**Binary Imbalanced Dataset Classification and Resampling Solutions**

Group member: Mingyue Li, Yingjie Qiu

**Introduction**

In the field of machine learning, classification is one of the most essential research areas to study. In particular when it comes to dealing with data that is evenly distributed, the traditional classification methods are considered to be reasonably well developed. However, when the data from the real world are imbalanced — that is, when the sample size in one category of the outcome is much larger than that of the other categories — the conventional methods of classification will not function as well as they normally would. To put it another way, an imbalance in the data occurs when the sample sizes of the various data classes are not distributed equally [1].

When working on bioinformatics categorization problems, it is common to come across imbalanced datasets. This means that the number of negative samples is significantly higher than the number of positive samples. Specifically, the phenomenon of data imbalance will cause us to grossly underestimate the performance of the minority class of positive examples. The majority of classifiers perform admirably with datasets that are well-balanced but struggle when the datasets contain inconsistencies. In this kind of scenario, classification techniques have a tendency to favor the majority, and as a result, they have a low rate of accuracy when applied to minority groups [2].

According to Haibo, various techniques have been proposed to solve the class imbalance problems, which divided into four categories: (1) data-level strategies, such as resampling or combinations; (2) algorithmic strategies, such as cost-sensitive and boosting; (3) feature selection; and (4) ensemble level, such as neural networks and kernel classifier [3]. The first strategy rebalances the lopsided class distribution by either over-sampling the minor class, under-sampling the major class, or combining these two strategies. This can be done by either under-sampling the major class or over-sampling the minor class. The second strategy frequently involves the development or modification of algorithms in order to gain a greater understanding of the minority group.

In this project, we deal with binary imbalanced datasets by data-level strategies using five resampling methods: Random Over-Sampling Algorithm (ROS), Synthetic Minority Over-sampling Technique (SMOTE), Synthetic Minority Over-sampling Technique- Nominal Continuous (SMOTE-NC), Random Under-Sampling Algorithm (RUS), and Under-Sampling Based on Clustering Algorithm (SBC) [4-5]. The first three strategies are examples of oversampling, while the remaining techniques are examples of under sampling. Our objective is to assess how well each of the five resampling strategies performs and how efficient they are in the classification of “Abalone” dataset.

**Resampling Methods**

*Random Over-Sampling*

The Random Over-Sampling (ROS) algorithm is responsible for the generation of new samples by randomly duplicating minority samples with replacement in each instance. The sample size of the majority class and the intended percentage of the size of majority samples both play a role in determining the number of replications to be performed. It is planned to produce a collection that is completely balanced (i.e., the imbalanced ratio will be 1:1). Figure 1 represents the mechanism of this method.

Figure 1.Mechanism of Random over-sampling

Chart, diagram

Description automatically generated

*Random Under-Sampling*

The purpose of the random Under-Sampling (RUS) algorithm is to select the majority samples in a random manner without replacing them, and then to get rid of any other majority samples that were not selected. The sample size of the minority class and the intended percentage of the size of the minority samples are taken into consideration when determining the number of retained majority samples. It is planned to produce a collection that is completely balanced (i.e., the imbalanced ratio will be 1:1).

Figure 2. Mechanism of random under-sampling

Diagram

Description automatically generated

*Clustering-based under-sampling*

Due to insufficient sampling, all samples are divided up into groups using the under-Sampling Based on Clustering (SBC) method. Then, it takes into account the majority-minority sample ratio within the cluster to randomly pick the majority samples [5]. To accomplish this, a method known as sampling without replacement would be implemented.

Let's say we have a combined sample size of N from both the main (MA) and minority (MI) groups. *SizeMA* and *SizeMI* are used to denote the relative amounts of MA and MI. At first, SBC divides all of the data into *K* distinct groups. *SizeMA* and *SizeMI* represent the total amount of MA and MI in the *ith* cluster.

Then, pick at random the samples that make up the majority in each cluster. In the *ith* cluster, the number of majority samples that were chosen is:

where *m* is the expected percentage of minority samples that would be reached by majority samples.

Depending on the factors in the dataset, there are different ways to group things together. k-means would be used to cluster the data set with variables that are all continuous (numeric or integer). For the dataset with all categorical variables, k-modes would be used to group things together. Clustering would be done with k-prototypes [7-8] for the dataset with variables that are both continuous and categorical.

*Synthetic Minority Over-sampling Technique*

The Synthetic Minority Over-sampling Technique (SMOTE) uses the nearest neighbors of these cases to make new samples that are not from the minority [4]. The sample size of the majority class and the desired percentage of the sample size of the majority class decide how many new synthetic samples are made.

