Applying Adaptive Over-sampling Technique Based on Data Density and Cost-Sensitive SVM to Imbalanced Learning

Senzhang Wang\textsuperscript{1,2}, Zhoujun Li\textsuperscript{1,2,*}, Wenhan Chao\textsuperscript{1,2} and Qinghua Cao\textsuperscript{3}

Abstract—Resampling method is a popular and effective technique to imbalanced learning. However, most resampling methods ignore data density information and may lead to overfitting. A novel adaptive over-sampling technique based on data density (ASMOBD) is proposed in this paper. Compared with existing resampling algorithms, ASMOBD can adaptively synthesize different number of new samples around each minority sample according to its level of learning difficulty. Therefore, this method makes the decision region more specific and can eliminate noise. What's more, to avoid over generalization, two smoothing methods are proposed. Cost-Sensitive learning is also an effective technique to imbalanced learning. In this paper, ASMOBD and Cost-Sensitive SVM are combined. Experiments show that our methods perform better than various state-of-art approaches on 9 UCI datasets by using metrics of G-mean and area under the receiver operation curve (AUC).

Keywords-over-sampling; Cost-sensitive SVM; imbalanced learning

I. INTRODUCTION

In practical application, many datasets are imbalanced, i.e., some classes have much more instances than others. Imbalanced learning is common in many situations like information filtering [1] and fraud detection [2]. Datasets imbalance must be taken into consideration in classifier designing, otherwise the classifier may tend to be overwhelmed by the majority class and to ignore the minority class.

Resampling technique is an effective approach to imbalance learning. Many resampling methods are used to reduce or eliminate the extent of datasets imbalance, such as over-sampling the minority class, under-sampling the majority class and the combination of both methods. But [3] showed that under-sampling can potentially remove certain important instances and lose some useful information, and over-sampling may lead to overfitting. Over-sampling methods also suffer from noise and outliers [4].

Support Vector Machine (SVM) has been widely used in many application areas of machine learning. However, regular SVM is no longer suitable to imbalance-class especially when the datasets are extremely imbalanced. An effective approach to improve the performance of SVM used in imbalanced datasets is to bias the classifier so that it pays more attention to minority instances. This can be done by setting different misclassifying penalty [5].

We proposed an over-sampling algorithm based on data density in previous work [6]. However, this algorithm sometimes leads to overfitting. In this paper, an adaptive over-sampling algorithm with two smoothing methods to avoid overfitting is proposed. Compared with other over-sampling algorithms and our previous work, this algorithm can synthesize samples more efficiently and eliminate the effects of noise. Contributions of this paper are as follows:

--This novel method can effectively eliminate the noise compared with most other sampling methods like RO and SMOTE. Noise is recognized and no new samples are synthesized around it.

--Different number new samples are synthesized around each minority sample according to its level of learning difficulty. This level is related to the sample density information. To calculate the sample density, core-distance and reachability-distance are used [7]. We will elaborate this idea in section IV.

--To avoid overfitting, two smoothing methods are proposed. One is using a sigmoid function to smooth the disparity of new samples synthesized around each minority sample. The other is using linear interpolation method to tradeoff between our algorithm and SMOTE algorithm. Experiments show that both methods are effective.

The rest of the paper is organized as follows: Section II reviews related works. Section III gives an overview of performance measures. Section IV details our approach. Section V presents experimental results comparing our approach with other approaches. Section VI discusses the result and concludes this paper.

II. RELATED WORK

Resampling techniques are widely used in imbalanced learning such as random over-sampling (RO), random undersampling (RU) and over-sampling with informed generation of new samples. [4] proposed an algorithm-SMOTE to over-sampling minority datasets. This algorithm synthesizes new samples along the line between the minority and their selected nearest neighbors. The disadvantage of SMOTE is that it makes the decision regions larger and less specific [8]. SMOTE-ENN and SMOTE-Tomek are two popular methods combining sampling technique and data cleaning technique. Experiments in [3] show that these two
In this paper, we use G-mean and AUC as performance measures. The metrics and some other parameters we used are defined as follows:

\[
\text{acc}_+ = \text{sensitivity} = \text{Recall} = \frac{TP}{(TP + FN)} \quad (1)
\]
\[
\text{acc}_- = \text{specificity} = \frac{TN}{(TN + FP)} \quad (2)
\]
\[
G - \text{mean} = \sqrt{\text{acc}_+ \cdot \text{acc}_-} = \sqrt{\text{sensitivity} \cdot \text{specificity}} \quad (3)
\]

