Universiteit Leiden
The Netherlands

Learning class-imbalanced problems from the perspective of data intrinsic characteristics

Kong, J.

Learning Class-Imbalanced Problems from the Perspective of Data Intrinsic Characteristics

Proefschrift

ter verkrijging van
de graad van doctor aan de Universiteit Leiden,
op gezag van rector magnificus prof.dr.ir. H. Bijl,
volgens besluit van het college voor promoties
te verdedigen op woensdag 27 september 2023
klokke 13:45 uur

door

Jiawen Kong
geboren te Harbin, China
in 1995

Promotores:
Prof.dr. T.H.W. Bäck
Prof.dr. B. Sendhoff (TU Darmstadt, Germany)

Co-promotor:
Dr. W.J. Kowalczyk

Promotiecommissie: *(Dutch)*
Prof.dr. A. Plaat.
Prof.dr. M.M. Bonsangue
Dr. A.V. Kononova
Prof.dr. S. Mostaghim (Otto von Guericke University of Magdeburg, Germany)
Dr. M. López-Ibáñez (The University of Manchester, UK)

This research has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 766186 (ECOLE).

Contents

CHAPTER 1

Introduction

1.1 Background

Machine learning is a broad multidisciplinary research area which is based on many different branches of mathematics and science, including computer science, statistics, cognitive psychology, engineering, and optimisation theory (Soofi and Awan, 2017). Machine learning algorithms aim to learn from the data and build the models in order to make predictions on unseen samples. These algorithms have numerous applications, for instance, medical diagnosis (Acharya, Chowriappa, Fujita, Bhat, S. Dua, Koh, Eugene, Kongmebhol, and K. Ng, 2016), product recommendations (Misra, Wan, and McAuley, 2018), defect detection (Haddad, S. Yang, Karam, Ye, Patel, and Braun, 2018), video surveillance (Radtke, Granger, Sabourin, and Gorodnichy, 2014), computer vision (X. Zhang, Zhuang, W. Wang, and Pedrycz, 2018) and self-driving cars (Carranza-García, Lara-Benítez, García-Gutiérrez, and Riquelme, 2021).

Supervised and unsupervised learning are the two main sub-categories of machine learning algorithms. Supervised learning algorithms are trained with a given set of samples with input attributes and outputs, and the goal is to find the relationship between the input attributes and the output responses. On the other hand, no output responses are given when training the unsupervised learning algorithms, and the goal is to discover the structure of the data itself, for example clustering the data points.

Classification and regression problems are the two essential branches of supervised learning. Classification is an important research area in the field of machine learning and data mining. Classification predictive modelling refers to the task of estimating the mapping function from input attributes to discrete class labels. In other words, training a classification model is to predict the class labels of samples with given input attributes. For example, spam email filtering can be regarded as a two-class classification task, where the two class labels are *spam* and *not spam*. Regression predictive modelling refers to the task of estimating the mapping function from input attributes to continuous outputs, i.e. a regression predictive model returns a quantity. For example, predicting the house price based on the size, location, age, and other aspects of the house is a regression task, where these aspects of the house are the input attributes, and the house price is the continuous output.

Several machine learning algorithms have been proposed in the literature to handle classification tasks, for instance, logistic regression, decision tree, random forest, support vector machine, and k-Nearest Neighbour (KNN). Most of the proposed algorithms are designed under two main assumptions:

- The classes are equally distributed;
- The cost of classification errors, so-called misclassification costs, are equal.

However, both assumptions do not always hold in real-world applications. Many real-world classification problems suffer from significant differences in the number of samples in each class. Moreover, the classification costs in these real-world classification problems cannot be treated equally. Taking an example of cancer diagnosis, a cancer case is much less likely to occur than healthy cases, i.e. samples in the *cancer* class and *healthy* class are not equally frequent. In this problem, the rare samples (cancer) are more important, and their classification costs are higher. Failure to identify a cancer case will lead to a person's life loss. Such problems where one or more classes are underrepresented are known as *class-imbalance problems*. Standard classifiers aim to maximise the overall accuracy, i.e. the proportion of all correctly classified validation samples in percentage, and always perform poorly on such problems. For example, if a standard classifier is used to handle a cancer diagnosis task with 95 healthy cases and 5 cancer cases, even if it would classify all samples into *healthy* class and still achieve 95% accuracy overall. However, the classification accuracy on *cancer* class is 0%, which makes the classifier useless in practice.

Class-Imbalance Learning Strictly speaking, any dataset with an unequal class distribution can be considered imbalanced. However, in the imbalanced domain, a dataset is defined as imbalanced only when the samples in different classes have a significant or even extreme gap in the number of samples (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018). In other words, one or more classes significantly outnumber the other class(es) in an imbalanced dataset. The classes with more samples are called *majority* classes, while the underrepresented classes are called *minority* classes. The skewed distribution in the dataset will make the classifiers biased toward the majority class(es) and the minority class(es) to be overlooked. However, from the application point of view, the underrepresented (minority) class is usually the class of interest and has higher misclassification costs in the problem. For example, it is more important to correctly identify the minority class samples in medical diagnosis, email filtering, and fault detection. The price of misclassifying the minority samples would be a massive loss of money in fault diagnosis, an unqualified product in anomaly detection and a person's life in medical diagnosis. In contrast, the misclassification cost on majority class samples is only a double check. Hence, it is of vital importance to study class-imbalance problems.

Class-imbalance problems have caught growing attention from both academic and industrial fields. Many techniques have been developed to alleviate the influence of *class imbalance* and can be categorized into four broad groups.

1. *Data-level* approaches rebalance the class distribution directly via resampling the data space. These approaches manipulate the data directly and are easy to implement in real-world applications as a preprocessing step.

2. *Algorithm-level* approaches adapt the classification algorithms to force the learning bias toward the minority class. However, the adjustments always require a deep understanding of the corresponding algorithms, i.e. the adjustments are algorithm-specific.

3. *Cost-sensitive* learning techniques handle the class-imbalance problems by considering the unequal misclassification costs. This technique can be combined with *data-level* and *algorithm-level* approaches.

4. *Ensemble-based* methods in imbalanced learning domain usually combine an ensemble learning algorithm and one of the approaches above.

Data Complexity in Imbalanced Learning Domain The *class imbalance* was widely considered as the determinant for performance degradation. However, researchers have observed that in some cases, good classification performances can be achieved even in the presence of significant class imbalance for problems with low complexity, such as a linearly separable problem. This suggests that the class imbalance itself cannot be considered as the main reason for the performance degradation, and naturally extends the research direction to the study of data complexity. The difficulty of a supervised classification problem can be characterized by several data complexity measures in the literature, including (i) *feature overlapping measures*; (ii) *measures of the separability classes*; and (iii) *measures of geometry, topology and density of manifolds*. These measures are used to gain insights into the performance of data-level approaches in (Weng and Poon, 2006). The relationship between data complexity measures and imbalanced classification performance is studied in (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018). Small disjuncts are problematic under class-imbalance situations because they are difficult to distinguish from noise (Jo and Japkowicz, 2004; López, Fernández, García, Palade, and Herrera, 2013). It is also interesting to study the samples as different types: *safe*, *borderline*, *rare* samples and *outliers* according to their local characteristics (Napierala and Stefanowski, 2016) or their distances to the decision boundary (Kubat, Matwin, et al., 1997).

Motivated by the studies on *Data Complexity in Imbalanced Learning Domain* (López, Fernández, García, Palade, and Herrera, 2013; Luengo, Fernández, García, and Herrera, 2011; Prati, Batista, and Monard, 2004; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018; Weng and Poon, 2006), as well as the study showing that "*Better Data*" *is Better than* "*Better Data Miners*" (Agrawal and Menzies, 2018), this thesis mainly conducts the research on *Learning Class-Imbalanced Problems from the Perspective of Data Intrinsic Characteristics*.

1.2 Research Questions

In this thesis we focus on learning class-imbalanced problems from the perspective of data intrinsic characteristics. To achieve our objective, the following research questions are considered.

RQ1: What is the difference between the effectiveness of "classical" and "new" resampling techniques?

Resampling techniques have been proven effective in handling class-imbalance problems, and many resampling techniques have been proposed in the literature. However, most empirical studies and application work still focus on "classical" resampling techniques and do not take newly developed ones into account. Distinguishing the oversampling techniques into "classical" and "new" and studying their effectiveness will provide researchers with insights on choosing appropriate techniques.

RQ2: What is the relationship between data complexity measures and the choice of oversampling techniques?

Researchers in the imbalanced learning domain not only focus on developing novel approaches but also emphasize the importance of understanding the problem at a deeper level. The more complex the data, the more difficult the classification is. *Imbalance* is not the unique factor hindering the classification. It is also of vital importance to understand how other data characteristics influence the imbalanced classification performance and investigate the relationship between data complexity measures and the choice of oversampling techniques.

RQ3: What is the relationship between the degree of class overlap and the classification improvement obtained via hyperparameter tuning?

Hyperparameter optimisation has shown great effectiveness for many machine learning algorithms. When dealing with class-imbalance problems, hyperparameters in both resampling and classification algorithms should be considered in the experiments. A minor variation of these hyperparameters might influence the performance significantly. Nevertheless, this topic has not been studied in detail in the context of learning from imbalanced data. Therefore, we explore the potential of applying hyperparameter optimisation to construct high-quality classifiers for imbalanced data automatically. From **RQ2**, we already mentioned that data complexity influenced classification performance. Our question in this part becomes, will data complexity affect the classification improvement yielded via hyperparameter tuning?

RQ4: Can we take advantage of anomaly detection techniques to improve imbalanced classification?

The anomaly detection problem can be considered as a class-imbalance problem with an extreme imbalance in terms of class distribution. There are many techniques available in the literature to detect anomalies. Considering the similarity between the two problems, it is very interesting to study if we can improve performance for class-imbalance problems with anomaly detection ideas.

RQ5: How can the idea of four types of samples be effectively extended to multi-class imbalanced scenarios?

The idea of studying different types of samples (*safe, borderline, rare* samples and *outliers*) was first proposed and evaluated on binary imbalanced classification problems (Napierala and Stefanowski, 2016). Since the idea was proposed, it has attracted widespread attention in the field of imbalanced learning, and more than 200 papers have cited the original paper so far. Some studies then extended this idea to multi-class scenarios without considering the more complicated relationships among classes in multi-class imbalanced scenarios. Therefore, proposing improved sample type identification for multi-class imbalanced classification is worth studying.

RQ6: How applicable are the developed approaches to real-world problems?

The performances of the developed approaches are typically tested on benchmark problems. However, given that real-world problems are more complex, these approaches may fail in real-world situations. Therefore, it is important to validate the performance of the approaches on real-world applications.

1.3 Outline of the Thesis

This thesis is organised as follows.

Chapter 2 presents a gentle introduction to class imbalanced problems. It starts with the literature review on binary and multi-class class imbalance problems, including problem illustration, existing approaches and performance metrics. Moreover, it presents the data complexity measures and introduces studies on the data complexity in the imbalanced learning domain. Finally, it shows the benchmark datasets for learning from imbalanced problems and imbalanced applications in real-world scenarios.

Chapter 3 introduces an empirical investigation comparing several oversampling techniques. Apart from experimenting with imbalanced benchmark datasets, further exploration through data from a real-world inspired digital vehicle model is presented.

Chapter 4 presents our study on hyperparameter optimisation on class-imbalance problems. We consider optimising the hyperparameters in both resampling techniques and classification algorithms. Furthermore, we investigate the relationship between the degree of class overlap and the improvement yielded via hyperparameter tuning.

Chapter 5 introduces our idea of improving imbalanced classification via adding additional attributes. We propose introducing the outlier score, an important indicator to evaluate whether a sample is an outlier, as an additional attribute of the original imbalanced datasets. Apart from this, we also introduce the four types of samples (*safe, borderline, rare* samples and *outliers*) as another additional attribute.

Chapter 6 introduces our proposed improved sample type identification for multi-class imbalanced classification. We first show the drawbacks when applying the existing identification rule directly to multi-class scenarios. After that, we emphasize the importance of proposing a new identification rule for multi-class scenarios and introduce the improved type identification rule.

Chapter 7 presents the main conclusions of this thesis and the potential future research directions.

1.4 Publications

The main contributions of this thesis are based on the following publications:

- **Kong, J.**, Kowalczyk, W., Nguyen, D.A., Bäck, T. and Menzel, S., 2019, December. Hyperparameter optimisation for improving classification under class imbalance. In 2019 IEEE symposium series on computational intelligence (SSCI) (pp. 3072-3078). IEEE.

- **Kong, J.**, Rios, T., Kowalczyk, W., Menzel, S. and Bäck, T., 2020, May. On the performance of oversampling techniques for class imbalance problems. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 84-96). Springer, Cham.
- **Kong, J.**, Kowalczyk, W., Menzel, S. and Bäck, T., 2020, September. Improving imbalanced classification by anomaly detection. In International Conference on Parallel Problem Solving from Nature (pp. 512-523). Springer, Cham.
- **Kong, J.**, Kowalczyk, W., Jonker, K., Menzel, S. and Bäck, T., 2022, July. Improved Sample Type Identification for Multi-Class Imbalanced Classification with Real-World Applications. In International Conference on Data Science. (Accepted, publication in process)

Other work by the author:

- Rios, T., **Kong, J.**, van Stein, B., Bäck, T., Wollstadt, P., Sendhoff, B. and Menzel, S., 2020, December. Back to meshes: Optimal simulation-ready mesh prototypes for autoencoder-based 3D car point clouds. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 942-949). IEEE.
- Nguyen, D.A., **Kong, J.**, Wang, H., Menzel, S., Sendhoff, B., Kononova, A.V. and Bäck, T., 2021, October. Improved automated cash optimization with tree parzen estimators for class imbalance problems. In 2021 IEEE 8th international conference on data science and advanced analytics (DSAA) (pp. 1-9). IEEE.

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.

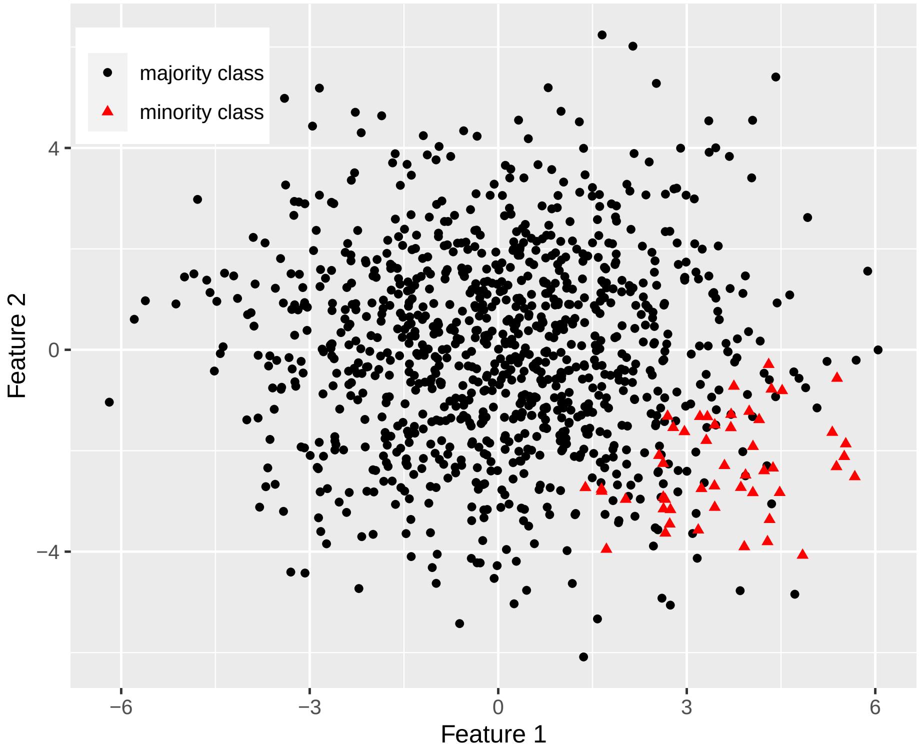


Figure 2.1: An example of a binary class imbalance problem with IR = 200.

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.

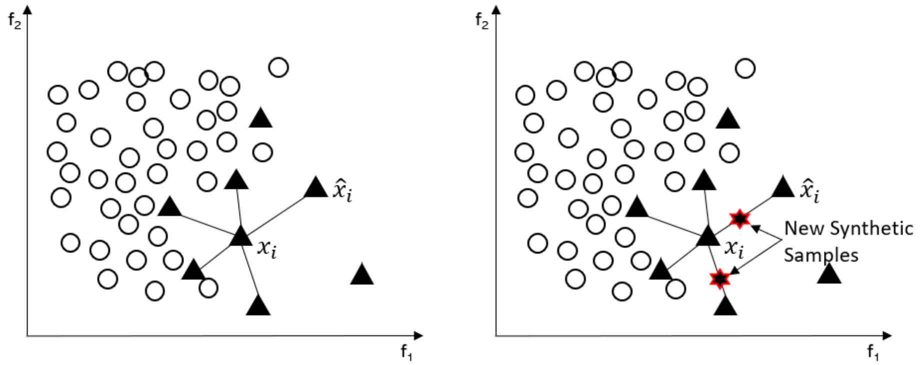


Figure 2.2: An illustration of how to generate synthetic samples through SMOTE. Example of K-nearest minority class neighbours for minority class sample $\boldsymbol{x_i}$ (K=5) (left) and new synthetic samples generated through SMOTE (right).

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

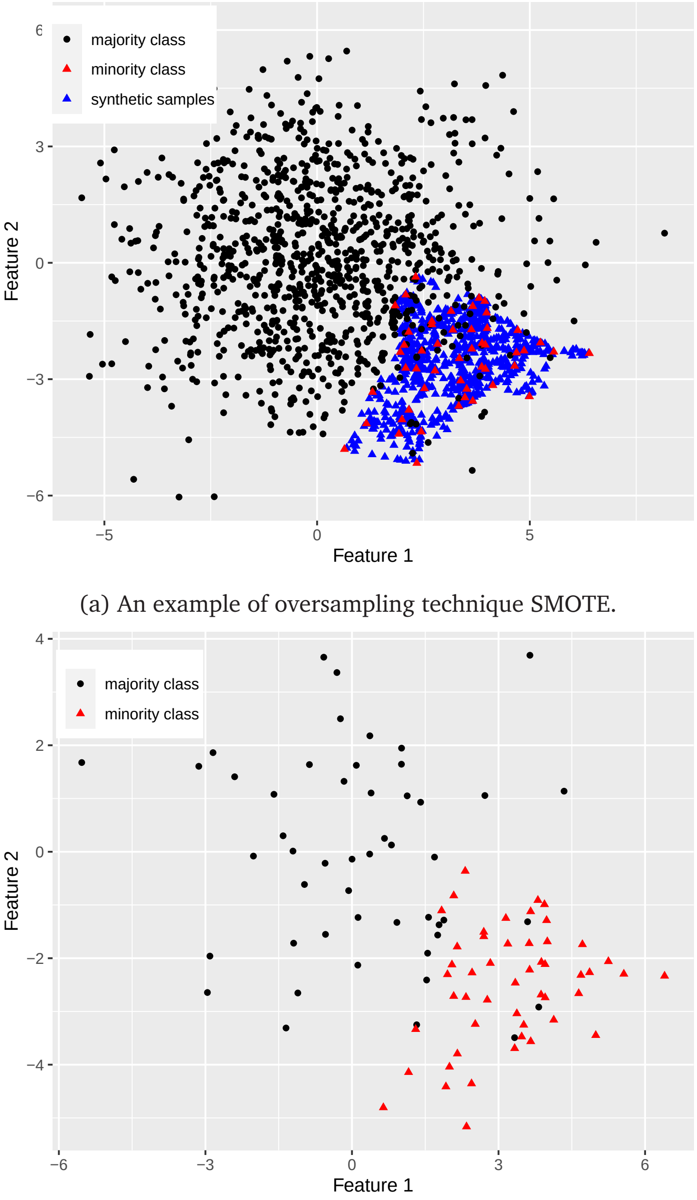
Ensemble-based classifiers, a combination of multiple classification algorithms, are known to produce better classification performance compared to a single classification algorithm (Rokach, 2010). Standard ensemble-based classifiers are not very effective to deal with skewed class distributions; however, 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

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



(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

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

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

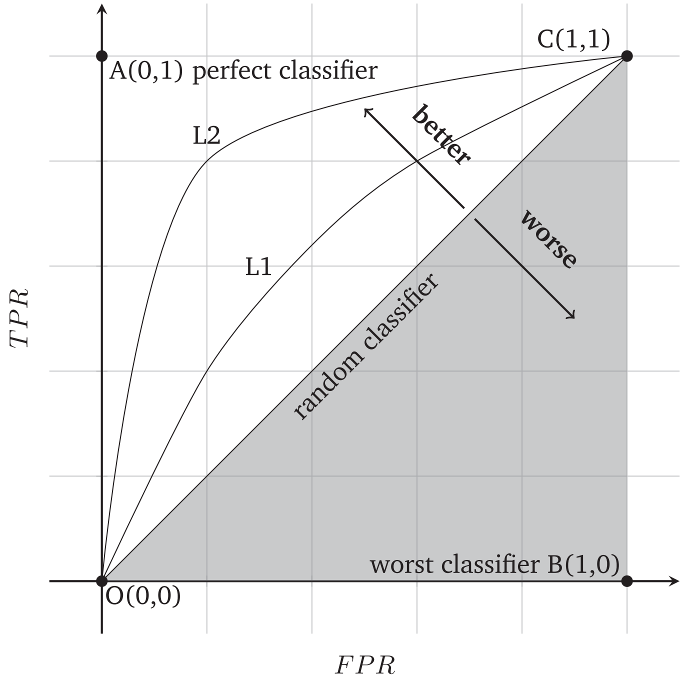


Figure 2.4: ROC space representation.

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

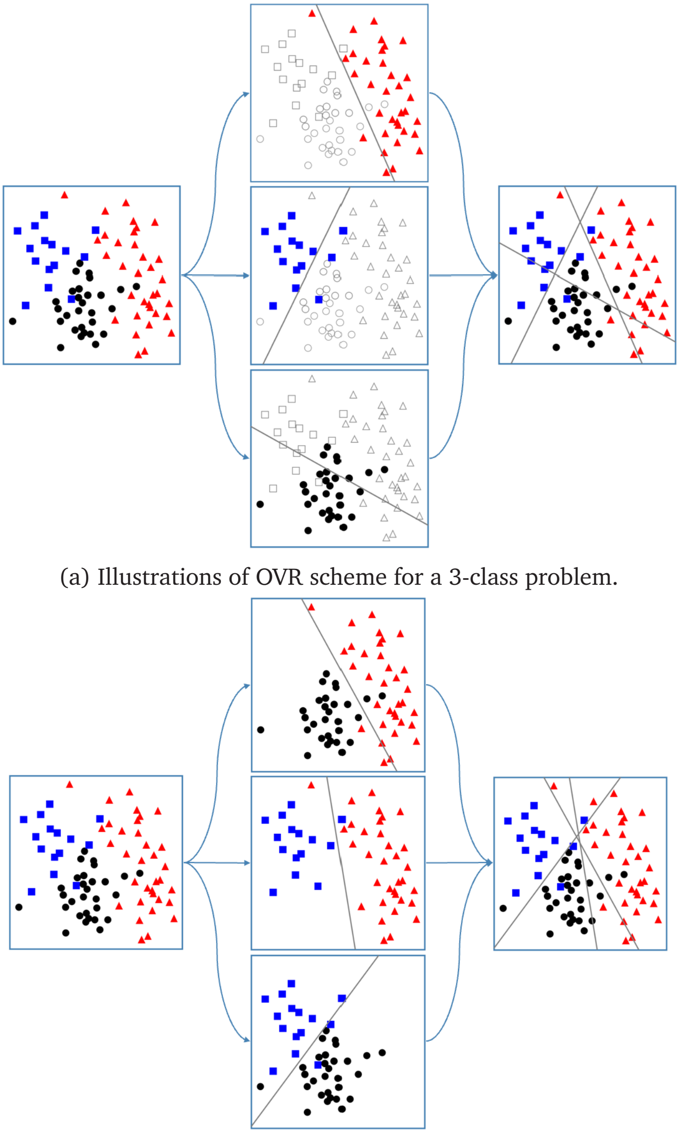
In this section, we first introduce the decomposition strategies for handling the multi-class imbalanced problems. After that, other methods, including preprocessing techniques and classification algorithms designed for multi-class scenarios, are described.

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.

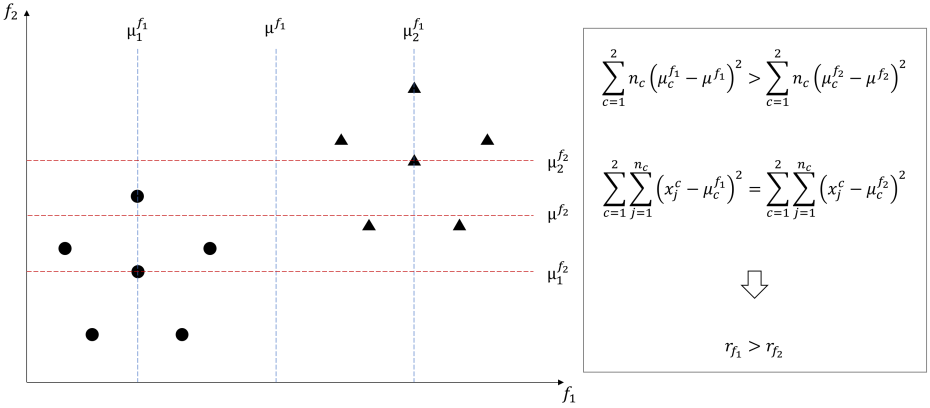

f_2
$\mu_1^{f_1}$
μ^{f_1}
$\mu_2^{f_1}$
$\mu_2^{f_2}$
μ^{f_2}
$\mu_1^{f_2}$
f_1
$\sum_{c=1}^{2} n_c\left(\mu_c^{f_1}-\mu^{f_1}\right)^2 > \sum_{c=1}^{2} n_c\left(\mu_c^{f_2}-\mu^{f_2}\right)^2$
$\sum_{c=1}^{2}\sum_{j=1}^{n_c}\left(x_j^c-\mu_c^{f_1}\right)^2 = \sum_{c=1}^{2}\sum_{j=1}^{n_c}\left(x_j^c-\mu_c^{f_2}\right)^2$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

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.

