




# Correlated Online k-Nearest Neighbors Regressor Chain for Online Multi-output Regression

Zipeng Wu<sup>1</sup>, Chu Kiong Loo<sup>1</sup>, and Kitsuchart Pasupa<sup>2</sup>(✉) 

<sup>1</sup> Faculty of Computer Science and Information Technology, University of Malaya,  
50603 Kuala Lumpur, Malaysia

s2013980@siswa.um.edu.my, ckloo.um@um.edu.my

<sup>2</sup> School of Information Technology, King Mongkut's Institute of Technology  
Ladkrabang, Bangkok 10520, Thailand

kitsuchart@it.kmitl.ac.th

**Abstract.** Online multi-output regression is a crucial task in machine learning with applications in various domains such as environmental monitoring, energy efficiency prediction, and water quality prediction. This paper introduces CONNRC, a novel algorithm designed to address online multi-output regression challenges and provide accurate real-time predictions. CONNRC builds upon the k-nearest neighbor algorithm in an online manner and incorporates a relevant chain structure to effectively capture and utilize correlations among structured multi-outputs. The main contribution of this work lies in the potential of CONNRC to enhance the accuracy and efficiency of real-time predictions across diverse application domains. Through a comprehensive experimental evaluation on six real-world datasets, CONNRC is compared against five existing online regression algorithms. The consistent results highlight that CONNRC consistently outperforms the other algorithms in terms of average Mean Absolute Error, demonstrating its superior accuracy in multi-output regression tasks. However, the time performance of CONNRC requires further improvement, indicating an area for future research and optimization.

**Keywords:** Multi output regression · Online machine learning · k-Nearest Neighbors

## 1 Introduction

Online machine learning, characterized by its real-time learning and adaptive capabilities [14], is increasingly emerging as a critical component in a wide range of applications, including river flow prediction and water quality assessment. The development of robust and efficient online machine learning algorithms is important, considering their aptitude for handling continuous data streams and adapting to evolving data patterns.

Despite the limited attention received by online multi-output regression, it is noteworthy that in an increasing number of applications, the need to predict multiple outputs rather than a single output is becoming more prevalent. Multi-output regression poses a greater challenge as it involves handling potential structural relationships between the outputs. Nevertheless, online multi-output regression holds significant potential in a variety of applications, e.g., Train Carriage Load prediction [22], water quality prediction [6].

Typically, multi-output regression can be categorized into global and local methods. Global methods output all output variables at once, while local methods often consist of multiple sub-models. However, existing multi-output regression algorithms are designed to learn a mapping from the input space to the output space on the entire training dataset, making them suitable for batch processing environments but challenging to apply in environments requiring online model updates.

Several online learning algorithms have been proposed [1, 3, 5, 8, 11, 15, 23] with Incremental Structured Output Prediction Tree (iSOUP-Tree) [15] and Passive-Aggressive (PA) [3] algorithms being notable examples. iSOUP-Tree is a tree-based method that simultaneously predicts all targets in multi-target regression tasks on data streams. On the other hand, PA is particularly suitable for single-output tasks and quickly adapts to data changes and updates model parameters analytically. It balances proximity to the current classifier and achieves a unit margin on the latest example. However, both algorithms face challenges in effectively capturing and exploiting correlations in multi-output regression tasks.

In this paper, we aim to address this problem by introducing a novel online multi-output regression algorithm called ‘‘Correlated Online k-Nearest Neighbors Regressor Chain’’ (CONNRC). Our proposed algorithm utilizes a maximum correlation chain structure to capture the associations among output variables while also leveraging the strengths of the k-nearest neighbors (kNN) algorithm.

