SAMknn Regressor for Online Learning in Water Distribution Networks

Jonathan Jakob, André Artelt, Martina Hasenjäger, Barbara Hammer

2022

Preprint:

This is a post-peer-review, pre-copyedit version of an article published in International Conference on Artificial Neural Networks (ICANN) 2022. The final authenticated version is available online at:
https://doi.org/10.1007/978-3-031-15934-3_62

SAM-kNN Regressor for Online Learning in Water Distribution Networks

Jonathan Jakob[1], André Artelt[1,3(✉)], Martina Hasenjäger[2], and Barbara Hammer[1]

[1] Bielefeld University, Bielefeld, Germany
aartelt@techfak.uni-bielefeld.de
[2] Honda Research Institute, Offenbach, Germany
[3] University of Cyprus, Nicosia, Cyprus

Abstract. Water distribution networks are a key component of modern infrastructure for housing and industry. They transport and distribute water via widely branched networks from sources to consumers. In order to guarantee a working network at all times, the water supply company continuously monitors the network and takes actions when necessary – e.g. reacting to leakages, sensor faults and drops in water quality. Since real world networks are too large and complex to be monitored by a human, algorithmic monitoring systems have been developed. A popular type of such systems are residual based anomaly detection systems that can detect events such as leakages and sensor faults. For a continuous high quality monitoring, it is necessary for these systems to adapt to changed demands and presence of various anomalies.

In this work, we propose an adaption of the incremental SAM-kNN classifier for regression to build a residual based anomaly detection system for water distribution networks that is able to adapt to any kind of change.

Keywords: SAM-kNN regressor · Incremental · Anomaly detection

1 Introduction

Water is the foundation of (our) life – we need water for drinking, cooking, hygiene and farming. Water distribution networks (WDNs), which distribute water from the supplier to the customers, are therefore considered as critical infrastructure. A major problem for water utility companies (and society in general) are anomalies that cause loss or contamination of water – e.g. leakages such as pipe bursts, sensor faults, pollution, cyber-physical attacks, etc. [1,3,15].

J. Jakob and A. Artelt—Contributed equally.

We gratefully acknowledge funding from the VW-Foundation for the project *IMPACT* funded in the frame of the funding line *AI and its Implications for Future Society*, and funding from the European Research Council (ERC) under the ERC Synergy Grant Water-Futures (Grant agreement No. 951424).

E. Pimenidis et al. (Eds.): ICANN 2022, LNCS 13531, pp. 752–762, 2022.
https://doi.org/10.1007/978-3-031-15934-3_62

Because of water shortages, among others caused by climate change, (drinking) water becomes an increasingly valuable resource that should not be wasted. However, it was estimated that leakages in WDNs lead to a loss of more than 45 million m^3 of drinking water in developing countries – even highly developed countries such as the island of Cyprus looses approx. up to 25% of their drinking water due to leaky pipes [11].

Because of the increasing availability of sensors (e.g. pressure sensors) in WDNs, water utility companies nowadays use computer systems for (autonomously) monitoring their networks [14]. These systems are realized using methods from engineering, statistics and machine learning (ML) [2]. While many different and successful methods for anomaly detection and localization have been proposed [17], these methods are usually not able to adapt to an occurring change or anomaly. These systems might be able to detect anomalies but once the anomaly is detected, they are "blind" for everything else that happens while the detected anomaly is present – the systems must be recalibrated or refitted which becomes challenging because a large amount of data (and therefore collection time) is needed. Adaptation to changes – in case of WDNs anomalies or simply changes in the water consumption behavior of the customers (i.e. changed demand) – can be natural handled by online learning methods.

Online learning [4] can be considered as a sub-field of machine learning which deals with models that are trained incrementally – i.e. they can learn from a data stream instead of a fixed training set only. For example, they can be used for electricity price prediction [20] or electric load forecasting [19].

In this work, we contribute to online learning for regression problems and water distribution networks as a particular field of application. More specifically, our contributions are:

- We propose SAM-kNN regression, a memory based online learner for regression problems.
- We evaluate our proposed SAM-kNN regression method in the context of anomaly detection in water distribution networks.

The remainder of this paper is structured as follows: After briefly reviewing related work in Sect. 2, we introduce the problem setting we are considering in this work (see Sect. 3). Next, in Sect. 4 we propose SAM-kNN regression for online learning, which we empirically evaluate in the context of anomaly detection in water distribution networks (see Sect. 5). Finally, this work closes with a summary and conclusion in Sect. 6.

