2008 20th IEEE International Conference on Tools with Artificial Intelligence

Addressing Class Imbalance in Non-Binary Classification Problems

Naeem Seliya and Zhiwei Xu
Computer and Information Science
University of Michigan – Dearborn
4901 Evergreen Rd., Dearborn, MI 48128, USA
{nseliya; zwxu}@umich.edu

Taghi M. Khoshgoftaar
Computer Science and Engineering
Florida Atlantic University
777 Glades Rd., Boca Raton, FL 33431, USA
taghi@cse.fau.edu

Abstract

The problem of class imbalance in machine learning is quite real and cumbersome when it comes to building a useful and practical classification model. We present a unique insight into addressing class imbalance for classification problems that involve three or more categories, i.e. non-binary. This study is different than related works in the literature because most works focus on addressing class imbalance only for binary classification problems, even if it means transforming a non-binary dataset into a binary classification problem. We propose an effective, yet simple approach to alleviating class imbalance issues when the classification problem involves more than two classes. The process, with four different methods, is based on applying random undersampling and random oversampling to different parts of the dataset for achieving better classification performance. The proposed data sampling methods are evaluated in the context of two real-world datasets obtained from the UCI Repository for Machine Learning Databases, and two commonly used classification algorithms: C4.5 and RIPPER. Our results demonstrate that the multi-group classification accuracy increases significantly in most cases after the proposed data sampling methods are applied. The positive outcome of this study motivates us to further our research on class imbalance and non-binary classification problems.

Keywords: Machine learning, class imbalance, non-binary classifiers, data sampling, artificial intelligence.

1. INTRODUCTION

Machine learning is an integral part of the artificial intelligence domain. The task of learning from past heuristic data and using the learnt knowledge to predict, decide, and analyze future events is a commonly observed task across various application domains – e.g., credit card fraud detection [17], software quality modeling [6], manufacturing process [17], etc.

Classifying domain-specific entities into different categories (i.e. prediction) is one of the most common machine learning tasks observed in artificial intelligence. A classifier is typically built using known prior knowledge in the form of a training data, and the subsequent model is then applied to predict the unknown classification of target data points. The efficacy and usefulness of such a classifier is very much dependent on the characteristics of the training data used [13][15].

One such characteristic of the training data is the relative distribution of its instances among the different classes. If the size of one class is relatively small (skewed) compared to the other classes, then it is very likely that the training process may not sufficiently learn trends of instances in that class. In contrast, a class that has a very high proportion of instances in the training data is likely to be represented too much in the trained model. It is intuitive to see how such a class imbalance problem can adversely affect the training process of a classifier.

Existing literature related to the class imbalance problem has primarily focused on datasets that contain only two classes (binary classification, with a majority class and a minority class). However, that does not exclude the occurrence of class imbalance among more than one class in a dataset consisting of multiple groups. Many real-world classification scenarios mandate machine learning with multiple categories. This is clearly observed in the various datasets included in the UCI Repository for Machine Learning Databases [2].

The contribution of this study is an effective, yet simple, solution to addressing the class imbalance problem for classification scenarios that involve more than two categories. We consider combining the random undersampling and random

1082-3409/08 $25.00 © 2008 IEEE
DOI 10.1109/ICTAI.2008.120

oversampling techniques (these are explained in the next section) in our data sampling process for alleviating the class imbalance problem. The combination of the two techniques is such that different parts of the dataset are subjected to a different data sampling process, i.e. either random undersampling or random oversampling.

The proposed data sampling approach for non-binary classification problems is presented and evaluated with two real-world datasets obtained from the UCI repository. Both datasets consist of more than two classes, and both have relatively skewed distributions of instances among their respective classes. Our empirical results clearly demonstrate the benefits of the proposed approach for addressing the class imbalance problem. In most cases, the classification performance improves compared to training without the proposed data sampling process.

The remainder of the paper is structured as follows. Section 2 provides a summary on existing data sampling techniques and contrasts them with what is being proposed here. Section 3 details the proposed data sampling process for non-binary classification problems. Section 4 discusses the two classifiers we use in our case studies. Section 5 presents the case studies of our empirical work, including dataset descriptions, results, and discussions. In Section 6 we summarize our work and provide some suggestions for future research directions.

