International Journal of Internet and Distributed Systems, 2024, 6, 1-23
https://www.scirp.org/journal/ijids
ISSN Online: 2327-7165
ISSN Print: 2327-7157

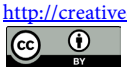
Bayesian Classifier Based on Robust Kernel Density Estimation and Harris Hawks Optimisation

Bi Iritie A-D Boli, Chenghao Wei

School of Computer Science, Hubei University of Technology, Wuhan, China
Email: chenghao.wei@hbut.edu.cn

How to cite this paper: Boli, B.I.A-D and Wei, C.H. (2024) Bayesian Classifier Based on Robust Kernel Density Estimation and Harris Hawks Optimisation. *International Journal of Internet and Distributed Systems*, **6**, 1-23.
https://doi.org/10.4236/ijids.2024.61001

Received: February 1, 2024
Accepted: February 22, 2024
Published: February 25, 2024



Abstract

In real-world applications, datasets frequently contain outliers, which can hinder the generalization ability of machine learning models. Bayesian classifiers, a popular supervised learning method, rely on accurate probability density estimation for classifying continuous datasets. However, achieving precise density estimation with datasets containing outliers poses a significant challenge. This paper introduces a Bayesian classifier that utilizes optimized robust kernel density estimation to address this issue. Our proposed method enhances the accuracy of probability density distribution estimation by mitigating the impact of outliers on the training sample's estimated distribution. Unlike the conventional kernel density estimator, our robust estimator can be seen as a weighted kernel mapping summary for each sample. This kernel mapping performs the inner product in the Hilbert space, allowing the kernel density estimation to be considered the average of the samples' mapping in the Hilbert space using a reproducing kernel. M-estimation techniques are used to obtain accurate mean values and solve the weights. Meanwhile, complete cross-validation is used as the objective function to search for the optimal bandwidth, which impacts the estimator. The Harris Hawks Optimisation optimizes the objective function to improve the estimation accuracy. The experimental results show that it outperforms other optimization algorithms regarding convergence speed and objective function value during the bandwidth search. The optimal robust kernel density estimator achieves better fitness performance than the traditional kernel density estimator when the training data contains outliers. The Naïve Bayesian with optimal robust kernel density estimation improves the generalization in the classification with outliers.

Keywords

Classification, Robust Kernel Density Estimation, M-Estimation, Harris

Hawks Optimisation Algorithm, Complete Cross-Validation

1. Introduction

Data mining systematically extracts valuable knowledge from vast quantities of noisy data [1]. Anomalous data records in production are expected to be encountered for numerous reasons. These abnormal data can disturb actual knowledge and potentially result in incorrect outcomes. Hence, it is imperative to eradicate the disruption caused by outliers [2]. In addition to recognizing and deleting outliers [3], it is usually possible to dynamically attenuate the impact of these outliers on the learning model.

Improving classification accuracy for continuous data sets is a critical issue for Bayesian classifiers due to their reliance on accurate probability density estimation. Continuous data often exhibit complexities such as noise and outliers, which can skew the density estimates and lead to misclassification. Accurate density estimation is essential for determining the posterior probabilities that inform the classification decision. When outliers are present, traditional methods may produce biased estimates, adversely affecting the classifier's performance. Therefore, enhancing classification accuracy through robust methods like optimized kernel density estimation is paramount, as it mitigates the influence of outliers and improves the overall reliability of the classifier's predictions.

This study presents a specific technique to enhance the classifier's generalization in classification when the training data includes outliers. The proposed approach utilizes a new Bayesian classifier that relies on Optimised Robust Kernel Density Estimation (ORKDE). This method employs the Harris Hawks Optimisation (HHO) [4] as its optimization strategy to minimize the Complete Cross-validation (CCV) objective function for determining the optimal bandwidth.

Additionally, M-estimation techniques can ensure the model's robustness during estimation. This procedure effectively eliminates outliers that disrupt the legitimate data. The method employs the Iterative Weighted Least Squares Method (IWLSM) along with the Hampel cost function to reduce the impact of outliers [5] and leads to a precise calculation of the data distribution. The ORKDE technique is employed for class conditional probability calculations in Bayesian classifiers. The outcomes of this method are used to infer class labels for the test samples.

The study is organized as follows: the initial section discusses the classification learning issue when polluted samples are included and analyses its impact on a classification model's decision boundary. The second section provides the estimation basic principles of the Bayesian classifier for attributes with continuous values. The third section explores the basic tenets of Robust Kernel Density Estimation (RKDE). The fourth section discusses the influence of bandwidth on estimates and provides the HHO technique for optimizing bandwidth. Experimental results conclusively validate the excellent performance of the proposed method.

2. Description of Outlier Problem in Classification

Suppose there are training data samples:

$$X_1,\cdots,X_i,\cdots,X_L \overset{\text{i.i.d}}{\sim} \theta_1\phi_1+\theta_2\phi_2+\cdots+\theta_\Lambda\phi_\Lambda \tag{1}$$

Let X_i represent a random sample with d-dimension, where each sample is independently and identically distributed according to an unknown density function ϕ_1. The remaining samples follow a distribution function for contaminated data. θ_1 represents the proportion of samples generated by ϕ_1, and it always exists $\theta_1 \geq \theta_2+\theta_3+\cdots+\theta_\Lambda$. In a binary model problem, the decision boundary reflects how the model's preferences are influenced by the distribution of the outliers [6]. In **Figure 1**, l_{a_2} demonstrates that the density of outlier samples generated by the ϕ_2 distribution has a harmful impact on l_{a_1}. The model erroneously identifies the test samples depicted as pentagons during the testing. Biassed Bayesian inference is often caused by the misrepresentation of sample decision boundaries caused by inaccurate distribution estimations. The key to fixing the problem is achieving precise density estimates, even when outlier data is included. Methods for estimating general probability distributions can be classified into parametric estimation [6] and non-parametric estimation [7]. Parametric estimation methods assume that data is drawn from a known probability distribution with unknown parameters, allowing the inference problem to be framed as a parametric solution. These techniques are widely used in statistical and machine-learning applications. However, making assumptions about the underlying data distribution can be challenging in real-world scenarios, often requiring substantial prior knowledge. This limitation hinders the ability to draw conclusive insights about the distribution's structure and parameters [8]. In some cases, the mathematical representation of the problem is crucial for its resolution, particularly in outlier detection and robust estimation. Parametric methods may struggle when data deviates from the assumed distribution or is complex.

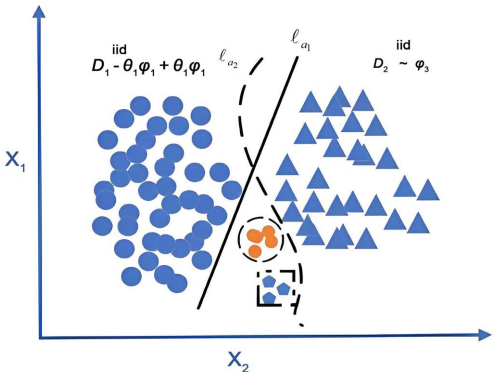


Figure 1. Decision boundary for outlier perturbation.

Furthermore, these outlier problems tend to occur in regions where formulating and thoroughly describing modeling assumptions is naturally challenging. The benefit of non-parametric estimation is its ability to avoid assuming a

predetermined form for the general distribution [9]. Instead, it seeks to extract the density information directly from the data and construct a statistical model to characterize it independently. This approach also offers a more accurate depiction of the density distribution in the complex region.

Non-parametric models usually have relevant advantages, such as fewer overall assumptions, broad applicability, and ease of understanding [10]. Such estimation strategies are significantly less restrictive. Typical estimation methods include histogram estimation [11], which can be used to estimate the frequency variation of the data. However, the resulting curve is not smooth, does not reflect the density function's local details, and is unsuitable for outliers. Kernel Density Estimation (KDE) [12], also known as Parzen window estimation, can be considered an additive model composite applied to the density function estimation. Assuming that the data $X_1, X_2, \cdots, X_n \in \mathbb{R}^d$ is taken from an unknown continuous distribution $\phi(x)$, the density estimation at point x is defined as $\tilde{\phi}(x) = \frac{1}{n}\sum_{i=1}^{n} K_H(x, X_i)$, where K is called the kernel function, H is the bandwidth parameter, and the kernel function itself satisfies the conditions $K_H(\cdot,\cdot) \geq 0$, $\int\left(K_H(x,\cdot)\right)\mathrm{d}x = 1$.

Standard kernel functions include the Triangular kernel, Epanechikov kernel, Gaussian kernel, etc. Mathematically, the estimate can be considered the average of the kernel mapping for each sample. Outliers can easily affect such a method, and it isn't easy to estimate the density function accurately [13]. The weighting strategy needs to be considered, which treats each sample unequally according to its importance.

