




# Efficient online tree, rule-based and distance-based algorithms

Saulo Martiello Mastelini   [ Institute of Mathematics and Computer Science - University of São Paulo | [mastelini@alumni.usp.br](mailto:mastelini@alumni.usp.br) ]

André Carlos Ponce de Leon Ferreira de Carvalho  [ Institute of Mathematics and Computer Science - University of São Paulo | [andre@icmc.usp.br](mailto:andre@icmc.usp.br) ]

 Institute of Mathematics and Computer Science - University of São Paulo, 400 Trabalhador São Carlense Av., Centro, São Carlos, SP, 13566-590, Brazil.

**Received:** 30 October 2024 • **Accepted:** 14 May 2025 • **Published:** 18 June 2025

**Abstract** The fast development of technology resulted in the constant production of data in different forms and from different sources. Contrary to what was observed in the first machine learning (ML) research works, there might be too much data to handle with traditional algorithms. Changes in the underlying data distributions might also render traditional ML solutions useless in real-world applications. Online ML (OML) aims to create solutions able to process data incrementally, with limited computation resource usage, and to deal with time-changing data distributions. Unfortunately, we have seen a recent growing trend in creating OML algorithms that solely focus on predictive performance and overlook computational costs. In regression tasks, the problem is even more pronounced when considering some of the most popular OML solutions: decision trees, decision rules, and ensembles of these models. We created improved and efficient OML algorithms from the algorithmic families mentioned, focusing on decreasing time and memory costs while maintaining competitive predictive performance. In this paper, we present an overview of the main contributions discussed in the Ph.D. thesis of the main author, which was awarded the best thesis prize by the Brazilian Computer Society in the 37th edition of the competition. Our proposals are either novel standalone OML algorithms or additions that can be paired with any existing tree or decision rule regressors.

**Keywords:** Online machine learning, Hoeffding trees, Ensembles, Regression, Online nearest neighbors, River library

## 1 Introduction

In the last decades, we have seen a steep increase in the produced and monitored data [Bahri *et al.*, 2021; Palli *et al.*, 2024]. As the years passed, rich and varied data sources concerning different real-world phenomena became widely available. With the prevalence of data and the development of technology, concerns about data privacy have also emerged, which resulted in the creation of new laws focusing on data in the international domain<sup>1</sup> and also in the Brazilian<sup>2</sup> context. These sets of laws become crucial as most of the gadgets and digital tools we use nowadays are also data sensors. In response to the increased data availability and the improvements in infrastructure and hardware, which naturally occurred, computational and pattern recognition tools also had to evolve [Gama, 2010; Russell and Norvig, 2016].

Traditional Machine Learning (ML) algorithms were originally intended to use all available data for training and make no changes once trained. Thus, this type of solution may not be able to deal with the huge amount of incoming training data [Gama, 2010; Bifet *et al.*, 2018]. The performance of traditional ML solutions also catastrophically degrades in the presence of changes in the underlying distribution of the data used for learning. This data change phenomenon is commonly referred to as concept drift [Bayram *et al.*, 2022; Agrahari and Singh, 2022] and naturally occurs in real-world predictive tasks. The fact that data nowadays arrives continu-

ously as a data stream has led to the creation of incremental or online ML algorithms. Online ML (OML) seeks to create learning models that are trained incrementally, that is, one instance at a time, and are also capable of adapting to changes in the underlying data distribution. These algorithms also operate under constrained computational resources due to the potentially unbounded nature of their input data. Ideally, an OML algorithm should only process each datum once and then discard that input [Bifet and Gavaldà, 2009]. By doing so, the learning model must be able to extract and store efficiently any useful information that may be extracted from the inputs. It must also be ready to provide predictions at any time. As a direct consequence, reducing the time requirements of OML solutions is also of paramount importance. That is, besides using a limited amount of memory OML models should be able to process data as fast as possible so that the incoming data streams can be processed in their entirety.

In the last two decades, plenty of OML solutions were designed to deal with the mentioned aspects of data stream mining. Decision Trees (DT) and ensembles of these incremental learning models were established among the most popular and effective OML solutions [Gama, 2010; Bifet *et al.*, 2018; Gomes *et al.*, 2017]. Hoeffding Trees (HT) [Domingos and Hulten, 2000] are a family of DTs that powers most of the tree-based solutions in OML. These DT algorithms rely on the Hoeffding inequality and a statistical bound obtained from this expression [Hoeffding, 1963] to decide, incrementally, when the tree model has gathered enough statistics about the relationship between input features and the

<sup>1</sup><https://gdpr-info.eu/>

<sup>2</sup>[https://www.planalto.gov.br/ccivil\\_03/\\_ato2015-2018/2018/lei/l13709.htm](https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm)

target variable(s) to enable growth. Due to practical limitations, HTs cannot store incoming data samples; instead, they save summary statistics to represent the incoming data stream characteristics.

