Journal of Data Intelligence and Predictive Systems (JDIPS) **2025, 1(1): 1-7**
https://www.longmanpublishers.com/

Combining metric learning with multi-ratio undersampling to achieve imbalanced classification

Hinata
Tokai Gakuen University, Miyoshi, Aichi, Japan

Abstract: One problem that impairs the classification performance of several classification techniques is class imbalance. One often used method to address the class imbalance is resampling; however, it still has a limited data space, which further impairs performance. This research proposes MMEnsemble, an undersampling-based unbalanced classification framework that integrates metric learning into a multi-ratio undersampling-based ensemble, as a solution to this problem. Determining the proper sample ratio in the multi-ratio ensemble approach is another issue that this system resolves. Twelve real-world datasets were used to assess it. In terms of recall and ROC-AUC, it surpassed the state-of-the-art methods of metric learning, undersampling, and oversampling; in terms of Gmean and F-measure metrics, it performed similarly to them.
Keywords: Imbalanced classification, Undersampling, Ensemble, Metric learning.

1. Introduction

In real-world applications, class imbalance [15] is a significant issue that impairs classification performance, particularly when minority classes are involved. When there are significantly more examples in one class than in other classes, this is referred to as class imbalance in datasets. Classifiers are biased toward the majority class due to the significant disparity in the number of examples. Class imbalance has been noted and addressed in a number of fields, including computer networks [12], software engineering [26], and the fields of medicine [7], economics [25], and agriculture [28].

Resampling has been extensively researched as a successful remedy for class imbalance [8, 19, 5, 6]. Resampling methods can be broadly divided into two groups: undersampling (e.g., EasyEnsemble [19] and RUSBoost [27]) and oversampling (e.g., SMOTE [8] and SWIM [6]).
A straightforward and effective resampling method for addressing class imbalance is undersampling [10]. The problem has been solved by integrating multiple undersampled datasets in an ensemble fashion in addition to employing single-shot undersampling [19, 16, 27].

According to Figure 1, a preliminary analysis of multiple datasets on the effects of various undersampling ratios revealed that preferences for classes vary depending on the undersampling ratio. The ratio of the sampled majority size to the minority size is known as the sampling ratio. The majority class is considered the negative class in this study, whereas the minority class is considered the positive class. The majority examples are chosen at random when the sampling ratio is 1.0, ensuring that the number of sampled examples is equal to that of the minority examples. When the ratio is less than 1.0, it indicates that there are fewer sampled majority cases than minority ones. This is referred to as severe undersampling in this work, while moderate undersampling is its opposite. The Abalone dataset's true positive and negative ratios for various sample ratios are shown in the figure. It suggests that 1.0 might not be the optimal balanced ratio because classifiers trained with severely undersampled datasets favor the minority class, while those trained with moderately undersampled datasets favor the majority class.

This study presents MMEnsemble, a unique undersampling-based ensemble framework consisting of asset-based weighting, multi-ratio ensemble, and metric learning.

• **Metric Learning:** LMNN [34] and other metric learning techniques learn a data transformation that makes it possible to identify instances in various classes. Selecting subsets of training examples for metric learning enhances the classification performance of an imbalanced classification, according to recent metric learning algorithms [33]. This concept serves as the foundation for MMEnsemble, which integrates metric learning into an ensemble that relies on undersampling.
• **Multi-ratio Ensemble:** The sampling ratio is a crucial parameter when using undersampling. It establishes how many drawn majority cases there are. According to a recent study [17], the classification performance is enhanced when multiple sample ratios are used in an ensemble fashion.
• **Asset-based Weighting:** Weak classifiers with varying sampling ratios have varying assets when using an ensemble based on multi-ratio undersampling. The majority class may be accurately classified by a classifier with a high sampling ratio, whereas the minority class may be correctly classified by a classifier with a small

sample ratio. MMEnsem-ble presents a weighting strategy that prioritizes classifiers that are able to accurately identify instances that are challenging for the other classifiers to categorize in order to capture the assets. The following is a summary of this paper's contributions.

The innovative framework known as MMEnsemble: The architecture known as MMEnsemble is made up of asset-based weighting, multi-ratio undersampling-based ensemble, and met-ric learning. By using an ensemble of base classifiers in different sampling ratios and automatic weighting schemes, this framework eliminates the drawback of metric learning with regard to class imbalance by applying undersampling beforehand. It also relieves users of the burden of selecting sampling ratios in undersampling.

