CGMOS: Certainty Guided Minority OverSampling

Xi Zhang, Di Ma, Lin Gan, Shanshan Jiang, Gady Agam

Abstract—Handling imbalanced datasets is a challenging problem that if not treated correctly results in reduced classification performance. Imbalanced datasets are commonly handled using minority oversampling, whereas the SMOTE algorithm is a successful oversampling algorithm with numerous extensions. SMOTE extensions do not have a theoretical guarantee during training to work better than SMOTE and in many instances their performance is data dependent. In this paper we propose a novel extension to the SMOTE algorithm with a theoretical guarantee for improved classification performance. The proposed approach considers the classification performance of both the majority and minority classes. In the proposed approach CGMOS (Certainty Guided Minority Over-Sampling) new data points are added by considering certainty changes in the dataset. The paper provides a proof that the proposed algorithm is guaranteed to work better than SMOTE for training data. Further experimental results on 30 real-world datasets show that CGMOS works better than existing algorithms when using 6 different classifiers.

1 INTRODUCTION

In many real world problems, the distribution of data between classes is imbalanced. Learning from imbalanced datasets is an important research problem with many applications.

The fundamental issue in imbalanced learning is the ability of imbalanced data to significantly compromise the performance of standard learning algorithms [1]. Generally, there are three primary reasons that can cause this problem [2].

The first reason is that the lack of data in the minority class makes it difficult to detect regularities within the minority class. Thus, the learned decision boundaries are less likely to approximate the true decision boundaries.

Second, many classification algorithms utilize a general bias for better generalization and to avoid overfitting during learning. However, such bias can adversely affect the ability to learn the minority class. Inductive bias also plays a key role with respect to the minority class. Most classification algorithms prefer more common classes in the presence of uncertainty (i.e., they are biased in favor of the class priors).

Last but not least, noise exerts a greater impact on the minority class, because in this case it is more difficult for a classifier to distinguish noise from minority data. This is especially so in extreme cases where the number of noisy samples is greater than actual minority samples. The problem of overfitting rises again, when modifying the classifier to learn the minority data correctly.

To address these problems, numerous research efforts have been devoted to imbalanced learning in recent years. The majori of techniques that solve the imbalanced learning problem fall into two categories: cost-sensitive methods and sampling-based methods. In the next section, we review related work on sampling-based methods [3].

1.1 Related work

A number of solutions to the class-imbalance problem were previously proposed both at the data and algorithmic levels [4]. There are mainly three groups of methods that can solve imbalanced learning problem [1], including sampling methods, cost sensitive methods, and kernel methods. Sampling-based methods are very effective and easy to use when solving imbalanced learning problems. In addition, sampling-based methods can be used together with methods in the other two groups to further improve performance. In such approaches a sampling technique is used to modify an imbalanced dataset to produce a balanced distribution. It has been shown that for most imbalanced datasets, sampling techniques do improve classification accuracy.

The basic sampling methods include undersampling and oversampling. Undersampling reduces majority class samples while oversampling increases minority class samples. While several works achieving data balance through undersampling have been proposed in the past [4] [5], more research efforts have been devoted to oversampling due to the fact that oversampling does not discard information.

