Performance and model complexity on imbalanced datasets using resampling and cost-sensitive algorithms

Jairo da Silva Freitas Júnior JAIRO.FREITAS@ALUNO.UFABC.EDU.BR
Paulo Henrique Pisani PAULO.PISANI@UFABC.EDU.BR
Center of Mathematics, Computing and Cognition, Federal University of ABC, Brazil

Editor: Nuno Moniz, Paula Branco, Luís Torgo, Nathalie Japkowicz, Michał Woźniak and Shuo Wang.

Abstract

Imbalanced datasets occur across industries, and many applications with high economical interest deal with them, such as fraud detection and churn prediction. Resampling is commonly used to overcome the tendency of machine learning algorithms to favor the majority class error minimization, while cost-sensitive algorithms are less used. In this paper, cost-sensitive algorithms (BayesMinimumRisk, Thresholding, Cost-Sensitive Decision Tree and Cost-Sensitive Random Forest) and resampling techniques (SMOTE, SMOTETomek and TomekLinks) combined with kNN, Decision Tree, Random Forest and AdaBoost were compared on binary classification problems. The results were analyzed with respect to relative performance over different imbalance ratios. The influence of these techniques for handling the class imbalance on the machine learning models complexities was also investigated. The experiments were performed using synthetic datasets and 90 real-world datasets.

Keywords: imbalanced dataset; resampling; cost-sensitive algorithm; machine learning.

1. Introduction

Imbalanced datasets are common in many applications, such as fraud detection, churn prediction and intrusion detection. Class imbalance can harm classification performance in the presence of class overlap (Prati et al., 2004) and decomposition (when a class is composed of two or more subconcepts) (Stefanowski, 2013). In these scenarios, where class separability is hard and the data is imbalanced, most machine learning algorithms tend to favor the minimization of the majority (most prevalent) class error, leading to lower performance on the minority class.

Many approaches to deal with imbalanced datasets have been proposed. One of the most popular categories of techniques for handling the class imbalance is resampling. Resampling techniques create or remove examples to balance the dataset. On the other hand, a less studied category of techniques is the cost-sensitive algorithms (Haixiang et al., 2017), which associate costs to the different combinations of classification errors and minimize the total classification cost. Compared to resampling, cost-sensitive algorithms can have some advantages including the preservation of the original data, capacity to assign distinct costs for all possible combinations of classification errors in multi-class classification, and, frequently, the capacity to specialize the classification costs at the example level. Considering these advantages, it makes sense to investigate comparatively the performance of the aforementioned techniques.

Weiss et al. (2007) compared a cost-sensitive decision tree with random oversampling and undersampling. They found that the cost-sensitive decision tree was more suitable for larger datasets, while oversampling for smaller ones. Seiffert et al. (2008) compared two cost-sensitive techniques (Thresholding and ExampleWeighting), with four resampling methods, considering two machine learning algorithms (C4.5 tree and RIPPER). In their results, ExampleWeighting usually was better than Thresholding. They also found that undersampling tends to outperform the other techniques, which does not agree with Weiss et al. (2007). In another study, López et al. (2012) compared the performance of two SMOTE-based techniques against cost-sensitive implementations of C4.5 tree, SVM, kNN and a fuzzy hybrid genetic based machine learning rule generation algortihm. They found that the techniques were statistically equivalent.

These results illustrate that establishing which properties of the data and which machine learning algorithms favors cost-sensitive or resampling techniques is an open question. Also, to the best of our knowledge, no investigation have been done on how each technique for handling the class imbalance may impact the complexity of the obtained models, which is the main contribution of this paper to the field of learning with imbalanced domains. Therefore, two research hypothesis are discussed: the relative performance of the techniques for handling the class imbalance may vary according to which machine learning algorithm they are combined with; and, the complexity of the obtained models may be affected by these techniques.

Next sections are organized as follows: Section 2 introduces the techniques for handling class imbalance; Section 3 shows the experiment design; Section 4 describes the datasets used in the experiments; Section 5 shows and discusses the results; and, Section 6 presents the final conclusions and future work.

2. Techniques for handling class imbalance

This section describes the techniques for handling class imbalance used in this work. SMOTE (Chawla et al., 2002) is one oversampling technique that generates new examples through interpolation of nearest neighbors. First, it identifies k-nearest neighbors for each example of the minority class that also belong to the minority class. Then, to generate a synthetic example, one example from the minority class is randomly chosen (x_1) and also one of its k-nearest neighbors (x_2). Finally, a random value between 0 and 1 (λ) is taken and the synthetic example is created from the equation:

$$x_{new} = x_1 + \lambda \times (x_2 - x_1) \tag{1}$$

TomekLinks (Tomek, 1976) is an undersampling heuristic-based technique. If two examples from distinct classes are closer to each other than to any other example, they form a Tomek link. The technique then removes the majority example from this link, that is believed to be noise in the data. This technique intends to reduce the imbalance and denoise the dataset. SMOTETomek (Batista et al., 2003, 2004) is an hybrid approach that combines SMOTE and TomekLinks. First, SMOTE is applied to balance the dataset, then Tomek links are identified and the two examples from the link are removed. The purpose of the last step is to improve class separability. Thresholding (Sheng and Ling, 2006) is a cost-sensitive algorithm that seeks to determine a new classification threshold for a cost-insensitive model in order to minimize its classification cost. Another probability-based cost-sensitive algorithm is BayesMinimumRisk, that tries to quantify the risk associated with the assignment of each class for an instance, finally making the choice that minimizes the risk (Ghosh et al., 2006). Formally, let i and j be two different classes, x an example, $C(i,j)$ the cost of confounding i as j, and $P(j|x)$ the probability of x being of class j (that we get from a cost-insensitive model). The Bayes Risk for class i is