• **Better Classification Performance:** MMEnsemble surpasses the state-of-the-art methods in an experiment utilizing 12 real-world datasets, particularly for recall and ROC-AUC metrics, and it performs similarly to them on Gmean and F-measure metrics. Higher recall scores can be obtained using this method, according to this experiment, which would be helpful for a variety of real-world applications.
This is how the remainder of the paper is structured. The relevant work on resampling-based techniques is introduced in Section 2. MMEnsemble is thoroughly explained in Section 3, and the experimental evaluation utilizing 12 real-world datasets is presented in Section 4. This paper is finally concluded in Section 5.

2. Related Work: Resampling Approaches

The three main categories of methods for addressing class imbalance are algorithm change [32], cost-adjustment [9], and resampling. Because it has demonstrated reliable performance and is applicable to all classifiers, resampling is frequently utilized. Oversampling and undersampling are the two broad categories into which sampling techniques can be divided.

2.1. Methods Based on Oversampling

Copying minority examples at random to equal the number of minority and majority examples is a straightforward oversampling technique. This method is prone to over-fitting. Oversampling techniques produce artificial minority instances that are near the minority in order to address the overfitting issue. The most often used synthetic oversampling technique is SMOTE [8]. It uses the nearest neighbor technique to create artificial minority examples. The generated examples may readily overlap with majority examples because SMOTE does not take majority examples into account. The categorization performance suffers as a result. More recent methods have included majority instances to the resampling procedure in order to get over SMOTE's drawback. Data purification methods used by SMOTE-Tomek [4] and SMOTE-ENN [5] include Edit Nearest Neighbours [35] and Tomek link removal [30]. Similarly, there are more sophisticated methods (e.g., SVM-SMOTE [23], borderline-SMOTE [13], and ADASYN [14]). SWIM is one of the most advanced synthetic oversampling techniques [6]. To create synthetic minority instances, SWIM makes use of each minority example's density in relation to the distribution of majority examples. Extensive studies were carried out by [18] to examine a wide range of SMOTE variations and compare them with various dataset types. ProWSyn [2] and PolyFitSMOTE [11] performed the best in this experiment.

2.2. Techniques Based on Undersampling

Three types of undersampling-based methods can be distinguished: boosting and bagging ensembles, as well as example selection. The process of selecting majority examples that are anticipated to improve categorization is known as example selection. Major techniques select examples of the majority that are difficult to differentiate from instances of the minority. Majority examples that are near minority instances are sampled by NearMiss [22]. A hardness property called instance hardness [29] shows how likely it is that an example would be incorrectly labeled.

A learning technique called "boosting ensemble" progressively modifies the bulk of samples. Boosting approaches eliminate a portion of the majority samples for every iteration. A boosting technique called BalanceCascade [19] eliminates majority samples that are accurately classified. A weighted random undersampling technique called RUS-Boost [27] is used to exclude majority examples that are probably going to be correctly identified. To enhance model performance, EUSBoost [21] presents an adaptive boundary choice technique and a cost-sensitive weight change. The best in this category is Trainable Undersampling [24]. It uses reinforcement learning to train a classifier.

Multiple weak classifiers are combined in a bagging ensemble, each of which is trained in a voting fashion on separate pieces of undersampled training data. One of the earliest bagging techniques that used undersampled training data was Ensemble of Undersampling [16]. An ensemble-of-ensemble method called EasyEnsemble

[19] groups AdaBoost classifiers for every piece of undersampled training data in a bagging fashion. A thorough experiment on bagging and boosting strategies is documented in [10]. It demonstrates that the top-performing methods are RUSBoost and EasyEnsemble, which outperform methods based on oversampling. Multiple sample ratios are incorporated into EasyEnsemble by MUEnsemble [17]. The most advanced bagging-based method that considers cost-sensitive learning and metric learning is called DDAE [36].

MMEnsemble, the suggested approach, falls under the bagging category. The fact that MMEnsemble uses metric learning to get over the problem of limited data spaces in resampling techniques sets it apart from the others (apart from DDAE). DDAE and MMEnsemble differ in two important ways. One is the control of undersampling (also known as data block); DDAE undersamples based on the number of blocks, independent of the imbalance ratio of datasets, whereas MMEnsemble undersamples data based on the sampling ratio. The other is the weak classifier selection; MMEnsemble employs the AdaBoost classifier, whereas DDAE utilizes the closest neighbor classifier, which is thought to fit metric learning. It's also crucial to note that MMEnsemble only has one hyper-parameter, which is a significantly smaller parameter space than DDAE, which has (at least) three.

