A Global comparison of solutions in The Imbalanced Learning

Michal Dawid Kowalski
University of Porto
Porto, Portugal
up202401554@up.pt

ABSTRACT

Class imbalance is a common problem faced by scientists working with data in many fields. This phenomenon occurs, among others, in healthcare diagnosing rare diseases, environment exploring rare weather events, legal predicting rare legal outcomes, as well as finance, fraud detection, marketing or technology. The classification of imbalanced data is a new problem that rises in the machine learning framework and it is the a significant problem raised for the researches and the use of various sampling techniques is necessary to improve classification performance [1]. This report presents the global comparison of several selected methods for handling imbalanced dataset and evaluate their impact on classifier performance.

1 INTRODUCTION

The imbalance of the data is defined as a dataset whose proportion of classes is severely skewed. Classification performance of existing models tends to deteriorate due to class distribution imbalance [2]. In that case handling mentioned data issues can be a challenging task in many classification cases. This type of data is characterized by having a minority class that significantly differs in the number of records compared to the remaining classes, called majority classes. For example, a database imbalance can be considered a case with the low *Imbalance Ratio* (IR), where the minority class is only 10% of the data cases compared to 90% belonging to the majority class.

Basically, these heavily skewed datasets are common in real-world examples, making it crucial to develop effective solutions to reduce their impact on computational processes and models predictions. Learning from unevenly distributed samples can decrease both accuracy and reliability from the trained model [3]. ML models trained on imbalanced datasets often become biased toward the majority class, resulting in weak recognition or misclassifications of the minority class.

Solutions for imbalanced data can be applied at two levels: data-level, introducing preprocessing techniques called: undersampling, oversampling, or hybrid sampling, and algorithm-level, using algorithms such as cost-sensitive or ensemble models optimized for imbalanced datasets [4]. Following to the preprocessing, bunch of python libraries can be used for this purpose, like *imblearn* or *smote_variants*. This became the objective of the research into the use of various techniques using mentioned libraries.

2 METHODOLOGY

This section describes the approach and all the steps taken to compare solutions in the Imbalanced Learning. The techniques such as *RandomOversampling*, *RandomUndersampling*, *SMOTE* and different variants of *SMOTE* were identified and analyzed in that exercise.

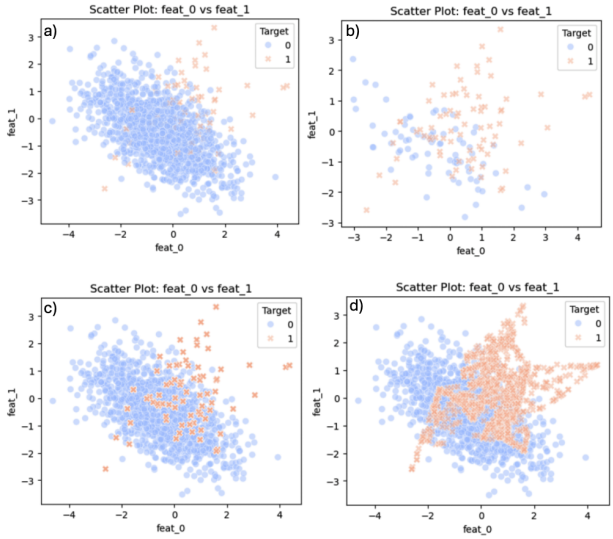

Figure 1: Class Distribution a) before, b) after Random Under-Sampling, c) after Random OverSampling , d) after SMOTE

2.1 Dataset

The dataset for this assignment was synthetically generated using a function *make_classification* from *sklearn.datasets*. After setting the appropriate parameters, the created database contains 2000 samples, 2 classes (0,1), 4 features, of which 3 are informative and 1 is redundant to make a problem complex.

The *class_sep* parameter is set to 0.5 to introduce overlapping between the classes, while the class weights are set to [0.95, 0.05], meaning class 0 (the majority class) significantly outweighs class 1 (the minority class). The resulting *Imbalance Ratio* (IR) is approximately 18.05. Both of imbalance and overlapping data points complicates the task of differentiating between them. The model must address both the class imbalance and the lack of clear separation in the feature space. Before training the classifier model, the data was split into training and test data in a ratio of 70/30.

2.2 Metrics

To assess the model performance after applying different techniques in training data preprocessing, common metrics are used:

- *Precision* - when the cost of false positives is high,
- *Recall* - when the cost of false negatives is high,
- *F1 Score* - the combination of the Precision and Recall (balance between them),

Table 1: Metric for the Global comparison of solutions in the Imbalanced Learning using RF Model

Metrics	Non-Bal.	UnderSampl.	OverSampl.	SMOTE	Border-SMOTE	SMOTE-Tomek	ADASYN	SMOTE-OUT
Precision	0.87	0.19	0.58	0.30	0.39	0.25	0.25	0.29
Recall	0.43	0.77	0.50	0.67	0.57	0.67	0.70	0.70
F1-Score	0.58	0.30	0.54	0.42	0.46	0.36	0.37	0.41
Accuracy	0.97	0.82	0.96	0.91	0.93	0.88	0.88	0.90

- *Accuracy* - it can be not ideal, because even with a high value for majority class recognition it can also have issues to identify any instances of the minority class.

