An empirical evaluation of imbalanced data techniques from a practitioner's point of view

submitted to the special issue on Imbalanced Learning

Jacques Wainer, Rodrigo A. Franceschinelli

Abstract—This research tested the following well known techniques to deal with imbalanced data on 82 different real life data sets (sampled to imbalance rates of 5%, 3%, 1%, and 0.1%): class weight, SMOTE, Underbagging, and a baseline (just the base classifier). As base classifiers we used SVM with RBF kernel, random forests, and gradient boosting machines and we measured the quality of the resulting classifier using 5 different metrics (Area under the curve, Accuracy, F-measure, G-mean, and Matthew's correlation coefficient). The best technique strongly depends on the metric used to measure the quality of the classifier. For AUC and accuracy class weight and the baseline perform better; for F-measure and MCC SMOTE performs better; and for G-mean, Underbagging.

Index Terms—Imbalance classification; empirical comparisons; multiple metrics; strong base classifiers

1 Introduction

A class imbalance binary classification problem is a classification task where the training data (and one expect the test or future data) does not contain the same proportion of examples from both classes. The *imbalance rate* is the proportion of the minority class on the whole dataset. Although there is no accepted threshold below which one would say the problem is imbalanced, the published research usually deal with imbalance rates of 10% to 1%. Traditionally the minority class in an imbalanced problem is called the *positive class*, and the majority is the *negative*.

The problem with imbalanced datasets is that it is believed that usual classifiers will not perform well when training on data sets with imbalance - since classifiers are internally optimising some global metric, it is believed that the classifier may "swallow" the minority class in order to maximize the global metric only on the majority class. If internally the classifier is trying to minimize the sum of the square distance between the prediction and the correct value (as in a logistic regression) a reasonable minimal solution, if the imbalance is high enough is to disregard all minority data, and try to minimize the sum only for the majority class.

Thus, techniques or methodologies were created to improve the classification of imbalanced problems. In this paper we will call them *techniques*. Haibo and Yunqian [1] place the imbalanced data techniques into four main classes:

- **sampling** These techniques will add or remove data points from the training set so that it present a less skewed distribution to the base classifier. Among the sampling techniques: random oversampling [2], random undersampling [3], and synthetic sampling with data generation [4], [5]. In general terms, oversampling approaches will randomly repeat examples of the minority class to achieve a balanced data set to use as training; undersampling approaches will remove random majority class data until balance is achieved. Undersampling can be random, or informed, so that the remaining negative examples are very informative to define the decision frontier. Also undersampling may be associated with bagging - different versions of the original data set will be generated using random undersampling and each version will be used to train the classifier - a bagging of the resulting classifier instances will be the final classifier. Finally, synthetic data generation will generate new positive examples until the resulting data set is balanced.
- **cost sensitive** These techniques add a cost to the classification construction algorithm or optimisation so that errors have different costs, and thus the classifier would "try harder" to predict the minority class. A simple solution is to weight the data differently by classes, but the different classifier algorithms must be adapted to take the weight into account. A classifier independent approach is to incorporate the cost to a boosting approach [6], [7].
- **kernel based approaches**. There has been some research specific on modifying SVM for imbalance data [8], [9]. We will not discuss further these approaches.
- **one-class approaches**. One-class classifiers, such as one-class SVM [10] or auto-encoders can be used to learn the rare class [11]. We will not discuss further these approaches.

Another way of classifying the techniques is whether they are **internal** or **external** [2]. Internal techniques aim at changing the formulation of a particular classifier so that it can deal better with imbalanced data. Cost sensitive

- *Both authors are at Institute of Computing, University of Campinas, SP 13083-852 , Brazil.*
  E-mail: wainer@ic.unicamp.br, rodrigoaf61@gmail.com

approach such as class weight and kernel based approaches are internal.

External techniques assume an unmodified classifier which we will call a **base classifier**, and construct a larger classifier by usually combining different instances of the base classifier trained on different samples of the original data. All sampling approaches are external, as are the MetaCost [6] framework for a cost sensitive approach. Another family of external techniques is the boosting formulations for imbalanced data [7], [12], [13]. Some of these boosting techniques combine standard boosting ideas (data weight and vote weight) with sampling (under or over or a combination) to define the subset of data that will be used to train the following boost.

This research attempts to answer the question of what to do with an imbalanced data set from the point of view of a practitioner. We assume that a practitioner will prefer:

1) using powerful base classifiers, such as random forests, gradient boosting machines or SVM with RBF kernel. These three families were found to be the three best classifiers for a large set of real life problems [14], [15].
2) using already implemented external or internal approaches.
3) using some of the standard or most common metrics to measure the quality of the resulting classifier.

There has been some research that compare different techniques (as opposed to the majority of research that *propose* a new technique - and obviously compare it to some of the others). Three of those published research inform this research: Prati et al [16], Galar et al. [17], and López et al. [18] and they are discussed in detail in Section 1.1. One characteristic of these researches is that in general they use less powerful base classifiers, such as decision trees, kNN, naive Bayes, and so on. Two of them use SVMs among the classifiers tested, which is one of our classifiers. It is unclear if the results they report remain the same if more powerful classifiers are used.

