CHAPTER 6

CLASS IMBALANCE LEARNING METHODS FOR SUPPORT VECTOR MACHINES

Rukshan Batuwita* and Vasile Palade†

* Singapore-MIT Alliance for Research and Technology Centre; † University of Oxford.

Abstract

Support Vector Machines is a very popular machine learning technique. Despite of all its theoretical and practical advantages, SVMs could produce suboptimal results with imbalanced datasets. That is, an SVM classifier trained on an imbalanced dataset can produce suboptimal models which are biased towards the majority class and have low performance on the minority class, like most of the other classification paradigms. There have been various data preprocessing and algorithmic techniques proposed in the literature to alleviate this problem for SVMs. This chapter aims to review these techniques.

6.1 INTRODUCTION

Support Vector Machines (SVMs) [1, 2, 3, 4, 5, 6, 7] is a popular machine learning technique, which has been successfully applied to many real-world classification problems from various domains. Due to its theoretical and practical advantages, such as solid mathematical background, high generalization

Imbalanced Learning: Foundations, Algorithms, and Applications,. By Haibo He and Yunqian Ma

capability and ability to find global and non-linear classification solutions, SVMs have been very popular among the machine learning and data mining researchers.

Although SVMs often work effectively with balanced datasets, they could produce suboptimal results with imbalanced datasets. More specifically, an SVM classifier trained on an imbalanced dataset often produces models which are biased towards the majority class and have low performance on the minority class. There have been various data preprocessing and algorithmic techniques proposed to overcome this problem for SVMs. This chapter is dedicated to discuss these techniques. In section 6.2 of this chapter we present some background on the SVM learning algorithm. In section 6.3, we discuss why SVMs are sensitive to the imbalance in datasets. Section 6.4 presents the existing techniques proposed in the literature to handle the class imbalance problem for SVMs. Finally, section 6.5 summarizes the chapter.

6.2 INTRODUCTION TO SUPPORT VECTOR MACHINES

In this section, we briefly review the learning algorithm of SVMs, which has been initially proposed in [1, 2, 3]. Let us consider that we have a binary classification problem represented by a dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$, where $x_i \in \Re^n$ represents an n-dimensional data point and $y_i \in \{-1, 1\}$ represents the label of the class of that data point, for $i = 1, \ldots, l$. The goal of the SVM learning algorithm is to find the optimal separating hyperplane which effectively separates these data points into two classes. In order to find a better separation of the classes, the data points are first considered to be transformed into a higher dimensional feature space by a non-linear mapping function Φ. A possible separating hyperplane residing in this transformed higher dimensional feature space can be represented by,

$$w \cdot \Phi(x) + b = 0 \tag{6.1}$$

where w is the weight vector normal to the hyperplane. If the dataset is completely linearly separable, the separating hyperplane with the maximum margin (for a higher generalization capability) can be found by solving the following maximal margin optimization problem:

$$\begin{aligned} & \min(\frac{1}{2} w \cdot w) \\ s.t. \quad & y_i(w \cdot \Phi(x_i) + b) \geq 1 \\ & i = 1, \ldots, l \end{aligned} \tag{6.2}$$

However, in most real-world problems, the datasets are not completely linearly separable even though they are mapped into a higher dimensional feature space. Therefore, the constrains in the above optimization problem in Eq.(6.2) are relaxed by introducing a set of slack variables, $\xi_i \geq 0$. Then the soft margin optimization problem can be reformulated as follows:

$$\begin{aligned} & \min(\frac{1}{2} w \cdot w + C \sum_{i=1}^{l} \xi_i) \\ s.t. \quad & y_i(w \cdot \Phi(x_i) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad i = 1, \ldots, l \end{aligned} \tag{6.3}$$

The slack variables $\xi_i > 0$ hold for misclassified examples, and therefore the penalty term $\sum_{i=1}^{l} \xi_i$ can be considered of as a measure of the amount of total misclassifications (training errors) of the model. This new objective function given in Eq.(6.3) has two goals. One is to maximize the margin and the other one is to minimize the number of misclassifications (the penalty term). The parameter C controls the trade-off between these two goals. This quadratic optimization problem can be easily solved by representing it as a Lagrangian optimization problem, which has the following dual form:

$$\begin{aligned} & \max_{\alpha_i} \{ \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \Phi(x_i) \cdot \Phi(x_j) \} \\ s.t. \quad & \sum_{i=1}^{l} y_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, l \end{aligned} \tag{6.4}$$

