mathematics

MDPI

Article

Density-Based Clustering to Deal with Highly Imbalanced Data in Multi-Class Problems

Julio Cesar Munguía Mondragón [1], Eréndira Rendón Lara [1], Roberto Alejo Eleuterio [1,*,†], Everardo Efrén Granda Gutirrez [2,†] and Federico Del Razo López [1]

[1] Division of Postgraduate Studies and Research, National Technological of Mexico (TecNM), Instituto Tecnológico de Toluca, Metepec 52149, Estado de Mexico, Mexico; jmunguiam1@toluca.tecnm.mx (J.C.M.M.); erendonl@toluca.tecnm.mx (E.R.L.); fdelrazol@toluca.tecnm.mx (F.D.R.L.)
[2] University Center at Atlacomulco, Autonomous University of the State of Mexico (UAEMex), Atlacomulco 50400, Estado de Mexico, Mexico; eegrandag@uaemex.mx
* Correspondence: ralejoe@toluca.tecnm.mx
† These authors contributed equally to this work.

Abstract: In machine learning and data mining applications, an imbalanced distribution of classes in the training dataset can drastically affect the performance of learning models. The class imbalance problem is frequently observed during classification tasks in real-world scenarios when the available instances of one class are much fewer than the amount of data available in other classes. Machine learning algorithms that do not consider the class imbalance could introduce a strong bias towards the majority class, while the minority class is usually despised. Thus, sampling techniques have been extensively used in various studies to overcome class imbalances, mainly based on random undersampling and oversampling methods. However, there is still no final solution, especially in the domain of multi-class problems. A strategy that combines density-based clustering algorithms with random undersampling and oversampling techniques is studied in this work. To analyze the performance of the studied method, an experimental validation was achieved on a collection of hyperspectral remote sensing images, and a deep learning neural network was utilized as the classifier. This data bank contains six datasets with different imbalance ratios, from slight to severe. The experimental results outperform the classification measured by the geometric mean of the precision compared with other state-of-the-art methods, mainly for highly imbalanced datasets.

Keywords: density-based clustering algorithms; sampling methods; deep neural networks

MSC: 68-xx; 62-xx

Citation: Munguía Mondragón, J.C.; Rendón Lara, E.; Alejo Eleuterio, R.; Granda Gutirrez, E.E.; Del Razo López, F. Density-Based Clustering to Deal with Highly Imbalanced Data in Multi-Class Problems. *Mathematics* **2023**, *11*, 4008. https://doi.org/10.3390/math11184008

Academic Editor: Liangxiao Jiang

Received: 6 September 2023
Revised: 19 September 2023
Accepted: 20 September 2023
Published: 21 September 2023

1. Introduction

Many machine learning and data mining applications require working with unbalanced datasets [1]. Although it is not a new topic, the results in diverse applications where this problem occurs indicate that it is still a current challenge for the data science community [2], especially in new Big Data scenarios [3].

An unbalanced dataset refers to a situation in which there exists a substantial disparity in the distribution of instances belonging to specific classes when compared to the sample sizes of one or more other classes. This imbalance can have profound implications for various data-driven tasks, especially in the realm of machine learning. The imbalance causes a bias in the results of classical classifiers, leading to a misinterpreted efficiency, even in the most recent cases, such as those based on deep learning [2]. Unbalanced data usually imply the misrecognition of the less represented classes, while the more populated classes are well recognized [4]. The trouble of class imbalance is present in many real-life problems: fraud detection, medical diagnosis, and anomaly identification, among others [5]. For example, in diagnosing some illnesses, a class of particular interest could have only a

few patients with a rare cancer, while another class has numerous healthy patients, leading to a tricky identification of the little-represented class. In data mining, the interest class with few representative instances is named minority class, or positive class [4,6].

Various methods to address the class imbalance problem can be documented in the literature [7]. They could be grouped into three categories: (a) data-level (or external), (b) algorithm-level (or internal), and (c) methods that consider the prediction errors (cost-sensitive) [8]. The strategies applied at the data level involve manipulating the original dataset. This manipulation can include either augmenting the minority class by introducing synthetic instances (oversampling) or reducing the number of instances in the majority class (undersampling) [9]. Also, the hybrid methods combine both oversampling and undersampling; for example, the undersampling clustering method [10] or the combination of oversampling and heuristics (Tomek Links) [11]. Most of the oversampling methods generate synthetic instances of the minority class until a relative balance in the dataset is achieved; they are based on the Synthetic Minority Over-Sampling Technique (SMOTE) [12], as it occurs in Borderline-SMOTE (B-SMOTE) [13], or the adaptative approach ADASYN (Adaptive Synthetic) [14]. In contrast, the clustering techniques are being popular because of their effectiveness to address class imbalance; for example, the Fast Clustering-Based Undersampling method (Fast-CBUS) [15], the Clustering Based Oversampling (CBOS) [16], or the SMOTE and Clustered Undersampling Technique (SCUT) [17]. Those methods use clustering to reduce the loss or distortion by eliminating instances from the majority classes or creating instances in the minority classes.

The relevance of clustering techniques has been studied, and some authors have demonstrated advancements in density-based clustering techniques for diverse data types and applications, emphasizing improved performance in terms of cluster quality, robustness against noise, and computational efficiency. For example, in a specific application, ref. [18] introduces the MDPC-AD algorithm (Mixture Distance-Based Improved Density Peaks Clustering), designed for gearbox fault diagnosis. It combines density-based and distance-based clustering techniques to handle mixed data types (numerical and categorical). Likewise, ref. [19] introduces an uncertain data object clustering method based on distribution similarity, using measures like KL-divergence and J-divergence along with k-medoids and modified DBSCAN. It is applied to PDF file databases and evaluated on real and synthetic datasets. While it may not be directly applicable to all databases, it provides an interesting alternative for assessing DBSCAN and related algorithms.

In relation to advanced clustering-based approaches, ref. [20] presents the DCSNE (Density-based Clustering using Graph Shared Neighbors and Entropy) algorithm. DCSNE excels in cluster quality, noise handling, and computational efficiency. It comprises four steps: similarity graph construction, density calculation, outlier detection, and merging density regions. It outperforms other clustering methods on synthetic, gene expression, and real datasets. In another example, ref. [21] presents an entropy-based density peak clustering technique for gene expression datasets. It leverages entropy and the Extreme Clustering algorithm to tackle high dimensionality and noise in gene expression data. This technique excels in cluster quality, noise robustness, and biological significance.

