IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 4, 2020, accepted November 12, 2020, date of publication November 16, 2020,
date of current version November 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3038341

Combining Linear Classifiers Using Probability-Based Potential Functions

PAWEL TRAJDOS AND ROBERT BURDUK
Department of Systems and Computer Networks, Wroclaw University of Science and Technology, 50-370 Wroclaw, Poland
Corresponding author: Pawel Trajdos (pawel.trajdos@pwr.edu.pl)

This work was supported by the National Science Centre, Poland, under Grant 2017/25/B/ST6/01750.

ABSTRACT The score function can be used as a measure for evaluating predicted probabilities of the classification models. In multiple classifiers systems, one of the problems is the diversity of the way of determining the scoring function of individual base classifiers. To alleviate this limitation, in this article, we propose a novel concept of calculating a scoring function defined by the probability-based potential function. The proposed potential functions take into account the distance of the recognized object from the decision boundary as well as a prior probability of the class labels. The proposed score function has the same nature for all linear base classifiers, which defined the multiple classifiers model. Additionally, the proposed method is compared with other ensemble algorithms based on homogeneous linear base classifiers. The experiments on seventy databases demonstrate the effectiveness of our method. To discuss the results of our experiments, we use multiple classification performance measures dedicated to standard and imbalanced datasets. The statistical analysis of the experiments is also performed.

INDEX TERMS Ensemble of classifiers, linear classifier, score function, supervised learning.

I. INTRODUCTION

The idea of building an ensemble of classifiers (EoC) is to compose a single strong model from the pool of weak or different ones. In general, EoCs improve the possibilities of individual base models (base classifiers) by building a more stable and accurate complex model [1]. The real-world classification problems solved with EoC was already mentioned in paper [2] because EoC increase the performance of individual classification models. Since then, many publications have appeared that indicate the practical applications of EoC. The network intrusion detection approach used EoC was proposed in [3] whereas paper [4] presents usefulness of EoC in detection cross-site scripting attack for web security. EoC have been also applied in many industrial fields like: the optimal stacking ensemble for remaining useful life estimation was proposed in [5], classification of cutting tools [6] or in the in-line detection of surface defects on glass substrates of thin-film transistor liquid crystal displays [7]. In addition, EoCs are used in other applications such as: the marine sediments classification [8], the land cover type classification [9] or in medical diagnostics [10]. The EoC classifiers play also an important role in the multi-label classification problems. In those classification problems, multiple classes may be simultaneously assigned to one object [11]. A possible solution to so posed classification is to decompose the multi-label problem into a series of multi-class classification tasks [12]. Then the results of created classifiers are combined to form a multi-label solution. These decomposition and aggregation are usually done using EoC approaches [13]–[15].

In general, the procedure for creating an EoC can be divided into three major steps: generation, selection, and fusion [16]. In the final phase of the EoC building, the outputs of each base classifier are combined. The classifier outcome can be represented by the class label, a subset of class labels ordered by plausibility or a vector of all possible labels with the corresponding scoring function that can reflect a measure expressed as a probability. This article focuses on a new proposition of the score function, which is defined by a probability-based potential function. This proposition significantly expands the concept of the scoring function presented in the earlier authors' work, because as proposed in [17] a scoring function is based on the distance of the recognized object from the decision boundary of a given base classifier and the distance to the class centroid. On the other hand, the article [18] presents the weighted scoring function based on Manhattan distance that uses the location of the cluster centroids and the distance to the decision boundary.

The associate editor coordinating the review of this manuscript and approving it for publication was Alberto Cano.

Given the above, the main objectives of this work can be summarized as follows:

- A proposal of a new scoring function based on a probability-based approach dedicated to linear base classifiers. We propose two versions of this potential function.
- The proposal of a new scoring function is used in the fusion process of homogeneous linear base classifiers.
- A new experimental setup for the comparison of the proposed method with other base classifiers fusion methods using different classification performance measures.

The outline of the paper is as follows: In the next section (Section II), related works are outlined. The proposed method is presented in Section III. In Section IV, the experiments that were carried out are discussed, while results and the discussion are present in Section V. Finally, we conclude the paper in Section VI.

II. RELATED WORK

In general, a classifier is a function that maps the feature space $\mathbb{X}$ into a set of class labels $\mathbb{M}$ [19]. Usually, if we talk about a classifier, we implicitly assume that the classifier is built with the use of a kind of supervised learning procedure. That is a procedure that incorporates information extracted from the training set [19]. The training set consists of training samples (taken from the feature space) and the information about the class points to which these samples belong to. In this article, it is assumed that the input space $\mathbb{X}$ is a $d-$ dimensional Euclidean space $\mathbb{X} = \mathbb{R}^d$. The paper is focused on linear (binary) classifiers. Consequently, each object from the input space $\boldsymbol{x} \in \mathbb{X}$ belongs to one of two available classes, so the output space is: $\mathbb{M} = \{-1; 1\}$.

A. LINEAR BINARY CLASSIFIER

A linear classifier produces a decision boundary that is described using a hyperplane π defined by the following equation:

$$\pi : \langle \boldsymbol{n}; \boldsymbol{x} \rangle + b = 0, \tag{1}$$

where $\boldsymbol{n}$ is a unit normal vector of the decision hyperplane, b is the distance from the hyperplane to the origin and $\langle \cdot; \cdot \rangle$ is a dot product defined as follows [20]:

$$\langle \boldsymbol{a}; \boldsymbol{b} \rangle = \sum_{i=1}^{d} a_i b_i \quad \forall \boldsymbol{a}, \boldsymbol{b} \in \mathbb{X}. \tag{2}$$

For each instance $\boldsymbol{x}$, the linear classifier ψ produces the discriminant function [21]:

$$\omega(\boldsymbol{x}) = \langle \boldsymbol{n}; \boldsymbol{x} \rangle + b. \tag{3}$$

When the normal vector of the plane is a unit vector, the absolute value of the discriminant function equals the perpendicular distance from the decision hyperplane to the point $\boldsymbol{x}$. The sign of the discriminant function depends on the site of the plane where the instance $\boldsymbol{x}$ lies.

Consequently, the decision of the linear classifier is determined using the sign of the discriminant function:

$$\psi(\boldsymbol{x}) = \text{sign}\left(\omega(\boldsymbol{x})\right). \tag{4}$$

The decision-plane coefficients ($\boldsymbol{n}$ and b) are obtained in a supervised learning procedure using the training set $\mathcal{T}$ containing $|\mathcal{T}|$ (where $|\cdot|$ is the cardinality of a set) pairs of feature vectors x and corresponding class labels m:

$$\mathcal{T} = \left\{(\boldsymbol{x}^{(1)}, m^{(1)}), (\boldsymbol{x}^{(2)}, m^{(2)}), \ldots, (\boldsymbol{x}^{(|\mathcal{T}|)}, m^{(|\mathcal{T}|)})\right\}, \tag{5}$$

where $\boldsymbol{x}^{(k)} \in \mathbb{X}$ and $m^{(k)} \in \mathbb{M}$.

B. ENSEMBLE OF LINEAR CLASSIFIERS

An ensemble classifier (or multiclassifier) is a set of properly trained classifiers whose decisions are then combined to produce the final decision of the system [22]. There are a few main reasons for building classifier ensembles [23].

First, the classifier training procedure may be interpreted as a procedure for exploring the hypothesis space. The goal is to find the best hypothesis that fits the training data. Unfortunately, a single classifier can search only a limited subspace of the entire hypothesis space. What is more, the optimization process connected with the classifier training may get stuck in one of the local optima. Those problems may be dealt with by employing a set of diverse classifiers [24].

Solving practical classification problems involves dealing with limited training/validation data. This may result in finding a set of classifiers which achieve the same classification quality. Combining responses of multiple classifiers may prevent the ensemble from choosing the wrong model [24].

The process of developing an ensemble system is divided into two main tasks: choosing the ensemble building strategy and choosing the method of output combination [22], [24].

The main goal of the ensemble building step is to provide the system with a set of accurate and diverse classifiers. Diversity of base classifiers is even more important than their high accuracy because combining classifiers whose predictions are identical gives no improvement over prediction of a single classifier [24]. There are two well-known ways of building a diverse set of classifiers. One is to build an ensemble of classifiers based on the different learning paradigm (heterogeneous ensemble) [25]. The other is to build a set of homogeneous classifiers (the same learning paradigm) which are learned on different training data. The most widely used methods of creating homogeneous ensembles are bagging [26], boosting [27], and random subspaces [28].

The second step of the ensemble creating process is to develop a method that combines outputs of the classifiers forming the ensemble (a combiner). Two methods can be used to combine the responses of base classifiers, namely output weighting methods and meta-learning [29]. In the meta-learning methods there is a need to learn at least two levels of classifiers/regressors. Classifiers on the first level are learned using object description and classifiers on higher levels are learned using outputs of classifiers from the lower

level [21], [30]. The output weighting methods can be divided into [31]:

- voting based [32] and support based [33];
- trainable [32] and untrainable [34];
- static [32], [34] and dynamic [33].

The idea of constructing ensemble classifiers has been and still is widely explored [22], [35], [36]. This is because they proved to be an efficient tool for solving many classification problems across multiple domains. Ensembles of linear classifiers have also been utilized in many practical problems like medical diagnosis [37], robotics [38] or bioinformatics [39] to name only a few. The linear classifiers owe their popularity to their low computational complexity and fair accuracy that may be achieved. They are also less overfitting prone. [37].

Now, let us define an ensemble of linear classifiers as a set of N classifiers:

$$\Psi = \left\{\psi^{(1)}, \psi^{(2)}, \cdots, \psi^{(N)}\right\}. \tag{6}$$

This article is focused mainly on the methods of combining the outputs of linear classifiers constituting the ensemble. In the literature, we may find multiple methods of doing so. The simplest strategy to combine the outcomes of multiple classifiers is to apply the majority voting scheme:

$$\omega(\boldsymbol{x}) = \sum_{i=1}^{N} \operatorname{sign}\left[\omega^{(i)}(\boldsymbol{x})\right], \tag{7}$$

where $\omega^{(i)}(\boldsymbol{x})$ is the value of the discriminant function provided by the classifier $\psi^{(i)}$ for point x. However, this simple yet effective strategy ignores the values of the discriminant function $\omega(\boldsymbol{x})$.

Another simple strategy is model averaging [40]. The output of such a model is calculated by simply averaging the values of the discriminant functions:

$$\omega(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} \omega^{(i)}(\boldsymbol{x}). \tag{8}$$

It is easy to notice that the model-averaging approach has a major drawback. That is, the discriminant function of a linear classifier is unbounded, and it grows with the distance from the decision plane. Consequently, a classifier-related decision plane placed far from the real decision boundary will produce a high value of the discriminant function that may negatively affect the ensemble. For the same reason, the outliers may acquire an abnormally high value of the discriminant function which may also affect the decision of the ensemble.