3. Bayesian Classifier

The Bayesian classifier is a traditional fundamental classifier, and it is designed to minimize the theoretical average risk at a predetermined cost [14]. The method has a strong theoretical foundation in statistical analysis and aims to maximize the Bayesian *posteriori* probability. It also involves calculating the *posteriori* probability by using Bayesian theory. The class with the highest *posteriori* probability is then selected as the label for the test object. Suppose the dataset $D = \{X_{\text{train}}, X_{\text{test}}\}$, where $X_{\text{train}} = \{x_1, \cdots, x_i, \cdots, x_N\}$, $X_{\text{test}} = \{x_1, \cdots, x_j, \cdots, x_M\}$, and $x_i, x_j \in \mathbb{R}^d$, the label of any training sample x_i belongs to $C = \{c_1, \cdots, c_k, \cdots, c_W\}$. At this point, the Bayesian maximized posterior can be expressed as:

$$\begin{aligned} &\arg\max_{k=1,2,\cdots,W} P(c_k \mid X = x_{\text{test}}) \\ &= \arg\max_{k=1,2,\cdots,W} \frac{P(c_k)P(X = x_{\text{test}} \mid c_k)}{P(X)} \\ &\propto \arg\max_{k=1,2,\cdots,W} P(c_k)P(X = x_{\text{test}} \mid c_k) \end{aligned} \tag{2}$$

The calculation of the value $P\left(X = x_j^{\text{test}} \mid c_k\right)$ in Equation (2) is essential when all of the attribute values are continuous. Assuming that the attribute values are

independent, the calculation of this value uses $P\left(X = x_{jq}^{\text{test}} \mid c_k\right)$. Non-parametric estimation methods can extract the density information for a continuous random variable. If the sample contains outliers, it is necessary to use a robust estimate approach to accurately calculate the probability value and the class-conditional probability density value [15].

4. Robust Kernel Density Estimator

RKDE achieves resilience in estimating density by integrating traditional kernel density estimation with the M-estimation statistic technique. The KDE process for a Positive Semi-Definite (PSD) kernel can be viewed as the inner product of the average sample values in the Reproducing Kernel Hilbert Space (RKHS) [16]. Outliers in the original space can change the sample means in the Hilbert space. The method used the M-estimation technique to get accurate estimates of the means in the RKHS. RKDE employs a Kernelized Iteratively Reweighted Least-Squares (KIRWLS) approach to minimize the robust loss function. The convergence of this method has also been demonstrated [17].

4.1. Kernel Density Estimation Using the Hilbert Space Mapping

There is a mapping function $\Phi : \mathbb{R}^d \to \mathscr{H}$, where $\mathscr{H}$ represents the Hilbert space. The kernel function $K\left(x, X_i\right) = \left\langle \Phi\left(x\right), \Phi\left(X_i\right) \right\rangle$ can be defined as the inner product of the Hilbert space resulting from the mapping. By examining the KDE from this viewpoint, the function can be formulated as the following equation:

$$\tilde{\phi}\left(x\right) = \frac{1}{n}\sum_{i=1}^{n}\left\langle \Phi\left(x\right), \Phi\left(X_i\right) \right\rangle = \left\langle \Phi\left(x\right), \frac{1}{n}\sum_{i=1}^{n}\Phi\left(X_i\right) \right\rangle \tag{3}$$

According to the Equation (3), the KDE can be understood as the inner product between the mean of $\Phi\left(X_i\right)$ and the mapping of $\Phi\left(x\right)$. Due to the property of the RKHS, the estimated KDE value is equal to the mean of the sample mapping in the RKHS. The average value is highly affected by the extreme values among the samples' mapping and needs to be calculated by using a robust estimator [18]. Replace the sample mean with a robust M-estimation.

$$\hat{\eta} = \underset{\eta \in \mathscr{H}}{\operatorname{argmin}} \sum_{i=1}^{n} \psi\left(\left\| \Phi\left(X_i\right) - \eta \right\|\right) \tag{4}$$

By using the $\hat{\eta}$ and weighted form, the RKDE formula is expressed as follows.

$$\begin{aligned} \hat{\phi}\left(x\right) &= \left\langle \Phi\left(x\right), \hat{\eta} \right\rangle = \left\langle \Phi\left(x\right), \sum_{i=1}^{n} a_i \Phi\left(x_i\right) \right\rangle \\ &= \sum_{i=1}^{n} a_i \left\langle \Phi\left(x\right), \Phi\left(X_i\right) \right\rangle = \sum_{i=1}^{n} a_i K\left(x, X_i\right) \end{aligned} \tag{5}$$

The resulting kernel density estimator is provided in a weighted form.

$$\hat{\phi}\left(x\right) = \sum_{i=1}^{n} a_i K_h\left(X, X_i\right) \tag{6}$$

In contrast to the conventional KDE, this expression incorporates weighted

samples, eliminating the use of equal weighting. The weights adhere to the conditions $a_i \geq 0$, $\sum_{i=1}^{n} a_i = 1$, with a tendency to provide lower weights to outlier data points. And ψ in the Equation (4) is the robust loss function, and the value of $\left\|\Phi(X_i) - \eta^{(k)}\right\|$ can be calculated by this way.

$$\begin{aligned}\left\|\Phi(X_i) - \eta^{(k)}\right\|^2 &= \left\langle \Phi(X_i) - \eta^{(k)}, \Phi(X_i) - \eta^{(k)} \right\rangle \\ &= \left\langle \Phi(X_i), \Phi(X_i) \right\rangle - 2\left\langle \Phi(X_i), \eta^{(k)} \right\rangle + \left\langle \eta^{(k)}, \eta^{(k)} \right\rangle\end{aligned} \tag{7}$$

There is also $\eta^{(k)} = \sum_{j=1}^{n} a_j^{(k-1)} \Phi\left(X_j\right)$, at the same time.

$$\begin{aligned}\left\langle \Phi(X_i), \Phi(X_i) \right\rangle &= K(X_i, X_i) \\ \left\langle \Phi(X_i), \eta^{(k)} \right\rangle &= \sum_{j=1}^{n} a_j^{(k-1)} K\left(X_i, X_j\right) \\ \left\langle \eta^{(k)}, \eta^{(k)} \right\rangle &= \sum_{j=1}^{n}\sum_{i=1}^{n} a_j^{(k-1)} a_i^{(k-1)} K\left(X_j, X_i\right)\end{aligned} \tag{8}$$

4.2. M-Estimator and Kernelized Iterative Reweighting Algorithm

M-estimator is a robust estimator that extends the maximum likelihood estimator to handle contamination in the distribution [19]. The M-estimator effectively resolved the Equation (1), which resembles the maximum likelihood estimator but employs a distinct loss function. The maximum likelihood estimator for multivariate functions can be used to substitute the likelihood function with ρ in the following manner.

$$\hat{\theta} = \arg\max_{\theta} \sum_{i=1}^{n} \ln f\left(x_i; \theta\right) = \arg\min_{\theta} \sum_{i=1}^{n} \rho\left(\left\|x_i - \theta\right\|\right) \tag{9}$$

The function ρ is strictly convex or a continuous symmetric function in the positive semi-axis. If ρ is derivable, the M-estimator can also be expressed as the following equation.

$$\sum_{i=1}^{n} \psi\left(x_i - \hat{\theta}\right) = 0 \tag{10}$$

This evaluation function is derivative-related to the function, and the solution of Equation (10) is equivalent to Equation (9) if ρ is a convex function and its derivative exists everywhere. By selecting appropriate evaluation functions ψ, M estimation can achieve efficient estimation while ensuring essential robustness performance. Standard robust estimation functions include the Huber and the Hampel loss functions. The Huber function is defined in the following equation:

$$\rho(u) = \begin{cases} \dfrac{u^2}{2}, & \text{if } |u| \leq \gamma \\ \gamma|u| - \dfrac{\gamma^2}{2}, & \text{if } \gamma < |u| \end{cases} \tag{11}$$

Then, we can make the ψ function.

$$\psi(u)=\begin{cases}-\gamma, & \text{if } u<-\gamma\\ u, & \text{if } -\gamma\le u\le\gamma\\ \gamma, & \text{if } u>\gamma\end{cases} \tag{12}$$

To control the parameters γ, for outliers, at that time, it is a constant, but when $\|x-\eta\|<\gamma$, and the $\psi(u)=u$, it increases monotonically. Using the Huber activation function, the objective function in Equation (6) can be calculated using an iterative algorithm to optimize weights.

$$\psi(u)=\begin{cases}u, & 0\le u<\gamma_1\\ \gamma_1, & \gamma_1\le u<\gamma_2\\ \dfrac{\gamma_3-u}{\gamma_3-\gamma_2}, & \gamma_2\le u<\gamma_3\\ 0, & \gamma_3\le u\end{cases} \tag{13}$$

Algorithm 4.2 outlines the computational procedure for the kernelized iterative reweighting algorithm. During the update, the weights are calculated based on Equation (8), Equations. (5), and (9). After each iteration, the weights of the samples are normalized and reintroduced into the formula for further calculations. The computation terminates and outputs the optimal weights if the value is below a specified threshold.

Algorithm 1 Kernelized Iterative Reweighting Algorithm

1: **Input:** Sample data $X_1, X_2, \ldots, X_n$, stop threshold value V
2: **Output:** Optimal objective function fitness value and corresponding optimized parameters $A^{(*)}=[a_1^*, a_2^*, \ldots, a_n^*]$
3: Initialize $A^{(0)}$ and iteration variable $k=0$
4: Compute $\left\|\Phi(X_i)-\eta^{(k)}\right\|$ using the Eqs. (7) and (8)
5: Calculate $\tilde{a}_i^{(k+1)}=\frac{\psi(\|\Phi(x_i)-\eta(k)\|)}{\|\Phi(x_i)-\eta(k)\|}$
6: Normalizing $\tilde{a}_i^{(k+1)}$, and generate the $A^{(k+1)}$
7: Compute $\hat{\eta}^{(k+1)}=\sum_{i=1}^{n}\psi(\|\Phi(X_i)-(\eta^k)\|)$
8: If $\hat{\eta}^{(k+1)}-\hat{\eta}^k<V$ is satisfied, the objective function converges and outputs the optimal weight vector $A^{(*)}$, otherwise, $k\leftarrow k+1$ and go to step 4.

