

Article

Regression Analysis Under Interval-Valued Targets as an Imprecise Classification Problem

Lev Utkin ^{†,‡}, Stanislav Kogan ^{†,‡}, Andrei Konstantinov ^{†,‡} and Vladimir Muliukha ^{*,†,‡}

Higher School of Artificial Intelligence Technologies, Peter the Great St. Petersburg Polytechnic University, 195251 St. Petersburg, Russia

* Correspondence: vladimir.muliukha@spbstu.ru

[†] These authors contributed equally to this work.

[‡] Current address: Peter the Great St. Petersburg Polytechnic University, Polytechnicheskaya, 29, 195251 St. Petersburg, Russia.

Abstract

Regression analysis with interval-valued outcomes presents a fundamental challenge in modeling data where uncertainty is inherent rather than incidental. Such data, arising naturally in fields ranging from meteorology to finance, require methods that preserve information about both central tendency and dispersion. We introduce a novel class of attention-based regression models that reformulates interval-valued regression as a multiclass classification task. The key idea behind the model is in partitioning the outcome domain into basic intervals derived from training data intersections and representing each interval-valued observation as a set of feasible discrete probability distributions over these intervals. This imprecise probabilistic representation allows us to train a classification-style model by minimizing the expected log-likelihood over all consistent distributions. We propose two training algorithms: a Monte Carlo sampling approach and a more efficient joint optimization method that simultaneously updates both the constrained probability distributions and model parameters. The model incorporates a kernel-based aggregation mechanism using trainable dot-product attention, where attention weights are computed from input features but applied to the probability distributions over basic intervals. Numerical experiments with real datasets illustrate the approach. By introducing the class of attention-based models for interval-valued regression, this work offers a novel perspective on applying machine learning to uncertain data. Codes implementing the proposed models are publicly available.

Keywords: imprecise probabilities; interval-valued targets; classification; regression; attention mechanism



Academic Editor: Stefano Mariani

Received: 28 January 2026

Accepted: 21 February 2026

Published: 24 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

In many real-world scenarios, data uncertainty is not merely noise to be eliminated but an intrinsic characteristic of the phenomenon being measured. This is elegantly captured through interval-valued data, where an observation is defined not by a scalar but by a lower and upper bound, forming a continuous interval. Such data arises naturally in numerous contexts: daily minimum and maximum temperatures in meteorology, low and high prices of a stock in a trading day, confidence intervals around survey results, ranges in biomedical measurements (e.g., systolic and diastolic blood pressure), and fluctuation bands in industrial processes. Modeling this datatype directly, rather than reducing it to a

midpoint, preserves valuable information about the variability and inherent uncertainty within the outcome variable.

The task of modeling such data leads to the field of Interval-Valued Regression. The core challenge extends beyond predicting a single value; it involves accurately predicting both the center (location) and the spread (width) of the outcome interval simultaneously. This necessitates a fundamental departure from standard regression paradigms. Ignoring the interval structure by, for instance, modeling only the midpoints, results in a critical loss of information regarding the data's dispersion and can lead to biased and overconfident inferences. Conversely, treating the lower and upper bounds as independent separate outcomes fails to capture their intrinsic dependency and shared underlying structure.

Many models and methods handling interval-valued data have been proposed in recent years [1–6]. They apply different frameworks, including imprecise probabilities [7], evidence theory or Dempster–Shafer theory [8,9], and interval analysis [10]. While these approaches provide rigorous theoretical foundations, many remain computationally intensive, difficult to scale to high-dimensional data, or lack seamless integration with modern machine learning pipelines. As datasets grow in size and complexity, the need for flexible, interpretable, and scalable models that respect the interval structure while leveraging advances in deep learning has become urgent. Recent efforts in neural interval regression, distributional regression, and conformal prediction offer promising pathways toward this goal, yet a unified framework that balances statistical coherence, computational efficiency, and practical usability remains an open and vital challenge.

We propose a novel approach for regression with interval-valued output variables that is motivated by results provided in [11] for a survival analysis task. The core idea is to partition the outcome domain into a set of basic intervals, defined as the intersections of the initial intervals observed in the training data. For each instance, we then define an imprecise discrete probability distribution that is a set of possible discrete probability distributions over these basic intervals. We refer to these intervals as basic intervals. This imprecise distribution effectively serves as a probabilistic representation of the original interval-valued outcome of the corresponding instance. In other words, we replace interval-valued outcomes in the training set with a set of discrete probability distributions over the basic intervals (with the imprecise distribution).

This reformulation enables us to train a classification-style model using the log-likelihood function as a loss function, where the probabilities correspond to the mass assigned to each basic interval. However, a key challenge arises: because the distributions are imprecise (i.e., each instance corresponds to a set of possible probability distributions), the loss function itself becomes set-valued. To address this, we adopt a strategy based on minimizing the expected loss over all feasible precise probability distributions consistent with the interval constraints.

Two computational approaches are proposed to solve this optimization problem. The first employs Monte Carlo sampling [12]: numerous precise probability distributions are sampled from the set defined by each interval-valued observation, and the loss is averaged across these samples during training. The second approach performs joint optimization: it simultaneously updates both the underlying probability distributions (subject to interval constraints) and the parameters of the classification model, avoiding explicit sampling altogether. This method is computationally more efficient, as it sidesteps the need to generate and aggregate multiple distributions per instance.

An additional innovation lies in the use of kernel-based aggregation via the Nadaraya–Watson regression [13,14], enhanced with trainable dot-product attention weights [15,16]. Crucially, the attention mechanism operates not on the input feature vectors directly, but on the probability distributions over the basic intervals, though the attention weights

themselves are computed as functions of the input features. The output of this attention module is a single aggregated probability distribution over the basic intervals for all instances. By minimizing the expected log-likelihood loss, we jointly learn the attention parameters and the model's predictive structure.

This design enables natural extensions: for example, the approach can be augmented with the multi-head attention or integrated into deeper neural architectures. Importantly, our approach constitutes a novel mechanism for applying attention to interval-valued data, which can be regarded as an area largely unexplored in the literature.

Finally, the following properties of the proposed approach distinguish it from the existing ones. First of all, most machine learning models that deal with interval-valued targets predict only intervals for each new instance, which can be regarded as providing limited information. Often, we need to know a probability distribution over the predicted domain, which, in fact, encompasses all information about the prediction. In contrast to existing interval-valued models, the proposed model provides such a probability distribution. Importantly, this predicted distribution is not based on any assumptions about the type of distribution, that is, a fully nonparametric case is considered. The second distinctive idea behind the proposed model is that the regression task is reduced to a classification task with a finite number of classes characterized by probabilities. This allows us to apply the softmax operation to predict class probabilities and a probability distribution over the predicted domain. The third important difference between the proposed model and existing interval-valued models is that it originally applies the attention mechanism. This enables the model to learn and implement complex multidimensional functional dependencies and provides an inherent way to construct a family of models trained on interval-valued data, which can be implemented using transformers. Moreover, these models can be incorporated into transformers as a component. Fourth, the proposed model considers a probability distribution from a set of distributions as a training parameter that can be learned by minimizing the corresponding loss function using the standard training algorithm. To the best of our knowledge, there are no machine learning models dealing with interval-valued targets that use at least some of the above features.

Our contributions can be summarized as follows:

1. A novel attention-based regression model for interval-valued outputs, formulated as an imprecise classification task where classes correspond to the basic intervals derived from the training data. The model's prediction is a probability distribution over these intervals, capturing both central tendency and uncertainty.
2. Two algorithms are proposed to train the model. The first uses Monte Carlo sampling to approximate the expected loss over feasible probability distributions. The second performs joint optimization of the probability distributions (constrained by the interval bounds) and the model parameters, offering greater computational efficiency.
3. Numerical experiments with real and synthetic data are provided to justify the approach. Codes implementing the proposed models are publicly available at: <https://github.com/NTAILab/iprob> (assessed on 28 January 2026)

The paper is organized as follows. Related work devoted to models handling interval-valued data can be found in Section 2. Section 3 provides basic definitions of the Nadaraya–Watson regression jointly with the attention mechanism. An approach to represent the interval regression problem as the imprecise classification is considered in Section 4. This section also contains two algorithms for training the model. Numerical experiments with synthetic and real data illustrating the proposed approach are given in Section 5. Concluding remarks are provided in Section 6.

2. Related Work

A substantial body of research has been devoted to modeling interval-valued data, both as input features and output variables. Machine learning models with interval-valued feature vectors have been studied by several authors [17–19]. Billard and Diday [20] introduced foundational covariance and correlation measures for interval-valued data, enabling statistical analysis in symbolic domains. Tree-based regression methods for interval outputs were proposed in [21], with random forest extensions following in [22]. A multiview classification framework for interval-valued data was developed in [23], while domain adaptation under interval uncertainty was addressed in [24].

One of the most influential paradigms in this field is the Center-and-Range Method (CRM), originally introduced in [25]. This approach represents each interval as a pair (c, r) , where c denotes the center (midpoint) and r the radius (half-width), thereby transforming interval-valued regression into a bivariate prediction task. CRM has since been extended to regularized neural networks [26] and nonparametric kernel-based regressions [27], where the use of kernel functions ensures mathematical coherence in range estimation. Lim [28] further advanced this line of work by proposing nonparametric additive models for interval data that combine parametric components with smooth, data-driven additive terms.

Nonlinear regression approaches for interval-valued outcomes have also been explored, including panel data models [29] and nonlinear kernel-based methods [30]. More recently, a quantile regression neural network based on the center–radius representation (QRANN-CR) was introduced in [31], demonstrating improved predictive accuracy under heteroscedastic uncertainty. Ordinal classification for interval-valued functional data was considered in [32].

Despite these advances, spatial dependence in interval-valued data has received limited attention. Freitas et al. [33,34] initiated efforts to model geospatial patterns in interval data, linking regional variability to underlying spatial processes. Huiyuan Wang and Ruiyuan Cao [35] proposed an interval-valued linear regression model that captures linear relationships between real-valued predictors and interval-valued responses, laying the groundwork for spatially aware formulations.

In the realm of deep learning, Jiang et al. [36] introduced the Interval Dual Convolutional Neural Network to predict interval-valued stock prices, leveraging convolutional architectures to extract spatiotemporal patterns. Meanwhile, Zhong et al. [37] integrated deep quantile regression with interpretability constraints, enabling both accurate uncertainty quantification and statistical inference.