The main contributions of this paper are as follows:

- We propose a novel online multi-output regression algorithm, CONNRC, which extends the kNN algorithm to handle online multi-output regression tasks. This algorithm leverages a maximum correlation chain structure to capture the association between output variables, thus addressing the challenge of handling potential structural relationships between the outputs in multi-output regression tasks.
- We comprehensively evaluate the proposed CONNRC algorithm on six real-world datasets. The evaluation includes a comparison with the existing online multi-output regression algorithm, iSOUP-Tree [15], and several classic online learning algorithms, including Hoeffding Adaptive Tree (HAT) [1], Adaptive Model Rules (AMRules) [5], Stochastic Gradient Trees (SGT) [8], and PA [3]. Our results demonstrate that CONNRC consistently outperforms other state-of-the-art algorithms in terms of average Mean Absolute Error (aMAE), signifying its superior accuracy in online multi-output regression tasks.

## 2 Related Work

The related work for this study can be broadly categorized into two main areas: online multi-output regression and classic online learning methods.

### 2.1 Online Multi-output Regression

Online multi-output regression is a crucial technique for modeling, predicting, and compressing multi-dimensional correlated data streams. The MORES method [11] dynamically learns the structure of regression coefficients and residual errors to improve prediction accuracy. It introduces three modified covariance matrices to extract necessary information from all seen data for training and sets different weights on samples to track the evolving characteristics of data streams. The iSOUP-Tree method [15] learns trees that predict all targets simultaneously. The iSOUP-optionTree extends the iSOUP-Tree through the use of option nodes and can be used as a base learner in ensemble approaches like online bagging and online random forest. The MORStreaming method [23] is another notable online multi-output regression method. It uses an instance-based model to make predictions and consists of two algorithms: an online algorithm based on topology networks to learn the instances and an online algorithm based on adaptive rules to learn the correlation between outputs automatically.

### 2.2 Classic Online Regression

Classic online learning methods have been extensively studied and applied in various domains. These methods typically focus on single-output regression tasks. The PA [3] is a well-known online learning algorithm for regression tasks. It is a margin-based online learning algorithm and has an analytical solution to update model parameters as new sample(s) arrive. The HAT [1] is a tree-based method for online regression. It uses an ADWIN concept-drift detector at each decision node to monitor possible changes in the data distribution. If a drift is detected in a node, an alternate tree begins to be induced in the background. When enough information is gathered, the node where the change was detected is swapped by its alternate tree. Lastly, SGT [8, 12] is a tree-based method for regression. It directly minimizes a loss function to guide tree growth and update their predictions, differentiating it from other incremental tree learners that do not directly optimize the loss but a data impurity-related heuristic. AMRules [5] is an online regression algorithm that constructs rules with attribute conditions as antecedents and linear combinations of attributes as consequents. It employs a Page-Hinkley test to identify and adapt to changes in the data stream, and includes outlier detection to prevent model skewing due to anomalous examples.

## 3 Proposed Method

The Correlated Online k-Nearest Neighbors Regressor Chain (CONNRC) algorithm, inspired by the Regressor Chain (RC) [21] and its variants [13, 17, 19, 20],

is designed for online multi-output regression tasks. It uses a chain structure to leverage correlations among multiple output variables, improving prediction performance. While the kNN algorithm forms its basis, CONNRC enhances it by incorporating correlation information, addressing a limitation of traditional online kNN in online multi-output regression tasks.

Throughout this paper, we use the following notation: a vector of output variables  $\mathbf{y} = (y_1, \dots, y_t, \dots, y_N)$ , where  $t = 1, \dots, N$ , with  $y_t$  representing the  $t$ -th sample and  $N$  being the total number of samples.  $\mathbf{Y}$  denotes a matrix of output variables,  $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_m)$ , where  $\mathbf{y}_i$  is the  $i$ -th vector of output variables and  $m$  is the total number of output variables.  $\mathbf{X}$  represents the input feature space, which is a matrix with  $N$  samples and  $d$  feature variables. The correlation coefficient of  $\mathbf{y}_i$  and  $\mathbf{y}_j$  can be denoted as  $c_{i,j}$ ,  $c_{i,j} = \text{corr}(\mathbf{y}_i, \mathbf{y}_j)$ . This is represented as  $\mathbf{COE}_{m \times m} = \text{corr}(\mathbf{Y})$ , where  $\mathbf{COE}$  is the correlation matrix of size  $m \times m$ , with  $m$  being the number of output variables. By summing each row of  $\mathbf{y}_i$ , we obtain the cumulative correlation value  $c_i$ , which indicates the degree of association between  $\mathbf{y}_i$  and all other variables in  $\mathbf{Y}$ . The correlation among output variables is calculated using the Spearman correlation coefficient, which has the advantage that there is no requirement for the distribution of the data, and the effect of the size of the absolute values can be ignored.  $\mathbf{I}$  represents the index vector corresponding to  $c_i$  that is the order of meta-model in the chain structure. It is the descending rank of each output variable cumulative correlation value  $c_i$ ,  $\mathbf{I} = (I_1, \dots, I_m)$ . This information serves as a guide for the order in which the kNN models are updated, based on the assumption that the output variables with higher cumulative correlation are more informative and should be predicted first.

