Tackling Class Imbalance with Ranking

Ricardo Cruz*, Kelwin Fernandes*†, Jaime S. Cardoso*† and Joaquim F. Pinto Costa‡
*INESC TEC, Portugal
Email: {rpcruz, kafc, jaime.cardoso}@inesctec.pt
†Faculty of Engineering, University of Porto, Portugal
‡Faculty of Sciences, University of Porto, Portugal
Email: jpcosta@fc.up.pt

***Abstract*—In classification, when there is a disproportion in the number of observations in each class, the data is said to be class imbalance. Class imbalance is pervasive in real world applications of data classification and has been the focus of much research. The minority class contributes too little to the decision boundary because the learning process learns from each observation in isolation. In this paper, we discuss the application of learning pairwise rankers as a solution to class imbalance. We compare ranking models to alternatives from the literature.**

I. Introduction

It is not uncommon in classification problems for data to be class imbalance; that is, the class distribution is not uniform, sometimes dramatically not. This is true in such fields as medicine where more people are cleared as negative in screening than are accused positives by such tests. In such cases, a naive application of learning algorithms will produce uninteresting models that have very good overall accuracy, at the expense of the minority class.

Several approaches have been proposed in tackling this problem, which usually involve:

A. ***Pre-processing*** step changing the class priors by undersampling the majority class and/or creating new synthetic examples of the minority class [1], or even changing class priors by changing class labels themselves (e.g. MetaCost [2]);
B. ***Training with costs*** instead of maximizing accuracy, the training algorithm maximizes weighted accuracy, so that the cost of misclassifying a class is inversely proportional to its frequency;
C. ***Post-processing*** by tweaking the decision boundary by such measures as changing a threshold after which one class is selected, sometimes with the aid of a ROC curve;
D. ***Ensembles*** by which each model within the ensemble is trained with balanced subsets of the data, coupled with the previous preprocessing techniques.

This list is by no means meant to be exhaustive. One-class models are also sometimes used to identify the minority class, though they do not usually produce very interesting results [3, see Table 4]. On the other hand, some rule induction models can be made to prioritize one class, and have been found to produce interesting results [4].

In this work we propose adding pairwise rankers to the repertoire of such techniques. We will first introduce some of the current techniques from the literature in more detail before delving into learning pairwise rankers. Previous work had found that rankers can produce better AUC curves [5].

A. Pre-processing

Stratification is the most popular pre-processing approach. It works either by undersampling from the majority class which has the side benefit of greatly improving training times when class imbalance is severe. Another approach, sometimes used in conjunction, is oversampling by creating new synthetic samples. A common algorithm is known as SMOTE [1], where new observations are created in between two existing observations using Euclidean distances. SMOTE has been extended by other algorithms; for example, MSMOTE [6]. These extensions improve SMOTE by adding new heuristics, most notably by identifying outliers and refraining from using them for the oversample, as well as identifying boundary points.

Also worth of notice is MetaCost which works by first assigning a probability to each observation as belonging to each class, by using a bagging of models, and secondly by calculating a threshold below which any observation of the majority class is assigned to the minority class [2]. In other words, it balances class priors by actually changing the class of those majority observations for which the underlying estimator is less certain about.

The immediate advantage of pre-processing is that it is model agnostic: class imbalance can then be solved as a separate problem. This is especially important when one is unsure of the most appropriate learner, and would prefer to tackle class imbalance as a separate issue.

B. Training with costs

Training by explicitly defining costs seems like the most direct approach. Instead of minimizing total misclassification, FP+FN, we minimize the weighted misclassification, $w_{\mathrm{P}}\mathrm{FP}+w_{\mathrm{N}}\mathrm{FN}$, where to the weights w_{P} and w_{N} are assigned the inverse frequency of their class priors (P and N stand for Positives and Negatives, respectively, while TP and FP are True and False Positives, conversely for TN and FN).

Unfortunately, adding cost-sensitivity to the training algorithm is not always straight-forward and is sometimes cumbersome. Taking SVMs as an example, suggestions have been made to introduce costs in the feature space transformation by changing the kernel function [7], introducing different

penalties for the positive and negative SVM slack variable ξ [8], among other approaches.

Furthermore, cost training sometimes saturates and cannot expand beyond the limits of the data, which pre-processing methods can help. It has been found that pre-processing approaches are in fact oftentimes superior [2].

