IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON MACHINE LEARNING DESIGNS, IMPLEMENTATIONS AND TECHNIQUES

Received March 25, 2020, accepted April 16, 2020, date of publication April 27, 2020, date of current version May 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2990721

Weighted Scoring in Geometric Space for Decision Tree Ensemble

JEDRZEJ BIEDRZYCKI AND ROBERT BURDUK
Faculty of Electronics, Wroclaw University of Science and Technology, 50-370 Wroclaw, Poland
Corresponding author: Robert Burduk (robert.burduk@pwr.edu.pl)

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

ABSTRACT In order to improve the classification performance of a single classification model, Multiple Classifier Systems (MCS) are used. One of the most common techniques utilizing multiple decision trees is the random forest, where diversity between base classifiers is obtained by bagging the training dataset. In this paper, we propose the algorithm that uses horizontal partitioning the learning set and uses decision trees as base models to obtain decision regions. In the proposed approach feature space is divided into disjoint subspace. Additionally, the location of the subspace centroids, as well as the size and location of decision regions, are used in order to determine the weights needed in the last process of creating MCS, i.e. in the integration phase. The proposed algorithm was evaluated employing multiple open-source benchmarking datasets, compared using accuracy and Matthews correlation coefficient performance measures with two existing MCS methods – random forest and majority voting. The statistical analysis confirms an improvement in recognition compared to the random forest. In addition, we proved that for infinitely dense space division proposed algorithm is equivalent to majority voting.

INDEX TERMS Decision tree, ensemble classifier, majority voting, multiple classifier system, random forest.

I. INTRODUCTION

One of the ways of improving the predictive performance of a single machine learning model is using a committee of classifiers, which is widely known as Multiple Classifier System (MCS) or Ensemble of Classifiers (EoC) [44]. The idea of building MCS is to compose a single strong classifier from the pool of weak ones. Integrating different hypotheses of individual machine learning models reduces the risk of choosing an incorrect one and therefore, improves the overall predictive performance [39]. In general, the procedure of creating EoC can be divided into three major steps [25]:

- Generation – a phase where base classifiers are trained and the pool of base classifiers is created.
- Selection – an optional phase where only several models from the committee are taken to the next phase.
- Integration – a process of combining outputs of multiple classifiers to obtain a single one, integrated model classification. This step is optional if the selection phase results in a single classification model.

The associate editor coordinating the review of this manuscript and approving it for publication was Robert P. Schumaker.

In the generation phase, each base model can be trained using the injection of randomness into the training set or partitioning of the dataset [38]. In horizontal partitioning, the original dataset is divided into several sets that include all features while in vertical partitioning each base model uses a subset of all features [9]. With this procedure, different base models are obtained. Another way to create diversity in the base models is to train them with varied parameter values.

During the selection phase, the competence of each base classifier can be used [5]. The simplest way to do this is by measuring the classification quality for each model. The worst classifiers can be then omitted in the integration phase and the best classifiers can have a greater impact on the integration process. However, it was noticed, that using local competence provides better results compared to generalizing over the whole feature space, since models can perform differently depending on the classification area.

The integration phase can be performed using different types of the classifier output: (1) a class label, (2) a subset of labels ordered by plausibility, (3) a vector of all possible labels with the corresponding support values [25]. One of the most used methods to integrate the class labels of base classifiers is the majority vote rule (MV). In this method,

each base model has the same impact on the final decision of EoC. In the weighted majority voting rule, the integration phase includes probability estimators or other factors of base models to the final decision of MCS [6], [29].

One interesting approach to building the MCS system is clustering and selection algorithm proposed in [24]. This algorithm has its modification presented in [19] and a good performance proven for imbalanced datasets [27]. In general, the clustering and selection algorithms divide the feature space into subspace by K-means or other clustering procedure and select one base classifier for each subspace using performance measure.

In this paper, we propose an algorithm that uses:

- Division of feature space into disjoint subspaces.
- Horizontal partitioning of the original dataset in order to train base models, in our proposal decision trees are used.
- The location of the subspace centroids, as well as the size and location of decision regions defined by base classifiers are used in order to determine the weights needed in the integration phase.

Mutual location in geometric space, subspace centroids and the decision regions define the class label assigned to each subspace. Thus the integration process in the proposed algorithm takes into account the geometric space.

The objectives of this work are as follows:

- A proposal of a new MCS algorithm that uses in the integration phase mutual location of the subspace centroids and the decision regions.
- Proving that for infinitely dense space division the proposed algorithm is equivalent to majority voting.
- A new experimental setup to compare the proposed method with majority voting and random forest algorithms.

The paper is structured as follows: In the next section related work is presented. Section III introduces base concept of supervised classification. The proposed algorithm is presented in section IV. In Section V the experiments that were carried out are presented, while results and discussion are present in section VI. Finally, we conclude the paper in section VII.

II. RELATED WORK

Considerations about classifiers integration using their geometrical representation have been studied for over a decade now [33]. Based on operations in geometrical space generated by real-valued features this procedure has proven itself to be effective in comparison to others, commonly used integration techniques such as majority voting [7]. In one of the previous papers authors have shown significant improvement in the classification by applying weighted mean and median functional to decision boundaries of the SVM classifiers [8].

