Vol.66: e23220365, 2023
https://doi.org/10.1590/1678-4324-2023220365
ISSN 1678-4324 Online Edition

Article - Engineering, Technology and Techniques

A Big Data Cleaning Method for Drinking-Water Streaming Data

Rong-Li Gai[1]
https://orcid.org/0000-0001-8826-7479

Hao Zhang[1]
https://orcid.org/0000-0003-0170-2509

Dang Ngoc Hoang Thanh[2*]
https://orcid.org/0000-0003-2025-8319

[1]Dalian University, Department of Information Engineering, Dalian, China; [2]University of Economics Ho Chi Minh City, College of Technology and Design, Ho Chi Minh City, Vietnam.

Editor-in-Chief: Alexandre Rasi Aoki
Associate Editor: Raja Soosaimarian Peter Raj

Received: 24-May-2022; Accepted: 18-Apr-2023

*Correspondence: thanhdnh@ueh.edu.vn; Tel.: (+84)0987391215 (D.N.H.T.).

HIGHLIGHTS

- Proposing a drinking-water data cleaning model with the combination of nonlinear partial differential equations and the CART decision tree.
- Preprocessing the data by removing outlier elements not following the normal distribution.
- Preprocessing the missing data by a modified classifier of AdaBoost.
- Implementing the proposed model with Big Data technology based on Hadoop architecture.
- Overall performance of the proposed method competes with other similar cutting-edge methods.

Abstract: A HA_Cart_AdaBoost model is proposed to clean the data in drinking-water-quality data. First, the data that do not follow the normal distribution are regarded as outliers and eliminated. Next, the optimal control theory of nonlinear partial differential equations (PDEs) is introduced into the cart decision tree, and the cart decision with the specified depth is used. As a weak classifier of AdaBoost, the tree uses the HA_Cart_AdaBoost model to compensate for the eliminated data, then it fits and predicts the missing values of the data stream, realizes the cleaning of drinking-water-quality data, and finally uses the big data Hadoop architecture for real-time storage and analysing streaming data. The experimental results show that compared with the most advanced data cleaning methods, after the optimal control theory of nonlinear PDEs is introduced into the cart decision tree, the stability and accuracy of the HA_Cart_AdaBoost model for water quality data cleaning are greatly improved. Taking pH as an example, the HA_Cart_AdaBoost model shows a minimum improvement of 2.25% and a maximum improvement of 53.33% in terms of RMSE, and a minimum improvement of 13.51% and a maximum improvement of 78.08% in terms of MAE.

Keywords: CART decision tree; partial differential equation; water data cleaning; AdaBoost algorithm; Hadoop architecture.

INTRODUCTION

Cleaning drinking-water streaming data is crucial for ensuring the safety and quality of drinking water. However, with the increasing volume and velocity of data in the era of big data, traditional cleaning methods are no longer sufficient. Therefore, there is a growing need for innovative big data cleaning methods for drinking-water streaming data. The motivation of the paper is to address this need by proposing a novel big data cleaning method specifically designed for drinking-water streaming data. The paper recognizes the challenges posed by the high volume, velocity, and variety of drinking-water data and seeks to develop a method that can handle these challenges effectively. The proposed method utilizes advanced data processing techniques, including data fusion, outlier detection, and data imputation, to identify and correct errors, anomalies, and missing values in the streaming data. The method also incorporates machine learning algorithms to automate the cleaning process and improve its accuracy and efficiency. The ultimate goal of the paper is to provide water management authorities and other stakeholders with a reliable and efficient tool for cleaning drinking-water streaming data. By ensuring the accuracy and quality of the data, the proposed method can help prevent waterborne diseases, reduce water treatment costs, and improve overall public health and safety. In summary, the paper's motivation is to develop a cutting-edge big data cleaning method that can address the challenges posed by the high volume, velocity, and variety of drinking-water streaming data and provide a reliable and efficient tool for water management authorities and other stakeholders to ensure the safety and quality of drinking water. However, the data are nonlinear and non-normal. Drinking water quality data is prone to abnormal phenomena such as missing and outlying data [1], which bring many difficulties to the analysis, treatment, and application of drinking water quality data. To this end, the data processing of residents' drinking water quality usually adopts data cleaning methods to identify abnormal data characteristics and propose and compensate for non-normal data to ensure the credibility of the data. However, while processing residential drinking water data, there are various characteristics of abnormal data, such as outlier data, duplicate data, and missing data, among others [2]. The LOF-SVM and LOF-BP models do not purify water quality data well due to the difficulty of identifying and compensating for anomalous data [3]. Residents are becoming increasingly concerned about the safety of their drinking water. Thus, data cleaning of drinking-water-quality data is critical.

In view of the problem that the characteristics of abnormal water quality data are difficult to identify experts and scholars have proposed a series of data cleaning methods [4]. For example, Li Panhong and coauthors [4]. proposed a double interpolation method, using the linear interpolation method to estimate parameters and then using Lagrange linear interpolation method to obtain the correction value of the interpolated data abnormal data compensation values. However, when the drinking water quality treatment process, the variables change into nonlinear time-varying characteristics, the data changes dramatically, and the linear interpolation method often cannot accurately compensate for the outliers. Liu Junqing and coauthors [6]. proposed a compensation algorithm for periodic time series, which decomposes the water quality data into trend terms, periodic terms, and residual terms with time series decomposition and replaces the outliers with the sum of the trend terms and periodic terms before the outliers are excluded. This method can fit the changing trend of water quality data under many circumstances and interpolate within the changing trend to compensate for outliers. However, the method relies too much on the change of abnormal data over time, and the compensation effect of data with complex changes and continuous anomalies is poor [7]. Zeng Chen [8] and coauthors proposed the ADAPTIVE-EWT-MFE method, the ADAPTIVE-EWT-MFE method uses EWT and MFE to clean water quality data. It decomposes the data into IMFs and introduces an adaptive threshold parameter based on MFE to filter high-frequency noise without distorting noise-free data. This method improves noise-cleaning performance and has potential for detecting patterns in high-frequency data at low SNRs. One potential disadvantage of the ADAPTIVE-EWT-MFE method is that it may require a certain level of expertise in EWT and MFE to correctly apply and interpret the results. This could limit its use for researchers or analysts who are unfamiliar with these techniques. Another possible drawback is that the method's performance may be affected by the quality and characteristics of the input data, such as the presence of outliers or non-stationary behavior. Therefore, careful consideration should be given to the data used and the method's application to achieve optimal results. Jianzhuo Yan and coauthors [9] proposes a time series framework to clean complex water quality data, using Pauta criterion and NLDIW-PSO SVR. Pearson's correlation coefficient was used to reduce dimensionality, and the model's performance was evaluated with R2 and Pearson's correlation coefficient. The framework is effective and can handle general time series data. One potential disadvantage of this time series framework is that it requires careful consideration of the input data and the application of the various techniques used, such as Pauta criterion and NLDIW-PSO SVR. Additionally, the effectiveness of the framework may be limited by the quality and characteristics of the input data, such as the presence of outliers or non-stationary behavior. Therefore,

careful evaluation and validation of the results are necessary to ensure the reliability and accuracy of the cleaning framework. Meng Q and coauthors [10] proposed a new method to clean incomplete water quality data based on improved BIRCH clustering. The method accurately detects and corrects missing and outlier values, normalises missing data and applies it to BIRCH data. A potential limitation of the method is that it may not be as effective in situations where the data has high variability or non-variability - stationary behaviour. In addition, the accuracy of the clustering and neural network models used to detect and fill in missing values may be affected by the quality and characteristics of the input data, such as the presence of outliers or noisy data. Due to a large amount of data on water quality and many other parameters, abnormal data may be accompanied by multiple abnormal features. Currently, the cleaning of water quality data is mostly limited to cleaning a single type of data; cleaning various abnormal characteristic data is lacking. When traditional cluster analysis, rough set theory, model matching, genetic neural network, and expert system data cleaning models are used, meeting the stringent quality requirements of drinking water is difficult [11].