2. RELATED WORK

We limit our discussion to existing data sampling techniques for addressing the class imbalance problem. To our knowledge, such existing techniques are either majority undersampling or minority oversampling techniques since they only focus on binary classification situations when addressing the class imbalance problem. Their application to non-binary classification situations is not yet clear – hence, the focus of our study.

In a binary classification problem, majority undersampling removes instances from the majority, i.e. larger class, with the aim of improving bias of the minority class instances. The majority undersampling techniques include random undersampling, Wilson's editing, and one-sided selection. Random undersampling is an effective technique in which a portion of the majority class instances are removed at random from the dataset [13][14][15]. Wilson's editing strives to remove noisy instances of the majority class based on a k-nearest-neighbor (k-NN, with k = 3) algorithm that classifies each instance in the training dataset using the remaining instances [1][16]. One-sided selection aims to remove both noisy and redundant instances of the majority class from the training dataset [7]. Tomek links [12] are used to remove incorrectly classified instances and borderline instances which lie close to the boundary between the two classes in the feature space.

Minority oversampling adds instances to the minority class group, also striving to increase bias of the minority class instances. The minority oversampling techniques include random oversampling, cluster-based oversampling, Synthetic Minority Oversampling Technique (SMOTE), and Borderline-SMOTE. Random oversampling augments the training dataset by randomly duplicating instances from the minority class [13]. SMOTE [3] creates new artificial (synthetic) minority data instances rather than simply duplicating from the existing instances. It creates attribute values for the new data instances by extrapolating values from the k-nearest-neighbors to each of the original minority class examples.

Borderline-SMOTE [4] is a modified version of SMOTE, and extrapolates the new data instances only from those minority instances that exist close to a minority class boundary in the feature space. Cluster-based oversampling alleviates the imbalance between the majority and minority classes, and helps balance instance grouping within the two classes [5]. The algorithm creates independent clusters from the minority and majority classes, and then randomly oversampling each of the majority clusters, except the largest cluster. This is done with replacement until all of the majority clusters contain the same number of instances as the largest cluster.

In contrast to the above techniques, we present a simple, but effective methodology that addresses the class imbalance problem when the real-world classification scenario mandates more than two groups. We do not reformulate the given dataset into a binary classification problem as is done in existing works related to class imbalance. As explained later, our process applies random undersampling and random oversampling to different parts of a given dataset while striving to alleviate the class imbalance problem.

3. PROPOSED APPROACH

To our knowledge, this paper presents a unique focus on addressing class imbalance problem in classification modeling tasks that involve three or more categories. Basic research in related literature is relatively well founded with techniques that address class imbalance with classification problems that involve only two (binary) categories.

Many of the existing works on class imbalance modify real-world datasets that have three of more classes into a dataset that has only two categories [3][5][13][14]. This is typically done by selecting the smallest category of instances as the minority class, and then combining instances from all other categories and forming the majority class. Moreover, this is done to suit the existing data sampling techniques (discussed earlier) that require the dataset to have only two classes.

We argue that the practice of modifying real-world datasets to form a binary classification problem is not reflective of real-world situations and is not always practical. For example, a given machine learning dataset may contain multiple categories that are relatively very small, yet important, compared to the other categories in the dataset. One may need to address the class imbalance problem for multiple, skewed classes in the same dataset.

An effective, yet simple approach is proposed here for addressing class imbalance when the classification problem mandates using a machine learning dataset that has more than two categories, i.e. non-binary. Moreover, a given dataset can have more than one skewed class, as is the case in the two real-world datasets used in this study. The proposed data sampling process is described in the remainder of this section.

Consider a dataset D consisting of instances grouped into $\{C_1, C_2, \ldots, C_m\}$ classes, where C_i is the ith class and $i = \{1, 2, \ldots, m\}$. Let D′ be the dataset after data sampling has been performed. The following steps detail our data sampling process, which allows four different methods of data sampling prior to classifier training.