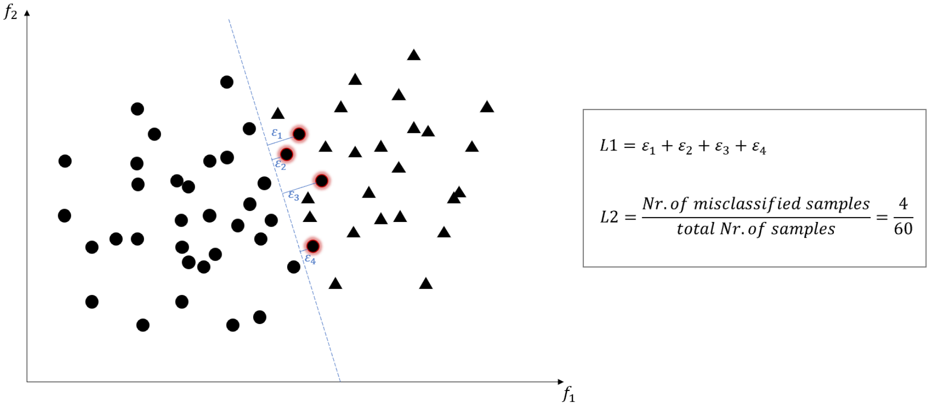

f_2
f_1
ε_1
ε_2
ε_3
ε_4
$L1 = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4$
$L2 = \frac{Nr.of\ misclassified\ samples}{total\ Nr.of\ samples} = \frac{4}{60}$

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 subdomains, 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.

CHAPTER 3

An Empirical Investigation Comparing Several Oversampling Techniques

Many resampling approaches have been developed in the imbalance learning domain, most empirical studies and application work are still based on the “classical" resampling techniques and do not take newly developed resampling techniques into account. In this chapter, we investigate the effectiveness of six oversampling techniques (both “classical" and new ones) and study the relationship between data complexity measures and the choice of oversampling techniques. This chapter is structured as follows. First, Section 3.1 briefly introduces our work in this chapter. Then, in Section 3.2, the research related to our work is presented including the relevant background knowledge on six resampling approaches and data complexity measures. In Section 3.3, the experiments, including introduction on the datasets, cross-validation and experimental setup are introduced. Section 3.3 also contains the results and discussions of our experiments. Further exploration through data from a real-world inspired digital vehicle model is presented in Section 3.4. Section 3.5 concludes the chapter and outlines further research.

3.1 Introduction

The classification problem under class imbalance has caught growing attention from both, academic and industrial field. Due to recent advances, the progress in technical assets for data storage and management as well as in data science enables practitioners from industry and engineering to collect a large amount of data with the purpose of extracting knowledge and acquire hidden insights. An example may be illustrated from the field of computational design optimization where product parameters are modified to generate digital prototypes which performances are evaluated by numerical simulations, or based on equations that express human heuristics and preferences. Here, many parameter variations usually result in valid and producible geometries but in the final steps of the optimization, i.e. in the area where the design parameters converge to a local/global optimum, some geometries are generated which violate given constraints. Under this circumstance, a database would contain a large number of designs which are according to specifications (even if some may be of low performance) and a smaller number of designs which eventually violate pre-defined product requirements. By far, the resampling techniques have proven to be effective in handling imbalanced benchmark datasets (López, Fernández, García, Palade, and Herrera, 2013). However, the empirical study and application work in the imbalanced learning domain are mostly focusing on “classical" resampling techniques like SMOTE, ADASYN, and MWMOTE etc (J. Li, L.-s. Liu, Fong, R. K. Wong, Mohammed, Fiaidhi, Sung, and K. K. Wong, 2017; Luengo, Fernández, García, and Herrera, 2011; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018), although there are many recently developed resampling techniques.

In this chapter, we set up several experiments on 19 benchmark datasets to study the effectiveness of six oversampling techniques (Kong, Rios, Kowalczyk, Menzel, and Bäck, 2020b), including SMOTE, ADASYN, MWMOTE, RACOG, wRACOG and RWO-Sampling. For each data set, we also compute seven data complexity measures to investigate the relationship between data complexity measures and the choice of resampling techniques, since researchers have pointed out that studying the data complexity of imbalanced datasets is of vital importance (Luengo, Fernández, García, and Herrera, 2011) and it may affect the choice of resampling techniques (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018). We also perform experiments on a real-world inspired vehicle dataset. Results of our experiments demonstrate that in most cases oversampling techniques that take into account the minority class distribution (RACOG, wRACOG, RWO-Sampling) perform better and RACOG exhibits the best performance among the six reviewed approaches. Results on our real-world inspired vehicle dataset further validate this conclusion. No obvious relationship between data complexity measures and the choice of resampling techniques is found in our experiments. However, we find that the F1v value, a measure for evaluating the overlap between classes which most researchers ignore (Luengo, Fernández, García, and Herrera, 2011; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018), has a strong negative correlation with the potential after-sampled Area Under curve (AUC) value.

3.2 Related Work

Many effective sampling approaches have been developed in the imbalanced learning domain and the synthetic minority oversampling technique (SMOTE) is the most famous one among all. Currently, more than 90 SMOTE extensions have been published in scientific journals and conferences (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018). Most of the review papers and applications are based on the “classical" resampling techniques and do not take new oversampling techniques into account. In this chapter, we briefly review six oversampling approaches, including both, “classical" ones (SMOTE, ADASYN, MWMOTE) and new ones (RACOG, wRACOG, RWO-Sampling) (Barua, Islam, Yao, and Murase, 2012; Chawla, Bowyer, Hall, and Kegelmeyer, 2002; Das, Krishnan, and Cook, 2014; H. He, Bai, E. A. Garcia, and S. Li, 2008; H. Zhang and M. Li, 2014). The six reviewed oversampling techniques can be divided into two groups according to whether they consider the overall minority class distribution. Among the six approaches, RACOG, wRACOG, and RWO-Sampling take into account the overall minority class distribution while the other three do not. Apart from developing new approaches to solve the class-imbalance problem, various studies have pointed out that it is important to study the characteristics of the imbalanced dataset (López, Fernández, García, Palade, and Herrera, 2013; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018). In (López, Fernández, García, Palade, and Herrera, 2013), authors emphasize the importance of studying the overlap between the two-class samples. In (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018), authors set up several experiments with the KEEL benchmark datasets (Alcalá-Fdez, Fernández, Luengo, Derrac, García, Sánchez, and Herrera, 2011) to study the relationship between various data complexity measures and the potential AUC value. It is also pointed out in (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018) that the distinctive inner procedures of oversampling approaches are suitable for particular characteristics of the data. Hence, apart from evaluating the effectiveness of the six reviewed oversampling approaches, we also aim to investigate the relationship between data complexity measures and the choice of resampling techniques.

3.2.1 Oversampling Techniques

As mentioned above, we investigate six oversampling techniques in two groups: “classical" ones (SMOTE, ADASYN, MWMOTE) and “new" ones (RACOG, wRACOG, RWO-sampling), depending on whether they consider the overall minority class distribution. SMOTE has been introduced in Section 2 and in the following, the remaining five oversampling techniques are introduced.

ADASYN

The adaptive synthetic (ADASYN) sampling technique aims to adaptively generate minority class samples according to their distributions (H. He, Bai, E. A. Garcia, and S. Li, 2008). The samples which are harder to learn are given higher importance and will be oversampled more often in the data generation process. The key point is to determine a weight/sampling importance ($\hat{r}_i$) for each minority class sample. Weight $\hat{r}_i$ of a minority class sample $\boldsymbol{x_i}$ is defined as (H. He, Bai, E. A. Garcia, and S. Li, 2008)

$$\hat{r}_i = \frac{r_i}{\sum_{i=1}^{m_s} r_i}, \quad r_i = \frac{\Delta_i}{K}, \quad i = 1, ..., m_s, \tag{3.1}$$

where m_s is the number of minority class samples, Δ_i is the number of samples in the K Nearest Neighbours (K-NN) of $\boldsymbol{x_i}$ that belong to the majority class. For a specific minority class sample, a higher value of r_i corresponds to a higher difficulty to learn. The number of synthetic samples that will be generated for different minority class samples are proportional to their sampling importance (H. He, Bai, E. A. Garcia, and S. Li, 2008)

$$g_i = \hat{r}_i \cdot G, \tag{3.2}$$

where G is the total number of synthetic minority class samples that need to be produced. Figure 3.1 shows an example of the sampling importance for different minority class samples.

Compared to SMOTE, the only difference in ADASYN oversampling procedure is that more synthetic samples will be generated for harder minority class samples.

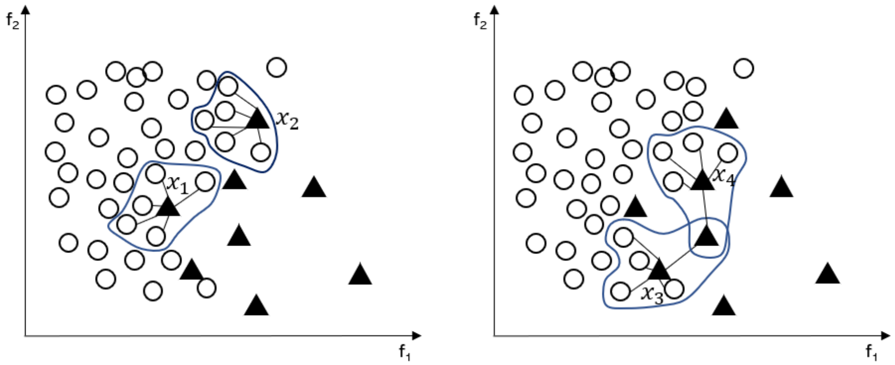


Figure 3.1: Example of sampling importance for different minority class samples. According to definition, $r_1 = r_2 = 1, r_3 = r_4 = 0.8$ and $\hat{r}_1 = \hat{r}_2 > \hat{r}_3 = \hat{r}_4$, indicating the sampling importance of sample $\boldsymbol{x_1}, \boldsymbol{x_2}$ is higher than $\boldsymbol{x_3}, \boldsymbol{x_4}$ and more synthetic samples will be produced for $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$.

In this way, the ADASYN not only provides less learning bias but puts more focus on the difficulty to learn minority class samples.

MWMOTE

Compared to other oversampling techniques, the majority weighted minority oversampling technique (MWMOTE) improves the sample selection scheme and the synthetic sample generation scheme (Barua, Islam, Yao, and Murase, 2012). MWMOTE first finds the informative minority class samples (S_{imin}) by removing the “noise” minority class samples and finding the borderline majority class samples. Then, every sample in S_{imin} is given a selection weight (S_w), according to the distance to the decision boundary, the sparsity of the located minority class cluster and the sparsity of the nearest majority class cluster. These weights are converted into the selection probability (S_p), which will be used in the synthetic sample generation stage. Different from the K-NN-based approach, MWMOTE adopts a clustering algorithm to generate the synthetic samples. The cluster-based synthetic sample generation process proposed in MWMOTE can be described as, 1) cluster all minority class samples into M clusters; 2) select a minority class sample x from S_{imin} according to S_p and randomly select another sample y from the same cluster of x; 3) use the same equation (Eq. 2.1.1) employed in the K-NN-based approach to generate the synthetic sample; 4) repeat 1) – 3) until the required number of synthetic samples is generated.

RACOG and wRACOG

The oversampling approaches can effectively increase the number of minority class samples and achieve a balanced training dataset for classifiers. However, the oversampling approaches introduced above heavily rely on *local* information of the minority class samples and do not take the overall distribution of the minority class into account. Hence, the *global* information of the minority class samples cannot be guaranteed. In order to tackle this problem, Das et al. (Das, Krishnan, and Cook, 2014) proposed RACOG (RApidly COnverging Gibbs) and wRACOG (Wrapper-based RApidly COnverging Gibbs).

In both algorithms, the n-dimensional probability distribution of the minority class is optimally approximated by Chow-Liu's dependence tree algorithm and the synthetic samples are generated from the approximated distribution using Gibbs sampling (Das, Krishnan, and Cook, 2014). The minority class data points are chosen as initial values to start the Gibbs sampler. Instead of running an "exhausting" long Markov chain, the two algorithms produce multiple relatively short Markov chains, each starting with a different minority class sample. RACOG selects the new minority class samples from the Gibbs sampler using a predefined *lag* and this selection procedure does not take the usefulness of the generated samples into account. On the other hand, wRACOG considers the usefulness of the generated samples and selects those samples which have the highest probability of being misclassified by the existing learning model (Das, Krishnan, and Cook, 2014).

RWO-Sampling

Inspired by the central limit theorem, Zhang et al. (H. Zhang and M. Li, 2014) proposed the random walk oversampling (RWO-Sampling) approach to generate the synthetic minority class samples which follows the same distribution as the original training data. Given an imbalanced dataset with multiple attributes, the mean and the standard deviation for the ith attribute a_i ($i \in \{1, 2, 3, ..., m\}$) in minority class data can be calculated and denoted by μ_i and σ_i. Under the central limit theorem, as the number of the minority class samples approaches infinity, the following formula holds:

$$\frac{\mu_i - \mu_i'}{\sigma_i'/\sqrt{n}} \rightarrow N(0,1), \tag{3.3}$$

where μ_i' and σ_i' are the real mean and standard deviation for attribute a_i.

In order to add m synthetic samples to the n original minority class samples, we first select at random m examples from the minority class and then for each of the selected examples $\boldsymbol{x} = (x_1, ..., x_m)$ we generate its synthetic counterpart by replacing $a_i(j)$ (the ith attribute in x_j, $j \in \{1, 2, ..., m\}$) with $\mu_i - r_i \cdot \sigma_i/\sqrt{n}$, where μ_i and σ_i denote the mean and the standard deviation of the ith feature restricted to the original minority class, and r_i is a random value drawn from the standard Gaussian distribution. We can repeat the above process until we reach the required mount of synthetic examples. Since the synthetic sample is achieved by randomly walking from one real sample, this oversampling is called random walk oversampling.

3.2.2 Data Complexity

The motivation for studying the data complexity in imbalanced data is that some researchers (Luengo, Fernández, García, and Herrera, 2011; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018) find no clear relationship between imbalance ratio (IR) and the classification performance obtained via resampling. From their empirical studies, they conclude that IR is not a sufficient measure to identify the potential performance improvement of the data-level approaches (Luengo, Fernández, García, and Herrera, 2011). Therefore, they analyse the resampling techniques through data complexity measures (detailed introduced in Section 2.3) in further studies. One of the main results is that the Fisher discriminant ratio (F1) is informative in characterising the imbalanced classification performance. Following the idea of studying the data complexity in the context of class imbalance, Chen et al. (L. Chen, Fang, Shang, and Tang, 2018) studied the relationship between class overlap and class imbalance in software defect prediction problems. In (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018), authors conduct detailed experiments to study the relationship between data complexity measures and imbalanced classification performance. In their regression analysis across a range of datasets, the obtained regression model can predict the AUC performance based on complexity measures with an average $R^2 = 0.72$.

3.3 Experiments

In this section, we introduce the information on the datasets used in our experiments. Then, the cross-validation in imbalanced learning is described. After that, the experimental setup and results are given.

3.3.1 Information on the Datasets

The experiments reported in this chapter are based on 19 two-class imbalanced datasets from the KEEL-collection (Alcalá-Fdez, Fernández, Luengo, Derrac, García, Sánchez, and Herrera, 2011). The 19 collected binary datasets are manually decomposed from four multi-class datasets: *ecoli*, *glass*, *vehicle* and *yeast*. Detailed information on the datasets are given in Table 3.1 & Table 3.2.

Table 3.1: Information on datasets divided into 4 groups.

Datasets	**#Attributes**	**#Samples**	**Imbalance Ratio (IR)**
ecoli{1,2,3,4}	7	336	{ 3.36, 5.46, 8.6, 15.8 }
glass{0,1,2,4,5,6}	9	214	{ 2.06, 1.82, 11.59, 15.47, 22.78, 6.38 }
vehicle{0,1,2,3}	18	846	{ 3.25, 2.9, 2.88, 2.99 }
yeast{1,3,4,5,6}	8	1484	{ 2.46, 8.1, 28.1, 32.73, 41.4 }

Table 3.2: Further description of the datasets (Alcalá-Fdez, Fernández, Luengo, Derrac, García, Sánchez, and Herrera, 2011)

Datasets	**Description**
ecoli	This is a protein localization sites classification dataset, which contains 8 classes. It is artificially modified into 4 binary datasets, where the sample proportions are { 77:259 ; 52:284 ; 35:301 ; 20:316 }.
glass	This is a glass identification dataset, which contains 6 classes. It is artificially modified into 6 binary datasets, where the sample proportions are { 70:144 ; 76:138 ; 17:197 ; 13:201 ; 9:205 ; 29:185 }.
vehicle	This is a vehicle silhouettes dataset, which contains 4 classes. It is artificially modified into 4 binary datasets, where the sample proportions are { 199:647 ; 217:629 ; 218:628 ; 212:634 }.
yeast	This is a protein localization sites classification dataset, which contains 10 classes. It is artificially modified into 5 binary datasets, where the sample proportions are { 429:1055 ; 163:1321 ; 51:1433 ; 44:1440 ; 35:1449 }.

3.3.2 Cross-Validation in Imbalanced Learning

Cross-validation (CV) is an effective technique to assess classification performance. It allows different portions of the data for training and testing a model (Bishop and Nasrabadi, 2006). In traditional k-fold CV, the original dataset is randomly partitioned into k folds, where k-1 folds are used to train the model and the left one is retained as a validation fold to test the model performance. Then, every fold iterates to be the validation fold to ensure that all folds are used for training and testing the model. After k iterations, the final performance can be estimated by averaging the k results. One significant advantage of this procedure is that the validation fold is unseen in the training process. In the imbalanced learning domain, data-level approaches are commonly used to deal with the imbalance in the datasets. Some researchers emphasize the importance of correctly understanding the joint use of CV and data-level approaches. They point out that a poorly designed CV procedure for imbalanced datasets will result in overfitting and overoptimism problems (Lusa et al., 2015; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018).

According to *Oxford English Dictionary* [1], overfitting is a statistical term with definition “the production of an analysis which corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably". This term is then extended to machine learning, which means the learning model is highly fitted to the training data and, therefore, has poor ability to generalise on unseen data. The CV technique can alleviate the overfitting problem in most cases. However, when learning from imbalanced data, some oversampling techniques produce exact replicas of some samples (Lusa et al., 2015). Too many same patterns in the training set will result in overfitting of the model even with CV technique.

Overoptimism occurs when the training and test sets contain exact or similar replicas of some patterns (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018). For example, suppose we first obtain a balanced dataset through oversampling approaches and then perform cross-validation when dealing with imbalanced datasets. In this way, since the synthetic samples share similar patterns with the original sample, samples with similar patterns may appear in both training and test set, which will lead to the overoptimism problem. In our experiments, we perform k-fold stratified CV before applying the six introduced oversampling techniques.

[1] https://www.oed.com/

The stratified folds ensure that the imbalance ratio in the training set is consistent with the original dataset.

3.3.3 Experimental Setup

In this chapter, six oversampling approaches (using the R package `imbalance` (Cordón, García, Fernández, and Herrera, 2018)), which have been reviewed in Section 3.2.1, are applied to the 19 two-class imbalanced datasets in Table 3.1. Every collected dataset is divided into 5 stratified folds for cross-validation and only the training set is oversampled, where the stratified fold ensures that the imbalance ratio in the training set is consistent with the original dataset and only oversampling the training set avoids the over-optimism problem (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019).

In this chapter, we aim to study the effectiveness of different oversampling approaches and investigate the relationship between data complexity measures and the choice of oversampling techniques. Therefore, we calculate the 7 data complexity measures (Table 2.2) for each dataset. In our 30 experiments for each dataset, we calculate the 7 data complexity measures for every training set using the R package `ECoL` (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019) (Table 3.3). Since we use 5 stratified cross-validations, we average each data complexity measure for these 5 training sets and define it to be the data complexity measure for the dataset.

In a binary classification problem, the confusion matrix can provide intuitive classification results. In the class imbalance domain, it is widely admitted that *Accuracy* tends to result in a deceptive evaluation of the performance. Instead of *Accuracy*, the Area Under the ROC Curve (AUC) and geometric mean (GM) are used to evaluate the performance (details can be checked in Section 2.1).

Table 3.3: Data complexity for 19 collected datasets.

Dataset	**F1**	**F1v**	**F2**	**F3**	**L1**	**L2**	**L3**
ecoli1	0.8785	0.1248	0.0229	0.5814	0.0523	0.0955	0.0586
ecoli2	0.9154	0.1323	0.0000	0.7175	0.0514	0.0806	0.0588
ecoli3	0.9248	0.1557	0.0058	0.4257	0.0516	0.0771	0.0629
ecoli4	0.9291	0.0614	0.0005	0.3584	0.0088	0.0163	0.0152
glass0	0.9525	0.3728	0.0000	0.7002	0.1181	0.2232	0.1873
glass1	0.9808	0.5749	0.0068	0.8896	0.2046	0.3409	0.3378
glass2	0.9913	0.3540	0.0000	0.5279	0.0732	0.0794	0.0778
glass4	0.9497	0.0956	0.0027	0.2784	0.0312	0.0441	0.0378
glass5	0.9753	0.1312	0.0000	0.1402	0.0061	0.0186	0.0154
glass6	0.8373	0.0435	0.0095	0.3775	0.0252	0.0260	0.0185
vehicle0	0.9156	0.0812	0.0001	0.5425	0.0103	0.0261	0.0082
vehicle1	0.9720	0.2606	0.0003	0.9362	0.0929	0.1758	0.1397
vehicle2	0.9735	0.0760	0.0024	0.7702	0.0172	0.0300	0.0142
vehicle3	0.9730	0.3075	0.0006	0.9595	0.1041	0.1818	0.1595
yeast1	0.9638	0.4407	0.0000	0.9587	0.1553	0.2496	0.2418
yeast3	0.9554	0.1343	0.0000	0.4588	0.0433	0.0510	0.0365
yeast4	0.9802	0.2013	0.0000	0.8734	0.0332	0.0344	0.0338
yeast5	0.9580	0.1049	0.0000	0.1139	0.0142	0.0224	0.0182
yeast6	0.9791	0.1468	0.0000	0.6514	0.0225	0.0232	0.0238

3.3.4 Experimental Results and Discussion

The AUC results for C5.0 decision tree and SVM in our experiments are presented in Table 3.4 and Table 3.5. Geometric mean results can be found in Table 3.6 and Table 3.7. In the experimental results of the decision tree, we observe that RACOG outperforms the other 5 oversampling techniques in 8 out of 19 datasets. The same conclusion can also be drawn from the experimental results of SVM. It is worth mentioning that RACOG costs more time than the other five considered oversampling techniques due to the execution of the Markov chain in its data generation process. From our experimental results, we conclude that, in most cases, oversampling approaches which consider the minority class distribution (RACOG, wRACOG and RWO-Sampling) perform better.

It was expected that data complexity can provide some guidance for choosing the oversampling technique, however, from our experimental results, no obvious relationship between data complexity and the choice of oversampling approaches can be concluded. This is because the 6 introduced oversampling approaches are designed for common datasets and do not take a specific data characteristic into account.

According to our experimental results, although the data complexity measures cannot provide guidance for choosing the most promising oversampling approaches, we find that there is a strong correlation between the potential best AUC (after oversampling) and some of the data complexity measures. From Figure 3.2 and Table 3.8, it can be concluded that the potentially best AUC value that can be achieved through C5.0 decision tree and oversampling techniques has an extreme negative correlation with the F1v value and the linearity measures. In the imbalanced learning domain, many researchers focus on studying data complexity measures. In (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019), the authors propose that the potentially best AUC value after resampling can be predicted through various data complexity measures. However, they did not consider the F1v measure, which has the strongest correlation with AUC value according to our findings. Hence, we recommend using F1v to evaluate the overlap in imbalanced datasets.

Table 3.4: AUC results for C5.0 decision tree.

Dataset	Baseline	SMOTE	ADASYN	MWMOTE	RACOG	wRACOG	RWO
ecoli1	0.9408	0.9428	0.9342	0.9414	**0.9453**	0.9384	0.9432
ecoli2	0.8736	**0.9190**	0.9102	0.9112	0.9133	0.8987	0.9143
ecoli3	0.7765	0.9170	0.9013	0.9049	**0.9204**	0.8648	0.9126
ecoli4	0.8403	**0.9271**	0.8832	0.9235	0.9244	0.8896	0.9020
glass0	0.8179	0.8328	0.8254	0.8345	**0.8470**	0.8391	0.8364
glass1	0.6995	0.7391	0.7440	0.7473	**0.7588**	0.7493	0.6944
glass2	0.7309	0.8189	**0.8201**	0.7995	0.8159	0.7960	0.7125
glass4	0.8461	0.9227	0.9203	0.9126	0.9216	0.8542	**0.9252**
glass5	0.9950	0.9927	0.9931	0.9935	0.9940	**0.9952**	0.9932
glass6	0.9341	0.9357	0.9306	0.9385	**0.9388**	0.9386	0.9354
vehicle0	0.9722	0.9730	0.9736	0.9723	0.9737	**0.9739**	0.9679
vehicle1	0.7430	0.7993	0.7916	0.7977	0.7970	**0.8000**	0.7738
vehicle2	0.9735	0.9722	0.9748	0.9757	0.9803	**0.9815**	0.9766
vehicle3	0.7858	0.8001	0.7954	0.8115	**0.8158**	0.8117	0.7907
yeast1	0.7318	0.7380	0.7282	0.7473	**0.7536**	0.6766	0.7279
yeast3	0.9335	0.9594	0.9580	0.9602	**0.9642**	0.9551	0.9422
yeast4	0.7769	**0.9020**	0.8989	0.8884	0.8549	0.8142	0.8367
yeast5	0.9555	0.9769	**0.9773**	**0.9773**	0.9761	0.9688	0.9772
yeast6	0.7307	0.8792	0.8850	0.8789	0.8806	0.7815	**0.8868**

Table 3.5: AUC results for SVM.