This paper proposes an HA_CART_Adaboost algorithm to clean and improve the quality of dirty data in the water environment dataset due to the failure of water environment monitoring instruments or sensors. Firstly, data that do not conform to a normal or approximately normal distribution are considered as outliers and are directly rejected; secondly, the optimal control theory of non-linear partial differential equations is introduced into the CART decision tree, and the CART decision tree of specified depth is used as an AdaBoost weak classifier to process the data set after the outliers are removed and to compensate for the data, while effectively fitting and predicting missing values in the dataset. The experimental results show that the combined cleaning effect of HA_CART_AdaBoost is more accurate than that of SVM and BP, and the data cleaning effect is better suited to the data cleaning problem of water environment type.

MATERIAL AND METHODS

Water quality data preprocessing

Characteristics of abnormal data on drinking water quality

In water quality data collection, data cleaning is mainly based on exploratory analysis, and two types of abnormal data are processed, namely missing values and outliers [12]; their characteristics are as follows:

(1) Missing values

Missing values refer to the data that have not been collected within a certain period or the presence of null values [**Erro! Fonte de referência não encontrada.**]. During drinking-water-quality data collection, machine failures and packet loss are prone to occur, resulting in continuous or intermittent loss of data. In addition, problems may be encountered in the server that collects data, resulting in the loss of data files.

(2) Outliers

Outliers refer to skewed data that deviate significantly from the rest of the data [**Erro! Fonte de referência não encontrada.**]. Drinking water has stringent quality requirements. In addition, the water quality varies with time, which leads to large deviations in the measurement values obtained using water quality collection equipment. Because the water quality collection equipment collects streaming data, the data is continuous and uninterrupted; thus, the data skew phenomenon might occur during the data collection process, leading to many outliers [15].

Handling Missing Values and Outliers

There are two main types of anomalous data types in water quality data collection: Missing values and Outliers. Therefore, the water quality data is pre-treated first. For missing values in the data, if the deletion rate of the variable is high (greater than 80%), the coverage rate is low, and the importance is low, the variable can be deleted directly. The removal rate refers to the percentage of features or variables removed from a dataset during feature selection or feature engineering. Feature selection is the process of selecting a set of relevant features to improve the performance of a model, while feature engineering involves creating new features from existing ones. The removal rate is often used as a measure of the effectiveness of a feature selection or engineering method; the higher the removal rate, the more effective the method. Coverage refers to the proportion of text analysed or covered by a particular model or algorithm. Coverage is an important metric for assessing model performance, particularly for tasks such as text classification and sentiment analysis. Importance refers to the extent to which a particular feature or variable has an impact on the model output or prediction, with features with high importance having a strong influence on the model output, while features with lower importance have little or no impact. For data outliers, we use the 3σ criterion [16] to determine. If the sample is normally distributed or approximately normally distributed, more than 99% of the

data are considered to be within 3 standard deviations of the upper and lower means. Specifically, the probability of this value distribution in (μ-3σ, μ+3σ) is 99.73%, and the data with the maximum or minimum value exceeding this range are outliers and are excluded. In the training of the tree model, the tree model has high robustness to extreme values, and there is no information loss, which will not affect the training effect of the model.

Algorithm

Introducing Partial Differential Equations into CART Decision Tree

The CART algorithm is an example of a binary tree problem. CART stands for Classification and Regression Trees, which is a machine learning algorithm used for both classification and regression analysis. It creates a decision tree that recursively splits the data into subsets based on the values of one of the features, ultimately producing a set of rules that can be used to predict the target variable. In classification problems, the target variable is categorical, while in regression problems, it is continuous. The CART algorithm is widely used in data mining, bioinformatics, and many other fields. Unlike the C4.5 algorithm [17], which may have multiple choices, ID3 [18] employs the information gain selection feature, with the gain being the first option. The information gain ratio is used in C4.5 to select features to alleviate the problem of excessive information gain caused by too many eigenvalues. The CART algorithm consists primarily of a regression tree and a classification tree. The commonly used criteria for the two differ slightly: regression tree is a fitting problem and is more concerned with the quality of the fitting effect, whereas classification tree is a classification problem and is akin to the probability estimation of discrete variables measured using a Gini coefficient [19] similar to entropy. The CART classification tree algorithm uses the Gini coefficient to replace the information gain ratio. The Gini coefficient represents the impurity of the model; the smaller the Gini coefficient, the lower the impurity and the better the feature [20].

A model of (m-1) trees is used, and the residuals for each sample r_{im} are computed.

$$r_{im}=-\left[\frac{\partial L(y_i,F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \tag{1}$$

$F(x_i)$ is replaced with $\hat{y}_i$, which is the partial derivative with respect to x_i.

$$\frac{\partial L(y_i,F(x_i))}{\partial \hat{y}_i}=\frac{\partial \frac{1}{2}(y_i-\hat{y}_i)^2}{\partial \hat{y}_i}=-(y_i-\hat{y}_i) \tag{2}$$

$F(x)=F_{m-1}(x)$, which means for using the model to calculate $\hat{y}_i$, that is, to use (m-1) generated trees to predict each sample, $r_{im}=y_i-\hat{y}_i$.

For each leaf node,

$$\gamma_{jm}=\arg\min\sum_{x\in R_{ij}} L(y_i,F_{m-1}(x_i)+\gamma) \tag{3}$$

In the newly constructed tree m, the output γ_{jm} of each node j is determined to minimize the loss of this node.

Solving Eqs. (1)-(3), we obtain

$$L(y_1,F_{m-1}(x_1)+\gamma)=-y_1(F_{m-1}(x_1)+\gamma)+\log(1+e^{F_{m-1}(x1)+\gamma}) \tag{4}$$

The trick here is to first use the second-order Taylor formula to approximate the formula and then obtain the derivative, that is, the second-order Taylor expansion with γ as a variable and the rest of the terms as constants, as follows:

$$L(y_1,F_{m-1}(x_1)+\gamma)\approx L\left(y_1,F_{m-1}(x_1)\right)+L'\left(y_1,F_{m-1}(x_1)\right)\gamma+\frac{1}{2}L''\left(y_1,F_{m-1}(x_1)\right)\gamma^2 \tag{5}$$

Taking the derivative, we obtain

$$\frac{dL}{d\gamma}=L'\left(y_1,F_{m-1}(x_1)\right)+L''(y_1,F_{m-1}(x_1))\gamma \tag{6}$$

When the loss is the lowest, Eq. (6) is equal to 0; thus, we can determine γ as follows:

$$\gamma_{11}=\frac{-L'\left(y_1,F_{m-1}(x_1)\right)}{L''(y_1,F_{m-1}(x_1))} \tag{7}$$

The numerator in Eq. (7) is the residual r. The denominator is the second-order differential of the loss function and is calculated as follows:

$$L''\left(y_1,F(x)\right)=\frac{dL'}{d\log(odds)}$$

$$=\frac{d}{d\log(odds)}\left[-y_i+\frac{e^{\log(odds)}}{1+e^{\log(odds)}}\right]$$

$$=\frac{d}{d\log(odds)}[e^{\log(odds)}\left(1+e^{\log(odds)}\right)^{-1}] \tag{8}$$

$$=e^{\log(odds)}\left(1+e^{\log(odds)}\right)^{-1}-e^{2\log(odds)}\left(1+e^{\log(odds)}\right)^{-2}$$

$$=\frac{e^{\log(odds)}}{\left(1+e^{\log(odds)}\right)^2}$$

where $\frac{e^{\log(odds)}}{1+e^{\log(odds)}}$ is p and $\frac{1}{1+e^{\log(odds)}}$ is 1 - p; thus,$L''=p(1-p)$. Then, the output of this node is $\gamma_{11}=\frac{r_{11}}{p_{10}(1-p_{10})}$.

