International Journal of Innovative Research and Scientific Studies, 8(3) 2025, pages: 1703-1720

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

Dynamic weighted cluster-sampling: An optimized cohesive method for improving data quality in the context of big data

Benabderrahmane MOUTASSEM[1], Laouni DJAFRI[2*], Abdelkader GAFOUR[3]

[1,3]*Department of Computer Science, Djilali Liabes University, Sidi Bel Abes, Algeria.*
[1]*Laboratory of Environmental and Energy Systems, Ali Kafi University Center, Tindouf, Algeria.*
[2]*Department of Mathematics, Ibn Khaldoun University, Tiaret, Algeria.*
[2]*LIM laboratory, Ibn Khaldoun University, Tiaret, Algeria.*
[3]*EEDIS laboratory, Djillali Liabes University, Sidi Bel Abbes, Algeria.*

Corresponding author: Laouni DJAFRI (*Email: laouni.djafri@univ-tiaret.dz*)

Abstract

In the field of data mining, imbalanced big data has emerged as a critical challenge, characterized by a disproportionate distribution of classes within large datasets. This phenomenon often results in biased models that underperform on minority classes, compromising the overall effectiveness of predictive analytics. Standard machine learning algorithms may struggle to accurately classify underrepresented instances, leading to predictions that reflect majority class tendencies rather than the true underlying patterns. To effectively address these challenges, it is imperative to employ advanced methods. This work presents a novel hybrid approach designed to mitigate the challenges of imbalanced big data classification effectively by employing clustering and sampling methods. Our proposed approach aims to reduce data volume, enhance veracity (improving performance metrics), and accelerate execution time, all while preserving essential attributes and ensuring data reliability. The results demonstrate that our approach achieves superior accuracy, AUC, F1-score, and G-means metrics compared to scenarios lacking data balancing strategies. Furthermore, we evaluate our proposed method against current methods in the field using large imbalanced datasets. Notably, our method exhibits an impressive accuracy rate approaching 100%, with improvements ranging from 17% to 22% across all performance metrics assessed, thus underscoring its effectiveness in addressing the challenges associated with imbalanced big data classification.

Keywords: Big data mining, clustering, cross-validation, imbalanced data, machine learning, sampling.

DOI: 10.53894/ijirss.v8i3.6878
Funding: This study received no specific financial support.
History: Received: 24 March 2025 / **Revised:** 25 April 2025 / **Accepted:** 28 April 2025 / **Published:** 9 May 2025

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Publisher: Innovative Research Publishing

1. Introduction

The realm of scientific computing is currently experiencing a surge in diverse data sources characterized by fine granularity and low latency, collectively termed "Big Data." This type of data surpasses the storage, processing, and analytical capabilities of traditional databases. Whether generated by users or automated processes, Big Data offers deeper insights into contemporary realities. Laney [1] described Big Data in 2001 through three dimensions: Volume, Variety, and Velocity, known as the three V's. Machine learning encounters substantial challenges in managing Big Data, particularly due to the extensive execution time required by algorithms. To mitigate these issues, frameworks like Hadoop [2] and the MapReduce model have been developed, enabling machine learning to function in distributed environments [3, 4]. Research has largely focused on enhancing algorithms via distributed architectures and parallel computing [5, 6]. For instance, one study [7] improved the Random Forest method by introducing dual pruning, which allows for parallel computations under the MapReduce algorithm, thereby boosting classification performance in Big Data. Another work [8] examined the impact of Random Under-Sampling (RUS) on imbalanced Big Data using Hadoop and Spark, revealing that RUS significantly improved classification performance in datasets with up to 12 million instances based on the Geometric Mean [9].

In recent years, there has been increasing scholarly interest in the challenges posed by imbalanced data [10-16]. Imbalanced data arises when class distributions are unequal, often resulting in learning difficulties. The majority classes represent a large portion of the dataset [17] while minority classes are underrepresented. For example, fraudulent transaction datasets frequently exhibit such imbalances, where only 5% of observations indicate fraud [18]. This issue is prevalent across various fields, including banking [18, 19], medical diagnostics [20], and reservoir identification in oil transport [21, 22]. Traditional classifiers aim to enhance overall accuracy [23] but often favor the majority class in imbalanced scenarios, leading to significant misclassification of minority classes [17, 24].

This paper presents the proposed approach, which integrates a clustering method and the SMOTE technique to tackle class imbalance in Big Data. The approach involves applying the K-Means (Two Levels) algorithm twice: first to cluster big datasets, then using SMOTE to generate synthetic examples of the minority class within each cluster. K-Means is reapplied for under-sampling the majority classes in each cluster. Results indicate that the proposed method is highly effective in addressing class imbalance.

The paper is organized as follows: Section 2 reviews the approach to class imbalance, highlighting recent methods in Big Data. Section 3 discusses the research methodology, including cluster sampling and model construction, emphasizing the selection of appropriate evaluation metrics for machine learning models. Section 4 presents experimental results and a detailed discussion, while Section 5 concludes with insights and future research directions.

2. Related Work

The last decade has witnessed widespread adoption of supervised learning algorithms for classification tasks involving imbalanced datasets [25]. Despite their popularity, these approaches often fall short in capturing essential data characteristics, leading to suboptimal prediction accuracy. Multiple solutions have emerged [15, 26, 27] to tackle imbalanced data challenges, particularly in the big data domain, which we examine in this work. The scientific community has demonstrated increasing attention to imbalanced learning challenges [10, 13, 16]. Such datasets present unique difficulties in classification tasks, primarily due to the disproportionate representation of different classes in training samples. Researchers have developed three main solution categories: data manipulation approaches, algorithmic modifications, and cost-based strategies [12, 28, 29]. Data manipulation approaches seek to rebalance class distributions through sampling modifications [30]. These methodologies restructure datasets while preserving the original loss function mechanics, either through minority class enhancement or majority class reduction [31, 32]. The field has produced numerous sampling innovations, from basic random sampling techniques [30] to more sophisticated approaches like SMOTE [33] and specialized oversampling methods [34, 35], each presenting distinct benefits and limitations [36]. Algorithmic modifications focus on enhancing existing classification methods to minimize majority class bias. This has led to the development of specialized boosting algorithms, particularly for traditional classifiers such as decision trees [37] and support vector machines [38]. Combined methodologies, particularly those incorporating ensemble techniques [39], integrate multiple strategies for handling class imbalances. Two prominent ensemble approaches are Bagging and Boosting. The former creates multiple classifiers using different data subsets, combining their outputs to minimize prediction variance [40]. The latter develops independent weak learners and aggregates their predictions through voting or averaging mechanisms [41]. Additionally, stacking employs a hierarchical approach, where initial model predictions serve as input features for a meta-model, which generates final predictions through weighted averaging [42].

The field of Big Data analysis has seen significant advancements in dimensionality reduction techniques aimed at optimizing predictive capabilities. Contemporary research largely centers on distributed computing architectures that facilitate parallel processing of massive datasets [6, 7] with particular attention to K-Means clustering and sampling methodologies. A groundbreaking contribution by Lin and colleagues [34] presented a clustering-based under-sampling framework for handling imbalanced data. Their dual-strategy approach reduces the majority class samples to match the minority class quantities through clustering. While their first method utilizes cluster centroids as majority class representatives, their second approach employs nearest-neighbor selection around these centers. Comparative analysis demonstrated the superior performance of their second strategy against five contemporary methods. Building on this foundation, researchers [43] developed the Cluster-Based Instance Selection (CBIS) methodology, which synthesizes clustering analysis with selective instance sampling. This innovative approach subdivides the majority class samples into distinct clusters before eliminating non-representative instances. When implemented with bagging and boosting-based MLP ensemble classifiers, CBIS demonstrated consistent superiority across various clustering algorithms and instance selection

methods. A notable application-specific study [44] extended these concepts to autism gene prediction, enhancing Lin's methodology through refined K-Means clustering for majority class sample reduction. Their results revealed marked improvements over both unbalanced approaches and existing methodologies, opening new avenues in autism-related gene identification. Addressing data quality concerns, the SMOTE-TLNN-DEPSO framework45 was introduced to eliminate noise and borderline cases through an innovative two-layer natural neighbors approach. This method incorporates a sophisticated hybrid optimization algorithm combining differential evolution with particle swarm optimization, proving particularly effective for datasets with attribute noise. A recent innovation [25] tackled imbalanced learning through a comprehensive tabular data sampling approach. This method combines K-nearest neighbors for efficient normal sample reduction with a novel Tabular Auxiliary Classifier GAN (TACGAN) for attack sample generation. The integration of dual loss functions in TACGAN's generator ensures minimal information loss, resulting in high-quality balanced datasets through the merger of undersampled normal data and synthesized attack samples.

3. Proposed Work

We propose the TLKMeans-S approach (Two-Level K-Means and SMOTE) to tackle the issue of imbalanced data by integrating Two-Level K-Means (for clustering (L1) and undersampling (L2)) with the SMOTE technique (for oversampling). Initially, we cluster the dataset to uncover natural groupings, helping to understand underlying patterns. Next, SMOTE generates synthetic examples of the minority class to enhance its representation by creating new data points between neighboring minority instances. We also perform undersampling to decrease the majority class instances, striving for a balanced distribution. These oversampling and undersampling processes are conducted separately to ensure accuracy. By combining these techniques, TLKMeans-S aims to balance classes and improve the performance of machine learning models on imbalanced datasets. This innovative integration of clustering, undersampling, and oversampling presents a promising solution for effectively managing imbalanced data, ultimately enhancing the accuracy and reliability of predictive models. The working principle of our proposed method, TLKMeans-S, is visually depicted in Figure 1.

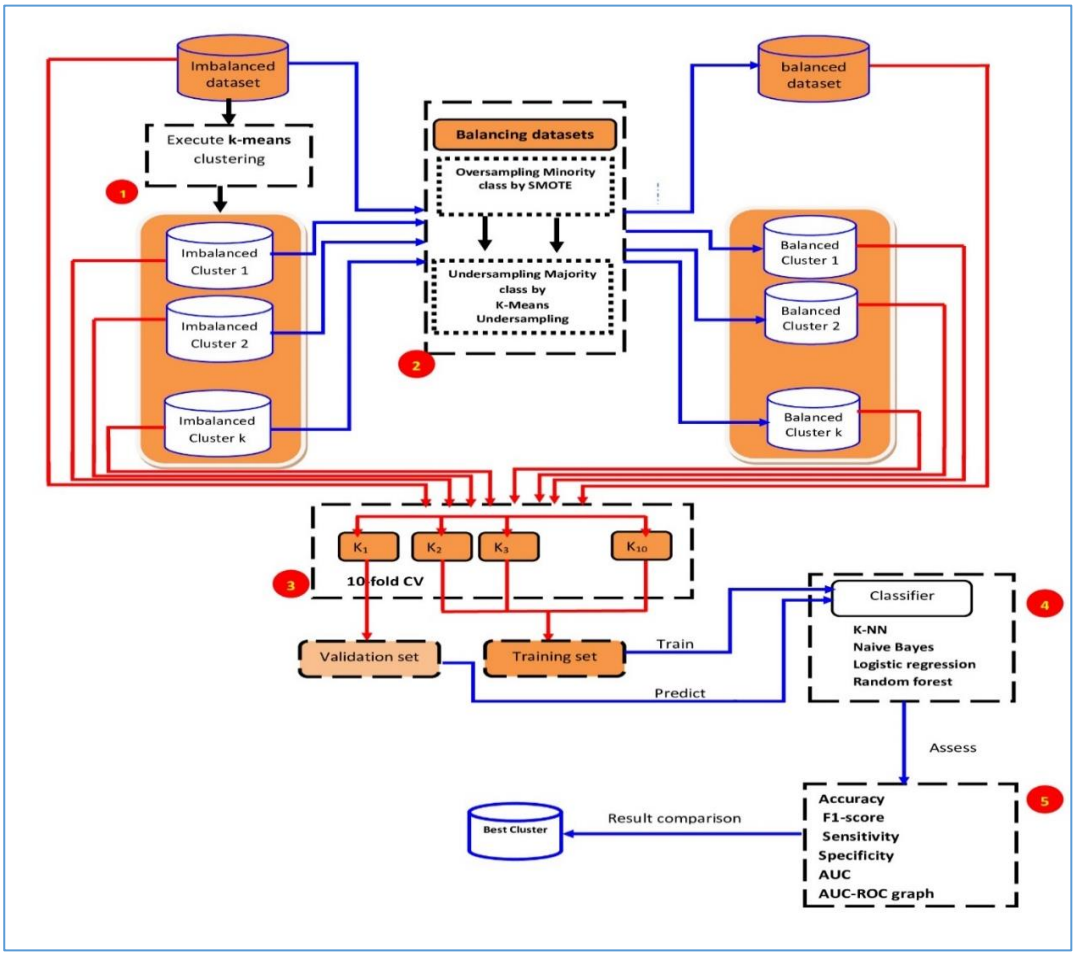


Figure 1.
Operating scenario of our proposed TLKMeans-S approach.