A kNN model is initialized for each output variable, and a window of recent data points is maintained. The maximum window size,  $w$ , is a parameter that can be adjusted based on the specific requirements of the task. The window always contains the most relevant data points for prediction, which is ensured by a mechanism that adds the current point to the window if its distance to the nearest neighbor is greater than a certain threshold, *min\_distance\_keep*, and removes the oldest point in the window if the window is full.

For each training example in the stream,  $(\mathbf{x}, \mathbf{y})$ , the kNN model for each output variable is updated. For the first output variable, the model  $kNN_1$  is updated with the input features and the corresponding output value  $(\mathbf{x}, y_1)$ . For subsequent output variables, the model  $kNN_i$  is updated with the input features and the predicted values of the previous output variables  $(\mathbf{x}, y_i)$ .

After updating the model, the  $k_{neighbors}$  nearest neighbors for the current point  $(\mathbf{x}, y_i)$  are computed from the window. If the distance to the nearest neighbor is greater than a certain threshold, the current point is added to the window. If the window is full, the oldest point in the window is removed. This process ensures that the window always contains the most relevant data points for prediction. The pseudocode of CONNRC is illustrated in Algorithm 1.

---

**Algorithm 1:** Correlated Online k-Nearest Neighbors Regressor Chain

---

**Data:** Historical training dataset  $D = (\mathbf{X}_{history}, \mathbf{y}_{history})$ , and stream data  $(\mathbf{X}, \mathbf{y})$

**Result:** Online Multi-output regression model  $h_j, j = \{1, \dots, m\}$

$\mathbf{COE}_{m \times m} = \text{corr}(\mathbf{Y})$  ;

$c_l = \sum \mathbf{COE}(:, l), l = \{1, \dots, m\}$  ;

$[c', \mathbf{I}] = \text{rank}(c_l, \text{'decending'})$ , where  $\mathbf{I} = (I_1, \dots, I_m)$  ;

**for**  $y_i$  *in*  $\mathbf{y}_{1,m}$  **do**

    Let  $kNN_i$  be a kNN model for the output  $y_i$ ;

    Initialize the window  $W$  for  $kNN_i$  with a maximum window size  $w$ ;

**end**

**for** *all training examples*  $(\mathbf{x}, \mathbf{y})$  *in stream* **do**

**if**  $i = 1$  **then**

        Update  $kNN_1$  with  $(\mathbf{x}, y_1)$ ;

**else**

$\mathbf{x} = \mathbf{x}_i \cup (y_1 \dots y_{i-1})$ ;

        Update  $kNN_i$  with  $(\mathbf{x}, y_i)$ ;

**end**

**for** *each*  $y_i$  *in*  $\mathbf{y} = (y_{I_1}, \dots, y_{I_m})$  **do**

        Compute the  $k_{neighbors}$  nearest neighbors **for**  $(\mathbf{x}, y_i)$  *from the window*

**do**

            Calculating the Euclidean distance between  $(\mathbf{x}, y_i)$  and all points in the window;

            Selecting the  $k_{neighbors}$  points with the smallest distances;

**end**

**if** *the distance to the nearest neighbor*  $>$   $min\_distance\_keep$  **then**

            Add  $(\mathbf{x}, y_i)$  to the window of  $kNN_i$ ;

**if**  $\text{size}(W) > w$  **then**

                Remove  $W[0]$ ;

**end**

**end**

**end**

**end**

---

In this algorithm, the historical dataset is denoted as  $D = (\mathbf{X}_{history}, \mathbf{y}_{history})$ . For the current experiment, we utilize the first 25% of the entire dataset as our training data.