A geometric approach to the classification problem by Voronoi cells utilization was recently examined by Polianskii and Pokorny [31]. The authors consider Voronoi cells instead of points as basic objects being classified. Boundaries of cells are associated with labels of the closest training objects. Then, integration over the boundaries with regard to associated labels is performed to obtain the most probable class. This approach was tested using SVM, the nearest neighbor and random forest classifiers.

Nearest neighbor classifiers are proven to be efficient when testing which Voronoi cell an object belongs to [3], because there is no need to calculate the geometry of every Voronoi cell. An efficient search lookup was proposed by Kushilevitz *et al.* [26]. The algorithm employs a space–efficient data type that allows to approximate the nearest neighbor in time nearly quadratic regarding dimensionality.

However, the nearest neighbor algorithms are difficult in usage when it comes to specifying the number of prototypes. Using too many leads to high computational complexity. Too few prototypes can cause an oversimplified representation of the decision space especially for datasets that are not linearly separable, have island–shaped decision space, etc. Multiple solutions for this problem were proposed. One way to achieve this is to apply Generalized Condensed Nearest Neighbor rule to obtain a set of prototypes [22]. Each prototype is an object of the training dataset. Another approach was proposed by Gou *et al.* [16]. The first step of the algorithm is to apply kNN algorithm to obtain fixed number of prototypes for every class. Then local mean vectors are calculated to transform the set of prototypes to better represent the decision space distribution.

Decision tree belongs to the simplest and most intuitive machine learning algorithms. They work by recursively partitioning the classification space [37]. Although it has been proposed more than three decades ago [34], decision tree and many its derivatives are very commonly used today [43]. Easy representation, low cost and high quality make decision trees one of the most powerful and popular approaches in data science [37].

It has been noticed, that local quality for each of the base classifiers might differ. The objective of classifier selection is to choose one or a subset of possible base classifiers to perform classification over a region. If the division is known a priori, the selection is called static. Otherwise, models are tested for their quality for the new pattern [32]. Kim and Ko [23] favor local confidence over averaging the quality of classification over the entire space. Combining complementary characteristics of the base models is proven to outperform individual classifiers and several other integration methods.

An interesting approach is combining weighting with local confidence [41]. The authors of the mentioned article notice, that a classifier trained on a subset of training data should be limited to the area it spans in an impact on the resulting classifier.

The problem of generalization of majority voting was studied in [2]. The authors are using a probability estimate calculated as the percentage of properly classified validation objects over geometric constraints. Separately are considered regions that are functionally independent. A significant

improvement in the classification quality was observed when using the proposed algorithm, although knowledge of the domain is needed to provide a proper division. The authors are using a retinal image and classify over anatomic regions.

Another variation on weighted majority voting is class–wise majority voting covered in [35]. Weights are determined for each label separately over the entire validation dataset. This can lead to the improvement of the performance of the resulting integrated classifier.

One of the most popular ensemble methods is a Random Forest introduced by Breiman in 2001 [4]. This technique has proven itself to be very powerful and many related algorithms appeared over the years. In article [14] 179 different classifications algorithms were evaluated using 121 datasets. The results show, that random forest outperforms the majority of examined classifiers. It operates on a pool of base classifiers – decision trees trained on different subsets of the training dataset. An object under test is classified by every model and results are gathered. Finally, majority voting is applied to obtain the most common label.

Extreme Gradient Boosting (XGB) is among the most widely spread, especially in machine learning competitions [11], [40], [42]. The algorithm works by training subsequent decision trees, where consecutive models minimize the value of a loss function generated by its predecessor [15]. Another implementation of Gradient Boosting Decision Tree aiming at performance, especially in case of high dimensionality, is LightGBM [21]. Without loss of performance in classification, the process of training a model can be sped up up to 20 times.

The diversity between base classifiers can be obtained using vertical or horizontal partitioning [37]. It has been proven that for datasets of extreme sizes (very large or small) horizontal partitioning (splitting data into disjoint subsets) outperforms other ensemble methods like bagging or boosting [10]. This provides great possibilities in parallelizing model learning in a distributed environment like p2p network [28].

III. BASIC CONCEPT

The recognition algorithm Ψ maps the feature space X to the set of class labels $\Omega = \{\omega_1, \omega_2, \ldots, \omega_C\}$ according to the general formula:

$$\Psi : X \to \Omega. \tag{1}$$

The classification goal is to assign a given object $x \in X$ into one of the predefined class labels $\omega_i \in \Omega$. Let us assume that K different decision trees $\Psi_1, \Psi_2, \ldots, \Psi_K$ are used to solve the classification task. As a result of all the classifiers' actions, their K responses are obtained. All K base classifiers are applied to make the final decision of MCSs.

The majority voting method allows counting base classifiers outputs as a vote for a class and assigns the input pattern to the class with the greatest count of votes. It is defined as follows:

$$\Psi_{MV}(x) = \arg\max_{\omega_i} \sum_{k=1}^{K} I(\Psi_k(x), \omega_i), \tag{2}$$

where $I(\cdot)$ is the indicator function with the value 1 in the case of the correct classification of the object described by the feature vector x, i.e. when $\Psi_k(x) = \omega_i$.

