Proceedings of Machine Learning Research 1:1–13, 2010 Workshop Title

Undersampled Majority Class Ensemble for highly imbalanced binary classification

Paweł Ksieniewicz PAWEL.KSIENIEWICZ@PWR.EDU.PL
Department of Systems and Computer Networks
Faculty of Electronics
Wrocław University of Science and Technology

Editor: Editor's name

Abstract

Following work tries to utilize an ensemble approach to solve a problem of highly imbalanced data classification. Paper contains a proposition of UMCE – a multiple classifier system, based on *k-fold division* of the *majority class* to create a pool of classifiers breaking one *imbalanced problem* into many balanced ones while ensuring the presence of all available samples in the training procedure. Algorithm, with five proposed fusers and a pruning method based on the statistical dependencies of the classifiers response on the testing set, was evaluated on the basis of the computer experiments carried out on the benchmark datasets and two different base classifiers.

Keywords: classification, classifier ensemble, undersampling, imbalanced data

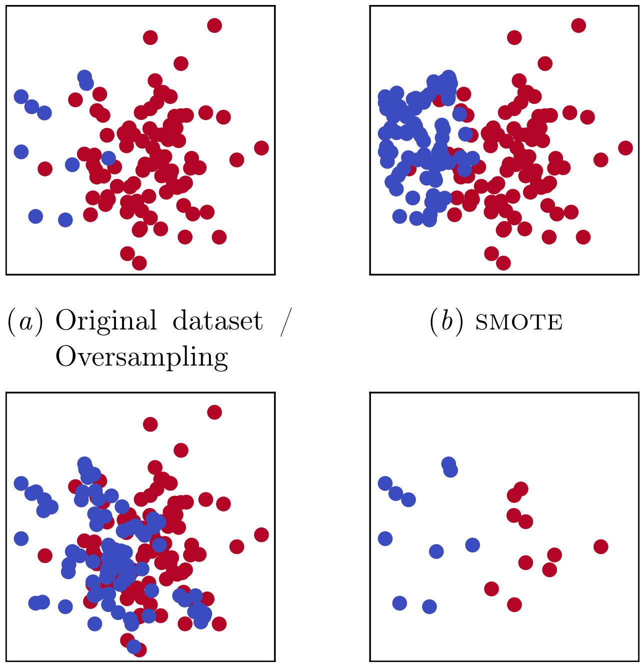
1. Introduction

Most of existing classification models benefit from the assumption that there are no significant disparities between the classes of the considered problem. Nevertheless, in the real world, there are many situations in which the number of objects from one of the classes (called the *majority class*) significantly exceeds the number of objects of the remaining classes (*minority classes*), which often leads to decisions biased towards the *majority class*. However, when considering cases such as spam filtering, medical tests or fraud detection, we may come to the conclusion that the cost of making an incorrect decision against a minority class is much greater than in other cases. The above-mentioned problem is called in the literature the *imbalanced data classification* (Wang et al., 2017; Sun et al., 2009).

Following work focuses on the binary classification of the highly imbalanced problems, with an IR (*imbalanced ratio*) greater than 9, which is an important issue not only in the context of the construction of appropriate models, but even in a proper quality measurement (Elazmeh et al., 2006). One of the important problems is also the fact that the number of patterns in the *minority class* may be so small that it will not allow to achieve the appropriate discriminatory power of the model, which may lead to its *overfitting* (Chen and Wasikowski, 2008). Most of these problems are the subject of extensive research (Bunkhumpornpat et al., 2009; Chawla et al., 2002).

One of the possible approaches to solve such problems are *inbuild mechanisms*, trying to adapt existing classification models to balance the accuracy between classes. Popular solution of this kind is the learning approach without counter-examples, using *one-class* *classification* (Japkowicz et al., 1995; Krawczyk et al., 2014), where the aim is to get to know the decision boundaries within *minority classes*. The solution may also be the *cost sensitive solutions*, assuming the asymmetric *loss function* (Lopez et al., 2012; He and Garcia, 2009).

Another approach, more connected with the scope of following paper, is the group of *data preprocessing methods*, which focuses on reducing the number of *majority class* objects (*undersampling*) or generating patterns of *minority class* (*oversampling*) to balance a dataset. Graphical overview of methods from this group is presented in Figure 1.



(a) Original dataset / Oversampling

(b) SMOTE

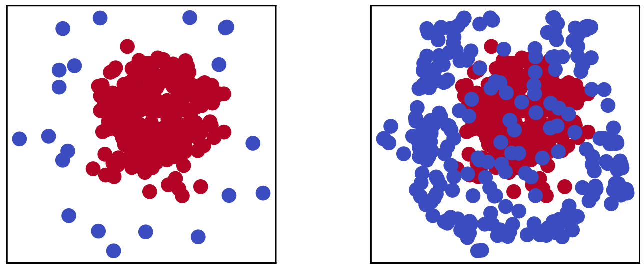
(c) ADASYN

(d) Undersampling

Figure 1: Examples of data preprocessing methods.

These algorithms are addressing the task of balancing the number of objects within the problem classes. In the case of basic *oversampling*, new objects are created as random copies of those already existing in the training set[1]. Currently, the most common kind of *oversampling* is SMOTE (Chawla et al., 2011), shown in Figure 1(b), creating new, synthetic objects based on k averaged examples nearest to a random points from the space occupied by a minority class. An active version of SMOTE is the ADASYN algorithm (He et al., 2008), shown in Figure 1(c), which takes into account the difficulty of synthetic samples. This approach allows to solve the problem of repeating samples in the training set, but can also lead to *overfitting*, which is presented in Figure 2.

