CHAPTER 2

Preliminaries

In this chapter, a gentle introduction to class imbalanced problems is presented. This chapter is structured as follows. First, in Section 2.1 we give an example of a binary class imbalance problem and introduce the existing approaches and the performance metrics in the binary class imbalance domain. Next, in Section 2.2 the methods and performance metrics in multi-class scenarios are presented. Then, in Section 2.3 we address the importance of data complexity in the imbalanced datasets and present the data complexity measures. Finally, in Section 2.4 the benchmark datasets for learning from imbalanced problems and several imbalanced applications are discussed.

2.1 Binary Class Imbalance Learning

Most studies in the imbalanced learning domain are devoted to the binary scenario, where the number of samples in one class is significantly higher than in the other. An example of a binary class imbalance problem is shown in Figure 2.1, where the Imbalance Ratio (IR) is the ratio of the number of majority class samples to the number of minority class samples (Orriols-Puig and Bernadó-Mansilla, 2009). The figure clearly illustrates that the minority class is underrepresented due to the lack of samples, and in real-world applications, the minority class is usually the class of interest. For instance, if we consider Figure 2.1 as an example from the car industry, we need to perform quality control, i.e. differentiate the qualified and unqualified cars. In this case, it is much more critical to identify unqualified cars correctly. The consequence of undetected unqualified cars could be severe accidents, whereas a false classification of qualified cars only requires a double check. The ideal case is to get a 100% accuracy on both classes. However, the current classification techniques are not perfect, and in order to ensure the overall accuracy, they tend to bias toward the majority class and produce poor accuracy or even neglect the accuracy of the minority class (0% accuracy). Class imbalance is not the only reason leading to performance degradation. Data complexity also significantly influences the imbalanced classification; detailed information on this will be given in Section 2.3. This section reviews the existing approaches and performance metrics for binary imbalanced learning.

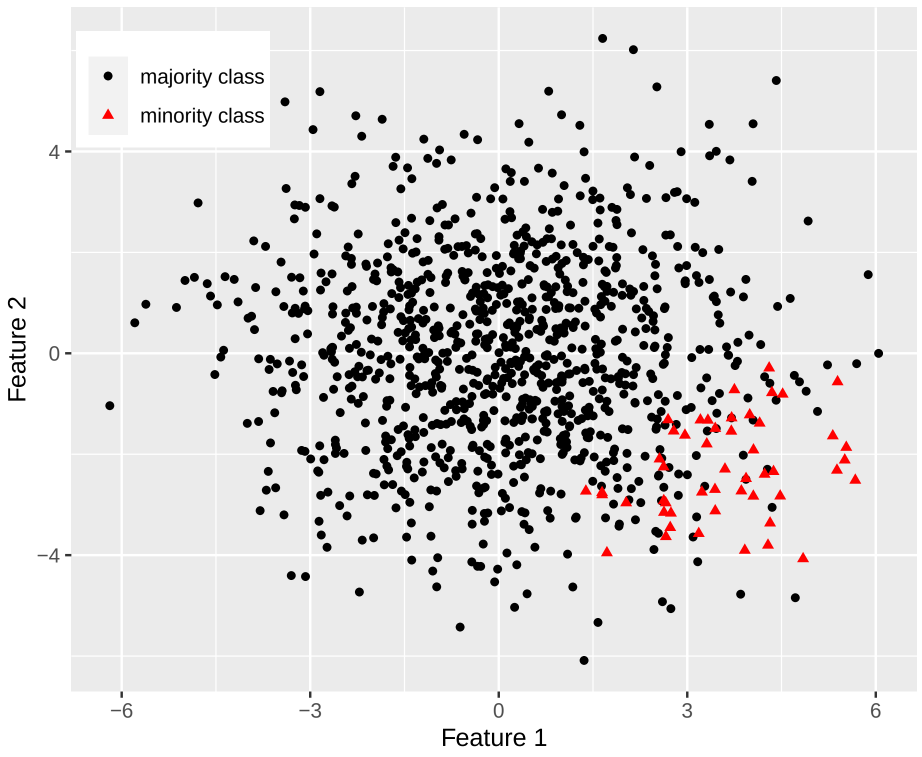


Figure 2.1: An example of a binary class imbalance problem with IR = 200.

2.1.1 Existing Approaches

Many techniques have been developed to improve the minority class accuracy in class imbalance problems. These techniques can be grouped into four broad categories based on how they deal with the problem.

Data-level approaches

Data-level approaches, also known as **resampling** techniques, adjust the data space directly in order to produce relatively balanced data distribution for standard classifiers. Resampling techniques consist of three groups, oversampling, undersampling and hybrid methods. For a clear description, the following notations are used in this section. For a training dataset S with N samples, i.e. $|S| = N$ and $S = \{(\boldsymbol{x_n}, y_n)\}, n = 1, 2, ..., N$, where $\boldsymbol{x_n}$ belongs to an instance space X and y_i belongs to a label set associated with $\boldsymbol{x_n}$.

Oversampling balances the class distribution by replicating existing samples in the minority class or generating new artificial samples for the minority class. One of the most representative oversampling approaches is the Synthetic Minority Oversampling TEchnique (SMOTE). SMOTE works by creating artificial minority class samples to produce balanced data. The artificial samples are generated based on the randomly chosen minority class samples and their K-Nearest Neighbours. A new synthetic sample $\boldsymbol{x_s}$ can be generated according to the following equation (H. He and E. A. Garcia, 2009):

$$\boldsymbol{x_s} = \boldsymbol{x_i} + \delta \cdot (\boldsymbol{\hat{x}_i} - \boldsymbol{x_i}); \tag{2.1}$$

where $\boldsymbol{x_i}$ is the minority class sample to oversample, $\boldsymbol{\hat{x}_i}$ is a randomly selected neighbour from its K-nearest minority class neighbours and δ is a random number, where $\delta \in [0, 1]$, as described in (Chawla, Bowyer, Hall, and Kegelmeyer, 2002). Figure 2.2 illustrates how the synthetic samples are created in the SMOTE technique.