1. Determine the sizes of all classes in D, i.e. $\{|C_1|, |C_2|, \ldots, |C_m|\}$ and sort them in a descending order. All instances are now sorted based on their associated class size, from largest class to smallest class. However, instances within a given class are not sorted. At this point D′ = D, since no data sampling has occurred.

2. Select the method to use for obtaining modified class sizes for performing data sampling, either random undersampling or random oversampling. Four methods are considered:

   a. Standard Mean (SMean)
      i. Compute the mean of all class sizes in D, and denote this size as SM.
      ii. For a class C_i in D, if $|C_i| > SM$, then instances from C_i are randomly removed (i.e. random undersampling) from D′ until $|C_i| = SM$.

          For a class C_k in D, if $|C_k| \leq SM$, then instances from C_k are randomly added (i.e. random oversampling) to D′ until $|C_k| = SM$.

   b. Modified Mean (MMean)
      i. Compute the mean of all class sizes in D, and denote this size as SM.
      ii. Divide the sorted classes (from Step 1) into two groups, *Upper* and *Lower*, by using SM as the dividing threshold. Hence, the size of a class belonging to the *Upper* group is greater than SM, while the size of a class belonging to the *Lower* group is lesser than or equal to SM.
      iii. Compute MM_{Upper} as the mean class size among all classes belonging to the *Upper* group, MM_{Lower} as the mean class size among all classes belonging to the *Lower* group.
      iv. For a class C_i in D, if $|C_i| > MM_{Upper}$, then instances from C_i are randomly removed (i.e. random undersampling) from D′ until $|C_i| = MM_{Upper}$.

          For a class C_k in D, if $MM_{Upper} \geq |C_k| > SM$, then instances from C_k are randomly added (i.e. random oversampling) to D′ until $|C_k| = MM_{Upper}$.

          For a class C_q in D, if $SM \geq |C_q| > MM_{Lower}$, then instances from C_q are randomly removed (i.e. undersampling) from D′ until $|C_q| = MM_{Lower}$.

          For a class C_t in D, if $|C_t| \leq MM_{Lower}$, then instances from C_t are randomly added (i.e. oversampling) to D′ until $|C_t| = MM_{Lower}$.

Compared to method (a), our rationale for developing and using this modified approach was to minimize a high rate of undersampling from the larger classes, and also minimize a high rate of oversampling from the smaller classes. This seemed intuitive because if the size of the largest class in a dataset was relatively very high than sizes of the other classes, one stands to possibly loose useful information via too much undersampling. Finally, this modified data sampling method seems more suited for addressing class imbalance in a classification problem consisting of three or more classes.

c. Standard Median (SMedian)
The steps of this method are similar to those for SMean, i.e., (a) above. The only difference here is that SM now represents the median class size among all classes in D.

d. Modified Median (SMedian)
The steps of this method are similar to those for MMean, i.e., (b) above. The differences here are that SM now represents the median class size among all classes in D, and MM_{Upper} and MM_{Lower} now respectively represent the median class sizes among all classes in the *Upper* and *Lower* groups.

3. Obtain the modified dataset, D′. This resampled machine learning dataset is now ready for classification modeling.

The proposed data sampling methods are implemented as a custom filter Java class that is used as a plug-in for the Java-based WEKA data mining and machine learning tool [17]. In our study we use WEKA 3.5.6 version for our empirical case studies.

4. SELECTED CLASSIFIERS

The two non-binary classifiers used in our study are briefly described in this section. We selected these two learners because of their common use among machine learning practitioners and their provision of handling a dataset with three or more classes. However, the proposed data sampling methods can be applied with any other non-binary classifier because data sampling is a preprocessing step during classification modeling.

- **C4.5 Decision Tree**: The C4.5 algorithm is an inductive supervised learning system which employs decision trees to represent a quality model. We use the J48 classification model, which is WEKA's implementation of the C4.5 algorithm [10][17]. Starting with the root node, the algorithm recursively selects an independent variable to split the data into sub-trees, until stopping criteria is satisfied. When determining which attribute to split the data, C4.5 works to maximize the purity of the resulting child nodes.

