Multi-class classification problems for the k-NN algorithm in the case of missing values

1st Urszula Bentkowska, University of Rzeszów, Rzeszów, Poland, ududziak@ur.edu.pl
2nd Jan G. Bazan, University of Rzeszów, Rzeszów, Poland, bazan@ur.edu.pl
3rd Marcin Mrukowicz, University of Rzeszów, Rzeszów, Poland, mmrukowicz@ur.edu.pl
4th Lech Zaręba, University of Rzeszów, Rzeszów, Poland, lzareba@ur.edu.pl
5th Piotr Molenda, University of Rzeszów, Rzeszów, Poland, pmole@ur.edu.pl

***Abstract*—In this contribution methods for improving the quality of multi-class classification by the k nearest neighborhood classifiers in the case of large number of missing values in data sets are considered. Two versions of classifiers are compared. In the first case the aggregation of certainty coefficients of the individual classifiers with the use of the arithmetic mean is applied. In the second case interval modelling and interval-valued aggregation functions are involved. It is proved that the classifier which uses interval methods entails a much slower decrease in classification quality.**

***Index Terms*—multi-class classifier, k-NN classifier, missing values, interval-valued aggregation functions**

I. Introduction

Binary classifiers are one of the supervised machine learning methods that help in extracting knowledge. However, in real world applications we often encounter problems that involve more than two classes. It is the so called multi-class problem. Classification problems in multiple classes may be solved in diverse ways. One of the most popular methods is a binarization or a decomposition (cf. [1]). In multi-class problems there are more decision constraints than in the case of a binary problem. This is why decomposition methods are often applied in order to simplify the problem. There exist different decomposition strategies. The most popular are one-versus-one (OVO) and one-versus-all (OVA) (cf. [2]). A detailed overview of both methods, presenting their advantages and disadvantages, with an experimental study for several well-known algorithms was provided in [1].

In this contribution we consider a problem of k nearest neighborhood (k-NN) based multi-class classifiers in the case of missing values in data test tables. In the case of missing values in data tables a problem of lowering the performance of classifiers is observed (cf. [3], [4], [5], [6]). Our aim is to show superiority of the proposed interval methods over numerical ones. Interval-valued fuzzy set theory, as one of the extensions of fuzzy sets theory, is a promising tool for solving diverse real-life problems (cf. [7], [8], [9], [10], [11]). We propose an approach of creating the so called uncertainty intervals (cf. [12], [13], [14]) for the k-NN and then aggregating the obtained intervals with the use of interval-valued aggregation functions. We prove that this method has better results than the other applied method where an aggregation function is just the numerical arithmetic mean. Aggregation functions proved to be an effective tool in many application areas (cf. [15]). Here we use interval-valued aggregation functions which are recently developed by several authors and successfully applied in real life problems (cf. [16]).

There are many implementations of the k-NN method in various software libraries and programming languages. The implementation of our classifiers was made in Python language with usage of its libraries NumPy, pandas and scikit-learn. We proved that the proposed interval method obtained statistically significantly better results than the numerical ones which was justified with the use of Statistica programme.

The paper is organized as follows. In Section II, we recall the notion and examples of interval-valued aggregation functions. Next, in Section III, details of the considered classifiers are provided. Finally, in Sections IV-V implementation details and discussion on the results are given.

II. Interval-valued aggregation functions

We recall definition of an interval-valued fuzzy set and some comparability relations used in interval-valued fuzzy settings.

Definition 1 (cf. [14])**.** An interval-valued fuzzy set F on X is a mapping $F : X \to L^I$ such that $F(x) = [\underline{F}(x), \overline{F}(x)] \in L^I$ for $x \in X$, where

$$L^I = \{[\underline{x}, \overline{x}] : \ \underline{x}, \overline{x} \in [0,1], \ \underline{x} \leqslant \overline{x}\}.$$

The well-known classical monotonicity (partial order) for intervals is of the form (cf. [20])

$$[\underline{x}, \overline{x}] \preceq [\underline{y}, \overline{y}] \Leftrightarrow \underline{x} \leqslant \underline{y}, \ \overline{x} \leqslant \overline{y}. \tag{1}$$

We will also use the following comparability relations on L^I (cf. [21]):

$$[\underline{x}, \overline{x}] \preceq_\pi [\underline{y}, \overline{y}] \Leftrightarrow \underline{x} \leqslant \overline{y}, \tag{2}$$

$$[\underline{x}, \overline{x}] \preceq_\nu [\underline{y}, \overline{y}] \Leftrightarrow \overline{x} \leqslant \underline{y}. \tag{3}$$

The listed comparability relations (1)-(3) follow from the epistemic settings of interval-valued fuzzy sets which represents the uncertainty of the membership value of a given object in an interval-valued fuzzy set. Further examples of comparability relations, including orders between intervals one may find for example in [22], [23], [24]. In [25] the admissible linear orders were introduced and the general method to build

978-1-7281-6932-3/20/$31.00 ©2020 IEEE

different linear orders for the family of intervals L^I was presented.

Definition 2 ([25])**.** An order $\leq_{L^I}$ on L^I is called admissible if it is linear and satisfies that for all $x, y \in L^I$, such that $x \preceq y$, then $x \leq_{L^I} y$.

Example 1. Well-known examples of admissible (linear) orders on L^I are presented below:

• Xu and Yager order defined as
$[\underline{x}, \overline{x}] \leq_{XY} [\underline{y}, \overline{y}]$ if and only if

$$\underline{x} + \overline{x} < \underline{y} + \overline{y} \text{ or } (\overline{x} + \underline{x} = \overline{y} + \underline{y} \text{ and } \overline{x} - \underline{x} \leqslant \overline{y} - \underline{y}).$$

• The first lexicographical order (with respect to the first variable), $\leq_{Lex1}$ defined as $[\underline{x}, \overline{x}] \leq_{Lex1} [\underline{y}, \overline{y}]$ if and only if

$$\underline{x} < \underline{y} \text{ or } (\underline{x} = \underline{y} \text{ and } \overline{x} \leqslant \overline{y}).$$

• The second lexicographical order (with respect to the second variable), $\leq_{Lex2}$ defined as
$[\underline{x}, \overline{x}] \leq_{Lex2} [\underline{y}, \overline{y}]$ if and only if

$$\overline{x} < \overline{y} \text{ or } (\overline{x} = \overline{y} \text{ and } \underline{x} \leqslant \underline{y}).$$

In this paper we will use interval-valued aggregation functions, possible aggregation functions and necessary aggregation functions (cf. [26]) which follow from the notions of comparability relations (1)-(3). Firstly, we recall the concept of an aggregation function on the unit interval $[0, 1]$.