Next, the output of the right node is calculated by using the second-order Taylor formula:

$$L\left(y_2,F_{m-1}(x_2)+\gamma\right)+L\left(y_3,F_{m-1}(x_3)+\gamma\right) \tag{9}$$

$$\approx L\left(y_2,F_{m-1}(x_2)\right)+L'\left(y_2,F_{m-1}(x_2)\right)\gamma+\frac{1}{2}L''\left(y_2,F_{m-1}(x_2)\right)\gamma^2+L\left(y_3,F_{m-1}(x_3)\right)+L'\left(y_3,F_{m-1}(x_3)\right)\gamma+\frac{1}{2}L''\left(y_3,F_{m-1}(x_3)\right)\gamma^2$$

Taking the derivative of Eq. (9) and making the result 0, γ can be calculated as

$$\gamma_{21}=\frac{-L'\left(y_2,F_{m-1}(x_2)\right)-L'\left(y_3,F_{m-1}(x_3)\right)}{L''\left(y_2,F_{m-1}(x_2)\right)+L''\left(y_3,F_{m-1}(x_3)\right)} \tag{10}$$

The output value for any leaf node is:

$$\gamma_{jm}=\frac{\sum_{i=1}^{R_{ij}} r_{im}}{\sum_{i=1}^{R_{ij}} P_{i,m-1}(1-P_{i,m-1})} \tag{11}$$

Next, the model $F_m(x)$ is:

$$F_m(x)=F_{m-1}(x)+v\sum_{j=1}^{J_m}\gamma_m \tag{12}$$

where v is the learning rate.

Finally, after looping M times or when the total residual is lower than the preset threshold, the execution of the cart decision tree algorithm with the partial differential equations (PDEs) is completed.

AdaBoost algorithm

The AdaBoost algorithm is an effective and practical boosting algorithm and improves the boosting algorithm proposed by Freund and Schapire in 1995. The smallest weight coefficient weak classifier is selected from the trained weak classifiers by adjusting the sample and classifier weights, and then combined to form a strong classifier [21]. The training set is used to train the weak classifier, and the next weak classifier is trained on a different weight set from the sample. The weight is determined by the difficulty of classifying each sample, which is estimated by the output of the classifier in the previous step.

During training using the sample training set, the AdaBoost algorithm selects the key classification feature sets, gradually trains the weak classifiers, selects the best weak classifier with an appropriate threshold, and finally selects the best weak classifier for each iteration of training. We divide the training set into a training set and a validation set. We train the weak classifier on the training set and evaluate its performance under different thresholds on the validation set, and pick the best performing threshold on the validation set. Thus, good and weak classifiers are used to obtain strong classifiers [22-23].

The AdaBoost algorithm has a high detection rate and is not prone to over-adaptation [24]. However, a large training sample set is required to achieve high detection accuracy. Each sample in the dataset has many features, and in each iteration, a weak classifier is trained to correspond to each sample. Therefore, the computational complexity of training the optimal weak classifier from the huge number of features increases.

Hadoop architecture

Hadoop mainly includes HDFS and MapReduce [25]. Hadoop is a distributed storage with highly fault-tolerant, suitable for applications with very large data; MapReduce is a program writing entity model for parallel operations on large-scale data (>1 TB).

(1) HDFS

HDFS is a distributed file system that is highly fault tolerant and can be widely deployed on cheap PCs. It accesses the application's data in the streaming access mode, which greatly increases the data throughput of the overall system, making it ideal for applications with very large datasets.

HDFS adopts a master-slave architecture [26]. A classic HDFS cluster contains two parts, one parameter node and multiple data nodes. The parameter nodes can both store and manage the metadata of files in the HDFS file system. Usually, only one machine in the cluster runs the NameNode instance. The DataNode saves the data in the file, and each machine in the cluster runs a DataNode instance. The DataNode communicates regularly with the NameNode through the heartbeat mechanism. NameNode and DataNode are key components of the Hadoop Distributed File System (HDFS), which is used for storing and processing large data sets across a distributed computing environment. The NameNode is the central node in the HDFS cluster, responsible for managing metadata about the files and directories stored in the file system. This includes information such as file names, permissions, and block locations. The NameNode maintains this information in memory and also persists it to disk. The DataNode, on the other hand, is responsible for storing the actual data blocks that make up the files in the HDFS cluster. Each DataNode stores one or more blocks of data, and communicates regularly with the NameNode to report on the health and availability of the blocks it manages. Together, the NameNode and DataNodes work together to provide a fault-tolerant, scalable, and highly available distributed file system for storing and processing large data sets.

(2) MapReduce

MapReduce is a programming model for parallel operations on large-scale datasets [27]. Map (mapping) and Reduce (simplification) adopt the idea of "divide and conquer." First, the cluster allocates tasks to multiple nodes, performs parallel computing on the tasks, and finally integrates the results to obtain the final computing results. The MapReduce framework performs multi-node computing, involving task scheduling, load balancing, and fault-tolerant processing.

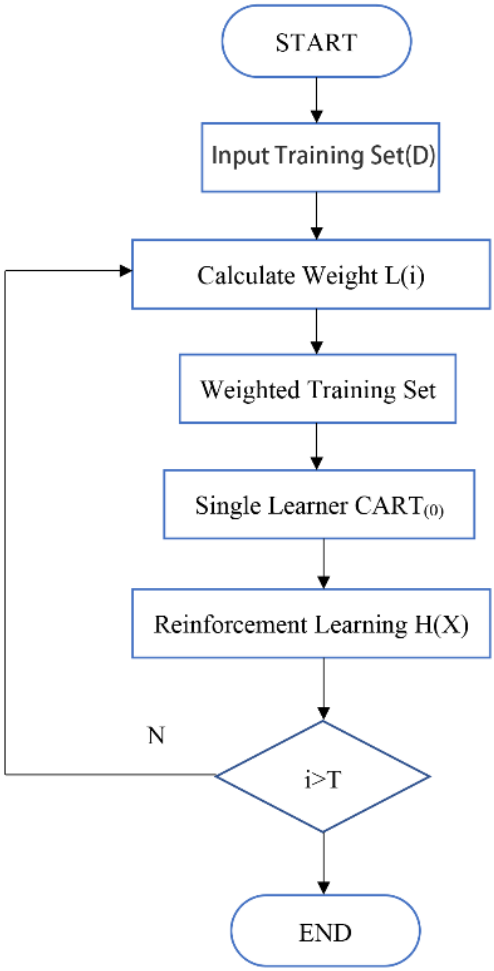


Figure 1. Flowchart of CART decision tree and AdaBoost fusion algorithm

HA_Cart_AdaBoost model

The CART classification tree algorithm is added because it is a pruning algorithm. HA_Cart_AdaBoost is composed of multiple cart decision trees as weak classifiers. In each iteration, a weak classifier is used to train the dataset; then, the weight of the dataset that is wrongly classified by the trainer is increased, and the weight of the correct classification is reduced. A weak learner is used to train this dataset, and it iterates continuously and finally merges to produce a strong learner, inputs the data into the learner, stores the cleaned data in HDFS, and uses MapReduce for real-time calculation. To better improve the fitting accuracy and solve the overfitting problem, in this paper, the cart decision tree is used as the strong classifier of the AdaBoost algorithm. The flowchart of the fusion algorithm is shown in Figure 1.

The algorithm takes the training set D, the threshold of the Gini coefficient, and the threshold of the number of samples as the input. The output is the decision tree T. The algorithm starts from the root node and recursively builds the CART classification tree by using the training set. The steps involved are described as follows:

Step 1: For the dataset D, in the current node, if the number of samples is less than the threshold or if there is no feature, return to the decision subtree and stop the recursion at the current node.

Step 2: Calculate the Gini coefficient of the sample set D. If the Gini coefficient is less than the threshold, return to the subtree of the decision tree and stop the recursion at the current node.

Step 3: Calculate the Gini coefficient of each feature of the current node in the dataset D.

Step 4: From the calculated Gini coefficients of each feature value pair in dataset D, select feature A with the smallest Gini coefficient and the corresponding feature value a. According to this optimal characteristic and optimal characteristic value, divide the dataset into two parts, D1 and D2, and establish the left and right nodes of the current node. The dataset D of the node is D1, and the dataset D of the right node is D2.

Step 5: Recursively call steps 1-4 on the left and right child nodes to generate a decision tree.

When making predictions on the generated decision tree, if sample A in the test set falls on a leaf node and there are multiple training samples in the node, then the category prediction for A uses the category with the highest probability in the leaf node.