C. Post-processing

This step consists in varying the threshold by which the class is chosen in a binary classifier (e.g. neural network), or it could mean varying the bias in a SVM model to adjust the decision boundary [9].

D. Ensembles

Several ensembles have been proposed recently. Easy Ensemble is a high performing bagging technique where each model is trained from undersampled pools of the data, in which each pool has the same number of positive and negative observations [10]. An extension to this model is Balance Cascade whereby undersampling of the majority class is guided to remove observations that have been correctly classified by the previous model in the ensemble, also [10].

Other methods, such as SMOTE Boost, SMOTE Bagging, IIVOTES or RUSBOOST, on the other hand, create new synthetic observations based on the harder to classify cases of minority observations; this is in opposition to traditional boosting methods which assign a distribution of weights to the observations.

Ensembles tend to triumph in recent literature [3]. It does not seem completely clear however whether what contributes to these gains is the combination of ensemble and stratification, or whether it is simply the ensemble since they are not usually benchmarked against ensembles of the other approaches.

II. Ranking for Class Imbalance

One possible family of methods to tackle the class imbalance problem is pairwise ranking algorithms, in particular pairwise scoring rankers. The term document is typically used in the literature to refer to observation because of its genesis and tight connection to information retrieval techniques [11].

In ranking, document $\mathbf{x}_i$ is compared with another document $\mathbf{x}_j$, and we are interested in predicting whether $\mathbf{x}_i \succ \mathbf{x}_j$, meaning $\mathbf{x}_i$ is "preferred" to $\mathbf{x}_j$. The three big umbrellas of rankers are:

- **Pointwise**, in which each document $\mathbf{x}_i$ is trained individually and a score function, $f(\mathbf{x}_i)$, is given based on its relevance;
- **Pairwise**: each document $\mathbf{x}_i$ is compared against all others $\mathbf{x}_j$, and if $\mathbf{x}_i \succ \mathbf{x}_j$, then we train a function f so that:
  - **pairwise scoring ranker:** if $\mathbf{x}_i \succ \mathbf{x}_j$ then $f(\mathbf{x}_i) > f(\mathbf{x}_j)$, with $f\colon X \to \mathbb{R}$;
  - **pairwise non-scoring ranker:** the decision function is such that it decides which of two documents is preferred, $f\colon X^2 \to X$;
- **Listwise**, where the training loss function is based on all documents and their scores.

We propose to consider pairwise scoring rankers for the class imbalance problem. Ranking algorithms for classification have been found to make highly competitive classifiers [12]. And, as we will see, there is no class imbalance when doing pairwise ranking. Since we are comparing each observation of one class to every observation of the other class, the ensuing training process is necessarily balanced.

In the same spirit of the category of models presented in the introduction, we can see the ranking process as being composed of the following steps: pre-processing, training, and post-processing (see diagram in Figure 1).

A. Pre-processing

In the case of binary classification, pairwise rankers are trained so that for every two observations, $(\mathbf{x}_i, \mathbf{x}_j)$ and respective class labels (y_i, y_j), a transformation f is applied so that $\mathbf{x}_i \succ \mathbf{x}_j$ if $\mathrm{P}(y_i = 1) > \mathrm{P}(y_j = 1)$, and $\mathbf{x}_i \prec \mathbf{x}_j$ otherwise. Here we take 1 as being the minority class, without loss of generality.

In order to illustrate the ranking approach, among many potential pairwise scoring rankers, here three are considered. Any others could be adapted in an analogous manner. These were selected because they are a) pairwise scoring, and b) they encompass major families of rankers:

TABLE I
RANKING MODELS EXPLORED

Family	Ranker	Reference
Linear SVM	RankSVM	[13]
Neural Networks	RankNet	[14]
AdaBoost	RankBoost	[15]

In **RankSVM**, data is transformed into the space of differences, so the original dataset $\mathbf{X}$ becomes $\mathbf{X}'$, where $\mathbf{x'}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ and $\mathbf{x'}_{ji} = \mathbf{x}_j - \mathbf{x}_i$, for all pairs (i, j) such that $y_i \neq y_j$, with $y'_{ij} = y_i$ and $y'_{ji} = y_j$.

