mathematics

MDPI

Article

ASIDS: A Robust Data Synthesis Method for Generating Optimal Synthetic Samples

Yukun Du †, Yitao Cai †, Xiao Jin †, Hongxia Wang *, Yao Li and Min Lu

School of Statistics and Data Science, Nanjing Audit University, Nanjing 211815, China; mg2108129@stu.nau.edu.cn (Y.D.); ma2208133@stu.nau.edu.cn (Y.C.); mg2108123@stu.nau.edu.cn (X.J.); 223080612@stu.nau.edu.cn (Y.L.); lumin@nau.edu.cn (M.L.)

* Correspondence: hxwang@nau.edu.cn

† These authors contributed equally to this work.

Abstract: Most existing data synthesis methods are designed to tackle problems with dataset imbalance, data anonymization, and an insufficient sample size. There is a lack of effective synthesis methods in cases where the actual datasets have a limited number of data points but a large number of features and unknown noise. Thus, in this paper we propose a data synthesis method named Adaptive Subspace Interpolation for Data Synthesis (ASIDS). The idea is to divide the original data feature space into several subspaces with an equal number of data points, and then perform interpolation on the data points in the adjacent subspaces. This method can adaptively adjust the sample size of the synthetic dataset that contains unknown noise, and the generated sample data typically contain minimal errors. Moreover, it adjusts the feature composition of the data points, which can significantly reduce the proportion of the data points with large fitting errors. Furthermore, the hyperparameters of this method have an intuitive interpretation and usually require little calibration. Analysis results obtained using simulated original data and benchmark original datasets demonstrate that ASIDS is a robust and stable method for data synthesis.

Keywords: data synthesis; unknown noise; interpolation; sample optimization; robust and stable

MSC: 62-11; 68T09

Citation: Du, Y.; Cai, Y.; Jin, X.; Wang, H.; Li, Y.; Lu, M. ASIDS: A Robust Data Synthesis Method for Generating Optimal Synthetic Samples. *Mathematics* **2023**, *11*, 3891. https://doi.org/10.3390/math11183891

Academic Editors: Su Chen and Michael Pokojovy

Received: 16 August 2023
Revised: 9 September 2023
Accepted: 11 September 2023
Published: 13 September 2023

1. Introduction

Synthetic data present an effective solution to the challenges of inadequate or low-quality samples, particularly in the era of big data. The use of data synthesis methods to generate synthetic data provides a cost-effective and efficient alternative to collecting and labeling large amounts of real-world data. Furthermore, these methods can address privacy concerns associated with real-world data, making it safer to share and analyze [1]. Recently, there has been a surge in the use of synthetic data in machine learning, with various synthetic methods being developed [2,3].

Representative data synthesis methods can be classified into three categories. The first category entails techniques such as interpolation, extrapolation, and other methods [4] that generate additional data points representative of the underlying distribution. These data synthesis methods can extract more information from the dataset, thereby enhancing model generalization. For image data, deep-learning-based methods such as VAE and GAN can also be used to generate new data [5,6]. These methods are useful for improving the quality of the dataset and model generalization. The second category is meant to address the issue of dataset imbalance, such as with SMOTE [7] and some of its enhanced versions [8–10]. These methods can synthesize minority class samples to balance the dataset and improve the model's performance. Finally, data sharing and research require sensitive data privacy protection. Synthetic data can help protect sensitive information while still enabling data sharing and research. Normally, we can add random noise to protect original data, like

in differential privacy methods [11], generating synthetic data for sharing and research purposes. Overall, these methods help optimize the representation and use of data in various applications.

For some actual tasks, original data often contain a multitude of features and unknown noise [12]. Since the synthetic method involves generating new data based on existing data, the quality of the new points depends on the quality and quantity of the original data. If the quality of the original data is poor or the quantity is insufficient, then the synthetic data may have limitations in terms of quality [13]. However, there is a lack of effective synthetic methods for datasets with a restricted size and complex noise that can expand the size of the dataset.

Based on the purpose of improving the quality and quantity of the dataset, and motivated by piecewise linear interpolation and spline interpolation, we propose a robust and stable data synthesis method named Adaptive Subspace Interpolation for Data Synthesis (ASIDS), which aims to adaptively adjust the sample size and structure of the original dataset containing unknown noise. The idea is to divide the original feature space into several subspaces with an equal number of samples, and then perform linear interpolation for the samples in the adjacent subspaces. This method achieves sample optimization via two aspects. First, it can adaptively adjust the size of the dataset, and the expanded data typically contain minimal errors. Second, it adjusts the structure of the samples, which can significantly reduce the proportion of samples with large errors, thereby minimizing the impact of the noise in model generalization. Compared with other methods, ASIDS is particularly suitable for data containing unknown noise. Its main purpose is to expand the sample size and optimize the sample structure to uncover more hidden information in the data. Also, the samples synthesized by ASIDS tend to have smaller errors.

The rest of this paper is organized as follows: The existing interpolation research is reviewed in Section 2. Section 3 details the concept of the proposed ASIDS method and provides proof of the effectiveness of this method. The experimental results are presented and analyzed in Section 4. Finally, we conclude this paper in Section 5.

2. Related Work

Traditional interpolation methods are based on the function values of known data points for extrapolation and prediction. For a given dataset, there are generally two cases. In the case of two known data points and interpolation in between, the interpolation method can be selected based on distance, such as in nearest neighbor interpolation [14]. If it is assumed that the unknown point between these two data points is consistent with the straight line between them, a linear function, such as linear interpolation or piecewise linear interpolation [15,16], can be used to fit and interpolate. For interpolation among multiple given points, a commonly used method is to predict the value of the unknown point by constructing a high-degree polynomial based on the given points, such as is performed with Lagrange interpolation or Newton interpolation [17,18]. However, with an increase in the number of data points and the degree of the polynomial, there is a risk of overfitting and numerical instability (Runge's phenomenon) [19]. Another solution is to construct a global smooth function by fitting a low-degree polynomial in a local region. In comparison to high-degree polynomial interpolation methods, this method has better smoothness and numerical stability, such as is seen in spline interpolation [20]. Nevertheless, as it is necessary to fit multiple local low-degree polynomials, this method can lead to relatively high computational complexity.

When expanding the sample size using interpolation methods, the selection of node positions and quantities has a significant impact on the accuracy and stability of the interpolation results. Typically, equidistant nodes [21] are used for interpolation position selection, which means the nodes are equally spaced within the interpolation interval. Chebyshev nodes [22] are selected within the interpolation interval to satisfy certain conditions for better fitting of the function. In addition, the choice of the number of interpolations can lead to instability of the data and insufficient validation accuracy. One way to determine

the appropriate number of interpolations is by comparing the fitting degree of the model trained under different interpolation numbers with the original data [23], but this may lead to overfitting and the model having a poor generalization ability.

3. Proposed Method

In this section, we will explain the proposed ASIDS method in detail. ASIDS consists of two algorithms: K-Space and K-Match. To clearly illustrate the proposed method, we will first provide an overview of ASIDS and then introduce the specific algorithms involved.