Another major theoretical foundation for handling interval uncertainty is the theory of random sets [38]. Within this framework, Petit et al. [39] proposed a nonparametric regression method grounded in a fuzzy extension of Dempster–Shafer evidence theory. Li et al. [40] developed a constrained linear regression model using random set principles, enforcing consistency with interval bounds. Denoeux [41] recently introduced an evidential likelihood-based approach for quantifying prediction uncertainty in neural networks, employing Gaussian approximations and output linearization to derive credible intervals from deep models. Feature selection for interval-valued data was addressed in [42], highlighting the need for dimensionality reduction techniques tailored to imprecise inputs. A regression model with the uncertain targets was studied in [43]. The SVM modification for the uncertain targets was proposed in [44].

Surveys on belief functions and their role in representing uncertainty in machine learning are provided in [45,46]. Utkin and Coolen [47] formulated regression and classification under set-valued risk functionals derived from inferential bounds on probability distributions, where the bounding distributions depend explicitly on model parameters.

Supporting this paradigm, SVMs and one-class classifiers for interval-valued data were developed in [48–50].

Interesting new results on models with interval-valued targets are currently presented in [51–54].

Attention mechanisms have revolutionized modern machine learning, significantly enhancing performance in classification and regression tasks through dynamic feature weighting. Comprehensive surveys of attention architectures can be found in [55–62]. However, none of the existing works incorporate attention mechanisms directly into regression problems with interval-valued outcomes. While attention has been applied to feature vectors or sequence representations, no prior method has modeled attention over probability distributions induced by interval uncertainty, nor has it learned attention weights that operate on the structure of interval-valued targets themselves.

Our aim is to bridge this gap by introducing a novel class of attention-based regression models specifically designed for interval-valued outputs, wherein trainable attention weights are computed from input features but applied to aggregated probability distributions over discretized outcome intervals. This enables the model to dynamically emphasize informative regions of the interval space, adaptively weigh contributions from training instances, and produce calibrated, interpretable predictive distributions, all while respecting the inherent imprecision of interval-valued data.

3. Preliminaries

Attention Mechanism

The idea of the attention mechanism can clearly be explained by using the Nadaraya–Watson kernel regression model [13,14]. If there is a training set $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ consisting of n instances, where $\mathbf{x}_i \in \mathbb{R}^m$ is a feature vector and $y_i \in \mathbb{R}$ is the corresponding label, then the regression output prediction $y(\mathbf{x})$, associated with a new input feature vector $\mathbf{x}$, can be estimated as the weighted average in the form of the Nadaraya–Watson kernel regression model [13,14]:

$$y(\mathbf{x}) = \sum_{i=1}^n \alpha(\mathbf{x}, \mathbf{x}_i, \mathbf{w}) y_i. \quad (1)$$

Here, $\alpha(\mathbf{x}, \mathbf{x}_i, \mathbf{w})$ is the attention weight with training parameters $\mathbf{w}$, which measures how far the feature vector $\mathbf{x}$ is from the feature vector $\mathbf{x}_i$. The closer $\mathbf{x}$ to $\mathbf{x}_i$, the greater the corresponding weight $\alpha(\mathbf{x}, \mathbf{x}_i, \mathbf{w})$. Generally, arbitrary distance functions satisfying the above condition can be regarded as the attention weights. One of the sets of the functions is the kernel set because a kernel K can be regarded as a scoring function estimating how close vector $\mathbf{x}_i$ is to vector $\mathbf{x}$. Hence, the attention weights can be represented as:

$$\alpha(\mathbf{x}, \mathbf{x}_i, \mathbf{w}) = \frac{K(\mathbf{x}, \mathbf{x}_i, \mathbf{w})}{\sum_{j=1}^n K(\mathbf{x}, \mathbf{x}_j, \mathbf{w})}. \quad (2)$$

In terms of the attention mechanism [63], vector $\mathbf{x}$, vectors $\mathbf{x}_i$ and labels y_i are called as the *query*, *keys* and *values*, respectively. Weights $\alpha(\mathbf{x}, \mathbf{x}_i, \mathbf{w})$ can be extended by incorporating trainable parameters. For example, if we take the Gaussian kernel with a trainable vector of parameters $\mathbf{w} = (w_1, \dots, w_n)$, then the attention weight can be represented as:

$$\alpha(\mathbf{x}, \mathbf{x}_i, \mathbf{w}) = \text{softmax}\left(-\|\mathbf{x} - \mathbf{x}_i\|^2 | \mathbf{w}\right). \quad (3)$$

There also exist several definitions of attention weights and the corresponding attention mechanisms; for example, the additive attention [63] and the multiplicative or dot-product attention [15,16]. The dot-product attention is used below.

4. Interval-Valued Regression as a Classification Task

Suppose that there is a dataset $\mathcal{D} = \{(\mathbf{x}_1, [y_1^l, y_1^u]), \dots, (\mathbf{x}_n, [y_n^l, y_n^u])\}$. Here, $[y_i^l, y_i^u]$, $i = 1, \dots, n$ are the interval-valued observations; $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector. Let Y be a target random variable of a regression model. We aim to train a the model such that it produces a target conditional density function $\rho(y | \mathbf{x}) = \Pr\{Y = y | \mathbf{x}\}$ for a new feature vector $\mathbf{x}$. It should be noted that some of the intervals can have infinite left or right bounds; for example, $y_i^l \rightarrow -\infty$ and $y_j^u \rightarrow \infty$.

When intervals are rather small, the task can be solved by many methods, for example by using the mean values of the intervals and their variance. However, the intervals can be very large. In this case, most methods provide unsatisfactory results.

Consider a partition of the y -axis into discrete intervals:

$$(-\infty, \infty) = [-\infty, z_1] \cup [z_1, z_2] \cup \dots \cup [z_{T-2}, z_{T-1}] \cup [z_{T-1}, \infty), \quad (4)$$

such that

$$-\infty < z_1 < z_2 < \dots < z_{T-1} < \infty. \quad (5)$$

The partition divides the y -axis into T intervals denoted as $\tau_i = [z_{i-1}, z_i]$ for $i = 1, \dots, T$. The target conditional density function $\rho(y | \mathbf{x})$ in this case will be step-wise and it can be represented as a discrete probability distribution $\mathbf{p}(y | \mathbf{x}) = (p_1(\mathbf{x}), \dots, p_T(\mathbf{x}))$, where $p_i(\mathbf{x})$ is the probability that $y \in \tau_i$.

Suppose that the i -th interval $[y_i^l, y_i^u]$ covers m_i intervals $\tau_{i_1}, \dots, \tau_{i_1+m_i-1}$. Denote a set of the corresponding indices $\{i_1, \dots, i_1 + m_i - 1\}$ as J_i .

Let us define the probability $\pi_j^{(i)}$ that an event corresponding to the i -th feature vector is observed in the interval τ_j . This generates a probability distribution over intervals $\pi^{(i)} = (\pi_1^{(i)}, \dots, \pi_T^{(i)})$, $i = 1, \dots, n$, such that there holds

$$\pi_j^{(i)} = \begin{cases} 0, & j \notin J_i, \\ [0, 1], & j \in J_i. \end{cases} \quad (6)$$

The above means that we do not know the precise probabilities over intervals with indices from J_i . However, we know that there holds

$$\sum_{j=1}^T \pi_j^{(i)} = 1. \quad (7)$$

Since the probabilities $\pi_j^{(i)}$, $j \in J_i$ are imprecise, then there is a set $\mathcal{R}^{(i)}$ of possible distributions $\mathfrak{B}^{(i)}$ for the i -th feature vector.

Let us consider the *classification task* with T classes such that the k -th class corresponds to the k -th interval τ_k . Then, the Nadaraya–Watson kernel regression can be applied to find the class probability distribution $\mathbf{p}(\mathbf{x}) = (p_1(\mathbf{x}), \dots, p_T(\mathbf{x}))$ of a new instance $\mathbf{x}$. Since a part of probabilities $\pi_k^{(i)}$ are interval-valued, then it is obvious that the predicted class probabilities $p_1, \dots, p_T$ are also interval-valued. They are determined as follows:

$$p_k(\mathbf{x}) = \sum_{i=1}^n a(\mathbf{x}, \mathbf{x}_i, \mathbf{W}) \pi_k^{(i)}, \quad k = 1, \dots, T, \quad (8)$$

where $a(\mathbf{x}, \mathbf{x}_i, \mathbf{W})$ is an attention weight expressed through kernels $K(\mathbf{x}, \mathbf{x}_i, \mathbf{w})$ (see (2)), $i = 1, \dots, n$; $\mathbf{W}$ is a set of trainable parameters.

The attention weights satisfy the condition

$$\sum_{i=1}^n a(\mathbf{x}, \mathbf{x}_i, \mathbf{W}) = 1. \quad (9)$$

The above expressions can be extended to a more general form. Introduce a matrix $\mathbf{K} \in \mathbb{R}^{n \times d}$ of the input keys (n feature vectors from $\mathcal{D}$) and the matrix $\mathbf{Q} \in \mathbb{R}^{n \times d}$ of the input queries (the same feature vectors). The matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ of attention weights can be computed as [15,16]:

$$\mathbf{A} = \frac{(\mathbf{Q}\mathbf{W}_Q)(\mathbf{K}\mathbf{W}_K)}{\sqrt{d}}. \quad (10)$$

where $\mathbf{W}_K \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_Q \in \mathbb{R}^{d \times d}$ are trainable matrices.

Let $\mathbf{a}_k = (a_{k1}, \dots, a_{kn})$ be the k -th row of the matrix $\mathbf{A}$ and $\pi_k = (\pi_k^{(1)}, \dots, \pi_k^{(n)})^T \in \mathbb{R}^n$ be the vector of probabilities of the k -th interval over all objects from $\mathcal{D}$. Then, the k -th element $p_k(\mathbf{x})$ of the probability distribution $\mathbf{p}(y | \mathbf{x})$ can be regarded as the average of the k -th interval probabilities with weights $\mathbf{a}_k$ and can be written as follows:

$$p_k(\mathbf{x}) = \mathbf{a}_k \pi_k, \quad k = 1, \dots, T. \quad (11)$$

This implies that we have to solve the classification task and find matrices $\mathbf{W}_Q$ and $\mathbf{W}_K$. The loss function for solving the task is the log-likelihood function defined as

$$\mathcal{L}(\mathbf{p} | \mathbf{W}_Q, \mathbf{W}_K) = \sum_{i=1}^n l(\mathbf{p}(\mathbf{x}_i)) = - \sum_{i=1}^n \log \left(\sum_{j \in J_i} p_j(\mathbf{x}_i) \right). \quad (12)$$

We use also regularization for matrices $\mathbf{W}_Q, \mathbf{W}_K$:

$$L_2 = \eta (\|\mathbf{W}_Q\|^2 + \|\mathbf{W}_K\|^2). \quad (13)$$

One of the problems that can occur during training with the above loss function is a fragmented probability distribution, i.e., when very small class probabilities are interspersed between large class probabilities. To solve this problem, we add the following Laplacian smoothness regularization

$$L_{Lap} = \frac{\lambda_1}{2} \sum_{i=1}^n \sum_{j=1}^n v_{ij} \|\mathbf{p}(\mathbf{x}_i) - \mathbf{p}(\mathbf{x}_j)\|^2, \quad (14)$$

where v_{ij} is a parameter of similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$, for example the kernels $K(\mathbf{x}_i, \mathbf{x}_j)$; λ_1 is the hyperparameter.