The simplest form of oversampling is duplication of minority class samples. This approach decreases the overall level of class imbalance, but may lead to overfitting [6]. SMOTE [7] is a fundamental approach for oversampling using data synthesis. To balance the dataset, SMOTE randomly selects a seed sample and synthesizes a new sample by applying a linear interpolation between the seed sample and one of its neighbors. Large research efforts have been devoted to feature space data synthesis based on SMOTE. Several methods integrate data synthesis as a part of the learning procedure. For example, by introducing SMOTE in each iteration of boosting. SMOTEBoost [8] increases the number of minority class samples and focus on these cases in each boosting iteration. Using the same idea of boosting, DataBoost-IM [9] and RAMOBoost [10] discover samples difficult to classify during each iteration of boosting, which are used to guide the oversampling in both the majority and minority classes.
In another group of minority oversampling approaches, the data synthesis procedure is independent of the learning processes. Such methods give preferences to different regions of a dataset by assigning weights to samples in the dataset. These weights can then generate a probability distribution which is used for randomly drawing samples. In such approaches the data synthesis can be completed in one step. Methods in this group include Borderline-SMOTE [11], Adasyn [12], [13] and MWMOTE [14]. All of these methods synthesize more samples along decision boundaries. However, these methods do not have objective functions to systematically guide the process of oversampling and so do not have a systematic way to decide on where new data should be synthesized. Thus, such approaches cannot measure the impact of each synthetic sample. As a result, there are several potential problems. One is that the oversampling procedure may sacrifice the performance of the majority class in order to improve the performance of the minority class in the classification. Another is that synthetic minority samples themselves can be misclassified and affect the performance in the minority class.

1.2 Novel Contribution

The proposed approach, CGMOS, is a member of the SMOTE family that can achieve data oversampling in a single step. To address some of the shortcomings in existing approaches, we propose a novel oversampling strategy by systematically considering the performance of both minority and majority classes. Based on a Bayesian classification framework, our proposed approach computes the influence of minority data addition on the certainty of the entire dataset. CGMOS thus can synthesize new samples that will improve the overall certainty of the entire dataset in classification. We prove that during training CGMOS is guaranteed to perform better than SMOTE when using Bayesian classification. To validate the proof, We further show experimentally that CGMOS outperforms known approaches when tested on real-world data set collections using different classifiers.

2 Problem Formulation

In this paper, we address the binary classification problem for imbalanced datasets. Let \( D = \{(x_j, y_j)\}_{j=1}^n \) be a training dataset, where \( x_j \in \mathbb{R}^m \) are features and \( y_j \in \{l_{mjr}, l_{mnr}\} \) are ground truth class labels. We begin by formally defining the certainty of imbalanced binary classification using a Bayesian framework, where a kernel density estimation (KDE) is used to estimate the samples’ probability density function (PDF). We then show how CGMOS can synthesize more samples according to the certainty estimation.

2.1 Definition of Certainty

Suppose \((x_j, y_j)\) is any tuple in the training dataset \( D \), where \( x_j \) is a feature vector and \( y_j \) is the ground truth label of \( x_j \).

A Bayesian classifier maps \( x_j \rightarrow l, l \in \{l_{mjr}, l_{mnr}\} \) using following rule.

\[
l = \begin{cases} 
  l_{mnr} & \text{if } \frac{P(l_{mnr}|x_j)}{P(l_{mjr}|x_j)} > 1 \\
  l_{mjr} & \text{otherwise}
\end{cases}
\]

where the posterior probability \( P(l|x_j) \) is computed using Bayes’ rule:

\[
P(l|x_j) = \frac{P(x_j|l)P(l)}{P(x_j)}; \quad l \in \{l_{mjr}, l_{mnr}\}
\]

Uncertainty is commonly used in machine learning algorithms. In this work, we use the posterior probability \( P(y_j|x_j) \) to define certainty. This is because in classification, the posterior probabilities \( P(y_j|x_j) \) reflect the certainty of assigning a sample to a correct label, where higher numbers indicate classification results with a stronger certainty.

**Definition 1. (Certainty)** Let \((x_j, y_j)\) be any tuple in \( D \), where \( x_j \) is a feature vector and \( y_j \) is the ground truth label of \( x_j \). The certainties for samples in the majority and minority class are respectively defined as:

\[
C(y_j = l_{mjr}|x_j) = P(y_j = l_{mjr}|x_j)
\]

(1)

\[
C(y_j = l_{mnr}|x_j) = P(y_j = l_{mnr}|x_j)
\]

(2)

It should be noted that in the case of binary classification the definition of certainty above is related up to some constants to the uncertainty defined in [15] based on margin confidence.