Definition 3 (cf. [27], p. 6)**.** A function $A : [0, 1]^n \to [0, 1]$, $n \in \mathbb{N}$, $n \geqslant 2$, which is increasing in each variable, i.e., for all $s_1, \ldots, s_n, t_1, \ldots, t_n \in [0, 1]$

$$(\underset{1 \leqslant i \leqslant n}{\forall} s_i \leqslant t_i) \Rightarrow A(s_1, \ldots, s_n) \leqslant A(t_1, \ldots, t_n), \quad (4)$$

is called an aggregation function if $A(0, \ldots, 0) = 0$, $A(1, \ldots, 1) = 1$.

The notion of an aggregation function may be naturally extended from the domain $[0, 1]$ to the domain L^I in the following way.

Definition 4 (cf. [28])**.** An operation $\mathcal{A} : (L^I)^n \to L^I$ is called an aggregation function on L^I if it is increasing, i.e.,

$$\underset{\mathbf{x}_i, \mathbf{y}_i \in L^I}{\forall} \mathbf{x}_i \preceq \mathbf{y}_i \Rightarrow \mathcal{A}(\mathbf{x}_1, \ldots, \mathbf{x}_n) \preceq \mathcal{A}(\mathbf{y}_1, \ldots, \mathbf{y}_n) \quad (5)$$

and $\mathcal{A}(\underbrace{\mathbf{0}, ..., \mathbf{0}}_{n\times}) = \mathbf{0}$, $\mathcal{A}(\underbrace{\mathbf{1}, ..., \mathbf{1}}_{n\times}) = \mathbf{1}$, where $\mathbf{0} = [0, 0]$, $\mathbf{1} = [1, 1]$.

Below we give some construction methods of interval-valued aggregation functions on L^I. These methods involve the notion of an aggregation function on $[0, 1]$.

Definition 5 (cf. [29])**.** $\mathcal{A} : (L^I)^n \to L^I$ is said to be a representable aggregation function on L^I if there exist two aggregation functions $A_1, A_2 : [0, 1]^n \to [0, 1]$, $A_1 \leqslant A_2$ such that, for every $\mathbf{x}_1 = [\underline{x}_1, \overline{x}_1]$, $\mathbf{x}_2 = [\underline{x}_2, \overline{x}_2]$, ..., $\mathbf{x}_n = [\underline{x}_n, \overline{x}_n] \in L^I$ it holds that

$$\mathcal{A}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = [A_1(\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n), A_2(\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)].$$

By replacing in the monotonicity condition (5) the classical order $\preceq$ with the relations $\preceq_\pi$ or $\preceq_\nu$ we obtain the following types of aggregation operators.

Definition 6 ([26])**.** An operation $\mathcal{A} : (L^I)^n \to L^I$ is called a possible aggregation function (pos-aggregation function) if

$$\underset{\mathbf{x}_i, \mathbf{y}_i \in L^I}{\forall} \mathbf{x}_i \preceq_\pi \mathbf{y}_i \Rightarrow \mathcal{A}(\mathbf{x}_1, \ldots, \mathbf{x}_n) \preceq_\pi \mathcal{A}(\mathbf{y}_1, \ldots, \mathbf{y}_n) \quad (6)$$

and $\mathcal{A}(\underbrace{\mathbf{0}, ..., \mathbf{0}}_{n\times}) = \mathbf{0}$, $\mathcal{A}(\underbrace{\mathbf{1}, ..., \mathbf{1}}_{n\times}) = \mathbf{1}$.

Definition 7 ([26])**.** An operation $\mathcal{A} : (L^I)^n \to L^I$ is called a necessary aggregation function (nec-aggregation function) if

$$\underset{\mathbf{x}_i, \mathbf{y}_i \in L^I}{\forall} \mathbf{x}_i \preceq_\nu \mathbf{y}_i \Rightarrow \mathcal{A}(\mathbf{x}_1, \ldots, \mathbf{x}_n) \preceq_\nu \mathcal{A}(\mathbf{y}_1, \ldots, \mathbf{y}_n) \quad (7)$$

and $\mathcal{A}(\underbrace{\mathbf{0}, ..., \mathbf{0}}_{n\times}) = \mathbf{0}$, $\mathcal{A}(\underbrace{\mathbf{1}, ..., \mathbf{1}}_{n\times}) = \mathbf{1}$.

We may also consider interval-valued aggregation functions with respect to linear orders.

Definition 8 ([30])**.** $\mathcal{A} : (L^I)^n \to L^I$ is called an aggregation function with respect to the admissible linear order $\leq_{L^I}$, if

$$\underset{\mathbf{x}_i, \mathbf{y}_i \in L^I}{\forall} \mathbf{x}_i \leq_{L^I} \mathbf{y}_i \Rightarrow \mathcal{A}(\mathbf{x}_1, \ldots, \mathbf{x}_n) \leq_{L^I} \mathcal{A}(\mathbf{y}_1, \ldots, \mathbf{y}_n)$$

and $\mathcal{A}(\underbrace{\mathbf{0}, ..., \mathbf{0}}_{n\times}) = \mathbf{0}$, $\mathcal{A}(\underbrace{\mathbf{1}, ..., \mathbf{1}}_{n\times}) = \mathbf{1}$.

