Improving Imbalanced Classification by Anomaly Detection

Jiawen Kong¹, Wojtek Kowalczyk¹, Stefan Menzel² and Thomas Bäck¹

¹ Leiden University, Leiden, the Netherlands
{j.kong, w.j.kowalczyk, t.h.w.baeck}@liacs.leidenuniv.nl
² Honda Research Institute Europe GmbH, Offenbach, Germany
{stefan.menzel}@honda-ri.de

Abstract. Although the anomaly detection problem can be considered as an extreme case of class imbalance problem, 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. However, in this paper, we gain additional information in another way, by introducing additional attributes. We propose to introduce the outlier score and four types of samples (safe, borderline, rare, outlier) as additional attributes in order to gain more information on the data characteristics and improve the classification performance. According to our experimental results, introducing additional attributes can improve the imbalanced classification performance in most cases (6 out of 7 datasets). 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). The proposed idea of introducing additional attributes is simple to implement and can be combined with resampling techniques and other algorithmic-level approaches in the imbalanced learning domain.

Keywords: Class imbalance · Anomaly detection · Borderline samples.

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

* This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement number 766186.
to be efficient in handling imbalanced datasets, including resampling techniques and algorithmic-level approaches [5, 9, 15], 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 [4]. 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 paper, 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 [2], 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 [14,16], as another additional attribute. 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).

The remainder of this paper is organized as follows. In Section 2, the research related to our work is presented, also including the relevant background knowledge on four resampling approaches, outlier score and the four types of samples. In Section 3, the experimental setup is introduced in order to understand how the results are generated. Section 4 gives the results and further discussion of our experiments. Section 5 concludes the paper and outlines further research.

2 Related Work

As mentioned in the Introduction, we propose to introduce two additional attributes into the imbalanced datasets in order to gain more information on the data characteristics and improve the classification performance. Introducing additional attributes can be regarded as a data preprocessing method, which is independent of resampling techniques and algorithmic-level approaches, and can also be combined with these two approaches. In this section, the background knowledge related to our experiment is given, including resampling techniques (Section 2.1), the definition of four types of samples in the imbalance learning domain (Section 2.3) and the outlier score (Section 2.2).

2.1 Resampling techniques

In the following, we introduce two oversampling techniques (SMOTE and ADASYN) and two undersampling techniques (NCL and OSS).
Oversampling Techniques The synthetic minority oversampling technique (SMOTE) is the most famous resampling technique [3]. SMOTE produces synthetic minority samples based on the randomly chosen minority samples and their $K$-nearest neighbours. The new synthetic sample can be generated by interpolation between the selected minority sample and one of its $K$-nearest neighbours. The main improvement in the adaptive synthetic (ADASYN) sampling technique is that the samples which are harder to learn are given higher importance and will be oversampled more often in ADASYN [7].

Undersampling Techniques One-Sided Selection (OSS) is an undersampling technique which combines Tomek Links and the Condensed Nearest Neighbour (CNN) Rule [4,11]. In OSS, noisy and borderline majority samples are removed with so-called Tomek links [17]. The safe majority samples which have limited contribution for building the decision boundary are then removed with CNN. Neighbourhood Cleaning Rule (NCL) emphasizes the quality of the retained majority class samples after data cleaning [12]. The cleaning process is first performed by removing ambiguous majority samples through Wilson’s Edited Nearest Neighbour Rule (ENN) [19]. Then, the majority samples which have different labels from their three nearest neighbours are removed. Apart from this, if a minority sample has different labels from its three nearest neighbours, then the three neighbours are removed.

2.2 Four Types of Samples in the Imbalance Learning Domain

Napierala and Stefanowski proposed to analyse the local characteristics of minority examples by dividing them into four different types: safe, borderline, rare examples and outliers [14]. 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 (Appendix A) is applied to calculate the distance between different examples. Given the number of neighbours $k$ (odd), the label to a minority example can be assigned through the ratio of the number of its minority neighbours to the total number of neighbours ($R_{\text{min}}$) according to Table 1. The label for a majority example can be assigned in a similar way.

| Type       | Safe (S) | Borderline (B) | Rare (R) | Outlier (O) |
|------------|----------|----------------|----------|-------------|
| Rule       | $\frac{k+1}{2k} \leq R_{\text{min}} \leq 1$ | $\frac{k-1}{2k} \leq R_{\text{min}} \leq \frac{k+1}{2k}$ | $0 < R_{\text{min}} < \frac{k-1}{2k}$ | $R_{\text{min}} = 0$ |
| E.G. given the neighbourhood of a fixed size $k = 5$ |
| Rule       | $\frac{3}{5} \leq R_{\text{min}} \leq 1$ | $\frac{2}{5} \leq R_{\text{min}} \leq \frac{3}{5}$ | $0 < R_{\text{min}} < \frac{2}{5}$ | $R_{\text{min}} = 0$ |
2.3 Outlier Score

Many algorithms have been developed to deal with anomaly detection problems and the experiments in this paper 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 by Breunig et al. in 2000 [2]. 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 1.