2.3 Model Selection

To test the algorithms, a *Random Forest Classifier* model was implemented using the *sklearn* library. The RF classifier was chosen because it is a robust model which can handle both imbalanced data and complex features relations well. Its ability to perform feature selection and handle overfitting makes it a proper model for this task.

2.4 Solutions in the Imbalanced Datasets

The *imblearn* Python package allows for the implementation of various resampling techniques to handle the class imbalance, such as *RandomOversampling, RandomUndersampling* and *SMOTE*, which leads to balance the distribution of classes in the training data.

The *smote-variants* Python package aims to support research and applications by offering 85 different oversampling techniques [4] . It provides a simple way to create synthetic samples for the minority class, also helping to improve model performance on imbalanced datasets. So, from these tools, *Borderline-SMOTE, SMOTE+TomekLinks, ADASYN* and *SMOTEOUT* were chosen to check their impact on the classification.

Borderline-SMOTE creates synthetic samples near the decision boundary, SMOTE+TomekLinks combines SMOTE to generate new samples with TomekLinks to remove borderline instances that are difficult to classify, ADASYN generates more synthetic samples for the minority class based on where the data points are hard to classify, SMOTEOUT removes noisy points from the minority class and creates synthetic samples from the remaining data.

3 RESULTS

Initially, the classifier model was trained using an unbalanced database to provide a baseline for further research. Then, data resampling techniques were performed using functions provided by the mentioned python libraries.

The class distribution of the dataset both before and after preprocessing is shown in Figure 1. The raw data is obviously unbalanced and exhibits a great deal of class overlap prior to any modifications. After applying Random Undersampling, the number of majority class instances is reduced to the number of minority class instances, but all SMOTE algorithms generate synthetic data to balance the minority class with the majority. In contrast, Random Oversampling method duplicates existing instances from the minority class to achieve dataset balance.

The results in Table 1 shows the impact of resampling algorithms on model performance compared to the non-balanced case. The baseline showed the highest precision (0.87) and accuracy (0.97), but it struggled with recall (0.43), indicating a poor minority class extraction, what may be problematic during the classification process. All resampling methods reduced precision metric due to changes in class distribution, which impacted the model's accuracy in positive predictions while balancing recall to improve minority class detection.

Undersampling significantly increased recall (0.77), but reduced precision (0.19) and accuracy (0.82), making this method unsuitable for a proper performance with the lowest f1-score (0.30). Probably model was trained with not sufficient volume of the data. In contrast, Random Oversampling method led to only moderate gains comparing with a baseline, maintaining high accuracy (0.96), which can be caused by bunch of repeated duplications of minority samples. This tendency often leads to the model overfitting.

SMOTE techniques offered a better trade-off, improving recall and F1-score while maintaining precision in range between (0.25-0.39). The SMOTE techniques tested showed various results compared to the baseline. SMOTE improved recall significantly, reaching 0.67, but precision dropped to 0.30, showing it is more sensitive to the minority class, though with more false positives. Borderline-SMOTE, which is basically focused on samples nearby the decision boundary, had a better precision of 0.39 and improved recall to 0.57 as well. SMOTE+TomekLinks, which also removes noisy instances from the majority class, improved recall to 0.67, but precision stayed low at 0.25. ADASYN achieved the best recall at 0.70, but still struggled with precision, similar to other techniques. SMOTEOUT, which combines SMOTE with outlier removal, showed improved recall to 0.70 while also maintaining a precision of 0.29. All approaches, taken together, increased recall but also reduced precision, highlighting the trade-off in dealing with imbalanced datasets.

In essence, SMOTE helps improve recall by balancing the dataset, but it can lower precision because the synthetic data may not always perfectly represent the minority class, leading to misclassifications.

4 CONCLUSIONS

Resampling methods are powerful tools to improve classifiers performance, mostly enhancing their ability to classify the minority class. The comparison of various resampling techniques, including SMOTE-based methods and random resampling strategies, clearly highlighted the trade-offs between recall and precision.

While SMOTE and its hybrids improved model's recall, they often resulted in a decrease in precision. Random Undersampling helped balance the class distribution but caused a drop in precision due to reduced representation of the majority class. Random

oversampling improved precision but did not achieve optimal results, because of many duplicates. The choice of resampling method ultimately depends on the objectives of the task: if improving recall is the priority, SMOTE-based techniques seem to be suitable, while if precision is more important, Random Oversampling or Borderline-SMOTE might offer a better balance, while also improving the classification of the minority class. Selecting appropriate data preprocessing techniques should be preceded by testing the classifier model.

REFERENCES

[1] Spelmen Vimalraj and Porkodi Dr.R. A review on handling imbalanced data. 12 2018.

[2] Joonho Gong and Hyunjoong Kim. Rhsboost: Improving classification performance in imbalance data. *Computational Statistics & Data Analysis*, 111, 02 2017.

[3] Vitor Werner de Vargas, Jorge Aranda, Ricardo Costa, Paulo Pereira, and Jorge Barbosa. Imbalanced data preprocessing techniques for machine learning: a systematic mapping study. *Knowledge and Information Systems*, 65:1–27, 11 2022.

[4] J Zhang, X Cui, J Li, and R Wang. Imbalanced classification of mental workload using a cost-sensitive majority weighted minority oversampling strategy. *Cognition, Technology & Work*, 19:633–653, 2017.