Binary classification: Ensemble Methods Utilizing Decision Theory Tools

Oksana Pichugina[a,b], Lyudmyla Kirichenko[c,d] and Tamara Radivilova[c]

[a]National Aerospace University "Kharkiv Aviation Institute", 17 Chkalova Street, Kharkiv, 61070 Ukraine
[b]University of Toronto, 27 King's College Circle, Toronto, M5S 1A1, Canada
[c]Kharkiv National University of Radio Electronics, 14 Nauki Avenue, Kharkiv, 61166 Ukraine
[d]Wroclaw University of Science and Technology, 27 Wyspianskiego, Wroclaw, 50-370 Poland

Abstract

Several ensemble methods of binary classification are presented. They are based on the use of decision theory tools at the stage of aggregating the results of binary classification and obtaining refined solutions to classification problems. Results of software implementation and computational experiments on benchmark instances are presented. The experiment conducted on unbalanced benchmark instances from KEEL–dataset repository demonstrates a noticeable improvement in the quality characteristics of binary classification such as accuracy and balanced accuracy. The presented approach is expected to be promising for complex classification instances such as unbalanced classification ones.

Keywords

Binary classification, Ensemble method, Decision Theory, priority vector, aggregation, accuracy

Introduction

The process of predicting classes for a given set of points is called classification. These classes are also called categories or labels. For instance, detecting spam in e-mail correspondence can be considered a classification problem, determining whether a given message is spam. This is a so-called binary classification problem in which only two classes participate: class 0 – regular messages, and class 1 - spam. Similarly, the problem of detecting fraudulent financial transactions can be seen as a binary classification problem, where class 0 is normal transactions, and class 1 is fraudulent. A distinctive feature of these two problems is that the number of elements of class 1 is much less than the number of other elements and the total number of observations. This imbalance greatly complicates qualitative classification and requires the development of classification theory in order to derive new approaches to solving complex classification problems.

Classification is traditionally referred to as a supervised machine learning class. Here, a machine learns from a training set in which some characteristics of a training set and its labels are known. After that, elements of the instance for which labels are unknown are put into the machine as input, while labels are obtained as output playing the role of predictions. There are many practical applications of classification in finance, economics, medicine, engineering, management, and so on [1, 2, 3, 4, 5, 6, 7].

Despite the huge amount of data involved in machine learning, the capabilities of modern computers make it possible in some cases to solve the same problem repeatedly in order to obtain a more

2nd International Workshop of IT-professionals on Artificial Intelligence (ProfIT AI 2022), December 2-4, 2022, Łódź, Poland
o.pichugina@khai.edu (O. Pichugina); lyudmyla.kirichenko@nure.ua (L. Kirichenko); tamara.radivilova@nure.ua (T. Radivilova)
0000-0002-7099-8967 (O. Pichugina); 0000-0002-2780-7993 (L. Kirichenko); 0000-0001-5975-0269 (T. Radivilova)

accurate solution to the problem. That is, in this case, the solution accuracy is a higher priority than the resources of computers used. In particular, in supervised machine learning, which deals with two main problems – classification and regression – the direction of the so-called aggregation or ensemble methods has been developed last years, where the same problem is solved in several ways, and then the results obtained are aggregated, and the final solution is formed. In this paper, we will present several ensemble methods for solving the binary classification problem based on the local priority vector, traditionally used in the Analytic Hierarchy Process [8, 9, 10].

It will be shown that these approaches to aggregation are promising and, in some cases, give a significant improvement in the quality of classification compared to the results of applying standard methods of conventional classification.

1. Prerequisites

1.1. Classification problems typology

A **classification problem** (**CP**) [11, 12] is a problem of identifying to which class a new instance belongs based on available class membership for samples in a **training set** (**TS**).

Instances to which classification is applied form a **test set** (**TeS**). A TS-instance is given by a feature tuple z and a class label y. At the same time, a TeS-instance is represented by a feature tuple, while a class label is unknown and needs to be found.

z can be a numeric vector but not necessarily since features presented in z-components characterize certain instance' properties, categorical, ordinal, integer-valued, or real-valued. If categorical or ordinal features are present, a preprocessing stage is required to map x into Euclidean space:

$$z \xrightarrow{\phi} x \in \mathbb{X} \subset \mathbb{R}^m. \tag{1}$$

Let us assume that the mapping (1) is done, and we deal with the training and test sets

$$\text{TS: } \{\langle x_i, y_i \rangle\}_{i \in J_n}; \tag{2}$$

$$\text{TeS: } \{x'_i\}_{i \in J_{n'}}, \tag{3}$$

where $J_n = \{1, ..., n\}$, $x_i \in \mathbb{R}^m$, $i \in J_n$ and $x'_i \in \mathbb{R}^m$, $i \in J_{n'}$.

A **classification algorithm** (**CA**) is intended to train a **classifier**, which is a function mapping an instances' space $\mathbb{X}$ into a class label space $\mathbb{Y}$:

$$f : \mathbb{X} \to \mathbb{Y}. \tag{4}$$

Let

$$\mathcal{C} = \{C_0, ..., C_l\} \tag{5}$$

be a set of classes. To $\mathbb{X}$ we will refer to as an instances' space and to a set

$$\mathbb{Y} = J_l^0 \tag{6}$$

as a class label space (hereafter, $J_l^0 = \{0, ..., l\}$).