where α_i are Lagrange multipliers, which should satisfy the following Karush-Kuhn-Tucker (KKT) conditions:

$$\alpha_i(y_i(w \cdot \phi(x_i) + b) - 1 + \xi_i) = 0, \quad i = 1, \ldots, l \tag{6.5}$$

$$(C - \alpha_i)\xi_i = 0, \quad i = 1, \ldots, l \tag{6.6}$$

An important property of SVMs is that it is not necessary to know the mapping function $\phi(x)$ explicitly. By applying a kernel function, such that $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$, we would be able to transform the dual optimization problem given in Eq.(6.4) into Eq.(6.7)

$$\max_{\alpha_i}\{\sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i,x_j)\} \tag{6.7}$$
$$s.t. \quad \sum_{i=1}^{l} y_i\alpha_i = 0, \quad 0 \le \alpha_i \le C, \quad i = 1,\ldots,l$$

By solving Eq.(6.7) and finding the optimal values for α_i, w can be recovered as in Eq.(6.8)

$$w = \sum_{i=1}^{l}\alpha_i y_i \phi(x_i) \tag{6.8}$$

and b can be determined from the KKT conditions given in Eq.(6.5). The data points having non-zero α_i values are called support vectors. Finally, the SVM decision function can be given by:

$$f(x) = sign(w\cdot\Phi(x) + b) = sign(\sum_{i=1}^{l}\alpha_i y_i K(x_i,x) + b) \tag{6.9}$$

6.3 SVMS AND CLASS IMBALANCE

Although SVMs often produce effective solutions for balanced datasets, they are sensitive to the imbalance in the datasets and produce sub-optimal models. [8, 9, 10] have studied this problem closely and proposed several possible reasons as to why SVMs can be sensitive to class imbalance, which are discussed below.

6.3.1 Weakness of the soft margin optimization problem

It has been identified that the separating hyperplane of an SVM model developed with an imbalanced dataset can be skewed towards the minority class [8], and this skewness can degrade the performance of that model with respect to the minority class. This phenomenon can be explained as follows.

Recall the objective function of the SVM soft-margin optimization problem, which was given in Eq.(6.3) previously.

$$\begin{aligned} & \min(\frac{1}{2} w \cdot w + C \sum_{i=1}^{l} \xi_i) \\ s.t. \quad & y_i(w \cdot \Phi(x_i) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad i = 1, \ldots, l \end{aligned} \tag{6.10}$$

The first part of this objective function focuses on maximizing the margin, while the second part attempts to minimize the penalty term associated with the misclassifications, where the regularization parameter C can also be considered as the assigned misclassification cost. Since we consider the same misclassification cost for all the training examples (i.e., same vale of C for both positive and negative examples), in order to reduce the penalty term, the total number of misclassifications should be reduced. When the dataset is imbalanced, the density of majority class examples would be higher than the density of minority class examples even around the class boundary region, where the ideal hyperplane would pass through (throughout this chapter we consider the majority class as the negative class and the minority class as the positive class). This is also pointed out in [9], that the low presence of positive examples make them appear further from the ideal class boundary than the negative examples. As a consequence, in order to reduce the total number of misclassifications in SVM learning, the separating hyperplane can be shifted (or skewed) towards the minority class. This shift/skew can cause the generation of more false negative predictions, which lowers the model's performance on the minority positive class. When the class imbalance is extreme, the SVMs could produce models having largely skewed hyperplanes, which would even recognize all the examples as negatives [10].

6.3.2 The imbalanced support-vector ratio

[9] has experimentally identified that as the training data gets more imbalanced, the ratio between the positive and negative support vectors also becomes more imbalanced. They have hypothesized that as a result of this imbalance, the neighbourhood of a test instance close to the boundary is more likely to be dominated by negative support vectors, and hence the decision function is more likely to classify a boundary point as negative. However, [10] has argued against this idea by pointing out that due to the constraint $\sum_{i=1}^{l} y_i \alpha_i = 0$ (given in Eq.(6.4)), α_i of each positive support vector, which are less in numbers than the negative support vectors, must be larger in magnitude than the α_i values associated with the negative support vectors. These α_i act as weights in the final decision function (Eq.6.9), and hence larger α_i in the positive support vectors receive higher weights than the negative support vectors, which can reduce the effect of imbalance in support vectors up to some extent. [10] has further argued that this could be the reason why SVMs

do not perform too badly compared to other machine learning algorithms for moderately skewed datasets.

In the remaining sections of this chapter we review the methods found in the literature to handle the class imbalance problem for SVMs. These methods have been developed as both data preprocessing methods (called external methods) and algorithmic modifications to the SVM algorithm (called internal methods).