Regarding the multi-ratio ensemble, MUEnsemble employs heuristic weighting (i.e., weighting based on Gaussian functions), whereas MMEnsemble uses weak classifier assets that are derived from the process of verifying weak classifiers. Therefore, extensive hyper-parameter adjustment is needed to capture the features of poor classifiers. The asset-based weighting system outperforms the heuristic weighting in MUEnsemble, according to the experimental evaluation presented in this study.

3. MMEnsemble

Complex The three phases of MMEnsemble—the multi-ratio ensemble phase, the metric learning phase, and the multi-ratio undersampling phase—are depicted in Figure 2. MUEnsemble [17] is replicated in the first step, which involves selecting multiple undersampled sets of examples with ri from the training data for each sampling ratio ri ∈ R (where R is a preset collection of sampling ratios). In the second stage, metric learning is carried out for every drawn set, and this altered drawn set is used to train a base ensemble classifier known as MLEnsemble. Using asset-based weighting, the final ensemble classifier is built in the final phase using the |R| basis classifiers from the previous phase. The technical specifications of MLEnsemble and the ensemble with asset-based weighting are presented in the sections that follow.

3.1. MLEnsemble, the Base Ensemble Classifier

MLEnsemble is a metric learning bagging classifier. Algorithm 1 provides a summary of its process. To get specific groups of examples, the training data are sampled several times with replacement (Line 2). Each set is used to train a metric learner, which turns the set into a sufficient data space for differentiating instances of various classes (Lines 3-4). A weak classifier is trained using the altered set (Line 5).

3.2. Use asset-based weighting to assemble

Typically, the weighted voting mechanism is employed in ensemble methods. These techniques frequently give all base classifiers the same weights and are unaware of class imbalance. Conversely, the weights of base classifiers must be carefully designed because they are more sensitive in the case of an ensemble of base classifiers with varying sampling ratios. [17] demonstrated the superiority of heuristic weighing with a Gaussian function over equal weighting.

4. Evaluation of Experiments

MMEnsemble was assessed in this experiment to provide answers to the following queries.
Q1: Does MMEnsemble perform better than the most advanced imbalanced classification techniques of oversampling, undersampling, and metric learning?
Q2: Does the combination of multi-ratio ensemble and metric learning work well?
Q3: Does the categorization performance get better with asset-based weighting? and how does the selection of its hyper-parameter k (Equation 2) affect things?

4.1. Configuration

Datasets: The OpenML dataset [31] and the KEEL repository [1] provided the datasets used in the experiment. The total number of records (#records), minority instances (#minor), dimensionality (#dim), and imbalance ratio (IR), which is #major, are all displayed in Table 1. The OpenML dataset included D1–D6, whereas the KEEL repository provided the remaining data.

Table 1. Datasets

ID	Name	#records	#minor	#dim	IR
D1	cm1	498	49	21	9.2
D2	kc3	458	43	39	9.7
D3	mw1	403	31	37	12.0
D4	pc1	1,109	77	21	13.4
D5	pc3	1,563	160	37	8.8
D6	pc4	1,458	178	37	7.2
D7	yeast1-7	459	30	7	14.3
D8	abalone9-18	731	42	8	16.4
D9	yeast6	1,484	35	8	41.4
D10	abalone19	4,174	32	8	129.4
D11	wine3-5	691	10	11	68.1
D12	abalone20	1,916	26	8	72.7

Evaluation: Recall, Gmean, F2, and AUC were the evaluation measures. The true positives, false negatives, true negatives, and false positives are represented by the letters TP, FN, TN, and FP. The number of positive (minority) cases that are correctly classified is measured by recall = TP. Gmean is equal to -TP+FN.
The geometric mean of the recalls for both classes is Recall · TNR, where TNR = TN.

F = (1+β2)Precision = T P, and β establishes the weight on the recall. Recall Precision is the harmonic mean of recall and precision. Since higher recalls are favored in many real-world applications, β was set to 2 in this experiment. The receiver operation characteristic curve's area under the curve is known as the AUC.
The experimental procedure was carried out 50 times in order to accurately estimate these evaluation metric values. The classifiers were trained on the training set and assessed on the test set after a dataset was randomly divided into 70% for training and 30% for testing. The macro average of the 50 trials served as the overall metric scores.

