K-Nearest Neighbour Classification for Interval-Valued Data

Vu-Linh Nguyen, Sébastien Destercke, Marie-Hélène Masson

K-nearest neighbour classification for interval-valued data

Vu-Linh NGUYEN[1], Sébastien DESTERCKE[1], Marie-Hélène MASSON[1,2]

[1] UMR CNRS 7253 Heudiasyc, Sorbonne universités,
Université de technologie de Compiègne CS 60319 - 60203 Compiègne cedex, France
[2] Université de Picardie Jules Verne, France
{linh.nguyen, sebastien.destercke, mylene.masson}@hds.utc.fr

Abstract. This paper studies the problem of providing predictions with a K-nn approach when data have partial features given in the form of intervals. To do so, we adopt an optimistic approach to replace the ill-known values, that requires to compute sets of possible and necessary neighbours of an instance. We provide an easy way to compute such sets, as well as the decision rule that follows from them. Our approach is then compared to a simple imputation method in different scenarios, in order to identify those ones where it is advantageous.
Key words

1 Introduction

The K-nearest neighbor method (K-nn) is a simple but efficient classification method [1, 6, 16]. In classical K-nn, each label is assigned a predicted score and the one with the highest score will be considered as the optimal label of the target instance [1, 6, 16]. Such procedures usually assume that all training data are precisely specified.

In this paper, we are interested in the case where the features of some training data are imprecisely known, that is are known to lie in an interval. In this case, the notion of nearest neighbour is no longer well-defined, and the learning process has to be modified accordingly. Learning from interval-valued or partial data is not new, but has regained some interest in the last few years [14, 15, 2, 13]. In practice, such interval-valued data can come from imprecise measurement devices, imperfect knowledge of an expert, or can also be the result of the summary of a huge data set.

In this paper, we intend to apply some generic learning procedures fitted to partial data [9, 8] to the specific case of interval-valued K-nn methods. It should be noted that while the problem of modelling the uncertainty or the imprecision of the decision within a K-nn procedure applied to precise data has been well-treated in the literature (see *e.g* [10], [3] and[7]), few works deal with the problem of applying a K-nn method to interval-valued data [2]. Imputation methods [4] offer a way to solve this problem by replacing imprecise data by precise values, but typically do not aim at improving as much as possible the method accuracy.

Instead, maximax or optimistic approaches [8] do intend to improve as much as possible the resulting accuracy.

In order to derive our K-nn method, we will adopt an approach similar to the one we previously successfully implemented for partially specified labels [11]. This approach was based on the use of two sets, the sets of possible and necessary predicted labels. These sets correspond to the sets of labels that would be predicted for at least one or all replacement(s) of the partial features, respectively. An important step of our approach will be to determine those sets for the case of interval-valued data from the sets of necessary and possible neighbours of an instance. We deal with this issue in Section 3, after having introduced our notations and settings in Section 2.

Our adaptation of the K-nn procedure, following the maximax approach put forward by Hüllermeier [8] to build predictive models from partial data, will then be derived in Section 4. We then provide some experimental results on several data sets in Section 5.

2 Preliminaries

In our setting, we assume that we have an imprecise training set $\mathbf{D} = \{(\mathbf{X}_n, y_n) | n = 1, \ldots, N\}$, used to make predictions, with the imprecise features $\mathbf{X}_n \subset \mathbb{R}^P$ and the precise label $y_n \in \Omega = \{\lambda_1, \ldots, \lambda_M\}$. We assume that X_n^p contains the precise value x_n^p in form of a of closed interval, or in other words, $X_n^p = \left[a_n^p, b_n^p\right]$. We are interested into predicting the class of a target instance $\mathbf{t}$, whose features are precisely known.

Let us first remind that in case of precise data, the Euclidean distance between a training instance $\mathbf{x}_n$ and a target instance $\mathbf{t}$ is given by

$$d(\mathbf{x}_n, \mathbf{t}) = \left(\sum_{p=1}^{P} \left(x_n^p - t^p\right)^2 \right)^{1/2}. \tag{1}$$

Then for a given target instance $\mathbf{t}$ and a number of nearest neighbours K, its nearest neighbour set in $\mathbf{D}$ will be denoted by $\mathbf{N_t} = \{\mathbf{x}_k^{\mathbf{t}} | k = 1, \ldots, K\}$ where $\mathbf{x}_k^{\mathbf{t}}$ is its k-th nearest neighbour. In the unweighted version of K-nn, the optimal prediction of $\mathbf{t}$ is

$$h(\mathbf{t}) = \arg\max_{\lambda \in \Omega} \sum_{\mathbf{x}_k^{\mathbf{t}} \in \mathbf{N_t}} \mathbb{1}_{\lambda = \mathbf{y}_k^{\mathbf{t}}}, \tag{2}$$

with $\mathbb{1}_A$ is the indicator function of A ($\mathbb{1}_A = 1$ if A is true and 0 otherwise). The idea of the above method is to allow each nearest neighbour to give a vote for its label and the one with the highest number of votes is considered as the optimal prediction.

However, in case of imprecise feature data, there may be some uncertainty about what is the nearest neighbour set $\mathbf{N_t}$ of a target instance $\mathbf{t}$. As a consequence, Eq. (2) is no-longer applicable in order to make a decision on $\mathbf{t}$, the target instance $\mathbf{t}$ is ambiguous, and this ambiguity must be resolved in some way

to make a decision. We will first focus on the problem of determining the ambiguity when having to decide the class of $\mathbf{t}$. Denote by $\mathbf{L}$ the set of all possible precise replacements of training set $\mathbf{D}$:

$$\mathbf{L} = \left\{ \mathbf{l} = \{(\mathbf{x}_n, y_n) | \mathbf{x}_n \in \mathbf{X}_n, n = 1, \ldots, N\} \right\}. \tag{3}$$

