mathematics

MDPI

Article

Bi-Partitioned Feature-Weighted *K*-Means Clustering for Detecting Insurance Fraud Claim Patterns

Francis Kwaku Combert [1], Shengkun Xie [2,*] and Anna T. Lawniczak [1]

[1] Mathematics and Statistics, University of Guelph, Guelph, ON N1G 2W1, Canada; fcombert@uoguelph.ca (F.K.C.); alawnicz@uoguelph.ca (A.T.L.)
[2] Global Management Studies, Ted Rogers School of Management, Toronto Metropolitan University, Toronto, ON M5B 2K3, Canada
* Correspondence: shengkun.xie@torontomu.ca; Tel.: +1-416-979-5000 (ext. 543474)

Abstract: The weighted *K*-means clustering algorithm is widely recognized for its ability to assign varying importance to features in clustering tasks. This paper introduces an enhanced version of the algorithm, incorporating a bi-partitioning strategy to segregate feature sets, thus improving its adaptability to high-dimensional and heterogeneous datasets. The proposed bi-partition weighted *K*-means (BPW *K*-means) clustering approach is tailored to address challenges in identifying patterns within datasets with distinct feature subspaces, such as those in insurance claim fraud detection. Experimental evaluations on real-world insurance datasets highlight significant improvements in both clustering accuracy and interpretability compared to the classical *K*-means, achieving an accuracy of approximately 91%, representing an improvement of about 38% over the classical *K*-means algorithm. Moreover, the method's ability to uncover meaningful fraud-related clusters underscores its potential as a robust tool for fraud detection. Beyond insurance, the proposed framework applies to diverse domains where data heterogeneity demands refined clustering solutions. The application of the BPW K-means method to multiple real-world datasets highlights its clear superiority over the classical K-means algorithm.

Keywords: *K*-means clustering; machine Learning; feature selection; insurance fraud detection

MSC: 62H20; 62J12; 62P05

Academic Editor: Manuel Alberto M. Ferreira

Received: 30 November 2024
Revised: 12 January 2025
Accepted: 21 January 2025
Published: 28 January 2025

Citation: Combert, F.K.; Xie, S.; Lawniczak, A.T. Bi-Partitioned Feature-Weighted *K*-Means Clustering for Detecting Insurance Fraud Claim Patterns. *Mathematics* **2025**, *13*, 434. https://doi.org/10.3390/math13030434

1. Introduction

The Royal Canadian Mounted Police (RCMP) has emphasized that tackling scams requires a collaborative effort involving consumers, institutions, researchers, and stakeholders. The rapid expansion of digital landscapes has intensified the need to combat fraudulent activities. Alarming statistics from the Canadian Anti-Fraud Center reveal a significant issue: reported fraud cases in 2022 soared to USD 530 million, marking a 40% increase from 2021 [1]. This highlights the urgent need for innovative solutions and proactive measures in detecting fraudulent claims. Specifically, the surge in fraudulent activities in auto insurance claims poses a significant threat to the stability and fairness of insurance systems. Moreover, when fraudulent claims go undetected, insurers bear the financial burden, leading to an increase in insurance premiums for all policyholders. This ripple effect fosters distrust and financial strain. All of the above factors have motivated our work to enhance existing clustering methods for detecting insurance fraud claim patterns.

In recent years, advanced data mining techniques—such as regression, association, sequential patterns, classification, clustering, and others—have emerged as significant contributors to insurance analysis [2,3]. These techniques have enabled the insurance industry to better predict fraudulent insurance claims and premiums tailored to individual clients. Additionally, they play a vital role in the initial screening process for evaluating claims, thereby reducing reliance on manual evaluations and mitigating financial losses over time [4]. To minimize fraudulent activities further, it is essential to enhance fraud detection capabilities by improving existing algorithms like *K*-means clustering [5–7].

It is crucial to distinguish between identifying fraud claim clusters and classifying claims as fraudulent or non-fraudulent. Fraud claim clustering focuses on grouping claims based on shared characteristics without prior knowledge of their fraudulent nature. This method uncovers patterns in the data, flagging clusters that deviate from the norm for further investigation. It identifies potential anomalies but does not explicitly label claims as fraudulent or non-fraudulent. In contrast, classifying claims as fraudulent or non-fraudulent involves training a model on labeled data to categorize individual claims as either fraudulent or non-fraudulent.

This paper contributes to the field of fraud detection by enhancing the identification of fraudulent claim clusters through the development of a novel unsupervised algorithm. The proposed bi-partition weighted *K*-means (BPW *K*-means) algorithm introduces a significant advancement in clustering methodologies. This algorithm contributes to addressing critical challenges and improves the capabilities of traditional *K*-means clustering algorithms. The BPW *K*-means algorithm stands out due to its flexibility, achieved through the introduction of weight parameters and feature bi-partitions, as shown in Figure 1. These weight parameters enable the algorithm to adapt to specific feature characteristics within a dataset, enhancing its robustness across various datasets and making it suitable for diverse applications. Additionally, the method incorporates feature ranking using existing algorithms, such as random forest [8], adding a new dimension to its functionality. By leveraging the efficiency of feature ranking methods, the BPW *K*-means algorithm improves both the interpretability and quality of its clustering compared to traditional *K*-means approaches. In summary, the novel BPW *K*-means algorithm advances traditional clustering techniques, paving the way for further investigation and refinement of unsupervised learning methods.

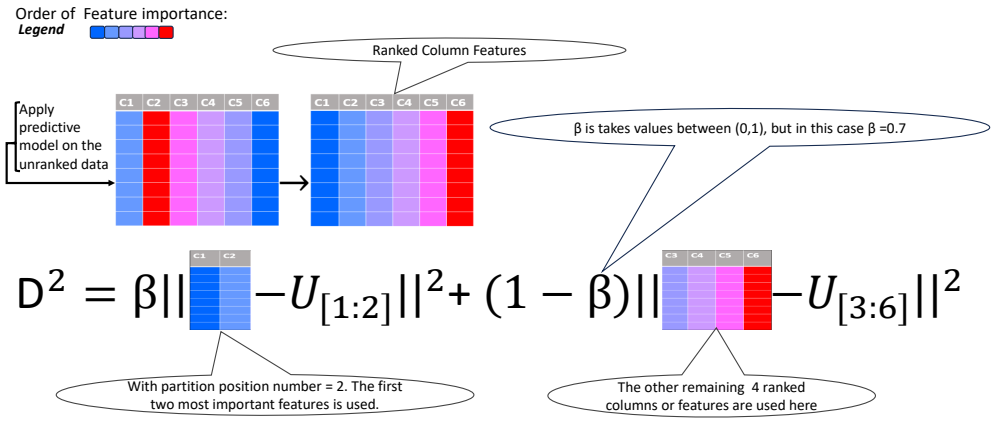


Figure 1. Illustration of the basic idea of weighted *K*-means under bi-partition. Example with parameters: colBiPartitionNum (i.e., bi-partition number) = 2, and $\beta = 0.7$.

The remaining sections of this paper are organized as follows. In Section 2, we review the existing methods and research studies related to fraud detection and improving clustering performance. Section 3 describes the classical *K*-means algorithm and introduces the bi-partition weighted *K*-means (BPW *K*-means) algorithm. Additionally, it discusses

clustering performance metrics, the Rand index (RI), the adjusted Rand index (ARI), and algorithms for measuring and validating clustering quality metrics. An exploratory analysis of insurance claims data from the United States, focusing on fraudulent and non-fraudulent cases obtained from Kaggle [9], is presented in Section 4. Additionally, the significance of key features in the Vehicle Fraud Insurance dataset is analyzed by using two feature selection methods: random forest and Relief. In Section 5, we discuss the applications of the BPW *K*-means algorithm to insurance fraud claim detection and compare the clustering performance of the BPW *K*-means algorithm with the classical *K*-means across different clustering groupings. Section 6 discusses the practical implications of the research results presented in Sections 4 and 5. In Section 7, the performance of the proposed BPW K-means method is evaluated using three publicly available datasets from the Machine Learning Repository at the University of California, Irvine (UCI) [10]: the Iris dataset, the Sirtuin6 dataset, and the Wholesale Customers dataset. Lastly, in Section 8, we conclude our study and provide additional remarks regarding potential directions for future research.

2. Related Works

As previously stated, fraud detection has emerged as a critical area of study. Researchers have employed various methods and techniques rooted in data analytics, machine learning, and artificial intelligence to identify and mitigate fraudulent activities. The study by Nian et al. in Ref. [11] introduced a spectral ranking-based method for detecting fraudulent activities in auto insurance claims. This approach combines spectral optimization with the analysis of a Laplacian matrix to rank suspicious cases, thereby identifying fraud without labeled data. Their method effectively captures the strength of the interdependencies among feature variables, making it suitable for both global and local anomaly detection. However, the authors tested their method only on artificial datasets generated by software, and its performance on real-world datasets remains uncertain. Similarly, Yang et al. in Ref. [12] proposed a multimodal learning framework that combines structured data, text, and images to detect fraud. They utilized advanced deep learning models like BERT (bidirectional encoder representations from transformers) and ResNet (residual neural network), which improved accuracy by capturing subtle fraud indicators. Other studies in Ref. [13–15] introduced hybrid models based on convolutional neural networks (CNNs) and long short-term memory (LSTM) networks. These models enhance feature extraction by identifying key patterns in data for fraud detection. Despite their improved performance, CNNs and LSTMs, like other deep learning models, are often referred to as "black-box" systems, offering limited transparency about how individual features contribute to the final classification. This poses challenges in fraud detection, as auditors and regulatory bodies require clear justifications for classifying claims as fraudulent. In contrast, Ref. [16] combined blockchain technology, gradient-boosting decision trees, and graph neural networks (GNNs) for fraud detection in financial transactions. Their method outperformed CNN-based approaches but the authors acknowledged potential biases and errors due to data limitations or overreliance on specific types of fraudulent behavior.

The authors of Ref. [17] reviewed the regulatory efforts and fraud scandals, emphasizing the importance of policy integration; however, this study lacks technical details on machine learning applications. Meanwhile, [18] conducted an extensive review of over 50 studies on auto fraud detection and prevention, highlighting the scarcity of publicly available datasets, as noted in Ref. [15]. A regional case study on Ontario's auto insurance anti-fraud initiative, Ref. [19], demonstrated the effectiveness of government and regulatory collaborations in curbing fraud.