The construction process of the HA_Cart_AdaBoost algorithm is described as follows:

Hypothetical training set can be presented:

$$T=\{(X_1,Y_1),(X_2,Y_2)\cdots(X_n,Y_n)\} \tag{13}$$

Initialize the weight distribution of training data as:

$$D1=(W|1,W|2\cdots\cdots W|n)\,,\text{where } W|i=\frac{1}{n},i=1,2,3\cdots\cdots n \tag{14}$$

Use the training set Dm with weight distribution for learning to obtain the basic classifier:

$$G_m(x):x\rightarrow\{-1,+1\} \tag{15}$$

Calculate the classification error of Gm(x) on the training set:

$$\varepsilon_m=P\left(G_m(X_i)\neq y_i\right)=\sum w_{mi}I\left(G_m(X_i)\neq y_i\right) \tag{16}$$

Calculate the weight coefficient ∂_m of the Gm(x) model:

$$\partial_m=\frac{1}{2}*\log_2\left(\frac{1-\xi m}{\xi_m}\right) \tag{17}$$

Determine the weight distribution of the weight training set:

$$D_{m+1}=\left(W_{m+1,1},W_{m+1,2},W_{m+1,i}\cdots,W_{m+1,n}\right),W_{m+1,\,i}=\frac{W_{m,i}}{Z_m}\,e^{-\alpha_m y_i G_m(X_i)} \tag{18}$$

where Z_m is the normalization factor:

$$Z_m=\sum_{i=1}^{n}W_{m,i}e^{-\alpha_m y_i G_m(x_i)} \tag{19}$$

Build a linear combination of classifiers:

$$f(x)=\sum_{m=1}^{M}\alpha_m G_m(x) \tag{20}$$

Obtain the final classifier:

$$G(x)=\text{sign}\left(f(x)\right)=\text{sign}\left[\sum_{m=1}^{M}\partial_m G_m(x)\right] \tag{21}$$

The HA_Cart_AdaBoost model is used to learn a strong classifier of water quality data. The process for training of the water quality data is as follows:

The training samples used for weight training are listed in Table 1.

Table 1. Training samples

Serial Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
X	0	1	2	3	4	5	6	7	8
Y	1	1	1	-1	-1	-1	1	1	1

For m = 1, the data weight distribution is presented in Table 2.

Table 2. D1 dataset weight distribution

Serial Number	0	1	2	3	4	5	6	7	8
X	1	1	1	-1	-1	-1	1	1	1
Y	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

On the training data with weight distribution D1, the error rate is the lowest when the threshold v is 2.5; thus, the basic classifier is

$$G_1(x)=\begin{cases}1, x<2.5\\ -1, x>2.5\end{cases} \tag{22}$$

The error rate of $G_1(x)$ on the training dataset is

$$\varepsilon_1=P\left(G_1(X_i)\neq y_i\right)=0.3 \tag{23}$$

Calculate the coefficients of G_1:

$$\partial_1=\frac{1}{2}*\log_2\left(\frac{1-\xi_1}{\xi_1}\right) \tag{24}$$

Update the weight distribution of the dataset:

$$\begin{aligned}D_2&=\left(W_{2,1},W_{2,2},\cdots,W_{2,n}\right)\\ &=(0.0582,0.0582,0.0582,0.0582,0.0582,0.0582,0.1976,0.1976,0.1976)\end{aligned} \tag{25}$$

Therefore, the first-generation classifier is obtained as follows:

$$f_1(x)=0.6112G_1(x) \tag{26}$$

The classifier sign($f_1(x)$)has three misclassified points on the training dataset; thus, training is continued. For m = 2, the data weight distribution is listed in Table 3:

Table 3. D2 dataset weight distribution

X	0	1	2	3	4	5	6	7	8
Y	1	1	1	-1	-1	-1	1	1	1
W1	0.0582	0.0582	0.0582	0.0582	0.0582	0.0582	0.1976	0.1976	0.1976

On the training data whose weight distribution is D2, the error rate is the lowest when the threshold v is 8.5; thus, the basic classifier is

$$G_2(x)=\begin{cases}1, x<8.5\\ -1, x>8.5\end{cases} \tag{27}$$

The error rate of $G_2(x)$ on the training dataset is

$$\varepsilon_2=P\left(G_2(X_i)\neq y_i\right)=0.0582*3=0.1746 \tag{28}$$

Calculate the coefficients of G_2:

$$\partial_2=\frac{1}{2}*\log_2\left(\frac{1-\xi_2}{\xi_2}\right)=1.1205 \tag{29}$$

Update the weight distribution of the dataset:

$$D_3=\left(W_{2,1},W_{2,2},\cdots,W_{2,n}\right)=(0.0236,0.0236,0.0236,0.02218,0.2218,0.2218,0.0801,0.0801,0.0801) \tag{30}$$

Therefore, the second-generation classifier is obtained as follows:

$$f_2(x)=0.6112G_1(x)+1.1205G_2(x) \tag{31}$$

The classifier sign($f_2(x)$)has three misclassified points on the training dataset; thus, training is continued, until there are no misclassification points.

Repeat the above steps, the third-generation classifier is obtained as follows:

$$f_3(x)=0.6112G_1(x)+1.1205G_2(x)+1.631G_3(x) \quad (32)$$

The classifier sign($f_3(x)$) has zero misclassified points on the training dataset; thus, the loop is ended, and the final strong classifier is obtained.

The HA_Cart_AdaBoost model trains samples of the data and assigns them weights. These weights form a weight vector D, and the dimension is equal to the number of samples in the dataset. Initially, all these weights are equal. First, a weak classifier is trained on the training dataset, and the error rate of that classifier is calculated. The weak classifier is then trained again on the same dataset, but this time the weight of each sample in the dataset is adjusted based on the classifier's error rate. The weight of the correctly classified sample falls, while the weight of the incorrectly classified sample rises, but the sum of these weights remains constant at 1.

The final classifier assigns different coefficients of determination alpha based on the classification error rates of these trained weak classifiers, and classifiers with low error rates get high coefficients of determination, thus playing a key role in data prediction.

The calculation of alpha is based on the error rate:

$$alpha=0.5*\ln(1-\varepsilon/\max(\varepsilon,1e\text{-}16)) \quad (33)$$

where ε is the number of correctly classified samples/total number of samples, and max(ε,1e-16) is for preventing the error rate from causing the denominator to be 0.

Store the cleaned data in Hadoop's HDFS. The specific process is as follows:

First, the Client calls the create() method of the Distributed File System object to create a file output stream; Distributed File System creates an RPC call to the namenode to create a new file in the filesystem's namespace; the Namenode performs various checks to ensure that the file does not exist , and the client has permission to create the file. If these checks pass, the namenode will record a record for creating a new file, otherwise, the file creation fails, throw an IOException to the Client, Distributed File System returns a FSDataOutputStream formation to the Client, and the Client can start writing data; DFSOutputStream divides it into packets and write to the internal queue. The DataStreamer handles the data queue, and its responsibility is to ask the namenode to allocate a new block suitable for storing the data backup based on the list of datanodes. This set of datanodes forms a pipeline - let's say the number of replicas is 3, there are 3 nodes in the pipeline, the DataStreamer streams the packet to the first datanode in the pipeline, the datanode stores the packet and sends it to the pipeline Similarly, the second datanode stores the packet and sends it to the third one; DFSOutputStream also maintains an internal packet queue to wait for the datanode's ack queue. When all datanode confirmation information in the pipeline is received, the data packet will be deleted from the confirmation queue; after the client finishes writing the data, it will call the close() method on the data stream; write all the remaining data packets into the datanode pipeline, and before exercising the namenode and sending the file write complete signal.

After extracting the water quality data from Hadoop's HDFS, use mapreduce for data analysis.When an MR (MapReduce) program is launched, the MRAppMaster is the first component to start. The MRAppMaster calculates the number of map tasks required for the job based on the job description and requests the cluster to start the corresponding number of map task processes. Each map task performs data processing on a given data slice range. The map task uses the input format specified by the user to read the data using a RecordReader, creates input key-value pairs, and applies the map() method defined by the user for logical operations. The output key-value pairs generated by the map() method are collected into a cache and sorted based on the partition key. The sorted key-value pairs are then written to disk files continuously. Once all map tasks have completed, the MRAppMaster starts the reduce task processes based on the customer-specified parameters and informs them of the data range or partition to be processed. The reduce task process retrieves the output results of several map tasks from the machines where they are running, based on the location of the data notified by the MRAppMaster. The reduce task locally re-merges and sorts the retrieved output data and then groups the key-value pairs based on the same key. The reduce() method defined by the user is called to perform logical operations on each group and collect the result key-value pairs. Finally, the output format specified by the user is used to store the result data externally.