Below we present the representatives of aggregation functions belonging to the mentioned classes of aggregation functions on L^I which were applied in our experiments. Namely,

$$\mathcal{A}_1(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n) = \left[\frac{\underline{x}_1 + \underline{x}_2 + ... + \underline{x}_n}{n}, \frac{\overline{x}_1 + \overline{x}_2 + ... + \overline{x}_n}{n}\right], \quad (8)$$

where $\mathcal{A}_1$ is a representable aggregation function on L^I, it is a pos-aggregation functions and a nec-aggregation function (cf. [26]) and it is an interval-valued aggregation function with respect to the linear orders $\leq_{Lex1}$, $\leq_{Lex2}$ and $\leq_{XY}$ (cf. [30]).

$$\mathcal{A}_2(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n) = [\frac{\underline{x}_1 + \underline{x}_2 + ... + \underline{x}_n}{n}, \max(\frac{\underline{x}_1 + \overline{x}_2 + ... + \overline{x}_n}{n}, ..., \frac{\overline{x}_1 + \cdots + \overline{x}_{n-1} + \underline{x}_n}{n})], \quad (9)$$

where $\mathcal{A}_2$ is a nec-aggregation function and it is neither a representable aggregation function nor a pos-aggregation function (cf. [26]).

$$\mathcal{A}_3(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n) = \left[\frac{\underline{x}_1 + ... + \underline{x}_n}{n}, \frac{\overline{x}_1^2 + ... + \overline{x}_n^2}{\overline{x}_1 + ... + \overline{x}_n}\right] \quad (10)$$

and its more general version $\mathcal{A}_4$, where $p \in \mathbb{N}$, $p \geqslant 2$,

$$\mathcal{A}_4(\mathbf{x}_1, ..., \mathbf{x}_n) = \left[\frac{\underline{x}_1 + ... + \underline{x}_n}{n}, \frac{\overline{x}_1^p + ... + \overline{x}_n^p}{\overline{x}_1^{p-1} + ... + \overline{x}_n^{p-1}}\right]. \quad (11)$$

$\mathcal{A}_3$ and $\mathcal{A}_4$ are pos-aggregation functions but they are neither aggregation functions nor nec-aggregation functions, where

it used the convention $\frac{0}{0} = 0$. The upper bounds of these aggregation operators were built with the use of *Lehmer means* (cf. [31], p. 185) which do not fulfil the classical monotonicity (4) and as a result $\mathcal{A}_3$, $\mathcal{A}_4$ do not fulfil the condition (5) but they fulfil condition (6) (cf. [26]).

$$\mathcal{A}_5(\mathbf{x}_1, ..., \mathbf{x}_n) = \left[\sqrt{\frac{\underline{x}_1^2 + ... + \underline{x}_n^2}{n}}, \sqrt[3]{\frac{\overline{x}_1^3 + ... + \overline{x}_n^3}{n}} \right], \quad (12)$$

$$\mathcal{A}_6(\mathbf{x}_1, ..., \mathbf{x}_n) = \left[\sqrt[3]{\frac{\underline{x}_1^3 + ... + \underline{x}_n^3}{n}}, \sqrt[4]{\frac{\overline{x}_1^4 + ... + \overline{x}_n^4}{n}} \right], \quad (13)$$

where $\mathcal{A}_5$ and $\mathcal{A}_6$ are representable aggregation functions, pos-aggregation functions and interval-valued aggregation functions with respect to the linear orders $\leq_{Lex1}$ and $\leq_{Lex2}$ (cf. [32]) but they are not nec-aggregation functions (cf. [26]).

$$\mathcal{A}_7(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = [\min(\frac{\overline{x}_1 + \underline{x}_2 + \cdots + \underline{x}_n}{n}, ..., \frac{\underline{x}_1 + \cdots + \underline{x}_{n-1} + \overline{x}_n}{n}), \frac{\overline{x}_1 + \overline{x}_2 + \cdots + \overline{x}_n}{n}]. \quad (14)$$

Aggregation function (14) is a nec-aggregation function but it is neither a pos-aggregation function (cf. [26]) nor a representable aggregation function.

$$\mathcal{A}_8(\mathbf{x}_1, ..., \mathbf{x}_n) = \left[\sqrt[n]{\underline{x}_1 \cdot ... \cdot \underline{x}_n}, \frac{\overline{x}_1^2 + ... + \overline{x}_n^2}{\overline{x}_1 + ... + \overline{x}_n} \right], \quad (15)$$

$$\mathcal{A}_9(\mathbf{x}_1, ..., \mathbf{x}_n) = \left[\sqrt{\frac{\underline{x}_1^2 + ... + \underline{x}_n^2}{n}}, \frac{\overline{x}_1^3 + ... + \overline{x}_n^3}{\overline{x}_1^2 + ... + \overline{x}_n^2} \right] \quad (16)$$

where for $\mathcal{A}_8$, $\mathcal{A}_9$ it is used the convention $\frac{0}{0} = 0$. $\mathcal{A}_8$ and $\mathcal{A}_9$ are pos-aggregation functions but they are neither aggregation functions nor nec-aggregation functions (cf. [26]).

$$\mathcal{A}_{10}(\mathbf{x}_1, ..., \mathbf{x}_n) = \left[\sqrt{\frac{\underline{x}_1^2 + ... + \underline{x}_n^2}{n}}, \sqrt{\frac{\overline{x}_1^2 + ... + \overline{x}_n^2}{n}} \right], \quad (17)$$

where $\mathcal{A}_{10}$ is a representable aggregation function, a pos-aggregation function and a nec-aggregation function (cf. [26]). Moreover, it is also an interval-valued aggregation function with respect to the linear orders $\leq_{Lex1}$ and $\leq_{Lex2}$ (cf. [32]). Some of these aggregation operators were successfully used in the binary classification problems for the optimization of k-NN algorithm in the case of missing values [33].

III. Decomposition technique in the multi-class classifiers

In classification the problem of missing values is usually solved in the following three ways. In the first method the missing value is treated as a normal value that carries certain information. However, this approach does not really consider missing values, because they are treated as normal ones. In the second approach classification of objects is done only on the basis of those attributes on which there are no "empty spaces". This approach requires the creation of specific classifiers that can classify test objects based only on some attributes. A typical example of such classifier is a classifier based on decision rules. It is worth noting, that this type of a classifier will work much worse if it classifies test objects with missing values compared to the case when there are no such values in the test object. The third approach, which seems to be the most commonly used in practice (cf. [34]), is that when classifying a test object having empty spaces, before classifying the object, empty spaces are filled with a specific value determined based on the training data. This method of data imputation usually consists in searching for the most frequently occurring value of a symbolic attribute or the average value of a numerical attribute. Another method was applied in OvaExpert, one of the real-life diagnosis support systems for ovarian tumor where the missing values of attributes were replaced with the whole intervals representing the given feature (cf. [3], [5], [6], [16]). This approach proved to be effective to make high-quality decisions under incomplete information and uncertainty. In our approach we also consider the so called uncertainty intervals but they are constructed in another way and they are used in another step of classification than in the mentioned papers.