2.2 PDF Estimation

There are two general ways to estimate a density function: parametric or non-parametric. In this work we use a non-parametric model so as to not depend on a specific distribution model. We use kernel density estimation (KDE) [16] [17] to estimate the likelihood \( P(x_j|l), l \in \{l_{mjr}, l_{mnr}\} \).

Assuming that the data is independent and identically distributed (i.i.d) and drawn from some distribution with an unknown density \( P(x_j|l) \), we have using KDE:

\[
P(x_j|l) = \frac{\sum_{k=1}^{n} K\left(\frac{x_j-x_k}{h_k}\right)I(y_k = l)}{\sum_{k=1}^{n} I(y_k = l)}
\]

(3)

where \( l \in \{l_{mjr}, l_{mnr}\}, I(\cdot) \) is an indicator function, and \( K(\cdot) \) is a kernel function which has zero mean and integrates to one. Given any sample \( x_k \), the bandwidth \( h_k \) of the sample \( x_k \) controls the effective range of the kernel and smoothness of the density function. Intuitively one wants to choose \( h_k \) as small as the data allows to exhibit as many underlying structures of the data as possible. Small bandwidth, however, will result in a noisy estimate. In this work, for any sample \( x_k \), we calculate a bandwidth \( h_k \) as a scaled average distance between \( x_k \) and its \( q \) nearest neighbors:

\[
h_k = \sigma \cdot \frac{\sum_{x \in N(x_k)} \|x - x_k\|}{q}
\]

(4)

where \( N(x) \) is the set of the \( q \) nearest neighbors of \( x_k \) and \( \sigma > 0 \) is a scale factor applied to the distance. We will discuss selection of parameters \( \sigma \) and \( q \) in Section 4.
2.3 Oversampling Seed Selection

In most classification algorithms, samples close to decision boundaries have less certain classification results. In order to achieve better predictions for such samples, many existing approaches synthesize data directly along the boundaries. However, this is risky and the expected performance improvement is not guaranteed. There are two primary reasons. First, samples from both classes are mixed in regions near the boundaries. Synthetic samples if added to these regions are less predictable and hard to learn. Second, adding synthetic minority samples to these regions may adversely impact the majority class, which may in turn decrease the performance of the majority class in classification. Instead of unguided oversampling near the boundaries, our proposed approach targets adding samples by considering the certainties of both the majority and minority classes before and after adding the samples. The synthetic samples thus are added to locations that can improve the overall certainty of the original data and boost the performance of the classification.

CGMOS uses a similar procedure as SMOTE when synthesizing a new sample. The sample is produced by interpolating between one seed sample and some of its neighbors. However, instead of randomly drawing a seed sample for interpolation, CGMOS assigns each sample \((x_i, y_i) \in D\) a weight \(W(x_i)\) which is used to determine the probabilities of \(x_i\) being chosen for interpolation. A higher weight results in a higher probability of a point being selected.

To compute \(W(x_i)\), we suppose that a new sample will be added to the same location as \(x_i\). The weight \(W(x_i)\) is computed as a relative certainty change comparing the certainty before and after the sample is added. With a new sample added at location \(x_i\), we update the certainty for all \((x_j, y_j) \in D\) and denote it as \(C_{+i}(y_j|x_j)\).

**Definition 2. (Relative Certainty Change)** The relative certainty change of label \(y_j\) assigned to feature \(x_j\) due to adding a minority example at location \(x_i\) is defined by:

\[
R_{+i}(y_j|x_j) = \frac{C_{+i}(y_j|x_j) - C(y_j|x_j)}{C(y_j|x_j)}
\]

where \(C(y_j|x_j)\) is the certainty before addition.