6.4 EXTERNAL IMBALANCE LEARNING METHODS FOR SVMS: DATA PREPROCESSING METHODS

6.4.1 Resampling methods

All the data preprocessing methods discussed in the other chapters of this book can be used to balance the datasets before training SVM models. These methods include random and focused under/oversampling methods and synthetic data generation methods like SMOTE [11]. Resampling methods have been successfully applied to train SVMs with imbalanced datasets in different domains [10, 11, 12, 13, 14, 15, 16].

Especially, [17] presents an efficient focused oversampling method for SVMs. In this method, first the separating hyperplane found by training an SVM model on the original imbalanced dataset is used to select the most informative examples for a given classification problem, which are the data points lying around the class boundary region. Then, only these selected examples are balanced by oversampling as opposed to blindly oversampling the complete dataset. This method reduces the SVM training time significantly while obtaining the comparable classification results to the original oversampling method.

Support cluster machines (SCMs) method presented in [18] can be viewed as another focused resampling method for SVMs. This method first partitions the negative examples into disjoint clusters by using the kernel-k-means clustering method. Then it trains an initial SVM model using the positive examples and the representatives of the negative clusters, namely, the data examples representing the cluster centres. With the global picture of the initial SVMs, it approximately identifies the support vectors and non-support vectors. Then a shrinking technique is used to remove the samples which are most probably not support vectors. This procedure of clustering and shrinking is performed iteratively several times until convergence.

6.4.2 Ensemble learning methods

Ensemble learning has also been applied as a solution for training SVMs with imbalanced datasets [19, 20, 21, 22]. Generally, in these methods, the majority class dataset is separated into multiple subdatasets such that each of these sub-datasets has a similar number of examples as the minority class

dataset. This can be done by random sampling with or without replacement (bootstrapping), or through clustering methods. Then a set of SVM classifiers is developed so that each one is trained with the same positive dataset and a different negative sub-dataset. Finally, the decisions made by the classifier ensemble are combined by using a method such as majority voting. In addition, special boosting algorithms, such as Adacost [23], RareBoost [24] and SMOTEBoost [25], which have been used in class imbalance learning with ensemble settings, could also be applied with SVMs.

6.5 INTERNAL IMBALANCE LEARNING METHODS FOR SVMS: ALGORITHMIC METHODS

In this section we present the algorithmic modifications proposed in the literature to make the SVM algorithm less sensitive to class imbalance.

6.5.1 Different Error Costs (DEC)

As we pointed out in section 6.3 above, the main reason for the SVM algorithm to be sensitive to class imbalance would be that the soft margin objective function given in Eq.(6.10) assigns the same cost (i.e., C) for both positive and negative misclassifications in the penalty term. This would cause the separating hyperplane to be skewed towards the minority class, which would finally yield a suboptimal model. The DEC method is a cost-sensitive learning solution proposed in [8] to overcome this problem in SVMs. In this method, the SVM soft margin objective function is modified to assign two misclassification costs, such that C^+ is the misclassification cost for positive class examples, while C^- is the misclassification cost for negative class examples, as given in Eq.(6.11) below. Here we also assume positive class to be the minority class and negative class to be the majority class.

$$\begin{aligned} \min(\frac{1}{2} w \cdot w + C^+ \sum_{i|y_i=+1}^{l} \xi_i + C^- \sum_{i|y_i=-1}^{l} \xi_i) \\ s.t. \quad y_i(w \cdot \Phi(x_i) + b) \geq 1 - \xi_i \\ \xi_i \geq 0, \quad i = 1, \ldots, l \end{aligned} \tag{6.11}$$

By assigning a higher misclassification cost for the minority class examples than the majority class examples (i.e.,$C^+ > C^-$), the effect of class imbalance could be reduced. That is, the modified SVM algorithm would not tend to skew the separating hyperplane towards the minority class examples to reduce the total misclassifications as the minority class examples are now assigned with a higher misclassification cost. The dual Lagrangian form of this modified objective function can be represented as follows:

$$\max_{\alpha_i}\{\sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i,x_j)\} \quad (6.12)$$

$$s.t. \quad \sum_{i=1}^{l} y_i\alpha_i = 0, \quad 0 \le \alpha_i^+ \le C^+, \quad 0 \le \alpha_i^- \le C^-, i = 1,\ldots,l$$

where α_i^+ and α_i^- represent the Lagrangian multipliers of positive and negative examples, respectively. This dual optimization problem can be solved in the same way as solving the normal SVM optimization problem. As a rule of thumb, [10] has reported that the reasonably good classification results from the DEC method could be obtained by setting the C^-/C^+ equals to the minority to majority class ratio.

