characteristic of streams [11].

Thus, data stream mining deals with challenges on the following dimensions: i) speed: data is processed in limited time; ii) memory efficiency: previously processed data needs to be discarded; iii) non-iterative: data is processed only once; and iv) adaptive: the learning model must be adjusted to concept drifts [16].

As on classification problems, in the context of data stream regression the predictive model needs to adapt to changes in concept. Therefore, for dynamically changing data, traditional supervised learning methods are inappropriate and can cause loss of model performance. To try to maintain the best possible precision, the trained model needs to adapt using new input data. Therefore, predictive models need to be trained incrementally, either by continuous updating or by retraining using recent batches of data [8].

Although there are some proposed approaches to data stream regression, when compared to data stream classification studies, it was not yet much explored by the scientific community, as it is considered a more complex problem and therefore requires more effort to solve [26].

In machine learning, a concept drift means that the statistical properties of the target variable, which the model is trying to predict, change over time in an unforeseen way [14]. Therefore, this results in problems in the predictions as the model becomes less accurate over time. Concept drift detection methods for adaptive machine learning can be classified into two main categories: passive and active. In passive approaches, learning models are updated without prior detection of the concept drift [6], therefore, they tend to be slower in detecting the change. In active mode, approaches are focused on first detecting the deviation/change of concept and, later, accelerating the process of updating the model so that the change is adsorbed by the model as soon as possible, making the detection process faster than in the passive one [7].

To deal with concept drifts in non-stationary data, passive approaches are able to adapt with continuous updates, but require more computational efforts because they are slower [19]. On the other hand, even though active approaches tend to be faster, they introduce to the learning process one more task to be performed, which is the execution of the drift detection algorithm. In this work, the use of active concept drift detection will be investigated in order to explore its potential benefits and limitations in relation to passive concept drift detection in data stream regression.

Among the studies on classification and regression of data streams, the ensemble of models stands out as the most robust and accurate. Looking specifi-

cally at classifier ensembles, boosting algorithms are a promising approach, as they are able to produce an accurate general model from the combination of several moderately imprecise models [23]. The basic idea of these algorithms is to iteratively apply classifiers and combine their solutions to get a better predictive result [18]. Two of its most famous algorithms are *AdaBoost* and *Gradient Boosting*, which work by sequentially adding models that correct the residual prediction errors of the previous model, usually made up of decision trees.

However, building models based on the boosting approach can be slow, especially for large data sets. Therefore, to mitigate this problem, the algorithm eXtreme Gradient Boosting (XGBoost) was proposed, which creates individual trees using several processors and the data is organized to minimize the search time, which decreases the training time and improves the performance [22]. The XGBoost algorithm is effective for a wide range of classification and regression problems [4], however, its application in data streams is still not much explored. The main studies are dedicated to the classification of data streams, such as *Adaptive Extreme Gradient Boosting* (AXGB). AXGB uses an incremental strategy to create the set, instead of using the same data to select each function, it uses subsamples of data taken from non-overlapping windows [20]. An adaptation of AXGB was proposed in [3], called AFXGB, which made use of an alternating model training strategy instead of an ensemble of classifiers. It achieved great improvement on running time while keeping the same level of accuracy of AXGB.

Therefore, considering that AFXGB technique has shown good results in classification problems, but has not yet been applied in data stream regression, this work explores AFXGB for regression (AFXGBReg) combined with active concept drift detection techniques. We compare AFXGBReg runtime and mean squared error with other data streams regressors synthetic datasets with different kinds of concept drifts.

The paper is organized as follows: Section 2 presents an overview of concept drift detector techniques and data stream regressors, Section 3 describes the proposed algorithm and in Section 4 we present the testing methodology and discuss the results. Lastly, Section 5 features the conclusion and future work.

2 Related Works

This section presents the summary of the literature review that was carried out on themes involving decision trees, data stream regression and strategies to deal with concept drifts.

On the matter of dealing with concept drifts, the detection of drifts can be done in two ways [11]: i) Active: designed to detect concept drifts using different types of detectors, so if there is a concept deviation, the model is updated and ii) Passive: the model is continually updated whenever new data becomes available, regardless of whether the change is taking place or not [17].