Unfortunately, in recent years, we saw only minor changes and improvements in the available solutions, with not nearly as many breakthroughs as the first OML algorithms offered compared to their original, batch-based, traditional ML counterparts. Most recent new OML algorithms, often ensembles of HTs, focus mostly on predictive performance alone. To exacerbate the situation, in many cases, the slight obtained predictive performance improvements come with a steep increase in memory and time costs [Korycki and Krawczyk, 2020; Cano and Krawczyk, 2022]. This problematic situation was even more pronounced when considering regression tasks, which had not yet received a lot of attention from the research community. Efforts to improve the computational costs of classification decision trees were performed over a decade ago [Pfahring et al., 2008] and enabled substantially decreasing the costs of the original HT algorithm [Domingos and Hulten, 2000]. Regression, on the other hand, had not yet seen the same kind of efforts and every regression-based HT proposed after the original one [Ikonomovska et al., 2011b] did not change the original strategy for enabling tree growth, focusing instead on reducing the prediction error.

We did not follow the same trend and took the opposite path by exploring ways to reduce the memory and time costs of regression HTs and forests composed of this type of DTs. Incremental decision rules directly benefit from any improvements made for HT regressors (HTR) because this family of incremental learning models also relies on the Hoeffding inequality and performs decision splits similar to the HTs. By focusing primarily on the computational costs of HTRs, we believe we can boost the best recent developments in the OML research field for regression by improving their efficiency and potential for application in real-world learning tasks. To achieve the desired reduction in computational costs, we employed multiple strategies to improve existing OML algorithms [Mastelini and de Carvalho, 2021; Mastelini et al., 2021], and we also devised an accurate and efficient novel ensemble regressor [Mastelini et al., 2022]. We also tackled reducing the computational costs of distance-based algorithms. More specifically, we focused on incremental k-Nearest Neighbors (k-NN) algorithms that operate in sliding data windows (First-in, First-out – FIFO). In this case, our contribution was task-agnostic and applicable to both classification and regression tasks, as well as other yet-to-be-explored possibilities.

In the remainder of this Special Issue paper, we will explore the contributions brought by the Ph.D. thesis<sup>3</sup> to the data stream literature and society, in general. The next sections are organized as follows. Section 2 presents the problem tackled in the thesis, as well as the objective, research questions, and hypothesis that guided all the research development. Section 3 presents the related work upon which the research was built. We summarize our main contributions

to the research and application fields in Section 4. Last, we present our concluding remarks in Section 5.

## 2 Problem, objective, research questions, and hypotheses

Although many new streaming-based OML solutions were created, they mostly focus on predictive performance and often overlook the needed computational resources. This becomes more apparent when considering resource-constrained applications and reducing the environmental impacts related to energy usage [García-Martín et al., 2019; García Martín, 2020]. By surveying the available literature, we identified the following specific problems related to the scope of the thesis.

**Problem:** most existing HT regressors (HTR) and ensembles thereof were too computationally costly for real-world applications. Besides, in the specific case of sliding window-based k-NN-based algorithms, we realized that the existing solutions perform a complete scan of the data buffer elements, which might become impractical as the amount of stored examples increases.

Hence, we envisioned a central principle that guided the whole development of the thesis.

**Objective:** to develop efficient tree and rule-based incremental regressors, and distance-based algorithms.

We formulated three research questions to guide our work and to direct us toward the contributions we ultimately achieved:

- Q1:** Can the computational costs of HTRs be reduced without significant impacts on predictive performance?
- Q2:** Can efficient online tree-based ensembles be created while keeping competitive predictive performance to state-of-the-art solutions?
- Q3:** Is there an efficient and alternative strategy to perform nearest neighbor search queries on a data buffer that is constantly updated using a FIFO data ingestion policy?

Given the research questions and the literature review, we formulated the following hypotheses to pursue further:

- (Hyp. 1)** *The use of summarization and data sampling techniques can significantly reduce computation costs for building incremental trees and decision rule-based regressors, without significantly impacting prediction accuracy.*
- (Hyp. 2)** *Efficient and incremental graph-based search structures can be created to perform nearest neighbor search queries using arbitrary distance measures.*

In the thesis, we used **Hyp. 1** to answer **Q1** and **Q2**, and **Hyp. 2** as the starting point to address **Q3**.

We performed an in-depth literature review to obtain a broad view of the research field. This was a continuous and ongoing action as time progressed and we developed our research contributions. We performed several exploratory analyses of sampling, data compression, and other statistical and computational tools to fulfill our objective.

<sup>3</sup>The full thesis text can be publicly accessed via the following address: <https://repositorio.usp.br/item/003152320>

### 3 Theoretical foundation and related work

HTs, the main base algorithm for thesis development, work by creating a single root node and subsequently creating additional decision nodes and leaves [Domingos and Hulten, 2000; Ikonovska *et al.*, 2011b]. To do so, HTs keep structures called attribute observers (AO) to monitor input statistics [Mastelini and de Carvalho, 2021]. Namely, each leaf node in an HT carries one AO per input feature. Categorical features are easier to monitor due to their inherent discrete nature. Continuous features, i.e., real numbers, are more challenging to manage because there are no predefined partitions for making tree splits. HTRs and decision rule regressors work by monitoring how each input contributes to reducing the overall variance in the target variable [Ikonovska *et al.*, 2011b,a; Duarte *et al.*, 2016]. The input feature and cut point combination that provides the maximum Variance Reduction (VR) is chosen to create splits and expand the tree or decision rule structure. Hence, regression AOs use incremental variance estimators to evaluate the VR for each monitored split candidate.