2 Related Work

Incremental or online learning is a machine learning paradigm in which a model is updated after each data sample that is fed into it. This paradigm is especially suited for large data sets, that are too big to be processed in batch fashion, or in situations where data becomes available only sample after sample – e.g. in the form of a potentially infinite stream of data. Incremental learning has made

great strides in recent years [5], with most applications set in a classification environment [7,9,10,12,18,21].

One particular system is the SAM-kNN classifier [13]. This incremental algorithm was created to perform on long data streams, using its internal memory structure to alleviate the problem of catastrophic forgetting when frequent concept changes are expected.

On the other hand, only a few approaches that utilize incremental learning for regression problems exist. In [19] the authors use an incremental variant of the support vector regressor (SVR) to build models for electricity price prediction. [20] uses a similar SVR, paired with phase space reconstruction for time series to facilitate electrical load forecasting. In [6] a wide range of incremental regression algorithms are compared for their use in exoskeleton control.

However, all of these publications utilize standard incremental algorithms and do not explicitly build a new model to work with. We, on the other hand, propose a reformulation of the SAM-kNN classifier for regression as a standalone algorithm.

3 Problem Setting

Incremental or online regression is the task of predicting a response variable $y \in \mathbb{R}$ from a stream $S = \{x_1, x_2, x_3, ...\}$ of variables $X \in \mathbb{R}^n$. Hereby, a new instance of the incremental model is learned for each incoming sample of the data stream.

We work on water distribution networks that have several internal nodes n. These nodes are equipped with sensors, that measure water pressure and flow rate. Each sensor provides read outs at specific time intervals t. For every node, this creates a potentially infinite data stream of sensors values $S = \{(s_1, y_1), (s_2, y_2), (s_3, y_3), ...\}$, where $s_i \in \mathbb{R}^{n-1}$ represents the sensor values of all but one node in the network and the predictor variable $y_i \in \mathbb{R}$ represents the sensor value at the remaining node.

This means, that we use the read outs of $n-1$ nodes to predict the value of the nth node using an incremental regression algorithm. Said algorithm processes the stream S instance after instance by generating a sequence of models $H = \{h_1, h_2, h_3, ...\}$, where $h_{i-1}(s_i) = \hat{y}_i$. After each prediction the true value y_i is revealed and a new model h_i is learned.

We use the *Interleaved train test error (ITTE)* as a cost function to train the model:

$$E(S) = \sqrt{\frac{1}{t}\sum_{i=1}^{t}(h_{i-1}(s_i) - y_i)^2} \tag{1}$$

This ITTE measures the *Root Mean Squared Error (RMSE)* over every model h_i up to a given time point t.

Whenever the local error $h_{i-1}(s_i) - y_i$ exceeds a certain threshold, this means, that more water than predicted flows through the observed node. This is taken as

an indication for a water leak and an alarm will be triggered. Being an incremental algorithm, our model will then automatically adjust to the new circumstances so that accurate prediction of the water flow will be maintained throughout the leak. Our model, which will be explained in detail in the next section, has the capability to remember long term concepts and therefore, as soon as the water leak is fixed, it will revert back to the normal circumstances.

4 Model

Our proposed model is an adaption of the Self Adjusting Memory (SAM) [13], an incremental classifier, to regression. This approach is based on two distinct internal memories, the Short-Term (STM) and the Long-Term memory (LTM). Hereby, the STM is a dynamic sliding window over the last m samples, that is supposed to only hold the most recent concept of the data stream:

$$M_{ST} = \{(x_i, y_i) \in \mathbb{R}^n \times \mathbb{R} \mid i = t - m + 1, ..., t\} \tag{2}$$

The LTM, on the other hand, is a collection of p samples, which hold older concepts, that do not contradict the STM and might still be of use in the future:

$$M_{LT} = \{(x_i, y_i) \in \mathbb{R}^n \times \mathbb{R} \mid i = 1, ..., p\} \tag{3}$$

Additionally, there is the combined memory (CM), which is a simple union of the STM and the LTM:

$$M_C = M_{ST} \cup M_{LT} \tag{4}$$

Each memory induces a kNN regressor that can be used independently from the others. To determine which kNN is used for every new incoming data sample, the ITTE (see Sect. 3) is tracked for all sub-models and the one with the lowest current ITTE is chosen.