One of the most popular active concept drift detector algorithms is the Adaptive Sliding Window (ADWIN). It monitors a sequence of real value inputs using

sliding windows [2]. It uses two windows: a reference window and a test window. ADWIN notices the occurrence of concept drift by identifying a large difference between the averages of the two subwindows. Once a drift point is detected, all old data samples before that drift time point are discarded [25]. A more recent method for concept drift detection is the KSWIN, which is based on the Kolmogorov-Smirnov (KS) [15] statistical test. Similar to ADWIN, KSWIN also implements two functions: one for adding the new one-dimensional data in the local sliding window and the second to inform if the deviation was detected or not. Another algorithm called Drift Detection Method (DDM) monitors the error rate, assuming it will decrease as the number of instances increases and the data distribution is stationary in the incremental learning process [24]. DDM was the first algorithm to define an warning level and change level for concept change detection [1]. If the confidence level of the observed error rate reaches the warning level, DDM starts building a new learner by using the old learner for predictions. If the change has reached the concept drift level, the old learner will be replaced by the new learner for other prediction tasks [16]. DDM works best with sudden change data where concepts can gradually pass without triggering the change level [8].

In [14] it was developed a work involving regression in data streams with passive adaptation to concept drifts, called *Rival Learner*. The algorithm uses an ensemble of regressors and has two submodels, one based on historical samples, called the global model, and the other based on samples from current sliding windows, called the local model. These two models compete with each other to see who has the best performance and the winner takes over as the global model. In the tests performed by the authors, when a concept deviation occurs, the proposed algorithm keeps the model effectively updated without the need for active detection.

An algorithm called Adaptive eXtreme Gradient Boosting (AXGB) was presented in [20], using XGBoost for classification of data streams with concept drifts. The central logic of AXGB is the incremental creation/updating of an ensemble of classifiers, with a fixed size, and when this size is reached the oldest classifiers are discarded to make way for the new models. AXGB uses ADWIN to detect concept drifts, triggering a routine to update the ensemble based on the change detection signal. In [3] it was proposed an adaptation of AXGB [20] called AFXGB. In order to reduce its training and testing time, it eliminated the set of classifiers from the original proposal and made use of an alternating model training strategy. This means that the classifier has a lifetime and before this lifetime is reached, an alternate model is trained in background to replace the current model. Using this strategy the model becomes incremental without the need to use a set of models and it also avoids over-specialization.

On the ensemble of classifiers and active concept drift detectors, in [25] it was developed a framework called *Performance Weighted Probability Averaging Ensemble* (PWPAE). The proposed framework is based on learning a set of models that uses combinations of two detection methods, ADWIN and DDM, and two drift adaptation methods, Adaptive Random Forest (ARF) and Streaming

Random Patches (SRP), to build basic learners. The results presented by the authors show that the algorithm reached high levels of accuracy compared to recent works.

Regarding data stream regression, in [10] it was proposed an adaptation to Adaptive Random Forest (ARF) for data stream regression task, called ARF-Reg. The algorithm works in a voting system responsible for averaging the individual predictions to obtain the final prediction, with its base learner being a regression tree, with the FIMT-DD algorithm. The update dynamics in ARF-Reg are both passive and active using ADWIN as concept drift detectors. As a result, ARF-Reg achieved considerably small error rates, especially in real-world scenarios, thus showing its effectiveness.

It is noticed that the regression task, despite being one of the most common topics in the area of machine learning, is not a topic widely discussed and studied in the analysis of data streams. Therefore, this work aims to adapt the AXGB algorithm to support data stream regression with active concept change detection.

3 Proposal: Adaptive Fast XGBoost Regressor

The original proposal of AXGB uses an ensemble of XGBoost classifiers, however this considerably harmed the performance of the model. Therefore, the AFXGB algorithm proposed by [3] will be used as the basis of this work, an adaptation to the AXGB that obtained a shorter training time while maintaining the same accuracy as the original AXGB. In the proposal of AFXGB, only one XGBoost model is trained and incrementally updated. However, after this classifier exceeds a limit of trained data, another classifier is trained as the replacement, thus avoiding the overfitting of the first one.