There were more efforts to improve HTs [Domingos and Hulten, 2000] in classification tasks [Pfahring *et al.*, 2008] than in regression. Indeed, the usual configuration of HT classifiers nowadays has a  $O(1)$  cost to both monitor a new instance and to query split candidate points [Kirkby, 2007; Pfahring *et al.*, 2008]. In the case of regression, most of the research effort has been put toward reducing the predictive error, often overlooking training costs [Ikonovska *et al.*, 2015; Osojnik *et al.*, 2018; Gomes *et al.*, 2018].

The original version of HTR [Ikonovska *et al.*, 2011b] introduced a binary search tree (BST) structure to monitor tree-splitting statistics. This algorithm, dubbed Extended BST (E-BST), also had a mechanism to deactivate nodes that did not hold promising split points. Still, a BST has a  $O(n)$  memory and split point query costs and an  $O(\log n)$  cost per point insertion. The insertion cost, in the worst-case scenario, can become  $O(n)$  when the input data is already ordered. There were also attempts to limit the number of stored BST nodes to reduce computation costs by hard-coding a maximum number of points that could be stored [Duarte and Gama, 2015; Duarte *et al.*, 2016]. An improved version of E-BST was also proposed to reduce the number of stored splitting-enabling statistics [Osojnik, 2017].

Nonetheless, none of the proposed solutions to improve the E-BST changes its asymptotic costs. We proposed several alternatives to the original and modified versions of the E-BST algorithm. Our proposed AOs effectively reduce the asymptotic costs involved in building HTRs, regarding the memory requirements and the running time. Naturally, the proposed AOs can be directly applied to Hoeffding bound-based decision rule algorithms and tree ensembles.

Lastly, in the realm of distance-based algorithms, we leverage existing state-of-the-art graph search indices [Dong *et al.*, 2011; Shimomura *et al.*, 2021] to create a novel, efficient, and general-purpose incremental structure to perform nearest neighbor search queries [Mastelini *et al.*, 2024]. To the best of our knowledge, the proposed solution is the first of its

kind available for incremental nearest-neighbor search problems that can work with arbitrary distance metrics.

## 4 Main results

Our main contributions were mainly reflected in papers published in renowned scientific diffusion venues. We will present the evolution of the thesis development and its main research products chronologically, by briefly summarizing the main findings of each main publication related to the thesis.

### 4.1 Exploring ensemble strategies for regression and identifying literature gaps

Through a collaboration with multiple international researchers, we explored numerous design aspects of online ensemble algorithms in Gomes *et al.* [2020]. Despite the various research works that had been devoted to ensemble classifiers, by that time, regression ensembles were relatively unexplored [Ikonovska *et al.*, 2015; Gomes *et al.*, 2018; Osojnik *et al.*, 2018]. It was also not clear whether or not common strategies for building incremental forests or other ensembles for classification had the same level of efficacy in regression tasks.

Hence, in Gomes *et al.* [2020] we explored in-depth the incremental Random Patches algorithm for regression, and different choices for its construction. These choices included different strategies for individual ensemble members' response aggregation, the choice of the base learning algorithm, and the strategy to detect and react to concept drifts. In our empirical findings, we observed that HTRs compared favorably to k-NN when composing an ensemble solution. We also observed that an ensemble of model trees [Ikonovska *et al.*, 2011a], i.e., those with a prediction model in their leaves (such as linear regression models), powered by a median aggregation of tree predictions stood out in terms of predictive error compared to the common strategy of using the mean prediction at the tree leaves and averaging individual trees' responses. Lastly, we observed that periodically resetting trees in the forest (at fixed or random intervals) was not significantly worse than relying on concept drift detectors for managing the forest members.

While running the extensive set of empirical experiments for the paper, another aspect of the ensemble became inherently clear. Regression trees and incremental nearest neighbor-based algorithms were not efficient. In the HTR case, both memory usage and time needed significant improvements. The popular incremental k-NN algorithms, while fairly efficient in terms of memory usage and having a negligible time cost to monitor a new element, were still prone to an exhaustive search each time a prediction had to be made. These practical experiences concerning the algorithms became the research fuel for the years to come in the Ph.D. pursuit.

## 4.2 A first attempt to speed up trees

The main cost in HTR construction comes from their AOs. Our first attempt to make HTRs faster consisted of avoiding part of the AO cost by skipping certain input features during split attempts, and thus, not directly changing the AO algorithm.

As previously mentioned, the traditional E-BST AO used in HTRs had a linear cost in terms of memory and split point query. The solution we proposed in Mastelini and Ponce de Leon Ferreira de Carvalho [2020] did not get rid of these costs neither the logarithm cost involved in the input monitoring. Nonetheless, by adding incremental estimators of the correlation between each input feature in a tree leaf, and the target, we implemented a filtering mechanism before split attempts. In this scheme, we only considered the  $k$  input features (a user-defined parameter) most correlated with the target, in absolute terms, as candidates to expand the tree structure. Although the modified HTR became generally faster than the vanilla incremental trees, two problems became apparent due to the modifications.

Firstly, relying on the linear correlation as a proxy to decide the most promising features to perform split attempts sometimes led to predictive performance degradation. This finding becomes clear when we analyze the VR mechanism used by HTRs to guide the splits. Picking the  $k$  features most correlated with the target does not guarantee the best VR will be found in this variable group. It is also unclear how such modifications to the original HTR algorithm would play out when building tree ensembles or dealing with concept drift. Secondly, adding correlation estimators to the HTRs also imposes additional memory costs that were not desired.