4.1 Model Parameters

The proposed model has three parameters that are continuously adapted during deployment:

1. The size m of the STM sliding window
2. The data samples in the LTM
3. The ITTEs of the three sub-models

Additionally, there are three hyperparameters that can be chosen robustly and are set before deployment:

1. The number of neighbours k
2. The minimum size L_{min} of the STM
3. The maximum size L_{max} of the LTM

4.2 Model Adaption

Whenever a new data sample arrives, it is added to the STM, which means that this memory grows continuously. However, since it is supposed to hold only the most recent concept, a reduction of the STM window size is performed on a regular basis. This is facilitated by testing smaller window sizes at every iteration and choosing the one that is optimizing the ITTE. Tested windows are:

$$M_l = \{(x_{t-l+1}, y_{t-l+1}), ..., (x_t, y_t)\} \tag{5}$$

where $l \in \{m, m/2, m/4...\}$ and $l \geq L_{min}$.

$$M_{ST_{t+1}} = \underset{S \in \{M_m, M_{m/2}, ...\}}{\operatorname{argmin}} E(S) \tag{6}$$

Whenever the STM is shrunk in size, the data samples O_t that fall out of the sliding window are not discarded.

$$O_t = M_{ST_t} \setminus M_{ST_{t+1}} \tag{7}$$

Instead, they undergo a cleaning process, and those, that are still consistent with the new STM are added to the LTM. Afterwards, the whole of the LTM is cleaned as well, to ensure consistency with the STM at all times. When the LTM reaches its maximum size, samples get discarded in a way that ensures minimal information loss.

4.3 Cleaning Process

The process to clean a set of samples with respect to the STM is defined in the following way:
A set A is cleaned by another set B regarding an example $(x_i, y_i) \in B$

$$clean : (A, B, (x_i, y_i)) \mapsto \hat{A} \tag{8}$$

where $A, B, \hat{A} \subset \mathbb{R}^n \times \mathbb{R}$ and $(x_i, y_i) \in B$.

$\hat{A}$ is defined in five steps:

1. Determine the k nearest neighbours of x_i in $B \setminus (x_i, y_i)$ and find the maximum distance

$$\Delta_x^* = \max \left\{ d(x_i, x) \mid x \in N_k(x_i, B \setminus (x_i, y_i)) \right\} \tag{9}$$

2. Compute the maximum weighted difference of y_i and $y \in N_k(x_i, B \setminus (x_i, y_i))$

$$\Delta_y^* = \max \left\{ \left(\frac{y_i - y}{e^{\frac{x_i - x}{\Delta_x^*}}} \right) \mid y \in N_k(x_i, B \setminus (x_i, y_i)) \right\} \tag{10}$$

3. Determine all samples in A that are within Δ_x^* of x_i

$$C = \left\{(x, y) \in A \mid d(x_i, x) < \Delta_x^*\right\} \tag{11}$$

4. Compute the weighted differences of y_i and $y \in C$

$$\Delta_y = \left\{ \left(\frac{y_i - y}{e^{\frac{x_i - x}{\Delta_x^*}}} \right) \mid y \in C \right\} \tag{12}$$

5. Discard samples from C that have a larger weighted difference than Δ_y^*

$$\hat{A} = A \setminus \{(x, y) \in C \mid \Delta_y(x) > \Delta_y^*\} \tag{13}$$

Furthermore, the cleaning operation for the full set B

$$clean : (A, B) \mapsto \hat{A}_{|B|} \tag{14}$$

is defined by iteratively applying the former cleaning for all $(x_i, y_i) \in B$

$$\begin{aligned} \hat{A}_0 &= A \\ \hat{A}_{t+1} &= clean(\hat{A}_t, B, (x_{t+1}, y_{t+1})) \end{aligned}$$

In summary, when the STM is shrunk in size, the process to clean the discarded set O_t is described by the operation:

$$clean(O_t, M_{ST_{t+1}}) \tag{15}$$

After that, the LTM is cleaned as well:

$$clean(M_{LT_t}, M_{ST_{t+1}}) \tag{16}$$

4.4 Compression of the LTM

When new samples are added to the LTM while it reaches maximum capacity, old samples need to be discarded. To avoid a significant information loss, samples are discarded in an iterative process one after another until $|M_{LT}| < L_{max}$ again. Hereby, the data sample with the lowest distance to any other two samples is chosen for every iteration of the discarding process.