$$R(i|x) = \sum_j C(i,j)P(j|x), \tag{2}$$

and the class assigned by BayesMinimumRisk is

$$L_{BMR}(x) = \underset{i}{\operatorname{argmin}}\, R(i|x). \tag{3}$$

Thresholding and BayesMinimumRisk are called cost-sensitive meta-algorithms, since they do not learn and minimize the cost-function simultaneously. Different from them, the Cost-Sensitive Decision Tree (Bahnsen et al., 2015) considers the classification cost during the induction and pruning of a decision tree, producing a tree that is natively cost-sensitive. The main difference between the Cost-Sensitive Decision Tree and the standard implementation of a decision tree is the replacement of the error measure (e.g. Gini coefficient) by a cost measure. Finally, a Cost-Sensitive Random Forest (Bahnsen, 2015) is obtained by creating different Cost-Sensitive Decision Trees on random subsamples of the training set, and then combining them by some approach (majority voting in this study).

3. Experiment design

Each dataset was splitted into train and test subsets, with 70% and 30% of the examples on each, respectively. To select the best hyperparameters from the search space, grid search was applied on using a 5-fold cross-validation. The classification metric used in this work was balanced accuracy. Four machine learning algorithms were used in this study: Decision Tree, Random Forest, kNN and AdaBoost (the implementation available in *Scikit-Learn* (Pedregosa et al., 2011) was adopted). For consistency with the original work on PMLB (see Section 4.2), which also used these algorithms, the same hyperparameter space used by the benchmark for the machine learning algorithms was adopted in this study (see Table 1). The datasets were standardized using *RobustScaler*.

Seven techniques for handling the class imbalance were compared, three resampling techniques and four cost-sensitive algorithms. The resampling techniques used were SMOTE, SMOTETomek and TomekLinks. The cost-sensitive algorithms evaluated were Thresholding, BayesMinimumRisk, Cost-Sensitive Decision Tree and Cost-Sensitive Random Forest. The implementations from the library *Imbalanced-Learn* (Lemaitre et al., 2017) were used for the resampling techniques and from *CostCla*[1] for the cost-sensitive algorithms. Table 2 shows the hyperparameter space for each technique. The values for *k_neighbors* were based on the work of Maciejewski and Stefanowski (2011). The values for the cost matrices

[1] http://albahnsen.github.io/CostSensitiveClassification/

Algorithm	Hyperparameter space
Decision Tree	*min_impurity_decrease* : $[0; 2.5 \times 10^{-4}; 5 \times 10^{-4}; ...; 5 \times 10^{-3}]$ *max_features* : [0.1; 0.25; 0.5; 0.75; 1.0; *sqrt; log2*] *criterion* : [*gini; entropy*]
Random Forest	*n_estimators*: [10; 50; 100; 500] *min_impurity_decrease* : $[0; 2.5 \times 10^{-4}; 5 \times 10^{-4}; ...; 5 \times 10^{-3}]$ *max_features* : [0.1; 0.25; 0.5; 0.75; 1.0; *sqrt; log2*] *criterion* : [*gini; entropy*]
kNN	*n_neighbors* : [1; 2; ... ; 24; 25; 50; 100] *weights* : [*uniform; distance*]
AdaBoost	*n_estimators* : [10; 50; 100; 500] *learning_rate* : [0.01; 0.1; 0.5; 1; 10; 50; 100]

Table 1: Hyperparameter space for each machine learning algorithm used in this study.

Technique	Hyperparameter space
SMOTE	*sampling_strategy* : *minority* *k_neighbors* : [3; 5; 7]
SMOTETomek	The same values used for SMOTE
TomekLinks	*sampling_strategy* : *majority*
Thresholding	Let $C(i, j)$ be the cost of predicting i as j: • $C(0, 0)$: 0 • $C(1, 1)$: 0 • $C(0, 1)$: 1 • $C(1, 0)$: [2; 5; 10; b], where b is inversely proportional to the Imbalance Ratio
BayesMinimumRisk	The same cost matrix (C) used by Thresholding
Cost-Sensitive Decision Tree	*cost_mat*: The same cost matrix (C) used by Thresholding *max_features*: [0.1; 0.25; 0.5; 0.75; 1.0; *sqrt; log2*] *criterion*: [*gini_cost; entropy_cost; direct_cost*] *min_gain*: $[0; 2.5 \times 10^{-4}; 5 \times 10^{-4}; ...; 5 \times 10^{-3}]$ *pruned*: *True*
Cost-Sensitive Random Forest	*cost_mat*: The same cost matrix (C) used by Thresholding *n_estimators*: $[10; 50; 100; 500]$ *max_features*: [0.1; 0.25; 0.5; 0.75; 1.0; *sqrt; log2*] *combination*: *majority_voting*

Table 2: Hyperparameter space for each technique used in this study for handling the class imbalance.

were based on Domingos (1999) (one configuration was added, which is the cost inversely proportional to the Imbalance Ratio, since it is commonly used). Whenever possible, the machine learning algorithms hyperparameter values were replicated on its cost-sensitive implementations.

4. Datasets

This section describes the datasets used in the study. Section 4.1 introduces synthetic datasets with controlled imbalance ratios, class overlap and class decomposition. Section 4.2 presents and characterizes the Penn Machine Learning Benchmark (PMLB), which is used to validate and complement the analysis on the synthetic datasets. Section 4.3 presents a brief comparison of them.