To each replacement $\mathbf{l} \in \mathbf{L}$ corresponds a well-defined nearest neighbour set $\mathbf{N}_\mathbf{t}^\mathbf{l}$, on which Eq. (2) can be applied to find the optimal prediction(s) as follows

$$h_\mathbf{l}(\mathbf{t}) = \arg\max_{\lambda \in \Omega} \sum_{\mathbf{x}_k^\mathbf{t} \in \mathbf{N}_\mathbf{t}^\mathbf{l}} \mathbb{1}_{\lambda = \mathbf{y}_k^\mathbf{t}}. \tag{4}$$

The sets of possible and necessary predicted labels are then defined as the sets of labels predicted for at least one replacement and for all possible replacements, respectively. Formally, this gives

$$\mathbf{PL_t} = \left\{ \lambda \in \Omega | \exists \mathbf{l} \in \mathbf{L} \text{ s.t } \lambda \in h_\mathbf{l}(\mathbf{t}) \right\} \tag{5}$$

and

$$\mathbf{NL_t} = \left\{ \lambda \in \Omega | \forall \mathbf{l} \in \mathbf{L} \text{ s.t } \lambda \in h_\mathbf{l}(\mathbf{t}) \right\}. \tag{6}$$

A target instance $\mathbf{t}$ is said to be ambiguous if and only if $\mathbf{PL_t} \neq \mathbf{NL_t}$. As we will see in the next section, determining such sets in the case of interval-valued features requires to compute the sets of necessary and possible neighbours. If the instance $\mathbf{t}$ is non-ambiguous, then the predictive value is clear and nothing needs to be done. If it is ambiguous, then an additional procedure must be performed to pick a prediction within $\mathbf{PL_t}$. In this paper, we will adopt a maximax approach presented in Section 4.

3 Determining ambiguous instances

This section focus on determining whether a given instance is ambiguous, and what are the resulting possible and necessary label sets. In order to do so, we will first have to determine the possible and necessary neighbours.

3.1 Determining interval ranks

Given an imprecise training data set $\mathbf{D}$ and a precise instance $\mathbf{t}$, Groenen *et al.*[5] provides simple formulae to determine the imprecise distance $d(\mathbf{X}_n, \mathbf{t}) = \left[\underline{d}(\mathbf{X}_n, \mathbf{t}), \overline{d}(\mathbf{X}_n, \mathbf{t})\right]$ of $\mathbf{X}_n \in \mathbf{D}$ with respect to $\mathbf{t}$:

$$\overline{d}(\mathbf{X}_n, \mathbf{t}) = \left(\sum_{p=1}^{P} \left[|c_n^p - t^p| + r_n^p \right]^2 \right)^{1/2}, \tag{7}$$

and

$$\underline{d}(\mathbf{X}_n, \mathbf{t}) = \left(\sum_{p=1}^{P} \max\left[0, |c_n^p - t^p| - r_n^p\right]^2 \right)^{1/2}, \tag{8}$$

where $c_n^p = (b_n^p + a_n^p)/2$ and $r_n^p = (b_n^p - a_n^p)/2$, $p = 1, \ldots, P$. Such interval distance allow us to define a partial order on the set $\mathbf{D}$ of training instance as follows

$$\mathbf{X}_i \succeq \mathbf{X}_j \text{ if } \underline{d}(\mathbf{X}_i, \mathbf{t}) \geq \overline{d}(\mathbf{X}_j, \mathbf{t}) \tag{9}$$

where $\mathbf{X}_i \succeq \mathbf{X}_j$ means that $\mathbf{X}_i$ is farther than $\mathbf{X}_j$ from $\mathbf{t}$. As demonstrated by Patil and Taille [12, Sec. 4.1], this partial order then allows us to derive interval rank values as we have that

$$\mathbf{X}_i \succeq \mathbf{X}_j \Rightarrow r(\mathbf{X}_i) \geq r(\mathbf{X}_j),$$

where $r(\mathbf{X}_i)$ is the rank that can be assigned to $\mathbf{X}_i$. Once the relation $\succeq$ is determined, $\mathbf{D}$ is a poset (partially ordered set) and the corresponding relation matrix, denoted by ζ, is a $N \times N$ matrix defined as

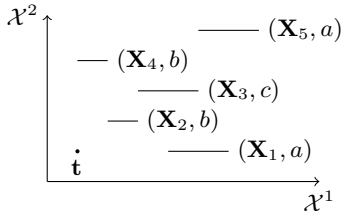
$$\zeta_{i,j} = \begin{cases} 1 \text{ if } \mathbf{X}_i \succeq \mathbf{X}_j \\ 0 \text{ otherwise.} \end{cases} \tag{10}$$

The results given by Theorems 1 and 2 in [12, Sec. 4.1] imply that each instance $\mathbf{X}_n$ can be associated to an imprecise rank which measures how close it is to the target instance $\mathbf{t}$ *i.e* $\mathbf{r}_n = [\underline{r}_n, \overline{r}_n]$ where

$$\underline{r}_n = \sum_{j=1}^{N} \zeta_{n,j} \text{ and } \overline{r}_n = N + 1 - \sum_{j=1}^{N} \zeta_{j,n}. \tag{11}$$

Example 1. Let us consider an example where $|\mathbf{D}| = 5$ and target instance $\mathbf{t}$ as illustrated in Figure 1. Using the relation (9), the corresponding ζ matrix is given in Table 1.