It is possible that by using less powerful base classifiers, the researchers are exploring more clearly the differences between the compared techniques - if the classifier by itself cannot do much, most of the observed difference will be the effect of the techniques themselves. In this case, the goal of the research is to inform other scientists (as indeed they informed us) as to which techniques are probably more useful for further exploration and development. But those results may be misleading to a practitioner, who we believe, will start by using more powerful classifiers.

Regarding the second point above, we believe that a practitioner will prefer to use either already implemented techniques, or if none of them are available, the practitioner would prefer implementing the simpler techniques. Thus in this research, when previous empirical research has pointed out a set of techniques as "winning" techniques, we select from them the ones that are either widely available or are easier to implement. In particular, [17] empirically determined that SMOTEBagging and RUSBoost are good ensemble techniques, but we considered RUSBoost easier to implement and that was the one we incorporated into this research. Similarly, [18] lists SMOTE+ENN and SMOTE as the (empirically determined) best oversampling techniques, and we decided to test only SMOTE since there are many implementations available; they also list MetaCost and class weight as good cost based solutions, and we only tested class weight in this research.

Finally regarding the third point above, the similar published researches only use AUC as the quality metric. We do not know that a practitioner should necessarily use AUC as metric. As we discuss in Section 2.2 there are other metrics that can be used in situations of imbalanced data, and as this paper will show, the best techniques are **very** dependent on the metric used! So we will perform all comparisons using 5 different quality metrics (AUC, Accuracy, F-measure, G-mean, and Matthew's correlation coefficient), all of which have been argued in the literature as useful metrics. We will give a small emphasis on AUC, to compare our results with the literature, but results in all other 4 metrics will be displayed and discussed.

1.1 Related literature

This research follows and extends three previous researches with a similar goal of empirically testing different techniques for imbalanced data [16], [17], [18]. The main comparative characteristics of the three papers are listed below in Table 1.

Prati et al. [16] model the problem as how much does each technique recover the losses when an artificially imbalanced data set is created from a balanced one. Thus, the balanced AUC is the "correct" or "limit" AUC, and they measure how much each technique approaches this limit when dealing with versions of the data set with up to 1% imbalance. They show that SVM is the base classifier less sensitive to imbalance. They also show that in general the loss in relation to the "correct" AUC, is small (5%) for imbalance rates up to 10%. For more severe imbalances (up to 1%) there is a loss of about 20%. The research reports the detailed comparisons of the different techniques for each of 7 base classifiers, but they conclude that the effect of the different techniques is limited, only recovering on average 30% of the loss due to class imbalance.

Galar et al. [17] focus on ensemble based techniques to deal with imbalanced data. They compare 7 different proposals for cost-based boosting where the weight update rules of AdaBoost are changed to take into consideration the cost of making a minority class error; 4 different algorithms in the family of boosting based ensembles, where sampling techniques of adding or removing data for the training set of each boost are used; 4 different bagging-based ensembles, which use sampling and bagging; and 2 algorithms classified as hybrid. They use only C4.5 as the base classifier. They conclude that SMOTEBagging, RUSBoost, and UnderBagging are the best techniques in terms of the AUC values, and RUSBoost seems to be the less computationally complex solution.

Lopez et al. [18] compare a 7 synthetic minority sampling techniques; 3 cost sensitive learning; and 5 ensemble based techniques, using three base classifiers (kNN, C4.5, and SVM). They first compare the techniques within each family and then compare the best in each family to each other. Within the synthetic minority sampling they conclude

that SMOTE and SMOTE+ENN perform better. For the cost sensitive, class weight is the winning technique. And among the ensembles, RUSBoost, SMOTEBagging, and EasyEnsemble perform better. When comparing the winners among themselves and a no-techniques approach (which we call baseline in this paper), they concluded that in general class weight, SMOTE and SMOTE+ENN perform better than the other techniques and better than the baseline.

[18] then explore artificially altered data sets that make it more explicit the problems with imbalanced data. They discuss that the low number of the minority examples in data can bring about different problems, not always present in all data sets, and that different techniques may address one or more of these problems:

- small disjuncts
- low density on the positive region
- overlapping of classes
- impact of noisy data in imbalanced domains
- the problem of bordeline data
- the impact of differences between training and testing data for the positive class

The paper studies the effect of the different techniques on data sets that make each of these problems more salient.

2 METHODS

2.1 Techniques

In this research we tested the following techniques to deal with imbalanced data. These are techniques that the previous empirical research [16], [17], [18] has singled out as the "winning" techniques, and when there were more than one of the techniques within the same family we selected either the one with publicly available implementation or the simpler one to implement (according to our second assumption about the practitioner's preferences).