The following division of classification problems (CPs) is common [11, 12, 13, 14]:

1. Depending on **the number of classes**:

a) **Binary classification problems** (Binary CPs, **BCPs**) if the number of classes is two, i.e., $l = 1$;
b) **Multi-class classification problems** (Multiclass CPs, **MCPs**) if $l > 1$;

2. Depending on **the proportions of the classes' sizes**:
a) **Balanced classification problems** (**BaCPs**), if the difference in classes' sizes is statistically insignificant;
b) **Imbalanced classification problems** (**class imbalance problems, ICPs**) if this difference is significant.

1.2. Classification metrics

Quality of classification is assessed by different metrics [11, 15, 16] such as the accuracy, balanced accuracy, recall, precision, F-score, G-mean, etc., which utilize a confusion matrix.

A **confusion matrix** (**CM**) is a matrix of the dimension $s+1$. Each its row represents the instances in an **actual class** (with an **actual label**), while its column represents the instances in a **predicted class** (with a **predicted label**). Namely,

$$CM = (n_{ij})_{i,j\in J_s^0},$$

where n_{ij} is a number of instances from an actual class C_i, whose predicted class is C_j (with an actual label y_i and a predicted label y_j).

The number of n_{ii} is called the **true prediction** of the class C_i ($i \in J_s^0$).

Let

$$n_i' = \sum_{j=0}^{s} n_{ij},\ i \in J_s^0,$$

be a distribution of the classes in $\mathcal{C}$, where n_i' be a class C_i-size. Then the number n of all $\mathcal{C}$-instances satisfies a relation

$$n = \sum_{i=0}^{s} n_i' = \sum_{i,j=0}^{s} n_{ij}.$$

In these notations, the listed above classification metrics are represented as follows.

Classification accuracy (accuracy) is calculated as a sum of all true predictions divided by n:

$$AC = \frac{1}{n}\sum_{i=0}^{s} n_{ii}. \tag{7}$$

AC is inadequate in reflecting the classifier's performance on classifying each single class, especially on small classes [17, 16]. That is why, in ICPs, other metrics are more commonly used. Among them are the following two:

$$R_i = \frac{n_{ii}}{n_i'},\ i \in J_s^0; \tag{8}$$

$$P_j = \frac{n_{jj}}{\sum_i n_{ij}}, j \in J_s^0. \tag{9}$$

After replacing j by i the later becomes

$$P_i = \frac{n_{ii}}{\sum_j n_{ji}}, i \in J_s^0. \tag{10}$$

R_i, P_i are called a **recall** (the true positive rate, **TPR**) and a **precision** of a class C_i, respectively. By themselves, neither of these two characteristics are adequate in reflecting the performance of any classifier on the class C_i. Therefore, they are commonly integrated in a metric called the **F-score (F-measure)** for a class C_i [18]:

$$F_i = \frac{2 \cdot P_i \cdot R_i}{P_i + R_i}, i \in J_s^0.$$

A peculiarity of F-score is that it is high if both the recall and precision are high.

If we are interested in the performances of all classes, the classification performance of each class should be equally represented in the evaluation metric. G-mean [19] is such a measure, which is the geometric mean of recall values of every class:

$$G_{mean} = \left(\prod_{i=0}^{s} R_i \right)^{1/(s+1)}.$$

Another metric is the **balanced classification accuracy**:

$$BAC = \frac{1}{s+1} \sum_{i=0}^{s} R_i.$$

The formula (8) can be rewritten as follows:

$$R_i = \frac{TP_i}{TP_i + FN_i}, i \in J_s^0, \tag{11}$$

where

$$TP_i = n_{ii}, FN_i = n_i^{'} - TP_i, \tag{12}$$

are called a **true positive** and a **false negative** for a class C_i, respectively ($i \in J_s^0$).

Note, if $i \neq j$, then the value n_{ij} is an **error** of prediction for an C_i-instance that it elongs to a class C_j.

In these notations, a sum across the whole row i is $n_i^{'} = TP_i + FN_i$. A **false positive** for C_i, denoted by FP_i, is a sum of a column i excluding TP_i. A **true negative** for C_i is $TN_i = n - (TP_i - FN_i - FP_i)$, i.e., it is a sum of elements of a CM except for the row i and column i [20].

Using this terminology, formulas (7), (8), and (10) become:

$$AC = \frac{1}{n} \sum_{i=0}^{s} TP_i; \tag{13}$$

$$R_i = \frac{TP_i}{TP_i + FN_i}, i \in J_s^0; \tag{14}$$

$$P_i = \frac{TP_i}{TP_i + FP_i}, i \in J_s^0. \tag{15}$$

If BCPs are solved, a positive class is considered as the main class

$$C = C_1,$$

and the subindex i is omitted in formulas (11)-(15) yielding most common expression for an accuracy, a recall, and a precision:

$$AC = \frac{1}{n} (TP + TN), \tag{16}$$

$$R = \frac{TP}{TP+FN}, \tag{17}$$
$$P = \frac{TP}{TP+FP}, \tag{18}$$

where

$$FN = n_{00},\ FP = n_{01},\ FN = n_{10}, TP = n_{11}.$$

Respectively, in these notations, the expressions for BAC, F-score, and G_{mean} are as follows:

$$BAC = \frac{1}{2}\left(\frac{TN}{TN+FP} + \frac{TP}{FN+TP}\right); \tag{19}$$
$$F = \frac{2 \cdot P \cdot R}{P+R}; \tag{20}$$
$$G_{mean} = \left(\frac{TN}{TN+FP} \cdot \frac{TP}{FN+TP}\right)^{1/2}. \tag{21}$$

One more often used method for evaluating classification performance in BCPs is Receiver Operating Characteristics analysis (**ROC**) [21, 15, 11].