In all others, data is transformed so that $\mathbf{X}_i = \{\mathbf{x}_i, y_i\}$ becomes $\mathbf{X}'_{ij} = \{\mathbf{x_i}, y'_{ij}\}$, for all combinations (i, j), (and ditto for the symmetric relation (j, i)), where y'_{ij} denotes a relation of preference between $\mathbf{x}_i$ and $\mathbf{x}_j$ ($y'_{ij} = 1$ if $\mathbf{x}_i \succ \mathbf{x}_j$, or -1 otherwise). In **RankBoost**, y'_{ij} denotes the class of i, $y'_{ij} = y_i$ (and all combinations such that $y_i = y_j$ are omitted), while in **RankNet** y'_{ij} represents the ranking probability we aim to estimate,

$$y'_{ij} = \begin{cases} 0, & \text{if } y_i < y_j \\ 1, & \text{if } y_i > y_j \\ 0.5, & \text{if } y_i = y_j. \end{cases}$$

The data with which the ranking estimator is trained is therefore usually bigger, and so training times tend to be slower than ordinary classification methods. In general, the transformed dataset $N' \in \mathcal{O}(N^2)$, but in cases like RankSVM or RankBoost which use only pairs of opposite classes, $N' = 2N_0N_1$ and, because of class imbalance, $N_0 \gg N_1$, so $N' \approx 2N_0$.

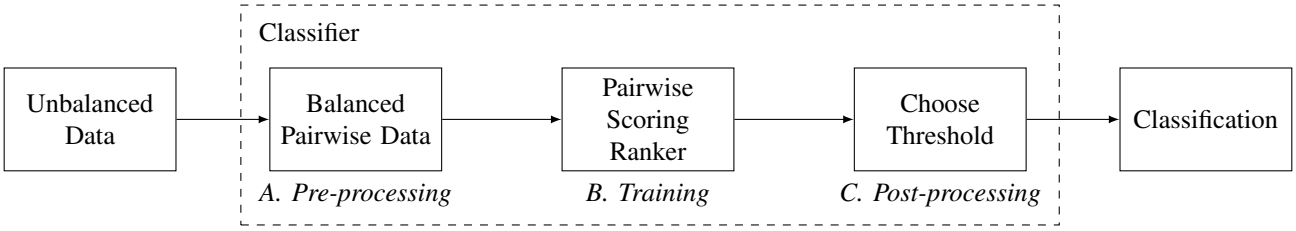

Fig. 1. Schematic of the pairwise ranking classifier applied to class imbalance data.

B. Training

When it comes to training, **RankSVM** makes use of a linear SVM as a base estimator to classify observations within the space of differences, where the decision rule $\mathbf{w}\cdot(\mathbf{x}_i-\mathbf{x}_j)>0$ can be transformed into a scoring function since $\mathbf{w}\cdot(\mathbf{x}_i-\mathbf{x}_j)>0\equiv\mathbf{w}\cdot\mathbf{x}_i>\mathbf{w}\cdot\mathbf{x}_j\equiv s(\mathbf{x}_i)>s(\mathbf{x}_j)$. In **RankNet**, a neural network is used to estimate y'_{ij} which denotes the probability $\mathrm{P}(\mathbf{x}_i\succ\mathbf{x}_j)$.

RankBoost, like AdaBoost from the same authors, trains a base estimator, at each iteration t, using an underlying distribution of weights for each pair, for which $D_{ij}=D_{ji}$. The difference from AdaBoost is in that α_t is a function of the number of pairs whose order has been correctly or incorrectly estimated: $\alpha_t=\frac{1}{2}\log\frac{W_{-1}}{W_{+1}}$, where $W_{b=\{-1,+1\}}=\sum_{i,j}D_{ij}I_{f_t(\mathbf{x}_i)-f_t(\mathbf{x}_j)=b}$.

Loss functions are therefore hinge loss, logistic loss, and exponential loss for RankSVM, RankNet, and RankBoost, respectively.