Undersampling eliminates the samples in the majority class to equalize the number of samples in each class. The majority class samples can be removed randomly or according to the preset strategies. Hybrid methods are the hybridization of oversampling and undersampling. There are various ways to perform these three groups of techniques (oversampling, undersampling and hybrid methods). Figure 2.3 shows examples of two resampling techniques, Synthetic Minority Oversampling TEchnique (SMOTE) and Random Undersampling (RUS), where RUS adjusts the data distribution by randomly deleting samples from the majority class. Detailed descriptions of various resampling techniques will be given in the following chapters.

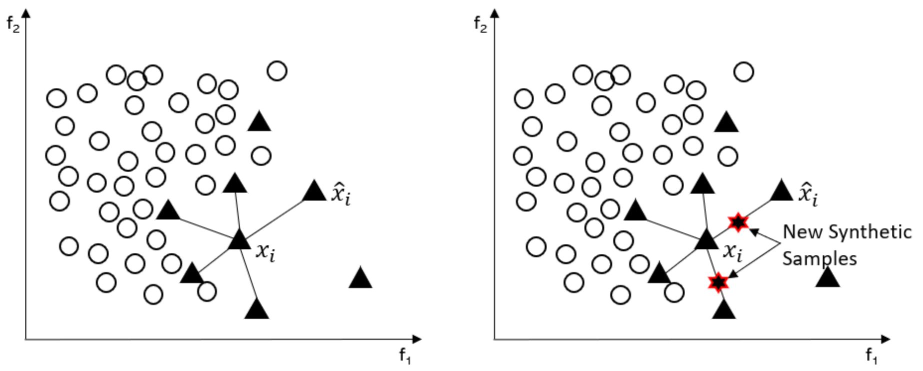


Figure 2.2: An illustration of how to generate synthetic samples through SMOTE. Example of K-nearest minority class neighbours for minority class sample $\boldsymbol{x_i}$ (K=5) (left) and new synthetic samples generated through SMOTE (right).

Algorithm-level approaches

Algorithm-level approaches do not deal with the data distribution. Instead, they modify the classical classification algorithms to alleviate the bias towards the majority class caused by the significant imbalanced data distribution. An in-depth understanding of the classification algorithms is required to perform appropriate modifications since one needs to precisely identify which part in the algorithm hinders the classification performance on imbalanced datasets (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018). An example of modifying Support Vector Machines (SVMs) is to emphasise more weight on support vectors belonging to minority class so that the decision boundary shift towards minority class (Imam, Ting, and Kamruzzaman, 2006). Another example of adapting Decision Trees is to use Hellinger distance as the split function instead of Gini index (Cieslak, Hoens, Chawla, and Kegelmeyer, 2012). The main idea is to avoid the selecting criteria in favour of the majority class. An exhaustive review of the algorithm-level approaches on class imbalance problems can be found in (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018).

Cost-sensitive learning

Most standard machine learning classification algorithms assume symmetric misclassification costs for each class (Thai-Nghe, Gantner, and Schmidt-Thieme, 2010). However, this assumption is violated in class imbalance problems since the cost of misclassifying samples in the minority class is much higher than that in the majority class. Cost-sensitive methods handle class imbalance problems via considering the costs associated with misclassifying samples (Elkan, 2001; H. He and E. A. Garcia, 2009). This learning framework can be combined with data-level approaches by adding costs to specific samples and can also be combined with algorithm-level approaches by adapting the misclassification cost in the learning process (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018).

Ensemble learning

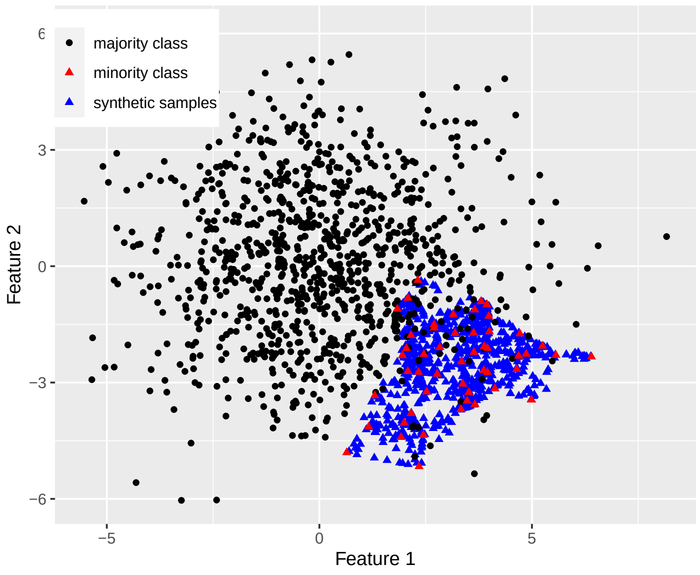
Ensemble-based classifiers, a combination of multiple classification algorithms, are known to produce better classification performance compared to a single classification algorithm (Rokach, 2010). Standard ensemble-based classifiers are not very effective to deal with skewed class distributions; however, they can be easily adapted to handle class imbalance problems. In the imbalanced learning domain, the most straightforward approach for adapting the ensemble-based classifiers is to include a resampling technique as a preprocessing step before learning base classifiers (Błaszczyński, Deckert, Stefanowski, and Wilk, 2010), e.g. SMOTEBoost (Chawla, Lazarevic, Hall, and Bowyer, 2003) and SMOTEBagging (S. Wang and Yao, 2009). Ensemble-based classifiers can also be combined with cost sensitive learning mainly in two ways in the literature, cost-sensitive Boosting (Sun, Kamel, A. K. Wong, and Y. Wang, 2007) and ensembles with cost-sensitive base classifiers (B. X. Wang and Japkowicz, 2010).