A ROC graph (ROC curve) depicts the relative trade-offs between TP and FP. It is plotted with R against **the false positive rate**:

$$FPR = \frac{FP}{FP + TN}, \tag{22}$$

where R is on y-axis and FPR is on the x-axis.

AUC (**Area Under The Curve**) is an area under the ROC curve [21].

1.3. Ensemble techniques

Combining classifiers and, as a consequence, combining results of classification, is one of the commonly accepted methods for improving the quality of classification and increasing the reliability of these results [22, 23, 24].

Combining methods are widely used in regression [22] and classification [23, 24] problems. These methods are called **ensemble algorithms** (**EAs**). Two subclasses in this group of methods can be singled out – classification EAs (**ECAs**) and regression EAs (**ERAs**).

Collections of EAs used together are called **ensemble systems** (**ESs**) [23]. Among them are classification ESs (**ECSs**) and regression ESs (**ERSs**). ESs are designed not only for a formation of a collection of predictions obtained in different combining ways, but also for analysis and comparison of these results in order to form a single solution to a prediction problem under consideration.

EAs utilize an idea ”the more diverse the training set, base classifiers, and feature set, the better the performance of the ES” [23].

Based on that, in [25], six strategies were described for designing an EAs:

1. different initialization;
2. different parameter choice;
3. different architecture;
4. different classifiers;
5. different training sets;
6. different feature sets.

Two of the listed strategies are most commonly used, namely, different training sets (also known as **Homogeneity Scenario** [26] and different classifiers (known as **Heterogeneity Scenario**). Respectively, ESs are divided into homogeneous ESs (**HoESs**) and heterogeneous ESs (**HeESs**). Thus, ECSs can be either classification HoESs (**HoECSs**) or classification HeESs (**HeECSs**).

Further, we will focus on ECSs. Elements of HoECSs and HeECSs we will call homogeneous ECAs (**HoECAs**) and heterogeneous ECAs (**HeECAs**), respectively.

In a HoECA, base classifiers are generated from applying a base CA on different training sets formed from the original training set. The predicted labels obtained by these classifiers are then combined in final predicted labels of the test set. In this category of ensemble methods are Random Forest, Adaptive Boosting, Bagging, Random Subspace (see [23] and references therein), etc.

In contrast to HoECAs, in a HeECA, different CAs are applied on the same training set, thus producing a set of different base classifiers, which outputs are called **meta-data** [27]. Then metadata are combined in final predicted labels obtained as a result of applying this HeECA.

Remark 1. *Note that HoECAs involve a large number of different "weak" classifiers. Therefore, a base CA should not be fine-tuned. At the same time, HeECAs commonly utilize a small number of "strong" classifiers, hence involved base CAs need a preliminary tuning. Both groups of ECAs utilize probabilities, i.e., they deal with probabilistic CAs [13], thus preventing involving deterministic (non-probabilistic) CAs such as SVM, SGB, Kernel Logistic Regression, Logistic Model Trees etc. [28].*

HeECAs are also divided into two groups – **fixed** HeECAs and **trainable** HeECAs [23]. The main difference between these two is that, when combining, fixed ones do not take into consideration the label information in the meta-data of training set while trainable methods do.

Fixed HeECAs combines base classifiers' results by **sum**, **product**, **min**, **max**, **median, majority vote** rules etc. [29]. Among trainable HeECAs are the Stacking Algorithm, Inference-based Combiner, Multiple Response Linear Regression, SCANN, Decision Template (see [23] and references therein) and so on.

In this research, we set the task of constructing a universal HeECA consisting of fixed and trainable HeECAs and involving deterministic and probabilistic CAs. We expect that using well-known HoECAs along with standard CAs will result in designing a highly strong classifier, which outperforms existing ones.

2. The proposed heterogeneous ensemble classification system

2.1. Denotations

Let us introduce some notations

- $J_n = \{1, ..., n\}$, $J_n^0 = J_n \cup \{0\}$;
- $CA = \left\{ CA_j \right\}_{j \in J_m}$ – is a set of base classification algorithms (CAs);
- $EA = \left\{ ECA_{j'} \right\}_{j' \in J_{m'}}$ – is a set of our ensemble classification algorithms (methods) (**EAMs**), which is a HeECS (further referred to as an **expert assessment based heterogeneous ensemble classification system, a expert assessment based HeECS, EA-HeECS)**.

  $ECA = \left\{ ECA_{j'} \right\}_{j' \in J_{m'}}$ – is a set of our ensemble classification algorithms (**EAMs**, **ECAs**);
- $\{O_i\}_{i \in J_n}$ – is a set of instances (observations, examples, samples, statistical units);

- $\langle X|Y\rangle$ – is a tuple of observations on samples and their labels:
  - $X = (\overline{x}_i)_{i\in J_n} = (x_{il})_{i\in J_n, l\in J_L}$ – is a real-valued matrix of observations on L independent (explanatory) variables;
  - $Y = (y_i)_{i\in J_n}$ – is a vector of **actual labels** thus

$$y_i \in \mathbb{Y}, i \in J_n. \tag{23}$$

  - $\widehat{Y} = (\widehat{y}_i)_{i\in J_n}$ – is a vector of **predicted labels** such that

$$\widehat{y}_i = f(\overline{x}_i) \in \mathbb{Y},\ i \in J_n. \tag{24}$$

- I – is a **K-fold split** of J_n, i.e., it is a partition of J_n into K subsets of nearly equal size:

$$\begin{aligned} I = \{I_k\}_{k\in J_K} : \quad & \bigcup_{k=1}^{K} I_k = J_n,\ I_k \cap I_{k'} = \emptyset, \forall k \neq k'; \\ & n_k \approx n_{k'}, \forall k \neq k', \\ \text{where} \qquad & n_k = |I_k|, k \in J_K. \end{aligned} \tag{25}$$