ROC curves can be thought of as the representative of the family of best decision boundaries for relative costs of TP and FP. On the ROC curve the X-axis represents FPR = FP/ (TN + FP) and the y-axis represents TPR = TP/ (TP + FN). AUC is the area below the curve. Figure 1 shows an illustration. Figure 1 is the ROC of sick dataset in UCI repository. The line y=x represents the scenario of random guess. The larger the AUC is the better the performance of classifier is.

![Figure 1](image)

**Figure 1:** Example of ROC curve. The line y=x represents randomly guessing. The more the ROC tilts toward the top-left corner, the larger the AUC of the ROC and the better performance of the classifier.

### IV. ASMOBD AND ASMOBD-CS

Regular over-sampling algorithms, like over-sampling with random replacement and SMOTE, ignore the density and distribution information of the datasets and suffer from the problem of outliers and noise.

SMOTE-ENN and SMOTE-Tomek are two effective algorithms to eliminate the noise in comparison to SMOTE, but experiments in [21] show that these algorithms may not provide better performance than random over-sampling when the number of minority class is large. In this section, firstly, the novel over-sampling algorithm ASMOBD we proposed will be described in detail; secondly, to avoid overfitting, two smoothing methods are proposed; thirdly, SVM with different error costs will be described briefly.

#### A. Adaptive Over-sampling Technique Based on samples Density (ASMOBD)

The proposed algorithm can adaptively synthesize different number new samples around each minority sample according to its level of learning difficulty. As shown in Fig. 2, the minority sample's level of learning difficulty depends on both the local minority sample density and local majority sample density. What's more, to avoid over generalization, an imbalanced factor which is determined by the ratio

<table>
<thead>
<tr>
<th>Actual Positive Class</th>
<th>Predicted Positive Class</th>
<th>Predicted Negative Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Positive Class</td>
<td>TP (True Positive)</td>
<td>TN (True Negative)</td>
</tr>
<tr>
<td>Actual Negative Class</td>
<td>FP (False Positive)</td>
<td>FN (False Negative)</td>
</tr>
</tbody>
</table>
between local majority and minority samples is proposed.

**Figure 2:** Example of ASMOBD. Triangles and circles represent samples of two different classes. Dash circles represent core-distance of sample A, B, C, D and E. A is easy to be classified correctly, so a small number of new samples will be synthesized. B and C are hard to be classified, so more new samples will be synthesized around D in order to avoid overfitting, because too many majority samples are in its core-distance. E is noise, so no new samples will be synthesized around it.

Fig. 2 shows the intuitive idea of our method. The triangles represent the minority samples, the circles represent the majority samples and the black triangles represent the minority samples around which new samples will be synthesized (except noise E). The level of learning difficulty for minority samples is determined by three factors: the local density of minority samples, the local density of majority samples and the local imbalanced ratio. The minority sample is easy to be classified when its local minority density is large and local majority density is small. On the contrary, the minority sample is hard to be classified when its local minority density is small and local majority density is large.

More new samples will be synthesized around the minority samples which are harder to be classified correctly while less new samples will be synthesized around the minority samples which are easier to be classified correctly. To avoid over generalization, an imbalanced factor which is determined by the ratio between local majority and minority samples is proposed. In Fig. 2, though sample D is very hard to be classified correctly, less new samples will be synthesized around it for its high imbalanced ratio. For noise, no new samples will be synthesized around it like sample E.