Since the biggest problem with the correct classification of test data (with missing values) will have classifiers using all conditional attributes, in our experiments we apply a typical example of such classifier, i.e. the k-nearest neighbors classifier. The test object data can be classified using the k-NN method with diverse parameters k. Therefore, a conflict appears between classifiers that operate on the basis of different k values. In order to resolve this conflict, we suggest aggregation of the classification results of individual classifiers (the aggregation methods have been already applied in [3], [5], [6], [16]).

There are many variants and several parameters that can be used for the k-NN method (cf. [17], [18], [19]). One of the most important parameters is the distance used, which can be chosen in many ways. We apply here only the Euclidean distance since it is the most commonly applied and it is suitable for the numerical values that we analyze. Furthermore, in this contribution we do not study the influence of the distance on the performance of the classifiers. Other types of distances (e.g. Manhattan distance) are planned to be applied in the further development of the experiments.

The aim of the experiments presented in this paper was to compare numerical modelling with interval-valued modelling. We compared the performance of multi-class classifiers MM

and MF. In the construction of these multi-class classifiers we used decomposition method one-vs-all which uses binary versions of algorithms given in [33]. However, we applied here more examples of aggregation operators than in [33], and the algorithms were implemented in Python.

We briefly recall the most important issues connected with the applied binary classifiers.

A. Construction of the binary classifiers

Binary algorithm BM (cf. Algorithm 1, [33]) is based on the aggregation of classical k-NN classifiers with the use of numerical arithmetic mean. Binary algorithm BF (cf. Algorithm 2, [33]) involves interval-valued aggregation of uncertainty intervals by individual classifiers where a special way of coping with missing values was proposed.

Algorithm 1: Binary classifier BM (cf. [33])

Input:
1) training data set represented by decision table $\mathbf{T} = (U, A, d)$, where $n = card(U)$ and $l = card(A)$,
2) collection $C_1, ..., C_m$ of k-NN classifiers for different k, where, e.g., $k \in \{5, 10, 20, 30\}$,
3) test object u.

Output: The certainty coefficient representing the probability of belonging the object u to the "main class"

```
begin
    for i := 1 to m do
        Compute certainty coefficient ("main class" membership probability) for the given test object u using the classifier C_i and assign it to p_i
    end
    Determine the final certainty coefficient p for the object u by aggregating (with a use of the arithmetic mean) the certainty coefficients p_1,...,p_m.
end
```

In the case of classifier BM the known method (cf. WEKA API library, [35], [36]) of coping with missing values was applied. This mechanism works when calculating the distance between the values on a given attribute. The difference between the numerical values of the two objects (only numerical values occurred in the analyzed data sets) is calculated as the absolute value of the difference of these values. If both values of the attributes are given, then the absolute value of the difference is determined for these values. If both values of the attributes are missing, then their difference is equal to 1. If one of the values of attributes is missing and the other one is equal to v, then the difference is taken as the value $\max(v, 1 - v)$, as a result this difference is as high as possible. Therefore, we see that when classifying objects by the classical k-NN method, if the test objects have missing values of attributes, then it results in a sense in "moving" these objects from the training objects (without missing values). As a result, the test object with a large number of missing values may be too strongly moved away from the training objects and it can get closer to the training objects from the wrong class. It makes this object wrong classified. Therefore, we proposed another method of coping with missing values, i.e. Algorithm BF (cf. Algorithm 2). In the proposed method in BF, by randomly filling missing values, we obtain objects that are less distanced from the proper training objects. Namely, in the case of algorithm BF the method of determining the uncertainty interval by single classifiers is applied. This method consists in the fact that a test object having missing values is classified by a given k-NN classifier in a specific way. Namely, during this classification, many classifications of different objects are actually made, which are constructed based on the test object. The construction of these objects is based on the fact that missing values in the object are filled in various ways based on the values from the training data. The ideal situation here would be that during the classification procedure, all possible test objects that can be generated from the given test object are classified by inserting empty values in all possible ways of the attribute values from the training data for the given attribute. The result of each such classification is the certainty value of belonging to the main class. Thanks to this value, the uncertainty interval may be computed by determining the minimum of these values (lower end of the interval) and the maximum (upper end of the interval). We propose the Monte Carlo method of choosing the test objects. This method consists in the fact that in the space of all possible objects that can be generated for the given test object with missing values, we select a random sample (the draw being made in accordance with the distribution of variable). Then, we classify only objects from this sample and on the basis of the obtained classification results, we estimate the lower and upper end of the uncertainty interval. Further details on the construction of the binary classifier BF may be found in [33].

B. Construction of the multi-class classifiers

We have applied in our experiments the basic decomposition method i.e. one-versus-all (OVA) technique that transforms an s-class classification problem, $s > 2$, into a number of binary problems. The choice of this technique for the first step of our study on the multiclass version of the problem was motivated by the results presented in [1]. On the basis of the performed experiments, it was stated that decomposition techniques such as OVA and OVO obtain better classification results than the multiclass version of the k-NN. However, no significant differences were found between OVO and OVA strategies and the original multiclass version of the k-NN. We applied the OVA technique for both multi-class versions of the classifiers denoted by MM and MF. In the case of OVA a binary classifier is used to distinguish between a single class and the remaining ones. As a result the number of s binary classifications are needed and a score vector $(\mathbf{p_1}, ..., \mathbf{p_s})$ is used. The output class is taken from the classifier with the

Algorithm 2: Binary classifier BF (cf. [33])