The partition (25) induces K pairs of **test sets**:

$$TeS_k,\ k \in J_K, \tag{26}$$

and **training sets**:

$$TS_k, k \in J_K, \tag{27}$$

such that

$$TeS_k = \{\overline{x}_i, y_i\}_{i\in I_k}, TS_k = \{\overline{x}_i, y_i\}_{i\notin I_k}, k \in J_K, \tag{28}$$

The collection (26) defines a partition of the dataset $\langle X, Y\rangle$. At the same time, the sets (27) defines a decomposition of the tuple, where each sample $\langle x, y\rangle \in \langle X, Y\rangle$ is presented $k-1$ times.

CAs yield predictions of labels, namely,

$$\widehat{y}_{ij}, \widehat{y}'_{ij'},\ i \in J_n,\ j \in J_m,\ j' \in J_{m'},$$

will be a predicted label of O_i obtained by $CA_j \in CA$ or $ECA_{j'} \in EA - HeECS$, respectively.

Note that since in CAs classification models are built and optimized on training sets, values

$$\widehat{y}_{ij}, \widehat{y}'_{ij'}, i \in I_k, j \in J_m, j' \in J_{m'}, k \in J_K, \tag{29}$$

are real predictions, while

$$\widehat{y}_{ij}, \widehat{y}'_{ij'}, i \notin I_k, j \in J_m, j' \in J_{m'}, k \in J_K, \tag{30}$$

are labels' predictions are compatible with the actual labels

$$y_i, i \notin I_k, k \in J_K, \tag{31}$$

Therefor, they can be used for evaluating quality of classification on training sets and tuning CAs if necessary.

Let us unite the predictions (29) matrices:

$$Y^k = \left(\widehat{y}_{ij}\right)_{i\in I_k, j\in J_m}, \; Y'^k = \left(\widehat{y}'_{ij}\right)_{i\in I_k, j'\in J_{m'}}, k \in J_K, \tag{32}$$

by CAs and ECAs, respectively.

For training sets (26), imbalanced ratios are:

$$IR_k = \frac{|TS_k| - n'_{k1}}{n'_{k1}} = \frac{n - n_k - n'_{k1}}{n'_{k1}}, \quad \text{where} \;\; n'_{k1} = \sum_{i\notin I_k} y_i \tag{33}$$

is the size of C_1 in TS_k, $k \in J_K$.

Note that, in ICPs, an imbalanced ratio IR can be given prior. In this case,

$$IR_k = IR, k \in J_K,$$

is reasonably to use instead of (33).

Also, in some ECAs, weights of CAs playing a role of experts in solving CPs are used. The weights can also be assigned prior, but we will use a posterior information obtained as a result of comparison of (30) with (31). Namely, we introduce vectors of relative weights of CAs for a fold k:

$$W^k = \left(w_j^k\right)_{j\in J_m}, \; k \in J_K, \tag{34}$$

where w_j^k is a weight of a CA_j, which depends on a quality of classification achieved by this method on a training set TS_k.

The vectors (34) must satisfy the following conditions:

$$W^k \geq 0, \;\; \left\|W^k\right\|_1 = 1, k \in J_K. \tag{35}$$

They can be found in different ways depending on our preferences, either the accuracy, balanced accuracy, recall, precision, G-mean, F-score, AUC, or another classification measure **M** we aim to improve.

All these approaches can be combined in the following way. Let

$$\begin{aligned} &AC^k = \left(AC_j^k\right)_j, BAC^k = \left(BAC_j^k\right)_j, R^k = \left(R_j^k\right)_j, \\ &P^k = \left(P_j^k\right)_j, G^k = \left(G_j^k\right)_j, F^k = \left(F_j^k\right)_j, \\ &AUC^k = \left(AUC_j^k\right)_j, M = \left(M_j^k\right)_j, k \in J_K, \end{aligned} \tag{36}$$

where

$$AC_j^k, BAC_j^k, R_j^k, P_j^k, G_j^k, F_j^k, AUC_j^k, M_j^k, k \in J_K, j \in J_m,$$

be values of AC, BAC, R, P, G-mean, F-score, AUC, and the metric $M \in [0, 1]$ achieved by a CA_j applied on a TS_k;

$$\alpha = (\alpha_r)_{r\in J_8} \in \mathbb{R}_+^8 \; : \; \|\alpha\|_1 = 1 \tag{37}$$

be a vector of weights of the listed metrics.