When computing \(W(x_i)\), CGMOS considers the relative certainty changes of examples from both the majority and the minority classes. \(W(x_i)\) is computed as the average value of relative certainty changes of all samples in the dataset.

\[
W(x_i) = 1 + \frac{1}{n} \sum_{j=1}^{n} R_{+i}(y_j|x_j)
\]

Given \(W(x_i)\) for all \(x_i \in D\), it is easy to see \(W(x_i) > 0\). We compute a normalization factor \(z\) so that \(\frac{1}{n} \sum_{i=1}^{n} W(x_i) = 1\). Therefore, the oversampling procedure can randomly choose a sample for interpolation according \(W(x_i)/z\). The interpolation phase of CGMOS is the same as SMOTE.

A demonstration of CGMOS is shown in Fig. 1. In this figure, samples in both the majority and minority classes are randomly drawn based on Gaussian distribution, where the means of the two datasets are on the same horizontal line, and the mean of the majority is to the right of the minority. The majority class contains 2000 samples and the minority class contains 400 samples. Color in part 1 of the figure indicates the certainty of each example with respect to its class, where red indicates high certainty. We highlight 3 regions (A, B, C) in the minority class. Samples in region A have relative high certainties, sample in region B has low certainties and region C is a boundary region in which samples have the lowest uncertainties. Part 2 of the figure shows the weight of each example as computed by our approach where red indicates high values. Region B has higher values and is where CGMOS will synthesize most of the samples.

To show the certainty changes induced by adding samples at different locations of the dataset, in part 3 of the figure we add one minority sample and move its location with a fixed step size from left to right on a horizontal line passing through the two classes. We then compute the relative certainty changes for all samples in both classes. As can be observed, by measuring relative certainty changes, CGMOS will assign a higher weight to samples in region B. The figure also shows that by oversampling more in region B, the certainty of the entire dataset gets improved, because the relative certainty changes are positive.

3 Theoretical Guarantee Over SMOTE

Several existing approaches claim handling imbalanced learning better than SMOTE. Such claims are normally validated using empirical tests without a theoretical guarantee and in some instances may not extend to new datasets. In this section we provide a theoretical guarantee showing that CGMOS is expected to work better than SMOTE in training process.

Let \(D = \{(x_j, y_j)\}_{j=1}^{n}\) be a training dataset. Let \(W(D) = \{W(x_i)\}_{i=1}^{n}\) be the sample weights computed using Eqn. 6.

**Lemma 1.** Given a set of weights \(\{W(x_i)\}_{i=1}^{n}\) as defined above and a normalization factor \(z\) given by \(z = \sum_{i=1}^{n} W(x_i)\), it must be that \(\sum_{i=1}^{n} W(x_i)^2 \geq \frac{2}{n}\).
Theorem 1. The expected average gain in CGMOS is higher or equal to that of SMOTE.

Proof For CGMOS the expected average gain is given by:

\[
E_p = E[\bar{r}_{+i}] = \sum_{i=1}^{n} \bar{r}_{+i} \frac{W(x_i)}{z}
\]

4 RESULTS AND DISCUSSION

4.1 Datasets

30 real-world datasets were randomly chosen from the UCI machine learning repository [18] for empirical testing of CGMOS. Most of the datasets were released within the past 10 years. As some of the datasets contain samples of more than two classes, we convert such datasets to a binary classification problem by keeping the class with the least data and merging all other classes. A summary of the test collections is provided in Table 1.

4.2 Compared Approaches

According to a survey of imbalanced learning [1], there are mainly three groups of methods addressing imbalanced learning: sampling methods, cost sensitive methods, and kernel methods. The proposed CGMOS belongs to the sampling group. Thus, we compare CGMOS to five other oversampling methods in this group: SMOTE [7], Borderline-SMOTE [11], ADASYN [12], MWMOTE [14] and RAMOBoost [10]. Since oversampling by duplication is broadly used in many applications as a baseline, we add it to our evaluation as well. To demonstrate the improvement of these oversampling strategies, we include in the comparison raw data with no oversampling. It should be noted that sampling methods are often combined with cost sensitive methods and kernel methods to further boost learning. [3] [8] [9].