- **class weight**. This technique weights the positive examples with a higher value so that erring the positive examples is costlier than negative examples. This is an internal technique and so it depends on changing the base classifier appropriately. Fortunately, the R implementations of random forest, gradient boosting, and SVM already accept a class weight parameter. The same is true for the Python implementation of such algorithms. So this is a low cost alternative for the practitioner. In fact, the Scikit-learn implementation of the three base classifier algorithms assume by default that the class weight is set as we did in this paper.
  We fixed the class weight for the negative class as 1 and for the positive as the inverse of the imbalance rate, that is, for an imbalance rate of 1%, the positive class has a weight of 100, and the negative weight 1.
- **SMOTE** Synthetic minority over-sampling technique [4] is probably the best known method to generate new positive data. For each positive data point, the method will search among the k nearest neighbours for those that are also positive, and create a new data point (randomly placed) in the line segment that joins the original point and each of those positive neighbours.
  We used the SMOTE implementation of the R package MLR [19]. We generated new examples until the two classes were balanced and used $k = 5$ (number of nearest neighbours).
- **Underbagging** is the bagging of the base classifier, where each instance is trained on a balanced sample of the data set, where the positive class is preserved, and the negative class is under-sampled. We created n (a hyperparameter) of these balanced sets for each data set and trained the base classifier on them. The final classifier is the bagging of the n base classifiers.
- **baseline** is the null case - the use of a single base classifier, without any different weighting for the classes.
- **RUSBoost** Besides external techniques and class weight we will test a special purpose classifier that was well ranked in both [17] and [18]. RUSBoost [13] is a boosting technique that uses random undersampling (removing negative data) for training the different boosts, and ADABoost.M1 [21] for the weight update and voting weights. The weight updates for the examples are computed on the whole training set, not the undersampled set.
  RUSBoost as proposed by [13] is a general boosting technique, that is, it can be applied on any base classifier. For this comparison we are not using it as a general boosting technique (and apply it on our three strong base classifiers). We fixed the base classifier as the CART decision tree, and we use RUSBoost as a special purpose classifier for imbalanced data.
  We implemented our R version of RUSBoost based on the one available at https://github.com/SteveOhh/RUSBoost. The undersampling was set to achieve a 50% imbalance rate. The number of boosts $nboost$ is a hyperparameter.

2.2 Metrics

There has been a large body of discussion on how to assess the quality of classifiers on imbalance problems, and a large set of metrics proposals that address one or more of the problems.

The main problem with the standard metric, accuracy – the proportion of correct predictions – is that on very imbalanced problems, say 1% of imbalance, a "non-classifier" that always predict the negative class (the majority class), will be correct 99% of the time! But in many applications that uses imbalance data, the costs of wrongly predicting a positive and negative class are not the same: the cost of missing a positive case is much more costly than missing a negative case.

Different metrics have been proposed to "balance" the mistakes of the classifier on both the positive and negative classes [22].

In this paper we will focus on 5 metrics:

- **accuracy** abbreviated as **acc**
- **area under the ROC curve** abbreviated as **auc**
- **F-measure** abbreviated as **f1**
- **G-mean** [23] abbreviated as **gmean**
- **Matthew's correlation coefficient** [24] abbreviated as **mcc**

TABLE 1
Summary of the relevant literature

	Prati et al. [16]	Galar et al. [17]	Lopes et al. [18]
Number of data sets	22	44	66
Imbalance range	from 50% to 1% with steps between 5% and 10%	17 data sets with imbalance between 1% to 5% the rest between 10% and 35%	40% to 1% (mean 10%
Algorithms	C4.5, C4.5Rules, CN2, RIPPER, Neural networks, Naive Bayes, SVM	C4.5	C4.5, kNN, SVM
Techniques	**Sampling**: SMOTE, ADASYN, SMOTE BORDERLINE, Random Oversampling **Cost**: MetaCost	**Sampling**: SMOTE **Ensembles**: AdaBoost, AdaBoost.M1, AdaBoost.M2, Bagging, AdaC2, RUSBoost, SMOTEBoost, MSMOTEBoost, UnderBagging, OverBagging, UnderOverBagging, SMOTEBagging, MSMOTEBagging, IIVotes, EasyEnsemble, BalanceCascade	**Sampling**: SMOTE, SMOTE+ENN, borderline SMOTE, ADASYN, SL-SMOTE, SPIDER2, DBSMOTE **Cost**: Cl-Weight, MetaCost, CS-Classifier WEKA **Ensemble**: Adaboost.M1, AdaBoostC2, RUSBoost, SMOTEBagging, EasyEnsemble
Metric	AUC	AUC	AUC
"Winning" techniques	Random oversampling	SMOTEBagging, RUSBoost, UnderBagging	SMOTE, SMOTE+ENN Cl-Weight, MetaCost RUSBoost, SMOTEBagging

Let us denote TP as the number of true positives, that is, the correct positive predictions of the classifier; TN as the number of true negatives; FP is the number of false positives, that is data that the classifier predicted as positive which were in fact negative; and FN as the number of false negatives. Then:

$$\text{acc} = \frac{TP+TN}{TP+FP+TN+FN}$$
$$\text{precision} = \frac{TP}{TP+FP}$$
$$\text{recall} = \frac{TP}{TP+FN}$$
$$\text{specificity} = \frac{TN}{TN+FP}$$
$$\text{f1} = \frac{2}{\frac{1}{\text{precision}}+\frac{1}{\text{recall}}} = \frac{2TP}{2TP+FP+FN}$$
$$\text{gmean} = \sqrt{\text{recall}\times\text{specificity}} = \sqrt{\frac{TP}{TP+FN}\times\frac{TN}{TN+FP}}$$
$$\text{mcc} = \frac{TP\times TN - FP\times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

Accuracy is the proportion of correct predictions. Notice that TP+FN is the total number of positive examples, and thus recall is the accuracy for positive examples (the proportion of true predictions for the positive examples). TN+FP is the total number of negative examples, and thus specificity is the accuracy for the negative examples. Precision is the probability that the classifier is correct if it outputs the positive class (TP+FP are all the cases the classifier predicted as positive).