  The purity of the node is measured by its information value or its entropy. C4.5 will choose the attribute which provides the greatest information gain ratio for the split as it tries to form nodes with an information value as close to zero as possible, i.e., a pure node. To avoid overfitting, C4.5 implements two types of pruning: subtree replacement and subtree raising.

- **RIPPER**: The Repeated Incremental Pruning to Produce Error Reduction algorithm is a rule-based learner that generates classification rules [17]. Using a “separate and conquer” strategy, a rule-based learner initially creates a rule that satisfies a set of instances of one class. The algorithm then removes the instances covered by that rule from the training dataset and then continues generating more rules using the remaining instances.

  The RIPPER algorithm seeks to maximize the information gain, and creates the conditions for its rules by greedily selecting the condition that yields the greatest information gain. To find a condition for the rule's antecedent, RIPPER evaluates all possible values for all of the nominal attributes in the dataset and checks thresholds between the possible values for numeric attributes. To avoid model over-fitting during training, RIPPER prunes the rules using a method called reduced-error pruning. We use the default RIPPER parameter settings in WEKA for our case studies.

We use the default parameter settings in WEKA for both C4.5 and RIPPER. The task of parameter optimization for both learners is beyond the scope of our study.

5. EMPIRICAL CASE STUDIES

Case Study Datasets

The two datasets used in our case studies are described in this section. Both datasets were obtained from the UCI Repository for Machine Learning Databases [2].

- **Glass**: This dataset consists of 214 instances. The nominal class attribute identifies each instance as one of six different types of glass. The nine independent attributes are all numeric and describe the levels of different oxides in the glass sample. The smallest class consists of 9 instances, while the largest class consists of 76 instances.

- **Satimage**: The Landsat Satellite database is one of the Statlog Project databases. A given instance is characterized by 36 independent, numeric attributes which are the 9 byte pixel values for each of the four spectral images. The original dataset has six classes, where each represents a specific soil type for the section of land in the 3x3 image. In our study we combine the training data (4435 instances) and test data (2000 instances) into one large dataset. This is done since our focus is on addressing the class imbalance problem in non-binary classification problems, and we wanted to consider one very large dataset in addition to one very small dataset (i.e., Glass). The smallest class has 626 instances, while the largest class has 1533 instances.

Results and Discussion

The 10-fold cross-validation performances for C4.5 and RIPPER using the two case study datasets are shown in Tables 1 and 2, respectively. The tables present the classification accuracies based on using one of the four proposed data sampling methods, or not using any data sampling prior to training the classifiers. Since our focus is on multi-group (non-binary) classification, the overall classification accuracies are presented instead of the more commonly used performance metrics for binary classification such as false positive, false negative, true positive, true negatives, Recall, Precision, etc.

With C4.5 and Glass, there is generally a clear improvement in the overall classification accuracy compared to modeling with no data sampling for addressing the class imbalance problem. The latter yielded 65.89% accuracy, while three of the four proposed data sampling methods yielded greater than 75% accuracy. A *z-test* at an $\alpha = 0.05$ (95% significance) validated the positive improvements. The MMean method yielded a modest 69.16% accuracy, and is still greater than using no data sampling prior to classification modeling.

In the case of C4.5 and Satimage, while an improvement over the base case (i.e. No Sampling with 85.84% accuracy) was generally observed, the performance increase was rather modest in most cases. The highest improvement of 3.41 percentage points was obtained with the SMean data sampling method. Based on *z-test*, three of the four data sampling methods were significant at an $\alpha = 0.10$, but not at an $\alpha = 0.05$. Our overall conclusion with the case study involving C4.5 and the two datasets is that the proposed data sampling methods generally provided significant benefits compared to not using any data sampling prior to classification modeling (non-binary).

The case study involving RIPPER with the two datasets yielded results (Table 2) that generally followed similar trends observed in Table 1. With RIPPER and Glass, the base case yielded 63.77% accuracy, compared to about 70% or higher accuracy with the four data sampling methods. However, classification accuracy with RIPPER for Glass was always lower than those obtained with C4.5. We note that identifying the best classification algorithm in the context of class imbalance is not the focus of this study, but is a candidate empirical investigation for future work.

