A new method of building a more effective ensemble classifiers

1st Jan G. Bazan
University of Rzeszów
Rzeszów, Poland
bazan@ur.edu.pl

2nd Paweł Drygaś
University of Rzeszów
Rzeszów, Poland
paweldr@ur.edu.pl

3rd Lech Zaręba
University of Rzeszów
Rzeszów, Poland
lzareba@ur.edu.pl

4th Piotr Molenda
University of Rzeszów
Rzeszów, Poland
pmole@ur.edu.pl

***Abstract*—In this contribution the new method for selecting classifiers to build an ensemble classifier is presented. Sometimes we get many classifiers that classify an object based on various premises (attributes). Many of them are of low quality, therefore a new classifier is being built that takes into account the weight of individual classifiers. In general, it gives better results than individual classifiers. However, the use of all classifiers (especially those of low quality) does not always give satisfactory results. Therefore, we present a method that allows to eliminate some classifiers while increasing the quality of classification.**

***Index Terms*—aggregation function, classifiers**

I. Introduction

The main task of the classification constituting one of the important methods of data mining is the creation of models, called classifiers (also classifying algorithms or decision algorithms), describing dependencies between the given class (category) of objects and their characteristics. Discovered classification models are then used to classify new objects of the unknown class membership (see e.g., [20], [3], [4]). We will consider a classification problem based on a finite set of observations, consisting in assigning to one of two classes (e.g. positive and negative decision). This finite set of observations will be represented using data tables. In this representation individual observations correspond to rows of a given data table and attributes to columns of a given data table. In this paper, we consider decision tables of the form $\mathbf{T} = (U, A, d)$ in Pawlak's sense (cf. [22]) for representation of data tables, where U is a set of objects (rows) in the data table, A is a set of attributes or columns in the data table, and d is a distinguished attribute from the set A called a decision attribute (in this paper, we consider problems for the case of a 2-class classification, e.g., for decision classes YES and NO or for decision classes 1 and 0, etc.).

The classifier assigns to the object a certain weight (classification coefficient) to classify the object. For the threshold parameter $t \in (0, 1)$, if the classification weight of the test object obtained from the classifier is greater than t, the object is classified into the main class (e.g., YES). However, if the weight is less than or equal to t, then the object is classified into a subordinate class (e.g., NO).

When classifying objects, we can construct different classifiers. Often the decisions obtained differ for a some elements. Therefore, a conflict appears between the classifiers that operate on the basis of different sources or parameters, which must be resolved in order to finally classify the test object. For this purpose we suggest aggregation of values obtained by the individual classifiers using aggregation operators. As a result, we build a new compound classifier. Such complex classifiers are often called ensemble classifiers in the literature [25].

Moreover, sometimes many of constructed individual classifiers are of low quality, therefore a new classifier takes into account the weight of individual classifiers. In general, it gives better results than individual classifiers. However, the use of all classifiers and their later aggregation is sometimes very expensive. In addition, because some classifiers are of low quality, they degrade the quality of the final classifier. Therefore, we present a method that allows you to eliminate some classifiers while increasing the quality of classification.

Well, our approach is characterized by, compared to the majority of existing ones, that classifiers are not only aggregated, but dynamically selected when testing a particular test object. The purpose of this selection is to improve the global quality of the classification and it is based on the raw results of the test object classification by all aggregated classifiers or based on certain selection parameters learned from training data.

The paper is structured as follows. In Section II, notions connected with aggregation operators are recalled. In Section III, the motivation to consider new versions of classifier are provided as well as a description of them is given.

II. Agregations operators

Firstly, we recall definition of an aggregation function. More details can be found in [10], [16], [21], [6], [8], [12], [19], [24]

Definition 1 (cf. [9])**.** A function $A : [0, 1]^n \to [0, 1]$, $n \in \mathbb{N}$, $n \geq 2$, which is increasing in each variable, i.e.,

$$(\forall_{1 \leq i \leq n} x_i \leq y_i) \Rightarrow A(x_1, \ldots, x_n) \leq A(y_1, \ldots, y_n), \quad (1)$$

for all $x_1, \ldots, x_n, y_1, \ldots, y_n \in [0, 1]$ is called an aggregation function (aggregation operator) if $A(0, \ldots, 0) = 0$, $A(1, \ldots, 1) = 1$.