Then the vector (34) is defined as follows:

$$W^k = \frac{1}{A^k}\left(\alpha_1 AC^k + \alpha_2 BAC^k + \alpha_3 R^k + \alpha_4 P^k + \alpha_5 G^k + \alpha_6 F^k + \alpha_7 AUC^k + \alpha_8 M^k\right), \tag{38}$$

where

$$A^k = \left\|\alpha_1 AC^k + \alpha_2 BAC^k + \alpha_3 R^k + \alpha_4 P^k + \alpha_5 G^k + \alpha_6 F^k + \alpha_7 AUC^k + \alpha_8 M^k\right\|_1, \ k \in J_K.$$

For instance, a choice of $\alpha = \alpha^1 = \left(1, 0^7\right)$ – means that we are interested in increasing AC only;
$\alpha = \alpha^2 = \left(0, 1, 0^6\right)$ – implies that we focus in higher BAC first of all;
$\alpha = \alpha^3 = \left(0^2, 1, 0^5\right)$ – that we are interested in higher R;
$\alpha = \alpha^4 = \left(0^4, 1, 0^3\right)$ – that we attempt to increase G-mean;
$\alpha = \alpha^5 = \left(0^5, 1, 0^2\right)$ – that we wish to maximize AUC;
$\alpha = \alpha^6 = \left((1/6)^3, 0, (1/6)^3, 0\right)$ – that all listed standard classification metrics, except for R and **M** are involved, and they are equal, etc.

2.2. New ECAs description

Fix $k \in J_K$ and $j \in J_m$. To the vector

$$\widehat{y}_j^k = \left(\widehat{y}_{ij}\right)_{i \in I_k}, \tag{39}$$

a local priority vector (LPV) [9, 10, 30]

$$p_j^k = \left(p_{ij}\right)_{i \in I_k}, \tag{40}$$

is associated such that

$$p_{ij}/p_{i'j} = \begin{cases} IR_k, \text{ if } O_i \in C_1, O_{i'} \in C_0; \\ (IR_k)^{-1}, \text{ if } O_i \in C_0, O_{i'} \in C_1; \quad \left(i, i' \in I_k\right) \\ 1, \quad \text{otherwise;} \end{cases} \tag{41}$$

$$\left\|p_j^k\right\|_1 = 1. \tag{42}$$

Remark 2. *A vector p_j^k will have at most 2 different coordinates, therefore, in order to find it, an auxiliary vector can be formed with unit coordinates for instances with 0 predicted label and the rest coordinates equal to IR_k. Then, this vector needs a normalization yielding a vector satisfying (41) and (42).*

Based on an assumption that IR_k is an imbalanced ratio for TeS_k, i.e., imbalanced ratios are the same for training and test sets, we split the test set TeS^k into classes C_0, C_1 such that

$$|C_0| = n_k^0, \ |C_1| = n_k^1 \ : \ n_k^1 = \left\lceil \frac{n_k}{1 + IR_k} \right\rceil, n_k^0 = n_k - n_k^1. \tag{43}$$

2.2.1. ECA1 (based on utilizing the geometric mean of expert estimates)

The LPVs (40) are combined in the following way:

- Find an auxiliary vector

$$z_1^{'k} = \left(z_{i1}^{'}\right)_{i \in I_k} \ : \ z_{i1}^{'} = \left(\prod_{j=1}^{m} p_{ij}\right)^{1/m}, \ i \in I_k. \tag{44}$$

- Find a threshold value

$$thr_1^k \ : \ z'_{i_1 1} \geq z'_{i_2 1} \geq ... \geq z'_{i_{n_k^1} 1} = thr_1^k > z'_{i_{n_k^1+1} 1} \geq ... \geq z'_{i_{n_k} 1}. \tag{45}$$

- Assign

$$\widehat{y}'^k_{i1} = \begin{cases} 1, \ if \ z'_{i1} \geq thr_1^k; \\ 0, \ otherwise. \end{cases} \quad (i \in I_k). \tag{46}$$

2.2.2. ECA2 (based on using the weighted geometric mean of expert estimates)

First, we generalize (45), (46) in the following way:

$$thr_{j'}^k \ : \ z'_{i_1 j'} \geq z'_{i_2 j'} \geq ... \geq z'_{i_{n_k^1} j'} = thr_{j'}^k > z'_{i_{n_k^1+1} j'} \geq ... \geq z'_{i_{n_k} j'}; \tag{47}$$

$$\widehat{y}'^k_{ij'} = \begin{cases} 1, \ if \ z'_{ij'} \geq thr_{j'}^k; \\ 0, \ otherwise. \end{cases} \quad (i \in I_k), \tag{48}$$

where $j' \in J_{m'}$, thus replacing in the formulas the sub-index 1 by j'.

Now, for ECA2, the LPVs (40) are combined using the weights (34), formulas (47), (48) are applied with

$$j' = 2,$$

and the auxiliary vector is

$$z'^k_{j'} = z'^k_2 = \left(z'_{i2}\right)_{i \in I_k} \ : \ z'_{i2} = \prod_{j=1}^{m} p_{ij}^{w_j^k}, \ i \in I_k. \tag{49}$$

2.2.3. ECA3 (the majority voting)

Here, we first assign

$$j' = 3;$$

then find a vector of votes and a threshold:

$$z'^k_3 = \left(z'_{i3}\right)_{i \in I_k} \ : \ z'_{i3} = \sum_{j=1}^{m} \widehat{y}_{ij}^k, \ i \in I_k; \tag{50}$$

$$thr_3^k = \frac{m}{2}. \tag{51}$$

Finally, apply the formula (48).

2.2.4. ECA4 (the weighted majority voting)

In this case, we follow a scheme:

- set

$$j' = 4;$$

- find a vector of weighted votes and a threshold:

$$z'^k_4 = \left(z'_{i4}\right)_{i \in I_k} \ : \ z'_{i4} = \sum_{j=1}^{m} w_j^k \widehat{y}_{ij}^k, \ i \in I_k; \tag{52}$$

$$thr_4^k = thr_3^k = \frac{m}{2}, \tag{53}$$

- finally, (48) is applied.