Input:
1) data set represented by $\mathbf{T} = (U, A, d)$,
2) collection $C_1, ..., C_m$ of k-NN classifiers, e.g., $k \in \{5, 10, 20, 30\}$,
3) fixed parameter r, e.g., $r = 10$,
4) aggregation function $\mathcal{A}$,
5) test object u

Output: The certainty coefficient representing the probability of belonging the object u to the "main class"

begin
  if *exists at least one missing value in the object* u **then**
    for $i := 1$ **to** m **do**
      Choose randomly with the Monte Carlo method r objects $u_1, ..., u_r$ on the basis of object u, where any object u_j is constructed as follows, $j \in \{1, ..., r\}$:
      begin
        Copy values of attributes from u to u_j;
        For each attribute whose value in u_j is missing, replace it with a randomly selected value from the range of possible values for this attribute (from the training data);
      end
      Compute certainty coefficient for objects $u_1, ..., u_r$ using the classifier C_i and assign these values to $p_1, ..., p_r$;
      Compute $min\{p_1, ..., p_r\}$ and assign it to min_i;
      Compute $max\{p_1, ..., p_r\}$ and assign it to max_i;
    end
    Determine the uncertainty interval $[down(u), up(u)]$ for the object u by aggregating (with the use of $\mathcal{A}$) the intervals $[min_1, max_1]$, ..., $[min_m, max_m]$;
    Determine the final certainty coefficient $\mathbf{p} = \frac{down(u)+up(u)}{2}$ for the object u;
  else
    for $i := 1$ **to** m **do**
      Compute certainty coefficient ("main class" membership probability) for the given test object u using the classifier C_i and assign it to p_i ;
    end
    Determine the uncertainty interval $[down(u), up(u)]$ for the object u by aggregating (with the use of $\mathcal{A}$) the intervals $[p_1, p_1]$, ..., $[p_m, p_m]$;
    Determine the final certainty coefficient $\mathbf{p} = \frac{down(u)+up(u)}{2}$ for the object u;
  end
end

largest positive answer, i.e., $arg\max_{i=1,...,s} \mathbf{p_i}$, where $\mathbf{p_i}$ for $i = 1, ..., s$ denotes the certainty coefficient returned by the given algorithm representing the probability of belonging the object to the "main class" (cf. Algorithm 1 and Algorithm 2). In some cases, this positive output is not unique and some tie-breaking techniques are required.

The computational time complexity of the multi-class classifier MF is not large and the algorithm can be used in practical applications wherever the k-NN algorithm can be used. If we assume that the classical k-NN algorithm works in time of order $O(n \cdot l)$, where n is the number of objects in the set U, and l is the number of attributes in the set A, then it is easy to see that the time complexity of the Algorithm MF (with the binary version given by Algorithm 2) is of order $O(s \cdot m \cdot r \cdot n \cdot l)$, where m is the number of classifiers from the collection $C_1, ..., C_m$, r is the parameter of Monte Carlo method, s is the number of binary classifications performed.

IV. Implementation details

The implementation of both MM and MF classifiers was made in Python language with the usage of its libraries NumPy, pandas and scikit-learn. In fact the binary versions of these algorithms, denoted as BM and BF were implemented directly. The only difference is that binary versions of classifiers in [33] return decision classes and due to the needs of scikit-learn the present implementations return $\mathbf{p}$, i.e., the probability of belonging to the main class. The multi-class versions of both algorithms MM and MF were received by using scikit-learn's OneVsRestClassifier class. This class is intended to transform a binary classifier into multi-class classifier (this class made a previously descripted decomposition of a multi-class classifier into a number of binary problems). For each decision class there is trained a binary classifier, which distinguishes between this class as a main class and all other classes as a subordinate class. OneVsRestClassifier class holds the obtained probabilities as a vector and the final decision of an instance is determined based on it (the vector is normalized and the decision with maximum probability is chosen).

Since random insert of missing data in each binary algorithm BF is independent, there is a possibility, especially in small datasets and for small k that testing object will be "moved" to different training objects each time (perhaps with different decision). Due to this fact, it is possible that multi-class MF decision vector will be a vector of zeros. In such case, which may be considered a tie, we choose the first decision class. This approach gives sufficient classification quality results, however more approaches should be proposed and studied in the future. To measure performance of both algorithms the AUC (Area under ROC) was applied. We used a multi-class extension of the commonly known binary AUC, proposed by Hand and Till in [37]. This measure was recently implemented in leading data mining Python library, i.e., scikit-learn [38].

We examined our algorithm on 8 data sets, downloaded from University of California, Irvine repository UCI [39]. We provide their full names in UCI repository and a shortcut name in parentheses if full name is long. The data sets are

the following: Iris, Balance Scale, Vertebral Column (Vertebral), Seeds, Wall-Following Robot Navigation Data (Wall-Following), Breast Tissue, Wireless Indoor Localization (Wifi), Leaf. Because our approach is intended to work with numerical data, we have only taken such attributes into consideration. In some data sets, there is a situation, where there exist multiple versions of a data set with different number of attributes or decisions (for example data set may have a multi-class or a binary decision attribute version). Moreover, in some cases we reduced the number of objects. Firstly, in order to balance the unbalanced data. Secondly, to use one validation method for all data sets (the methods may differ depending on the number of objects). The experiments were made using stratified k-fold cross validation.

TABLE I
DATASETS INFORMATION

UCI data	Objects	Attributes	Classes
Iris	150	4	3
Balance Scale	625	4	3
Vertebral	310	6	3
Seeds	210	7	3
Wifi	2000	7	4
Wall-Following	5456	24	4
Breast Tissue	106	10	6
Leaf	340	7	30

Preprocessing of data sets details (cf. [40], [41], [42]) are as follows:

- Iris - all conditional attributes and all objects were taken.
- Balance Scale - all conditional attributes and all objects were taken.
- Vertebral - a three decision classes version was chosen (also binary version exists), all conditional attributes and all objects were taken.
- Seeds - all conditional attributes and all objects were taken.
- Wifi - all conditional attributes and all objects were taken.
- Wall-Following - the 4 conditional attributes version was taken; for all decision classes we randomly selected 200 objects, so we have chosen 800 objects in total.
- Breast Tissue - all conditional attributes were chosen (except of id) and all objects were chosen.
- Leaf - this data set contains color photos and their binarized versions (not for all classes). It is worth to mention that description of the data set claims that it contains 40 different classes. It is true that there is 40 classes of color photos. However, in fact there is 30 classes in csv file, which is equal to number of binarized photos classes. We have chosen all attributes and all objects.