4.3 Base classifiers

We match the compared classifiers to classifiers used in other SMOTE extension evaluations. Six well-known classifiers are tested in experiments. The first is the Bayesian classifier based on kernel density estimation described in Section 2 (b-kde). The second is a K nearest neighbors classifier (knn). The third is a support vector machine classifier using RBF kernel (svm). The fourth one is a neural network (nn) with one hidden layer. We use in addition two ensemble methods: a random forest implementing the C4.5 decision tree [19] (rf) and Adaboost.M1 [20]. All hyper-parameters of the classifiers tested were determined by cross validation to ensure the best performance of each method.
### 4.4 Evaluation metric

Finding an appropriate evaluation metric for different tasks is challenging, since different evaluation metrics are designed for different purposes. The datasets used in this paper cover from financial application to medical treatment. To achieve an general evaluation and avoid bias, we follow the method in [2] [11] [12] [14] [10] and use different metrics to evaluate the performance of the proposed CGMOS oversampling algorithm.

Among these evaluation metrics, the most frequently adopted ones are **Precision** and **Recall** when the focus of evaluation is focus on one specific class such as problems in text classification, information extraction, natural language processing and bioinformatics. In these areas of application the number of examples belonging to one class is often substantially lower than the overall number of examples, which basically are imbalance learning problems. **Precision** and **Recall** are defined as:

\[
\text{Precision} = \frac{TP}{TP + FP} \\
\text{Recall} = \frac{TP}{TP + FN}
\]

However, these two metrics share an inverse relationship between each other. A quick inspection on the **Precision** and **Recall** formulas readily yields that solely use each of these two metrics only provide a limit view of an algorithm under test. As **Recall** provides no insight to how many examples are incorrectly labeled as positive and **Precision** cannot assert how many positive examples are labeled incorrectly. Specifically, the **F-score** combines **Precision** and **Recall** as measure of the effectiveness of classification in terms of a ratio of the weighted importance on either **Recall** or **Precision**, which is defined as:

\[
F\text{-score} = \frac{(1 + \beta^2) \cdot \text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}}
\]

We use $\beta = 1$ to treat **Precision** and **Recall** equally in all evaluations. As a result, **F-score** provides more insight into the functionality of a classifier.

As **F-score** measures the harmonic mean of **Precision** and **Recall**, we also compute **G-score** which is the geometric mean of **Precision** and **Recall** and is able to evaluate the degree of inductive bias in terms of a ratio of positive accuracy and negative accuracy [1].

\[
G\text{-score} = \sqrt{\text{Precision} \cdot \text{Recall}}
\]

As both **F-score** and **G-score** concentrate their measures on one class (positive examples) [21], to have a general way of comparing our test results, we altered the positive examples between the majority and minority classes when computing **F-score** and **G-score**. Thus we show **F-score** and **G-score** for the majority and the minority classes separately.

Although, both **F-score** and **G-score** are great evaluation metrics, they are still less effective in some situations. So we also employ the ROC graphs [22] [23] [24] in the evaluation. ROC graph is a two-dimensional graph, while **TP rate** and **FP rate** are its X axis and Y axis respectively.

An ROC graph basically manifest its usefulness by showing relative trade-off between benefit (true positive) and cost (false positive). One attractive property make ROC graph a good metric in imbalanced learning lies in the facts that ROC curve is insensitive to changes in class distribution. Because of this property, it is easier to see the performances of models trained by dataset oversampled by different algorithms. The goal in ROC space is to let curves be as close to upper-left-hand corner as possible, in which case the ratio between benefit and cost is maximized. To compare all test results in a more straightforward way, we also compute area under an ROC curve (AUC) which reduce the ROC performance to a single scalar value representing expected performance of the ROC curve.

### 4.5 Results

This section presents the performance of CGMOS and all the other methods on 30 real-world datasets. The same experiment procedure as the one in the experiments of the artificial dataset was conducted. All results are averaged from 10 rounds of 10-folds cross-validations. A summary of the experiment results is shown in Table 2 and ROC graphs are shown in Figure 2.