3.1. Overview

ASIDS is mainly based on linear interpolation to increase the size and improve the quality of the original dataset. The idea is to divide the original feature space into several subspaces with an equal number of samples, and then perform linear interpolation for the samples in the adjacent subspaces. This method requires two hyperparameters (k and η) in advance. Parameter k is the number of samples existing in each feature subspace, while η is the number of equidistant nodes interpolated per unit distance in the linear interpolation of the samples. This proposed method is illustrated in Figure 1.

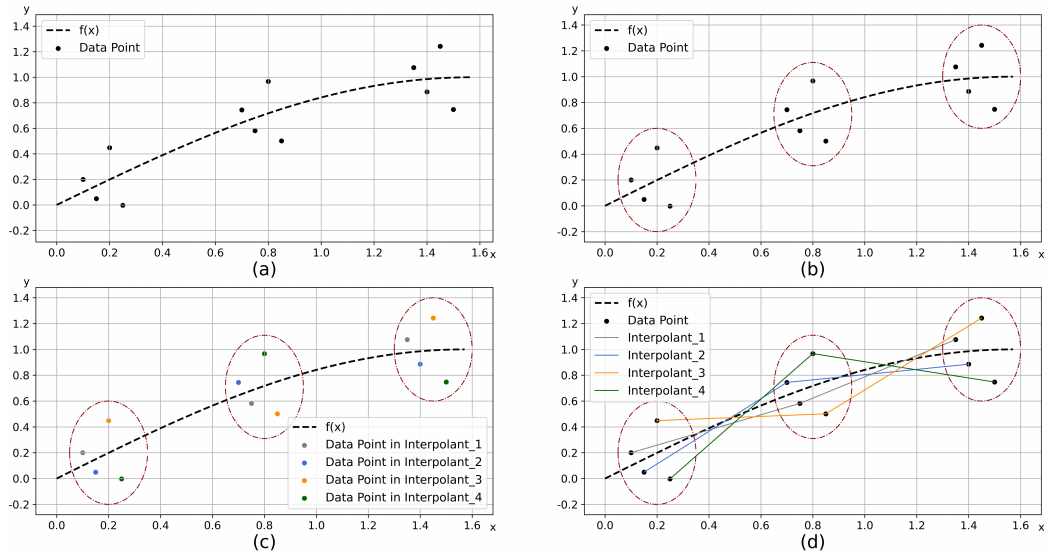

Figure 1. Overview of the proposed approach for ASIDS. (**a**) The dataset contains multiple noisy data points, and the true functional relationship between x and y is f(·). (**b**) In the first step, we use the K-Space algorithm to perform unsupervised clustering on the dataset and divide the original feature space into several subspaces. (**c**) Interpolation matching of data points between adjacent subspaces is performed using the K-Match algorithm, and data points with the same color belong to the same class to be interpolated. (**d**) Piecewise linear interpolation is performed on data points under different classes, where equidistant data points are inserted onto lines of different colors in adjacent subspaces.

The dataset $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^{n}$ is given and is assumed to be contaminated with unknown noise, where $\boldsymbol{x}_i \in \mathcal{X} = R^p$ and $y_i \in \mathcal{Y} = R$. Assuming $\dot{y}_i = f(\dot{\boldsymbol{x}}_i)$, where $\dot{\boldsymbol{x}}_i$ is the actual value of $\boldsymbol{x}_i$, $\dot{y}_i$ is the actual value of y_i, and $f(\cdot)$ is a continuous function, then it can be taken to represent the real relationship between $\boldsymbol{x}_i$ and y. Consider the model:

$$\dot{y}_i + \epsilon_{i,y} = f(\dot{\boldsymbol{x}}_i + \boldsymbol{\epsilon}_{i,x}) + \epsilon_i, \tag{1}$$

where $\epsilon_{i,y}$ is the noise in $\dot{y}_i$, $\boldsymbol{\epsilon}_{i,x}$ is the noise in $\dot{\boldsymbol{x}}_i$, and ϵ_i represents the error term. Expression (1) can be rewritten as

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i. \tag{2}$$

Let $\boldsymbol{x}_0 = \{x_0^1, \ldots, x_0^p\}$, where $x_0^j = inf\{x_i^j\}_{i=1}^n$ for $\forall j = 1, \ldots, p$, and call $\boldsymbol{x}_0$ the sample minimum point.

Given the hyperparameter k, we provide an unsupervised clustering method called K-Space. As is shown in Figure 1b, the space can be partitioned into n/k subspaces, each containing k samples; i.e., $\mathcal{X} = \cup_{s=1}^{n/k}\mathcal{X}_s$, $\mathcal{X}_i \cap \mathcal{X}_j = \emptyset, i, j = 1, 2, \ldots, \frac{n}{k}$, and $i \neq j$. The datasets corresponding to different subspaces are defined with $D = \cup_{s=1}^{n/k} D_s$, $D_s = \{(\boldsymbol{x}_i^s, y_i^s)\}_{i=1}^k$, and $\boldsymbol{x}_i^s \in \mathcal{X}_s$. For two adjacent subspaces, since $f(\cdot)$ is a continuous function, we assume that it can be approximated as a linear function $g(\cdot)$, and then Equation (2) can be transformed into

$$y_i = g(\boldsymbol{x}_i) + \epsilon_i + \epsilon_i', \tag{3}$$

where ϵ_i' is the linear fitting error term. When the distance between two adjacent subspaces approaches zero and the measurements of the subspaces tend to be zero, then $\epsilon_i' \to 0$ is obtained. Next, we will perform sample interpolation between adjacent subspaces.

We need to calculate the centers of each cluster, as follows:

$$\overline{\boldsymbol{x}}^s = \frac{1}{k} \sum_{\boldsymbol{x}_i^s \in D_s} \boldsymbol{x}_i^s. \tag{4}$$

To achieve $\epsilon_i' \to 0$, we need to ensure that the interpolation is performed between clusters that are as close in distance as possible. For $\{D_s\}_{s=1}^{n/k}$, we define $D_{(1)}$, whose cluster center has the minimum distance to the sample minimum point $\boldsymbol{x}_0$, and define $D_{(d)}$, whose cluster center has the minimum distance to the center of $D_{(d-1)}$, $D_{(d-1)} \neq D_{(1)}, \ldots, D_{(d-1)}$, and $d > 1$.

$$D_{(1)} = \underset{D_s \in D}{argmin}\, dist(\boldsymbol{x}_0, \overline{\boldsymbol{x}}^s), \tag{5}$$

$$D_{(d)} = \underset{\{D_s \in D\}, D_s \neq D_{(1)}, \ldots, D_{(d-1)}}{argmin} dist(\overline{\boldsymbol{x}}_0^{(d-1)}, \overline{\boldsymbol{x}}^s), \tag{6}$$

where $\overline{\boldsymbol{x}}^{(d-1)}$ is the center of $D_{(d-1)}$. Interpolation is performed on $\{D_{(d)}\}_{d=1}^{n/k}$ sequentially according to the order of the d values, and interpolation is carried out only between adjacent subspaces (i.e., interpolate between $D_{(1)}$ and $D_{(2)}$, between $D_{(2)}$ and $D_{(3)}$, and so on).

When performing linear interpolation between adjacent subspaces, we should pair the k samples from the first subspace with an equal number of samples from the second subspace. The interpolation rules between adjacent subspaces are as follows:

1. Linear interpolation can only be performed between two samples belonging to different adjacent subspace sets.
2. Interpolation must be performed for each sample.
3. Participation of each sample point is restricted to a single interpolation instance.