1. Since the characteristics of the new patterns will be identical to those already present in the dataset, we can consider Figure 1(a), an illustration of the original dataset, also as the presentation of pattern distribution after oversampling.



(*a*) Original dataset (*b*) SMOTE

Figure 2: Example of wrong SMOTE oversampling.

In the case of *undersampling*, shown in Figure 1(*d*), in which we draw as many objects from the majority class as are present in the minority class, there is no risk of erroneous mixing of the classes distribution.

The last group of methods to be mentioned here are *hybrid approaches*, combining *over-* and *undersampling* algorithms with *ensemble classifiers* (Galar et al., 2012). The *Bagging* and *Boosting* variants, such as *AdaBoost.NC* (Wang et al., 2010) or *SMOTEBoost* (Chawla et al., 2003), have become particularly popular in this area.

The main contributions of this work are:

- a method of establishing a homogenous *ensemble* using a *k-fold undersampling* of *majority class*,
- proposition of five *fusers* to generate *ensemble* decision,
- a *pruning* method adjusting the decision rule to the *testing set*,
- implementation and experimental evaluation of proposed method.

2. *Undersampled Majority Class Ensemble*

2.1. Establishing ensemble

Complex oversampling methods, such as SMOTE or ADASYN, despite the large possibilities in most of the problems in imbalanced domain, are not applicable to extreme situations where the *minority class* is represented by only a few samples, which makes it impossible to designate the nearest neighbors to create a new synthetic object. This could lead to the use of *undersampling* in such problems, but it is characterized, due to high randomness, by a strong instability in a situation of high IR (*imbalance ratio*), which does not allow for the development of a reliable solution.

A popular answer to the above-mentioned problem are the *ensemble* methods of *Bagging* or *Boosting*, characterized by random sampling with replacement of the training set, breaking a large problem, into a set of smaller ones. This work proposes a basic method, which also breaks the imbalanced task, but with ensuring the use of all the patterns available in the data set, but without a risk of overlapping. Its description may be found in Algorithm 1.

Algorithm 1: Training classifier ensemble from multiple balanced training datasets separated from one imbalanced dataset of binary problem
Given a dataset DS:

1. Divide DS into subsets of minority- $MinC$ and majority-class $MajC$
2. Calculate imbalanced ratio IR as the proportion of the number of patterns in $MinC$ and $MajC$
3. Establish k by rounding IR to nearest integer
4. Perform a *shuffled k-fold division* of $MajC$ to produce a set of subsets $MajC_1, MajC_2, \ldots, MajC_k$
5. For every i in range to k
    6. Join $MajC_i$ with $MinC$ to prepare a training set TS_i,
    7. Train classifier Ψ_i on TS_i and add it into ensemble

After dividing the dataset with imbalanced binary problem into separated minority ($MinC$) and majority class ($MajC$), we are calculating the IR (*imbalanced ratio*) between given classes. Rounding IR to the nearest integer value k allows us to find the optimal division coefficient of the majority class samples in the context of maximizing the balance between the $MinC$ and any $MajC_i$ subsets while ensuring that all $MajC$ patterns are used in learning process with no overlapping between the individual $MajC_i$'s. Each of k classifiers Ψ_i is trained on union of $MinC$ and $MajC_i$ sets.

Extending pool with oversampling As an extension of the method of classifier ensemble construction, it is also proposed to expand its pool by a model learned on an additional data set, which is a full set of data subjected to *oversampling*. It is worth testing if the knowledge gained from this method may be a valuable contribution to the ensemble decision. Due to impossibility of using SMOTE or ADASYN for oversampling the minority class with only few instances, only its basic variant will be employed.

2.2. Fuser design

In addition to ensuring the diversity of the classifiers pool, which we achieve by a homogenous committee built on disjoint subsets of the majority class supplemented by minority patterns, the key aspect of the hybrid classification system is the appropriate design of its *fuser* – the element responsible for making decisions based on the answers of the base classifiers.

There are two groups of solutions here. The first are based on component *decisions* of the committee, most often employing the *majority voting* to produce a final decision. The decision rules proposed in this work are, however, part of the second group, where the *fuser* is carried out by *averaging* (or *accumulating*) the *support vectors* received from the members of a pool. It should be remembered that in such methods, it is necessary to use a *probabilistic classification model*, which also requires *quantitative* and not *qualitative* *data*, so we need to reject classification algorithms such as *Support Vector Machines*, whose probabilistic interpretation becomes reliable only in cases of large training sets.

Five accumulative fusers were proposed to analyze:

1. **REG** — regular accumulation of support.

   A basic method without weighing the members of a committee.

2. **WEI** — accumulation weighted after members of a committee.

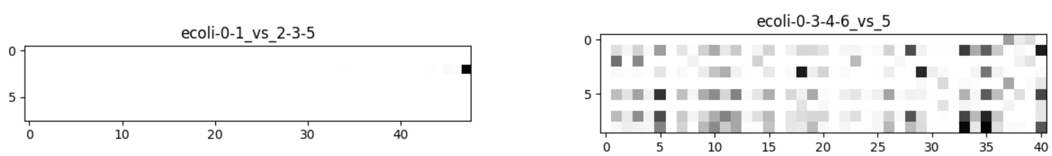
   The weight of the classifier in the pool is its quality achieved for the training set. We can not use here the measure of *accuracy*, which does not fit with the task of the imbalanced classification, so a *balanced accuracy* was chosen (Brodersen et al., 2010).

3. **NOR** — same as **WEI**, but with normalization of weights,

   To reward classifiers with a higher *discriminative power*, weights are subjected to normalization by a *MinMaxScaler*.

4. **CON** — accumulation weighted by tested patterns.