**Algorithm 1: Local Outlier Factor (LOF) algorithm [2]**

| Input | X - input data X = \(\{X_1, \ldots, X_n\}\)  |
|-------|-------------------------------------------------|
| n     | the number of input examples                     |
| k     | the number of neighbours                         |
| Output| LOF score of every \(X_i\)                       |

1. initialization;
2. calculate the distance \(d(\cdot)\) between every two data points;
3. for \(i = 1\) to \(n\) do
4. calculate \(k\)-distance \((X_i)\): the distance between \(X_i\) and its \(k\)th neighbour;
5. find out \(k\)-distance neighbourhood \(N_k(X_i)\): the set of data points whose distance from \(X_i\) is not greater than \(k\)-distance \((X_i)\);
6. for \(j = 1\) to \(n\) do
7. calculate reachability distance:
   \[
   \text{reach-dist}_k(X_i, X_j) = \max\{k\text{-distance}(X_j), d(X_i, X_j)\};
   \]
8. calculate local reachability density:
   \[
   \text{lrd}_k(X_i) = \frac{1}{\text{avg-reach-dist}_k(X_i)} = \frac{1}{\left(\sum_{o \in N_k(X_i)} \text{reach-dist}_k(X_i, X_j)\right)};
   \]
   intuitively, the local reachability density of \(X_i\) is the inverse of the average reachability distance based on the \(k\)-nearest neighbours of \(X_i\);
9. calculate LOF:
   \[
   \text{LOF}_k(X_i) = \frac{\sum_{o \in N_k(X_i)} \text{lrd}_k(X_j)}{|N_k(X_i)|} \cdot \text{lrd}_k(X_i)
   = \frac{\sum_{o \in N_k(X_i)} \text{lrd}_k(X_j)}{|N_k(X_i)|};
   \]
   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 \(X_i\).
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.

### 3 Experimental Setup

In this paper, we propose to introduce the four types of samples and the outlier score as additional attributes of the original imbalanced dataset, where the former can be expressed as $R_{\text{min}}$ (Table 1) and the latter can be calculated through Python library PyOD [20].

The experiments reported in this paper are based on 7 two-class imbalanced datasets, including 6 imbalanced benchmark datasets (given in Table 2) and a 2D imbalanced chess dataset, which is commonly used for visualising the effectiveness of the selected techniques in the imbalanced learning domain [4]. Imbalance ratio (IR) is the ratio of the number of majority class samples to the number of minority class samples. For each dataset, we consider four scenarios, whether to perform resampling techniques on the original datasets and whether to perform resampling techniques on the datasets with additional attributes. For each scenario 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. Each collected dataset is divided into 5 stratified folds (for cross-validation) and only the training set is oversampled, where the stratified fold is to ensure that the imbalance ratio in the training set is consistent with the original dataset and only oversampling the training set is to avoid over-optimism problem [15]. Our code will be available on Github ([https://github.com/FayKong/PPSN2020](https://github.com/FayKong/PPSN2020)) for the convenience of reproducing the main results and figure.

| 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                 |
In this paper, we evaluate the performance through several different measures, including Area Under the ROC Curve (AUC), precision, recall, F-Measure (F1) and Geometric mean (Gmean) [13]. These performance measures can be calculated as follows.

\[
AUC = \frac{1 + TP_{rate} - FP_{rate}}{2}, \quad TP_{rate} = \frac{TP}{TP + FN}, \quad FP_{rate} = \frac{FP}{FP + TN};
\]

\[
GM = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{FP + TN}}; \quad F_M = \frac{(1 + \beta)^2 \times \text{Recall} \times \text{Precision}}{\beta^2 \times \text{Recall} + \text{Precision}};
\]

where \(TP, FN, FP, TN\) indicate True Positives, False Negatives, False Positives and True Negatives in the standard confusion matrix for binary classification.