The CONNRC algorithm has several advantages. First, it considers the correlation among multiple output variables by correlated chain structure, which can improve the prediction performance in multi-output regression tasks. Second, it is based on the simple yet effective kNN algorithm, which makes it easy to implement and understand. It is worth noting that CONNRC consists of two hyperparameters: the maximum window size ( $w$ ) and the threshold for adding points to the window ( $min\_distance\_keep$ ). These hyperparameters need to be tuned. Furthermore, the computation of the  $k_{neighbors}$  nearest neighbors can be time-consuming, especially when the window size is large.

## 4 Experiment Framework

In this section, we present the experimental framework used to evaluate the performance of our proposed method, CONNRC, in comparison with several state-of-the-art online regression algorithms. All experiments were implemented in Python 3.9 and executed on a PC with an Intel Core i7 12700 processor (4.90 GHz) and 16 GB of RAM.

### 4.1 Datasets

Our proposed method, CONNRC, underwent evaluation on six real-world datasets, each exhibiting unique characteristics and posing specific challenges. These datasets were chosen based on their previous utilization in online multi-output regression research [4, 15] and their role as benchmark datasets in the field of multi-output regression [21]. The datasets used in our evaluation are as follows:

- Water Quality Prediction [6]: It comprises 16 input attributes that are associated with physical and chemical water quality parameters. Additionally, it includes 14 target attributes that represent the relative presence of plant and animal species in Slovenian rivers.
- Supply Chain Management Prediction [9, 16]: It is derived from the Trading Agent Competition in Supply Chain Management (TAC SCM) tournament from 2010. This consists of two sub-datasets. The dataset consists of 16 regression targets, where each target corresponds to either the mean price for the next day (SCM1D) [9] (as the first sub-dataset) or the mean price for a 20-day period in the future (SCM20D) [16] (as the second sub-dataset) for each product within the simulation.
- River flow Prediction [21]: This dataset consists of hourly flow observations collected from 8 sites within the Mississippi River network in the United States. The data was obtained from the US National Weather Service. Each row of the dataset contains the most recent observation for each of the eight sites and time-lagged observations from 6, 12, 18, 24, 36, 48, and 60 h in the past.
- Energy Building Prediction [18]: The focus of this dataset is the prediction of heating load and cooling load requirements for buildings. The prediction is based on eight building parameters, including glazing area, roof area, overall height, and other relevant factors.
- Sea Water Quality Prediction [10]: The Andromeda dataset focuses on predicting future values for six water quality variables, namely temperature, pH, conductivity, salinity, oxygen, and turbidity. The dataset pertains specifically to the Thermaikos Gulf of Thessaloniki in Greece. The predictions are made using a window of five days and a lead time of five days, indicating that the model aims to forecast the values of these variables five days ahead based on the observations from the previous five days.

These datasets represent a variety of application domains and provide a comprehensive basis for evaluating the performance of the proposed method.

## 4.2 Experiment Settings

To evaluate the performance of each multi-output method, we employed the prequential strategy introduced in [7]. In this strategy, a sample is used to update the model after it has been evaluated by this model. This approach allows us to simulate a real-time learning environment, which is a key characteristic of online machine learning. We also employed progressive validation, as suggested in [2]. This method is considered the canonical way to evaluate a model’s performance, as it allows us to accurately assess how a model would have performed in a production scenario. In progressive validation, the dataset is transformed into a sequence of queries and responses. For each step, the model is tasked with predicting an observation or undergoing an update, and the samples are processed one after the other.