C. Post-processing

As discussed, a pairwise scoring ranker produces a score function $f\colon X\to\mathbb{R}$, with which we predict a class $\{0,1\}$ using a scoring threshold. We have chosen the threshold T that maximizes the F_1 score, which is more appropriate than accuracy for class imbalance, and is defined as:

$$\mathrm{F}_1=\frac{2\,\mathrm{TP}}{2\,\mathrm{TP}+\mathrm{FN}+\mathrm{FP}}.$$

Using the training data, we have $s_i=f(\mathbf{x}_i)$ which we order, and use each midpoint $s'_i=\frac{s_i+s_{i+1}}{2}$ as possible candidates for threshold T, so that

$$T=\arg\max_{s'_i}\ \mathrm{F}_1(s'_i).$$

For fairness, the F_1 score metric is also used when cross-validating the best parameters of the models we are comparing.

In the following experiments tables, we also make use of these scores to calculate the area under curve (AUC) of ROC curves. The ROC curve is a common measure to evaluate how correctly classified observations would be if the decision threshold T was changed. What should constitute this decision threshold is not always obvious; in the SVM, this could be the distance to the separating hyperplane, as was used, or varying the bias [9]. In ranking models, the ROC can easily be drawn by choosing different scores s_i for T.

III. Experiments

Fifteen empirical datasets are considered (see Table II). Most datasets used have N (the number of observations) in the order of thousands, to ensure what is being tested is "relative rarity", and avoid "absolute rarity" issues [16]. Some of these are multinomial classification datasets which were converted to binary classification using the class label mentioned in the "Minority" column. These samples are based on [17]. All others are binary classification datasets. "IR" is the imbalance ratio (N_1/N). Datasets in all proceeding tables are ordered by IR.

Overlap is a measure of how intertwined the observations from the two classes are. There is a big amount of literature on the role of overlapping in class imbalance [21], with some authors arguing these problems are often conflated [22]. Our measure of overlap is defined as the ratio of minority observations whose closest neighbor is an observation of the majority class. We compare models' correlation to IR and overlap in the results.

Experiments are done by cross validation through a bootstrap process: each sample is randomly split into 40 folds of 80-20% stratified splits of train-test. The average of F_1 and ROC AUC for each dataset is exhibited. The best scores are presented in bold, as well as all statistically identical scores, using a paired difference Student's t-test with a 95%

TABLE II
Datasets

Dataset	Minority	N	Features	**IR**	Overlap
sonar	—	208	60	0.466	0.216
breast-cancer	—	699	9	0.345	0.075
german	—	1000	24	0.300	0.563
haberman	—	306	3	0.265	0.605
transfusion	—	748	4	0.238	0.629
vehicle	van	846	18	0.235	0.090
CTG	—	2126	22	0.222	0.172
hepatitis	—	143	14	0.203	0.621
segment	1	2310	19	0.143	0.009
winequality-red	7,8	1599	11	0.136	0.512
vowel	1	990	13	0.091	0.011
abalone	9vs18	731	7	0.057	0.595
glass	6	214	9	0.042	0.556
car	good	1728	6	0.040	0.667
yeast	ME1	1484	8	0.030	0.341

Acknowledgments: Datasets come courtesy of the UCI Machine Learning repository [18]. The breast cancer dataset was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg [19]. The vehicle dataset is originally from the Turing Institute, Glasgow, Scotland. Wine-quality is originally from [20].