Considering the classification results of the minority class, it can be observed that the proposed approach outperforms most of the compared methods under all classification algorithms in terms of **F-score** and **G-score**. For **F-score** and **G-score** of
A summary of AUC, Precision, Recall, F-score, and G-score of all competitors for the majority and minority classes produced by 6 classifiers on the artificial datasets.
The majority class, the proposed approach in most cases is only second to the original data without oversampling. This is because the original dataset is imbalanced and it favors the majority class more than the minority class during classification. Overall, CGMOS achieves the best AUC over all tests. This is because the proposed approach takes into account both of the majority and minority classes and increases the certainties of the two classes while oversampling.

The same conclusion can be made from the ROC curves shown in Fig. 2. It could be seen from the ROC curves that the proposed approach has the highest values almost everywhere. The proposed approach achieves the best result when random forest is used as the classifier. For b-kde as the classifier, the proposed approach gets the largest improvement since the design of the proposed approach uses b-kde for certainty computations.

To get a closer view of the performances of all compared methods on each dataset, we show the AUC results of CGMOS approach uses b-kde for certainty computations. It could be seen from above table that CGMOS achieves best AUC measures for 24 datasets out of 30. By average, the AUC of CGMOS is at least 2 percent higher than all other competitors.

| AUC of 8 oversampling algorithms over all 30 datasets used in our evaluation. The AUC is averaged over all 6 base classifiers used in the evaluation. It could be seen from above table that CGMOS achieves best AUC measures for 24 datasets out of 30. By average, the AUC of CGMOS is at least 2 percent higher than all other competitors. | Average |
|---|---|---|---|---|---|---|
| Original | Dup | SMOTE | B-SMOTE | MWMOTE | ADASYN | RAMOboost |
| BreastCancer | 0.992 | 0.992 | 0.993 | 0.992 | 0.998 | 0.991 | 0.992 |
| BloodService | 0.733 | 0.653 | 0.648 | 0.649 | 0.651 | 0.720 | 0.714 |
| BreastTissue | 0.984 | 0.899 | 0.946 | 0.932 | 0.917 | 0.937 | 0.908 |
| CarEvaluation | 0.997 | 0.995 | 0.845 | 0.997 | 0.994 | 0.996 | 0.997 |
| Card'graphy | 0.977 | 0.976 | 0.939 | 0.962 | 0.956 | 0.925 | 0.957 |
| CharacterTraj | 0.985 | 0.962 | 0.717 | 0.985 | 0.978 | 0.981 | 0.988 |
| Chess | 0.977 | 0.974 | 0.959 | 0.973 | 0.977 | 0.974 | 0.975 |
| ClimateSim | 0.908 | 0.908 | 0.861 | 0.902 | 0.863 | 0.901 | 0.901 |
| Contraceptive | 0.724 | 0.705 | 0.699 | 0.712 | 0.702 | 0.705 | 0.702 |
| Fertility | 0.673 | 0.615 | 0.594 | 0.634 | 0.392 | 0.604 | 0.639 |
| Haberman | 0.651 | 0.625 | 0.597 | 0.600 | 0.593 | 0.594 | 0.587 |
| ILPD | 0.707 | 0.687 | 0.693 | 0.715 | 0.703 | 0.702 | 0.693 |
| ImgSeg | 0.999 | 0.998 | 0.999 | 0.997 | 0.998 | 0.989 | 0.997 |
| Leaf | 0.906 | 0.870 | 0.782 | 0.852 | 0.752 | 0.836 | 0.839 |
| Libras | 0.945 | 0.925 | 0.859 | 0.929 | 0.886 | 0.936 | 0.922 |
| MultipleFs | 0.998 | 0.998 | 0.997 | 0.998 | 0.997 | 0.997 | 0.996 |
| Parkinson | 0.841 | 0.676 | 0.692 | 0.834 | 0.791 | 0.837 | 0.842 |
| PlanRelax | 0.472 | 0.457 | 0.494 | 0.469 | 0.445 | 0.467 | 0.488 |
| QSAR | 0.901 | 0.886 | 0.879 | 0.895 | 0.863 | 0.886 | 0.886 |
| SPECT | 0.820 | 0.772 | 0.803 | 0.808 | 0.811 | 0.752 | 0.801 |
| SPECTF | 0.819 | 0.819 | 0.800 | 0.805 | 0.816 | 0.812 | 0.825 |
| SeismicBumps | 0.743 | 0.735 | 0.712 | 0.727 | 0.740 | 0.732 | 0.761 |
| Statlog | 0.998 | 0.992 | 0.996 | 0.998 | 0.990 | 0.996 | 0.996 |
| PlatesFaults | 0.956 | 0.928 | 0.844 | 0.954 | 0.920 | 0.943 | 0.956 |
| TAEvaluation | 0.748 | 0.682 | 0.644 | 0.703 | 0.671 | 0.707 | 0.665 |
| UserKnowledge | 0.958 | 0.837 | 0.919 | 0.953 | 0.947 | 0.951 | 0.950 |
| Vertebral | 0.890 | 0.839 | 0.869 | 0.855 | 0.829 | 0.860 | 0.860 |
| Customers | 0.952 | 0.930 | 0.943 | 0.946 | 0.884 | 0.902 | 0.946 |
| Yeast | 0.925 | 0.792 | 0.844 | 0.907 | 0.898 | 0.900 | 0.906 |
| Average | 0.864 | 0.827 | 0.808 | 0.844 | 0.830 | 0.842 | 0.842 |
We prove that CGMOS can achieve better classification results on improvement of the certainties for samples in both classes. We show the results of 6 different classifiers: b-kde, knn, svm, nn, rf and Adaboost.M1. Curves in blue are the results of the proposed CGMOS.