In our comparison, we assessed the performance of our proposed algorithm, CONNRC, in comparison to the state-of-the-art iSOUP-Tree, as well as several classic online learning algorithms: PA, AMRules, HAT, and SGT. It’s important to note that our task involves multi-output regression, while these classical online learning algorithms were originally designed for single-output tasks. We trained separate models for each output variable to adapt them to the multi-output setting, denoting them with the prefix “MT-”. The reason for choosing single-output classical online learning algorithms as baselines is that online multi-output regression currently receives little attention, and few open-source algorithms are available for this purpose. These baseline algorithms represent a diverse range of approaches to online regression, providing a comprehensive benchmark for evaluating the performance of our proposed method, CONNRC.

## 4.3 Performance Evaluation

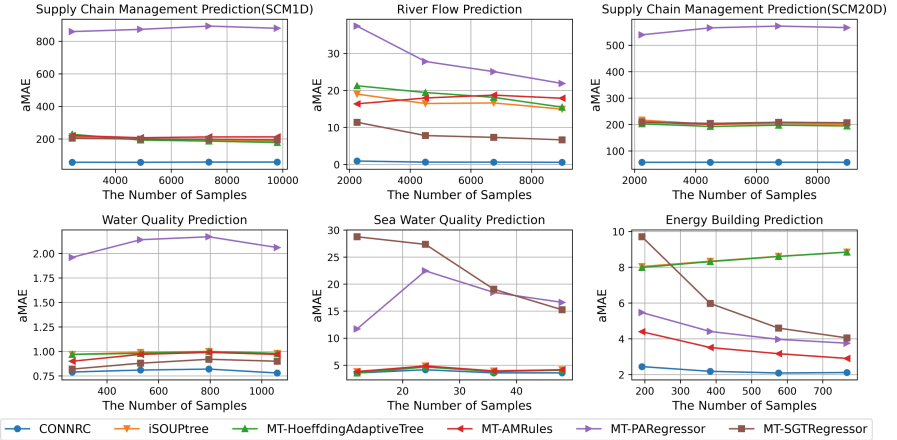
To evaluate the overall performance of the model, we conducted performance evaluations at regular intervals, incrementally increasing the data sample by 25%. This approach allowed us to monitor the model’s performance as the dataset size grew and gain a comprehensive understanding of its effectiveness across various sample sizes. In our evaluation, we considered several metrics to assess the performance of the model, including average Mean Absolute Error (aMAE), memory usage, and time. It is important to note that smaller values for all three metrics indicate better performance. The aMAE can be calculated by the following equation:

$$\text{aMAE} = \frac{1}{m} \sum_m^{t=1} \frac{1}{N} \sum_N^{i=1} |y_i^t - \hat{y}_i^t|. \quad (1)$$

## 5 Experiment Results and Discussion

Figure 1 illustrates the accuracy performance of the algorithms based on the average Mean Absolute Error (aMAE). Our proposed method, CONNRC, consistently outperforms the other algorithms by achieving the lowest aMAE values

across all sample sizes and datasets. The performance of CONNRC remains stable and promising at every regular checkpoint, whereas the performance of the other algorithms fluctuates. This underscores the effectiveness of CONNRC in capturing and leveraging the correlations among multiple outputs, resulting in enhanced prediction accuracy.