A compromise between the above-mentioned methods is to transform the discriminant function by applying a kind of sigmoid function to it [21]. The sigmoid function is a monotonic function that has finite upper and lower bounds:

$$\widetilde{\omega}^{(i)}(\boldsymbol{x}) = \left(1 + \exp\left[-\omega^{(i)}(\boldsymbol{x})\right]\right)^{-1}. \tag{9}$$

As a consequence, the distance-specific information is not lost, and the impact of the misplaced decision boundaries is reduced. This approach is equivalent to applying a kind of the logistic regression on the values of the discriminant function [41]. The value of the discriminant function may also be used to estimate the conditional probability of a class given the instance x [41], [42]. The normalized outputs are then simply averaged:

$$\omega(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} \widetilde{\omega}^{(i)}(\boldsymbol{x}). \tag{10}$$

The other issue with combining linear classifiers is that the discriminant function of the linear classifier grows monotonically with the distance to the decision plane. It means that the linear classifier ignores the data spread along the normal vector of the decision plane and it is implicitly assumed that the distribution is uniform. However, in many real-world datasets objects are distributed unevenly. An example of this situation is visualised in Fig. 1. The figure presents a binary, two-dimensional, banana-shaped dataset and the decision boundary. As we can see, the objects are placed in one cluster located in the intersection of intervals $x_1 \in [-1.5; 2.5]$, $x_2 \in [-1.5; 2]$. Outside this area, there are no class-specific objects. Consequently, the discriminant function generated by the classifier should be low outside this area. Unfortunately, a linear classifier ignores this fact and its support will grow (along the normal vector n of the decision boundary) outside this area. Transforming the discriminant function using a monotonic function, such as a sigmoid function, does not change the situation at all. This is because far from the decision boundary the discriminant function approaches its upper (lower) limit. Being close to the limit still indicates high support for a particular class in the area where there are no class-specific instances.

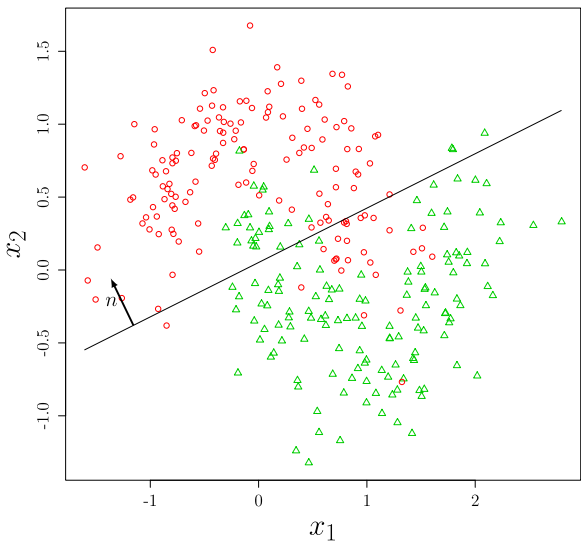

FIGURE 1. The linear decision boundary for binary, two-dimensional, banana-shaped data. The plot also shows the normal vector of the decision plane n.

Ignoring the class-spread-related information does not change the outcome of the single classifier since the sign

of the discriminant function remains the same. However, our previous research has shown that employing this information may improve the classification quality for heterogeneous ensembles of linear classifiers [17]. In the previously-proposed approach, the discriminant function is transformed using a non-monotonic function derived below:

$$g\big(\omega(\boldsymbol{x})\big) = \omega(\boldsymbol{x}) \exp\left[-\zeta\left(\omega(\boldsymbol{x})\right)^2 + 0.5\right]\sqrt{2\zeta}, \qquad (11)$$

where ζ is a coefficient that determines the position and steepness of peaks (positive and negative peaks) [see Fig. 2]. This coefficient should be tuned during the training procedure. The translation constants 0.5 and the scaling factor $\sqrt{2\zeta}$ guarantee that the maximum and minimum values (peaks) of the discriminant functions are 1 and -1 respectively. The above-mentioned non-monotonic function is visualised in Fig. 2. The figure shows the shape of the function for different values of ζ.

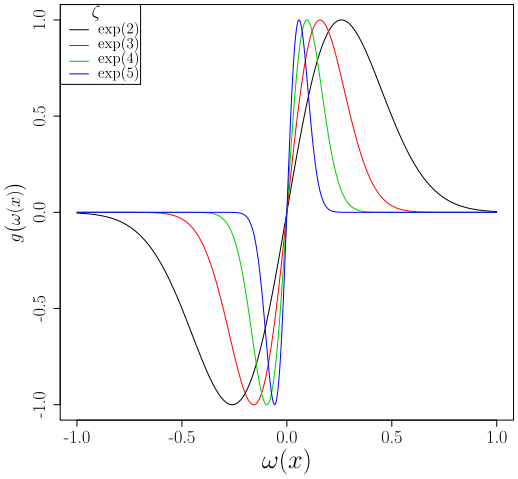

FIGURE 2. Visualisation of the potential function $g(z)$ [see (11)] for different values of ζ.

Using this transformation, the prediction of the ensemble is calculated as follows:

$$\omega(\boldsymbol{x}) = \sum_{i=1}^{N} g\left(\omega^{(i)}(\boldsymbol{x})\right). \qquad (12)$$

After combining the base classifiers, the final prediction of the ensemble is obtained according to the rule (4).

Harnessing the above-mentioned transformation allows the ensemble to improve the classification quality. This is due to the function being tuned so that the potential is near zero in the areas where there are no training points. However, when the data distribution is imbalanced, the performance may degrade [17]. The other drawback of this method is that the ζ coefficient controls the position and the steepness of the peaks simultaneously. This can be seen in Fig. 2. The higher the value of ζ is, the closer the peak is to the decision boundary, and it is narrower. The solution may be to use an asymmetric, data-driven potential function. The potential function should possess the following properties:

- The function should be bounded within a given interval;
- The function should not be a sigma-shaped one;
- The function should not be an odd function;
- The function should allow multiple peaks;
- The peaks should indicate areas where the density of class-specific instances is high.

III. PROPOSED METHOD

In this section, the probability-based potential function is introduced. The concept description is preceded by the discussion that motivates the usage of such a potential function.

A. POTENTIAL FUNCTION

The discussion has brought us to the point where we realise that an asymmetric, data-driven potential function is needed. This section describes such a function based on a probabilistic framework. The harnessing of the probabilistic model means that x and m are realizations of random variables $\mathbf{X}$ and $\mathbf{M}$, respectively. The joint distribution $P(\mathbf{X}, \mathbf{M})$ is also given. Then, the value of the discriminant function $\omega(\boldsymbol{x})$ of a linear classifier (described by its normal vector $\boldsymbol{n}$ and offset b) is also a realisation of a random variable defined as follows:

$$\mathbf{W} = \langle \boldsymbol{n}; \mathbf{X} \rangle + b. \qquad (13)$$

We denote its probability density function by $w(\omega)$. This one-dimensional distribution describes the data spread along the line defined by the normal vector n of the decision plane. More precisely, it describes the distribution of distances of points from the decision plane. This random variable is also jointly distributed with $\mathbf{M}$: $P(\mathbf{W}, \mathbf{M})$.

Taking this into consideration, the potential function β should be proportional to the conditional probability of class 1 given $\omega(\boldsymbol{x})$:

$$\beta(\boldsymbol{x}) \propto P\Big(\mathbf{M} = 1 | \omega(\boldsymbol{x})\Big). \qquad (14)$$

The conditional probability is expressed as follows:

$$P\Big(\mathbf{M} = 1 | \omega(\boldsymbol{x})\Big) = \frac{w\Big(\omega(\boldsymbol{x}) | \mathbf{M} = 1\Big) P(\mathbf{M} = 1)}{w\Big(\omega(\boldsymbol{x})\Big)} \qquad (15)$$

$$= \frac{w\Big(\omega(\boldsymbol{x}) | \mathbf{M} = 1\Big) P(\mathbf{M} = 1)}{\sum_{m \in \mathbb{M}} w\Big(\omega(\boldsymbol{x}) | \mathbf{M} = m\Big) P(\mathbf{M} = m)}. \qquad (16)$$

The potential function $\beta\Big(\omega(\boldsymbol{x})\Big)$ is given as follows:

$$\beta\Big(\omega(\boldsymbol{x})\Big) = \frac{\exp\Big[w\big(\omega(\boldsymbol{x}) | \mathbf{M} = 1\big) P(\mathbf{M} = 1)\Big]}{\sum_{m \in \mathbb{M}} \exp\Big[w\big(\omega(\boldsymbol{x}) | \mathbf{M} = m\big) P(\mathbf{M} = m)\Big]} - 0.5. \qquad (17)$$

To turn the probability into the potential function, the following modifications have been applied.

- When an outlying object x is considered, the sum $\sum_{i \in \mathbb{M}} w\Big(\omega(\boldsymbol{x}) | \mathbf{M} = m\Big) P(\mathbf{M} = m)$ may be close

to zero. To avoid numerical problems, the softmax transformation is applied. The application of the softmax normalization also gives the potential function a desired shape. That is, for outliers lying far from the decision boundary, the potential function is close to zero [because the fraction in the (17) is close to 0.5].
- To get the potential bounded within an interval $[-0.5; 0.5]$ (the sign of the potential indicates the class), we have to subtract 0.5 from the probability.

The above-defined potential function takes into account prior class probabilities $P(\mathbf{M} = m)$. Using these probabilities may not be suitable for imbalanced classification problems. If we want the potential function using only class-conditional densities, we simply assume $P(\mathbf{M} = m) = 0.5\ \forall m$. It gives the following potential function:

$$\beta_b\Big(\omega(\boldsymbol{x})\Big) = \frac{\exp\Big[w\big(\omega(\boldsymbol{x})|\mathbf{M}=1\big)\Big]}{\sum_{i\in\mathbb{M}}\exp\Big[w\big(\omega(\boldsymbol{x})|\mathbf{M}=m\big)\Big]} - 0.5. \quad (18)$$

Given the above-defined potential function and the ensemble of linear classifiers, the outcome of the ensemble is calculated using simple averaging of the values of the potential function:

$$\omega(\boldsymbol{x}) = \sum_{i=1}^{N}\beta^{(i)}\Big(\omega(\boldsymbol{x})\Big), \quad (19)$$

where $\beta^{(i)}\Big(\omega(\boldsymbol{x})\Big)$ is the classifier-specific potential.

B. PROBABILITY ESTIMATION

In the previous section the potential function has been developed. It was assumed that all needed probabilities are known. Unfortunately, in the real-world classification tasks, the probabilities have to be estimated from the training data. This section describes the techniques used to estimate the probabilities needed by the potential function.

First, the linear classifier ψ is trained using the training set $\mathcal{T}$. Then, the training set is divided into two datasets containing objects belonging to class -1 and 1 respectively:

$$\mathcal{T}^{(m)} = \left\{(\boldsymbol{x}^{(k)}, m^{(k)})|m^{(k)} = m\right\}, \quad (20)$$

Given those sets, the prior probabilities may be easily estimated as:

$$\hat{P}(\mathbf{M} = m) = \frac{|\mathcal{T}^{(m)}|}{|\mathcal{T}|}. \quad (21)$$

Estimating the class-conditional densities $w\Big(\omega(\boldsymbol{x})|\mathbf{M} = m\Big)$ is a more complicated task. This is due to the entire probability density function have to be estimated instead of a single value. To avoid any assumptions about the distribution type, the non-parametric estimation technique must be used [43], [44]. Two widely-known examples of such techniques are histograms and kernel estimators [45]. The histogram-based estimators are often criticized because the shape of estimated distribution strongly depends on the width and the number of bins. What is more, due to the binning approach the histogram loses some information coming from the sample [44]. However, the kernel-based estimators are devoid of these drawbacks [43], [44].