Definition 2 ([9])**.** Let $n \geq 2$. $A : \mathbb{R}^n \to \mathbb{R}$ is a mean (average function) if it is increasing and idempotent, i.e., for all $x, y \in \mathbb{R}^n$ (1) is satisfies and

$$\forall_{x \in \mathbb{R}} A(x, ..., x) = x.$$

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

Lemma 1. *For every mean A we have*

$$\forall x \in \mathbb{R}^n \min_{1\leq k\leq n} x_k \leq A(x_1, ..., x_n) \leq \max_{1\leq k\leq n} x_k. \quad (2)$$

From the above lemma we see that the mean can be restricted to any interval. Our domain of interest is the interval $[0, 1]$. In this case, the mean is the aggregation function.

Example 1. Let $\varphi : [0, 1] \to [0, 1]$ be an increasing bijection and $x, w \in [0, 1]^n$. We remind here two important examples of aggregation function: the quasi-arithmetic mean (cf. [1])

$$A(x_1, ..., x_n) = \varphi^{-1}(\frac{1}{n}\sum_{k=1}^{n} \varphi(x_k)),$$

and the generalized weighted average (cf. [9])

$$A(x_1, ..., x_n) = \varphi^{-1}(\sum_{k=1}^{n} w_k\varphi(x_k)).$$

In this paper we use the aggregation method based on the arithmetic mean for experiments. The reason for this is that this aggregation in practice gives good classification results obtained by the ensemble classifiers based on it, very often better than other aggregations known from the literature.

III. Classifiers

During classification, the classifier assigns a certain classification weight to the object (see Fig.1). For a set range of the threshold parameters $t \in (0, 1)$, the test objects are tested in such a way that if the classification weight of the test object obtained from the classifier is greater than t, the object is classified into the main class (e.g., YES). However, if the weight is less than or equal to t, then the object is classified into a subordinate class (e.g., NO). In this way, we obtain the decision value for the test object, which may be correct (consistent with the actual decision in the test table) or incorrect (we make a mistake in the classification).

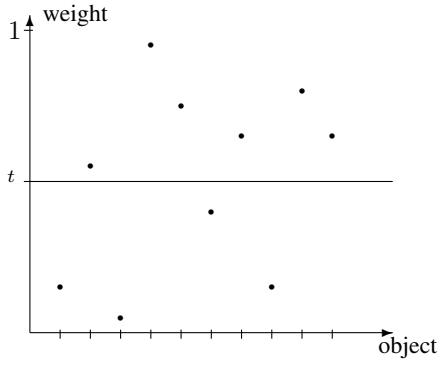

Fig. 1. Certain classification weights assigned by the classifier to the object and the threshold parameter $t \in (0, 1)$

To calculate the global classification quality of a given classifier with the fixed parameter t we use the accuracy (ACC) of the classification which is the quotient of the number of correct classifications to the number of all classifications.

Using the following notion:

- TP - True Positives - elements from the main class classified into the main class
- TN - True Negatives - elements from the subordinate class classified into the subordinate class
- FP - False Positives - elements from the subordinate class classified into the main class
- FN - False Negatives - elements from the main class classified into the subordinate class
- ACC - accuracy

we can calculate the accuracy according to the following formula

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}$$

We can also put above information in the table, where the rows contain elements from the main class and subordinate class, respectively, while the columns contain elements classified by the classifier into the main class and subordinate class, respectively.

TABLE I
Table of parameters

Elements from	classified by the classifier as main class	subordinate class
main class	TP	FN
subordinate class	FP	TN

Accuracy calculated for the test objects from the main class is called sensitivity (TPR - true positive rate), and the accuracy calculated for the test objects from a subordinate class we call specificity (TNR - true negative rate). In addition, we will consider FPR - false positive rate. Using the above notion we can calculate mentioned parameters according to the following formulas

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP}$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$$

$$FPR = 1 - TNR$$

If the sensitivity is unsatisfactory, e.g., in medicine when trying to predict the occurrence of a disease of a patient, it may turn out that the sensitivity of the classification to the main class "sick" is too low, we can balance between sensitivity and specificity, i.e., increasing sensitivity at the expense of decreasing specificity. This approach leads to the concept of the ROC curve (receiver operating characteristic curve), where each point of the ROC curve corresponds to one setting of the classifier's performance (the parameter t).

ROC shows the dependence of sensitivity on error of the first type (FPR) during calibration of the classifier (at various threshold settings).

AUC is the indicator of the quality of a classifier which is the area under the ROC (cf. [14], [23]). The greater is the AUC value the better is the classifier.