**Fig. 1.** Comparison of aMAE by different algorithms at regular intervals of 25% data sample incremental

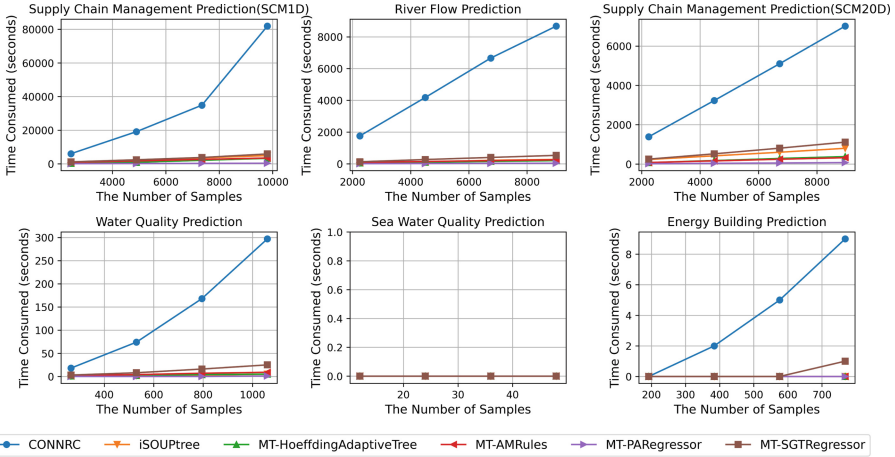
While CONNRC demonstrates superior accuracy compared to other methods, it also exhibits the highest time consumption across all datasets, as depicted in Fig. 2. This can be attributed to the time complexity associated with searching for the nearest neighbors in the kNN algorithm, indicating a potential area for future improvement. Nevertheless, it is important to note that CONNRC still maintains reasonable efficiency in certain cases, suggesting a balance between accuracy and computational speed.

It is worth noting that although MT-PARegressor demonstrates the poorest accuracy in 4 out of 6 datasets, it stands out as the fastest algorithm. This highlights the potential trade-off between accuracy and computational efficiency when selecting an algorithm for specific applications.

Figure 3 illustrates the memory usage comparison among the algorithms. It is evident that MT-PARegressor exhibits the highest efficiency in terms of memory consumption across all datasets. In contrast, in most cases, CONNRC and MT-SGTRegressor tend to consume more memory. However, it is worth noting that CONNRC demonstrates reasonable efficiency, particularly in memory usage, on certain datasets, such as the River Flow dataset.

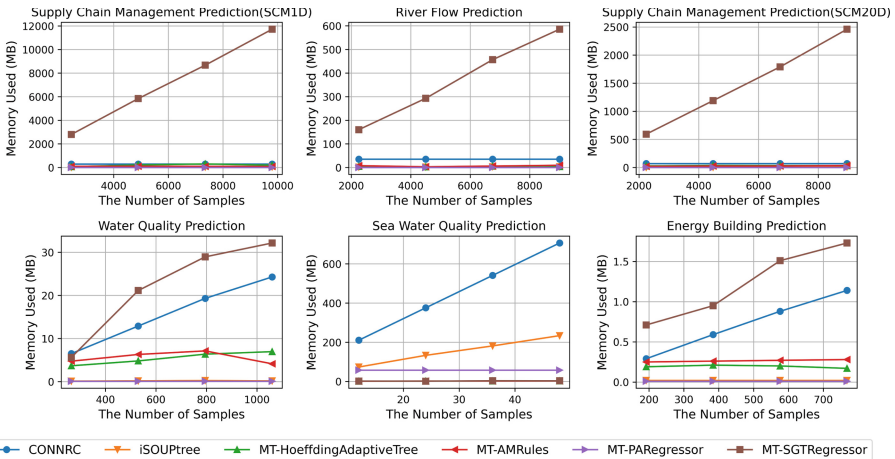
One key difference between the online kNN algorithm and other online learning algorithms lies in their approach to handling data. While both algorithms learn from each sample once, the kNN algorithm retains the data, allowing its performance to be comparable to batch algorithms that have the ability to





**Fig. 2.** Comparison of computational time by different algorithms at regular intervals of 25% data sample incremental

review each sample multiple times. On the other hand, typical online learning algorithms, which do not store data, often do not achieve the same level of robustness and accuracy as batch algorithms. In the case of CONNRC, our proposed method, we leverage the advantage of data retention inherited from kNN. Additionally, CONNRC incorporates a relevance chain structure to capture and exploit the correlations among structured multi-outputs. Moreover, the neighboring meta-models in CONNRC utilize the output of the preceding meta-model in the chain as latent information. Similar to knowledge distillation, this process



**Fig. 3.** Comparison of memory usage by different algorithms at regular intervals of 25% data sample incremental

enhances CONNRC's robustness against concept drift, allowing it to adapt to changing data patterns.