5. Optimized Robust Kernel Density Estimation

Optimizing RKDE is achieved by a combination of advanced statistical techniques to enhance the performance of Bayesian classifiers, particularly in the presence of outliers. The process begins with M-estimation to minimize the influence of extreme values by iteratively updating parameter estimates based on weighted least squares. The weights assigned to each sample are derived from their distance to the estimated mean, allowing the method to dampen outliers' impact on the overall density estimation [20]. The influence of bandwidth on the accuracy of estimations is a critical aspect that must be addressed in the kernel density estimation process. The method employs CCV to ascertain the optimal bandwidth for the kernel density estimator. This process entails partitioning the dataset into multiple training and validation subsets, thereby enabling a comprehensive assessment of the estimator's performance across a range of bandwidth values. By systematically

evaluating the integrated squared error (ISE) for each bandwidth, the method identifies the bandwidth that minimizes the ISE, thereby ensuring that the estimator is finely tuned to the specific characteristics of the dataset. This tailored approach enhances the accuracy of the density estimates and mitigates the adverse effects of outliers, ultimately leading to more reliable probabilistic assessments in the context of Bayesian classification. The integration of CCV into the methodology underscores the importance of bandwidth selection as a pivotal factor in achieving robust and accurate density estimation [21].

5.1. Complete Cross-Validation

CCV is a proper way to obtain the optimal bandwidth. Unbiased Cross-Validation (UCV), Biased Cross-Validation (BCV) and Bootstrap comprise the CCV. The ISE is defined as the following equation.

$$\mathrm{ISE}(h)=\int\left(\hat{\phi}(x)-f(x)\right)^2\mathrm{d}x \tag{14}$$

Determining that

$$\mathrm{ISE}(h)=R\left(\hat{\phi}(x)\right)-2\int\hat{\phi}(x)f(x)\mathrm{d}x+R\left(f(x)\right) \tag{15}$$

the roughness of the estimated value is denoted by $R\left(\hat{\phi}\right)=\int\cdots\int\hat{\phi}(x)^2\,\mathrm{d}x$. This term can be ignored since $R\left(f(x)\right)$ does not vary with *h*. Using a leave-one-out estimator, cross-validation is performed for calculation.

$$\hat{\phi}_{-i}(x_i)=\frac{1}{(n-1)h_1\cdots h_d}\sum_{j\neq 1}^{n}\left\{\prod_{k=1}^{d}K\left(\frac{x_{ik}-x_{jk}}{h_k}\right)\right\} \tag{16}$$

Allows us to estimate the second term in Equation (15) by noting that

$$\mathbb{E}\hat{\phi}_{-i}(x_i)=\int\hat{\phi}(x)f(x)\mathrm{d}x \tag{17}$$

the equation Equation (18) implies an approximation.

$$\mathrm{UCV}(h)=R\left(\hat{\phi}\right)-\frac{2}{n}\sum_{i=1}^{n}\hat{\phi}_{-i}(x_i) \tag{18}$$

Hence, the expected value of UCV is equal to the expected value of ISE subtracted by the constant. Therefore, an alternative term for this approach is unbiased cross-validation. The generalization of least-squares cross-validation to arbitrary dimensions can be achieved by using the multivariate product kernel. It is constructed as follows: let X be an $n\times d$ data matrix of random vectors $x_1,x_2,\cdots,x_n$, where X_i are independent observations sampled from a multivariate density $f(x)$ of dimension *d*. Let x_{ij} denote the *j*th entry of x_i. Given is the multivariate product kernel estimator of $f(x)$

$$\hat{\phi}(x)=\frac{1}{2n}\sum_{i=1}^{n}\prod_{j=1}^{d}\frac{1}{h_j}K\left(\frac{x_{ij}}{h_j}\right) \tag{19}$$

where h_j is the smoothing parameter for the *j*th dimension. The smoothness of $\hat{\phi}(x)$ when using the standard Normal kernel for K is demonstrated

$$R\left(\hat{\phi}\right)=\frac{1}{\left(2\sqrt{\pi}\right)^{d} nh_1\cdots h_d}+\frac{1}{\left(2\sqrt{\pi}\right)^{d} n^2h_1\cdots h_d}\sum_{i=1}^{n}\sum_{j\neq i}^{n}\exp\left(-\frac{1}{4}\sum_{k=1}^{d}\delta_{ijk}^2\right) \tag{20}$$

where $\delta_{ijk}=\left(x_{ik}-x_{jk}\right)\big/h_k$. The second part of the UCV estimator in Eq. (18) is given explicitly by the coming equation.

$$\frac{2}{n}\sum_{i=1}^{n}\hat{\phi}_{-i}\left(x_i\right)=\frac{2}{\left(\sqrt{2\pi}\right)^{d} n^2h_1\cdots h_d}\sum_{i=1}^{n}\sum_{j\neq i}^{n}\exp\left(-\frac{1}{2}\sum_{k=1}^{d}\delta_{ijk}^2\right) \tag{21}$$

For the sake of simplicity, the factor $n-1$ has been substituted with a large *n*. The multivariate least-squares cross-validation function, $\mathrm{UCV}\left(h_1,\cdots,h_d\right)$, is obtained.

$$\begin{aligned}\mathrm{UCV}\left(h_1,\cdots,h_d\right)=&\frac{1}{\left(2\sqrt{\pi}\right)^{d} nh_1\cdots h_d}+\frac{1}{\left(2\sqrt{\pi}\right)^{d} n^2h_1\cdots h_d}\\&\times\sum_{i=1}^{n}\sum_{j\neq i}^{n}\left[\exp\left(-\frac{1}{4}\sum_{k=1}^{d}\delta_{ijk}^2\right)-\left(2\times 2^{d/2}\right)\exp\left(-\frac{1}{2}\sum_{k=1}^{d}\delta_{ijk}^2\right)\right]\end{aligned} \tag{22}$$

The BCV method integrates bias correction into the UCV method [22] for addressing potential biases that may occur from the iterative exclusion of individual data points. The following equation expresses the mean integrated squared error (MISE).

$$\mathrm{MISE}\left(h\right)=\int_{\mathbb{R}^d}E\left(\hat{\phi}\left(x\right)-f\left(x\right)\right)^2\mathrm{d}x \tag{23}$$

The general form of BCV using the asymptotic form is the following equation [23].

$$\begin{aligned}\mathrm{BCV}\left(h_1,\cdots,h_d\right)=&\frac{1}{\left(2\sqrt{\pi}\right)^{d} nh_1\cdots h_d}+\frac{1}{4n\left(n-1\right)h_1\cdots h_d}\\&\times\sum_{i=1}^{n}\sum_{j\neq i}^{n}\left[\left(\sum_{k=1}^{d}\delta_{ijk}^2\right)^2-\left(2d+4\right)\left(\sum_{k=1}^{d}\delta_{ijk}^2\right)+\left(d^2+2d\right)\right]\prod_{k=1}^{d}\phi\left(\delta_{ijk}\right)\end{aligned} \tag{24}$$

Extensive research has been conducted on applying bootstrap in univariate kernel density estimation. A simplistic method for utilizing the bootstrap involves resampling $x_1^*,\cdots,x_n^*$ from the original dataset and subsequently creating a bootstrap density estimate in the following manner.

$$\hat{\phi}^*\left(x\right)=\frac{1}{nh}\sum_{i=1}^{n}K\left(\frac{x-x_1^*}{h}\right)=\frac{1}{n}\sum_{i=1}^{n}K_h\left(x-x_1^*\right) \tag{25}$$

From Equation (23), the global MISE criterion can be computed from the pointwise MSE criterion.

$$\mathrm{MSE}\left(X\right)=E\left[\left(\hat{\phi}\left(x\right)-f\left(x\right)\right)^2\right] \tag{26}$$

This formula's naive bootstrap estimate of $\mathrm{MSE}\left(x\right)$ *expresses*.

$$\mathrm{MSE}^*\left(X\right)=E^*\left[\left(\hat{\phi}^*\left(x\right)-\hat{\phi}\left(x\right)\right)^2\right] \tag{27}$$

However, with the naive bootstrap, we can obtain the following,

$$E^*\hat{\phi}^*(x) = EK_h(x - x_1^*) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \hat{\phi}(x) \tag{28}$$

here, x_1^* has a discrete uniform distribution over the original data. Thus, the simple bootstrap estimate of the MSE in Equation (27) fails because the bias, $E^*\hat{\phi}^*(x) - \hat{\phi}(x)$, vanishes and only the variance is estimated.

Reducing bias has been approached in many ways, limiting the resampling size to be less than the original sample size and building the estimator [24] [25], and it is recommended using smoothed bootstrap to calculate MSE with Equation (27) [26]. To clarify, x_1^* is resampled from the kernel estimate $\hat{\phi}(x) = n^{-1}\sum_{i=1}^{n} K_h(x - x_i)$ instead of the original data. The smoothed bootstrap technique using a Normal kernel does not require resampling because the calculations can be written in a closed form. The new MISE of the estimator is the following form,

$$\hat{\mathrm{MISE}}(h) = \int_{R^d} E^*\left\{\hat{\phi}^*(x) - \hat{\phi}(x)\right\}^2 \mathrm{d}x \tag{29}$$

where $\hat{\phi}(x)$ is the multivariate product kernel estimator given in Equation (19). $\hat{\phi}^*(x)$ is a multivariate produce kernel estimator calculated with data sampled from $\hat{\phi}(x)$, and the expectation, E^*, is taken for the density $\hat{\phi}(x)$. The aim is to choose h to minimize the value of Equation (29).