The source code in Python and steps to reproduce results can be found in [43].

V. DISCUSSION ON THE RESULTS

The first expected conclusion is that the increase of the level of missing values resulted in the decrease of classification quality measured by AUC. However, the classification method MF always gave better classification results than the method MM in the case of appearance of missing values. The level of missing values was: 0, 0.01, 0.03, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5 (for example, if the parameter was 0.2, then 20% of the missing values was randomly entered into every attribute column). In Table II we present only some of the levels of missing values and some of the data sets. The full results can be found in [44].

The observed results were justified with the use of statistical tests in the Statistica program [45]. Using the pairwise Wilcoxon test it can be noticed that there are statistically significant differences in the distribution of the average values of AUC obtained for the algorithm MF and MM ($p = 0.01729$). Pearson correlation coefficient (related to the further results with linear regression), determined for both methods, proved with the level of $p < 0.0001$ the negative correlation between the missing values and the values of AUC (Kendall rank correlation coefficient and Spearman's rank correlation coefficient were also applied with similar results).

Moreover, we considered the linear regression of changes in the average AUC values depending on the levels of missing values (cf. Figure 1). The linear correlations proved to be very strong. We proved that the gradients of the lines are equal. The test has shown that on the level of $p < 0.01$ such equality holds. Moreover, the straight line for the algorithm MF lies statistically significantly above the straight line for the algorithm MM. Therefore, the applied tests proved that there are statistically significant changes between the MF method and the MM method. Since the experiments for each data set were repeated 10 times, we report in the tables also the standard deviation ([44]). The values of standard deviation are small (particularly for the method MF) which means that the results of experiments are stable. In Figure 1, the values of standard deviation for both methods are depicted by dashed lines.

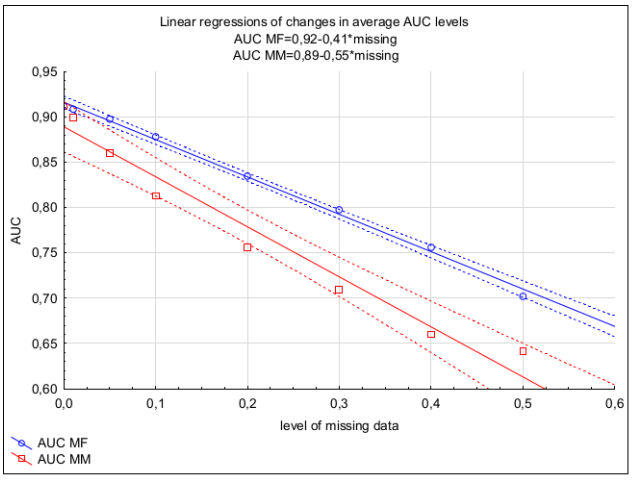

Fig. 1. Changes in average AUC depending on the levels of missing values

We also performed further analysis with respect to the

aspect of dependency/independency of experiments in the context of amount of missing values. Treating the experiments of extracting samples of missing values from the data sets as dependent experiments in the case of MF, using the Friedman test, we obtained statistically significant differences in the average level of AUC depending on the missing values ($p < 0.00001$). In this case, by comparing each two samples, for two different amounts of missing values, we always get statistically significantly different AUC ($p < 0.00001$ with Bonferroni correction) for each comparison. Similar results were obtained for the method MM. Namely, in the case of MM using the Friedman test we obtained statistically significant differences in the level of average AUC depending on the missing values ($p < 0.00001$). For the method MM, comparing each two samples, for two different amounts of missing values (except for the missing values $0,4$ and $0,5$, with $p = 0,1$), we get statistically significant differences in the level of AUC for each comparison ($p < 0,0003$ with Bonferroni correction). Additionally, the analysis for both methods was performed with respect to the number of neighbors k. The analysis for the case $k = 2,4$ and $k = 3,5$, performed separately for each case, gave similar results to the general case (without distinguishing parameters k) discussed above. Furthermore, the performance of the method MF with respect to the specific parameters for this method, i.e., the type of an aggregation applied and the parameter r, was analyzed. It turned out that a choice of an aggregation function may increase the performance of the classifier (with respect to the data set and the level of missing values) but these are not statistically significant changes. Statistically significant results for the method BF (involving interval modelling but other data sets) were obtained for some aggregation operators (cf. [33]). The aggregation operator $\mathcal{A}_6$, and next $\mathcal{A}_2$, $\mathcal{A}_3$ obtained the best results (with statistically significant values) with respect to the respective level of missing value. It seems that the reason for non-obtaining statistically significant results with respect to the aggregation operator in the multi-class case is the decomposition method applied here for the multi-class classification. The significance of single results of the binary classifiers and aggregation operators are declining. If it comes to the parameter r, there are statistically significant changes between the average values of AUC (analysis of variance and the non-parametrical Kruskal-Wallis test were considered with $p < 0.002$ in both cases). In order to perform pairwise comparison with respect to considered values of r we applied post-hoc test with Bonferroni correction. For $r = 2$ the values of AUC are statistically smaller than that of $r = 5$ and $r = 10$ (with $p < 0.03$ in both cases). It may be explained by the role of r in the method F, where it was used to determine the uncertainty intervals. For a small number of r the domain knowledge reflected by the interval may be lacking stability. The increase of r was the advantage but it is enough to increase r to the reasonable value. The post-hoc test with Bonferroni correction has shown that there are no statistically significant changes between values $r = 5$ and $r = 10$.

To sum up, the presented method MF has advantages and disadvantages. The main advantage is good classification result but the choice of the aggregation operator to obtain the best result depends on the data set and the level of missing values. Moreover, the presented algorithm MF is dedicated for numerical attributes. In its present form it is not suitable for categorical attributes.