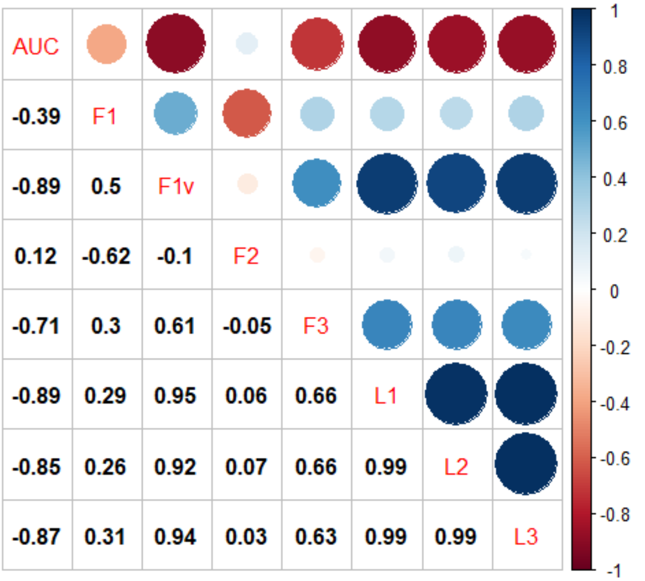
Dataset	Baseline	SMOTE	ADASYN	MWMOTE	RACOG	wRACOG	RWO
ecoli1	**0.9518**	0.9483	0.9435	0.9471	0.9458	0.9513	0.9455
ecoli2	0.9580	0.9563	0.9590	0.9564	0.9958	**0.9602**	0.9552
ecoli3	0.9459	0.9508	0.9462	0.9502	**0.9518**	0.9512	0.9485
ecoli4	**0.9949**	0.9922	0.9905	0.9908	0.9907	0.9948	0.9900
glass0	0.8390	0.8515	0.8475	0.8489	**0.8535**	0.8461	0.8527
glass1	0.7741	0.7765	0.7764	0.7749	**0.7802**	0.7777	0.7770
glass2	0.8206	0.8483	0.8471	0.8414	0.8609	0.8296	**0.8626**
glass4	**0.9863**	0.9855	0.9862	0.9853	0.9836	0.9862	0.9856
glass5	0.9698	**0.9807**	0.9806	0.9797	0.9785	0.9708	0.9776
glass6	0.9800	0.9773	0.9744	0.9739	0.9766	**0.9809**	0.9755
vehicle0	0.9956	**0.9959**	0.9954	0.9948	0.9950	0.9951	0.9906
vehicle1	0.8609	0.8889	0.8886	**0.8913**	0.8822	0.8812	0.8487
vehicle2	0.9952	0.9953	**0.9960**	0.9949	0.9948	0.9955	0.9943
vehicle3	0.8492	**0.8724**	0.8717	0.8709	0.8676	0.8611	0.8492
yeast1	0.7803	0.7874	0.7875	0.7826	**0.7959**	0.7768	0.7911
yeast3	**0.9730**	0.9685	0.9678	0.9689	0.9716	0.9727	0.9686
yeast4	0.8416	0.8843	0.8838	0.8878	**0.8990**	0.8703	0.8853
yeast5	0.9804	0.9867	**0.9871**	0.9868	0.9837	0.9827	0.9865
yeast6	0.8334	0.9264	0.9158	0.9272	**0.9295**	0.8709	0.9191

Table 3.6: Geometric mean results for C5.0 decision tree.

Dataset	**Baseline**	**SMOTE**	**ADASYN**	**MWMOTE**	**RACOG**	**wRACOG**	**RWO**
ecoli1	0.8319	0.8851	0.8769	0.8861	**0.8865**	0.8727	0.8562
ecoli2	0.8519	0.8829	0.8742	0.8784	**0.8854**	0.8701	0.8720
ecoli3	0.7173	**0.8281**	0.8100	0.8173	0.7762	0.7527	0.7458
ecoli4	0.8276	0.8617	0.8415	0.8610	**0.8681**	0.8442	0.8540
glass0	0.7691	0.7799	0.7727	0.7846	**0.7879**	0.7773	0.7829
glass1	0.7082	0.7179	0.7181	0.7193	0.7205	**0.7233**	0.6879
glass2	0.3966	0.6083	**0.6194**	0.5702	0.4938	0.5286	0.4399
glass4	0.6838	**0.8513**	0.8427	0.8344	0.8047	0.6930	0.8388
glass5	0.8868	0.9121	0.9030	0.9087	0.8850	**0.9199**	0.9076
glass6	0.8828	0.9069	0.8853	**0.9078**	0.8969	0.8792	0.8947
vehicle0	0.9158	0.9155	0.9201	0.9240	**0.9249**	0.9215	0.9228
vehicle1	0.6271	0.7104	0.7031	0.7089	0.7054	**0.7119**	0.6475
vehicle2	0.9455	0.9534	0.9587	0.9569	0.9491	0.9509	**0.9596**
vehicle3	0.6439	0.7119	0.7084	0.7113	**0.7121**	0.7059	0.6454
yeast1	0.6335	0.6893	0.6917	0.6925	**0.7024**	0.6461	0.6307
yeast3	0.8668	0.9106	0.9156	0.9067	**0.9184**	0.8959	0.8853
yeast4	0.5011	0.7006	0.6879	**0.7390**	0.6466	0.5725	0.5000
yeast5	0.8394	0.9305	**0.9399**	0.9288	0.9058	0.8669	0.8629
yeast6	0.6224	0.7688	0.7831	**0.7880**	0.7501	0.7076	0.7060

Table 3.7: Geometric mean results for SVM. “—" means that TP+FN=0 or TP+FP=0 and the performance metric cannot be computed.

Dataset	**Baseline**	**SMOTE**	**ADASYN**	**MWMOTE**	**RACOG**	**wRACOG**	**RWO**
ecoli1	0.8292	0.8810	0.8844	0.8782	**0.8845**	0.8622	0.8801
ecoli2	0.7278	**0.9326**	0.9179	0.9324	0.9297	0.7399	0.9306
ecoli3	0.6108	0.8722	0.8668	0.8748	**0.8764**	0.6615	0.8729
ecoli4	0.7132	0.9079	0.8987	0.9017	**0.9191**	0.7158	0.8992
glass0	0.7234	0.7900	**0.7909**	0.7850	0.7866	0.7741	0.7881
glass1	0.6419	0.6908	0.6883	0.6894	**0.6951**	0.6942	0.6861
glass2	—	0.7138	0.7080	0.7207	**0.7592**	—	0.7664
glass4	0.7079	0.8606	0.8692	0.8603	0.8658	0.7181	**0.8776**
glass5	0.0283	0.6663	0.6664	0.6644	0.6899	0.0679	**0.7630**
glass6	0.8374	0.8862	**0.8926**	0.8799	0.8889	0.8459	0.8818
vehicle0	0.9525	**0.9731**	0.9730	0.9682	0.9693	0.9677	0.9599
vehicle1	0.5668	0.8176	**0.8199**	0.8183	0.8073	0.8020	0.6520
vehicle2	0.9621	0.9728	**0.9754**	0.9727	0.9657	0.9687	0.9591
vehicle3	0.5115	0.8017	0.8048	**0.8056**	0.7986	0.7943	0.6347
yeast1	0.5888	0.7123	0.7123	0.7107	**0.7193**	0.6864	0.7162
yeast3	0.8428	0.8978	0.9023	0.8956	**0.9141**	0.8658	0.9020
yeast4	0.0084	0.7484	0.7527	0.7560	**0.8021**	0.3774	0.7525
yeast5	0.6463	0.9255	0.9278	0.9245	0.9342	0.7618	**0.9377**
yeast6	0.3701	0.8257	0.8063	0.8279	**0.8541**	0.5605	0.8310



AUC	●	●	●	●	●	●	●
-0.39	F1	●	●	●	●	●	●
-0.89	0.5	F1v	●	●	●	●	●
0.12	-0.62	-0.1	F2	●	●	●	●
-0.71	0.3	0.61	-0.05	F3	●	●	●
-0.89	0.29	0.95	0.06	0.66	L1	●	●
-0.85	0.26	0.92	0.07	0.66	0.99	L2	●
-0.87	0.31	0.94	0.03	0.63	0.99	0.99	L3

Figure 3.2: Correlation matrix. (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2019).

Table 3.8: Results of Hypothesis Test.

Measure	Correlation Coefficient	P-value	Correlation Level
F1	-0.3872	0.1014	medium
F1v	-0.8928	2.736×10^{-7}	extreme
F2	0.1156	0.6374	none
F3	-0.7138	0.0006	high
L1	-0.8876	4.013×10^{-7}	extreme
L2	-0.8523	3.611×10^{-6}	extreme
L3	-0.8699	1.304×10^{-6}	extreme

3.4 Efficient Oversampling for Engineering Vehicle Mesh Dataset

In this section, we propose the application of the reviewed methods on the quality prediction of geometric computer aided engineering (CAE) models. For some CAE applications, like e.g. aerodynamic performance evaluation, engineers discretize the geometric models using surface meshes (undirected graphs). Each mesh consists of a set of nodes (vertices), and a set of edges connecting the nodes to form faces and volumes (elements). In computer simulations, equations describing the physical phenomena are solved with respect to the vertices allowing to approximate the solution between nodes and calculate performance features of a design, e.g. drag values as aerodynamic design quality. The meshes are generated from an initial geometric representation, e.g. non-uniform rational B-Splines (NURBS) or stereolithography (STL) representations, using numerical algorithms, such as sweep-hull for Delaunay triangulation (Sinclair, 2016), polycube (Livesu, Vining, Sheffer, Gregson, and Scateni, 2013) etc.

In most cases, the quality of the mesh plays an important role concerning the accuracy and fidelity of the results (Knupp, 2008). Engineers use different types of metrics to infer the quality of the mesh, but it is common sense that increasing the number and uniformity of the elements in the mesh improves the accuracy of the simulation results. However, the computational effort associated with meshing is proportional to the target level of refinement. Therefore, a match between accuracy and available computational resources is often required, especially for cases that demand iterative geometric modifications, such as shape optimization.

Shape morphing techniques address this issue by operating on the mesh nodes through a polynomial-based lower-dimensional representation. Such techniques avoid re-meshing of the simulation domain, thus, speeding up the optimization process. Several cases of optimization using morphing techniques are published in the literature (Menzel, Olhofer, and Sendhoff, 2005; Menzel and Sendhoff, 2008; Olhofer, Bihrer, Menzel, Fischer, and Sendhoff, 2009; Sieger, Menzel, and Botsch, 2015). For our experiments, we implemented the free form deformation (FFD) method presented in (Sederberg and Parry, 1986). To prepare design deformations based on FFD, the geometry of interest is embedded in a uniform parallelepiped lattice, where a trivariate Bernstein polynomial maps the position of the control points of the lattice to the nodes of the mesh, as an $IR^3 \to IR^3$ function. Therefore,

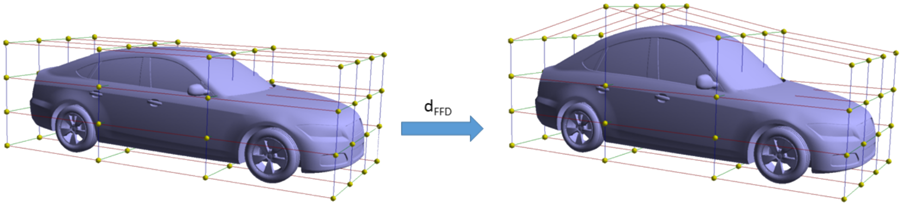


Figure 3.3: Example of free form deformation applied to a configuration of the TUM DrivAer model (Heft, Indinger, and Adams, 2012) using a lattice with four planes in each direction.

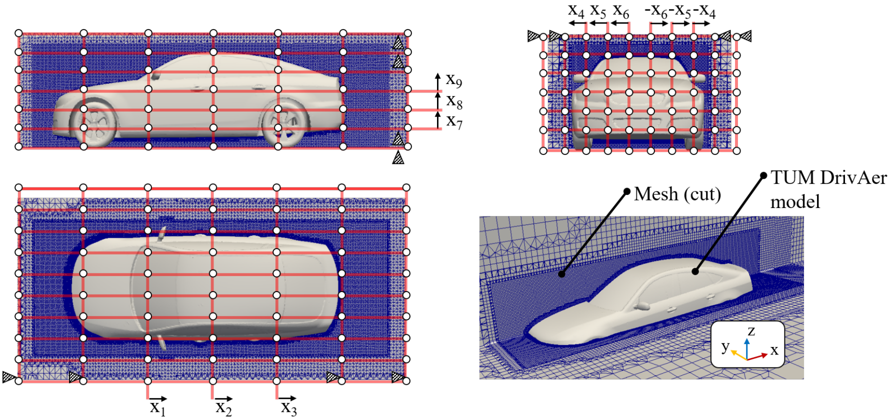


Figure 3.4: Free form deformation lattice used to generate the data set for the experiments.

by deforming the lattice, the nodes of the mesh are moved accordingly (Figure 3.3).

In order to embed the geometry in the lattice, a local coordinate system is defined taking as vector basis the unitary vectors $(\vec{s}, \vec{t}, \vec{u})$, normal to the faces of the parallelepiped and origin in $\mathbf{v_0}$. Then, the coordinates of the mesh nodes are described according to the new basis, using the following linear transformation:

$$\mathbf{v} = \mathbf{v_0} + S\vec{s} + T\vec{t} + U\vec{u} \tag{3.4}$$

where $\mathbf{v}$ is the mesh node described in global coordinate system and the new coordinates S, T and U belong to the interval $[0, 1]$. Given the set that contains the points $\mathbf{p}_{ijk}$ defined by the intersection of the planes that form the lattice, the coordinates of any mesh node can be calculated using the trivariate Bernstein polynomial, defined as

$$\begin{aligned}\mathbf{v_{FFD}} = &\sum_{i=0}^{l}\binom{l}{i}(1-S)^{l-i}S^{i}\left\{\sum_{j=0}^{m}\binom{m}{j}(1-T)^{m-j}T^{j}\right.\\ &\left.\left[\sum_{k=0}^{n}\binom{n}{k}(1-U)^{n-k}U^{k}\mathbf{p}_{ijk}\right]\right\}\end{aligned} \tag{3.5}$$

where $\mathbf{v_{FFD}}$ is the deformed point; l, m, n are respectively the number of control planes in the $\vec{s}$-, $\vec{t}$- and $\vec{u}$-direction.

The continuity of the surfaces is ensured by the mathematical formulation of the FFD up to the order of $k-1$, where k is the number of planes in the direction of interest, but the mesh quality is not necessarily maintained. The designer can either avoid models with ill-defined elements by applying constraints to the deformations, which might be unintuitive, or eliminate them by performing regular quality assessments. Addressing this issue, we propose the classification of the deformation parameters with respect to the quality of the output meshes, based on a data set of labeled meshes. Further than reducing the risk of generating infeasible meshes for CAE applications, our approach avoids unnecessary computation to generate the deformed meshes, which is aligned with the objective of increasing the efficiency of shape optimization tasks.

3.4.1 Generation of a Synthetic Data Set

For the experiments we adopted the computational fluid dynamics (CFD) simulation of a configuration of the TUM DrivAer model (Heft, Indinger, and Adams, 2012). The simulation model is deformed using the discussed FFD algorithm, realized as a lattice with 7 planes in x- and z-directions, and 10 in y-direction (Fig. 3.4). The planes closer to the boundaries of the control volume are not displaced in order to enable a smooth transition from the region affected by the deformations to the original domain. Assuming symmetry of the shape with respect to the vertical plane (xz) and deformations caused by the displacement of entire control planes only in the direction of their normal vectors, it yields a design space with 9 parameters. To generate the data set, the displacements x_i were sampled from a random uniform distribution and constrained to the volume of the lattice, allowing the overlap of planes.

The initial mesh was generated using the algorithms *blockMesh* and *snappyHexMesh* of OpenFOAM®[2]. We automatically generated 994 valid meshes based on the FFD algorithm implemented in Python and evaluated them using the OpenFOAM checkMesh algorithm. The metrics used to define the quality of the meshes were the number of warnings raised by the *checkMesh* algorithm, the maximum skewness and maximum aspect ratio. We manually labeled the feasible meshes according to the rules shown in Table 3.9. The imbalance ratios after manually labelling are also given in Table 3.9. Please note that the input attributes are exactly the same for all three sets of datasets, only the “class" labels are different. In this way, the values of data complexity measures (Table 3.10) for the three datasets vary from each other.

3.4.2 Experimental Results and Discussion

The experimental results on the digital vehicle dataset are given in Table 3.11. In line with our conclusions for the KEEL-dataset experiments (Section 3.3.4), we find that RACOG outperforms the other 5 oversampling techniques in 2 out of 3 datasets. Therefore, combining our experimental results on both benchmark and real-world inspired datasets, we conclude that RACOG performs the best out of the considered 6 oversampling approaches. Moreover, we find that applying the oversampling techniques can improve the performance by around 10% for our digital vehicle datasets. We also calculate the data complexity measures for our digital vehicle datasets and our findings on the correlation between the potential AUC value and the data complexity measures remain consistent with the conclusions in Section.

Table 3.9: Feasible meshes labeling rule.

Dataset	#Attribute	#Sample	#Warnings	Max skewness	Max aspect ratio	IR
set1	9	994	<4	<6	<10	3.76
set2	9	994	<2	<5.8	<10.3	6.83
set3	9	994	<4	<5.6	<10.3	12.43

Table 3.10: Data complexity HRI.

Dataset	F1	F1v	F2	F3	L1	L2	L3
set1	0.9809	0.4360	0.3123	0.9072	0.1737	0.2103	0.2115
set2	0.9950	0.7030	0.1619	0.8900	0.1133	0.1278	0.1325
set3	0.9840	0.2854	0.0962	0.7953	0.0693	0.0744	0.0709

[2] https://www.openfoam.com

Table 3.11: Experimental Results (AUC) on Digital Vehicle Dataset.

Dataset	Baseline	SMOTE	ADASYN	MWMOTE	RACOG	wRACOG	RWO
set1	0.7927	0.8332	0.8279	0.8458	**0.8512**	0.8436	0.8240
set2	0.5864	0.7619	0.7517	0.7590	**0.7633**	0.7437	0.7583
set3	0.6511	0.8215	0.8169	**0.8341**	0.8246	0.8114	0.8065

Table 3.12: Experimental Results (Geometric mean) on Digital Vehicle Dataset.

Dataset	Baseline	SMOTE	ADASYN	MWMOTE	RACOG	wRACOG	RWO
set1	0.6975	0.7557	0.7492	0.7577	**0.7612**	0.7530	0.7295
set2	0.2685	0.6685	0.6622	0.6670	**0.6781**	0.6520	0.6414
set3	0.3373	0.6657	0.6683	**0.6878**	0.6725	0.6573	0.5952

3.5 Conclusions

In this work, we reviewed six oversampling techniques, including “classical" ones (SMOTE, ADASYN and MWMOTE) and new ones (RACOG, wRACOG and RWO-Sampling), in which the new ones consider the minority class distribution while the “classical" ones do not. The six reviewed oversampling approaches were applied to 19 benchmark imbalanced datasets and an imbalanced real-world inspired vehicle dataset to investigate their effectiveness. Seven data complexity measures were considered in order to find the relationship between data complexity measures and the choice of resampling techniques. According to our experimental results, two main conclusions can be derived:

- In our experiment, in most cases, oversampling approaches which consider the minority class distribution (RACOG, wRACOG and RWO-Sampling) perform better. For both benchmark datasets and our real-world inspired dataset, RACOG performs best. However, the trade-off between performance improvement and the time cost should be considered while using RACOG.

- No obvious relationship between data complexity measures and the choice of resampling techniques can be derived from our experimental results. However, we find that the F1v value has a strong correlation with the potential best AUC value (after resampling) while only rarely researchers in the imbalance learning domain consider F1v value for evaluating the overlap between classes.

In this chapter, we applied the oversampling techniques for benchmark datasets and our digital vehicle dataset and evaluated their effectiveness. In the next chapter, we will study hyperparameter optimisation on class-imbalance problems.

CHAPTER 4

Hyperparameter Optimisation on Class-Imbalance Problems

Although the class-imbalance classification problem has caught a huge amount of attention, hyperparameter optimisation has not been studied in detail in this field. Both classification algorithms and resampling techniques involve some hyperparameters that can be tuned. In this chapter, we study hyperparameter optimisation on class-imbalance problems and investigate the relation between the degree of class overlap and the improvement yielded via hyperparameter tuning. This chapter is divided as follows. First, Section 4.1 shows the motivation and provides a brief introduction on our work. After that, in Section 4.2, the resampling techniques used in this chapter and the background knowledge on hyperparameter optimisation are presented. In Section 4.3, the information on the datasets, the experimental setup as well as the experimental results and discussion are introduced. Section 4.4 concludes the chapter and outlines the further work.

4.1 Introduction

Over years of development, many techniques have proven to be efficient in handling imbalanced datasets. These methods can be divided into data-level approaches and algorithmic-level approaches (Bhowan, Johnston, M. Zhang, and Yao, 2012; Ganganwar, 2012; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018), where the data-level approaches aim to produce balanced datasets and the algorithmic-level approaches aim to adjust classical classification algorithms in order to make them appropriate for handling imbalanced datasets.

By far, the most commonly used approach for handling imbalanced data is a combination of resampling techniques and machine learning classification algorithms (López, Fernández, Moreno-Torres, and Herrera, 2012). Research works also focused on these two separate parts, developing new resampling techniques and adjusting machine learning algorithms to be more appropriate for imbalanced datasets. Both resampling techniques and machine learning algorithms involve some hyperparameters that are set to some default values and could be tuned. A minor variation of these hyperparameters might influence the performance significantly. However, hyperparameter optimisation has not been studied yet in detail in the context of learning from imbalanced data, where both components could be tuned simultaneously.

Previous research has considered the hyperparameters for the classifiers for class-imbalance problems (Thai-Nghe, Busche, and Schmidt-Thieme, 2009), but the hyperparameters in resampling techniques are not included. Agrawal et al. (Agrawal and Menzies, 2018) take the hyperparameters in SMOTE into account and propose an auto-tuning version of SMOTE. In this chapter, we explore the potential of applying hyperparameter optimisation for the automatic construction of high-quality classifiers for imbalanced data. In our research, we experiment with a small collection of imbalanced datasets and two classification algorithms: Random Forest and SVM. In each experiment we consider six scenarios for hyperparameter optimisation (see Table 4.1). For classification algorithms, we consider two conditions, algorithms with default hyperparameters (A_d) and algorithms with optimised hyperparameters (A_o). For resampling approaches, we consider three conditions, no resampling applied (R_n), resampling applied with default hyperparameters (R_d) and resampling applied with optimised hyperparameters (R_o).

Table 4.1: Six scenarios in our experiments.

Scenario	**Classification Algorithms**	**Resampling Approaches**
(1) $A_d + R_n$	Default hyperparameters	No
(2) $A_o + R_n$	Optimised hyperparameters	No
(3) $A_d + R_d$	Default hyperparameters	Default hyperparameters
(4) $A_o + R_d$	Optimised hyperparameters	Default hyperparameters
(5) $A_d + R_o$	Default hyperparameters	Optimised hyperparameters
(6) $A_o + R_o$	Optimised hyperparameters	Optimised hyperparameters

Apart from developing new techniques to deal with imbalanced datasets, the data complexity in the dataset itself has caught an increasing attention in recent studies of class-imbalance problems. As we stated in Chapter 3.2.2, it has been shown that the degradation of machine learning algorithms for imbalanced datasets is not directly caused by class imbalance, but is also related to the degree of class overlapping (Prati, Batista, and Monard, 2004), and the classification algorithms are more sensitive to noise than to class imbalance (López, Fernández, García, Palade, and Herrera, 2013). It is also concluded that data complexity may influence the choice of resampling methods (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018). Hence, in this chapter, we consider the hyperparameter optimisation for both resampling techniques and classification algorithms. Furthermore, the relation between the degree of class overlap and the improvement achieved via hyperparameter tuning is investigated.

The results of our experiments demonstrate that an improvement can be obtained by applying hyperparameter tuning. In the six scenarios, optimising the hyperparameters for both classification algorithms and resampling approaches gives the best performance for all six datasets. Further study shows that the data complexity of the original data, especially the overlap between classes, influences whether a significant improvement can be achieved through hyperparameter optimisation. Compared to imbalanced datasets with high class overlap, hyperparameter optimisation works more efficiently for imbalanced datasets with low class overlap. In addition, we point out that resampling techniques are not effective for all datasets, and their effectiveness is also affected by data complexity in the original datasets. Hence, we recommend studying the data complexity of imbalanced datasets before resampling the samples and optimising the hyperparameters. Our work in this chapter has received more than 20 citations from other researchers till the end of 2022, which indicates our contributions to this topic.

4.2 Related Works

This section first introduces the resampling techniques used in this chapter. Then, the definition of hyperparameter optimisation and the related literature in the class-imbalance domain are given in Section 4.2.2.

4.2.1 Resampling Techniques

This section describes four resampling techniques in our experiments, two oversampling and two hybrid approaches. The two oversampling techniques, SMOTE and ADASYN, have been introduced in detail in the previous chapters. Therefore, we only provide details on the two hybrid approaches, SMOTETL and SMOTEENN.

SMOTETL

In a classification problem, a Tomek link is defined as follows (Tomek, 1976): given two samples $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ from different classes, $d(\boldsymbol{x_i}, \boldsymbol{x_j})$ the distance between $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$, and $\boldsymbol{x_l}$ is a random sample in the dataset. The pair $(\boldsymbol{x_i}, \boldsymbol{x_j})$ is defined as a Tomek link if the following requirements hold,

$$\forall \boldsymbol{x_l},\ d(\boldsymbol{x_i}, \boldsymbol{x_j}) < d(\boldsymbol{x_i}, \boldsymbol{x_l}) \text{ and } d(\boldsymbol{x_i}, \boldsymbol{x_j}) < d(\boldsymbol{x_j}, \boldsymbol{x_l}). \tag{4.1}$$

From the definition, a Tomek link is a pair of samples from different classes that are the nearest neighbours for each other, and the samples in Tomek links are either noise or borderline (Batista, Prati, and Monard, 2004).