## 6 Conclusion and Future Work

This paper introduced and evaluated CONNRC, an online multi-output regression algorithm, using six different datasets. Our experimental results demonstrated that CONNRC consistently outperformed other algorithms in terms of accuracy, as measured by the aMAE metric. This indicates that CONNRC is capable of providing more accurate predictions in online multi-output regression tasks compared to the evaluated algorithms. However, it is important to note that the time performance of CONNRC was relatively slower compared to the other algorithms. This could be a potential limitation in scenarios where real-time predictions are crucial and time efficiency is a primary concern. While CONNRC demonstrated reasonable efficiency in terms of memory usage, it was not the most memory-consuming among the tested algorithms.

Despite these limitations, the superior accuracy performance of CONNRC underscores its potential for online multi-output regression tasks. The algorithm's ability to adapt to changes in the data stream and make accurate predictions is a significant advantage in many real-world applications, such as environmental monitoring, energy efficiency prediction, and water quality prediction.

For future work, we have plans to improve the efficiency of CONNRC, particularly its time performance. One potential approach is to enhance the nearest neighbor search in the kNN component of the algorithm, as it currently constitutes the most time-consuming part. Through optimization of this process, we aim to significantly reduce the prediction time, making CONNRC more suitable for real-time applications. Given the potential application scenarios involving massive streaming data, we will seek large datasets to validate the performance of our algorithms in such environments.

## References

1. Bifet, A., Gavaldà, R.: Adaptive learning from evolving data streams. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) IDA 2009. LNCS, vol. 5772, pp. 249–260. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-03915-7\\_22](https://doi.org/10.1007/978-3-642-03915-7_22)
2. Blum, A., Kalai, A., Langford, J.: Beating the hold-out: bounds for K-fold and progressive cross-validation. In: Ben-David, S., Long, P.M. (eds.) Proceedings of the Twelfth Annual Conference on Computational Learning Theory, COLT 1999, Santa Cruz, CA, USA, 7–9 July 1999, pp. 203–208. ACM (1999). <https://doi.org/10.1145/307400.307439>
3. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. *J. Mach. Learn. Res.* **7**, 551–585 (2006)
4. Duarte, J., Gama, J.: Multi-target regression from high-speed data streams with adaptive model rules. In: 2015 IEEE International Conference on Data Science