Consequently, the class-conditioned densities are estimated using the kernel estimator [46]:

$$\hat{w}\Big(\omega(\boldsymbol{x})|\mathbf{M}=m\Big) = \frac{1}{h\left|\mathcal{T}^{(m)}\right|}\sum_{i=1}^{|\mathcal{T}^{(m)}|}K\left(\frac{\omega\left(\boldsymbol{x}\right)-\omega\left(x^{(i)}\right)}{h}\right), \quad (22)$$

where h is the smoothing parameter (bandwidth) and $K(\cdot)$ is the kernel function. The kernel estimation technique uses a 'smooth' kernel function that is centered at each data point and then the values coming from the centered kernels are summed up to form the estimated probability density function. The estimation result of a smiple distribution is visualised in Fig. 3.

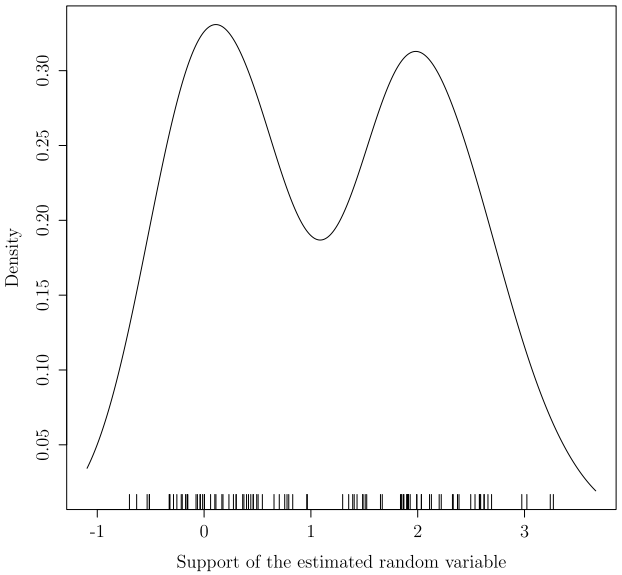

FIGURE 3. Example of the kernel estimation. The vertical lines over the abscissa represent the realisations of the random variable being estimated. The realizations are generated using normal distributions with standard deviation equal to 0.5 and mean values equal to zero and two respectively. The number of realizations is 50 for each of the normal distributions. The estimation was made using Gaussian kernel and bandwidth selected using the Silverman's rule [44].

The kernel function usually meets the following properties [44], [47]:

$$K(t) \in [0; \infty)\,\forall t \in \mathbb{R}, \quad (23)$$

$$K(t) = K(-t)\,\forall t \in \mathbb{R}, \quad (24)$$

$$\int_{-\infty}^{\infty} K(t)\,dt = 1, \quad (25)$$

Many different types of kernel functions have been described so far in the literature [44]. However, for the symmetric kernels described above the shape of the kernel function has little impact on estimator properties [48], [49].

On the other hand, the Gaussian kernels are told to produce most 'smooth' estimators and, due to this fact, they are most frequently used [44]. The impact of the kernel shape on the resulting estimation is shown in Fig. 4.

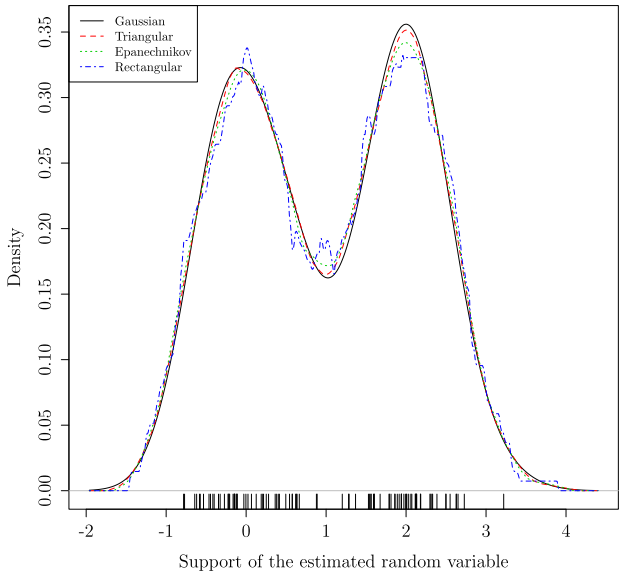

FIGURE 4. Example of the kernel estimation. The vertical lines over the abscissa represent the realisations of the random variable being estimated. The realizations are generated using normal distributions with standard deviation equal to 0.5 and mean values equal to zero and two respectively. The number of realizations is 50 for each of the normal distributions. The estimation was made using different kernels and bandwidth selected using the Silverman's rule [44]. As we can see, the shapes of the estimated distributions are quite similar for all of the investigated kernels.

In this study, we decided to use the Gaussian (The kernel is the pdf. of the normal distribution) kernel:

$$K_g(t) = \frac{1}{\sqrt{2\pi}} exp(-0.5t^2). \tag{26}$$

In contrast to the kernel shape choice, the choice of the kernel bandwidth is critical [44]. On one hand, too large value of the smoothing parameter causes the estimator to be over smoothed. The over smoothed estimator may loose important details of the density function. For example the over smoothed estimator may loose information about multiple modes when the peaks are to close to each other [43], [44]. The impact of the bandwidth on the estimation result is shown in Fig. 5.

On the other hand, too small bandwidth value may cause the estimator to show many insignificant details of the density function. For example, it may show multiple modes when estimating unimodal distribution [44]. The literature presents many techniques for finding the proper value of the kernel bandwidth [47], [50], [51]. However, for Gaussian kernels, the Silverman rule of thumb is often used in practice due to ist simplicity and the ability to provide quite good results [44].

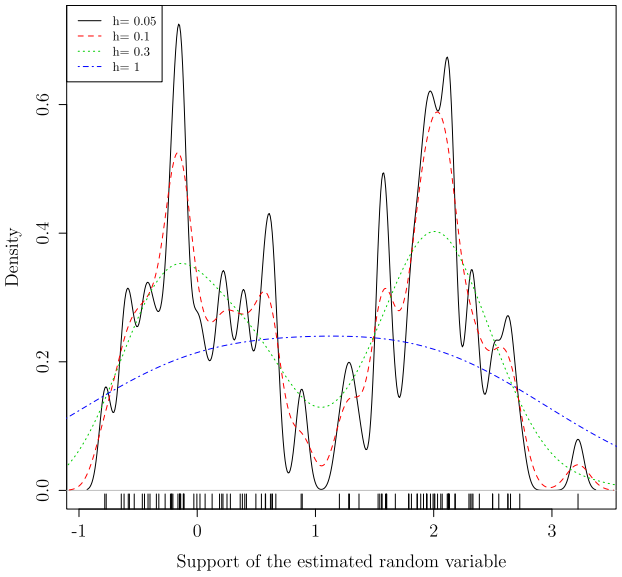

FIGURE 5. Example of the kernel estimation. The vertical lines over the abscissa represent the realisations of the random variable being estimated. The realizations are generated using normal distributions with standard deviation equal to 0.5 and mean values equal to zero and two respectively. The number of realizations is 50 for each of the normal distributions. The estimation was made gaussian kernel. The estimation was done using different values of bandwidth parameter.

In our work, we also decided to select the bandwidth using Silverman's rule of thumb [48]:

$$h = 0.9 \min(\hat{\sigma}, \frac{\text{IQR}}{1.34}) n^{-\frac{1}{5}}, \tag{27}$$

where $\hat{\sigma}$ is the sample standard deviation, IQR is the interquantile ratio of the sample and n is the sample size [52].

C. TOY EXAMPLES

In this section, simple examples of the potential-function-construction process are presented.

1) PROBABILITY ESTIMATION AND THE POTENTIAL FUNCTION

In this subsection, a simple example of constructing the potential field out of balanced dataset is given

The training dataset and the constructed linear classifier (constructed using the nearest centroid rule [53]) for this set are presented in Fig. 6. The estimated class-conditioned densities are shown in Fig. 7. The potential function is visualised in Fig. 8.

2) ENSEMBLE-SPECIFIC DISCRIMINANT FUNCTIONS

In this subsection, we visualise the ensemble-specific discriminant functions obtained using different approaches to combining linear classifiers. In the examples, we used an imbalanced two-dimensional dataset. As a base classifier the Fisher LDA classifier is used. [54] The base classifiers of the ensemble are trained using the bagging approach and the number of the base classifiers is three.

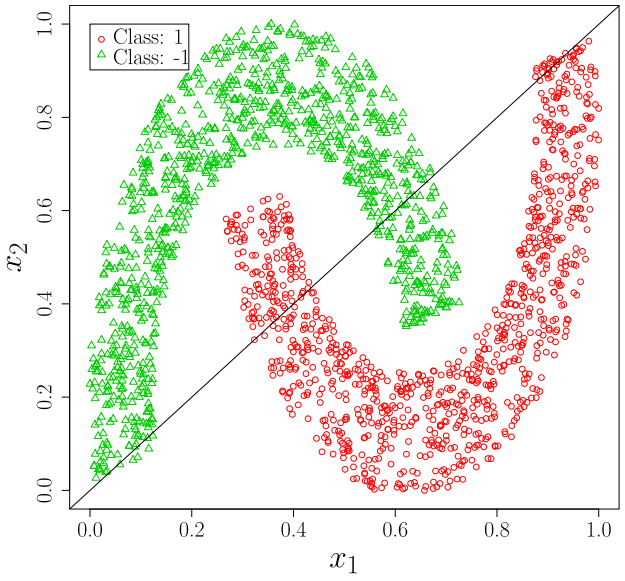

FIGURE 6. Linear decision boundary for binary, two-dimensional, banana-shaped data. The solid line presents the decision boundary obtained using the nearest centroid algorithm.

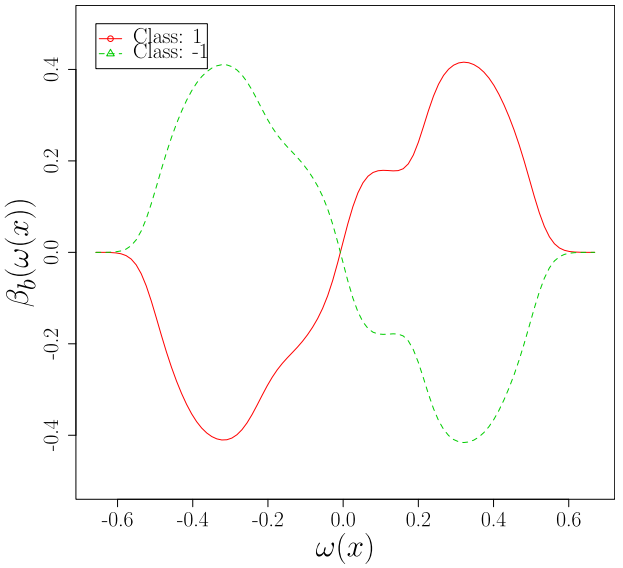

FIGURE 8. The potential function β_b calculated using the given dataset. The solid line presents the potential function defined in (18). The dashed one is the alternative potential related to the other class.