Baseline Methods: MMEnsemble was contrasted with the most advanced techniques for ensemble approach, metric learning, and resampling (both oversampling and undersampling). A cutting-edge method of metric learning that addresses class imbalance is IML [33]. To enhance the data transformation, IML iteratively chooses training samples and integrates LMNN [34]. Based on the thorough experiment in [18] and an initial assessment of the datasets in this experiment, ProWSyn [2] was chosen for the oversampling strategy for resampling. The state-of-the-art for the ensemble and undersampling method is DDAE [36], which also incorporates metric learning. The results of IML and DDAE were taken from the DDAE paper [36] because this experiment uses the same dataset as DDAE. However, in [36], MWMOTE [3] is chosen as the state-of-the-art oversampling method, [18], and the initial evaluation demonstrated ProWSyn's superiority.

MMEnsemble was compared to EasyEnsemble [20], MUEnsemble [17], and MLEnsemble (in this article), which are metric learning including EasyEnsemble, a multi-ratio undersampling-based ensemble, and an undersampling-based ensemble, respectively, in order to address Q2. The distinction between MMEnsemble and EasyEnsemble demonstrates the advantages of combining multi-ratio ensemble and metric learning. Likewise, the distinction between MMEnsemble and MUEnsemble demonstrates the advantage of metric learning in enhancing performance for the unbalanced classification. ProWSyn, EasyEnsemble, MLEnsemble, and MUEnsemble parameters were specified. In ProWSyn, EasyEnsemble, and MLEnsemble, the sampling ratio was set to 1.0. LMNN was used as the metric learning technique, and its k value was set to 3. Gaussian weighting was used with parameters, μ and σ2, of 1.0 and 0.2 for MUEnsemble. The predefined set R of sampling ratios is set to {0.2, 0.4,..., 2.0}, and the best σ2 was experimentally explored from {0.1, 0.2,..., 1.0}. As with the previous methods, μ was fixed to 1.0. R is the same as MUEnsemble, k of the asset-based weighting was selected from {0.1, 0.2,..., 5.0}, and the base classifier, MLEnsemble, was established as previously for MMEnsemble.

4.2. Findings

The experimental results are presented from three angles in order to address the questions: a comparison over the k parameter and different weighting schemes (corr. Q3), an overall comparison (corr. Q1), and the ablation study (corr. Q2).

4.2.1. Comprehensive Analysis

The metric scores of MMEnsemble using the most advanced techniques are displayed in Table 2. The highest scores in a row are bolded in the table. In terms of recall and AUC, MMEnsemble completely outperformed IML

and ProWSyn, and it was comparable to DDAE in terms of Gmean and F2 metrics. Notably, MMEnsemble nearly exceeded the others on the recall metric and completely outperformed them on the AUC metric. This dominance of MMEnsemble is practically useful because a high recall is preferred for real-world applications. Conversely, the F2 and Gmean scores were similar to DDAE. MMEnsemble clearly beat DDAE on datasets D5, D6, D8, D9, and D11; on the remaining datasets, MMEnsemble was either equivalent to or worse than DDAE. The poor TNR and precision scores for MMEnsemble, which resulted from the asset-based weighting in the weighting scheme design, were the reason of this. The purpose of asset-based weighting is to highlight the basic classifiers that accurately categorize cases that others are unable to. This raises the possibility that there will be more false positives.

4.2.2. The Combination's Effect

A comparison of MMEnsemble and its fundamental methodologies is presented in Table 3. The comparisons between EasyEnsemble and MLEnsemble and MUEnsemble demonstrate that the addition of either the multi-ratio ensemble or the metric learning could somewhat improve the classification performance. The architectural distinction between MUEnsemble and MMEnsemble is whether or not metric learning is applied; as a result, MMEnsemble was integrated with Gaussian weighting (Equation 1) in order to examine the performance increase brought about by the difference. This comparison demonstrated MMEnsemble's superiority over MUEnsemble, i.e., the metric learning effectively enhanced the data space in the data sets for every sampling ratio. Furthermore, MMEnsemble with the asset-based weighting performed better than MUEnsemble with the Gaussian weighting, as can be observed by comparing the columns of MMEnsemble in Table 2 and Table 3.

4.2.3. The Impact of Weighting Based on Assets

The impact of the hyper-parameter k on the asset-based weighting is depicted in Figure 3. The primary conclusion is that as k grew, the recall scores decreased. This is due to the fact that the basis classifiers that are able to correctly identify examples that the other base classifiers have misclassified are assigned larger weights as k increases. Higher TNR and precision result from this; so, Gmean and F2 scores rise as k grows, and AUC values progressively rise as well.