Despite making some progress in speeding up the regression trees, it became clear that directly optimizing the AO algorithms was the best direction to follow as the next step. Another empirical finding stood out when making the first steps towards changing the HTR construction: the incremental variance estimators used for VR computation [Ikononovska *et al.*, 2011b; Duarte *et al.*, 2016] were not reliable and, sometimes, produced theoretically impossible negative variance values. Thus, improving the paramount variance calculation became another additional objective to fulfill.

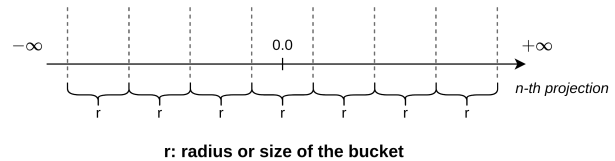
## 4.3 Creating more reliable and fast regression tree building mechanisms

In Mastelini and de Carvalho [2021], we made pivotal advancements in solving two of the open problems identified in the preceding papers. Firstly, we replaced the so-called naive incremental variance estimators [Knuth, 2014], which are prone to numerical cancellation and produced the negative variance estimates, with robust variance estimators based on the well-renowned Welford's algorithm [Schubert and Gertz, 2018]. To this end, we devised complementary formulae to those proposed in Chan *et al.* [1982] to handle the merging of partial estimates of the sample variance which was needed in the E-BST algorithm.

Our experiments showed the effectiveness of the new variance estimators compared to the ones consistently used in multiple regression-based solutions built upon the Hoeffding

inequality framework. Currently, all tree, rule-based, and forest regressor algorithms in River [Montiel *et al.*, 2021], which use VR as the split heuristic, are based on the aforementioned variance estimators.

In addition to introducing robust variance estimators, Mastelini and de Carvalho [2021] also proposed the Quantization Observer (QO) splitter. This AO relies on quantizing numerical input features into intervals of fixed width (controlled via a quantization radius) and significantly reduces the costs for input monitoring, split point query, and memory. The QO quantization scheme is presented in Figure 1.



**Figure 1.** Quantization scheme used in QO: partitions of size  $r$  are created for each numerical input feature.

QO was compared against the original E-BST and a proposed modification of it dubbed TE-BST (Truncated E-BST), which firstly reduces the number of decimal places in the inputs before inserting them into the E-BST. The experimental comparison in the paper considered the AOs in isolation and showed that QO stood out compared to its contenders. TE-BST also performed generally better than vanilla E-BST. Both QO and TE-BST produced split points that were close to the exhaustive, and thus exact, strategy employed by E-BST. Thus, the next major research goal became evaluating the contributions achieved in the aforementioned paper when applied directly to HTRs.

## 4.4 Fast and multi-branch trees

We evaluated the potential of QO as AO in Mastelini *et al.* [2021]. In this work, we compared HTRs using E-BST and TE-BST against the ones using QO as the main split mechanism. Besides that, we leveraged the inherent quantization capabilities of QO to create multi-way splits rather than binary splits. Instead of selecting a single best split option, we also considered the possibility of creating one tree branch for each partition created by QO for a given input feature. When the quantization radius used in QO is increased, such multi-way splits become manageable from a computational cost perspective. In Figure 2 we illustrate the difference between the traditional HTRs using only binary splits for numerical features and the ones also enabling multi-way splits.

The resulting multi-branch enabled HTRs showed to be competitive in terms of predictive error while significantly reducing the tree depth compared to the HTRs using strictly binary splits for numerical features. The multi-branch trees became wider rather than deeper, as one would expect when creating multiple partitions per split in each numerical input. All the variations of HTRs using QO as their AO algorithm used significantly fewer computational resources compared to the ones using E-BST or TE-BST. Our results align with the results previously reported in Mastelini and de Carvalho [2021].

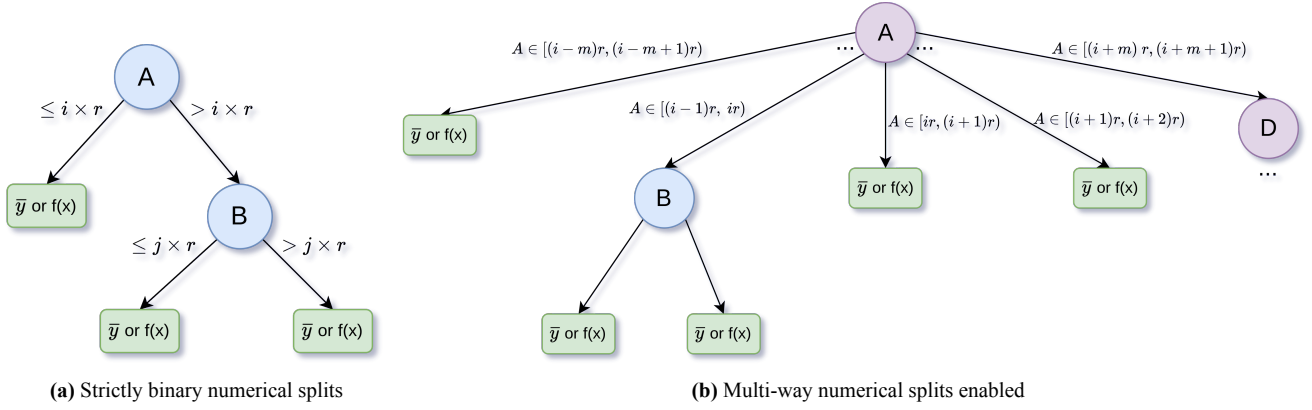


Figure 2. Comparing HTRs using the traditional strictly binary splits and a mix of multi-way and binary splits.