By employing the standard kernel for K in Equation (19) and evaluating Equation (29), it derives the multivariate bootstrap criterion function minimize across $h_1, \cdots, h_d$. We eliminated the $i = j$ terms in the sums to reduce the inflated variance of this resampling strategy [27] [28].

$$\begin{aligned} B(h_1, \cdots, h_d) = {} & \frac{1}{\left(2\sqrt{\pi}\right)^d n h_1 \cdots h_d} + \frac{1}{\left(2\sqrt{\pi}\right)^d n^2 h_1 \cdots h_d} \\ & \times \sum_{i=1}^{n}\sum_{j\neq i}^{n}\left[\frac{n-1}{2^{d/2} n}\exp\left\{-\frac{1}{8}\sum_{k=1}^{d}\delta_{ijk}^2\right\} + \exp\left\{-\frac{1}{4}\sum_{k=1}^{d}\delta_{ijk}^2\right\}\right. \\ & \left. - \frac{2\times 2^{d/2}}{3^{d/2}}\exp\left\{-\frac{1}{6}\sum_{k=1}^{d}\delta_{ijk}^2\right\}\right] \end{aligned} \tag{30}$$

CCV leverages components' advantages while minimizing their downsides. It analyses complex, noisy intelligence data well. This combination completes the RKDE bandwidth selection analysis. From Equation (22), Equations. (24) and (30), CCV can be expressed as the following equation.

$$\mathrm{CCV} = \mathrm{UCV}(h_1, \cdots, h_d) + \mathrm{BCV}(h_1, \cdots, h_d) + B(h_1, \cdots, h_d) \tag{31}$$

5.2. Harris Hawks Optimisation Algorithm

The HHO algorithm is then applied to optimize the CCV objective function. HHO mimics the hunting strategies of Harris hawks and is particularly effective in exploring the parameter space for optimal solutions [29]. The implementation involves initializing a population of hawks, each representing a potential solution in the bandwidth search space. The fitness of each hawk is evaluated based on the

CCV error, and their positions are iteratively updated to converge toward the optimal bandwidth. This optimization process is crucial as it ensures that the bandwidth selection is not only practical but also efficient, leading to improved classification performance. This hunting method comprises two phases, which are exploration and exploitation. Each phase mimics hawk predation behaviors [30]. Time decreases the prey's escape energy, represented by EG in the following formula.

$$EG = 2EG_0\left(1-\frac{t}{T}\right) \tag{32}$$

Here, EG_0 is equal to $2r_1-1$ that represents the initial energy state, r_1 is a random value $(0,1)$, and EG_0 fluctuates between $[-1,1]$ during each iteration. The maximum and current iteration numbers are T and t, respectively. Harris Hawks capture tactics depend on EG. They are in the exploration phase when $|EG|$ is greater than 1. Otherwise, they are in the exploitation phase.

In the exploration phase, hawks follow target animals and other flock members' erratic prey-seeking movements to establish their location [31]. The following equation describes the mathematical model.

$$p(t+1)=\begin{cases} p_{rand}(t)-r_1\left|p_{rand}(t)-2r_2p(t)\right|, & q\geq 0.5 \\ \left(p_{prey}(t)-p_m(t)\right)-r_3\left[LB+r_4(UB-LB)\right], & q<0.5 \end{cases} \tag{33}$$

In each iteration, r_1, r_2, r_3, r_4, and q, which are random values in $[0,1]$. Searching upper and lower boundaries are UB and LB. In the next iteration, $p(t+1)$ represents the hawks' position, $p_{prey}(t)$ represents the optimal position of prey, $p(t)$ is the current hawk position, p_{rand} means a randomly selected hawk, and $p_m(t)$ is the average position of the current population. The following formula calculates the average position of hawks,

$$p_m(t)=\frac{1}{N}\sum_{i=1}^{N}p_i(t) \tag{34}$$

where $p_i(t)$ represents each hawk's location in the iteration t.

In the exploitation phase, hawks chase and catch the victim during this period. It can be modeled by four predatory behaviors: soft encirclement, aggressive encirclement, gradual swift fall, and gradual rapid descent. HHO chooses a different strategy based on the EG and the probability r.

When $|EG|\geq 0.5$, $r\geq 0.5$, the HHO carries out the soft besiege. In this stage, the prey still has the energy to escape, and the hawks use a weak encircle strategy to deplete it and launch a surprise attack. The following equation models the behavior

$$p(t+1)=\Delta p(t)-EG\left|J\times p_{prey}(t)-p(t)\right| \tag{35}$$

and

$$\begin{aligned} \Delta p(t) &= p_{prey}(t)-p(t) \\ J &= 2\left(1-r_5\right) \end{aligned} \tag{36}$$

where $\Delta p(t)$ is the difference between the prey position and the Harris Hawks' current position in iteration t, r_5 is a random value between (0, 1), and J is the rabbit's random jump strength when escaping. To replicate rabbit motions, J changes randomly in each iteration.

When $|EG| < 0.5$, $r \geq 0.5$, the HHO executes the hard besiege. In this stage, the prey has no energy to escape, and the Harris hawks use a hard encircling to hunt the prey for a final assault. The current position is updated according to the following equation.

$$p(t+1) = p_{prey}(t) - EG\left|\Delta p(t)\right| \tag{37}$$

When $|EG| \geq 0.5$, $r < 0.5$, the HHO fulfills the soft besiege with progressive rapid dive. In this stage, the prey has enough energy to escape from the eagles, and the Harris hawks will gradually dive and softly surround the target. The *levy* function is used in the position update process to simulate the prey's escape mode and jumping action. This process is modeled as follows equation.

$$\begin{aligned} Y &= p_{prey}(t) - EG\left|J \times p_{prey}(t) - p(t)\right| \\ Z &= Y + S \times LF(D) \end{aligned} \tag{38}$$

D is the problem dimension and S is the D-dimensional random row vector. LF is a levy flight function expressed as Eq. (39).

$$LF(\beta) = 0.01 \times \frac{u * \sigma}{|v|^{\frac{1}{\beta}}},\ \sigma = \left(\frac{r\left(1 + \beta * \sin\frac{\pi\beta}{2}\right)}{r\left(\frac{1+\beta}{2}\right) * \beta * 2^{\frac{\beta-1}{2}}} \right)^{\frac{1}{\beta}} \tag{39}$$

r is the escape probability, v and u are random values in the range [0, 1], and the specificity is 1.5. The final hawk's position is determined as in Equation (40).

$$p(t+1) = \begin{cases} Y, & \text{if } F\left(p(t)\right) > F(Y) \\ Z, & \text{if } F\left(p(t)\right) > F(Z) \end{cases} \tag{40}$$

where F is the fitness function.

When $|EG| < 0.5$, $r < 0.5$, the HHO carries out the hard besiege with a progressive rapid dive. In this stage, the prey is exhausted and has low escape energy; the Harris Hawks surround the prey through the hard besiege with a progressive rapid dive. They attempt to reduce the distance between their average position and the target prey. The position update formula is expressed in this equation.

$$p(t+1) = \begin{cases} Y, & \text{if } F\left(p(t)\right) > F(Y) \\ Z, & \text{if } F\left(p(t)\right) > F(Z) \end{cases} \tag{41}$$

where Y and Z are updated by Equation (42).

$$\begin{aligned} Y &= p_{prey}(t) - EG\left|Jp_{prey}(t) - p_m(t)\right| \\ Z &= Y + S \times LF(D) \end{aligned} \tag{42}$$

The algorithm 5.2 relates the basic flow of Harris Hawks Optimization.

Algorithm 2 The proposed HHO-CCV algorithm

Input: Fitness CCV function F, Population size N, Maximum number of iterations $MaxItr$, convergence threshold $Threshold$
Output: Optimal objective function fitness value $F(x_{\text{prey}})$ and optimised parameters x_{prey}
Initialize: Generate random population $(x_i; i\varepsilon(1,N))$, $EG_0 = 2\text{rand}(0,1) - 1$ and $Itr = 0$
while $Itr < MaxItr$ and not converged **do**
  for each hawk (x_i) **do**
    Evaluate $F(x_i)$ using CCV Eq. (31)
  end for
  Set x_{prey} with the minimum fitness value x_i as the best location of the rabbit
  for each hawk (x_i) **do**
    Update the initial energy EG_0 and jump strength J using Eq. (36)
    Update EG using Eq. (32)
    Set r using $rand(0,1)$
    if $|EG| \geq 1$ **then**
      Update the x_{prey} using Eq. (33)
    else
      if $|EG| \geq 0.5, r \geq 0.5$ **then**
        Update the x_{prey} using Eq. (35)
      else if $|EG| < 0.5, r \geq 0.5$ **then**
        Update the x_{prey} using Eq. (37)
      else if $|EG| \geq 0.5, r < 0.5$ **then**
        Update the x_{prey} using Eq. (40)
      else if $|EG| < 0.5, r < 0.5$ **then**
        Update the x_{prey} using Eq. (41)
      end if
    end if
  end for
  $Itr+ = 1$
end while
Return $F(x_{\text{prey}}), x_{\text{prey}}$

6. Experimental Results and Analysis

This study employs synthetic data to validate its effectiveness in three components. Many datasets from different distributions are created to evaluate various optimization methods to find the optimal bandwidth. Subsequently, the density estimation performance of ORKDE, RKDE, and KDE is assessed under identical optimization bandwidths using different probability density functions. The final classification comparison employs ORKDE, RKDE, and KDE with Naïve Bayes.