Remark 3. *ECA3 and ECA4 can be implemented in a slightly different way in a manner of ECA1, ECA2, if the formula* (47) *is used for deriving* thr_3^k, thr_4^k *instead of* (51), (53).

2.2.5. ECA5 (the iterative method of finding experts estimates)

Set

$$j' = 5.$$

Let J_T^0 be a set of iterations and $t \in J_T^0$ be an iteration index,

$$W_t^k = \left(w_{jt}^k\right)_{j \in J_m} \tag{54}$$

be a vector of the weights of CAs for a fold k on iteration t ($t \in J_T^0$, $k \in J_K$).

For the vector (54), constraints similar to (35) hold, namely,

$$W_t^k \geq 0, \quad \left\| W_t^k \right\|_1 = 1, k \in J_K, t \in J_T^0, \tag{55}$$

2.2.6. ECA5 outline

- **Input**. $k \in J_K$, $\varepsilon > 0$, a matrix

  $$P^k = \left(p_j^k\right)_{j \in J_m},$$

  where vectors p_j^k, $j \in J_m$, are found by (40).

- **Step 0. Initialization**. $t = 0$, all the weights (55) are equal, thus

  $$W_0^k = \left(\frac{1}{m}, ..., \frac{1}{m}\right) \in \mathbb{R}^m.$$

- **Step 1.** Set $t = t + 1$.

- **Step 2.** Find an estimate of y^k on iteration t denoted by

  $$\widehat{y}^{k(t)} = \left(\widehat{y}_i^{(t)}\right)_{i \in I_k} \tag{56}$$

  in the following way:

  $$\widehat{y}^{k(t)} = P^k W_{t-1}^k. \tag{57}$$

- **Step 3.** A vector W_t^k is evaluated based on W_{t-1}^k in a such a way that its components increases for those $j \in J_m$, whose vectors of estimates $\widehat{y}_j^k$ are closer to (57), and become lower for the rest of CAs. For that, an auxiliary parameter

  $$\lambda^{k(t)} = \left(\widehat{y}^{k(t)}\right)^T \left(P^k \mathbf{1}\right), \tag{58}$$

  where $\mathbf{1} \in \mathbb{R}^n$ is a vector of units, is calculated first. Then W_t^k is found with the help of (57), (58):

  $$W_t^k = (\lambda^{k(t)})^{-1} \left(P^k\right)^T \widehat{y}^{k(t)}. \tag{59}$$

- **Step 4.** If the given accuracy ε is not achieved yet, i.e.,

$$\frac{\left\| \widehat{y}^{k(t)} - \widehat{y}^{k(t-1)} \right\|_1}{\left\| \widehat{y}^{k(t-1)} \right\|_1} > \varepsilon, \tag{60}$$

  then go to Step 1. Otherwise, set $T = t$ and terminate.

- **Outputs** are:

$$W^k = W_T^k,\ z_5^{'k} = \left(z_{i5}^{'} \right)_{i \in I_k} = \widehat{y}^{k(T)},\ i \in I_k.$$

Then (47), (48) are applied.

2.3. DS-ECS modification and genaralization

To an approach of assigning the CAs' weights described earlier in this section, where the whole training set TS_k is used for that, we will refer to as **Approach 1** (**A1**). Another approach to assign weights of CAs, that can be uses in ECAs (further referred to as **Approach 2**, **A2**), is based on an observation that, to be more realistic, it might be beneficial to split, in addition, the training sets (27) into auxiliary test and training sets:

$$TeS_k^{'}, TS_k^{'} : TeS_k^{'} \cup TS_k^{'} = TS_k, TeS_k^{'} \cap TS_k^{'} = \emptyset, k \in J_K, \tag{61}$$

for instance, making a 10%/90% split, additionally perform training on $TS_k^{'}$, and then assign the weights depending on the quality of classification achieved on $TeS_k^{'}$. In this case, (34) is replaced by

$$W^{'k} = \left(w_j^{'k} \right)_{j \in J_m}, k \in J_K, \tag{62}$$

where $w_j^{'k}$ is the weight of CA_j, which reflects a quality of classification reached on the auxiliary test set $TeS_k^{'}$ by applying CA_j on an auxiliary training set $TS_k^{'}$.

Now, for Approach 2, the formula (36) becomes:

$$\begin{aligned} &AC^{'k} = \left(AC_j^{'k} \right)_j, BAC^{'k} = \left(BAC_j^{'k} \right)_j, R^{'k} = \left(R_j^{'k} \right)_j, \\ &P^{'k} = \left(P_j^{'k} \right)_j, G^{'k} = \left(G_j^{'k} \right)_j, F^{'k} = \left(F_j^{'k} \right)_j, \\ &AUC^{'k} = \left(AUC_j^{'k} \right)_j, M = \left(M_j^{'k} \right)_j, k \in J_K, \end{aligned} \tag{63}$$

where $AC_j^{'k}, BAC_j^{'k}, R_j^{'k}, P_j^{'k}, G_j^{'k}, F_j^{'k}, AUC_j^{'k}, M_j^{'k}, k \in J_K$ are AC, BAC, R, P, G-mean, F-score, AUC, **M** evaluated on $TS_k^{'}$ after applying CA_j ($j \in J_m$). Respectively, (38) becomes:

$$\begin{aligned} &W^{'k} = \frac{1}{A^{'k}} \left(\alpha_1 AC^{'k} + \alpha_2 BAC^{'k} + \alpha_3 R^{'k} + \alpha_4 P^{'k} + \alpha_5 G^{'k} + \alpha_6 F^{'k} + \alpha_7 AUC^{'k} + \alpha_8 M^{'k} \right), \\ &A^{'k} = \left\| \alpha_1 AC^{'k} + \alpha_2 BAC^{'k} + \alpha_3 R^{'k} + \alpha_4 P^{'k} + \alpha_5 G^{'k} + \alpha_6 F^{'k} + \alpha_7 AUC^{'k} + \alpha_8 M^{'k} \right\|_1, \end{aligned} \tag{64}$$

where $k \in J_K$.