4.1. Synthetic datasets

The synthetic datasets used in this study were part of a previous research and are publicly available on GitHub [2]. Synthetic datasets with various imbalance ratios, levels of class overlap and minority class decomposition were obtained. In total, 135 datasets were created, each with 10,000 examples, 2 continuous predictor variables and a binary target variable. The majority class was drawn from a uniform distribution and covers the entire space around and below the minority class. The minority class is composed of 2, 4 or 8 subconcepts (as in Jo and Japkowicz (2004)). Here, each subconcept is a circular cluster obtained from a two-dimensional Gaussian distribution with radius $r \in \{0.25; 0.5; 1.0\}$ (all subconcepts in the same dataset have the same r). Different levels of class overlap were produced by establishing a exclusion region within each subconcept. The exclusion region is an area within the subconcept, concentric with the subconcept, where only minority examples exist. The size of the exclusion region is determined by its radius, measured in standard deviations of the subconcept Gaussian distribution, and the levels varied in this study were $e \in \{0, 1, 2\}$. The imbalance ratio was also varied between 1:1, 1:2, 1:4, 1:9 and 1:99 (5 of the 9 levels varied in Batista et al. (2005), including the minimum and maximum). Figure 1 illustrates two datasets with the same imbalance ratio, 4 subconcepts with the same radius, but varying the exclusion radius. Note the halo around each subconcept in Figure 1(b) that results from a high exclusion radius, while in Figure 1(a) the exclusion radius was set to zero and the two classes completely overlap.

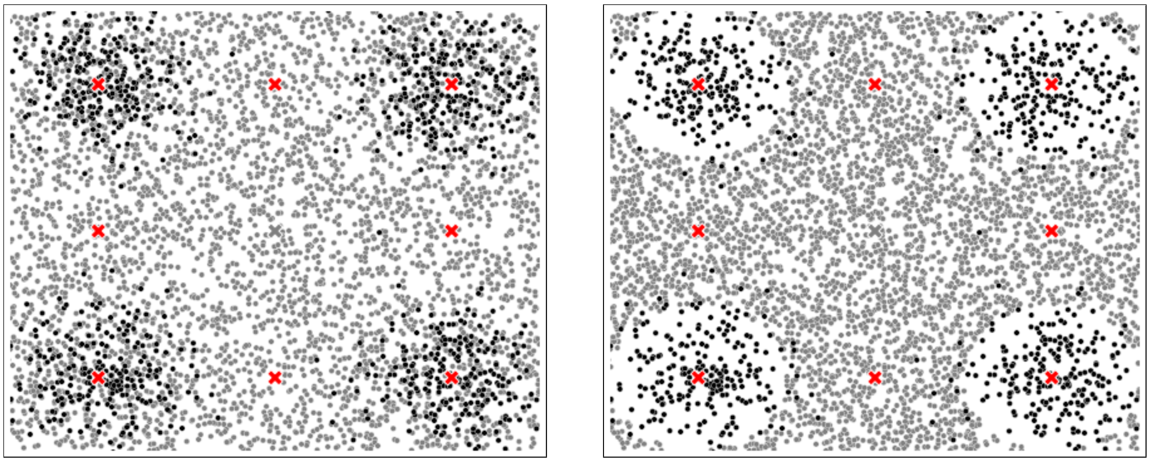
4.2. Penn Machine Learning Benchmark (PMLB)

All datasets marked as appropriate for binary classification on the Penn Machine Learning Benchmark (PMLB) (Olson et al., 2017) were used in this study, which totals 90 datasets. The PMLB datasets are very diverse with respect to the quantity of examples and predictors, as shown in Figure 2(a). Most of the PMLB datasets are imbalanced, but the imbalance ratio is usually not extreme (Figure 2(b)).

4.3. Comparison of synthetic datasets and PMLB

In addition to the difference in quantity of examples and predictor variables (that are fixed on the synthetic datasets and vary on PMLB) and imbalance ratio, other characteristics of the data are relevant to evaluate the obtained results. Two of them are class overlap and decomposition. Since these features are hard to obtain from real-world datasets, the methodology from Han et al. (2005), that classifies each example as safe, borderline or

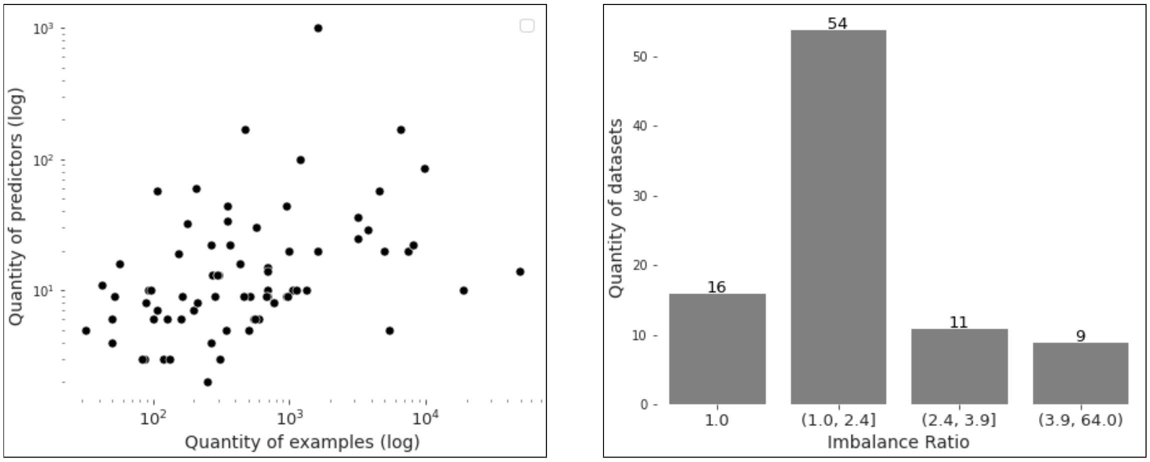
[2] https://github.com/jairojuunior/cost_sensitive_vs_resampling



(*a*) Exclusion radius: 0 std.

(*b*) Exclusion radius: 2 std.

Figure 1: Two synthetic datasets with Imbalance Ratio of 1:4, 4 subconcepts on the minority class, each with a radius of 1.0 and varying exclusion radius (e). Red x markers indicate the center of the subconcepts (including where additional subconcepts would be placed).