A. Classification algorithm

When classifying objects, we can construct different classifiers (based on different systems or based on different data sources, e.g., using several diagnostic devices). Often the decisions obtained differ for a certain class of test elements. Therefore, a conflict appears between the classifiers that operate on the basis of different sources or parameters, which must be resolved in order to finally classify the test object. To get a final decision, we should create a new classifier that will take into account previous results. For this purpose we suggest aggregation of values obtained by the individual classifiers. As a result, we build a new compound classifier.

In this article, we use aggregating of the classification weights obtained by individual classifiers, and we propose the Algorithm 1 (method WAS - weight arithmetic mean selection). This algorithm uses M aggregation, which is based on an arithmetic mean.

Algorithm 1: Classification of a test object by the M classifier

Input:
1) training data set represented by decision table $\mathbf{T} = (U, A, d)$,
2) collection $C_1, ..., C_m$ of classifiers,
3) test object u,
4) aggregation M,
5) threshold parameter t, e.g., $t = 0.6$.

Output: The membership of the object u to the main class or to the subordinate class

begin
  for $i := 1$ **to** m **do**
    Compute a certain weight ("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 weight p for the object u by aggregating (with a use of the aggregation operator M e.g., arithmetic mean) the weights p_1,...,p_m.
  if $p > t$ **then**
    return u *belongs to the "main class"*
  else
    return u *belongs to the "subordinate class"*
  end
end

B. Modification of the algorithm

Some classifiers assigns a weight to an object that differ from the weights of other classifiers or their aggregation obtained in the Algorithm 1

That is why we decided to check their impact on the quality of the classification. In other words, we decided to check whether eliminating these classifiers would improve the quality of the classification.

The problem that appeared here was the choice of classifiers that we remove, or in other words - the choice of classifiers that we will use for classification.

Here we propose three methods to choose from classifier which we will use later in the classification.

The first method is to choose those classifiers that give the weights closest to the aggregate value, i.e., those that are the most distant from the aggregate value are rejected (method WTS weight threshold selection), see Algorithm 2.

Algorithm 2: Classification of a test object by the WTS classifier.

Input:
1) data set represented by decision table $\mathbf{T} = (U, A, d)$, with $card\ U = n$,
2) collection $C_1, ..., C_m$ of classifiers,
3) test object u,
4) aggregation M,
5) threshold parameter t, e.g., $t = 0.6$,
6) parameter ε, e.g., $\varepsilon = 0.8$.

Output: The membership of the object u to the main class or to the subordinate class

begin
  for $i := 1$ **to** m **do**
    Compute a certain weight ("main class" membership probability) for the given test object u using the classifier C_i and assign it to p_i
  end
  Determine the weight p' for the object u by aggregating (with a use of the aggregation operator M e.g., arithmetic mean) the weights p_1,...,p_m.
  for $i := 1$ **to** m **do**
    Compute a distance d_i between p' and p_i for the given test object u
  end
  Choose the classifiers for which $d_i < \varepsilon$. In this way we receive the sets $K = \{C_{s_1}, ..., C_{s_k}\}$
  Determine the final weight p for the object u by aggregating (with a use of the aggregation operator M) the weights p_{s_1},...,p_{s_k}.
  if $p > t$ **then**
    return u *belongs to the "main class"*
  else
    return u *belongs to the "subordinate class"*
  end
end

Unfortunately, for small values of ε we can get an empty set of K. That is why we suggest modifying this algorithm by selecting a certain percentage of classifiers for each test

object. The selection method will be presented in Algorithm 3 (method WPS - weight percent selection).

Algorithm 3: Classification of a test object by the WPS classifier.

Input:
1) data set represented by decision table $\mathbf{T} = (U, A, d)$, with $card\ U = n$,
2) collection $C_1, ..., C_m$ of classifiers,
3) test object u,
4) aggregation M,
5) threshold parameter t, e.g., $t = 0.6$,
6) parameter $r \in (0, 1]$.