6.5.2 One class learning

[26, 27] have presented two extreme rebalancing methods for training SVMs with highly imbalanced datasets. In the first method they have trained an SVM model only with the minority class examples. In the second method, the DEC method has been extended to assign a $C^- = 0$ misclassification cost for the majority class examples and $C^+ = 1/N^+$ misclassification cost for minority class examples, where N^+ is the number of minority class examples. From the experimental results obtained on several heavily imbalanced synthetic and real-world datasets, these methods have been observed to be more effective than general data rebalancing methods.

6.5.3 zSVM

zSVM is another algorithmic modification proposed for SVMs in [28] to learn from imbalanced datasets. In this method, first an SVM model is developed by using the original imbalanced training dataset. Then, the decision boundary of the resulted model is modified to remove its bias towards the majority (negative) class. Consider the standard SVM decision function given in Eq.(6.9), which can be rewritten as follows:

$$\begin{aligned} f(x) &= sign(\sum_{i=1}^{l}\alpha_i y_i K(x_i,x) + b) \\ &= sign(\sum_{i=1}^{l_1}\alpha_i^+ y_i K(x_i,x) + \sum_{j=1}^{l_2}\alpha_j^- y_j K(x_j,x) + b) \end{aligned} \quad (6.13)$$

where α_i^+ are the coefficients of the positive support vectors, α_j^- are the coefficients of the negative support vectors, and l_1 and l_2 represent the number of positive and negative training examples, respectively. In the zSVM method, the magnitude of the α_i^+ values of the positive support vectors are increased by multiplying all of them by a particular small positive value z. Then, the modified SVM decision function can be represented as follows:

$$f(x) = sign(z * \sum_{i=1}^{l_1} \alpha_i^+ y_i K(x_i, x) + \sum_{j=1}^{l_2} \alpha_i^- y_i K(x_j, x) + b) \tag{6.14}$$

This modification of α_i^+ would increase the weights of the positive support vectors in the decision function, and therefore it would decrease its bias towards the majority negative class. In [28], the value of z giving the best classification results for the training dataset was selected as the optimal value.

6.5.4 Kernel modification methods

There have been several techniques proposed in the literature to make the SVM algorithm less sensitive to the class imbalance by modifying the associated kernel function.

6.5.4.1 Class boundary alignment [9] has proposed a variant of SVM learning method, where the kernel function is conformally transformed to enlarge the margin around the class boundary region in the transformed higher dimensional feature space to have improved performance. [29] has improved this method for imbalanced datasets by enlarging more of the class boundary around the minority class compared to the class boundary around the majority class. This method is called the class boundary alignment (CBA) method which can only be used with the vector space representation of input data. [30] has further proposed a variant of the CBA method for the sequence representation of imbalanced input data by modifying the kernel matrix to have the similar effect, which is called the Kernel Boundary Alignment (KBA) method.

6.5.4.2 Kernel target alignment In the context of SVM learning, a quantitative measure of agreement between the kernel function used and the learning task is important from the both theoretical and practical point of view. Kernel target alignment method has been proposed as a method for measuring the agreement between a kernel being used and the classification task in [31]. This method has been improved for imbalanced datasets learning in [32].

6.5.4.3 Margin calibration The DEC method described previously modifies the SVM objective function by assigning a higher misclassification cost to the

positive examples than the negative examples to change the penalty term. [33] has extended this method to modify the SVM objective function not only in terms of the penalty term, but also in terms of the margin to recover the biased decision boundary. As proposed in this method, the modification first adopts an inversed proportional regularized penalty to reweight the imbalanced classes. Then it employs a margin compensation to lead the margin to be lopsided, which enables the decision boundary drift.

6.5.4.4 Other kernel-modification methods There have been several imbalance learning techniques proposed in the literature for other kernel-based classifiers. These methods include the kernel classifier construction algorithm proposed in [34] based on orthogonal forward selection (OFS) and the regularized orthogonal weighted least squares (ROWLSs) estimator, kernel neural gas (KNG) algorithm for imbalanced clustering [35], the P2PKNNC algorithm based on the k-nearest neighbors classifier and the P2P communication paradigm [36], Adaboost relevance vector machine (RVM) [37], among others.