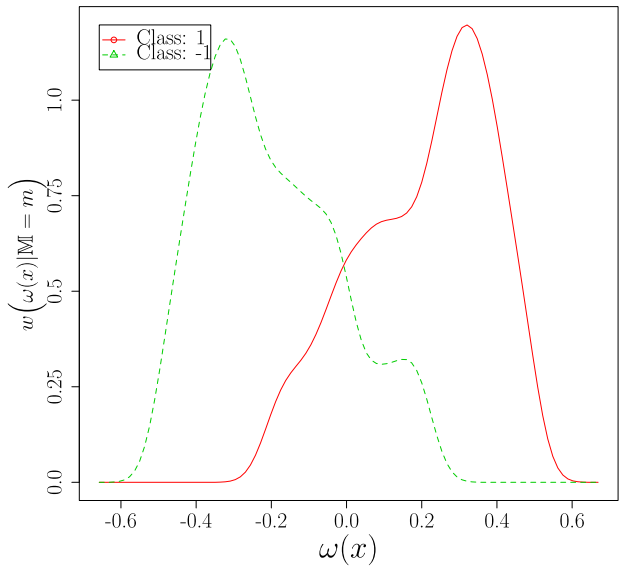

FIGURE 7. Class conditioned pdfs. calculated using the kernel estimator.

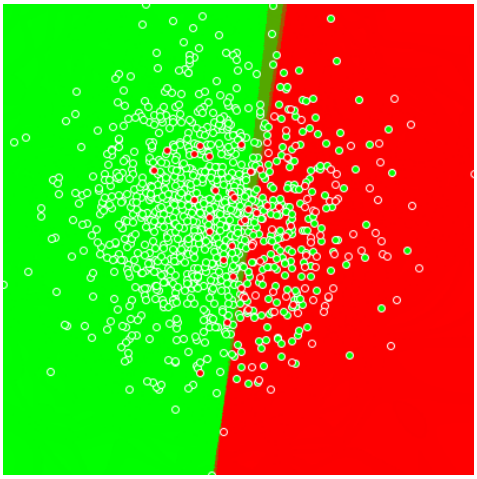

FIGURE 9. Discriminant function for the ensemble classifier using majority voting approach.

The visualisations of discriminant functions are shown in Fig. 9 – 12. In the figures, objects are shown using red and green dots with a white border. The values of the discriminant functions are shown using the background colour. Shades between red and green indicate the negative and positive values of the discriminant functions.

As we can see in Fig. 9, for the response of the majority voting ensemble is rather crisp. The shades between red and green are visible only in the area where the decision areas of base classifier do not overlap.

For the ensemble using the model averaging approach, which is shown in Fig. 10, no clear decision boundary is visible. The values of the discriminant function decrease along the horizontal axis of the plot. Consequently, the objects that are more distant from the decision boundary, the higher the absolute value of the discriminant function is.

The discriminant function generated by the ensemble using β potential function is visualised in Fig. 11. It may be seen that the green colour dominates the plot. It means that the ensemble is biased towards the majority class. However, we may see that absolute value of the discriminant function for the majority class is the highest in the area where the value of class-specific probability-density-function is the highest.

The discriminant function generated by the ensemble using β_b potential function is visualised in Fig. 12. In this case, the ensemble is not biased towards the majority class. As expected, the class-specific valiues of the discriminant

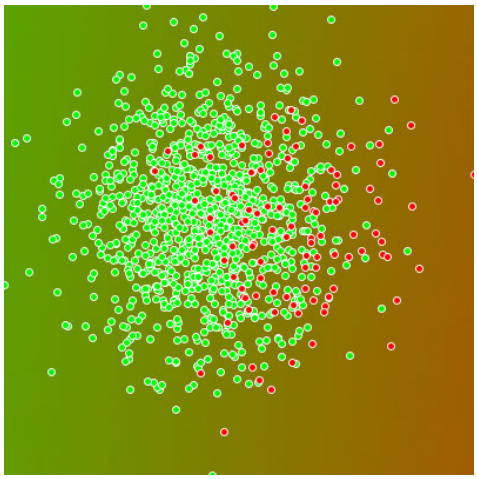

FIGURE 10. Discriminant function for the ensemble classifier using model averaging approach.

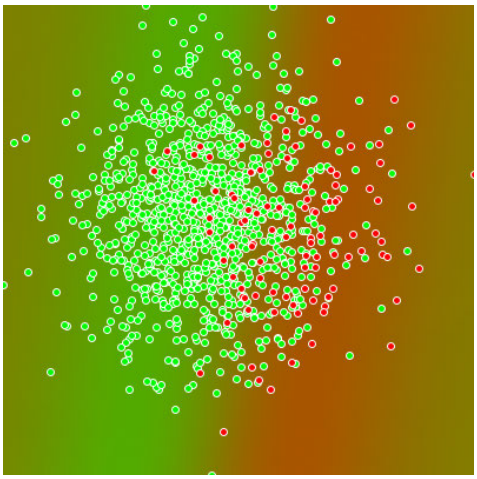

FIGURE 12. Discriminant function for the ensemble using β_b potential function.

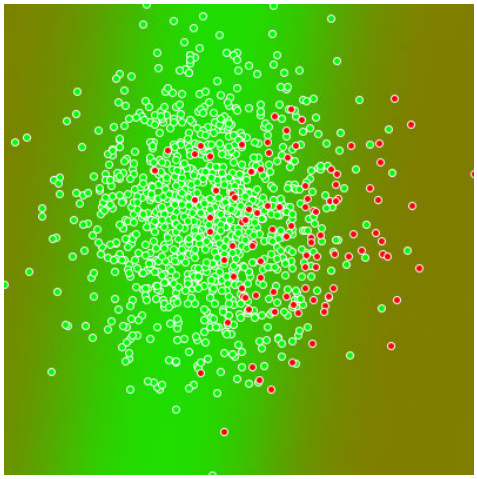

FIGURE 11. Discriminant function for the ensemble using β potential function.

function are the highest in the areas where many class-specific objects are placed.

IV. EXPERIMENTAL SETUP

In the experimental study, the proposed method was used to combine classifiers using a homogeneous ensemble of classifiers. The ensembles were created using a bagging approach [40]. The generated ensembles consist of 11 classifiers learned by using the bagging method containing 80% of the number of instances from the original dataset. The instances for the given base learner are chosen randomly.

During the experiment, the following ensembles were considered:

- ψ_{RA} – Random forest ensemble [55];
- ψ_{RO} – Rotation forest ensemble [56];
- ψ_{MV} – bagged classifiers were combined using the majority-voting approach (7);
- ψ_{MA} – bagged classifiers were combined using the model-averaging approach (8);
- ψ_{SM} – outputs of the linear classifiers were normalized using the softmax rule, and then combined according to (10);
- ψ_{PF} – the approach using the potential function proposed in [17]. Classifiers were combined using model averaging;
- ψ_{B1} – the ensemble combined using β function;
- ψ_{B2} – the ensemble combined using β_b function.

The following base classifiers were used to build the above-mentioned ensembles (except random forest and rotation forest):

- ψ_{FLDA} – Fisher LDA [54];
- ψ_{LR} – Logistic regression classifier [57];
- ψ_{PER} – perceptron classifier [58];
- ψ_{NC} – nearest centroid (Nearest Prototype) [53] with the Euclidean distance;
- ψ_{SVM} – SVM classifier with the linear kernel (no kernel) [59].

The classifiers used were implemented in the WEKA framework [60]. If not stated otherwise, the classifier parameters were set to their defaults. The multi-class problems were dealt with using One-vs-One decomposition [14]. The source code of the proposed methods is available online .[1]

For each of the employed kernel estimators the kernel bandwidth was selected using the Silverman's rule [48]. The gaussian kernel is used.

To evaluate the proposed methods, six classification-quality criteria are used:

- Macro-averaged:
  - false discovery rate (1 − precision, FDR);
  - false negative rate (1 − recall, FNR);
  - Matthews correlation coefficient (MCC);

[1] https://github.com/ptrajdos/piecewiseLinearClassifiers/tree/master

- Micro-averaged:
  - false discovery rate (1 − precision, FDR);
  - false negative rate (1 − recall, FNR);
  - Matthews correlation coefficient (MCC).

Macro and micro averaged measures were used to assess the performance for the majority and minority classes. This is because the macro-averaged measures are more sensitive to the performance for minority classes [61]. The criteria are bounded in the interval $[0, 1]$, where zero denotes the best classification quality. The results obtained using the MCC criterion are also normalised to fit the above-mentioned property.

The experimental procedure was conducted using the ten-fold cross-validation procedure. The data folds were generated using methods implemented in WEKA software. The random seed used to generate them is zero.

Following the recommendation of [62] the statistical significance of the obtained results was assessed using the two-step procedure. The first step was to perform the Friedman test [62] for each quality criterion separately. Since the multiple criteria were employed, the family-wise errors (FWER) should be controlled [63]. To do so, the Bergmann-Hommel [63] procedure of controlling FWER of the conducted Friedman tests was employed. When the Friedman test shows that there is a significant difference within the group of classifiers, the pairwise tests using the Wilcoxon signed-rank test [62] were employed. To control FWER of the Wilcoxon-testing procedure, the Bergmann-Hommel approach was employed [63]. For all tests, the significance level was set to $\alpha = 0.01$.

Table 1 displays the collection of the 70 benchmark sets that were used during the experimental evaluation of the proposed methods. The table is divided into three columns. Each column is organized as follows. The first column contains the names of the datasets. The remaining ones contain the set-specific characteristics of the benchmark sets: the number of instances in the dataset $|S|$; dimensionality of the input space d; the number of classes C; average imbalance ratio IR.

The datasets come from the Keel [2] repository. The datasets are available online .[3]

During the dataset-preprocessing stage, a few transformations on the datasets were applied. The PCA method [64] was applied and the percentage of covered variance was set to 0.95. The attributes were also normalized to have zero mean and unit variance.

V. RESULTS AND DISCUSSION

To compare multiple algorithms on multiple benchmark sets, the average ranks approach is used. In this approach, the winning algorithm achieves rank equal to '1', the second achieves rank equal to '2', and so on. In the case of ties, the ranks of algorithms that achieve the same results are averaged. To provide a visualization of the average ranks the radar plots are employed. In the radar plot, each of the radially arranged axes represents one quality criterion. In the plots, the data is visualized in such a way that the lowest ranks are closer to the centre of the graph. Consequently, higher ranks are placed near the outer ring of the graph. Graphs are also scaled so that the inner ring represents the lowest rank recorded for the analyzed set of classifiers, and the outer ring is equal to the highest recorded rank. The radar plots are presented on Fig. 13 – 17.

[2]https://sci2s.ugr.es/keel/category.php?cat=clas

[3]https://github.com/ptrajdos/MLResults/blob/master/data/KeelData.tar.xz

TABLE 1. The characteristics of the benchmark sets.