Output: The membership of the object u to the main class or to the subordinate class

begin
  for $i := 1$ **to** m **do**
    Compute a certain weight ("main class" membership probability) for the given test object u using the classifier C_i and assign it to p_i
  end
  Determine the weight p' for the object u by aggregating (with a use of the aggregation operator M e.g., arithmetic mean) the weights p_1,...,p_m.
  for $i := 1$ **to** m **do**
    Compute a distance d_i between p' and p_i for the given test object u
  end
  Choose $100 \cdot r$ % classifiers that are closest to the aggregate value p'. In this way we receive the sets $K = \{C_{s_1}, ..., C_{s_k}\}$
  Determine the final weight p for the object u by aggregating (with a use of the aggregation operator M) the weights p_{s_1},...,p_{s_k}.
  if $p > t$ **then**
    return *u belongs to the "main class"*
  else
    return *u belongs to the "subordinate class"*
  end
end

The third construction, which differs from the previous two, consists in the selection of classifiers that have been recognized as the most stable based on the training set. Based on training data, we determine accuracy, sensitivity and specificity for each classifier(see Fig. 2). We choose those for which the distance between the point of intersection of sensitivity and specificity and the maximum value of accuracy is the smallest. The selection method will be presented in Algorithm 4 (method SSAS - sensitivity, specificity, accuracy selection).

IV. Experiments

According to the above algorithms, we have m classifiers as input. Because of this, we need a set of classifiers to check our algorithms. Here we will use *k-NN* classifiers. Therefore we remind this concept.

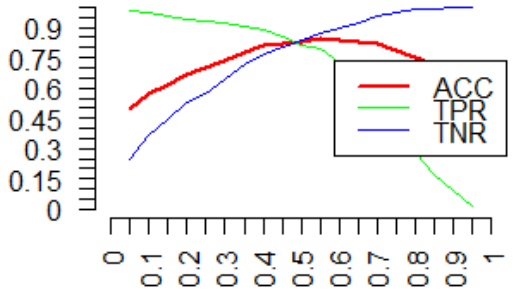

Fig. 2. Graph of dependence of accuracy, sensitivity and specificity on the threshold parameter s

A. k-NN algorithm

In 1951 Fix and Hodges introduced a non-parametric method for pattern classification that has since become known as the k-nearest neighbor algorithm [15]. Next, some of the formal properties of the k-nearest neighbor rule were obtained [11]. The k-NN algorithm is a method for classifying objects based on the k closest training examples in a feature space. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (k is a positive integer). If $k = 1$, then the object is simply assigned to the class of its nearest neighbor. It is a type of instance-based learning.

For classification also a useful technique can be used, to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the decision than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/d$, where d is the distance to the neighbor ([13]). There were considered also another methods [7], [17], [18]) and their applications ([5]).

B. Use of Algorithms

The experiments have been performed on the 9 data sets obtained from UC Irvine (UCI) Machine Learning repository. They are listed in the Table II.

To get the C_i classifiers we use the k-nearest neighbor algorithm with different values of k. In this algorithm the Euclidean metric is applied for measuring distances. Each data set is divided into two training and test parts, in the proportion of 50% to 50%. The training part of the data is used to construct the C_i classifiers. Each experiment is repeated 10 times and the average AUC and standard deviation are reported using test part of data. We assume that all analyzed data have only two decision classes. In this work, we will mainly present the results for the data from the sets *diabetes* and *red wine*.

Consecutively, Table III shows examples of experimental results for diabetes data using Algorithms 2-4. Table IV and V shows the average AUC for the results of experiments for individual algorithms and Fig. 3, 4 shows the graphical interpretation of these results.

Algorithm 4: Classification of a test object by the SSAS classifier.

Input:

1) training data set represented by decision table $\mathbf{T} = (U, A, d)$, with $card\ U = n$,
2) collection $C_1, ..., C_m$ of classifiers,
3) collection of weight thresholds $T = \{t_1, ..., t_z\}$ used during computation of ROC curve,
4) test object u,
5) aggregation M,
6) threshold parameter t, e.g., $t = 0.6$,
7) parameter ε, e.g., $\varepsilon = 0.8$.

Output: The membership of the object u to the main class or to the subordinate class