   In order to reward classifiers with greater *"certainty"* for given object, the decision for each pattern is weighted by the absolute difference between class support, for the needs of research called the *contrast*. Individual classifiers in the pool do not have to be better or worse for each of the tested patterns. This is illustrated in Figure 3, where we can see two cases of ensembles. There are tested patterns on the X axis and classifiers in the pool on the Y axis. A white square means the *contrast* of 1, and therefore a *sure* decision, and the black square the *contrast* of 0, which describes the pattern that is exactly on the decision boundary.



(a) Example of a *"sure"* ensemble (b) Example of *"unsure"* ensemble

Figure 3: Illustration of the *contrast* in committees built on two different datasets.

5. **NCI** — accumulation weighted by a product of normalized weights and a *contrast*.

The proposed method of constructing the committee makes its size directly dependent on the IR, which, given the highly unbalanced data (for example with IR greater than 40), leads to the construction of an extensive hybrid model. Therefore, the method of prunning it to a smaller size was also considered.

2.3. Ensemble pruning

Typical methods of *ensemble pruning* follow the phase of training the committee, for example, by eliminating the classifiers that achieve the lowest quality on the *training* or separated *validation set*. This paper proposes a method of *response pruning* based on the assumption that during the testing phase we analyze not just a single test pattern, but the entire *testing set*.

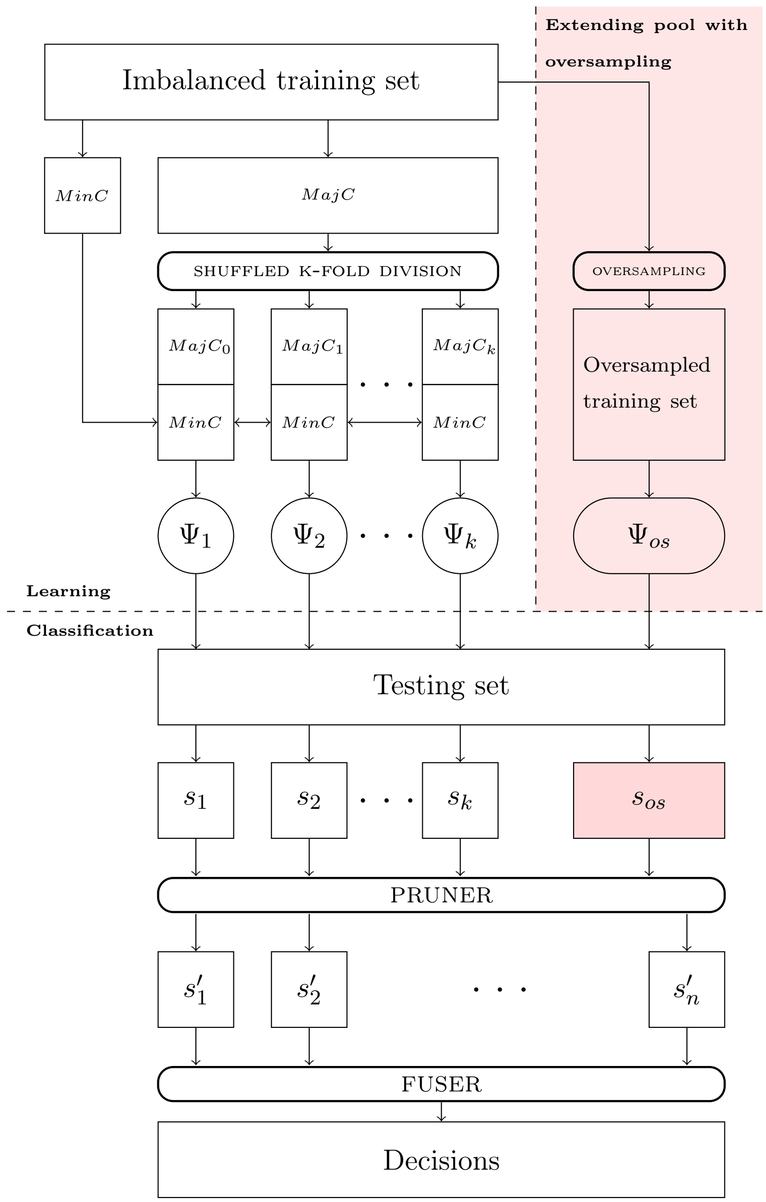


Figure 4: Diagram of *Undersampled Majority Class Ensemble* structure

Ensemble, receiving a *testing set*, generates *support vectors* (s_i) for each classified object, so, with a binary problem, we can treat received support for one of the problem classes as values from the *random variables* to analyze their mutual statistical dependence.

In the proposed method, using the signed-rank test, we are *clustering* the pool of k (or $k+1$ on the *oversampling* variation of a method) classifiers to n groups (where $n \leq k$), to average the support and weight classes within groups to create a new set of supports from s'_1 to s'_n, passed later on to *fuser*. It is important to denote, that in the considered case of pruning, we ignore the possible situation in which the answer Ψ_1 is dependent on Ψ_2, the answer Ψ_2 is dependent on Ψ_3, but Ψ_1 is not dependent on Ψ_3. This is an interesting issue that will be addressed in future research, but to clarify the proposal, a simplified approach has been used.

The scheme of the full decision model of the proposed method is shown in Figure 4.

3. Experiment design

For the experimental evaluation of the proposed method, a collection of datasets made available with KEEL (Alcalá-Fdez et al., 2011) was used, focusing on a section containing highly unbalanced data, with IR greater than 9 (Fernández et al., 2009). From among the available datasets, 40 were selected, presenting only binary problems with quantitative attributes. A review of selected datasets, including information on their number of features, the number of patterns in each class and the imbalance ratio is presented in Table 1.