Fig. 1. Example with $|\mathbf{D}| = 5$



$\mathbf{t} = (1, 1)$
$[\underline{d}(\mathbf{X}_1, \mathbf{t}), \overline{d}(\mathbf{X}_1, \mathbf{t})] = [3, 5]$
$[\underline{d}(\mathbf{X}_2, \mathbf{t}), \overline{d}(\mathbf{X}_2, \mathbf{t})] = [1, 1.4]$
$[\underline{d}(\mathbf{X}_3, \mathbf{t}), \overline{d}(\mathbf{X}_3, \mathbf{t})] = [2.8, 4.4]$
$[\underline{d}(\mathbf{X}_4, \mathbf{t}), \overline{d}(\mathbf{X}_4, \mathbf{t})] = [3, 3.2]$
$[\underline{d}(\mathbf{X}_5, \mathbf{t}), \overline{d}(\mathbf{X}_5, \mathbf{t})] = [5.6, 7]$

Table 1. The corresponding ζ matrix for example in Figure 1

	$\mathbf{X}_1$	$\mathbf{X}_2$	$\mathbf{X}_3$	$\mathbf{X}_4$	$\mathbf{X}_5$	$\sum_r$
$\mathbf{X}_1$	1	1	0	0	0	2
$\mathbf{X}_2$	0	1	0	0	0	1
$\mathbf{X}_3$	0	1	1	0	0	2
$\mathbf{X}_4$	0	1	0	1	0	2
$\mathbf{X}_5$	1	1	1	1	1	5
$\sum_c$	2	5	2	2	1	

By applying (11), we can easily compute the imprecise ranks of the training instances.

$$([\underline{r}_1, \overline{r}_1], [\underline{r}_2, \overline{r}_2], [\underline{r}_3, \overline{r}_3], [\underline{r}_4, \overline{r}_4], [\underline{r}_5, \overline{r}_5]) = ([2,4], [1,1], [2,4], [2,4], [5,5]). \quad (12)$$

3.2 Determining the possible label set

Let us now focus on the problem of determining whether a given label λ is a possible prediction for $\mathbf{t}$. Denoting by $\mathbf{R_t} = \{\mathbf{r}_n = [\underline{r}_n, \overline{r}_n] | n = 1, \ldots, N\}$ the imprecise ranks of the instances in $\mathbf{D}$, we can easily determine the sets of possible and necessary neighbours as

$$\mathbf{PN_t} = \{\mathbf{X}_n | \underline{r}_n \leq K\} \quad (13)$$

and

$$\mathbf{NN_t} = \{\mathbf{X}_n | \overline{r}_n \leq K\}. \quad (14)$$

We have that $\mathbf{X}_n \in \mathbf{NN_t}$ if it is in the set of neighbours $\mathbf{X}_n \in \mathbf{N_t^l}$ for any replacement $\mathbf{l}$, while $\mathbf{X}_n \in \mathbf{PN_t}$ if $\mathbf{X}_n \in \mathbf{N_t^l}$ only for some replacement $\mathbf{l} \in \mathbf{L}$. For each label $\lambda \in \Omega$, we can then compute its minimum number of votes

$$s_{\mathbf{t}}^{small}(\lambda) = \left|\left\{\mathbf{X}_n | \mathbf{X}_n \in \mathbf{NN_t}, y_n = \lambda\right\}\right|, \quad (15)$$

given by its necessary neighbours. From s^{small} can then be deduced the maximal and minimal number of votes it can receive from K neighbours, according to the following formulae

$$s_{\mathbf{t}}^{max}(\lambda) = \min\left[K - \sum_{\lambda' \neq \lambda} s_{\mathbf{t}}^{small}(\lambda'), \left|\left\{\mathbf{X}_n | \mathbf{X}_n \in \mathbf{PN_t}, y_n = \lambda\right\}\right|\right], \quad (16)$$

and

$$s_{\mathbf{t}}^{min}(\lambda) = \max\left[s_{\mathbf{t}}^{small}(\lambda), K - \sum_{\lambda' \neq \lambda} s_{\mathbf{t}}^{max}(\lambda')\right]. \quad (17)$$

These scores are simply derived from the fact that, among the K neighbours, at least $s^{small}(\lambda)$ among them must give their votes to label λ. This is proved in the next Lemma, where it is shown that $s_{\mathbf{t}}^{min}(\lambda)$ and $s_{\mathbf{t}}^{max}(\lambda)$ are the minimum and maximum number of votes that can be given to λ over all replacements $\mathbf{l} \in \mathbf{L}$.

Lemma 1. *Given number of nearest neighbours K, a target instance $\mathbf{t}$, the corresponding maximum and minimum score vectors $\left(s_{\mathbf{t}}^{min}(\lambda_1),\ldots,s_{\mathbf{t}}^{min}(\lambda_M)\right)$ and $\left(s_{\mathbf{t}}^{max}(\lambda_1),\ldots,s_{\mathbf{t}}^{max}(\lambda_M)\right)$, then for any $\lambda \in \Omega$, we have that*

$$s_{\mathbf{t}}^{min}(\lambda) = \min_{\mathbf{l}\in\mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda) \text{ and } s_{\mathbf{t}}^{max}(\lambda) = \max_{\mathbf{l}\in\mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda) \tag{18}$$

and consequently, we have that, for $\forall \mathbf{l} \in \mathbf{L}$,

$$s_{\mathbf{t}}^{max}(\lambda) \geq s_{\mathbf{t}}^{\mathbf{l}}(\lambda) \geq s_{\mathbf{t}}^{min}(\lambda), \forall \lambda \in \Omega. \tag{19}$$

Proof. The relation that $s_{\mathbf{t}}^{max}(\lambda) = \max_{\mathbf{l}\in\mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda)$ can be simply proved by observing that $K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{small}(\lambda^{'})$ bounds the number of instance that could be in the set of nearest neighbours and have λ for label, while the value $|\{\mathbf{X}_n | \mathbf{X}_n \in \mathbf{PN_t}, y_n = \lambda\}|$ simply gives the maximal number of such elements that are available within the set of possible neighbours, and that may be chosen freely to be/not be in the neighbour set, as long as they remain lower than the bound $K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{small}(\lambda^{'})$. So, maximising this number of elements simply provides $s_{\mathbf{t}}^{max}(\lambda)$.