The operating principle of our approach mode consists of the following steps.

- Clustering: The dataset is partitioned into k clusters using the K-Means algorithm, grouping data based on similarities.
- Sampling and Balancing: The dataset is balanced through a combination of SMOTE for oversampling the minority class and K-Means undersampling to reduce the majority class instances, achieving a more equitable distribution.

- Evaluation: Cross-validation is employed to validate model performance, assessing generalization by splitting the data into multiple training and testing subsets.
- Model Training: Various supervised machine learning models, including Logistic Regression, Naive Bayes, kNN, and Random Forest, are trained for prediction, selected for their diverse characteristics.
- Comparison: Metrics from all models are evaluated and compared to identify the best clustering and classification combination for the dataset.

```
ALGORITHM TLKMeans-S

/* Global Variables and Data Structures */
INPUT:
    D: original_dataset[n] // Original dataset with n instances
    K: integer             // Number of clusters
    minority_class: class  // Class with fewer instances
    majority_class: class  // Class with more instances

OUTPUT:
    best_model: classifier   // Best performing model
    best_cluster: cluster    // Best performing cluster
    metrics: evaluation[]    // Performance metrics
/*********************************************
 * PHASE 1: CLUSTERING
 *********************************************/
PROCEDURE ClusterData(D, K)
    // Determine optimal K using Elbow Method
    best_k ← ElbowMethod(D)
    // Initialize centroids
    centroids[K] ← SelectRandomPoints(D, K)
    clusters[K] ← InitializeEmptyClusters()

    REPEAT
        // Assign points to nearest centroid
        FOR each point p in D DO
            min_dist ← INFINITY
            assigned_cluster ← 0
            FOR i FROM 1 TO K DO
                dist ← EuclideanDistance(p, centroids[i])
                IF dist < min_dist THEN
                    min_dist ← dist
                    assigned_cluster ← i
                END IF
            END FOR
            clusters[assigned_cluster].add(p)
        END FOR
        // Update centroids
        FOR i FROM 1 TO K DO
            centroids[i] ← CalculateMean(clusters[i])
        END FOR
    UNTIL convergence_reached
    RETURN clusters
END PROCEDURE

  /*****************************************
 * PHASE 2: DATA SAMPLING AND BALANCING
 *****************************************/
PROCEDURE Sample_BalanceDatasets(clusters)
  balanced_clusters[] ← []
  FOR each cluster C in clusters DO
       // Apply SMOTE for minority class
       synthetic_samples ← SMOTE(C.minority_class)
       // Apply K-Means undersampling for majority class
      reduced_majority ← KMeansUndersampling(C.majority_class)
       // Combine balanced data
  balanced_cluster←Combine(synthetic_samples,reduced_majority)
       balanced_clusters.add(balanced_cluster)
  END FOR
  RETURN balanced_clusters
END PROCEDURE
```

```
/*****************************************
 * PHASE 3: MODEL EVALUATION
 *****************************************/
PROCEDURE EvaluateModels(datasets)
    classifiers ← [LogisticRegression, NaiveBayes, kNN, RandomForest]
    metrics_list ← []

    FOR each dataset D in datasets DO
        // Perform 10-fold cross-validation
        folds ← CreateFolds(D, 10)

        FOR each classifier C in classifiers DO
            fold_metrics ← []

            FOR i FROM 1 TO 10 DO
                test_set ← folds[i]
                train_set ← folds - folds[i]

                model ← TrainModel(C, train_set)
                predictions ← model.predict(test_set)

                // Calculate metrics
                metrics ← CalculateMetrics(predictions, test_set.labels)
                fold_metrics.add(metrics)
            END FOR

            // Average metrics across folds
            avg_metrics ← CalculateAverageMetrics(fold_metrics)
            metrics_list.add(avg_metrics)

        END FOR
    END FOR
    RETURN SelectBestModel(metrics_list)
END PROCEDURE

/*****************************************
 * MAIN PROCEDURE
 *****************************************/
PROCEDURE Main()
    // Phase 1: Clustering
    clusters ← ClusterData(original_dataset, initial_K)

    // Phase 2: Balance datasets
    balanced_clusters ← BalanceDatasets(clusters)

    // Phase 3: Evaluate all models
   all_datasets ← Combine(original_dataset, clusters, balanced_clusters)
   best_model, best_cluster, metrics ← EvaluateModels(all_datasets)

    // Output results
    Output Results(best_model, best_cluster, metrics)
END PROCEDURE

FUNCTION ElbowMethod(data)
    wcss ← []  // Within-Cluster Sum of Squares

    FOR k FROM 1 TO max_k DO
        cluster_result ← KMeans(data, k)
        wcss.add(CalculateWCSS(cluster_result))
    END FOR

    RETURN FindElbowPoint(wcss)
END FUNCTION
```

Figure 2.
TLKMeans-S Algorithm Steps: Clustering, Balancing, and Model Evaluation.

The K-Means clustering algorithm is commonly applied to large datasets [45] and is well-suited for parallelization [46]. It operates by defining the number of clusters, k, randomly selecting k initial centroids, and calculating the distances between these centroids and all other points. Each point is then assigned to the nearest centroid, and the centroids are updated until they no longer change. The goal is to minimize the within-cluster sum of squares (WCSS), which measures the variance within each cluster.

The WCSS of the k cluster is given by equation 1.

$$W(C_k) = \sum_{\text{xi}\in C_k}(x_i - m_k)^2 \qquad (1)$$

The total WCSS for all the clusters is given by equation 2:

$$\sum_{k=1}^{n}\sum_{\text{xi}\in C_k}(x_i - m_k)^2 \qquad (2)$$

To determine the optimal number of clusters, the Elbow method48 is utilized, plotting the explained variation (WCSS) against the number of clusters and identifying the point where the curve bends, indicating the ideal k value.

Next, the dataset is balanced using the SMOTE algorithm [33] for oversampling minority classes and K-Means undersampling to reduce majority classes. SMOTE generates synthetic data for the minority class, aiming for an 80% increase, while the majority class is adjusted to match the number of minority instances. Metric estimates are obtained through 10-fold cross-validation, where the dataset is divided into ten folds, each containing 10% of the data. The algorithm is trained on samples from the other folds and tested on the current fold during each iteration. The balanced dataset is then used to train

various supervised machine learning models, including Logistic Regression, Naive Bayes, kNN, and Random Forest. To evaluate performance, we utilize metrics such as accuracy, F1-score, sensitivity, specificity, AUC, and ROC curves [47]. The results from the four classifiers are analyzed and compared to identify the most effective combination of clustering and classification techniques.

4. Experiment and Results

In this section, we discuss and analyze the effectiveness of the proposed approach (TLKmeans-S).

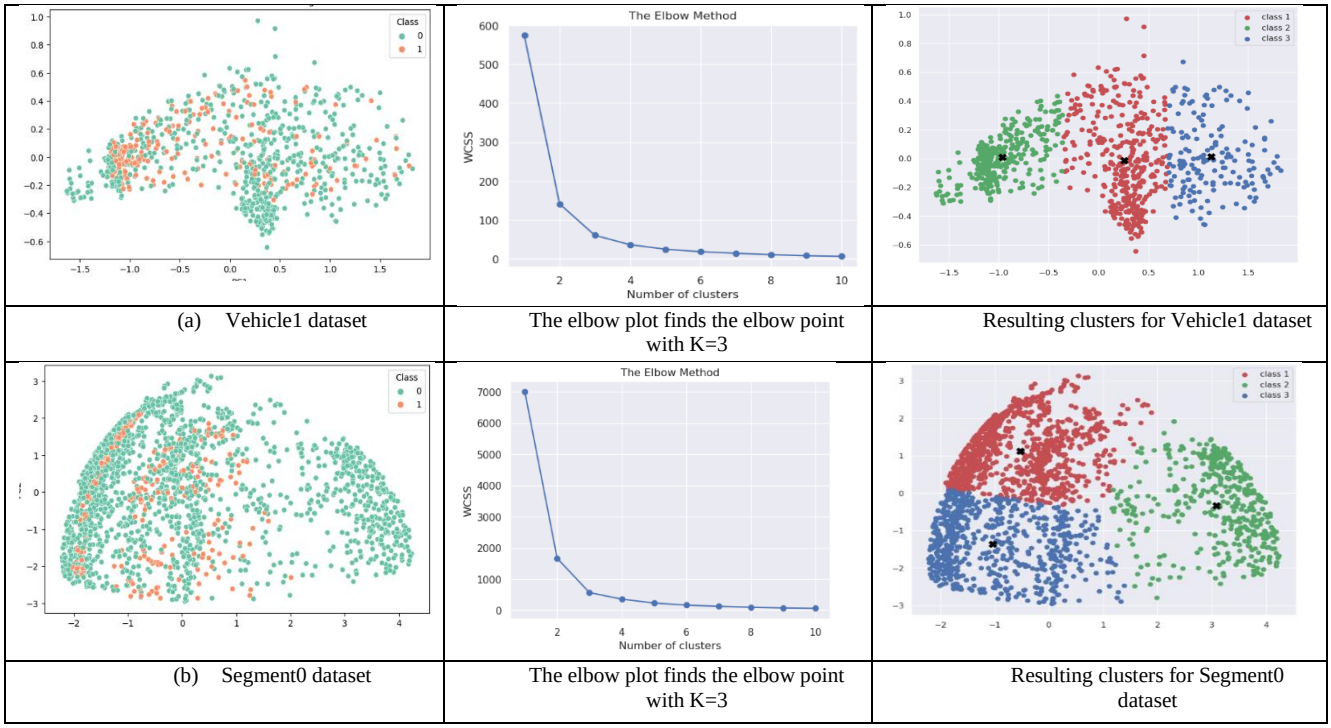
As an initial step, we carefully selected three datasets of varying sizes (small, medium, and big) to demonstrate the superiority of the proposed approach across different data sizes within the realm of imbalanced data. Next, we generalize our approach in the context of big data.

Table 1.
Details of the imbalanced datasets used in the first experimentation.

Datasets [1]	***Attributes (R/I/N)***	***Instances***	***Origin***	***IR***
Vehicle 1	18 (0/18/0)	846	Real world	2.9
Segment 0	10 (19/0/0)	2308	Real world	6.02
Credit card	31(30/1/0)	284807	Real world	577.9

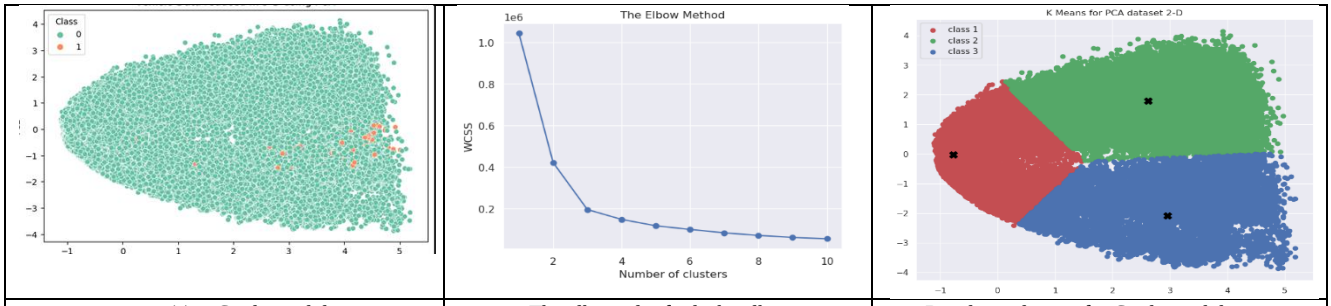
[a] https://sci2s.ugr.es/keel/imbalanced.php#sub60.