Another regularization penalizes large changes between the probabilities of adjacent intervals:

$$L_{Adj} = \lambda_2 \sum_{i=1}^n \sum_{j=1}^{T-1} |p_{j+1}(\mathbf{x}_i) - p_j(\mathbf{x}_i)|. \quad (15)$$

The main difficulty of solving the task is to handle the interval-valued probabilities $\pi_j^{(i)}$. We propose two approaches.

The idea is to average the loss over all possible probability distributions which are produced by the interval-valued probabilities. If the set of all probability distributions produced by intervals $[0, 1]$ in (6) for all objects is denoted as $\mathcal{P} = \mathcal{P}^{(1)} \cup \dots \cup \mathcal{P}^{(n)}$, where

$\mathcal{P}^{(i)}$ is the set of all class probability distributions for the i -th object, then the optimization problem for training the classification model can be written as follows:

$$\min_{\mathbf{W}_Q, \mathbf{W}_K} \mathbb{E}_{(\pi^{(1)}, \dots, \pi^{(n)}) \in \mathcal{P}} \mathcal{L}(\mathbf{p} \mid \mathbf{W}_Q, \mathbf{W}_K). \quad (16)$$

The next question is how to find the expectation in the optimization problem. The first approach is to use the Monte Carlo sampling scheme.

4.1. The Monte Carlo Sampling Approach

The Monte Carlo sampling method [12] is employed to approximate complex quantities of interest by generating a large number of random samples. For each instance i , M discrete probability distributions $\tilde{\pi}^{(i,1)}, \dots, \tilde{\pi}^{(i,M)}$ are generated from its associated set $\mathcal{P}^{(i)}$, where M is a predefined hyperparameter. These distributions are randomly sampled, typically using the Dirichlet distribution [64]. This procedure yields a set of n matrices $\mathbf{S}^{(i)} \in \mathbb{R}^{M \times T}$ for $i = 1, \dots, n$. Each matrix $\mathbf{S}^{(i)}$ contains M sampled probability distributions from $\mathcal{P}^{(i)}$. The dimensionality of the unit simplex from which these distributions are drawn is determined by the number of intervals $\tau_j, j \in J_i$ included in the interval $[y_i^l, y_i^u]$ and is equal to $T - m_i$. Elements of each $\tilde{\pi}^{(i,k)}, k = 1, \dots, M$, with indices $j \notin J_i$ are equal to 0.

The attention-weighted probability matrix for the i -th instance is subsequently computed as:

$$\mathbf{P}^{(i)} = \mathbf{A} \mathbf{S}^{(i)} \in \mathbb{R}^{M \times T}, \quad (17)$$

where $\mathbf{A}$ is a weight matrix defined above.

The output $\mathbf{P}^{(i)}$ provides a refined estimate of event probabilities across M Monte Carlo generations and T time intervals. Each element $p_k^{(m)}(\mathbf{x}_t)$, representing the probability for class k and instance $\mathbf{x}_t$ in the m -th generation, is computed as a weighted sum:

$$p_k^{(m)}(\mathbf{x}_t) = \sum_{i=1}^n a_{k,i} \mathbf{S}_{m,t}^{(i)}. \quad (18)$$

Here, $a_{k,i}$ is an element of the weight matrix $\mathbf{A}$; $\mathbf{S}_{m,t}^{(i)}$ is an element of the matrix $\mathbf{S}^{(i)}$ corresponding to the m -th generation and the t -th interval; $p_k^{(m)}(\mathbf{x}_t)$ is the k -th element of the class probability distribution $\mathbf{p}(\mathbf{x}_t)$ for the t -th instance obtained through the m -th generation of the probability distribution from the set $\mathcal{P}^{(t)}$.

The model operates by minimizing a loss function that is averaged over multiple Monte Carlo generations. The core learning objective is to find trainable parameters $\mathbf{W}_Q$ and $\mathbf{W}_K$ producing the class probability distributions $\mathbf{p}^{(m)}(\mathbf{x}_i), m = 1, \dots, M, i = 1, \dots, n$. The loss function can be written as follows:

$$\mathcal{L}(\mathbf{p} \mid \mathbf{W}_Q, \mathbf{W}_K) = \sum_{i=1}^n \sum_{m=1}^M l(\mathbf{p}^{(m)}(\mathbf{x}_i)) + L_{Lap} + L_{Adj} + L_2. \quad (19)$$

The loss function $\mathcal{L}(\mathbf{p} \mid \mathbf{W}_Q, \mathbf{W}_K)$ is minimized to learn the parameters of the attention mechanism. Learning is designed as a two-stage optimization process. Initially, the model learns the attention weights. After fixing the optimized weights $\mathbf{W}_Q, \mathbf{W}_K$, the final class probability distribution is further refined. This is done by initializing the probabilities with the averaged values from the Monte Carlo generations and then optimizing them directly using the per-generation loss function (12). This second-stage optimization enhances the model's predictive accuracy.

The training procedure is outlined in Algorithm 1.

Algorithm 1 The training algorithm for the Monte Carlo sampling approach**Require:** Training set $\mathcal{D}$; hyperparameter M ; the number of epochs E **Ensure:** Matrices $\mathbf{W}_Q, \mathbf{W}_K$; probability distributions $\mathbf{p}(\mathbf{x})$

- 1: Initialize randomly matrices $\mathbf{W}_Q, \mathbf{W}_K$ and compute the attention matrix $\mathbf{A}$ using (10)
- 2: **while** Number of epochs $\leq E$ **do**
- 3: Generate probability distributions $\tilde{\pi}^{(i,1)}, \dots, \tilde{\pi}^{(i,M)}$ from $\mathcal{P}^{(i)}$ by using the Dirichlet distribution
- 4: Form the matrix $\mathbf{S}^{(i)}, i = 1, \dots, n$
- 5: Compute the matrix $\mathbf{P}^{(i)}$ from (17), $i = 1, \dots, N$
- 6: Learn $\mathbf{W}_Q, \mathbf{W}_K$ using the loss function (19)
- 7: **end while**
- 8: Fine-tune the final probability estimates by taking the mean of the generated distributions $\tilde{\pi}^{(i,1)}, \dots, \tilde{\pi}^{(i,M)}$ from the last epoch and then optimizing them directly using the loss function (19)

During inference, the class probability distribution $\mathbf{p}(\mathbf{x}) = (p_1(\mathbf{x}), \dots, p_T(\mathbf{x}))$ for a new instance $\mathbf{x}$ is computed using the trained attention mechanism from (11). The attention weights $\mathbf{W}_Q, \mathbf{W}_K$ are fixed. The probability vectors $\pi_1, \dots, \pi_T$ in that equation are replaced with the averaged Monte Carlo distributions

$$\tilde{\pi}^{(i)} = M^{-1} \sum_{m=1}^M \tilde{\pi}^{(i,m)}, \quad (20)$$

generated for the training instances in the final epoch of Algorithm 1.

4.2. The Second Approach

The second proposed approach is principally distinguished by its joint learning algorithm, in which the class probability distributions and the attention weights are optimized simultaneously. This integrated approach circumvents the necessity for the Monte Carlo generation procedure of the probability distributions $\tilde{\pi}^{(i,1)}, \dots, \tilde{\pi}^{(i,M)}$, thereby streamlining the training process. The implementation of this learning paradigm initializes the probability logits to zero. At every epoch, the class probabilities $\pi_k^{(i)}$ are derived by applying the softmax operation to these logits, a step that intrinsically ensures the normalization condition $\sum_{k=1}^T \tilde{\pi}_k^{(i)} = 1$. These learned parameters $\tilde{\pi}_k^{(i)}$ serve as precise analogues to the interval-valued probabilities $\pi_k^{(i)}$ found in the previous model, with the key distinction that their values are directly learned from the data rather than generated stochastically.

The class probability $p_j(\mathbf{x}_i)$ for the interval $j \in J_i$ is defined according to the approach established in (11). However, a key distinction is that the probability vector π_k in (11) is now the precise $\tilde{\pi}_k$ learned parameter $\tilde{\pi}_k^{(i)}$ rather than a generated quantity. The overall objective for training the model remains the minimization of the total loss from (19). The loss function is now defined with an added regularization term:

$$\begin{aligned} \mathcal{L}(\tilde{\mathbf{p}} \mid \mathbf{W}_Q, \mathbf{W}_K, \tilde{\pi}_1, \dots, \tilde{\pi}_T) = & - \sum_{i=1}^n \log \left(\sum_{j \in J_i} \tilde{p}_j(\mathbf{x}_i) \right) \\ & + L_{caut} + L_{Lap} + L_{Adj} + L_2. \end{aligned} \quad (21)$$

Here, $\tilde{\mathbf{p}}(\mathbf{x}_i) = (\tilde{p}_1(\mathbf{x}_i), \dots, \tilde{p}_T(\mathbf{x}_i))$ is the precise predicted probability distribution for the i -th instance. The regularization loss function L_{caut} is defined as:

$$L_{caut} = \gamma \sum_{i=1}^n \sum_{k=1}^T \tilde{\pi}_k^{(i)} \cdot \log(\tilde{\pi}_k^{(i)}), \quad (22)$$

where γ is a hyperparameter controlling the regularization strength.

The new regularization component L_{caut} represents the negative entropy of the learned distribution $\tilde{\pi}^{(i)}$. This regularization term is crucial for the stability of the model. By penalizing low entropy, i.e., distributions that are overly confident or “peaky”, it prevents the model from making overly optimistic and potentially overfitted decisions during the joint optimization of the attention weights and the probability parameters. This same entropy regularization is applied during the final fine-tuning step in the model to ensure a consistent and stable optimization process when directly adjusting the probability values.