2.1.2 Performance Metrics

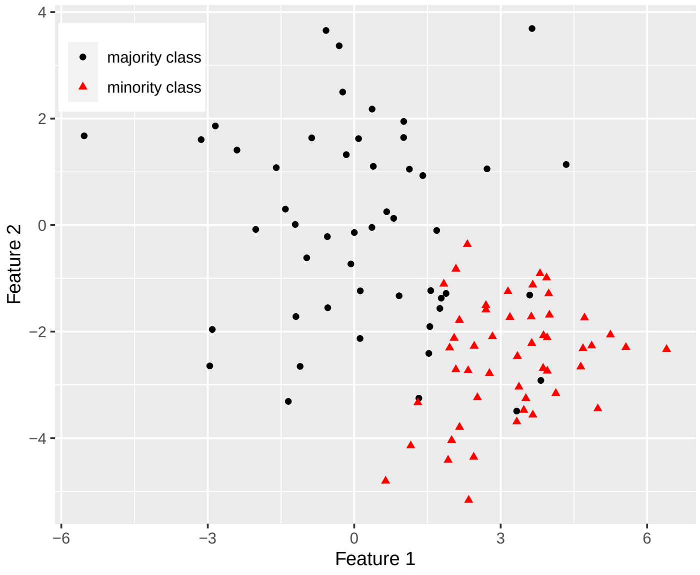
When dealing with classification tasks, *accuracy* and *error rate* are the most frequently used performance metrics (H. He and E. A. Garcia, 2009). In a binary classification problem, the confusion matrix (see Table 2.1) can provide classification results.

According to the confusion matrix (see Table 2.1), *accuracy* and *error rate* can be computed as

$$\begin{aligned} \textit{accuracy} &= \frac{TP+TN}{TP+FN+FP+TN}, \\ \textit{error rate} &= 1 - \textit{accuracy}. \end{aligned} \tag{2.2}$$



(a) An example of oversampling technique SMOTE.



(b) An example of undersampling technique RUS.

Figure 2.3: Examples of two resampling techniques with (a) SMOTE and (b) RUS.

Table 2.1: Confusion matrix for a binary classification problem

	Positive prediction	**Negative prediction**
Positive class	True Positives (TP)	False Negatives (FN)
Negative class	False Positives (FP)	True Negatives (TN)

However, the two metrics have some drawbacks when dealing with imbalanced datasets. Firstly, they may give a deceptive evaluation in imbalanced scenarios. For example, let us assume in a binary class-imbalance classification problem, the majority-class and minority-class samples take 95% and 5% of the total samples respectively. Even if the classifier predicts all the samples as majority class, the accuracy is still 95%, which makes the classifier seems extremely efficient but neglects the minority class. Moreover, the two metrics above assume the cost of misclassifying different class samples is the same. However, in imbalanced classification, the cost of misclassifying minority class samples are generally higher. In bank transactions, for instance, failing to detect a fraud case will result in a massive loss of money, while classifying a safe transaction into a fraud will require a double check. Considering the facts above, the accuracy does not reflect the actual effectiveness of an algorithm in imbalanced domains.

In lieu of accuracy, *recall*, *precision*, *F-Measure (FM)* and *G-Mean (GM)* are frequently adopted to assess the classification performance in imbalanced scenarios. These measures are computed by

$$\begin{aligned} \textit{Precision} &= \frac{TP}{TP+FP}, \\ \textit{Recall} &= \frac{TP}{TP+FN}, \\ FM &= \frac{(1+\beta)^2 \times \textit{Recall} \times \textit{Precision}}{\beta^2 \times \textit{Precision} + \textit{Recall}}, \\ GM &= \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{FP+TN}}, \end{aligned} \tag{2.3}$$

where β is a coefficient which controls the relative importance of *precision* and *recall*. It is a positive real coefficient indicating the importance of *recall* is β times as *precision*. β is normally set to 1, indicating the same importance of *precision* and *recall*.

In the literature, *precision* and *recall* are also referred as positive predictive value and true positive rate, reflecting the exactness and completeness respectively (H. He and E. A. Garcia, 2009). *Precision* measures the proportion of correctly classified positive samples to all positive predictions, whereas *recall* measures the proportion of correctly classified positive samples to all positive samples. *F-Measure* achieves the trade-off between *precision* and *recall* via adjusting the coefficient β (Baeza-Yates, Ribeiro-Neto, et al., 1999). *G-Mean* is the geometric mean of positive accuracy and negative accuracy (Kubat, Matwin, et al., 1997), it considers performances on both majority and minority classes.

The Receiver Operating Characteristic (ROC) curve (Fawcett, 2004; Fawcett, 2006) is a graphical evaluation technique which assesses the classification ability of a binary classifier. It is a graphical plot depicting all possible trade-offs between true positive rate (TPR) and false positive rate (FPR) (S. Wang, 2011a), which are defined as

$$\begin{aligned} TPR &= \frac{TP}{TP+FN}; \\ FPR &= \frac{FP}{FP+TN}. \end{aligned} \tag{2.4}$$

The ROC space is illustrated in Figure 2.4. According to the definition, a perfect classifier can be represented as $TPR = 1$ and $FPR = 0$, see broken line OAC. The worst classifier corresponds to broken line OBC with $TPR = 0$ and $FPR = 1$, indicating the classifier always makes wrong predictions. The diagonal from left bottom to the right top corner corresponds to a random-guessing classifier with $TPR = FPR$. The ROC space is divided into two parts by this diagonal, where the upper half indicates good classification results (better than random) and the lower half indicates bad classification results (worse than random). L1 and L2 represent two ROC curves, and the classifier corresponding to L2 outperforms the classifier corresponding to L1.