This means that having a pool of arbitrary classifiers and an object to classify, we assign to the object a label that is indicated by most of the models in the pool. There are different approaches to handle ties (two or more labels are mode), for example random draw from conflicting classes [25].

In the majority vote method each of the individual classifiers takes an equal part in building EoC. Given definition for binary classification is sufficient provided the number of classifiers is odd (to eliminate ties). Otherwise the definition must be modified, usually using random draws or weighting.

The most basic application of MCS to decision trees is random forest. Decision trees are trained on different subsets of training data to obtain different models. Afterwards majority voting is applied (formula (2) is used where decision tree classifiers are inserted in the place of Ψ_k) [18].

IV. PROPOSED METHOD

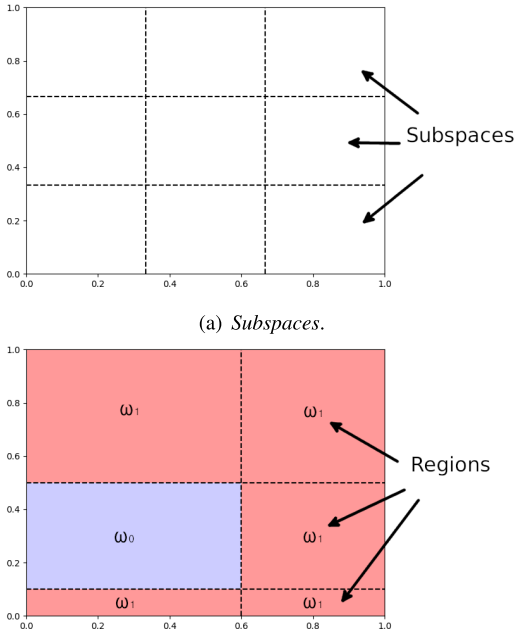
Every decision tree divides space into the finite set of n-dimensional cubes associated with the given class. Let us consider a two-dimensional space as a special case, where decision boundaries can be represented as rectangles. This paper proposes a method of combining multiple decision trees using this representation.

The proposed idea is based on the division of the classification space into rectangular (cubic in case of more dimensions) subsets and assigning classes to them. Further, they will be referred to as classification regions for brevity.

The whole dataset (training and testing subsets) generates a cubic space, that can be divided into smaller ones. Those cubical sets (further referred to as subspaces) are of the same shape as the original dataset but of a different size, since the cube is divided into the same amount of parts along every dimension (feature axis). *Subspaces* and *classification regions* are depicted in fig. 1. Three different levels of granularity were examined: every edge was divided into 20, 40 and 60 parts. For every region a midpoint is calculated.

For each subspace a candidate is resolved as a label of the classification region that spans the midpoint of the considered subspace. For every such candidate weight is derived based on the area (volume) of the classification region. Finally, all intermediate results are aggregated by summing their weights and the class with the largest weight is assigned to the subspace. Since the resulting classifier assigns labels to every rectangular region of competence, it is also a decision tree.

On the other hand, since the classification space is divided into equal subspaces and for all objects within the subspace their label is determined as the label assigned to the midpoint, the resulting classifier can be considered as 1-NN (Nearest



(a) *Subspaces.*

(b) *Classification regions.*

FIGURE 1. Graphical explanation of ***subspace*** and ***classification region.***

Neighbor) classifier with training objects located in the center points of every subspace and subspaces themself become Voronoi cells (areas of points whose closest training point is the midpoint of that area). This eases reasoning about the resulting classifier and simplifies its representation. On the other hand this technique can be useful as a cardinality reduction method and can be applied as an intermediate step in the classification process. The resulting centers of the subspaces can be used as a training set for another classification model and it's resolution can be customized (cardinality of the training set can be any number of the form res^n, where $res \in \mathbb{N}_+$ and n denotes dimensionality).

To keep the notation consistent the classifier that maps the classification region into a label will be denoted as $\Upsilon(A) \equiv \forall_{x \in A} \Psi(x)$. Note that all objects from the classification region A must be classified with the same label by the classifier Ψ.

Let S_m be the m-th subspace, R_l^k – l-th classification region of k-th classifier with label $\omega_i = \Upsilon(R_l^k)$ ($\omega_i \in \Omega$) and x – classified object. Let us denote by M number of partitions of the classification space into subspaces along one dimension. Then, for n-dimensional problem M^n subspaces will be considered. Notice, that k-th decision tree can be completely represented as R^k. If we define $\delta_S(S_m, x)$ as 1 if S_m spans x and 0 otherwise, $\delta_R(R_l^k, S_m)$ as 1 if the midpoint of S_m lies within R_l^k and 0 otherwise, i.e.

$$\delta_S(S_m, x) = \begin{cases} 1 & \text{if } x \in S_m \\ 0 & \text{if } x \notin S_m \end{cases}$$

$$\delta_R(R_l^k, S_m) = \begin{cases} 1 & \text{if } mid(S_j) \in R_l^k \\ 0 & \text{if } mid(S_j) \notin R_l^k \end{cases}$$