The authors of Ref. [20] investigated the integration of explainable AI (XAI) techniques in banking fraud detection, emphasizing the need for transparency and interpretability

to balance performance with regulatory compliance. Similarly, ref. [21] compared several machine learning algorithms—logistic regression, extreme gradient boosting (XGB), decision tree, KNN, and random forest—for detecting fraudulent claims. Logistic regression achieved an F1 score of 83; however, when tested with new datasets, random forest delivered the best performance. These models were implemented by using Python's PySpark MLlib module. The authors noted challenges related to data quality in some datasets, which negatively impacted prediction performance, and suggested refining models for different fraud cases to adapt to evolving fraud patterns. A follow-up study, ref. [22], extended the work presented in Ref. [21] by incorporating linear discriminant analysis (LDA), which achieved a superior F1 score of 87, concluding that LDA outperforms other methods. Additionally, ref. [23] explored the use of clustering algorithms for automating fraud detection in group life insurance claims during audits. Their research focused on anomaly identification through cluster analysis, flagging clusters characterized by large beneficiary payments or long delays between claim submission and payment for further investigation. They employed the *K*-means clustering algorithm using open-source software like WEKA (i.e., Waikato Environment for Knowledge Analysis, Ref. [23]). In their paper, the authors formed eight clusters based on two attributes when clustering their dataset; however, they did not explain the rationale behind this specific choice. Furthermore, their acknowledgment that cluster analysis serves as a preliminary step toward integrating technology into auditing, suggests a need for future research to develop new methods and improve upon existing ones, aligning with the objectives of this study.

Other unsupervised methods such as principal component analysis (PCA), introduced in Refs. [24,25], are foundational techniques for dimensionality reduction. PCA transforms data into uncorrelated components by identifying the directions (principal components) along which the data vary the most. This process decomposes the data into orthogonal components and it preserves the most important features in the data. However, its linear nature limits its ability to capture complex relationships. Independent component analysis (ICA), introduced by Comon in Ref. [26], aims to find statistically independent components and excels in tasks like blind source separation, which is a technique in signal processing that is used to recover original source signals from a set of observed mixed signals, without prior knowledge about the sources. However, ICA's performance can be compromised when the assumptions of independent and non-Gaussian sources are violated, Ref. [27]. The Laplacian score (LS), proposed by He et al. in Ref. [28], evaluates feature importance by assessing locality-preserving power. It measures the variance of each feature within the local neighborhood of the data, selecting features that best conform to the intrinsic geometry of the data distribution. LS has been effective in applications where preserving local structures is important, such as clustering and manifold learning. The mutual information score (MIS) rooted in information theory, evaluates variable dependencies, making it versatile for feature selection, but can be computationally demanding and data-intensive, Ref. [29]. In contrast, random forest, introduced by Breiman (2001), not only excels in classification and regression but also provides feature importance scores based on decision tree impurity reduction, enhancing interpretability, Ref. [8,30]. The ReliefF algorithm, an extension of the original Relief algorithm from Ref. [31], and later improved on in Ref. [32], excels at identifying key features by comparing data samples or observations with their nearest neighbors. One of the strengths of ReliefF is its ability to handle noisy and incomplete data, as well as its capacity to capture feature interactions. However, its performance depends heavily on parameter choices, particularly the number of nearest neighbors, and struggles with high-dimensional datasets [33].

3. Methods

In this section, the classical *K*-means method and its mathematical formulation will first be briefly presented. Next, the proposed bi-partition weighted *K*-means algorithm (BPW *K*-means) will be described. Additionally, some label-based clustering performance measures will be introduced.

3.1. Classical K-Means

The classical *K*-means algorithm [34–37] iteratively updates cluster centroids and reassigns data points to the nearest centroid until convergence is achieved. Convergence is typically determined by checking whether the cluster assignments and centroids no longer change significantly between iterations or when the maximum number of iterations is reached. A key characteristic of the classical *K*-means algorithm is its assumption that all features of a dataset contribute equally. The method uses the Euclidean distance $\|\cdot\|$ as its metric.

Consider an input dataset $X = \{\mathbf{x}_i\}_{i=1}^{m}$ where $\mathbf{x}_i \in \mathbb{R}^n$, m represents the number of rows (i.e., the number of observations), and n represents the number of columns (i.e., the number of features). Let $k \in \mathbb{Z}^+ \setminus \{1\}$ be the number of clusters, and let $C = \{C_1, C_2 \ldots, C_k\}$ be the disjoint grouping of data X into k subsets, such that each C_j has a corresponding centroid $\boldsymbol{\mu}_j$, where $j = 1, 2, \ldots, k$. Following the formulation in Ref. [34], and using the *K*-means clustering method to cluster points (observations) in a dataset involves minimizing the following objective function:

$$\underset{C}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} \left\| \mathbf{x} - \boldsymbol{\mu}_j \right\|^2, \tag{1}$$

in which the within-cluster sum of squares is minimized for a given number of clusters (k).

The *K*-means method is straightforward in its approach to iteratively minimize the objective function in Equation (1). It is fast and computationally efficient, especially for large datasets but it is sensitive to outliers. Since the method attempts to minimize the sum of squared distances, a single large outlier can significantly distort the centroid of a cluster, leading to incorrect assignment of data points to clusters. Additionally, the *K*-means algorithm is sensitive to the initial choice of centroids, and poor initialization can result in suboptimal clustering quality.

3.2. Distinguishing BPW K-Means from Fuzzy Clustering: Principles, Performance, and Applications

We aim to clearly differentiate the proposed algorithm from the fuzzy clustering method. The BPW *K*-means algorithm and fuzzy clustering algorithm operate on distinct principles and yield different results. In the BPW *K*-means algorithm, each data point is assigned to one specific cluster based on its distance to the centroids, meaning the clustering is a hard-clustering type, and each point belongs to only one and only one cluster. In contrast, fuzzy C-means (FCM) allows each data point to belong to multiple clusters with varying degrees of membership, meaning a point can be partially associated with several clusters, reflecting uncertainty in the data [38]. Additionally, the centroids generated by the proposed method are recalculated as the mean of all data points assigned to each cluster, leading to clear and distinct cluster boundaries. In fuzzy clustering, centroids are calculated based on membership values, which consider the degree to which each point belongs to multiple clusters. As a result, the BPW *K*-means produces outputs that are easier to interpret with each data point belonging to a specific cluster, providing a clear categorization of the data. In contrast, fuzzy clustering may be harder to interpret due to overlapping memberships.

Theoretically, since the BPW *K*-means is a variant of *K*-means, it has lower computational complexity and can be faster for large datasets, as it only needs to compute the closest centroid for each point. In contrast, the fuzzy clustering algorithm generally has higher computational complexity due to the need to calculate membership values for each point across all its clusters, making it slower, particularly for large datasets.

In this paper, "Feature Bi-Partition Ranked" refers to ordering the features of a dataset according to their importance using a predictive model, while "Feature Bi-Partition Unranked" leaves the features in their original order, without altering the positions of the data columns. In this study, we observed that the BPW *K*-means performed slightly better with ranked datasets compared to unranked datasets.

Also, under the BPW *K*-means method, the distance metric used is the Euclidean distance $\|\cdot\|$ metric. Users must provide the following inputs: the dataset, the desired number of clusters (k), a bi-partition number to indicate the position (point) to split the dataset columns, and a weight value β between 0 and 1 to be applied to one of the resulting bi-partitions. Note in this study, we tested β values of $0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8$ and 0.9, thus, starting from a β value of 0.1 and incrementing it by 0.1, up to a β value of 0.9. Also, we tested column bi-partition numbers starting from 1 and going up to the column size of the dataset. The β value and bi-partition number pair that yielded the highest accuracy was selected as the optimal parameter set.

3.3. Bi-Partition Weighted K-Means

In this section, we introduce the primary contributions of this paper. We begin with an example, illustrated in Figure 1, to demonstrate how our method works. Next, we delve into a more formal mathematical formulation of our method and present the pseudocode for the algorithm.

The example in Figure 1 uses a dataset with eight rows and six columns. A bi-partition number of 2 is chosen because the user suspects that the first two columns are the most important features, while the remaining four features are less important. A weight value β is set to 0.7. The legend in the figure uses a color gradient, with blue representing the most important features and red representing the least important, transitioning through four other colors in descending order of importance. A predictive model is applied to the dataset to rank its features, as shown by the color-coded illustration. The first two most important (blue) features are selected along with their corresponding centroids ($U_{[1:2]}$). Note that for the dimensional agreement, the first two most important features and their corresponding centroids still retain the same dimensions as the given dataset. The entries of the remaining less important features (in this case, four features) and their corresponding centroids are set to zero . The Euclidean distances between these selected features and their corresponding centroids are computed, and the weight value $\beta = 0.7$ is applied to the calculated distances. Similarly, the remaining four less important features and their corresponding centroids ($U_{[3:6]}$) are selected. These four features and their corresponding centroids also retain the same dimensions as the given dataset, while the entries of the first two important features and their centroid parts are set to zero. The Euclidean distances for this second portion are then calculated, and a weight of $1 - \beta$ is applied. The final distance, denoted by D, is the sum of these two weighted distances, and the clustering of the data observations is performed using this weighted distance metric. After presenting the main points of the introduced algorithm in this example, we proceed to its formal description.

Let us assume datasets $A, A^{d_1} \in \mathbb{R}^{m \times n}$, where m is the number of rows (observations) and n is the number of columns (features). Let $p \in \mathbb{Z}^+$, such that $1 \leq p \leq n$, be the data column bi-partition number. This number refers to the column-wise position where a split (partition) occurs within the dataset features. Let $U, U^{d_1} \in \mathbb{R}^{k \times n}$ denote some k

centroids. The index d_1 indicates that the matrices have been partitioned at column p. Then, a bi-partition of A and U at their respective column p is defined as follows:

$$A_{ij}^{d_1} = \begin{cases} A_{ij} & \text{if} \quad 1 \leq j \leq p \\ 0 & \text{if} \quad p < j \leq n, \end{cases} \quad \forall \quad 1 \leq i \leq m, \tag{2}$$

$$A_{ij} - A_{ij}^{d_1} = \begin{cases} 0 & \text{if} \quad 1 \leq j \leq p \\ A_{ij} & \text{if} \quad p < j \leq n, \end{cases} \quad \forall \quad 1 \leq i \leq m, \tag{3}$$

and,

$$U_{sj}^{d_1} = \begin{cases} U_{sj} & \text{if} \quad 1 \leq j \leq p \\ 0 & \text{if} \quad p < j \leq n, \end{cases} \quad \forall \quad 1 \leq i \leq m, \tag{4}$$

$$U_{sj} - U_{sj}^{d_1} = \begin{cases} 0 & \text{if} \quad 1 \leq j \leq p \\ U_{sj} & \text{if} \quad p < j \leq n, \end{cases} \quad \forall \quad 1 \leq i \leq m, \tag{5}$$

The key difference between the BPW K-means and the classical K-means is that the Euclidean distances between the features (column dimensions) of an observation and its corresponding centroid are bi-partitioned, with different weights applied to each partition. In the proposed method, the features are divided into two parts, resulting in a weighted distance calculation. Our method seeks to group the m observations into distinct k clusters (set C), where $(k \leq m)$ and $C = \{C_1, C_2, \ldots, C_k\}$, with the most minimal overall weighted sum of squared distance. The objective function of the proposed method is given as follows:

$$D^2 = \sum_{i=1}^{m} \left(\min_{s \in \{1,2,\ldots,k\}} \sum_{j=1}^{n} \left[\beta \left\| A_{ij}^{d_1} - U_{sj}^{d_1} \right\|^2 + (1-\beta) \left\| (A_{ij} - A_{ij}^{d_1}) - (U_{sj} - U_{sj}^{d_1}) \right\|^2 \right] \right), \tag{6}$$

subjected to $0 \leq \beta \leq 1$, where k is the number of required clusters; i is the index of observations in a dataset; s is the index of centroids; j is the index of column features in a dataset; min stands for minimum; n is the number of features; m is the number of observations; $||\cdot||$ is the Euclidean distance; β is the weight between 0 and 1, to be applied to the bi-partition distances; and D is the final weighted distance.