Oversampling techniques aim to balance the class distribution via expanding the minority class space. However, some synthetic minority class samples may invade the majority class space, making the decision boundary blur. To alleviate this problem, Batista et al. (Batista, Prati, and Monard, 2004) proposed to apply Tomek links as an additional data cleaning method after SMOTE, and named the new technique SMOTETL. In the SMOTETL technique, the first step is (1) to oversample the minority classes using SMOTE and then (2) to identify the Tomek links. After that, (3) the Tomek links for the oversampled samples are removed. In this way, the SMOTETL technique provides a more clear decision boundary by removing part of the samples in the overlapping region. Figure 4.1 gives an example of clearing Tomek links for oversampled samples.

SMOTEENN

Similar to SMOTETL, SMOTEENN is also a hybrid method that combines oversampling and data cleaning techniques. SMOTEENN uses Wilson's Edited Nearest Neighbours (ENN) (D. L. Wilson, 1972) to remove any sample that has a different class from at least two of its three nearest neighbours (Lorena, L. P.

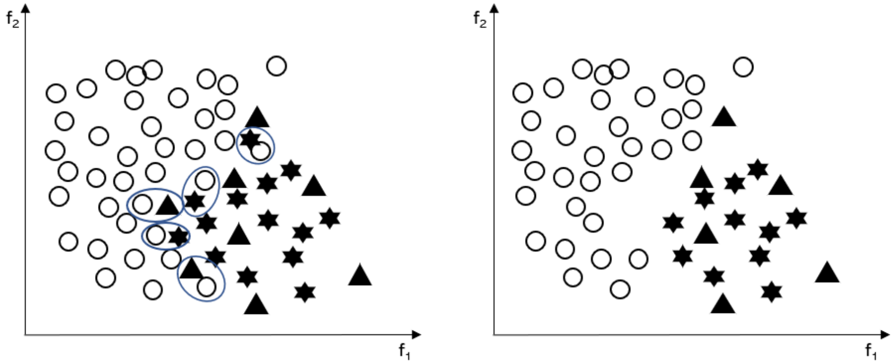


Figure 4.1: Example of clearing Tomek links for oversampled samples. ○ and the black △ indicate the majority and minority class samples respectively. The star indicates the synthetic samples and each blue circle indicates a Tomek link.

Garcia, Lehmann, Souto, and Ho, 2018). For a binary class-imbalance problem, SMOTEENN is implemented as follows: (1) the training set is oversampled via SMOTE, then (2) for each sample in the training set, its three nearest neighbours are found. After that, (3) any sample whose label contradicts the label of at least two of its three nearest neighbours is removed. According to the ENN procedure, more samples are removed than the Tomek links, i.e. ENN provides a deeper data cleaning (Lorena, L. P. Garcia, Lehmann, Souto, and Ho, 2018).

4.2.2 Hyperparameter Optimisation

Most machine learning algorithms involve several hyperparameters, which have to be set before the training process. Compared with randomly selecting the hyperparameters in a learning algorithm, choosing a set of optimal hyperparameters can improve the performance of the algorithm. For instance, in Random Forest, the choice of the depth of a decision tree and the number of trees in a forest will have an influence on the performance. To determine the optimal combination of hyperparameters for a given problem/dataset naturally leads to the well-established hyperparameter optimisation (or hyperparameter tuning) task.

Let $\mathcal{A}$ denote a typical machine learning algorithm with n hyperparameters, $\boldsymbol{\lambda}$ denote a vector of hyperparameters and $\boldsymbol{\Lambda}$ denote the hyperparameter configuration space, i.e. $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$. A learning algorithm with hyperparameters $\boldsymbol{\lambda}$ is represented by

$\mathcal{A}_{\boldsymbol{\lambda}}$ (Feurer and Hutter, 2019). Given a dataset $\boldsymbol{X}$, the goal to find the optimal set of hyperparameters $\boldsymbol{\lambda}^*$ so as to minimize the predefined loss function $\mathcal{L}(\cdot)$ can be represented by (Bergstra, Bardenet, Bengio, and Kégl, 2011; Claesen and De Moor, 2015)

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda} \in \boldsymbol{\Lambda}} \mathcal{L}(\boldsymbol{X}^{(te)}; \mathcal{A}_{\boldsymbol{\lambda}}(\boldsymbol{X}^{(tr)}), \tag{4.2}$$

where $\boldsymbol{X}^{(tr)}$ and $\boldsymbol{X}^{(te)}$ are the training set and validation set, which are given.

There are many approaches for performing hyperparameter optimisation. *Grid search* is a traditional way of tuning hyperparameters. It starts with dividing the search space into a discrete grid. Then, grid search performs an exhaustive search on every combination of the hyperparameters, which always requires much time. *Random search* is similar to grid search but replaces the exhaustive searching on every combination with randomly selecting the combinations to test. *Bayesian hyperparameter optimisation* approaches provide a less expensive way to optimise the hyperparameters. Its strategy keeps tracking previously evaluated results and uses the obtained information to form a surrogate probabilistic model of the objective function (Bergstra, Bardenet, Bengio, and Kégl, 2011; Bergstra, Yamins, and Cox, 2013). The hyperparameters for evaluation by the objective function are selected by applying a criterion to the surrogate function, and this criterion is defined by a selection function, e.g. Expected Improvement. The optimisation procedure is described below.

- Form a surrogate probabilistic model of the objective function;
- Optimise the selection function over the surrogate model;
- Find the hyperparameter values which maximise the Expected Improvement;
- Evaluate these hyperparameters on the objective function;
- Update the surrogate according to the new performance;
- Iterate the 2nd - 5th step until time or other constraint is met.

Compared to the original objective function, the surrogate model is less expensive to optimise because it chooses the next candidate hyperparameters worth evaluating instead of wasting time on unworthy hyperparameters. In practice, there are many software packages based on Bayesian hyperparameter optimisation, e.g. Spearmint, SMAC, HyperOpt, SPOT, etc. In this chapter, a python library[1], HyperOpt (Bergstra, Komer, Eliasmith, Yamins, and Cox, 2015), is used to perform the hyperparameter optimisation for classification algorithms.

In the field of imbalanced learning, the most basic methods are combining the resampling techniques and machine learning classification algorithms; both involve some hyperparameters that could be tuned. Hyperparameters in classifiers are widely considered in classification tasks and this is also true in the imbalanced learning domain. For example, (Thai-Nghe, Busche, and Schmidt-Thieme, 2009) searches the best hyperparameters for their classifiers when improving academic performance prediction by dealing with class imbalance. In (Shekar and Dagnew, 2019), researchers perform a grid search-based hyperparameter tuning on Random Forest classifier when their imbalanced microarray cancer data. Some studies also take the hyperparameters in resampling techniques into account. In (Douzas, Bacao, and F. Last, 2018), authors tune the k nearest neighbours in SMOTE-related resampling techniques. A representative study on hyperparameters in oversampling techniques is (Agrawal and Menzies, 2018), They take the hyperparameters in SMOTE into account and propose SMOTUNED, an auto-tuning version of SMOTE. In their experiments, SMOTUNED improved the performance dramatically, e.g. improvements in AUC up to 60% compared to SMOTE. In this chapter, we perform a detailed study on hyperparameter optimisation for class imbalance problems, i.e. considering six combinations of hyperparameters in both classification algorithm and resampling techniques (see Table 4.1 in Section 4.1).

4.3 Experiments

In this section, we introduce the information on the datasets used in our experiments. Then, the experimental setup is described. After that, the experimental results and discussions are given.

4.3.1 Information on the Datasets

The experiments reported in this chapter are based on six imbalanced datasets from the KEEL-collection (Alcalá-Fdez, Sánchez, S. Garcia, Jesus, Ventura, Garrell, Otero, Romero, Bacardit, Rivas, et al., 2009). Detailed information on the datasets are

[1] available at: http://hyperopt.github.io/hyperopt/

shown in Table 4.2. The overlap between classes is calculated by the Directional-vector Maximum Fisher's Discriminant Ratio (*F1v*). Lower *F1v* value indicates higher overlap between classes (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018).

Table 4.2: Information on the datasets.

Dataset	#Attributes	#Examples	#Classes	IR	F1v value
glass1	9	214	2	1.82	0.57
glass6	9	214	2	6.38	0.04
yeast3	8	1484	2	8.1	0.13
yeast4	8	1484	2	28.1	0.20
ecoli3	7	336	2	8.6	0.16
abalone19	8	4174	2	129.44	0.31

4.3.2 Experimental Setup

As mentioned in Section 4.1, we experiment with six imbalanced datasets, two algorithms and four resampling techniques. Thus, in our experiment, we have $6 \cdot 2 \cdot 5 = 60$ settings tested on each data set, with 6 scenarios, 2 classifiers, and 5 resampling approaches (including none). My co-authored work (D. A. Nguyen, Kong, H. Wang, Menzel, Sendhoff, Kononova, and Bäck, 2021) studies hyperparameter optimisation on class-imbalance problems more extensively, it includes experiments with more imbalanced datasets.

The hyperparameter optimisation for the classification algorithm is done through HyperOpt. Hyperparameters in resampling approaches includes the number of neighbours, imbalance ratio after resampling and etc. In our experiment, hyperparameter optimisation for resampling approaches is done through grid search. Whenever we optimise hyperparameters with "HyperOpt", the AUC loss (1-AUC) is set as the objective function to minimise and the number of iterations is set to 500. For each experiment, we repeated 30 times with different random seeds. After that, the paired t-tests were performed on each 30 AUC values to test if there is significant difference between the results of each scenario on a 5% significance level.

4.3.3 Experimental Results and Discussions

The experimental results are presented in Table 4.3 to investigate the importance of hyperparameter optimisation for imbalanced datasets. For all the six datasets in our experiment, we observe that optimising the hyperparameters for both classifiers and resampling approaches gives the best performance. The statistical hypothesis tests mentioned in Section 4.3.2 are performed on the AUC values of scenario $(A_d + R_d)$ and $(A_o + R_o)$. The test results indicate that there is enough statistical evidence showing the performance improvements are significant for datasets “glass1", “yeast4" and “abalone19". In other words, applying the hyperparameter optimisation does not bring significant improvement for datasets “glass6", “yeast3" and “ecoli3". This experimental result demonstrates that significant improvement can be achieved by performing hyperparameter optimisation for datasets with high *F1v* values. That is to say, hyperparameter optimisation works efficiently for datasets with low overlap between classes.

Furthermore, comparing the AUC values of scenario $(A_d + R_n)$ and $(A_d + R_d)$, for datasets “glass6", “yeast3" and “ecoli3, resampling techniques does not improve the classification performance. Thus, we can conclude that oversampling techniques are not effective for datasets with high overlap. The generated synthetic samples might bring additional noise and make the class overlap even higher. Another point worth mentioning is that, compared to datasets with high overlap, we expected the classification algorithms would perform better on datasets with low overlap. However, the experimental results are contrary to our presupposition. This is because the complexity of a classification problem is not only determined by the overlap between classes but also related to other types of complexity, such as linearity measures.

In the end, we can also observe that there is no specific combination of classifiers and resampling techniques that can provide the best performance for all datasets. For a given dataset, the best combination of classifiers and resampling approaches might depend on the data complexity itself.

4.4 Conclusions and Future Work

In this chapter we considered six scenarios of hyperparameter optimisation for classification algorithms and resampling approaches. Two main conclusions can be derived according to our experimental results:

Table 4.3: Experimental results (AUC) for two classification algorithms regarding six scenarios.
The grey shade and no shade indicate the experimental results for SVM and Random Forest respectively.
p-values indicate the statistical evidence of t-tests between experimental results of scenario $(A_o + R_o)$ and $(A_d + R_d)$.
Dataset with * indicates the results of scenario $(A_o + R_o)$ is significantly higher than results of scenario $(A_d + R_d)$.

		Resampling Approaches (SVM *vs.* Random Forest)									
Scenarios	Dataset	NONE		SMOTE		ADASYN		SMOTETL		SMOTEENN	
$A_d + R_n$	glass1*	0.6753	0.8301	—	—	—	—	—	—	—	—
$A_o + R_n$		0.8309	0.8345	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.7165	0.8401	0.7253	0.8456	0.7416	0.8420	0.7484	0.8126
$A_o + R_d$		—	—	0.8360	0.8537	0.8390	0.8527	0.8423	0.8479	0.8435	0.8278
$A_d + R_o$		—	—	0.7322	0.8599	0.7370	0.8498	0.7437	0.8463	0.7518	0.8216
$A_o + R_o$		—	—	0.8508	0.8649	0.8592	0.8631	0.8659	0.8527	0.8673	0.8379
p-value		—	—	$\ll 0.05$	0.0060	$\ll 0.05$	0.0133	$\ll 0.05$	0.0022	$\ll 0.05$	0.0100
$A_d + R_n$	glass6	0.9768	0.9884	—	—	—	—	—	—	—	—
$A_o + R_n$		0.9848	0.9892	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9749	0.9862	0.9727	0.9849	0.9768	0.9880	0.9761	0.9870
$A_o + R_d$		—	—	0.9807	0.9893	0.9787	0.9877	0.9832	0.9886	0.9840	0.9883
$A_d + R_o$		—	—	0.9796	0.9888	0.9744	0.9870	0.9805	0.9896	0.9795	0.9905
$A_o + R_o$		—	—	0.9850	0.9897	0.9833	0.9883	0.9861	0.9917	0.9857	0.9910
p-value		—	—	0.0693	0.1633	0.1819	0.1166	0.3067	0.1513	0.0603	0.1279
$A_d + R_n$	yeast3	0.9688	0.9624	—	—	—	—	—	—	—	—
$A_o + R_n$		0.9712	0.9700	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9642	0.9662	0.9601	0.9670	0.9659	0.9653	0.9649	0.9693
$A_o + R_d$		—	—	0.9663	0.9731	0.9655	0.9727	0.9701	0.9669	0.9689	0.9743
$A_d + R_o$		—	—	0.9671	0.9693	0.9628	0.9696	0.9684	0.9705	0.9668	0.9722
$A_o + R_o$		—	—	0.9704	0.9759	0.9683	0.9756	0.9733	0.9742	0.9716	0.9787
p-value		—	—	0.3890	0.1529	0.1256	0.0567	0.6166	0.0585	0.2084	0.0573

Table 4.4: Experimental results (AUC) for two classification algorithms regarding six scenarios.
The grey shade and no shade indicate the experimental results for SVM and Random Forest respectively.
p-values indicate the statistical evidence of t-tests between experimental results of scenario ($A_o + R_o$) and ($A_d + R_d$).
Dataset with * indicates the results of scenario ($A_o + R_o$) is significantly higher than results of scenario ($A_d + R_d$).

		Resampling Approaches (SVM *vs.* Random Forest)									
Scenarios	Dataset	NONE		SMOTE		ADASYN		SMOTETL		SMOTEENN	
$A_d + R_n$	yeast4*	0.8479	0.9211	—	—	—	—	—	—	—	—
$A_o + R_n$		0.8739	0.9389	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9025	0.9165	0.8998	0.9123	0.9019	0.9257	0.9079	0.9237
$A_o + R_d$		—	—	0.9132	0.9300	0.9076	0.9293	0.9089	0.9312	0.9093	0.9327
$A_d + R_o$		—	—	0.9098	0.9345	0.9059	0.9319	0.9102	0.9327	0.9122	0.9291
$A_o + R_o$		—	—	0.9178	0.9393	0.9105	0.9346	0.9147	0.9389	0.9201	0.9364
p-value		—	—	$\ll 0.05$	0.0075	0.0133	0.0013	0.0061	0.0036	0.0385	0.0355
$A_d + R_n$	ecoli3	0.9540	0.9359	—	—	—	—	—	—	—	—
$A_o + R_n$		0.9551	0.9535	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9528	0.9310	0.9505	0.9303	0.9508	0.9300	0.9514	0.9329
$A_o + R_d$		—	—	0.9559	0.9338	0.9519	0.9395	0.9549	0.9384	0.9529	0.9385
$A_d + R_o$		—	—	0.9562	0.9419	0.9528	0.9396	0.9569	0.9417	0.9571	0.9416
$A_o + R_o$		—	—	0.9581	0.9432	0.9543	0.9407	0.9573	0.9444	0.9598	0.9450
p-value		—	—	0.4507	0.1337	0.3408	0.1532	0.4436	0.0773	0.3596	0.0575
$A_d + R_n$	abalone19*	0.7373	0.7239	—	—	—	—	—	—	—	—
$A_o + R_n$		0.7687	0.8077	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.8051	0.7934	0.8053	0.7971	0.8051	0.7946	0.8060	0.8034
$A_o + R_d$		—	—	0.8478	0.8328	0.8484	0.8347	0.8473	0.8331	0.8496	0.8395
$A_d + R_o$		—	—	0.8088	0.8095	0.8097	0.8023	0.8089	0.8077	0.8108	0.8090
$A_o + R_o$		—	—	0.8494	0.8389	0.8503	0.8402	0.8488	0.8391	0.8511	0.8414
p-value		—	—	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$

1. In our experiment, the results of scenario ($A_o + R_o$) outperform the other five scenarios. Especially for imbalanced datasets with low class overlap, applying hyperparameter optimisation for both classification algorithms and resampling approaches can significantly improve the performance. Nevertheless, the time consumption caused by hyperparameter optimisation is not negligible. Therefore, we recommend studying the data complexity and considering the trade-off between time cost and potential improvement before optimising the hyperparameters.

2. Based on our experimental results, we find oversampling techniques does not give performance improvement for imbalanced datasets with high class overlap. This further emphasizes the importance of learning the data complexity before dealing with the imbalanced datasets.

In future work, more data complexity measures will be considered in order to study the relation between hyperparameter optimisation and data complexity in detail. Additionally, more attention should be put on developing techniques which can efficiently handle complex imbalanced datasets. Finally, we observe the best choice of classifiers and oversampling techniques depends on the dataset itself. Therefore, another study worth exploring would be to produce a semi-automatic approach which can help choosing the best combination of resampling approaches, machine learning algorithms and hyperparameter optimisation strategies.

CHAPTER 5

Improving Imbalanced Classification via Adding Additional Attributes

The anomaly detection problem can be considered as an extreme case of class imbalance problem, however, very few studies consider improving class imbalance classification with anomaly detection ideas. Most data-level approaches in the imbalanced learning domain aim to introduce more information to the original dataset by generating synthetic samples. In this chapter, we introduce our proposed idea on improving imbalanced classification via adding additional attributes. First, Section 5.1 shows the motivation and provides a brief introduction on our work. After that, in Section 5.2, the background knowledge on anomaly detection and four types of samples in imbalanced datasets are presented. In Section 5.3, the information on the datasets, the experimental setup as well as the experimental results and discussion are introduced. Section 5.4 concludes the chapter and outlines the further work.

5.1 Introduction

The imbalanced classification problem has caught growing attention from many fields. In the field of computational design optimization, product parameters are modified to generate digital prototypes and the performances are usually evaluated by numerical simulations which often require minutes to hours of computation time. Here, some parameter variations (minority number of designs) would result in valid and producible geometries but violate given constraints in the final steps of the optimization. Under this circumstance, performing proper imbalanced classification algorithms on the design parameters could save computation time. In the imbalanced learning domain, many techniques have proven to be efficient in handling imbalanced datasets, including resampling techniques and algorithmic-level approaches (Ganganwar, 2012; Kong, Kowalczyk, D. A. Nguyen, Bäck, and Menzel, 2019; M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018), where the former aims to produce balanced datasets and the latter aims to make classical classification algorithms appropriate for handling imbalanced datasets. The resampling techniques are standard techniques in imbalance learning since they are simple and easily configurable and can be used in synergy with other learning algorithms (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018). The main idea of most oversampling approaches is to introduce more information to the original dataset by creating synthetic samples. However, very few studies consider the idea of introducing additional attributes to the imbalanced dataset.

The anomaly detection problem can be considered as an extreme case of the class imbalance problem. In this chapter, we propose to improve the imbalanced classification with some anomaly detection techniques. We propose to introduce the outlier score, which is an important indicator to evaluate whether a sample is an outlier (Breunig, Kriegel, R. T. Ng, and Sander, 2000), as an additional attribute of the original imbalanced datasets. Apart from this, we also introduce the four types of samples (safe, borderline, rare and outlier), which have been emphasized in many studies (Napierala and Stefanowski, 2016; Skryjomski and Krawczyk, 2017), as another additional attribute. The paper contributed to this chapter has been published in *Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part I 2020 Aug 31 (pp. 512-523)*, titled "*Improving imbalanced classification by anomaly detection*". In our experiments, we consider four scenarios, i.e. four different combinations using the additional attributes and performing resampling techniques. The results of our experiments demonstrate that introducing the two proposed additional attributes can improve the imbalanced classification performance in most cases. Further study shows that this performance improvement is mainly contributed by a more accurate classification in the overlapping region of the two classes (majority and minority classes).

5.2 Related Works

This section first introduces the resampling techniques used in this chapter. Then, the anomaly detection problem is introduced.

5.2.1 Resampling Techniques

This work is based on five resampling techniques in our experiments, one oversampling, two undersampling and two hybrid approaches. The oversampling technique SMOTE and the two hybrid approaches, SMOTETL and SMOTEENN; have been introduced in detail in the previous chapters. Therefore, we only provide details on the two undersampling approaches, OSS and NCL.

OSS

One-Sided Selection (OSS) (Kubat, Matwin, et al., 1997) is an undersampling technique which combines Tomek Links and the Condensed Nearest Neighbour (CNN) Rule. Detailed information on Tomek Links is given in 4.2.1. CNN was first introduced by Hart in 1968 (Hart, 1968) together with the concept of a consistent subset. By definition, a subset $\hat{E}$ is consistent with E ($\hat{E} \subseteq E$), if the 1-NN rule (K-NN rule, where $K = 1$) built with samples in $\hat{E}$ can correctly classify samples in E. In OSS, the following three groups of samples are removed (Kubat, Matwin, et al., 1997):

- Majority class samples which suffer from class-label noise.
- Majority class samples which are close to the decision boundary. They are susceptible to variations, and even a tiny variation in training data or classification model can make them fall on the wrong side of the decision boundary.
- Majority class samples which have limited contribution for building the decision boundary. Although they are harmless but they increase the classification costs.

The first two groups of samples are removed with so-called Tomek links. The third group of samples are removed with CNN. The remainder of the majority class samples and all the minority class samples are used to construct the classifiers. Algorithm 1 summarizes the OSS procedure.

Algorithm 1: One-Side Selection (OSS) (Kubat, Matwin, et al., 1997)

```
Input  : $S$ - Original training set
Output : $T$ - Undersampled training set
Select a subset $C$ ($C \subseteq S$), which contains all minority class samples and
  one randomly selected majority class sample;
Classify $S$ using the 1-NN rule built with $C$. Add all misclassified samples in
  $S$ to subset $C$ and now $C$ is a consistent subset of $S$;
Remove from $C$ all majority class samples belonging to Tomek links. The
  remaining set is referred to as $T$.
```

NCL

Neighbourhood Cleaning Rule (NCL) (Laurikkala, 2001) emphasizes the quality of the retained samples after data cleaning and can be used for multi-class problems. Suppose C are the classes of interest, and the rest of the data are referred as R. The cleaning process is first performed by removing any ambiguous sample in R whose label differs from the class of at least two of its three neighbours through the Wilson's Edited Nearest Neighbours (ENN, introduced in 4.2.1) (D. L. Wilson, 1972). In addition, NCL performs a deeper cleaning in the neighbourhoods of samples in C. For a sample in C, if its label differs from the classification given by its three nearest neighbours, the neighbours belonging to R are removed. In this step, special considerations are paid to small-size classes (details in Algorithm 2). In the binary scenario, NCL can be described as follows: if a majority class sample has a different label from the classification given by its three nearest neighbours, this majority class sample is removed. Additionally, if the label of a minority class sample contradicts the classification given by its three nearest neighbours, then the neighbours belonging to the majority class are removed.

5.2.2 Anomaly Detection

Anomaly detection, also referred to as outlier detection, is the process of identifying irregular patterns in the datasets (Chandola, Banerjee, and Kumar, 2009). The behaviours of these patterns deviate significantly from the majority of the data. Such examples can be found in various applications, including fraud detection in credit cards, medical diagnosis in health care, quality control in the manufacturing field, etc.

Many algorithms have been developed to deal with anomaly detection problems

Algorithm 2: Neighbourhood Cleaning Rule (NCL) (Laurikkala, 2001)

```
Input  : $S$ - Original training set
Output : $T$ - Undersampled training set
Split training set $S$ into the classes of interest $C$ and the rest $R$;
Identify the noisy data $D_1$ in $R$ with ENN;
Identify the samples in $C$ which are misclassified by their 3 nearest
  neighbours and referred to as $C_m$;
for each class $R_i \in R$ do
    if $x \in Ri \cap C_m$ and $|R_i| \geqslant \frac{1}{2} \times |C|$ then
        move $x$ into $D_1$;
end
Remove $D_1$ from $S$ and the undersampled training set is $T = S - D_1$.
```

and the experiments in this chapter are mainly performed with the nearest-neighbour based local outlier score (LOF). Local outlier factor (LOF), which indicates the degree of a sample being an outlier, was first introduced in (Breunig, Kriegel, R. T. Ng, and Sander, 2000). The LOF of an object depends on its relative degree of isolation from its surrounding neighbours. Several definitions are needed to calculate the LOF and are summarized in the following Algorithm 3.

According to the definition of LOF, a value of approximately 1 indicates that the local density of data point x_i is similar to its neighbours. A value below 1 indicates that data point x_i locates in a relatively denser area and does not seem to be an anomaly, while a value significantly larger than 1 indicates that data point x_i is alienated from other points, which is most likely an outlier.

5.2.3 Four Types of Samples in Imbalanced Datasets

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 idea has been introduced in detail in Section 2.3.2.