Recall is at tension with both recall and precision - it is possible to trade one for the other. f1 balances these two conflicting measures by taking the harmonic means of the metrics. Similarly, there is a tension between the accuracy of the positive and negative classes (recall and specificity). G-means balances these two conflicting metrics by taking the geometric mean of them.

We do not know of an intuitive interpretation of the mcc.

Area under the curve is not defined in terms of the false and true positives and negatives, but instead by the way each classifier ranks a set of examples - classifiers usually internally, but sometimes explicitly, compute a score value for each data, and we assume that higher scores are associated with the positive class. The usual way of interpreting AUC is by construction a receiver operating characteristic (ROC) curve , which is constructed by computing the recall and specificity of the classifier for different values of the score threshold above which the classifier decides that the example is positive. The area under that curve is the AUC.

A more intuitive definition is that the AUC is an estimate of the probability that a random positive example will have a higher score than a randomly selected negative example. Under this interpretation, assuming that $f(x)$ is the score for example x, X_+ is the set of positive examples, and X_- the set of negative examples, then

$$auc = \frac{1}{|X_+|\times|X_-|}\sum_{p\in X_+, n\in X_-} H(f(p)-f(n)) \quad (1)$$

where $H(x) = +1$ if $x > 0$, and 0 otherwise, is the Heaviside step function. The formula just counts the number of times the score of a positive example is higher than the score of a negative example, divided by the number of pairs.

There are other metrics proposed in the literature [22], [25]. In this paper we focus on the 5 metrics listed above: auc, acc, f1, gmean, and mcc. Accuracy and F-measure are

very commonly used metrics and thus we included in this research. In particular, although accuracy has been singled out as a problematic metric for imbalanced problems we believe that it is one of the metrics that should be evaluated. If there is no difference in error costs for the positive and negative classes then accuracy is the correct metric to be used in terms of selecting the classifier with lowest expected costs. Equal costs for errors may be a reasonable approximation when the original problem is a multiclass problem (where all classes have the same error costs) and one is using a one-versus-one or one-versus-all solution. If the original problem is binary then it is more probable that the costs of errors of misclassification are not the same for both classes, and thus accuracy is a less useful metric.

AUC is the metric used by three of the papers that inform this research [16], [17], [18]. Matthew correlation coefficient (mcc) has been argued as a good quality metric for imbalanced problems by recent publications [26], [27]. Finally, G-mean is a less common quality metric, but it has been used in some published research [20], [28], [29], [30].

2.3 Data and classifiers

In this paper we used the some of the UCI public data sets collected by [14]. For each data set:

- If the data set was multiclass, we selected as positive the class with the lowest frequency just above 5% or the class with highest frequency, if all frequencies were below 5%. All the other classes would be named as negative.
- Random positive examples were removed to achieve a 5%, 3%, 1% and 0.1% imbalance rates, provided there were at least 10 positive cases.
- If the original data set had already an imbalance below 5%, then random examples negative class was removed to achieve the 5% imbalance.

Table 2 displays the number of data sets for each imbalance level. At the lowest imbalance (5%) we started with 82 different data sets, but for the more severe levels (1% and 0.1%) we could use only produce 40 of them.

TABLE 2
Number of data sets for each imbalance rate.

Imbalance rate	number of datasets
0.1%	40
1.0%	40
3.0%	66
5.0%	82

As base classifiers we used:

- **random forest (rf)**. We used the R randomForest implementation. The hyperparameter range was: mtry $\in$ [1,number of dimensions] and ntree $\in [2^4, 2^{12}]$.
- **SVM with RBF kernel (svm)**. We used the R implementation. The hyperparameters were $C \in [2^{-5}, 2^{15}]$ and $\gamma \in [2^{-15}, 2^3]$
- **Extreme gradient boosting (xgb)**. We used the R implementation. Hyperparameters: max_depth $\in [1, 6]$, eta $\in [0.005, 0.05]$, nrounds $\in \{20, 40, 60, \ldots 140\}$

Besides the base classifiers, two other algorithms have hyperparameters:

- **RUSBoost** is treated in this paper as a special purpose classifier for imbalanced problems based on boosting of decision trees. The number of trees is a hyperparameter whose possible values are nboost $\in \{10, 20, 30, 40, 60\}$.
- **Underbagging** is a general technique that combines sampling and bagging and can be used with any base classifier. Underbagging also has a hyperparameter – the number of bags n which was selected from $n \in \{10, 20, 30, 40, 60\}$. The search for the best value of this hyperparameter is performed in parallel with the search for the base classifier's hyperparameter.