6.1. Comparative Experiments of Intelligent Optimisation Algorithms Based on CCV

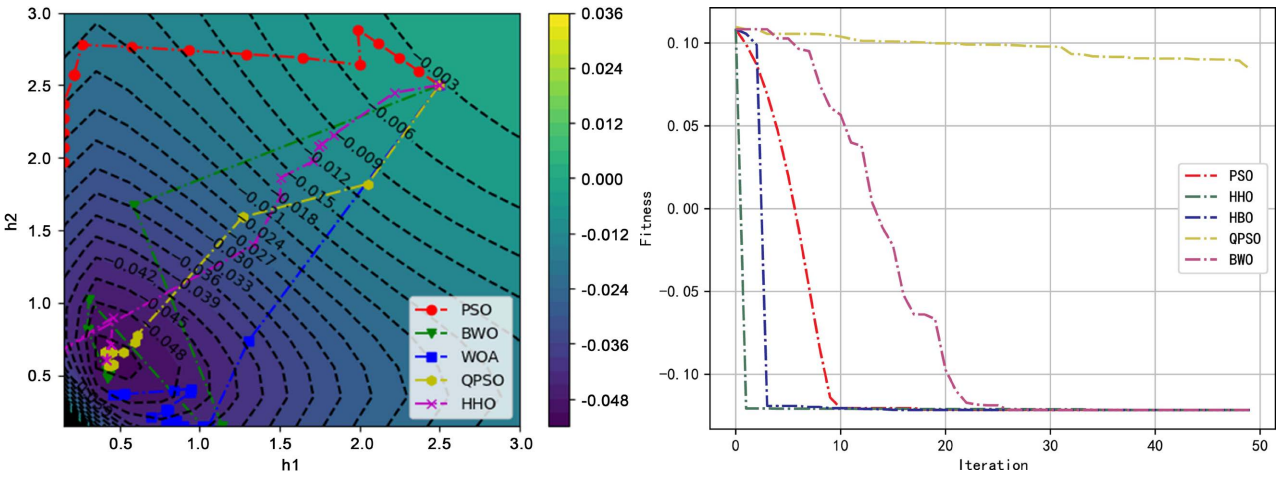
This experiment compares CCV optimization techniques. HHO started with a population size of 50 and a threshold of 0.0003. In the Quantum Particle Swarm Optimisation (QPSO) algorithm [32], a population size of 50 and a control parameter of 0.5 were used. With an initial population size of 50, the Black Widow Optimisation algorithm [33] was limited to a maximum of 50 iterations. The computation's optimization algorithms have a 100-iteration restriction, and each algorithm is run 50 times independently. **Figure 2(a)** depicts many optimization methods' search paths, showing that the optimization path of HHO is relatively optimal. **Table 1** gives the optimized pairs of values of UCV under the conditions of data generated by different distributions, and the optimization algorithm is run independently 50 times for each simulated data, with a sample size of 500, where

the Gaussian function has the parameters: [0, 0], δ: [[3, 1.2], [1.2, 3]], and the T-distribution function has the parameters L: [0, 0], with the degree of freedom DF = 5, and the values S: [[2, 0.1], [0.1, 2]]; the Cauchy distribution function with parameters L: [0, 0], S: [[2, 0.1], [0.1, 2]] and degrees of freedom $DF1$ = 1; and the Laplace distribution L: [0, 0], S: [[2, 0.1], [0.1, 2]].

Upon analyzing the 2-D CCV findings, it is evident that HHO outperforms the other four optimization methods. Figure 2(b) compares the convergence of optimization algorithms for Gaussian distributed data with two-dimensional bandwidth conditions. The results demonstrate that the HHO algorithm surpasses other algorithms in terms of convergence speed and value. The experiment surpasses the outcomes of various algorithms for HHO in terms of the schematic search path, the convergence speed of the objective function, and the final convergence value.

Table 1. CCV comparison with different optimization methods.

Distribution	PSO	BWO	WOA	QPSO	HHO
Gaussian	−6.8 ± 0.4	−6.8 ± 0.1	−6.7 ± 0.6	−6.7 ± 0.6	−6.9 ± 0.2
T-distribution	3.4 ± 0.2	2.3 ± 0.5	2.3 ± 0.5	1.1 ± 0.2	0.8 ± 0.2
Cauchy	0.8 ± 0.2	0.6 ± 0.2	0.7 ± 0.3	0.5 ± 0.2	0.4 ± 0.1
Laplace	2.1 ± 0.4	1.8 ± 0.4	1.1 ± 0.2	0.3	1.2 ± 0.3



(a) Search paths under 2-D

(b) Objective function convergence

Figure 2. Comparison of different methods.

6.2. Experiments Comparing KDE and RKDE Distribution Fitting

The HHO-based optimization bandwidth was used as the bandwidth value for all estimators in the experimental comparison of function fitting effects. The MISE metric compares the differences between the estimated results and the proper distribution function. The experiences are done for both one-dimensional and two-dimensional data. For the one-dimensional comparison experiments, 300 sample points are generated using a standard normal distribution with 0 as the mean

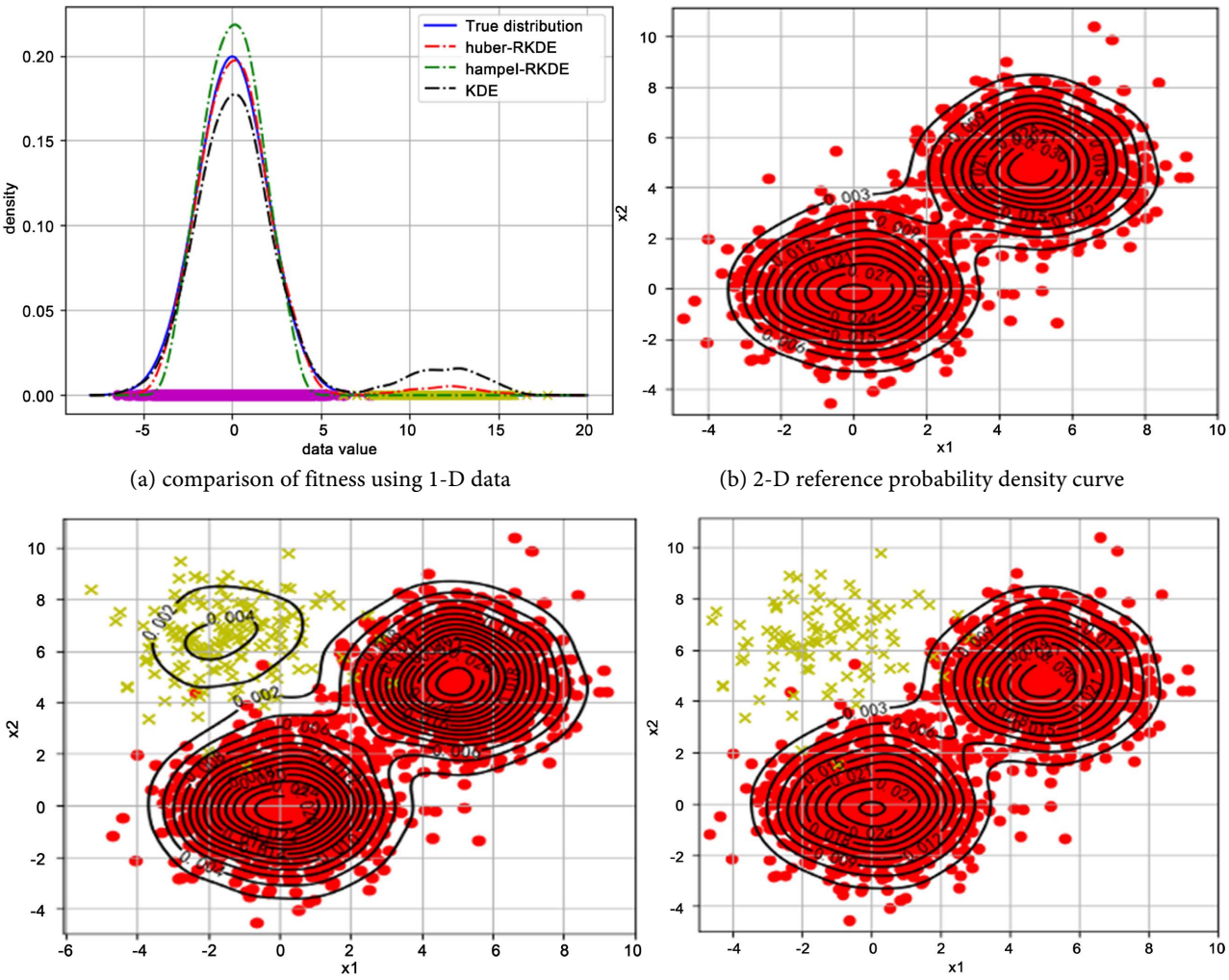
value, 0.5 as the variance value, and 40 outlier sample points using a uniform distribution in the interval from 6 to 17. In this case, γ in the Huber function is taken from the 75% quantile of the mapping after each calculation, and the parameters γ_1, γ_2 and γ_3 of the Hampel function are taken as the 25%, 50% and already 75% quantiles, respectively. Figure 3 shows that the traditional KDE has the highest density values in the outlier section, even when the bandwidth value is optimized. This result means that the main density components are shown less accurately. Compared to the Hampel function, the red curve of the RKDE with the Huber function is the closest to the original reference function curve among all the fitted curves. Although there is a non-zero density value of the outliers corresponding to the red curve for the fitting part of its outliers, the fitting of the principal components is optimal compared to the other methods. The RKDE using the Hampel loss function completely suppresses the density values of outlier points for one-dimensional data. Still, there is a significant deviation in the fitting of the principal function.