Although the incremental AFXGB model update strategy can handle passive detection of concept change as the model is updated based on more recent data, adjustment for abrupt concept drifts is not adequately supported, due to its delay in the perception of change. Therefore, this work proposes the inclusion of active concept drift detection techniques to AFXGB, while porting it to regression. We call it Adaptive Fast XGBoost Regressor (AFXGBReg).

Therefore, this work proposes to explore the following active concept change detection techniques: ADWIN, KSWIN and DDM. These detectors are intended to identify concept drifts and trigger the model update mechanism. This mechanism consists of resetting the size of the data window coming from the stream used to update the model, causing the model to be updated faster, resulting in a quicker response to the change in concept.

3.1 Implementation

A pseudo-code for the proposed algorithm is presented in Algorithm 1. The modifications implemented AFXGB started with the inclusion of an active detection step of concept drift performed by the set of detectors composed by: ADWIN,

KSWIN and DDM. Afterwards, the classification model was ported to regression and a new flag which allows the execution of the algorithm with a window-size *reset* mechanism was implemented, happening when the regressor is replaced by the new one.

Algorithm 1: Adaptive Fast XGBoost Regressor algorithm

Input: (x, y) ϵ Data Stream
Data: w = maximum number of samples on the window, M = XGBoost classifier model, W = window, SW = sliding window of data, *life_time* = lifetime of each classifier, *training_time* = training time of each regressor, count = how many times the window has run

```
Adds (x, y) to window (W)
if |W| > w then
    if M <> NULL then
        if training_time ≥ (life_time - cont) then
            # Checks if it is inside the training time of next regressor
            nextM = trains next XGBoost regressor;
        else if count ≥ life_time then
            # resets counter and starts to use the new classifier
            count = 0;
            M = nextM;
            if active_reset == 'Y' then
                w = 0;
        M = loads the previous model;
        M' = trains regressor with W using M and adds to M;
        M = M';
        Saves new M model recently trained;
    else
        M = Trains regressor with W;
        Saves new M model recently trained;
    if detect_drift == True then
        # Calculate the absolute error
        error = abs(self.predict(X) - y);
        foreach active_detector of D do
            if drift_detection(active_detector, error) then
                # If any active concept drift detector detects a drift based on
                  the absolute error, the window size is reset to zero.
                w = 0;
                # In the first detection of concept drift by some detector the
                  loop is broken, advancing to next instances
                break;
    W = W - W';
    count = count + 1; # counts how many times the window has run
returns M;
```

As input, the set x and y belonging to the data stream are present: with x being the input data while y is the value that will be predicted. In the algorithm, when the sliding window of data coming from the stream is full, the algorithm checks if there is any already trained XGBoost regression model (Line 3). If it does not exist, an XGBoost regressor is trained (Line 17). If it already exists, other conditions are tested with the proposal of verifying the life time of the regressor (Lines 4 to 12) and then the regressor is loaded (Line 13) and updated (Line 15).

The adaptation to port XGBoost for regression is carried out in order to change the objective function of the algorithm in the configurable parameters of boosting of the XGB to *reg:squarederror*, having as an internal metric for data validation by the algorithm the Root Mean Square Error (RMSE). In addition, the prediction function was also transformed, in order to return floating values from X instances instead of labels as in classification. With the model loaded, an array of active detectors is iterated to detect a concept drift. Each detector must be complementary to each other, in order to aggregate in accuracy/MSE or R^2, not considerably increasing the execution time or memory weight of the model (Lines 20 to 21). For this implementation, ADWIN, KSWIN and DDM detectors were chosen.

Since AXGB was developed for classification, the instances passed to each detector were the ones incorrectly classified. Now, to make the portability of the algorithm for regression, a function was created (Line 22) to calculate the Mean Absolute Error (MAE), whose value is treated in order to identify changes in concept by the active detection algorithms. MAE is calculated by the absolute difference between observed (actual) and predicted (hypothesis) values. Each time a change in concept is detected by some detector, the size of the sliding window is reset and the iteration loop over all other detectors for that instant analyzed is broken by means of a *break* (Line 28).

With the implementation of the break after a detection, the final execution time of the algorithm is reduced without harming the calculated MSE, because if all the detectors used detected a change in concept at a certain moment, they would all need to be called, making it more costly in time and memory the process, as well as causing an unnecessary *reset* of the same sliding window times the number of detectors used.