The multi-branch trees could be potentially easier to interpret due to their compactness while retaining the predictive performance of deeper vanilla HTR models. We also argue that in the case of using adaptive mechanisms to build/rebuild parts of the tree model due to concept drift [Bifet and Gavaldà, 2009; Manapragada *et al.*, 2022; Esteban *et al.*, 2024], the multi-branch HTRs would become more resilient to performance drops, as smaller portions of their structure would need to be rebuilt. This situation is still open for investigation.

One limitation we identified related to HTRs using QO was their apparent inefficiency when applied in ensembles. The gains previously observed did not seem to hold when composing tree committees for regression. As we found out, it was due to the common choice of hyperparameter values when creating tree ensembles. This practical aspect, which we will discuss further next, motivated us to create a new tree ensemble algorithm for regression.

#### 4.5 Extremely fast regression forests

Historically, algorithms like Adaptive Random Forest (ARF) [Gomes *et al.*, 2018] and other bagging-based incremental ensembles for regression [Gomes *et al.*, 2020] or classification [Gomes *et al.*, 2018, 2021] perform split attempts at small time intervals to early increase model diversity in the forest. The seminal ARF algorithm, by default, considers 50 instances between split attempts. In this setting, QO, despite being asymptotically more efficient than E-BST in terms of running time, results in only marginal gains, if any. In case one increases the interval between split attempts, QO will show the same remarkable results as presented in Mastelini and de Carvalho [2021] and Mastelini *et al.* [2021]. Still, we wanted to go a step further and create a more efficient ensemble algorithm specially catered towards regression.

In Mastelini *et al.* [2022], we proposed Online Extra Trees (OXT) to overcome the mentioned limitations of regression ensembles. OXT uses many of the successful components of popular incremental forest algorithms, such as randomly selecting input features in each split attempt and using foreground and background HTRs to deal with concept drift [Gomes *et al.*, 2018]. By default, OXT also uses model trees as the constituent base models, which were shown to overcome simple regression tree models in online regression

tasks [Ikonovska *et al.*, 2011a, 2015; Gomes *et al.*, 2020].

On the other hand, OXT introduces two important design aspects responsible for its effectiveness. Firstly, rather than relying on online bagging, like almost every tree-based incremental ensemble algorithm, our proposal relies on sub-bagging. In other words, rather than resampling instances with replacement, OXT samples data without replacement. This motivation comes from the fact that the input streams are potentially infinite and the HT framework is expected to converge to the model yielded by a batch decision tree algorithm given enough observations in a stationary data stream [Domingos and Hulten, 2000]. Hence, we allow OXT's trees to use a subset of the incoming instances for learning, which naturally speeds up the input monitoring.

Moreover, trees in OXT evaluate a single random split point per feature rather than comparing multiple candidates. In each tree leaf, the input feature with the best randomly chosen split point is chosen to further expand the tree structure. This new random AO has a  $O(1)$  cost to monitor inputs, similarly to QO. Nonetheless, the split candidate query also has a  $O(1)$  cost, which significantly surpasses all preceding AO algorithms for regression. We illustrate the main conceptual differences and similarities between ARF and OXT in Figure 3. In our experiments, we not only observe that OXT is much faster and uses fewer memory resources compared to ARF, but also is generally more accurate than the latter.

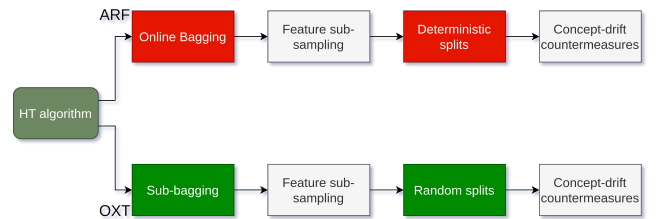


Figure 3. The ARF and OXT ensemble algorithms compared in terms of their main conceptual design choices. The main differences between the two algorithms are highlighted in red and green.

#### 4.6 Creating faster nearest neighbor queries in sliding windows

The last contribution of the thesis deviated from the HT framework and regression. We devised a new graph-based al-

gorithm to tackle nearest neighbor search in sliding windows. Back when developing Gomes *et al.* [2020], while comparing multiple regression algorithms against different strategies to compose ensemble regressors, we realized the potential of k-NN as an effective OML tool compared to other more complex algorithms. In the online setting, k-NN is typically performed using a sliding window of the instances most recently observed. Albeit simple, this strategy is capable of producing highly accurate predictions. The main drawback of this strategy lies in the time complexity of the solution, which historically relied on a linear scan of the data buffer to find the most similar items to the query, i.e., the instance for which one wants to make a prediction. For that reason, practical applications of k-NN were limited to small sliding windows, which might limit the accuracy of the predictions.