In the case of RIPPER and Satimage, improvement over the base case (i.e. 86.51%) was generally not noticeable. The largest improvement in overall accuracy was 1.58 percentage points with the SMean data sampling method. Our overall impression with the case study involving RIPPER and the two datasets is that the proposed data sampling methods generally provided good, but modest, improvements compared to not using any data sampling, especially with the Satimage dataset. Like others [11], we have noted earlier that various characteristics of a dataset can impact, positively or adversely, the outcome of a machine learner.

A comparison among the four data sampling methods presented in this study (Tables 1 and 2) provided relatively mixed results. In the case of C4.5, SMean provides the best classification accuracy for both Glass and Satimage. However, for Satimage the other three data sampling methods provide competitive performances. In the case of

RIPPER, SMean once again provides the best accuracy for both Glass and Satimage. However, again for Satimage the other three data sampling methods are competitive.

A comparison among the respective standard and modified methods (Tables 1 and 2) provides some interesting results. With C4.5 and Glass, SMean is clearly better than MMean. However, this improvement is not impressive with C4.5 and Satimage. With RIPPER and Glass, once again SMean provides better performance than MMean; however, with Satimage the two are again relatively competitive. In the case of SMedian and MMedian, both provided relatively similar results when either classifier is applied to either datasets.

Table 1: C4.5 Classification Accuracy with Proposed Data Sampling Methods

Sampling Method	Glass	Satimage
SMean	82.87	89.25
MMean	69.16	87.09
SMedian	75.36	88.71
MMedian	76.31	86.73
No Sampling	65.89	85.84

Table 2: RIPPER Classification Accuracy with Proposed Data Sampling Methods

Sampling Method	Glass	Satimage
SMean	75.93	88.09
MMean	71.03	86.60
SMedian	69.63	86.90
MMedian	73.09	86.71
No Sampling	63.77	86.51

Table 3: Resampled Instance Counts with Glass Dataset

Class	SMean	MMean	SMedian	MMedian	Original
I	36	73	23	70	70
II	36	73	23	70	76
III	36	17	23	13	17
IV	36	17	23	13	13
V	36	17	23	13	9
VI	36	17	23	70	29
Total	216	214	138	249	214

The class-wise numbers of instances for the Glass and Satimage datasets are shown in Tables 3 and 4, respectively. The tables show both the Original class-wise instance count and after each of the four data sampling methods are applied to obtain D′, i.e. the final resampled dataset. Both datasets have six classes, as shown by the Class column. The last row provides the total of each column.

Table 4: Resampled Instance Counts with Satimage Dataset

Class	SMean	MMean	SMedian	MMedian	Original
I	1073	1466	1033	1508	1533
II	1073	679	1033	703	703
III	1073	1466	1033	1508	1358
IV	1073	679	1033	703	626
V	1073	679	1033	703	707
VI	1073	1466	1033	1508	1508
Total	6438	6435	6198	6633	6435

In the case of Glass, the original dataset increases from 214 to 249 instances when the MMedian method is used. On the other hand the original dataset decreases from 214 to 138 instances when SMedian is used. In the case of SMean a negligible increase of 2 instances is observed, while no change in size is observed with MMean.

In the case of Satimage, MMedian increases the original dataset to 6633 instances, while SMedian decreases the dataset to 6198 instances. There is either no, or very minor change in size when SMean or MMean are used. We note that is it is by chance that MMean for both datasets did not change the original size. The median based data sampling methods seems to make the most changes in sizes of the two datasets, as compare to the mean based data sampling methods.

6. CONCLUSION

The problem of class imbalance in machine learning is addressed in the context of non-binary classification problems. While existing literature on the class imbalance problem presents good solutions, they are all limited to a two-group or binary classification problem. However, it is not uncommon to have a machine learning task where a classification model is to be built for classifying domain-specific instances into three or more groups. There is no existing solution that addresses class imbalance in the case of non-binary classification problems.