In this paper, four heuristics to face the data imbalance in multi-class problems are studied; they consist of a combination of clustering and sampling techniques. The general idea is to eliminate some instances of the majority classes using density-based clustering algorithms to avoid the loss of relevant information, and then SMOTE to add instances to the minority classes and reduce the imbalance. Hyperspectral images with multiple classes and high imbalance were used as datasets to validate the proposed method, and a deep neural network (DNN) was used as a classifier. The mean geometric value was used to quantify the efficiency of the method, and Friedman's test was used to rank the different approaches. The main contribution of this research lies in the study of density-based clustering methods to deal with the problem of class imbalance in multiple classes and in scenarios with severe imbalance, which have been little studied in the recent literature. The main findings and highlights of this work can be summarized as follows:

- We introduce a new density-based undersampling technique (based on SMBD, Spatially Motivated Balancing by Density) which leverages the DBSCAN and HDBSCAN clustering algorithms.
- An improved version of the SCUT algorithm is presented, enhancing its performance and versatility.
- Our proposed method underwent testing on a database consisting of highly imbalanced and multi-class hyperspectral remote sensing images.
- To assess the effectiveness of our approach, we employed the geometric mean value as a quantitative measure. Additionally, we applied Friedman's test to rank various methodologies, comparing SMBD against conventional algorithms typically used for addressing class imbalance.
- The results obtained not only surpassed those achieved with unbalanced images but also outperformed classical approaches, emphasizing the efficacy of our proposed method.

2. Related Literature

There is a relatively large number of publications on class imbalance, mainly on data-level approaches [22]. The solutions presented primarily lie in changing the distribution of classes in the dataset, either through undersampling or oversampling. The undersampling techniques reduce the size of the majority classes, randomly eliminating some of their instances. In contrast, the oversampling methods add instances to the minority classes by duplicating or interpolating existing ones. Random undersampling methods have demonstrated promising results in terms of the computational cost [23] because they use small groups of the majority classes in the training stage, favoring a fast and efficient learning process [9]. However, information loss can occur [11], so more sophisticated techniques have been proposed to remove instances of the majority classes more selectively: examples of this include neighborhood-based algorithms [24,25] and clustering-based techniques [16].

2.1. Neighborhood-Based Methods

Several variants of neighborhood-based methods are currently in use [23,26]. The condensed nearest neighbor (CNN) method, also known as Hart algorithm [27], eliminates instances far from the decision borders determined by the nearest neighbor technique because they are considered irrelevant for the learning process. Opposite to the Hart algorithm, the Tomek Links (TL) method eliminates the instances from the majority or minority classes that lie close together, i.e., close to the decision border [28]. In [24], the One-Sided Selection (OSS) methodology is proposed; first, the CNN eliminates the redundant instances; then, other instances considered as noise are deleted by TL. In [25], the Wilson Editing method [29] is used to eliminate instances that were misclassified by the nearest neighbor technique. In [30], the Neighborhood Cleaning Rule (NCL), which is an adaptation of the Wilson Editing method, where only the instances from the majority class can be deleted, is used.

2.2. Clustering-Based Techniques

Clustering algorithms are the new approach to using subsampling methods but avoiding information loss [10]. Ofek et al. [15] propose the Fast-CBUS, where the K-means algorithm [31] groups the instances from the minority class. The same number of instances from the majority class are selected for each group within the minority class. The selected samples from minority and majority classes are used to train a classifier. In [32], the Clustering Large Applications (CLARA) algorithm [31] is used to group instances from the majority class. The Silhouette index estimates the optimal number of groups, and instances from each group are randomly selected, thus reducing the size of the majority class. The C4.5 classifier was utilized to test the proposed methodology, and the Area Under the Curve (AUC) was used to evaluate the overall efficiency. In [16], the method CBOS is proposed

to classify binary class (two classes); this method uses the K-means algorithm to group instances from the minority class. Synthetic instances between the centroid and the group instances are created for each group. In [17], the SCUT algorithm is proposed to balance multiple-class datasets. The class balancing is achieved depending on the mean size of the classes, P (number of instances per class). Then, the Expectation-Maximization [33] algorithm is applied for majority classes with a size greater or equal than P, while SMOTE is used to generate synthetic instances in the minority classes with a size less than P.

2.3. Density-Based Clustering Algorithms

Density-based clustering algorithms are extensively used in data mining applications: mainly geo-localization and medical imaging segmentation [34]. The success of these algorithms is because they can group the data into different sizes, shapes, and densities [31]. The groups are based in dense regions of data separated by low-density regions, which are considered noise or atypical data, which could be eliminated from the dataset [35]. Clustering algorithms can also be used to reduce the size of the majority class [32]. Although a relatively large variety of heuristics to perform clustering exists, this work uses only density-based algorithms; specifically, the Density-Based Spatial Clustering of Applications with Noise, namely DBSCAN [36], and the Hierarchical DBSCAN, or HDBSCAN [37].

DBSCAN is one of the most relevant density-based clustering methods. Proposed by Ester et al. [36], this algorithm introduces the concept of core-point, noise-point (or outlier), and border-point. It requires two input parameters to create a cluster: the maximum radius of the neighborhood (rv) and the minimum number of data points within the radius of a neighborhood ($minPts$). The following definitions help to understand the DBSCAN process, which is summarized by Algorithm 1.

Definition 1 (Core-point). *An object $p \in D$, related to rv and $minPts$, is a core-point if $\|N_{rv}(p)\| \geq minPts$, where $\|N_{rv}(p)\|$ are neighbor objects within rv from p.*

Definition 2 (Density-reachable point). *An object $q \in D$, related to rv and $minPts$, is density-reachable from $p \in D$ (directly or transiently) if a catenation of objects $p_1, p_2, ..., p_n$ in D exists, with $p_1 = p, p_n = q$ so that $p_{i+1} \in N_{rv}(p_i)$, with p_i being a core-point.*

Definition 3 (Density-connectivity). *An object $p \in D$ is connected by density to $q \in D$, related to rv and $minPts$, if an object $v \in D$ exists so that p and q are density-reachable from v.*

Definition 4 (Density-based clustering). *A density-based group C, related to rv and $minPts$, is a nonempty subset of D, which satisfies, first, that $\forall p, q \in D$: if $p \in C$ and q is density-reachable from p, related to rv and $minPts$. Consequently, $q \in C$; second, that $\forall p, q \in C$: p is density-connected to q related to rv and $minPts$.*

Algorithm 1 Summary of the DBSCAN algorithm.

Input: *D, rv, $minPts$; // D collection of objects, rv maximum radius of the neighborhood, $minPts$ minimum number of datapoints within rv.*
Output: *C // Identified clusters*

1: A random object p is selected. If p belongs to a group, then another object is selected until the selected object is not in a group.
2: If p is a core-point, then a group with all the points reachable from p within rv exists.
3: If p is a border-point (or an outlier), then another object is chosen.
4: If all objects have been visited, the process ends, and all grouped objects C are returned; otherwise, the process is repeated from step 2.
5: **return** C**;**

HDBSCAN was proposed in [37] as an improved version of DBSCAN, which only requires $minPts$ as an input parameter. It can use different values of rv, resulting in groups of distinct densities. The following definitions are useful to understand HDBSCAN, which is summarized by Algorithm 2.