To evaluate the classification performance using clustering-based sampling methods, four distinct classifiers were developed: Logistic Regression, k-Nearest Neighbor (k-NN), Naïve Bayes, and Random Forest [7]. Furthermore, in this work, the chosen method for model assessment is referred to as 'K-fold cross validation (KFold-CV)'. This technique involves dividing the original dataset into K subsets. For our experiment, we will use K = 10 partitions. Additionally, we do not rely solely on accuracy as the evaluation metric for classifiers. We also compute metrics such as F1-score, Sensitivity, Specificity, AUC, and AUC-ROC curve. The clustering-based sampling method utilizes the K-Means algorithm to partition the dataset into K clusters. Figure 3 illustrates the datasets before and after applying the K-Means algorithm. Figures Figure 3 (a), Figure 3 (b), and Figure 3 (c) indicate that the optimal number of clusters, determined by the elbow method, is 3. Each cluster is treated as an individual dataset. In Test 1, we calculate the metrics for the four classifiers using the original imbalanced datasets. Following this, Test 2 involves recalculating the metrics after balancing the datasets through SMOTE and K-Means undersampling methods. Figures 4, 5, and 6 display the datasets and the resulting clusters before and after the balancing process.



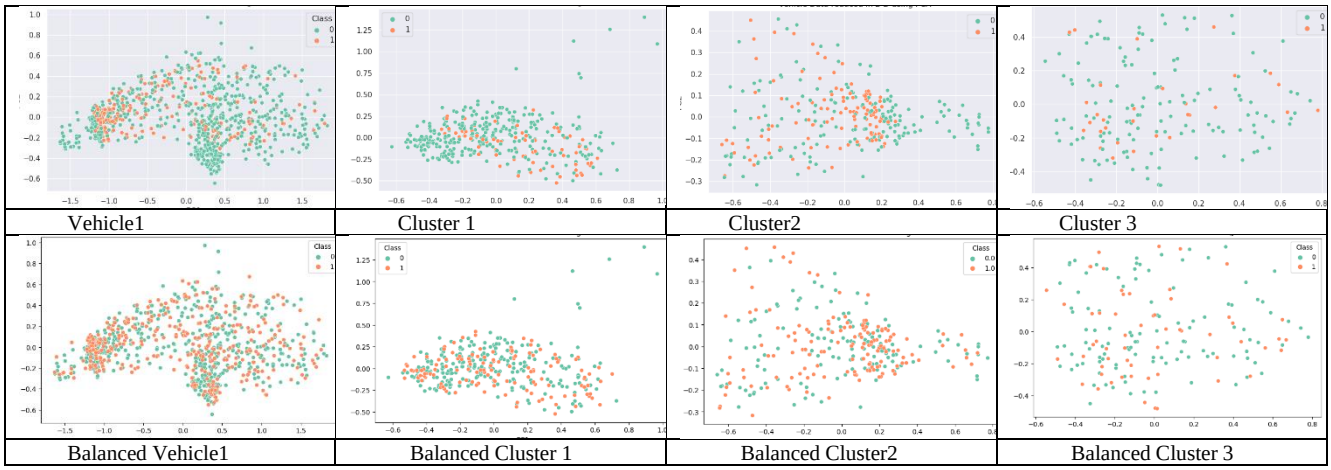
(a) Vehicle1 dataset | The elbow plot finds the elbow point with K=3 | Resulting clusters for Vehicle1 dataset

(b) Segment0 dataset | The elbow plot finds the elbow point with K=3 | Resulting clusters for Segment0 dataset



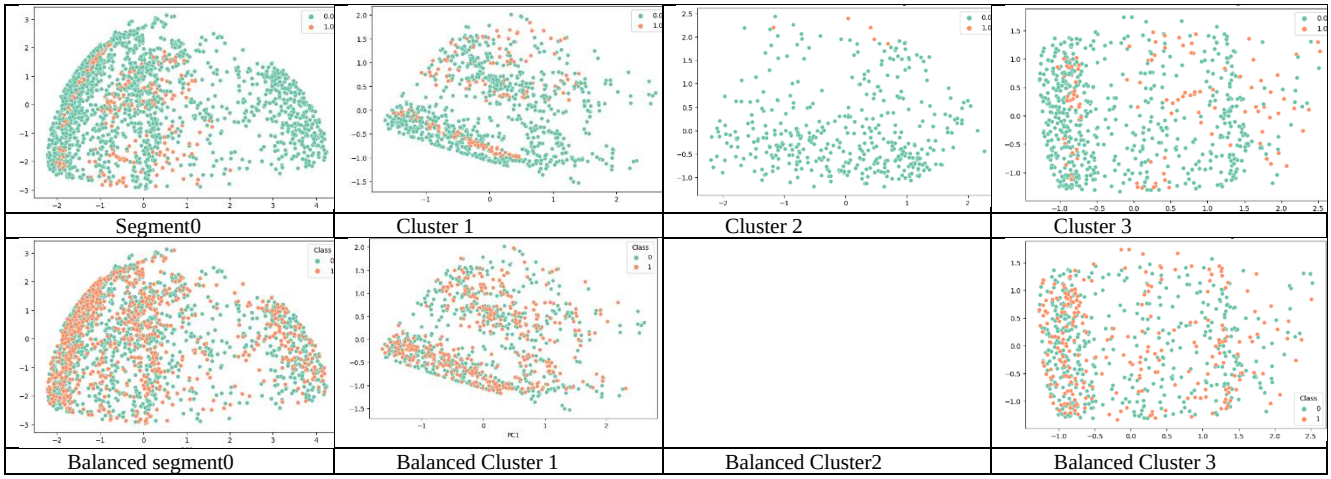
(c) Credit card dataset	The elbow plot finds the elbow point with K=3	Resulting clusters for Credit card dataset

Figure 3.
K-Means clustering algorithm with elbow method.



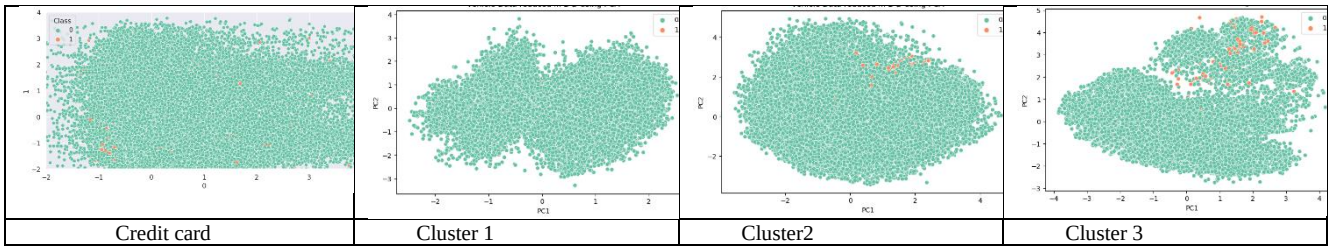
Vehicle1	Cluster 1	Cluster2	Cluster 3
Balanced Vehicle1	Balanced Cluster 1	Balanced Cluster2	Balanced Cluster 3

Figure 4.
Vehicle 1 dataset before and after balancing.

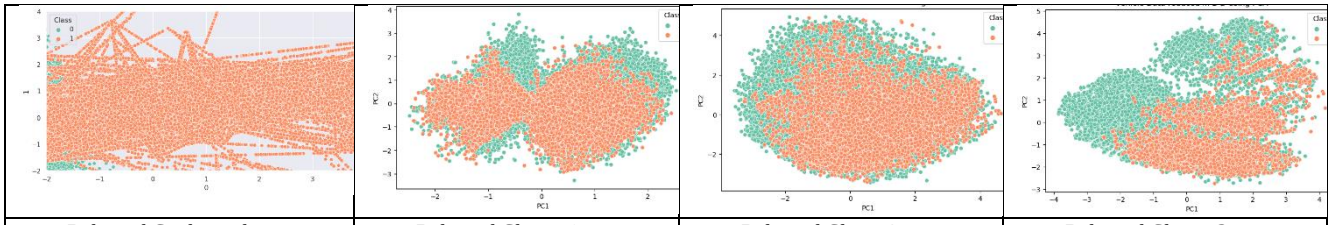


Segment0	Cluster 1	Cluster 2	Cluster 3
Balanced segment0	Balanced Cluster 1	Balanced Cluster2	Balanced Cluster 3

Figure 5.
Segment 0 dataset before and after balancing.



Credit card	Cluster 1	Cluster2	Cluster 3



Balanced Credit card	Balanced Cluster 1	Balanced Cluster2	Balanced Cluster 3

Figure 6.
Credit card big dataset before and after balancing.

Table 2 presents the total number of samples, minority samples, and majority samples for the three datasets and their clusters, both before and after balancing.

Table 2.
Datasets with class distribution.

	Vehicle 1 dataset			Segment 0 dataset			credit card dataset		
	Class 0	**Class 1**	**Total**	**Class 0**	**Class 1**	**Total**	**Class 0**	**Class 1**	**Total**
Original dataset	629	217	846	1979	329	2308	284315	492	284807
Cluster 1	311	70	381	884	194	1078	222249	247	222496
Cluster 2	181	117	298	442	5	447	34421	154	34575
Cluster 3	137	30	167	653	130	783	27645	91	27736
Original dataset balanced	503	503	1003	1583	1583	3166	227452	227452	454904
Cluster 1 balanced	248	248	496	707	707	1414	177799	177799	355598
Cluster 2 balanced	144	144	288	Nan	Nan	Nan	27536	27536	55072
Cluster 3 balanced	109	109	218	522	522	1044	22116	22116	44232

Test 1: In the first step, we utilize the complete datasets (vehicle 1, segment 0, and credit card) to evaluate the performance of the four classifiers. The results for various metrics are detailed in Tables 3-5. In the second step, we apply the clustering algorithm to partition the dataset into k clusters. We then assess the metrics for the classifiers within each cluster, with the results displayed in Tables 3-5.

Test 2: We replicate the process from Test 1 after balancing the datasets. Initially, we balance the original dataset using SMOTE and K-Means undersampling techniques and evaluate its performance with the four classifiers. Next, we apply the clustering algorithm to divide the dataset into k clusters and subsequently balance each cluster using SMOTE and K-Means undersampling. We then evaluate the metrics for the classifiers for the balanced clusters.

In this section, we conduct an empirical comparison of our proposed method against benchmark methods to address several questions regarding the learning algorithms. Before presenting the results, it's essential to discuss key considerations related to imbalanced datasets. Working with imbalanced data poses inherent challenges in machine learning, and it's crucial that our model evaluation remains unbiased. Therefore, it is generally advisable to use the F1-score rather than accuracy for such datasets. The AUC score is particularly beneficial as it incorporates prediction probabilities, offering a more comprehensive evaluation than the F1-score. Consequently, we recommend prioritizing the AUC over accuracy for imbalanced datasets [48-50].

Table 3 presents the classification outcomes for the Vehicle1 dataset as determined by four different classifiers: logistic regression, Naive Bayes, k-Nearest Neighbors (kNN), and Random Forest. The table displays the accuracy, F1-score, and AUC metrics for both Test 1 and Test 2.

Table 3.
Performance metrics evaluation of the four classifiers on vehicle1 dataset.