A data density based clustering algorithm OPTICS was proposed in [7], which could identify clustering structure. The core-distance and reachability-distance proposed in this algorithm fully reflect the density information of the datasets and noisy samples will be judged by the two distance. The core-distance and reachability-distance are defined as follows:

**Definition 1** [7]. Core-distance of an object \( p \): Let \( p \) be an object from a database \( D \), let \( \varepsilon \) be a distance value, let \( N_\varepsilon(p) \) be \( \varepsilon \)-neighborhood of \( p \), let \( \text{card}(N_\varepsilon(p)) \) denote the cardinality of the set \( N_\varepsilon(p) \), let \( \text{MinPts} \) be a natural number and let \( \text{MinPts} \cdot \text{distance}(p) \) be the distance from \( P \) to its \( \text{MinPts} \)’ neighbor. Then, the core-distance of \( p \) is defined as \( \text{core} - \text{distance}_{\varepsilon} \cdot \text{MinPts} (p) = \)

\[
\begin{align*}
\text{UNDEFINED} & \quad \text{if Card}(N_\varepsilon(p)) < \text{MinPts} \\
\text{Max} & \quad \text{if Card}(N_\varepsilon(p)) \geq \text{MinPts} \\
\text{Max} & \quad \text{else}
\end{align*}
\]

**Definition 2** [7]. Reachability-distance object \( p \) w.r.t. object \( a \): Let \( p \) and \( a \) be objects from a database \( D \), let \( N_\varepsilon(a) \) be the \( \varepsilon - \text{neighborhood} \) of \( a \), and let \( \text{MinPts} \) be a natural number. Then, the reachability-distance of \( p \) with respect to \( a \) is defined as \( \text{reachability} - \text{distance}_{\varepsilon} \cdot \text{MinPts} (p,a) = \)

\[
\begin{align*}
\text{UNDEFINED} & \quad \text{if } N_\varepsilon(a) < \text{MinPts} \\
\text{Max} & \quad \text{if } N_\varepsilon(a) \geq \text{MinPts} \\
\text{Max} & \quad \text{else}
\end{align*}
\]

Intuitively, the core-distance of an object \( p \) is the smallest distance between \( p \) and an object in its \( \varepsilon - \text{neighborhood} \) such that \( p \) would be a core object. The reachability-distance of an object \( p \) with respect to another object \( o \) is the smallest distance such that \( p \) is directly density-reachable from \( o \) if \( o \) is a core object.

**Definition 3.** Noise based on core-distance and reachability-distance: Let \( p \) and \( a \) be objects from a database \( D \). If \( \text{core} - \text{distance}_{\varepsilon} \cdot \text{MinPts} (p) \) is beyond the threshold \( T_1 \), and \( \text{reachability} - \text{distance}_{\varepsilon} \cdot \text{MinPts} (p,a) \) is beyond the threshold \( T_2 \) ( \( T_1 \) and \( T_2 \) are all predefined), then \( p \) is a noise.

\[
\text{NOISE}_p = \begin{cases} 1 & \text{if } \text{core} - \text{distance}_{\varepsilon} \cdot \text{MinPts} (p) > T_1 \text{ and } \text{reachability} - \text{distance}_{\varepsilon} \cdot \text{MinPts} (p,a) > T_2 \\ 0 & \text{else} \end{cases}
\]

According to the above definitions, Density of new samples synthesized around each minority sample is computed by the formula below:

\[
\text{DF}_{\varepsilon} = \eta \cdot \varepsilon_i + (1 - \eta) \cdot \text{card}_{\text{majority}}(N_\varepsilon(x_i)).
\] (4)

\( \text{DF}_{\varepsilon} \) means the density of new samples synthesized around the sample \( x_i \). \( \varepsilon_i \) represents the core-distance of sample \( x_i \). \( \text{card}_{\text{majority}}(N_\varepsilon(x_i)) \) is the number of majority samples which are included in the hyper sphere with a radius of core-distance of sample \( x_i \). \( \varepsilon_i \) and \( \text{card}_{\text{majority}}(N_\varepsilon(x_i)) \) are all normalized. \( \eta \) is weighting coefficient tradeoff...
between \( \varepsilon \) and \( N_{\text{le}} \). We set \( \eta \) to 0.5, and in most cases, 0.5 is an appropriate value for \( \eta \) in our experiment.