TABLE III
FAMILY: LINEAR SVM

Linear SVM	F_1						*ROC AUC*					
Sample	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost
sonar	**0.741**	0.723	**0.734**	0.723	0.724	0.725	**0.832**	0.823	**0.826**	0.823	0.823	0.816
breast-cancer-wisconsin	**0.957**	0.953	0.954	**0.955**	0.954	0.953	0.994	**0.994**	0.994	0.994	**0.994**	**0.994**
german	**0.618**	0.568	**0.622**	0.616	**0.617**	0.598	**0.808**	**0.809**	**0.809**	0.806	0.804	**0.808**
haberman	**0.485**	0.188	**0.476**	**0.484**	**0.469**	0.253	**0.685**	**0.689**	**0.690**	0.677	0.668	**0.688**
transfusion	0.516	0.154	**0.528**	0.518	0.522	0.174	**0.758**	**0.758**	**0.757**	0.753	0.752	**0.758**
vehicle-van	**0.937**	**0.940**	0.930	0.932	0.934	0.927	**0.995**	0.995	0.995	0.995	0.994	0.994
CTG	0.925	0.931	0.915	0.917	0.919	**0.934**	**0.993**	**0.993**	**0.993**	0.993	0.992	0.993
hepatitis	**0.634**	**0.606**	**0.651**	**0.630**	**0.632**	**0.648**	**0.882**	**0.877**	**0.884**	**0.882**	**0.871**	**0.881**
segment-1	0.986	**0.991**	0.988	**0.990**	**0.990**	0.990	0.996	**0.998**	0.997	**0.998**	**0.998**	**0.999**
winequality-red-7,8	**0.517**	0.228	0.479	0.482	0.491	0.346	**0.858**	**0.857**	**0.858**	0.856	0.850	0.855
vowel-1	**0.457**	0.180	**0.445**	0.442	0.421	0.273	**0.892**	0.884	**0.893**	0.887	0.850	0.870
abalone-9vs18	**0.652**	0.502	0.473	0.493	0.500	**0.632**	0.948	**0.952**	**0.950**	0.948	0.926	**0.950**
glass-6	**0.695**	0.000	0.234	0.236	0.226	0.010	**0.984**	0.507	0.763	0.752	0.761	0.436
car-good	**0.476**	0.064	0.422	0.438	0.391	0.274	**0.959**	0.958	**0.959**	0.958	0.941	0.955
yeast-ME1	**0.612**	0.523	0.556	0.562	0.564	0.571	**0.986**	**0.986**	**0.986**	0.985	0.984	0.986
Average	0.681	0.503	0.627	0.628	0.624	0.554	0.905	0.872	0.890	0.887	0.881	0.865
Winner	80%	20%	40%	26%	26%	20%	80%	66%	73%	13%	20%	46%

TABLE IV
FAMILY: ADABOOST

AdaBoost	F_1						*ROC AUC*					
Sample	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost
sonar	0.787	**0.824**	**0.824**	**0.824**	**0.824**	0.818	0.891	**0.917**	**0.917**	**0.917**	**0.917**	**0.916**
breast-cancer	0.937	0.932	0.932	0.937	0.934	**0.948**	**0.990**	0.989	0.989	0.990	0.989	**0.991**
german	**0.597**	0.541	0.541	**0.588**	0.583	**0.587**	0.783	**0.794**	**0.794**	**0.792**	**0.793**	**0.796**
haberman	0.419	0.375	0.375	**0.464**	**0.458**	0.441	0.638	0.663	0.663	0.671	0.673	**0.692**
transfusion	**0.502**	0.418	0.418	**0.511**	**0.509**	0.493	0.718	**0.745**	**0.745**	0.737	0.737	**0.740**
vehicle-van	**0.928**	0.901	0.901	0.905	0.909	0.906	**0.993**	0.991	0.991	0.991	0.991	0.989
CTG	0.973	0.972	0.972	0.970	0.971	**0.979**	0.996	**0.997**	**0.997**	0.996	**0.997**	0.996
hepatitis	**0.578**	0.525	0.525	**0.581**	**0.596**	**0.596**	0.822	0.808	0.808	**0.833**	**0.842**	**0.846**
segment-1	**0.993**	0.990	0.990	0.987	0.988	0.988	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	1.000
winequality-red-7,8	**0.520**	0.431	0.431	0.509	0.511	**0.528**	**0.867**	**0.868**	**0.868**	0.864	0.863	**0.869**
vowel-1	**0.692**	0.443	0.443	0.610	0.549	0.633	**0.953**	0.946	0.946	**0.948**	0.930	0.939
abalone-9vs18	**0.369**	0.318	0.318	0.287	0.298	**0.377**	0.803	**0.820**	**0.820**	0.793	0.791	**0.822**
glass-6	**0.801**	0.670	0.670	**0.825**	**0.777**	0.713	**0.996**	**0.998**	**0.998**	**0.993**	0.989	0.982
car-good	**0.573**	0.388	0.388	**0.596**	0.515	0.401	0.974	**0.977**	**0.977**	0.976	0.965	0.919
yeast-ME1	**0.667**	**0.671**	**0.671**	0.654	0.635	**0.698**	0.982	**0.986**	**0.986**	**0.985**	0.982	**0.986**
Average	0.689	0.627	0.627	0.683	0.671	0.674	0.894	0.900	0.900	0.899	0.897	0.899
Winner	73%	13%	13%	46%	33%	46%	40%	66%	66%	46%	33%	60%