TABLE II
SELECTED RESULTS

Level of missing values	Dataset	Parameters, algorithms and their highest AUC			
		k, r, agg	MF	k	MM
0.0	Iris	(3,5), 2, A1	0.997	(3,5)	0.997
	Balance	(3,5), 2, A6	0,742	(2,4)	0,736
	Vertebral	(2,4), 2, A5	0,877	(2,4)	0,875
	Seeds	(3,5), 2, A1	0,986	(3,5)	0,986
	Wifi	(3,5), 2, A9	0,996	(3,5)	0,996
	Wall	(3,5), 2, A1	0,908	(3,5)	0,908
	Breast	(3,5), 2, A6	0,885	(3,5)	0,881
	Leaf	(3,5), 2, A5	0,930	(3,5)	0,929
0.1	Iris	(3,5), 10, A6	0,997	(2,4)	0.910
	Balance	(3,5), 10, A1	0,768	(3,5)	0,726
	Vertebral	(3,5), 2, A8	0,871	(2,4)	0,722
	Seeds	(2,4), 5, A6	0,983	(3,5)	0,943
	Wifi	(3,5), 10, A2	0,992	(3,5)	0,930
	Wall	(3,5), 10, A3	0,913	(3,5)	0,816
	Breast	(3,5), 2, A6	0,895	(3,5)	0,771
	Leaf	(3,5), 10, A5	0,849	(3,5)	0,739
0.3	Iris	(2,4), 10, A1	0,978	(3,5)	0,766
	Balance	(3,5), 5, A6	0,733	(2,4)	0,681
	Vertebral	(3,5), 10, A5	0,841	(3,5)	0,652
	Seeds	(2,4), 5, A6	0,969	(3,5)	0,858
	Wifi	(3,5), 10, A7	0,964	(3,5)	0,789
	Wall	(3,5), 10, A1	0,876	(3,5)	0,656
	Breast	(3,5), 5, A5	0,780	(3,5)	0,746
	Leaf	(3,5), 5, A6	0,629	(2,4)	0,599
0.5	Iris	(2,4), 10, A1	0,933	(3,5)	0,774
	Balance	(2,4), 10, A1	0,685	(3,5)	0,649
	Vertebral	(2,4), 10, A4	0,770	(3,5)	0,560
	Seeds	(3,5), 5, A2	0,895	(3,5)	0,853
	Wifi	(2,4), 10, A1	0,885	(2,4)	0,708
	Wall	(2,4), 10, A1	0,794	(2,4)	0,583
	Breast	(3,5), 10, A8	0,729	(3,5)	0,635
	Leaf	(3,5), 5, A5	0,579	(2,4)	0,553

VI. CONCLUSIONS

In this contribution it was considered the problem of missing values in the data sets. An effective method of dealing with missing values was proposed. It was proved that interval modeling and aggregation methods allow to obtain a much slower decrease of the multi-class classification quality comparing to the simple aggregation method of k-NN classifiers. In the case of interval modelling it was shown that diverse interval-valued aggregation operators enable to choose the one which is the most suitable for a given data set and a level of missing values. For the future work we plan to develop our algorithm for multi-class classification in the case of missing values with other decomposition methods or generally other methods of multi-class classification. Moreover, we would like to adjust our algorithm to the wider range of attribute types.

ACKNOWLEDGEMENTS

This work was partially supported by the Centre for Innovation and Transfer of Natural Sciences and Engineering Knowledge of University of Rzeszów, Poland, the project RPPK.01.03.00-18-001/10. The presented work is also partly a continuation of the research carried out under the grant of the Polish National Science Centre DEC-2013/09/B/ST6/01568.

REFERENCES

[1] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes", Pattern Recognition, vol. 44, pp. 1761–1776, 2011.

[2] A.C.Lorena, A.C.Carvalho, and J.M.Gama, "A review on the combination of binary classifiers in multiclass problems", Artificial Intelligence Review, vol. 30, pp. 19–37, 2008.

[3] K. Dyczkowski, A. Wójtowicz, P. Żywica, A. Stachowiak, R. Moszyński, and S. Szubert, "An intelligent system for computer-aided ovarian tumor diagnosis', in *Intelligent Systems'2014*, D. Filev et al. eds., Cham, pp. 335–343, Advances in Intelligent Systems and Computing, vol. 323, Springer, 2015.

[4] M. Štěpnička, N. Cao, L. Běhounek, M. Burda, and A. Dolný, "Missing values and dragonfly operations in fuzzy relational compositions", Int. J. Approx. Reason., vol. 113, pp. 149–170, 2019.

[5] A. Wójtowicz, P. Żywica, A. Stachowiak, and K. Dyczkowski, "Solving the problem of incomplete data in medical diagnosis via interval modeling", Applied Soft Computing, vol. 47, pp. 424–437, 2016.

[6] P. Żywica, K. Dyczkowski, A. Wójtowicz, A. Stachowiak, S. Szubert, and R. Moszyński, "Development of a fuzzy-driven system for ovarian tumor diagnosis", Biocybernetics and Biomedical Engineering, vol. 36, no. 4, pp. 632–643, 2016.

[7] M. Elkano, J.A. Sanz , M. Galar, B. Pękala, U. Bentkowska, H. Bustince, "Composition of interval-valued fuzzy relations using aggregation functions", Inform. Sci., vol. 369, pp. 690–703, 2016.

[8] U. Bentkowska, B. Pękala, H. Bustince, J. Fernandez, A. Jurio, K. Balicki, "N-reciprocity property for interval-valued fuzzy relations with an application to group decision making problems in social networks", International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 25 (Suppl. 1), pp. 43–72, 2017.

[9] B. Pękala, "Operations on interval matrices", in Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), M. Kryszkiewicz et al. eds., pp. 613–621, vol. 4585, Springer, 2007.

[10] B. Pękala, U. Bentkowska, H. Bustince, J. Fernandez, M. Galar, "Operators on intuitionistic fuzzy relations", Proc. FUZZ IEEE 2015, 6 pages, 2015.

[11] B. Pękala, Uncertainty Data in Interval-Valued Fuzzy Set Theory: Properties, Algorithms and Applica-tions, vol. 367, Studies in Fuzziness and Soft Computing, Springer, 2019.

[12] D. Dubois and H. Prade, Possibility Theory, Plenum Press, New York, 1988.