4 Experimental Results and Discussion

Like other studies [7, 13], 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. Please note that we did not tune the hyperparameters for the classification algorithms and the resampling techniques [9]. The experimental results with the two additional attributes (four types of samples and LOF score) are presented in Table 3. We can observe that introducing outlier score and four types of samples as additional attributes can significantly improve the imbalanced classification performance in most cases. For 5 out of 7 datasets (2D chess dataset, glass1, yeast4, wine quality and page block), only introducing additional attributes (with no resampling) gives better results than performing resampling techniques.

| Methods | Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|---------|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE    | NO  | 0.8482 | 0.5743 | 0.6692 | 0.6208 | 0.8847 | 0.8265 | 0.8265 |
|         | YES | 0.9771 | 0.9557 | 0.9078 | 0.9226 | 0.9460 | 0.9859 | 0.9859 |
| SMOTE   | NO  | 0.9554 | 0.9122 | 0.9174 | 0.9346 | 0.9513 | 0.9524 | 0.9524 |
|         | YES | 0.9784 | 0.9191 | 0.9061 | 0.9064 | 0.9453 | 0.9433 | 0.9433 |
| ADASYN  | NO  | 0.8482 | 0.5743 | 0.6692 | 0.6208 | 0.8847 | 0.6172 | 0.1434 |
|         | YES | 0.9771 | 0.9557 | 0.9078 | 0.9226 | 0.9460 | 0.9954 | 0.9226 |
| NCL     | NO  | 0.7836 | 0.1241 | 0.6652 | 0.4952 | 0.5641 | 0.5520 | 0.1076 |
|         | YES | 0.9715 | 0.9543 | 0.9667 | 0.8988 | 0.9716 | 0.9937 | 0.9937 |
| OSS     | NO  | 0.9743 | 0.9321 | 0.9381 | 0.9316 | 0.9640 | 0.9937 | 0.9937 |
|         | YES | 0.9743 | 0.9321 | 0.9381 | 0.9316 | 0.9640 | 0.9937 | 0.9937 |
Improving Imbalanced Classification by Anomaly Detection

### Methods

| Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE | NO | 0.9329 | 0.6883 | 0.8011 | 0.7214 | NO | 0.9327 | 0.6897 | 0.8009 | 0.7211 |
| YES | YES | 0.8760 | 0.6585 | 0.7305 | 0.7051 | YES | 0.8787 | 0.6615 | 0.7322 | 0.7069 |

**glass1 dataset**

| Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE | NO | 0.9329 | 0.6883 | 0.8011 | 0.7214 | NO | 0.9327 | 0.6897 | 0.8009 | 0.7211 |
| YES | YES | 0.8760 | 0.6585 | 0.7305 | 0.7051 | YES | 0.8787 | 0.6615 | 0.7322 | 0.7069 |

**ecoli4 dataset**

| Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE | NO | 0.9329 | 0.6883 | 0.8011 | 0.7214 | NO | 0.9327 | 0.6897 | 0.8009 | 0.7211 |
| YES | YES | 0.8760 | 0.6585 | 0.7305 | 0.7051 | YES | 0.8787 | 0.6615 | 0.7322 | 0.7069 |

**yeast4 dataset**

| Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE | NO | 0.9329 | 0.6883 | 0.8011 | 0.7214 | NO | 0.9327 | 0.6897 | 0.8009 | 0.7211 |
| YES | YES | 0.8760 | 0.6585 | 0.7305 | 0.7051 | YES | 0.8787 | 0.6615 | 0.7322 | 0.7069 |

**wine quality dataset**

| Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE | NO | 0.9329 | 0.6883 | 0.8011 | 0.7214 | NO | 0.9327 | 0.6897 | 0.8009 | 0.7211 |
| YES | YES | 0.8760 | 0.6585 | 0.7305 | 0.7051 | YES | 0.8787 | 0.6615 | 0.7322 | 0.7069 |

**pag block dataset**

| Add | AUC | Precision | Recall | F1 | Gmean | AUC | Precision | Recall | F1 | Gmean |
|-----|-----|-----------|--------|----|-------|-----|-----------|--------|----|-------|
| NONE | NO | 0.9329 | 0.6883 | 0.8011 | 0.7214 | NO | 0.9327 | 0.6897 | 0.8009 | 0.7211 |
| YES | YES | 0.8760 | 0.6585 | 0.7305 | 0.7051 | YES | 0.8787 | 0.6615 | 0.7322 | 0.7069 |
According to our experimental setup, we notice that introducing the outlier score focuses on dealing with the minority samples since the outlier score 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 is given in Figure 1 in order to further study the reason for the performance improvement.

From both the experimental results in Table 3 and the visualisation in Figure 1, 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.

Fig. 1. [top left]. Original imbalanced 2D chess dataset. [top right]. Classification performance for original chess dataset. The red-circled points indicate the misclassified points. [bottom left]. Classification performance for SMOTE-sampled chess dataset. [bottom right]. Classification performance for chess dataset with additional attributes.