4.5 Final Model

The complete pseudocode of our proposed model is given in Algorithm 1.

Algorithm 1. SAM-kNN Regression

Input: Data stream S, one s_i at a time
Output: $\hat{y}_i$ for every s_i

$M_{ST}, M_{LT} = \{s_0, ..., s_{L_{min}}\}$ ▷ Initialize STM and LTM
$E_{ST}, E_{LT}, E_C = 0$ ▷ Initialize tracked errors
for $s_i \in S \setminus \{s_0, ..., s_{L_{min}}\}$ **do** ▷ Loop over the remaining data stream
  $BM = argmin(E_{ST}, E_{LT}, E_C)$ ▷ Find best memory with lowest error
  $\hat{y}_i = kNN_{BM}(s_i)$ ▷ Predict with kNN of best memory
  Update E_{ST}, E_{LT}, E_C
  $M_{ST} = M_{ST} \cup \{s_i\}$ ▷ Add current sample to STM
  Evaluate smaller STM sizes
  if STM is reduced **then**
    $O_t = M_{ST_t} \setminus M_{ST_{t+1}}$ ▷ Take discarded samples from STM
    $clean(O_t, M_{ST_{t+1}})$ ▷ Clean discarded samples with respect to new STM
    $M_{LT} = M_{LT} \cup clean(O_t, M_{ST_{t+1}})$ ▷ Add cleaned samples to LTM
    $clean(M_{LT}, M_{ST_{t+1}})$ ▷ Clean new LTM with respect to new STM
  end if
end for

5 Experiments

We empirically evaluate our proposed method in an online scenario for detecting leakages and sensor faults in water distribution networks – all experiments are implemented in Python[1].

5.1 Data

We use a version of the L-Town water distribution network as used in [16], as a prominent realistic benchmark for anomaly detection – we only use Area A which consists of 661 nodes, 766 links, and 29 (optimally placed) pressure sensors. We build and simulate 10 scenarios where the first 5 scenarios each contain a single leakage – we vary position, time and size of the leakage – and the remaining 5 scenarios each contain a single different sensor fault (position is varied):

- Scenario 6: Sensor measurement is overflowing over time – i.e. it is going to infinity over time.
- Scenario 7: Gaussian noise is added to the sensor measurement.
- Scenario 8: A constant value is added to the sensor measurement.
- Scenario 9: Sensor measurement is set equal to zero.
- Scenario 10: Sensor measurement is shifted by a small amount.

Each scenario is simulated (using WNTR [8]) for 3 months and pressure sensors are sampled every 5 min.

[1] Implementation is available on GitHub: https://github.com/andreArtelt/SAM-kNN-Regressor_OnlineLearning_WDNs.

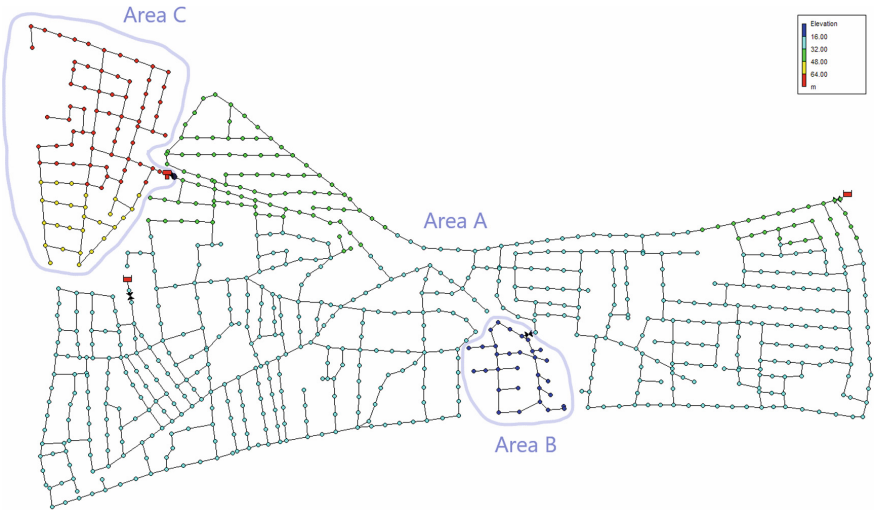

Fig. 1. L-Town network [16] – we only use "Area A" where we have 29 pressure sensors.