TABLE V
FAMILY: NEURAL NETWORKS

Neural Networks	F_1						*ROC AUC*					
Sample	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost
sonar	**0.805**	**0.804**	**0.801**	0.731	0.732	0.725	**0.881**	**0.896**	**0.895**	0.830	0.829	0.817
breast-cancer	0.946	0.942	0.947	**0.955**	**0.954**	**0.955**	0.975	0.989	0.991	0.994	0.993	**0.994**
german	0.513	0.540	0.543	**0.618**	0.609	**0.623**	0.665	0.744	0.721	**0.808**	0.807	**0.809**
haberman	0.444	0.361	0.459	0.435	0.431	**0.497**	0.604	**0.675**	**0.678**	0.676	0.663	**0.688**
transfusion	0.495	0.391	0.506	0.492	0.502	**0.525**	0.549	**0.764**	0.753	0.757	0.755	0.758
vehicle-van	**0.944**	**0.940**	**0.941**	0.593	0.596	0.596	**0.982**	**0.996**	**0.996**	0.985	0.985	0.923
CTG	0.961	**0.965**	**0.963**	0.919	0.919	0.925	0.993	0.997	**0.998**	0.993	0.993	0.993
hepatitis	**0.600**	0.502	0.528	0.520	0.515	**0.611**	**0.828**	0.803	0.801	0.795	0.791	**0.846**
segment-1	**0.989**	**0.990**	**0.975**	0.984	0.984	0.986	**0.999**	**0.999**	**0.999**	0.996	0.996	0.996
winequality-red-7,8	0.477	0.482	**0.508**	0.316	0.317	0.269	0.555	0.823	**0.843**	0.790	0.786	0.655
vowel-1	0.397	**0.946**	0.855	0.480	0.482	0.379	0.516	**0.989**	0.973	0.963	0.954	0.899
abalone-9vs18	**0.511**	0.485	0.362	0.301	0.347	0.358	0.801	**0.917**	0.907	**0.927**	**0.922**	0.889
glass-6	0.024	0.000	0.136	**0.350**	**0.347**	0.290	0.442	0.338	0.556	**0.683**	**0.698**	0.610
car-good	**0.839**	**0.849**	0.737	0.447	0.402	0.392	0.959	**0.996**	0.982	0.966	0.951	0.953
yeast-ME1	**0.653**	0.564	0.528	0.603	0.583	0.597	0.950	**0.986**	0.983	**0.986**	0.984	**0.986**
Average	0.640	0.651	0.653	0.583	0.581	0.582	0.780	0.861	0.872	0.876	0.874	0.854
Winner	46%	40%	33%	20%	13%	33%	26%	60%	40%	26%	13%	33%

confidence level.

Furthermore, a 5-fold validation is performed for SVM and neural networks to find the best parameter: the regularization coefficient C and hidden nodes H, respectively, choosing between 5 parameters along the range $C \in [0.01, 100]$, and $H \in [F, F^2]$, with F being the number of features. SVM was trained using a linear kernel with liblinear. Stochastic gradient descent was used with learning rate = 1.0, and 1000 as the epochs maximum. The data was normalized for both. AdaBoost and RankBoost were trained as an ensemble of 50 binary classifiers.

Four variants of the baseline model are provided in each column: loss function with weights using inverse class frequencies, and also together with SMOTE [1] and MSMOTE [6] with number of neighbors $k = 5$ applied to equalize frequencies, as well as with MetaCost [2] using an ensemble of $m = 50$ with the other parameters being $n = N$, $p = \text{False}$ and $q = \text{True}$.

All implementations from our work including the dataset folds are made publicly available at http://pong.inesctec.pt/~rpcruz/ijcnn2016/ (mostly Python was used).

IV. Results

The ranking models here considered have performed statistically significantly better than their counterparts from the literature, especially with regard to the F_1 score (Table III). In Linear SVM, when ranking won, it won by a much bigger margin than when it lost, $0.068/-0.009$, relative to the second performer and best performer.

The lower performance from the ROC AUC scores could suggest the threshold selection (section II-C) is partly responsible for the gain.