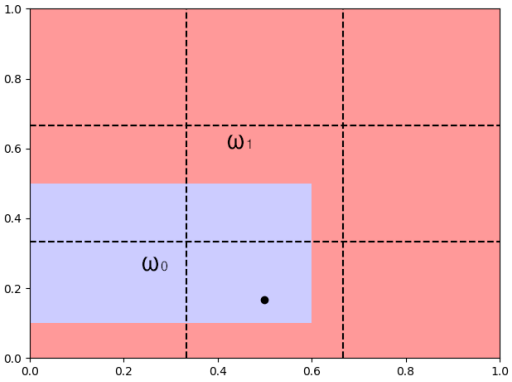


FIGURE 2. The marked midpoint lies within a classification region labeled with ω_0, thus every object in this subspace is assigned label ω_0 with weight defined by the function 5.

and $f_m(R_l^k)$ as a weighting function, we can formalize the proposed algorithm as:

$$\Psi_T(x) = \arg\max_{\omega_i} \sum_{k=1}^{K} \sum_{m=1}^{M^n} \sum_{l=1}^{|R^k|} \delta_S(S_m, x) \delta_R(R_l^k, S_m) f_{\omega_i}(R_l^k). \tag{3}$$

Knowing, that $mid(A)$ is the midpoint of the cubic region A, the equation [3] can be rewritten as:

$$\Psi_T(x) = \arg\max_{\omega_i} \sum_{k=1}^{K} \sum_{m=1}^{M^n} \sum_{l=1}^{|R^k|} \delta(x, S_m, R_l^k) f_{\omega_i}(R_l^k), \tag{4}$$

where $\delta(x, S_m, R_l^k) = \delta_S(S_m, x)\delta_S(R_l^k, mid(S_m))$.

The effective computational complexity is reduced as most terms of the equation [4] are omitted because either $\delta_S(S_m, x)$ or $\delta_S(R_l^k, mid(S_m))$ resolves to 0.

In this paper only the proportional and inversely proportional weighting function was examined: $f_{vol}(A) = volume(A), f_{inv}(A) = \frac{1}{volume(A)}$.

Notice, that f_{ω_i} depends on label ω_i. It assigns weight to the label, thus it is a shorthand:

$$f_{\omega_i}(R_l^k) = f_{wt}(R_l^k) I(\Upsilon_k(R_l^k), \omega_i), \tag{5}$$

where $I(\cdot)$ is the indicator function from equation [2] and $wt \in \{vol, inv\}$.

Fig. 2 shows an example of weight calculation. Since the midpoint of the considered subspace lies within the classification region classified with the label ω_0, label for every object in this subspace is assigned to ω_0 with weight calculated using formula [5].

The algorithm is depicted in fig. 3. Given the subspace and weighting function proportional to the area, the first classifier (fig. (a)) assigns weight to label ω_0 and the second (fig. (b)) – to ω_1, because the midpoint lies in the respective classification regions. Weight associated with label ω_0 is smaller than the one associated with ω_1, because it's proportional to the area of classification regions. The resulting classifier (fig. (c)) aggregates calculated weights and assigns label ω_1

Algorithm 1: Algorithm to Obtain Integrated Decision Tree Using Cubic Subspaces

Input : K – number of base classifiers $(\Psi_1, \Psi_2, \ldots, \Psi_K)$, M - number of edge divisions, n - number of dimensions, dataset

Output: Integrated decision tree Ψ_T

1 Select the most informative features and normalize the dataset.
2 Divide the dataset into $K+1$ subsets (K for training base classifiers and 1 for testing).
3 Train base classifiers $\Psi_1, \Psi_2, \ldots, \Psi_K$ to obtain their geometrical representation (classification regions with labels).
4 Divide the feature space into M^n identical cubic subspaces.
5 For each subspace determine the classification region that spans the midpoint of the considered subspace and evaluate competence of the label using weighting function using formula 5.
6 Sum weights for every label over every classifier and assign to the subspace label with the greatest sum according to [4].

TABLE 1. Descriptions of datasets used in experiments (name with abbreviation, number of instances, number of features, imbalance ratio).

Dataset	#inst	#f	Imb
QSAR biodegradation (bio)	1055	41	0.51
Liver Disorders (BUPA) (bup)	345	6	0.73
Cryotherapy (cry)	90	7	0.88
Banknote authentication (dba)	1372	5	0.80
Haberman's Survival (hab)	306	3	0.36
Ionosphere (ion)	351	34	0.56
Ultrasonic flowmeter diagnostics (met)	540	173	0.69
Climate model simulation crashes (pop)	540	18	0.09
Seismic-bumps (sei)	2584	19	0.07
Breast Cancer (Diagnostic) (wdb)	569	30	0.59
Breast Cancer (Original) (wis)	699	9	0.54

to the entire subspace, since this is the label with the greatest sum of weights.

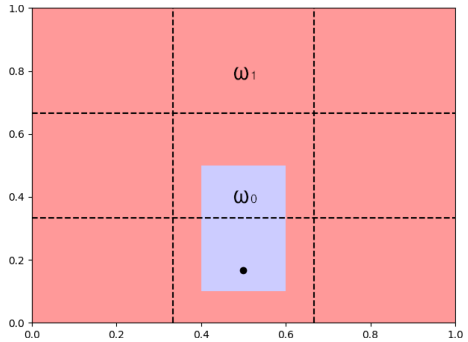
Lemma: Majority Voting with an arbitrary weighting function that depends on the region only is a special case of the presented algorithm for the infinitely dense space division.