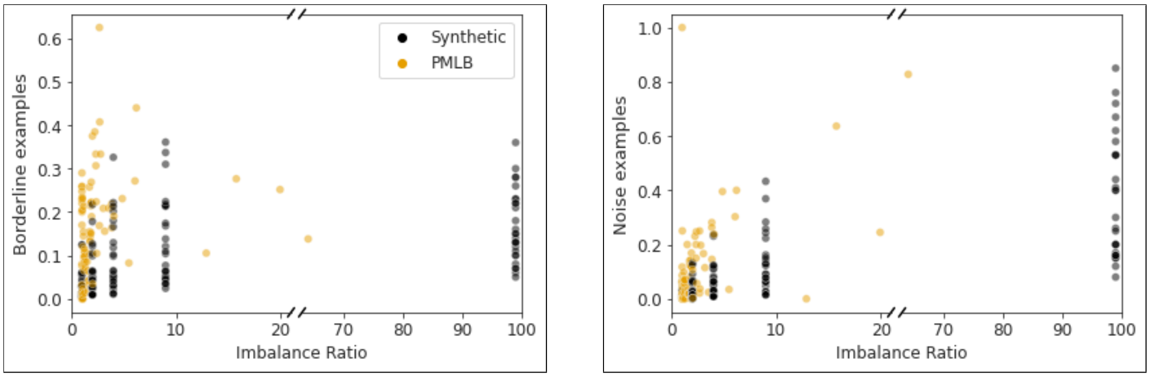


(*a*) Quantity of examples and predictors.

(*b*) Imbalance Ratio.

Figure 2: Characterization of PMLB datasets with respect to the quantity of predictors and examples and imbalance ratio.

noise from its local neighborhood, was adopted. In our case, if 3 or more out the 5-nearest neighbors of the minority example were also of the minority class, it was classified as safe. If just 1 or 2 of its neighbors are of the minority class, then it was labeled as bordeline. Otherwise, it was classified as noise. Figure 3 shows the variation of borderline and noise examples as a function of the imbalance ratio. The results show that, for the imbalance ratio interval between 1:1 and 1:4, where most of PMLB datasets are concentrated, the variation in borderline and noise examples is similar between PMLB and synthetic datasets. It was also observed a tendency for higher variation in the percentage of noise examples as the imbalance ratio increases.



(*a*) Borderline examples

(*b*) Noise examples

Figure 3: Scatterplot of borderline and noise examples per imbalance ratio for synthetic and PMLB datasets.