DS	Features	Samples ALL	Samples MAJ	Samples MIN	IR
ecoli-0-1-3-7-vs-2-6	7	281	274	7	39.14
ecoli4	7	336	316	20	15.80
glass-0-1-6-vs-2	9	192	175	17	10.29
glass-0-1-6-vs-5	9	184	175	9	19.44
glass2	9	214	197	17	11.59
glass4	9	214	201	13	15.46
glass5	9	214	205	9	22.78
page-blocks-1-3-vs-4	10	472	444	28	15.86
shuttle-c0-vs-c4	9	1829	1706	123	13.87
shuttle-c2-vs-c4	9	129	123	6	20.50
vowel0	13	988	898	90	9.98
yeast-0-5-6-7-9-vs-4	8	528	477	51	9.35
yeast-1-2-8-9-vs-7	8	947	917	30	30.57
yeast-1-4-5-8-vs-7	8	693	663	30	22.10
yeast-1-vs-7	7	459	429	30	14.30
yeast-2-vs-4	8	514	463	51	9.08
yeast-2-vs-8	8	482	462	20	23.10
yeast4	8	1484	1433	51	28.10
yeast5	8	1484	1440	44	32.73
yeast6	8	1484	1449	35	41.40
ecoli-0-1-4-6-vs-5	6	280	260	20	13.00
ecoli-0-1-4-7-vs-2-3-5-6	7	336	307	29	10.59
ecoli-0-1-4-7-vs-5-6	6	332	307	25	12.28
ecoli-0-1-vs-2-3-5	7	244	220	24	9.17
ecoli-0-1-vs-5	6	240	220	20	11.00
ecoli-0-2-3-4-vs-5	7	202	182	20	9.10
ecoli-0-2-6-7-vs-3-5	7	224	202	22	9.18
ecoli-0-3-4-6-vs-5	7	205	185	20	9.25
ecoli-0-3-4-7-vs-5-6	7	257	232	25	9.28
ecoli-0-3-4-vs-5	7	200	180	20	9.00
ecoli-0-4-6-vs-5	6	203	183	20	9.15
ecoli-0-6-7-vs-3-5	7	222	200	22	9.09
ecoli-0-6-7-vs-5	6	220	200	20	10.00
glass-0-1-4-6-vs-2	9	205	188	17	11.06
glass-0-1-5-vs-2	9	172	155	17	9.12
glass-0-4-vs-5	9	92	83	9	9.22
glass-0-6-vs-5	9	108	99	9	11.00
yeast-0-2-5-6-vs-3-7-8-9	8	1004	905	99	9.14
yeast-0-2-5-7-9-vs-3-6-8	8	1004	905	99	9.14
yeast-0-3-5-9-vs-7-8	8	506	456	50	9.12

Table 1: Summary of imbalanced datasets chosen for evaluation

As may be observed in the summary, the experiments are based on datasets with relatively small spatiality (up to 13 dimensions), with imbalance ratio from 9 to even 40. The datasets provided by KEEL, to ensure easy comparison between results presented in various research, are already pre-divided into five parts, which forces the use of *k-fold cross-validation* with 5 folds in experiments (Alpaydin, 2009).

In the task of imbalanced data classification, due to its strong bias towards majority class, the *accuracy* measure is not a proper tool. For a reliable result, a measure of *balanced accuracy* is given as test results.

Both the implementation of the proposed method and the experimental environment have been constructed using the *scikit-learn* library (Pedregosa et al., 2011) in version *0.20.dev0*[2]. Among the available classification models, the MLP (*Multilayer Perceptron*) and SVC (*Support Vector Machine*) were rejected. First one was not able to build a correct model due to the lack of convergence on the small datasets (minority class of data chosen for experiments is often represented by only two patterns in cross-validated folds) and second one, whose probabilistic interpretation is measurable only with sufficiently large data sets, did not allow credible construction of a fuser. As base classifiers, the following algorithms were used:

- *Gaussian Naive Bayes* (GNB) (Chan et al., 1982),
- *Decision Tree Classifier* (DTC) — with *Gini* criterion (Loh, 2011).

To provide a comparative result for the method presented in the following paper, each base classifier was also tested for (*i*) the raw, imbalanced dataset and its (*ii*) under- and (*iii*) oversampled versions. Undersampling, due to high instability of results, was repeated five times on each fold. Used statistical analysis tool was a paired dependency between the classifier, which achieved the highest result and each of the others, calculated using the signed-rank *Wilcoxon* test (Wilcoxon, 1945).

The full implementation of the proposed method, content of the following paper and the script allowing to reconstruct the presented research may be found in the *git* repository[3].

4. Experimental evaluation

The results of the conducted research, for individual base classifiers, are presented in Tables 2 and 3. They were divided to present in individual sections a *balanced accuracy* achieved by particular variations of the method proposed in the following paper. In the first division stage, we show the impact of inclusion of the classifier built on the *oversampled* dataset, in the second, the use of the proposed *pruning* method, and in the third – employed *fuser*. It gave the number of 20 algorithm variations.

The presented results were supplemented by a balanced accuracy achieved by the classifier built on a full, *imbalanced dataset* (**Full**), a set after *undersampling* (**US**) and an *oversampling* (**OS**). The table cells marked in green indicate the best result for a dataset or the result statistically dependent on it, calculated in accordance with previously described assumptions of the experiments.