Definition 5 (Core-distance). *The distance to the core $d_{core}(p)$ of an object $p \in D$ is the separation from p to its nearest neighbors ($minPts$, p included).*

Definition 6 (rv-core). *An object $p \in D$ is an rv-core for each value of rv greater than or equal to the core distance of p, related to $minPts$, i.e., if $d_{core}(p) \leq rv$.*

Definition 7 (Mutual reachability distance). *The distance beween two objects $p \in D$ and $q \in D$, related to $minPts$, defined as $d_{mreach}(p,q) = max\{d_{core}(p), d_{core}(q), d(p,q)\}$.*

Definition 8 (Graph of mutual reachability). *A complete graph G_{minPts} is formed by the objects in D as the vertices and the mutual reachability distance as the weight of each object, related to $minPts$, between the respective pairs of objects.*

Algorithm 2 Summary of the HDBSCAN algorithm.

Input: D, rv, $minPts$; *// D collection of objects, $minPts$ minimum number of objects.*
Output: C *// Identified groups*

1: Calculate the core-distance, related to $minPts$, for all objects in D.
2: Calculate a Minimum Spanning Tree (MST) of G_{minPts} [38].
3: Extend the MST to obtain MST_{ext}, by adding an edge with the rv-core distance as a weight for each vertice.
4: Extract the hierarchy as a dendrogram of MST_{ext}:
5: **repeat**
6: Assign the same label to all objects (unique group) for the tree's root.
7: Iteratively eliminate all edges of MST_{ext} in descending order of weights. If a tie occurs, the edges are simultaneously eliminated.
8: **repeat**
9: Before an elimination, establish the weight of the edges to be removed as equivalent to the value of the current hierarchical level dendrogram.
10: Next to an elimination, assign labels to the connected components containing the final vertices of deleted edges to obtain the next hierarchical level. Assign a new group label to an element if at least an edge exists; otherwise, assign a null label (noise).
11: **until** all edges has been removed;
12: **until** all MST_{ext} nodes are analyzed;
13: **return** C;

3. Methodology

The studied strategy is derived from the SCUT method exhibited in [17]. Our approach balances the multiple-class dataset by performing undersampling and oversampling with respect to the average number of samples in each class of the dataset. D is a dataset with C_0,...,C_n classes ($D = C_0 \cup C_1 \cup ... \cup C_n$) and t_0,..., t_n is the number of samples per class ($\|D\| = \sum_{i=0}^{n} t_i = \sum_{i=0}^{n} \|C_i\|$); M is the average number of samples per class ($M = \sum_{i=0}^{n} t_i/n$). If $t_i > M$, an undersampling procedure is performed upon density-based techniques, DBSCAN or HDBSCAN, (in the original work [17], the method Expectation-Maximization (EM) is used). In contrast, if $t_i < M$, SMOTE is applied. In Algorithm 3, the steps of the studied strategy are presented.

The SMDB procedure (Spatially Motivated Balancing by Density) is shown in Algorithm 4. Its input is a class C, and it returns CB as a subset of size M. This algorithm uses a density-based clustering method, either DBSCAN or HDBSCAN, to reduce the size of the class, such that $\|CB\| < \|C\|$. To adjust the reduction ratio, we use the criteria of grouping only

the classes whose size is greater than the average plus an additional percentage Δ (20%), thus avoiding that the algorithm could eliminate an excessive number of objects. On the other hand, for classes with a size less than $M + \Delta$, the SMOTE technique is applied until a subset of objects of size M is obtained.

Algorithm 3 Proposed strategy derived from SCUT.

Input: D (Dataset containing n classes);
Output: DB (Balanced dataset containing n classes of size M each);

Split the classes on D in $C_0, C_1, C_2, \ldots, C_n$, disjoint subsets;
Calculate $t_i = \|C_i\|$, for $i = 0, 1, ..., n$;
Calculate $M = \sum_{i=0}^{n} t_i / n$
$DB = \emptyset$
for $i = 0$ **to** n **do**
  if $t_i > M$ **then**
    $CB_i \leftarrow$ SMBD(C_i, M); /*SMBD *is the density-based undersampling procedure**/
  end if
  if $t_i < M$ **then**
    $CB_i \leftarrow SMOTE(C_i, M)$;
  end if
  if $t_i = M$ **then**
    $CB_i = C_i$;
  end if
  $DB = DB \cup CB_i$;
end for
return DB;

Algorithm 4 SMBD: density-based undersampling procedure.

Input: Subset C containing the objects from a specific class in D ($C \in D$), Δ (tolerance percentage);
Output: Balanced subset CB of a particular class, where $(\|CB\| == M) \wedge (CB \in C)$;

$Limit = (M * \Delta) + M$;
if $\|C\| > Limit$ **then**
  $G_0, G_1, \ldots, G_k \leftarrow DBSCAN(C, rv, minPts)$;
  $G_0, G_1, \ldots, G_k \leftarrow HDBSCAN(C, minPts)$;
  /* G_j, *is the j-th group obtained by the clustering algorithm, where* $j = 0, 1, ..., k$ */
  for $j = 0$ **to** k **do**
    $g\_G_j = |G_j|$; /* *Number of samples of* G_j *cluster* /*
  end for
  for $j = 0$ **to** k **do**
    $T = T + g\_G_j$; /* *T is the sum of samples all clusters* */
  end for
  for $j = 0$ **to** k **do**
    $size_j = \left[\frac{g\_G_j}{T}\right] * M$; /**Number of samples to be selected from each cluster* */
  end for
  $CB = \varnothing$;
  for $j = 0$ **to** k **do**
    $G'_j = Select(size_j, G_j)$; /**Randomly select* $size_j$ *samples from* G_j**/
    $CB = CB \cup G'_j$;
  end for
else
  $CB \leftarrow RUS(M, C)$; /**Randomly select M samples from class C**/
end if
return (CB);

The proposed methodology in Algorithm 3 performs the undersampling process shown in Algorithm 4 which uses either DBSCAN or HDBSCAN. However, other techniques could also be applied. In this study, the K-means clustering algorithm was used as a baseline [33,39]. Additionally, Algorithms 1 and 2 were adapted by calculating the maximum radius of the neighborhood, rv, and the best value for the minimum number of points, $minPts$, for each particular algorithm. Different values of rv and $minPts$ were proven by *greedy* techniques [40], and the Silhouette validation index [31] acted as optimization criteria, to identify the best collection of clusters.