begin
  for $i := 1$ **to** m **do**
    For the C_i classifier, perform calculations such as for determining the points of the ROC curve based on the training table and the collection T, obtaining a list of points $LP_i = P_1, ..., P_z$; assume that each point on this list is the triple (sensitivity, specificity, accuracy).
    Based on the list of LP_i determine the point $P_{ss} = \{sen_{ss}, spec_{ss}, acc_{ss}\} \in LP_i$ for which the distance between sensitivity and specificity is the smallest (closest to the point of intersection of sensitivity and specificity)
    Based on the list of LP_i determine the point $P_a = \{sen_a, spec_a, acc_a\} \in LP_i$ for which the value of accuracy is the highest
    Compute the distance d_i between $\frac{sen_{ss}+spec_{ss}}{2}$ and acc_a.
  end
  Choose $\varepsilon \cdot m$ classifiers for which the distance d_i is the smallest. In this way we receive the sets $K = \{C_{s_1}, ..., C_{s_k}\}$
  for $i := 1$ **to** k **do**
    Compute a certain weight ("main class" membership probability) for the given test object u using the classifier C_{s_i} and assign it to p_{s_i}
  end
  Determine the final weight p for the object u by aggregating (with a use of the aggregation operator M) the weights p_{s_1},...,p_{s_k}.
  if $p > t$ **then**
    return u *belongs to the "main class"*
  else
    return u *belongs to the "subordinate class"*
  end
end

As we can see on the Fig. 3 and Fig. 4 the A4 method gives the highest of all the AUC and is the most stable value without deviation up or down.

TABLE II
EXPERIMENTAL DATA SET DETAILS

UCI data	Objects	Attributes	Classes
australian	690	15	2
biodeg	1055	43	2
breast_cancer	699	11	2
diabetes	768	9	2
german	1000	25	2
ozone	2536	74	2
parkinson	1040	29	2
red_wine	1599	12	2
rethinopathy	1151	20	2

TABLE III
EXAMPLE OF RESULTS OF EXPERIMENTS FOR DIABETES DATA SET

Method	data reduction	AUC	STDDEV
WPS	0.5	0.813	0.002
$SSAS$	1.0	0.813	0.003
WTS	0.9	0.812	0.003
WTS	1.0	0.812	0.003
WPS	0.2	0.811	0.003
WPS	0.7	0.811	0.003
$SSAS$	0.8	0.811	0.004
WPS	0.8	0.81	0.002
$SSAS$	0.3	0.81	0.005
$SSAS$	0.6	0.81	0.003
$SSAS$	0.7	0.81	0.004
WTS	0.8	0.809	0.003
$SSAS$	0.4	0.809	0.004
$SSAS$	0.5	0.809	0.004
$SSAS$	0.9	0.809	0.002
$SSAS$	0.2	0.808	0.004
WPS	0.6	0.807	0.004

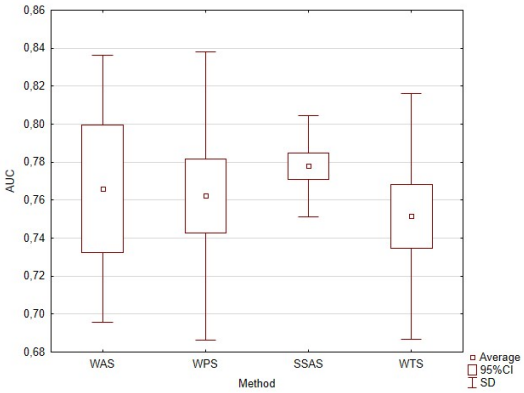

Fig. 3. The results of AUC for *diabetes*

V. CONCLUSIONS

In this paper, we presented three concepts of selecting classifiers to build a multi-classifier. During the experiments, we show that Algorithm 4 is the most effective, and that there are stable values obtained using Algorithm 4, while the other methods have high AUC dispersion. Therefore, we presented a method that allows to eliminate some classifiers

TABLE IV
THE AVERAGE AUC FOR INDIVIDUAL ALGORITHMS USING DATA SET *diabetes*

Method	average of AUC	STDDEV
WAS	0,765	0,071
WTS	0,752	0,065
WPS	0,762	0,076
SSAS	0,778	0,027

TABLE V
THE AVERAGE AUC FOR INDIVIDUAL ALGORITHMS USING DATA SET *red wine*

Method	average of AUC	STDDEV
WAS	0,781	0,081
WPS	0,768	0,062
WTS	0,782	0,078
SSAS	0,800	0,016

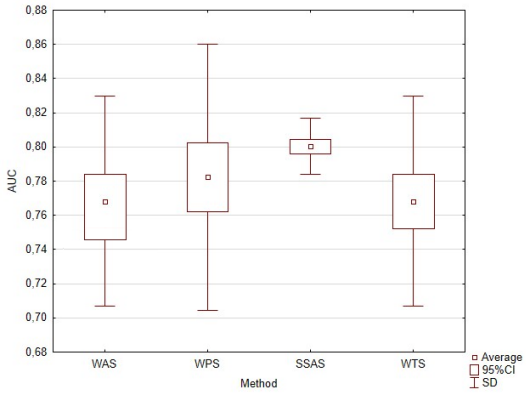

Fig. 4. The results of AUC for *red wine*

while increasing the quality of classification. In addition, if the classifiers use different attributes, using the reduction of classifiers we can decide which attributes are important and which can be omitted.