First, SMOTE will figure out who each minority case's *k*-nearest neighbors are among all the minority cases that don't include itself. Depending on how much oversampling is needed, the new samples would be chosen at random from the *N* nearest neighbors of one group sample. If *N* is less than *k*, *N* would be picked from the *k* nearest friends without replacement. If not, *N* would be chosen by replacement.

Calculate the difference between each minority sample and its *N* nearest neighbors in the second step of the process. Add this difference to the minority sample by multiplying it by a random number between 0 and 1. To make a new synthetic minority sample, SMOTE will pick a random point on the line section between two specific features. Here is how a new sample should be written:

Where  is one minority sample and is the selected sample from the *k*-nearest neighbors of . is the uniform distribution. Due to the above formula, it is important to note that SMOTE can only deal with datasets whose indicators are all continuous.

*Synthetic Minority Over-sampling Technique-Nominal Continuous*

Synthetic Minority Over-sampling Technique-Nominal Continuous (SMOTE-NC) is an improved SMOTE algorithm that can manage mixed datasets of continuous and categorical predictors, according to Chawla et al.'s research [4].

For mixed datasets, the Euclidean distance would be different than normal to find the *k*-nearest neighbors. First, the median of all continuous variables' standard deviations (Med) for the minority class would be found. It gives less weight to categorical predictors that are different between two groups. So, the distance between sample x and sample y is:

Where n is the number of continuous predictors and m is the number of different categorical predictors between sample x and y.

As before, the continuous predictors are made using SMOTE to generate new synthetic minority data. The values seen most frequently in its k-nearest neighbors are assigned to the categorical predictors. Keep in mind that SMOTE-NC can only handle mixed datasets that contain both continuous and categorical variables.

**Test of the methods**

*Datasets*

One real-world imbalanced biological datasets, *abalone,* were used to assess these methods. It can be downloaded through [*https://archive.ics.uci.edu/ml/datasets.php*](https://archive.ics.uci.edu/ml/datasets.php)*.*

The abalone dataset has 731 observations and 8 variables. Except the outcome *Class*, there are 7 continuous predictors. The class labeled as "positive" contains a total of 42 abalones, while the class labeled as "negative" contains 689 abalones. The imbalanced ratio (IR) is about 16.4. The description of variables is shown in Table 1.

Table 1. Data dictionary

|  |  |
| --- | --- |
| Attribute | Description |
| Length | Longest shell measurement |
| Diameter | Perpendicular to length |
| Height | Height with meat in shell |
| Whole weight | Height of whole abalone |
| Shucked weight | Weight of meat |
| Viscera weight | Gut weight (after bleeding) |
| Shell weight | Weight after being dried |

*Machine learning methods*

We created experiments on the chosen dataset, which requires a binary classification, to demonstrate the efficacy of the existing methods. The classification will make use of four different machine learning-based algorithms: a Support Vector Machine (SVM), a Decision Tree, a Random Forest, and a naive Bayesian classifier.

*Design of experiment*

Classification performance of all classifiers has been compared in six scenarios to show the efficacy of the current method: with no resampling, with random over-sampling, with random under-sampling, with under-sampling based on clustering, with synthetic minority over-sampling technique, and with synthetic minority over-sampling technique-nominal continuous. Training samples were resampled to train the classifiers. A 30% test group was used to prevent overfitting. In order to validate the robustness of our discovery, we have carried out 100 iterations and the mean outcome of the performance attained on the evaluation dataset is presented in the subsequent sections.

Classifiers are evaluated according to their accuracy, sensitivity, specificity, and area under the curve (AUC). The definitions of these metrices are listed below:

* Accuracy: percentage of correctly classified patterns. This metric is determined by dividing the number of patterns that are correctly classified and belong to both classes by the total number of patterns.
* Sensitivity (True Positive Rate): The probability of obtaining a positive test result given that the individual is truly positive.
* Specificity (True Negative Rate): The probability of obtaining a negative test result given that the individual is truly negative.
* AUC: a comprehensive evaluation of performance that encompasses all conceivable classification thresholds.

**Results**

The utilization of resampling techniques resulted in a decrease in classification accuracy as opposed to an improvement when compared to the original imbalanced dataset, as determined by the implementation of five distinct machine learning classifiers. The results indicate that the implementation of resampling techniques significantly enhanced the predictive capacity for rare occurrences. The study revealed that under-sampling techniques, specifically SBC and RUS, exhibited superior sensitivity performance in comparison to over-sampling methods, namely ROS and SMOTE. The SMOTE method yielded the poorest sensitivity performance. It is noteworthy that the utilization of over-sampling techniques exhibited greater performance in the identification of majority groups. The utilization of resampling techniques resulted in an enhancement of the classification's overall performance, as measured by the AUC metric. The performance of SBC, ROS, and RUS exhibited similarity in general. The results of all resampling methods were summarized in Table 2 and Figure 3-6.