4. Experimental Set-Up

4.1. Datasets

The experimental validation of the proposed methodology was performed in six hyperspectral remote sensing datasets obtained from the GIC repository [41]. We proposed to study remote sensing hyperspectral images, which are collected using airborne or satellite sensors, because these images are gathered across many contiguous spectral bands. This means that each pixel in the picture has a spectrum, which is a plot of the amount of light reflected or emitted at different wavelengths. Hyperspectral images can have hundreds or even thousands of spectral bands, which gives them much more spectral information than traditional aerial images. Remote sensing spectral classification has become essential for spatial data analysis tasks where each pixel is represented as a spectral feature [42]. Therefore, these images contain many classes (land regions), and many times, some classes are underrepresented because they were only captured in a few pixels from the entire image, i.e., they are imbalanced datasets.

In the proposed dataset, the AVIRIS sensor captured data from the Pine Indian region in Northwestern Indiana, yielding a dataset with 220 spectral reflectance bands spanning the wavelength range from 0.4 to 2.5 µm. The Salinas image was also obtained by the AVIRIS sensor in California's Salinas Valley. The ROSIS sensor obtained the PaviaC and PaviaU images during a flight over Pavia in northern Italy; PaviaC's image is of the city center and the university, while PaviaU is just the university. The KSC imagery corresponds to the Kennedy Space Center by the AVIRIS sensor. Finally, the images in Botswana were captured by NASA's EO-1 sensor over the Okavango River Delta in Botswana.

In Table 1, dataset attributes can be observed, which include pixel count, band count, class count, the size of the majority class (C_{maj}), and the size of the minority class (C_{min}). Afterwards, the imbalance ratio (IR) was calculated between the majority class, C_{maj}, and the minority class, C_{min}, i.e., the biggest and smallest classes, respectively: $IR = \|C_{maj}\| / \|C_{min}\|$.

Table 1. Description of the datasets.

Dataset	Pixels	Bands	Classes	C_{maj}	C_{min}	IR
Indian	21,025	220	17	10776	20	533.8
Salinas	111,104	224	17	56,975	916	62.2
PaviaC	1,201,216	102	10	635,488	2685	236.7
PaviaU	372,100	103	10	164,624	947	173.8
Botswana	377,856	145	15	374,608	95	3943.2
KSC	314,368	176	14	309,157	105	2944.4

To evaluate the effectiveness of the classifier, each dataset was split into two disjoint subsets by the hold-out method [43], where the training dataset (TRD) contains 70% of the images, and the testing (TD) dataset represents the remaining 30%, so that TRD $\cap$ TD $== \oslash$. It is noteworthy that this dataset division was executed before any data balancing procedures, and these balancing techniques were exclusively applied to the image training subset. Meanwhile, the testing subset remained unaltered throughout the process, ensuring an equitable evaluation of the proposed scheme's performance. Furthermore, it is essential

to highlight that this separation of training and testing was performed at the image level: each image within the dataset may exhibit distinct pixel distributions, reflecting the unique hyperspectral information it encapsulates. Importantly, no image pre-processing techniques were applied before the data balancing process. The latter allows for a comprehensive evaluation of the classifier's ability to handle the inherent variations within the dataset, ensuring that performance assessments are executed in real conditions.

4.2. Artificial Neural Network Architecture

The artificial neural network used as a classifier for this work was constructed upon that presented in Ref. [11], and it consists of a DNN. The test-and-trial approach was used to find the number of hidden layers and neurons per layer, so the results of this work can be directly compared with Ref. [11]. Although several schemes for the automatic identification of the optimal number of neurons in a DNN exist [44], the test-and-trial method is widely used because of its suitability to be adapted to many scenarios. In addition, for the purpose of this study, we consider that the configuration of four to six hidden layers is appropriate to demonstrate the effectiveness of the methods to balance the dataset.

Table 2 presents the specifications of the deep neural networks employed in the experimental validation of this study. In this table, IL represents the quantity of neurons in the input layer of the DNN, while HL_k denotes the number of neurons in each respective hidden layer (indexed as k); finally, OL signifies the neuron count within the output layer. Furthermore, the free parameters of the DNN were obtained from [11], where 500 epochs were used for the Indian, Salinas, and PaviaU datasets, while 250 epochs were used for KSC, Botswana, and PaviaC. The number of samples was set to be 100 for the Indian dataset and 1000 for the rest of the datasets. The DNN was developed with Tensorflow 2.0 and Keras 2.3.1 frameworks. The training of the DNN was performed with the training subset (TRD, see Section 4.1).

Table 2. Deep neural network architecture shows the number of neurons in each layer.

Dataset	IL	HL_1	HL_2	HL_3	HL_4	HL_5	HL_6	OL
Indian	224	60	60	60	60	-	-	17
Salinas	220	60	60	60	60	60	60	17
PaviaU	103	40	40	40	40	40	-	10
KSC	176	60	60	60	60	60	60	14
Botswana	145	30	30	30	30	30	30	15
PaviaC	102	40	40	40	40	40	40	10

Finally, to perform the experiments, we used a Windows-based computer (Windows 10) equipped with an Intel Core i5 processor (up to 4.5 GHz, 18 MB L3 cache, 12 cores, 16 threads), NVIDIA GeForce RTX 3050 Mobile GPU (4 GB), 16 GB DDR5-4800 SDRAM (2 $\times$ 8 GB), and 512 GB PCIe NVMe.

5. Results and Discussion

Results of the DNN classification of the testing subset (TD, see Section 4.1) are presented in this section to evaluate different balancing methods. The Original method (i.e., the unbalanced datasets) is used as a reference to evaluate the effectiveness of the balancing methods, which are categorized into four types of heuristics:

1. Classical approaches: ROS (Random Oversampling) and SMOTE.
2. SCUT approaches: SCUT and SCUT+KM (where the EM was replaced by K-means.)
3. Density-based methods: SCUT+DBS and SCUT+HDBS, where the clustering algorithm is substituted by DBSCAN and HDBSCAN, respectively.
4. Density-based with *greedy* variants: SCUT+DBS$_v$ and SCUT+HDBS$_v$, where v refers to the use of *greedy* techniques during the optimization of rv and $minPts$.

In the proposed strategy, as outlined in Algorithm 3, the class sizes (denoted as t_i) were obtained for each individual class within the database, grouped under C_i. The pivotal criterion for determining the appropriate balancing approach hinges on whether a class's size surpasses the average class size (M) computed for the entire database. If a class's size exceeds the database-wide average (M), the Spatially Motivated Balancing by Density (SMBD) procedure is initiated, which leverages clustering algorithms such as DBSCAN or HDBSCAN (as detailed in Algorithm 4). On the other hand, if a class's size falls below the M threshold, the SMOTE algorithm is utilized as an oversampling solution. To ensure optimal performance and maintain a consistent balance among all classes, we introduce a fixed approximation ratio of 20 % relative to the average class size of each particular database analyzed; this ratio serves as a stopping criterion for the balancing process. As a result, by the conclusion of the balancing procedure, all classes are approximately equal in size, aligning with the M value calculated for the entire database. This approach guarantees that every class, within all studied databases, maintains a balanced and equitable representation.