The number of matching schemes is $k!$. As is shown in Figure 1c, we provide a matching method called K-Match. Supposing $\epsilon_i' \to 0$, then this method can select a good-performing matching scheme of $\{(\boldsymbol{x}_i^{(d)}, y_i^{(d)}), (\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)})\}_{i=1}^k$ from $k!$.

Assuming $\boldsymbol{x}$ and y are continuous variables, and given another hyperparameter η, then the number of samples inserted using the linear interpolation method between $D_{(d)}$ and $D_{(d+1)}$ is $\sum_{i=1}^{k} \lfloor \eta \cdot dist(\boldsymbol{x}_i^{(d)}, \boldsymbol{x}_i^{(d+1)}) \rfloor$. Taking $(\boldsymbol{x}_i^{(d)}, y_i^{(d)}) \in D_{(d)}$ and $(\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)}) \in D_{(d+1)}$ as an example, then $\{(\boldsymbol{x}_{(d,d+1)}^{(m,i)}, y_{(d,d+1)}^{(m,i)})\}_{m=1}^{\lfloor \eta \cdot dist(\boldsymbol{x}_i^{(d)}, \boldsymbol{x}_i^{(d+1)}) \rfloor}$ is the set of inserted samples, and the linear interpolation formula is defined as

$$\boldsymbol{x}_{(d,d+1)}^{(m,i)} = \boldsymbol{x}_i^{(d)} + m \cdot \frac{\boldsymbol{x}_i^{(d+1)} - \boldsymbol{x}_i^{(d)}}{\lfloor \eta \cdot dist(\boldsymbol{x}_i^{(d)}, \boldsymbol{x}_i^{(d+1)}) \rfloor + 1}, \tag{7}$$

$$y_{(d,d+1)}^{(m,i)} = y_i^{(d)} + m \cdot \frac{y_i^{(d+1)} - y_i^{(d)}}{\lfloor \eta \cdot dist(y_i^{(d)}, y_i^{(d+1)}) \rfloor + 1}. \tag{8}$$

After ASIDS processing, the original dataset will be optimized. The main steps of the ASIDS algorithm are summarized in Algorithm 1.

Algorithm 1: ASIDS.

Input: Dataset $D = \{(x_i, y_i\}_{i=1}^{n}$; hyperparameters k and η
Output: Optimized dataset D'

Perform unsupervised clustering using K-Space to obtain $D = \{D_s\}_{s=1}^{n/k}$, where $D_s = \{x_i^s, y_i^s\}_{i=1}^{k}$
Calculate cluster centers of each cluster: $\{\overline{x}^s\}_{s=1}^{n/k}$
Obtain $\{D_{(d)}\}_{d=1^{n/k}}$ based on Equations (5) and (6)
for $d = 1, \ldots, \frac{n}{k} - 1$ **do**
    Match samples between $D_{(d)}$ and $D_{(d+1)}$ using K-Match to obtain $\{(x_i^{(d)}, y_i^{(d)}), (x_i^{(d+1)}, y_i^{(d+1)})\}$
    for $i = 1, \ldots, k$ **do**
        Synthesize new samples $\{(x_{(d,d+1)}^{(m,i)}, y_{(d,d+1)}^{(m,i)})\}_{m=1}^{\lfloor \eta \cdot dist(x_i^{(d)}, x_i^{(d+1)}) \rfloor}$ using Equations (7) and (8)
        $D' \leftarrow \{(x_{(d,d+1)}^{(m,i)}, y_{(d,d+1)}^{(m,i)})\}_{m=1}^{\lfloor \eta \cdot dist(x_i^{(d)}, x_i^{(d+1)}) \rfloor}$
    end
end

The assumptions of ASIDS are as follows:

1. $f(\cdot)$ is a continuous function.
2. The linear fitting error is $\epsilon_i' \to 0$.
3. x and y are continuous variables.

3.2. K-Space

The implementation of ASIDS requires an unsupervised clustering method to partition the feature space into multiple subspaces, each containing k samples. Based on this, we propose the K-Space clustering method. The clustering method has the following performance parameters:

1. Each subspace contains an equal number of samples, i.e., $D = \cup_{s=1}^{n/k} D_s, D_s = \{(x_i^s, y_i^s)\}_{i=1}^{k}$;
2. Each sample belongs to only one subset, i.e., $D_i \cap D_j = \emptyset, i \neq j$.

Maintaining continuity and similarity between adjacent subspaces is essential for synthesizing data via multiple linear interpolations in ASIDS. Our objective is to minimize the linear fitting error ϵ_i', which helps to satisfy ASIDS assumption 2 as much as possible.

To determine the sample set D_s for subspace $\mathcal{X}_s$, it is necessary to determine the first sample x_1^s in D_s.

$$x_1^s = \underset{x: x \in D, x \notin D_1, \ldots, D_{s-1}}{argmin} dist(x, \overline{x}^{s-1}), \tag{9}$$

where $s = 1, \ldots, \frac{n}{k}$, $\overline{x}^{s-1}$ is the cluster center of D_{s-1}, and $\overline{x}^0 = x_0$. We define $D_s = \{x_1^s\}$ and determine x_d^s as follows:

$$x_d^s = \underset{x: x \in D, x \notin D_1, \ldots, D_s}{argmin} dist(x, \overline{x}^s), \tag{10}$$

where $d = 2, \ldots, k$. x_d^s is obtained and $D_s \leftarrow D_s \cup \{x_d^s\}$ is updated.

The main steps of the K-Space algorithm are summarized in Algorithm 2.

Algorithm 2: K-Space.

Input: Dataset $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$; hyperparameter k
Output: $\{D_s\}_{s=1}^{n/k}$

Obtain the sample minimum point $\boldsymbol{x}_0$
for $s = 1, \ldots, \frac{n}{k}$ **do**
  $\boldsymbol{x}_1^s = \underset{\boldsymbol{x}:\boldsymbol{x}\in D, \boldsymbol{x}\notin D_1,\ldots,D_{s-1}}{argmin} dist(\boldsymbol{x}, \overline{\boldsymbol{x}}^{s-1})$
  $D_s = \{\boldsymbol{x}_1^s\}$
  for $d = 2, \ldots, k$ **do**
    $\boldsymbol{x}_d^s = \underset{\boldsymbol{x}:\boldsymbol{x}\in D, \boldsymbol{x}\notin D_1,\ldots,D_s}{argmin} dist(\boldsymbol{x}, \overline{\boldsymbol{x}}^{s})$
    $D_s \leftarrow D_s \cup \{\boldsymbol{x}_d^s\}$
  end
end

3.3. K-Match

We can calculate the total error of the matching scheme to measure the quality of the scheme; for the sake of simplicity, let $\mathcal{X} = R$, and we calculate it as follows:

$$\sum_{i=1}^{k} S(x_i^{(d)}, x_i^{(d+1)}) = \sum_{i=1}^{k} \int_{x_i^{(d)}}^{x_i^{(d+1)}} |f(x) - L_i(x)| dx, \tag{11}$$

where $L_i(x)$ is the linear expression passing through the points $(x_i^{(d)}, y_i^{(d)})$ and $(x_i^{(d+1)}, y_i^{(d+1)})$.