Table 2. Summary of performance in “Abalones” dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Accuracy | Sensitivity | Specificity | AUC |
| *Random Forest* | | | | |
| Imbalanced | 0.95 | 0.083 | 1 | 0.769 |
| Balanced by SBC | 0.688 | 1 | 0.67 | 0.923 |
| Balanced by ROS | 0.945 | 0.167 | 0.99 | 0.772 |
| Balanced by RUS | 0.775 | 0.667 | 0.782 | 0.721 |
| Balanced by SMOTE | 0.931 | 0.25 | 0.971 | 0.699 |
| *SVM* | | | |  |
| Imbalanced | 0.945 | 0 | 1 | 0.502 |
| Balanced by SBC | 0.697 | 0.833 | 0.689 | 0.789 |
| Balanced by ROS | 0.743 | 0.833 | 0.738 | 0.806 |
| Balanced by RUS | 0.803 | 0.833 | 0.801 | 0.826 |
| Balanced by SMOTE | 0.963 | 0.333 | 1 | 0.665 |
| *Decision Trees* | | | |  |
| Imbalanced | 0.922 | 0.083 | 0.971 | 0.623 |
| Balanced by SBC | 0.748 | 0.667 | 0.908 | 0.769 |
| Balanced by ROS | 0.881 | 0.417 | 0.864 | 0.735 |
| Balanced by RUS | 0.697 | 0.833 | 0.689 | 0.721 |
| Balanced by SMOTE | 0.908 | 0 | 0.961 | 0.696 |
| *Naïve Bayes* | | | |  |
| Imbalanced | 0.876 | 0.417 | 0.903 | 0.66 |
| Balanced by SBC | 0.61 | 0.833 | 0.597 | 0.779 |
| Balanced by ROS | 0.743 | 0.833 | 0.738 | 0.801 |
| Balanced by RUS | 0.807 | 0.583 | 0.82 | 0.760 |
| Balanced by SMOTE | 0.904 | 0.25 | 0.942 | 0.597 |

Figure 3. Comparison of performance using random forest

Chart, bar chart

Description automatically generated

Figure 4. Comparison of performance using SVM

**Chart, bar chart

Description automatically generated**

Figure 5. Comparison of performance using Decision tree

**Chart, bar chart

Description automatically generated**

Figure 6. Comparison of performance using Naïve Bayes

**Chart, bar chart

Description automatically generated**

**Discussion and conclusions**

We try to deal with imbalanced dataset with moderate IR using data-level strategies. Through a combination of novel over-sampling and under-sampling procedures, it is possible to balance the original dataset and to improve the overall classification performance. In general, resampling method enable the binary imbalanced data be more balanced and lead to a better performance of classifiers, especially when we are more concerned on the minority class. Considering the obtained results concerning the different classifiers, we can conclude that under-sampling methods (SBC & RUC) outperformed over-sampling methods (ROS & SMOTE) in identifying rare events. Overall, machine learning classifiers have more power to distinguish between classes after resampling the data. However, their ability varies may according to different IRs. A theoretical work to demonstrate the characteristics can be considered as future work.

Since over-sampling methods perform well in identifying majority while under-sampling is good at identifying minority, a mixed method that combine these two benefits can be considered for future analysis.

**References**

[1] Daskalaki, Sophia, Ioannis Kopanas, and Nikolaos Avouris. "Evaluation of classifiers for an uneven class distribution problem." *Applied artificial intelligence* 20.5 (2006): 381-417

[2] Longadge, Rushi, and Snehalata Dongre. "Class imbalance problem in data mining review." *arXiv preprint arXiv:1305.1707* (2013).

[3] Ma, Yunqian, and Haibo He, eds. "Imbalanced learning: foundations, algorithms, and applications." (2013).

[4] Chawla, Nitesh V., et al. "SMOTE: synthetic minority over-sampling technique." *Journal of artificial intelligence research* 16 (2002): 321-357.

[5] Yen, Show-Jane, and Yue-Shi Lee. "Cluster-based under-sampling approaches for imbalanced data distributions." *Expert Systems with Applications* 36.3 (2009): 5718-5727.

[6] Ja, Hartigan, and Wong Ma. "Algorithm as 136: A k-means clustering algorithm." *Applied statistics* 28.1 (1979): 100.

[7] Huang, Zhexue. "A fast clustering algorithm to cluster very large categorical data sets in data mining." *Dmkd* 3.8 (1997): 34-39.

[8] Huang, Zhexue. "Extensions to the k-means algorithm for clustering large data sets with categorical values." *Data mining and knowledge discovery* 2.3 (1998): 283-304.