Name	$\|S\|$	d	C	IR	Name	$\|S\|$	d	C	IR
abalone	4174	10	28	162.59	mammographic	830	5	2	1.03
adult	45222	103	2	2.02	marketing	6876	13	9	1.80
appendicitis	106	7	2	2.52	monk-2	432	6	2	1.06
australian	690	18	2	1.12	movement_libras	360	90	15	1.00
automobile	159	61	6	4.30	mushroom	5644	92	2	1.31
balance	625	4	3	2.63	newthyroid	215	5	3	3.43
banana	5300	2	2	1.12	nursery	12960	26	5	435.25
bands	365	19	2	1.35	optdigits	5620	64	10	1.02
breast	277	38	2	1.71	page-blocks	5472	10	5	58.12
bupa	345	6	2	1.19	penbased	10992	16	10	1.04
car	1728	21	4	10.08	phoneme	5404	5	2	1.70
chess	3196	38	2	1.05	pima	768	8	2	1.43
cleveland	297	13	5	5.08	post-operative	87	21	3	21.86
coil2000	9822	85	2	8.38	ring	7400	20	2	1.01
connect-4	67557	126	3	3.52	saheart	462	9	2	1.44
contraceptive	1473	9	3	1.37	satimage	6435	36	6	1.66
crx	653	42	2	1.10	segment	2310	19	7	1.00
dermatology	358	34	6	2.43	shuttle	57999	9	7	1326.03
ecoli	336	7	8	23.56	sonar	208	60	2	1.07
fars	100968	362	8	610.12	spambase	4597	57	2	1.27
flare	1066	37	6	2.90	spectfheart	267	44	2	2.43
german	1000	59	2	1.67	splice	3190	287	3	1.77
glass	214	9	6	3.91	tae	151	5	3	1.03
haberman	306	3	2	1.89	texture	5500	40	11	1.00
hayes-roth	160	4	3	1.37	thyroid	7200	21	3	19.76
heart	270	13	2	1.13	tic-tac-toe	958	27	2	1.44
hepatitis	80	19	2	3.08	titanic	2201	3	2	1.55
housevotes	232	16	2	1.07	twonorm	7400	20	2	1.00
ionosphere	351	33	2	1.39	vehicle	846	18	4	1.03
iris	150	4	3	1.00	vowel	990	13	11	1.00
kr-vs-k	28056	40	18	20.96	wdbc	569	30	2	1.34
led7digit	500	7	10	1.16	wine	178	13	3	1.23
letter	20000	16	26	1.06	wisconsin	683	9	2	1.43
lymphography	148	38	4	15.77	yeast	1484	8	10	17.08
magic	19020	10	2	1.42	zoo	101	21	7	4.84

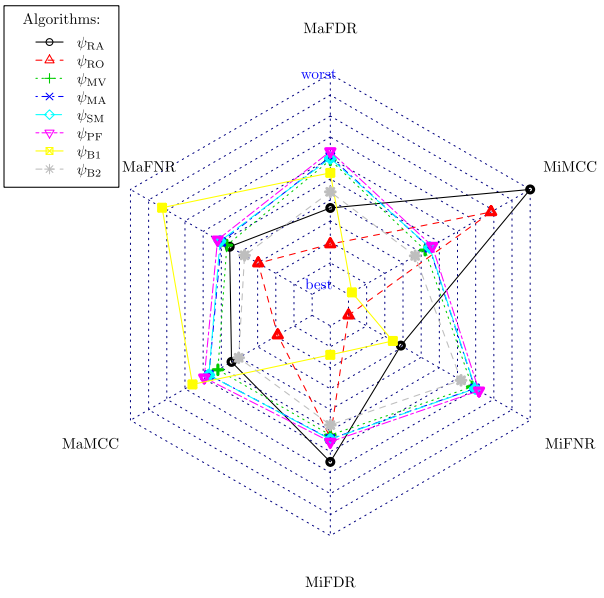

FIGURE 13. Radar plot for the ψ_{FLDA}.

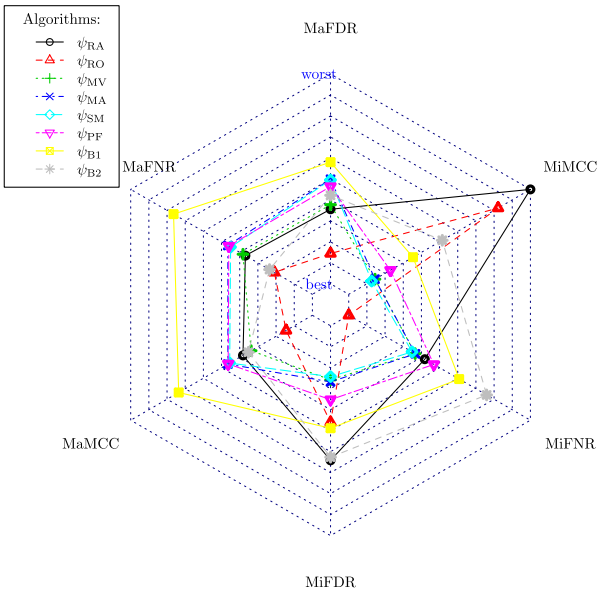

FIGURE 14. Radar plot for the ψ_{LR}.

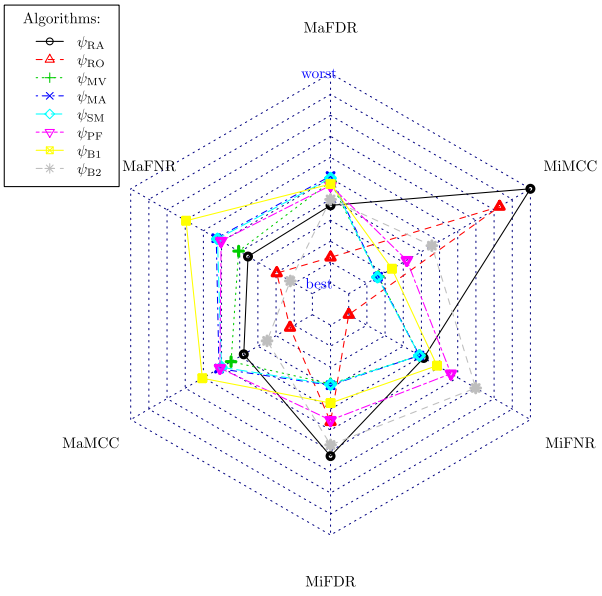

FIGURE 15. Radar plot for the ψ_{PER}.

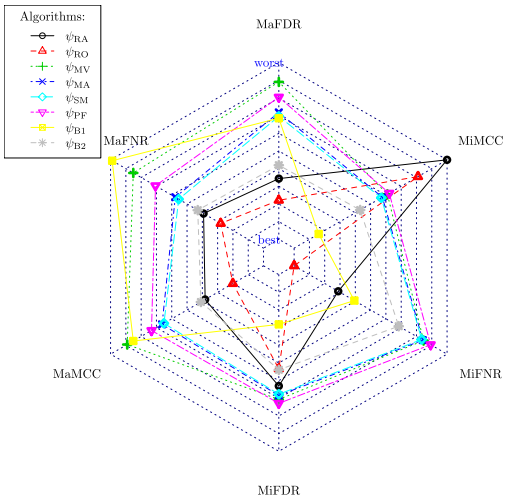

FIGURE 16. Radar plot for the ψ_{NC}.

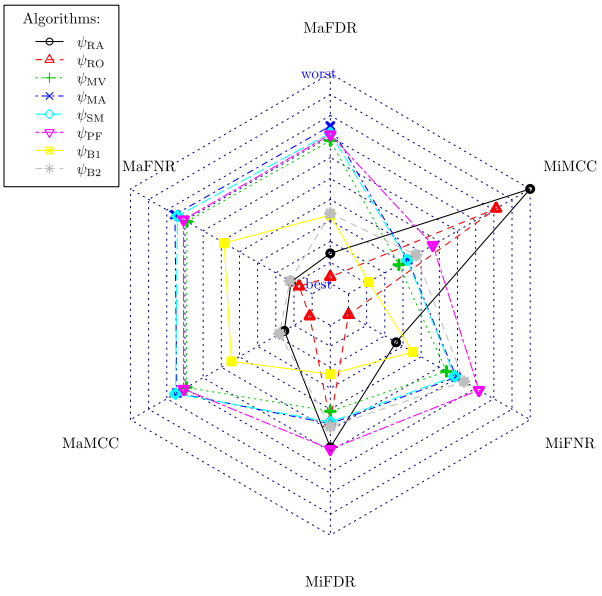

FIGURE 17. Radar plot for the ψ_{SVM}.

The numerical results are given in Table 2 to 6. Each table is structured as follows. The first row contains the names of the investigated algorithms. Then, the table is divided into six sections – one section is related to a single evaluation criterion. The first row of each section is the name of the quality criterion investigated in the section. The second row shows the p-value of the Friedman test. The third one shows the average ranks achieved by algorithms. The following rows show p-values resulting from the pairwise Wilcoxon test. The p-value equal to 0.000 informs that the p-values are lower than 10^{-3} and p-value equal to 1.000 informs that the value is higher than 0.999. P-values lower than α are bolded. Consequently, the bolded results show that there is a significant difference between classifiers.

Before we begin the analysis of the classification quality of the proposed methods, let us analyse the reference algorithms first. We start with the analysis of the results connected with ensembles built using linear classifiers only. For the above-presented experimental setup there are almost no significant differences between the reference methods. For ψ_{NC} classifier on the other hand, for macro-averaged FDR measure ψ_{MV} classifier performs significantly worse than the other reference ensemble algorithms based on the linear classifiers. It means that for ψ_{NC}, the majority voting strategy fails to identify the minority class. This may be due to the fact that ψ_{NC} completely ignores inter and intra-class variation.

Now, the results related to tree-based ensembles (ψ_{RA} and ψ_{RO}) are analysed. For macro-averaged FDR and MCC, the ψ_{RO} algorithm is significantly better than the ensembles built using linear classifiers for four of five base classifiers.

TABLE 2. Statistical evaluation: the Wilcoxon test for the ensembles based on ψ_{FLDA} classifier.

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFDR							
Frd.	**4.436e-05**							
Rank	3.950	3.229	4.907	4.971	4.943	5.079	4.650	4.271
ψ_{RA}		0.656	0.024	0.028	0.027	0.021	0.016	0.144
ψ_{RO}			**0.005**	**0.007**	**0.006**	**0.003**	**0.001**	0.016
ψ_{MV}				1.000	1.000	1.000	1.000	0.060
ψ_{MA}					1.000	1.000	1.000	0.060
ψ_{SM}						1.000	1.000	0.060
ψ_{PF}							1.000	0.045
ψ_{B1}								1.000
Nam.	MiFDR							
Frd.	**4.436e-05**							
Rank	5.150	4.664	4.614	4.693	4.700	4.750	3.014	4.414
ψ_{RA}		1.000	0.804	0.804	0.804	0.804	0.041	0.804
ψ_{RO}			1.000	1.000	1.000	1.000	0.904	1.000
ψ_{MV}				1.000	1.000	1.000	**0.000**	1.000
ψ_{MA}					1.000	1.000	**0.000**	1.000
ψ_{SM}						1.000	**0.000**	1.000
ψ_{PF}							**0.000**	1.000
ψ_{B1}								**0.000**

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFNR							
Frd.	**4.436e-05**							
Rank	4.336	3.679	4.400	4.529	4.557	4.614	5.893	3.993
ψ_{RA}		0.985	1.000	1.000	1.000	1.000	**0.000**	1.000
ψ_{RO}			0.485	0.485	0.485	0.368	**0.000**	0.985
ψ_{MV}				1.000	1.000	1.000	**0.005**	0.485
ψ_{MA}					1.000	1.000	**0.000**	0.485
ψ_{SM}						1.000	**0.001**	0.485
ψ_{PF}							**0.006**	0.368
ψ_{B1}								**0.000**
Nam.	MiFNR							
Frd.	**1.631e-20**							
Rank	3.643	2.436	5.286	5.350	5.357	5.443	3.457	5.029
ψ_{RA}		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.699	**0.000**
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{MV}				1.000	1.000	1.000	**0.000**	1.000
ψ_{MA}					1.000	1.000	**0.000**	1.000
ψ_{SM}						1.000	**0.000**	1.000
ψ_{PF}							**0.000**	1.000
ψ_{B1}								**0.000**

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	Macro Matthews							
Frd.	**4.507e-05**							
Rank	4.293	3.229	4.607	4.814	4.814	4.921	5.193	4.129
ψ_{RA}		0.301	0.483	0.301	0.266	0.301	**0.003**	0.753
ψ_{RO}			**0.009**	**0.009**	**0.009**	**0.009**	**0.000**	0.026
ψ_{MV}				0.771	0.771	0.771	0.483	0.483
ψ_{MA}					1.000	1.000	0.667	0.144
ψ_{SM}						1.000	0.667	0.144
ψ_{PF}							0.771	0.080
ψ_{B1}								**0.006**
Nam.	Micro.Matthews							
Frd.	**5.492e-23**							
Rank	6.621	5.721	4.200	4.300	4.307	4.364	2.514	3.971
ψ_{RA}		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{RO}			**0.001**	**0.001**	**0.001**	**0.001**	**0.000**	**0.001**
ψ_{MV}				1.000	1.000	1.000	**0.000**	1.000
ψ_{MA}					1.000	1.000	**0.000**	1.000
ψ_{SM}						1.000	**0.000**	1.000
ψ_{PF}							**0.000**	1.000
ψ_{B1}								**0.000**

TABLE 3. Statistical evaluation: the Wilcoxon test for the ensembles based on ψ_{LR} classifier.