While not the main point of the work, it is worth noticing that, as other authors have argued [21], data's overlap (from Table II) explains better model's F_1 scores than imbalance ratio (IR), as measured by Spearman's ρ ($\rho \in [-1, 1]$ with high/low $|\rho|$ meaning high/low correlation), see Table VI. More importantly, rankers, when compared to the other models within their family, produce models least correlated to the imbalance ratio. It was already visible from Table III, which is ordered by IR, that rankers gains are concentrated in the bottom (the more unbalanced). And, while overlap explains scores better than IR, as already stated, no systemic tendency is apparent, and so all gains from rankers seem to accrue to solving the IR problem.

TABLE VI
Correlations: Data Complexity

Spearman's ρ	Ranking	Baseline	Weights	SMOTE	MSMOTE	MetaCost
Linear SVM						
IR	0.312	0.477	0.576	0.555	0.544	0.401
Overlap	-0.185	-0.302	-0.293	-0.285	-0.293	-0.293
AdaBoost						
IR	0.115	0.224	0.224	0.148	0.208	0.238
Overlap	-0.668	-0.609	-0.609	-0.613	-0.601	-0.674
Neural Networks						
IR	0.210	0.135	0.277	0.398	0.407	0.463
Overlap	-0.645	-0.756	-0.791	-0.650	-0.647	-0.584

TABLE VII
Correlations: Inter-Family

Spearman's ρ	Baseline	Weights	SMOTE	MSMOTE	MetaCost
RankSVM	0.710	0.742	0.751	0.756	0.715
RankBoost	0.842	0.842	0.888	0.861	0.862
RankNet	0.736	0.754	0.610	0.614	0.656

TABLE VIII
Correlations: Intra-Family

Spearman's ρ	RankSVM	RankBoost	RankNet
RankSVM	1.000	0.365	0.235
RankBoost	0.365	1.000	0.555
RankNet	0.235	0.555	1.000

Finally, we compare correlations within and between families of models. This can help in differentiating, on one hand, whether rankers are competing classifiers or if, on the other hand, rankers are alternative models that learn different data patterns. Correlation is, again, measured by Spearman's ρ. Naturally, the correlation between any two models will be high since we are using datasets that were chosen because they have different IRs, and IR is (inversely) correlated to model's performance, and is therefore a confounder. We control for IR's correlation using Fisher's partial correlation formula. This does not affect the relative correlations between any two models, but it reduces the overall magnitude of correlations to be more aligned to use cases when random samples from the same population, having the same IR, are used.

Table VII and VIII clearly show rankers more closely follow the decision function of their family of models than that of the rest of the rankers. Ranking techniques are therefore an extra technique of tackling class imbalance to try to improve a currently employed solution.

V. Discussion

There is a latent benefit when considering rankers as possible classifiers; a latent benefit that has not so far been discussed. Rankers can use extra information about the order of classes. This means that data collection is not as constrained to broad categories such as "healthy" and "sick", or "credit-worthy" and "not credit-worthy". Rankers can make use of extra subtlety in the classification by having a gradient of classes. A tangent point is that in many real world applications it might make sense to express the data from the get-go in terms of pairwise comparisons. It is often more intuitive for the human classifier to compare observations than to assign labels.

This was not a focus of this discussion, but one inconvenience is that training times are usually higher, possibly insuperably higher for very big datasets. We have however only implemented and experimented with the three major ranking families while ignoring the more recent progress.

Further work is required to more finely tuned ranking solutions, as well as combining rankers with current pre-

processing and ensemble solutions. Tackling imbalance in multi-class problems and reducing training times are other problems of interest. The ranking threshold decision could possibly be solved using a SVM to separate classes, or, more elegantly, while training.

VI. Conclusion

Almost two hundred papers have been published since 2012, just by searching titles by "class imbalance" as reported by Google Scholar. It is not clear that ranking is a superior solution, but it is a very competitive and promising alternative that we felt was sorely lacking in the literature.

Some classical ranking models were compared with conventional classification models. This work shows promising efficiency improvements from training using pairwise scoring ranking models. These models have been in general superior, and when their performance was worse, it was worse by a smaller margin than when the performance was positive. It is clear class imbalance performance can be improved by combining these models with other approaches from the literature.

It was also found that performance scores of ranking models tend to correlate with those of their underlying models, and so they may be seen as potential improvements on top of traditional classifiers.

Acknowledgment

This work is co-financed by: "NORTE-01-0145-FEDER-000016" of the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF)'s COMPETE 2020 Programme within project "POCI-01-0145-FEDER-006961", and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013.