By extending a consolidated graph-based nearest neighbor search algorithm [Dong *et al.*, 2011; McInnes *et al.*, 2018] to the online setting, we were able to apply k-NN incrementally with increasingly large sliding window sizes. In Mastelini *et al.* [2024], the proposed Sliding Window-based Incremental Nearest Neighbors (SWINN) was shown to be much faster than performing a linear scan of the sliding window as the number of stored instances increases. By increasing the data window size, SWINN was able to increase the predictive performance in the classification and regression downstream tasks in which it was evaluated. Different from our previous contributions, SWINN is a generic solution that can be used in multiple OML tasks. Besides classification and regression, as evaluated in the paper, anomaly and outlier detection are other possible venues for future exploration left open for research with SWINN.

## 4.7 Discussion

Table 1 summarizes all the previously discussed main contributions of the thesis in terms of asymptotic cost analysis. In the case of decision trees, decision rules, and ensembles, the insertion, memory, and search query costs imply the time cost to insert a new point into the decision rule/tree structure, the total memory costs related to a single input feature, and the time cost to evaluate the best-split decision for a given input feature, respectively. In the expressions,  $n$  is the number of observed data stream points, and  $q$  is the number of stored elements in the QO algorithm. In the case of the nearest neighbor search solution,  $B$  is the branching factor, that is, the number of neighbors in the nearest neighbor search graph (which is data-dependent);  $h$  is the number of hops needed to go from a starting vertex in the graph to a target vertex (also data-dependent); and  $c$  is the number of neighbors explored for a certain vertex (a user-defined parameter). Insertion, memory, and search query costs correspond to the time cost to insert a single element in the data buffer, the total memory cost of the nearest neighbor search structure (the data buffer, or the data buffer plus the search graph, in our proposal), and the time cost to search for nearest neighbors, respectively.

In all the cases, the proposed algorithms significantly improve the preceding solutions in terms of computational costs. In the published papers, we also show that our proposals are not only computationally efficient but also competi-

tive in terms of predictive performance [Mastelini and de Carvalho, 2021; Mastelini *et al.*, 2021]. Our proposed online regression ensemble surpassed the state-of-the-art both in predictive performance and resource usage efficiency [Mastelini *et al.*, 2022]. It is worth noting that in the case of the nearest neighbor search, the insertion and memory costs had to be slightly increased in order to decrease the main source of computationally intensive operations: the search queries. As a result, the overall running time of the solution was significantly decreased compared to the original, exhaustive, nearest neighbor search algorithm.

A last highly impacting sub-product of the Ph.D. research was the joint creation, in collaboration with researchers from multiple countries, of the River Montiel *et al.* [2021] Python library for OML, from which the Ph.D. author is an active maintainer. River has since become the most popular tool for researchers and practitioners who want to develop OML solutions. Besides housing every theoretical collaboration mentioned so far, developing parts of the River Library fostered multiple collaborations with researchers from many countries that also resulted in additional publications that are not necessarily directly related to OML and the main thesis goals. These publications, while out of the scope of this current work and the general goals of the thesis development, demonstrate the power of open-source research in OML and how the whole community can benefit from open and reproducible research results.

## 5 Conclusion

By identifying research gaps in the existing literature, we followed the opposite path of most ongoing research in OML. Rather than focusing on predictive performance alone, we pursued building more efficient algorithms as the main objective, without sacrificing accuracy as a secondary goal. The main focal points of the research were decision trees, decision rules, and decision tree ensemble algorithms for regression, which are among the most popular practical solutions in OML applications. Regression tasks were historically overshadowed by classification tasks, which further motivated us to focus on this type of predictive task.

We envision multiple possible paths for future research and expansion of our work. Firstly, we would like to explore the performance of QO and its multi-branch trees when applied in ensembles properly parameterized, as discussed previously. We also want to explore strategies to improve the quantization used in QO and possibly select a custom quantization radius per input feature dynamically, without adding high computational cost overhead. We would like to explore alternatives to further foster model diversity in Online Extra Trees, considering the limitations of Hoeffding Trees on how long evidence must be gathered to enable splits, without incurring extra computational costs. Lastly, we would like to apply SWINN in machine learning tasks other than classification and regression, for instance, anomaly detection.

The active participation in the River open-source project, from which the thesis author is one of the founders and maintainers, is a constant inspiration to pursue further the original objectives envisioned during the thesis development. The

**Table 1.** Summary of the main contributions of the thesis in terms of asymptotic analysis

Task	Insertion		Memory		Search query		Main idea
	Before	Thesis	Before	Thesis	Before	Thesis	
Decision tree and decision rule building	$O(\log n)$	$O(1)$	$O(n)$	$O(q, q \ll n)$	$O(n)$	$O(q \log q)$	Quantization
Online regression ensemble		$O(1)$		$O(1)$		$O(1)$	Sampling
Nearest neighbor search	$O(1)$	$O(B(c^2 + 1) + c^h)$	$O(n)$	$O(n + Bn)$	$O(n)$	$O(c^h)$	Proximity graphs

community of users and researchers that gathered around the River project is also an effective source of information to better understand the needs of data stream mining practitioners and the research gaps that still need to be addressed.