Associated with the ROC curve, the Area Under the ROC Curve (AUC) can be computed by estimating the area using quadrature, i.e. the AUC value varying in [0, 1]. It is used as an evaluation criterion for comparing the performance of different classifiers (Fawcett, 2004; Fawcett, 2006). If we rank the samples according to the predicted score produced by the classifier, AUC can be understood as the probability that the classifier will rank a randomly selected positive sample

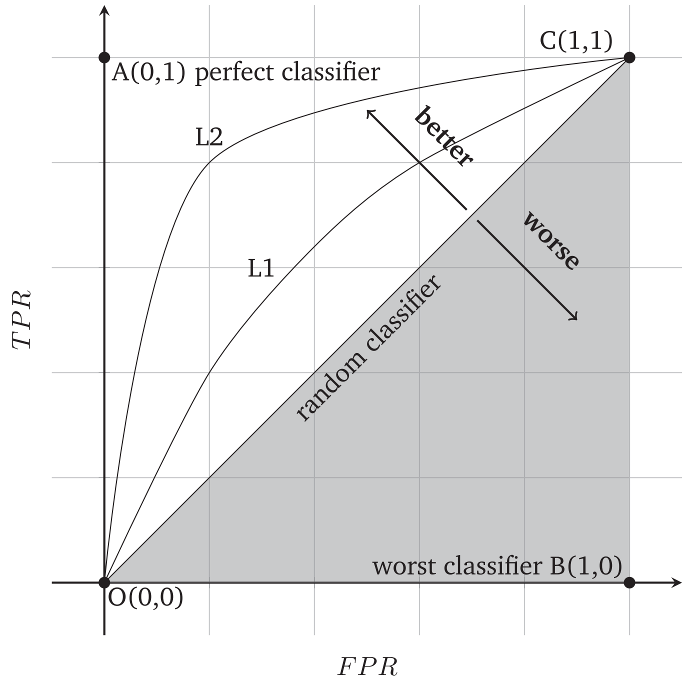


Figure 2.4: ROC space representation.

higher than a randomly selected negative sample (Hand and Till, 2001). The AUC of the random classifier is 0.5 and is highlighted in gray in Figure 2.4. The AUC value of a perfect classifier is equal to 1.

2.2 Multi-Class Imbalance Learning

Most studies in the imbalanced learning domain devote to the binary imbalanced scenario. However, a significant number of imbalanced real-world applications contain more than two classes, for instance, image classification, protein classification and medical diagnosis. The increasing number of classes poses new challenges for learning from multi-class imbalanced problems. First of all, more decision boundaries need to be defined during the multi-class classification process. Another challenging issue is that the imbalance among classes becomes more complicated as there will be multi-majority and multi-minority classes (S. Wang, Minku, and Yao, 2016). The data complexity, an important cause of the degradation in binary case (López, Fernández, García, Palade, and Herrera, 2013), is more sophisticated. Several solutions designed for binary imbalanced classification are extended to multi-class scenarios. In this section, we review the existing approaches and performance metrics for multi-class imbalanced learning.

2.2.1 Existing Approaches

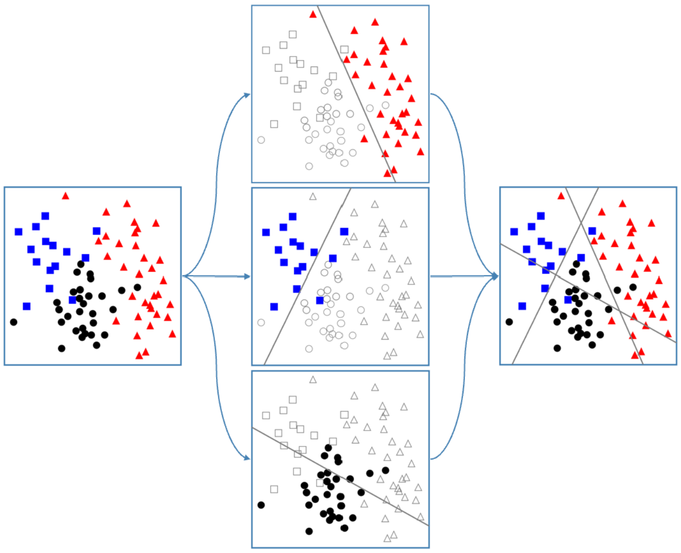
In this section, we first introduce the decomposition strategies for handling the multi-class imbalanced problems. After that, other methods, including preprocessing techniques and classification algorithms designed for multi-class scenarios, are described.

Decomposition Strategies

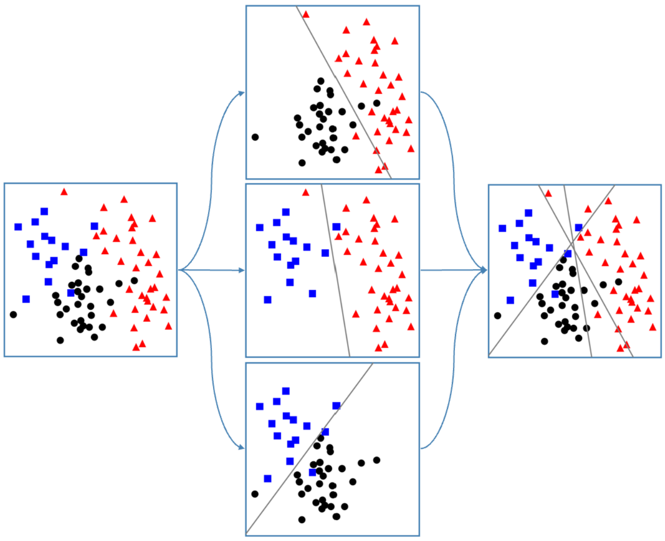
Class decomposition is an intuitive method to deal with multi-class imbalanced problems (Galar, Fernández, Barrenechea, Bustince, and Herrera, 2011). After transforming the multi-class problem into multiple subsets, the existing approaches for handling the binary scenarios can be applied directly. Among several decomposition strategies, One-vs-Rest (OVR) and One-vs-One (OVO) are the most commonly used in the literature.

Suppose there are C classes in the multi-class imbalanced problem. In the OVR decomposition, each of the C classes is trained against the remaining $(C-1)$ classes (Rifkin and Klautau, 2004). In other words, a C-class imbalanced problem is decomposed into C binary classification problems. When predicting the final label for a test sample, each binary classifier provides a prediction with confidence, and the prediction with the highest confidence is usually determined as the final label for this test sample. An illustration of the OVR scheme for a 3-class problem is shown in Figure 2.5a. While OVR provides the convenience of treating multi-class scenarios as binary scenarios, it also brings further imbalance into the binary subsets. In addition, all the individual classifiers are trained with the complete dataset; this ensures that no information is dropped in the training procedure. However, this also preserves the overlapping regions, a factor leading to the degradation of the classification performance (López, Fernández, García, Palade, and Herrera, 2013).