The regularization L_{caut} can be explained from a different perspective. By considering each interval-valued target as representing “incomplete knowledge” of a continuous quantity, we face the question of which probability distribution to select from the set of possibilities. A “diffuse” prediction is generally more appropriate, as it better captures our epistemic uncertainty regarding the exact value within the interval (we lack information to specify a precise distribution). In contrast, a “concentrated” prediction relies on strong, often unjustified, assumptions about the internal structure of the interval. Thus, in the absence of compelling evidence for concentration at specific points, the “diffuse” (high-entropy) prediction is preferable. This approach is more methodologically sound, guarding against false overconfidence. Consequently, we minimize the negative entropy to promote higher entropy in our predictions.

An implementation of training the model is presented as Algorithm 2. Logits of $\tilde{\pi}_k^{(i)}$ in the algorithm are initialized as follows: the matrix $\mathbf{S}^{(i)}$ is sampled in the same way as in the Monte Carlo approach, but with a larger number of samples (for example, 1000). Then, it is averaged and the logits of $\tilde{\pi}_k^{(i)}$ are taken from these averaged values.

Algorithm 2 The training algorithm for the second approach

Require: Training set $\mathcal{D}$; the number of epochs E

Ensure: Matrices $\mathbf{W}_Q, \mathbf{W}_K$; probability distributions $\tilde{\mathbf{p}}(\mathbf{x})$

- 1: Initialize randomly matrices $\mathbf{W}_Q, \mathbf{W}_K$ and compute the matrix $\mathbf{A}$ using (10)
 - 2: Initialize logits of $\tilde{\pi}_k^{(i)}$ as zeros
 - 3: **while** Number of epochs $\leq E$ **do**
 - 4: Compute $\tilde{\pi}_k^{(i)}$ using the softmax operation applied to logits
 - 5: Compute the probability distribution $\tilde{\mathbf{p}}(\mathbf{x}_i)$ using $\mathbf{A}$ and $\tilde{\pi}_k^{(i)}$
 - 6: Learn $\mathbf{W}_Q, \mathbf{W}_K$ and $\tilde{\pi}_k^{(i)}$ using the loss functions (21)
 - 7: Compute logits of $\tilde{\pi}_k^{(i)}$
 - 8: **end while**
-

The important question is how this approach aligns with its decision-theoretic justification. The problem is that we are essentially selecting a single distribution $\pi^{(i)}$ for each instance from the set of distributions $\mathcal{P}^{(i)}$ using the proposed algorithm, and this selection is not aligned with the well-known decision strategies, such as a robust strategy or a strategy for minimizing expected loss. Since the goal of model training is to obtain a probability distribution $\mathbf{p}(\mathbf{x})$ over intervals for a new instance, the core idea of the approach is that a probability distribution $\pi^{(i)}$ is “optimal” if it minimizes the loss function on the test data. Namely the test data define whether the selected distributions are “optimal” or not.

Let us look at the proposed algorithm from another point of view. In fact, we move away from the decision-theoretic concept of the imprecision of probability distributions on elementary intervals in an explicit form and instead treat it analogously to, for example, the weights of connections in neural networks, which are parameters that we do not know a priori but, through updating and adjustment, attempt to tune by minimizing a loss function so that they yield predictions close to those observed in the training sample. Just as a neural network predicts

the class probabilities using an entropy-based loss function, our model does the same, except that instead of connection weights it utilizes probability distributions derived from individual interval-valued observations. After training, the results are evaluated on test data, similarly to how a neural network with weights is assessed. Here, imprecision serves as a way to represent sets of distributions and to enable the development of new models by constraining these sets of distributions within certain imprecise models [7].

Suppose that the test set consists of N pairs $(\mathbf{x}_i^*, [y_i^{l*}, y_i^{u*}])$, $i = 1, \dots, N$. We also suppose that, for every instance from the test set, the discrete probability distribution $\tilde{\mathbf{p}}(\mathbf{x}_i^*)$ over intervals $\tau_1, \dots, \tau_T$ can be predicted by using the proposed model. The loss function in this case is

$$\mathcal{L}(\tilde{\mathbf{p}}) = - \sum_{i=1}^N \log \left(\sum_{j \in J_i^*} \tilde{p}_j(\mathbf{x}_i^*) \right). \quad (23)$$

where J_i^* is a set of all intervals τ_j that are included into the interval $[y_i^{l*}, y_i^{u*}]$.

4.3. Modifications of the Loss Function

One of the difficulties in training the models is that the number of classes in the obtained classification task may be larger than the number of instances. To overcome this difficulty, we propose to use modifications of the loss function (12) considering the neighboring $2k$ intervals around each interval (k intervals on the right and k intervals on the left) and summing their probabilities. This approach has been used in [11] to improve model performance. The term in the loss function (12)

$$- \sum_{i=1}^n \log \left(\sum_{j \in J_i} p_j(\mathbf{x}_i) \right), \quad (24)$$

is replaced with the term

$$- \sum_{i=1}^n \log \left(\sum_{j \in J_i} \sum_{l=j-k}^{j+1+k} p_l(\mathbf{x}_i) \right). \quad (25)$$

Another modification is to weigh all probabilities around each interval τ_i with weights ψ formed by the Gaussian distribution centered on $\bar{\tau}_j$ of the interval τ_j (i.e., with an expectation/mean of $\bar{\tau}_j$) and the standard deviation ϑ (a hyperparameter). The corresponding term of the loss function becomes

$$- \sum_{i=1}^n \log \left(\sum_{j \in J_i} \sum_{l=1}^T \psi(\bar{\tau}_l, \bar{\tau}_j, \vartheta^2) \cdot p_l(\mathbf{x}_i) \right). \quad (26)$$

5. Numerical Experiments

The codes implementing the proposed models are publicly available at: <https://github.com/NTAILab/iprob> (assessed on 28 January 2026).

We study the proposed model on synthetic datasets: *Linear*, *Spiral*, and *Power*. They are generated in accordance with the following functions of the same name.

1. *Linear*: Examples are generated into two steps. At the first step, precise values of y are generated using a linear function as follows:

$$y = 2 \cdot x^{(1)} + 4 \cdot x^{(2)} + 8 \cdot x^{(3)} + \epsilon. \quad (27)$$

Here, $x^{(i)}$ is the i -th feature of the vector $\mathbf{x}$ uniformly generated from $[0, 1]$, i.e., $x^{(i)} \sim \mathcal{U}(0, 1)$; $\epsilon \sim \mathcal{N}(\mu, \sigma)$ is noise generated in accordance with the normal distribution with the expectation μ and the standard deviation σ , where $\mu = 0$, $\sigma = 0.2$. At the

second step, target intervals $[y^l, y^u]$ are generated for every y , obtained at the first step, in accordance with the normal distribution $\mathcal{N}(\mu = B, \sigma = 1)$, where the parameter B is changed from 1 to 8 for studying how the performance of the model depends on the averaged interval width.

2. *Spiral*: Feature vectors $\mathbf{x} \in \mathbb{R}^5$ are generated by using the Archimedean spiral as follows:

$$\mathbf{x} = (t \sin(t), t \cos(t), \dots, t \sin(t \cdot d/2), t \cos(t \cdot d/2)), \quad (28)$$

for even d values, and

$$\mathbf{x} = (t \sin(t), t \cos(t), \dots, t \sin(t \cdot \lceil d/2 \rceil)). \quad (29)$$

for odd d values. Precise values of y are generated as $y = at + b$, where $a \sim \mathcal{U}(0.6, 1)$, $b \sim \mathcal{U}(0.6, 1)$, $t \sim \mathcal{U}(1, 4)$. Target intervals are generated in the same way as was done in the Linear dataset, i.e., in accordance with the normal distribution $\mathcal{N}(\mu = B, \sigma = 1)$.

3. *Power*: Feature vectors $\mathbf{x} \in \mathbb{R}^5$ are generated by using the following representation:

$$\mathbf{x} = (t^{1/\sqrt{d}}, t^{2/\sqrt{d}}, \dots, t^{d/\sqrt{d}}). \quad (30)$$

Target values are computed as:

$$y = a \cdot \exp\left(-\frac{(t-s)^2}{b}\right). \quad (31)$$

Parameters of generation are $a \sim \mathcal{U}(9, 10)$, $b \sim \mathcal{U}(0.5, 1)$, $t \sim \mathcal{U}(0, 7)$; the parameter s is 3.5.

For the Monte Carlo sampling approach, the number of samples is 10 and the number of epochs for training is 3000, of which 1000 are for training the model and 2000 are for fine-tuning $\tilde{\pi}^{(i,1)}, \dots, \tilde{\pi}^{(i,M)}$. For the second approach, the number of epochs for training is 3000.

During training, a method similar to cross-validation is used for selecting Query and Key. The randomly shuffled train set is divided into $1/br$ parts, where br is the batch rate. The first part is selected as Query and the remaining parts as Key. Then the second part as Query and the rest as Key, and so on until all parts have served as Query. After this, the epoch ends and the train set is randomly shuffled again. The smaller the batch rate, the less noisy the epoch-wise train graph.

For all experiments related to interval widths, the batch rate is 0.05. The number of instances in this task is 200. Interval widths B range from 1 to 8. The number of examples in the test set is also 200.

In order to study how the performance of the model depends on the sizes of the training set, the number of instances n is taken from 50 to 300 and the batch rate is set to 0.1. The interval width B for these experiments is set to 4. The number of instances in the test set is 300.

It is also supposed that $\pi_j^{(i)}$ in (6) is equal to a small value ε when $j \notin J_i$ to avoid infinite values calculating $\log(\pi_j^{(i)})$.

Hyperparameter optimization was carried out with the Optuna framework [65], which relies on Bayesian optimization. For each model, a customized search space is defined as follows: the parameter k in Equation (25) changes in the interval $[0, 10]$, the parameter ϑ is in $[2, q]$, where q is the number of intervals τ_i . The coefficients of the regularization terms γ , η , λ_1 , and λ_2 change in intervals $[0.00001, 0.001]$, $[0.00001, 0.01]$, $[0.001, 0.1]$, and $[0.001, 0.2]$, respectively.