		Logistic Regression		Naive Bayes		kNN		Random Forest	
	Measure	Test 1	Test 2	Test 1	Test 2	Test 1	Test 2	Test 1	Test 2
Original dataset	Accuracy	0.80861	0.84499	0.68556	0.6868	0.76829	0.82808	0.79681	0.8659
	F1-score	0.56879	0.85016	0.50916	0.69675	0.53728	0.84626	0.53753	0.87487
	AUC	0.87278	0.91882	0.71431	0.72796	0.82928	0.8928	0.87143	0.94416
Cluster 1	Accuracy	0.82422	0.90331	0.7587	0.73976	0.81619	0.87506	0.81646	0.89122
	F1-score	0.45866	0.90625	0.5175	0.76209	0.45361	0.88888	0.41857	0.89901
	AUC	0.90337	0.95148	0.79653	0.81785	0.8396	0.92541	0.88093	0.97059
Cluster 2	Accuracy	0.86943	0.86798	0.70517	0.71207	0.67529	0.68387	0.72851	0.78128
	F1-score	0.82616	0.87123	0.69783	0.7453	0.60301	0.71995	0.65774	0.80384
	AUC	0.93854	0.93724	0.80161	0.80867	0.77489	0.78539	0.84975	0.85867
Cluster 3	Accuracy	0.8261	0.90411	0.65662	0.76169	0.83162	0.86688	0.83199	0.91299
	F1-score	NaN	0.90965	0.40699	0.79058	NaN	0.88248	NaN	0.91369
	AUC	0.86081	0.95008	0.74267	0.82207	0.73974	0.90905	0.7696	0.97843
Average clusters	Accuracy	0.83992	0.8918	0.70683	0.73784	0.77437	0.8086	0.79232	0.86183
	F1-score	0.64241	0.89571	0.54077	0.76599	0.52831	0.83044	0.53815	0.87218
	AUC	0.90091	0.94627	0.78027	0.8162	0.78474	0.87328	0.83343	0.9359

The comparison of classifier performance between Test 1 and Test 2 revealed significant improvements across all classifiers when datasets were balanced. Specifically, Logistic Regression showed increases of 3.64% in accuracy, 28.14% in F1-score, and 4.60% in AUC, with Cluster 3 demonstrating the best performance. Naïve Bayes exhibited a 0.12% rise in accuracy, an 18.76% increase in F1-score, and a 1.63% boost in AUC, notably with Cluster 3 achieving a 10.51% gain in accuracy and a 38.36% rise in F1-score. kNN also improved, with accuracy rising by 5.98%, F1-score by 30.90%, and AUC by 6.35%, particularly in Cluster 1. Random Forest showed the most impressive results, with a 6.90% increase in accuracy, a 33.73% rise in F1-score, and a 7.27% improvement in AUC, especially in Cluster 3. Overall, Test 2 consistently outperformed Test 1, indicating that balanced datasets enhance performance, with Cluster 3 emerging as the top performer, particularly excelling in AUC metrics. The Random Forest classifier was the most effective across all metrics, highlighting its superiority when datasets are balanced.

We will compare the results of each classifier by examining the performance graphs across different clusters, focusing specifically on three key metrics: accuracy, F1-score, and AUC.

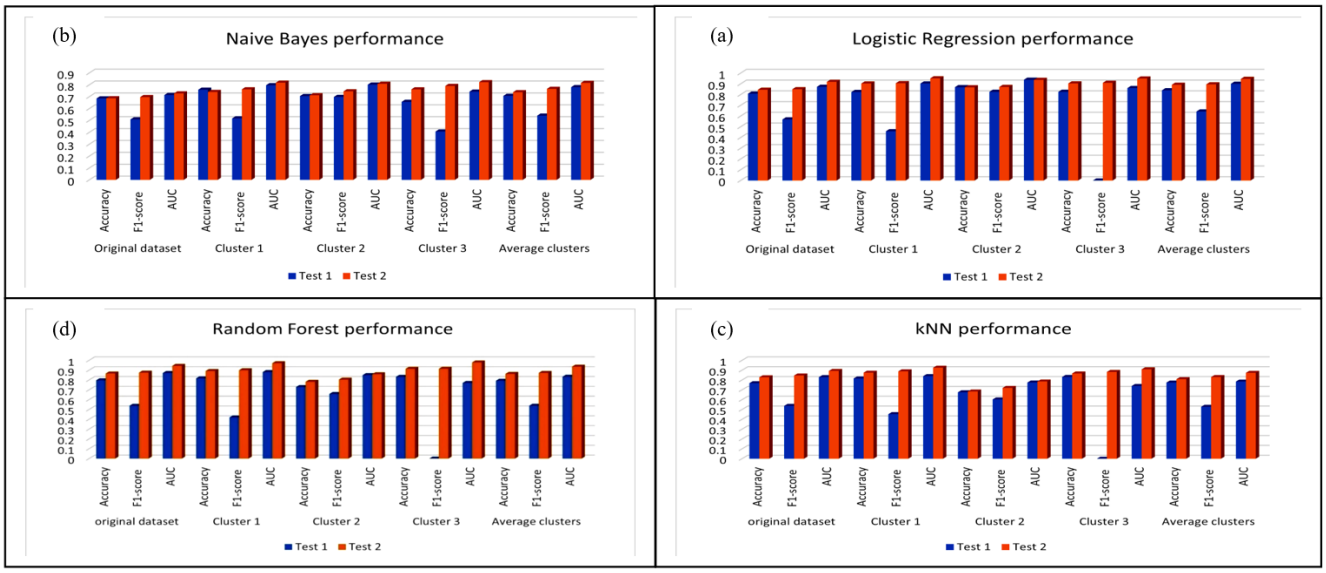
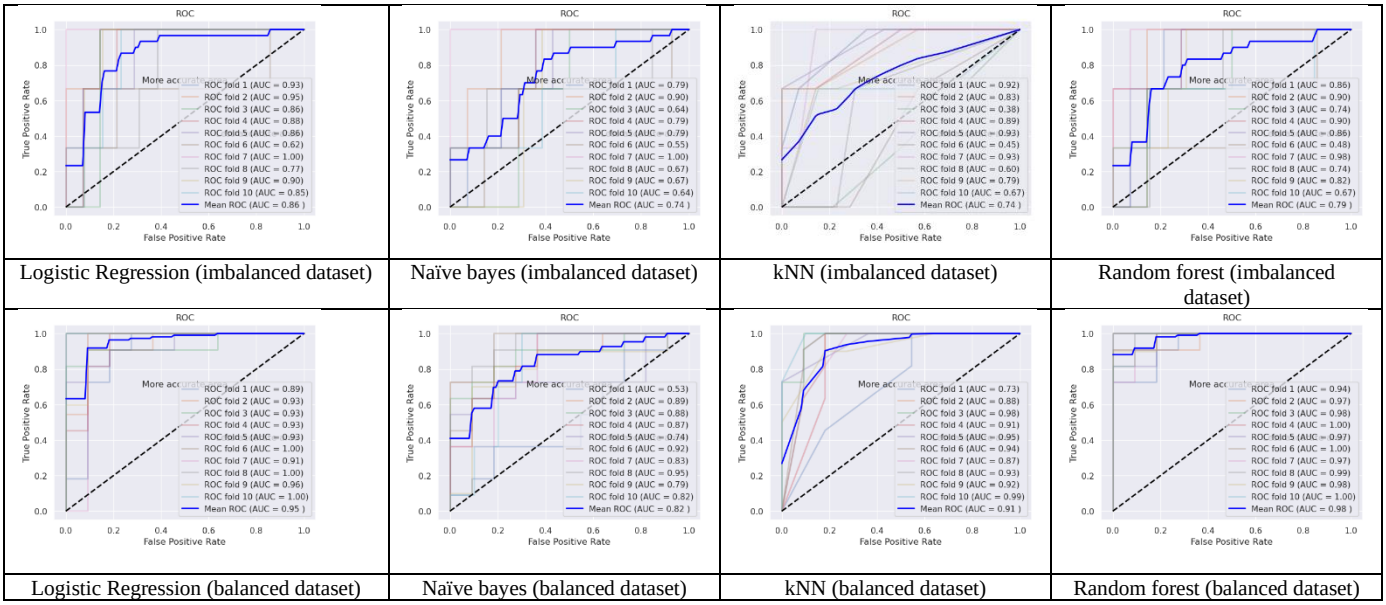
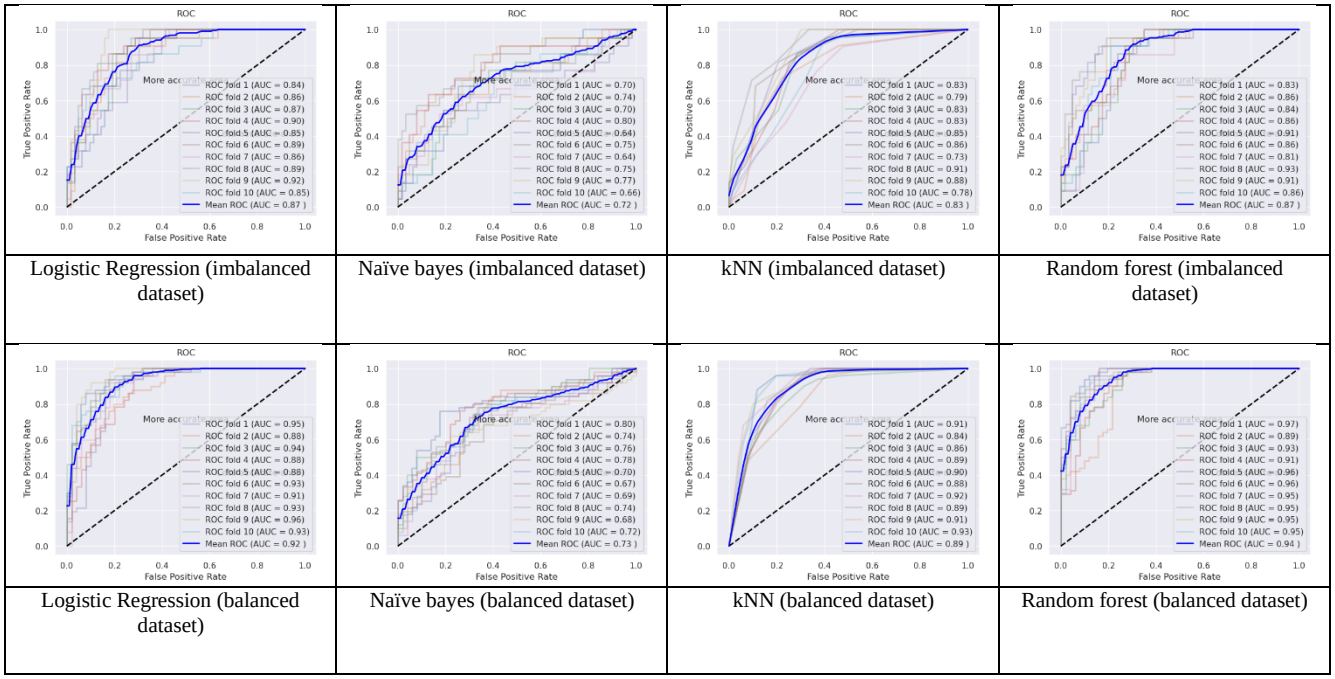


Figure 7.
Performance metrics classifiers on vehicle1 dataset.



Logistic Regression (imbalanced dataset)	Naïve bayes (imbalanced dataset)	kNN (imbalanced dataset)	Random forest (imbalanced dataset)
Logistic Regression (balanced dataset)	Naïve bayes (balanced dataset)	kNN (balanced dataset)	Random forest (balanced dataset)

Figure 8.
AUC-ROC Curve of the four classifiers on vehicle1 dataset.



Logistic Regression (imbalanced dataset)	Naïve bayes (imbalanced dataset)	kNN (imbalanced dataset)	Random forest (imbalanced dataset)
Logistic Regression (balanced dataset)	Naïve bayes (balanced dataset)	kNN (balanced dataset)	Random forest (balanced dataset)

Figure 9.
AUC-ROC Curve of the four classifiers on cluster 3 (vehicle1 dataset).