2.4 Experimental setup

Each data set for each imbalanced level we randomly selected 20% of the data set as test. The proportion of classes was the same in both the training and test set but we did not impose any other constraint on the partitioning (against the warnings of [31]) . We performed a 3-fold cross validation on the training set to select the best set of hyperparameters.

The hyperparameters were selected from 10 random samples from the ranges described above. The hyperparameters were selected based on each of the 5 metrics used.

Once the hyperparameters were selected, we trained the classifier with those hyperparameters using the whole training set, and measure it on the test set using the same metric used to select the hyperparameters. For the case of underbagging, all base classifiers in the ensemble used the same set of hyperparameters.

We repeated the process 3 times, that is, we have three measures of the quality of the classifier, which we average to have a better estimate.

Formally, we used a 3-times repeated, 20% holdout procedure with a nested 3-fold for selecting the hyperparameters.

We are interested in answering four questions for the practitioner, and thus we perform three comparisons (for each metric). We are also interested in answering one question to the scholars working on the problem of imbalance data.

- The first question is "what is in general the best technique for imbalance data (for each metric)?". This question is independent of the base classifier, the data set, and the imbalance rate, and thus these factors are marginalised out of the comparison: we compare the 5 techniques, for all combinations of base classifiers, imbalance rates, and data sets.
- The second questions is "what is the best technique for moderately imbalance data?" and "what is the best technique for severely imbalanced data?". In our experiment we consider the 5% imbalance as moderate, and the 0.1% imbalance as severe. Thus we compare the 5 techniques on all classifiers, and all data sets with imbalance of 5%, and the same for all data sets with imbalance of 0.1%.
- If the practitioner have control over which base classifier to use, we would like to answer the question

"What is the best combination of base classifier and technique?". Thus we compare all combinations of techniques and classifiers (5×3) for all data sets and all imbalance rates.

- Although we mentioned that we assume that a practitioner will prefer an out-of-the-box implementation, we believe that if a less known solution if clearly a good one, the practitioner would use it (and that would suggest that such technique should be soon placed "in-the-box"). The question we would like to answer is "How does the best combination of standard classifiers and techniques compare to the best special purpose solution?". In this case we follow some research [17], [18] that found that RUSBoost as one of the best ensemble techniques and we use a RUSBoost of decision trees as a special purpose solution for imbalanced problems. We compare the best combination of classifier and technique (discovered above) with a RUSBoost solution, for all data sets and all imbalance rates.
- Finally, the previous research in particular [16], determined that SVM is the base classifier less sensitive to imbalance of the data. But none of the previous research had evaluated random forests or gradient machines. Therefore we would like to answer the question "what is the best base classifiers?" fixed for the baseline technique, for all datasets and imbalance rates. We believe that this answer will be of value to scholars in the area.

2.5 Statistical analysis

We broadly follow Demsar's procedure [32] to compare multiple classifiers on multiple data sets but with some of the modifications as proposed by [33], [34]. In particular, instead of using the Nemenyi test as proposed by Demsar, we used the Friedman post hoc test, with multi-comparison adjustment of the p-value following Bergmann and Hommel [35] (we used the Holm procedure [36] for the case where more than 9 classifiers were compared). We used the R package `scmamp` [37] to perform the computations.

Demsar proposed a graphical representation of the resulting comparisons. We will use a more compact representation, using a table to both display the order of the different comparisons, and whether the differences are significative (with 95% confidence) or not [38]. The top left corner of Table 3 is an example of such table. The mean rank, which orders the different techniques is displayed in the second column. The third column shows the statistical significance: if two lines share a letter then the difference between the corresponding results is **not** statistically significant. In the case of first pane of Table 3, the difference between first and second lines are not statistically significant, but both of them have significant differences with the third and forth. The third and fourth are also not significantly different. The first and second panes of Table 9 in the appendix show more complex examples where the lines are not grouped into not-statistically-different blocks.

2.6 Reproducibility

The imbalanced datasets, the programs that run the experiments, the results of all experiments, and the R program to perform the analysis are available at https://figshare.com/s/96b3d7f8d3f74de4b6e3.

3 Results

3.1 General comparison

Table 3 reports the comparison of all techniques, for the 5 metrics. The second column "rank" is the mean rank of the technique across all databases (lower is better). The third column indicates the groups of not-statistically significant differences, as discussed in above. For AUC and accuracy, baseline and class weight perform equally well. For F-measure, SMOTE performs better than the alternatives. For G-mean, underbagging is the sole winner, and for MCC, all techniques but underbagging perform equally well.

TABLE 3
Comparisons of the techniques across all algorithms, and imbalance rates for all 5 metrics

algorithm	rank		algorithm	rank	
auc			acc		
baseline	2.29	a	baseline	1.95	a
weight	2.34	a	weight	1.96	a
smote	2.47	b	smote	2.41	b
underbagging	2.91	c	underbagging4	c	
f1			gmean		
smote	2.26	a	underbagging	1.46	a
weight	2.51	b	smote	2.57	b
baseline	2.52	b	baseline	2.97	c
underbagging	2.71	c	weight	3.00	c
mcc					
smote	2.37	a			
baseline	2.48	a			
weight	2.50	a			
underbagging	2.65	b			

3.2 Different imbalance rates

3.3 Best combination

For AUC, rf+baseline, rf+smote, rf+weight, xgb+baseline, xgb+weight, and xgb+smote are the best combinations, with no significant difference between them. For accuracy: xgb+baseline, xgb+weight, rf+weight, and rf+baseline are the best combinations.