Algorithm 3: Local Outlier Factor (LOF) algorithm (Breunig, Kriegel, R. T. Ng, and Sander, 2000)

```
Input  : $\mathbf{x}$ - input data $\mathbf{x} = (x_1, ..., x_n)$
         $n$ - the number of input examples
         $k$ - the number of neighbours
Output : LOF score of every $x_i$
initialization;
calculate the distance $d(\cdot)$ between every two data points;
for $i = 1$ to $n$ do
    calculate $k\text{-}distance(x_i)$: the distance between $x_i$ and its $k$th neighbour;
    find out $k$-distance neighbourhood $N_k(x_i)$: the set of data points whose
      distance from $x_i$ is not greater than $k\text{-}distance(x_i)$;
    for $j = 1$ to $n$ do
        calculate reachability distance:
            $$reach\text{-}dist_k(x_i, x_j) = \max\{k\text{-}distance(x_j), d(x_i, x_j)\};$$
        calculate local reachability density:
            $$\begin{aligned} lrd_k(x_i) &= 1/avg\text{-}reach\text{-}dist_k(x_i) \\ &= 1/\left(\frac{\sum_{o \in N_k(x_i)} reach\text{-}dist_k(x_i, x_j)}{|N_k(x_i)|}\right); \end{aligned}$$
          intuitively, the local reachability density of $x_i$ is the inverse of the
          average reachability distance based on the $k$-nearest neighbours of
          $x_i$;
        calculate LOF:
            $$\begin{aligned} LOF_k(x_i) &= \frac{\sum_{o \in N_k(x_i)} lrd_k(x_j)}{|N_k(x_i)| \cdot lrd_k(x_i)} \\ &= \frac{\sum_{o \in N_k(x_i)} \frac{lrd_k(x_j)}{lrd_k(x_i)}}{|N_k(x_i)|} \end{aligned}$$
          the LOF of $x_i$ is the average local reachability density of $x_i$'s
          $k$-nearest neighbours divided by the local reachability density of $x_i$.
    end
end
```

5.3 Experiments

5.3.1 Information on the Datasets

The experiments reported in this chapter are based on 7 two-class imbalanced datasets, including 6 imbalanced benchmark datasets (given in Table 5.1) and a 2D imbalanced chess dataset, which is commonly used for visualising the effectiveness of the selected techniques in the imbalanced learning domain (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018). Figure 5.1 shows the 2D imbalanced chess dataset.

Table 5.1: Information on benchmark datasets (Alcalá-Fdez, Fernández, Luengo, Derrac, García, Sánchez, and Herrera, 2011).

Datasets	**#Attributes**	**#Samples**	**Imbalance Ratio (IR)**
glass1	9	214	1.82
ecoli4	7	336	15.8
vehicle1	18	846	2.9
yeast4	8	1484	28.1
wine quality	11	1599	29.17
page block	10	5472	8.79

5.3.2 Experimental Setup

In this chapter, we propose introducing the *outlier score* and the *four types of samples* as additional attributes of the original imbalanced dataset. The LOF algorithm is an unsupervised anomaly detection method which computes the local density deviation of a given data point relative to its neighbours. Hence, calculating the *outlier score* does not require the information of class labels on either training or test samples. In our experiments, we calculate the LOF values for all samples (before splitting the training and test set). The Python library `PyOD` (Y. Zhao, Nasrullah, and Z. Li, 2019) is used directly to calculate the LOF values. Unlike computing LOF values, computing different types of samples requires the information of class labels, see Table 2.3. However, the labels of test samples should be assumed unknown in

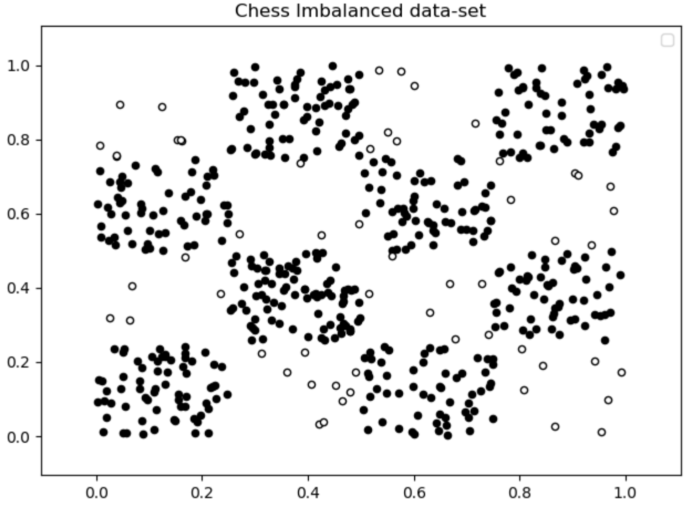


Figure 5.1: Original imbalanced 2D chess dataset. Black points indicate the majority class samples and white points indicate minority class samples.

the training process. Therefore, we use the steps below to add *four types of samples* as an additional attribute "type".

1. Split the data into training and test set.

2. Compute the types for samples in training set. We use positive numbers ($R_{\frac{min}{all}}$) to indicate the types of minority class samples, and negative numbers ($-R_{\frac{maj}{all}}$) for types of majority class samples. For example, for a safe minority class sample with $R_{\frac{min}{all}} = 1$, we add "1" as the "type" value. For a borderline majority class sample with $R_{\frac{maj}{all}} = \frac{3}{5}$, we add "$-\frac{3}{5}$" as the "type" value.

3. Treat training set and test set as a whole. Then, given the number of neighbours k, for each sample in the test set, find the k nearest neighbours belonging to the training set.

4. For a sample in the test set, average the "type" values of its k nearest neighbours belonging to the training set. The average is the "type" value for this sample.

Each dataset was experimented with five resampling techniques and our proposed method. For each method of each dataset, we repeat the experiments 30 times with different random seeds. After that, the paired t-tests were performed on each of the 30 performance metric values to test if there is significant difference between the results of each scenario on a 5% significance level. We perform 5-fold stratified cross validation in our experiments.

5.3.3 Experimental Results and Discussion

Like other studies (H. He, Bai, E. A. Garcia, and S. Li, 2008; López, Fernández, García, Palade, and Herrera, 2013), we also use `SVM` and `Decision Tree` as the base classifiers in our experiments to compare the performance of the proposed method and the existing methods. Our purpose in this chapter is not to achieve the best performance of a certain method under fine hyperparameter tuning. Hence, we did not tune the hyperparameters for the classification algorithms and the resampling techniques (Kong, Kowalczyk, D. A. Nguyen, Bäck, and Menzel, 2019). The experimental results with the two additional attributes (four types of samples and LOF score) are presented in Table 5.2, 5.3, 5.4 and 5.5. Before discussing the experimental results, it is worth mentioning that NCL will not be effective if no samples meet the removal conditions. In this case, NCL will produce the same results as dealing with the original dataset, i.e. row “None" and row “NCL" can be exactly the same in the tables. The tables contain much information, and we will discuss them separately below.

- Scenarios where adding additional attributes performs significantly better than resampling techniques:
  - *2D chess* dataset with `SVM`;
  - *glass1* dataset with `SVM`;
  - *ecoli4* dataset with `Decision Tree`.
- Scenarios where adding additional attributes produces competitive performances to resampling techniques:
  - *vehicle1* dataset with `SVM`;
  - *yeast4* dataset with `Decision Tree` and `SVM`;
  - *wine quality* dataset with `Decision Tree` and `SVM`.
- Scenarios where both resampling techniques and our proposed method do not improve the imbalanced classification performances:

  - *glass1* dataset with `Decision Tree`;
  - *ecoli4* dataset with `SVM`;
  - *page block* dataset with `Decision Tree` and `SVM`.

- Scenarios where adding additional attributes degrades the imbalanced classification performance:
  - *2D chess* dataset with `Decision Tree`;
  - *vehicle1* dataset with `Decision Tree`.

We conclude that in most cases, adding additional attributes produces significantly better or competitive classification performance, except for two scenarios *2D chess* dataset with `Decision Tree` and *vehicle1* dataset with `Decision Tree`. Further feature importance analysis shows that due to the high correlation between the added “type" attribute and class labels, `Decision Tree` uses only the added “type" attribute for classification when dealing with these two datasets. This results in the degradation of these two scenarios. Hence, it is recommended to implement the proposed method with feature-insensitive classifiers.

According to our experimental setup, we notice that introducing the local outlier factor focuses on dealing with the minority samples since the local outlier factor indicates the degree of a sample being an outlier. Meanwhile, introducing four types of samples (safe, borderline, rare and outlier) puts emphasis on separating the overlapping region and safe region. The visualisation of different scenarios for the *2D chess* dataset with `SVM` is given in Figure 5.2 in order to further study the reason for the performance improvement.

From both the experimental results and the visualisation in Figure 5.2, we can conclude that, for the *2D chess* dataset, the experiment with the two additional attributes outperforms the experiment with the classical resampling technique SMOTE. The figure also illustrates that the proposed method has a better ability to handle samples in the overlapping region.

5.4 Conclusions and Future Work

In this chapter, we propose to introduce additional attributes to the original imbalanced datasets in order to improve the classification performance. Two

Table 5.2: Experimental results on *2D chess* and *glass1* datasets with `SVM` and `Decision Tree`.
Row “**Add**" indicates our proposed methods. Bold numbers indicate that the performance is statistically better than the unbold ones. We only bold up to two statistically best results; if more than two competitive results exist, no numbers will be bolded. “—" means that TP+FN=0 or TP+FP=0 and the performance metric cannot be computed.

2D chess dataset										
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.8232	0.7775	0.5324	0.6203	0.7131	0.8284	—	—	—	—
SMOTE	**0.8584**	0.6422	**0.7102**	**0.6646**	**0.8183**	0.5848	0.1564	0.4847	0.2340	0.5743
NCL	0.8232	0.7775	0.5324	0.6203	0.7131	0.8284	—	—	—	—
OSS	0.7569	0.4197	0.5227	0.4554	0.6813	0.6385	0.2611	0.0266	0.0479	0.08478
SMOTEENN	0.8135	0.5519	0.6534	0.5885	0.7763	0.5932	0.1503	**0.5371**	0.2331	**0.5846**
SMOTETL	**0.8586**	0.6581	**0.7018**	**0.6696**	**0.8150**	0.5818	0.1644	0.4824	0.2414	0.5777
Add	0.6033	**0.9000**	0.2106	0.3374	0.4479	**0.8422**	**0.8533**	0.3333	**0.4750**	0.5673
glass1 dataset										
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.6987	0.6056	0.6199	0.5998	0.6764	0.6765	**0.6351**	0.5400	0.5735	**0.6554**
SMOTE	0.6973	0.5650	0.6455	0.5954	0.6700	0.7159	0.5104	0.7255	0.5787	0.6097
NCL	0.7050	0.5850	0.6635	0.6087	0.6772	0.6887	0.5983	0.5800	0.5783	**0.6572**
OSS	0.6885	0.5532	0.6707	0.5999	0.6701	0.6779	0.5882	0.5734	0.5639	0.6397
SMOTEENN	0.6812	0.5407	**0.6905**	0.5955	0.6578	0.7239	0.5098	**0.7596**	**0.5834**	0.5943
SMOTETL	0.6942	0.5793	0.6401	0.5995	0.6723	0.7097	0.5071	0.7192	0.5748	0.6064
Add	0.6954	**0.6633**	0.6050	**0.6139**	0.6773	**0.7885**	—	—	—	—

Table 5.3: Experimental results on *ecoli4* and *vehicle1* datasets with `SVM` and `Decision Tree`.
Row “**Add**" indicates our proposed methods. Bold numbers indicate that the performance is statistically better than the unbold ones. We only bold up to two statistically best results; if more than two competitive results exist, no numbers will be bolded. “—" means that TP+FN=0 or TP+FP=0 and the performance metric cannot be computed.

	ecoli4 dataset									
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.8446	0.7241	0.6433	0.6432	0.7694	0.9919	**0.8889**	0.8000	0.7993	0.8797
SMOTE	0.8803	0.7894	0.7233	0.7080	0.8323	0.9898	0.8299	0.8000	0.7281	0.8470
NCL	0.8446	0.7241	0.6433	0.6432	0.7694	0.9919	**0.8889**	0.8000	0.7993	0.8797
OSS	0.8398	0.6276	0.7267	0.5827	0.7788	0.9867	0.8389	0.8133	0.7504	0.8648
SMOTEENN	0.8105	0.7115	0.7133	0.6251	0.7626	0.9908	0.8238	0.8400	0.7492	0.8585
SMOTETL	0.8685	0.7867	0.7083	0.7001	0.8247	0.9896	0.8295	0.8000	0.7275	0.8465
Add	**0.9058**	**0.8571**	**0.8500**	**0.8032**	**0.9031**	0.9439	0.8571	**0.8500**	0.8032	**0.9031**
	vehicle1 dataset									
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.6699	0.5018	0.4301	0.4575	0.6004	0.8673	**0.7074**	0.3593	0.4747	0.5824
SMOTE	0.7172	0.5293	0.5407	0.5323	0.6687	0.8950	0.5560	0.9269	0.6940	0.8287
NCL	0.7298	**0.5573**	0.5734	0.5635	0.6932	0.8617	0.6063	0.5623	0.5808	0.6987
OSS	0.7055	0.4717	0.5952	0.5241	0.6739	0.8698	0.5747	0.6961	0.6271	0.7537
SMOTEENN	**0.7624**	0.5004	**0.7521**	**0.5993**	**0.7436**	0.8654	0.4949	**0.9388**	0.6467	0.7890
SMOTETL	0.7169	0.5331	0.5377	0.5333	0.6686	0.8945	0.5548	0.9226	0.6919	0.8268
Add	0.6755	0.5210	0.4973	0.5065	0.6440	0.8935	0.5584	0.9077	0.6902	0.8239

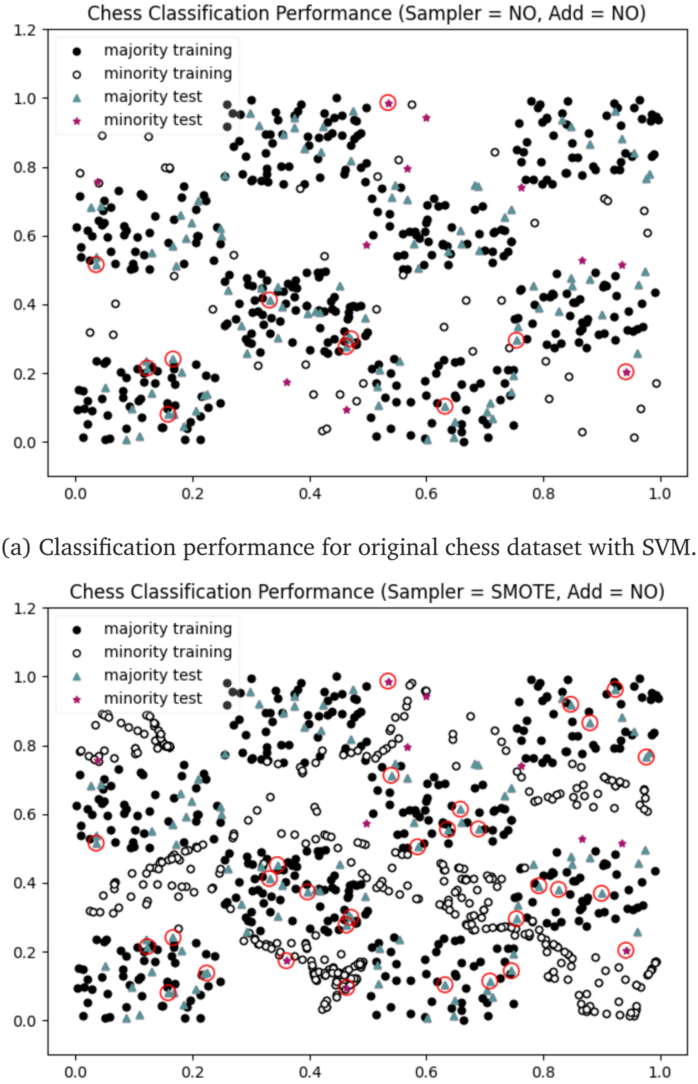
Table 5.4: Experimental results on *yeast4* and *wine quality* datasets with SVM and Decision Tree. Row “**Add**" indicates our proposed methods. Bold numbers indicate that the performance is statistically better than the unbold ones. We only bold up to two statistically best results; if more than two competitive results exist, no numbers will be bolded. “—" means that TP+FN=0 or TP+FP=0 and the performance metric cannot be computed.

yeast4 dataset										
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.6845	**0.2959**	0.2245	0.2465	0.4503	0.8481	—	—	—	—
SMOTE	0.7338	0.2598	0.3968	0.3026	0.6042	0.9033	0.2170	0.6693	0.3182	0.7735
NCL	0.6845	**0.2959**	0.2245	0.2465	0.4503	0.8481	—	—	—	—
OSS	0.7050	0.2838	0.3727	0.3044	0.5827	0.8452	**0.2794**	0.0310	0.0551	0.0961
SMOTEENN	**0.7663**	0.2170	**0.5552**	0.3071	**0.7101**	0.9083	0.1937	**0.7247**	0.2982	**0.7953**
SMOTETL	0.7258	0.2769	0.4042	**0.3166**	0.6112	0.9046	0.2166	0.6674	**0.3174**	0.7724
Add	0.7388	0.2610	0.3709	0.2990	0.5891	0.8977	0.2050	0.6491	0.3009	0.7589

wine quality dataset										
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.5760	**0.1739**	0.1292	0.1283	0.2910	0.6415	—	—	—	—
SMOTE	0.5539	0.0698	0.1720	0.0970	0.3627	0.6934	**0.1180**	0.4292	**0.1825**	0.5948
NCL	0.5760	**0.1739**	0.1292	0.1283	0.2910	0.6415	—	—	—	—
OSS	0.5534	0.0796	0.1735	0.1035	0.3610	0.4466	—	—	—	—
SMOTEENN	**0.6043**	0.0902	**0.3413**	**0.1404**	**0.5262**	0.6995	0.1006	**0.4722**	0.1642	0.6097
SMOTETL	0.5545	0.0709	0.1793	0.0991	0.3669	0.6942	**0.1169**	0.4253	**0.1811**	0.5926
Add	**0.6126**	0.0910	**0.3418**	**0.1429**	**0.5320**	0.7007	—	—	—	—

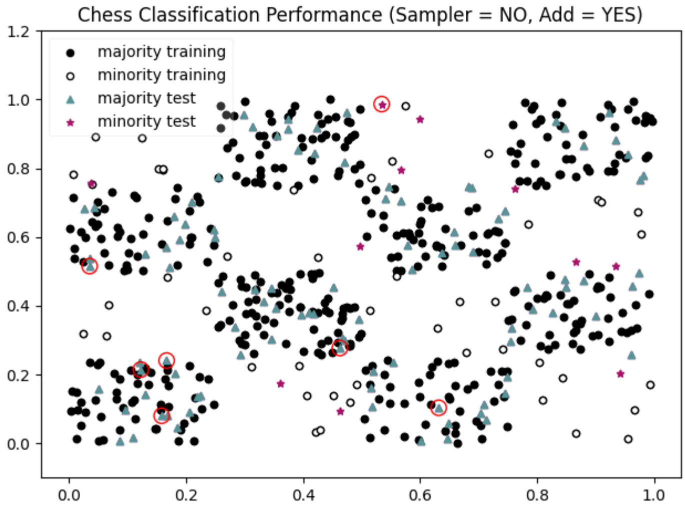
Table 5.5: Experimental results on *page block* dataset with SVM and Decision Tree.
Row "**Add**" indicates our proposed methods. Bold numbers indicate that the performance is statistically better than the unbold ones. We only bold up to two statistically best results; if more than two competitive results exist, no numbers will be bolded. "—" means that TP+FN=0 or TP+FP=0 and the performance metric cannot be computed.

page block dataset										
Methods	**Decision Tree**					**SVM**				
	AUC	**Precision**	**Recall**	**F1**	**Gmean**	**AUC**	**Precision**	**Recall**	**F1**	**Gmean**
NONE	0.9104	0.7840	0.7340	0.7498	0.8451	0.9753	0.8625	0.7025	0.7596	0.8287
SMOTE	0.9198	0.7434	0.8150	0.7708	0.8871	0.9728	0.6621	0.9057	0.7533	0.9239
NCL	0.9163	0.7978	0.7884	0.7857	0.8760	0.9728	0.8471	0.7182	0.7606	0.8364
OSS	0.9228	0.7203	0.8092	0.7514	0.8810	0.9552	0.8280	0.6649	0.7208	0.8044
SMOTEENN	0.9280	0.6994	0.8744	0.7692	0.9137	0.9721	0.6384	0.9120	0.7409	0.9246
SMOTETL	0.9209	0.7425	0.8220	0.7737	0.8908	0.9729	0.6620	0.9061	0.7530	0.9239
Add	0.8491	0.8269	0.7126	0.7559	0.8349	0.9709	0.5669	0.9537	0.7027	0.9340



(a) Classification performance for original chess dataset with SVM.

(b) Classification performance for SMOTE-sampled chess dataset with SVM.



(c) Classification performance for chess dataset with additional attributes with SVM.

Figure 5.2: Classification performance for chess dataset under different scenarios. The red-circled points indicate the misclassified points.

additional attributes, namely four types of samples and outlier score, and the resampling techniques (SMOTE, NCL, OSS, SMOTEENN and SMOTETL) are considered and experimentally tested on seven imbalanced datasets. According to our experimental results, two main conclusions can be derived:

1. In most cases, introducing these two additional attributes can improve or produce competitive class imbalance classification performance. For some datasets, only introducing additional attributes gives better classification results than only performing resampling techniques.

2. The proposed additional attribute “type" is highly correlated with class labels in some datasets. Hence, it is recommended to implement the proposed method with feature-insensitive classifiers.

3. An analysis of the experimental results also illustrates that the proposed method has a better ability to handle samples in the overlapping region.

In this chapter, we only validate our newly proposed method with five resampling techniques and seven benchmark datasets. As future work, other anomaly detection techniques, such as the clustering-based local outlier score (CBLOF) (Z. He, Xu, and Deng, 2003) and histogram-based outlier score (HBOS) (Goldstein and Dengel, 2012) could be included in the analysis. Future work could also consider an extension of this research for engineering datasets (Kong, Rios, Kowalczyk, Menzel, and Bäck, 2020a), especially for the design optimization problems mentioned in our Introduction. Detailed analysis of the feature importance and how the proposed method affects the classification performance in the overlapping region would also be worth studying.

CHAPTER 6

Improved Sample Type Identification for Multi-Class Imbalanced Classification

The idea of studying different types of samples was first proposed and evaluated on binary imbalanced classification problems and then extended to multi-class scenarios. However, simply extending the identification rule in binary scenarios to multi-class scenarios results in several problems. In this chapter, we introduce our proposed sample type identification for multi-class imbalanced classification. First, Section 6.1 shows the motivation and briefly introduces on our work. After that, in Section 6.2, the literature review and problems when extending to multi-class scenarios are presented. In Section 6.3, detailed information on the new identification rule is given. In Section 6.4, the information on the datasets, the experimental setup as well as the experimental results and discussion are introduced. In addition, a real-world application is described in Section 6.5. Section 6.6 concludes the chapter and outlines the further work.

6.1 Introduction

Despite the progress for several years, learning from imbalanced data is still a challenging problem in machine learning. Solving imbalanced classification problems refers to the predictive modelling of data comprising a high or even extreme imbalance in the sample distribution. Since machine learning models assume that the sample distribution is relatively balanced, the nature of imbalanced data violates this assumption, thus the class imbalance is commonly considered the determinant factor for the degradation of classification performance (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018; Ganganwar, 2012). However, several studies in the literature have pointed out that the data characteristics also play a crucial role in dealing with imbalanced problems (López, Fernández, García, Palade, and Herrera, 2013; Napierała, Stefanowski, and Wilk, 2010; Prati, Batista, and Monard, 2004). Here, Napierala and Stefanowski proposed to consider samples from minority class consisting of four types of samples: *safe, borderline, rare* samples and *outliers* (Napierala and Stefanowski, 2016). They studied the influence of these four types of samples on binary imbalanced classification, where the datasets are composed of two classes and one class significantly outnumbers the other. Other researchers then extended this idea to develop new techniques to improve imbalanced classification in both binary and multi-class scenarios (Kong, Kowalczyk, Menzel, and Bäck, 2020; Lango and Stefanowski, 2018; B. Liu and Tsoumakas, 2019). However, the relationships among classes are more complicated in multi-class scenarios since there are more than two classes in the datasets. Simply extending the idea of four types of samples from binary to multi-class scenarios without changing the identification rule will cause several problems.

In this chapter, we first recall the identification rule for the four types of samples as proposed in the literature (Napierala and Stefanowski, 2016). Then, we show the drawbacks when applying this identification rule to multi-class scenarios and emphasize the importance of proposing a new identification rule for multi-class scenarios. We find mainly two drawbacks: (1) a higher percentage of unsafe (*borderline, rare* and *outliers*) samples and (2) false identification of *outliers*. As a consequence, we propose a new identification rule for the four types of samples to handle the drawbacks mentioned above and validate the effectiveness of the new rule with benchmark datasets. In these experiments, we consider oversampling different types of samples before performing the classification, where oversampling is a data-level approach to handle the imbalance in the datasets. Experimental results on benchmark and real-world data show that the proposed rule can significantly improve the classification performance on minority class(es) when a high imbalance exists in the datasets.

Class imbalance is present in many real-world classification tasks, for instance, medical diagnosis (Mazurowski, Habas, Zurada, Lo, Baker, and Tourassi, 2008), email filtering (Bermejo, Gámez, and Puerta, 2011), fault diagnosis (Krawczyk, Galar, Jeleń, and Herrera, 2016), etc. Most of class imbalance applications in the literature have been devoted to binary classification problems. Most of the multi-class imbalanced benchmark datasets contain only a small number of attributes and a limited number of samples (Alcalá-Fdez, Fernández, Luengo, Derrac, García, Sánchez, and Herrera, 2011; D. Dua and Graff, 2017). Therefore, our work makes an additional contribution by introducing a challenging industrial surface defects dataset, with 172 attributes, 27 classes and 12496 samples. Experimental results on this industrial dataset also confirm the effectiveness and usefulness of our proposed rule for real-world applications.

6.2 Related Works

In this section, we first introduce the existing rule for identifying types of samples in binary scenarios from the related literature (Section 6.2.1). Then, we show the drawbacks when extending this idea from binary to multi-class scenarios (Section 6.2.2), which motivates our own research presented in Section 6.3.

6.2.1 Studies on Types of Samples in Binary Scenarios

It is essential to recall the identification of types of samples in binary scenarios. Napierala and Stefanowski first proposed the idea of identifying minority class samples in four categories: *safe, borderline, rare* samples and *outliers* (Napierala and Stefanowski, 2016), the latter three are called *unsafe* samples. The majority class samples can also be categorized into these four types. The general rule to identify the four types is as follows.

- 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.