## Declarations

## Acknowledgements

The authors would like to thank Professor João Gama for providing valuable guidance during the internship performed by the thesis author at the University of Porto, Portugal. The PhD thesis author would also like to thank the core River development team and the close friends and family around them for all the collaboration and wonderful friendship forged throughout the years.

## Funding

This research was funded by FAPESP through the grants #2018/07319-6 and #2021/10488-7.

## Authors' Contributions

SMM and ACPLFC conceptualized this study. SMM is the main contributor and writer of this manuscript. ACPLFC read and approved the final manuscript.

## Competing interests

The authors declare that they have no competing interests.

## Availability of data and materials

The datasets used in the thesis that originated this study are publicly available in open machine learning platforms, such as UCI, Kaggle and OpenML. The code is available in the River library.

## References

- Agrahari, S. and Singh, A. K. (2022). Concept drift detection in data stream mining: A literature review. *Journal of King Saud University-Computer and Information Sciences*, 34(10):9523–9540. DOI: 10.1016/j.jksuci.2021.11.006.
- Bahri, M., Bifet, A., Gama, J., Gomes, H. M., and Mani, S. (2021). Data stream analysis: Foundations, major tasks and tools. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(3):e1405. DOI: 10.1002/widm.1405.
- Bayram, F., Ahmed, B. S., and Kassler, A. (2022). From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems*, 245:108632. DOI: 10.1016/j.knosys.2022.108632.
- Bifet, A. and Gavaldà, R. (2009). Adaptive learning from evolving data streams. In *International Symposium on Intelligent Data Analysis*, pages 249–260. Springer. DOI: 10.1007/978-3-642-03915-7\_22.
- Bifet, A., Gavaldà, R., Holmes, G., and Pfahringer, B. (2018). *Machine Learning for Data Streams with Practical Examples in MOA*. MIT Press. Available at: <https://moa.cms.waikato.ac.nz/book/>.
- Cano, A. and Krawczyk, B. (2022). ROSE: robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams. *Machine Learning*, pages 1–39. DOI: 10.1007/s10994-022-06168-x.
- Chan, T. F., Golub, G. H., and LeVeque, R. J. (1982). Updating formulae and a pairwise algorithm for computing sample variances. In *COMPSTAT 1982 5th Symposium held at Toulouse 1982*, pages 30–41. Springer. DOI: 10.1007/978-3-642-51461-6\_3.
- Domingos, P. and Hulten, G. (2000). Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80, Boston, MA, USA. ACM. DOI: 10.1145/347090.347107.
- Dong, W., Moses, C., and Li, K. (2011). Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on World wide web*, pages 577–586. DOI: 10.1145/1963405.1963487.
- Duarte, J. and Gama, J. (2015). Multi-target regression from high-speed data streams with adaptive model rules. In *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*, pages 1–10, Campus des Cordeliers, Paris, France. IEEE. DOI: 10.1109/DSAA.2015.7344900.
- Duarte, J., Gama, J., and Bifet, A. (2016). Adaptive model rules from high-speed data streams. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(3):1–22. DOI: 10.1145/2829955.
- Esteban, A., Cano, A., Zafra, A., and Ventura, S. (2024). Hoffding adaptive trees for multi-label classification on data streams. *Knowledge-Based Systems*, page 112561. DOI: 10.1016/j.knosys.2024.112561.
- Gama, J. (2010). *Knowledge discovery from data streams*. Chapman and Hall/CRC. DOI: 10.3233/978-1-60750-611-9-125.
- García Martín, E. (2020). *Energy Efficiency in Machine Learning: Approaches to Sustainable Data Stream Mining*. PhD thesis, Blekinge Tekniska Högskola. Available at: [https://www.researchgate.net/publication/283666663\\_Energy\\_Efficiency\\_in\\_Data\\_Stream\\_Mining](https://www.researchgate.net/publication/283666663_Energy_Efficiency_in_Data_Stream_Mining).