1 recorded the best AUC at 0.90. For the Naive Bayes classifier (Figure 7 (b)), Test 1 revealed that cluster 1 had the highest accuracy of 80%, while cluster 2 excelled in F1-score with 0.78 and AUC of 0.82. In Test 2, cluster 3 outperformed with an accuracy of 85%, F1-score of 0.81, and AUC of 0.88. For the kNN classifier (Figure 7 (c)), Test 1 showed that cluster 3 achieved the highest accuracy of 82%, cluster 2 led in F1-score with 0.80, and cluster 1 had the best AUC at 0.86. In Test 2, cluster 1 topped all metrics with an accuracy of 86%, F1-score of 0.83, and AUC of 0.89. Finally, for the Random Forest classifier (Figure 7 (d)), Test 1 indicated that cluster 3 had the highest accuracy at 85%, cluster 2 achieved the best F1-score of 0.84, and cluster 1 recorded the highest AUC at 0.91. In Test 2, cluster 3 delivered the best metrics across the board, achieving an impressive accuracy of 90%, F1-score of 0.88, and AUC of 0.95.

In general, these results provide valuable insights into the performance of each classifier across different clusters, highlighting where optimal results are achieved for each metric. The AUC-ROC curves (Figure 8 and Figure 9) further illustrate significant improvements after dataset balancing, with a 7% increase for the original dataset and a 12% increase for cluster 3. The Random Forest classifier emerged as the best performer with an AUC of 98%, indicating its superior ability to classify the positive class in this dataset compared to Logistic Regression, Naive Bayes, and kNN.

Table 4 presents a comprehensive overview of the performance metrics, including accuracy, F1-score, and AUC, for both Test 1 and Test 2 evaluations of the Segment0 dataset. The results encompass three individual clusters, alongside the average metrics calculated across all clusters collectively.

Table 4.
Performance metrics evaluation of the four classifiers on Segment 0 dataset.

		Logistic Regression		Naive Bayes		kNN		Random Forest	
	Measure	**Test 1**	**Test 2**	**Test 1**	**Test 2**	**Test 1**	**Test 2**	**Test 1**	**Test 2**
Original dataset	Accuracy	0.9974	0.99684	0.83317	0.89261	0.9922	0.99463	0.9974	0.99747
	F1-score	0.99082	0.99683	0.63208	0.90225	0.97259	0.99467	0.99072	0.99748
	AUC	0.99904	0.99933	0.98207	0.98552	0.99933	0.99904	0.99992	1
Cluster 1	Accuracy	0.99907	0.99787	0.82193	0.89037	0.99072	0.99364	0.99814	0.99929
	F1-score	0.9973	0.99789	0.6625	0.90008	0.97405	0.99374	0.99473	0.99928
	AUC	1	0.99998	0.97515	0.98085	0.99971	0.99927	1	1
Cluster 2	Accuracy	0.99907	NaN	0.82193	NaN	0.99072	NaN	0.9972	NaN
	F1-score	0.9973	NaN	0.6625	NaN	0.97405	NaN	0.99188	NaN
	AUC	1	NaN	0.97515	NaN	0.99971	NaN	1	NaN
Cluster 3	Accuracy	0.99234	0.98564	0.75479	0.84483	0.99362	0.9952	0.99745	0.99713
	F1-score	0.97631	0.98558	0.57816	0.86609	0.98089	0.99526	0.992	0.99712
	AUC	0.99553	0.9983	0.99623	0.99739	0.99569	0.99802	0.99953	1
Average` clusters	Accuracy	0.99682	0.99176	0.79955	0.8676	0.99169	0.99442	0.9976	0.99821
	F1-score	0.9903	0.99174	0.63438	0.88309	0.97633	0.9945	0.99287	0.9982
	AUC	0.99851	0.99914	0.98217	0.98912	0.99837	0.99864	0.99984	1

The results presented in Table 4 illustrate the varied performance of several classifiers, highlighting trends between Test 1 and Test 2 evaluations. Logistic Regression experienced a slight overall accuracy decrease of 0.06%, yet demonstrated improvements in F1-score by 0.60% and AUC by 0.03%. Notably, cluster 1 saw a 0.12% drop in accuracy, accompanied by a 0.06% F1-score enhancement, while cluster 3 exhibited a more pronounced 0.67% accuracy decline but significant gains in F1-score (0.93%) and AUC (0.28%). In contrast, Naive Bayes showcased substantial improvements, with an accuracy increase of 5.94%, a remarkable 27.02% rise in F1-score, and a 0.34% boost in AUC, particularly excelling in cluster 3 with a 9% accuracy increase and a 28.79% F1-score improvement, leading to an overall average enhancement of 6.80% in accuracy and 24.87% in F1-score across clusters. Meanwhile, Random Forest displayed negligible changes, with only a 0.01% accuracy increase and slight declines in F1-score (0.68%) and AUC (0.01%); cluster 1 suffered a 0.12% accuracy drop and a 0.45% F1-score reduction, while cluster 3 saw a minor accuracy increase of 0.03% but declines in both F1-score (0.51%) and AUC (0.05%).

From Figure 10, we observe an enhancement in performance metrics, particularly in the AUC measure, across all classifiers when compared to the average clusters and the original dataset. For the logistic regression classifier (Figure 10 (a)), we noted improvements of 0.26% and 0.29% in T1 and T2, respectively, relative to the original dataset. In the case of the Naive Bayes classifier (Figure 10 (b)), improvements of 19.93% and 16.81% were recorded for T1 and T2, respectively, compared to the original dataset. The KNN classifier (Figure 10 (c)) also showed improvements of 0.64% in T1 and 0.26% in T2 relative to the original dataset. For the Random Forest classifier (Figure 10 (d)), we observed stability in the AUC for both scenarios (average clusters and original dataset), along with an increase in accuracy compared to the original dataset.

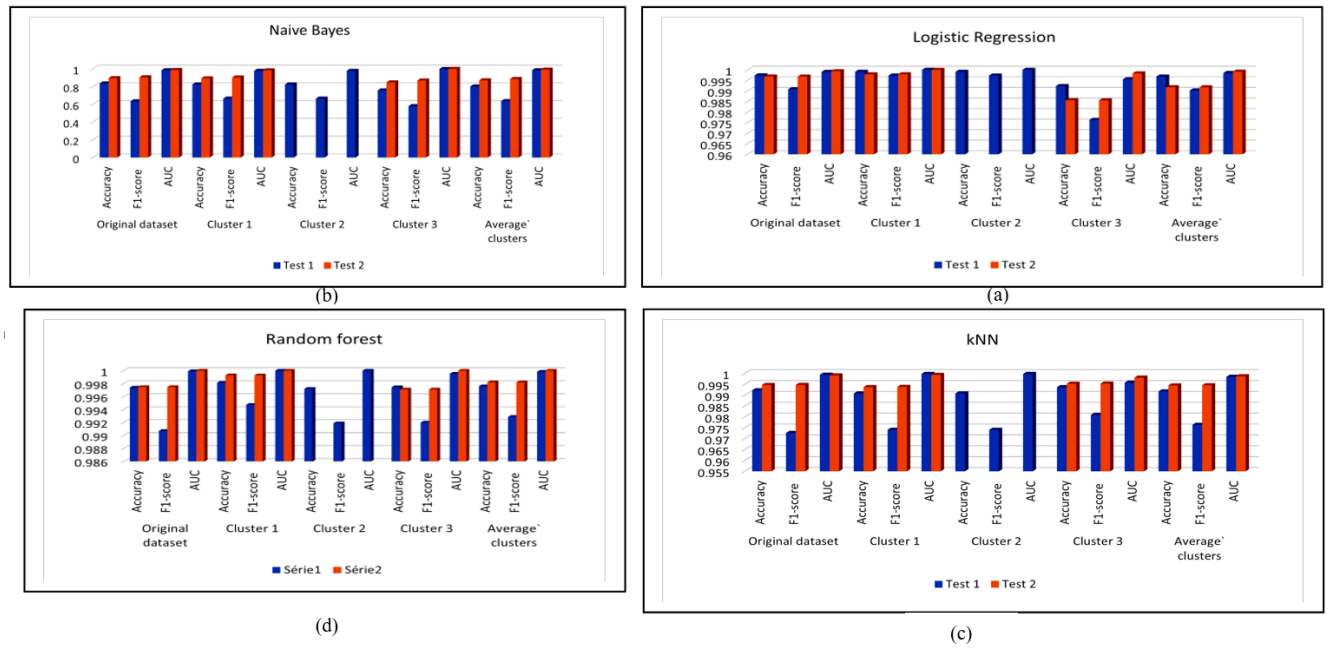


Figure 10.
Performance metrics of four classifiers on the Segment0 dataset.

Table 5 presents the classification results for the credit card dataset, which is large and extremely imbalanced, organized into four sub-tables for the Logistic Regression, Naive Bayes, kNN, and Random Forest classifiers. Each sub-table details

performance metrics, including accuracy, F1-score, and AUC for both Test 1 and Test 2. The results are categorized for the entire dataset, as well as for the three clusters, alongside the average metrics computed across the three clusters.

Table 5.
Performance metrics evaluation of the four classifiers on Credit card dataset.

		Logistic Regression		**Naive Bayes**		**kNN**		**Random Forest**	
	Measure	Test 1	Test 2	Test 1	Test 2	Test 1	Test 2	Test 1	Test 2
original dataset	Accuracy	0.99919	0.95973	0.9778	0.9189	0.99951	0.99861	0.99953	0.98751
	F1-score	0.72546	0.95878	0.11445	0.91409	0.84564	0.99861	0.84919	0.98737
	AUC	0.97595	0.992	0.96091	0.9601	0.92477	0.99959	0.99992	0.99966
Cluster 1	Accuracy	0.99938	0.9643	0.98135	0.91277	0.99968	0.99914	0.9997	0.99535
	F1-score	0.66489	0.96357	0.08878	0.90658	0.83923	0.99914	0.84997	0.99533
	AUC	0.95586	0.99524	0.95315	0.96001	0.91076	0.99977	0.9681	0.99994
Cluster 2	Accuracy	0.99873	0.96752	0.9648	0.92902	0.99873	0.99586	0.99873	0.99731
	F1-score	0.83341	0.96718	0.18384	0.92676	0.84938	0.99588	0.84064	0.99731
	AUC	0.98328	0.99664	0.96826	0.95996	0.93723	0.9986	0.97717	0.99992
Cluster 3	Accuracy	0.99874	0.96082	0.97725	0.90862	0.99895	0.99828	0.99906	0.9993
	F1-score	0.76349	0.96027	0.19704	0.90314	0.81249	0.99829	0.83275	0.9993
	AUC	0.97274	0.9934	0.97666	0.97718	0.91195	0.99941	0.98259	0.99995
Average clusters	Accuracy	0.99895	0.96421	0.97447	0.9168	0.99912	0.99776	0.99916	0.99732
	F1-score	0.75393	0.96367	0.15655	0.91216	0.8337	0.99777	0.84112	0.99731
	AUC	0.97063	0.99509	0.96602	0.96572	0.91998	0.99926	0.97595	0.99994

From Table 5, we notice that the classification results across different classifiers reveal notable trends and performance variations between Test 1 and Test 2. For the Logistic Regression classifier, the original dataset saw a decrease in accuracy of 3.95%, but significant improvements were noted in the F1-score (23.33%) and AUC (1.60%). Cluster-wise, Cluster 1 led with an F1-score increase of 29.87% and an AUC boost of 3.94%, despite a 3.51% drop in accuracy. The Naive Bayes classifier exhibited a 5.89% decline in accuracy overall, yet a remarkable 79.96% increase in F1-score, with minor changes in AUC. Cluster 1 again performed best in F1-score and AUC. The kNN classifier showed minimal accuracy loss (0.09%), but a substantial F1-score improvement (15.30%) and AUC increase (7.48%). Cluster 1 excelled in both metrics, while Cluster 3 led in AUC improvements. Lastly, the Random Forest classifier experienced a slight accuracy decrease (1.20%) but significant enhancements in F1-score (13.82%) and minor AUC changes. Cluster 3 demonstrated the most favorable outcomes, outperforming others in accuracy and F1-score measures. As a result, Cluster 3 emerged as the best-performing sample across classifiers, suggesting distinct characteristics that enhance model performance.