Let us now prove that $s_{\mathbf{t}}^{min}(\lambda) = \min_{\mathbf{l}\in\mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda)$, recalling that we just proved that $s_{\mathbf{t}}^{max}(\lambda)$ is reachable for some replacement. We are going to focus on two cases:

1. $s_{\mathbf{t}}^{small}(\lambda) \geq K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{max}(\lambda^{'})$, meaning that $s_{\mathbf{t}}^{min}(\lambda) = s_{\mathbf{t}}^{small}(\lambda)$, hence for every replacement there is at least $s_{\mathbf{t}}^{small}(\lambda)$ nearest neighbors of label λ. Furthermore, $s_{\mathbf{t}}^{small}(\lambda) \geq K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{max}(\lambda^{'})$ implies that $\sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{max}(\lambda^{'}) + s_{\mathbf{t}}^{small}(\lambda) \geq K$, meaning that we can choose the remaining $K - s_{\mathbf{t}}^{small}(\lambda)$ neighbours so that they vote for other labels. In other words, we can find a replacement $\mathbf{l}$ where $s_{\mathbf{t}}^{small}(\lambda) = s_{\mathbf{t}}^{\mathbf{l}}(\lambda)$, proving that $s_{\mathbf{t}}^{min}(\lambda) = \min_{\mathbf{l}\in\mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda)$ in the first case.
2. $s_{\mathbf{t}}^{small}(\lambda) < K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{max}(\lambda^{'})$, meaning that $s_{\mathbf{t}}^{min}(\lambda) = K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{max}(\lambda^{'})$. First note that for any replacement we cannot have $s_{\mathbf{t}}^{\mathbf{l}}(\lambda) < K - \sum_{\lambda'\neq\lambda} s_{\mathbf{t}}^{max}(\lambda^{'})$, otherwise the set of nearest neighbour would be necessarily lower than K. $s_{\mathbf{t}}^{min}(\lambda)$ then reaches this lower bound by simply taking the replacement $s^{\mathbf{l}}$ for which we have $s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'}) = s_{\mathbf{t}}^{max}(\lambda^{'})$, proving that $s_{\mathbf{t}}^{min}(\lambda) = \min_{\mathbf{l}\in\mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda)$ in the second case.

□

For a label $\lambda_m \in \Omega$, the relations among scores (19) and the definition of the possible label set (5) imply that λ_m is a possible label ($\lambda_m \in \mathbf{PL_t}$) if and only if there is a replacement $\mathbf{l} \in \mathbf{L}$ with a score vector $\left(s_{\mathbf{t}}^{\mathbf{l}}(\lambda_1),\ldots,s_{\mathbf{t}}^{\mathbf{l}}(\lambda_M)\right)$ such that

$$\sum_{i=1}^{M} s_{\mathbf{t}}^{\mathbf{l}}(\lambda_i) = K, \tag{20}$$

and

$$\min\left(s_{\mathbf{t}}^{\mathbf{l}}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right) \geq s_{\mathbf{t}}^{\mathbf{l}}(\lambda_i) \geq s_{\mathbf{t}}^{min}(\lambda_i), i = 1,\ldots,M. \tag{21}$$

The condition $\sum_{i=1}^{M} s_{\mathbf{t}}^{\mathbf{l}}(\lambda_i) = K$ simply ensures that $\mathbf{l}$ is a legal replacement. The constraint (21) then ensures that all other labels have a score lower than $s_{\mathbf{t}}^{\mathbf{l}}(\lambda_m)$ for the replacement $\mathbf{l}$ (note that $\min(s_{\mathbf{t}}^{\mathbf{l}}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_m)) = s_{\mathbf{t}}^{\mathbf{l}}(\lambda_m)$), and that their scores are bounded by Eq. (19).

The question is now to know whether we can instantiate such a vector making a winner of λ_m. To achieve this task, we will first maximise its score, such that $s_{\mathbf{t}}^{\mathbf{l}}(\lambda_m) = s_{\mathbf{t}}^{max}(\lambda_m)$. The scores of all other labels λ_i is also lower-bounded by $s_{\mathbf{t}}^{min}(\lambda_i)$, meaning that among the K neighbours we choose in $\mathbf{l}$, only $K - s_{\mathbf{t}}^{max}(\lambda_m) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i)$ remain to be fixed in order to specify the score vector. Then we can focus on the relative difference between $s_{\mathbf{t}}^{min}(\lambda_i)$ and the additional number of chosen neighbours voting for λ_i. Solving the problem defined by Eqs. (20), (21) is equivalent to determine a score vector $(w(\lambda_1), \ldots, w(\lambda_{m-1}), w(\lambda_{m+1}), \ldots, w(\lambda_M))$ whith $w(\lambda_i) = s_{\mathbf{t}}^{\mathbf{l}}(\lambda_i) - s_{\mathbf{t}}^{min}(\lambda_i)$, $\forall \lambda_i \neq \lambda_m$, s.t.

$$\sum_{i=1, i\neq m}^{M} w(\lambda_i) = K - s_{\mathbf{t}}^{max}(\lambda_m) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i), \tag{22}$$

$$\min\left(s_{\mathbf{t}}^{max}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right) - s_{\mathbf{t}}^{min}(\lambda_i) \geq w(\lambda_i) \geq 0, \forall \lambda_i \neq \lambda_m. \tag{23}$$

Eq. (22) again ensures that the replacement is a legal one (the number of neighbours sums up to K), and Eq. (23) ensures that λ_m is a winning label. Also note that if $\exists \lambda_i \in \Omega \setminus \{\lambda_m\}$ s.t $s_{\mathbf{t}}^{max}(\lambda_m) < s_{\mathbf{t}}^{min}(\lambda_i)$, then there no chance for λ_m to be a possible label.

We will now give a proposition allowing to determine in an easy way if a label belongs to the set of possible labels.