The number of new samples synthesize around each samples is computed by the formula below:

\[
N_i = \frac{DF_i \cdot N}{\sum_{j=1}^{DF_i} N} \tag{5}
\]

\( N \) means the total number of new samples synthesized. The algorithm of synthesizing new samples is similar to SMOTE. The only difference is no new samples are synthesized around noisy samples and the number of new samples around each minority sample is different. The noisy sample is judged by the formula below:

\[
\begin{aligned}
\text{NOISE}_i = \begin{cases} 
1 & \text{if } CD_i > \frac{1}{N_{\text{min}}} \sum_{i=1}^{N_{\text{min}}} CD_i \cdot t_1 \\
RD_i > \frac{1}{N_{\text{min}}} \sum_{i=1}^{N_{\text{min}}} RD_i \cdot t_2 \\
0 & \text{else}
\end{cases}
\end{aligned} \tag{6}
\]

\( \text{NOISE}_i \) is 1 if \( x_i \) is a noisy sample and 0 if not. Array CD[] and RD[] store the core-distance and reachability-distance of all the minority samples. \( N_{\text{min}} \) is the number of minority samples. \( t_1 \) and \( t_2 \) are noise threshold coefficients predefined. In our experiment, we set \( t_1 \) and \( t_2 \) to 4. It means that if the core-distance and reachability-distance of sample \( x_i \) is 4 times larger than the average core-distance and reachability-distance, sample \( x_i \) is considered to be a noise.

**B. Sigmoid Function Smoothing**

The method we proposed above can effectively synthesize different number of new samples around each minority sample according to its level of learning difficulty. However, overfitting may still exist. For example, some minority samples will synthesize a large number of new samples while some minority samples will not synthesize any new samples. To address this problem, two smoothing methods are proposed.

The first approach is using a sigmoid function to smoothing equation (4). Smoothing details is as follows:

\[
\begin{align*}
DF \_ S_i &= DF_i - \text{Balance} \_ \text{ratio}_i \\
&= \eta \cdot \varepsilon + (1 - \eta) \cdot \text{card}_\text{majority}(N_c(x_i)) - \text{Balance} \_ \text{ratio}_i \\
\text{Balance} \_ \text{ratio}_i &= \text{abs}\left(\frac{2}{1 + \exp[-a \cdot \text{Ratio}_i]} - 1\right) \\
\text{Ratio}_i &= \frac{\text{Maj} \_ \text{num}}{\text{Min} \_ \text{num}} \tag{7, 8, 9}
\end{align*}
\]

\( DF \) is smoothed by subtracting a value \( \text{Balance} \_ \text{ratio} \), which is calculated by a sigmoid function. In this function, \( a \) is a weighting coefficient and \( \text{Ratio}_i \) is the imbalance ratio between the local majority samples and local minority samples. In our experiment, \( a \) is set to 0.05 or 0.1. \( \text{Maj} \_ \text{num} \) and \( \text{Min} \_ \text{num} \) mean the number of majority samples and minority samples in the \( \varepsilon \)-neighbourhood of sample \( x_i \) respectively.

**C. Linear Interpolation Smoothing**

The second smoothing approach we proposed is linear interpolation. This approach is a combination between SMOTE and ASMOBD. SMOTE ignores the data density information and ASMOBD may lead to overfitting because of overemphasizing data density information. We combine both methods.

The linear interpolation smoothing is proposed as follows:

\[
DF \_ L_i = \mu \cdot DF_i + (1 - \mu) \cdot k \tag{10}
\]

In equation (10), \( \mu \) is the smoothing coefficient. \( k \) is the proportion of over-sampling with SMOTE method. Equation (10) shows that linear interpolation smoothing method is a compromise between SMOBD and SMOTE. Intuitively, this method is a compromise between individuation and generality.