The data stream of sensor measurements is post processed by using a sliding window of size 4 – we average all samples dimensional wise in this time window, so that we end up with 29 dimensional samples.

5.2 Setup

We compare our proposed SAM-kNN regressor (see Sect. 4) to several other online learning regressors. In order to justify the introduced overhead of the SAM architecture, we compare its performance to vanilla regressors (kNN regression and linear regression) wrapped as online learners by using the river toolbox[2].

For each pressure sensor, we build a corresponding virtual sensor based on all other pressure sensors – i.e. we try to predict (using a regressor) the pressure based on the past pressure values of all other pressure sensors (see Sect. 3). These virtual sensors are then used for a residual based anomaly detection – i.e. an alarm (detected anomaly) is raised when the predicted pressure value deviates too much from the observed pressure measurement. For each scenario, the processed data stream is fed as batches of 200 samples to the regressors (realizing the virtual sensors).

For each regressor, we evaluate the performance of the resulting anomaly detector – since we are interested in the detection of a single anomaly (leakage and sensor fault), we report true positives (TP), false positives (FP) and false negatives (FN). Note that the true positives and false negatives are always either 0 or 1 because we only check whether an alarm was raised when the single anomaly was present or not – however, for the false positives, we count every single false alarm.

[2] https://github.com/online-ml/river.

5.3 Results

The results for leakage detection, for each regressor and each scenario, are shown in Table 1. Likewise, the results for sensor fault detection are shown in Table 2. We observe the same/similar effects for both types of anomalies (leakages and sensor faults): We observe that linear regression completely fails to detect the anomaly – this indicates that a linear model is not sufficient to model the hydraulic dynamics in the water distribution network and hence fails to detect any anomalies. There is only one exception: For scenario 6 the linear model is able to detect the sensor fault – recall that in this particular scenario the sensor fault is characterized by a slowly overflowing sensor measurements (i.e. the pressure value goes to infinity), which is expected to be easily detected because it is a very "loud" fault. The kNN model shows good performance in detecting the anomaly but has a huge number of false positives – i.e. it is too sensitive and raises lots of false alarms. Our proposed SAM-kNN shows the best performance – it is able to consistently detect the anomalies while having a small number of false positives only. The huge reduction of the false positives in comparison

Table 1. Leakages: evaluation of residual based anomaly detection in water distribution networks – note that each scenario consists of approx. 23000 samples.

Method	Metric	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
SAM-kNN	TP	1	1	1	1	1
	FP	48	20	3	20	17
	FN	0	0	0	0	0
kNN	TP	1	1	1	1	1
	FP	17057	19216	11146	19082	18751
	FN	0	0	0	0	0
Linear regression	TP	0	0	0	0	0
	FP	0	0	0	0	0
	FN	1	1	1	1	1

Table 2. Sensor faults: Evaluation of residual based anomaly detection in water distribution networks – note that each scenario consists of approx. 23000 samples.

Method	Metric	Scenario 6	Scenario 7	Scenario 8	Scenario 9	Scenario 10
SAM-kNN	TP	1	1	1	1	1
	FP	155	20	157	96	156
	FN	0	0	0	0	0
kNN	TP	1	1	1	1	1
	FP	18596	18596	18596	18596	18596
	FN	0	0	0	0	0
Linear regression	TP	1	0	0	0	0
	FP	0	0	0	0	0
	FN	0	1	1	1	1

to vanilla kNN indicates that the overhead introduced by the SAM architecture actually pays off.

6 Summary and Conclusion

Inspired by the SAM-kNN classifier, we proposed SAM-kNN regressor as an online learner for regression. In contrast to other online learners, our proposed method comes with a memory component which allows it to remember past concepts quite easily. We empirically evaluated our proposed online learner in an anomaly detection scenario for a realistic water distribution network – our proposed online learner consistently outperforms other standard online learners.

Although our proposed method shows good performance for leakage and sensor fault detection (two very common anomalies), it is unclear, how well it performs for other (more complex) types of anomalies such as cyber-physical attacks, etc. Furthermore, another challenge is high-dimensional data – in this work our data had 29 dimensions which is already somewhat high but can still be managed by our kNN based method. However, in case of really high dimensional data, kNN will encounter performance problems – e.g. some kind of integrated dimensionality reduction might be required. We leave these aspects as future work.