A comparison between uniform weighting and asset-based weighting is presented in Table 4. All base classifiers received the same weights thanks to uniform weighting. Uniformly weighted MMEnsemble tended to have low results for the other metrics but good recall scores. This suggests that the number of cases classified to the minority class rises when the average performance of base classifiers trained on datasets with varying sampling ratios is taken into account.

It is significant that MLEnsem-ble with the asset-based weighting demonstrated a comparable classification performance to that with the uniform weighting, despite the fact that the specifics are left out for space reasons. This is due to the fact that the base classifiers in MLEnsemble have a similar classification tendency, meaning that cases that are successfully categorized are nearly always among them. Consequently, the weights assigned to these classifiers, as determined by Equation 2, acquire comparable values. This observation suggests that asset-based weighting works well for ensemble classifiers whose base classifier classification tendencies are different from one another. Base classifiers are trained for varying sample ratios, so their tendencies vary from one another. This type of ensemble classifier is what MMEnsemble is.

4.3. Knowledge Acquired

Question 1: Does MMEnsemble perform better than the most advanced imbalanced classification techniques of oversampling, undersampling, and metric learning? — While MMEnsemble was comparable to DDAE in Gmean and F2, it attained the state-of-the-art in terms of recall and AUC. This suggests that while MMEnsemble can increase recall, its precision and TNR performance are constrained. Improving MMEnsemble for these metrics without compromising high recall is a viable future step, even though many real-world applications require a larger recall. A high TNR and precision with high recall is optimal.
Q2: Is it effective to combine multi-ratio ensemble and metric learning? — Indeed, all indicators show improved classification performance as a result of the combination. When MMEnsemble and MLEnsemble were compared, the multi-ratio ensem-ble was shown to increase performance, and when MMEnsemble and MUEnsemble were compared, the metric learning was found to improve performance.
Q3: Does the categorization performance get better with asset-based weighting? and how does the selection of its hyper-parameter k (Equation 2) affect things?

For all metrics, asset-based weighting outperformed the two weighting schemes (uniform and Gaussian) in classification; however, because it is sensitive to recall, the hyper-parameter k needs to be carefully chosen. While the Gmean and F2 rise with increasing k, recall falls. This suggests that classifiers perform better in terms

of precision and TNR when k is increased. As a result, k can be adjusted based on user preferences for precision or recall.

5. Conclusion

This research proposed MMEnsemble, a unique ensemble framework based on undersampling. To get over the problem of inadequate data space in the earlier undersampling-based ensemble approaches, MMEnsemble combines three methods: metric learning, multi-ratio ensemble, and asset-based weighting. MMEnsem-ble outperformed the state-of-the-art techniques, according to an experimental evaluation, particularly in terms of recall and AUC metrics. The main drawback of MMEnsemble (as well as the other approaches) is that while it can increase recall scores, precision is lost.

References

1. J. Alcala´-Fdez, A. Ferna´ndez, J. Luengo, J. Derrac, and S. Garc´ıa. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *J. Multiple Valued Log. Soft Comput.*, 17(2-3):255–287, 2011.
2. S. Barua, M. M. Islam, and K. Murase. ProWSyn: Proximity Weighted Synthetic Oversampling Technique for Imbalanced Data Set Learning. In *PAKDD 2013*, pages 317–328, 2013.
3. S. Barua, M. M. Islam, X. Yao, and K. Murase. MWMOTE-Majority Weighted Minority Oversam- pling Technique for Imbalanced Data Set Learning. *IEEE Trans. Knowl. Data Eng.*, 26(2):405–425, 2014.
4. G. E. A. P. A. Batista, A. L. C. Bazzan, and M. C. Monard. Balancing Training Data for Automated Annotation of Keywords: a Case Study. In *II Brazilian Workshop on Bioinformatics*, pages 10–18, 2003.
5. G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6(1):20–29, 2004.
6. C. Bellinger, S. Sharma, N. Japkowicz, and O. R. Zaane. Framework for extreme imbalance classification: SWIM-sampling with the majority class. *KAIS*, 2019.
7. S. Bhattacharya, V. Rajan, and H. Shrivastava. ICU Mortality Prediction: A Classification Algo- rithm for Imbalanced Datasets. In *AAAI 2017*, pages 1288–1294, 2017.
8. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.*, 16:321–357, 2002.
9. C. Elkan. The Foundations of Cost-Sensitive Learning. In *IJCAI 2001*, pages 973–978, 2001.
10. M. Galar, A. Ferna´ndez, E. B. Tartas, H. B. Sola, and F. Herrera. A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 42(4):463–484, 2012.
11. S. Gazzah and N. E. B. Amara. New Oversampling Approaches Based on Polynomial Fitting for Imbalanced Data Sets. In *DAS 2008*, pages 677–684, 2008.
12. S. E. Go´mez, L. Herna´ndez-Callejo, B. C. Mart´ınez, and A. J. Sa´nchez-Esguevillas. Exploratory study on Class Imbalance and solutions for Network Traffic Classification. *Neurocomputing*, 343:100–119, 2019.
13. H. Han, W. Wang, and B. Mao. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In *ICIC 2005*, pages 878–887, 2005.
14. H. He, Y. Bai, E. A. Garcia, and S. Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *IJCNN 2008*, pages 1322–1328, 2008.
15. H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.*, 21(9):1263–1284, 2009.
16. P. Kang and S. Cho. EUS SVMs: Ensemble of Under-Sampled SVMs for Data Imbalance Problems. In *ICONIP 2006*, pages 837–846, 2006.
17. T. Komamizu, R. Uehara, Y. Ogawa, and K. Toyama. MUEnsemble: Multi-ratio Undersampling- Based Ensemble Framework for Imbalanced Data. In *DEXA 2020*, pages 213–228, 2020.
18. G. Kova´cs. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, 83:105662, 2019. (IF-2019=4.873).