As we can see in Table 2, which presents the quality of classification using the GNB algorithm, there were only two datasets, where the lone best solution was to train the

2. At the time of conducting research, only the development version of the package already has the implementation of *balanced accuracy* measure.
3. https://github.com/w4k2/umce

Without oversampled set										With oversampled set													
Without pruning					With pruning					Without pruning					With pruning					OS	US	Full	Dataset
REG	WEI	CON	NOR	NC	REG	WEI	CON	NOR	NC	REG	WEI	CON	NOR	NC	REG	WEI	CON	NOR	NC				
.843	.843	.843	.834	.839	.845	.845	.845	.834	.837	.830	.828	.828	.817	.817	.845	.845	.845	.834	.834	.807	.835	.825	*ecoli-0-1-3-7-vs-2-6*
.685	.735	.785	.807	.807	.717	.767	.792	.875	.896	.779	.823	.841	.910	.896	.740	.792	.842	.898	.918	.860	.765	.878	*ecoli4*
.613	.613	.616	.582	.582	.580	.580	.574	.616	.588	.585	.582	.585	.580	.580	.577	.577	.574	.613	.585	.577	.574	.580	*glass-0-1-6-vs-2*
.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.989	.941	.967	.941	*glass-0-1-6-vs-5*
.641	.641	.644	.641	.641	.641	.641	.641	.641	.641	.619	.616	.619	.641	.641	.641	.641	.641	.641	.641	.610	.629	.591	*glass2*
.731	.731	.781	.779	.781	.776	.776	.776	.774	.771	.731	.731	.781	.788	.789	.779	.776	.779	.776	.776	.731	.728	.587	*glass4*
.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.938	.943	.938	*glass5*
.818	.818	.818	.817	.818	.831	.831	.831	.821	.821	.821	.821	.819	.817	.818	.832	.832	.831	.823	.823	.789	.816	.763	*page-blocks-1-3-vs-4*
.995	.995	.995	.994	.994	.994	.994	.994	.994	.994	.995	.995	.995	.991	.991	.994	.994	.994	.991	.991	.990	.994	.991	*shuttle-c0-vs-c4*
.959	.959	.955	.963	.959	.984	.988	.988	.992	.992	.959	.959	.959	.980	.976	.988	.988	.988	.992	.992	.986	.950	.996	*shuttle-c2-vs-c4*
.909	.909	.909	.909	.909	.909	.909	.909	.909	.909	.909	.909	.909	.911	.911	.909	.909	.910	.910	.909	.906	.906	.917	*vowel0*
.650	.680	.650	.751	.727	.733	.747	.737	.761	.767	.673	.704	.674	.759	.733	.734	.751	.748	.761	.767	.498	.620	.504	*yeast-0-5-6-7-9-vs-4*
.614	.601	.614	.624	.619	.570	.589	.588	.636	.606	.570	.583	.578	.613	.614	.561	.581	.581	.636	.620	.540	.588	.544	*yeast-1-2-8-9-vs-7*
.610	.598	.604	.628	.606	.553	.569	.561	.572	.558	.586	.601	.594	.627	.605	.581	.558	.552	.588	.556	.541	.570	.547	*yeast-1-4-5-8-vs-7*
.714	.714	.717	.721	.718	.689	.696	.683	.713	.705	.694	.701	.653	.721	.718	.696	.682	.677	.713	.714	.588	.699	.604	*yeast-1-vs-7*
.785	.795	.785	.824	.815	.807	.807	.815	.838	.841	.818	.823	.817	.824	.813	.800	.824	.840	.838	.841	.533	.733	.561	*yeast-2-vs-4*
.773	.773	.773	.773	.773	.773	.773	.773	.773	.773	.796	.798	.797	.773	.773	.773	.773	.773	.773	.773	.614	.775	.657	*yeast-2-vs-8*
.618	.643	.616	.792	.781	.699	.785	.768	.819	.809	.744	.752	.722	.797	.770	.722	.790	.769	.818	.808	.526	.645	.551	*yeast4*
.936	.936	.936	.936	.936	.931	.935	.937	.950	.951	.895	.911	.906	.936	.935	.928	.933	.934	.949	.950	.782	.918	.831	*yeast5*
.699	.708	.704	.828	.767	.832	.860	.845	.886	.878	.675	.681	.695	.820	.736	.821	.858	.841	.886	.878	.628	.779	.650	*yeast6*
.894	.900	.898	.898	.902	.687	.763	.763	.910	.910	.896	.896	.896	.898	.890	.712	.863	.863	.912	.912	.883	.672	.877	*ecoli-0-1-4-6-vs-5*
.630	.630	.630	.630	.630	.630	.630	.630	.662	.663	.630	.630	.630	.663	.663	.630	.630	.630	.662	.663	.663	.638	.630	*ecoli-0-1-4-7-vs-2-3-5-6*
.617	.617	.637	.710	.732	.617	.617	.617	.838	.811	.757	.813	.835	.852	.839	.617	.617	.617	.846	.846	.860	.638	.735	*ecoli-0-1-4-7-vs-5-6*
.558	.558	.558	.598	.558	.558	.558	.558	.698	.698	.618	.618	.638	.675	.638	.558	.578	.578	.718	.718	.639	.570	.638	*ecoli-0-1-vs-2-3-5*
.666	.691	.691	.691	.691	.616	.641	.641	.716	.741	.716	.741	.816	.814	.814	.641	.666	.664	.789	.814	.800	.662	.782	*ecoli-0-1-vs-5*
.673	.678	.675	.686	.681	.653	.681	.706	.825	.848	.728	.728	.728	.722	.722	.672	.728	.728	.870	.870	.638	.674	.754	*ecoli-0-2-3-4-vs-5*
.585	.605	.605	.593	.593	.563	.608	.588	.643	.646	.605	.603	.605	.591	.595	.588	.608	.610	.648	.648	.588	.588	.563	*ecoli-0-2-6-7-vs-3-5*
.870	.870	.870	.901	.895	.898	.898	.898	.895	.895	.890	.887	.884	.882	.874	.898	.898	.895	.898	.898	.704	.736	.784	*ecoli-0-3-4-6-vs-5*
.633	.633	.633	.656	.659	.613	.613	.613	.801	.786	.675	.709	.707	.696	.696	.613	.633	.633	.793	.789	.728	.695	.775	*ecoli-0-3-4-7-vs-5-6*
.808	.833	.833	.867	.872	.714	.739	.736	.900	.900	.819	.839	.817	.844	.822	.733	.786	.783	.903	.903	.738	.647	.817	*ecoli-0-3-4-vs-5*
.773	.776	.773	.801	.801	.731	.778	.781	.903	.903	.792	.795	.828	.901	.901	.778	.778	.801	.903	.901	.894	.694	.854	*ecoli-0-4-6-vs-5*
.603	.603	.603	.585	.605	.548	.508	.508	.605	.603	.603	.603	.603	.603	.603	.548	.508	.508	.603	.603	.548	.567	.508	*ecoli-0-6-7-vs-3-5*
.747	.757	.738	.747	.750	.615	.710	.715	.865	.838	.812	.832	.863	.853	.853	.637	.760	.760	.857	.863	.851	.664	.780	*ecoli-0-6-7-vs-5*
.656	.623	.656	.630	.630	.620	.620	.620	.620	.620	.592	.625	.592	.630	.597	.586	.620	.620	.620	.620	.599	.615	.577	*glass-0-1-4-6-vs-2*
.600	.587	.594	.605	.616	.519	.533	.539	.626	.642	.600	.619	.612	.548	.548	.536	.523	.523	.574	.580	.518	.515	.519	*glass-0-1-5-vs-2*
.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.994	.984	.994	*glass-0-4-vs-5*
.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.995	.950	.983	.945	*glass-0-6-vs-5*
.581	.576	.576	.576	.576	.576	.576	.576	.788	.784	.689	.707	.711	.775	.776	.576	.576	.576	.790	.786	.782	.596	.670	*yeast-0-2-5-6-vs-3-7-8-9*
.895	.895	.895	.895	.895	.896	.898	.897	.896	.895	.803	.837	.817	.895	.899	.902	.900	.896	.902	.899	.525	.741	.577	*yeast-0-2-5-7-9-vs-3-6-8*
.606	.606	.606	.616	.616	.607	.607	.607	.615	.607	.650	.635	.637	.616	.625	.623	.612	.607	.615	.607	.537	.633	.557	*yeast-0-3-5-9-vs-7-8*