signed-ranks test is that the median difference between pairs of observations is zero.

The test results are shown in Table[4] It could be seen from the table that the p-value of all tests is smaller than 0.05 and pass the test.

5 Conclusion

In this paper, we address the imbalanced binary classification problem by proposing a novel minority oversampling strategy. Different from existing approaches, CGMOS does not randomly synthesize new data along decision boundaries. Instead, CGMOS computes the Bayes classification certainties for both the majority and minority classes and then synthesize new samples based on improvement of the certainties for samples in both classes. We prove that CGMOS can achieve better classification results compared with SMOTE. In addition, experimental results show that CGMOS outperforms known oversampling techniques using various metrics.
REFERENCES

[1] H. He and E. A. Garcia, “Learning from imbalanced data,” Knowledge and Data Engineering, IEEE Transactions on, vol. 21, no. 9, pp. 1263–1284, 2009.

[2] G. M. Weiss, “Mining with rarity: a unifying framework,” ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 7–19, 2004.

[3] N. V. Chawla, N. Japkowicz, and A. Kotcz, “Editorial: special issue on learning from imbalanced data sets,” ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 1–6, 2004.

[4] X.-Y. Liu, J. Wu, and Z.-H. Zhou, “Exploratory undersampling for class-imbalance learning,” Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, vol. 39, no. 2, pp. 539–550, 2009.

[5] J. Zhang and I. Mani, “Knn approach to unbalanced data distributions: A case study involving information extraction,” in Intl Conf. Machine learning, workshop learning from imbalanced data sets, 2003.

[6] C. Drummond and R. C. Holte, “C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling,” in Workshop on Learning from Imbalanced Data Sets II, International Conference on Machine Learning, 2003, pp. 1–8.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.

[8] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, “Smoteboost: Improving prediction of the minority class in boosting,” Subseries of Lecture Notes in Computer Science, p. 107, 2003.