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFDR							
Frd.	**3.446e-04**							
Rank	4.236	3.329	4.329	4.850	4.836	4.700	5.200	4.521
ψ_{RA}		0.954	0.349	0.153	0.153	0.297	0.021	0.459
ψ_{RO}			0.086	0.039	0.039	0.047	**0.001**	0.086
ψ_{MV}				1.000	1.000	1.000	1.000	1.000
ψ_{MA}					1.000	1.000	1.000	1.000
ψ_{SM}						1.000	1.000	1.000
ψ_{PF}							1.000	1.000
ψ_{B1}								1.000
Nam.	MiFDR							
Frd.	**1.620e-06**							
Rank	5.464	4.686	3.814	3.857	3.750	4.221	4.807	5.400
ψ_{RA}		1.000	0.016	0.016	0.016	0.016	0.068	0.572
ψ_{RO}			0.572	0.572	0.572	0.572	0.847	1.000
ψ_{MV}				1.000	1.000	1.000	0.157	**0.000**
ψ_{MA}					1.000	0.991	0.328	**0.000**
ψ_{SM}						0.991	0.328	**0.000**
ψ_{PF}							0.572	**0.000**
ψ_{B1}								0.022

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFNR							
Frd.	**3.790e-07**							
Rank	4.293	3.607	4.350	4.707	4.643	4.686	5.993	3.721
ψ_{RA}		0.958	0.142	0.142	0.142	0.142	**0.000**	1.000
ψ_{RO}			0.056	0.047	0.047	0.047	**0.000**	0.958
ψ_{MV}				0.958	0.958	0.958	**0.001**	**0.008**
ψ_{MA}					1.000	1.000	**0.005**	**0.008**
ψ_{SM}						1.000	**0.006**	**0.008**
ψ_{PF}							**0.003**	**0.008**
ψ_{B1}								**0.000**
Nam.	MiFNR							
Frd.	**3.908e-13**							
Rank	4.507	2.707	4.279	4.286	4.207	4.721	5.321	5.971
ψ_{RA}		**0.000**	1.000	1.000	1.000	1.000	0.298	**0.000**
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{MV}				1.000	1.000	1.000	0.148	**0.000**
ψ_{MA}					1.000	1.000	0.298	**0.000**
ψ_{SM}						1.000	0.298	**0.000**
ψ_{PF}							0.598	**0.000**
ψ_{B1}								0.018

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	Macro Matthews							
Frd.	**9.634e-07**							
Rank	4.350	3.329	4.157	4.707	4.664	4.700	5.871	4.221
ψ_{RA}		0.312	0.467	0.312	0.312	0.312	**0.003**	0.930
ψ_{RO}			0.063	0.023	0.023	0.026	**0.000**	0.059
ψ_{MV}				0.467	0.467	0.467	**0.005**	1.000
ψ_{MA}					1.000	1.000	0.033	0.467
ψ_{SM}						1.000	0.035	0.467
ψ_{PF}							0.029	0.467
ψ_{B1}								**0.001**
Nam.	Micro.Matthews							
Frd.	**9.850e-34**							
Rank	7.007	6.243	3.329	3.329	3.250	3.693	4.229	4.921
ψ_{RA}		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.001**
ψ_{MV}				0.881	0.881	0.881	0.112	**0.000**
ψ_{MA}					0.881	0.787	0.205	**0.000**
ψ_{SM}						0.787	0.205	**0.000**
ψ_{PF}							0.355	**0.000**
ψ_{B1}								0.015

TABLE 4. Statistical evaluation: the Wilcoxon test for the ensembles based on ψ_{PER} classifier.

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFDR							
Frd.	**7.159e-04**							
Rank	4.307	3.243	4.793	4.907	4.864	4.714	4.743	4.429
ψ_{RA}		0.954	0.122	0.078	0.078	0.078	0.078	0.629
ψ_{RO}			**0.010**	**0.010**	**0.010**	**0.010**	**0.005**	0.086
ψ_{MV}				1.000	1.000	1.000	1.000	1.000
ψ_{MA}					1.000	1.000	1.000	1.000
ψ_{SM}						1.000	1.000	1.000
ψ_{PF}							1.000	1.000
ψ_{B1}								1.000
Nam.	MiFDR							
Frd.	**4.801e-04**							
Rank	5.393	4.693	3.914	3.943	3.929	4.657	4.307	5.164
ψ_{RA}		1.000	0.021	0.021	0.021	0.092	0.021	0.243
ψ_{RO}			0.573	0.573	0.573	0.736	0.573	1.000
ψ_{MV}				1.000	1.000	0.064	1.000	**0.001**
ψ_{MA}					1.000	0.131	1.000	**0.008**
ψ_{SM}						0.131	1.000	**0.008**
ψ_{PF}							0.573	0.187
ψ_{B1}								0.014

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFNR							
Frd.	**6.890e-09**							
Rank	4.236	3.557	4.457	4.986	4.957	4.871	5.700	3.236
ψ_{RA}		0.639	0.065	0.026	0.026	0.016	**0.001**	1.000
ψ_{RO}			0.026	0.012	0.012	0.012	**0.000**	0.639
ψ_{MV}				0.159	0.218	0.639	**0.007**	**0.000**
ψ_{MA}					1.000	1.000	0.218	**0.000**
ψ_{SM}						1.000	0.218	**0.000**
ψ_{PF}							0.218	**0.000**
ψ_{B1}								**0.000**
Nam.	MiFNR							
Frd.	**2.447e-10**							
Rank	4.486	2.714	4.386	4.386	4.386	5.129	4.800	5.714
ψ_{RA}		**0.000**	1.000	1.000	1.000	0.222	1.000	**0.004**
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{MV}				1.000	1.000	0.060	1.000	**0.001**
ψ_{MA}					1.000	0.119	1.000	**0.004**
ψ_{SM}						0.119	1.000	**0.004**
ψ_{PF}							0.697	0.187
ψ_{B1}								**0.010**

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	Macro Matthews							
Frd.	**6.062e-06**							
Rank	4.336	3.243	4.636	4.921	4.864	4.900	5.314	3.786
ψ_{RA}		0.192	0.135	0.107	0.107	0.080	0.019	1.000
ψ_{RO}			**0.007**	**0.003**	**0.003**	**0.005**	**0.000**	0.107
ψ_{MV}				0.807	0.898	1.000	0.107	0.019
ψ_{MA}					1.000	1.000	0.898	**0.003**
ψ_{SM}						1.000	0.898	**0.003**
ψ_{PF}							0.898	**0.005**
ψ_{B1}								**0.000**
Nam.	Micro.Matthews							
Frd.	**1.214e-31**							
Rank	7.007	6.279	3.414	3.407	3.393	4.086	3.736	4.679
ψ_{RA}		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{MV}				1.000	1.000	0.074	1.000	**0.001**
ψ_{MA}					1.000	0.083	1.000	**0.004**
ψ_{SM}						0.110	1.000	**0.004**
ψ_{PF}							0.601	0.110
ψ_{B1}								**0.010**

What is more, for macro-averaged FDR it is better than the above-mentioned ensembles for two (ψ_{NC} and ψ_{SVM}) out of five base classifiers. On the other hand, in terms of all macro-averaged measures, ψ_{RA} classifier is significantly better than the ensembles based on the linear classifiers for two (ψ_{NC} and ψ_{SVM}) out of five base classifiers. These results show that ψ_{RA} and ψ_{RO} tend to be better at recognizing minority classes.

The situation changes for the micro-averaged quality criteria. For those criteria, ψ_{RA} and ψ_{RO} algorithms tend to be significantly better in terms of FNR and worse in terms of MCC. These results indicate that, for majority classes, ψ_{RA} and ψ_{RO} tend to make more false-positive predictions than the ensembles based on the linear base classifiers.

A. ψ_{B1} VS REFERENCE

Observing the tables for ψ_{B1} classifier, we may find three different patterns of behaviour.

First for ψ_{FLDA} (Tab. 2, Fig. 13) and ψ_{NC} (Tab. 5, Fig. 16) classifiers, ψ_{B1} get a significantly worse score for

TABLE 5. Statistical evaluation: the Wilcoxon test for the ensembles based on ψ_{NC} classifier.

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFDR								MaFNR								Macro Matthews							
Frd.	**1.826e-14**								**1.702e-14**								**7.402e-15**							
Rank	3.450	2.943	5.714	5.000	4.907	5.357	4.864	3.764	3.650	3.186	5.550	4.414	4.321	4.950	6.121	3.807	3.607	2.857	5.714	4.771	4.721	5.057	5.550	3.721
ψ_{RA}		0.418	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.001**		0.319	**0.000**	**0.006**	**0.008**	**0.004**	**0.000**	0.021		0.192	**0.000**	**0.000**	**0.001**	**0.000**	**0.000**	**0.010**
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.001**			**0.000**	**0.004**	**0.005**	**0.003**	**0.000**	0.016			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.001**
ψ_{MV}				0.084	0.077	0.116	0.418	**0.000**				**0.003**	**0.003**	0.021	0.829	**0.000**				0.012	0.012	0.034	0.349	**0.000**
ψ_{MA}					0.980	0.492	0.980	**0.000**					0.653	0.034	**0.000**	**0.008**					0.349	0.284	0.284	**0.000**
ψ_{SM}						0.492	0.980	**0.000**						0.021	**0.000**	0.034						0.284	0.284	**0.001**
ψ_{PF}							0.980	**0.000**							**0.005**	**0.008**							0.287	**0.000**
ψ_{B1}								0.205								**0.000**								**0.000**
Nam.	MiFDR								MiFNR								Micro.Matthews							
Frd.	**6.416e-05**								**9.151e-30**								**2.173e-15**							
Rank	4.636	4.236	4.950	4.857	4.829	5.043	3.193	4.257	3.221	2.029	5.529	5.507	5.479	5.721	3.657	4.857	6.164	5.379	4.550	4.414	4.386	4.600	2.693	3.814
ψ_{RA}		1.000	1.000	1.000	1.000	1.000	1.000	1.000		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.003**	**0.000**		**0.000**	**0.004**	**0.002**	**0.002**	**0.004**	**0.000**	**0.002**
ψ_{RO}			1.000	1.000	1.000	1.000	1.000	1.000			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**			0.074	0.061	0.061	0.061	**0.000**	0.054
ψ_{MV}				1.000	1.000	1.000	**0.000**	1.000				1.000	1.000	1.000	**0.000**	1.000				0.937	0.937	0.937	**0.000**	0.937
ψ_{MA}					1.000	1.000	**0.000**	1.000					1.000	1.000	**0.000**	0.669					0.937	0.937	**0.000**	0.605
ψ_{SM}						1.000	**0.000**	1.000						1.000	**0.000**	0.669						0.937	**0.000**	0.605
ψ_{PF}							**0.000**	1.000							**0.000**	0.487							**0.000**	0.605
ψ_{B1}								**0.000**								**0.000**								**0.001**

TABLE 6. Statistical evaluation: the Wilcoxon test for the ensembles based on ψ_{SVM} classifier.