Besides, ECA_2, ECA_4, which use the weights of CAs, can be adapted to usage of (64), namely, (49) and (52) become:

$$z_2^{'k} = \left(z_{i2}^{'} \right)_{i \in I_k} : z_{i2}^{'} = \prod_{j=1}^{m} p_{ij}^{w_j^{'k}},\ i \in I_k; \tag{65}$$

$$z_4^{'k} = \left(z_{i4}^{'} \right)_{i \in I_k} : z_{i4}^{'} = \sum_{j=1}^{m} w_j^{'k} \widehat{y}_{ij}^{k},\ i \in I_k. \tag{66}$$

Remark 4. *We will refer to these modifications as $ECA_2^{'}, ECA_4^{'}$, respectively.*

3. Computational Experiment

For validating of the proposed ensemble approaches, 7 conventional classification algorithms implemented in R were selected:

1. Linear Logistic Regression (**GLM**) [31];
2. k-Nearest Neighbors (**KNN**) [32];
3. Linear Support Vector Machine (**SVM**) [33];
4. Kernel SVM (**KSVM**) [34];
5. Naive Bayes (**NB**) [35];
6. Decision Tree (**DT**) – C4.5 [36], CART [2];
7. Random Forest (**RF**) [2]

and 2 unbalanced benchmark instances from KEEL–dataset repository [37] were used – Pima Indians Diabetes (Pima) and Haberman Breast Cancer (Haberman) for which standard classification algorithm gives rather low quality – accuracy around 70%.

As a result of application of our ensemble approaches, accuracy and balanced accuracy was improved by 1% and 6%, respectively. The best method is ECA2.

Conclusion

In this paper, we offer new ensemble methods for binary classification which use Decision Theory tools for combining conventional classifiers' results. Our aggregative techniques work better on datasets, where standard methods are weak such as Haberman and Pima. Thus, the proposed approaches are promising for application in complex real-world classification problems.

Acknowledgments

The work was partially supported by Beethoven Grant No. DFG NCN 2016/23/G/ST1/04083.

References

[1] L. Kirichenko, T. Radivilova, V. Bulakh, Machine Learning in Classification Time Series with Fractal Properties, Data 4 (2019) 5. doi:10.3390/data4010005.

[2] L. Breiman, J. Friedman, C. J. Stone, R. A. Olshen, Classification and Regression Trees, 1st ed., Chapman and Hall/CRC, Boca Raton, Fla., 1984.

[3] D. Forsyth, Applied Machine Learning, Springer International Publishing, 2019. doi:10.1007/978-3-030-18114-7.

[4] P. Vuttipittayamongkol, E. Elyan, A. Petrovski, On the class overlap problem in imbalanced data classification, Knowledge-Based Systems 212 (2021) 106631. doi:10.1016/j.knosys.2020.106631.

[5] V. A. Perepelitsa, N. K. Maksishko, I. V. Kozin, Using a model of cellular automata and classification methods for prediction of time series with memory 42 (2006) 807–816. doi:10.1007/s10559-006-0121-4.

[6] L. Kirichenko, O. Pichugina, T. Radivilova, K. Pavlenko, Application of wavelet transform for machine learning classification of time series, in: S. Babichev, V. Lytvynenko (Eds.), Lecture Notes in Data Engineering, Computational Intelligence, and Decision Making, Lecture Notes on Data Engineering and Communications Technologies, Springer International Publishing, 2023, pp. 547–563. doi:10.1007/978-3-031-16203-9_31.

[7] L. Kirichenko, O. Pichugina, H. Zinchenko, Clustering time series of complex dynamics by features, in: Selected Papers of the VIII International Scientific Conference "Information Technology and Implementation" (IT&I-2021). Conference Proceedings, volume 3132 of *CEUR Workshop Proceedings*, 2021, pp. 83–93. ISSN: 1613-0073.

[8] O. Pichugina, Decision Making Tools For Choice Software Development Environment, in: 2020 IEEE KhPI Week on Advanced Technology (KhPIWeek), 2020, pp. 450–454. doi:10.1109/KhPIWeek51551.2020.9250109.

[9] T. L. Saaty, J. M. Alexander, Conflict Resolution: The Analytic Hierachy Approach, Praeger Pub, New York, 1989.

[10] T. L. Saaty, Analytic Hierarchy Process, in: S. I. Gass, M. C. Fu (Eds.), Encyclopedia of Operations Research and Management Science, Springer US, 2013, pp. 52–64. doi:10.1007/978-1-4419-1153-7_31.

[11] C. Drummond, Classification, in: C. Sammut, G. I. Webb (Eds.), Encyclopedia of Machine Learning, Springer US, Boston, MA, 2010, pp. 168–171. doi:10.1007/978-0-387-30164-8_111.

[12] E. Alpaydin, Introduction to machine learning, Adaptive computation and machine learning, 2nd ed., MIT Press, Cambridge, Mass, 2010. OCLC: ocn317698631.