3.1.1 Drift Detectors Scheduling

In addition to choosing which detectors would integrate the algorithm, an analysis was also necessary considering their order of execution and scheduling, as well as hyperparameter values for each detector. Analyzing in depth the characteristics of the chosen detectors, the sensitivity for detecting changes in concept by the KSWIN detector was identified through experiments and empirical analysis. Considering aspects related to the amount of concept changes by the KSWIN detector and detection quality by the ADWIN detector, the detectors were staggered in an orderly manner:

1. **ADWIN**: It was chosen as the first detection option due to its assertiveness for recurrent and gradual concept changes, while maintaining a good detection rate for abrupt changes;
2. **DDM**: Effective and neutral performance for abrupt concept changes;
3. **KSWIN**: As a last option, if a change in concept is not detected by any of the other detectors mentioned above, KSWIN has a high quantitative percentage of detections for changes in concept, especially abrupt ones.

3.1.2 Parameters

- **Learning Rate (eta)**: a value between 0 and 1. The learning rate applies a weighting factor to new trees added to the XGBoost model. When next to 0 the algorithm will make less corrections, this resulting in more trees and slower processing time.
- **Maximum depth (*max_depth*)**: the maximum depth which the tree can reach. Increasing this number created a more complex model and increases the memory consumption.
- **Maximum window size (*max_window_size*)**: maximum size of the window which stores the data from the data stream. The algorithm updated the models only when maximum window size is reached.
- **Classifier life time (*life_time*)**: Lifetime of each alternate classifier. This value is based on the number of times which the sliding window was reset.
- **Training time (*training_time*)**: training time of each alternate classifier. This value is based on the number of times which the sliding window was reset.
- **Reset window size when regressor is exchanged (*active_reset*)**: Boolean used to identify whether the window size reset strategy in the regressor exchange will be used or not;
- **Concept drift (*detect_drift*)**: This boolean determines whether the concept change with the array of detectors will be applied during learning.

4 Results assessment

This section presents the results of tests performed with the AFXGBReg algorithm. The Section 4.1 outlines the methodology that was used for the tests and the Section 4.2 details the results obtained.

4.1 Testing methodology

To evaluate of the proposed algorithm, six implementations were considered. Of these six algorithms, two were proposed by the author and four external for comparison, are described below:

- AFXGBReg-Dr: algorithm proposed in this work, using alternating regressors, ensemble of active detectors of concept drift and window reset in concept drift detections as well as in the step where the regressor is replaced by a new one previously trained.

- AFXGBReg-D: algorithm also proposed in this work, using alternate regressors, ensemble of active detectors of concept drift and window reset **ONLY** in detections of concept drift.
- HTR (Hoeffding Tree Regressor): is a regression adaptation of the incremental tree algorithm of the same name for classification. HTR uses Hoeffding's inequality to control its node division decisions.
- HTRA (Hoeffding Adaptive Tree Regressor): similar to the previous implementation, but with the addition of the tree using the ADWIN concept change detector and PERCEPTRON (single layer neural network) to make predictions.
- KNN (k-Nearest Neighbors regressor): nonparametric regression method where predictions are obtained by aggregating the values of stored samples from n_nearest neighbors against a query sample.
- ARFReg (Adaptive Random Forest Regressor): adaptation for regression of the existing algorithm for classification, using the ADWIN concept drift detector.

Evaluated aspects include runtime, Mean Squared Error (MSE), and the ability to adapt to different types of concept drift. The evaluation method selected is *Prequential Evaluation*, where the data are analyzed sequentially when they arrive at the model.

The databases used for the tests were taken from the *scikit-multiflow* library provided by [21], being synthetic databases generated through the *Hyperplane Generator* and *ConceptDriftStream* generation methods. Table 1 presents the bases selected for the evaluation.

Table 1. Datasets used on evaluation.

Dataset	Drift Type	# Instances	# Drifts
CDS_A	Abrupt	500.000	10
HYP	Incremental	500.000	1
CDS_G	Gradual	500.000	3

The Prequential Evaluation simulation was executed 5 times for each algorithm and dataset to obtain the average MSE, training time, testing time and total time.