	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}	ψ_{RA}	ψ_{RO}	ψ_{MV}	ψ_{MA}	ψ_{SM}	ψ_{PF}	ψ_{B1}	ψ_{B2}
Nam.	MaFDR								MaFNR								Macro Matthews							
Frd.	**3.088e-23**								**5.323e-27**								**2.347e-26**							
Rank	3.164	2.693	5.421	5.714	5.571	5.543	3.929	3.964	3.050	2.864	5.457	5.736	5.679	5.536	4.600	3.079	3.207	2.621	5.471	5.693	5.721	5.536	4.429	3.321
ψ_{RA}		0.298	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.022		0.426	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.414		0.192	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.115
ψ_{RO}			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.008**			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	0.265			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.004**
ψ_{MV}				0.274	0.696	0.897	**0.000**	**0.000**				0.581	0.664	0.664	0.080	**0.000**				1.000	1.000	1.000	0.011	**0.000**
ψ_{MA}					0.422	0.696	**0.000**	**0.000**					0.414	0.664	**0.010**	**0.000**					1.000	1.000	**0.002**	**0.000**
ψ_{SM}						0.696	**0.000**	**0.000**						0.664	**0.010**	**0.000**						1.000	**0.002**	**0.000**
ψ_{PF}							**0.000**	**0.000**							0.020	**0.000**							**0.004**	**0.000**
ψ_{B1}								0.495								**0.000**								**0.000**
Nam.	MiFDR								MiFNR								Micro.Matthews							
Frd.	**9.509e-03**								**1.234e-15**								**8.084e-25**							
Rank	5.007	4.493	4.286	4.536	4.500	5.043	3.543	4.593	3.664	2.564	4.829	5.057	5.021	5.579	4.050	5.236	6.764	5.979	3.729	3.936	3.921	4.514	3.036	4.121
ψ_{RA}		1.000	0.574	0.574	0.574	1.000	0.117	0.635		**0.000**	**0.003**	**0.002**	**0.002**	**0.000**	0.061	**0.000**		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
ψ_{RO}			1.000	1.000	1.000	1.000	0.968	1.000			**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**			**0.000**	**0.000**	**0.000**	**0.001**	**0.000**	**0.002**
ψ_{MV}				1.000	1.000	0.117	0.247	1.000				1.000	1.000	0.061	0.108	1.000				1.000	1.000	0.060	0.099	1.000
ψ_{MA}					1.000	0.471	0.023	1.000					1.000	0.188	**0.010**	1.000					1.000	0.147	0.012	1.000
ψ_{SM}						0.574	0.028	1.000						0.188	0.012	1.000						0.147	0.014	1.000
ψ_{PF}							**0.000**	0.968							**0.000**	0.534							**0.000**	0.658
ψ_{B1}								0.017								**0.008**								**0.010**

the macro-averaged FNR and a significantly better score for micro-averaged FNR and MCC measures. On the other hand, for the micro-averaged FDR, the B1 ensemble is significantly better than ψ_{RA} and ψ_{RO}. It means that for those classifiers ψ_{B1} classifier makes more false-negative predictions for the minority class than the reference ensembles built using linear classifier. Moreover, it tends to be biased towards the majority class. However not as biased as ψ_{RA} and ψ_{RO}.

The second pattern of behaviour may be observed for ψ_{LR} (Tab. 3, Fig. 14) and ψ_{PER} (Tab. 4, Fig. 15). For those classifiers, ψ_{B1} is also worse than the reference methods in terms of macro-averaged FNR, but for ensembles built using linear base classifiers it is comparable in terms of micro-averaged measures. It means that its bias towards the majority class is smaller for those base classifiers.

For the above-considered cases, the bias towards the majority class is an effect of employing the prior class probabilities in the potential function. Employing those probabilities moves the decision boundary towards the majority class.

The third pattern is observed for ψ_{SVM} base classifier (Tab. 6, Fig. 17). For this base classifier, ψ_{B1} outperforms the linear-base-classifier-based reference ensembles in terms of macro-averaged measures. ψ_{B1} is also comparable to them in terms of micro averaged measures. Contrary to the previously observed patterns, for this base classifier, ψ_{B1} is not biased towards the majority class. What is more, it allows to improve the classification quality for the minority classes. This effect may be related to the separation margin that is enforced by the SVM-based classifiers. The harnessing of the prior class probabilities moves the decision boundary towards the majority class, however, the separation-margin is so wide that it does not cause a bias towards the majority class. The situation changes when ψ_{B1} is compared to ψ_{RA} and ψ_{RO} in terms of the macro-averaged criteria. In this case ψ_{B1} is significantly worse than the tree-based-reference ensembles.

B. ψ_{B2} VS REFERENCE

Comparing ψ_{B2} classifier with the reference methods allows us to observe the following patterns.

For ψ_{FLDA} base classifier (Tab. 2, Fig. 13), there are no significant differences between the ψ_{B2} ensemble and the reference methods.

For most base classifiers, except ψ_{FLDA}, ψ_{B2} the ensemble is significantly better than linear-classifier-based ensembles in terms of macro-averaged FNR measure. Additionally, for ψ_{NC} (Tab. 5, Fig. 16), and ψ_{SVM} (Tab. 6, Fig. 17), the investigated classifier is significantly better in terms of the macro-averaged FDR. What is most important, for ψ_{PER}, ψ_{NC}, and ψ_{SVM}, it is also better in terms of macro-averaged MCC measure. It means that ψ_{B2} classifier tends to

outperform the linear-classifier-based reference ensembles in recognizing objects of minority classes.

The situation changes when comparing ψ_{B2} ensemble with the tree-based ensembles (ψ_{RA} and ψ_{RO}). In this case for ψ_{NC}, and ψ_{SVM} base classifiers, the ψ_{B2} ensemble is significantly worse in terms of macro-averaged FDR and MCC.

For the majority classes the situation is quite different. For ψ_{NC} (Tab. 5, Fig. 16), and ψ_{SVM} (Tab. 6, Fig. 17), when comparing to the linear-classifier-based ensembles, there are no significant differences in terms of the micro-averaged measures. It means that for those base classifiers ψ_{B2} ensemble is able to improve the classification quality for minority classes without harming the recognition quality of the majority classes. For ψ_{LR} and ψ_{PER}, on the other hand, ψ_{B2} tends to be worse in terms of the micro averaged measures. Consequently, for those classifiers, ψ_{B2} ensembles are biased towards minority classes.

The situation is a bit different when comparing ψ_{B2} with the tree-based ensembles. In this comparison, ψ_{B2} tends to be significantly better in terms of the micro-averaged MCC criterion and significantly worse in terms of micro-averaged FNR. It means that the ψ_{B2} ensemble is less biased towards the majority classes.

C. ψ_{B1} VS ψ_{B2}

For all the investigated base classifiers, the result pattern is very clear. That is, in terms of macro-averaged FNR and MCC, ψ_{B2} is always significantly better than ψ_{B1}. It means that for the minority class ψ_{B1} makes significantly more false-negative predictions than ψ_{B2}. In other words, employing prior probabilities makes the ensemble based on ψ_{B1} potential function less sensitive to the minority class.

On the other hand, for micro-averaged FNR and MCC measures, ψ_{B1} is better than ψ_{B2} for three out of five base classifiers (ψ_{FLDA} (Tab. 2, Fig. 13), ψ_{PER} (Tab. 4, Fig. 15), ψ_{NC} (Tab. 5, Fig. 16)). For two remaining base classifiers (ψ_{LR} (Tab. 3, Fig. 14) and ψ_{SVM} (Tab. 6, Fig. 17)), no significant differences are observed. It means that ψ_{B1} is far better at identifying majority class examples. This fact which is connected with the low classification quality for macro-averaged measures means that ψ_{B1} classifier is biased towards the majority class. This is disadvantageous because, in many practical imbalanced classification problems, the minority class is the class of most interest [65].

D. MAIN FINDINGS

Given the above, the main advantages of the proposed method can be summarized as follows:

- Ensemble ψ_{B2} is better than ψ_{B1} in dealing with imbalanced data.
- In general, ψ_{B2} ensemble is comparable to ψ_{RA} and ψ_{RO} in terms of macro-averaged quality measures. It means that the classification quality for examples coming from the minority classes is similar to the classification quality obtained by non-linear ensembles.
- In general, in terms of the macro-averaged measures ψ_{B2} ensemble is better than other ensembles built using linear classifiers i.e. ψ_{MV}, ψ_{MA}, ψ_{SM} and ψ_{PF}

Likewise given the above, the main disadvantages of the proposed method can be summarized as follows:

- Creating the potential function for ψ_{B1} and ψ_{B2} needs more computational burden than creating potential function in ψ_{MV}, ψ_{MA} and ψ_{SM}. This is because the kernel estimators is employed. What is more, the bandwidth parameter of the kernel must be chosen. This generates an additional computational cost.
- Generally, for macro-averaged measures ψ_{B1} ensemble is worse than the reference methods. It means, that ψ_{B1}, due to employing prior class probabilities, is biased towards majority classes.

VI. CONCLUSION

In this article, a new method of combining linear classifiers has been proposed. Outputs of the base classifiers constituting the ensemble are combined via the potential functions. Two potential functions based on class-conditional probabilities have been developed. One of them ignores class prior probabilities.

The proposed methods have been compared to four reference methods. The comparison was done in terms of six different quality criteria. The experiments were conducted using a large set of benchmark datasets (70 benchmark sets).

The experimental evaluation shows that the potential function that ignores prior probabilities outperforms most of the reference methods in terms of macro-averaged quality criteria. It means that the method is significantly better at recognizing minority class objects.

In this study, a simple bagging approach was used to constitute a diverse set of base classifiers. This simple, yet effective method allowed to achieve quite interesting results. Nevertheless, we believe that harnessing an ensemble building scheme tailored to the proposed potential function will allow to improve the classification quality achieved by the proposed method. Future research should be aimed at this issue.

REFERENCES

[1] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining Knowl. Discovery*, vol. 8, no. 4, p. e1249, Feb. 2018.
[2] N. C. Oza and K. Tumer, "Classifier ensembles: Select real-world applications," *Inf. Fusion*, vol. 9, no. 1, pp. 4–20, Jan. 2008.
[3] V. Dutta, M. Choraś, M. Pawlicki, and R. Kozik, "A deep learning ensemble for network anomaly and cyber-attack detection," *Sensors*, vol. 20, no. 16, p. 4583, Aug. 2020.
[4] Y. Zhou and P. Wang, "An ensemble learning approach for XSS attack detection with domain knowledge and threat intelligence," *Comput. Secur.*, vol. 82, pp. 261–269, May 2019.
[5] F. Li, L. Zhang, B. Chen, D. Gao, Y. Cheng, X. Zhang, Y. Yang, K. Gao, and Z. Huang, "An optimal stacking ensemble for remaining useful life estimation of systems under multi-operating conditions," *IEEE Access*, vol. 8, pp. 31854–31868, 2020.
[6] P. Heda, I. Rojek, and R. Burduk, "Dynamic ensemble selection—Application to classification of cutting tools," in *Computer Information Systems and Industrial Management*. Cham, Switzerland: Springer, 2020, pp. 345–354.