[13] D. Dubois and H. Prade, "Gradualness, uncertainty and bipolarity: Making sense of fuzzy sets", Fuzzy Sets Syst., vol. 192, pp. 3–24, 2012.

[14] L.A. Zadeh, "The Concept of a Linguistic Variable and its Application to Approximate Reasoning-I", Inform. Sci., vol. 8, pp. 199–249, 1975.

[15] G. Beliakov, H. Bustince, and T. Calvo, A Practical Guide to Averaging Functions, Studies in Fuzziness and Soft Computing, Springer, Switzerland, 2016.

[16] K. Dyczkowski, Intelligent Medical Decision Support System Based on Imperfect Information. The Case of Ovarian Tumor Diagnosis, Studies in Computational Intelligence, Springer, 2018.

[17] T. Bailey and A. Jain, "A note on distance-weighted k-nearest neighbor rules", IEEE Trans. Systems, Man, Cybernetics, vol. 8, pp. 311–313, 1978.

[18] J. G. Bazan, S. Bazan-Socha, M. Ochab, S. Buregwa-Czuma, T. Nowakowski, and M. Woźniak, "Effective construction of classifiers with the k-NN method supported by a concept ontology", Knowl. Inf. Syst., vol. 62, pp. 1497–1510, 2020.

[19] S.A. Dudani, "The distance-weighted k-nearest-neighbor rule", IEEE Trans. Syst. Man Cybern., SMC, vol. 6, pp. 325–327, 1976.

[20] U. W. Kulish and W. L. Miranker, Computer arithemtic in theory and practice, Academic Press, 1981.

[21] B. Pękala, U. Bentkowska, and B. De Baets, "On comparability relations in the class of interval-valued fuzzy relations", Tatra Mountains Math. Publ., vol. 66, pp. 91–101, 2016.

[22] K. Dembczyński, S. Greco, and R. Słowiński, "Rough set approach to multiple criteria classification with imprecise evaluations and assignments", Eur. J. Oper. Res., vol. 198, pp. 626–636, 2009.

[23] S. Karmakar and A.K. Bhunia, "A comparative study of different order relations of intervals", Reliab. Comput., vol. 16, pp. 38–72, 2012.

[24] D.S. Scot, "Outline of a mathematical theory of computation", in 4th Annual Princeton Conference on Information Sciences and Systems, pp. 169–176, 1970.

[25] H. Bustince, J. Fernandez, A. Kolesárová, and R. Mesiar, "Generation of linear orders for intervals by means of aggregation functions", Fuzzy Sets Syst., vol. 220, pp. 69–77, 2013.

[26] U. Bentkowska, "New types of aggregation functions for interval-valued fuzzy setting and preservation of pos-B and nec-B-transitivity in decision making problems", Inform. Sci., vol. 424, pp. 385–399, 2018.

[27] T. Calvo, A. Kolesárová, M. Komorniková, R. Mesiar, "Aggregation operators: properties, classes and construction methods", in T. Calvo et al. (eds.) Aggregation Operators, pp. 3–104. Physica-Verlag, Heidelberg, 2002.

[28] M. Komorníková and R. Mesiar, "Aggregation functions on bounded partially ordered sets and their classification", Fuzzy Sets Syst., vol. 175, pp. 48–56, 2011.

[29] G. Deschrijver, "Arithmetic operators in interval-valued fuzzy set theory", Inform. Sci., vol. 177, pp. 2906–2924, 2007.

[30] H. Zapata, H. Bustince, S. Montes, B. Bedregal, G.P. Dimuro, Z. Takáč, M. Baczyński, J. Fernandez, "Interval-valued implications and interval-valued strong equality index with admissible orders", Internat. J. Approx. Reason., vol. 88, pp. 91–109, 2017.

[31] P.S. Bullen, D.S. Mitrinović, P.M. Vasić, Means and Their Inequalities, Reidel, Dordrecht, 1988.

[32] U. Bentkowska, Interval-valued methods in Classifications and Decisions, Studies in Fuzziness and Soft Computing, vol. 378, Springer 2020.

[33] U. Bentkowska, J.G. Bazan, W. Rząsa, and L. Zaręba, "Application of interval-valued aggregation to optimization problem of k-NN classifiers for missing values case", Inform. Sci., vol. 486, pp. 434–449, 2019.

[34] T. De Waal, J. Pannekoek, S. Scholtus, Handbook of statistical data editing and imputation, John Wiley and Sons, vol. 563, 2011.

[35] E. Frank, M.A. Hall, and I.H. Witten, The WEKA Workbench, Online Appendix for Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, Fourth Edition, 2016.

[36] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten, The WEKA Data Mining Software: An Update, SIGKDD Explorations, vol. 11, Issue 1, 2009.

[37] D. J. Hand and R. J. Till, "A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems", Machine Learning, vol. 45, pp. 171–186, 2001.

[38] https://scikit-learn.org/stable/modules/model_evaluation.html#roc-metrics

[39] D. Dua and C. Graff, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2019.

[40] R. Bhatt, "Fuzzy-Rough Approaches for Pattern Classification: Hybrid measures, Mathematical analysis, Feature selection algorithms, Decision tree algorithms, Neural learning, and Applications", Amazon Books, 2017.

[41] J. G. Rohra, B. Perumal, S. J. Narayanan, P. Thakur, and R. B. Bhatt, "User Localization in an Indoor Environment Using Fuzzy Hybrid of Particle Swarm Optimization & Gravitational Search Algorithm with Neural Networks", in Proceedings of Sixth International Conference on Soft Computing for Problem Solving, pp. 286–295, 2017.

[42] P. F.B. Silva, A. R.S. Marcal, and R. M. Almeida da Silva, "Evaluation of Features for Leaf Discrimination", Springer Lecture Notes in Computer Science, vol. 7950, pp. 197–204, 2013.

[43] https://github.com/furoDMGroup/Multi-class-classification-problems-for-the-k-NN-in-the-case-of-missing-values

[44] https://furodmgroup.github.io/Multi-class-classification-problems-for-the-k-NN-in-the-case-of-missing-values/

[45] http://www.statsoft.com