The geometric mean (*g-mean*) was used to assess the performance of the tested methods. Geometric mean is a mathematical metric used to calculate the central tendency or average of a set of values $x_1, x_2, x_3, ..., x_n$. Unlike the more common arithmetic mean, which adds up all values and divides by the number of values, the geometric mean involves multiplying all values together and then taking the n-th root, where n is the number of values in the dataset. Equation (1) represents the basic calculation of *g-mean*.

$$g\text{-}mean = \sqrt[n]{(x_1 \cdot x_2 \cdot x_3 \cdot \ldots \cdot x_n)} \quad (1)$$

Moreover, the geometric mean can be applied in the context of machine learning, mainly when dealing with imbalanced classification problems, to assess the performance of the classification model, especially in scenarios where accuracy might be misleading due to class imbalances [45]. Thus, the geometric mean is a way to evaluate the performance of a classification model, especially in situations where the distribution of classes is highly imbalanced, as occurs in our tested dataset. *g-mean* is related to other common metrics, such as Sensitivity and Specificity [46]; however, for multi-class problems, Equation (1) exhibited before can be applied to calculating *g-mean* from the individual precision of each class. Consequently, the metric represents a weighted precision score that considers the efficacy of predicting both the minority and the majority classes.

Thus, the average of the geometric mean from five repetitions of each balancing method, and the Original subset, is reported in Table 3. The intention of performing five repetitions is to reduce the random effects and ensure that results could follow a trend. The best results are marked in bold font.

Table 3. Average of geometric mean for the tested methods. The best scores are highlighted in bold.

Method	Indian	Salinas	Botswana	KSC	PaviaC	PaviaU
Original	0.000	0.882	0.000	0.000	0.239	0.579
SMOTE	**0.824**	0.960	0.698	0.687	0.884	0.862
ROS	0.822	**0.961**	0.709	0.659	0.899	0.874
SCUT	0.763	0.935	0.882	0.853	0.905	0.858
SCUT+KM	0.754	0.944	0.842	**0.858**	0.908	0.878
SCUT+HDBS	0.760	0.946	**0.891**	0.848	0.904	**0.886**
SCUT+DBS	0.764	0.946	0.877	0.843	0.909	0.883
SCUT+DBS $_v$	0.768	0.946	0.888	0.837	**0.911**	0.882
SCUT+HDBS $_v$	0.787	0.944	0.885	0.845	0.909	0.879

The class imbalance negatively affects the precision of the classifier, especially in the more unbalanced datasets (Table 3). In this sense, the Original subset obtained *g-mean* = 0 for Indian, Botswana, and KSC databases; however, relatively low *g-mean* values were also observed for the other datasets, except for Salinas, which is the dataset with the lowest $IR = 62.2$ (i.e., the more balanced dataset). In contrast, when the datasets are processed by balancing methods, all results are better than the Original case, which agrees with the reported literature. Nevertheless, our objective is to demonstrate the effectiveness of the density-based clustering methods in comparison with other approaches.

Table 3 shows that classical methods (ROS and SMOTE) work fine when slightly unbalanced datasets are processed. This is the case of Salinas, PaviaU, and PaviaC datasets, where the *g-mean* values are noticeably more significant than the Original, and even good results are also obtained for the moderate imbalance present in Indian dataset ($IR = 533.8$). In contrast, the highly imbalanced datasets, KSC and Botswana, which reported an IR of 2944.4 and 3943.2, respectively, exhibit improved classification performance when density-based approaches are utilized. It suggests that the more class imbalance is present, the more effective the density-based clustering approaches. These algorithms are also adequate for slightly unbalanced datasets (Salinas, PaviaU, and PaviaC); however, they seem to not be the best choice for low or moderate imbalance (the case of Indian dataset). It could be attributed to an excessive elimination of useful data from the majority classes, but also to the very reduced size of the minority class in this dataset, which contains only 20 points.

The following plots show a graphical representation of the classifier's performance using the different balancing techniques analyzed in this paper. In Figure 1, we portray a comprehensive analysis of classification performance, as measured by the geometric mean (*g-mean*), across various class imbalance ratios (IR). This figure highlights the efficacy of different oversampling and clustering-based techniques in addressing class imbalances. Two traditional oversampling methods, ROS and SMOTE, are evaluated alongside two clustering-based algorithms, SCUT and SCUT plus K-means. The performance trends across different datasets reveal a striking pattern. When dealing with datasets featuring relatively small class imbalance ratios (e.g., Salinas with an IR of 62.2, PaviaU with an IR of 173.8, and PaviaC with an IR of 236.7), all selected machine learning techniques exhibit commendable performance. However, the situation takes a noteworthy turn when confronted with highly imbalanced datasets, such as Indian with an IR of 533.8, KSC with an IR of 2944.4, and Botswana with an IR of 3943.2. In these scenarios, classical oversampling techniques in Figure 1 demonstrate notably poor classification precision, consistently falling below the 0.7 thresholds. Conversely, the clustering-based algorithms, SCUT and SCUT plus K-means, outperform their traditional counterparts in handling highly imbalanced datasets. Nonetheless, it becomes evident that further improvement is required when addressing extreme class imbalances.

Figure 2 delves deeper into the quest for improved classification performance, particularly in the face of extreme class imbalance. Here, we explore the results of combining SCUT with density-based techniques, namely DBS and HDBS, as well as their variants incorporating greedy strategies (DBS_v and $HDBS_v$). The *g-mean*, as a measure of classification performance, is once again assessed across varying class imbalance ratios. Remarkably, the density-based techniques in Figure 2 consistently outshine the other methods, achieving remarkable performance even in the most severely imbalanced datasets. These techniques attain *g-mean* scores of up to 0.911, showcasing their effectiveness in handling extreme class imbalances. It is clear from these results that, when confronted with highly skewed class distributions, the combination of SCUT with density-based methods offers a powerful solution for enhancing classification precision.

In summary, Figures 1 and 2 underscore the importance of choosing a suitable methodology when dealing with class-imbalanced datasets. While traditional oversampling techniques struggle to maintain classification precision in highly imbalanced scenarios, clustering-based methods demonstrate improved performance. However, the density-based strategies in Figure 2 emerge as the best choices, consistently delivering acceptable

classification results even in the most challenging imbalance situations. These findings provide valuable insights for seeking robust solutions for class imbalance mitigation in machine learning tasks on hyperspectral remote sensing images.