Theorem 1. *Let $\mathcal{X}_{(d)}$ and $\mathcal{X}_{(d+1)}$ be two adjacent subspaces; the datasets corresponding to different subspaces are $D_{(d)}, D_{(d+1)}$, and $(x_i^{(d)}, y_i^{(d)}) \in D_{(d)}, (x_i^{(d+1)}, y_i^{(d+1)}) \in D_{(d+1)}$. Consider the model $y_i = f(x_i) + \epsilon_i$, and let $\epsilon_i^{(d)} = y_i^{(d)} - f(x_i^{(d)})$. For $\forall i = 1, 2, \ldots, k$, suppose that $\epsilon_i' \to 0$; then, $E\left(\frac{S(x_i^{(d)}, x_i^{(d+1)})}{|x_i^{(d)} - x_i^{(d+1)}|}\right) < E\left(\frac{|\epsilon_i^{(d)}| + |\epsilon_i^{(d+1)}|}{2}\right)$.*

Proof of Theorem 1. Since $\epsilon_i' \to 0$, according to Equation (3) the model can be transformed into

$$y_i = g(x_i) + \epsilon_i,$$

where $g(\cdot)$ is a linear function. According to Equation (11), it follows that

$$S(x_i^{(d)}, x_i^{(d+1)}) = \int_{x_i^{(d)}}^{x_i^{(d+1)}} |g(x) - L_i(x)| dx.$$

When $\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0$, let (x', y') be the intersection point between $y = L_i(x)$ and $y = g(x)$. We can simplify $S(x_i^{(d)}, x_i^{(d+1)})$ using basic geometric area calculations, and according to the law of iterated expectations (LIE),

$$
\begin{aligned}
E\left(\frac{S(x_i^{(d)}, x_i^{(d+1)})}{|x_i^{(d)} - x_i^{(d+1)}|}\right) &= \frac{E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} \geq 0\Big)P(\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} \geq 0)}{|x_i^{(d)} - x_i^{(d+1)}|} \\
&+ \frac{E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0\Big)P(\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0)}{|x_i^{(d)} - x_i^{(d+1)}|} \\
&= \frac{E\Big(|\epsilon_i^{(d)}| + |\epsilon_i^{(d+1)}|\Big)P(\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} \geq 0)}{2} \\
&+ \frac{\Big(h_1 \cdot E|\epsilon_i^{(d)}| + h_2 \cdot E|\epsilon_i^{(d+1)}|\Big)P(\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0)}{2},
\end{aligned}
$$

where $h_1 = \frac{|x' - x_i^{(d)}|}{|x_i^{(d)} - x_i^{(d+1)}|}$ and $h_2 = \frac{|x' - x_i^{(d+1)}|}{|x_i^{(d)} - x_i^{(d+1)}|}$. Since $P(\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} \geq 0) + P(\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0) = 1$, and $h_1 + h_2 = 1$, it follows that $E\left(\frac{S(x_i^{(d)}, x_i^{(d+1)})}{|x_i^{(d)} - x_i^{(d+1)}|}\right) < E\left(\frac{|\epsilon_i^{(d)}| + |\epsilon_i^{(d+1)}|}{2}\right)$.

□

If our approach is to randomly select a matching scheme, the validity of this method can be proved by Theorem 1. However, randomly selecting a matching scheme does not guarantee the uniqueness of the results, and it also does not guarantee that we will necessarily select a good-performing matching scheme. We found that for $x_i^{(d)}$ and $x_i^{(d+1)}$, if $\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0$, there will be a better interpolation effect.

Theorem 2. *Let* $y_i^{(d)} = f(x_i^{(d)}) + \epsilon_i^{(d)}, y_i^{(d+1)} = f(x_i^{(d+1)}) + \epsilon_i^{(d+1)}$. *Suppose that* $\epsilon_i' \to 0$; *then, we can obtain* $E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} < 0\Big) < E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)} \cdot \epsilon_i^{(d+1)} \geq 0\Big)$.

Proof of Theorem 2. Since $\epsilon_i' \to 0$, as based on the proof of Theorem 1, we can obtain

$$
\begin{aligned}
&E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)}\epsilon_i^{(d+1)} \geq 0\Big) \\
&= \frac{|x_i^{(d)} - x_i^{(d+1)}| \cdot E|\epsilon_i^{(d)}| + |x_i^{(d)} - x_i^{(d+1)}| \cdot E|\epsilon_i^{(d+1)}|}{2},
\end{aligned}
$$

$$
\begin{aligned}
&E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)}\epsilon_i^{(d+1)} < 0\Big) \\
&= \frac{|x' - x_i^{(d)}| \cdot E|\epsilon_i^{(d)}| + |x' - x_i^{(d+1)}| \cdot E|\epsilon_i^{(d+1)}|}{2}.
\end{aligned}
$$

It follows that

$$
E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)}\epsilon_i^{(d+1)} < 0\Big) < E\Big(S(x_i^{(d)}, x_i^{(d+1)})|\epsilon_i^{(d)}\epsilon_i^{(d+1)} \geq 0\Big).
$$

□

According to Theorem 2, we can match samples with the opposite signs of ϵ_i to achieve a good data synthesis effect. Therefore, the core idea of K-Match is to judge whether the sign of ϵ_i is positive or negative for each sample, and then interpolate the samples with opposite signs as much as possible.

In K-Match, we need to choose an appropriate linear regression method to fit the dataset $D_{(d)} \cup D_{(d+1)}$ based on the performance of the noise. For example, lasso regression, locally weighted linear regression (LWLR) [24], and other methods can be used [25,26]. In our experiments, we used the OLS or SVR method to fit and obtain $\hat{g}(\cdot)$. Notably, the kernel function is linear in SVR. According to Equation (3), and supposing that the linear fitting error is $\epsilon_i' \to 0$ for dataset $D_{(d)} \cup D_{(d+1)}$, we can obtain

$$\epsilon_i = y_i - \hat{g}(\boldsymbol{x}_i). \tag{12}$$

Then, we sort the samples in dataset $D_{(d)}$ in ascending order according to the value of ϵ_i, and obtain $\{(\boldsymbol{x}_i^{(d)}, y_i^{(d+1)})\}_{i=1}^k$, whereas we sort the samples in dataset $D_{(d+1)}$ in descending order and obtain $\{(\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)})\}_{i=1}^k$. As is shown in Figure 1d, the sorted datasets $D_{(d)}$ and $D_{(d+1)}$ are combined into the matching scheme $\{(\boldsymbol{x}_i^{(d)}, y_i^{(d)}), (\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)})\}_{i=1}^k$, the main steps of the K-Space algorithm are summarized in Algorithm 3.

Algorithm 3: K-Match.