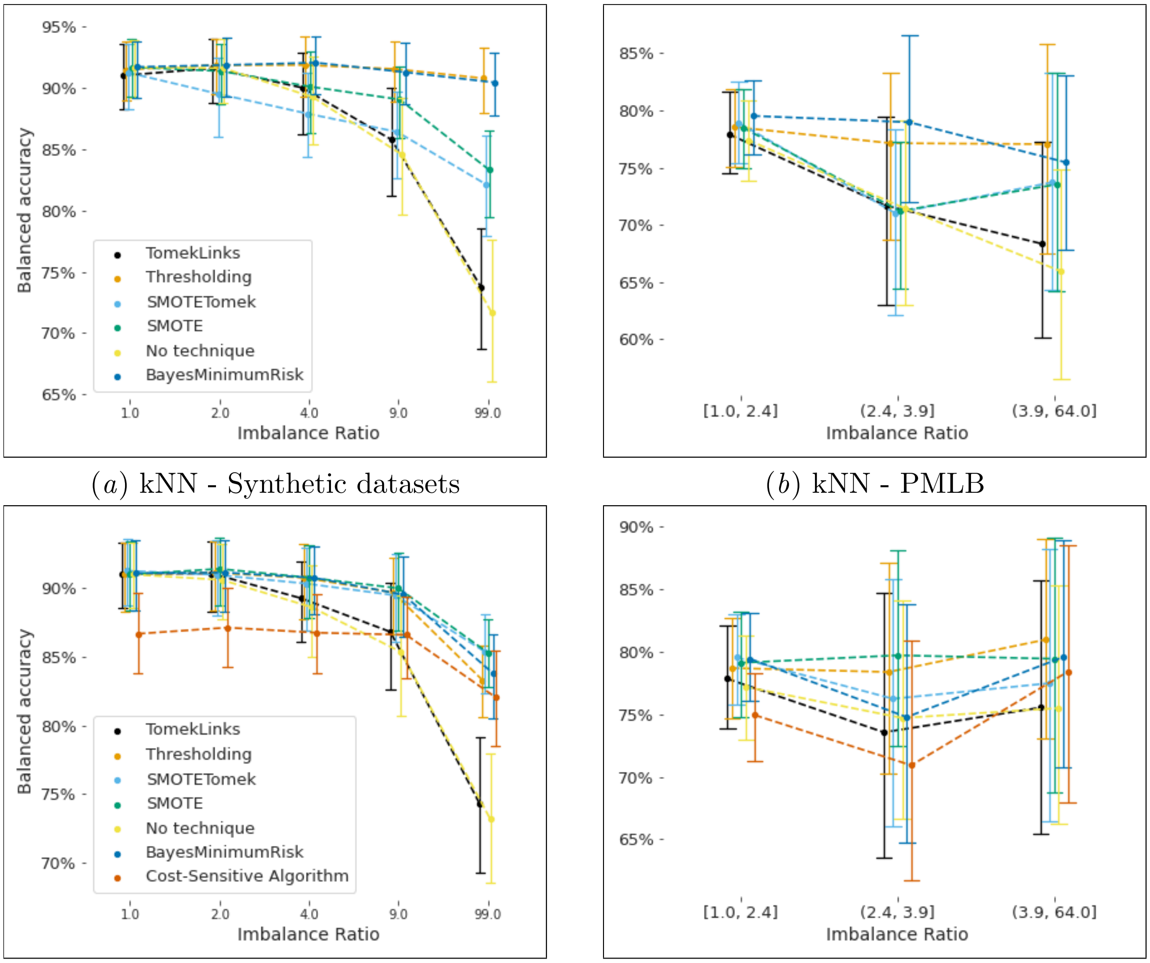
5. Results

This section presents and discusses the obtained results. In Figures 4-6, each point represent the mean of the results and its confidence interval for $\alpha = 0.05$ obtained via bootstrap. The results on the synthetic datasets and PMLB are always displayed side-by-side. Since the PMLB datasets are much more heterogeneous, the confidence interval on the PMLB results are always higher than on the synthetic datasets. The results are organized as follows: Section 5.1 compares the performance of the techniques for handling the class imbalance on each machine learning algorithm; Section 5.2 analyzes the impact of the techniques to handle class imbalance on model complexity. Part of the results presented in Section 5.1 are from the research carried out during an undergraduate project (Freitas Júnior, 2022), which discusses additional comparative performance analysis with respect to other dataset parameters (e.g. exclusion radius on synthetic datasets).

5.1. Predictive performance comparison

Figure 4(*a*) shows the balanced accuracy of kNN combined with the techniques for handling the class imbalance, as a function of the imbalance ratio, on synthetic datasets, and

Figure 4(*b*) presents the results obtained on PMLB. On both datasets, BayesMinimumRisk and Thresholding tended to obtain superior relative performance. Following them were SMOTE and SMOTETomek, which seemed to be favored by higher imbalance ratios. Finally, TomekLinks performance was close to the non-application of a technique (this behavior was observed on all machine learning algorithms). It will be shown in the next pages that cost-sensitive algorithms combined with kNN yielded models with higher values of k-neighbors (usually by a factor of 2) compared to resampling techniques, what possibly produced more reasonable probabilities for the cost-sensitive algorithms, explaining its high balanced accuracy. On the other hand, Zhang and Li (2011) showed that oversampling techniques (including SMOTE) can increase the variance errors of kNN, which may explain the intermediate performance of SMOTE and SMOTETomek.



(*a*) kNN - Synthetic datasets

(*b*) kNN - PMLB

(*c*) Decision Tree - Synthetic datasets

(*d*) Decision Tree - PMLB

Figure 4: Balanced accuracy as a function of the Imbalance Ratio for different combinations of techniques for handling the class imbalance with **kNN** and **Decision Tree**. In this Figure, the "Cost-Sensitive Algorithm" is the Cost-Sensitive Decision Tree.

Figure 4(c) shows the balanced accuracy of Decision Tree combined with the techniques for handling the class imbalance, as a function of the imbalance ratio, on synthetic datasets, and Figure 4(d) presents the results obtained on PMLB. SMOTE and Thresholding had relative superior performance on both datasets, even though the most extreme imbalance on synthetic datasets (1:99) had a high impact on Thresholding. The Cost-Sensitive Decision Tree (line "Cost-Sensitive Algorithm" on the plot) achieved competitive balanced accuracy only on datasets with high imbalance ratios. Decision Trees are known to produce poor estimates of class probabilities (Provost and Domingos, 2003). It possibly explains why Thresholding, a technique that relies on a single classification threshold, outperformed BayesMinimumRisk in this experiment. Previously, Chawla (2003) found that SMOTE has a tendency to improve Decision Tree classification performance over other resampling techniques. A similar tendency was also observed in our results.