The evaluation of classifiers across three key metrics: accuracy, F1-score, and AUC yields insightful findings. For the Logistic Regression classifier (Fig. 10 (a)), Test 1 revealed that Cluster 1 had the highest accuracy (99.99% in Test 1 and 99.98% in Test 2), while Cluster 2 excelled in F1-score, showing an improvement of 10% in Test 1 compared to the original dataset. In the Naive Bayes classifier (Fig. 10 (b)), Cluster 1 led in accuracy for Test 1 (99.96%), whereas Cluster 2 demonstrated the best F1-score, with an improvement of 8% in Test 1 compared to the original dataset. Cluster 3 also showed an improved F1-score, with a 3% enhancement in Test 1. The kNN classifier (Fig. 10 (c)) maintained consistent performance metrics, with no significant improvements in Test 1 or Test 2. Lastly, for the Random Forest classifier (Fig. 10 (d)), Cluster 1 delivered the best accuracy and F1-score in Test 1, while Cluster 3 showed the most favorable results across all metrics in both Test 1 and Test 2.

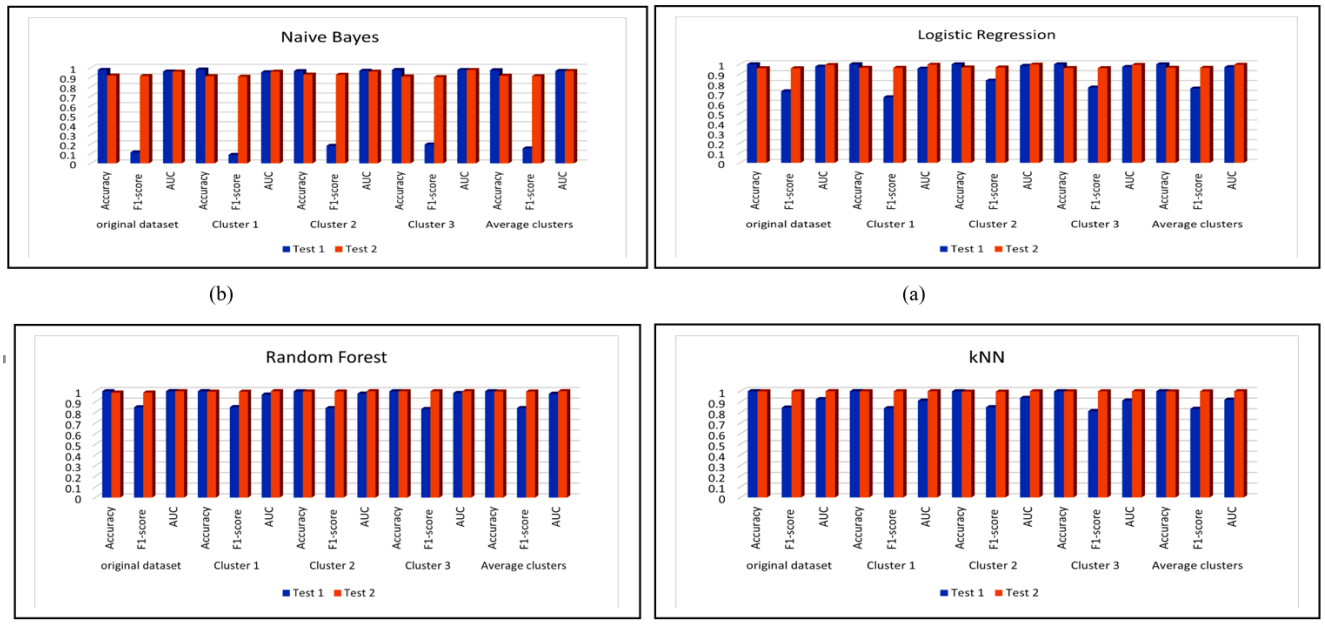


Figure 11.
Performance metrics of four classifiers on the Credit card dataset.

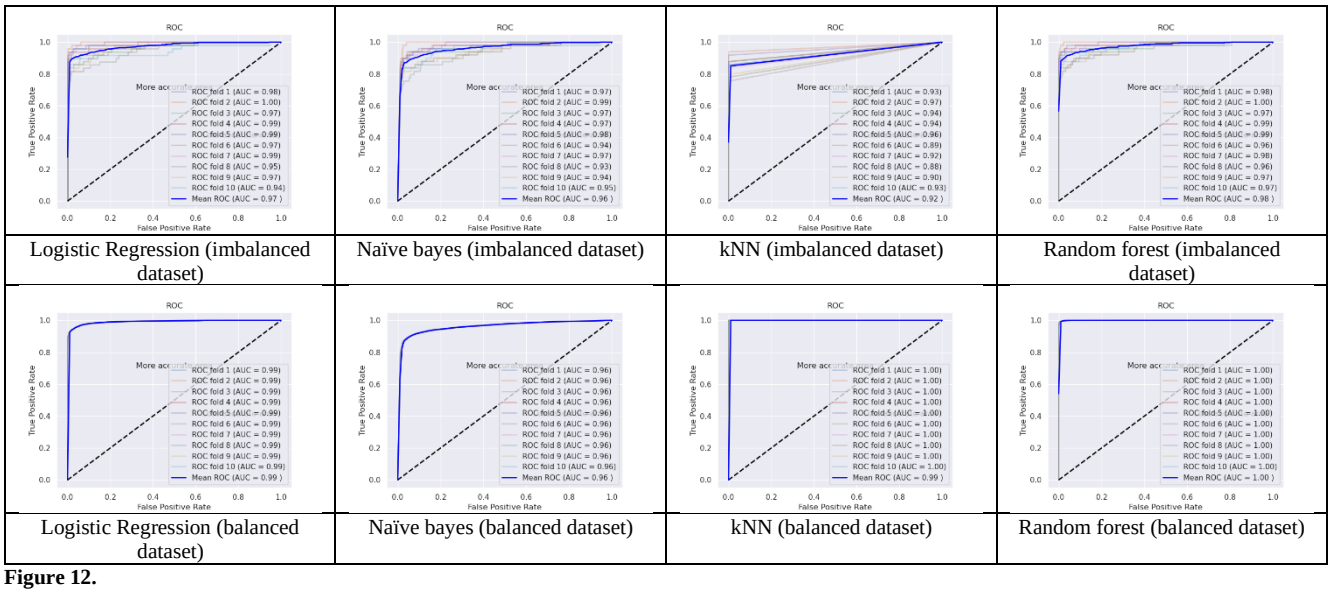


Figure 12.
AUC-ROC Curve of the four classifiers on credit card dataset.

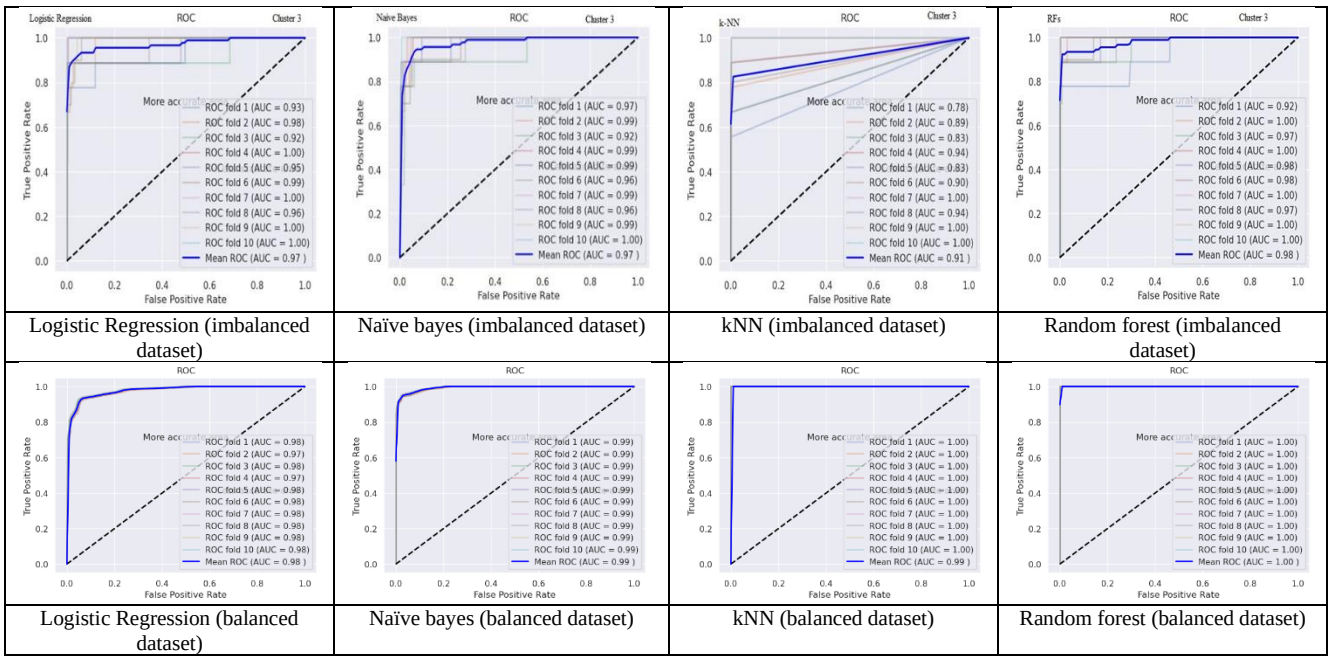


Figure 13.
AUC-ROC Curve of the four classifiers on cluster 3 (credit card dataset).

After a thorough analysis of the AUC-ROC curves from Figure 12, we observe that the maximum results achieved by all classifiers remain consistent and identical in both experiments, both without data balancing and after applying SMOTE for balancing. Notably, the Naive Bayes classifier exhibits a slight improvement of 1% in performance after balancing. The AUC values obtained from the ROC curves underscore the excellent performance of all four classifiers, indicating their robust ability to distinguish between classes effectively. This consistent performance across both experimental conditions highlights the reliability of these classifiers in handling the dataset.

In our evaluation of classifiers on imbalanced datasets, we employed data balancing techniques like SMOTE, which significantly enhanced classification outcomes, evidenced by a notable 2% performance improvement in both the original dataset and Cluster 3 (Figure 13). Among the classifiers tested, Random Forest was the top performer, achieving a perfect accuracy of 100% and surpassing Logistic Regression, Naive Bayes, and kNN in AUC-ROC curves, highlighting its robustness in identifying positive instances. Our extensive evaluation compared performance metrics from two tests, with Test 2 showing improvements across various datasets: for the Vehicle1 dataset, accuracy increased by 0.30% to 4.36%, F1-score by 8.75% to 18.76%, and AUC by 2.05% to 3.99%. In the Segment0 dataset, accuracy improved by 0.02% to 5.94%, and F1-score by 0.20% to 23.98%, while AUC saw a marginal improvement of 0% to 0.38%. The Credit Card dataset displayed a remarkable F1-score enhancement of 14.93% to 72.97% and an AUC improvement of 0% to 4%, though accuracy declined by 0.04% to 5.23%. Despite a slight trade-off in accuracy for some datasets, our proposed method effectively addressed class imbalance, demonstrating superior results in key metrics compared to existing machine learning methods [46, 51-55]. Notably, our method achieved a 3.29% improvement in accuracy and 5.58% in sensitivity over previously proposed methods [55]. This work advances techniques for handling class imbalance and lays a foundation for further exploration in various domains.

We emphasize the significant role of runtime in processing imbalanced data, particularly in the context of big data, and thus, we will present the findings related to runtime efficiency. Table 6 showcases the runtime of classifiers across different datasets and their clusters, both before and after balancing. This assessment is geared towards evaluating the scalability of models and the influence of sample size, with execution times graphically illustrated in Figures 14-16 for the four classifiers across the three datasets.