Experiment and Analysis

To ensure the accuracy of the experiment, real data collected from a reservoir in DaLian, LiaoNing in December 2021 were used as the experimental data. Smart terminals containing sensors were installed to collect temperature, residual chlorine, dissolved oxygen and pH data.

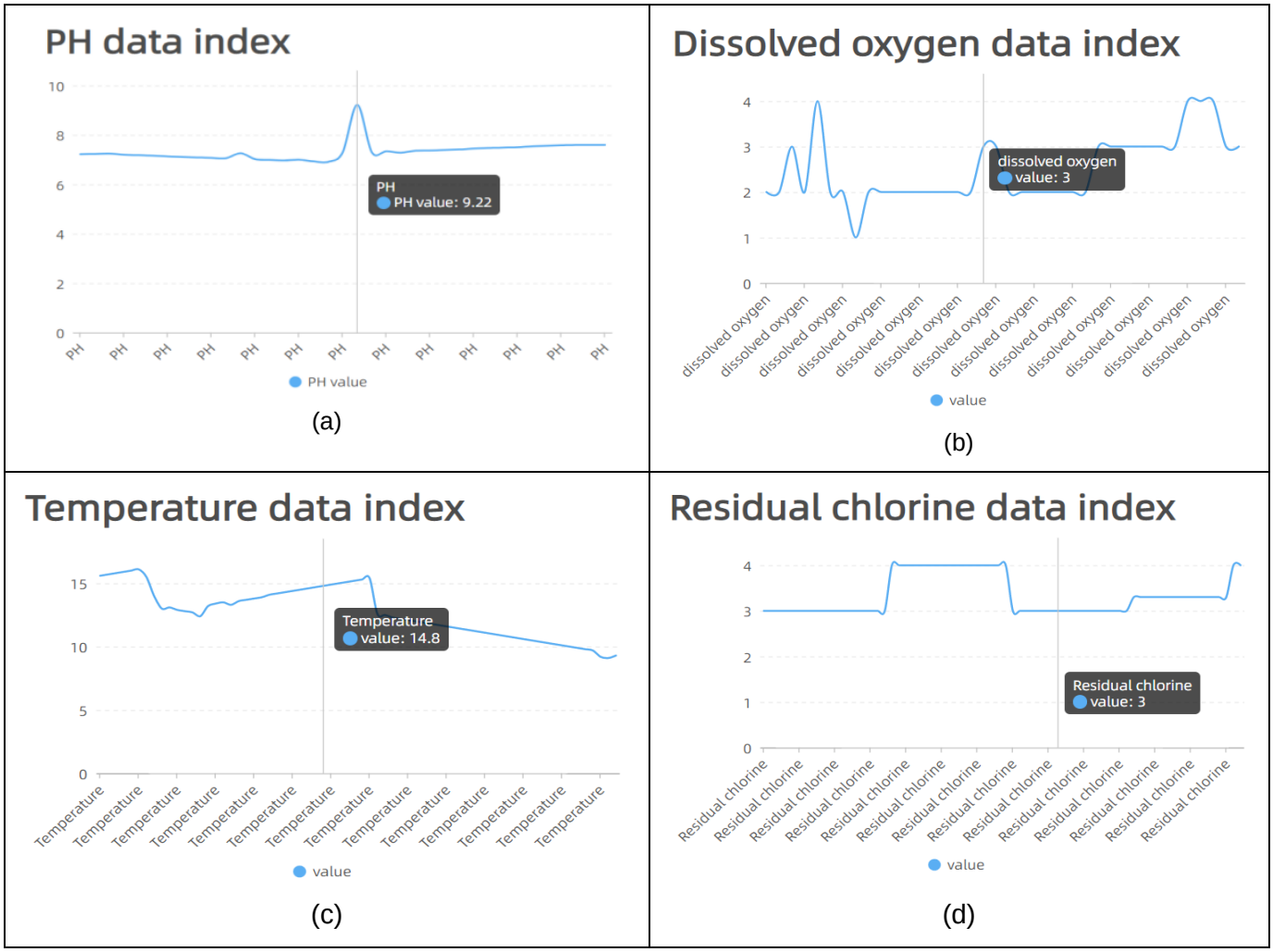


Figure 2. Analysis of data changes of key variables: (a) pH index; (b) Dissolved oxygen index; (c) Temperature index; (d) Residual chlorine index

First, the data change analysis of the four key variables of the cleaning process was performed; the results are shown in Figure 2.

From Figure 2, it can be seen that the trends of dissolved oxygen and residual chlorine are unstable, and when the pH variable value is 9.22, it is considered an outlier. Two variables, namely, pH and dissolved oxygen, were selected in the data cleaning experiment. In total, 500 sets of HA_Cart_AdaBoost training data, 100 compensation training data, and 100 sets of test data were selected for each variable. The data cleaning experiments are described in this section.

Outlier data removal

In data analysis and modelling, outliers can cause significant deviations and distort the results. As such, removing outlier data is an important aspect of data cleaning. The HA_Cart_AdaBoost model employs the 3σ criterion to detect outliers. While removing outliers can improve the accuracy of models, it is important to be cautious when doing so. Sometimes, outliers may contain valuable information or be indicative of rare but significant events. Thus, it is essential to carefully consider the reasons for removing outliers and the potential impact on the analysis and conclusions drawn from the data. Furthermore, the 3σ criterion assumes that the data is normally distributed, which may not always be the case. In situations where the distribution is skewed or has heavy tails, the criterion may incorrectly classify data points as outliers or fail to identify true outliers. To address these issues, alternative methods for detecting outliers, such as robust statistics or clustering

algorithms, can be used. Additionally, exploratory data analysis and visualization techniques can help identify potential outliers and provide insight into the underlying distribution of the data. The results are shown in Figure 3.

Figure 3. Outlier screening and elimination

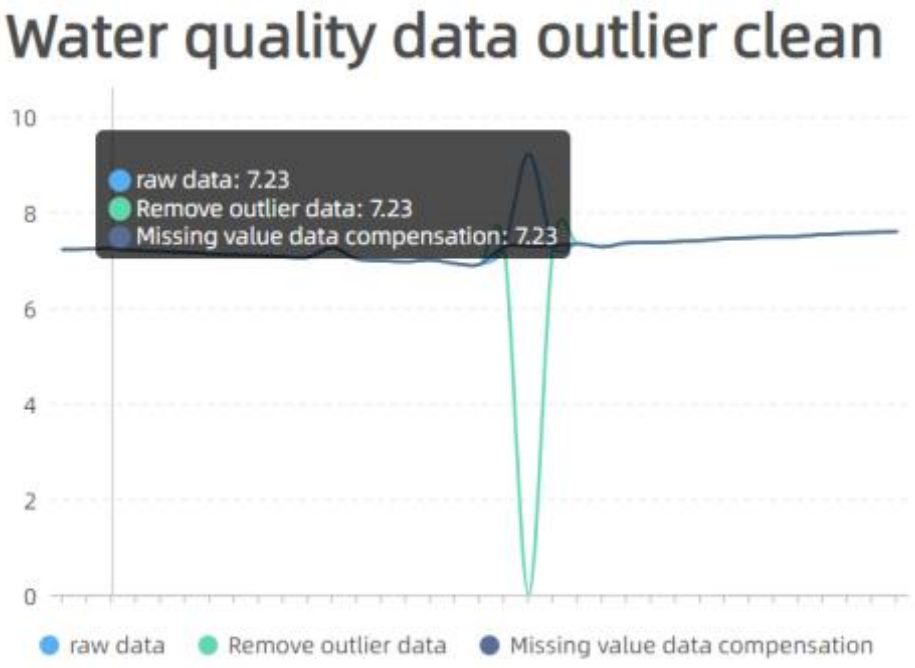


Figure 4. Outlier screening and elimination

By observing Figures 3 and 4, it can be noticed that there are outliers in the datasets. In Figure 3, a data point with a value of 9.22 is considered an outlier, whereas in Figure 4, a data point with a value of 7.23 is also regarded as an outlier. We use the 3σ criterion to detect and remove the outliers.