We proposed an effective, yet simple strategy to alleviate the adverse effects class imbalance can have on a multi-group classification problem. Our approach, with four different application methods, is based on performing random undersampling and random oversampling on different parts of the machine learning dataset for achieving better

classification accuracy. Case studies of two real-world datasets are used to empirically evaluate the proposed data sampling approach. The impact on classification accuracy is observed by using the C4.5 decision tree and RIPPER machine learners.

Our results demonstrate that the multi-group classification accuracy increases significantly in most cases after the proposed data sampling methods are applied. The positive outcome of this study motivates us to further our research on class imbalance and non-binary classification problems.

Some future directions will include: investigating other existing data sampling techniques (such as Cluster-based oversampling, SMOTE, Borderline-SMOTE, Wilson's editing, etc.) for their applicability to non-binary classification problems; developing new ways to determine, for a given machine learning dataset, which parts should be undersampled and which parts should be oversampled; and, further empirical validation by performing additional case studies other datasets and/or machine learners.

ACKNOWLEDGMENTS

We are grateful to Michael D. Gilbert for his assistance with machine learning experiments.

REFERENCES

[1] R. Barandela, R. M. Valdovinos, J. S. Sanchez, and F. J. Ferri, "The imbalanced training sample problem: under or over sampling?" In *Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition* (SSPR/SPR'04), pp. 806-814, 2004.

[2] C. Blake and C. Merz, "*UCI Repository of Machine Learning Databases*", Department of Information and Computer Sciences, University of California, Irvine. 1998.

[3] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique", *Journal of Artificial Intelligence and Research*, vol. 16, pp. 321-357, 2002.

[4] H. Han, W. Y. Wang, and B. H. Mao, "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning", In *International Conference on Intelligent Computing* (ICIC'05), *Lecture Notes in Computer Science*, no. 3644, pp. 878-887, 2005, Springer-Verlag.

[5] T. Jo and N. Japkowicz. "Class imbalances versus small disjuncts", *ACM SIGKDD Explorations*, vol. 6, no. 1, pp. 40-49, 2004.

[6] T. M. Khoshgoftaar and N. Seliya, "Comparative assessment of software quality classification techniques: an empirical case study", *Empirical Software Engineering Journal*, vol. 9, no. 3, pp. 229-257, 2004, Kluwer Publishing.

[7] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: one sided selection", In *Proceedings of the 14th International Conference on Machine Learning*, pp. 179-186, 1997.

[8] A. Orioles and E. Bernado-Mansilla, "Class imbalance problem in UCS classifier system: fitness adaptation", *IEEE Congress on Evolutionary Computation*, pp. 604-611, September 2005.

[9] F. Provost and T. Fawcett, "Robust classification for imprecise environments", *Machine Learning*, vol. 42, pp. 203-231, 2001.

[10] J. R. Quinlan, "*C4.5: Programs for Machine Learning*", Morgan Kaufmann, San Mateo, CA, 1993.

[11] M. Shepperd and G. Kadoda, "Comparing Software Prediction Techniques Using Simulation", *IEEE Transactions on Software Engineering*, 27(11): 1014-1022, 2001.

[12] I. Tomek, "Two modifications of CNN", *IEEE Transactions on Systems, Man and Communications*, vol. 6, pp. 769-772, 1976.

[13] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental perspectives on learning from imbalanced data", In *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, June 2007.

[14] G. M. Weiss, "Mining with rarity: a unifying framework", *ACM SIGKDD Explorations*, vol. 6, no. 1, pp. 7-19, 2004.

[15] G. M. Weiss and F. Provost, "Learning when training data are costly: the effect of class distribution on tree induction", *Journal of Artificial Intelligence Research*, vol. 19, pp. 315-354, 2003.

[16] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data sets" *IEEE Transactions on Systems, Man and Cybernetics*, vol. 2, pp. 408-421, 1972.

[17] I. H. Witten and E. Frank, "*Data Mining: Practical Machine Learning Tools and Techniques*", 2nd ed., Morgan Kaufmann, San Francisco, CA, 2005.