Proof: First let us notice, that for any training point x there are only one S_x and R_s^k that fulfil the following:

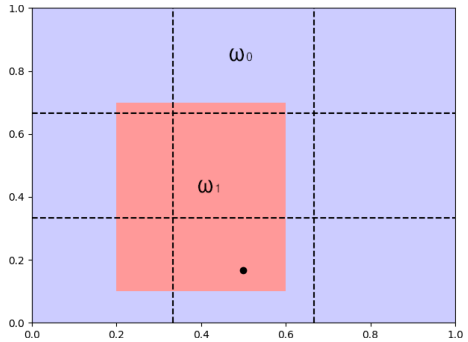
$$\exists S_x \in \{S_m\} : x \in S_x$$
$$\forall_k \exists R_s^k \in \{R_l^k\} : s = mid(S_x) \wedge s \in R_s^k. \quad (6)$$

This is the consequence of exclusiveness of subspaces and subregions: $S_i \cap S_j = \emptyset \wedge R_i^k \cap R_j^k = \emptyset$ for every $i \neq j$.

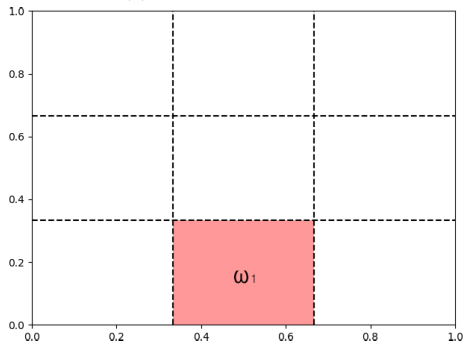
The special case of the proposed algorithm is when the division into subspaces becomes infinitely dense. This means, that the size of every subspace becomes infinitely small and



(a) First base classifier.



(b) Second base classifier.



(c) Resulting classification of subspace.

FIGURE 3. Label calculation using two base classifiers and mapping function proportional to area.

shrinks to a single point, which results in:

$$\lim_{|S_x| \to 0} mid(S_x) = x. \quad (7)$$

Combining equations [6] and [7] we can obtain from [4]:

$$\lim_{M \to \infty} \Psi_T(x) = argmax_{\omega_i} \sum_{k=1}^{K} f_{\omega_i}(R_x^k), \quad (8)$$

where R_x denotes the decision tree region, that spans x. This proves the lemma. ■

The consequence of the proven lemma is that majority voting is a special case of the presented algorithm for the constant weighting function (without weighting) and infinitely dense partition into subspaces.

V. EXPERIMENTAL SETUP

The pool of classifiers consisted of 5 decision trees of depth of 3. Decision tree implementation from scala library spark

TABLE 2. ACC and mean rank for the proportional weighting function.

	bio	bup	cry	dba	hab	ion	met	pop	sei	wdb	wis	rank
Ψ_{mv}	0.723	0.600	0.778	0.907	0.709	0.783	0.654	0.892	0.932	0.924	0.953	**2.32**
Ψ_{rf}	0.719	0.576	0.738	0.899	0.697	0.770	0.577	0.893	0.931	0.924	0.936	4.41
Ψ_{vol}^{20}	0.723	0.594	0.784	0.909	0.706	0.774	0.684	0.890	0.932	0.925	0.953	4.41
Ψ_{vol}^{40}	0.723	0.592	0.764	0.904	0.699	0.778	0.678	0.890	0.932	0.923	0.953	3.23
Ψ_{vol}^{60}	0.723	0.614	0.764	0.913	0.709	0.778	0.677	0.890	0.932	0.918	0.953	2.64

TABLE 3. ACC and mean rank for the inversely proportional weighting function.

	bio	bup	cry	dba	hab	ion	met	pop	sei	wdb	wis	rank
Ψ_{mv}	0.723	0.600	0.778	0.907	0.709	0.783	0.654	0.892	0.932	0.924	0.953	**2.23**
Ψ_{rf}	0.719	0.576	0.738	0.899	0.697	0.770	0.577	0.893	0.931	0.924	0.936	4.41
Ψ_{inv}^{20}	0.723	0.583	0.776	0.909	0.704	0.774	0.684	0.890	0.932	0.925	0.953	2.59
Ψ_{inv}^{40}	0.723	0.588	0.764	0.904	0.699	0.778	0.678	0.890	0.932	0.923	0.953	3.14
Ψ_{inv}^{60}	0.723	0.614	0.764	0.913	0.709	0.778	0.677	0.890	0.932	0.918	0.953	2.64

TABLE 4. MCC and mean rank for the proportional weighting function.

	bio	bup	cry	dba	hab	ion	met	pop	sei	wdb	wis	rank
Ψ_{mv}	0.415	0.169	0.583	0.815	0.140	0.567	0.328	-0.004	0.000	0.833	0.898	2.55
Ψ_{rf}	0.401	0.115	0.463	0.797	0.078	0.538	0.140	0.000	-0.001	0.831	0.860	4.50
Ψ_{vol}^{20}	0.415	0.166	0.594	0.819	0.141	0.543	0.378	-0.005	0.000	0.835	0.898	**2.32**
Ψ_{vol}^{40}	0.415	0.129	0.594	0.808	0.121	0.554	0.360	-0.005	0.000	0.831	0.898	3.14
Ψ_{vol}^{60}	0.415	0.205	0.594	0.826	0.140	0.554	0.364	-0.005	0.000	0.820	0.898	2.50