Table 2: Balanced accuracy scores obtained using GNB as a base classifier

Without oversampled set										With oversampled set													
Without pruning					With pruning					Without pruning					With pruning					OS	US	Full	Dataset
REG	WEI	CON	NOR	NC	REG	WEI	CON	NOR	NC	REG	WEI	CON	NOR	NC	REG	WEI	CON	NOR	NC				
.705	.805	.703	.811	.811	.809	.809	.809	.823	.823	.714	.716	.718	.747	.653	.809	.816	.816	.827	.827	.639	.772	.739	*ecoli-0-1-3-7-vs-2-6*
.865	.865	.866	.903	.903	.878	.878	.878	.878	.878	.849	.849	.851	.835	.835	.878	.878	.878	.859	.859	.817	.858	.861	*ecoli4*
.785	.785	.782	.713	.760	.794	.788	.788	.846	.846	.777	.716	.722	.654	.599	.783	.805	.805	.761	.761	.575	.656	.552	*glass-0-1-6-vs-2*
.931	.931	.931	.934	.937	.934	.934	.934	.940	.940	.934	.934	.934	.954	.954	.934	.934	.934	.946	.946	.869	.898	.886	*glass-0-1-6-vs-5*
.779	.779	.785	.792	.792	.833	.828	.828	.766	.766	.802	.805	.807	.670	.677	.782	.782	.782	.703	.703	.630	.654	.609	*glass2*
.880	.880	.873	.895	.885	.903	.903	.903	.908	.908	.895	.900	.900	.915	.918	.903	.903	.903	.913	.913	.815	.829	.799	*glass4*
.934	.934	.937	.949	.949	.934	.934	.934	.951	.951	.946	.949	.949	.956	.963	.939	.939	.939	.956	.956	.933	.873	.898	*glass5*
.987	.987	.988	.991	.990	.990	.990	.990	.990	.990	.990	.990	.989	.994	.994	.990	.990	.990	.991	.991	.998	.959	.996	*page-blocks-1-3-vs-4*
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	.000	1	*shuttle-c0-vs-c4*
1	1	1	1	1	1	1	1	1	1	1	.950	.950	.950	.950	1	1	1	1	1	.995	.965	.950	*shuttle-c2-vs-c4*
.964	.964	.964	.965	.965	.966	.964	.964	.964	.964	.958	.958	.958	.956	.956	.966	.966	.966	.968	.968	.925	.942	.943	*vowel0*
.770	.769	.769	.784	.782	.784	.782	.782	.787	.787	.772	.774	.777	.728	.715	.776	.771	.771	.753	.753	.660	.746	.648	*yeast-0-5-6-7-9-vs-4*
.685	.687	.665	.653	.651	.721	.721	.721	.714	.714	.685	.679	.641	.602	.598	.740	.737	.737	.726	.726	.623	.631	.631	*yeast-1-2-8-9-vs-7*
.596	.598	.599	.601	.601	.619	.605	.605	.610	.610	.591	.599	.576	.540	.549	.618	.618	.618	.608	.608	.541	.569	.533	*yeast-1-4-5-8-vs-7*
.697	.698	.697	.691	.706	.701	.690	.690	.680	.680	.726	.731	.735	.710	.696	.717	.717	.717	.718	.718	.603	.692	.651	*yeast-1-vs-7*
.951	.951	.951	.958	.958	.943	.943	.943	.957	.957	.937	.937	.937	.901	.902	.937	.937	.937	.918	.918	.831	.904	.843	*yeast-2-vs-4*
.771	.772	.795	.775	.770	.817	.817	.817	.748	.748	.801	.808	.817	.716	.716	.760	.755	.755	.770	.770	.692	.726	.690	*yeast-2-vs-8*
.853	.853	.842	.854	.872	.854	.852	.852	.858	.858	.840	.833	.818	.710	.684	.845	.845	.845	.862	.862	.621	.805	.664	*yeast4*
.961	.961	.959	.960	.960	.965	.965	.965	.966	.966	.966	.966	.955	.962	.963	.967	.966	.966	.968	.968	.850	.932	.843	*yeast5*
.858	.858	.858	.865	.862	.847	.847	.847	.852	.852	.872	.863	.855	.765	.768	.852	.850	.850	.855	.855	.751	.813	.746	*yeast6*
.902	.902	.900	.904	.904	.883	.883	.883	.890	.890	.860	.860	.858	.842	.819	.867	.863	.863	.867	.867	.800	.813	.775	*ecoli-0-1-4-6-vs-5*