The AFXGBReg hyperparameters were selected empirically based on numerous simulations and were defined as the following: No. Estimators = 30, Learning Rate = 0.3, Max Window Size = 1000, Max Tree Depth = 6. The active drift detectors hyperparameters were defined as: *adwin_delta* = 0.0000001, *kswin_alpha* = 0.0000001, *kswin_window_size* = 100, *kswin_stat_size* = 30, *ddm_min_num_instances* = 30, *ddm_warning_level* = 2, *ddm_out_control_level* = 3.

Table 2. Average MSE and rankings for the six algorithms considered in the study over 3 datasets.

Dataset	AFXGBReg-Dr	AFXGBReg-D	HTR	HTRA	KNN	ARFReg
CDS_A	**3459.59** (1)	3833.69 (2)	9806.61 (6)	7971.14 (5)	7190.04 (4)	5064.42 (3)
HYP	0.14 (5.2)	0.15(5.6)	0.109 (2.2)	0.105 (2.4)	0.13 (4.2)	**0.10** (1.4)
CDS_G	**4037.43**(1)	4399.73 (2)	11587.09 (6)	9268.18 (5)	7663.68 (4)	5273.96 (3)
Avg. MSE	2499.06	2744.52	7131.27	5746.48	4951.28	3446.16
Avg. rank	2.4	3.2	4.733	4.133	4.067	2.467

4.2 Analysis of the results

We present on Table 2 the average MSE of the 5 executions of the considered models for each dataset. The best MSE is highlighted, and the corresponding ranking of the model for that dataset is next to the MSE. The model rank is defined by ordering all model's MSE for that dataset from lowest to highest, this meaning that the smallest ranking is the better one.

In the following sections we analyze the MSE, runtime and concept drift recovery results.

4.2.1 MSE analysis

It is noticed that the average MSE of all algorithms varies a lot. To confirm the performance difference between the algorithms, we apply a pair-wise *post-hoc* test for multiple comparisons [5]. The *post-hoc* test applied was a Nemenyi test using the algorithms average ranking. This test determines a Critical Difference (CD), meaning that two algorithms whose ranking difference is greater that the critical difference are considered significantly different. The Nemenyi resulted on a CD of 1.38 and pair-wise comparisons are shown on Figure 1.

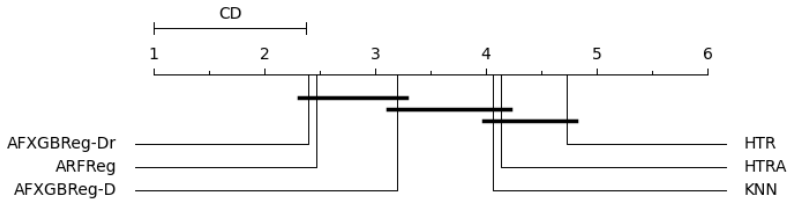

Fig. 1. Nemenyi test of model's average rankings.

It is noticeable that AFXGBReg-Dr and ARFReg have similar performances, followed by AFXGBReg-D. Since the difference of average ranks between these three models is smaller than the value of CD, this test confirms that the difference in regression capability is not significant. This also means that the models

AFXGBReg-Dr, ARFReg and AFXGBReg-D are superior than the other models in terms of MSE.

4.2.2 Runtime analysis

We now focus on comparing the simulated models runtimes, especially of AFXGBReg-Dr, AFXGBReg-D and ARFReg, which were the ones who presented best performance of MSE. Table 3 presents the average training time, average testing time and average total time on minutes of all simulations on all datasets.

Table 3. Average times of simulations for the algorithms considered in the study.

	Time (minutes)		
Model	**Avg. training**	**Avg. testing**	**Avg. Total**
AFXGBReg-Dr	3.53	1.88	5.41
AFXGBReg-D	3.53	1.87	5.40
HTR	1.52	0.24	1.75
HTRA	5.89	0.23	6.12
KNN	0.22	3.79	4.01
ARFReg	179.30	3.27	182.57

It is apparent that HTR and KNN have the smaller total times. This is expected because of the algorithms simplicity, since they were not developed for data streams. On the other hand, they also had worse MSE performance than other models. Regarding the superior MSE group, AFXGBReg-Dr and AFXGBReg-D have similar times, as expected since the algorithms are related, but ARFReg shows an outstanding training time when compared to them. The average total time of ARFReg model is 33 times larger than AFXGBReg models.