The following measures for studying and estimating the model under different conditions are applied:

- **Brier Score (BS):** Mean squared error for the probabilistic forecast defined for n instances as:

$$BS = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1, \dots, T} (\tilde{p}_j(\mathbf{x}_i) - I[j \in J_i]) \right)^2. \quad (32)$$

- **LogLoss (Logarithmic Score):** Based on the logarithm of the likelihood:

$$LL = -\frac{1}{n} \sum_{i=1}^n \left(\log \left(\sum_{j \in J_i} \tilde{p}_j(\mathbf{x}_i) \right) + \log \left(1 - \sum_{j \notin J_i} \tilde{p}_j(\mathbf{x}_i) \right) \right) \quad (33)$$

The experimental design employed a nested cross-validation approach to ensure robust performance estimation. The outer loop consists of four iterations of five-fold stratified cross-validation, preserving the distribution of intervals in each fold, with shuffling enabled and random seeds fixed for reproducibility. Within each training fold of the outer loop, an additional three-fold stratified cross-validation is performed for hyperparameter optimization.

The interval width parameter k , which determines the granularity of predictions for interval-valued instances, is varied between 3 and 10 across different training instances.

5.1. Study of the Model Properties Using Synthetic Data

5.1.1. Linear Dataset

Figure 1 illustrates examples of probability distributions of Y obtained for testing instances using the Monte Carlo approach (the top row) and the second approach (the bottom row). Lower and upper bounds for intervals $[y_i^l, y_i^u]$ of the instance targets are depicted by vertical dotted lines. From the figure, it is evident that the Monte Carlo approach yields sufficiently “noisy” results, which is related to the limited number of Dirichlet distribution generations over the intervals. At the same time, increasing the number of generations significantly extends the model training time. The second approach is largely free from noise in predicting the probability distribution function over the intervals but predicts “sharper” distribution functions, which is due to the insufficient influence of the regularization L_{caut} . However, intensifying it through the parameter γ weakens the main loss function and leads to a shift in the predicted probability distribution.

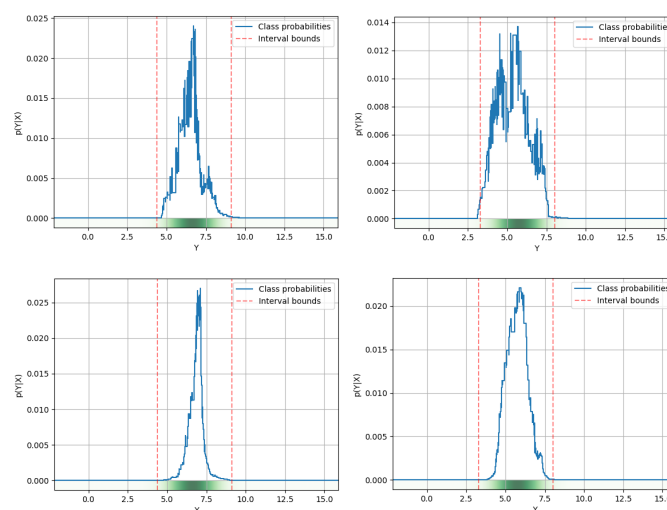


Figure 1. Linear dataset: Probability distributions of the predicted target Y obtained using the Monte Carlo approach (the top row) and the second approach (the bottom row).

Figure 2 illustrates how the performance of the models depends on the average interval length B (top graphs) and on the number n of training instances (bottom graphs) when they are trained on the linear dataset. The Brier score and Log loss are used to characterize the Monte Carlo approach (solid line) and the second approach (dashed line). The top graphs in the figure show that all accuracy measures for both models improve as the interval length B increases. At first glance, this seems paradoxical since uncertainty of the observations increases, yet accuracy improves. This can be explained if we return to the Nadaraya–Watson kernel regression in (8). When intervals $[y_i^l, y_i^u]$ are large, many other intervals cover every subinterval $\tau_k, k = 1, \dots, T$. This implies that the number of non-zero probabilities $\pi_k^{(i)}$ in the sum (8) for every k is rather large, providing more data to train each model. Another reason for the improvement in accuracy is that testing instances also have large target intervals, which simplifies the task of having the predicted probability distributions fall into these intervals. The bottom graphs in Figure 2 show how the Brier score and Log loss of the models depend on the number n of training instances. The graphs reveal that all accuracy measures for both models improve as the number of training instances increases. It is interesting to note that the Monte Carlo approach outperforms the second approach when the number of instances is less than 200. However, this relationship changes when the number of instances exceeds 200.

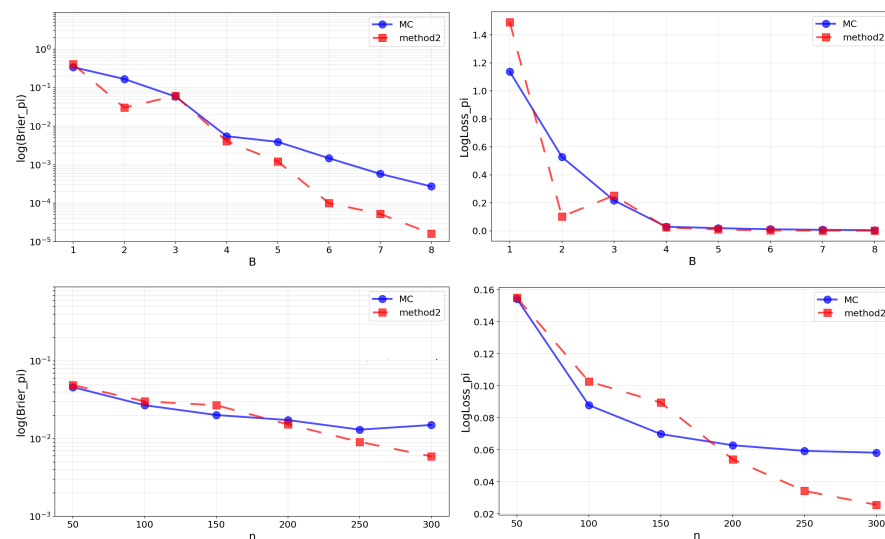


Figure 2. Linear dataset: Comparison of loss functions (the Brier score loss and the Log loss) for the Monte Carlo approach (solid line) and the second approach (dashed line) as functions of the average interval length B (**top graphs**) and the interval numbers n (**bottom graphs**).

Figure 3 illustrates training and testing loss functions: the Brier loss (left graphs) and the Log loss (right graphs) were calculated using the Monte Carlo approach (the top row) and the second approach (the bottom row). It is interesting to note that the loss functions for the Monte Carlo approach trained during the preliminary steps 1–7 of Algorithm 1, when matrices $\mathbf{W}_Q$ and $\mathbf{W}_K$ are being learned, hardly change. However, applying fine-tuning sharply reduces the losses for both training and testing. In contrast to the Monte Carlo approach, the loss functions in the second approach decrease smoothly as the number of training and testing epochs increases.

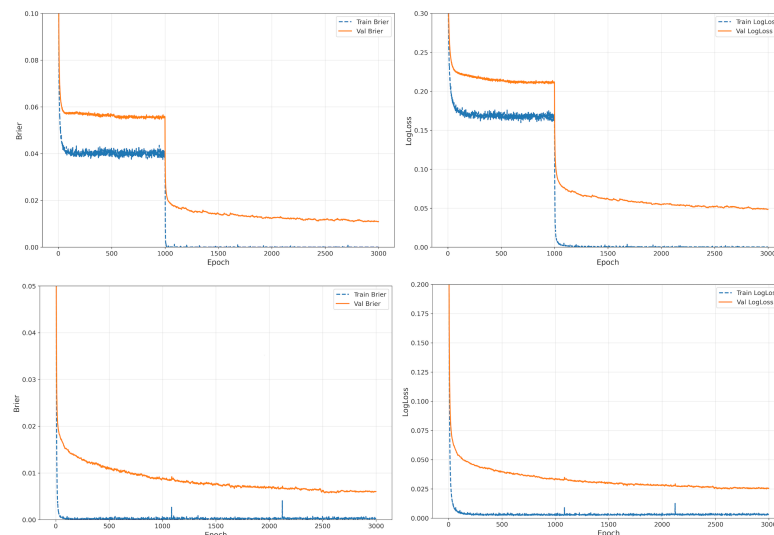


Figure 3. Linear dataset: Training and testing loss functions (Brier, LogLoss) for the Monte Carlo approach (**the top row**) and the second approach (**the bottom row**) by 300 intervals.

5.1.2. Spiral Dataset

The same analysis is applied to the case of the spiral dataset. Figure 4 shows the probability distribution functions of Y for the testing instances, obtained using the Monte Carlo approach (top row) and the second approach (bottom row). Figure 5 illustrates model performance as a function of the average interval length B (top graphs) and the number of training instances n (bottom graphs). In contrast to the trends observed for the linear dataset, the second approach yields better results across all considered values of B and n .

Figure 6 illustrates the training and testing loss functions. As shown, their behavior for the spiral dataset closely resembles that of the linear dataset.

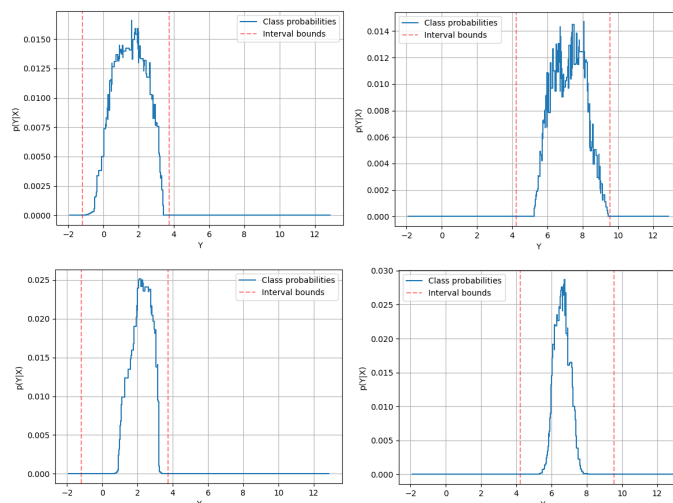


Figure 4. Spiral dataset: Probability distributions of the predicted target Y obtained using the Monte Carlo approach (**the top row**) and the second approach (**the bottom row**).