[9] H. Guo and H. L. Viktor, “Learning from imbalanced data sets with boosting and data generation: the databoost-im approach,” ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 30–39, 2004.

[10] S. Chen, H. He, E. Garcia et al., “Ramoboost: Ranked minority oversampling in boosting,” Neural Networks, IEEE Transactions on, vol. 21, no. 10, pp. 1624–1642, 2010.

[11] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-smote: a new oversampling method in imbalanced data sets learning,” in Advances in intelligent computing. Springer, 2005, pp. 878–887.

[12] H. He, Y. Bai, E. A. Garcia, and S. Li, “Adasyn: Adaptive synthetic sampling approach for imbalanced learning,” in Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on. IEEE, 2008, pp. 1322–1328.

[13] S. Barua, M. M. Islam, and K. Murase, “A novel synthetic minority oversampling technique for imbalanced data set learning,” in Neural Information Processing. Springer, 2011, pp. 735–744.

[14] S. Barua, M. M. Islam, X. Yao, and K. Murase, “Mmote-majority weighted minority oversampling technique for imbalanced data set learning,” Knowledge and Data Engineering, IEEE Transactions on, vol. 26, no. 2, pp. 405–425, 2014.

[15] M. Sharma and M. Bilgic, “Most-surely vs. least-surely uncertain,” in Data Mining (ICDM), 2013 IEEE 13th International Conference on, 2013, pp. 667–676. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6729551

[16] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, “Background and foreground modeling using nonparametric kernel density estimation for visual surveillance,” Proceedings of the IEEE, vol. 90, no. 7, pp. 1151–1163, 2002.

[17] X. Zhang, M. L. King, and R. J. Hyndman, “A bayesian approach to bandwidth selection for multivariate kernel density estimation,” Computational Statistics & Data Analysis, vol. 50, no. 11, pp. 3009–3031, 2006.

[18] M. Lichman, “UCI machine learning repository,” 2013.

[19] J. R. Quinlan, C4.5: Programs for Machine Learning. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[20] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” in Proceedings of the Thirteenth International Conference on Machine Learning (ICML 1996), L. Saitta, Ed. Morgan Kaufmann, 1996, pp. 148–156. [Online]. Available: http://www.biostat.wisc.edu/~kbroman/teaching/statgen/2004/refs/freund.pdf

[21] M. Sokolova, N. Japkowicz, and S. Szpakowicz, “Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation,” in AI 2006: Advances in Artificial Intelligence. Springer, 2006, pp. 1015–1021.

[22] T. Fawcett, “Roc graphs: Notes and practical considerations for researchers,” Machine learning, pp. 1–38, 2004.

[23] M. Dzeroski, “An introduction to roc analysis,” Pattern recognition letters, vol. 27, no. 8, pp. 861–874, 2006.

[24] C. Mohri, “Confidence intervals for the area under the roc curve,” in Advances in neural information processing systems, 2005, p. 305.

[25] G. E. Batista, R. C. Prati, and M. C. Monard, “A study of the behavior of several methods for balancing machine learning training data,” ACM Sigkdd Explorations Newsletter, vol. 6, no. 1, pp. 20–29, 2004.

[26] 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, pp. 315–354, 2003.

[27] J. L. Devore, Probability and Statistics for Engineering and the Sciences. Duxbury Press, 2011.

[28] S. McKillup, Statistics Explained: An Introductory Guide for Life Scientists. Cambridge University Press, 2006.

[29] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” The Journal of Machine Learning Research, vol. 7, pp. 1–30, 2006.

[30] G. W. Corder and D. I. Foreman, Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach. Wiley, 2009.

[31] B. F. PhD, High-Yield(TM) Behavioral Science (High-Yield Series). LWL, 2008.

[32] D. W. Zimmerman, “Teacher’s corner: A note on interpretation of the paired-samples t test,” Journal of Educational and Behavioral Statistics, vol. 22, no. 3, pp. 349–360, 1997.