- and Advanced Analytics, DSAA 2015, Campus des Cordeliers, Paris, France, 19–21 October 2015, pp. 1–10. IEEE (2015). <https://doi.org/10.1109/DSAA.2015.7344900>
5. Duarte, J., Gama, J., Bifet, A.: Adaptive model rules from high-speed data streams. *ACM Trans. Knowl. Discov. Data* **10**(3), 30:1–30:22 (2016). <https://doi.org/10.1145/2829955>
  6. Dzeroski, S., Demsar, D., Grbovic, J.: Predicting chemical parameters of river water quality from bioindicator data. *Appl. Intell.* **13**(1), 7–17 (2000). <https://doi.org/10.1023/A:1008323212047>
  7. Gama, J.: *Knowledge Discovery from Data Streams*. CRC Press (2010)
  8. Gouk, H., Pfahringer, B., Frank, E.: Stochastic gradient trees. In: Lee, W.S., Suzuki, T. (eds.) *Proceedings of The 11th Asian Conference on Machine Learning, ACML 2019, 17–19 November 2019, Nagoya, Japan*. *Proceedings of Machine Learning Research*, vol. 101, pp. 1094–1109. PMLR (2019)
  9. Groves, W., Gini, M.: Improving prediction in TAC SCM by integrating multivariate and temporal aspects via PLS regression. In: David, E., Robu, V., Shehory, O., Stein, S., Symeonidis, A. (eds.) *AMEC/TADA -2011. LNBIP*, vol. 119, pp. 28–43. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-34889-1\\_3](https://doi.org/10.1007/978-3-642-34889-1_3)
  10. Hatzikos, E.V., Tsoumakas, G., Tzanis, G., Bassiliades, N., Vlahavas, I.P.: An empirical study on sea water quality prediction. *Knowl. Based Syst.* **21**(6), 471–478 (2008). <https://doi.org/10.1016/j.knosys.2008.03.005>
  11. Li, C., Wei, F., Dong, W., Wang, X., Liu, Q., Zhang, X.: Dynamic structure embedded online multiple-output regression for streaming data. *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(2), 323–336 (2019). <https://doi.org/10.1109/TPAMI.2018.2794446>
  12. Mastelini, S.M., de Leon Ferreira de Carvalho, A.C.P.: Using dynamical quantization to perform split attempts in online tree regressors. *Pattern Recognit. Lett.* **145**, 37–42 (2021). <https://doi.org/10.1016/j.patrec.2021.01.033>
  13. Melki, G., Cano, A., Kecman, V., Ventura, S.: Multi-target support vector regression via correlation regressor chains. *Inf. Sci.* **415**, 53–69 (2017). <https://doi.org/10.1016/j.ins.2017.06.017>
  14. Montiel, J., et al.: River: machine learning for streaming data in Python. *J. Mach. Learn. Res.* **22**, 110:1–110:8 (2021)
  15. Osojnik, A., Panov, P., Dzeroski, S.: Tree-based methods for online multi-target regression. *J. Intell. Inf. Syst.* **50**(2), 315–339 (2018). <https://doi.org/10.1007/s10844-017-0462-7>
  16. Pardoe, D., Stone, P.: The 2007 TAC SCM prediction challenge. In: Ketter, W., La Poutré, H., Sadeh, N., Shehory, O., Walsh, W. (eds.) *AMEC/TADA -2008. LNBIP*, vol. 44, pp. 175–189. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-15237-5\\_13](https://doi.org/10.1007/978-3-642-15237-5_13)
  17. Read, J., Martino, L.: Probabilistic regressor chains with monte Carlo methods. *Neurocomputing* **413**, 471–486 (2020). <https://doi.org/10.1016/j.neucom.2020.05.024>
  18. Tsanas, A., Xifara, A.: Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy Build.* **49**, 560–567 (2012)
  19. Wu, Z., Lian, G.: A novel dynamically adjusted regressor chain for taxi demand prediction. In: *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, 19–24 July 2020*, pp. 1–10. IEEE (2020). <https://doi.org/10.1109/IJCNN48605.2020.9207160>

20. Wu, Z., Loo, C.K., Pasupa, K., Xu, L.: An interpretable multi-target regression method for hierarchical load forecasting. In: Tanveer, M., Agarwal, S., Ozawa, S., Ekbal, A., Jatowt, A. (eds.) *Neural Information Processing - 29th International Conference, ICONIP 2022, Virtual Event, 22–26 November 2022, Proceedings, Part VII*. CCIS, vol. 1794, pp. 3–12. Springer, Singapore (2022). [https://doi.org/10.1007/978-981-99-1648-1\\_1](https://doi.org/10.1007/978-981-99-1648-1_1)
21. Xioufis, E.S., Tsoumakas, G., Groves, W., Vlahavas, I.P.: Multi-target regression via input space expansion: treating targets as inputs. *Mach. Learn.* **104**(1), 55–98 (2016). <https://doi.org/10.1007/s10994-016-5546-z>
22. Yu, H., Lu, J., Liu, A., Wang, B., Li, R., Zhang, G.: Real-time prediction system of train carriage load based on multi-stream fuzzy learning. *IEEE Trans. Intell. Transp. Syst.* **23**(9), 15155–15165 (2022). <https://doi.org/10.1109/TITS.2021.3137446>
23. Yu, H., Lu, J., Zhang, G.: MORStreaming: a multioutput regression system for streaming data. *IEEE Trans. Syst. Man Cybern. Syst.* **52**(8), 4862–4874 (2022). <https://doi.org/10.1109/TSMC.2021.3102978>