- García-Martín, E., Rodrigues, C. F., Riley, G., and Grahn, H. (2019). Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134:75–88. DOI: 10.1016/j.jpdc.2019.07.007.
- Gomes, H. M., Barddal, J. P., Enembreck, F., and Bifet, A. (2017). A survey on ensemble learning for data stream classification. *ACM Computing Surveys (CSUR)*, 50(2):23. DOI: 10.1145/3054925.
- Gomes, H. M., Barddal, J. P., Ferreira, L. E. B., and Bifet, A. (2018). Adaptive random forests for data stream regression. In *26th European Symposium on Artificial Neural Networks, ESANN 2018, Bruges, Belgium, April 25–27, 2018*. Available at: [https://www.ppgia.pucpr.br/~jean.barddal/assets/pdf/arf\\_regression.pdf](https://www.ppgia.pucpr.br/~jean.barddal/assets/pdf/arf_regression.pdf).
- Gomes, H. M., Montiel, J., Mastelini, S. M., Pfahringer, B., and Bifet, A. (2020). On Ensemble Techniques for Data Stream Regression. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE. DOI: 10.1109/IJCNN48605.2020.9206756.
- Gomes, H. M., Read, J., Bifet, A., and Durrant, R. J. (2021). Learning from evolving data streams through ensembles of random patches. *Knowledge and Information Systems*, pages 1–29. DOI: 10.1007/s10115-021-01579-z.
- Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*. DOI: 10.1007/978-1-4612-0865-5\_26.
- Ikononovska, E., Gama, J., and Džeroski, S. (2011a). Incremental multi-target model trees for data streams. In *Proceedings of the 2011 ACM symposium on applied computing*, pages 988–993. ACM. DOI: 10.1145/1982185.1982402.
- Ikononovska, E., Gama, J., and Džeroski, S. (2011b). Learning model trees from evolving data streams. *Data mining and knowledge discovery*, 23(1):128–168. DOI: 10.1007/s10618-010-0201-y.
- Ikononovska, E., Gama, J., and Džeroski, S. (2015). Online tree-based ensembles and option trees for regression on evolving data streams. *Neurocomputing*, 150:458–470. DOI: 10.1016/j.neucom.2014.04.076.
- Kirkby, R. B. (2007). *Improving hoeffding trees*. PhD thesis, The University of Waikato. Available at: <https://researchcommons.waikato.ac.nz/server/api/core/bitstreams/125555d3-de75-43d0-a15c-42ad2d61a13c/content>.
- Knuth, D. E. (2014). *Art of computer programming, volume 2: Seminumerical algorithms*. Addison-Wesley Professional. Book.
- Korycki, Ł. and Krawczyk, B. (2020). Adaptive Deep Forest for Online Learning from Drifting Data Streams. *arXiv preprint arXiv:2010.07340*. DOI: 10.48550/arXiv.2010.07340.
- Manapragada, C., Salehi, M., and Webb, G. I. (2022). Extremely fast hoeffding adaptive tree. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 319–328. IEEE. DOI: 10.1109/ICDM54844.2022.00042.
- Mastelini, S. M. and de Carvalho, A. C. P. d. L. F. (2021). Using dynamical quantization to perform split attempts in online tree regressors. *Pattern Recognition Letters*, 145:37–42. DOI: 10.1016/j.patrec.2021.01.033.
- Mastelini, S. M., Montiel, J., Gomes, H. M., Bifet, A., Pfahringer, B., and de Carvalho, A. C. (2021). Fast and lightweight binary and multi-branch Hoeffding Tree Regressors. In *2021 International Conference on Data Mining Workshops (ICDMW)*, pages 380–388. IEEE. DOI: 10.1109/ICDMW53433.2021.00053.
- Mastelini, S. M., Nakano, F. K., Vens, C., de Leon Ferreira, A. C. P., et al. (2022). Online Extra Trees Regressor. *IEEE Transactions on Neural Networks and Learning Systems*. DOI: 10.1109/TNNLS.2022.3212859.
- Mastelini, S. M. and Ponce de Leon Ferreira de Carvalho, A. C. (2020). 2CS: correlation-guided split candidate selection in Hoeffding tree regressors. In *Brazilian Conference on Intelligent Systems*, pages 337–351. Springer. DOI: 10.1007/978-3-030-61380-8\_23.
- Mastelini, S. M., Veloso, B., Halford, M., de Leon Ferreira, A. C. P., Gama, J., et al. (2024). SWINN: Efficient nearest neighbor search in sliding windows using graphs. *Information Fusion*, 101:101979. DOI: 10.1016/j.inffus.2023.101979.
- McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*. DOI: 10.48550/arXiv.1802.03426.
- Montiel, J., Halford, M., Mastelini, S. M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H. M., Read, J., Abdessalem, T., and Bifet, A. (2021). River: machine learning for streaming data in Python. *Journal of Machine Learning Research*, 22(110):1–8. Available at: <https://www.jmlr.org/papers/v22/20-1380.html>.
- Osojnik, A. (2017). *Structured output prediction on data streams*. PhD thesis, Ph. D. thesis, Jozef Stefan International Postgraduate School. Available at: [https://kt.ijs.si/wp-content/uploads/2021/11/phd\\_aljaz\\_osojnik.pdf](https://kt.ijs.si/wp-content/uploads/2021/11/phd_aljaz_osojnik.pdf).
- Osojnik, A., Panov, P., and Džeroski, S. (2018). Tree-based methods for online multi-target regression. *Journal of Intelligent Information Systems*, 50(2):315–339. DOI: 10.1007/s10844-017-0462-7.
- Palli, A. S., Jaafar, J., Gilal, A. R., Alsughayyir, A., Gomes, H. M., Alshanqiti, A., and Omar, M. (2024). Online Machine Learning from Non-stationary Data Streams in the Presence of Concept Drift and Class Imbalance: A Systematic Review. *Journal of Information and Communication Technology*, 23(1):105–139. DOI: 10.32890/jict2024.23.1.5.
- Pfahringer, B., Holmes, G., and Kirkby, R. (2008). Handling numeric attributes in hoeffding trees. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 296–307. Springer. DOI: 10.1007/978-3-540-68125-0\_27.
- Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited., Book.
- Schubert, E. and Gertz, M. (2018). Numerically stable parallel computation of (co-) variance. In *Proceedings of the 30th international conference on scientific and statistical database management*, pages 1–12. DOI: 10.1145/3221269.3223036.
- Shimomura, L. C., Oyamada, R. S., Vieira, M. R., and Kaster,

D. S. (2021). A survey on graph-based methods for similarity searches in metric spaces. *Information Systems*, 95:101507. DOI: 10.1016/j.is.2020.101507.