6.5.5 Active learning

Active learning methods, as opposed to conventional batch learning, have also been applied to solve the problem of class imbalance for SVMs. [38] and [39] have proposed an efficient active learning strategy for SVMs to overcome the class imbalance problem. This method iteratively selects the closest instance to the separating hyperplane from the unseen training data and adds it to the training set to retrain the classifier. With an early stopping criterion, the method can significantly decrease the training time in large scale imbalanced datasets.

6.5.6 Fuzzy SVMs for class imbalance learning (FSVM-CIL)

All the methods presented so far attempt to make SVMs robust to the problem of class imbalance. It has been well studies in the literature that SVMs are also sensitive to the noise and outliers present in datasets. Therefore, it can be argued that although the existing class imbalance learning methods can make the SVM algorithm less sensitive to the class imbalance problem, it can still be sensitive to noise and outliers present in datasets, which could still result in suboptimal models. In fact, some class imbalance learning methods, such as random oversampling and SMOTE, can make the problem worse by duplicating the existing outliers and noisy examples or introducing new ones. Fuzzy SVMs for Class Imbalance Learning (FSVM-CIL) is an improved SVM method proposed in [40] to handle the problem of class imbalance together with the problem of outliers and noise. In this section, we present this method with more details.

6.5.6.1 The Fuzzy SVM method As mentioned previously, the standard SVM algorithm considers all the data points with equal importance and assigns the

same misclassification cost for those in its objective function. We have already pointed out that this can cause SVM to produce sub optimal models on imbalanced datasets. It has also been found out that the same reason of considering all the data points with equal importance can also cause SVMs to be sensitive to the outliers and noise present in a dataset. That is, the presence of outliers and noisy examples (especially, around the class boundary region) can influence the position and orientation of the separating hyperplane causing the development of suboptimal models.

In order to make the SVMs less sensitive to outliers and noisy examples, a technique called Fuzzy SVMs (FSVMs) have been proposed in [41]. The FSVM method assigns different fuzzy membership values, $m_i; m_i \geq 0$ (or weights), for different examples to reflect different importance in their own classes, where more important examples are assigned higher membership values, while less important ones (such as outliers and noise) are assigned lower membership values. Then, the SVM soft margin optimization problem is reformulated as follows:

$$\begin{aligned} &\min(\frac{1}{2}w \cdot w + C\sum_{i=1}^{l} m_i\xi_i) \\ s.t. \quad &y_i(w \cdot \Phi(x_i) + b) \geq 1 - \xi_i \\ &\xi_i \geq 0, \quad i = 1, \ldots, l \end{aligned} \tag{6.15}$$

In this reformulation of the objective function, the membership m_i of a data point x_i is incorporated into the penalty term, such that a smaller m_i could reduce the effect of the associated slack variable ξ_i in the objective function (if the corresponding data point x_i is treated as less important). In another view, if we consider C as the cost assigned for a misclassification, now each data point is assigned with a different misclassification cost, m_iC, which is based on the importance of the data point in its own class, such that more important data points are assigned higher costs, while less important ones are assigned lower costs. Therefore, the FSVM algorithm can find a more robust separating hyperplane through maximizing the margin by allowing some misclassification for less important examples, like the outliers and noise.

In order to solve the FSVM optimization problem, Eq.(6.15) can be transformed into the following dual Lagrangian form:

$$\max_{\alpha_i}\{\sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i, x_j)\} \tag{6.16}$$

$$s.t. \quad \sum_{i=1}^{l} y_i\alpha_i = 0, \quad 0 \leq \alpha_i \leq m_iC, \quad i = 1, \ldots, l$$

The only difference between the original SVM dual optimization problem given in Eq.(6.7) and the FSVM dual optimization problem given in Eq.(6.16) is the upper bound of the values that α_i could take. By solving this dual problem in Eq.(6.16) for optimal α_i, w and b can be recovered in the same way as in the normal SVM learning algorithm. The same SVM decision function in Eq.(6.9) applies for FSVMs method as well.

6.5.6.2 FSVM-CIL method However, the standard FSVM method is still sensitive to the class imbalance problem, since the assigned misclassification costs do not consider the imbalance of the dataset. [40] has improved the standard FSVM method by combining it with the DEC method, which is called the FSVM-CIL. In the FSVM-CIL method, the membership values for data points are assigned in such a way to satisfy the following two goals:

1. To suppress the effect of between class imbalance.
2. To reflect the within class importance of different training examples in order to suppress the effect of outliers and noise.