Table 6.
Runtime of classifiers for Test 1 and Test 2.

Vehicle1	**original dataset**	**cluster1**	**cluster2**	**cluster3**
logistic regression	0.332	0.28	0.259	0.155
Naive Bayes	0.029	0.034	0.038	0.025
kNN	0.149	0.043	0.052	0.046
Random Forest	5.289	4.131	4.524	3.924
a. Vehicle1 dataset				
Segment0	original dataset	cluster1	cluster2	cluster3
logistic regression	0.346	0.189	0.058	0.169
Naive Bayes	0.048	0.047	0.014	0.031
kNN	0.382	0.163	0.033	0.098
Random Forest	6.872	4.555	1.897	4.571
b. Segment0 dataset				
Credit card	original dataset	cluster1	cluster2	cluster3
logistic regression	34.283	35.918	9.559	4.883
Naive Bayes	4.137	3.156	0.631	0.41
kNN	1623.613	938.321	24.643	12.678
Random Forest	4822.329	2393.983	314.266	196.373
c. Credit card dataset				
Vehicle1	original dataset	cluster 1	cluster 2	cluster3
logistic regression	0.413	0.282	0.182	0.165
Naive Bayes	0.031	0.032	0.028	0.028
kNN	0.241	0.081	0.038	0.038
Random Forest	5.473	4.647	4.149	3.846
a. Vehicle1 dataset				
Segment0	original dataset	cluster1	cluster2	cluster3
logistic regression	0.333	0.204	NaN	0.224
Naive Bayes	0.054	0.034		0.036
kNN	0.343	0.185		0.118
Random Forest	10.054	5.689		5.564
b. Segment0 dataset				
Credit card	original dataset	cluster1	cluster2	cluster3
logistic regression	133.853	84.165	12.175	9.471
Naive Bayes	7.192	4.676	0.696	0.54
kNN	3888.993	2047.857	49.416	32.812
Random Forest	5004.4	3131.446	454.579	277.991
c. Credit card dataset				

All classifiers exhibit a more substantial reduction in runtime for clusters compared to the original datasets, with kNN demonstrating the most notable reduction in both tests. In the Vehicle1 and Segment0 datasets, the classifiers are ranked based on runtime reduction as follows: Logistic Regression, Random Forest, and Naive Bayes. Conversely, for the Credit Card dataset, the ranking is: Random Forest, Logistic Regression, and Naive Bayes. The kNN classifier stands out for its exceptional performance in reducing execution time, achieving reductions ranging from 71.49% to 77.45% for the Vehicle1 dataset, 65.66% to 91.23% for the Segment0 dataset, and a remarkable 99.16% to 99.22% for the Credit Card dataset. Particularly noteworthy are the profound runtime reductions observed in the Credit Card dataset, with reductions ranging from 85.76% to 99.22% in Test 1 and from 92.92% to 99.16% in Test 2.

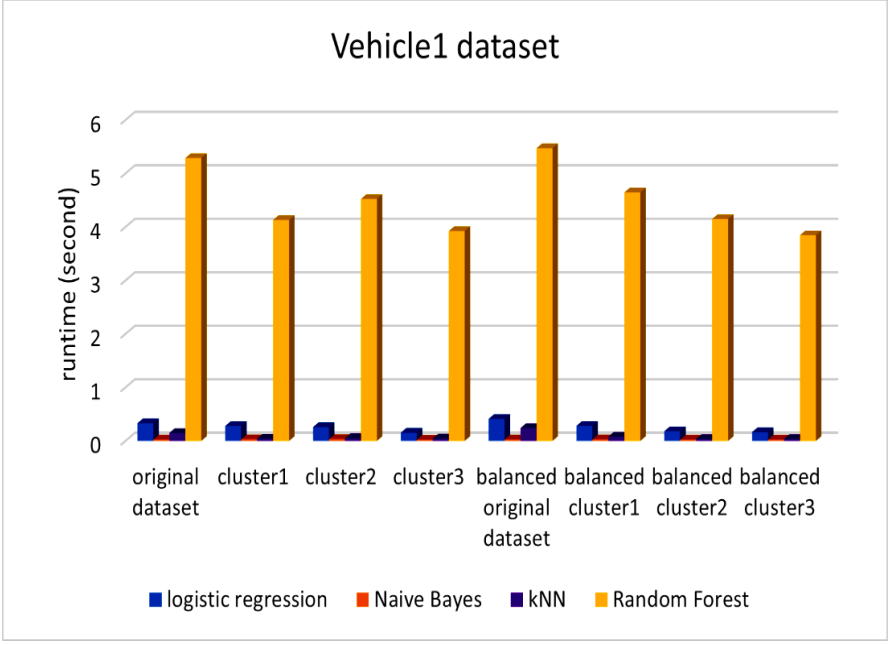


Figure 14.
Classifiers Runtime in seconds for Test 1 and 2 on Vehicle1 dataset.

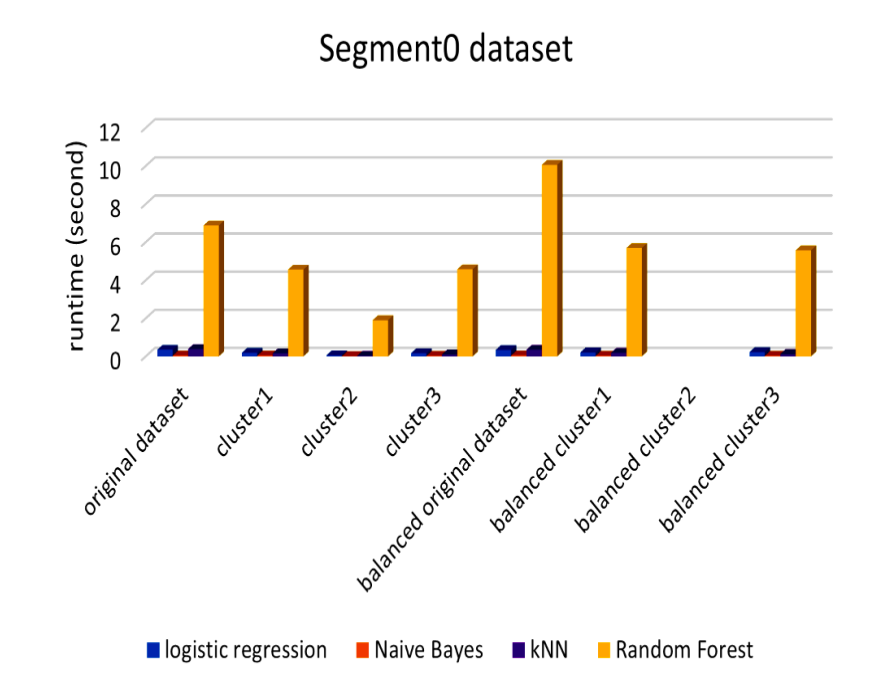


Figure 15.
Classifiers Runtime in seconds for Test 1 and 2 on Segment0 dataset.

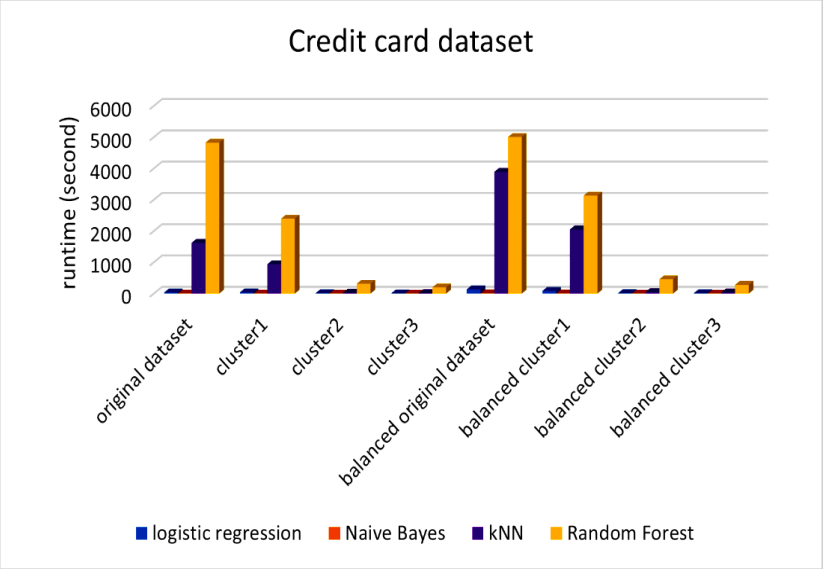


Figure 16.
Classifiers Runtime in seconds for Test 1 and 2 on Credit card dataset.

Based on previous experiments, we conclude that the proposed TLKMeans-S method is effective for handling imbalanced data of all sizes, including small, medium, and large datasets. Big data presents a significant challenge in terms of class imbalance. Therefore, we will now discuss some recent methods that have been compared in the context of imbalanced big data classification. To further validate our proposed method's applicability in real-world scenarios and demonstrate its superiority, we conducted a separate experiment using large and highly imbalanced original datasets.

Table 7.
Performance Metrics of TLKMeans-S in the Context of Big Data.

Methods	Stack-AdaB Kumari, et al. [56]			ECSEL Daud, et al. [57]			SMOTE-kTLNN Sun, et al. [58]			TLKMeans-S		
CICIDS2017	0.718	0.673	0.742	0.748	0.681	0.753	0.811	0.780	0.737	0.874	0.843	0.877
Sharafaldin, et al. [59]	0.786	0.742	0.759	0.860	0.875	0.872	0.847	0.885	0.862	0.914	0.905	0.932
KDD Cup-99[1]	0.841	0.785	0.867	0.848	0.880	0.861	0.797	0.783	0.798	0.833	0.922	0.964
Mammography[2]	0.798	0.785	0.766	0.890	0.888	0.867	0.849	0.825	0.869	0.944	0.955	0.936
BoT_IoT[3]	0.874	0.785	0.862	0.811	0.795	0.822	0.874	0.881	0.897	0.978	0.996	0.962
Higgs Boson[4]												

The experimental results presented in Table 7 demonstrate the superior performance of TLKMeans-S in handling imbalanced big datasets. The proposed method achieves exceptional performance metrics: 99.60% F1-score, 96.4% accuracy, and 97.8 G-means. Comparative analysis reveals significant improvements over existing methods, with increases of 16.70% in F1-score, 21.09% in accuracy, and 16.6% in G-means.

[a] https ://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[b] https://www.bcsc-research.org/data/mammography_dataset/digitial-mammo-dataset-download.
[c] https ://www.kaggle.com/datasets/vigneshvenkateswaran/bot-iot
[d] https ://archive.ics.uci.edu/dataset/280/higgs

5. Conclusion and Future Work

In this paper, we introduce the TLKMeans-S approach to tackle the issue of imbalanced large dataset classification. Our proposed approach integrates a novel hybrid cluster-sampling framework that combines oversampling and undersampling techniques, focusing on a two-level K-Means clustering-undersampling strategy. The process involves dividing the dataset into k clusters using a first-level K-Means clustering (initial balancing), treating each cluster as a separate dataset, and then applying SMOTE along with a second-level K-Means (final balancing) to achieve balanced classes. The cluster with the most favorable evaluation metrics is selected as the representative sample of the original dataset. Additionally, our work makes a significant contribution to big data mining processing by reducing data volume, improving performance metrics (veracity), and execution speed (velocity), and maintaining data reliability (validity).

In future work, we aim to develop tailored cluster-sampling methods to effectively address imbalanced classes in large datasets. Furthermore, we will explore the implementation and enhancement of cluster-sampling methods on advanced

[1] https ://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[2] https://www.bcsc-research.org/data/mammography_dataset/digitial-mammo-dataset-download.
[3] https ://www.kaggle.com/datasets/vigneshvenkateswaran/bot-iot
[4] https ://archive.ics.uci.edu/dataset/280/higgs

computing platforms such as GPUs, parallel CPUs, or distributed systems to improve runtime performance. Additionally, we plan to investigate the applicability of our method in addressing other challenges in supervised classification tasks, such as noisy data and missing data.