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] J. Aczél. Lectures on Functional Equations and their Applications, Academic Press, New York 1966.
[2] C. Alsina, M.J. Frank, B. Schweizer. Associative Functions. Triangular Norms and Copulas. World Scientific, New Jersey 2006.
[3] J.G. Bazan. *Hierarchical classifiers for complex spatio-temporal concepts*. Transactions on Rough Sets, IX, LNCS 5390, 2008, 474–750.
[4] J. G. Bazan, M. Szczuka. *The Rough Set Exploration System*. Transactions on Rough Sets, III, LNCS 3400, 2005, 37–56.
[5] J.G. Bazan, S. Bazan-Socha, M. Ochab, S. Buregwa-Czuma, T. Nowakowski, M. Woniak. *Effective construction of classifiers with the k-NN method supported by a concept ontology*. Knowledge and Information Systems (2019). https://doi.org/10.1007/s10115-019-01391.
[6] G. Beliakov, A. Pradera and T. Calvo. Aggregation Functions: A Guide for Practitioners. in the series: Studies in Fuzziness and Soft Computing, 221, Springer, Berlin, Heidelberg, 2007.
[7] S. Bermejo, J. Cabestany. *Adaptive soft k-nearest-neighbour classifiers*. Pattern Recognition, 33, 2000, 1999–2005.
[8] T, Calvo, G. Mayor and R. Mesiar. Aggregation Operators: New Trends and Applications. in the series: Studies in Fuzziness and Soft Computing, 97, Springer, Berlin, Heidelberg, 2002.
[9] T. Calvo, A. Kolesrov, M. Komornikov, R. Mesiar. Aggregation operators: properties, classes and construction methods. In: T. Calvo et al. (eds.) Aggregation Operators, pp. 3–104, Physica-Verlag, Heidelberg, 2002.
[10] T. Calvo, G. Mayor. *Remarks on two types of extended aggregation functions*, Tatra Mount. Math. Publ. 16, 2000, 235–253.
[11] T.M. Cover, P.E. Hart. *Nearest neighbor pattern classification*. IEEE Trans. Inform. Theory, IT 13 (1), 1967, 21–27.
[12] P. Drygaś. On the structure of continuous uninorms. *Kybernetika*, 43, 2007, 183–196.
[13] S.A. Dudani. *The distance-weighted k-nearest-neighbor rule*. IEEE Trans. Syst. Man Cybern., SMC 6, 1976, 325–327.
[14] T. Fawcett. *An introduction to ROC analysis*. Pattern Recognition Letters 27 (8), 2006, 861–874.
[15] E. Fix, J.L. Hodges. Discriminatory analysis, nonparametric discrimination: Consistency properties, Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
[16] M. Grabisch, J.L. Marichal, R. Mesiar, and E. Pap. Aggregation Functions, Encyclopedia of Mathematics and its applications, 127. Cambridge University Press, New York, 2009.
[17] A. Jozwik. *A learning scheme for a fuzzy k-nn rule*. Pattern Recognition Letters 1, 1983, 287–289.
[18] J.M. Keller, M.R. Gray, J.A. Givens. *A fuzzy k-nn neighbor algorithm*. IEEE Trans. Syst. Man Cybern., SMC 15 (4), 1985, 580–585.
[19] E.P. Klement, R. Mesiar, and E. Pap. Triangular norms. Kluwer, Dordrecht, 2000.
[20] D. Michie, D. J. Spiegelhalter, C. C. Taylor. Machine learning, neural and statistical classification. Ellis Horwood Limited, England, 1994.
[21] Y. Narukawa and V. Torra. Modeling Decisions: Information Fusion and Aggregation Operators, Springer, Berlin, Heidelberg, 2007.
[22] Z. Pawlak, A. Skowron. *Rudiments of rough sets*, Information Sciences 177, 2007, 3–27.
[23] J. A. Swets. *Measuring the accuracy of diagnostic systems*. Science 240, 1988, 1285–1293.
[24] R. R. Yager, A. Rybalov. *Uninorm aggregation operators*, Fuzzy Sets Syst., 80(1) (1996) 111–120.
[25] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, CRC Press, 2012.