For F-measure, xgb+smote is the best combination; for G-mean, rf+underbagging and xgb+underbagging. And for mcc, xgb+smote.

The full tables are in Appendix A.

3.4 Comparison to RUSBoost

Table 5 compares the best combination of base classifier and technique with the RUSBoost classifier. For each metric, the table list the mean rank and the name of the best combination, the mean rank of the RUSBoost, and the p-value of the Wilcoxon signed rank comparison. Note that for all metrics, all best combinations have a lower mean rank than RUSBoost, and that the difference is statistically significant (p-values below 5e-2).

TABLE 4
Comparisons of the techniques across all algorithms, and 5% and 0.1% imbalance rates for the auc metric

5%			0.1%		
algorithm	rank		algorithm	rank	
auc					
weight	2.36	a	baseline	2.21	a
baseline	2.37	a	weight	2.30	a
smote	2.40	a	smote	2.55	ab
underbagging	2.87	b	underbagging	2.93	b
acc					
baseline	1.96	a	weight	1.94	a
weight	2.00	a	baseline	1.99	a
smote	2.31	b	smote	2.48	b
underbagging	3.73	c	underbagging	3.60	c
f1					
smote	2.12	a	weight	2.39	a
baseline	2.57	b	smote	2.45	a
underbagging	2.64	b	baseline	2.46	a
weight	2.67	b	underbagging	2.70	a
gmean					
underbagging	1.49	a	underbagging	1.46	a
smote	2.39	b	smote	2.76	b
baseline	3.04	c	weight	2.87	b
weight	3.09	c	baseline	2.90	b
mcc					
smote	2.22	a	weight	2.39	a
baseline	2.55	b	baseline	2.40	a
weight	2.58	b	underbagging	2.59	a
underbagging	2.66	b	smote	2.62	a

TABLE 5
Comparisons of the best solution for each metric and RUSBoost

metric	best combination		rusboost	p.value
	rank	name	rank	
auc	1.21	rf+baseline	1.79	1e-15
acc	1.05	xgb+baseline	1.95	3e-37
f1	1.13	xgb+smote	1.87	6e-35
gmean	1.11	rf+underbagging	1.89	2e-31
mcc	1.15	xgb+smote	1.85	1e-33

3.5 Best base classifiers

Table 6 displays the comparison of each base classifier, when operating in baseline, for each metric. For AUC and accuracy, random forest and gradient boosting are equally good; for F-measure, G-mean and MCC gradient boosting performs better. In all cases SVM, is the worst algorithm.

TABLE 6
Comparison of the base classifiers using baseline for each metric.

algorithm	rank		algorithm	rank	
auc			acc		
rf	1.74	a	xgb	1.73	a
xgb	1.76	a	rf	1.87	a
svm	2.50	b	svm	2.40	b
f1			gmean		
xgb	1.66	a	xgb	1.63	a
rf	1.91	b	rf	1.95	b
svm	2.44	c	svm	2.42	c
mcc					
xgb	1.66	a			
rf	1.88	b			
svm	2.46	c			

4 Discussion

A very important aspect of the results reported above is that they are strongly related to the metric. As far as we know, this has not been reported in the literature, and it has important consequences to both practitioners and researchers. Practitioners must first define the metric used to assess the quality of the classifier before attempting different imbalance techniques. If the practitioner is using AUC as metric then the baseline or class weight is likely the best techniques, but they are the worst if one is using MCC, and both differences are significative.

Unfortunately, we cannot provide any advice to the practitioner as to which metric is the most useful or is the "correct" metric. But the reader should be aware of literature such as [39] and [40] which link performance metrics with expected classification costs when one does not know the operational conditions (ratio of imbalance and costs of errors for different classes) in advance.

The second conclusion is that this research seems to contradict the published results, in particular the results that use AUC as metric [16], [17], [18]. None of these papers points the baseline and class weight as the best alternatives for improving the AUC in imbalanced data.

The important difference between our research and theirs is that we use much more powerful base classifiers. [16] and [18] include SVMs among less powerful classifiers such as tree and rule induction, and naive bayes while we test only with SVMs, random forest, and boosting machines. We understand that there are good reasons to use simpler base classifiers if the goal of the research is to advance one's understanding of the techniques themselves. If the base classifier "is not of much help", all the differences observed are due to differences in the techniques themselves. But under the practitioner's perspective we adopted, the base classifiers will not be necessarily simple. In particular the two base classifiers we introduced (random forest and boosting machines) are particularly good in dealing with imbalanced data, and the techniques seem to hinder their ability to deal with the problem.

Table 7 shows the comparison on the AUC metric of the four techniques and RUSBoost as a special purpose classifier when using weaker base classifiers, in this case a 1-nearest neighbour and a CART decision tree. The results partially reproduce the conclusions of the previous research. [17] lists as winning techniques both RUSBoost and Underbagging which are the best two techniques in Table 7 and they are not statistically significantly different. We must notice that the conditions of our tests are similar to the ones in [17] which uses only a decision tree as base classifier. The results are less congruent to [18] which lists SMOTE, RUSBoost and class weight as winning strategies, obtained using knn, C4.5 and SVM (all with fixed hyperparameters) as base classifiers. In our case, baseline seems to perform better than class weight and underbagging is among the two best techniques which are not conclusions in [18].