The proposed method is designed to optimize clustering results by applying a predictive model to rank the features of a given dataset based on their importance. This ensembling step is known to improve clustering accuracy, as the contributions of each feature determine the strength of similarity within clusters, as stated in Refs. [39,40]. The steps of the proposed algorithm are detailed in Algorithm 1. The algorithm begins with empty centroids and distance vectors. After the initialization step, it computes the distances to the nearest centroids (Step 5). Within this step, when given a specific β value and feature bi-partition position number, the algorithm splits the dataset according to the bi-partition number. As described in the example above, the weighted distance is then calculated, such that β is applied to the first part of the bi-partition and simultaneously $1 - \beta$ is applied to the second part of the bi-partition. The overall within-cluster weighted sum of squares is then assigned to the left-hand side. Each observation is then assigned to its nearest centroid, thereby creating k clusters in Step 6. In Step 7, the BPW K-means updates the centroids by calculating the mean of each cluster. This process is repeated until the desired number of iterations is achieved, finally returning the updated centroids and clusters.

Algorithm 1. The bi-partition weighted *K*-means algorithm for clustering

Input: Data (scaled) A $\in \mathbb{R}^{m \times n}$, number of clusters k $\in \mathbb{Z}^+$, $k \leq m$; data column bi-partition number $p \in \mathbb{Z}^+$, $p \leq n$; a weight parameter $\beta \in (0, 1)$; $N_{iter} \in \mathbb{N}$ is the maximum number of iterations; convergence threshold $\epsilon > 0$.
Output: Clusters and Centroids.

Data Feature Importance Ranking:
Using any preferred feature selection algorithm or predictive model:

i: **Apply** model = Train a predictive model on (Data A)
ii: Feature importance = Use the model to rank the features based on their importance
iii: Select the top-ranked (Data *A*) features based on feature importance above
iv: **Set** New A = ranked and ordered features of A

then continue:

1: Initialize U **= Centroids** , and D **= distances** as empty vectors, $t = 0$.
2: **While** not converged and $t < N_{iter}$ **do**
3: **for** $i = 1 : m$ **do**
4: Compute **distances** from the nearest centroid
5: $D_i = \sqrt{\beta \sum_{q=1}^{p} \sum_{j=1}^{k} ||A_{iq} - U_{jq}||^2 + (1-\beta) \sum_{q=p+1}^{n} \sum_{j=1}^{k} ||A_{iq} - U_{jq}||^2}$
6: Assign each data point A_i to the closest centroid (Min D_i)
7: **Set** $U_t = U$, U_t stores old centroid.
8: **Update** $U_j = \frac{\sum_{j=1}^{k} A_j}{|A_j|}$, the mean of each cluster j.
9: Check for convergence
10: If $||U_j - U_t||^2 < \epsilon \; \forall j$ then
11: converged = true
12: $t = t + 1$, increase iteration counter
13: **Repeat** the process from line 3 to line 10 till N_{iter} iterations or convergence
14: **Return** { (Clusters j, U_j) | $j = 1, \ldots, k$ }, the final cluster assignments and their corresponding centroids.

The algorithm excels in cases where feature importance plays a crucial role in determining cluster assignments and it provides valuable clustering solutions for datasets with complex structures. The algorithm's ability to handle different bi-partition configurations, positions it as a versatile tool in machine learning.

Additionally, the BPW *K*-means algorithm introduces a new way of clustering by combining feature ranking with flexible parameter tuning. This approach helps the BPW *K*-means algorithm to prevent features contributing less to the clustering quality from impacting the overall clustering performance. This adaptability is crucial in real-world applications, such as the auto insurance industry, where imbalanced datasets are common. By incorporating the BPW *K*-means algorithm into our clustering methodologies, we aim to improve the robustness, reliability, and clustering quality in unsupervised learning.

Having introduced the core methodology of the proposed algorithm, it is essential to assess its effectiveness in comparison to the existing *K*-means method. To achieve this, we will now explore various clustering performance metrics, which will help us quantify and evaluate the algorithm's performance.

3.4. Clustering Performance Metrics

In this section, we discuss two widely used clustering performance metrics, which are the Rand index and the adjusted Rand index. These metrics will be employed to compare the performance of the proposed algorithm with the classical *K*-means. The results of this comparison will be presented in detail in tabular form later in this study.

3.4.1. Rand Index (RI)

From Refs. [41,42], it is known that the Rand index (RI) is a metric used to measure the similarity or agreement between two data clusterings or between a clustering result and a ground truth (i.e., the known labels of data). It calculates the percentage of data point pairs that are either correctly classified in the same cluster in both clusterings or correctly classified in different clusters in both clusterings. However, RI does not account for the possibility of agreement occurring by chance, which may limit its applicability in some scenarios. Nevertheless, it remains a valuable metric for assessing the quality of clustering algorithms.

Assume that A and B represent two different clustering results from two different algorithms, with both having n total points. In this study, A denotes the clustering result obtained from the BPW K-means algorithm, while B represents the true labels from the dataset. Let n_{ij} denote the number of points that belong to both cluster i in clustering A and cluster j in clustering B. Let $n_{.i}$ and $n_{.j}$ represent the number of points that belong only to cluster i in clustering A and cluster j in clustering B, respectively. Assume that c_1 is the number of clusters in clustering A and c_2 is the number of clusters in clustering B. Following the formulation of Ref. [41], the similarity between A and B can be calculated as follows:

$$RI = \frac{\binom{n}{2} + 2\sum_{i=1}^{c_1}\sum_{j=1}^{c_2}\binom{n_{ij}}{2} - \left[\sum_{i=1}^{c_1}\binom{n_{i\cdot}}{2} + \sum_{j=1}^{c_2}\binom{n_{\cdot j}}{2}\right]}{\binom{n}{2}}. \tag{7}$$

3.4.2. Adjusted Rand Index (ARI)

As noted in Refs. [41,42], the adjusted Rand index (ARI) is a refinement of the Rand index, which accounts for chance agreements between two clusterings. Unlike the Rand index, ARI adjusts for the possibility of random agreement, providing a more accurate measure of clustering similarity. The higher the ARI value, the closer the two clusterings are to each other. ARI ranges from -1 to 1, where the value of 1 indicates perfect agreement between the two clusterings, while 0 suggests a random agreement, and -1 implies complete disagreement between the clusterings.

It has been observed that when the Rand index and its expected value are both low and closely aligned, the ARI value tends to approach zero, indicating that the clustering result is similar to a random partition. The ARI modifies the Rand index as follows:

$$ARI = \frac{RI - Expected(RI)}{Max(RI) - Expected(RI)}. \tag{8}$$

The Rand index and adjusted Rand index are both metrics used to evaluate the similarity between two data clusterings or partitionings. These metrics are commonly employed in clustering and unsupervised machine learning to compare the quality of different algorithms or to assess the performance of a single algorithm across multiple runs. In this study, we will utilize both RI and ARI to evaluate the clustering performance of the proposed method. Additionally, we will compare these results with those obtained from K-means clustering. To ensure a robust evaluation, both algorithms will be executed 200 times, generating samples of metrics. For each iteration, the centroid updates for clustering will be limited to a maximum of 100 times (within algorithm maximum runs).

3.4.3. Algorithms for Measuring and Validating Clustering Quality Metrics

Algorithm 2 is designed to evaluate the performance of a clustering method by measuring how accurately the predicted clusters align with the true class labels of the given data points. Let us assume that A is a dataset with true class labels, b, and k denotes the number of clusters.

In Algorithm 2, the steps begin by initializing vectors **x**, **y**, and the integer *Result* as placeholders for storing predicted cluster labels, the proportion (accuracy) of correct label assignments within each cluster, and the final accuracy value, respectively. In steps 3–5 of Algorithm 2, the method applies a clustering method (e.g., BPW *K*-means) to partition dataset A into k clusters. The resulting predicted cluster labels are stored in the vector **x**. At steps 7–9 of Algorithm 2; the method counts how many of these predicted labels in each cluster match the true class labels in vector b. Within each cluster, the method identifies the most dominant class (the true label with the highest count of matches with the predicted labels) to N_j. Additionally, it computes N_{tot_j}, which represents the total number of data points within cluster j. The algorithm then computes the proportion $y_j = \frac{N_j}{N_{\text{tot}_j}}$ of correctly assigned labels in cluster $j \in \{1, \ldots, k\}$ by dividing N_j by N_{tot_j} at step 11. After calculating the proportion y_j for each cluster $j \in \{1, \ldots, k\}$, the algorithm computes the mean of the proportions (accuracies) by adding all values in y_j and dividing the sum of the accuracies (proportions) by the total number of clusters k. This average accuracy is then stored in the *Result* at step 12 in Algorithm 2. The metric (average accuracy) evaluates the clustering quality by measuring how accurately the clustering method has grouped the data relative to the true class labels. A higher metric value, close to 1, indicates better clustering, while a value near 0 suggests poorer clustering performance.

Algorithm 2. Accuracy function used for measuring the accuracy of clusters.