From the one-dimensional experimental results, the RKDE using the Huber function is the most effective in fitting the one-dimensional data. We made 1000 sample points with the distribution F_1, which has a Gaussian function mean of [0,0] and variance of [1, 0; 0, 1]. We also made 1000 samples with the distribution F_2, which has a Gaussian function mean of [5.5, 5.5], and variance of [1, 0; 0, 1], and 1000 samples with the distribution with a Gaussian function mean of [−2, 6.5] and variance of [2, 0; 0, 2]. The Gaussian distribution F_3 generates 100 outlier sample points. ORKDE's bandwidth values of both KDE and RKDE are from HHO-optimized bandwidth [0.34, 0.32]. Compared with Figure 3(b) of the reference function, the estimation of traditional KDE in Figure 3(c) produces estimations value in the region of the outlier points, which leads to some error in the central part, and the estimation results using ORKDE are more accurate in Figure 3(d). Table 2 gives the parameters, outlier points, and number of samples for generating 1-D and 2-D data distributions. The experimental results in Table 3 show that the MISE value of ORKDE is the smallest among all types of fitting methods.

6.3. Comparative Experiments of Bayesian Classifiers Based on Optimized RKDEs

This part of the experiment mainly compares the classifications using ORKDE-NB, RKDE-NB, KDE-NB, and common classifiers. The data in this part is simulated data, mainly used to obey several different distributions of data in different data sets, a simulation of binary classification, and a triple classification problem, which takes into account the different dimensions of the data, the dimensions in this experiment range from two-dimensional, three-dimensional, to four-dimensional. The parameter control of outliers in this classification process, $L_o = \frac{L_1 + L_2 + \cdots + L_{cn}}{cn}$, $S_o = \frac{S_1 + S_2 + \cdots + S_{cn}}{cn}$ specific parameters are shown in Table 5. Meanwhile, the ten-fold method is used for different datasets to verify

the classifier's effectiveness. The comparison of the effectiveness of each classifier for two-dimensional T-distribution is given separately in Table 4.



(a) comparison of fitness using 1-D data

(b) 2-D reference probability density curve

(c) 2-D KDE probability density estimation curves

(d) 2-D ORKDE probability density estimation curve

Figure 3. Comparative experiments of fitness.

Tables 6-9 give the results of testing the mean accuracy of the algorithms based on the simulated generation of relevant distribution experimental data in Table 5. In addition to comparing the relevant Bayesian classifiers, basic classifiers such as SVM, Decision Tree (DT), Random Forest (RF), XGboost, etc, are used. Among them, SVM adopts RBF kernel, and its parameter adopts lattice search method, $c \in [-8,8]$, $g \in [-8,8]$, with a step size of 0.1. Decision Tree adopts the *C*4.5 model combined with dichotomization to deal with continuous attributes. The parameters of the random forest were set in the experiment as criterion = "gini", *maxdepth* = None, *minsamplessplit* = 2, *minsamplesleaf* = 1, bootstrap = True. The parameters in the XGboost model were set as *learningrate* = 0.3, gamma = 2.5, *maxdepth* = 6, *minchildweight* = 3, *maxdeltastep* = 0.5; with the classification results in Tables 6-9, among the classification accuracies of various dimensions

in the binary classification, and tertiary classification, the classification accuracies of ORKDE- NB has the best classification accuracy in all cases. As the data dimensions increase, the accuracy of the same classifier decreases; this experimental result also fully demonstrates the effectiveness of the method proposed in this paper.

Table 2. Parameters for distribution function generation.

Distribution function	1-D distribution	2-D distribution
Gaussian	$\mu 1$: 0; Samples: 500; $\delta 1$: 0.5; Outliers: 20; $\mu 2$: 1; $\delta 2$: 3	$\mu 1$: [0, 0]; Samples: 500; $\delta 1$: [[3, 1.2], [1.2, 3]]; Outliers: 100; $\mu 2$: [0.5, 0.5]; $\delta 2$: [[3, 1.5], [1.5, 3]]
T-distribution	$\mu 1$: 0.5; $\delta 1$: 3; L1: 0; DF1 = 5; S1: 2.1; Samples: 500; Outliers: 20	$\mu 2$: [0.5, 0.5]; $\delta 2$: [[3, 1.5], [1.5, 3]]; L1: [0, 0]; df1 = 5; S1: [[2, 0.1], [0.1, 2]]; Samples: 500; Outliers: 20
Cauchy	$\mu 1$: 0.5; $\delta 1$: 3; L1: 0; Samples: 500; S1: 2.1; DF1 = 1; Outliers: 20	L1: [0, 0]; Samples: 500; s1: [[2, 0.1], [0.1, 2]]; DF1 = 1; Outliers: 20; $\mu 2$: [0.5, 0.5]; $\delta 2$: [[3, 1.5], [1.5, 3]]
Laplace's	L1: 0; Samples: 500; S1: 2.2; Outlier: 20; $\mu 1$: 1; $\delta 1$: 3	L1: [0, 0]; Samples: 500; s1: [[2, 0.1], [0.1, 2]]; Outliers: 20; $\mu 2$: [0.5, 0.5]; $\delta 2$: [[3, 1.5], [1.5, 3]]

Table 3. MISE comparison with different distributions.

Estimator	Gaussian	Gamma	T	Chi-square	Laplace	Gaussian	T	Laplace
			1-D				2-D	
KDE	4.76	2.80	3.36	9.33	2.35	4.27	5.56	9.15
RKDE	5.16	3.15	3.57	9.87	2.44	4.31	5.58	9.31
ORKDE	4.36	2.76	3.16	8.25	2.28	3.57	5.51	8.94

Table 4. Comparisons of two-dimensional T-distribution data.

Classifier	Precision	Recall	Accuracy
KDE-NB	0.78 ± 0.2	0.83 ± 0.2	0.81 ± 0.2
RKDE-NB	0.79 ± 0.3	0.84 ± 0.4	0.82 ± 0.4
ORKDE-NB	0.82 ± 0.1	0.85 ± 0.3	0.83 ± 0.1
SVM	0.80 ± 0.4	0.79 ± 0.2	0.78 ± 0.3
DecisionTree	0.81 ± 0.5	0.83 ± 0.7	0.80 ± 0.4
RandomForest	0.82 ± 0.2	0.84 ± 0.6	0.82 ± 0.5
XGboost	0.82 ± 0.3	0.84 ± 0.3	0.82 ± 0.2

The ORKDE method demonstrates significant advantages over traditional kernel density estimation methods. Its robustness to outliers is achieved through the combination of M-estimation and a weighted kernel density estimator, resulting in more accurate density estimates. Additionally, the adaptive bandwidth selection facilitated by CCV and HHO enhances the estimator's ability to adjust to the underlying data distribution, making it more versatile in real-world applications. In terms of applicability, the ORKDE method is designed to accommodate various data distributions, ensuring that the kernel density estimates accurately reflect the

Table 5. Simulation data set parameter description.