In the OVO decomposition, each of the C classes is trained against one of the remaining classes (Fürnkranz, 2002). Thus, a C-class imbalanced problem is decomposed into $C(C-1)/2$ binary problems. The final predictions are usually determined via the majority voting strategy. An illustration of the OVO scheme for



(a) Illustrations of OVR scheme for a 3-class problem.



(b) Illustrations of OVO scheme for a 3-class problem.

Figure 2.5: Illustrations of OVR and OVO scheme for a 3-class problem (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018).

a 3-class problem is shown in Figure 2.5b. Each binary classifier is only trained with pairs of classes; this makes the decision boundaries much simpler and properly addresses the overlapping issue. However, when pairing the classes, the number of binary classifiers increases in a quadratic rate of C (Tan, Gilbert, and Deville, 2003; S. Wang, 2011b). The training time can be long if C is large.

Approaches for Handling Multi-class Imbalanced Problems

The decomposition strategies are prevalent in addressing multi-class problems due to their straightforward idea and simple implementation. With the advantages of the decomposition strategies, many binary imbalanced approaches are extended to deal with multi-class imbalanced problems. Liao applied OVR and resampling techniques on the weld flaw classification problem specifically (Liao, 2008). Fernandez et al. reported a thorough experimental analysis on the combination of decomposition strategies and popular resampling techniques (Fernández, López, Galar, Del Jesus, and Herrera, 2013). They concluded that OVO and oversampling showed the best robustness in their experiments. Krawczyk proposed to embed a cost-sensitive Artificial Neural Networks (ANN) into OVO scheme for handling multi-class imbalanced data (Krawczyk, 2016). A classification framework has been proposed in (Sen, Islam, Murase, and Yao, 2015) to efficiently handle multi-class imbalanced problems. The framework is based on the OVR strategy and the boosting technique focuses on hard-to-learn samples in each base classifier. Meanwhile, oversampling techniques are applied to increase the sample weight in minority classes.

Despite applying a decomposition strategy, there are also ad-hoc approaches for multi-class imbalanced problems. The Static-SMOTE resampling technique (Fernández-Navarro, Hervás-Martínez, and Gutiérrez, 2011), inspired by SMOTE (Chawla, Bowyer, Hall, and Kegelmeyer, 2002), is proposed to handle multi-class imbalanced datasets. In Static-SMOTE, the oversampling procedure is performed in m steps, and m is the number of classes (Fernández, López, Galar, Del Jesus, and Herrera, 2013). The number of samples in the minimum size class is duplicated using SMOTE in each iteration. The Mahalanobis Distance-Based Oversampling Technique (MDO) (Abdi and Hashemi, 2015) is also proposed to oversample the minority classes in multi-class scenarios. Instead of randomly oversampling the samples, MDO guarantees that the artificial samples have the same Mahalanobis distance (Mahalanobis, 1936) from the considered class mean as other samples from the considered class. Considering the excellent ability of ensemble algorithms, Sun et al. (Sun, Kamel, and Y. Wang, 2006) proposed a cost-sensitive boosting algorithm to handle multi-class imbalanced problems. The core ideas are first to find an appropriate cost matrix, then to apply a Genetic Algorithm to search the optimum cost setup of each class. AdaBoost.NC (S. Wang, H. Chen, and Yao, 2010), a negative correlation learning algorithm, was proposed to address binary classification by introducing diversity among base classifiers. This work was extended to multi-class scenarios (S. Wang and Yao, 2012). Their experimental results reveal that combining AdaBoost.NC and oversampling techniques have a better ability to recognise samples from minority classes and achieve a high G-mean among classes even without decomposition strategies.

2.2.2 Performance Metrics

When choosing the performance metrics for multi-class imbalanced problems, both the performance for each class and the overall performance must be taken into account. The single-class performance metrics introduced in Section 2.1.2 are still suitable for multi-class scenarios. There is no standard performance metric to measure the overall classifier performance in the multi-class imbalanced learning domain. We consider two overall performance metrics in this thesis.

The average accuracy is commonly used to evaluate the multi-class imbalanced classification performance (Ferri, Hernández-Orallo, and Modroiu, 2009). It is computed by

$$\mathit{MAcc} = \frac{1}{C}\sum_{i=1}^{C} TPR_i. \tag{2.5}$$

The Multi-class Area Under the Curve (MAUC), an extension of AUC, is another commonly used to measure the multi-class classification performance of the whole dataset (Hand and Till, 2001). It is the average pairwise AUC values of all paired classes and is defined as

$$\mathit{MAUC} = \frac{2}{C \cdot (C-1)} \sum_{j<k} \hat{A}(j,k), \tag{2.6}$$

where $\hat{A}(j,k) = [\hat{A}(j|k) + \hat{A}(k|j)]/2$ is the measure of separability between classes j and k. $\hat{A}(j|k)$ indicates the probability that a sample randomly selected from class k has a lower probability for class j than randomly selected from class j, and $\hat{A}(k|j)$ is defined correspondingly. A detailed equation to compute $\hat{A}(j,k)$ can be found in (Hand and Till, 2001).

Apart from overall performance, one main aim of studying the imbalanced problem is to improve the classification accuracy on minority class(es) while not losing too much accuracy on majority class(es). In this thesis, we use *MinAcc*, the average accuracy on minority class(es), to measure the performance on minority class(es). It is computed by

$$\text{MinAcc} = \sum_{i \in \text{C}_{\text{minority}}} \text{TPR}_i / n_{\text{minority}}, \tag{2.7}$$

where $\text{C}_{\text{minority}}$ denotes the set of minority class indices, TPR_i is the true positive rate in class i, n_{minority} denotes the number of minority classes. If there is more than one class being underrepresented in multi-class imbalanced classification, one should manually define the value of n_{minority}.