19. X. Liu, J. Wu, and Z. Zhou. Exploratory Undersampling for Class-Imbalance Learning. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 39(2):539–550, 2009.
20. X. Liu, J. Wu, and Z. Zhou. Exploratory Undersampling for Class-Imbalance Learning. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 39(2):539–550, 2009.
21. W. Lu, Z. Li, and J. Chu. Adaptive Ensemble Undersampling-Boost: A novel learning framework for imbalanced data. *Journal of Systems and Software*, 132:272–282, 2017.
22. I. Mani and I. Zhang. kNN approach to unbalanced data distributions: a case study involving information extraction. In *ICML'2003 Workshop on Learning from Imbalanced Datasets*, volume 126, 2003.
23. H. M. Nguyen, E. W. Cooper, and K. Kamei. Borderline over-sampling for imbalanced data classification. *IJKESDP*, 3(1):4–21, 2011.
24. M. Peng, Q. Zhang, X. Xing, T. Gui, X. Huang, Y. Jiang, K. Ding, and Z. Chen. Trainable Undersampling for Class-Imbalance Learning. In *AAAI 2019*, pages 4707–4714, 2019.
25. A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi. Calibrating Probability with Under- sampling for Unbalanced Classification. In *SSCI 2015*, pages 159–166, 2015.
26. D. Rodr´ıguez, I. Herraiz, R. Harrison, J. J. Dolado, and J. C. Riquelme. Preliminary Comparison of Techniques for Dealing with Imbalance in Software Defect Prediction. In *EASE 2014*, pages 43:1–43:10, 2014.
27. C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse, and A. Napolitano. RUSBoost: A Hybrid Approach to Alleviating Class Imbalance. *IEEE Trans. Systems, Man, and Cybernetics, Part A*, 40(1):185–197, 2010.
28. A. Sharififar, F. Sarmadian, and B. Minasny. Mapping imbalanced soil classes using Markov chain random fields models treated with data resampling technique. *Computers and Electronics in Agriculture*, 159:110–118, 2019.
29. M. R. Smith, T. R. Martinez, and C. G. Giraud-Carrier. An instance level analysis of data complexity. *Machine Learning*, 95(2):225–256, 2014.
30. I. Tomek. Two Modifications of CNN. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772, 1976.
31. J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo. OpenML: networked science in machine learning. *SIGKDD Explor.*, 15(2):49–60, 2013.
32. H. Wang, Y. Gao, Y. Shi, and H. Wang. A Fast Distributed Classification Algorithm for Large- Scale Imbalanced Data. In *ICDM 2016*, pages 1251–1256, 2016.
33. N. Wang, X. Zhao, Y. Jiang, and Y. Gao. Iterative Metric Learning for Imbalance Data Classifi- cation. In *IJCAI 2018*, pages 2805–2811, 2018.
34. K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. In *NIPS 2005*, pages 1473–1480, 2005.
35. D. L. Wilson. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Trans. Systems, Man, and Cybernetics*, 2(3):408–421, 1972.
36. J. Yin, C. Gan, K. Zhao, X. Lin, Z. Quan, and Z. Wang. A Novel Model for Imbalanced Data Classification. In *AAAI 2020*, pages 6680–6687, 2020.