Since the idea was proposed, it has attracted widespread attention in the field of imbalanced learning, and more than 200 papers have cited the original paper so far. It appears in the citations of review papers as an important development in the imbalanced learning domain, and also in the citations of papers proposing new approaches as a source of inspiration. Various researchers confirmed the occurrence of the different types of samples in real-world data. They studied the influence of different types of minority class samples on binary imbalanced classification (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018), and concluded that the *unsafe* samples are the actual source of difficulty when learning from imbalanced problems (S. Wang, Minku, and Yao, 2018). Studies also focus on investigating the influence of minority class samples on the performance of SMOTE (Skryjomski and Krawczyk, 2017). This idea is also evaluated in real-world applications. For example, authors in (García, Marqués, and Sánchez, 2019) explored the effects of sample types on credit risk and corporate bankruptcy prediction.

6.2.2 Problems When Extending to Multi-class Scenarios

As the importance of learning different types of samples has received more and more attention, some studies extended this idea to multi-class imbalanced classification without changing the identification rule for the four types of samples (Lango and Stefanowski, 2018; Sáez, Krawczyk, and Woźniak, 2016; Sleeman IV and Krawczyk, 2021). However, the relationships among classes in multi-class imbalanced scenarios are more complicated than in binary scenarios, resulting in two main drawbacks if we follow the identification rule for binary scenarios.

- **A higher percentage of unsafe samples in minority classes.** In the identification rule in Table 2.3, the number of neighbours is set the same for all the classes when considering the neighbourhood information. However, this setting neglects the fact that, in multi-class imbalanced classification, minority classes contain significantly fewer samples than in the majority classes. Hence, choosing the same k for all classes in multi-class scenarios will result in a higher percentage of unsafe samples (*borderline, rare, outliers*) in minority classes, see orange triangles (△) in Figure 6.1. The methods we propose to handle this problem are described later in Section 6.3.1.

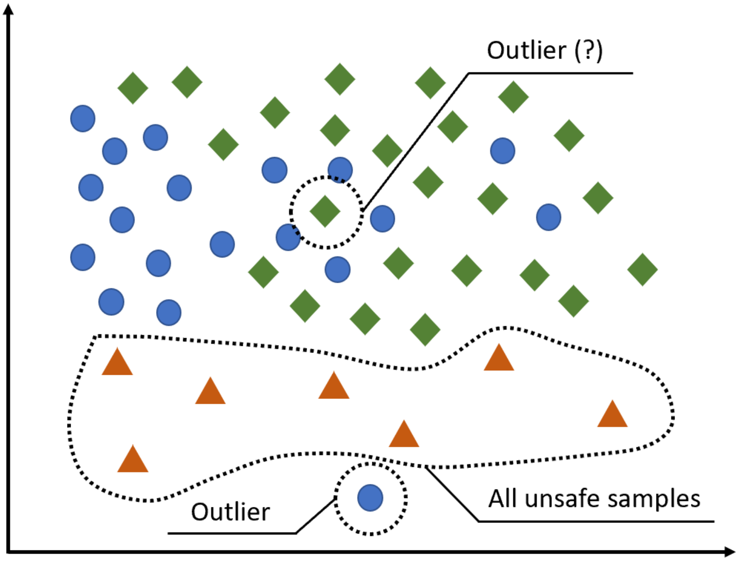


Figure 6.1: An artificial 2-dimensional dataset showing the drawbacks when simply extending the identification rule in binary scenarios to multi-class scenarios. Suppose $k = 5$, then according to the identification rule in Table 2.3, the orange triangles ($\triangle$) are all unsafe samples and the green diamond ($\diamond$) marked with the dotted circle is an outlier.

- **False identification of outliers.** In the identification rule in Table 2.3, *outliers* refer to the isolated samples surrounded by different classes. For example, following this rule, the blue circle at the bottom (Figure 6.1) is classified as an outlier. However, the rule also distinguishes the green diamond ($\diamond$) marked with the dotted circle (Figure 6.1) as an outlier. According to the geometric location of this sample, however, it is not an isolated sample far away from other samples in the same class. This indicates that the current rule leads to the false identification of some samples. In the case of multi-class problems, the relationships among classes are more complex, and our proposed idea to reduce the probability of false identification is detailed in Section 6.3.2.

José et al. (Sáez, Krawczyk, and Woźniak, 2016) analyzed the oversampling of different classes and types of samples with several benchmark multi-class imbalanced datasets. They calculate the percentage of each type of sample (safe/borderline/rare/outlier) using the identification rule for binary scenarios.

Related information on selected datasets is given in Table 6.1. We can observe that if there is a significant gap between the number of minority and majority class samples, over 60% of the minority class samples are considered outliers (see *C1* in *Balance* and *C1 & C2* in *Thyroid*). Hence, we confirm that the drawbacks above exist in multi-class benchmark datasets, and a new identification rule is required for distinguishing the types of samples in multi-class imbalanced scenarios.

Table 6.1: The number of samples of each class in the three selected datasets (detailed information on datasets shown in Table 6.5) and percentage of each type of sample (safe/borderline/rare/outlier) within the class (taken from José's work (Sáez, Krawczyk, and Woźniak, 2016)). "Cj" indicates class j, percentages are rounded to integer values.

Dataset	C1	C2	C3
Balance	49 (0/0/4/**96**)	288 (74/26/0/0)	288 (73/27/0/0)
Thyroid	17 (0/12/6/**82**)	37 (0/11/24/**65**)	666 (97/3/0/0)
Wine	48 (98/2/0/0)	59 (100/0/0/0)	71 (85/14/1/0)

6.3 New Identification Rule for Multi-class Scenarios

In Section 6.2.2, we pointed out two main drawbacks when extending the identification rule from binary to multi-class scenarios. In this section, we propose a new identification rule for multi-class scenarios to overcome these drawbacks.

6.3.1 Adjusting k according to Imbalance Ratio

In the literature, the same k is used when assigning the types for samples in both majority classes and minority classes, where k is the k in k-NN within the sampling methods. However, considering the enormous gap between the sample size of minority and majority classes, choosing the same k will result in a higher percentage of unsafe samples in the minority class (stated in Section 6.2.2). Hence, to ensure a reasonable proportion of different types of samples in minority class(es), a smaller k should be used when analysing the local characteristics of a minority class sample. Here, we propose to adjust k to k_j according to the class distribution, as follows:

$$k_j = \left\lceil \sqrt{\frac{n_j}{N/C}} \times k \right\rceil, \tag{6.1}$$

where $j = 1, \ldots, C$ denotes the class index, n_j is the number of samples in class j, C is the number of classes and $N = \sum_{j=1}^{C} n_j$ is the total number of samples in the dataset. The results of adjusting k as shown in Table 6.2 indicate that Equation (6.1) meets our requirements for choosing a larger k for majority class(es) and a smaller k for minority class(es).

Table 6.2: The number of samples of each class in the three selected datasets and k_j for each class. k is preset to 5 and “Cj" indicates class j.

Dataset	C1	C2	C3
Balance	49 $k_1 = 3$	288 $k_2 = 6$	288 $k_3 = 6$
Thyroid	17 $k_1 = 2$	37 $k_2 = 2$	666 $k_3 = 9$
Wine	48 $k_1 = 5$	59 $k_2 = 5$	71 $k_3 = 6$

6.3.2 Considering neighbourhood Information of the neighbours

In Section 6.2.2, we illustrated that only considering neighbours of a sample is insufficient to identify the type because the neighbourhood information might not adequately reflect the geometric location. Increasing k is a straightforward solution to expand neighbourhood information. However, this will also decrease the number of safe samples for both minority and majority class samples. For example, taking an extreme case, if k is large enough, all samples will be unsafe. Hence, we propose to consider neighbourhood information of the neighbours additionally, i.e. we also find the k nearest neighbours for the neighbours. In our proposed approach, the importance of neighbourhood information usually is higher than of neighbourhood information of the neighbours. A definition of “type score (TS)" of data sample x is given below,

$$\begin{aligned}\text{TS}(x) = &\overbrace{\alpha(x)\cdot\frac{n_x}{k_j}}^{\text{neighbourhood}} + \underbrace{(1-\alpha(x))\cdot\frac{N_x}{(k_j)^2}}_{\text{neighbourhood of the neighbours}} \\ \alpha(x) = &\begin{cases} 1-\frac{1}{k_j} & \text{if } k_j > 1 \\ 0.8 & \text{if } k_j = 1 \end{cases}\end{aligned} \tag{6.2}$$

where x belongs to class j, k_j is the number of nearest neighbours for sample x (see Section 6.3.1), n_x is the number of neighbours which share the same label with sample x, N_x is the number of neighbours of x's neighbours which share the same label with sample x, $\alpha(x)$ is the weight for the neighbourhood information of sample x. If $k_j = 1$, we set $\alpha(x) = 0.8$ (to avoid $\alpha(x) = 1 - \frac{1}{k_j} = 0$) to ensure the higher importance of neighbourhood information. Note that when considering the neighbourhood information of the neighbours, we also use k_j. The proposed identification rule to assign the four types of samples in multi-class scenarios is given in Table 6.3. Following the proposed identification rule, the percentage of each type of sample is recalculated and shown in Table 6.4. For datasets with a significant gap between minority and majority class sample sizes (*Balance* and *Thyroid*), the percentage of *outlier* type decreases from over 60% to less than 30% (compare with Table 6.1).

Table 6.3: Identification rule to assign types for samples in multi-class scenarios. Note that the thresholds can be adjusted (hand-tuned) depending on the given datasets.

Type	Safe	Borderline	Rare	Outlier
Rule	TS>0.75	0.5<TS≤0.75	0.05<TS≤0.5	TS≤0.05

6.4 Experiments

In this section, we introduce the information on the datasets used in our experiments. Then, the experimental setup is described. After that, the experimental results and discussions are given.

Table 6.4: The number of samples of each class in the three selected datasets and percentage of each type of sample (safe/borderline/rare/outlier) within the class. “Cj" indicates class j, percentages are rounded to integer values.

Dataset	C1	C2	C3
Balance	49 (0/0/78/**22**)	288 (70/24/6/0)	288 (70/23/7/0)
Thyroid	17 (6/24/47/**23**)	37 (8/13/49/**30**)	666 (99/1/0/0)
Wine	48 (98/2/0/0)	59 (100/0/0/0)	71 (76/13/8/3)

6.4.1 Information on the Datasets

The experiments in this chapter are based on 6 selected benchmark multi-class imbalanced datasets from the KEEL repository (Alcalá-Fdez, Fernández, Luengo, Derrac, García, Sánchez, and Herrera, 2011). The descriptions of the datasets are summarized in Table 6.5.

Table 6.5: Information on the benchmark datasets. AT, CL and NS indicate the number of attributes, the number of classes and the number of samples respectively.

Dataset	AT	CL	NS (in each class)
Balance	4	3	625 (49 / 288 / 288)
Contraceptive	9	3	1473 (333 / 511 / 629)
Glass	9	6	214 (9 / 13 / 17 / 29 / 70 / 76)
Thyroid	21	3	720 (17 / 37 / 666)
Wine	13	3	178 (48 / 59 / 71)
Winequality-red	11	6	1599 (10 / 18 / 53 / 199 / 638 / 681)

6.4.2 Experimental Setup

In this chapter, we (1) improve the rule for identifying the four types of samples for multi-class imbalanced problems and (2) investigate how oversampling for different types of sample combinations affects the classification performance. Our experimental setup is illustrated in Figure 6.2. We consider $\binom{4}{4}+\binom{4}{3}+\binom{4}{2}+\binom{4}{1} = 15$ (excluding *None*) combinations of the four types of samples and SMOTE (Chawla, Bowyer, Hall, and Kegelmeyer, 2002) to oversample these combinations in our experiments. To be specific, $\binom{4}{4}$ means we choose all four types of samples to be oversampled, $\binom{4}{3}$ means we choose three out of four types of samples to be oversampled, $\binom{4}{2}$ means we choose two out of four types of samples to be oversampled and $\binom{4}{1}$ means we choose only one type of samples to be oversampled. Three classifiers (C5.0, SVM and Nearest Neighbour) are used as classification algorithms, and 5-fold stratified cross-validation is used to preserve the original class distribution (M. S. Santos, Soares, Abreu, Araujo, and J. Santos, 2018).

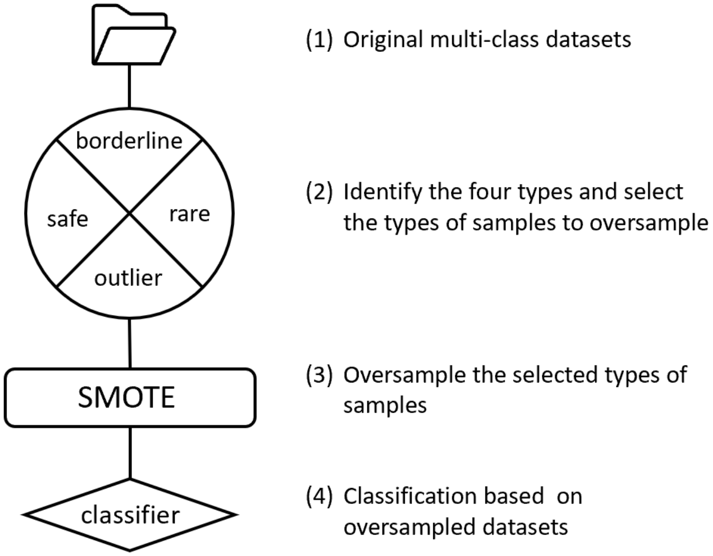

borderline
safe
rare
outlier
SMOTE
classifier
(1) Original multi-class datasets
(2) Identify the four types and select the types of samples to oversample
(3) Oversample the selected types of samples
(4) Classification based on oversampled datasets

Figure 6.2: Experimental setup to compare the effectiveness of the two different identification rules (inspired by (Sáez, Krawczyk, and Woźniak, 2016)). The comparison is done via changing the identification rule in step (2).

6.4.3 Experimental Results and Discussion

Experimental results of the decision tree C5.0 (average of 30 trials) on *Balance* and *Winequality-red* are given in Table 6.6 and Table 6.7. Note that there is one minority class in *Balance* and three minority classes in *Winequality-red*. Three main conclusions can be drawn from our experiments:

Table 6.6: Performance results of decision tree (C5.0) on the dataset *Balance*. “1 0 1 0" represents “safe(1) borderline(0) rare(1) outlier(0)", i.e. only safe and rare samples are oversampled. $R_{min/all}$ and *TS* indicate the different rules for identifying types of samples. “–" means that there are not enough samples to execute the k-nearest-neighbour algorithm in the oversampling step.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.0129	0.0129	0.7449	0.7449
1 1 1 0	–	0.1590	–	0.8179
1 1 0 1	**0.1546**	0.1374	0.8119	0.7712
1 0 1 1	0.1386	0.1600	**0.8138**	0.8216
0 1 1 1	0.0535	0.0676	0.7894	0.7934
1 1 0 0	0	0.0222	0.7534	0.7470
1 0 1 0	–	**0.1907**	–	**0.8219**
0 1 1 0	–	0.1301	–	0.8101
1 0 0 1	0.1151	0.1037	0.8092	0.7764
0 1 0 1	0.0474	0.0823	0.7825	0.7810
0 0 1 1	–	0	–	0.7348
1 0 0 0	0	0	0.7489	0.7537
0 0 1 0	–	0	–	0.7303
0 1 0 0	–	0	–	0.7481
0 0 0 1	–	–	–	–

- Taking different types of sample combinations into account in the oversampling technique can significantly improve the classification performance on minority class(es). At the same time, improved or competitive classification performance on the whole dataset can also be achieved. Please refer to the bold numbers, the best performance in the 15 combinations, in Table 6.6 and Table 6.7. This improvement can be explained by the fact that, when considering different combinations, one or several types of samples will be discarded. This can be regarded as an informed undersampling to balance the class distribution.

- From the performance comparison between two identification rules ($R_{min/all}$ and *TS*), it can be concluded that our proposed identification rule provide significantly better performance on classifying minority class(es). Moreover, there are less “–" in the experiments using the proposed identification rule, where “–" means that there are not enough samples to execute the k-nearest-neighbour algorithm in the oversampling step. Both points confirm the appropriateness of and improvement provided by the proposed rule.

- Only experimental results on the dataset *Winequality* are shown in this chapter. Experimental results on other datasets can be found in Appendix A. The relationship between imbalance ratio and *MinAcc* is shown in Figure 6.3. The imbalance ratio (IR) for multi-class classification in this chapter is defined as the average majority sample size to the average minority class sample size. It is worth mentioning that if the imbalanced ratio is not significant (< 4), oversampling different combinations of types of samples will not bring a significant improvement on minority classification performance. However, no linear relationship between the imbalance ratio and *MinAcc* can be concluded (see linear regression equation and R^2 in Figure 6.3). This is because the improvement is not only determined by the imbalance ratio, but also depends on the separability of classes.

Table 6.7: Performance results of C5.0 on the dataset *Winequality-red*. The huge difference in the corresponding positions of the two columns in *MinAcc* is caused by the significant difference between the four types of samples under the two identification rules, i.e., data distribution in different combinations varies a lot.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.0819	0.0819	**0.6751**	0.6751
1 1 1 0	–	0.0771	–	0.6581
1 1 0 1	0.0281	0.1219	0.6571	0.6637
1 0 1 1	0.0520	0.0588	0.6600	0.6627
0 1 1 1	0.0466	0.1170	0.6541	0.6534
1 1 0 0	–	–	–	–
1 0 1 0	–	0.0498	–	0.6576
0 1 1 0	–	0.0394	–	0.6548
1 0 0 1	**0.1305**	0.0444	0.6518	0.6584
0 1 0 1	0.0511	0.1140	0.6553	0.6601
0 0 1 1	0.0851	0.0680	0.6615	0.6637
1 0 0 0	–	0.0698	–	**0.6782**
0 0 1 0	–	0.0875	–	0.6616
0 1 0 0	–	–	–	–
0 0 0 1	0.0563	**0.1485**	0.6461	0.6453

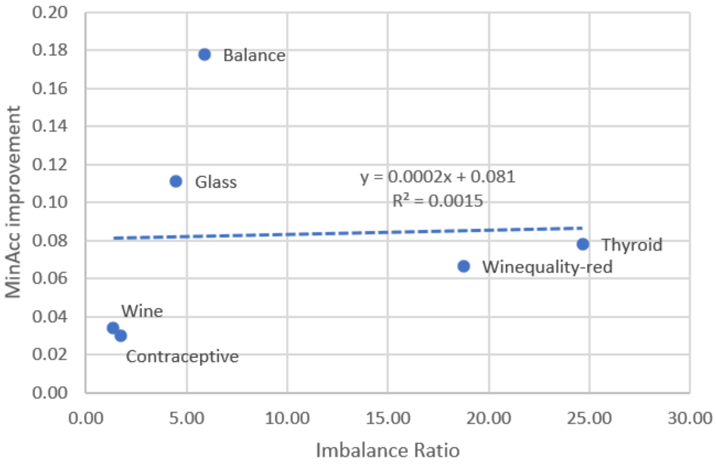


Figure 6.3: Relationship between imbalance ratio and *MinAcc*. The imbalance ratio (IR) for multi-class classification in this chapter is defined as the average majority sample size to the average minority class sample size.

6.5 Applications on the Detection of Surface Defects

In this section, we report our study on an imbalanced application for detecting surface defects. We first introduce the industrial problem. Then, the information on the surface defects dataset is given in Section 6.5.1. After that, the visualisation and preprocessing step on this high-dimensional dataset is described in Section 6.5.2. In Section 6.5.3, we evaluate our proposed sample identification rule on the surface defects dataset.

The surface of a steel product is one of the major quality aspects. Therefore, surface anomalies should be avoided or at least known. A camera-based Surface Inspection Systems (SIS) is used in various process lines to identify those anomalies in the industry (Neogi, Mohanta, and Dutta, 2014). Grey value images taken from the surface by the SIS contains information on the anomalies. These images of various anomalies occurring in production are assessed and gathered in defined classes within a defect library. Figure 6.4 shows a diagram of how to capture the defects images. The defect library is used to train and test classifiers (classification algorithms), and these classifiers are finally used to identify the new surface anomalies from production. Thus, a stable, accurate and high classification performance is a must in the quality check procedure. However, the imbalance in the number of various defect types makes it challenging to obtain a stable and accurate classification performance.

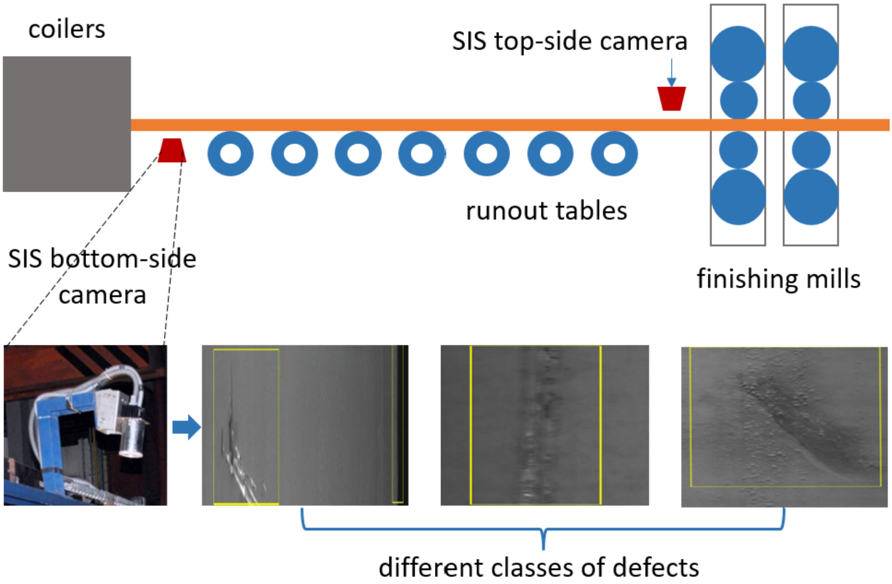


Figure 6.4: A diagram of how to capture the defects images. Defects images are from TATA steel official website[1], for example purpose.

6.5.1 Information on Surface Defects Dataset

The images captured by the SIS cameras will be processed in the feature extraction module. Relevant defect features, e.g. geometrical, textural and moment features, are extracted for the purpose of classification. Both the images and information after extraction will be stored in the defect library. The surface defects dataset used in this chapter is taken from a defect library after a certain selection (for privacy reasons). The dataset is after extraction and contains 12496 samples along with 173 attributes. After removing samples with missing values, there are 12456

[1] https://automation.tatasteel.com/products/rolling-mills/squins-surface-quality-inspection-system/

samples in total. The information on surface defects data for experiments is given in Table 6.8.

Table 6.8: Dimension of each record in the *surface defects* dataset after preprocessing. NS and “class" indicate the number of samples and class label respectively. There are 25 classes and 12456 samples in total

class	NS	class	NS	class	NS	class	NS
25	2012	1	385	11	282	20	134
17	1666	10	382	19	255	23	121
24	1211	12	379	22	243	6	71
15	1205	16	357	9	215	4	39
18	937	7	354	21	201		
3	623	5	312	27	165	Total	
2	457	13	296	8	154	25	12456

6.5.2 Visualisation and Preprocessing

Visualisation is an important step when dealing with real-world applications. It can provide some general information on the datasets, e.g. clusters. In the data-preprocessing step, missing values and redundant attributes are usually removed to provide high-quality data for future experiments.

Visualisation with t-SNE

Before experimenting with this real-world application dataset, we visualise the data to get some general information on the data. *T-distributed Stochastic neighbourhood Embedding* (t-SNE) (Van der Maaten and G. Hinton, 2008), a variation of *Stochastic neighbourhood Embedding* (SNE) (G. E. Hinton and Roweis, 2002), is a statistical technique for visualising high-dimensional data. It first converts high-dimensional Euclidean distance into conditional probability to characterise similarity among data points. Then, t-SNE models the similarity distribution among data points in the low-dimensional map. After that, it minimises the Kullback-Leibler divergence (KL divergence) between the joint distributions in high-dimensional and low-dimensional space.

t-SNE has been used for visualisation in various applications, consisting of medical research (Esteva, Kuprel, Novoa, Ko, Swetter, Blau, and Thrun, 2017), music analysis (Van den Oord, Dieleman, and Schrauwen, 2013), bioinformatics

(Baxevanis, Bader, and Wishart, 2020), etc. In this chapter, we use t-SNE to visualise the surface defects data from industry. As we discussed in Section 6.2.2, the relationships among classes in multi-class scenarios are more complicated than in binary scenarios. It is very intuitive from Figure 6.5 that as the number of classes increases, it gets more and more difficult to visualise the boundaries of different classes.

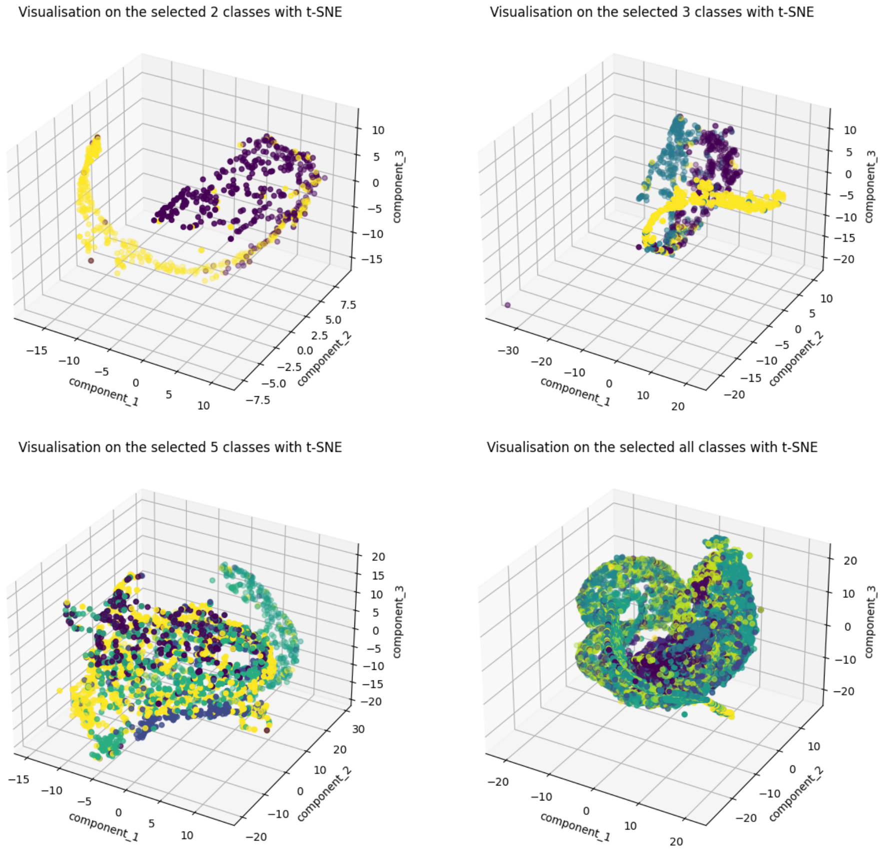


Figure 6.5: Visualisation on *surface defects* dataset with 2/3/5/all classes (top-left/top-right/bottom-left/bottom-right).