2.3 Data Complexity for Imbalanced Datasets

The class imbalance was widely considered as the main reason for performance degradation. However, there are highly imbalanced problems with good classification performance. This situation caught the attention of various researchers and they addressed the importance of data complexity in the imbalanced datasets (López, Fernández, García, Palade, and Herrera, 2013; Prati, Batista, and Monard, 2004). Weng et al. performed an analysis of the data complexity to gain some insights on their imbalanced datasets (Weng and Poon, 2006). In (Luengo, Fernández, García, and Herrera, 2011), authors concluded that, according to their experimental results, the imbalance ratio by itself cannot be considered as a determinant factor for degradation in performance. Researchers in (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018) analyzed the relationship between data complexity measures and the classification performance with and without applying the resampling techniques. They confirmed that the performance with oversampling techniques is related to the data complexity in a quasi-linear. This section first introduces two types of data complexity measures: feature overlapping measures and measures of separability of classes. After that, four types of samples in the imbalanced domain are described.

2.3.1 Overlapping and Class Separability

When studying the data complexity measures in binary classification problems, *feature overlapping measures* and *measures of the separability of classes* are commonly considered (Ho and Basu, 2002), where the former characterize how informative the features classify the classes and the latter try to quantify the linear separability of the classes (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019). A summary of the two types of measures is shown in Table 2.2.

Table 2.2: Summary of the data complexity measures. "Positive" and "Negative" indicate the positive and negative relation between measure value and data complexity respectively.

Measure	**Description**	**Relation**
F1	Maximum Fisher's Discriminant Ratio	Negative
F1v	The Directional-vector Maximum Fisher's Discriminant Ratio	Negative
F2	Volume of Overlapping Region	Positive
F3	Maximum Individual Feature Efficiency	Negative
L1	Sum of the Error Distance by Linear Programming	Positive
L2	Error Rate of Linear Classifier	Positive
L3	Non-Linearity of a Linear Classifier	Positive

Feature Overlapping Measures

The *maximum Fisher's discriminant ratio*, denoted by F1, measures the overlap between the feature values of different classes and is given by (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019):

$$F1 = \max_{i=1}^{m} r_{f_i}, \tag{2.8}$$

where m is the number of features, r_{f_i} is the discriminant ratio for each feature f_i. In a binary classification problem, r_{f_i} can be calculated as follows (Kong, Kowalczyk, D. A. Nguyen, Menzel, and Bäck, 2019; Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019):

$$r_{f_i} = \frac{\sum_{c=1}^{2} n_c (\mu_c^{f_i} - \mu^{f_i})^2}{\sum_{c=1}^{2} \sum_{j=1}^{n_c} (x_j^c - \mu_c^{f_i})^2}, \tag{2.9}$$

where n_c is the number of examples in class c, $\mu_c^{f_i}$ is the mean value of feature f_i across class c, μ^{f_i} is the mean value of feature f_i across all classes, and x_j^c represents the value of feature f_i for a sample from class c (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019). An example of F1 computation is given in Figure 2.6.

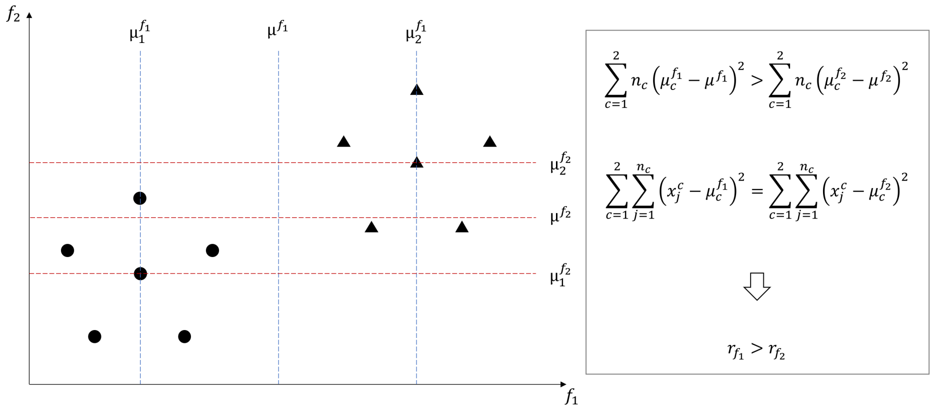

f_2
$\mu_1^{f_1}$
μ^{f_1}
$\mu_2^{f_1}$
$\mu_2^{f_2}$
μ^{f_2}
$\mu_1^{f_2}$
f_1
$\sum_{c=1}^{2} n_c\left(\mu_c^{f_1}-\mu^{f_1}\right)^2 > \sum_{c=1}^{2} n_c\left(\mu_c^{f_2}-\mu^{f_2}\right)^2$
$\sum_{c=1}^{2}\sum_{j=1}^{n_c}\left(x_j^c-\mu_c^{f_1}\right)^2 = \sum_{c=1}^{2}\sum_{j=1}^{n_c}\left(x_j^c-\mu_c^{f_2}\right)^2$
$r_{f_1} > r_{f_2}$

Figure 2.6: Example of F1 computation for a binary dataset (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019).

The *directional-vector maximum Fisher's discriminant ratio*, F1v, is a complement of F1 and a higher value of F1v indicates that there exists a vector which can separate different class samples after these samples are projected on it (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019; Orriols-Puig, Macia, and Ho, 2010). It computes the two-class Fisher's criterion defined in (Malina, 2001) as:

$$F1v = \frac{\boldsymbol{d}^t\boldsymbol{B}\boldsymbol{d}}{\boldsymbol{d}^t\boldsymbol{W}\boldsymbol{d}}, \tag{2.10}$$

where

- $\boldsymbol{d}$ is the directional vector on which the data are projected;
- $\boldsymbol{B} = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^t$ is the between-class scatter matrix and μ_1, μ_2 are the mean vector of the two classes;
- $\boldsymbol{W} = p\Sigma_1 + (1-p)\Sigma_2$ and p is the proportion of samples in one class and Σ_1 is the scatter matrix of the same class, and Σ_2 is the scatter matrix of the other class.