Input: Data (scaled) A $\in \mathbb{R}^{m \times n}$, label vector $b \in \mathbb{R}^m$ (true class labels) and a parameter $k \in \mathbb{N}$ is the number of clusters.
Output: Result

1: Initialize x and y as empty vectors, and Result as an integer.
2: **for any given** k, **do**
3: **Perform** clustering on A using any clustering method (e.g., BPW *K*-means)
4: Fit = Clustering Method (A, centers = k)
5: x = fit clusters (assign predicted cluster labels)
6: **for** $j = 1, \ldots k$ **do**
7: **Count** all label occurrences in predicted cluster j by comparing it to labels in b
8: N_j = Count of the most frequent true label in predicted cluster j
9: N_{tot} = total of all label counts in predicted cluster j
10: **then compute**
11: **append** $\mathrm{y}_j = \frac{N_j}{N_{tot_j}}$
12: **Return** $Results = \frac{\sum_{j=1}^{k} \mathrm{y}_j}{k}$

Algorithm 3 complements methods like Algorithm 2 by validating the robustness, consistency, and reliability of clustering performance metrics. Let $N_{iter} \in \mathbb{N}$ be the number of iterations. Step 1 of Algorithm 3 initializes an empty vector **z** to store metric values obtained from iterating a clustering performance metric method. Given a performance metric of a clustering method (such as the output from Algorithm 2, Rand index, or adjusted Rand index) and a specified number of iterations N_{iter}, Algorithm 3 operates as follows.

In steps 2–4 of Algorithm 3, each iteration i from 1 to N_{iter} computes the specified performance metric and assigns the resulting value to z_i. This process repeats until all iterations are completed. After all iterations, step 5 of Algorithm 3 computes and returns three key statistics: the maximum, mean, and standard deviation of the stored metric values in **z**. These statistics provide insights into the clustering method's effectiveness: the highest performance achieved, the average performance, and the variability in performance.

Algorithm 3. Validation method for clustering performance metrics.

Input: Performance metric of a clustering method (for example, using output from Algorithm 2, the Rand index, or the adjusted Rand index) and $N_{iter} \in \mathbb{N}$ is the number of iterations.
Output: Maximum, mean, and standard deviations of a given metric.

1: Initialize z as an empty vector to store N_{iter} metric values.
2: **for** $i = 1, \ldots, N_{iter}$ **do**
3: **append** z_i = output of **Algorithm** 2 or performance metric of any clustering method.
4: **Repeat** the process from line 3 till N_{iter} iterations
5: **Return** {Maximum of z, Mean of z, Standard Deviation of z}

After discussing the clustering performance metrics, it is essential to evaluate how our method will perform in real-world datasets. This leads us to the next phase of our study, where the method is applied to diverse real-world datasets.

4. Exploratory Analysis of Insurance Fraud Claims: Insights and Feature Rankings

In this section, we conduct an exploratory analysis of insurance claims data from the United States, focusing on fraudulent and non-fraudulent cases obtained from Kaggle [9]. This analysis leverages various graphical representations to visualize and understand the distribution of claims data across the northeastern United States. Additionally, we assess the importance of key features in the Vehicle Fraud Insurance dataset using two feature selection methods: random forest and Relief.

This exploratory phase serves as a foundational step, providing insights into the dataset's structure and feature relevance, which inform the subsequent application of the proposed BPW K-means and classical K-means clustering methods discussed in later sections.

The analyzed Insurance Claims dataset from the northeastern United States [9] includes fraudulent and non-fraudulent vehicle insurance claims. The analysis focuses on the following four data features for fraudulent cases: locations (latitude and longitude coordinates), relative insurance claim frequencies (the count of reported fraud cases at a location), average total insurance claim amounts (the mean of total fraud claim amount at a location), average vehicle insurance claim amounts (the mean of only vehicle fraud claim amount at a location), average injury insurance claim amounts (the mean of only injury fraud claim amount at a location) and average property insurance claim amounts (the mean of only property damage fraud claim amount at a location). The same data feature characterization applies to non-fraudulent cases: locations (latitude and longitude coordinates), relative insurance claim frequencies (the count of reported fraud cases at a location), average total insurance claim amounts (the mean total fraud claim amount at a location), average vehicle insurance claim amounts (the mean fraud claim amount for vehicles at a location), average injury insurance claim amounts (the mean fraud claim amount for injuries at a location), and average property insurance claim amounts (the mean fraud claim amount for property damage at a location).

To identify regions with a higher prevalence of fraudulent claim cases, we examined the geographical longitude and latitude coordinates of the claims in the dataset. It was observed that individual customer addresses had been altered, probably, to protect privacy and security. Consequently, some addresses could not be geocoded using Google's tool (Geocoding API https://developers.google.com/maps/documentation/geocoding/overview (accessed on 1 June 2023)) to convert addresses into latitude and longitude coordinates.

The considered dataset contains 1000 observations spanning seven unique cities ("Columbus", "Riverwood", "Arlington", "Springfield", "Hillsdale", "Northbend", and

"Northbrook") and seven unique state shortcodes ("SC", "VA", "NY", "OH", "WV", "NC", and "PA"). Despite thousands of unique street names, approximately 98 latitude and longitude coordinates were identified as corresponding to these addresses. Four coordinates were discarded as outliers after visualizing the statistics of the analyzed insurance claim data features on maps in Figures 2–6.

The maps of Figures 2–6 display, respectively, the following statistics: (Figure 2) Relative Insurance Claim Frequency; (Figure 3) Average Total Insurance Claim Amounts; (Figure 4) Average Vehicle Insurance Claim Amounts; (Figure 5) Average Injury Insurance Claim Amounts; (Figure 6) Average Property Insurance Claim Amounts. On these maps, the averages calculated for non-fraudulent claims for the considered features are represented by blue circles, while fraudulent claims are represented by red circles. The size of each circle corresponds to the magnitude of the average claim values for each respective feature of the vehicle insurance claim data. This visualization intuitively represents the differences in average values between fraudulent and non-fraudulent vehicle insurance claims across various locations. Note that the map legends use the abbreviations "Fraud" for fraudulent cases and "No-Fraud" for non-fraudulent cases. For brevity, these abbreviations are sometimes used in the following analysis.

In Figure 2, we can see that areas such as Arlington, OH, and Northbrook, OH, have high frequencies of fraudulent reported cases, while Arlington, PA, Hillsdale, OH, and Northbend, PA, exhibit higher frequencies of non-fraudulent cases. In Figure 3, high values of average total insurance claim amounts for fraudulent cases are evident in areas such as Riverwood, SC, Columbus, OH, and Northbrook, NC. Conversely, in Northbend, OH, significantly higher average values are observed for non-fraudulent total insurance claim amounts. In Figure 4, we can see that the areas of Riverwood SC, Northbrook, NC, and Columbus, OH, have higher average values of fraudulent vehicle insurance claim amounts. In contrast, areas such as Northbend, OH, Hillsdale, OH, and Arlington, OH, show higher average amounts for non-fraudulent cases. The map in Figure 5 reveals that areas such as Columbus, OH, Northbrook, OH, and Columbus, PA, have higher average values of fraudulent injury insurance claim amounts. Meanwhile, in Columbus, PA, Hillsdale, OH, and Columbus, NC, higher average values of non-fraudulent injury insurance claim amounts are observed. Finally, Figure 6 indicates that, in general, the average values of property insurance claim amounts for fraudulent and non-fraudulent claims are similar across different locations. However, notable exceptions include (1) Northbrook, WV, Springfield, SC, and Riverwood, SC, where the average values are higher for fraudulent claims, and (2) Hillsdale, PA, where the average value is high for non-fraudulent claims.

Figure 7a,b presents the analysis of feature rankings in the Vehicle Fraud Insurance dataset using two feature selection methods: random forest and Relief. Both methods produced similar feature rankings, with the primary difference being the reversed positions of the 'average property claim' and 'average injury claim' features. Notably, both methods identified the relative frequency of fraud as the most important feature, while Longitude and Latitude were deemed the least significant. Given the minimal differences in rankings, random forest was selected as the feature selection method for this study.

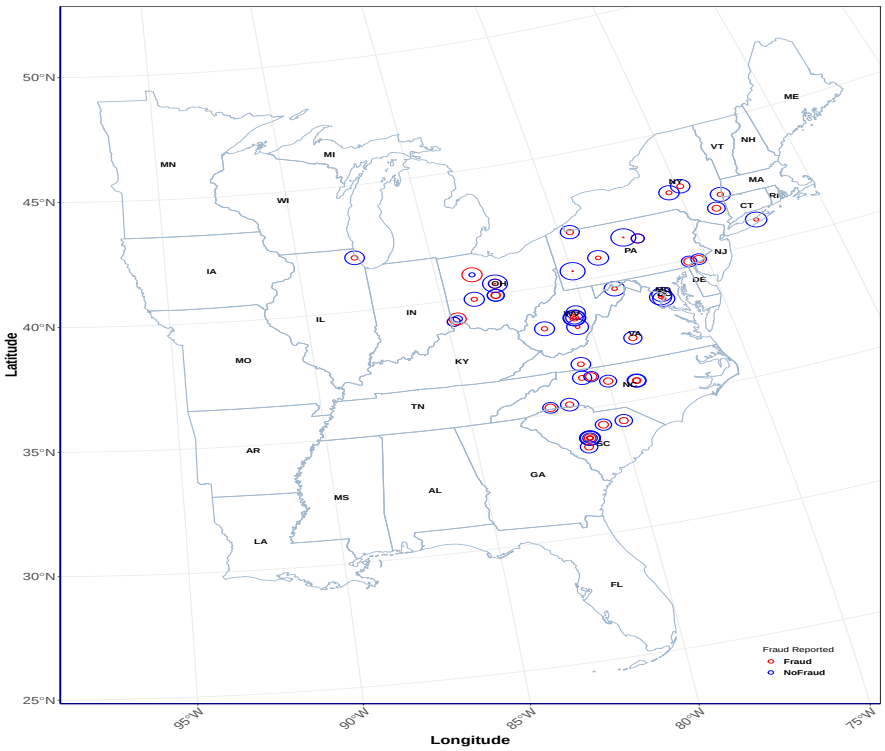


Figure 2. Relative insurance claim frequency of non-fraudulent (blue circles) and fraudulent (red circles) records in the northeast USA were calculated using the insurance dataset from Kaggle [9]. The size of each circle plot provides visual insight into the magnitudes of the respective claims.