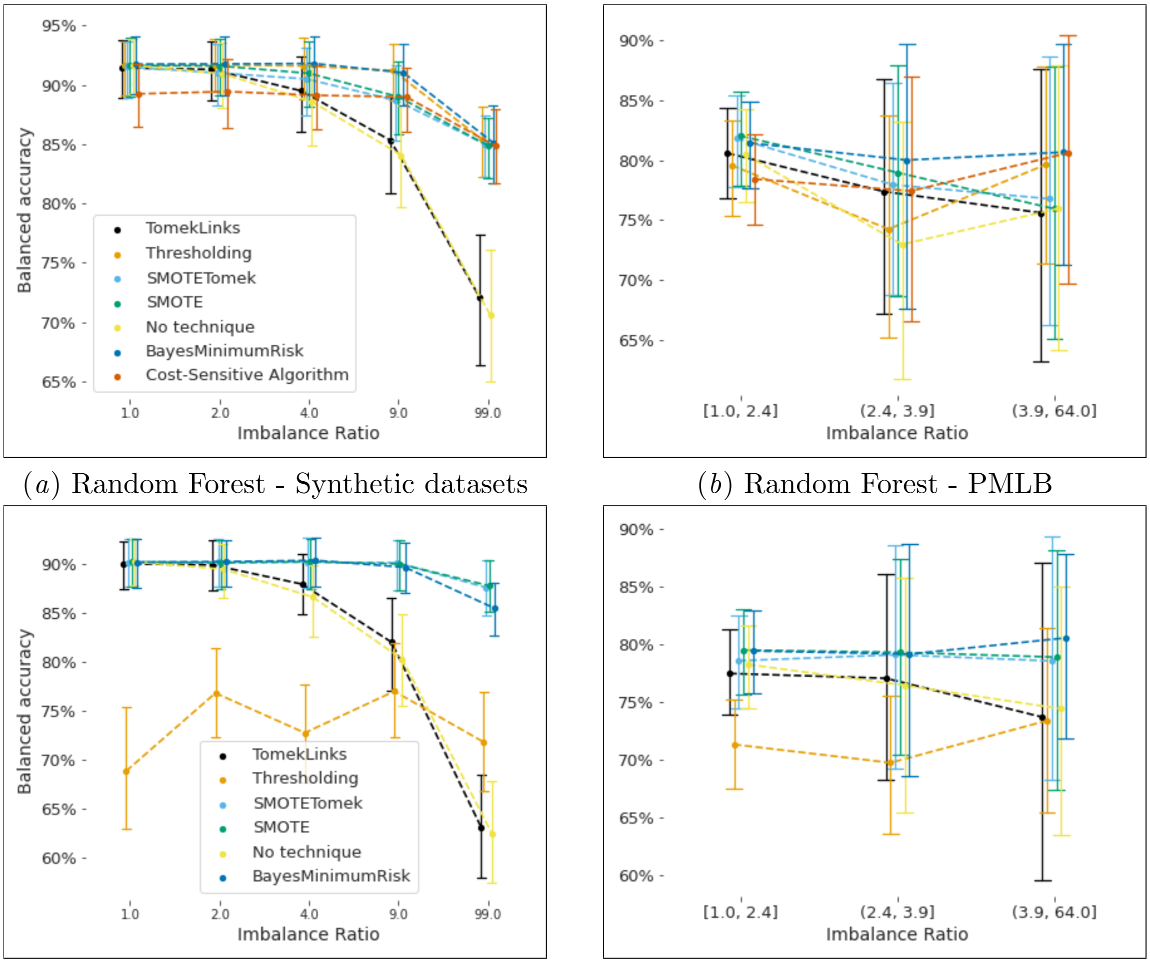
Figures 5(a) and 5(b) present the results for the experiments with Random Forest. BayesMinimumRisk showed a tendency for higher relative results on both synthetic (Figure 5(a)) and PMLB (Figure 5(b)) datasets. The Cost-Sensitive Random Forest, similar to the Cost-Sensitive Decision Tree from Figures 4(c) and 4(d), had a relative low performance on more balanced datasets. However, the Cost-Sensitve Random Forest started achieving competitive results on datasets with lower imbalance ratio (1:9 versus 1:99 on synthetic datasets and $(2.4, 3.9]$ versus $(3.9, 64.0]$ on PMLB). Even though Random Forest may not produce perfectly calibrated models, it's known to be one of the best learning algorithms to produce well-calibrated probabilities (Niculescu-Mizil and Caruana, 2005). This possibly explains the performance of BayesMinimumRisk on this experiment. The increase in relative performance of the Cost-Sensitive Random Forest compared to the Cost-Sensitive Decision Tree may be the result of ensembling, which tends to reduce variance while maintaining the low bias of weak learners.

The results obtained with AdaBoost are shown in Figures 5(c) and 5(d). BayesMinimumRisk, SMOTE and SMOTETomek obtained similar balanced accuracy and superior results compared to the other techniques. Thresholding showed a tendency for lower results than the non-application of a technique when combined with AdaBoost, which was not observed in the other machine learning algorithms. The models created with AdaBoost combined with Thresholding and other techniques were compared, and it was found that the Thresholding models had, on average, fewer estimators and, on PMLB datasets, a much lower learning rate (which controls the contributions of consecutive estimators to the ensemble prediction). At the moment of writing this paper, we had not reached a satisfactory hypothesis to explain why the grid-search with cross-validation performed in the experiments led to this result.

The findings from PMLB and synthetic datasets presented in this section overall agreed. A higher confidence interval was observed on PMLB results, which may be a consequence of the higher heterogeneity of its datasets. Since the synthetic datasets have a higher similarity among them, it tends to produce less noisy results.

5.2. Model complexity analysis

This section presents an investigation of the mean complexity of the models built by the combination of each algorithm and technique for handling the class imbalance. The com-



(*a*) Random Forest - Synthetic datasets

(*b*) Random Forest - PMLB

(*c*) AdaBoost - Synthetic datasets

(*d*) AdaBoost - PMLB

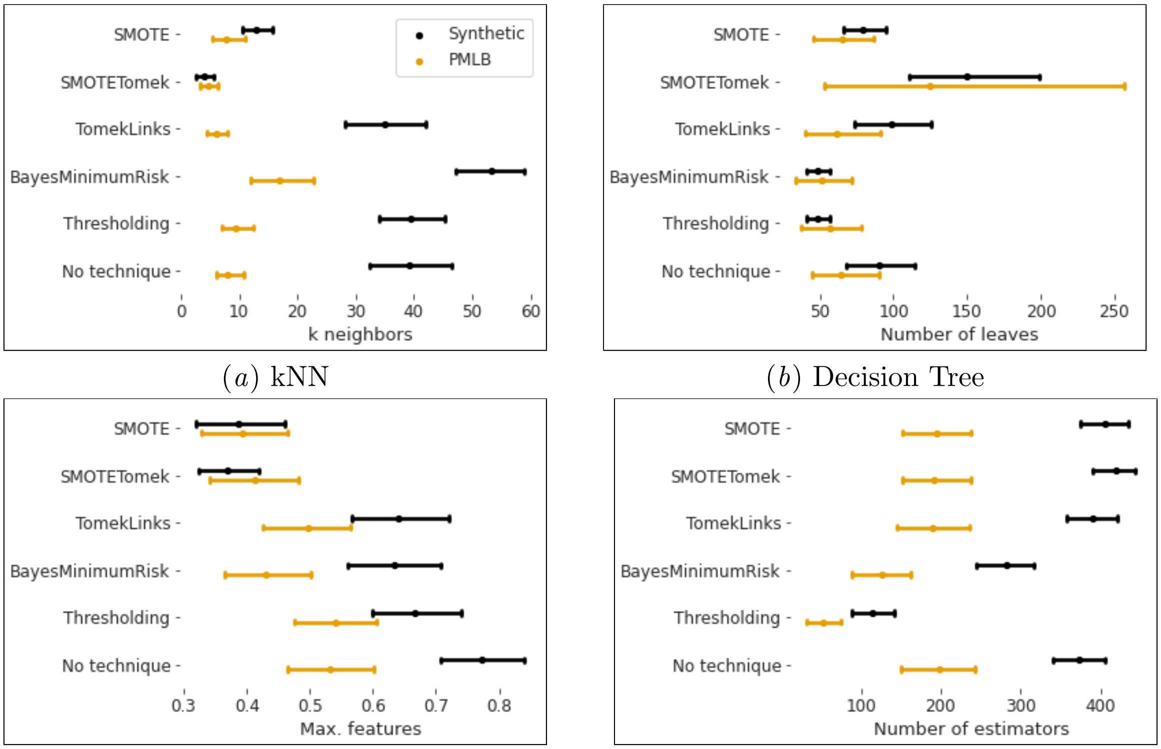
Figure 5: Balanced accuracy as a function of the Imbalance Ratio for different combinations of techniques for handling the class imbalance with **Random Forest** and **AdaBoost**. In this Figure, the "Cost-Sensitive Algorithm" is the Cost-Sensitive Random Forest.