Data Preprocessing

As we mentioned in Section 6.5.1, we have already deleted the missing values. Therefore, in this chapter, we focus on reducing dimensionality via feature correlation analysis, i.e. feature selection. *Correlation* is a statistical term to describe the linear relationship between two or more variables. When correlation happens in features (attributes), we call this *feature correlation*. In other words, if two features have a high correlation, we can predict one from the other. When training a predictive model based on a certain dataset, correlated features are considered redundant and we can delete one of them for simplification. As per the *Occam's razor*, "entities should not be multiplied beyond necessity" (Schaffer, 2015). (In Latin, *Entia non sunt multiplicanda praeter necessitatem* (Bauer, 2007).)

According to the information from the industry (which provides the surface defects data), the first 20 attributes in the surface defects dataset are only for internal recording, such as image number, date, top camera or bottom camera, etc. These features provide no information on the defects and can be directly deleted. After that, we calculate the feature correlation through *Pearson* correlation. From Figure 6.6, we can observe that many features are highly correlated. For our surface defects dataset, if the correlation between two features is higher than 0.7 (this number is suggested by the industrial expert in TATA company), one of them will be deleted. After removing the redundant features, there are 62 features left for future experiments.

6.5.3 Experiments on Surface Defects Dataset

Experimental results on the industrial surface defects dataset are given in Table 6.9. This real-world dataset is a multi-class imbalanced dataset with an extreme imbalance ratio. Significant improvements on both minority and overall classification performance can be observed in Table 6.9. This is consistent with our conclusions from the experiments on benchmark datasets in Section 6.4.3. Furthermore, the best performances out of 15 combinations are contributed mainly by "no outliers (1 1 1 0)", which also shows that the outlier type has a significant influence on the classification performance in real-world imbalanced problems. In addition, the proposed identification rule (TS) outperforms the other one on classifying minority class samples. This confirms that the proposed rule can better recognise the outliers in this real-world problem.

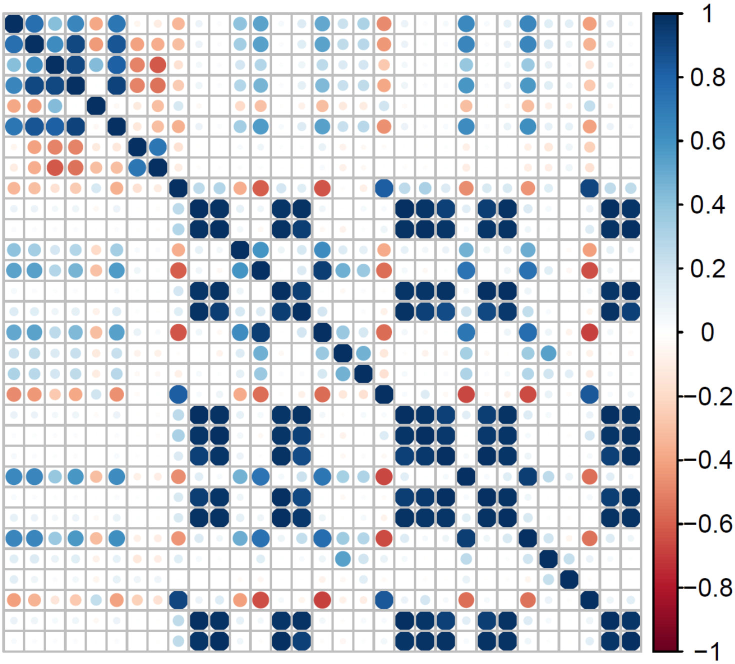


Figure 6.6: Visualisation of correlation matrix on 31 selected features. Positive correlations are displayed in blue and negative correlations in red color. Color intensity and the size of the circle are proportional to the correlation coefficients.

6.6 Conclusions and Future Work

The idea of introducing four types of samples (safe, borderline, rare and outlier) in binary imbalanced literature has been done already. This chapter introduces the drawbacks of extending this idea to multi-class imbalanced scenarios. We proposed a new identification rule to deal with these drawbacks and evaluated the effectiveness of this proposed rule on six benchmark datasets and a real-world application. According to our experimental results, the following conclusions can be derived:

- Oversampling different combinations of types of samples can provide better or competitive performance in classifying minority class(es) while not losing too much classification performance on majority class samples.
- The proposed identification rule for types of samples makes the percentage of

Table 6.9: Performance results of C5.0 in *surface defects* dataset."1 0 1 0" represents "safe(1) borderline(0) rare(1) outlier(0)", i.e. only safe and rare samples are oversampled. $R_{min/all}$ and *TS* indicate the different rules for identifying types of samples. "–" means that there are not enough samples to execute the k-nearest-neighbour algorithm in the oversampling step.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.5256	0.5256	0.8748	0.8748
1 1 1 0	**0.5361**	**0.5468**	0.8900	**0.8917**
1 1 0 1	0.4927	0.4780	**0.8924**	0.8881
1 0 1 1	0.5022	0.4994	0.8879	0.8880
0 1 1 1	0.5040	0.4923	0.8759	0.8746
1 1 0 0	–	–	–	–
1 0 1 0	–	0.5430	–	0.8914
0 1 1 0	0.5190	0.5301	0.8796	0.8794
1 0 0 1	0.4806	0.4754	0.8871	0.8857
0 1 0 1	0.4903	0.4671	0.8803	0.8758
0 0 1 1	0.4891	0.4944	0.8668	0.8679
1 0 0 0	–	–	–	–
0 0 1 0	–	–	–	–
0 1 0 0	–	–	–	–
0 0 0 1	–	–	–	–

each type of sample within the class more reasonable (avoiding all samples in the minority class considered as outliers).

- Our experimental results do not show significant improvement on datasets that are not highly imbalanced. Therefore, it is recommended to analyse the types of samples only when the dataset is highly imbalanced.
- The proposed identification rule can be applied to real-world multi-class imbalanced datasets and significantly improve the classification performance. When dealing with real-world problems, much attention should be paid to the sample type "outlier".

In future work, it is worth studying the relationship between imbalance ratio, separability of classes and performance improvement while analysing the four types of samples in the imbalanced learning domain. In addition, further study on applying the proposed identification rule to more real-world applications is encouraged. However, real-world data available in the machine learning community is rare due to confidentiality and the time-consuming generation.

We also would like to explore how these four types of samples can be used for interacting with and benefiting from the feedback of human experts in real-world applications. One scenario is, for example, the rule identifies some outlier samples and plans to delete these samples in future analysis. Then, the human experts check whether these are real outliers and provide feedback to the algorithm training process.

CHAPTER 7

Conclusions

As class-imbalance problems attract attention from academic and industrial fields, many approaches have been proposed to improve the imbalanced classification theoretically and practically. In this thesis, we mainly conducted research on *Learning Class-Imbalanced Problems from the Perspective of Data Intrinsic Characteristics*. In the following, Section 7.1 first summarizes the main contributions of the thesis as the answers to the research questions in the Introduction chapter. Then, the strengths and weaknesses of the research work (Chapters 3-6) are also discussed following the chapter order. Finally, the outlook on future research is provided in Section 7.2.

7.1 Summary

Chapter 1 introduced the scientific background and the motivation of this thesis. It showed the existence of class imbalance problems in real-world applications and emphasized the importance of learning from imbalanced data. Then, the outline of the thesis and relevant publications were given.

Chapter 2 provided the necessary literature review. It started with the visualisation of a binary class imbalance problem. After that, the methods and the performance metrics for both binary and multi-class class imbalance scenarios were presented. Furthermore, the studies on the data complexity in the imbalanced learning domain were introduced in detail. Finally, the imbalanced benchmark datasets and the real-world imbalanced application work were reviewed.

Chapter 3 investigated the effectiveness of several oversampling techniques, where the new ones (RACOG, wRACOG and RWO-sampling) take into account the minority class distribution, while the “classic" ones (SMOTE, ADASYN and MWMOTE) do not. These oversampling techniques were experimented with 19 benchmark datasets and our real-world inspired vehicle dataset. The experimental results first answered **research question 1**: In most cases, oversampling approaches considering the minority class distribution perform better. Different data complexity measures were taken into account with the original aim to answer **research question 2**. According to our experimental results, no apparent relationship between data complexity measures and the choice of resampling techniques can be derived. One noteworthy finding is that the F1v value strongly correlates with the potential best AUC value (after resampling).

Although “new" oversampling approaches showed effectiveness over “classical" ones in most cases, one main practical limitation must be taken into account. Due to the fact that “new" oversampling techniques consider the minority class distribution, implementing these techniques often requires more time compared to “classical" ones. When facing large datasets, huge time costs are inevitable. Therefore, the trade-off between performance improvement and time consumption must be considered while using “new" oversampling techniques.

Chapter 4 introduced our work on hyperparameter optimisation on class-imbalance problems. Both hyperparameters in resampling techniques and classification algorithms were optimised in our experiments. Further exploration of how data complexity affects the classification improvement yielded via hyperparameter optimisation answered our **research question 3**. Applying hyperparameter optimisation for both classification algorithms and resampling approaches can significantly improve the performance of imbalanced datasets with low class overlap. However, oversampling techniques and hyperparameter optimisation do not improve performance for imbalanced datasets with high class overlap.

Despite the fact that hyperparameter optimisation improves the classification performance significantly for imbalanced datasets with low class overlap, the optimisation process always involves hundreds to thousands of iterations. The additional time consumption is significant. Moreover, different resampling techniques contain different hyperparameters. Therefore, one needs to have an in-depth understanding of the resampling techniques in order to set the hyperparameters that need to be optimised.

Chapter 5 conducted research on improving imbalanced classification via adding additional attributes. We proposed introducing the outlier score and four types of samples as two additional attributes of the original imbalanced datasets. We compared the classification performance of our proposed method and the resampling techniques in the literature and concluded that adding additional attributes in most cases produces significantly better or competitive classification performance. This naturally leads to the answer of **research question 4**: we can take advantage of anomaly detection techniques to improve the imbalanced classification.

We must consider the following points when using our proposed method to improve the imbalanced classification. Firstly, we have shown that the proposed attribute “type" highly correlates with the class labels under certain circumstances. Hence, we recommend choosing feature-insensitive classification algorithms when implementing our proposed method. Furthermore, considering the fact that anomaly detection problems are imbalanced problems with extreme imbalance ratios, it is recommended to add the outlier score as an additional attribute when the imbalance ratio is relatively high (no less than 5).

Chapter 6 presented our improved sample type identification for multi-class imbalanced classification. We showed the drawbacks of the existing identification rule in multi-class scenarios, (i) a higher percentage of unsafe samples in minority classes and (ii) the false identification of outliers. The proposed rule answered the **research question 5**, we can improve the sample identification by adjusting k according to the imbalance ratio and considering neighborhood information of the neighbors. The proposed approach was tested on a challenging real-world problem, the steel surface defects detection task. The experimental results answered **research question 6**, showing the industrial applicability of our method.

We used two performance metrics to evaluate the experimental results, *MinAcc* for assessing the performance of minority class(es) and *MAUC* for measuring the overall performance of all classes. According to our experimental setup, the proposed identification rule significantly better classifies minority class(es) while producing competitive overall classification performance. Hence, one main limitation of the proposed method is that a significant better performance cannot be guaranteed if the given task only focuses on the overall performance.

7.2 Future Work

This thesis mainly conducted the research on *Learning Class-Imbalanced Problems from the Perspective of Data Intrinsic Characteristics*. Despite the achievements presented here that have revealed interesting insights, learning the data intrinsic characteristics in imbalanced datasets and how to efficiently use these characteristics to obtain guidance on choosing the imbalanced techniques still need to be completed. Furthermore, much work is yet to be done to apply the class imbalance techniques to handle complex real-world scenarios. Several possible future research directions for extending the work in this thesis are discussed as follows.

Software Tool for Learning from Class Imbalance Datasets As we have shown in this thesis, the class imbalance problem has been studied extensively from different aspects, including data interpolation, algorithm adjusting, cost-sensitive learning, data complexity etc. Given a class imbalance problem, one has to try several techniques and choose the best one for the specific situation. However, these techniques are available in different languages, *Python*, *R* and *C*, which makes it challenging for researchers to implement and compare. Therefore, software with the following functions would greatly contribute to the community.

- Main class-imbalance techniques, e.g. various resampling techniques, different algorithm-level approaches, cost-sensitive learning approaches and ensemble learning methods.

- Efficient hyperparameter optimisation algorithms to choose the optimal combination of hyperparameters.

- Data complexity analysis to provide some algorithm selection insights.

- Several benchmark examples to help beginners understand the functions of the software.

Comparative Study of Anomaly Detection and Class Imbalance Problem The anomaly detection problem can be considered a class imbalance problem with an extreme imbalance ratio. In this thesis, we proposed to add the *Local Outlier Score* as an additional attribute to gain more information for the original imbalanced dataset. In future work, other anomaly detection techniques, such as the clustering-based local outlier score (CBLOF) (Z. He, Xu, and Deng, 2003) and histogram-based outlier score (HBOS) (Goldstein and Dengel, 2012) could be included in the analysis. It is also interesting to explore other potential attributes to be added.

Data Complexity in Real-Time Processing In this thesis we mainly focused on stationary imbalanced datasets, whereas in many applications, such as fault diagnosis and bank commercial monitoring systems (H. M. Nguyen, Cooper, and Kamei, 2011), the data is constantly arriving and real-time analysis must be given. This scenario refers to the topic of *Online Class Imbalance Learning from Imbalanced Data Streams* (Fernández, García, Herrera, and Chawla, 2018; M. Last, 2002), which combines the difficulties of data stream mining and class imbalance problems (Fernández, García, Galar, Prati, Krawczyk, and Herrera, 2018; S. Wang, Minku, and Yao, 2014). In this type of problem, the new learning instances arrive in a time-based manner and the class distribution is dynamic. The imbalance ratio may evolve over time, making the relationship dynamic so that the algorithms with fixed imbalance ratio assumptions are not valid anymore. For example, when the imbalanced problem evolves into a balanced problem, it will lead to the failure of the previous imbalanced algorithm. When the majority class evolves into the minority one (or vice versa), the algorithm may even bring more imbalance bias to the problem. Thus, analysing the data complexity dynamically and adjusting the approaches accordingly would significantly benefit applications.

Appendices

APPENDIX A

Additional Experimental Results

Table A.1: Performance results of decision tree (C5.0) on the dataset *Contraceptive*. “1 0 1 0" represents “safe(1) borderline(0) rare(1) outlier(0)", i.e. only safe and rare samples are oversampled. $R_{min/all}$ and *TS* indicate the different rules for identifying types of samples.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.4154	0.4154	0.6736	0.6744
1 1 1 0	0.3925	0.3503	0.6734	0.6735
1 1 0 1	0.3800	0.3846	0.6714	0.6753
1 0 1 1	0.3978	0.3690	0.6607	0.6617
0 1 1 1	0.3530	0.3695	0.6670	0.6643
1 1 0 0	0.4296	**0.4360**	0.6807	**0.6834**
1 0 1 0	0.3773	0.3518	0.6689	0.6655
0 1 1 0	0.3865	0.3882	0.6737	0.6699
1 0 0 1	0.3814	0.3932	0.6669	0.6700
0 1 0 1	0.3988	0.3950	0.6725	0.6678
0 0 1 1	0.3963	0.3605	0.6679	0.6626
1 0 0 0	**0.4457**	0.4360	**0.6884**	0.6826
0 0 1 0	0.3666	0.3688	0.6698	0.6676
0 1 0 0	0.3899	0.4207	0.6771	0.6768
0 0 0 1	0.4343	0.4040	0.6841	0.6622

Table A.2: Performance results of decision tree (C5.0) on the dataset *Thyroid*. “1 0 1 0" represents “safe(1) borderline(0) rare(1) outlier(0)", i.e. only safe and rare samples are oversampled. $R_{min/all}$ and *TS* indicate the different rules for identifying types of samples. “–" means that there are not enough samples to execute the k-nearest-neighbor algorithm in the oversampling step.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.8648	0.8648	0.9813	0.9808
1 1 1 0	0.7789	0.7708	0.9829	0.9733
1 1 0 1	0.7221	0.7486	0.9726	0.9736
1 0 1 1	0.7227	0.7440	0.9703	0.9737
0 1 1 1	**0.9432**	**0.9350**	0.9831	**0.9830**
1 1 0 0	0.7011	–	0.9774	0.9712
1 0 1 0	–	0.7306	–	0.9765
0 1 1 0	0.7694	0.7756	**0.9838**	0.9815
1 0 0 1	0.7816	–	0.9735	0.9744
0 1 0 1	0.8224	–	0.9831	0.9814
0 0 1 1	–	–	–	–
1 0 0 0	–	–	–	–
0 0 1 0	–	–	–	–
0 1 0 0	–	–	–	–
0 0 0 1	–	–	–	–

Table A.3: Performance results of decision tree (C5.0) on the dataset *Wine*. "1 0 1 0" represents "safe(1) borderline(0) rare(1) outlier(0)", i.e. only safe and rare samples are oversampled. $R_{min/all}$ and *TS* indicate the different rules for identifying types of samples. "–" means that there are not enough samples to execute the k-nearest-neighbor algorithm in the oversampling step.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.9385	0.9297	0.9493	0.9495
1 1 1 0	0.9578	0.9600	**0.9619**	**0.9606**
1 1 0 1	0.9232	0.9192	0.9577	0.9560
1 0 1 1	0.9500	0.9800	0.9546	0.9553
0 1 1 1	–	–	–	–
1 1 0 0	0.9068	0.9436	0.9583	0.9556
1 0 1 0	0.8986	0.9378	0.9531	0.9547
0 1 1 0	–	–	–	–
1 0 0 1	**0.9618**	0.9374	0.9529	0.9492
0 1 0 1	–	–	–	–
0 0 1 1	–	–	–	–
1 0 0 0	0.9532	**0.9636**	0.9530	0.9475
0 0 1 0	–	–	–	–
0 1 0 0	–	–	–	–
0 0 0 1	–	–	–	–

Table A.4: Performance results of decision tree (C5.0) on the dataset *Glass*. “1 0 1 0" represents “safe(1) borderline(0) rare(1) outlier(0)", i.e. only safe and rare samples are oversampled. $R_{min/all}$ and *TS* indicate the different rules for identifying types of samples. “–" means that there are not enough samples to execute the k-nearest-neighbor algorithm in the oversampling step.

Combination	MinAcc		MAUC	
	$R_{min/all}$	TS	$R_{min/all}$	TS
1 1 1 1	0.6243	0.6291	0.8603	0.8605
1 1 1 0	**0.7357**	0.6778	0.8903	**0.8958**
1 1 0 1	0.4933	**0.7111**	**0.9010**	0.8925
1 0 1 1	0.4778	0.6156	0.8798	0.8840
0 1 1 1	0.5211	0.6522	0.8954	0.8952
1 1 0 0	–	–	–	–
1 0 1 0	–	–	–	–
0 1 1 0	–	–	–	–
1 0 0 1	–	–	–	–
0 1 0 1	–	–	–	–
0 0 1 1	–	–	–	–
1 0 0 0	–	–	–	–
0 0 1 0	–	–	–	–
0 1 0 0	–	–	–	–
0 0 0 1	–	–	–	–

Bibliography

Abdi, L. and Hashemi, S. (2015). “To combat multi-class imbalanced problems by means of over-sampling techniques”. In: *IEEE transactions on Knowledge and Data Engineering* vol. 28, no. 1, pp. 238–251.

Acharya, U. R., Chowriappa, P., Fujita, H., Bhat, S., Dua, S., Koh, J. E., Eugene, L., Kongmebhol, P., and Ng, K. (2016). “Thyroid lesion classification in 242 patient population using Gabor transform features from high resolution ultrasound images”. In: *Knowledge-Based Systems* vol. 107, pp. 235–245.

Agrawal, A. and Menzies, T. (2018). “Is" Better Data" Better Than" Better Data Miners"?” In: *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*. IEEE, pp. 1050–1061.

Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., and Herrera, F. (2011). “Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework.” In: *Journal of Multiple-Valued Logic & Soft Computing* vol. 17.

Alcalá-Fdez, J., Sánchez, L., Garcia, S., Jesus, M. J. del, Ventura, S., Garrell, J. M., Otero, J., Romero, C., Bacardit, J., Rivas, V. M., et al. (2009). “KEEL: a software tool to assess evolutionary algorithms for data mining problems”. In: *Soft Computing* vol. 13, no. 3, pp. 307–318.

Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999). *Modern information retrieval*. Vol. 463. ACM press New York.

Barua, S., Islam, M. M., Yao, X., and Murase, K. (2012). “MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning”. In: *IEEE Transactions on Knowledge and Data Engineering* vol. 26, no. 2, pp. 405–425.

Batista, G. E., Prati, R. C., and Monard, M. C. (2004). "A study of the behavior of several methods for balancing machine learning training data". In: *ACM SIGKDD explorations newsletter* vol. 6, no. 1, pp. 20–29.

Bauer, L. (2007). *Linguistics Student's Handbook*. Edinburgh University Press.

Baxevanis, A. D., Bader, G. D., and Wishart, D. S. (2020). *Bioinformatics*. John Wiley & Sons.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). "Algorithms for hyper-parameter optimization". In: *Advances in neural information processing systems* vol. 24.

Bergstra, J., Komer, B., Eliasmith, C., Yamins, D., and Cox, D. (July 2015). "Hyperopt: A Python library for model selection and hyperparameter optimization". In: *Computational Science & Discovery* vol. 8, p. 014008.

Bergstra, J., Yamins, D., and Cox, D. (2013). "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures". In: *International conference on machine learning*. PMLR, pp. 115–123.

Bermejo, P., Gámez, J. A., and Puerta, J. M. (2011). "Improving the performance of Naive Bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets". In: *Expert Systems with Applications* vol. 38, no. 3, pp. 2072–2080.

Bhowan, U., Johnston, M., Zhang, M., and Yao, X. (2012). "Evolving diverse ensembles using genetic programming for classification with unbalanced data". In: *IEEE Transactions on Evolutionary Computation* vol. 17, no. 3, pp. 368–386.

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*. Vol. 4. 4. Springer.

Błaszczyński, J., Deckert, M., Stefanowski, J., and Wilk, S. (2010). "Integrating selective pre-processing of imbalanced data with ivotes ensemble". In: *International conference on rough sets and current trends in computing*. Springer, pp. 148–157.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). "LOF: identifying density-based local outliers". In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104.

Cao, P., Yang, J., Li, W., Zhao, D., and Zaiane, O. (2014). "Ensemble-based hybrid probabilistic sampling for imbalanced data learning in lung nodule CAD". In: *Computerized Medical Imaging and Graphics* vol. 38, no. 3, pp. 137–150.

Carranza-García, M., Lara-Benítez, P., García-Gutiérrez, J., and Riquelme, J. C. (2021). “Enhancing object detection for autonomous driving by optimizing anchor generation and addressing class imbalance”. In: *Neurocomputing* vol. 449, pp. 229–244.

Chandola, V., Banerjee, A., and Kumar, V. (2009). “Anomaly detection: A survey”. In: *ACM computing surveys (CSUR)* vol. 41, no. 3, pp. 1–58.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* vol. 16, pp. 321–357.

Chawla, N. V., Lazarevic, A., Hall, L. O., and Bowyer, K. W. (2003). “SMOTEBoost: Improving prediction of the minority class in boosting”. In: *European conference on principles of data mining and knowledge discovery*. Springer, pp. 107–119.

Chen, L., Fang, B., Shang, Z., and Tang, Y. (2018). “Tackling class overlap and imbalance problems in software defect prediction”. In: *Software Quality Journal* vol. 26, no. 1, pp. 97–125.

Chen, Z., Yan, Q., Han, H., Wang, S., Peng, L., Wang, L., and Yang, B. (2018). “Machine learning based mobile malware detection using highly imbalanced network traffic”. In: *Information Sciences* vol. 433, pp. 346–364.

Cieslak, D. A., Hoens, T. R., Chawla, N. V., and Kegelmeyer, W. P. (2012). “Hellinger distance decision trees are robust and skew-insensitive”. In: *Data Mining and Knowledge Discovery* vol. 24, no. 1, pp. 136–158.

Claesen, M. and De Moor, B. (2015). “Hyperparameter search in machine learning”. In: *arXiv preprint arXiv:1502.02127*.

Cordón, I., García, S., Fernández, A., and Herrera, F. (2018). “Imbalance: oversampling algorithms for imbalanced classification in R”. In: *Knowledge-Based Systems* vol. 161, pp. 329–341.

Das, B., Krishnan, N. C., and Cook, D. J. (2014). “RACOG and wRACOG: Two probabilistic oversampling techniques”. In: *IEEE transactions on knowledge and data engineering* vol. 27, no. 1, pp. 222–234.

Douzas, G., Bacao, F., and Last, F. (2018). “Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE”. In: *Information Sciences* vol. 465, pp. 1–20.

Dua, D. and Graff, C. (2017). *UCI Machine Learning Repository*.

Elkan, C. (2001). “The foundations of cost-sensitive learning”. In: *International joint conference on artificial intelligence*. Vol. 17. 1. Lawrence Erlbaum Associates Ltd, pp. 973–978.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). “Dermatologist-level classification of skin cancer with deep neural networks”. In: *nature* vol. 542, no. 7639, pp. 115–118.

Fawcett, T. (2004). “ROC graphs: Notes and practical considerations for researchers”. In: *Machine learning* vol. 31, no. 1, pp. 1–38.

— (2006). “An introduction to ROC analysis”. In: *Pattern recognition letters* vol. 27, no. 8, pp. 861–874.

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., and Herrera, F. (2018). *Learning from imbalanced data sets*. Vol. 10. Springer.