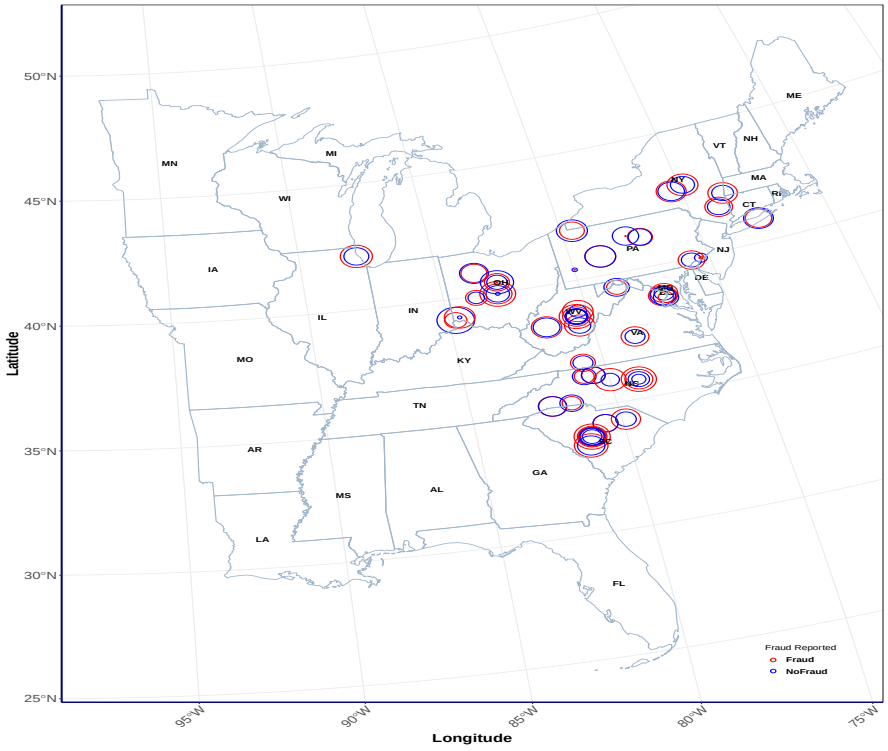


Figure 3. Average total insurance claim amounts of non-fraudulent (blue circles) and fraudulent (red circles) records in the northeast USA were calculated using the insurance dataset from Kaggle [9]. The size of each circle plot provides visual insight into the magnitudes of the respective claims.

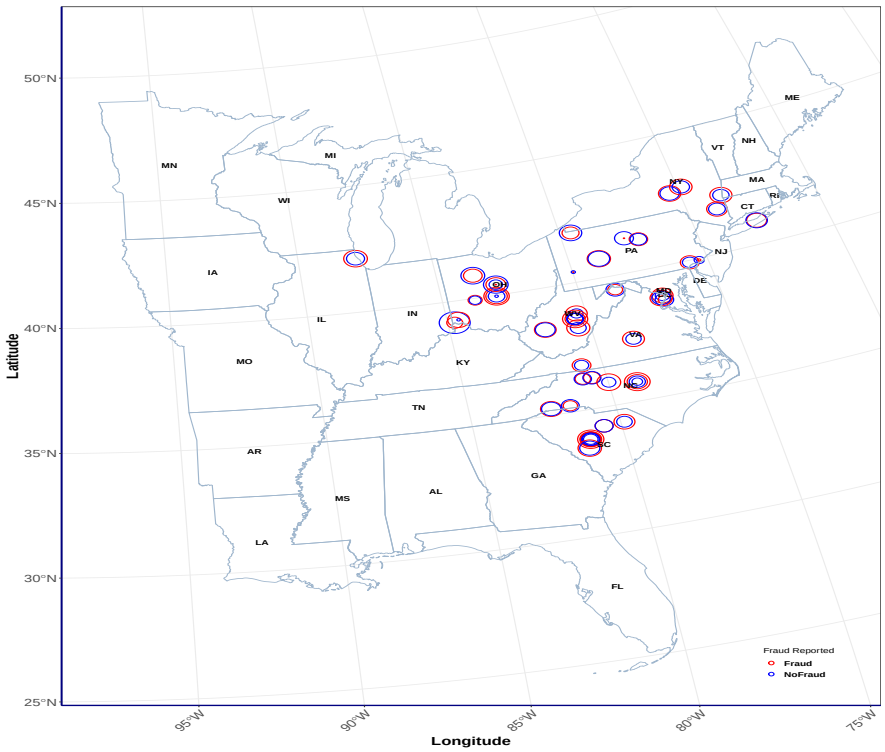


Figure 4. Average vehicle insurance claim amounts of non-fraudulent (blue circles) and fraudulent (red circles) records in the northeast USA were calculated using the insurance dataset from Kaggle [9]. The size of each circle plot provides visual insight into the magnitudes of the respective claims.

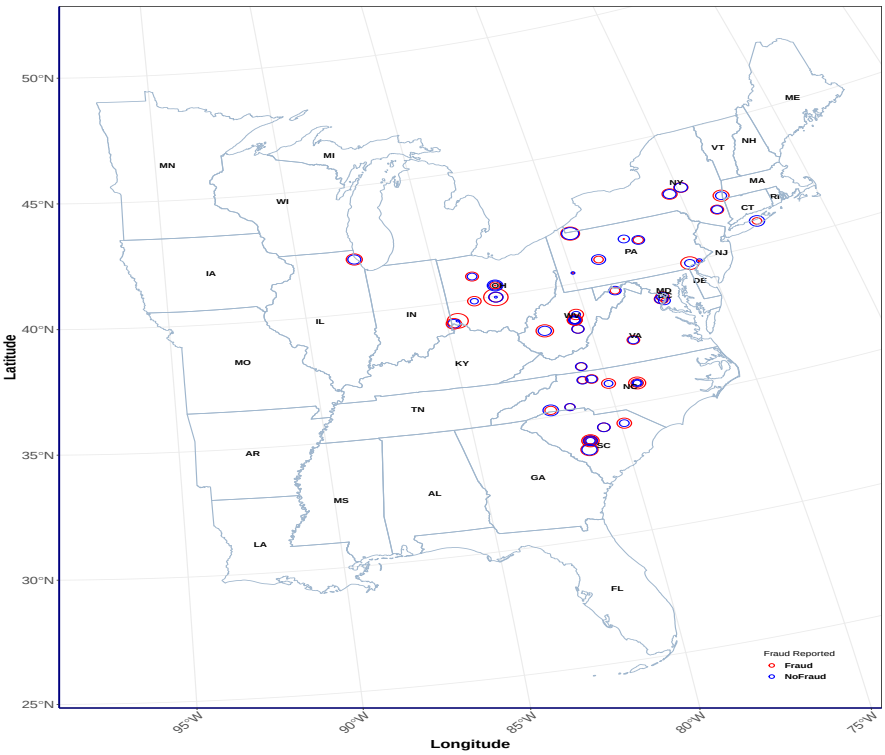


Figure 5. Average injury insurance claim amounts of non-fraudulent (blue circles) and fraudulent (red circles) records in the northeast USA were calculated using the insurance dataset from Kaggle [9]. The size of each circle plot provides visual insight into the magnitudes of the respective claims.

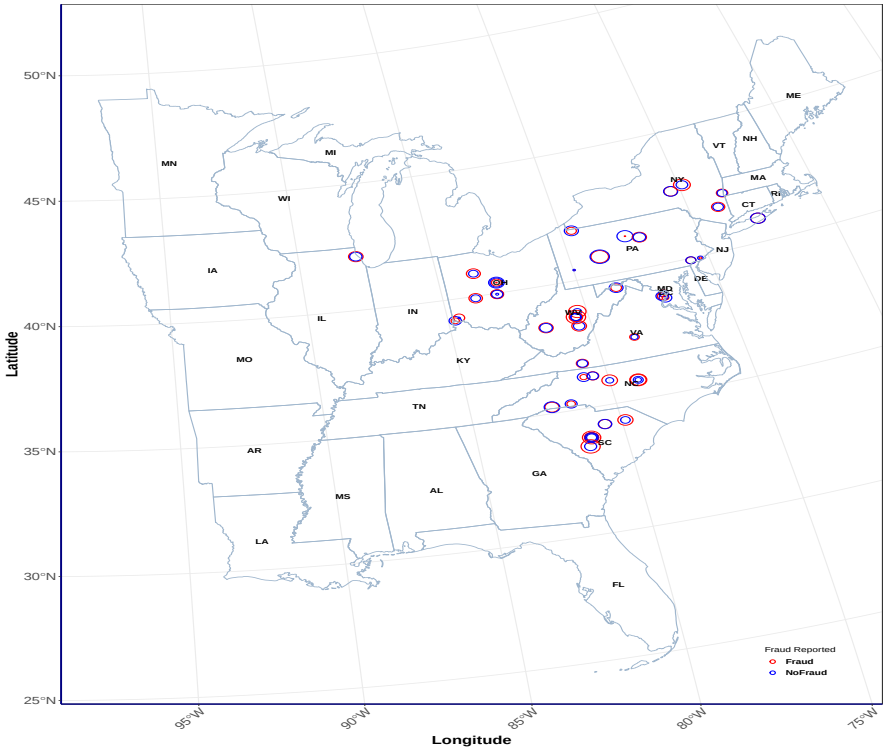
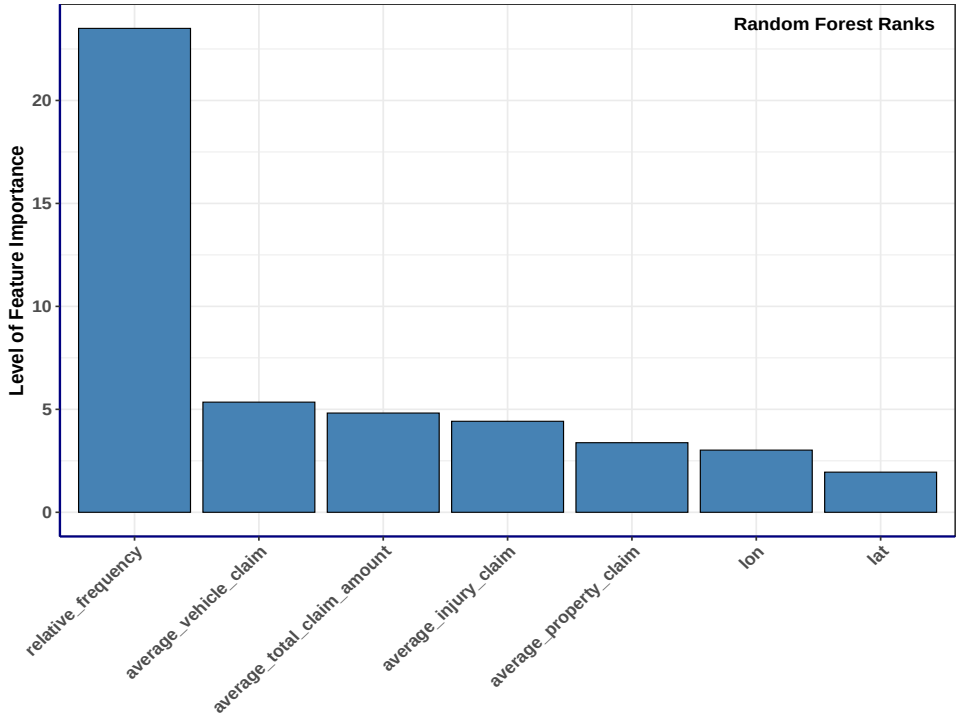
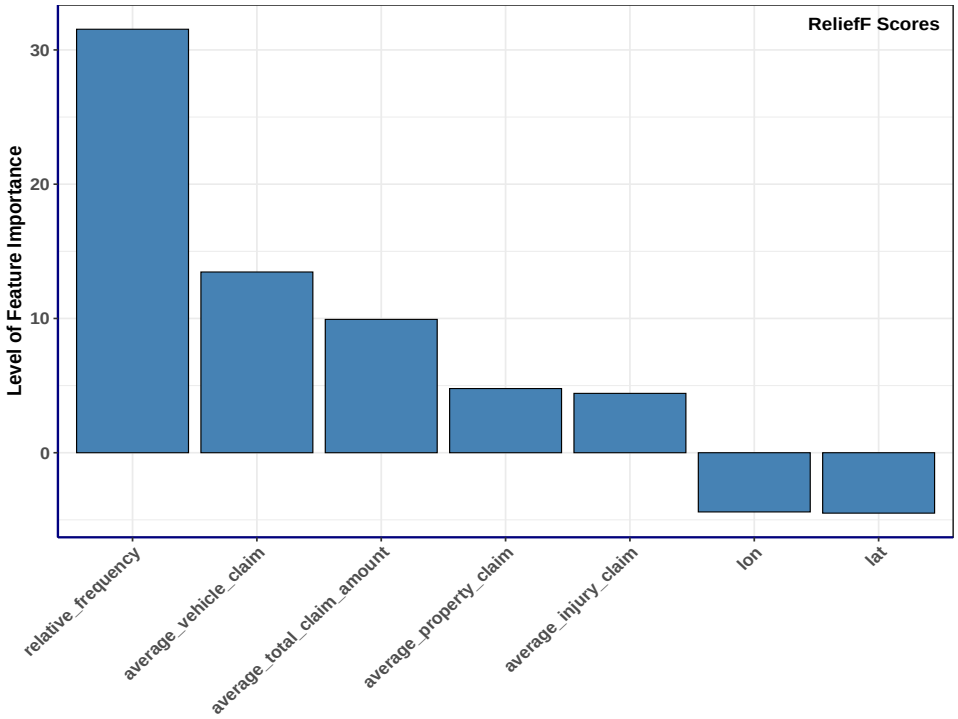