Compensation for missing value data

The dataset with the outliers removed is inputted into the HA_Cart_AdaBoost model, and the logarithmic missing data values are compensated. The results are shown in Figure 5.

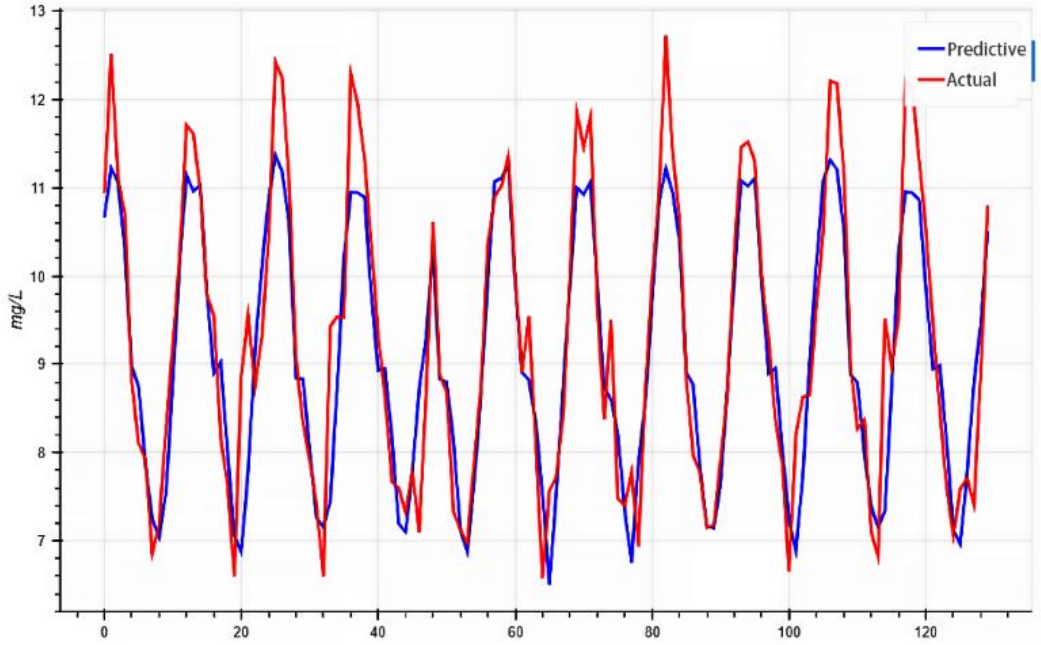


Figure 5. Compensation for missing value data

The results show that the prediction results of the model compensation value are almost the same as the actual data trend. The HA_Cart_AdaBoost model has high stability and accuracy, the compensation effect for this missing dataset is better, and the data adaptability is stronger.

Model testing and classification error rates

Based on the experimental HA_Cart_AdaBoost classifier training and testing, the error rates of the HA_Cart_AdaBoost model testing and classification after introducing the optimal control theory of nonlinear PDEs for the cart decision tree were compared for different numbers of classifiers (Table 4).

Table 4. Model testing and classification error rates

(a)Test and classification error rate of the HA_Cart_AdaBoost model without PDEs

Number of classifiers	Training error rate (%)	Test error rate (%)
1	0.28	0.27
10	0.23	0.24
50	0.19	0.21
100	0.19	0.22
500	0.16	0.25
1000	0.14	0.31
10000	0.11	0.33

(b)Test and classification error rate of the HA_Cart_AdaBoost model with PDEs

Number of classifiers	Training error rate (%)	Test error rate (%)
1	0.24	0.24
10	0.18	0.21
50	0.16	0.26
100	0.16	0.14
500	0.11	0.09
1000	0.11	0.09
10000	0.09	0.09

The experimental results revealed that when the optimal control theory of nonlinear PDEs was not introduced, the training error rate of the HA_Cart_AdaBoost classifier decreased with the increase in the number of classifiers. While the test error rate is the first process of reducing a parameter to a minimum value, and then the value is gradually increased, which is called overfitting.

After introducing the optimal control theory of nonlinear PDEs, the training error rate of the HA_Cart_AdaBoost classifier decreased with the increase in the number of classifiers, whereas the test error rate increased with the increase in the number of iterations and then gradually stabilized.

Appropriate measures should be adopted to avoid the overfitting problem. When introducing the optimal control theory of nonlinear PDEs into the model, as classifiers are drilled, a validation set is applied and the classification error rate of the validation set is continuously checked. When the error rate of the training set decreases and that of the verification set increases, the training is stopped.

Some studies have shown that for datasets with good performance, the test error rate of HA_Cart_AdaBoost gradually stabilizes with the increase in the number of iterations, without the overfitting problem, as shown in Table 4(a). However, the dataset used in this study cannot be called a “good performance” dataset. Therefore, high-quality datasets should be used in addition to introducing the optimal control theory of nonlinear PDEs.

Comparison with other models

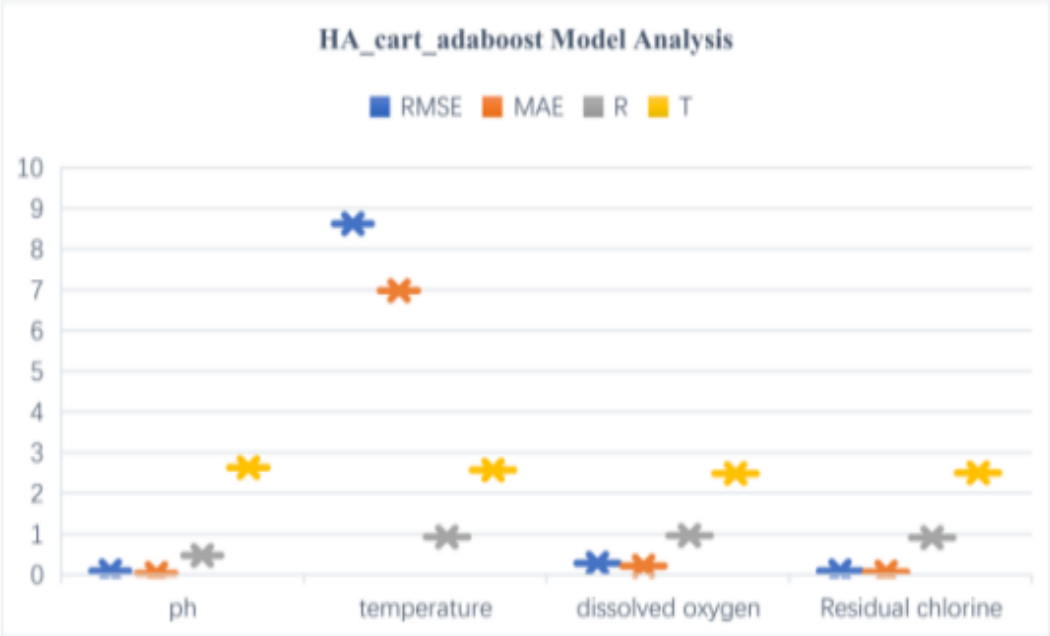
To evaluate the performance of various models on the prediction task and demonstrate the superiority of the HA_Cart_AdaBoost model in data cleaning over other models, the experimental data were compensated in the model, and the data cleaning results were analysed by using root mean square error (RMSE), mean absolute error (MAE) and determination coefficient (R) [**Erro! Fonte de referência não encontrada.**]. Precision represents the degree of fitting of two curves. LOF-SVM is a method that combines the Local Outlier Factor (LOF) algorithm with the Support Vector Machine (SVM) algorithm for anomaly detection and classification analysis. LOF-BP combines the Local Outlier Factor algorithm with the Back Propagation (BP) neural network algorithm for anomaly detection and classification analysis. These algorithms were used in the study to compare their performance in anomaly detection tasks. Fifteen groups of intermittent outlier data and 10 groups of continuous outlier data were randomly inserted into the 500 groups of test datasets, and the abnormal data were identified and eliminated using the 3σ criterion. To observe the distribution of missing data, the data were normalized and processed, and outliers were eliminated and replaced with 0 to form a missing dataset with 5% missing values.

The experimental results are presented in Table 5.