It is worth mentioning that AFXGBReg's runtime could be reduced even further by using a smaller lifetime, thus decreasing the model's build-up by resetting it more frequently, but this needs to be balanced with possible MSE increase.

4.2.3 Concept drift recovery analysis

To evaluate the ability of the models on recovering from a concept drift we analyzed the accuracy of each model over time after each drift. We selected one simulation of CDS_A to show the model's behavior on abrupt concept drift.

On Figure 2 we observe the models MSE for the simulation of CDS_A dataset, where both AFXGBReg models have a smaller peak of MSE than the others. This can be perceived in more detail on Figure 3 and 4, where the drifts of points 50k and 450k are highlighted. It is seen that HTR and HTRA have higher MSE peaks, and also that HTR and KNN do not recover completely from the drift. This happens because HTR and KNN do not have features to reset the model, accumulating obsolete concepts.

On the 50k CDS_A drift depicted by Figure 3, we can see that AFXGBReg models recovery is faster than other models, showing the effectiveness of the array of active concept drift detectors and model reset. This can be further observed on Table 4, where the MSE of all models is displayed for five data points related to the 50k concept drift of CDS_A.

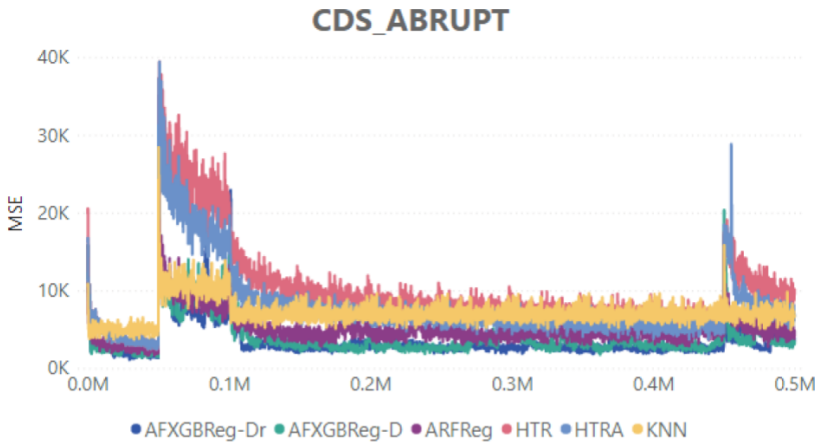

Fig. 2. MSE over time for evaluated models on CDS_A dataset.

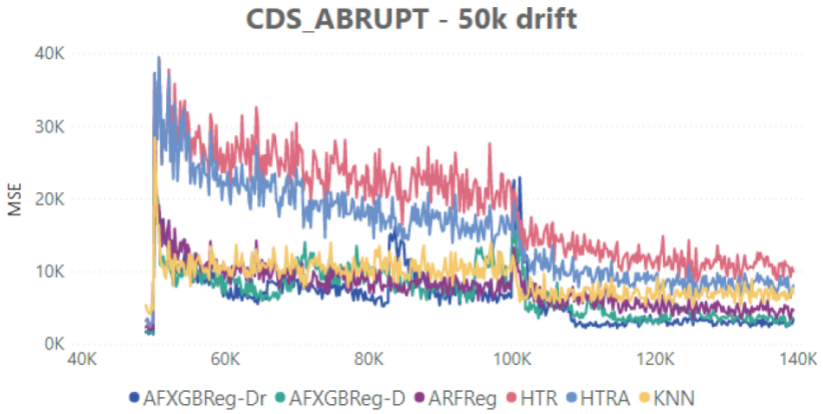

Fig. 3. Highlight of 50k concept drift of CDS_A simulation.