Proposition 1. *Given the number of nearest neighbours K, a target instance $\mathbf{t}$, its corresponding maximum and minimum score vectors $\left(s_{\mathbf{t}}^{min}(\lambda_1), \ldots, s_{\mathbf{t}}^{min}(\lambda_M)\right)$ and $\left(s_{\mathbf{t}}^{max}(\lambda_1), \ldots, s_{\mathbf{t}}^{max}(\lambda_M)\right)$. Assuming that $s_{\mathbf{t}}^{max}(\lambda_m) \geq s_{\mathbf{t}}^{min}(\lambda_i)$, for $\forall \lambda_i \in \Omega \setminus \{\lambda_m\}$, then λ_m is a possible label if and only if*

$$K \leq s_{\mathbf{t}}^{max}(\lambda_m) + \sum_{i=1, i\neq m}^{M} \min\left(s_{\mathbf{t}}^{max}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right) \tag{24}$$

Proof. ($\Rightarrow$) Let us prove that λ_m being a possible label implies (24). First, if $\lambda_m \in \mathbf{PL_t}$ and $\mathbf{l}$ is a legitimate replacement, we have that

$$w(\lambda_i) \leq \min\left(s_{\mathbf{t}}^{max}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right) - s_{\mathbf{t}}^{min}(\lambda_i), \ \forall i \neq m \tag{25}$$

otherwise λ_m would not be a winner, or we would give a higher score to λ_i than it actually can get (we would have $s^{\mathbf{l}}(\lambda_i) > s_{\mathbf{t}}^{max}(\lambda_i)$). Since for any replacement we have that Eq. (22) must be satisfied, we have necessarily

$$K - s_{\mathbf{t}}^{max}(\lambda_m) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i) = \sum_{i=1, i\neq m}^{M} w(\lambda_i).$$

If we replace $w(\lambda_i)$ by its upper bound (25), we get the following inequality

$$K - s_{\mathbf{t}}^{max}(\lambda_m) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i) \leq \sum_{i=1, i\neq m}^{M} \min\left(s_{\mathbf{t}}^{max}(\lambda_m)), s_{\mathbf{t}}^{max}(\lambda_i)\right) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i),$$

that is equivalent to the relation

$$K \leq s_{\mathbf{t}}^{max}(\lambda_m) + \sum_{i=1, i\neq m}^{M} \min\left(s_{\mathbf{t}}^{max}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right).$$

($\Leftarrow$) Let us now show that if the conditions given by Eqs. (22)-(23) are satisfied, then $\lambda_m \in \mathbf{PL_t}$. First remark that, once we have assigned the maximal score to λ_m and the minimal ones to the other labels, there remain

$$K - s_{\mathbf{t}}^{max}(\lambda_m) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i)$$

neighbours to choose from. We also know from (23) that at most

$$\sum_{i=1, i\neq m}^{M} \left[\min\left(s_{\mathbf{t}}^{max}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right) - s_{\mathbf{t}}^{min}(\lambda_i)\right]$$

neighbours can still be affected to other labels than λ_m without making it a loser. Clearly, if

$$K - s_{\mathbf{t}}^{max}(\lambda_m) - \sum_{i=1, i\neq m}^{M} s_{\mathbf{t}}^{min}(\lambda_i) \leq \sum_{i=1, i\neq m}^{M} \left[\min\left(s_{\mathbf{t}}^{max}(\lambda_m), s_{\mathbf{t}}^{max}(\lambda_i)\right) - s_{\mathbf{t}}^{min}(\lambda_i)\right],$$

we can reach the number of K neighbours without making λ_m a loser, or inversely letting λ_m be a winner for the chosen replacement, meaning that $\lambda_m \in \mathbf{PL_t}$. □

Example 2. Let us continue with the data set in Example 1 with value $K = 3$. From Table 1 and the interval ranks (12), we can see that

$$\mathbf{PN_t} = \{(\mathbf{X}_1, a), (\mathbf{X}_2, b), (\mathbf{X}_3, c), (\mathbf{X}_4, b)\}, \mathbf{NN_t} = \{(\mathbf{X}_2, b)\}.$$

Then the maximum and minimum scores for all the labels are

$$(s_{\mathbf{t}}^{min}(a), s_{\mathbf{t}}^{min}(b), s_{\mathbf{t}}^{min}(c)) = (0, 1, 0)$$
$$(s_{\mathbf{t}}^{max}(a), s_{\mathbf{t}}^{max}(b), s_{\mathbf{t}}^{max}(c)) = (1, 2, 1).$$

We will now determine whether a given label in $\Omega = \{a, b, c\}$ is a possible label. For label a, we have that

$$s_{\mathbf{t}}^{max}(a) + \min\left(s_{\mathbf{t}}^{max}(a), s_{\mathbf{t}}^{max}(b)\right) + \min\left(s_{\mathbf{t}}^{max}(a), s_{\mathbf{t}}^{max}(c)\right) = 1 + 1 + 1 = 3 \geq K,$$

hence $a \in \mathbf{PL_t}$. The same procedure applied to b and c gives the result $\mathbf{PL_t} = \{a, b, c\}$.

3.3 Determining necessary label set

Let us now focus on characterizing the set $\mathbf{NL_t}$. The following propositions gives a very easy way to determine it, by simply comparing the minimum score of a given label λ to the maximal scores of the others.

Proposition 2. *Given the maximum and minimum scores* $\left(s_{\mathbf{t}}^{min}(\lambda_1), \ldots, s_{\mathbf{t}}^{min}(\lambda_M)\right)$ *and* $\left(s_{\mathbf{t}}^{max}(\lambda_1), \ldots, s_{\mathbf{t}}^{max}(\lambda_M)\right)$*, then a given label* λ *is a necessary label if and only if*

$$s_{\mathbf{t}}^{min}(\lambda) \geq s_{\mathbf{t}}^{max}(\lambda^{'}), \forall \lambda^{'} \neq \lambda. \tag{26}$$

Proof. ($\Rightarrow$) We proceed by contradiction. Assuming that $\exists\ \lambda \in \mathbf{NL_t}$ and $\exists\ \lambda^{'} \in \Omega$ where $s_{\mathbf{t}}^{min}(\lambda) < s_{\mathbf{t}}^{max}(\lambda^{'})$, we show that we can always find a replacement $\mathbf{l} \in \mathbf{L}$ s.t $s_{\mathbf{t}}^{\mathbf{l}}(\lambda) < s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'})$, or in other words, $\exists\ \mathbf{l} \in \mathbf{L}$ s.t $\lambda \notin h_{\mathbf{l}}(\mathbf{t})$, and therefore λ is not necessary. Let us consider the two cases