plexity of kNN-based models was evaluated in terms of the value of k-neighbors. The Random Forest complexity was measured by the maximum number of features considered per split (*max_features*). AdaBoost-based models complexities were approximated by the number of estimators, and the Decision Tree-based models complexities were measured by the number of leaves of the trees.

Figure 7(*a*) shows the mean value of k-neightbors when kNN was combined with each technique for handling the class imbalance. Cost-senstive algorithms (BayesMinimumRisk and Thresholding) showed a tendency for producing kNN-based models with higher values of k-neighbors when compared to oversampling techniques (SMOTE and SMOTETomek) in both synthetic and PMLB datasets. The value of k-neighbors for the non-application of a technique for handling the class imbalance was similar to the value for Thresholding. While, in the PMLB datasets, the value of k-neighbors for TomekLinks was close to the other resampling techniques. This pattern was not observed in the synthetic datasets.

Before the application of SMOTE and SMOTETomek, most datasets had the majority of non-safe examples as borderline type. The creation of synthetic examples around borderline examples increases the likelihood of turning them safe, and, as a result, it can make the nearest neighborhood more reliable for the classifier. This might explain why the value for k-neighbors was low when these techniques were combined with kNN. However, when the cost-sensitive algorithms are applied, a larger neighborhood might be needed to reach enough minority examples to estimate the class probabilities. It was also noted that there was an equilibrium in the occurence of choices of uniform and distance based weights for resampling techniques, but more than 90% of the kNN-based models combined with BayesMinimumRisk or Thresholding used uniform weights. The choice of uniform weights for higher values of k-neighbors sounds reasonable, since distant minority examples could be underweighted otherwise.

Figure 6: Mean complexity of the models produced by the combination of machine learning algorithms and different techniques for handling class imbalance.



(a) kNN

(b) Decision Tree

(c) Random Forest

(d) AdaBoost

Figure 7(b) shows the mean number of leaves when Decision Tree was combined with each technique for handling the class imbalance. Decision Tree-based models combined with cost-sensitive algorithms (BayesMinimuRisk and Thresholding) tended to have a lower number of leaves compared to resampling techniques (SMOTE, SMOTETomek and TomekLinks). This pattern was more distinguished on the synthetic datasets.

To better understand this result, let the probability estimate by a Decision Tree for class i on leaf m be $p^i_m = |i_m|/|m|$, where $|i_m|$ is the number of examples of class i on leaf m and $|m|$ is the number of examples in m. If the number of examples in a dataset is fixed and the number of leaves in a tree decreases, then on average $|m|$ and $|i_m|$ increases. By the central limit theorem, if the sample size increases, the probability estimate tends to be more reliable. It might explain why Decision Tree-based models combined with cost-sensitive algorithms tended to have a lower number of leaves, although it needs more investigation before coming to this conclusion.

For the Random Forest-based models (Figure 7(c)), when combined with oversampling techniques (SMOTE and SMOTETomek), a tendency for a lower value of *max_features* was observed. BayesMinimumRisk showed a tendency for lower values of *max_features* on PMLB, while, on the synthetic datasets, the results were more aligned with the ones from Thresholding and TomekLinks. The maximum number of features considered per split is fundamental to generate diverse enough trees to benefit from averaging them (Sage et al., 2020). The lower its value, the higher is the randomization in the induction and, therefore, less similar the estimators tend to be. Since SMOTE (and for extension SMOTETomek) introduces correlation between some examples (Blagus and Lusa, 2013), the examples on each sampled subset used to train each estimator tend to be more correlated with the ones from other subsets than when using non-oversampled datasets, leading to more similar trees. Reducing *max_features*, in the case of SMOTE and SMOTETomek, possibly counteracts the increased similarity of subsets, producing a better final model.

Resampling techniques (SMOTE, SMOTETomek and TomekLinks) showed a tendency for producing AdaBoost-based models with higher number of estimators, as shown in Figure 7(d). In contrast, cost-sensitive algorithms combined with AdaBoost tended to have a lower number of estimators. The number of estimators for the non-application of a technique was comparable to the observations for the resampling methods.

AdaBoost-SAMME (Hastie et al., 2009) iteratively train additional classifiers on copies of the original dataset with more weights on the previously incorrectly classified instances (hard examples). Models with high number of estimators might have their probabilities skewed by the hard examples. Our hypothesis is that AdaBoost-based models combined with cost-senstive algorithms tend to favor a lower number of estimators to get less skewed probabilities.

A difference in complexity was also noted between Thresholding and BayesMinimumRisk when combined with kNN and AdaBoost, where BayesMinimumRisk models tended to have higher values of k-neighbors and number of estimators. Our hypothesis for this result is that, since Thresholding relies on a single threshold to separate the classes, the importance of probability calibration for it is smaller than for BayesMinimumRisk, reason why we believe the latter tends to be more complex.