Pseudo code of our algorithm is as follows:
1. **procedure** AS_S/AS_LI
2. **Input**: Dataset \( D \)
3. **Output**: new Dataset \( D_{\text{new}} \)
4. //Calculate the core-distance and reachability-distance of each minority sample using OPTICS
\[
\begin{align*}
\text{core-distance}(D_{\text{minority}}) & \leftarrow \text{OPTICS}(D,k,\text{threshold}) \\
\text{reachability-distance}(D_{\text{minority}}) & \leftarrow \text{OPTICS}(D,k,\text{threshold})
\end{align*}
\]
5. Eliminate noise according to core-distance and reachability-distance using Definition 3 and formula (6).
6. //Calculate the new synthesized samples density of each minority sample \( x_i \)
\[
DF_i = \eta \epsilon_i + (1 - \eta) \text{card}_{majority}(N_i(x_i)).
\]
7. Smoothing
   Method 1: \( DF_S = DF - \text{Balance}\_\text{ratio} \)
   Method 2: \( DF_L = \mu DF + (1 - \mu) k \)
8. //Calculate the number of new synthesized samples around each minority sample \( x_i \)
\[
N_i = \frac{DF_i \cdot N}{\sum DF_i}
\]
9. Synthesize new samples similar to SMOOTE method, and output new dataset \( D_{\text{new}} \)
10. **end procedure**

Fig. 5 shows an example of the results of the two smoothing methods we proposed. We use the sick dataset of UCI in this example. \( k \) value is 400%. The horizontal axis represents each minority sample and the vertical axis represents the number of new samples synthesized around each minority sample.

![Figure 5](image_url)

**Figure 5**: An example of the results of two smoothing methods

This example demonstrates without smoothing, large number of new samples will be synthesized around a few minority samples while no new samples will be synthesized around some minority samples. The two smoothing methods effectively reduce the imbalance.

For the linear interpolation smoothing method, a difficult, yet important problem is how to determine the value of parameter \( \mu \). Some experiments are made to test how parameter \( \mu \) influences the performance of classification.

Three datasets of UCI are used here: *hypothyroid*, *abalone* and *sick*. We use the G-mean value as performance measure. The experiments results are as follows:

![Figure 6](image_url)

**Figure 6**: G-mean for different \( \mu \) values for hypothyroid dataset

![Figure 7](image_url)

**Figure 7**: G-mean for different \( \mu \) values for abalone dataset

![Figure 8](image_url)

**Figure 8**: G-mean for different \( \mu \) values for sick dataset

The horizontal axis shows the proportion of over-sampling and the vertical axis shows the G-mean value of the method we proposed with linear interpolation smoothing. Four different \( \mu \) values are tested in our experiment, \( \mu = 0.2 \), \( \mu = 0.4 \), \( \mu = 0.6 \) and \( \mu = 0.8 \). The experiments results show that \( \mu = 0.4 \) and \( \mu = 0.6 \) are better than \( \mu = 0.2 \) and \( \mu = 0.8 \). For \( \mu = 0.2 \), SMOTE overwhelms ASMOBD and for \( \mu = 0.8 \), ASMOBD overwhelms SMOTE. The experiments results are also in accordance with intuition: the combination of two methods can improve performance.
D. SVM with different error costs and ASMOBD-CS

As mentioned above, regular SVM is invalid to the imbalanced datasets. [14] showed that with imbalanced datasets, the learned boundary is too close to the minority samples, so SVM should be biased in a way that will push the boundary away from the positive samples. [5] suggested using different error costs for the positive ($S_+$) and negative ($S_-$) classes. The classifier function equates to solve the quadratic programming problem as follows:

$$J(\omega, \omega_0, \xi) = \frac{1}{2} \left\| \omega \right\|^2 + C^+ \sum_{[i]:y_i=1} \xi_i + C^- \sum_{[i]:y_i=-1} \xi_i$$

(11)

Subject to: $y_i(\omega^T x_i + \omega_0) \geq 1 - \xi_i$

(12)

$\xi_i \geq 0, \; i = 1, \ldots, n_+ + n_-$

(13)

In (11), $C^+$ represents the cost of misclassifying the positive sample and $C^-$ represents the cost of misclassifying the negative sample. It was reported in [9] that the optimal result could be obtained when $C^-/C^+$ equals to the minority-to-majority class ratio. The $C^-/C^+$ is determined by formula below:

$$C^- = \frac{\text{Num Min}_\text{Minority} \times k}{\text{Num Min}_\text{Majority}}$$

(14)

$\text{Num Min}_\text{Minority}$ is the number of minority samples, $k$ is the proportion of over-sampling and $\text{Num Min}_\text{Majority}$ is the number of majority samples.