1. $K - \sum_{\lambda^{''} \neq \lambda} s_{\mathbf{t}}^{max}(\lambda^{''}) \geq s_{\mathbf{t}}^{small}(\lambda)$, then for $\forall \lambda^{''} \neq \lambda$, we give its the maximum score s.t $s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{''}) = s_{\mathbf{t}}^{max}(\lambda^{''})$ and give λ the score $s_{\mathbf{t}}^{\mathbf{l}}(\lambda) = K - \sum_{\lambda^{''} \neq \lambda} s_{\mathbf{t}}^{max}(\lambda^{''})$. Then it is clear that
$$s_{\mathbf{t}}^{\mathbf{l}}(\lambda) = K - \sum_{\lambda^{''} \neq \lambda} s_{\mathbf{t}}^{max}(\lambda^{''}) = s_{\mathbf{t}}^{min}(\lambda) < s_{\mathbf{t}}^{max}(\lambda^{'}) = s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'}).$$
2. $K - \sum_{\lambda^{''} \neq \lambda} s_{\mathbf{t}}^{max}(\lambda^{''}) < s_{\mathbf{t}}^{small}(\lambda)$, then we give λ a score $s_{\mathbf{t}}^{\mathbf{l}}(\lambda) = s_{\mathbf{t}}^{small}(\lambda)$ and give $\lambda^{'}$ a score $s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'}) = s_{\mathbf{t}}^{smax}(\lambda^{'})$. As we have
$$K < \sum_{\lambda^{''} \neq \{\lambda, \lambda^{'}\}} s_{\mathbf{t}}^{max}(\lambda^{''}) + s_{\mathbf{t}}^{small}(\lambda) + s_{\mathbf{t}}^{max}(\lambda^{'})$$
by assumption, we can choose $K - s_{\mathbf{t}}^{small}(\lambda) - s_{\mathbf{t}}^{smax}(\lambda^{'})$ nearest neighbours from at most $\sum_{\lambda^{''} \neq \{\lambda, \lambda^{'}\}} s_{\mathbf{t}}^{max}(\lambda^{''})$ possible nearest neigbours whose labels are not λ or $\lambda^{'}$. In such a replacement we have $s_{\mathbf{t}}^{\mathbf{l}}(\lambda) < s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'})$.

($\Leftarrow$) We are going to prove that (26) implies that the label $\lambda \in \mathbf{NL_t}$ is necessary. Let us first note that

$$\min_{\mathbf{l} \in \mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda) = s_{\mathbf{t}}^{min}(\lambda) \text{ and } \max_{\mathbf{l} \in \mathbf{L}} s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'}) = s_{\mathbf{t}}^{max}(\lambda^{'}), \forall \lambda^{'} \neq \lambda,$$

then (26) ensures that, for any replacement $\mathbf{l} \in \mathbf{L}$,

$$s_{\mathbf{t}}^{\mathbf{l}}(\lambda) \geq \min_{\mathbf{l} \in \mathbf{L}}(s_{\mathbf{t}}^{\mathbf{l}}(\lambda)) \geq \max_{\mathbf{l} \in \mathbf{L}}(s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'})) \geq s_{\mathbf{t}}^{\mathbf{l}}(\lambda^{'}), \forall \lambda, \neq \lambda,$$

which is sufficient to get the proof. □

Example 3. Consider the data set given in Example 2 with the maximum and minimum scores of the labels are

$$(s_{\mathbf{t}}^{min}(a), s_{\mathbf{t}}^{min}(b), s_{\mathbf{t}}^{min}(c)) = (0, 1, 0)$$
$$(s_{\mathbf{t}}^{max}(a), s_{\mathbf{t}}^{max}(b), s_{\mathbf{t}}^{max}(c)) = (1, 2, 1).$$

Then (26) implies that the necessary label set $\mathbf{NL_t} = \{b\}$.

4 Learning from interval-valued feature data

We are now going to present a maximax approach that can be used to make decision on interval-valued feature data. Let us first note that whenever the data is imprecise, the decision rule (2) is no longer well-defined. In case of *set-valued labels*, such a decision rule can be generalized as a maximax rule [9] where the optimal prediction of $\mathbf{t}$ is

$$h(\mathbf{t}) = \arg\max_{\lambda \in \Omega} \sum_{\mathbf{x}_k^{\mathbf{t}} \in \mathbf{N_t}} \mathbb{1}_{\lambda \in \mathbf{y}_k^{\mathbf{t}}}. \tag{27}$$

The idea of the above method is to assign for each label the highest number of votes that it could get. Let call such number of votes by optimal score, then the label with the highest optimal score will be considered as the optimal decision of $\mathbf{t}$. Note that in case of interval-valued feature data, as point out in Lemma 1, the score $s^{max}(\lambda)$ defined in (16) is nothing else but the optimal score of λ. Then the maximax approach can be then generalized for *interval-valued feature data* as follows

$$\begin{aligned} h(\mathbf{t}) &= \arg\max_{\lambda \in \Omega} s_{\mathbf{t}}^{max}(\lambda) \\ &= \arg\max_{\lambda \in \Omega} \left(\min \left[K - \sum_{\lambda' \neq \lambda} s_{\mathbf{t}}^{small}(\lambda'), \left| \left\{ \mathbf{X}_n | \mathbf{X}_n \in \mathbf{PN_t}, y_n = \lambda \right\} \right| \right] \right). \end{aligned} \tag{28}$$

The procedure to make predictions is summarized in Algorithm 1. It is then clear that if $\lambda \in h(\mathbf{t})$, then λ is the winner in at least one replacement $\mathbf{l} \in \mathbf{L}$. Of course, unless we have $|\mathbf{PL_t}| = |\mathbf{NL_t}| = 1$, we cannot be sure that λ is the prediction that fully precise data would have given us. It merely says that it is the most promising one, in the sense that it is the one with the highest potential score. We may suspect that the higher is $|\mathbf{PL_t}|$, the more likely we are to commit mistakes, as the ambiguity increases. It would then be interesting to wonder if we could reduce $|\mathbf{PL_t}|$ by querying the data and making some of their feature precise, using techniques similar to active learning. Yet, we leave the investigation of such an approach to future research.