Figure 6. Average property insurance claim amounts of non-fraudulent (blue circles) and fraudulent (red circles) records in the northeast USA were calculated using the insurance dataset from Kaggle [9]. The size of each circle plot provides visual insight into the magnitudes of the respective claims.



(**a**) Random Forest

Figure 7. *Cont.*



(**b**) Relief Method

Figure 7. Feature rankings of the Vehicle Fraud Insurance dataset (Kaggle [9]) obtained using random forest (**a**) and Relief (**b**) feature selection methods. The height of each bar represents the relative importance and contribution strength of each feature.

5. Clustering Performance of BPW *K*-Means vs. Classical *K*-Means on the Insurance Fraud Claims Dataset: Ranked vs. Unranked Features

In this section, we evaluate the performance of the proposed BPW *K*-means method by applying it to the Insurance Claims dataset from the northeastern United States [9]. We begin with an empirical search for the optimal β and bi-partition pair, introducing the principle for empirically selecting the best β value used in this study. Next, we perform clustering on the Insurance Claims dataset using the classical *K*-means and BPW *K*-means methods and compare their clustering performance. The section concludes with two specific analyses: (1) clustering performance with the bi-partition number fixed at 1, and (2) clustering performance with the β value fixed at 0.9.

5.1. Empirical Analysis of Optimal β and Bi-Partition Number Pairing

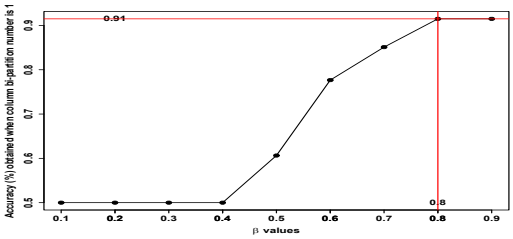
In this section, we empirically search for the optimal β and bi-partition pair and introduce the principle for empirically selecting the best β value used in this study. We will apply this principle in other examples considered in this paper.

Starting with the ranked Insurance Fraud Claims dataset, we evaluate the performance of the BPW *K*-means using various pairs of β values and bi-partition numbers. As shown in Figure 8, the two perpendicular red line segments in the plots intersect to highlight the optimal pairs of β values and bi-partition numbers, along with their corresponding potential accuracies. In this study, we selected the β value and bi-partition number based on the following empirical principle: we began with the minimum pair values (e.g., $\beta = 0.1$ and bi-partition number = 1) and incrementally increased one parameter while holding the other constant. This systematic approach yielded effective results and serves as a practical guideline for selecting β values when analyzing other datasets.

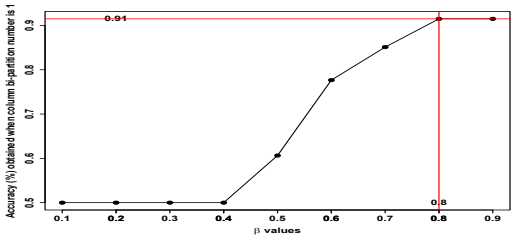
From Figure 8g–l, it is evident that Figure 8g presents the optimal β = 0.1 and bi-partition number 1 pair, where the proposed algorithm achieved an accuracy of 91%. As such, we identified β of 0.8 or 0.9 paired with bi-partition number 1 as the overall optimal parameter choices under the ranked dataset type. For the unranked Insurance Fraud Claims dataset, as shown in Figure 8a–c, the BPW K-means algorithm achieved higher accuracies of 91%, 93%, and 93%, respectively. These results were obtained with the following optimal parameter pairs: β = 0.8 or 0.9 with bi-partition number 1, β = 0.9 with bi-partition number 2, and β = 0.9 with bi-partition number 3. In contrast, Figure 8d–f recorded lower accuracies of 85%, 78%, and 62%, respectively. Based on these observations, we selected a β value of 0.9 paired with either bi-partition number 2 or 3 as the overall optimal parameter choices for the unranked dataset type. In the next step, we conducted further analysis using additional statistical metrics to gain deeper insights into the algorithm's performance based on these parameter settings.

In Output 1 (in the main body) and Output A1 (in Appendix A), the BPW *K*-means algorithm achieved an F1 score—a metric ranging from 0 to 1, with 1 representing the best possible score—of 91.67% on the Insurance Fraud Claims dataset. The corresponding accuracy of 91.49% further validates the results shown in Figure 8a,g. Additionally, Outputs A2 and A3 in Appendix A recorded F1 scores of 92.63% and 92.78%, respectively, while achieving the same approximate accuracy of 93%.

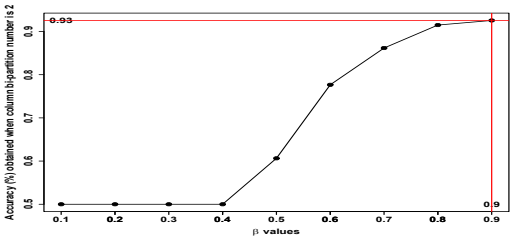
In contrast, Output A4 in Appendix A presents the statistical performance metrics of the classical *K*-means algorithm, which recorded consistently lower F1 scores and accuracies for both ranked and unranked Insurance Fraud Claims datasets. Specifically, it achieved an F1 score of 17.54% and an accuracy of 50%, demonstrating a significant underperformance when compared to the BPW *K*-means algorithm.



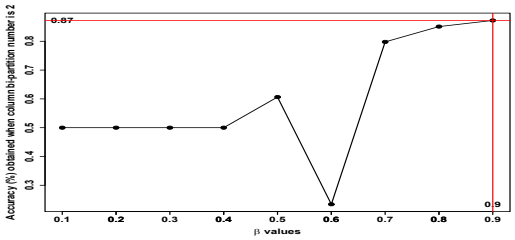
(**a**) Unranked type



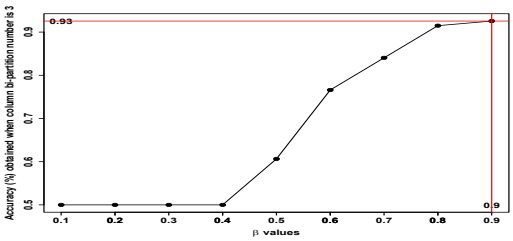
(**g**) Ranked type



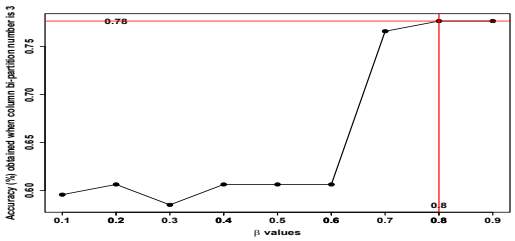
(**b**) Unranked type



(**h**) Ranked type

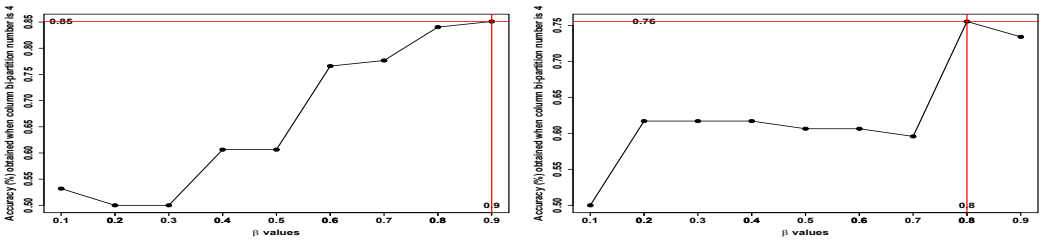


(**c**) Unranked type



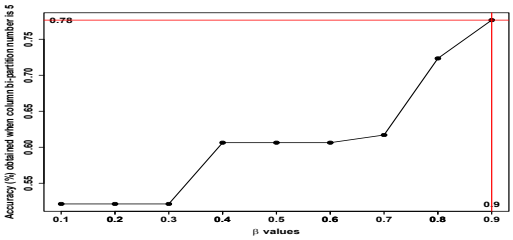
(**i**) Ranked type

Figure 8. *Cont.*

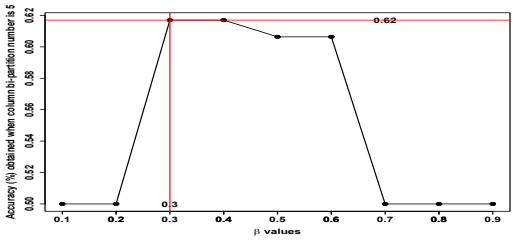


(**d**) Unranked type

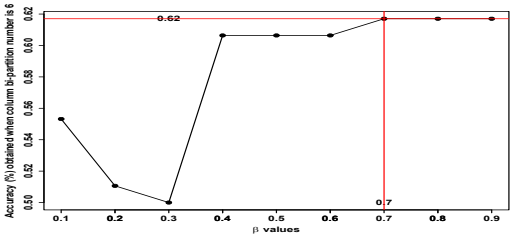
(**j**) Ranked type



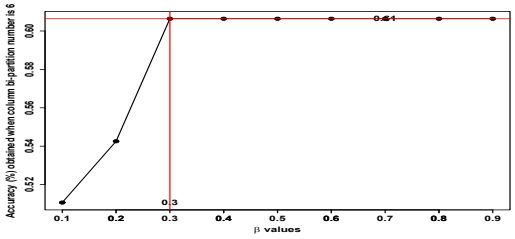
(**e**) Unranked type



(**k**) Ranked type



(**f**) Unranked type



(**l**) Ranked type

Figure 8. Plots of accuracies for the optimal β and bi-partition pair search under the BPW *K*-means algorithm. Six bi-partition numbers are plotted against β values in the range (0, 1). The left column represents Unranked features, and the right column represents Ranked features, using the Insurance Fraud Claims dataset from Kaggle [9].