With different error costs, the boundary is pushed more towards the majority samples. [14] showed that SVM with different error costs may obtain stronger cues from the majority samples than from the minority samples about the orientation of the plane. Consequently, the combination of the two methods can achieve better performance.

ASMOBD-CS combines ASMOBD and cost-sensitive SVM. Though imbalance rate is reduced by over-sampling, it still exists. To further reduce the imbalance rate, different error costs are proposed according to the reduced imbalance rate.

V. EXPERIMENT AND RESULT

A. Datasets

9 UCI datasets are used to test the algorithms we proposed. Information about these datasets is summarized in Table II. When more than two classes exist in the dataset, one class is considered to be positive and all the other classes are considered to be negative.

In our experiments, G-mean and area under curve (AUC) are used as metrics. For each dataset, we perform 5-fold cross validation. In each fold four out of five samples are selected to be training set, and the left one out of five samples is testing set. This process repeats 5 times so that all samples are selected in both training set and testing set.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Attributes</th>
<th>#Positive</th>
<th>#Negative</th>
<th>#Imbalance Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>9</td>
<td>32</td>
<td>4145</td>
<td>130</td>
</tr>
<tr>
<td>hypothyroid</td>
<td>29</td>
<td>95</td>
<td>3677</td>
<td>39</td>
</tr>
<tr>
<td>sick</td>
<td>29</td>
<td>231</td>
<td>3541</td>
<td>15</td>
</tr>
<tr>
<td>glass</td>
<td>9</td>
<td>29</td>
<td>185</td>
<td>6</td>
</tr>
<tr>
<td>car</td>
<td>6</td>
<td>69</td>
<td>1659</td>
<td>24</td>
</tr>
<tr>
<td>prima</td>
<td>8</td>
<td>268</td>
<td>500</td>
<td>2</td>
</tr>
<tr>
<td>hepatitis</td>
<td>19</td>
<td>32</td>
<td>123</td>
<td>4</td>
</tr>
<tr>
<td>segment</td>
<td>19</td>
<td>330</td>
<td>1980</td>
<td>6</td>
</tr>
<tr>
<td>auto-mgp</td>
<td>7</td>
<td>68</td>
<td>324</td>
<td>5</td>
</tr>
</tbody>
</table>

B. Experiment between ASMOBD and other over-sampling methods

We compare our methods with SMOTE and random over-sampling (RO). In our experiments, we use ASMOBD with two smoothing methods. The sigmoid function smoothing method is noted as AS_S and the linear interpolation smoothing method is noted as AS_LI.
Fig 9, Fig 10 and Fig 11 show the G-mean metric of each method for hypothyroid, abalone and sick datasets of UCI repository. The experiments results show that the AS_S and AS_LI methods we proposed are both better than RO and SMOTE methods for almost all over-sampling proportions for the three datasets we used.

C. Experiment between ASMOBD and ASMOBD with Cost-sensitive SVM

We also make experiments between ASMOBD and ASMOBD with cost-sensitive SVM. These experiments indicate the combination of over-sampling method and cost-sensitive learning can further improve the performance of imbalanced learning.

The two methods we proposed above with cost-sensitive SVM are noted as AS_S_CS and AS_LI_CS respectively. hypothyroid and abalone datasets are used in the experiments. The SMOTE with cost-sensitive SVM method (SDC) [14] is used to be a baseline method.

Fig 12 and Fig 13 show the G-mean metric of the experiment. The results indicate that the combination of over-sampling and cost-sensitive learning can further improve the G-mean metric of imbalanced learning. However, the improvement is not remarkable in our experiments. Moreover, with the increasing of over-sampling proportion, the performance of the two methods is not that big of a difference. Experiment also shows that AS_S_CS and AS_LI_CS are both better than SDC in the G-mean metric in the two datasets.

D. Experiment among 9 methods using 9 datasets

We compared 9 methods: Random over-sampling (RO), SMOTE, Borderline SMOTE, cost-sensitive SVM, SMOTE with cost-sensitive SVM (SDC), AS_S, AS_LI, AS_S_CS and AS_LI_CS, on 9 datasets described in TABLE II.