[7] P. Žuvela, M. Lovrić, A. Yousefian-Jazi, and J. J. Liu, "Ensemble learning approaches to data imbalance and competing objectives in design of an industrial machine vision system," *Ind. Eng. Chem. Res.*, vol. 59, no. 10, pp. 4636–4645, Mar. 2020.
[8] X. Li and P. Fan, "Study on characteristic spectrum and multiple classifier fusion with different particle size in marine sediments," *IEEE Access*, vol. 8, pp. 157151–157160, 2020.
[9] C. Vasilakos, D. Kavroudakis, and A. Georganta, "Machine learning classification ensemble of multitemporal Sentinel-2 images: The case of a mixed mediterranean ecosystem," *Remote Sens.*, vol. 12, no. 12, p. 2005, Jun. 2020.
[10] M. Topolski and K. Topolska, "Algorithm for constructing a classifier team using a modified PCA (principal component analysis) in the task of diagnosis of acute lymphocytic leukaemia type B-CLL," in *Hybrid Artificial Intelligent Systems* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2019, pp. 614–624, doi: 10.1007/978-3-030-29859-3_52.
[11] E. Gibaja and S. Ventura, "Multi-label learning: A review of the state of the art and ongoing research," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 4, no. 6, pp. 411–444, Nov. 2014.
[12] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *Int. J. Data Warehousing Mining*, vol. 3, no. 3, pp. 1–13, 2007.
[13] E. A. Cherman, J. Metz, and M. C. Monard, "A simple approach to incorporate label dependency in multi-label classification," in *Advances in Soft Computing*. Berlin, Germany: Springer, 2010, pp. 33–43, doi: 10.1007/978-3-642-16773-7_3.
[14] E. Hüllermeier and J. Fürnkranz, "On predictive accuracy and risk minimization in pairwise label ranking," *J. Comput. Syst. Sci.*, vol. 76, no. 1, pp. 49–62, Feb. 2010.
[15] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Mach. Learn.*, vol. 85, no. 3, pp. 333–359, Jun. 2011, doi: 10.1007/s10994-011-5256-5.
[16] M. Mohandes, M. Deriche, and S. O. Aliyu, "Classifiers combination techniques: A comprehensive review," *IEEE Access*, vol. 6, pp. 19626–19639, 2018.
[17] P. Trajdos and R. Burduk, "Combination of linear classifiers using score function—Analysis of possible combination strategies," in *Progress in Computer Recognition Systems* (Advances in Intelligent Systems and Computing). Cham, Switzerland: Springer, May 2019, pp. 348–359.
[18] P. Ksieniewicz and R. Burduk, "Clustering and weighted scoring in geometric space support vector machine ensemble for highly imbalanced data classification," in *Computational Science* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2020, pp. 128–140.
[19] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2012.
[20] I. R. Shafarevich, *Basic Notions of Algebra*. Berlin, Germany: Springer, 2005.
[21] L. I. Kuncheva, *Combining Pattern Classifiers*. Hoboken, NJ, USA: Wiley, Sep. 2014.
[22] M. Woźniak, M. Graña, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," *Inf. Fusion*, vol. 16, pp. 3–17, Mar. 2014.
[23] A. Jurek, Y. Bi, S. Wu, and C. Nugent, "A survey of commonly used ensemble-based classification techniques," *Knowl. Eng. Rev.*, vol. 29, no. 5, pp. 551–581, May 2013.
[24] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. 1st Int. Workshop Multiple Classifier Syst. (MCS)*. London, U.K. Springer-Verlag, 2000, pp. 1–15.
[25] M. A. Tahir, J. Kittler, and A. Bouridane, "Multilabel classification using heterogeneous ensemble of multi-label classifiers," *Pattern Recognit. Lett.*, vol. 33, no. 5, pp. 513–523, Apr. 2012, doi: 10.1016/j.patrec.2011.10.019.
[26] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996.
[27] Y. Freund and R. Shapire, "Experiments with a new boosting algorithm," in *Proc. 13th Int. Conf. Mach. Learn.*, 1996, pp. 148–156.
[28] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844,Aug. 1998.
[29] L. Rokach, *Pattern Classification Using Ensemble Methods*, vol. 75. Singapore: World Scientific, Nov. 2009.
[30] S. Rajagopal, P. P. Kundapur, and K. S. Hareesha, "A stacking ensemble for network intrusion detection using heterogeneous datasets," *Secur. Commun. Netw.*, vol. 2020, pp. 1–9, Jan. 2020, doi: 10.1155/2020/4586875.
[31] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, nos. 1–2, pp. 1–39, Nov. 2009.
[32] L. I. Kuncheva and J. J. Rodríguez, "A weighted voting framework for classifiers ensembles," *Knowl. Inf. Syst.*, vol. 38, no. 2, pp. 259–275, Dec. 2012.
[33] R. Valdovinos and J. Sánchez, "Combining multiple classifiers with dynamic weighted voting," in *Hybrid Artificial Intelligence Systems* (Lecture Notes in Computer Science), vol. 5572, E. Corchado, X. Wu, E. Oja, A. Herrero, and B. Baruque, Eds. Berlin, Germany: Springer, 2009, pp. 510–516, doi: 10.1007/978-3-642-02319-4_61.
[34] J. Kittler and F. M. Alkoot, "Sum versus vote fusion in multiple classifier systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 1, pp. 110–115, Jan. 2003.
[35] B. Krawczyk, L. L. Minkub, J. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Inf. Fusion*, vol. 37, pp. 132–156, Sep. 2017.
[36] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining Knowl. Discovery*, vol. 8, no. 4, p. e1249, Feb. 2018.
[37] C. O. Plumpton, L. I. Kuncheva, N. N. Oosterhof, and S. J. Johnston, "Naive random subspace ensemble with linear classifiers for real-time classification of fMRI data," *Pattern Recognit.*, vol. 45, no. 6, pp. 2101–2108, Jun. 2012.
[38] M. Trincavelli, S. Coradeschi, A. Loutfi, M. Pardo, and G. Sberveglieri, "Classification of odours for mobile robots using an ensemble of linear classifiers," *AIP Conf.*, vol. 1137, no. 1, pp. 475–478, 2009.
[39] A. Bertoni, R. Folgieri, and G. Valentini, "Bio-molecular cancer prediction with random subspace ensembles of support vector machines," *Neurocomputing*, vol. 63, pp. 535–539, Jan. 2005.
[40] M. Skurichina and R. P. W. Duin, "Bagging for linear classifiers," *Pattern Recognit.*, vol. 31, no. 7, pp. 909–930, Jul. 1998.
[41] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Adv. Large Margin Classifiers*, vol. 10, no. 3, pp. 61–74, 1999.
[42] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2002, pp. 694–699.
[43] P. Kulczycki, "Kernel estimators in industrial applications," in *Soft Computing Applications in Industry*, B. Prasad, Ed. Berlin, Germany: Springer, 2008, pp. 69–91, doi: 10.1007/978-3-540-77465-5_4.
[44] S. Węglarczyk, "Kernel density estimation and its application," in *Proc. ITM Web Conf.*, vol. 23, 2018, Art. no. 00037.
[45] D. W. Scott and S. R. Sain, "Multidimensional density estimation," in *Handbook of Statistics*, vol. 24, C. R. Rao, E. J. Wegman, and J. L. Solka, Eds. Amsterdam, The Netherlands: Elsevier, 2005, pp. 229–261, doi: 10.1016/s0169-7161(04)24009-3.
[46] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, no. 3, pp. 1065–1076, Sep. 1962, doi: 10.1214/aoms/1177704472.
[47] A. Goldenshluger and O. Lepski, "Bandwidth selection in kernel density estimation: Oracle inequalities and adaptive minimax optimality," *Ann. Statist.*, vol. 39, no. 3, pp. 1608–1632, Jun. 2011, doi: 10.1214/11-aos883.
[48] B. Silverman, *Density Estimation for Statistics and Data Analysis*. New York, NY, USA: Springer, 1986.
[49] P. Hall and J. S. Marron, "Choice of kernel order in density estimation," *Ann. Statist.*, vol. 16, no. 1, pp. 161–173, Mar. 1988, doi: 10.1214/aos/1176350697.
[50] P. Hall and J. S. Marron, "On the amount of noise inherent in bandwidth selection for a kernel density estimator," *Ann. Statist.*, vol. 15, no. 1, pp. 163–181, Mar. 1987, doi: 10.1214/aos/1176350259.
[51] B. A. Turlach, "Bandwidth selection in kernel density estimation: A review," Inst. Statistik Okonometrie Humboldt-Univ., Berlin, Germany, Tech. Rep. 9317, 1993.
[52] G. Upton, *Understanding Statistics*. Oxford, U.K.: Oxford Univ. Press, 1996.
[53] L. I. Kuncheva and J. C. Bezdek, "Nearest prototype classification: Clustering, genetic algorithms, or random search?" *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 28, no. 1, pp. 160–164, Feb. 1998.
[54] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition* (Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics). Hoboken, NJ, USA: Wiley, Mar. 1992.
[55] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/a:1010933404324.

[56] J. J. Rodriguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: A new classifier ensemble method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1619–1630, Oct. 2006, doi: 10.1109/tpami.2006.211.
[57] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*. New York, NY, USA: Springer, 1996.
[58] K. Gurney, *An Introduction to Neural Networks*. London, U.K.: Taylor & Francis, 1997.
[59] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[60] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, p. 10, Nov. 2009.
[61] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manage.*, vol. 45, no. 4, pp. 427–437, Jul. 2009.
[62] S. Garcia and F. Herrera, "An extension on 'statistical comparisons of classifiers over multiple data sets' for all pairwise comparisons," *J. Mach. Learn. Res.*, vol. 9, pp. 2677–2694, Dec. 2008.
[63] B. Bergmann and G. Hommel, "Improvements of general multiple test procedures for redundant systems of hypotheses," in *Multiple Hypothesenprüfung/Multiple Hypotheses Testing*. Berlin, Germany: Springer, 1988, pp. 100–115.
[64] M. Topolski, "The modified principal component analysis feature extraction method for the task of diagnosing chronic lymphocytic leukemia type b-CLL," *J. Universal Comput. Sci.*, vol. 26, no. 6, pp. 734–746, 2020.
[65] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert Syst. Appl.*, vol. 73, pp. 220–239, May 2017.

PAWEL TRAJDOS received the Ph.D. degree in computer science from the Wroclaw University of Technology, in 2018. Since then, he has been working with the Department of Systems and Computer Networks, Wroclaw University of Technology. His research interests include computational intelligence and machine learning with application to biology and medicine.

ROBERT BURDUK received the Ph.D. and D.Sc. degrees in computer science in 2003 and 2014, respectively.

He is currently a Professor of computer science with the Department of Systems and Computer Networks, Faculty of Electronics, Wroclaw University of Science and Technology, Poland. He has published more than 100 articles and edited four books, *Computer Recognition Systems* (Springer). His research interests include machine learning, classifier selection algorithms, and multiple classifier systems.

Dr. Burduk serves on program committees for numerous international conferences.