Let m_i^+ represents the membership value of a positive data point x_i^+, while m_i^- represents the membership of a negative data point x_i^- in their own classes. In the proposed FSVM-CIL method, these membership functions are defined as follows:

$$m_i^+ = f(x_i^+)r^+ \tag{6.17}$$
$$m_i^- = f(x_i^-)r^- \tag{6.18}$$

where $f(x_i)$ generates a value between 0 and 1, which reflects the importance of x_i in its own class. The values for r^+ and r^- were assigned in order to reflect the class imbalance, such that $r^+ = 1$ and $r^- = r$, where r is the minority to majority class ratio ($r^+ > r^-$) (this was following the findings reported in [10], where the optimal results from the DEC method could be obtained when C^-/C^+ equals to the minority to majority class ratio). According to this assignment of values, a positive class data point is assigned a misclassification cost m_i^+C, where m_i^+ takes a value in the [0,1] interval, while a negative class data point is assigned a misclassification cost m_i^-C, where m_i^- takes value in the $[0, r]$ interval, where $r < 1$.

In order to define the function $f(x_i)$ introduced in Eq.(6.17) and (6.18), which gives the within class importance of a training example, the following methods have been considered in [40].

A. $f(x_i)$ is based on the distance from the own class centre:

In this method, $f(x_i)$ is defined with respect to d_i^{cen}, which is the distance between x_i and its own class centre. The examples closer to the class centre

are treated as more informative and assigned higher $f(x_i)$ values, while the examples far away from the centre are treated as outliers or noise and assigned lower $f(x_i)$ values. Here, two separate decaying functions of d_i^{cen} have been used to define $f(x_i)$, which are represented by $f_{lin}^{cen}(x_i)$ and $f_{exp}^{cen}(x_i)$ as follows:

$$f_{lin}^{cen}(x_i) = 1 - (d_i^{cen}/(\max(d_i^{cen}) + \delta)) \tag{6.19}$$

is a linearly decaying function. δ is a small positive value used to avoid the case where $f(x_i)$ becomes zero.

$$f_{exp}^{cen}(x_i) = 2/(1 + \exp(d_i^{cen} * \beta)) \tag{6.20}$$

is an exponentially decaying function, where $\beta; \beta \in [0, 1]$ determines the steepness of the decay. $d_i^{cen} = \|x_i - \bar{x}\|^{\frac{1}{2}}$ is the Euclidean distance to x_i from its own class centre $\bar{x}$.

B. $f(x_i)$ is based on the distance from the preestimated separating hyperplane:

In this method, $f(x_i)$ is defined based on d_i^{sph}, which is the distance to x_i from the preestimated separating hyperplane as introduced in [42]. Here d_i^{sph} is estimated by the distance to x_i from the centre of the common spherical region, which can be defined as a hyper-sphere covering the overlapping region of the two classes, where the separation hyperplane is more likely to pass through. Both linear and exponential decaying functions are used to define the function $f(x_i)$, which are represented by $f_{lin}^{sph}(x_i)$ and $f_{exp}^{sph}(x_i)$ as follows:

$$f_{lin}^{sph}(x_i) = 1 - (d_i^{sph}/(\max(d_i^{sph}) + \delta)) \tag{6.21}$$

$$f_{exp}^{sph}(x_i) = 2/(1 + \exp(d_i^{sph} * \beta)) \tag{6.22}$$

where $d_i^{sph} = \|x_i - \bar{x}\|^{\frac{1}{2}}$ and $\bar{x}$ is the centre of the spherical region, which is estimated by the centre of the entire dataset, δ is a small positive value and $\beta \in [0, 1]$.

C. $f(x_i)$ is based on the distance from the actual separating hyperplane:

In this method, $f(x_i)$ is defined based on the distance from the actual separating hyperplane to x_i, which is found by training a conventional SVM model on the imbalanced dataset. The data points closer to the actual separating hyperplane are treated as more informative and assigned higher membership values, while the data points far away from the separating hyperplane are treated as less informative and assigned lower membership values. The following procedure is carried out to assign $f(x_i)$ values in this method:

1. Train a normal SVM model with the original imbalanced dataset

2. Find the functional margin d_i^{hyp} of each example x_i (given in Eq.(6.23)) (this is equivalent to the absolute value of the SVM decision value) with respect to the separating hyperplane found. The functional margin is proportional to the geometric margin of a training example with respect to the separating hyperplane.

$$d_i^{hyp} = y_i(w \cdot \Phi(x_i) + b) \tag{6.23}$$

3. Consider both linear and exponential decaying functions to define $f(x_i)$ as follows:

$$f_{lin}^{hyp}(x_i) = 1 - (d_i^{hyp}/(\max(d_i^{hyp}) + \delta)) \tag{6.24}$$

$$f_{exp}^{hyp}(x_i) = 2/(1 + \exp(d_i^{hyp} * \beta)) \tag{6.25}$$

where δ is a small positive value and $\beta \in [0, 1]$.