Input: Subset $D_{(d)} = \{(\boldsymbol{x}_i^{(d)}, y_i^{(d)})\}_{i=1}^k, D_{(d+1)} = \{(\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)}\}_{i=1}^k$

Output: Matching scheme $\{(\boldsymbol{x}_i^{(d)}, y_i^{(d)}), (\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)})\}_{i=1}^k$

1 Fit the dataset $D_{(d)} \cup D_{(d+1)}$ and obtain $\hat{g}(\cdot)$
2 Obtain $\{\epsilon_i\}$ using Equation (10)
3 Sort the samples in $D_{(d)}$ and $D_{(d+1)}$ according to the value of ϵ_i, and obtain $D'_{(d)}$ and $D'_{(d+1)}$
4 Combine $D'_{(d)}$ and $D'_{(d+1)}$ into $\{(\boldsymbol{x}_i^{(d)}, y_i^{(d)}), (\boldsymbol{x}_i^{(d+1)}, y_i^{(d+1)})\}_{i=1}^k$

3.4. Supplementary Notes

The proposed method can effectively expand the size of the dataset and adjust the dataset structure, reducing the proportion of samples that deviate significantly from the actual distribution, and thereby improve model generalization performance (see Figure 2).

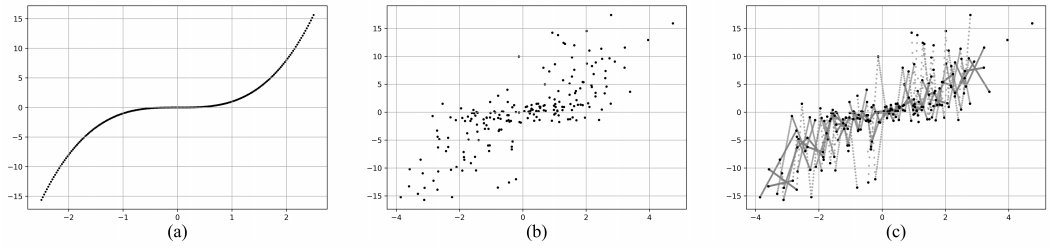

Figure 2. The synthetic data from ASIDS. (**a**) The true relationship between x and y is $y = x^3$. The sample size of the original data is 200 (**b**) Adding Gaussian noise. (**c**) Let $k = 6$ and $\eta = 100$; by processing with ASIDS, the sample size of the generated synthetic data was 3808.

The supplements to ASIDS are given below:

1. The choice of the hyperparameter k is crucial, as different datasets require different values of k. Conversely, the hyperparameter η tends to exhibit a better performance as its value increases, which will be illustrated in the experimental results in the following section.
2. It is necessary to normalize the data if there is a significant difference in the dimensional scale between the features of the data. This avoids the issue of generating an excessive number of samples.
3. In most cases, n/k is not an integer, and for the excess samples, we usually have two solutions for handling this. The first one is to use the LOF algorithm [27] to filter out

the excess samples that are not needed for ASIDS, as is shown in Figure 2c. Another solution is to treat the excess samples as a dataset D' of a subspace $D' = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n \bmod k}$. When interpolating between D' and other subspaces D'', we choose an appropriate linear regression method to fit the dataset $D' \cup D''$ and obtain $\hat{g}(\cdot)$. Then, we use the same method to sort D' and D''. Only $n \bmod k$ interpolations are performed, with each sample in D' being interpolated, while for D'', only $n \bmod k$ samples are interpolated. Moreover, we interpolate the samples with the opposite signs of ϵ_i as much as possible, as is shown in Figure 3.

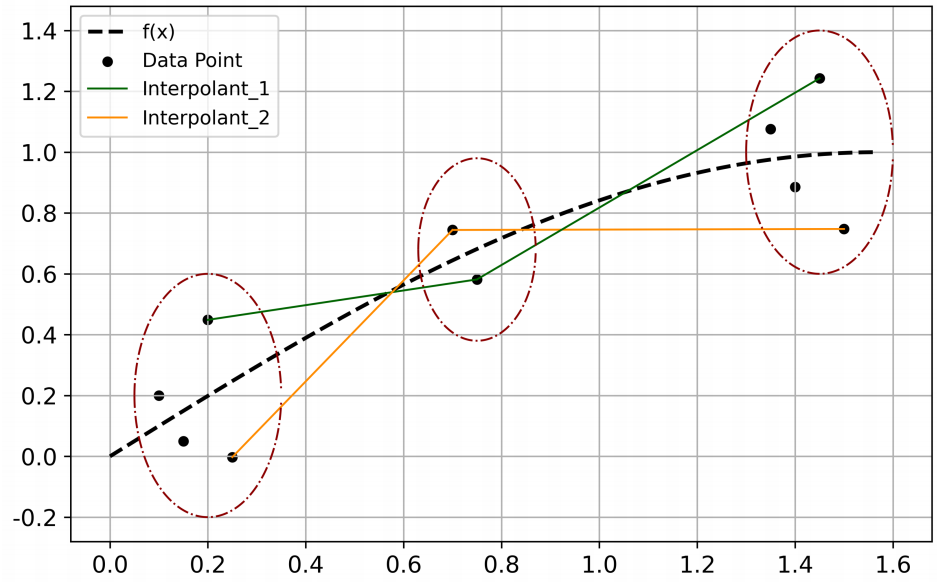

Figure 3. Interpolations in subspaces with different number of samples.

4. Verification of the Performances of ASIDS

For the artificial data, which are also known as $f(\cdot)$, we investigated the hyperparameter selection and optimization performance after processing with ASIDS. For the benchmark data, we studied the prediction performance using this method.

4.1. Simulated Datasets

Some verification indicators, including the proportion of samples with ϵ_i greater than α and the mean square error (MSE), can be selected to test the optimization effect of ASIDS on the original sample after processing as follows:

$$p(\alpha) = \frac{1}{n}\sum_{i=1}^{n} I(|f(\boldsymbol{x}_i) - y_i| > \alpha), \tag{13}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f(\boldsymbol{x}_i) - y_i)^2. \tag{14}$$

For the simulated datasets, $\{\boldsymbol{x}_i\}_{i=1}^{n}$ is generated using $N_p(\boldsymbol{0}, \boldsymbol{E})$. Let $\boldsymbol{W}_1 \in \mathbb{R}_{(p \times p_1)}$ and $\boldsymbol{W}_2 \in \mathbb{R}_{(p_1 \times 1)}$, where all elements of both $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are independently and identically distributed as $N(0,1)$. Consider that $y_i = f(\boldsymbol{x}_i) + \epsilon_i = tanh(\boldsymbol{x}_i'\boldsymbol{W}_1)\boldsymbol{W}_2 + \epsilon_i$.

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

For a given dataset, we typically cannot ascertain the distribution of noise. Hence, we constructed datasets that contain unknown noises [12]. The unknown noises are simulated by a mixture of noises that contain uniform noises and Gaussian noises. The generation of ϵ_i

is explained in [28]. We fixed the random effects of the generated datasets; the experiments were repeated 100 times for each simulated dataset. Table 1 shows six simulated datasets. Specifically, P is the number of features in each sample of the simulated datasets. P_1 is the dimension of the intermediate layer used to generate the y values.

Table 1. Simulated datasets.