References

[1] 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. [Online]. Available: http://arxiv.org/abs/1106.1813

[2] P. Domingos, "MetaCost: A General Method for Making Classifiers Cost-Sensitive," *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, vol. 55, pp. 155–164, 1999.

[3] L. Nanni, C. Fantozzi, and N. Lazzarini, "Coupling different methods for overcoming the class imbalance problem," *Neurocomputing*, vol. 158, pp. 48–61, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0925231215001411

[4] M. Kubat, R. Holte, and S. Matwin, "Learning when negative examples abound," in *Journal of Chemical Information and Modeling*, 1997, vol. 53, pp. 146–153. [Online]. Available: http://link.springer.com/10.1007/3-540-62858-4{_}79

[5] C. Cortes and M. Mohri, "AUC Optimization vs. Error Rate Minimization." *Advances in Neural Information Processing Systems*, pp. 313–320, 2003. [Online]. Available: https://papers.nips.cc/paper/2518-auc-optimization-vs-error-rate-minimization.pdf

[6] S. Hu, Y. Liang, L. Ma, and Y. He, "MSMOTE: Improving classification performance when training data is imbalanced," *2nd International Workshop on Computer Science and Engineering, WCSE 2009*, vol. 2, pp. 13–17, 2009.

[7] G. Wu and E. Chang, "Class-boundary alignment for imbalanced dataset learning," *The Twentieth International Conference on Machine Learning (ICML)*, no. 1, pp. 49–56, 2003. [Online]. Available: http://www.site.uottawa.ca/{~}nat/Workshop2003/Wu-final.pdf

[8] F. R. Bach, "Considering Cost Asymmetry in Learning Classifiers," *Jmlr*, vol. 7, pp. 1713–1741, 2006.

[9] H. Núñez, L. Gonzalez-Abril, and C. Angulo, "A post-processing strategy for SVM learning from unbalanced data," *ESANN 2011 proceedings, 19th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 195–200, 2010. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84887058524{\&}partnerID=tZOtx3y1

[10] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory Undersampling for Class Imbalance Learning," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 39, no. 2, pp. 539–550, 2009.

[11] H. Li, *Learning to Rank for Information Retrieval and Natural Language Processing*, 2011, vol. 4, no. 1.

[12] P. Flach and E. T. Matsubara, "A simple lexicographic ranker and probability estimator," *Machine Learning: ECML 2007*, pp. 575–582, 2007.

[13] R. Herbrich, T. Graepel, and K. Obermayer, "Support Vector Learning for Ordinal Regression A Risk Formulation for Ordinal Regression," *Proceedings of the Ninth International Conference on Artificial Neural Networks*, pp. 97–102, 1999. [Online]. Available: http://www.herbrich.me/papers/icann99{_}ordinal.pdf

[14] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd international conference on Machine learning - ICML '05*. New York, New York, USA: ACM Press, 2005, pp. 89–96. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1102351.1102363

[15] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An Efficient Boosting Algorithm for Combining Preferences," *The Journal of Machine Learning Research*, vol. 4, pp. 933–969, 2003.

[16] H. He and E. A. Garcia, "Learning from Imbalanced Data Sets," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263—-1284, 2010. [Online]. Available: http://www.aaai.org/Papers/Workshops/2000/WS-00-05/WS00-05-003.pdf

[17] S. Chen, H. He, and E. a. Garcia, "RAMOBoost: Ranked Minority Oversampling in Boosting," *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 21, no. 10, pp. 1624–42, 2010. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/20805051

[18] M. Lichman, "UCI Machine Learning Repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[19] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast Cancer Diagnosis and Prognosis Via Linear Programming," *Operations Research*, vol. 43, no. 4, pp. 570–577, 1995.

[20] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, nov 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167923609001377

[21] R. Prati, G. Batista, and M. Monard, "Class imbalances versus class overlapping: an analysis of a learning system behavior," *MICAI 2004: Advances in Artificial Intelligence*, pp. 312–321, 2004. [Online]. Available: http://www.springerlink.com/index/DJF08LA1YYVWJVJC.pdf

[22] M. Denil and T. Trappenberg, "Overlap versus imbalance," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6085 LNAI, pp. 220–231, 2010.