TABLE 5. MCC and mean rank for the inversely proportional weighting function.

	bio	bup	cry	dba	hab	ion	met	pop	sei	wdb	wis	rank
Ψ_{mv}	0.415	0.169	0.583	0.815	0.140	0.567	0.328	-0.004	0.000	0.833	0.898	**2.36**
Ψ_{rf}	0.401	0.115	0.463	0.797	0.078	0.538	0.140	0.000	-0.001	0.831	0.860	4.50
Ψ_{inv}^{20}	0.415	0.144	0.581	0.819	0.137	0.543	0.378	-0.005	0.000	0.835	0.898	2.68
Ψ_{inv}^{40}	0.415	0.142	0.594	0.808	0.121	0.554	0.360	-0.005	0.000	0.831	0.898	3.09
Ψ_{inv}^{60}	0.415	0.205	0.594	0.826	0.140	0.554	0.364	-0.005	0.000	0.820	0.898	**2.36**

was utilised. To conduct statistical tests numpy [30] and scipy [20] were used.

As referential classifiers majority voting and random forest were used. The experiments were conducted for edge division into 20, 40 and 60 parts. Edge division into M parts means creating M^n areas of competence, where n is the number of dimensions. In this paper only the two-dimensional feature space is considered, but the algorithm is easily applicable to any number of dimensions without any modifications.

Two antagonistic weighting methods were used: proportional and inversely proportional to the volume. Weight of every decision tree competence region can be calculated as its area (inverse of the area respectively). To assure even influence of every dimension, datasets were normalized to span over square space.

The experiments were conducted using open-source datasets available on platforms UCI Machine Learning Repository [13] and KEEL Data Set Repository [1]. Datasets are presented in table 1. The imbalance ratio is shown to stress the need to use metric sensitive for highly imbalanced datasets (MCC). For all datasets the feature selection process [17], [36] was performed to indicate two most informative features.

It's also important to notice, that no other requirements besides knowledge of extrema of the feature space is needed to apply the algorithm.

Integrated classifiers are designated as Ψ_{weight}^{M}, where *weight* denotes weighting function (*vol* for proportional to volume or *inv* for inversely proportional to volume) and M - number of divisions along the dimension. Referential classifiers are designated as Ψ_{alg}, where *alg* means algorithm used: majority voting – *mv* or random forest – *rf* .

VI. RESULTS AND DISCUSSION

The main aim of the experiments was to compare the quality of classification of the proposed method with referential algorithms. Statistical tests were performed to compare the improvement achieved by using the proportional and inversely proportional weighting function for decision trees. In order to compare the quality of the classification,

TABLE 6. p–values of ranked Friedman tests for the examined algorithms.

		Quality measure	
		ACC	MCC
Weighting	Proportional	0.004	0.006
	Inversely proportional	0.004	0.005

two classification measures were used: accuracy (ACC) and Matthews correlation coefficient (MCC).

ACC is the most commonly used quantity, but it reflects the quality of classifier very poorly, when applied to imbalanced datasets. Suppose the imbalance quotient for the binary classification problem ($\frac{\#\text{minor class objects}}{\#\text{major class objects}}$) equals $\frac{1}{9}$, then the model classifying every object with the most common label will receive a score of 90%. MCC takes the imbalance of dataset into account, what makes it more reliable in case of datasets used, where the imbalance reaches 0.07.

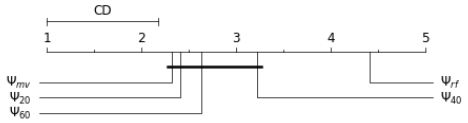
Tables 2 and 3 show the results of ACC and tables 4 and 5 – the results of MCC. Every experiment was conducted 10 times and the average is presented. Along with quality measures, average ranks obtained in nonparametric Friedman tests are written in the last column.

p-values of Friedman tests are shown in table6. They do not exceed the value of 0.01, what means, that not all algorithms perform equally. To determine which algorithms are odd, post–hoc Bonferroni–Dunn tests were conducted.

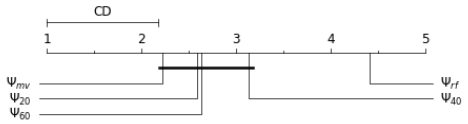
This test was carried out for each division of the feature space (the feature space was divided into 20^2, 40^2 and 60^2 areas of competence). The difference in the quality of classification in some cases was observed, because Bonferroni–Dunn test requires the difference in Friedman ranks to be at least 1.18 (3 algorithms are compared against referential, 11 datasets are used) to reject the null hypothesis at a significance level of $\alpha = 0.1$ [12].

The best results are marked with the bolded font. For every division granularity and every quality measure classifiers obtained using both weighting functions perform significantly better than random forest. Additionally, using the proportional weighting function provides better results than majority voting when comparing MCC.