The results for RUSBoost (Table 5) show that it is not worth using it as a special purpose classification algorithm for imbalanced data. As we discussed, RUSBoost is an external technique that combines boosting and sampling and could be used with any base classifier. By fixing the base

TABLE 7
Comparisons of the techniques using weaker classifiers (1-nn and cart) for AUC

algorithm	mean.rank	auc
rusboost	2.13	a
underbagging	2.20	a
smote	2.88	b
baseline	3.45	c
weight	4.34	d

classifier to a decision tree we were expecting that it would be a competitive classifier. In fact, it is competitive when compared to weaker base classifiers (Table 7) but not when compared to stronger classifiers. Surprisingly, RUSBoost is not competitive even when compared to gradient boosting - also a boosting ensemble based on decision trees, just like RUSBoost, but without any special consideration given to imbalance. Table 8 displays the mean rank of gradient boosting with baseline and RUSBoost, for the AUC metric. The table also displays the p-value of the Wilcoxon signed rank test between the two algorithms.

TABLE 8
Comparison between XGB+baseline and RUSBoost for AUC

xgb	rusboost	p.value
1.16	1.83	2.2e-16

Finally, Prati et al [16] point out that SVM is the best base classifier among the ones tested, but none of them were random forest or boosting machines. We discovered that for AUC, both these base classifiers perform significantly better than SVMs. We hope that this is a useful information to researchers, who now should include those algorithms when they want to compare alternative imbalance correcting techniques on stronger base classifiers.

5 Conclusions

The first important conclusion of this research is that the best technique strongly depends on the metric used. The practitioner must define which metric will be used to evaluate the quality of the classifier.

If the user is using AUC as the metric of quality then:

- baseline or class weight techniques are likely the best alternatives (regardless if the imbalance is moderate (5%) or severe (0.1%).
- if the practitioner has control over which base classifier to use, then one should opt for a gradient boosting or a random forest.

If one is using the standard accuracy as metric:

- baseline or class weight are the best techniques.
- the best classifier is gradient boosting.

If one is using the f-measure as metric:

- SMOTE is a good technique for any imbalance rate.
- best classifier is gradient boosting.

If one is using the G-mean as metric:

- Underbagging is the best technique for any imbalance rate.
- best classifier are random forest and gradient boosting.

If one is using the Matthew's correlation coefficient as metric:

- Almost all techniques perform the equally well in general, but SMOTE seems to perform better for less imbalanced data
- the best classifier is gradient boosting

Finally, more of a contribution to researchers, regardless of the metric used, random forest and gradient boosting outperform SVMs as base classifiers.

Appendix A
Best combinations - Tables

References

[1] H. He and Y. Ma, Eds., *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, 2013.

[2] A. Estabrooks, T. Jo, and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," *Computational intelligence*, vol. 20, no. 1, pp. 18–36, 2004.

[3] C. Drummond, R. C. Holte *et al.*, "C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," in *Workshop on learning from imbalanced datasets II*, vol. 11. Citeseer, 2003, pp. 1–8.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[5] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE, 2008, pp. 1322–1328.

[6] P. Domingos, "Metacost: A general method for making classifiers cost-sensitive," in *Proceedings ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 155–164.

[7] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.

[8] G. Wu and E. Y. Chang, "Kba: Kernel boundary alignment considering imbalanced data distribution," *IEEE Transactions on knowledge and data engineering*, vol. 17, no. 6, pp. 786–795, 2005.

[9] Y.-H. Liu and Y.-T. Chen, "Face recognition using total margin-based adaptive fuzzy support vector machines," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 178–192, 2007.

[10] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[11] B. Raskutti and A. Kowalczyk, "Extreme re-balancing for SVMs: a case study," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 60–69, 2004.

[12] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2003, pp. 107–119.

[13] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Rusboost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2010.

[14] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems," *Journal Machine Learning Reseach*, vol. 15, no. 1, pp. 3133–3181, 2014.

[15] J. Wainer, "Comparison of 14 different families of classification algorithms on 115 binary datasets," arXiv, Tech. Rep. 1606.00930, 2016. [Online]. Available: https://arxiv.org/abs/1606.00930

[16] R. C. Prati, G. E. Batista, and D. F. Silva, "Class imbalance revisited: a new experimental setup to assess the performance of treatment methods," *Knowledge and Information Systems*, vol. 45, no. 1, pp. 247–270, 2015.