distributions function	Binary classifcation dataset			Triple classifcation dataset		
	2-D	3-D	4-D	2-D	3-D	4-D
Gaussian	$\mu 1$ = [0, 0] $\delta 1$ = [[3, 1.2], [1.2, 3]] $\mu 2$ = [1, 1] $\delta 2$ = [[3, 1.5], [1.5, 3]]	$\mu 1$ = [0, 0, 0] $\delta 1$ = [[3, 1.2, 0], [1.2, 3, 0], [0, 0, 3]] $\mu 2$ = [1, 1, 1] $\delta 2$ = [[3, 1.2, 0], [1.2, 3, 0], [0, 0, 3]]	$\mu 1$ = [0, 0, 0, 0] $\delta 1$ = [[3, 0, 0, 0], [0, 3, 0, 0], [0, 0, 3, 0], [0, 0, 0, 3]] $\mu 2$ = [1, 1, 1, 1] $\delta 2$ = [[3, 0, 0, 0], [0, 3, 0, 0], [0, 0, 3, 0], [0, 0, 0, 3]]	$\mu 1$ = [0, 0] $\delta 1$ = [[3, 1.2], [1.2, 3]] $\mu 2$ = [1, 1] $\delta 2$ = [[3, 1.5], [1.5, 3]] $\mu 3$ = [1.5, 1.5] $\delta 3$ = [[3, 1.5], [1.5, 3]]	$\mu 1$ = [0, 0, 0] $\delta 1$ = [[3, 0, 0], [0, 3, 0], [0, 0, 3]] $\mu 2$ = [1, 1, 1] $\delta 2$ = [[3, 0, 0], [0, 3, 0], [0, 0, 3]] $\mu 3$ = [1.5, 1.5, 1.5] $\delta 3$ = [[3, 1, 0], [0, 3, 1], [0, 1, 3]]	$\mu 1$ = [0, 0, 0, 0], $\delta 1$ = [[3, 0, 0, 0], [0, 3, 0, 0], [0, 0, 3, 0], [0, 0, 0, 3]] $\mu 2$ = [1, 1, 1, 1] $\delta 2$ = [[3, 0, 0, 0], [0, 3, 0, 0], [0, 0, 3, 0], [0, 0, 0, 3]] $\mu 3$ = [1.5, 1.5, 1.5, 1.5] $\delta 3$ = [[3, 0, 0, 0], [0, 3, 0, 0], [0, 0, 3, 0], [0, 0, 0, 3]]
T-distribution	L1 = [0, 0] DF1 = 5 S1 = [[2, 0.1], [0.1, 2]] L2 = [2, 2] DF2 = 5 S2 = [[2, 0.1], [0.1, 2]]	L1 = [0, 0, 0] DF1 = 5 S1 = [[1, 0.1, 0], [0.1, 1, 0], [0, 0.1, 1]] L2 = [2, 2, 2] DF2 = 5 S2 = [[1, 0.1, 0], [0.1, 1, 0], [0, 0.1, 1]]	L1 = [0, 0, 0, 0] DF1 = 5 S1 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]] L2 = [2, 2, 2, 2] DF2 = 5 S2 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]	L1 = [0, 0] DF1 = 5 S1 = [[2, 0.1], [0.1, 2]] L2 = [2, 2] DF2 = 5 S2 = [[2, 0.1], [0.1, 2]] L3 = [1, 1] DF3 = 5 S3 = [[2, 0.1], [0.1, 2]]	L1 = [0, 0, 0] DF1 = 5 S1 = [[2, 0.1, 0], [2, 0.1, 0], [0.1, 0, 2]] L2 = [2, 2, 2] DF2 = 5 S2 = [[2, 0.1, 0], [2, 0.1, 0], [0.1, 0, 2]] L3 = [1, 1, 1] DF3 = 5 S3 = [[2, 0.1, 0], [0, 2, 0.1], [0.1, 0, 2]]	L1 = [0, 0, 0, 0] DF1 = 5 S1 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]] L2 = [2, 2, 2, 2] DF2 = 5 S2 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]] L3 = [1, 1, 1, 1] DF3 = 5 S3 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]]
Cauchy	L1 = [0, 0] DF1 = 1 S1 = [[2, 0.1], [0.1, 2]] L2 = [2, 2] DF2 = 1 S2 = [[2, 0.1], [0.1, 2]]	L1 = [0, 0, 0] DF1 = 1 S1 = [[1, 0.1, 0], [0.1, 1, 0], [0, 0.1, 1]] L2 = [2, 2, 2] DF2 = 1 S2 = [[1, 0.1, 0], [0.1, 1, 0], [0, 0.1, 1]]	L1 = [0, 0, 0, 0] DF1 = 1 S1 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]] L2 = [2, 2, 2, 2] DF2 = 1 S2 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]	L1 = [0, 0] DF1 = 1 S1 = [[2, 0.1], [0.1, 2]] L2 = [2, 2] DF2 = 1 S2 = [[2, 0.1], [0.1, 2]] L3 = [1, 1] DF3 = 5 S3 = [[2, 0.1], [0.1, 2]]	L1 = [0, 0, 0] DF1 = 1 S1 = [[2, 0.1, 0], [2, 0.1, 0], [0.1, 0, 2]] L2 = [2, 2, 2] DF2 = 1 S2 = [[2, 0.1, 0], [2, 0.1, 0], [0.1, 0, 2]] L3 = [1, 1, 1] DF3 = 1 S3 = [[2, 0.1, 0], [0, 2, 0.1], [0.1, 0, 2]]	L1 = [0, 0, 0, 0] DF1 = 5 S1 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]] L2 = [2, 2, 2, 2] DF2 = 2 S2 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]] L3 = [1, 1, 1, 1] DF3 = 3 S3 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]]
Laplace	L1 = [0, 0] DF1 = 1 S1 = [[2, 0.1], [0.1, 2]] L2 = [2, 2] DF2 = 1 S2 = [[2, 0.1], [0.1, 2]]	L1 = [0, 0, 0] DF1 = 1 S1 = [[1, 0.1, 0], [0.1, 1, 0], [0, 0.1, 1]] L2 = [2, 2, 2] DF2 = 1 S2 = [[1, 0.1, 0], [0.1, 1, 0], [0, 0.1, 1]]	L1 = [0, 0, 0, 0] DF1 = 1 S1 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]] L2 = [2, 2, 2, 2] DF2 = 1 S2 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]	L1 = [0, 0, 0, 0] DF1 = 1 S1 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]] L2 = [2, 2, 2, 2] DF2 = 1 S2 = [[1, 0.1, 0, 0], [0.1, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]	L1 = [0, 0, 0] DF1 = 1 S1 = [[2, 0.1, 0], [2, 0.1, 0], [0.1, 0, 2]] L2 = [2, 2, 2] DF2 = 1 S2 = [[2, 0.1, 0], [2, 0.1, 0], [0.1, 0, 2]] L3 = [1, 1, 1] DF3 = 1 S3 = [[2, 0.1, 0], [0, 2, 0.1], [0.1, 0,2]]	L1 = [0, 0, 0, 0] DF1 = 5 S1 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]] L2 = [2, 2, 2, 2] DF2 = 2 S2 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1], [0, 0, 0.1, 2]] L3 = [1, 1, 1, 1] DF3 = 3 S3 = [[2, 0.1, 0, 0], [0, 2, 0.1, 0], [0, 0, 2, 0.1],

Table 6. Comparisons with Gaussian multidimensional data.

Classifier	Accuracy					
	Binary classification			triple classification		
	2-D	3-D	4-D	2-D	3-D	4-D
KDE-NB	0.81	0.79	0.73	0.89	0.81	0.78
RKDE-NB	0.82	0.78	0.72	0.88	0.80	0.77
ORKDE-NB	0.82	0.83	0.76	0.90	0.86	0.79
SVM	0.78	0.78	0.74	0.80	0.83	0.75
DT	0.80	0.76	0.71	0.83	0.82	0.76
RF	0.82	0.79	0.75	0.84	0.82	0.77
XGboost	0.82	0.81	0.75	0.87	0.81	0.78

Table 7. Comparisons with T multidimensional data.

Classifier	Accuracy					
	binary classification			triple classification		
	2-D	3-D	4-D	2-D	3-D	4-D
KDE-NB	0.81	0.79	0.78	0.71	0.71	0.70
RKDE-NB	0.82	0.78	0.77	0.73	0.80	0.72
ORKDE-NB	0.83	0.83	0.76	0.75	0.82	0.76
SVM	0.78	0.78	0.73	0.73	0.82	0.71
DT	0.80	0.76	0.76	0.74	0.81	0.70
RF	0.82	0.75	0.75	0.75	0.80	0.73
XGboos	0.82	0.82	0.77	0.75	0.81	0.72

Table 8. Comparisons with Cauchy distribution data.

Classifier	Accuracy					
	binary classification			triple classification		
	2-D	3-D	4-D	2-D	3-D	4-D
KDE-NB	0.84	0.78	0.71	0.73	0.80	0.72
RKDE-NB	0.86	0.74	0.72	0.74	0.82	0.73
ORKDE-NB	0.87	0.77	0.73	0.76	0.83	0.75
SVM	0.79	0.72	0.76	0.77	0.84	0.71
DT	0.78	0.73	0.73	0.78	0.76	0.73
RF	0.76	0.74	0.72	0.73	0.78	0.74
XGboos	0.77	0.79	0.76	0.71	0.83	0.73

Table 9. Comparisons with Laplacian distribution data.

Classifier	Accuracy					
	binary classification			triple classification		
	2-D	3-D	4-D	2-D	3-D	4-D
KDE-NB	0.87	0.81	0.74	0.83	0.91	0.79
RKDE-NB	0.88	0.82	0.73	0.82	0.92	0.80
ORKDE-NB	0.89	0.83	0.76	0.84	0.93	0.82
SVM	0.83	0.80	0.71	0.80	0.80	0.79
DT	0.81	0.78	0.75	0.79	0.83	0.78
RF	0.80	0.80	0.75	0.84	0.85	0.77
XGboos	0.84	0.82	0.78	0.82	0.88	0.76

actual underlying structure of the data. This adaptability is essential in scenarios where datasets may contain varying degrees of contamination from outliers, allowing the method to maintain high performance across diverse conditions. Overall, the ORKDE method represents a significant advancement in robust statistical estimation techniques, providing a reliable framework for improving classification accuracy in Bayesian classifiers, especially when dealing with continuous datasets that exhibit complexities such as noise and outliers. The integration of M-estimation, CCV, and HHO not only enhances the accuracy of probability density estimation but also contributes to the overall reliability of classification decisions, making it a valuable tool in the field of machine learning and data analysis.