[13] C. C. Aggarwal, Data Classification: Algorithms and Applications, 1st ed., Chapman & Hall/CRC, 2014.

[14] N. Japkowicz, S. Stephen, The class imbalance problem: A systematic study, Intelligent Data Analysis (2002) 429–449.

[15] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, 4th ed., Morgan Kaufmann, Amsterdam, 2016.

[16] Y. Sun, M. S. Kamel, Y. Wang, Boosting for Learning Multiple Classes with Imbalanced Class Distribution, in: Sixth International Conference on Data Mining (ICDM'06), 2006, pp. 592–602. doi:10.1109/ICDM.2006.29, iSSN: 2374-8486.

[17] C. Drummond, R. C. Holte, Severe Class Imbalance: Why Better Algorithms Aren't the Answer, in: J. Gama, R. Camacho, P. B. Brazdil, A. M. Jorge, L. Torgo (Eds.), Machine Learning: ECML 2005, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2005, pp. 539–546. doi:10.1007/11564096_52.

[18] D. Lewis, W. A. Gale, Training text classifiers by uncertainty sampling, in: Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information, NY, New York, 1998, pp. 73–79. URL: /paper/Training-text-classifiers-by-uncertainty-sampling-Lewis-Gale/d9d4c586b985af2ec42e4fec24cd1806d9aefaff.

[19] M. Kubat, R. C. Holte, S. Matwin, Machine Learning for the Detection of Oil Spills in Satellite Radar Images, Machine Learning 30 (1998) 195–215. doi:10.1023/A:1007452223027.

[20] L. Abdi, S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling and boosting techniques, Soft Computing 19 (2015) 3369–3385. doi:10.1007/s00500-014-1291-z.

[21] A. P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (1997) 1145–1159. doi:10.1016/S0031-3203(96)00142-2.

[22] L. Abdi, S. Hashemi, To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling

Techniques, IEEE Transactions on Knowledge and Data Engineering 28 (2016) 238–251. doi:10.1109/TKDE.2015.2458858.

[23] T. T. Nguyen, X. C. Pham, A. W.-C. Liew, W. Pedrycz, Aggregation of Classifiers: A Justifiable Information Granularity Approach, IEEE Transactions on Cybernetics 49 (2019) 2168–2177. doi:10.1109/TCYB.2018.2821679.

[24] D. Ndirangu, W. Mwangi, L. Nderu, A Hybrid Ensemble Method for Multiclass Classification and Outlier Detection, International Journal of Sciences: Basic and Applied Research (IJSBAR) 45 (2019) 192–213.

[25] R. Duin, The combining classifier: to train or not to train?, in: Object recognition supported by user interaction for service robots, volume 2, 2002, pp. 765–770 vol.2. doi:10.1109/ICPR.2002.1048415, iSSN: 1051-4651.

[26] T. T. Nguyen, T. T. T. Nguyen, X. C. Pham, A. W.-C. Liew, A novel combining classifier method based on Variational Inference, Pattern Recognition 49 (2016) 198–212. doi:10.1016/j.patcog.2015.06.016.

[27] S. Džeroski, B. Ženko, Is Combining Classifiers with Stacking Better than Selecting the Best One?, Machine Learning 54 (2004) 255–273. doi:10.1023/B:MACH.0000015881.36452.6e.

[28] V. Rodrigues, S. Deusdado, Deterministic Classifiers Accuracy Optimization for Cancer Microarray Data, in: F. Fdez-Riverola, M. Rocha, M. S. Mohamad, N. Zaki, J. A. Castellanos-Garzón (Eds.), Practical Applications of Computational Biology and Bioinformatics, 13th International Conference, volume 1005, Springer International Publishing, Cham, 2020, pp. 154–163. doi:10.1007/978-3-030-23873-5_19.

[29] J. Kittler, M. Hatef, R. Duin, J. Matas, On combining classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1998) 226–239. doi:10.1109/34.667881.

[30] M. Brunelli, Introduction to the Analytic Hierarchy Process, SpringerBriefs in Operations Research, Springer International Publishing, 2015. doi:10.1007/978-3-319-12502-2.

[31] P. McCullagh, J. A. Nelder, Generalized Linear Models, 2nd ed., Chapman and Hall/CRC, Boca Raton, 1989.

[32] N. S. Altman, An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression, The American Statistician 46 (1992) 175–185. doi:10.2307/2685209.

[33] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (1995) 273–297. doi:10.1007/BF00994018.

[34] I. W. Tsang, J. T. Kwok, P.-M. Cheung, Core Vector Machines: Fast SVM Training on Very Large Data Sets, Journal of Machine Learning Research 6 (2005) 363–392. URL: http://jmlr.org/papers/v6/tsang05a.html.

[35] P. Domingos, M. Pazzani, On the Optimality of the Simple Bayesian Classifier under Zero-One Loss, Machine Learning 29 (1997) 103–130. doi:10.1023/A:1007413511361.

[36] J. R. Quinlan, C4.5: Programs for Machine Learning, 1st ed., Morgan Kaufmann, San Mateo, Calif, 1992.

[37] I. Triguero, S. González, J. M. Moyano, S. García, J. Alcalá-Fdez, J. Luengo, A. Fernández, M. J. d. Jesús, L. Sánchez, F. Herrera, KEEL 3.0: An Open Source Software for Multi-Stage Analysis in Data Mining, International Journal of Computational Intelligence Systems 10 (2017) 1238–1249. doi:10.2991/ijcis.10.1.82.