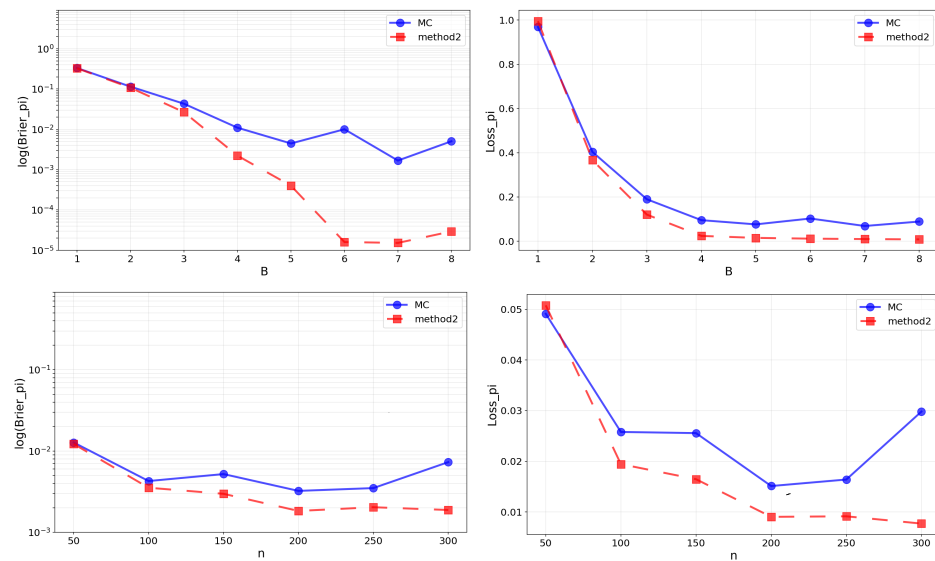


Figure 5. Spiral dataset: Comparison of loss functions (the Brier score loss and the Log loss) for the Monte Carlo approach (solid line) and the second approach (dashed line) as functions of the average interval length B (top graphs) and the interval numbers n (bottom graphs).

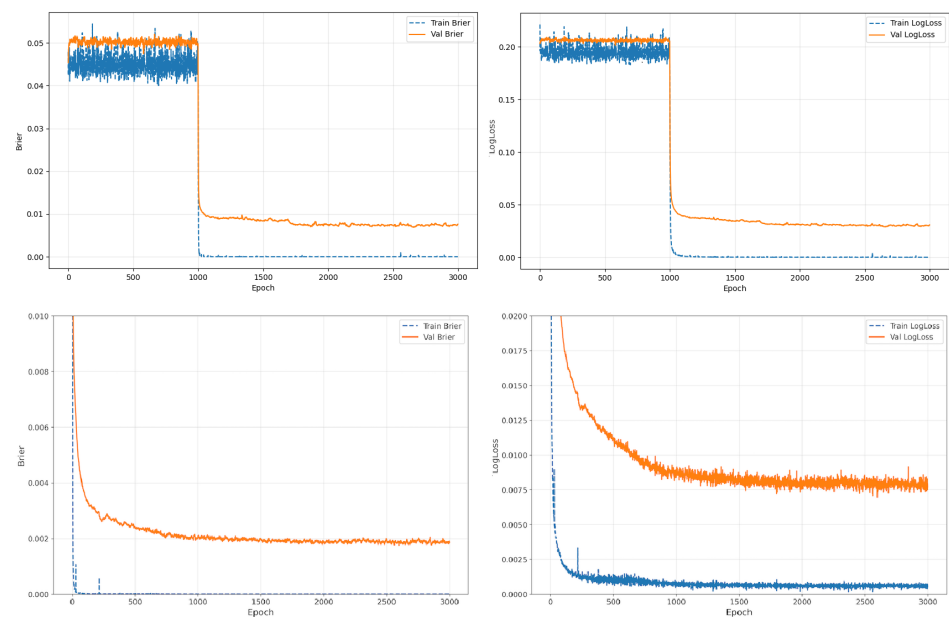


Figure 6. Spiral dataset: Training and testing loss functions (Brier, LogLoss) for the Monte Carlo approach (the top row) and the second approach (the bottom row) by 300 intervals.

5.1.3. Power Dataset

Similar results for the power dataset are represented in Figures 7–9, where one can observe that the models are perfectly trained on the dataset and provide accurate predictions.

Analyzing the above numerical experiments with synthetic data, we can conclude that the second approach outperforms the Monte Carlo approach in most cases. However, when the number of training examples is small, the Monte Carlo approach becomes preferable, despite its substantially higher computational cost.

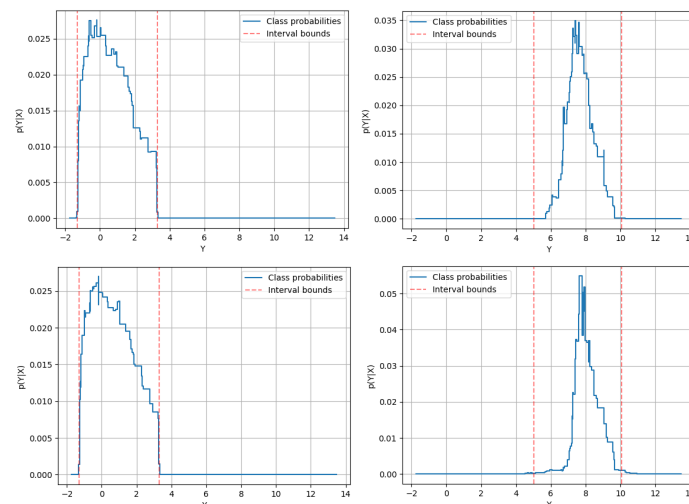


Figure 7. Power dataset: Probability distributions of the predicted target Y obtained using the Monte Carlo approach (the top row) and the second approach (the bottom row).

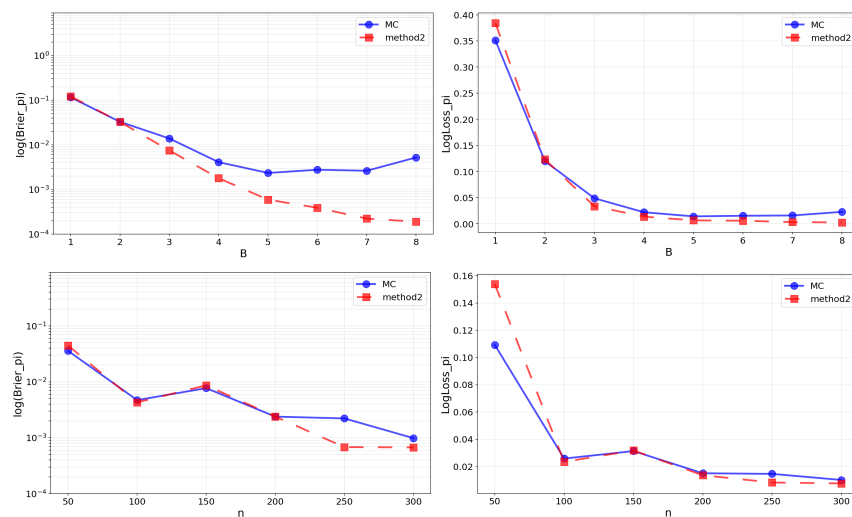


Figure 8. Power dataset: Comparison of loss functions (the Brier score loss and the Log loss) for the Monte Carlo approach (solid line) and the second approach (dashed line) as functions of the average interval length B (top graphs) and the interval numbers n (bottom graphs).

The Monte Carlo approach produces more robust, smoother probability distributions but is inherently “noisy” due to sampling limitations and its computational cost is high. In contrast, the second approach yields “sharper”, noise-free distributions but can be overconfident, as its regularization to promote caution (L_{caut}) is difficult to balance without degrading the primary learning objective.

In most experiments, predictive accuracy improves for both models as the target interval length increases. Predictive accuracy for both models improves with more training instances.

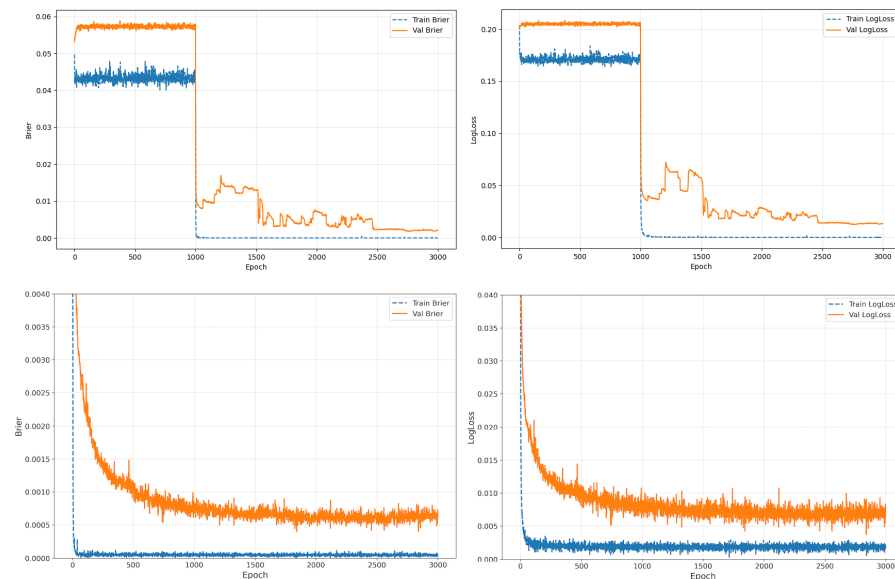


Figure 9. Power dataset: Training and testing loss functions (Brier, LogLoss) for the Monte Carlo approach (**the top row**) and the second approach (**the bottom row**) by 300 intervals.

There is no universally superior method. The choice depends on the data context. In particular, the Monte Carlo approach is preferable when dealing with smaller datasets, accepting higher computational cost and some output noise. The second approach is more efficient and provides cleaner predictions, becoming advantageous with larger datasets, though it requires careful tuning to avoid overly confident, potentially miscalibrated predictions.

Table 1 compares the performance of the models on three synthetic datasets (Linear, Spiral, Power) using two evaluation metrics, the Logarithmic score and the Brier score. The table clearly demonstrates that the M2 method provides more accurate probabilistic predictions than the Monte Carlo method across all tested synthetic datasets.

Table 1. Values of LogLoss and Brier score for models trained on synthetic datasets.

Dataset	Monte Carlo		M2	
	LogLoss	Brier	LogLoss	Brier
Linear	0.0581	0.0150	0.0256	0.0059
Spiral	0.0298	0.0073	0.0077	0.0019
Power	0.0102	0.0010	0.0076	0.0007

5.2. Real Data

The proposed models are studied by applying datasets that are taken from the open UCI Machine Learning Repository [66]. These datasets, with their numbers of features d and numbers of instances n , are given in Table 2. Detailed information can be found from the aforementioned data resources.