Simulated Datasets	ϵ_i Distribution	Sample Size	(P, P_1)
D_1	20%-N(0, 64) 30%-U(−8,8) 50%-N(0, 0.04)	500	(5, 3)
D_2	20%-N(0, 64) 30%-U(−8,8) 50%-N(0, 0.04)	200	(5, 3)
D_3	20%-N(0, 64) 30%-U(−8,8) 50%-N(0, 0.04)	1500	(5, 3)
D_4	20%-N(0, 64) 30%-U(−8,8) 50%-N(0, 0.04)	500	(1, 3)
D_5	20%-N(0, 64) 30%-U(−8,8) 50%-N(0, 0.04)	500	(20, 10)
D_6	40%-N(0, 64) 45%-U(−8,8) 15%-N(0, 0.04)	500	(5, 3)

4.1.1. Hyperparameter Selection

First, we investigated the determination of parameter k. To ensure that the number of feature subspaces was sufficient, we set the hyperparameter $\eta = 1$, letting $k = 1, 2, \ldots, \lfloor \frac{n}{2} \rfloor$. To calculate the changes in the MSE of the datasets at different values of k, we obtained $k' = \underset{k:k=1,2,\ldots,\lfloor \frac{n}{2} \rfloor, \eta=1}{argmin} MSE$. The change trend in the MSE index before and after the ASISO processing is shown in Figure 4. Then, by letting $\eta = 1, 2, \ldots, 30, k = k'$, we calculated the changes in the MSE of the datasets (Figure 5), and obtained $\eta' = \underset{\eta:\eta=1,2,\ldots,30, k=k'}{argmin} MSE$. At last, we let $k = k', \eta = 1$, and η', and we calculated the changes in $p(\alpha)$ of the simulated datasets (Figure 6).

As can be seen from Figure 4, ASIDS demonstrates good optimization performance for datasets with varying sample sizes or feature dimensions. The experimental results also show that the performance of ASIDS does not decline dramatically as the content of noise with large variances increases. Hence, it can deal with unknown noise well and has good robustness. In addition, it was shown by the experimental results that ASIDS exhibits good optimization performance for all datasets; thus, it is also a stable data synthesis method. As can be seen from Figure 5, the hyperparameter η has a generally monotonically decreasing relationship with the MSE, and the larger the value of η, the more significant the effect. From Figure 6, it can be observed that ASIDS can adaptively adjust the sample structure, which reduces the proportion of samples with large errors.

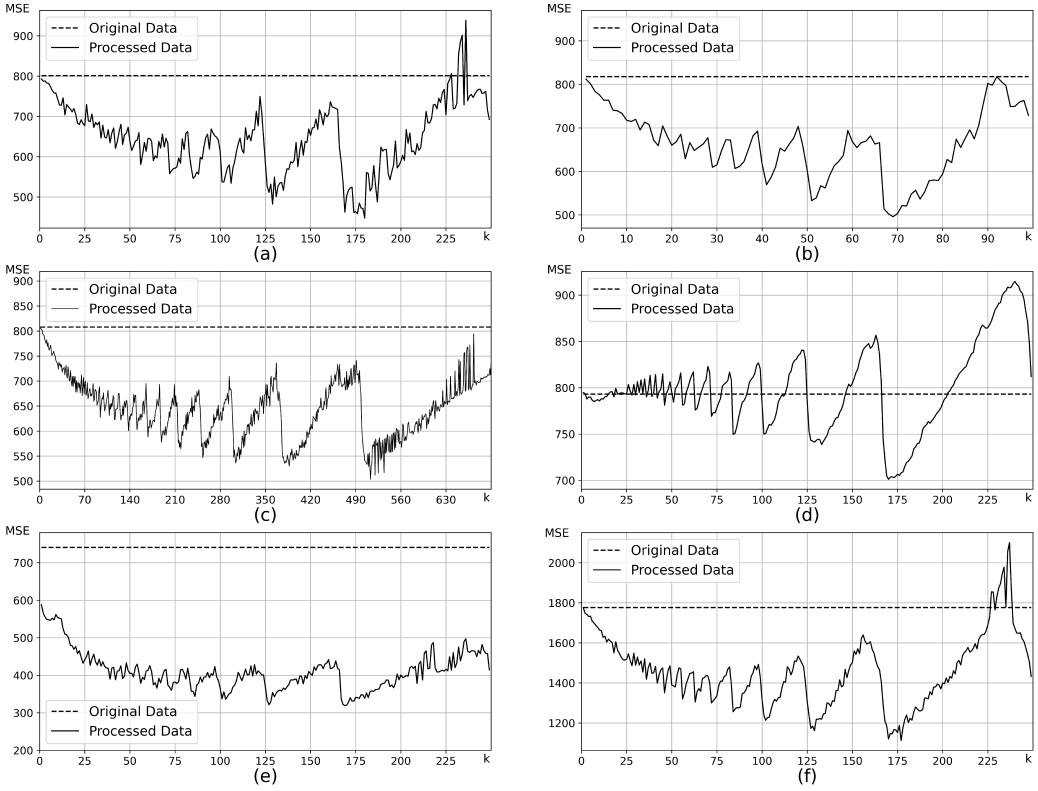

Figure 4. Changes in the MSE under different k values: (**a**) simulation study results of dataset D_1; (**b**) simulation study results of dataset D_2; (**c**) simulation study results of dataset D_3; (**d**) simulation study results of dataset D_4; (**e**) simulation study results of dataset D_5; and (**f**) simulation study results of dataset D_6, which will not be specifically mentioned below.

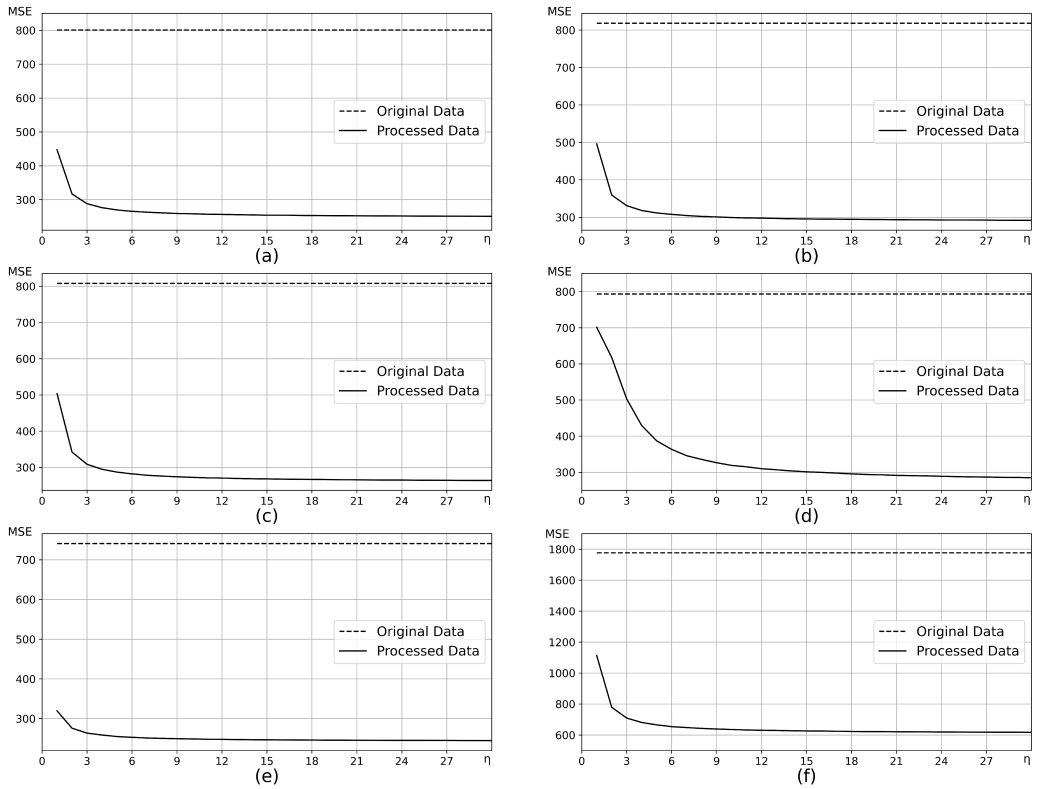

Figure 5. Changes in the MSE under different η values.