Following the aforementioned methods of assigning membership values for positive and negative training data points, several FSVM-CIL settings have been defined in [40]. These methods have been validated on 10 real-world imbalanced datasets representing a variety of domains, complexities and imbalanced ratios, which are highly likely to contain noisy examples and outliers. FSVM-CIL settings have resulted in better classification results on these datasets than the existing class imbalance learning methods applied for standard SVMs, namely, random oversampling, random undersampling, SMOTE, DEC and zSVM methods. [40] pointed out that better performance of FSVM-CIL method is due to its capability to handle outliers and noise in these datasets in addition to the class imbalance problem.

6.5.7 Hybrid Methods

There exist methods which have used the combination of both external and internal methods to solve the class imbalance problem for SVMs. The hybrid kernel machine ensemble (HKME) method [43] combines a standard binary SVM and a one-class SVM classifier to solve the problem of class imbalance. [10] has combined the SMOTE algorithm with the DEC method for SVMs for imbalanced dataset learning and shown to have better performance than the use of either of these methods alone.

6.6 SUMMARY

This chapter aimed to review the existing imbalance learning methods developed for SVMs. These methods have been developed as data pre-processing methods or algorithmic improvements. As it has been pointed out in the literature, the class imbalance learning method giving the optimal solution is often dataset dependent. Therefore, it is worth applying several of these available external and internal methods and compare the performances, when training an SVM model on an imbalanced dataset.

REFERENCES

1. V. Vapnik, *The nature of statistical learning theory.* Springer-Verlag New York, Inc., 1995.
2. C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
3. B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pp. 144–152, ACM Press, 1992.
4. N. Cristianinio and J. Shawe-Taylor, *An introduction to support Vector Machines: and other kernel-based learning methods.* Cambridge University Press, 2000.
5. B. Scholkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* Cambridge, MA, USA: MIT Press, 2001.
6. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.
7. C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.
8. K. Veropoulos, C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," in *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 55–60, 1999.

9. G. Wu and E. Chang, "Adaptive feature-space conformal transformation for imbalanced-data learning," in *Proceedings of the 20th International Conference on Machine Learning*, pp. 816–823, 2003.
10. R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proceedings of the 15th European Conference on Machine Learning*, pp. 39–50, 2004.
11. N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
12. J. Chen, M. Casique, and M. Karakoy, "Classification of lung data by sampling and support vector machine," in *Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2, pp. 3194–3197, 2004.
13. Y. Fu, S. Ruixiang, Q. Yang, H. Simin, C. Wang, H. Wang, S. Shan, J. Liu, , and W. Gao, "A block-based support vector machine approach to the protein homology prediction task in kdd cup 2004," *SIGKDD Exploration Newsletters*, vol. 6, pp. 120–124, Dec. 2004.
14. S. Lessmann, "Solving imbalanced classification problems with support vector machines," in *Proceedings of the International Confenece on Artificial Intelligence*, pp. 214–220, 2004.
15. R. Batuwita and V. Palade, "An improved non-comparative classification method for human microrna gene prediction," in *Proceedings of the International Conference on Bioinformatics and Bioengineering*, pp. 1–6, 2008.
16. R. Batuwita and V. Palade, "micropred: Effective classification of pre-mirnas for human mirna gene prediction," *Bioinformatics*, vol. 25, pp. 989–995, February 2009.
17. R. Batuwita and V. Palade, "Efficient resampling methods for training support vector machines with imbalanced datasets," in *Proceedings of the International Joint Conference on Neural Networks*, pp. 1–8, 2010.
18. J. Yuan, J. Li, , and B. Zhang, "Learning concepts from large scale imbalanced data sets using support cluster machines," in *Proceedings of the 14th annual ACM international conference on Multimedia*, pp. 441–450, ACM, 2006.
19. Z. Lin, Z. Hao, X. Yang, and X. Liu, "Several svm ensemble methods integrated with under-sampling for imbalanced data learning," in *Proceedings of the 5th International Conference on Advanced Data Mining and Applications*, pp. 536–544, Springer-Verlag, 2009.
20. P. Kang and S. Cho, "Eus svms: ensemble of under-sampled svms for data imbalance problems," in *Proceedings of the 13th international conference on Neural Information Processing*, pp. 837–846, Springer-Verlag, 2006.
21. Y. Liu, A. An, and X. Huang, "Boosting prediction accuracy on imbalanced datasets with svm ensembles," in *Proceedings of the 10th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*, pp. 107–118, 2006.
22. B. Wang and N. Japkowicz, "Boosting support vector machines for imbalanced data sets," *Knowledge and Information Systems*, vol. 25, pp. 1–20, Oct. 2010.