For each target value y_i in real data, the interval width h is generated from the normal distribution with the mean B and the standard deviation $B/10$. Bounds $[y_i^l, y_i^u]$ of intervals are computed as follows: $y_i^l = y_i - h/2$, $y_i^u = y_i + h/2$. The value B is taken as $0.4 \cdot (\max_{i=1, \dots, n} y_i - \min_{i=1, \dots, n} y_i)$.

Table 2. A brief introduction to the regression datasets.

Dataset	Abbreviation	d	n
Abalone	Abalone	8	4177
Concrete Compressive Strength	Concrete	8	1030
Wine red	RWine	11	1599
Wine white	WWine	11	4898
Average localization error	Ale	4	107
Yacht hydrodynamics	Yacht	6	308
Computer hardware	Hardware	10	209
Synchronous Machine	Synchronous	5	557

Values of the Logarithmic score and the Brier score for the real dataset are shown in Table 3. Unlike the synthetic data where M2 was consistently better, M2 with real data wins on some datasets (Abalone, Concrete, Yacht, RWine, WWine) by small margins. The Monte Carlo method wins on others (Ale, Hardware, Synchronous), often by large margins. The RWine dataset shows nearly identical performance. The Brier score patterns generally follow LogLoss, but with some interesting exceptions: in the Ale dataset, the LogLoss difference is huge (0.1323 vs 0.3177) but the Brier score difference is smaller (0.0369 vs. 0.0719). Similarly, in the Hardware and Synchronous datasets, Brier score differences are smaller than LogLoss differences.

Table 3. Values of LogLoss and Brier score for models trained on real datasets.

Dataset	Monte Carlo		M2	
	LogLoss	Brier	LogLoss	Brier
Abalone	0.2376	0.0543	0.2099	0.0517
Concrete	0.3610	0.1125	0.2302	0.0624
RWine	0.3538	0.0964	0.3611	0.0936
WWine	0.3213	0.0799	0.3093	0.0782
Ale	0.1323	0.0369	0.3177	0.0719
Yacht	0.6156	0.1430	0.2807	0.0953
Hardware	0.4172	0.0696	0.7304	0.0905
Synchronous	0.2739	0.0725	0.9939	0.0789

For illustrative purposes, we show the training and testing loss functions (Brier) of models trained on the Yacht (Figure 10) and Synchronous (Figure 11) datasets.

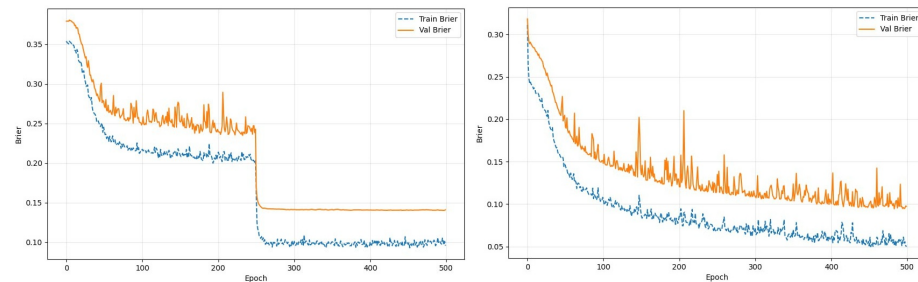


Figure 10. Yacht dataset: Training and testing Brier loss functions for the Monte Carlo approach (the left graph) and the second approach (the right graph).

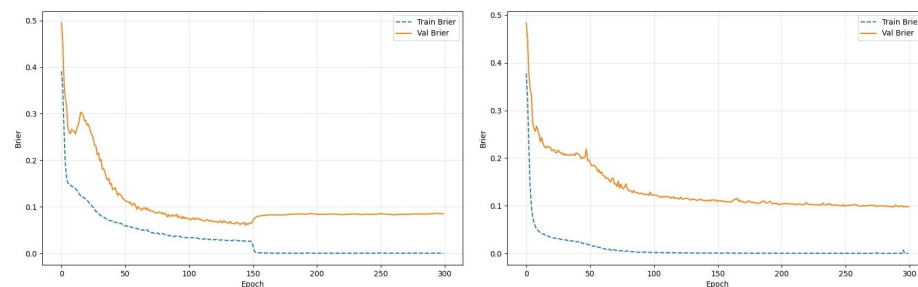


Figure 11. Synchronous dataset: Training and testing Brier loss functions for the Monte Carlo approach (the left graph) and the second approach (the right graph).

It can be seen from Figure 11, where loss functions are depicted for the Synchronous dataset, that the Monte Carlo algorithm outperforms the second approach. The main reason for this observation is that this dataset contains anomalous instances. Since the Monte Carlo algorithm is divided into two parts, where the first part does not learn the distribution π , then this algorithm is more robust to anomalies. The difference in the Brier loss and the LogLoss indicates that the model may be accurate in most instances but makes complete mistakes on some of them. The Brier loss is less sensitive to completely erroneous predictions than LogLoss. This is evidence of overconfident predictions.

6. Conclusions

This paper has introduced a class of attention-based models for interval-valued regression that fundamentally reinterprets the prediction of an interval as an imprecise classification task. In contrast to methods that treat the lower and upper bounds of an interval as independent regression targets or reduce the interval to a scalar midpoint, our approach preserves and directly models the intrinsic uncertainty encapsulated in the interval structure. The two proposed training algorithms offer distinct strategies for handling the imprecise distributions. The Monte Carlo (MC) approach approximates the expected loss by repeatedly sampling precise distributions from the feasible set defined by each interval, providing a straightforward but computationally intensive implementation. In contrast, the joint optimization (M2) approach directly learns a single, optimal precise distribution for each training instance alongside the model parameters, subject to the interval constraints and regularized to avoid overconfidence. This method is not only more computationally efficient but also, as evidenced by our experiments, tends to yield more accurate and stable predictions by avoiding the noise inherent in sampling.

The proposed models were rigorously evaluated on synthetic datasets (Linear, Spiral, Power) and eight real-world regression datasets. The numerical experiments reveal several key insights. First, the joint optimization approach (M2) consistently outperformed the Monte Carlo (MC) sampling method on synthetic data across all metrics (LogLoss and

Brier Score), demonstrating its superior predictive accuracy and efficiency in controlled settings. Furthermore, performance reliably improved with larger training set sizes for both methods. The real-data experiments presented a more nuanced picture. The MC approach, with its inherent sampling mechanism, can be more robust in the face of complex, noisy data distributions or when training data are limited, albeit at a higher computational cost. In contrast, the efficient M2 approach excels with larger datasets but may require careful tuning of its caution regularization (L_{caut}) to prevent overconfident predictions.

A distinctive positive feature of the employed attention mechanism is that it combines the probabilities of interval data by accounting for the distance between the corresponding feature vectors through attention weights. This makes the obtained results robust to anomalous observations. Robustness is ensured both with respect to feature vectors and interval-valued observations; specifically, anomalous feature vectors lead to substantially small attention weights that do not contribute significantly to the final probability distribution prediction, while anomalous target intervals generate probability distributions with regions that are attended to with nearly zero regions of the probability distributions for other interval observations. The attention mechanism also ensures the interpretability of the obtained results, in that the learned attention weights indicate which observations contributed the most to the prediction.

This work opens several directions for future research. The proposed algorithms are naturally extensible: the attention mechanism can be adapted to multi-head attention or integrated into deeper neural architectures to capture more complex feature interactions. Another idea is to consider a combination of different multi-head attention implementations, where some heads utilize probability distributions while others use feature vectors to implement self-attention. The core idea of treating imprecise observations as sets of distributions is general and could be applied to other domains with uncertain outputs. Furthermore, exploring connections with conformal prediction could yield models that provide rigorous, distribution-free uncertainty intervals alongside point predictions.

Author Contributions: Conceptualization, L.U. and A.K.; methodology, L.U. and V.M.; software, S.K. and A.K.; validation, S.K., V.M. and A.K.; formal analysis, A.K. and L.U.; investigation, L.U., A.K. and V.M.; resources, L.U. and V.M.; data curation, S.K. and V.M.; writing—original draft preparation, L.U.; writing—review and editing, A.K. and V.M.; visualization, S.K.; supervision, L.U.; project administration, V.M.; funding acquisition, L.U. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Russian Science Foundation under grant 25-11-00021.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in UCI Machine Learning Repository at <https://archive.ics.uci.edu/> (assessed on 28 January 2026).

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

References

1. Bentkowska, U. *Interval-Valued Methods in Classifications and Decisions*; Springer: Berlin/Heidelberg, Germany, 2019; Volume 378.
2. Billard, L.; Diday, E. *Symbolic Data Analysis: Conceptual Statistics and Data Mining*; John Wiley & Sons: Hoboken, NJ, USA, 2012.
3. Gawlikowski, J.; Tassi, C.; Ali, M.; Lee, J.; Humt, M.; Feng, J.; Kruspe, A.; Triebel, R.; Jung, P.; Roscher, R.; et al. A survey of uncertainty in deep neural networks. *Artif. Intell. Rev.* **2023**, *56*, 1513–1589.
4. Guo, Z.; Wan, Z.; Zhang, Q.; Zhao, X.; Chen, F.; Cho, J.H.; Zhang, Q.; Kaplan, L.M.; Jeong, D.H.; Jøsang, A. A survey on uncertainty reasoning and quantification for decision making: Belief theory meets deep learning. *arXiv* **2022**, arXiv:2206.05675.
5. Rodriguez, O. Shrinkage linear regression for symbolic interval-valued variables. *arXiv* **2024**, arXiv:2401.05471.