Apart from the visualisation, the feature importance (with Decision Tree) is also analysed in order to get an additional insight into the usefulness of the new attributes. Detailed importance score of each attribute is shown in Table 4. According to the feature importance analysis, we can conclude that the introduced
“four types of samples” attribute plays an important role in the decision tree classification process for all datasets in our experiment. For 3 out of 7 datasets, the introduced “outlier score” attribute provides useful information during the classification process. The conclusions above show that the two introduced attributes are actually used in the decision process and the “four types of samples” attribute is more important than the “outlier score” attribute.

Table 4. Feature importance analysis with Decision Tree.
The higher the “score” is, the more the feature contributes during the classification; “org” indicates the original attribute while “add” indicates the added attribute; grey cells indicate the three most useful attributes (after adding the two proposed attributes) in the decision tree classification process.

| 2D chess dataset (2 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.4636 | 0.5564 |       |       |
| YES | 0.0101 | 0.0097 | 0.0152 | 0.1699 |

| glass1 dataset (9 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.2661 | 0.0233 | 0.2294 | 0.1291 | 0.0003 | 0.0149 | 0.2634 | 0.0000 | 0.0726 | 0.2419 | 0.1677 |
| YES | 0.1770 | 0.0056 | 0.1527 | 0.1099 | 0.0000 | 0.0110 | 0.1400 | 0.0000 | 0.0056 | 0.1210 | 0.0000 |

| ecoli4 dataset (7 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.1093 | 0.0084 | 0.0988 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| YES | 0.0085 | 0.0337 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

| vehicle1 dataset (18 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.1304 | 0.0694 | 0.0892 | 0.0003 | 0.0000 | 0.0149 | 0.0000 | 0.0635 | 0.2419 | 0.0167 | 0.0167 |
| YES | 0.0216 | 0.0016 | 0.0137 | 0.0227 | 0.0000 | 0.0179 | 0.0000 | 0.0000 | 0.0000 |

| yeast4 dataset (5 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.0860 | 0.0414 | 0.0498 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| YES | 0.0146 | 0.0310 | 0.0252 | 0.0471 | 0.0000 | 0.0413 | 0.0000 | 0.0000 |

| wine quality dataset (11 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.3821 | 0.2448 | 0.1839 | 0.0010 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| YES | 0.1685 | 0.0297 | 0.0483 | 0.0016 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

| page block dataset (10 original & 2 added attributes) | Score | Attr | Add1 | Add2 |
|-----------------------------------------------|-------|------|------|------|
| NO | 0.0406 | 0.1412 | 0.1215 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| YES | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
5 Conclusions and Future Research

In this paper, we propose to introduce additional attributes to the original imbalanced datasets in order to improve the classification performance. Two additional attributes, namely four types of samples and outlier score, and the resampling techniques (SMOTE, ADASYN, NCL and OSS) 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 the class imbalance classification performance. For some datasets, only introducing additional attributes gives better classification results than only performing resampling techniques.
2) 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 paper, we only validate our idea with four resampling techniques and seven benchmark datasets. As future work, other anomaly detection techniques, such as the clustering-based local outlier score (CBLOF) [8] and histogram-based outlier score (HBOS) [6] could be included in the analysis. Future work could also consider an extension of this research for engineering datasets [10], 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.

References

1. Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F.: Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic & Soft Computing 17 (2011)
2. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: Lof: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data. pp. 93–104 (2000)
3. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research 16, 321–357 (2002)
4. Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., Herrera, F.: Learning from imbalanced data sets. Springer (2018)
5. Ganganwar, V.: An overview of classification algorithms for imbalanced datasets. International Journal of Emerging Technology and Advanced Engineering 2(4), 42–47 (2012)
6. Goldstein, M., Dengel, A.: Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. KI-2012: Poster and Demo Track pp. 59–63 (2012)
Appendix A

Heterogeneous Value Difference Metric (HVDM)

HVDM is a heterogeneous distance function that returns the distance between two vectors \( x \) and \( y \) [18], where the vectors can involve both nominal and numerical attributes. The HVDM distance can be calculated by [18]:

\[
HVDM(x, y) = \sqrt{\sum_{a=1}^{n} d_a^2(x_a, y_a)},
\]

(1)
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 \( a \)th attribute of vector \( x \) and \( y \) respectively. It is defined as follows:

\[
d_a(x, y) = \begin{cases} 
1, & \text{if } x \text{ or } y \text{ is unknown} \\
\text{norm}_\text{vdm}_a(x, y), & \text{if } a \text{th attribute is nominal} \\
\text{norm}_\text{diff}_a(x, y), & \text{if } a \text{th attribute is continuous}
\end{cases}
\] (2)

where

\[
\text{norm}_\text{vdm}_a(x, y) = \sqrt{\sum_{c=1}^{C} \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^2}, \quad \text{norm}_\text{diff}_a(x, y) = \frac{|x - y|}{4\sigma_a},
\] (3)

where

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