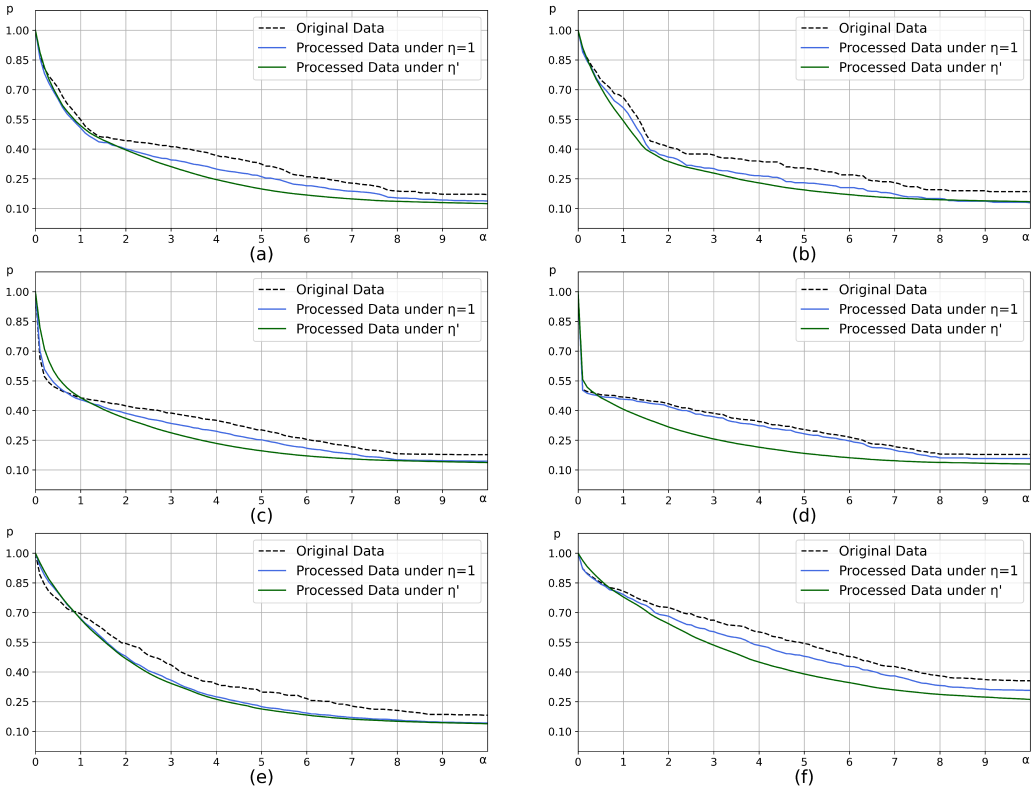

Figure 6. Changes in the $p(\alpha)$ under different η values.

4.1.2. Comparison of Optimization Performance

To verify the optimization performance of ASIDS, we compared it with three other methods: piecewise linear interpolation, linear extrapolation, and nearest neighbor interpolation. Specifically, we let $k = k'$ and $\eta = \eta'$ for ASIDS. Based on the given dataset, we used the samples generated by ASIDS as interpolation points to calculate the output values of linear extrapolation and nearest neighbor interpolation. Moreover, by letting $k = 1$ and $\eta = \eta'$ for ASIDS, we can regard it as piecewise linear interpolation. The experimental results show that ASIDS has the smallest MSE among all methods for each dataset (Figure 7).

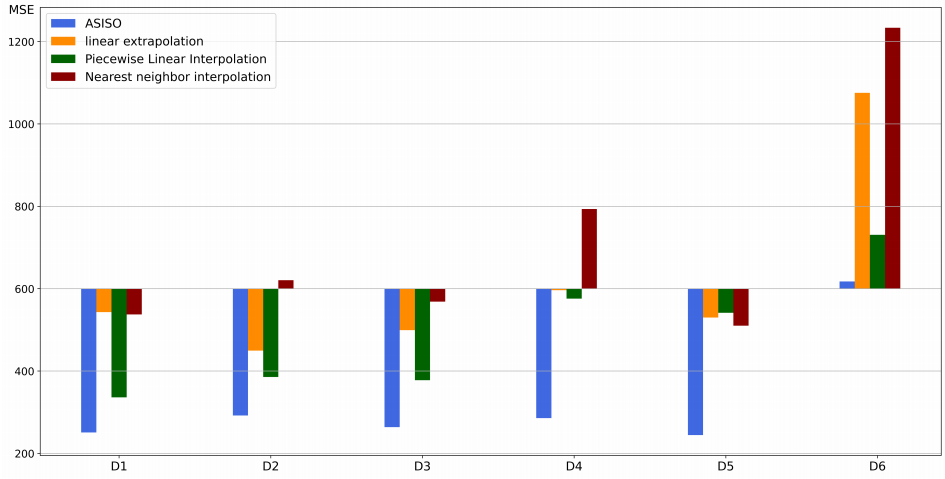

Figure 7. Comparison of MSEs of simulated datasets.

4.2. Benchmark Datasets

To verify the prediction performance of ASIDS, four benchmark datasets were used [29,30]. We partitioned the dataset into training and testing sets using a 7:3 ratio and normalized the data using the min–max normalization method. We preprocessed the training data using ASIDS, trained multiple machine learning models on the training set, and evaluated the prediction performance of the models on the testing set. The evaluation metric we used was the mean absolute error (MAE).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|. \tag{15}$$

In addition, we removed features that cannot be directly used, such as "Datetime" for bike sharing demand, "Month" for forest fires, and so on. We chose the K-nearest neighbor (KNN), random forest (RF), gradient boosting decision tree (GBDT), multilayer perceptron (MLP), and support vector regression (SVR) as the machine learning prediction models. Specifically, the kernel function is the radial basis function (RBF) in SVR. Moreover, we set the number of hidden layers to 3 for the MLP and used different numbers of neurons based on the input dimensionality and sample size. Prior to and subsequent to the application of ASIDS, the MLP retains an identical network architecture. Additionally, there are negligible variations in the count of base learners within ensemble methodologies, such as RF and GBDT. On a holistic level, these variations do not impart significant changes to the model's complexity.

The experimental results of the five models are shown in Table 2 (the parameters for both the models and algorithms across diverse datasets were optimized using grid search and cross-validation). It is evident that ASIDS performs well on each benchmark dataset. This method is highly applicable to all five models, and in most cases, it can improve the predictive performance. It is worth mentioning that the dataset contains many categorical features which are not continuous variables. Moreover, for sparse samples (Facebook Metrics and Forest Fires), it is difficult to guarantee that the linear fitting error is $\epsilon_i' \to 0$ when interpolating between subspaces. This indicates that even if there are violations of the ASIDS assumptions in practical applications, this method may still achieve good optimization results. This further demonstrates that this method is robust and stable.