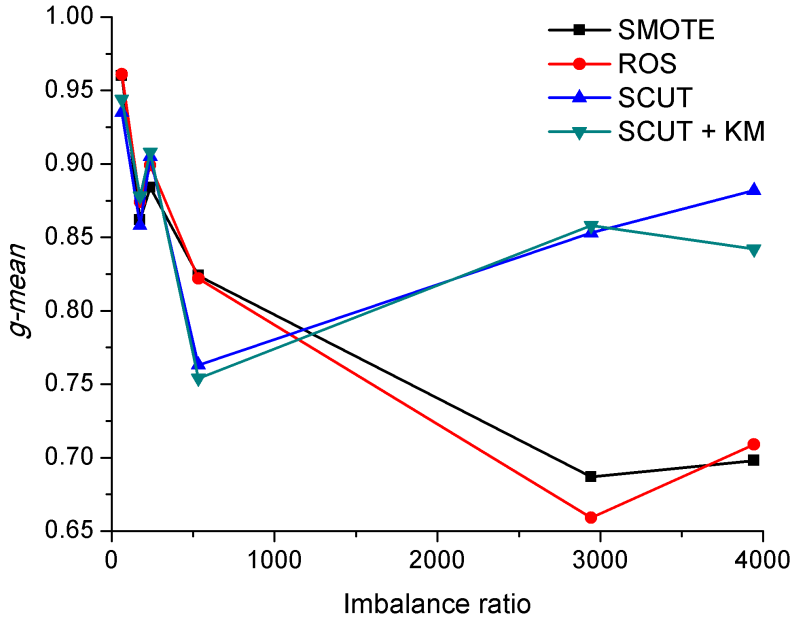

Figure 1. Classification performance (expressed as *g-mean*) after applying traditional oversampling techniques (ROS and SMOTE) and clustering-based algorithms (SCUT and SCUT plus K-means) as a function of class imbalance ratio (*IR*).

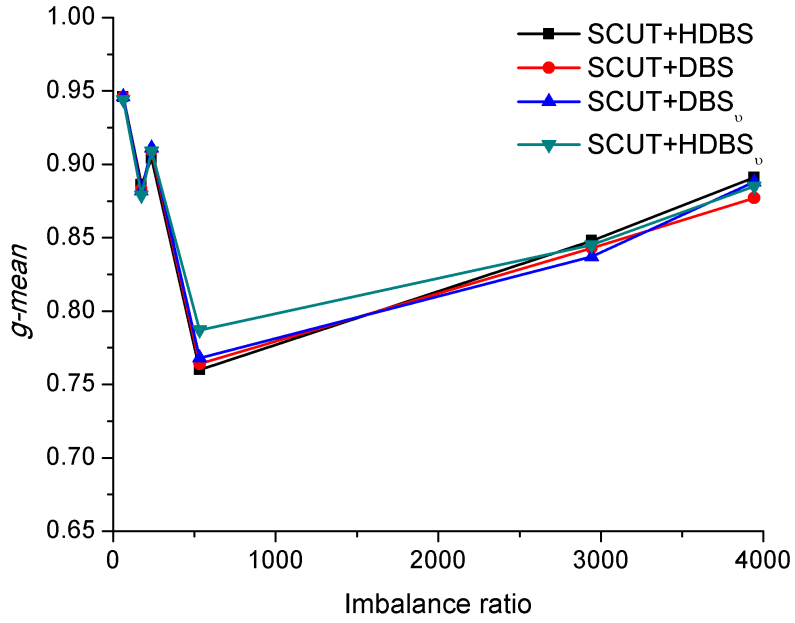

Figure 2. Classification performance (expressed as *g-mean*) after applying Clustered Undersampling (SCUT) plus density-based techniques (DBS and HDBS) and density-based techniques with greedy variants (DBS$_\upsilon$ and HDBS$_\upsilon$) as a function of class imbalance ratio (*IR*).

To provide an objective global measure about the relative performance of the balancing methods from experimental results, Friedman's test ranking was calculated [47]. The first rank is assigned to the best-evaluated method (considering the overall results from the classification performance for the six datasets), while the last rank belongs to the worst result. Table 4 presents Friedman's ranks assigned to the evaluated methods, ordered from

worst to best. The worst case is the Original (or unbalanced) reference experiment, which is in the last rank. The SMOTE algorithm and SCUT method are tied for the next place with a little difference with respect to the ROS and SCUT+KM (both tied for third place). The best place is obtained by (SCUT+DBS$_v$), i.e., the *greedy* variant of SCUT combined with DBSCAN. However, there is no significant difference between all the density-based algorithms, which go from 3.91 to 3.41, occupying the first places in the ranking.

Table 4. Friedman's test ranking for the selected methods.

Method	Ranking
Original	9.00
SMOTE	5.50
SCUT	5.50
ROS	5.17
SCUT+KM	5.17
SCUT+DBS	3.91
SCUT+HDBS$_v$	3.83
SCUT+HDBS	3.50
SCUT+DBS$_v$	3.41

At this point, it is important to highlight that the time complexity of our approach is not worse than the typical complexity of the evaluated algorithms. Because a combination of SMOTE and DBSCAN was used, the time complexity of the proposed strategy can be considered also as $\mathcal{O}(N^2)$. It can be concluded if the complexity of the different components is analyzed as follows:

The time complexity of SMOTE depends on its primary stages: (a) Generating Synthetic Samples, which is the core of SMOTE, is typically $\mathcal{O}(N \cdot M)$, where N is the number of minority class samples, and M is the number of synthetic samples to generate for each minority sample; (b) k-Nearest Neighbors Search, where the k-nearest neighbors for each minority class sample are found, is often dominated by the nearest neighbors search and can be up to $\mathcal{O}(N^2)$. Overall, the time complexity of SMOTE is often dominated by the nearest neighbor search and is commonly expressed as $\mathcal{O}(N^2)$ for the most time-consuming parts of the algorithm [48].

DBSCAN is a density-based clustering algorithm that identifies clusters based on the density of data points in the feature space. Its time complexity depends on the following typical stages: (a) Nearest Neighbor Search, with time complexity $\mathcal{O}(N^2)$; (b) Cluster Formation, which depends on the distribution of data points and the specific parameters of the algorithm, but its typical complexity can be linear or close to linear with respect to the number of data points, i.e., $\mathcal{O}(N)$. Therefore, the time complexity of DBSCAN is primarily determined by the nearest neighbor search, typically $\mathcal{O}(N^2)$ [49].