The directional vector $\boldsymbol{d}$ is calculated (Orriols-Puig, Macia, and Ho, 2010) by

$$\boldsymbol{d} = \boldsymbol{W}^{-1}(\mu_1 - \mu_2), \tag{2.11}$$

where the $\boldsymbol{W}^{-1}$ is the pseudo-inverse of $\boldsymbol{W}$ (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019; Orriols-Puig, Macia, and Ho, 2010).

The *volume of overlapping region*, denoted by F2, calculates the overlap ratio of all features (the width of the overlap interval in relation to the width of the entire interval) and returns the product of the ratios of all features (Orriols-Puig, Macia, and Ho, 2010), as shown below.

$$\begin{aligned} F2 &= \prod_i^m \frac{\mathit{overlap}(f_i)}{\mathit{range}(f_i)} \\ &= \prod_i^m \frac{\max\{0, \min\max(f_i) - \max\min(f_i)\}}{\max\max(f_i) - \min\min(f_i)}, \end{aligned} \tag{2.12}$$

where

$$\begin{aligned} \min\max(f_i) &= \min(\max(f_i^{c_1}), \max(f_i^{c_2})), \\ \max\min(f_i) &= \max(\min(f_i^{c_1}), \min(f_i^{c_2})), \\ \max\max(f_i) &= \max(\max(f_i^{c_1}), \max(f_i^{c_2})), \\ \min\min(f_i) &= \min(\min(f_i^{c_1}), \min(f_i^{c_2})), \end{aligned} \tag{2.13}$$

where $(f_i^{c_1})$ and $(f_i^{c_2})$ are the values of the feature i for the two classes.

The *maximum individual feature efficiency* (F3) computes the individual feature efficiency and returns the maximum value among all features (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019; Orriols-Puig, Macia, and Ho, 2010). For each feature, the overlapping region is taken into account, and the ratio of the number of examples not in the overlapping region to the total number of examples is returned as F3.

Linearity Measures

L1 and L2 measure to what extent the classes can be linearly separated using an SVM with a linear kernel (Orriols-Puig, Macia, and Ho, 2010), where L1 returns the sum of the distances of the misclassified samples to the linear boundary and L2 returns the error rate of the linear classifier. An example of L1 and L2 computation is given in Figure 2.7. L3 returns the error rate of an SVM with linear kernel on a test set, where the SVM is trained on training samples and the test set is manually created by performing linear interpolation on the two randomly chosen samples from the same class.

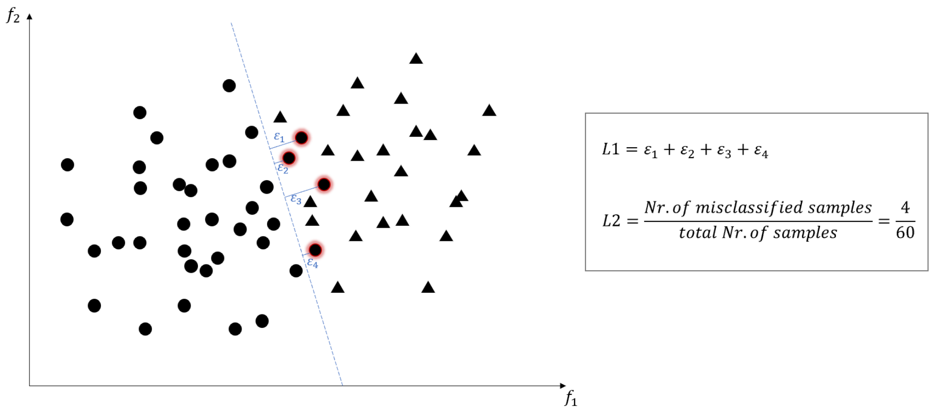

f_2
f_1
ε_1
ε_2
ε_3
ε_4
$L1 = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4$
$L2 = \frac{Nr.\,of\ misclassified\ samples}{total\ Nr.\,of\ samples} = \frac{4}{60}$

Figure 2.7: Example of L1 and L2 computation for a binary dataset (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019).

2.3.2 Types of Sample in Imbalanced Domain

Napierala and Stefanowski proposed to analyse the local characteristics of minority class samples by dividing them into four different types: *safe, borderline, rare* samples and *outliers* (Napierala and Stefanowski, 2016), the latter three are called *unsafe* samples. The identification of the type of an example can be done through modeling its k-neighbourhood. Considering that many applications involve both nominal and continuous attributes, the HVDM metric is applied to calculate the distance between different examples.

Heterogeneous Value Difference Metric (HVDM)

HVDM is a heterogeneous distance function that returns the distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ (D. R. Wilson and Martinez, 1997), where the vectors can involve both nominal and numerical attributes. The HVDM distance is defined by

(D. R. Wilson and Martinez, 1997):

$$HVDM(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{a=1}^{n} {d_a}^2(x_a, y_a)}, \tag{2.14}$$

where n is the number of attributes. The function $d_a(\cdot)$ returns the distance between x_a and y_a, where x_a, y_a indicate the ath attribute of vector x and y respectively. It is defined as follows:

$$d_a(x,y) = \begin{cases} 1, & \text{if } x \text{ or } y \text{ is unknown, i.e. NA} \\ \text{norm\_vdm}_a(x,y), & \text{if } a\text{th attribute is nominal} \\ \text{norm\_diff}_a(x,y), & \text{if } a\text{th attribute is continuous} \end{cases} \tag{2.15}$$

where

$$\text{norm\_vdm}_a(x,y) = \sqrt{\sum_{c=1}^{C} \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^2}, \quad \text{norm\_diff}_a(x,y) = \frac{|x-y|}{4\sigma_a}, \tag{2.16}$$

where

- C is the number of total output classes,
- $N_{a,x,c}$ is the number of instances which have value x for the ath attribute and output class c and $N_{a,x} = \sum_{c=1}^{C} N_{a,x,c}$,
- σ_a is the standard deviation of values of the ath attribute.