For each dataset, we calculate the G-mean and AUC metrics under different over-sampling proportions and average them. For example, the over-sampling proportion is from 100% to 1500% for abalone dataset. We sample from 100% to 1500% and calculate the average G-mean and AUC. The upper limit of over-sampling proportion depends on the imbalance ratio of each dataset.

G-mean and AUC metrics described in section III are both used in our experiments. The best experiments results are denoted by bold body and black underlines. Experiments results are shown in Table III and Table IV. Table III is the G-mean metric table and Table IV is the AUC metric table.

Results demonstrate that in eight out of nine datasets, AS_S_CS and AS_LI_CS have the highest G-means and AUC value. Comparison among RO, SMOTE, Borderline SMOTE, AS_S and AS_LI demonstrates that the two methods we proposed outperform other methods in most datasets. Experiments demonstrate AS_S and AS_LI synthesize new samples more effectively than SMOTE. SMOTE and other algorithms synthesize the same number of new samples around each minority sample, so some useful information is lost. Moreover, noise is an important factor to influence the performance improvement for SMOTE. The comparison between AS_S, AS_LI and AS_S_CS, AS_LI_CS demonstrates the combination of over-sampling and cost-sensitive SVM can further improve the performance of classifier.

VI. CONCLUSION

A novel adaptive over-sampling technique based on data density information is proposed in this paper. We also combine the new over-sampling method with cost-sensitive SVM. Empirical results show that our methods perform better than state-of-art approaches like RO, SMOTE, Borderline SMOTE and SDC on a variety of datasets by using G-mean, area under the receiver operation curve (AUC) metrics.

Though ASMOBD and ASMOBD-CS can achieve better performance in most cases, many problems still need to be addressed. Firstly, there are some parameters in our algorithms. The performance of our algorithms varies a lot with different values of parameters. How to find the best parameter value to achieve the best performance is a problem we need to solve in the future work. Secondly, more time is needed to synthesize the same number of new samples in our method than SMOTE. To compute density of each sample, reachability-distance and core-distance need to be computed firstly, which consumes much more time than that of SMOTE. Computation complexity reduction is another work we need to do in the future.

ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China [grant number 60973105, 90718017, 61170189], the Research Fund for the Doctoral Program of Higher Education [grant number 20111102130003] and the Fund of the State Key Laboratory of Software Development Environment [grant number SKLSDE-2011ZX-03].
TABLE III. G-MEAN METRIC OF EXPERIMENT RESULT

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Over-sampling proportion</th>
<th>RO</th>
<th>SMOTE</th>
<th>Bordeline SMOTE</th>
<th>CS</th>
<th>AS_S</th>
<th>AS_LI</th>
<th>SDC</th>
<th>AS_S_CS</th>
<th>AS_LI_CS</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>100% - 1500%</td>
<td>0.834</td>
<td>0.802</td>
<td>0.840</td>
<td>0.340</td>
<td>0.851</td>
<td>0.826</td>
<td>0.817</td>
<td>0.869</td>
<td>0.844</td>
</tr>
<tr>
<td>hypothyroid</td>
<td>100% - 1000%</td>
<td>0.860</td>
<td>0.860</td>
<td>0.920</td>
<td>0.781</td>
<td>0.969</td>
<td>0.974</td>
<td>0.863</td>
<td>0.986</td>
<td>0.982</td>
</tr>
<tr>
<td>sick</td>
<td>100% - 1000%</td>
<td>0.881</td>
<td>0.912</td>
<td>0.910</td>
<td>0.671</td>
<td>0.925</td>
<td>0.928</td>
<td>0.925</td>
<td>0.938</td>
<td>0.940</td>
</tr>
<tr>
<td>glass</td>
<td>100% - 500%</td>
<td>0.910</td>
<td>1.000</td>
<td>0.992</td>
<td>0.992</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>car</td>
<td>100% - 1000%</td>
<td>0.977</td>
<td>0.966</td>
<td>0.985</td>
<td>0.460</td>
<td>0.980</td>
<td>0.981</td>
<td>0.981</td>
<td>0.985</td>
<td>0.986</td>
</tr>
<tr>
<td>prima</td>
<td>100% - 500%</td>
<td>0.702</td>
<td>0.700</td>
<td>0.710</td>
<td>0.673</td>
<td>0.709</td>
<td>0.711</td>
<td>0.702</td>
<td>0.718</td>
<td>0.721</td>
</tr>
<tr>
<td>hepatitis</td>
<td>100% - 500%</td>
<td>0.873</td>
<td>0.869</td>
<td>0.940</td>
<td>0.542</td>
<td>0.900</td>
<td>0.894</td>
<td>0.890</td>
<td>0.913</td>
<td>0.910</td>
</tr>
<tr>
<td>segment</td>
<td>100% - 1000%</td>
<td>0.910</td>
<td>0.917</td>
<td>0.965</td>
<td>0.842</td>
<td>0.955</td>
<td>0.960</td>
<td>0.932</td>
<td>0.974</td>
<td>0.982</td>
</tr>
<tr>
<td>auto-mgp</td>
<td>100% - 800%</td>
<td>0.890</td>
<td>0.924</td>
<td>0.942</td>
<td>0.575</td>
<td>0.940</td>
<td>0.934</td>
<td>0.930</td>
<td>0.952</td>
<td>0.963</td>
</tr>
</tbody>
</table>