TABLE 9
Comparisons of the combination of base algorithm and technique for the auc, acc, and f1 metrics

auc			acc			f1		
algorithm	rank		algorithm	rank		algorithm	rank	
rf_baseline	4.98	a	xgb_baseline	3.97	a	xgb_smote	3.61	a
xgb_baseline	5.03	a	xgb_weight	4.05	a	rf_smote	5.06	b
xgb_weight	5.13	a	rf_weight	4.26	a	xgb_weight	5.07	b
rf_smote	5.19	a	rf_baseline	4.29	a	xgb_baseline	5.20	b
rf_weight	5.20	a	xgb_smote	5.30	b	rf_weight	5.81	b
xgb_smote	5.23	a	rf_smote	5.64	b	rf_baseline	5.84	b
rf_underbagging	6.50	b	svm_baseline	6.17	b	rf_underbagging	6.96	c
xgb_underbagging	6.91	bc	svm_weight	6.19	b	xgb_underbagging	7.18	c
svm_baseline	7.78	c	svm_smote	7.70	c	svm_underbagging	7.21	c
svm_weight	7.91	cd	svm_underbagging	9.48	d	svm_baseline	8.46	d
svm_smote	8.85	de	rf_underbagging	10.29	de	svm_weight	8.55	d
svm_underbagging	9.30	e	xgb_underbagging	10.66	e	svm_smote	9.04	d

TABLE 10
Comparisons of the combination of base algorithm and technique for the gmean and mcc metrics

algorithm	rank		algorithm	rank	
gmean			mcc		
rf_underbagging	2.83	a	xgb_smote	4.02	a
xgb_underbagging	3.06	a	xgb_baseline	4.99	ab
xgb_smote	4.62	b	xgb_weight	5.10	b
svm_underbagging	5.43	bc	rf_smote	5.29	b
rf_smote	6.09	cd	rf_baseline	5.73	b
xgb_weight	6.19	de	rf_weight	5.77	b
xgb_baseline	6.19	de	rf_underbagging	6.80	c
rf_weight	7.26	e	xgb_underbagging	6.90	c
rf_baseline	7.28	e	svm_underbagging	7.34	c
svm_baseline	9.28	f	svm_baseline	8.37	d
svm_weight	9.36	f	svm_weight	8.46	d
svm_smote	9.64	f	svm_smote	9.23	d

[17] 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)*, vol. 42, no. 4, pp. 463–484, 2012.

[18] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information Sciences*, vol. 250, pp. 113–141, 2013.

[19] B. Bischl, M. Lang, L. Kotthoff, J. Schiffner, J. Richter, E. Studerus, G. Casalicchio, and Z. M. Jones, "mlr: Machine learning in r," *Journal of Machine Learning Research*, vol. 17, no. 170, pp. 1–5, 2016.

[20] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539–550, 2009.

[21] Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," in *Icml*, vol. 96. Bari, Italy, 1996, pp. 148–156.

[22] N. Japkowicz, *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, 2013, ch. Assessment metrics for imbalanced learning, pp. 187–206.

[23] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine learning*, vol. 30, no. 2-3, pp. 195–215, 1998.

[24] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[25] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, p. 31, 2016.

[26] S. Boughorbel, F. Jarray, and M. El-Anbari, "Optimal classifier for imbalanced data using matthews correlation coefficient metric," *PloS one*, vol. 12, no. 6, p. e0177678, 2017.

[27] D. Chicco, "Ten quick tips for machine learning in computational biology," *BioData mining*, vol. 10, no. 1, p. 35, 2017.

[28] W. Zong, G.-B. Huang, and Y. Chen, "Weighted extreme learning machine for imbalance learning," *Neurocomputing*, vol. 101, pp. 229–242, 2013.

[29] S. Li, Z. Wang, G. Zhou, and S. Y. M. Lee, "Semi-supervised learning for imbalanced sentiment classification," in *Proceedings International Joint Conference on Artificial Intelligence*, 2011, p. 1826.

[30] R. Barandela, J. S. Sánchez, V. Garcıa, and E. Rangel, "Strategies for learning in class imbalance problems," *Pattern Recognition*, vol. 36, no. 3, pp. 849–851, 2003.

[31] V. López, A. Fernández, and F. Herrera, "On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed," *Information Sciences*, vol. 257, pp. 1–13, 2014.

[32] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.

[33] S. Garcia and F. Herrera, "An extension on"statistical comparisons of classifiers over multiple data sets"for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, no. Dec, pp. 2677–2694, 2008.

[34] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[35] B. Bergmann and G. Hommel, "Improvements of general multiple test procedures for redundant systems of hypotheses," in *Multiple Hypotheses Testing*. Springer, 1988, pp. 100–115.

[36] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian journal of statistics*, pp. 65–70, 1979.

[37] B. Calvo and G. Santafé Rodrigo, "scmamp: Statistical comparison of multiple algorithms in multiple problems," *The R Journal*, vol. 8, no. 1, 2016.

[38] H.-P. Piepho, "An algorithm for a letter-based representation of all-pairwise comparisons," *Journal of Computational and Graphical Statistics*, vol. 13, no. 2, pp. 456–466, 2004.

[39] J. Hernández-Orallo, P. Flach, and C. Ferri, "A unified view of performance metrics: translating threshold choice into expected classification loss," *Journal of Machine Learning Research*, vol. 13, no. Oct, pp. 2813–2869, 2012.

[40] D. J. Hand, "Measuring classifier performance: a coherent alternative to the area under the roc curve," *Machine learning*, vol. 77, no. 1, pp. 103–123, 2009.