23. W. Fan, S. Stolfo, J. Zhang, and P. Chan, "Adacost: Misclassification cost-sensitive boosting," in *Proceedings of the 16th International Conference on Machine Learning*, pp. 97–105, Morgan Kaufmann Publishers Inc., 1999.

24. M. Joshi, V. Kumar, and C. Agarwal, "Evaluating boosting algorithms to classify rare classes: Comparison and improvements," in *Proceedings of the IEEE International Conference on Data Mining*, pp. 257–264, IEEE Computer Society, 2001.

25. N. Chawla, A. Lazarevic, L. Hall, and K. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *Proceedings of the Principles of Knowledge Discovery in Databases*, pp. 107–119, 2003.

26. B. Raskutti and A. Kowalczyk, "Extreme re-balancing for svms: a case study," *SIGKDD Exploration Newsletters*, vol. 6, pp. 60–69, June 2004.

27. A. Kowalczyk and B. Raskutti, "One class svm for yeast regulation prediction," *SIGKDD Exploration Newsletters*, vol. 4, no. 2, pp. 99–100, 2002.

28. T. Imam, K. Ting, and J. Kamruzzaman, "z-svm: an svm for improved classification of imbalanced data," in *Proceedings of the 19th Australian joint conference on Artificial Intelligence: advances in Artificial Intelligence*, pp. 264–273, Springer-Verlag, 2006.

29. G. Wu and E. Chang, "Class-boundary alignment for imbalanced dataset learning," in *Proceeding of the International Conference on Machine Learning: Workshop on Learning from Imbalanced Data Sets*, pp. 49–56, 2003.

30. G. Wu and E. Chang, "Kba: Kernel boundary alignment considering imbalanced data distribution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 786–795, 2005.

31. N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, "On kernel-target alignment," in *Advances in Neural Information Processing Systems 14*, pp. 367–373, MIT Press, 2002.

32. J. Kandola and J. Shawe-taylor, "Refining kernels for regression and uneven classification problems," in *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2003.

33. C.-Y. Yang, J.-S. Yang, and J.-J. Wang, "Margin calibration in svm class-imbalanced learning," *Neurocomputing*, vol. 73, no. 1-3, pp. 397–411, 2009.

34. X. Hong, S. Chen, and C. Harris, "A kernel-based two-class classifier for imbalanced data sets," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 28–41, 2007.

35. A. Qin and P. Suganthan, "Kernel neural gas algorithms with application to cluster analysis," in *Proceedings of the 17th International Conference on Pattern Recognition*, pp. 617–620, IEEE Computer Society, 2004.

36. X.-P. Yu and X.-G. Yu, "Novel text classification based on k-nearest neighbor," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, pp. 3425–3430, 2007.

37. A. Tashk and K. Faez, "Boosted bayesian kernel classifier method for face detection," in *Proceedings of the Third International Conference on Natural Computation*, pp. 533–537, IEEE Computer Society, 2007.

38. S. Ertekin, J. Huang, and L. Giles, "Active learning for class imbalance problem," in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 823–824, ACM, 2007.

39. S. Ertekin, J. Huang, L. Bottou, and L. Giles, "Learning on the border: active learning in imbalanced data classification," in *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, pp. 127–136, ACM, 2007.

40. R. Batuwita and V. Palade, "Fsvm-cil: fuzzy support vector machines for class imbalance learning," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 3, pp. 558–571, 2010.

41. C.-F. Lin and S.-D. Wang, "Fuzzy support vector machines," *IEEE Transactions on In Neural Networks*, vol. 13, no. 2, pp. 464–471, 2002.

42. C.-F. Lin and S.-D. Wang, "Training algorithms for fuzzy support vector machines with noisy data," *Pattern Recognition Letters*, vol. 25, no. 14, pp. 1647–1656, 2004.

43. P. Li, K. Chan, and W. Fang, "Hybrid kernel machine ensemble for imbalanced data sets," in *Proceedings of the 18th International Conference on Pattern Recognition*, pp. 1108–1111, IEEE Computer Society, 2006.