TABLE IV. AUC METRIC OF EXPERIMENT RESULT

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Over-sampling proportion</th>
<th>RO</th>
<th>SMOTE</th>
<th>Bordeline SMOTE</th>
<th>CS</th>
<th>AS_S</th>
<th>AS_LI</th>
<th>SDC</th>
<th>AS_S_CS</th>
<th>AS_LI_CS</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>100% - 1500%</td>
<td>0.828</td>
<td>0.837</td>
<td>0.842</td>
<td>0.67</td>
<td>0.854</td>
<td>0.823</td>
<td>0.831</td>
<td>0.864</td>
<td>0.842</td>
</tr>
<tr>
<td>hypothyroid</td>
<td>100% - 1000%</td>
<td>0.869</td>
<td>0.869</td>
<td>0.927</td>
<td>0.804</td>
<td>0.971</td>
<td>0.980</td>
<td>0.863</td>
<td>0.982</td>
<td>0.973</td>
</tr>
<tr>
<td>sick</td>
<td>100% - 1000%</td>
<td>0.888</td>
<td>0.914</td>
<td>0.915</td>
<td>0.723</td>
<td>0.927</td>
<td>0.923</td>
<td>0.924</td>
<td>0.937</td>
<td>0.936</td>
</tr>
<tr>
<td>glass</td>
<td>100% - 500%</td>
<td>0.924</td>
<td>1.000</td>
<td>0.990</td>
<td>0.990</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>car</td>
<td>100% - 1000%</td>
<td>0.980</td>
<td>0.967</td>
<td>0.985</td>
<td>0.600</td>
<td>0.981</td>
<td>0.984</td>
<td>0.982</td>
<td>0.986</td>
<td>0.987</td>
</tr>
<tr>
<td>prima</td>
<td>100% - 500%</td>
<td>0.738</td>
<td>0.735</td>
<td>0.720</td>
<td>0.698</td>
<td>0.740</td>
<td>0.745</td>
<td>0.737</td>
<td>0.742</td>
<td>0.747</td>
</tr>
<tr>
<td>hepatitis</td>
<td>100% - 500%</td>
<td>0.875</td>
<td>0.870</td>
<td>0.944</td>
<td>0.607</td>
<td>0.900</td>
<td>0.894</td>
<td>0.893</td>
<td>0.918</td>
<td>0.910</td>
</tr>
<tr>
<td>segment</td>
<td>100% - 1000%</td>
<td>0.917</td>
<td>0.922</td>
<td>0.960</td>
<td>0.874</td>
<td>0.960</td>
<td>0.962</td>
<td>0.943</td>
<td>0.980</td>
<td>0.980</td>
</tr>
<tr>
<td>auto-mgp</td>
<td>100% - 800%</td>
<td>0.900</td>
<td>0.930</td>
<td>0.945</td>
<td>0.662</td>
<td>0.944</td>
<td>0.940</td>
<td>0.932</td>
<td>0.954</td>
<td>0.964</td>
</tr>
</tbody>
</table>

REFERENCES