7. Conclusions

This study introduces a Bayesian classifier based on ORKDE that offers substantial benefits, including enhanced robustness to outliers, improved classification accuracy, and adaptive bandwidth selection. These features collectively contribute to the method's effectiveness in handling complex datasets characterized by noise and outliers. However, certain limitations persist, particularly concerning the computational complexity associated with the HHO algorithm and the necessity for meticulous tuning of hyperparameters. Future research directions may encompass several avenues:

- Exploration of alternative optimization algorithms: Investigating other optimization techniques that could provide improved convergence rates or enhanced computational efficiency, thereby broadening the applicability of the ORKDE method.
- Extension of applicability: Conducting empirical evaluations of the method across a diverse array of datasets and application domains to further substantiate its robustness and effectiveness in varied contexts.
- Integration with deep learning approaches: Merging the ORKDE method with deep learning frameworks to capitalize on their capabilities in feature extraction and representation learning. This integration has the potential to further

enhance classification performance, particularly in complex scenarios where traditional methods may falter.

These proposed enhancements and explorations will significantly contribute to the ongoing development of robust classification methodologies, particularly in environments where data is prone to contamination from outliers and noise. The ORKDE method stands as a promising advancement in the field, offering a reliable framework for improving Bayesian classification accuracy and fostering further innovations in machine learning and data analysis.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Shu, X. and Ye, Y. (2023) Knowledge Discovery: Methods from Data Mining and Machine Learning. *Social Science Research*, **110**, Article 102817. https://doi.org/10.1016/j.ssresearch.2022.102817

[2] Su, S., Xiao, L., Ruan, L., Gu, F., Li, S., Wang, Z., *et al.* (2019) An Efficient Density-Based Local Outlier Detection Approach for Scattered Data. *IEEE Access*, **7**, 1006-1020. https://doi.org/10.1109/access.2018.2886197

[3] Wang, T., Li, Q., Chen, B. and Li, Z. (2017) Multiple Outliers Detection in Sparse High-Dimensional Regression. *Journal of Statistical Computation and Simulation*, **88**, 89-107. https://doi.org/10.1080/00949655.2017.1379521

[4] Guo, W., Xu, P., Dai, F., Zhao, F. and Wu, M. (2021) Improved Harris Hawks Optimization Algorithm Based on Random Unscented Sigma Point Mutation Strategy. *Applied Soft Computing*, **113**, Article 108012. https://doi.org/10.1016/j.asoc.2021.108012

[5] Zhang, X., Mallick, H., Tang, Z., Zhang, L., Cui, X., Benson, A.K., *et al.* (2017) Negative Binomial Mixed Models for Analyzing Microbiome Count Data. *BMC Bioinformatics*, **18**, Article No. 4. https://doi.org/10.1186/s12859-016-1441-7

[6] Lee, K.H. and Kim, M.H. (2022) Bayesian Inductive Learning in Group Recommendations for Seen and Unseen Groups. *Information Sciences*, **610**, 725-745. https://doi.org/10.1016/j.ins.2022.08.010

[7] Wang, Q. (2020) Multivariate Kernel Smoothing and Its Applications. *Journal of the American Statistical Association*, **115**, 486-486. https://doi.org/10.1080/01621459.2020.1721247

[8] Aggarwal, C.C. and Yu, P.S. (2008) Outlier Detection with Uncertain Data. *Proceedings of the* 2008 *SIAM International Conference on Data Mining*, Atlanta, 24-26 April 2008, 483-493. https://doi.org/10.1137/1.9781611972788.44

[9] Cao, K., Shi, L., Wang, G., Han, D. and Bai, M. (2014) Density-Based Local Outlier Detection on Uncertain Data. *Web-Age Information Management*, Macau, 16-18 June 2014, 67-71. https://doi.org/10.1007/978-3-319-08010-9_9

[10] Scott, D.W. (2015) Multivariate Density Estimation: Theory, Practice, and Visualization. Wiley. https://doi.org/10.1002/9781118575574

[11] Knuth, K.H. (2019) Optimal Data-Based Binning for Histograms and Histogram-Based Probability Density Models. *Digital Signal Processing*, **95**, Article 102581. https://doi.org/10.1016/j.dsp.2019.102581

[12] Kamalov, F. (2020) Kernel Density Estimation Based Sampling for Imbalanced Class Distribution. *Information Sciences*, **512**, 1192-1201. https://doi.org/10.1016/j.ins.2019.10.017

[13] Kim, J. and Scott, C.D. (2012) Robust Kernel Density Estimation. *The Journal of Machine Learning Research*, **13**, 2529-2565.

[14] Ou, G., He, Y., Fournier-Viger, P. and Huang, J.Z. (2022) A Novel Mixed-Attribute Fusion-Based Naive Bayesian Classifier. *Applied Sciences*, **12**, Article 10443. https://doi.org/10.3390/app122010443

[15] Yang, F. (2018) An Implementation of Naive Bayes Classifier. 2018 *International Conference on Computational Science and Computational Intelligence* (*CSCI*), Las Vegas, 12-14 December 2018, 301-306. https://doi.org/10.1109/csci46756.2018.00065

[16] Bertsimas, D. and Koduri, N. (2022) Data-Driven Optimization: A Reproducing Kernel Hilbert Space Approach. *Operations Research*, **70**, 454-471. https://doi.org/10.1287/opre.2020.2069

[17] Wang, S., Li, A., Wen, K. and Wu, X. (2020) Robust Kernels for Kernel Density Estimation. *Economics Letters*, **191**, Article 109138. https://doi.org/10.1016/j.econlet.2020.109138

[18] Vandermeulen, R.A. and Scott, C. (2014) Robust Kernel Density Estimation by Scaling and Projection in Hilbert Space. *Proceedings of the 27th International Conference on Neural Information Processing System*, Montreal, 8-13 December 2014, 1-8.

[19] López-Rubio, E., Palomo, E.J. and Domínguez, E. (2015) Robust Self-Organization with M-Estimators. *Neurocomputing*, **151**, 408-423. https://doi.org/10.1016/j.neucom.2014.09.024

[20] Silverman, B.W. (1986) Density Estimation for Statistics and Data Analysis. Chapman and Hall, 1-175.

[21] Duong, T. (2022) Bandwidth Selection for Kernel Density Estimation: A Review of Fully Automatic Selectors. *Advances in Statistical Analysis*, **35**, 159-188.

[22] He, Y., Ye, X., Huang, D., Huang, J.Z. and Zhai, J. (2021) Novel Kernel Density Estimator Based on Ensemble Unbiased Cross-Validation. *Information Sciences*, **581**, 327-344. https://doi.org/10.1016/j.ins.2021.09.045

[23] Sain, S.R., Baggerly, K.A. and Scott, D.W. (1994) Cross-Validation of Multivariate Densities. *Journal of the American Statistical Association*, **89**, 807-817. https://doi.org/10.1080/01621459.1994.10476814

[24] Hall, P. (1984) Central Limit Theorem for Integrated Square Error of Multivariate Nonparametric Density Estimators. *Journal of Multivariate Analysis*, **14**, 1-16. https://doi.org/10.1016/0047-259x(84)90044-7

[25] Ghorai, J.K. and Pattanaik, L.M. (1991) A Central Limit Theorem for the Weighted Integrated Squared Error of the Kernel Type Density Estimator under the Proportional Hazard Model. *Journal of Nonparametric Statistics*, **1**, 111-126. https://doi.org/10.1080/10485259108832514

[26] Miecznikowski, J.C., Wang, D. and Hutson, A. (2010) Bootstrap MISE Estimators to Obtain Bandwidth for Kernel Density Estimation. *Communications in Statistics-Simulation and Computation*, **39**, 1455-1469. https://doi.org/10.1080/03610918.2010.500108

[27] Taylor, C.C. (1989) Bootstrap Choice of the Smoothing Parameter in Kernel Density Estimation. *Biometrika*, **76**, 705-712. https://doi.org/10.1093/biomet/76.4.705

[28] Mojirsheibani, M. (2021) A Note on the Performance of Bootstrap Kernel Density Estimation with Small Re-Sample Sizes. *Statistics & Probability Letters*, **178**, Article

109189. https://doi.org/10.1016/j.spl.2021.109189

[29] Chen, X., Fu, M., Liu, Z., Jia, C. and Liu, Y. (2022) Harris Hawks Optimization Algorithm and BP Neural Network for Ultra-Wideband Indoor Positioning. *Mathematical Biosciences and Engineering*, **19**, 9098-9124. https://doi.org/10.3934/mbe.2022423

[30] Chen, L., Song, N. and Ma, Y. (2022) Harris Hawks Optimization Based on Global Cross-Variation and Tent Mapping. *The Journal of Supercomputing*, **79**, 5576-5614. https://doi.org/10.1007/s11227-022-04869-7

[31] Shehab, M., Mashal, I., Momani, Z., Shambour, M.K.Y., AL-Badareen, A., Al-Dabet, S., *et al.* (2022) Harris Hawks Optimization Algorithm: Variants and Applications. *Archives of Computational Methods in Engineering*, **29**, 5579-5603. https://doi.org/10.1007/s11831-022-09780-1

[32] Li, X., Fang, W. and Zhu, S. (2023) An Improved Binary Quantum-Behaved Particle Swarm Optimization Algorithm for Knapsack Problems. *Information Sciences*, **648**, Article 119529. https://doi.org/10.1016/j.ins.2023.119529

[33] Hu, G., Du, B., Wang, X. and Wei, G. (2022) An Enhanced Black Widow Optimization Algorithm for Feature Selection. *Knowledge-Based Systems*, **235**, Article 107638. https://doi.org/10.1016/j.knosys.2021.107638