Finally, HDBSCAN is an extension of the traditional DBSCAN algorithm that introduces a hierarchical approach to clustering. The time complexity of HDBSCAN shares similarities with DBSCAN but also presents some differences due to its hierarchical structure. The time complexity of this variant depends on the following stages: (a) Nearest Neighbor Search, with time complexity $\mathcal{O}(N^2)$; (b) Hierarchical Clustering, which is often linear $\mathcal{O}(N)$ or close to linear; and b) Cluster Formation, similar to DBSCAN, that has a time complexity linear or close to linear $\mathcal{O}(N)$. HDBSCAN combines the nearest neighbor search, hierarchical clustering, and cluster formation steps; thus, considering that the most time-consuming part of the algorithm is the nearest neighbor search, the overall complexity is also $\mathcal{O}(N^2)$. In our approach, three main techniques were used (SMOTE, DBSCAN and HDBSCAN); consequently, the overall complexity of the proposed scheme is also $\mathcal{O}(N^2)$ in the worst case.

6. Conclusions

This study explores the feasibility of using density-based clustering methods to improve the effectiveness of balancing algorithms when highly imbalanced multi-class datasets are used in identification problems.

The SCUT algorithm, which was designed to deal with the class imbalance in multiple-class datasets, was used in this paper in combination with DBSCAN and HDBSCAN. The results were compared with K-means and the classical ROS and SMOTE approaches. Six hyperspectral remote images were used to perform the experimental validation, and a DNN was used as a classifier. The g-mean and Friedman's ranks were used to measure the performance of the balancing methods because both g-mean and Friedman's ranks provide valuable insights into the performance of deep learning models in classification tasks. The g-mean helps to evaluate the model's effectiveness in handling imbalanced datasets, while Friedman's rank helps to compare and rank multiple models based on their performance.

Our results indicate that the combination of SCUT with DBSCAN and HDBSCAN is an effective alternative to process highly imbalanced datasets in the domain of multi-class problems. These methods obtained a better Friedman's rank than the classical algorithms. *Greedy* techniques were also used to an intensive selection of *rv* and *minPts* for the density-based algorithms, but there was no evident improvement. As a general trend, density-based algorithms help obtain better classification results than the original unbalanced datasets or the classical oversampling techniques or neighborhood clustering method of K-means.

Future works should be performed to go deep in evaluating the effectiveness of density-based strategies in situations of extreme imbalance due to their promising potential, as is presented in this study.

Author Contributions: All authors played pivotal roles in conceiving, designing, and interpreting this study. J.C.M.M. and E.R.L. undertook the responsibilities of data collection and analysis. The initial manuscript draft was written by J.C.M.M. and R.A.E. Subsequent revision and editing were expertly conducted by E.E.G.G. and F.D.R.L. The collaborative efforts of all authors included reviewing and providing feedback on earlier manuscript iterations, ultimately culminating in the approval of the final version. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported under grants of project 17006.23-P from TecNM.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to extend their sincere appreciation to supporting under grants of project 17006.23-P from TecNM.

Conflicts of Interest: All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. Du, G.; Zhang, J.; Li, S.; Li, C. Learning from class-imbalance and heterogeneous data for 30-day hospital readmission. *Neurocomputing* **2021**, *420*, 27–35. [CrossRef]
2. Johnson, J.M.; Khoshgoftaar, T.M. Survey on deep learning with class imbalance. *J. Big Data* **2019**, *6*, 27. [CrossRef]
3. Leevy, J.L.; Khoshgoftaar, T.M.; Bauder, R.A.; Seliya, N. A survey on addressing high-class imbalance in big data. *J. Big Data* **2018**, *5*, 42. [CrossRef]
4. Lin, W.C.; Tsai, C.F.; Hu, Y.H.; Jhang, J.S. Clustering-based undersampling in class-imbalanced data. *Inf. Sci.* **2017**, *409–410*, 17–26. [CrossRef]
5. Kumar, P.; Bhatnagar, R.; Gaur, K.; Bhatnagar, A. Classification of Imbalanced Data:Review of Methods and Applications. *IOP Conf. Ser. Mater. Sci. Eng.* **2021**, *1099*, 012077. [CrossRef]
6. Sun, Z.; Song, Q.; Zhu, X.; Sun, H.; Xu, B.; Zhou, Y. A novel ensemble method for classifying imbalanced data. *Pattern Recognit.* **2015**, *48*, 1623–1637. [CrossRef]
7. Li, P.; Liang, T.g.; Zhang, K.h. Imbalanced Data Set CSVM Classification Method Based on Cluster Boundary Sampling. *Math. Probl. Eng.* **2016**, *2016*, 1540628. [CrossRef]
8. Fernández, A.; García, S.; Galar, M.; Prati, R.; Krawczyk, B.; Herrera, F. *Learning from Imbalanced Data Sets*; Springer Nature AG: Cham , Switzerland, 2018. [CrossRef]