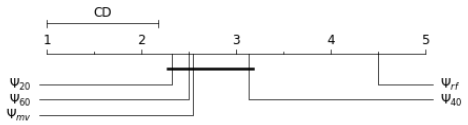
Fig. 4 depicts a statistical analysis of the obtained results. The graphs show rankings obtained in Friedman's tests for every algorithm. The lower the rank, the better the quality of the classifier. For the reference the critical difference between ranks is presented in the top–left corner. The bolded line spans over values, whose ranks differences are smaller than the critical value. This means, that according to Bonferroni–Dunn's test, they are indistinguishable.



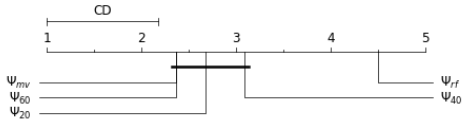
(a) Ranks of Friedman test of ACC for proportional weighting function.



(b) Ranks of Friedman test of ACC for inversely proportional weighting function.



(c) Ranks of Friedman test of MCC for proportional weighting function.



(d) Ranks of Friedman test of MCC for inversely proportional weighting function.

FIGURE 4. Comparison of quality of proposed algorithm, majority voting and random forest.

VII. CONCLUSION

In this article the algorithm of the decision tree integration in the geometric space along with two weighting functions were proposed. The algorithm is less restrictive than majority voting when it comes to the number of base classifiers, since the possibility of draw is low. The resulting classifier is also a decision tree, which makes it easy to reason about and serialize.

Because all subspaces are cubes of the same size, they are also Voronoi cells. This means, that the resulting model is also a nearest neighbor classifier with centroids placed in the centers of every subspace.

It was also proven, that the weighted majority voting is a special case of the presented algorithm, when the division granularity is infinite.

Eleven open-source benchmarking datasets were used in the experimental part to perform the statistical analysis of the results concerning two classification measures. MCC was used aside of ACC because of the high imbalance of the datasets used. MCC unlike ACC takes the imbalance into account, what makes it more reliable. Bonferroni–Dunn tests showed, that for all division densities the proposed algorithm resulted in the better classification quality than the random forest for datasets used. Additionally for proportional weighting function MCC of the integrated classifier outperformed the random forest.

REFERENCES

[1] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Multiple-Valued Log. Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2011.

[2] A. Hajdu, L. Hajdu, A. Jonas, L. Kovacs, and H. Toman, "Generalizing the majority voting scheme to spatially constrained voting," *IEEE Trans. Image Process.*, vol. 22, no. 11, pp. 4182–4194, Nov. 2013.

[3] G. Biau and L. Devroye, *Lectures Nearest Neighbor Method*. New York, NY, USA: Springer, 2015.