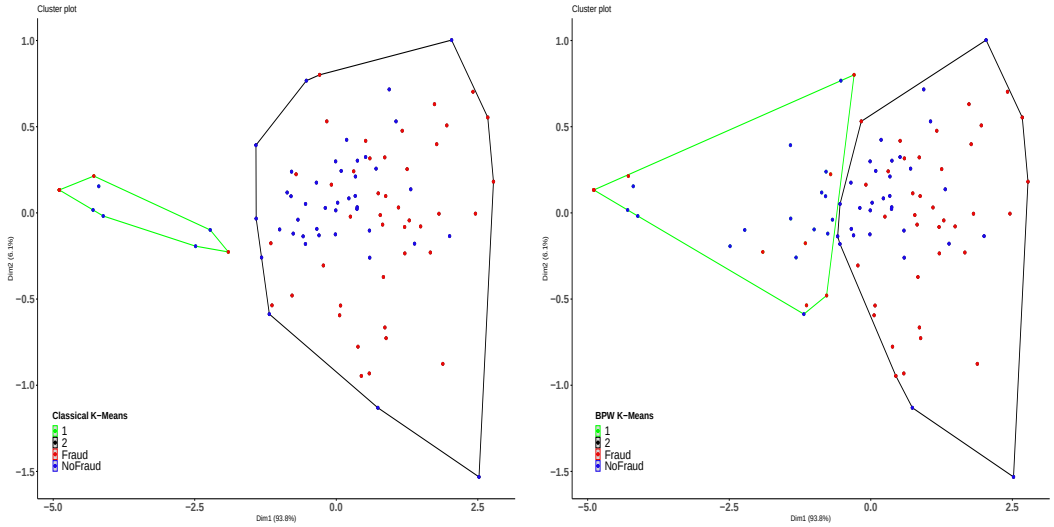
```
Output 1:
 Clustering Performance of the BPW K-Means Algorithm on
      the Insurance Fraud Claims Dataset---Ranked Features

Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 1


  Fraudulent Non-Fraudulent
1 44         3
2 5 42

 Overall Statistics

         Accuracy: 0.9149
           95% CI: (0.8392, 0.9625)
 No Information Rate: 0.5213
 P-Value [Acc > NIR]: <2e-16
```

```
                Kappa : 0.8298

 McNemar's Test P-Value : 0.7237

            Sensitivity : 0.8980
            Specificity : 0.9333
         Pos Pred Value : 0.9362
         Neg Pred Value : 0.8936
              Precision : 0.9362
                 Recall : 0.8980
                     F1 : 0.9167
             Prevalence : 0.5213
         Detection Rate : 0.4681
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.9156
```

Notice that, in what follows, we will use the following numbering convention for outputs: numbers with the capital letter "A" refer to outputs in Appendix A, while numbers without it refer to outputs in the main body of the paper.

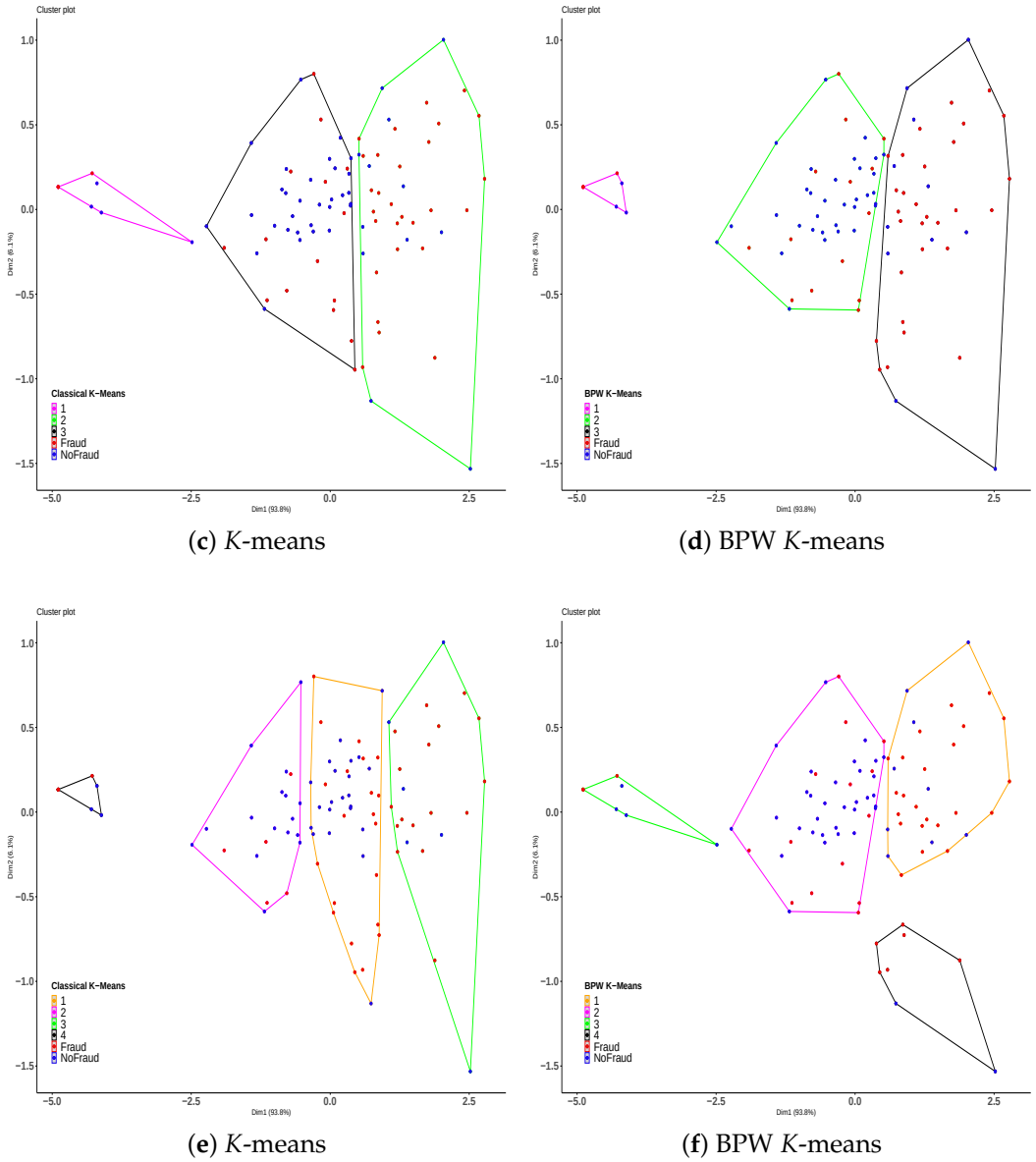
5.2. Classical K-Means and BPW K-Means Clustering of Insurance Claims

We compare different groups of clusters in the Insurance Fraud dataset using three key features: average total claim amount, average property claim, and average vehicle claim. These features are illustrated in Figure 9. These three features were specifically chosen to ensure visualization clarity and readability. The objective of the analysis was to maximize the correct clustering of members within each cluster group. Figure 9 demonstrates that the BPW *K*-means algorithm consistently excelled at achieving this goal compared to the classical *K*-means model. This trend is evident in Figure 9b,d,f, showing how the clustering performance of each model evolves as the number of clusters increases. The visual comparisons of different cluster groupings in the Insurance Fraud dataset confirm that the BPW *K*-means model outperforms the classical *K*-means model.



(**a**) *K*-means (**b**) BPW *K*-means

Figure 9. *Cont.*



(**c**) *K*-means (**d**) BPW *K*-means

(**e**) *K*-means (**f**) BPW *K*-means

Figure 9. Classical *K*-means (**left column**) and BPW *K*-means (**right column**) cluster plots of two features (**first row**), three features (**second row**), four features (**third row**) of fraudulent (red dots), non-fraudulent (blue dots) cases in the northeast United States calculated from the Vehicle Fraud Insurance dataset obtained from Kaggle [9].

The performance of the BPW *K*-means algorithm with respect to different β values (i.e., values between 0 and 1) and different bi-partition numbers (i.e., numbers from 1 to 6) is presented in Tables 1–4. The average accuracy (μ) (which is the mean of the accuracies achieved by the algorithm in multiple iterations) and standard deviation (σ) (being the square root of the sum of the squared deviations from the mean accuracies, divided by the total number of observed accuracies in iterations minus one) of the BPW *K*-means algorithm illustrate variations of clustering performance with different β values and bi-partition numbers.

We compared the BPW *K*-means clustering performance against the classical *K*-means algorithm. Notably, some experiments were conducted using a variant of *K*-means known as *K*-means++. However, the clustering results of *K*-means++ were not significantly different from those of the classical *K*-means when applied to the Insurance Claims and the Iris (to be discussed later) datasets. Consequently, we chose not to repeat the tables and

results for *K*-means++ and instead focused on the classical *K*-means algorithm in all the considered cases in this study.