The results from Figure 6 show that, overall, the technique used for handling the class imbalance can influence the complexity of the machine learning model. A consequence from this finding is the realization that the hyperparameter search space should be carefully chosen depending on the combination of learning algorithm and technique for handling class imbalance. This, however, does not mean that doing otherwise will result in low performance. In fact, the association between the complexity of these models with overfitting or underfitting was not addressed here and is a topic for future research.

6. Conclusions and future work

In this paper, the performance of cost-sensitive algorithms and resampling techniques were compared when combined with four machine learning algorithms. For kNN, a better performance of BayesMinimumRisk and Thresholding was observed, and these techniques also tended to yield models with higher values of k-neighbors. Decision Tree achieved better results with SMOTE, SMOTETomek and Thresholding, and cost-sensitive algorithms combined with Decision Trees resulted in a lower number of leaves. Random Forest showed better results with BayesMinimumRisk, and oversampling techniques combined with Random Forest tended to produce models with lower values of maximum number of features per split. Finally, BayesMinimumRisk, SMOTE and SMOTETomek achieved similar performance and superior results compared to other techniques on AdaBoost. A lower number of estimators was also observed on AdaBoost models combined with cost-sensitive algorithms.

Our results showed that the techniques for handling the class imbalance may influence the complexity of machine learning models, including the choice of hyperparameters. To the best of our knowledge, this study is the first to show it for these techniques. On future works, additional learning algorithms and techniques for handling the class imbalance can be explored. The impact of model calibration could also be evaluated, since it was previously shown that cost-sensitive algorithms tend to benefit from it (Bahnsen et al., 2014). Finally, extending the current study to multi-class settings is another alternative which could be investigated in future work.

References

Alejandro Correa Bahnsen. *Example-Dependent Cost-Sensitive Classification with Applications in Financial Risk Modeling and Marketing Analytics.* PhD thesis, University of Luxembourg, 2015.

Alejandro Correa Bahnsen, Aleksandar Stojanovic, Djamila Aouada, and Björn Ottersten. Improving credit card fraud detection with calibrated probabilities. In *Proceedings of the 2014 SIAM international conference on data mining*, pages 677–685. SIAM, 2014.

Alejandro Correa Bahnsen, Djamila Aouada, and Björn Ottersten. Example-dependent cost-sensitive decision trees. *Expert Systems with Applications*, 42(19):6609–6619, 2015.

Gustavo EAPA Batista, Ana LC Bazzan, Maria Carolina Monard, et al. Balancing training data for automated annotation of keywords: a case study. In *WOB*, pages 10–18, 2003.

Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

Gustavo EAPA Batista, Ronaldo C Prati, and Maria C Monard. Balancing strategies and class overlapping. In *International Symposium on Intelligent Data Analysis*, pages 24–35. Springer, 2005.

Rok Blagus and Lara Lusa. Smote for high-dimensional class-imbalanced data. *BMC bioinformatics*, 14(1):1–16, 2013.

Nitesh V Chawla. C4. 5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *Proceedings of the ICML*, volume 3, page 66. CIBC Toronto, ON, Canada, 2003.

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.

Pedro Domingos. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, page 155–164, New York, NY, USA, 1999. Association for Computing Machinery. ISBN 1581131437. doi: 10.1145/312129.312220.

Jairo da Silva Freitas Júnior. Comparativo de técnicas de aprendizado sensível ao custo e reamostragem em conjuntos de dados desbalanceados. Bachelor's thesis, Federal University of ABC, 2022.

Jayanta K Ghosh, Mohan Delampady, and Tapas Samanta. Bayesian inference and decision theory. *An Introduction to Bayesian Analysis: Theory and Methods*, pages 29–63, 2006.

Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220–239, 2017.

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360, 2009.

Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1):40–49, 2004.

Guillaume Lemaitre, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

Victoria López, Alberto Fernández, Jose G. Moreno-Torres, and Francisco 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. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2011.12.043.

T. Maciejewski and J. Stefanowski. Local neighbourhood extension of smote for mining imbalanced data. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 104–111, 2011. doi: 10.1109/CIDM.2011.5949434.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 625–632, 2005.

Randal S Olson, William La Cava, Patryk Orzechowski, Ryan J Urbanowicz, and Jason H Moore. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData mining*, 10(1):1–13, 2017.

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.

Ronaldo C Prati, Gustavo EAPA Batista, and Maria Carolina Monard. Class imbalances versus class overlapping: an analysis of a learning system behavior. In *Mexican international conference on artificial intelligence*, pages 312–321. Springer, 2004.

Foster Provost and Pedro Domingos. Tree induction for probability-based ranking. *Machine learning*, 52(3):199–215, 2003.

Andrew J Sage, Ulrike Genschel, and Dan Nettleton. Tree aggregation for random forest class probability estimation. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 13(2):134–150, 2020.

Chris Seiffert, Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. A comparative study of data sampling and cost sensitive learning. In *2008 IEEE international conference on data mining workshops*, pages 46–52. IEEE, 2008.

Victor S Sheng and Charles X Ling. Thresholding for making classifiers cost-sensitive. In *AAAI*, pages 476–481, 2006.

Jerzy Stefanowski. Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data. In *Emerging paradigms in machine learning*, pages 277–306. Springer, 2013.

Ivan Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772, 1976. doi: 10.1109/TSMC.1976.4309452.

Gary M Weiss, Kate McCarthy, and Bibi Zabar. Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? *Dmin*, 7(35-41): 24, 2007.

Xiuzhen Zhang and Yuxuan Li. An empirical study of learning from imbalanced data. In *Proceedings of the Twenty-Second Australasian Database Conference-Volume 115*, pages 85–94, 2011.