Acknowledgment. We acknowledge the bachelor thesis by Augustin Harter (Bielefeld University) and Yannik Sander (Bielefeld University) which served as a mental starting point for this work.

References

1. Alexander, A., Julius, T., Andrew, T., Ezera, A., Christine, A.: Contamination potentials of household water handling and storage practices in Kirundo subcounty, Kisoro district, Uganda (2019)
2. Chan, T.K., Chin, C.S., Zhong, X.: Review of current technologies and proposed intelligent methodologies for water distributed network leakage detection. IEEE Access **6**, 78846–78867 (2018)
3. Farley, M., Trow, S.: Losses in Water Distribution Networks. IWA Publishing (2003)
4. Gama, J.: A survey on learning from data streams: current and future trends. Prog. Artif. Intell. **1**(1), 45–55 (2012)
5. Gomes, H.M., Read, J., Bifet, A., Barddal, J.P., Gama, J.: Machine learning for streaming data: state of the art, challenges, and opportunities. In: ACM SIGKDD Explorations Newsletter, pp. 6–22 (2019)
6. Jakob, J., Hasenjäger, M., Hammer, B.: On the suitability of incremental learning for regression tasks in exoskeleton control. In: IEEE Symposium on Computational Intelligence in Data Mining (CIDM). IEEE, December 2021
7. Jodelet, Q., Liu, X., Murata, T.: Balanced softmax cross-entropy for incremental learning. In: Farkaš, I., Masulli, P., Otte, S., Wermter, S. (eds.) ICANN 2021. LNCS, vol. 12892, pp. 385–396. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86340-1_31

8. Klise, K.A., Murray, R., Haxton, T.: An overview of the water network tool for resilience (WNTR) (2018)
9. Lei, C.-H., Chen, Y.-H., Peng, W.-H., Chiu, W.-C.: Class-incremental learning with rectified feature-graph preservation. In: Ishikawa, H., Liu, C.-L., Pajdla, T., Shi, J. (eds.) ACCV 2020. LNCS, vol. 12627, pp. 358–374. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-69544-6_22
10. Li, H., Dong, W., Hu, B.G.: Incremental concept learning via online generative memory recall (2019). https://arxiv.org/abs/1907.02788
11. Liemberger, R., Marin, P., et al.: The challenge of reducing non-revenue water in developing countries-how the private sector can help: a look at performance-based service contracting (2006)
12. Liu, Y., Su, Y., Liu, A.A., Schiele, B., Sun, Q.: Mnemonics training: multi-class incremental learning without forgetting. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, June 2020. https://doi.org/10.1109/cvpr42600.2020.01226
13. Losing, V., Hammer, B., Wersing, H.: KNN classifier with self adjusting memory for heterogeneous concept drift. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 291–300 (2016). https://doi.org/10.1109/ICDM.2016.0040
14. Makropoulos, C., Savić, D.: Urban hydroinformatics: past, present and future. Water **11**(10), 1959 (2019)
15. Nikolopoulos, D., Moraitis, G., Bouziotas, D., Lykou, A., Karavokiros, G., Makropoulos, C.: Cyber-physical stress-testing platform for water distribution networks. J. Environ. Eng. **146**(7), 04020061 (2020)
16. Vrachimis, S.G., et al.: BattLeDIM: battle of the leakage detection and isolation methods (2020)
17. Wu, Y., Liu, S.: A review of data-driven approaches for burst detection in water distribution systems. Urban Water J. **14**(9), 972–983 (2017)
18. Wu, Y., et al.: Incremental classifier learning with generative adversarial networks (2018). https://arxiv.org/abs/1802.00853
19. Yan, J., Tian, C., Wang, Y., Huang, J.: Online incremental regression for electricity price prediction. In: Proceedings of 2012 IEEE International Conference on Service Operations and Logistics, and Informatics, pp. 31–35. IEEE (2012). https://doi.org/10.1109/SOLI.2012.6273500
20. Yang, Y., Che, J., Li, Y., Zhao, Y., Zhu, S.: An incremental electric load forecasting model based on support vector regression. Energy **113**, 796–808 (2016). https://doi.org/10.1016/j.energy.2016.07.092
21. Zhu, Q., He, Z., Ye, X.: Incremental classifier learning based on PEDCC-loss and cosine distance (2019). https://arxiv.org/abs/1906.04734