.847	.847	.848	.853	.865	.845	.842	.842	.833	.833	.849	.848	.851	.874	.875	.845	.843	.843	.854	.854	.795	.805	.814	*ecoli-0-1-4-7-vs-2-3-5-6*
.883	.881	.860	.856	.856	.869	.883	.883	.886	.886	.869	.869	.869	.859	.866	.869	.869	.869	.862	.862	.853	.801	.865	*ecoli-0-1-4-7-vs-5-6*
.820	.820	.842	.822	.822	.837	.837	.837	.797	.797	.804	.804	.804	.771	.773	.804	.804	.804	.786	.786	.745	.808	.762	*ecoli-0-1-vs-2-3-5*
.816	.816	.816	.816	.816	.825	.825	.825	.836	.836	.830	.830	.832	.843	.845	.839	.834	.834	.841	.841	.810	.826	.861	*ecoli-0-1-vs-5*
.834	.834	.834	.814	.811	.817	.817	.817	.847	.847	.856	.856	.856	.839	.842	.828	.828	.828	.836	.836	.828	.847	.808	*ecoli-0-2-3-4-vs-5*
.815	.815	.815	.820	.820	.820	.818	.818	.823	.823	.835	.835	.835	.803	.803	.820	.820	.820	.840	.840	.800	.797	.795	*ecoli-0-2-6-7-vs-3-5*
.907	.907	.904	.929	.932	.901	.901	.901	.907	.907	.915	.915	.915	.878	.881	.923	.915	.915	.893	.893	.810	.846	.781	*ecoli-0-3-4-6-vs-5*
.886	.886	.886	.884	.884	.884	.884	.884	.890	.890	.881	.881	.881	.894	.896	.885	.883	.883	.892	.892	.823	.812	.818	*ecoli-0-3-4-7-vs-5-6*
.928	.928	.928	.928	.928	.931	.931	.931	.908	.908	.917	.917	.917	.897	.897	.917	.917	.917	.897	.897	.854	.864	.856	*ecoli-0-3-4-vs-5*
.901	.901	.881	.862	.859	.906	.906	.906	.890	.890	.895	.895	.901	.861	.861	.901	.901	.901	.884	.884	.825	.858	.834	*ecoli-0-4-6-vs-5*
.828	.828	.828	.835	.832	.838	.838	.838	.845	.845	.850	.850	.853	.853	.853	.857	.857	.857	.863	.863	.847	.805	.828	*ecoli-0-6-7-vs-3-5*
.865	.865	.865	.878	.878	.875	.873	.873	.885	.885	.872	.872	.872	.870	.875	.875	.875	.875	.895	.895	.826	.815	.770	*ecoli-0-6-7-vs-5*
.810	.807	.804	.810	.810	.829	.829	.829	.747	.747	.814	.816	.808	.686	.692	.778	.778	.778	.759	.759	.659	.674	.615	*glass-0-1-4-6-vs-2*
.777	.777	.777	.783	.783	.743	.786	.786	.782	.782	.738	.738	.738	.574	.534	.738	.735	.735	.653	.653	.566	.638	.570	*glass-0-1-5-vs-2*
.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.982	.994	.946	.994	*glass-0-4-vs-5*
.929	.929	.939	.934	.944	.934	.934	.934	.965	.965	.949	.949	.954	.970	.970	.965	.960	.960	.965	.965	.960	.880	.940	*glass-0-6-vs-5*
.759	.759	.763	.758	.762	.767	.767	.767	.777	.777	.779	.780	.782	.749	.725	.782	.780	.780	.782	.782	.705	.717	.742	*yeast-0-2-5-6-vs-3-7-8-9*
.901	.900	.899	.894	.894	.893	.893	.893	.894	.894	.907	.907	.907	.890	.893	.907	.907	.907	.902	.902	.861	.862	.850	*yeast-0-2-5-7-9-vs-3-6-8*
.694	.694	.694	.717	.701	.714	.714	.714	.725	.725	.707	.714	.700	.598	.593	.717	.710	.710	.658	.658	.586	.642	.670	*yeast-0-3-5-9-vs-7-8*

Table 3: Balanced accuracy scores obtained using DTC as a base classifier

model on a full, imbalanced dataset, and one where the best solution were simple *over-* or *undersampling*. In the Table 3, showing the results for the DTC classifier, we are dealing with a similar situation in which, however, *undersampling* never turns out to be the best in the tested pool of solutions.