6. Shi, Y.; Wei, P.; Feng, K.; Feng, D.C.; Beer, M. A survey on machine learning approaches for uncertainty quantification of engineering systems. *Mach. Learn. Comput. Sci. Eng.* **2025**, *1*, 11.
7. Walley, P. *Statistical Reasoning with Imprecise Probabilities*; Chapman and Hall: London, UK, 1991.
8. Dempster, A. Upper and lower probabilities induced by a multi-valued mapping. *Ann. Math. Stat.* **1967**, *38*, 325–339.
9. Shafer, G. *A Mathematical Theory of Evidence*; Princeton University Press: Princeton, NJ, USA, 1976.
10. Alefeld, G.; Herzberger, J. *Introduction to Interval Computations*; Academic Press: New York, NY, USA, 1983.
11. Konstantinov, A.; Utkin, L.; Efremenko, V.; Muliukha, V.; Lukashin, A.; Verbova, N. Survival Analysis as Imprecise Classification with Trainable Kernels. *Mathematics* **2025**, *13*, 3040. <https://doi.org/https://doi.org/10.3390/math13183040>.
12. Kroese, D.; Taimre, T.; Botev, Z. *Handbook of Monte Carlo Methods*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
13. Nadaraya, E. On estimating regression. *Theory Probab. Its Appl.* **1964**, *9*, 141–142.
14. Watson, G. Smooth regression analysis. *Sankhya Indian J. Stat. Ser. A* **1964**, *26*, 359–372.
15. Luong, T.; Pham, H.; Manning, C. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics: Lisbon, Portugal, 2015; pp. 1412–1421. <https://doi.org/10.18653/v1/D15-1166>.
16. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
17. Ahn, J.; Peng, M.; Park, C.; Jeon, Y. A resampling approach for interval-valued data regression. *Stat. Anal. Data Min. Asa Data Sci. J.* **2012**, *5*, 336–348.
18. Neto, E.d.A.; De Carvalho, F.d.A. Constrained linear regression models for symbolic interval-valued variables. *Comput. Stat. Data Anal.* **2010**, *54*, 333–347.
19. Schollmeyer, G.; Augustin, T. On sharp identification regions for regression under interval data. In *Proceedings of the ISIPTA'13: Proceedings of the Eighth International Symposium on Imprecise Probability: Theories and Applications*; Cozman, F., Denœux, T., Destercke, S., Seidenfeld, T., Eds.; SIPTA: Compiègne, France, 2013; pp. 285–294.
20. Billard, L.; Diday, E. Regression analysis for interval-valued data. In *Proceedings of the Data Analysis, Classification and Related Methods: Proceedings of the Seventh Conference of the International Federation of Classification Societies*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 369–374.
21. Yeh, C.C.; Sun, Y.; Cutler, A. Tree-based regression for interval-valued data. *arXiv* **2022**, arXiv:2201.02948.
22. Gaona-Partida, P.; Yeh, C.C.; Sun, Y.; Cutler, A. Random forests regression for soft interval data. *Commun. Stat.-Simul. Comput.* **2024**, *54*, 4821–4840.
23. Ma, G.; Lu, J.; Fang, Z.; Liu, F.; Zhang, G. Multiview classification through learning from interval-valued data. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 9606–9620.
24. Ma, G.; Lu, J.; Liu, F.; Fang, Z.; Zhang, G. Domain adaptation with interval-valued observations: Theory and algorithms. *IEEE Trans. Fuzzy Syst.* **2024**, *32*, 3107–3120.
25. Neto, E.d.A.; De Carvalho, F.d.A. Centre and range method for fitting a linear regression model to symbolic interval data. *Comput. Stat. Data Anal.* **2008**, *52*, 1500–1515.
26. Yang, Z.; Lin, D.K.; Zhang, A. Interval-valued data prediction via regularized artificial neural network. *Neurocomputing* **2019**, *331*, 336–345.
27. Fagundes, R.; de Souza, R.; Cysneiros, F. Interval kernel regression. *Neurocomputing* **2014**, *128*, 371–388.
28. Lim, C. Interval-valued data regression using nonparametric additive models. *J. Korean Stat. Soc.* **2016**, *45*, 358–370.
29. Ji, A.B.; Li, Q.Q.; Zhang, J.J. Panel interval-valued data nonlinear regression models and applications. *Comput. Econ.* **2024**, *64*, 2413–2435.
30. Lima Neto, E.d.A.; de Carvalho, F.d.A. Nonlinear regression applied to interval-valued data. *Pattern Anal. Appl.* **2017**, *20*, 809–824.
31. Wang, H.; Cao, R. Deep Learning Quantile Regression for Interval-Valued Data Prediction. *J. Forecast.* **2025**, *44*, 1806–1825.
32. Alcacer, A.; Martinez-Garcia, M.; Epifanio, I. Ordinal classification for interval-valued data and interval-valued functional data. *Expert Syst. Appl.* **2024**, *238*, 122277.
33. Freitas, W.; de Souza, R.; Amaral, G.; De Bastiani, F. Exploratory spatial analysis for interval data: A new autocorrelation index with COVID-19 and rent price applications. *Expert Syst. Appl.* **2022**, *195*, 116561.
34. Freitas, W.; de Souza, R.; Amaral, G.; de Moraes, R. Regression applied to symbolic interval-spatial data. *Appl. Intell.* **2024**, *54*, 1545–1565.
35. Wang, X.; Li, S.; Denœux, T. Interval-valued linear model. *Int. J. Comput. Intell. Syst.* **2015**, *8*, 114–127.
36. Jiang, M.; Chen, W.; Xu, H.; Liu, Y. A novel interval dual convolutional neural network method for interval-valued stock price prediction. *Pattern Recognit.* **2024**, *145*, 109920.
37. Zhong, Q.; Wang, J.L. Neural networks for partially linear quantile regression. *J. Bus. Econ. Stat.* **2024**, *42*, 603–614.
38. Molchanov, I. *Theory of Random Sets*; Springer: London, UK, 2005.

39. Petit-Renaud, S.; Denoeux, T. Nonparametric regression analysis of uncertain and imprecise data using belief functions. *Int. J. Approx. Reason.* **2004**, *35*, 1–28.
40. Li, F.; Li, S.; Tang, N.; Denoeux, T. Constrained interval-valued linear regression model. In Proceedings of the 2017 20th International Conference on Information Fusion, IEEE, Xi'an, China, 10–13 July 2017; pp. 1–8.
41. Denoeux, T. Uncertainty Quantification in Regression Neural Networks Using Likelihood-Based. In Proceedings of the Belief Functions: Theory and Applications: 8th International Conference, BELIEF 2024, Belfast, UK, 2–4 September 2024; Proceedings; Springer Nature: Berlin/Heidelberg, Germany, 2024; Volume 14909, p. 40.
42. Peng, Y.; Zhang, Q. Feature selection for interval-valued data based on DS evidence theory. *IEEE Access* **2021**, *9*, 122754–122765.
43. Yao, K.; Liu, B. Uncertain regression analysis: An approach for imprecise observations. *Soft Comput.* **2018**, *22*, 5579–5582.
44. Zhang, H.; Sheng, Y. v-SVR with Imprecise Observations. *Int. J. Uncertain. Fuzziness -Knowl.-Based Syst.* **2025**, *33*, 235–252. <https://doi.org/10.1142/S0218488525500102>.
45. Campagner, A.; Ciucci, D.; Denoeux, T. Belief functions and rough sets: Survey and new insights. *Int. J. Approx. Reason.* **2022**, *143*, 192–215.
46. Liu, Z.; Letchmunan, S. Representing uncertainty and imprecision in machine learning: A survey on belief functions. *J. King Saud Univ.-Comput. Inf. Sci.* **2024**, *36*, 101904. <https://doi.org/10.1016/j.jksuci.2023.101904>.
47. Utkin, L.; Coolen, F. Interval-valued regression and classification models in the framework of machine learning. In *Proceedings of the Seventh Int. Symposium on Imprecise Probabilities: Theories and Applications; ISIPTA'11*; Coolen, F., de Cooman, G., Fetz, T., Oberguggenberger, M., Eds.; SIPTA: Innsbruck, Austria, 2011; pp. 371–380.
48. Utkin, L.; Chekh, A. A new robust model of one-class classification by interval-valued training data using the triangular kernel. *Neural Netw.* **2015**, *69*, 99–110. <https://doi.org/10.1016/j.neunet.2015.05.004>.
49. Utkin, L.; Zhuk, Y.; Chekh, A. A robust one-class classification model with interval-valued data based on belief functions and minimax strategy. In *Proceedings of the Machine Learning and Data Mining in Pattern Recognition; Lecture Notes in Computer Science*; Perner, P., Ed.; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8556, pp. 107–118. https://doi.org/10.1007/978-3-319-08979-9_9.
50. Utkin, L.; Zhuk, Y. An one-class classification support vector machine model by interval-valued training data. *Knowl.-Based Syst.* **2017**, *120*, 43–56. <https://doi.org/10.1016/j.knosys.2016.12.022>.
51. Cheng, X.; Cao, Y.; Li, X.; An, B.; Feng, L. Weakly supervised regression with interval targets. In Proceedings of the International Conference on Machine Learning. PMLR, Honolulu, HI, USA, 23–29 July 2023; pp. 5428–5448.
52. Nguyen, T.; Hocking, T. Interval Regression: A Comparative Study with Proposed Models. *arXiv* **2025**, arXiv:2503.02011, <https://doi.org/https://doi.org/10.48550/arXiv.2503.02011>.
53. Pukdee, R.; Ke, Z.; Gupta, C. Learning from Interval Targets. *arXiv* **2025**, arXiv:2510.20925, <https://doi.org/https://doi.org/10.48550/arXiv.2510.20925>.
54. Tretiak, K.; Schollmeyer, G.; Ferson, S. Neural network model for imprecise regression with interval dependent variables. *Neural Netw.* **2023**, *161*, 550–564. <https://doi.org/https://doi.org/10.1016/j.neunet.2023.02.005>.
55. Brauwiers, G.; Frasincar, F. A general survey on attention mechanisms in deep learning. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 3279–3298.
56. Chaudhari, S.; Mithal, V.; Polatkan, G.; Ramanath, R. An attentive survey of attention models. *ACM Trans. Intell. Syst. Technol.* **2021**, *12*, 53.
57. Correia, A.; Colombini, E. Attention, please! A survey of neural attention models in deep learning. *Artif. Intell. Rev.* **2022**, *55*, 6037–6124.
58. Hassanin, M.; Anwar, S.; Radwan, I.; Khan, F.; Mian, A. Visual attention methods in deep learning: An in-depth survey. *Inf. Fusion* **2024**, *108*, 102417.
59. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A Survey of Transformers. *arXiv* **2021**, arXiv:2106.04554.
60. Liu, F.; Huang, X.; Chen, Y.; Suykens, J. Random Features for Kernel Approximation: A Survey on Algorithms, Theory, and Beyond. *arXiv* **2021**, arXiv:2004.11154v5.
61. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. *Neurocomputing* **2021**, *452*, 48–62.
62. Tay, Y.; Dehghani, M.; Bahri, D.; Metzler, D. Efficient transformers: A survey. *ACM Comput. Surv.* **2022**, *55*, 1–28.
63. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
64. Rubinstein, R.; Kroese, D. *Simulation and the Monte Carlo Method*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2008; p. 345.

65. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2623–2631.
66. Kelly, M.; Longjohn, R.; Nottingham, K. The UCI Machine Learning Repository. Available online: <https://archive.ics.uci.edu> (accessed on 28 January 2026).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.