References

[1] D. Laney, "3D data management: Controlling data volume, velocity and variety," *META Group Research Note*, vol. 6, no. 70, pp. 1-10, 2001.

[2] T. White, *Hadoop: The definitive guide*. Sebastopol: O'Reilly, 2015.

[3] S. Del Río, V. López, J. M. Benítez, and F. Herrera, "On the use of mapreduce for imbalanced big data using random forest," *Information Sciences*, vol. 285, pp. 112-137, 2014. https://doi.org/10.1016/j.ins.2014.03.043

[4] K. Sarwar, S. H. Ripon, N. Dey, A. S. Ashour, and V. Santhi, "A MapReduce approach to diminish imbalance parameters for big deoxyribonucleic acid dataset," *Computer Methods and Programs in Biomedicine*, vol. 131, pp. 191-206, 2016. https://doi.org/10.1016/j.cmpb.2016.04.005

[5] B. Furht and F. Villanustre, *Introduction to big data*. Cham Springer, 2016.

[6] L. Djafri, "Dynamic distributed and parallel machine learning algorithms for big data mining processing," *Data Technologies and Applications*, vol. 56, no. 4, pp. 558-601, 2022. https://doi.org/10.1108/dta-06-2021-0153

[7] L. Djafri, D. Amar Bensaber, and R. Adjoudj, "Big Data analytics for prediction: Parallel processing of the big learning base with the possibility of improving the final result of the prediction," *Information Discovery and Delivery*, vol. 46, no. 3, pp. 147-160, 2018.

[8] A. Fernández, S. del Río, N. V. Chawla, and F. Herrera, "An insight into imbalanced big data classification: Outcomes and challenges," *Complex & Intelligent Systems*, vol. 3, pp. 105-120, 2017. https://doi.org/10.1007/s40747-017-0037-9

[9] L. Djafri and Y. Gafour, "Machine learning algorithms for big data mining processing: A review," presented at the International Conference on Artificial Intelligence and its Applications, Cham: Springer International Publishing, 2021.

[10] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263-1284, 2009. https://doi.org/10.1109/TKDE.2008.239

[11] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 04, pp. 687-719, 2009.

[12] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information Sciences*, vol. 250, pp. 113-141, 2013.

[13] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221-232, 2016.

[14] R. C. Prati, G. E. Batista, and D. F. Silva, "Class imbalance revisited: A new experimental setup to assess the performance of treatment methods," *Knowledge and Information Systems*, vol. 45, pp. 247-270, 2015.

[15] Z. Wang, T. Liu, X. Wu, and C. Liu, "A diagnosis method for imbalanced bearing data based on improved SMOTE model combined with CNN-AM," *Journal of Computational Design and Engineering*, vol. 10, no. 5, pp. 1930-1940, 2023. https://doi.org/10.1093/jcde/qwad081

[16] H. Patel, D. Singh Rajput, G. Thippa Reddy, C. Iwendi, A. Kashif Bashir, and O. Jo, "A review on classification of imbalanced data for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 16, no. 4, p. 1550147720916404, 2020.

[17] H. Liu, M. Zhou, and Q. Liu, "An embedded feature selection method for imbalanced data classification," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 703-715, 2019.

[18] M. E. Lokanan and K. Sharma, "Fraud prediction using machine learning: The case of investment advisors in Canada," *Machine Learning with Applications*, vol. 8, p. 100269, 2022. https://doi.org/10.1016/j.mlwa.2022.100269

[19] S. Ilyas, S. Zia, U. M. Butt, S. Letchmunan, and Z. un Nisa, "Predicting the future transaction from large and imbalanced banking dataset," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 1, pp. 1-14, 2020.

[20] Y.-S. Chen, "An empirical study of a hybrid imbalanced-class DT-RST classification procedure to elucidate therapeutic effects in uremia patients," *Medical & Biological Engineering & Computing*, vol. 54, no. 6, pp. 983-1001, 2016. https://doi.org/10.1007/s11517-016-1482-0

[21] X. Lan, C. Zou, Z. Kang, and X. Wu, "Log facies identification in carbonate reservoirs using multiclass semi-supervised learning strategy," *Fuel*, vol. 302, p. 121145, 2021. https://doi.org/10.1016/j.fuel.2021.121145

[22] K. Zhou, S. Li, X. Zhou, Y. Hu, C. Zhang, and J. Liu, "Data-driven prediction and analysis method for nanoparticle transport behavior in porous media," *Measurement*, vol. 172, p. 108869, 2021. https://doi.org/10.1016/j.measurement.2020

[23] M. Benabderrahmane, D. Laouni, and A.-K. Gaafour, "Big data veracity: Methods and challenges," 2022.

[24] S. Park, H. W. Lee, and J. Im, *Raking and relabeling for imbalanced data*. USA: Authorea Preprints, 2022.

[25] H. Ding, L. Chen, L. Dong, Z. Fu, and X. Cui, "Imbalanced data classification: A KNN and generative adversarial networks-based hybrid approach for intrusion detection," *Future Generation Computer Systems*, vol. 131, pp. 240-254, 2022. https://doi.org/10.1016/j.future.2022.01.026

[26] M. Priyadharshini, A. F. Banu, B. Sharma, S. Chowdhury, K. Rabie, and T. Shongwe, "Hybrid multi-label classification model for medical applications based on adaptive synthetic data and ensemble learning," *Sensors*, vol. 23, no. 15, p. 6836, 2023. https://doi.org/10.3390/s23156836

[27] S. Riyanto, S. S. Imas, T. Djatna, and T. D. Atikah, "Comparative analysis using various performance metrics in imbalanced data for multi-class text classification," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 6, pp. 1-9, 2023. https://doi.org/10.14569/IJACSA.2023.01406116

[28] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463-484, 2011. https://doi.org/10.1109/TSMCC.2011.2161285

[29] H. Kaur, H. S. Pannu, and A. K. Malhi, "A systematic review on imbalanced data challenges in machine learning: Applications and solutions," *ACM Computing Surveys (CSUR)*, vol. 52, no. 4, pp. 1-36, 2019. https://doi.org/10.1145/3343440

[30] H. Han, W.-Y. Wang, and B.-H. Mao, *Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning*. Berlin, Heidelberg: Springer, 2005.

[31] T. Wongvorachan, S. He, and O. Bulut, "A comparison of undersampling, oversampling, and SMOTE methods for dealing with imbalanced classification in educational data mining," *Information*, vol. 14, no. 1, p. 54, 2023. https://doi.org/10.3390/info14010054

[32] T. Zhu, X. Liu, and E. Zhu, "Oversampling with reliably expanding minority class regions for imbalanced data learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 6, pp. 6167-6181, 2022. https://doi.org/10.1109/TKDE.2022.3171706

[33] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321-357, 2002. https://doi.org/10.1613/jair.953

[34] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang, "Clustering-based undersampling in class-imbalanced data," *Information Sciences*, vol. 409, pp. 17-26, 2017. https://doi.org/10.1016/j.ins.2017.05.008

[35] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5718-5727, 2009. https://doi.org/10.1016/j.eswa.2008.06.108

[36] M. Bekkar, H. K. Djemaa, and T. A. Alitouche, "Evaluation measures for models assessment over imbalanced data sets," *Journal of Information Engineering and Applications*, vol. 3, no. 10, pp. 27–38, 2013.

[37] Y.-H. Shao, W.-J. Chen, J.-J. Zhang, Z. Wang, and N.-Y. Deng, "An efficient weighted Lagrangian twin support vector machine for imbalanced data classification," *Pattern Recognition*, vol. 47, no. 9, pp. 3158-3167, 2014.

[38] A. Mellor, S. Boukir, A. Haywood, and S. Jones, "Exploring issues of training data imbalance and mislabelling on random forest performance for large area land cover classification using the ensemble margin," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 105, pp. 155-168, 2015.

[39] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, pp. 185-197, 2009.

[40] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123-140, 1996.

[41] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119-139, 1997.

[42] I. El Mallahi, J. Riffi, and H. Ahmad, "Enhancing road traffic accident severity classification using the stacking method in machine learning models," *Preprints*, p. 202308, 2023. https://doi.org/10.20944/preprints202308.0169.v1

[43] C.-F. Tsai, W.-C. Lin, Y.-H. Hu, and G.-T. Yao, "Under-sampling class imbalanced datasets by combining clustering analysis and instance selection," *Information Sciences*, vol. 477, pp. 47-54, 2019. https://doi.org/10.1016/j.ins.2018.10.029

[44] X. T. Dang, D. H. Bui, T. H. Nguyen, T. Q. V. Nguyen, and D. H. Tran, "Prediction of autism-related genes using a new clustering-based under-sampling method," in *2019 11th International Conference on Knowledge and Systems Engineering (KSE), IEEE*, 2019, pp. 1-6.

[45] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, vol. 2, no. 3, pp. 283-304, 1998.

[46] F. H. Awad, M. M. Hamad, and L. Alzubaidi, "Robust classification and detection of big medical data using advanced parallel K-means clustering, YOLOv4, and logistic regression," *Life*, vol. 13, no. 3, p. 691, 2023. https://doi.org/10.3390/life13030691

[47] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, 2006. https://doi.org/10.1016/j.patrec.2005.10.010

[48] E. Zvornicanin, "Accuracy vs AUC in machine learning," 2024.

[49] S. Allwright, "AUC vs accuracy, which is the best metric?," 2024.

[50] J. Huang and C. X. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 299-310, 2005. https://doi.org/10.1109/TKDE.2005.50

[51] H. Ahmad, B. Kasasbeh, B. Aldabaybah, and E. Rawashdeh, "Class balancing framework for credit card fraud detection based on clustering and similarity-based selection (SBS)," *International Journal of Information Technology*, vol. 15, no. 1, pp. 325-333, 2023. https://doi.org/10.1007/s41870-022-00987-w

[52] A. Muaz, M. Jayabalan, and V. Thiruchelvam, "A comparison of data sampling techniques for credit card fraud detection," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, pp. 1-9, 2020.

[53] P. Gupta, A. Varshney, M. R. Khan, R. Ahmed, M. Shuaib, and S. Alam, "Unbalanced credit card fraud detection data: A machine learning-oriented comparative study of balancing techniques," *Procedia Computer Science*, vol. 218, pp. 2575-2584, 2023. https://doi.org/10.1016/j.procs.2023.01.231

[54] Y. Ding, W. Kang, J. Feng, B. Peng, and A. Yang, "Credit card fraud detection based on improved variational autoencoder Generative adversarial network," *IEEE Access*, vol. 11, pp. 83680-83691, 2023. https://doi.org/10.1109/ACCESS.2023.3302339

[55] J. Chung and K. Lee, "Credit card fraud detection: An improved strategy for high recall using KNN, LDA, and linear regression," *Sensors*, vol. 23, no. 18, p. 7788, 2023. https://doi.org/10.3390/s23187788

[56] R. Kumari, J. Singh, and A. Gosain, "SmS: SMOTE-stacked hybrid model for diagnosis of polycystic ovary syndrome using feature selection method," *Expert Systems with Applications*, vol. 225, p. 120102, 2023. https://doi.org/10.1016/j.eswa.2023.120102

[57] S. N. S. S. Daud, R. Sudirman, and T. W. Shing, "Safe-level SMOTE method for handling the class imbalanced problem in electroencephalography dataset of adult anxious state," *Biomedical Signal Processing and Control*, vol. 83, p. 104649, 2023. https://doi.org/10.1016/j.bspc.2023.104649

[58] P. Sun, Z. Wang, L. Jia, and Z. Xu, "SMOTE-kTLNN: A hybrid re-sampling method based on SMOTE and a two-layer nearest neighbor classifier," *Expert Systems with Applications*, vol. 238, p. 121848, 2024. https://doi.org/10.1016/j.eswa.2023.121848

[59] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSp*, vol. 1, no. 2018, pp. 108-116, 2018.