A clearer interpretation of the results may take place after the analysis of the Table 4, showing a summary of the results achieved by individual variations of the proposed method, presenting the number of datasets for which a given variation took part in the construction of the best solution.

Classifier	Full	US	OS	OSE		Pru.		Fuser				
				NO	YES	NO	YES	REG	WEI	CON	NOR	NCI
GNB	3	1	1	10	**12**	6	**12**	6	5	6	11	**12**
DTC	3	0	2	7	**8**	7	**8**	7	6	6	**8**	**8**

Table 4: Final summary of proposed method variations.
(OSE – extending pool by oversampled dataset, Pru. – usage of pruning)

As we may observe, both the extension of the classifier pool by the model built on the oversampled dataset as well as the proposed pruning method has a positive impact on the quality of the final solution. Among the fusers, the best performers are NOR – normalizing the calculated weights for the members of the committee and NCI - complementing NOR by the accumulated support with a stronger impact of the certainty of the decision. Even just the basic ensemble construction, in its simplest form without improvements and using the decision rule without weighting, allows to achieve better results than learning on a full dataset or basic under- or oversampling.

5. Conclusions

This paper presents UMCE (*Undersampled Majority Class Ensemble*) – a hybrid method for solving the problem of binary classification of datasets with a high *imbalance ratio*, based on *k-fold division* of the *majority class* samples to create an *ensemble* of classifiers breaking one *imbalanced problem* into many balanced problems. The basic division method has been supplemented with a variant extending the pool with the *oversampled* dataset and the *post-pruning* method based on the analysis of the statistical dependencies of the classifiers response on the testing set. For the *ensemble* it were also proposed five different *fusers*.

Computer experiments have shown, that this approach led to create a method solving targeted problem and able to outperform other possible basic solutions, proving that it may be employed for real-life appliance.

Acknowledgments

This work was supported by the Polish National Science Center under the grant no. UMO-2015/19/B/ST6/01597 and by the statutory fund of the Faculty of Electronics, Wroclaw University of Science and Technology.

References

Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011.

Ethem Alpaydin. *Introduction to machine learning.* MIT press, 2009.

Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M Buhmann. The balanced accuracy and its posterior distribution. In *Pattern recognition (ICPR), 2010 20th international conference on*, pages 3121–3124. IEEE, 2010.

C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. Safe-Level-SMOTE: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining, 13th Pacific-Asia Conference 2009, Bangkok, Thailand, April 27-30, 2009, Proceedings*, pages 475–482, 2009.

T. F. Chan, G. H. Golub, and R. J. LeVeque. Updating formulae and a pairwise algorithm for computing sample variances. In H. Caussinus, P. Ettinger, and R. Tomassone, editors, *COMPSTAT 1982 5th Symposium held at Toulouse 1982*, pages 30–41, Heidelberg, 1982. Physica-Verlag HD. ISBN 978-3-642-51461-6.

N V Chawla, K W Bowyer, L O Hall, and W P Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *arXiv.org*, June 2011.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

Nitesh V. Chawla, Aleksandar Lazarevic, Lawrence O. Hall, and Kevin W. Bowyer. *SMOTE-Boost: Improving Prediction of the Minority Class in Boosting*, pages 107–119. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. ISBN 978-3-540-39804-2. doi: 10.1007/978-3-540-39804-2_12. URL https://doi.org/10.1007/978-3-540-39804-2_12.

Xue-wen Chen and Michael Wasikowski. Fast: A ROC-based feature selection metric for small samples and imbalanced data classification problems. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 124–132, 2008.

William Elazmeh, Nathalie Japkowicz, and Stan Matwin. Evaluating misclassifications in imbalanced data. In *Proceedings of the 17th European Conference on Machine Learning*, ECML'06, pages 126–137, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-45375-X, 978-3-540-45375-8. doi: 10.1007/11871842_16. URL http://dx.doi.org/10.1007/11871842_16.

Alberto Fernández, María José del Jesus, and Francisco Herrera. Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets. *International Journal of Approximate Reasoning*, 50(3):561–577, 2009.

M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42 (4):463–484, July 2012. ISSN 1094-6977. doi: 10.1109/TSMCC.2011.2161285.

H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

H. He, Y. Bai, E. A. Garcia, and S. Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the International Joint Conference on Neural Networks, 2008, part of the IEEE World Congress on Computational Intelligence, 2008, Hong Kong, China, June 1-6, 2008*, pages 1322–1328, 2008.

Nathalie Japkowicz, Catherine Myers, and Mark Gluck. A novelty detection approach to classification. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 518–523, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8, 978-1-558-60363-9. URL http://dl.acm.org/citation.cfm?id=1625855.1625923.

Bartosz Krawczyk, Michal Wozniak, and Boguslaw Cyganek. Clustering-based ensembles for one-class classification. *Information Sciences*, 264:182–195, 2014.

Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

V. Lopez, A. Fernandez, J. G. Moreno-Torres, and F. Herrera. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. *Expert Systems with Applications*, 39(7):6585–6608, 2012.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Y. Sun, A. K. C. Wong, and M. S. Kamel. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(4):687–719, 2009.

S. Wang, H. Chen, and X. Yao. Negative correlation learning for classification ensembles. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2010. doi: 10.1109/IJCNN.2010.5596702.

Shuo Wang, Leandro L. Minku, and Xin Yao. A systematic study of online class imbalance learning with concept drift. *CoRR*, abs/1703.06683, 2017. URL http://arxiv.org/abs/1703.06683.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6): 80–83, 1945.