It may also happen that Equation (28) returns multiple instances that have the highest number of votes. We can then follow a different strategy, where we consider the result of the K-nn procedure for a peculiar replacement. Since every label receives its maximal number of votes by considering the lower distance $\underline{d}(\mathbf{X}_n, \mathbf{t})$, a quite simple idea is to consider the result obtained by Eq. (27) when we consider the replacement $\mathbf{l}$ giving $d(\mathbf{X}_n, \mathbf{t}) = \underline{d}(\mathbf{X}_n, \mathbf{t})$ for every $\mathbf{X}_n$.

5 Experiments

We run experiments on a contaminated version of 6 standard benchmark data sets described in Table 2. By contamination, we mean that we introduce artificially imprecision in these precise data sets. These data sets have various

Algorithm 1: Maximax approach for interval-valued training data.

Input: **D**-imprecise training data, **T**-test set, K-number of nearest neighbours
Output: $\{p(\mathbf{t})|\mathbf{t} \in \mathbf{T}\}$-predictions

foreach $\mathbf{t} \in \mathbf{T}$ **do**
  compute its zeta matrix ζ through (7)-(10);
  foreach $\mathbf{X}_n \in \mathbf{D}$ **do**
    compute imprecise rank $[\underline{r}_n, \overline{r}_n]$ defined in (11);
  determine the $\mathbf{PN_t}$ and $\mathbf{NN_t}$ defined in (13)-(14);
  foreach $\lambda \in \Omega$ **do**
    compute $s_{\mathbf{t}}^{max}(\lambda)$ through (15)-(16);
  determine $h(\mathbf{t})$ defined in (28);
  if $|h(\mathbf{t})| = 1$ **then**
    $p(\mathbf{t}) = h(\mathbf{t})$;
  else
    replace the imprecise distances by $\mathbf{d_t} = \{\underline{d}(\mathbf{X}_n, \mathbf{t})|n = 1, \ldots, N\}$;
    determine $p(\mathbf{t})$ by performing classical K-nn on $\mathbf{d_t}$;

numbers of classes and features, but have a relatively small number of instances, for the reason that handling imprecise data is mainly problematic in such situations: when a lot of data are present, we can expect that enough sufficiently precise data will exist to reach an accuracy level similar to the one of fully precise methods.

Table 2. Data sets used in the experiments

Name	# instances	# features	# labels
iris	150	4	3
seeds	210	7	3
glass	214	9	6
ecoli	336	7	8
dermatology	385	34	6
vehicle	846	18	4

Our experimental setting is as follows: given a data set, we randomly chose a training set **D** consisting of 10% of instances and the rest (90%) as a test set **T**, to limit the number of training samples. For each training instance $\mathbf{x}_i \in \mathbf{D}$ and each feature x_i^j, a biased coin is flipped in order to decide whether or not the feature x_i^j will be contaminated; the probability of contamination is p and we have tested different values of it ($\{0.2, 0.4, 0.6, 0.8\}$). In case x_i^j is contaminated, its precise value is transformed into an interval which can be asymmetric with respect to x_i^j. To do that, a pair of widths $\{l_i^j, r_i^j\}$ will be generated from two Beta distributions, $Beta(\alpha_l, \beta)$ and $Beta(\alpha_r, \beta)$. To control the skewness of the generated data, we introduce a so called unbalance parameter ϵ and assign $\{\alpha_l, \alpha_r\} = \{\beta * \epsilon, \beta/\epsilon\}$.

Then the generated interval valued data is $X_i^j = [x_i^j + l_i^j(\underline{D}^j - x_i^j), x_i^j + r_i^j(\overline{D}^j - x_i^j)]$ where $\underline{D}^j = \min_i(x_i^j)$ and $\overline{D}^j = \max_i(x_i^j)$. As usual when working with Euclidean distance based K-nn, data is normalized. Then, the proposed method is used to make predictions on the test set and its accuracy is compared with the accuracy of two other cases: classical K-nn when fully precise data is given, and a basic imputation method consisting in replacing an interval-valued data X_i^j by its middle value, i.e, $x_i^j = (\underline{X}_i^j + \overline{X}_i^j)/2$. The disambiguated data is used to make predictions under the classical K-nn procedure.

Because the training set is randomly chosen and contaminated, the results maybe affected by random components. Then, for each data set, we repeat the above procedure 100 times and compute the average results. The experimental results on the data sets described in Table 2 with several combinations of parameters (K, p, ϵ, β) are given in the Table 3, with the best results between imputation and the presented method put in bold (the precise case only serves as a reference value of the best accuracy achievable). These first results show that the difference between the two approaches is generally small. Surprisingly, this is true for all explored settings, even for skewed imprecision and high uncertainty ($\epsilon = 0.25$, $p = 0.8$). However, on the two data sets dermatology and vehicle, our approach really provides a significant, consistent increase of accuracy, and this even for low and balanced imprecision ($\epsilon = 1$, $p = 0.2$). In the future, we intend to do more experiments (varying K, increasing the number of data sets) and also try to understand the origin of the witnessed difference.

6 Conclusion

In this paper, we have proposed a maximax approach to deal with K-nn predictions when features are imprecisely specified. Our method mainly relies on identifying possible neighbours in an efficient manner, using the partial orders induced by distance intervals to do so. First experiments suggest that a simple imputation method could often work as well as the presented approach, but for some data sets our approach can bring a real advantage. Compared to imputation methods, our approach also provides us with information about how uncertain our prediction is, by identifying possible and necessary neighbours.

Such information is instrumental in the next step we envision for this work: determining which sample feature should be queried first to improve the overall algorithm accuracy, much like what we did for the case of partial labels [11]. Also, investigating the decision rules and querying procedure when both training and test data can be imprecise is another future direction though defining the partial order (9) is still a challenge.

Acknowledgement

This work was carried out in the framework of Labex MS2T and EVEREST projects, which were funded by the French National Agency for Research (Reference ANR-11-IDEX-0004-02, ANR-12-JS02-0005).