9. Liang, G.; Zhang, C. An efficient and simple under-sampling technique for imbalanced time series classification. In Proceedings of the CIKM '12: 21st ACM International Conference on Information and Knowledge Management, Maui, HI, USA, 29 October–2 November 2012; Association for Computing Machinery: New York, NY, USA, 2012. [CrossRef]
10. Tsai, C.F.; Lin, W.C.; Hu, Y.H.; Yao, G.T. Under-sampling class imbalanced datasets by combining clustering analysis and instance selection. *Inf. Sci.* **2019**, *477*, 47–54. [CrossRef]
11. Rendón, E.; Alejo, R.; Castorena, C.; Isidro-Ortega, F.J.; Granda-Gutierrez, E.E. Data Sampling Methods to Deal With the Big Data Multi-Class Imbalance Problem. *Appl. Sci.* **2020**, *10*, 1276. [CrossRef]
12. Chawla, N.; Bowyer, K.; Hall, L.; Kegelmeyer, W. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res. (JAIR)* **2002**, *16*, 321–357. [CrossRef]
13. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In Proceedings of the Advances in Intelligent Computing, ICIC 2005, Hefei, China, 23–26 August 2005; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3644, pp. 878–887.
14. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, 1–6 June 2008; pp. 1322–1328. [CrossRef]
15. Ofek, N.; Rokach, L.; Stern, R.; Shabtai, A. Fast-CBUS: A Fast Clustering-Based Undersampling Method for Addressing the Class Imbalance Problem. *Neurocomputing* **2017**, *243*, 88–102. [CrossRef]
16. Singh, N.; Dhall, A. Clustering Based Over Sampling - Learning from Class Imbalanced Data. In Proceedings of the NIPS Workshop on Challenges and Opportunities for AI in Financial Services: The Impact of Fairness, Explainability, Accuracy, and Privacy, Montreal, QC, Canada, 7 December 2018; pp. 1–4.
17. Agrawal, A.; Viktor, H.L.; Paquet, E. SCUT: Multi-class imbalanced data classification using SMOTE and cluster-based under-sampling. In Proceedings of the 2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K), Lisbon, Portugal, 12–14 November 2015; Volume 1, pp. 226–234.
18. Sharma, K.K.; Seal, A.; Yazidi, A.; Krejcar, O. A New Adaptive Mixture Distance-Based Improved Density Peaks Clustering for Gearbox Fault Diagnosis. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–16. [CrossRef]
19. Sharma, K.K.; Seal, A.; Yazidi, A.; Selamat, A.; Krejcar, O. Clustering Uncertain Data Objects Using Jeffreys-Divergence and Maximum Bipartite Matching Based Similarity Measure. *IEEE Access* **2021**, *9*, 79505–79519. [CrossRef]
20. Maheshwari, R.; Mohanty, S.K.; Mishra, A.C. DCSNE: Density-based Clustering using Graph Shared Neighbors and Entropy. *Pattern Recognit.* **2023**, *137*, 109341. [CrossRef]
21. Maheshwari, R.; Mishra, A.C.; Mohanty, S.K. An entropy-based density peak clustering for numerical gene expression datasets. *Appl. Soft Comput.* **2023**, *142*, 110321. [CrossRef]
22. Haixiang, G.; Yijing, L.; Shang, J.; Mingyun, G.; Yuanyue, H.; Bing, G. Learning from class-imbalanced data: Review of methods and applications. *Expert Syst. Appl.* **2017**, *73*, 220–239. [CrossRef]
23. Ankita, B.; Abha, J. Analysis of Focussed Under-Sampling Techniques with Machine Learning Classifiers. In Proceedings of the 2021 IEEE/ACIS 19th International Conference on Software Engineering Research, Management and Applications (SERA), Kanazawa, Japan, 20–22 June 2021; pp. 91–96. [CrossRef]
24. Kubat, M. Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. In Proceedings of the Fourteenth International Conference on Machine Learning, Washington, DC, USA, 12–14 December 2000.
25. Batista, G.E.A.P.A.; Prati, R.C.; Monard, M.C. A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. *SIGKDD Explor. Newsl.* **2004**, *6*, 20–29. [CrossRef]
26. Pereira, R.M.; Costa, Y.M.; Silla, C.N. MLTL: A multi-label approach for the Tomek Link undersampling algorithm. *Neurocomputing* **2020**, *383*, 95–105. [CrossRef]
27. Hart, P. The condensed nearest neighbor rule (Corresp.). *IEEE Trans. Inf. Theory* **1968**, *14*, 515–516. [CrossRef]
28. Tomek, I. An Experiment with the Edited Nearest-Neighbor Rule. *IEEE Trans. Syst. Man Cybern.* **1976**, *SMC-6*, 448–452. [CrossRef]
29. Wilson, D.L. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Trans. Syst. Man Cybern.* **1972**, *SMC-2*, 408–421. [CrossRef]
30. Laurikkala, J. Improving Identification of Difficult Small Classes by Balancing Class Distribution. In Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine, Cascais, Portugal, 1–4 July 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 63–66.
31. Kaufman, L.; Rousseeuw, P. *Finding Groups in Data. An Introduction to Cluster Analysis*; John Wiley and Sons Inc.: New York, NY, USA, 1990.
32. Nugraha, W.; Maulana, S.; Sasongko, A. Clustering Based Undersampling for Handling Class Imbalance in C4.5 Classification Algorithm. *J. Phys. Conf. Ser.* **2020**, *1641*, 012014. [CrossRef]
33. Duda, R.O.; Hart, P.E.; Stork, D.G. *Pattern Classification*, 2nd ed.; Wiley: New York, NY, USA, 2001.
34. Batuwita, R.; Palade, V. A New Performance Measure for Class Imbalance Learning. Application to Bioinformatics Problems. In Proceedings of the 2009 International Conference on Machine Learning and Applications, Miami Beach, FL, USA, 13–15 December 2009; pp. 545–550. [CrossRef]
35. Bhattacharjee, P.; Mitra, P. A survey of density based clustering algorithms. *Front. Comput. Sci.* **2020**, *15*, 151308. [CrossRef]

36. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, Portland, OR, USA, 2–4 August 2010; AAAI Press: Washington, DC, USA, 1996; pp. 226–231.
37. Campello, R.J.G.B.; Moulavi, D.; Sander, J. Density-Based Clustering Based on Hierarchical Density Estimates. In *Advances in Knowledge Discovery and Data Mining*; Pei, J., Tseng, V.S., Cao, L., Motoda, H., Xu, G., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 160–172.
38. Milková, E. THE MINIMUM SPANNING TREE PROBLEM: Jarník's solution in historical and present context. *Electron. Notes Discret. Math.* **2007**, *28*, 309–316. [CrossRef]
39. Awad, M.; Khanna, R. *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*; Apress: Berkeley, CA, USA, 2015. [CrossRef]
40. Vince, A. A framework for the greedy algorithm. *Discret. Appl. Math.* **2002**, *121*, 247–260. [CrossRef]
41. Graña, M.; Veganzons, M.; Ayerdi, B. Hyperspectral Remote Sensing Scenes [Database] from Grupo de Inteligencia Computacional. 2021. Available online: http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes (accessed on 15 February 2022).
42. Sumbul, G.; Demİr, B. A Deep Multi-Attention Driven Approach for Multi-Label Remote Sensing Image Classification. *IEEE Access* **2020**, *8*, 95934–95946. [CrossRef]
43. Xiong, Z.; Cui, Y.; Liu, Z.; Zhao, Y.; Hu, M.; Hu, J. Evaluating explorative prediction power of machine learning algorithms for materials discovery using k-fold forward cross-validation. *Comput. Mater. Sci.* **2020**, *171*, 109203. [CrossRef]
44. Vega-Gutierrez, H.R.; Castorena, C.; Alejo, R.; Granda-Gutierrez, E.E. Comparative study of methods to obtain the number of hidden neurons of an auto-encoder in a high-dimensionality context. *IEEE Lat. Am. Trans.* **2020**, *18*, 2196–2203. [CrossRef]
45. Kapil, S.; Chawla, M. Performance Evaluation of K-means Clustering Algorithm with Various Distance Metrics. *Int. J. Comput. Appl.* **2015**, *110*, 12–16. [CrossRef]
46. Tharwat, A. Classification assessment methods. *Appl. Comput. Inf.* **2021**, *17*, 168–192. [CrossRef]
47. Friedman, M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *J. Am. Stat. Assoc.* **1937**, *32*, 675–701. [CrossRef]
48. Juez-Gil, M.; Arnaiz-González, A.; Rodríguez, J.J.; López-Nozal, C.; García-Osorio, C. Approx-SMOTE: Fast SMOTE for Big Data on Apache Spark. *Neurocomputing* **2021**, *464*, 432–437. [CrossRef]
49. Zhao, Y.; Liu, X.; Li, X. An improved DBSCAN algorithm based on cell-like P systems with promoters and inhibitors. *PLoS ONE* **2018**, *13*, e0200751. [CrossRef] [PubMed]