Table 2. Experimental results for benchmark datasets.

Datasets	Processing	Hyperparameter	Testing MAE (10^{-2})				
			KNN	RF	MLP	SVR	GBDT
Bike Sharing	-	-	2.20	0.12	0.54	4.26	0.23
	ASIDS	$k = 150, \eta = 10$	2.00	0.06	0.25	4.12	0.19
Facebook	-	-	6.60	2.25	8.19	7.34	1.33
	ASIDS	$k = 2, \eta = 10$	6.41	1.59	2.02	7.36	1.12
Air Quality	-	-	2.99	2.57	4.80	3.87	2.58
	ASIDS	$k = 20, \eta = 100$	2.89	2.55	2.08	3.84	2.77
Forest Fires	-	-	4.06	4.93	10.58	7.36	4.49
	ASIDS	$k = 10, \eta = 100$	3.89	4.32	4.80	10.05	4.14

5. Conclusions

In this paper, we proposed a data synthesis method, ASIDS, which can adaptively adjust the size of the dataset, and the generated synthetic data typically contain minimal errors. Moreover, it can adjust the structure of the samples, which can significantly reduce the proportion of samples with large errors. The experimental results from the simulated datasets demonstrate that ASIDS can optimize the samples, and compared to other methods, the data generated using this method had smaller errors. ASIDS can deal with unknown noise better and has good robustness. The results from the benchmark datasets show that the proposed method is applicable for many machine models, and in most cases,

it can improve the model generalization. ASIDS has some deficiencies and limitations. In practical applications, it should be considered whether the real scenario data can satisfy the assumptions of ASIDS. From the experimental results, it can be seen that the choice of hyperparameters has a great influence on the results. Future work may focus on practical applications and its integration with advanced machine learning techniques, and study of how to automatically or effectively select hyperparameters.

Author Contributions: Writing—original draft, Y.D.; Software, Y.C.; Writing—Data curation & editing, X.J.; Methodology, H.W.; review, Y.L.; review, M.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Social Science Fund of China grant number 22BTJ021.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. ALRikabi, H.T.S.; Hazim, H.T. Enhanced data security of communication system using combined encryption and steganography. *iJIM* **2021**, *15*, 145. [CrossRef]
2. Kollias, D. ABAW: Learning from synthetic data & multi-task learning challenges. In *Computer Vision—ECCV 2022 Workshops*; Springer: Cham, Switzerland, 2022; pp. 157–172.
3. Mahesh, B. Machine learning algorithms—A review. *Int. J. Sci. Res. (IJSR)* **2020**, *9*, 381–386.
4. Lepot, M.; Aubin, J.B.; Clemens, F.H.L.R. Interpolation in time series: An introductive overview of existing methods, their performance criteria and uncertainty assessment. *Water* **2017**, *9*, 796. [CrossRef]
5. Chlap, P.; Min, H.; Vandenberg, N.; Dowling, J.; Holloway, L.; Haworth, A. A review of medical image data augmentation techniques for deep learning applications. *J. Med. Imaging Radiat. Oncol.* **2021**, *65*, 545–563. [CrossRef] [PubMed]
6. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48. [CrossRef]
7. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
8. Dablain, D.; Krawczyk, B.; Chawla, N.V. DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**. [CrossRef] [PubMed]
9. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In Proceedings of the International Conference on Intelligent Computing, Hefei, China, 23–26 August 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 878–887.
10. Bunkhumpornpat, C.; Sinapiromsaran, K.; Lursinsap, C. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining, Proceedings of the 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, 27–30 April 2009*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 475–482.
11. Ha, T.; Dang, T.K.; Dang, T.T.; Truong, T.A.; Nguyen, M.T. Differential privacy in deep learning: An overview. In Proceedings of the 2019 International Conference on Advanced Computing and Applications (ACOMP), Nha Trang, Vietnam, 26–28 November 2019; pp. 97–102.
12. Meng, D.; De La Torre, F. Robust matrix factorization with unknown noise. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 1337–1344.
13. Raghunathan, T.E. Synthetic data. *Annu. Rev. Stat. Its Appl.* **2021**, *8*, 129–140. [CrossRef]
14. Sibson, R. A brief description of natural neighbour interpolation. In *Interpreting Multivariate Data*; Wiley: New York, NY, USA, 1981; pp. 21–36.
15. Tachev, G.T. Piecewise linear interpolation with nonequidistant nodes. *Numer. Funct. Anal. Optim.* **2000**, *21*, 945–953. [CrossRef]
16. Blu, T.; Thévenaz, P.; Unser, M. Linear interpolation revitalized. *IEEE Trans. Image Process.* **2004**, *13*, 710–719. [CrossRef] [PubMed]
17. Berrut, J.P.; Trefethen, L.N. Barycentric lagrange interpolation. *SIAM Rev.* **2004**, *46*, 501–517. [CrossRef]
18. Musial, J.P.; Verstraete, M.M.; Gobron, N. Comparing the effectiveness of recent algorithms to fill and smooth incomplete and noisy time series. *Atmos. Chem. Phys.* **2011**, *11*, 7905–7923. [CrossRef]
19. Fornberg, B.; Zuev, J. The Runge phenomenon and spatially variable shape parameters in RBF interpolation. *Comput. Math. Appl.* **2007**, *54*, 379–398. [CrossRef]
20. Rabbath, C.A.; Corriveau, D. A comparison of piecewise cubic Hermite interpolating polynomials, cubic splines and piecewise linear functions for the approximation of projectile aerodynamics. *Def. Technol.* **2019**, *15*, 741–757. [CrossRef]
21. Habermann, C.; Kindermann, F. Multidimensional spline interpolation: Theory and applications. *Comput. Econ.* **2007**, *30*, 153–169. [CrossRef]
22. Ganzburg, M.I. The Bernstein constant and polynomial interpolation at the Chebyshev nodes. *J. Approx. Theory* **2002**, *119*, 193–213. [CrossRef]

23. Bové, D.S.; Held, L.; Kauermann, G. Objective Bayesian Model Selection in Generalized Additive Models With Penalized Splines. *J. Comput. Graph. Stat.* **2015**, *24*, 394–415. [CrossRef]
24. Cleveland, W.S. Robust locally weighted regression and smoothing scatterplots. *J. Am. Stat. Assoc.* **1979**, *74*, 829–836. [CrossRef]
25. Lichti, D.D.; Chan, T.O.; Belton, D. Linear regression with an observation distribution model. *J. Geod.* **2021**, *95*, 1–14. [CrossRef]
26. Liu, C.; Li, B.; Vorobeychik, Y.; Oprea, A. Robust linear regression against training data poisoning. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Dallas, TX, USA, 3 November 2017; pp. 91–102.
27. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; pp. 93–104.
28. Guo, Y.; Wang, W.; Wang, X. A robust linear regression feature selection method for data sets with unknown noise. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 31–44. [CrossRef]
29. Cukierski, W. Bike Sharing Demand. Kaggle. 2014. Available online: https://kaggle.com/competitions/bike-sharing-demand (accessed on 25 October 2014).
30. Dua, D.; Craff, C. UCI Machine Learning Repository. 2017. Available online: http://archive.ics.uci.edu/ml (accessed on 25 January 2017).