On data 50k the drift has not happened yet, so all models are at their lowest MSE, then on the following data register after the drift (50.2k) all models suffer a great increase in MSE, prominent on ARFReg, HTR and HTRA models. On the next observed data point of 50.4k the models start to adjust to the new

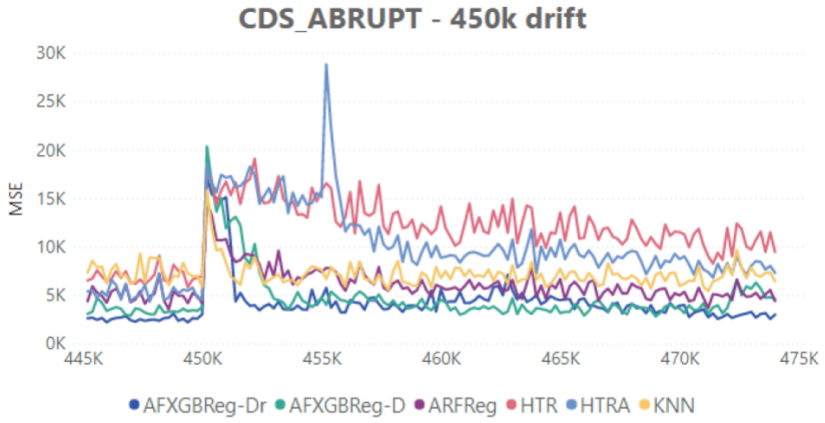

Fig. 4. Highlight of 450k concept drift of CDS_A simulation.

concept, decreasing MSE and are able to arrive on stable predictions for the new concept by point 55.6k. In this scenario it took about 5.6k of data volume for achieving full concept drift recovery. The last data point highlighted on the table is the lowest MSE obtained on this concept, achieved by AFXGBReg-Dr on 82.4k. We can see that none of the models were able to achieve MSE as low as they previously had, but in comparison to AFXGBReg variants and ARFReg, the other models had much larger error.

Table 4. MSE of models on data points before and after the 50k concept drift for the 1st simulation of CDS_A dataset.

# data	AFXGBReg-Dr	AFXGBReg-D	ARFReg	HTR	HTRA	KNN
50k	1,326	1,432	1,963	2,918	2,991	4,750
50.2k	26,358	30,496	35,018	37,292	37,220	28,408
50.4k	18,703	16,580	21,904	32,722	32,722	22,855
55.6k	8,520	8,726	11,613	28,335	24,888	11,196
82.4k	5,646	9,823	7,950	22,663	15,872	11,912

5 Conclusion and future work

In this work we proposed a Fast Adaptive XGboost Regression model with active concept drift detetction (AFXGBReg-D) to handle data streams. AFXGBReg uses an alternate model training strategy in order to reduce the model piling up complexity and adapt faster to concept drifts. It was also proposed a variation of AFXGBReg-D with a fixed window size reset each time the regressor is replaced by the new model, called AFXGBReg-Dr.

Comparing their MSE, speed and ability to adapt to concept drift with other models on synthetic datasets with different kinds of concept drifts, it was seen that AFXGBReg variations are able to achieve the same level of MSE of ARFReg, obtaining superior performance than the other models, as proven statistically. Meanwhile, AFXGBReg models are also 33 times faster than ARFReg, meaning that it is able to keep the same MSE performance while being much faster.

Regarding to concept drifts adaptation, AFXGBReg-Dr model presented a smaller peak in MSE and faster adaptation than the other models, including AFXGBReg-D. It was seen that it needs less volume of data to comeback from the MSE increase and it is able to keep a low long-term average MSE, since old concepts are forgotten when the model is resetted and substituted.

For future works we intend to explore AFXGBReg models with semi-supervised learning. Still, we intend to study more about other detection strategies besides the active one, as well as explore the AFXGBReg algorithm in order to make it increasingly performant for different types of concept drift.

Acknowledgments. We would like to specially thanks FAPESC – Fundação de Amparo à Pesquisa e Inovação do Estado de Santa Catarina – to partially funded this research work.