Table 3. Experimental Results: Accuracy of classifiers (%)

		iris	seeds	glass	ecoli	derma.	vehicle
$p = 0.2$, $\epsilon = 0.25$	Precise	91.55	84.88	49.70	75.21	82.26	53.55
	Imputation	88.93	83.79	47.30	74.40	80.20	49.45
	Maximax	**89.39**	**83.80**	**48.37**	**74.57**	**81.19**	**53.21**
$p = 0.2$, $\epsilon = 0.5$	Precise	91.57	85.15	50.46	74.98	81.76	53.65
	Imputation	89.07	**84.16**	47.41	**74.23**	77.41	50.35
	Maximax	**89.43**	83.92	**48.54**	74.13	**80.55**	**53.19**
$p = 0.2$, $\epsilon = 1$	Precise	91.35	85.39	50.49	75.11	82.13	53.65
	Imputation	88.80	**84.36**	47.48	**74.52**	75.12	50.76
	Maximax	**89.08**	84.31	**48.73**	74.35	**80.54**	**53.24**
$p = 0.4$, $\epsilon = 0.25$	Precise	91.44	85.31	50.34	75.33	82.26	53.54
	Imputation	87.70	83.83	46.70	**74.49**	75.87	49.88
	Maximax	**88.59**	**83.88**	**48.06**	74.02	**80.32**	**52.95**
$p = 0.4$, $\epsilon = 0.5$	Precise	91.14	85.26	50.20	75.47	82.04	53.50
	Imputation	87.00	**83.77**	46.31	**74.60**	75.14	49.70
	Maximax	**87.42**	83.61	**47.69**	73.87	**79.75**	**52.79**
$p = 0.4$, $\epsilon = 1$	Precise	91.11	85.33	50.18	75.36	82.24	53.52
	Imputation	**86.87**	**83.80**	46.17	**74.62**	73.10	49.77
	Maximax	86.59	83.52	**47.58**	73.57	**79.51**	**52.70**
$p = 0.6$, $\epsilon = 0.25$	Precise	92.53	84.59	50.82	74.54	81.10	53.25
	Imputation	80.46	**80.88**	43.56	**72.27**	75.38	43.41
	Maximax	**84.86**	80.85	**45.90**	69.48	**77.40**	**50.87**
$p = 0.6$, $\epsilon = 0.5$	Precise	92.00	85.39	50.97	74.86	81.98	53.38
	Imputation	80.06	**82.51**	44.04	**73.13**	73.28	45.10
	Maximax	**82.43**	82.06	**46.08**	70.24	**77.29**	**50.75**
$p = 0.6$, $\epsilon = 1$	Precise	91.66	85.57	51.01	74.83	81.97	53.46
	Imputation	80.22	**82.47**	44.37	**73.45**	68.41	46.48
	Maximax	**80.79**	82.16	**46.19**	70.47	**75.84**	**50.59**
$p = 0.8$, $\epsilon = 0.25$	Precise	91.62	85.46	50.74	74.97	81.91	53.40
	Imputation	79.13	**81.92**	44.34	**73.27**	69.42	44.52
	Maximax	**81.26**	81.86	**45.88**	70.19	**76.04**	**48.88**
$p = 0.8$, $\epsilon = 0.5$	Precise	91.27	85.29	50.85	74.92	82.08	53.44
	Imputation	78.53	81.95	44.33	**73.34**	69.00	44.18
	Maximax	**80.92**	**82.00**	**45.66**	70.17	**75.71**	**48.32**
$p = 0.8$, $\epsilon = 1$	Precise	91.16	85.35	50.71	75.00	82.18	53.45
	Imputation	78.58	82.04	44.25	**73.60**	66.67	44.71
	Maximax	**80.38**	**82.47**	**45.48**	70.46	**74.99**	**47.92**

Fixed parameters: $K = 3, \beta = 10$

Bibliography

[1] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[2] R. M. de Souza and F. d. A. De Carvalho. Clustering of interval data based on city–block distances. *Pattern Recognition Letters*, 25(3):353–365, 2004.

[3] T. Denoeux. A k-nearest neighbor classification rule based on dempster-shafer theory. *IEEE Transactions on Systems, Man and Cybernetics*, 25:804–813, 1995.

[4] A. R. T. Donders, G. J. van der Heijden, T. Stijnen, and K. G. Moons. Review: a gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.

[5] P. J. Groenen, S. Winsberg, O. Rodriguez, and E. Diday. I-scal: Multidimensional scaling of interval dissimilarities. *Computational Statistics & Data Analysis*, 51(1):360–378, 2006.

[6] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

[7] C. Holmes and N. Adams. A probabilistic nearest neighbour method for statistical pattern recognition. *J. Roy. Statist. Soc. Ser. B*, 64:295–306, 2002.

[8] E. Hüllermeier. Learning from imprecise and fuzzy observations: Data disambiguation through generalized loss minimization. *International Journal of Approximate Reasoning*, 55(7):1519–1534, 2014.

[9] E. Hüllermeier and J. Beringer. Learning from ambiguously labeled examples. *Intelligent Data Analysis*, 10(5):419–439, 2006.

[10] J. Keller, M. Gray, and J. Givens. A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man and Cybernetics*, 15:580–585, 1985.

[11] V.-L. Nguyen, S. Destercke, and M.-H. Masson. Querying partially labelled data to improve a k-nn classifier. In *AAAI*, pages 2401–2407, 2017.

[12] G. Patil and C. Taillie. Multiple indicators, partially ordered sets, and linear extensions: Multi-criterion ranking and prioritization. *Environmental and Ecological Statistics*, 11(2):199–228, 2004.

[13] A. P. D. Silva and P. Brito. Linear discriminant analysis for interval data. *Computational Statistics*, 21(2):289–308, 2006.

[14] L. V. Utkin, A. I. Chekh, and Y. A. Zhuk. Binary classification svm-based algorithms with interval-valued training data using triangular and epanechnikov kernels. *Neural Networks*, 80:53–66, 2016.

[15] L. V. Utkin and F. P. Coolen. Interval-valued regression and classification models in the framework of machine learning. In *ISIPTA*, volume 11, pages 371–380. Citeseer, 2011.

[16] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.