Fernández, A., García, S., Herrera, F., and Chawla, N. V. (Jan. 2018). “SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-Year Anniversary”. In: *J. Artif. Int. Res.* vol. 61, no. 1, pp. 863–905.

Fernández, A., López, V., Galar, M., Del Jesus, M. J., and Herrera, F. (2013). “Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches”. In: *Knowledge-based systems* vol. 42, pp. 97–110.

Fernández-Navarro, F., Hervás-Martínez, C., and Gutiérrez, P. A. (2011). “A dynamic over-sampling procedure based on sensitivity for multi-class problems”. In: *Pattern Recognition* vol. 44, no. 8, pp. 1821–1833.

Ferri, C., Hernández-Orallo, J., and Modroiu, R. (2009). “An experimental comparison of performance measures for classification”. In: *Pattern recognition letters* vol. 30, no. 1, pp. 27–38.

Feurer, M. and Hutter, F. (2019). “Hyperparameter optimization”. In: *Automated machine learning*. Springer, Cham, pp. 3–33.

Fürnkranz, J. (2002). “Round robin classification”. In: *The Journal of Machine Learning Research* vol. 2, pp. 721–747.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., and Herrera, F. (2011). “An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes”. In: *Pattern Recognition* vol. 44, no. 8, pp. 1761–1776.

Ganganwar, V. (2012). “An overview of classification algorithms for imbalanced datasets”. In: *International Journal of Emerging Technology and Advanced Engineering* vol. 2, no. 4, pp. 42–47.

García, V., Marqués, A. I., and Sánchez, J. S. (2019). “Exploring the synergetic effects of sample types on the performance of ensembles for credit risk and corporate bankruptcy prediction”. In: *Information Fusion* vol. 47, pp. 88–101.

Goldstein, M. and Dengel, A. (2012). “Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm”. In: *KI-2012: Poster and Demo Track*, pp. 59–63.

Haddad, B. M., Yang, S., Karam, L. J., Ye, J., Patel, N. S., and Braun, M. W. (2018). “Multifeature, Sparse-Based Approach for Defects Detection and Classification in Semiconductor Units”. In: *IEEE Transactions on Automation Science and Engineering* vol. 15, no. 1, pp. 145–159.

Hand, D. J. and Till, R. J. (2001). “A simple generalisation of the area under the ROC curve for multiple class classification problems”. In: *Machine learning* vol. 45, no. 2, pp. 171–186.

Hart, P. (1968). “The condensed nearest neighbor rule (corresp.)” In: *IEEE transactions on information theory* vol. 14, no. 3, pp. 515–516.

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

He, H. and Garcia, E. A. (2009). “Learning from imbalanced data”. In: *IEEE Transactions on knowledge and data engineering* vol. 21, no. 9, pp. 1263–1284.

He, Z., Xu, X., and Deng, S. (2003). “Discovering cluster-based local outliers”. In: *Pattern Recognition Letters* vol. 24, no. 9-10, pp. 1641–1650.

Heft, A. I., Indinger, T., and Adams, N. A. (2012). “Experimental and numerical investigation of the DrivAer model”. In: *ASME 2012 Fluids Engineering Division Summer Meeting*. American Society of Mechanical Engineers Digital Collection, pp. 41–51.

Hinton, G. E. and Roweis, S. (2002). “Stochastic neighbor embedding”. In: *Advances in neural information processing systems* vol. 15.

Ho, T. K. and Basu, M. (2002). “Complexity measures of supervised classification problems”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 3, pp. 289–300.

Imam, T., Ting, K. M., and Kamruzzaman, J. (2006). “z-SVM: An SVM for improved classification of imbalanced data”. In: *Australasian joint conference on artificial intelligence*. Springer, pp. 264–273.

Jo, T. and Japkowicz, N. (June 2004). “Class Imbalances versus Small Disjuncts”. In: *SIGKDD Explor. Newsl.* vol. 6, no. 1, pp. 40–49.

Knupp, P. (2008). “Measurement and Impact of Mesh Quality”. In: *46th AIAA Aerospace Sciences Meeting and Exhibit*, p. 933.

Kong, J., Kowalczyk, W., Menzel, S., and Bäck, T. (2020). “Improving Imbalanced Classification by Anomaly Detection”. In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 512–523.

Kong, J., Kowalczyk, W., Nguyen, D. A., Bäck, T., and Menzel, S. (2019). “Hyperparameter Optimisation for Improving Classification under Class Imbalance”. In: *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, pp. 3072–3078.

Kong, J., Kowalczyk, W., Nguyen, D. A., Menzel, S., and Bäck, T. (2019). “Hyperparameter Optimisation for Improving Classification under Class Imbalance”. In: *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE.

Kong, J., Rios, T., Kowalczyk, W., Menzel, S., and Bäck, T. (2020a). “On the Performance of Oversampling Techniques for Class Imbalance Problems”. In: *24th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) [Accepted]*. Springer.

— (2020b). “On the performance of oversampling techniques for class imbalance problems”. In: *Advances in Knowledge Discovery and Data Mining* vol. 12085, p. 84.

Krawczyk, B. (2016). “Cost-sensitive one-vs-one ensemble for multi-class imbalanced data”. In: *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 2447–2452.

Krawczyk, B., Galar, M., Jeleń, Ł., and Herrera, F. (2016). “Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy”. In: *Applied Soft Computing* vol. 38, pp. 714–726.

Kubat, M., Holte, R., and Matwin, S. (1997). “Learning when negative examples abound”. In: *European conference on machine learning*. Springer, pp. 146–153.

Kubat, M., Holte, R. C., and Matwin, S. (1998). “Machine learning for the detection of oil spills in satellite radar images”. In: *Machine learning* vol. 30, no. 2, pp. 195–215.

Kubat, M., Matwin, S., et al. (1997). “Addressing the curse of imbalanced training sets: one-sided selection”. In: *Icml*. Vol. 97. 1. Citeseer, p. 179.

Lango, M. and Stefanowski, J. (2018). “Multi-class and feature selection extensions of roughly balanced bagging for imbalanced data”. In: *Journal of Intelligent Information Systems* vol. 50, no. 1, pp. 97–127.

Last, M. (2002). “Online classification of nonstationary data streams”. In: *Intelligent data analysis* vol. 6, no. 2, pp. 129–147.

Laurikkala, J. (2001). “Improving identification of difficult small classes by balancing class distribution”. In: *Conference on artificial intelligence in medicine in Europe*. Springer, pp. 63–66.

Lee, T., Lee, K. B., and Kim, C. O. (2016). “Performance of machine learning algorithms for class-imbalanced process fault detection problems”. In: *IEEE Transactions on Semiconductor Manufacturing* vol. 29, no. 4, pp. 436–445.

Lertampaiporn, S., Thammarongtham, C., Nukoolkit, C., Kaewkamnerdpong, B., and Ruengjitchatchawalya, M. (2013). “Heterogeneous ensemble approach with discriminative features and modified-SMOTEbagging for pre-miRNA classification”. In: *Nucleic acids research* vol. 41, no. 1, e21–e21.

Li, J., Liu, L.-s., Fong, S., Wong, R. K., Mohammed, S., Fiaidhi, J., Sung, Y., and Wong, K. K. (2017). “Adaptive Swarm Balancing Algorithms for rare-event prediction in imbalanced healthcare data”. In: *PloS one* vol. 12, no. 7, e0180830.

Liao, T. W. (2008). “Classification of weld flaws with imbalanced class data”. In: *Expert Systems with Applications* vol. 35, no. 3, pp. 1041–1052.

Liu, B. and Tsoumakas, G. (2019). “Synthetic oversampling of multi-label data based on local label distribution”. In: *arXiv preprint arXiv:1905.00609*.

Livesu, M., Vining, N., Sheffer, A., Gregson, J., and Scateni, R. (2013). “PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction”. In: *Transactions on Graphics (Proc. SIGGRAPH ASIA 2013)* vol. 32, no. 6.

López, V., Fernández, A., García, S., Palade, V., and Herrera, F. (2013). “An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics”. In: *Information sciences* vol. 250, pp. 113–141.

López, V., Fernández, A., Moreno-Torres, J. G., and Herrera, F. (2012). "Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics". In: *Expert Systems with Applications* vol. 39, no. 7, pp. 6585–6608.

Lorena, A. C., Garcia, L. P., Lehmann, J., Souto, M. C., and Ho, T. K. (2018). "How Complex is your classification problem? A survey on measuring classification complexity". In: *arXiv preprint arXiv:1808.03591*.

Lorena, A. C., Garcia, L. P., Lehmann, J., Souto, M. C., and Ho, T. K. (2019). "How Complex Is Your Classification Problem?: A Survey on Measuring Classification Complexity". In: *ACM Computing Surveys (CSUR)* vol. 52, no. 5, p. 107.

Luengo, J., Fernández, A., García, S., and Herrera, F. (2011). "Addressing data complexity for imbalanced data sets: analysis of SMOTE-based oversampling and evolutionary undersampling". In: *Soft Computing* vol. 15, no. 10, pp. 1909–1936.

Lusa, L. et al. (2015). "Joint use of over-and under-sampling techniques and cross-validation for the development and assessment of prediction models". In: *BMC bioinformatics* vol. 16, no. 1, p. 363.

Mahalanobis, P. C. (1936). "On the generalized distance in statistics". In: National Institute of Science of India.

Malina, W. (2001). "Two-parameter Fisher criterion". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* vol. 31, no. 4, pp. 629–636.

Mazurowski, M. A., Habas, P. A., Zurada, J. M., Lo, J. Y., Baker, J. A., and Tourassi, G. D. (2008). "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance". In: *Neural networks* vol. 21, no. 2-3, pp. 427–436.

Menzel, S., Olhofer, M., and Sendhoff, B. (2005). "Application of Free Form Deformation Techniques in Evolutionary Design Optimisation". In: *6th World Congress on Structural and Multidisciplinary Optimization (WCSMO6)*. Ed. by Herskovits, J., Mazorche, S., and Canelas, A. Rio de Janeiro: COPPE Publication.

Menzel, S. and Sendhoff, B. (2008). "Representing the Change - Free Form Deformation for Evolutionary Design Optimization". In: *Evolutionary Computation in Practice*. Springer Berlin Heidelberg, pp. 63–86.

Misra, R., Wan, M., and McAuley, J. (2018). "Decomposing Fit Semantics for Product Size Recommendation in Metric Spaces". In: *Proceedings of the 12th*

ACM Conference on Recommender Systems. RecSys ’18. Vancouver, British Columbia, Canada: Association for Computing Machinery, pp. 422–426.

Napierala, K. and Stefanowski, J. (2016). “Types of minority class examples and their influence on learning classifiers from imbalanced data”. In: *Journal of Intelligent Information Systems* vol. 46, no. 3, pp. 563–597.

Napierała, K., Stefanowski, J., and Wilk, S. (2010). “Learning from imbalanced data in presence of noisy and borderline examples”. In: *International conference on rough sets and current trends in computing*. Springer, pp. 158–167.

Neogi, N., Mohanta, D. K., and Dutta, P. K. (2014). “Review of vision-based steel surface inspection systems”. In: *EURASIP Journal on Image and Video Processing* vol. 2014, no. 1, pp. 1–19.

Nguyen, D. A., Kong, J., Wang, H., Menzel, S., Sendhoff, B., Kononova, A. V., and Bäck, T. (2021). “Improved automated cash optimization with tree parzen estimators for class imbalance problems”. In: *2021 IEEE 8th international conference on data science and advanced analytics (DSAA)*. IEEE, pp. 1–9.

Nguyen, H. M., Cooper, E. W., and Kamei, K. (2011). “Online learning from imbalanced data streams”. In: *2011 International Conference of Soft Computing and Pattern Recognition (SoCPaR)*. IEEE, pp. 347–352.

Olhofer, M., Bihrer, T., Menzel, S., Fischer, M., and Sendhoff, B. (2009). “Evolutionary Optimisation of an Exhaust Flow Element with Free Form Deformation”. In: *4th European Automotive Simulation Conference, Munich*.

Orriols-Puig, A. and Bernadó-Mansilla, E. (2009). “Evolutionary rule-based systems for imbalanced data sets”. In: *Soft Computing* vol. 13, no. 3, pp. 213–225.

Orriols-Puig, A., Macia, N., and Ho, T. K. (2010). “Documentation for the data complexity library in C++”. In: *Universitat Ramon Llull, La Salle* vol. 196, pp. 1–40.

Prati, R. C., Batista, G. E., and Monard, M. C. (2004). “Class imbalances versus class overlapping: an analysis of a learning system behavior”. In: *Mexican international conference on artificial intelligence*. Springer, pp. 312–321.

Radtke, P. V., Granger, E., Sabourin, R., and Gorodnichy, D. O. (2014). “Skew-sensitive boolean combination for adaptive ensembles – An application to face recognition in video surveillance”. In: *Information Fusion* vol. 20, pp. 31–48.

Ren, F., Cao, P., Li, W., Zhao, D., and Zaiane, O. (2017). “Ensemble based adaptive over-sampling method for imbalanced data learning in computer aided detection of microaneurysm”. In: *Computerized Medical Imaging and Graphics* vol. 55, pp. 54–67.

Rifkin, R. and Klautau, A. (2004). “In defense of one-vs-all classification”. In: *The Journal of Machine Learning Research* vol. 5, pp. 101–141.

Rodriguez, D., Herraiz, I., Harrison, R., Dolado, J., and Riquelme, J. C. (2014). “Preliminary comparison of techniques for dealing with imbalance in software defect prediction”. In: *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, pp. 1–10.

Rokach, L. (2010). “Ensemble-based classifiers”. In: *Artificial intelligence review* vol. 33, no. 1, pp. 1–39.

Sáez, J. A., Krawczyk, B., and Woźniak, M. (2016). “Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets”. In: *Pattern Recognition* vol. 57, pp. 164–178.

Santos, M. S., Soares, J. P., Abreu, P. H., Araujo, H., and Santos, J. (2018). “Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches [research frontier]”. In: *ieee ComputatioNal iNtelligeNCe magaziNe* vol. 13, no. 4, pp. 59–76.

Schaffer, J. (2015). “What not to multiply without necessity”. In: *Australasian Journal of Philosophy* vol. 93, no. 4, pp. 644–664.

Sederberg, T. W. and Parry, S. R. (1986). “Free-form deformation of solid geometric models”. In: *ACM SIGGRAPH computer graphics* vol. 20, no. 4, pp. 151–160.

Sen, A., Islam, M. M., Murase, K., and Yao, X. (2015). “Binarization with boosting and oversampling for multiclass classification”. In: *IEEE transactions on cybernetics* vol. 46, no. 5, pp. 1078–1091.

Shekar, B. and Dagnew, G. (2019). “Grid search-based hyperparameter tuning and classification of microarray cancer data”. In: *2019 second international conference on advanced computational and communication paradigms (ICACCP)*. IEEE, pp. 1–8.

Sieger, D., Menzel, S., and Botsch, M. (2015). “On shape deformation techniques for simulation-based design optimization”. In: *New Challenges in Grid Generation and Adaptivity for Scientific Computing*. Springer, pp. 281–303.

Sinclair, D. (2016). “S-hull: a fast radial sweep-hull routine for Delaunay triangulation”. In: *arXiv preprint arXiv:1604.01428v1 [cs.CG]*.

Skryjomski, P. and Krawczyk, B. (Sept. 2017). “Influence of minority class instance types on SMOTE imbalanced data oversampling”. In: *Proceedings of the First*

International Workshop on Learning with Imbalanced Domains: Theory and Applications. Ed. by Luís Torgo, P. B. and Moniz, N. Vol. 74. Proceedings of Machine Learning Research. PMLR, pp. 7–21.

Sleeman IV, W. C. and Krawczyk, B. (2021). “Multi-class imbalanced big data classification on Spark”. In: *Knowledge-Based Systems* vol. 212, p. 106598.

Soofi, A. A. and Awan, A. (2017). “Classification techniques in machine learning: applications and issues”. In: *Journal of Basic & Applied Sciences* vol. 13, pp. 459–465.

Sun, Y., Kamel, M. S., and Wang, Y. (2006). “Boosting for learning multiple classes with imbalanced class distribution”. In: *Sixth international conference on data mining (ICDM’06)*. IEEE, pp. 592–602.

Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. (2007). “Cost-sensitive boosting for classification of imbalanced data”. In: *Pattern recognition* vol. 40, no. 12, pp. 3358–3378.

Tan, A. C., Gilbert, D., and Deville, Y. (2003). “Multi-class protein fold classification using a new ensemble machine learning approach”. In: *Genome Informatics* vol. 14, pp. 206–217.

Thai-Nghe, N., Busche, A., and Schmidt-Thieme, L. (2009). “Improving academic performance prediction by dealing with class imbalance”. In: *2009 Ninth International Conference on Intelligent Systems Design and Applications*. IEEE, pp. 878–883.

Thai-Nghe, N., Gantner, Z., and Schmidt-Thieme, L. (2010). “Cost-sensitive learning methods for imbalanced data”. In: *The 2010 International joint conference on neural networks (IJCNN)*. IEEE, pp. 1–8.

Tomek, I. (1976). “Two modifications of CNN”. In: *IEEE Trans. Systems, Man and Cybernetics* vol. 6, pp. 769–772.

Van den Oord, A., Dieleman, S., and Schrauwen, B. (2013). “Deep content-based music recommendation”. In: *Advances in neural information processing systems* vol. 26.

Van der Maaten, L. and Hinton, G. (2008). “Visualizing data using t-SNE.” In: *Journal of machine learning research* vol. 9, no. 11.

Wang, B. X. and Japkowicz, N. (2010). “Boosting support vector machines for imbalanced data sets”. In: *Knowledge and information systems* vol. 25, no. 1, pp. 1–20.

Wang, S. (2011a). “Ensemble diversity for class imbalance learning”. PhD thesis. University of Birmingham.

— (2011b). “Ensemble diversity for class imbalance learning”.

Wang, S., Chen, H., and Yao, X. (2010). “Negative correlation learning for classification ensembles”. In: *The 2010 international joint conference on neural networks (IJCNN)*. IEEE, pp. 1–8.

Wang, S., Minku, L. L., and Yao, X. (2014). “Resampling-based ensemble methods for online class imbalance learning”. In: *IEEE Transactions on Knowledge and Data Engineering* vol. 27, no. 5, pp. 1356–1368.

— (2016). “Dealing with Multiple Classes in Online Class Imbalance Learning.” In: *IJCAI*, pp. 2118–2124.

— (2018). “A systematic study of online class imbalance learning with concept drift”. In: *IEEE transactions on neural networks and learning systems* vol. 29, no. 10, pp. 4802–4821.

Wang, S. and Yao, X. (2009). “Diversity analysis on imbalanced data sets by using ensemble models”. In: *2009 IEEE symposium on computational intelligence and data mining*. IEEE, pp. 324–331.

— (2012). “Multiclass imbalance problems: Analysis and potential solutions”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* vol. 42, no. 4, pp. 1119–1130.

Weng, C. G. and Poon, J. (2006). “A data complexity analysis on imbalanced datasets and an alternative imbalance recovering strategy”. In: *2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings)(WI’06)*. IEEE, pp. 270–276.

Wilson, D. R. and Martinez, T. R. (1997). “Improved heterogeneous distance functions”. In: *Journal of artificial intelligence research* vol. 6, pp. 1–34.

Wilson, D. L. (1972). “Asymptotic properties of nearest neighbor rules using edited data”. In: *IEEE Transactions on Systems, Man, and Cybernetics*, no. 3, pp. 408–421.

Wu, Z., Lin, W., and Ji, Y. (2018). “An integrated ensemble learning model for imbalanced fault diagnostics and prognostics”. In: *IEEE Access* vol. 6, pp. 8394–8402.

Zhang, H. and Li, M. (2014). “RWO-Sampling: A random walk over-sampling approach to imbalanced data classification”. In: *Information Fusion* vol. 20, pp. 99–116.

Zhang, X., Zhuang, Y., Wang, W., and Pedrycz, W. (2016). “Transfer boosting with synthetic instances for class imbalanced object recognition”. In: *IEEE transactions on cybernetics* vol. 48, no. 1, pp. 357–370.

— (2018). “Transfer Boosting With Synthetic Instances for Class Imbalanced Object Recognition”. In: *IEEE Transactions on Cybernetics* vol. 48, no. 1, pp. 357–370.

Zhao, Y., Nasrullah, Z., and Li, Z. (2019). “Pyod: A python toolbox for scalable outlier detection”. In: *arXiv preprint arXiv:1901.01588*.

Zhu, L., Lu, C., Dong, Z. Y., and Hong, C. (2017). “Imbalance learning machine-based power system short-term voltage stability assessment”. In: *IEEE Transactions on Industrial Informatics* vol. 13, no. 5, pp. 2533–2543.

Zięba, M., Tomczak, J. M., Lubicz, M., and Świątek, J. (2014). “Boosted SVM for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients”. In: *Applied soft computing* vol. 14, pp. 99–108.

Samenvatting

Het ongebalanceerde klasse probleem is een uitdagende classificatie probleem en komt vaak voor in de praktijk in dagelijkse toepassingen. Er zijn verschillende technieken ontwikkeld om de onevenwichtige classificatieprestaties theoretisch en praktisch te verbeteren. Naast het ontwikkelen van nieuwe methodes, richten onderzoekers zich ook op het belang van het begrijpen van de data zelf, wat meer inzicht zal geven in wat de ongebalanceerde klasse prestaties daadwerkelijk belemmert.

In dit proefschrift is onderzoek gedaan naar het leren van ongebalanceerde klasse problemen vanuit het perspectief van data-intrinsieke kenmerken. Het empirische onderzoek waarbij verschillende algoritmen op data niveau werden vergeleken, toont aan dat over-sampling-benaderingen, rekening houdend met de minderheids-klasse-verdeling, in de meeste gevallen betere ongebalanceerde classificatieprestaties kunnen opleveren. Hoewel data complexe metingen geen richtlijn kunnen geven over de keuze van re-sampling technieken, vinden we dat de potentieel beste AUC-waarde kan worden voorspeld door de F1v-meting (de Directional-vector Maximum Fisher's Discriminant Ratio). Beide conclusies worden ook geverifieerd op een op de praktijk geïnspireerde voertuig mesh dataset van het Honda Research Institute.

Optimalisatie van hyperparameters is zeer effectief gebleken voor veel classificatiealgoritmen voor machine learning. De maximale diepte van de boom en het minimale aantal samples dat nodig is intern knooppunt te splitsen, zijn bijvoorbeeld cruciaal voor het afstemmen van de beslissingsboom om de beste prestaties te bereiken. we benadrukken daarom het belang van afstemming van hyperparameters voor benaderingen op data niveau.

Het afwijkingsdetectieprobleem is een ongebalanceerd-klasse-probleem met een extreem onevenwichtige verhouding. Technieken voor afwijkingsdetectieprobleem kunnen worden toegepast op ongebalanceerde-klasse problemen met nauwkeurige afstelling. In dit proefschrift stellen we voor om de Local Outlier Score te introduceren, wat een belangrijke indicator om te evalueren of een steekproef een outlier is, als een extra toepassing van de originele ongebalanceerde dataset. Dit voorstel is meer dan het lenen van kennis uit afwijkingsdetectie onderzoeksveld, maar geeft onderzoekers ook de mogelijkheid om inzicht te krijgen in de data in plaats van te over- of onder-samplen.

In het laatste deel van het proefschrift wordt een verbeterde sampling type identificatie voorgesteld voor het omgaan met ongebalanceerde classificatie met meerdere klassen en toegepast op een dataset uit de praktijk voor oppervlaktedefecten van TATA Steel. Ondertussen gaan we in op het belang van het begrijpen van de verschillende intrinsieke gegevenskenmerken voor binaire scenario's en scenario's met meerdere klassen.

Summary

The class-imbalance problem is a challenging classification task and is frequently encountered in real-world applications. Various techniques have been developed to improve the imbalanced classification performance theoretically and practically. Apart from developing new approaches, researchers also address the importance of understanding the data itself, which will provide more insight into what actually hinders the imbalanced classification performance.

This thesis conducted research on Learning Class-Imbalanced Problem from the perspective of Data Intrinsic Characteristics. The empirical investigation comparing several data-level algorithms shows that oversampling approaches considering the minority class distribution can provide better imbalanced classification performance in most cases. Although data complexity measures cannot provide any guidance on the choice of resampling techniques, we find the potential best AUC value can be predicted by the F1v measure (the Directional-vector Maximum Fisher's Discriminant Ratio). Both conclusions are also verified on a real-world inspired vehicle mesh dataset from Honda Research Institute

Hyperparameter optimisation has shown great effectiveness for many machine learning classification algorithms. For example, the maximum depth of the tree and the minimum number of samples required to split an internal node are critical for tuning the Decision Tree to achieve the best performance. Therefore, we emphasize the importance of hyperparameter tuning for data-level approaches.

The anomaly detection problem is a class-imbalance problem with an extreme imbalanced ratio. Techniques for anomaly detection problems can be applied to class-imbalanced problems with fine adjustment. In this thesis, we propose to introduce the Local Outlier Score, which is an important indicator to evaluate whether a sample is an outlier, as an additional attribute of the original imbalanced

dataset. This proposal is more than borrowing the knowledge from anomaly detection research field but also provides researchers with another possibility to acquire more insight from the data rather than undersampling/oversampling.

In the final part of the thesis, an improved sample type identification is proposed for dealing with multi-class imbalanced classification and applied on a real-world surface defects dataset from TATA Steel. Meanwhile, we address the importance of understanding the different data intrinsic characteristics for binary and multi-class scenarios.

Curriculum Vitae

Jiawen (Fay) Kong was born in Harbin, China in 1995. She had lived and studied in this northeastern city until she concluded her high-school degree in 2013. After that, she went to Shanghai to study Statistics and received her bachelor's degree in Applied Statistics in 2017. With the support of her parents, she then continued her study in Statistics in the UK and received her master's degree in Statistics in 2018. Fay worked as a PhD candidate under the supervision of Prof.dr. Thomas Bäck, Prof.dr. Bernhard Sendhoff and Dr. Wojtek Kowalczyk since October 2018. Her research interests include class-imbalance problem, anomaly detection and statistics. She was a member of the ECOLE (Experience-based Computation: Learning to Optimise) project. ECOLE is a Marie Curie ITN (Innovative Training Network) project, which provided her with very good opportunities to work as an early-stage researcher in Honda Research Institute, Germany and TATA Steel, the Netherlands.