Identification Rule to Assign Types of Sample

The four types of samples in binary scenario are determined by the neighbourhood information, taking a sample from minority class as an example:

- a sample is considered to be **safe** if the majority of the neighbours belongs to the same class;
- a sample is considered to be **borderline** if the proportion of the neighbours in both classes is approximately the same;
- a sample is considered to be **rare** if the majority of the neighbours belongs to a different class;

- a sample is considered to be an **outlier** if all the neighbours belongs to a different class.

Given the number of neighbours k, the label to a sample from minority class can be assigned through the ratio of the number of its neighbours from minority class to the total number of neighbours ($R_{\frac{min}{all}}$) according to Table 2.3. The label for a sample from majority class can be assigned in a similar way. Given the number of neighbours k, the label to a sample from majority class can be assigned through the ratio of the number of its neighbours from majority class to the total number of neighbours ($R_{\frac{maj}{all}}$).

Table 2.3: Identification rule to assign types for samples from minority class. $R_{\frac{min}{all}}$ is the ratio of the number of its neighbours from minority class to the total number of neighbours.

Type	Rule	Rule ($k = 5$)
Safe	$\frac{k+1}{2k} < R_{\frac{min}{all}} \leqslant 1$	$\frac{3}{5} < R_{\frac{min}{all}} \leqslant 1$
Borderline	$\frac{k-1}{2k} \leqslant R_{\frac{min}{all}} \leqslant \frac{k+1}{2k}$	$\frac{2}{5} \leqslant R_{\frac{min}{all}} \leqslant \frac{3}{5}$
Rare	$0 < R_{\frac{min}{all}} < \frac{k-1}{2k}$	$0 < R_{\frac{min}{all}} < \frac{2}{5}$
Outlier	$R_{\frac{min}{all}} = 0$	$R_{\frac{min}{all}} = 0$

2.4 Imbalanced Benchmark Datasets and Applications

This section first introduces one of the dataset repositories for learning from imbalanced benchmark datasets. After that, a gentle introduction to imbalanced applications is given.

2.4.1 KEEL-Dataset Repository

KEEL (Knowledge Extraction based on Evolutionary Learning) is an open-source software [1] which was initially developed to implement evolutionary algorithms and deal with some standard data mining tasks (Alcalá-Fdez, Fernández, Luengo,

[1]http://www.keel.es

Derrac, García, Sánchez, and Herrera, 2011), e.g. classification and regression. A dataset repository is also provided in KEEL [2], it provides a set of quality benchmark datasets, allowing comparative studies for various researchers.

Regarding imbalanced classification, there are various binary benchmark datasets with imbalanced ratios varying from 1.5 to 130. Most of the datasets can also be found in the UCI repository [3]; however, the datasets in UCI always require some preprocessing step, i.e. one has to deal with the missing values by himself/herself. Datasets in KEEL are in good structure and can be used in the experiments directly. Please note that many binary datasets in KEEL are artificially derived from multi-class classification problems using decomposition strategies. There are also 15 multi-class imbalanced benchmark datasets available. Most experiments in this thesis are based on the datasets in KEEL. Several experiments use the datasets from our industrial partners. Information on these datasets can be found in our Marie-Curie ITN project GitHub repository [4].

2.4.2 Imbalanced Applications

The imbalanced problems widely exist in many real-world scenarios. This section briefly reviews several imbalanced applications in engineering, information technology, bioinformatics, and medicine.

Back to the end of the 1990s, Kubat et al. (Kubat, R. Holte, and Matwin, 1997; Kubat, R. C. Holte, and Matwin, 1998; Kubat, Matwin, et al., 1997) dealt with the detection of oil spills in satellite radar images. It is very challenging to detect oil spills in satellites' radar images since they reflect less light. The class imbalance in the problem (41 oil spills and 896 images without oil spills) makes the problem even more challenging. These challenges drove them to propose one-side selection (OSS) to sample the data points. OSS will be introduced later in this thesis. The class imbalance applications are also widely studied in various engineering sub-domains, such as fault detection in semiconductors (T. Lee, K. B. Lee, and Kim, 2016), short-term voltage stability assessment (Zhu, Lu, Dong, and Hong, 2017), fault diagnosis in wind turbines (Wu, Lin, and Ji, 2018) and etc.

In information technology, software defect prediction is necessary for quality control in order to detect possible failures. Rodriguez et al. (Rodriguez, Herraiz,

[2]http://www.keel.es/datasets.php
[3]https://archive.ics.uci.edu/ml/index.php
[4]https://github.com/ECOLE-ITN

Harrison, Dolado, and Riquelme, 2014) compared the effectiveness of different approaches for handling the class imbalance in the problem. They concluded that combining the ensemble methods and feature selection scheme is robust in dealing with the proposed problem. Due to the current advances, applications network analysis and computer vision are also proposed, for instance, mobile malware detection (Z. Chen, Yan, Han, S. Wang, Peng, L. Wang, and B. Yang, 2018) and object recognition in images (X. Zhang, Zhuang, W. Wang, and Pedrycz, 2016).

One well-known application in Bioinformatics is protein identification. The detection of Micro RNAs is crucial due to their high importance in post-transcriptional regulation of gene expression of plants and animals (Lertampaiporn, Thammarongtham, Nukoolkit, Kaewkamnerdpong, and Ruengjitchatchawalya, 2013). The authors proposed a modified-SMOTEbagging for pre-miRNA classification. The imbalanced applications in medicine contain medicine quality (Zięba, Tomczak, Lubicz, and Świątek, 2014), lung nodule detection (Cao, J. Yang, W. Li, D. Zhao, and Zaiane, 2014), diagnosis of diabetes mellitus (Z. Chen, Yan, Han, S. Wang, Peng, L. Wang, and B. Yang, 2018), microaneurysm (Ren, Cao, W. Li, D. Zhao, and Zaiane, 2017) and other diseases.