[4] L. Breiman, "Random forests," *Mach. Learn. Arch.*, vol. 45, no. 1, pp. 5–32, 2001.
[5] A. S. Britto, Jr., R. Sabourin, and L. E. Oliveira, "Dynamic selection of classifiers—A comprehensive review," *Pattern Recognit.*, vol. 47, no. 11, pp. 3665–3680, 2014.
[6] R. Burduk, "Classifier fusion with interval-valued weights," *Pattern Recognit. Lett.*, vol. 34, no. 14, pp. 1623–1629, Oct. 2013.
[7] R. Burduk, "Integration base classifiers in geometry space by harmonic mean," in *Artificial Intelligence and Soft Computing*, L. Rutkowski, R. Scherer, M. Korytkowski, W. Pedrycz, R. Tadeusiewicz, and J. M. Zurada, Eds. Cham, Switzerland: Springer, 2018, pp. 585–592.
[8] R. Burduk and J. Biedrzycki, "Integration and selection of linear svm classifiers in geometric space," *J. Universal Comput. Sci.*, vol. 25, no. 6, pp. 718–730, Jun. 2019.
[9] J. Cao, G. Lv, C. Chang, and H. Li, "A feature selection based serial SVM ensemble classifier," *IEEE Access*, vol. 7, pp. 144516–144523, 2019.
[10] N. V. Chawla, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "Learning ensembles from bites: A scalable and accurate approach," *J. Mach. Learn. Res.*, vol. 5, pp. 421–451, Dec. 2004.
[11] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 785–794.
[12] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.
[13] D. Dua and C. Graff, *UCI Machine Learning Repository*. Oakland, CA, USA: Univ. of California, Irvine, School of Information and Computer Sciences, 2017.
[14] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3133–3181, 2014.
[15] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.
[16] J. Gou, Y. Zhan, Y. Rao, X. Shen, X. Wang, and W. He, "Improved pseudo nearest neighbor classification," *Knowl.-Based Syst.*, vol. 70, pp. 361–375, Nov. 2014.
[17] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Jan. 2003.
[18] D. Ruppert, "The elements of statistical learning: Data mining, inference, and prediction," *J. Amer. Stat. Assoc.*, vol. 99, no. 466, p. 567, Jun. 2004.
[19] K. Jackowski and M. Wozniak, "Algorithm of designing compound recognition system on the basis of combining classifiers with simultaneous splitting feature space into competence areas," *Pattern Anal. Appl.*, vol. 12, no. 4, pp. 415–425, Dec. 2009.
[20] E. Jones, T. Oliphant, and P. Peterson. *SciPy: Open Source Scientific Tools for Python*. Accessed: 2001. [Online]. Available: http://www.scipy.org
[21] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 3149–3157.
[22] S. R. Kheradpisheh, F. Behjati-Ardakani, and R. Ebrahimpour, "Combining classifiers using nearest decision prototypes," *Appl. Soft Comput.*, vol. 13, no. 12, pp. 4570–4578, Dec. 2013.
[23] E. Kim and J. Ko, "Dynamic classifier integration method," in *Multiple Classifier Systems*, N. C. Oza, R. Polikar, J. Kittler, F. Roli, Eds. Berlin, Germany: Springer, 2005, pp. 97–107.
[24] L. I. Kuncheva, "Clustering-and-selection model for classifier combination," in *Proc. 4th Int. Conf. Knowl.-Based Intell. Eng. Syst. Allied Technol. (KES)*, vol. 1, 2000, pp. 185–188.
[25] I. L. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. New York, NY, USA: Wiley, 2004.
[26] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, "Efficient search for approximate nearest neighbor in high dimensional spaces," *SIAM J. Comput.*, vol. 30, no. 2, pp. 457–474, Jan. 2000.
[27] P. Lopez-Garcia, A. D. Masegosa, E. Osaba, E. Onieva, and A. Perallos, "Ensemble classification for imbalanced data based on feature space partitioning and hybrid metaheuristics," *Int. J. Speech Technol.*, vol. 49, no. 8, pp. 2807–2822, Aug. 2019.
[28] P. Luo, H. Xiong, K. Lü, and Z. Shi, "Distributed classification in peer-to-peer networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2007, pp. 968–976.
[29] S. Mao, L. Jiao, L. Xiong, S. Gou, B. Chen, and S.-K. Yeung, "Weighted classifier ensemble based on quadratic form," *Pattern Recognit.*, vol. 48, no. 5, pp. 1688–1706, May 2015.
[30] T. Oliphant. *NumPy: A Guide to NumPy*. USA: Trelgol Publishing, 2006.
[31] V. Polianskii and T. F. Pokorny, "Voronoi boundary classification: A high-dimensional geometric approach via weighted Monte Carlo integration," in *Proc. 36th Int. Conf. Mach. Learn.*, vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds. Long Beach, CA, USA, Jun. 2019, pp. 5162–5170.
[32] M. P. Ponti, Jr., "Combining classifiers: From the creation of ensembles to the decision fusion," in *Proc. 24th SIBGRAPI Conf. Graph., Patterns, Images Tuts.*, Aug. 2011, pp. 1–10.
[33] O. Pujol and D. Masip, "Geometry-based ensembles: Toward a structural characterization of the classification boundary," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 6, pp. 1140–1146, Jun. 2009.
[34] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.
[35] A. F. R. Rahman, H. Alam, and M. C. Fairhurst, "Multiple classifier combination for character recognition: Revisiting the majority voting system and its variations," in *Document Analysis Systems V*, D. Lopresti, J. Hu, R. Kashi, Eds. Berlin, Germany: Springer, 2002, pp. 167–178.
[36] I. Rejer, "Genetic algorithms for feature selection for brain-computer interface," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 29, no. 5, 2015, Art. no. 1559008.
[37] L. Rokach, "Decision forest: Twenty years of research," *Inf. Fusion*, vol. 27, pp. 111–125, Jan. 2016.
[38] M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "Vote-boosting ensembles," *Pattern Recognit.*, vol. 83, pp. 119–133, Nov. 2018.
[39] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 8, no. 4, p. e1249, 2018.
[40] P. Robert Sheridan and W. M. Wang, "Extreme gradient boosting as a method for quantitative structure-activity relationships," *J. Chem. Inf. Model.*, vol. 56, no. 12, pp. 2353–2360, Dec. 2016.
[41] M. S. Zia, M. Hussain, and M. A. Jaffar, "A novel spontaneous facial expression recognition using dynamically weighted majority voting based ensemble classifier," *Multimedia Tools Appl.*, vol. 77, no. 19, pp. 25537–25567, Oct. 2018.
[42] S. B. Taieb and R. J. Hyndman, "A gradient boosting approach to the Kaggle load forecasting competition," *Int. J. Forecasting*, vol. 30, no. 2, pp. 382–394, Apr. 2014.
[43] P.-N. Tan, M. M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Boston, MA, USA: Addison-Wesley, 2005.
[44] A. Ulaş, M. Semerci, O. T. Yıldız, and E. Alpaydın, "Incremental construction of classifier and discriminant ensembles," *Inf. Sci.*, vol. 179, no. 9, pp. 1298–1318, Apr. 2009.

JEDRZEJ BIEDRZYCKI received the B.S. degree in biomedical engineering and the M.S. degree in computer science from the Wroclaw University of Science and Technology, Poland, in 2017 and 2018, respectively, where he is currently pursuing the Ph.D. degree in computer science with the Department of Systems and Computer Networks, Faculty of Electronics.

His major research interests are machine learning and multiple classifier systems.

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. His research interests cover among the others machine learning, classifier selection algorithms, and multiple classifier systems. He serves on program committees of numerous international conferences, published over 100 articles, and edited six books.