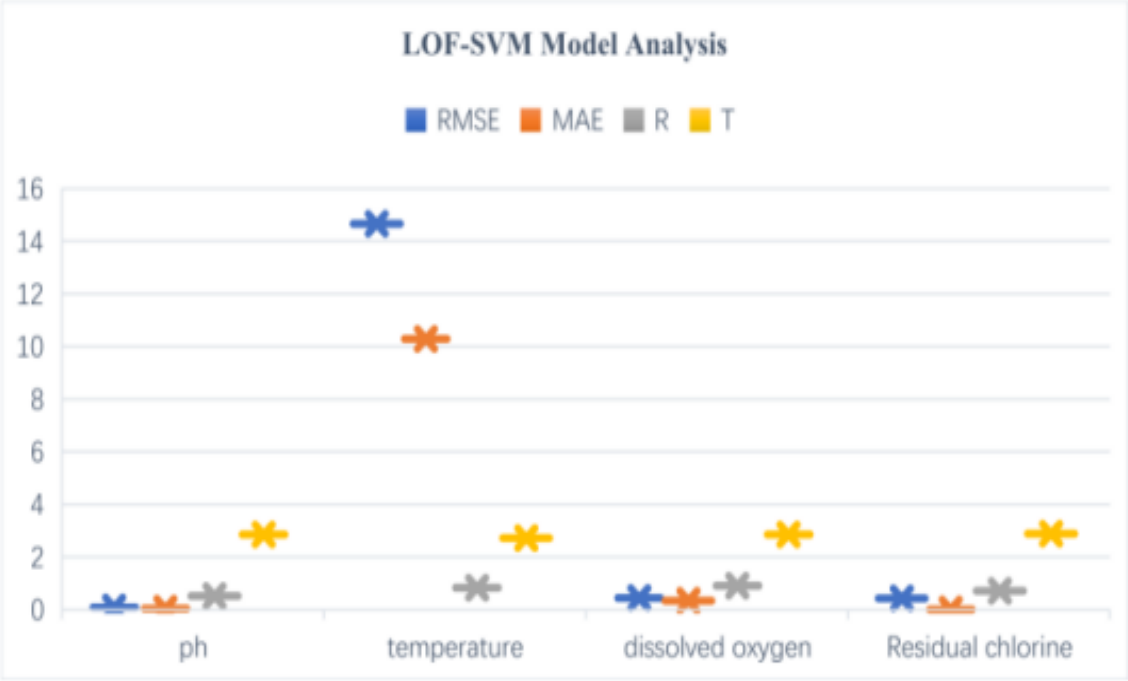
Table 5. Comparison of model experimental results

Variable	HA_Cart_AdaBoost				LOF-SVM				LOF-BP			
	RMSE	MAE	R	Ts	RMSE	MAE	R	Ts	RMSE	MAE	R	Ts
pH	0.091	0.032	0.471	2.635	0.089	0.037	0.522	2.853	0.195	0.146	-0.852	2.814
Temperature	8.624	6.979	0.932	2.574	14.672	10.29	0.834	2.724	47.02	38.32	-0.635	2.862
Dissolved	0.286	0.217	0.964	2.488	0.45	0.336	0.91	2.86	1.899	1.808	-0.61	2.747
Residual	0.091	0.068	0.914	2.505	0.429	0.017	0.709	2.888	0.119	0.036	-1.055	2.803

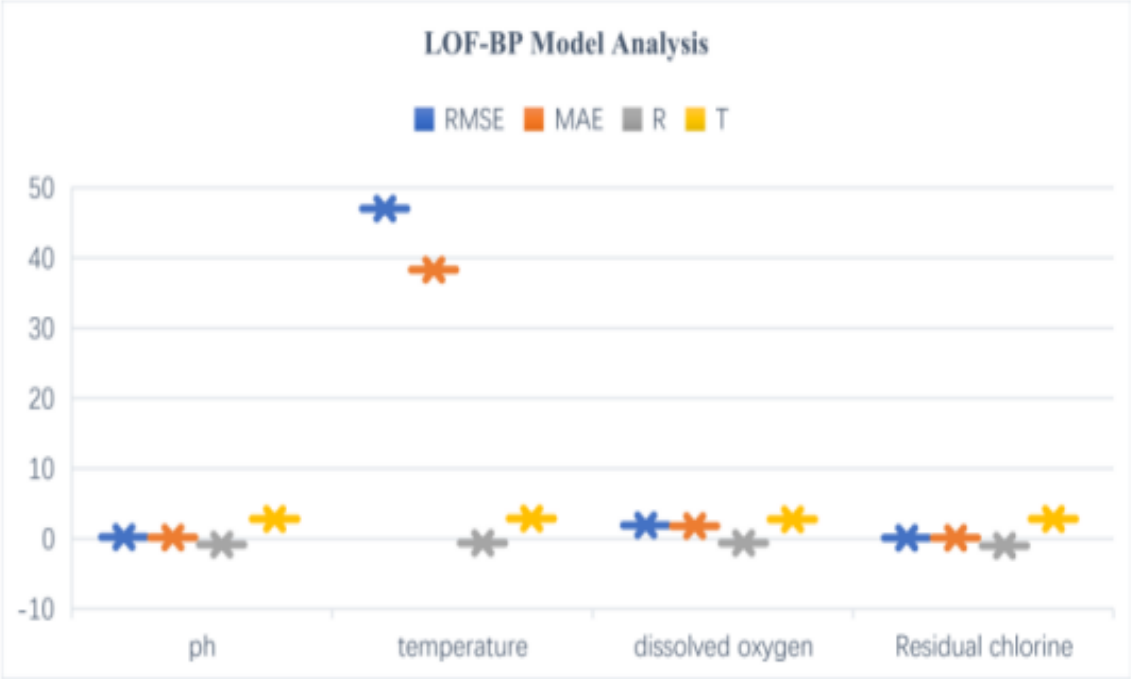
The model analysis results are shown in Figure 6.



(a)



(b)



(c)

Figure 6. Analysis of different models: (a) HA_Cart_AdaBoost; (b) LOF-BP; (c) LOF-SVM

The experimental results reveal that in the index evaluation of T, the HA_Cart_AdaBoost model takes the least time for data cleaning compared to the SVM and BP models. In the index evaluation of RMSE, the HA_Cart_AdaBoost model exhibits better RMSE in terms of temperature and dissolved oxygen. Moreover, R is larger; thus, the HA_Cart_AdaBoost model compensates better for the missing data and has a better fitting trend. Furthermore, the HA_Cart_AdaBoost model has the smallest MAE value among the two indexes of temperature and pH; thus, the HA_Cart_AdaBoost model has smaller error fluctuations. In summary, the comprehensive data-cleaning effect of the HA_Cart_AdaBoost model is better than that of the most advanced data-cleaning methods, LOF-BP and LOF-SVM, and has a higher accuracy rate and is more suitable for cleaning drinking-water-quality data.

RESULTS AND DISCUSSIONS

Based on the experimental results, the HA_Cart_AdaBoost model exhibits superior comprehensive performance and higher accuracy rate in cleaning drinking-water-quality data compared to the advanced data-cleaning methods, LOF-BP and LOF-SVM. This model not only takes less time for data cleaning, but also has better RMSE, R, and MAE values, indicating better compensatory ability for missing data and smaller error fluctuations. Overall, the HA_Cart_AdaBoost model shows promising potential for further optimization and expansion, including integrating with IoT technology for real-time monitoring and cleaning of water-quality data, and applying to other fields for wider data-cleaning needs. These efforts are expected to enhance the better utilization of water-quality data and improve water resource management and protection.

The HA_Cart_AdaBoost model has shown excellent comprehensive performance and high accuracy in cleaning water quality data. Looking ahead, the model can be further optimized and expanded. For example, by combining it with IoT technology and intelligent sensors, real-time monitoring and cleaning of water quality data can be achieved. Additionally, the model can be applied to other fields of data cleaning, such as atmospheric environmental data and soil pollution data, to expand its scope of application. In the future, the HA_Cart_AdaBoost model can be further optimized and improved to meet higher-level data cleaning needs. Advanced machine learning algorithms or deep learning techniques can be used to improve the accuracy and efficiency of data cleaning. Furthermore, research can be conducted on how to better handle different types of water quality data to adapt to a wider range of water quality monitoring needs. Additionally, combining the HA_Cart_AdaBoost model with other data processing technologies, such as anomaly detection and missing value imputation, is also a future research direction. These efforts are expected to promote better utilization of water quality data and improve water resource management and protection. In the future, further research could be conducted to investigate the performance of the HA_Cart_AdaBoost model on larger

datasets and in different water-quality monitoring scenarios. Additionally, the model could be compared with other state-of-the-art data-cleaning methods to further evaluate its effectiveness and suitability for various applications. Overall, the HA_Cart_AdaBoost model has great potential in improving the accuracy and reliability of drinking-water-quality data analysis and modelling.