References

1. Abbaszadeh, O., Amiri, A., Khanteymoori, A.R.: An ensemble method for data stream classification in the presence of concept drift. Frontiers of Information Technology & Electronic Engineering **16**(12), 1059–1068 (2015)
2. Barddal, J.P.: Vertical and horizontal partitioning in data stream regression ensembles. In: 2019 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE, Curitiba, BR (2019)
3. Bonassa, G.: Adaptação de classificador utilizando a biblioteca XGBoost para classificação rápida de fluxos de dados parcialmente classificados com mudança de conceito (2021)
4. Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., et al.: Xgboost: extreme gradient boosting. R package version 0.4-2 **1**(4), 1–4 (2015)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research **7**, 1–30 (2006)
6. Ditzler, G., Roveri, M., Alippi, C., Polikar, R.: Learning in nonstationary environments: A survey. Comp. Intell. Mag. **10**(4), 12–25 (Nov 2015). https://doi.org/10.1109/MCI.2015.2471196, https://doi.org/10.1109/MCI.2015.2471196
7. Elwell, R., Polikar, R.: Incremental learning of concept drift in nonstationary environments. IEEE Transactions on Neural Networks **22**(10), 1517–1531 (2011)
8. Gama, J., Žliobaitundefined, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. **46**(4) (mar 2014). https://doi.org/10.1145/2523813, https://doi.org/10.1145/2523813
9. Gamage, S., Premaratne, U.: Detecting and adapting to concept drift in continually evolving stochastic processes. In: Proceedings of

the International Conference on Big Data and Internet of Thing. p. 109–114. BDIOT2017, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3175684.3175723, https://doi.org/10.1145/3175684.3175723
10. Gomes, H.M., Barddal, J.P., Ferreira, L.E.B., Bifet, A.: Adaptive random forests for data stream regression. In: ESANN. IEEE, Curitiba, Paraná, Brasil (2018)
11. Krawczyk, B., Minku, L.L., Gama, J., Stefanowski, J., Woźniak, M.: Ensemble learning for data stream analysis: A survey. Information Fusion **37**, 132–156 (2017)
12. Laney, D.: 3D data management: Controlling data volume, velocity, and variety. Tech. rep., META Group, EUA (February 2001), http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf
13. Larson, D., Chang, V.: A review and future direction of agile, business intelligence, analytics and data science. International Journal of Information Management **36**(5), 700–710 (2016)
14. Liao, Z., Wang, Y.: Rival learner algorithm with drift adaptation for online data stream regression. In: Proceedings of the 2018 International Conference on Algorithms, Computing and Artificial Intelligence. ACAI 2018, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3302425.3302475, https://doi.org/10.1145/3302425.3302475
15. Lopes, R.H., Reid, I., Hobson, P.R.: The two-dimensional kolmogorov-smirnov test (2007)
16. Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, G.: Learning under concept drift: A review. IEEE Transactions on Knowledge and Data Engineering **31**(12), 2346–2363 (2018)
17. Mahdi, O.A., Pardede, E., Ali, N., Cao, J.: Fast reaction to sudden concept drift in the absence of class labels. Applied Sciences **10**(2), 606 (2020)
18. Mayr, A., Binder, H., Gefeller, O., Schmid, M.: The evolution of boosting algorithms. Methods of information in medicine **53**(06), 419–427 (2014)
19. Mehmood, H., Kostakos, P., Cortes, M., Anagnostopoulos, T., Pirttikangas, S., Gilman, E.: Concept drift adaptation techniques in distributed environment for real-world data streams. Smart Cities **4**(1), 349–371 (2021)
20. Montiel, J., Mitchell, R., Frank, E., Pfahringer, B., Abdessalem, T., Bifet, A.: Adaptive XGBoost for evolving data streams. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE, Hamilton, New Zealand (2020)
21. Montiel, J., Read, J., Bifet, A., Abdessalem, T.: Scikit-multiflow: A multi-output streaming framework. Journal of Machine Learning Research **19**(72), 1–5 (2018), http://jmlr.org/papers/v19/18-251.html
22. Ramraj, S., Uzir, N., Sunil, R., Banerjee, S.: Experimenting XGBoost algorithm for prediction and classification of different datasets. International Journal of Control Theory and Applications **9**, 651–662 (2016)
23. Schapire, R.E.: The boosting approach to machine learning: An overview. Nonlinear estimation and classification pp. 149–171 (2003)
24. Yan, M.M.W.: Accurate detecting concept drift in evolving data streams. ICT Express **6**(4), 332–338 (2020)
25. Yang, L., Manias, D.M., Shami, A.: Pwpae: An ensemble framework for concept drift adaptation in iot data streams. arXiv preprint arXiv:2109.05013 (2021)
26. Yu, H., Lu, J., Zhang, G.: Morstreaming: A multioutput regression system for streaming data. IEEE Transactions on Systems, Man, and Cybernetics: Systems pp. 1–13 (2021). https://doi.org/10.1109/TSMC.2021.3102978