CONCLUSION

In view of the cleaning of drinking water data, in this paper, a data cleaning method was performed by employing AdaBoost along with the optimal control theory of nonlinear PDEs. First, the HA_Cart_AdaBoost model was used to select the 3σ criterion to screen out outliers and eliminate outlier data. Then, data compensation was performed for missing datasets. The validity of the data cleaning algorithm of the HA_Cart_AdaBoost model was verified by taking the actual drinking-water-quality data. The HA_Cart_AdaBoost model uses the weak classifier as the basic classifier. In the proposed method, after inputting the data into the model, it is weighted using the weighted vector. In each iteration, the weighted vector is updated based on the weighted error rate of the weak classifier to proceed to the next iteration; moreover, the coefficient of the weak classifier is calculated in each iteration. The performance of the weak classifier in the final prediction classification depends on the size of the coefficients, and the combination between them is the advantage of the HA_Cart_AdaBoost model.

To sum up, in the mixed-type missing dataset, the comprehensive cleaning effect of HA_Cart_AdaBoost is superior to that of SVM and BP. The data cleaning effect is better and more suitable for data cleaning of drinking water-quality data for residents.

Funding: This research was funded by University of Economics Ho Chi Minh City (UEH), Vietnam. Fund receiver: Dr. Dang Ngoc Hoang Thanh.
Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

1. Chen Y, Miao D, Zhang H. Neighborhood outlier detection. Expert Syst Appl, 2010, 37(12):8745–9. DOI: 10.1016/j.eswa.2010.06.040.
2. Grubbs FE. Sample criteria for testing outlying observations. Ann Math Stat, 1950, 21:27–58. DOI: 10.1214/aoms/1177729885
3. Fei H, Li G. [An abnormal data detection algorithm for WSN based on K-means clustering]. Jisuanji Gongcheng, 2015, 41(7): 124–8. DOI: 10.3969/j.issn.1000-3428.2015.07.024. (Chinese)
4. Zhao R, Du B, Zhang L. A robust nonlinear hyperspectral anomaly detection approach. IEEE J Sel Top Appl Earth Obs Remote Sens IEEE J-STARS, 2014, 7(4): 1227–34. DOI: 10.1109/JSTARS.2014.2311995.
5. Li PH, Han ZQ, Dong ZH, Gao SB. [Research on SMV packet loss measures based on linear interpolation]. Shaanxi Electr Power, 2014, 42(4): 10-4. (Chinese)
6. Liu JQ, Tao T. [A method for detection and processing abnormal data of RO membrane pressure in waste water treatment]. Sichuan Environ, 2019, 38(1): 19- 23. (Chinese).
7. Breuning MM, Kriegel HP, Sander J, Raymond TYN. LOF: identifying density-based local outliers. ACM Sigmod Rec, 2000, 29(2): 93–104. DOI: 10.1145/335191.335388
8. Zeng C, Peng J, Jun L, Song Z, Zhenyu S, Zhihua L, Huan X, Igor VB, Alexei EH. An adaptive data cleaning framework: a case study of the water quality monitoring system in China. Hydrol Sci J, 2022, 67(7): 1114-29. DOI: 10.1080/02626667.2022.2060106.
9. Yan J, Chen X, Yu Y. A Data Cleaning Framework for Water Quality Based on NLDIW-PSO Based Optimal SVR. IEEE/WIC/ACM International Conference on Web Intelligence (WI), 2018, 336-41. DOI: 10.1109/WI.2018.00-71
10. Meng Q, Yan J. A data cleaning method for water quality based on improved hierarchical clustering algorithm. *Int* J Simul Process Model, 2019, 14(5): 442-51. DOI: 10.1504/IJSPM.2019.104120
11. Singh S, Tu H, Donat W, Pattipati K. Anomaly detection via feature-aided tracking and hidden Markov models. IEEE Trans Syst Man Cybern Syst, 2009, 39(1): 144–59. DOI: 10.1109/AERO.2007.352797
12. Raghuram J, Miller DJ, Kesidis G. Unsupervised low latency anomaly detection of algorithmically generated domain names by generative probabilistic modeling. J Adv Res, 2014, 5(4): 423-33. DOI: 10.1016/j.jare.2014.01.001
13. Emmanuel T, Maupong T, Mpoeleng D, Semong T, Mphago B, Tabona O. A survey on missing data in machine learning. J Big Data, 2021, 8:140. DOI: 10.1186/s40537-021-00516-9.
14. Aggarwal CC. An Introduction to Outlier Analysis. Outlier Analysis. Springer, 2017. DOI: 10.1007/978-3-319-47578-3_1
15. Purwar A, Singh SK. Hybrid prediction model with missing value imputation for medical data. Expert Syst Appl, 2015, 42(13): 5621–31. DOI: 10.1016/j.eswa.2015.02.050
16. Cao H, Zhou K, Chen X, Zhang X. Early chatter detection in end milling based on multi-feature fusion and 3σ criterion. Int J Adv Manuf Technol, 2017, 92: 4387-97. DOI: 10.1007/s00170-017-0476-x

17. Damanik IS, Windarto AP, Wanto A, Andani SR, Saputra W. Decision tree optimization in C4. 5 algorithm using genetic algorithm. J Phys Conf Ser, 2019, 1255(1):012012. DOI: 10.1088/1742-6596/1255/1/012012
18. Hssina B, Merbouha A, Ezzikouri H, Erritali M. A comparative study of decision tree ID3 and C4. 5. Int J Adv Comput Sci Appl, 2014, 4(2): 13-9. DOI: 10.14569/SpecialIssue.2014.040203
19. Jian J, Jianxiang W, Xiaoyi M, Yuding W, Renyong L. Equality of medical health resource allocation in China based on the Gini coefficient method. Iran. J. Public Health, 2015, 44(4):445.
20. Wang X, Yi Z, Wu H. Research and Improvement of Internet Financial Anti-Fraud Rules Based on Information Gain and Support. J Phys Conf Ser, 2018, 1069(1): 012104. DOI: 10.1088/1742-6596/1069/1/012104
21. Breiman L. Random forests. Mach. Learn., 2001, 45(1): 5–32. DOI: 10.1023/A:1010933404324
22. Tang F, Ishwaran H. Random Forest Missing Data Algorithms. Stat Anal Data Min, 2017, 10(6): 363–77. DOI: 10.1002/sam.11348.
23. He YG, Wu SS, Zhou H. [Analysis of medium and long-term power demand forecasting in Hunan province]. Hunan Electr Power, 2018, 38(6): 20–4. (Chinese)
24. Du X, Feng JY, Lu SQ, Shi W. [PM2. 5 concentration prediction model based on random forest regression analysis]. Telecom Sci, 2017, 33(7): 66-75. (Chinese).
25. Lee CW, Hsieh KY, Hsiao SY. A Dynamic Data Placement Strategy for Hadoop in Heterogeneous Environments. Big Data Res, 2014, 1:14-22. DOI: 10.1016/j.bdr.2014.07.002
26. Mohd U, Mengchen L, Min C. Job schedulers for Big data processing in Hadoop environment: testing real-life schedulers using benchmark programs. Curr For Rep, 2017, 3(4): 260-73. DOI: 10.1016/j.dcan.2017.07.008
27. Beyaz A, Gerdan D. Meta-learning-based prediction of different corn cultivars from color feature extraction. J Agric Sci, 2021, 27(1): 32-41. DOI: 10.15832/ankutbd.567407
28. Gerdan D, Beyaz A, Vatandaş M. Classification of Apple Varieties: Comparison of Ensemble Learning and Naive Bayes Algorithms in H2O Framework. J Agric Fac Gaziosmanpaşa Univ, 2020, 37 (1), 9-16. DOI: 10.13002/jafag4646