Table 1 provides a summary of the mean (μ) and standard deviation (σ) of accuracies computed by the BPW *K*-means algorithm for the unranked feature bi-partition type. We observed that when β is between 0 and 0.5, the mean and standard deviation remain relatively consistent across bi-partitions 1 through 6. However, when β is between 0.5 and 1, a significant increase in accuracies and their corresponding standard deviations was observed. Specifically, $\beta = 0.9$ with a bi-partition number of 1 recorded the highest average accuracy of 89%, followed by $\beta = 0.8$ with an average accuracy of 83%. This trend continued to decline as β decreased further toward 0. Additionally, it was noted that average accuracies decreased as the bi-partition number increased to 6. Based on these findings, the optimal parameter choice for unranked feature types is $\beta = 0.9$ and a bi-partition number of 1. In contrast, the classical *K*-means algorithm underperformed, achieving an average accuracy of only 50%.

Table 1. Average clustering accuracy for insurance claim data under bi-partition of unranked features.

Data Column Bi-Partition Number: BPW *K*-Means Feature Bi-Partition Unranked												
Average Accuracy												
	1		**2**		**3**		**4**		**5**		**6**	
β	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
0.1	0.50	0.00	0.50	0.00	0.50	0.00	0.53	0.01	0.52	0.01	0.54	0.02
0.2	0.50	0.00	0.50	0.00	0.50	0.00	0.50	0.00	0.52	0.01	0.51	0.00
0.3	0.50	0.00	0.50	0.00	0.50	0.00	0.50	0.00	0.52	0.01	0.50	0.00
0.4	0.50	0.00	0.50	0.00	0.50	0.00	0.58	0.05	0.58	0.05	0.58	0.05
0.5	0.58	0.05	0.58	0.05	0.57	0.05	0.58	0.05	0.58	0.05	0.58	0.05
0.6	0.63	0.12	0.62	0.10	0.61	0.09	0.60	0.08	0.57	0.05	0.57	0.05
0.7	0.77	0.17	0.68	0.16	0.60	0.14	0.63	0.09	0.58	0.05	0.58	0.05
0.8	0.83	0.17	0.63	0.28	0.59	0.16	0.64	0.13	0.62	0.09	0.57	0.06
0.9	0.89	0.05	0.70	0.29	0.60	0.19	0.63	0.14	0.64	0.09	0.56	0.06
Classical K-Mean Result												
Mean $\mu = 0.50$, Standard Deviation $\sigma = 0.00$												

Table 2 summarizes the mean (μ) and standard deviation (σ) of accuracies calculated by the BPW *K*-means algorithm for ranked feature types. Notably, the optimal average accuracy occurred at $\beta = 0.8$ and $\beta = 0.9$ with bi-partition numbers 1 and 2, achieving an average accuracy of 91%. This represents a significant improvement compared to the classical *K*-means algorithm, which recorded an average accuracy of 50%. This improvement is attributed to the dimensionality reduction techniques employed and the application of weights to distances associated with the dataset's features.

We also observed that the accuracy remains consistently high for $\beta \geq 0.5$. Additionally, the algorithm demonstrates a low and stable standard deviation across different β values, indicating reliable performance. Feature bi-partition (as shown in columns 1 to 6 of Tables 1–4), whether applied to ranked or unranked data features, does not significantly affect the average accuracies and standard deviations. These findings highlight that the BPW *K*-means performs consistently and robustly across variations in feature bi-partitions compared to the classical *K*-means algorithm.

Table 2. Average clustering accuracy for insurance claim data under bi-partition of ranked features.

Data Column Bi-Partition Number: BPW *K*-Means Feature Bi-Partition Unranked												
Average Accuracy												
	1		**2**		**3**		**4**		**5**		**6**	
β	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
0.1	0.50	0.00	0.50	0.00	0.60	0.00	0.50	0.00	0.50	0.00	0.51	0.00
0.2	0.50	0.00	0.50	0.00	0.61	0.00	0.62	0.00	0.50	0.00	0.54	0.00
0.3	0.50	0.00	0.50	0.00	0.59	0.00	0.62	0.00	0.62	0.00	0.61	0.00
0.4	0.50	0.00	0.50	0.00	0.61	0.00	0.62	0.00	0.62	0.00	0.61	0.00
0.5	0.61	0.00	0.61	0.00	0.61	0.00	0.61	0.00	0.61	0.00	0.61	0.00
0.6	0.78	0.00	0.23	0.00	0.61	0.00	0.61	0.00	0.61	0.00	0.61	0.00
0.7	0.85	0.00	0.80	0.00	0.77	0.00	0.60	0.00	0.50	0.00	0.61	0.00
0.8	0.91	0.00	0.85	0.00	0.78	0.00	0.76	0.00	0.50	0.00	0.61	0.00
0.9	0.91	0.00	0.87	0.00	0.78	0.00	0.73	0.00	0.50	0.00	0.61	0.00
Classical K-Mean Result												
Mean $\mu = 0.50$, Standard Deviation $\sigma = 0.00$												

Table 3 summarizes the average Rand index and average adjusted Rand index obtained by both the BPW *K*-means and *K*-means algorithms for unranked feature types. The results indicate that, for β values between 0 and 0.5, both the average Rand index and average adjusted Rand index remain consistent across bi-partitions 1 through 6. However, when β values range between 0.5 and 1, there is a notable performance improvement. Specifically, the highest average Rand index, 86%, was achieved at $\beta = 0.9$ and bi-partition numbers of 2 and 3, followed by an average Rand index of 84% at $\beta = 0.8$. Similarly, the average adjusted Rand index recorded 72%, which also occurred at $\beta = 0.9$ and the bi-partition numbers of 2 and 3. In comparison, the classical *K*-means algorithm demonstrated significantly poorer performance, achieving an average Rand index of 52% and an average adjusted Rand index of 5%. These results validate the improved clustering quality of the BPW *K*-means, particularly because the adjusted Rand index accounts for chance agreement when measuring clustering accuracy.

Table 3. Average Rand and adjusted Rand index for unranked features.

Data Column Bi-Partition Number: BPW *K*-Means Feature Bi-Partition Unranked												
Average Rand Index Average Adjusted Rand Index												
β	**1**	**2**	**3**	**4**	**5**	**6**	**1**	**2**	**3**	**4**	**5**	**6**
0.1	0.49	0.49	0.49	0.50	0.50	0.50	0.00	0.00	0.00	0.00	0.00	0.00
0.2	0.49	0.49	0.49	0.49	0.50	0.49	0.00	0.00	0.00	0.00	0.00	0.00
0.3	0.49	0.49	0.49	0.49	0.50	0.49	0.00	0.00	0.00	0.00	0.00	0.00
0.4	0.49	0.49	0.49	0.52	0.52	0.52	0.00	0.00	0.00	0.04	0.04	0.04
0.5	0.52	0.52	0.52	0.52	0.52	0.52	0.04	0.04	0.04	0.04	0.04	0.04
0.6	0.65	0.65	0.64	0.64	0.52	0.52	0.30	0.30	0.28	0.28	0.04	0.04
0.7	0.74	0.76	0.73	0.65	0.52	0.52	0.49	0.52	0.46	0.30	0.05	0.05
0.8	0.84	0.84	0.84	0.73	0.60	0.52	0.69	0.69	0.69	0.46	0.19	0.05
0.9	0.84	0.86	0.86	0.74	0.65	0.52	0.69	0.72	0.72	0.49	0.30	0.05
Classical K-Mean Result												
RI = 0.52 ARI = 0.05												

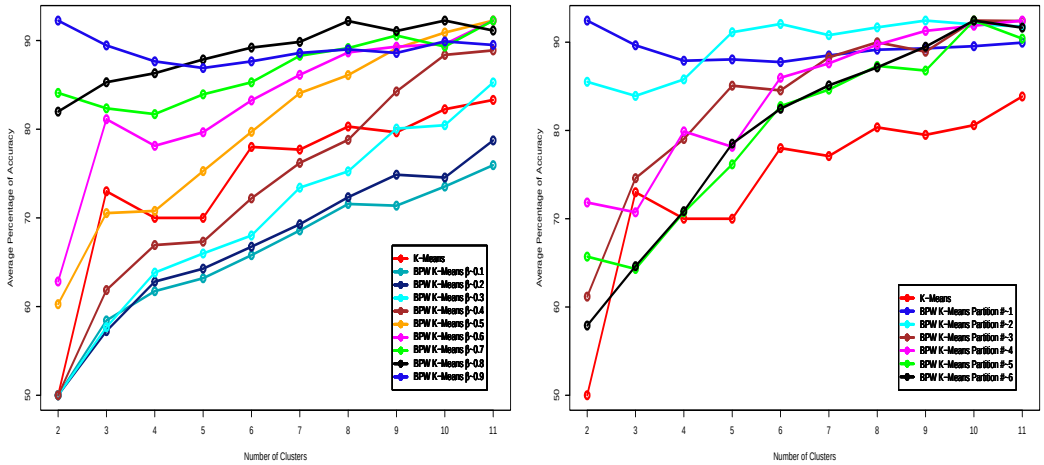
Table 4 presents the output of the average Rand index and average adjusted Rand index achieved by the BPW *K*-means and classical *K*-means algorithms for ranked feature bi-partition types. As with previous results, the findings show that for β values between 0 and 0.6, both the average Rand index and the average adjusted Rand index remain relatively constant across bi-partitions 1 through 6. In contrast, a significant performance improvement is observed when β values range from 0.6 to 1. Specifically, the highest average Rand index, 85%, was achieved at $\beta = 0.9$ with bi-partition number = 1, followed by 79% at $\beta = 0.8$. Similarly, the highest average adjusted Rand index, 69%, was achieved at $\beta = 0.9$ with bi-partition number = 1. In comparison, the classical *K*-means algorithm showed a less significant performance, with an average Rand index of 49% and an average adjusted Rand index of only 0.4%, highlighting the superiority of the proposed BPW *K*-mean algorithm. Collectively, these findings from Table 1 through Table 4 confirm that the BPW *K*-means algorithm significantly outperforms the traditional *K*-means algorithm in terms of clustering performance.

Table 4. Rand and adjusted Rand index for the ranked features.

Data Column Bi-Partition Number: BPW *K*-Means Feature Bi-Partition Unranked												
	Average Rand Index						**Average Adjusted Rand Index**					
β	**1**	**2**	**3**	**4**	**5**	**6**	**1**	**2**	**3**	**4**	**5**	**6**
0.1	0.49	0.49	0.50	0.50	0.49	0.49	0.00	0.00	0.01	0.01	0.01	0.01
0.2	0.49	0.49	0.50	0.50	0.50	0.50	0.00	0.00	0.01	0.01	0.01	0.00
0.3	0.49	0.49	0.50	0.50	0.50	0.51	0.00	0.00	0.01	0.01	0.01	0.02
0.4	0.49	0.49	0.51	0.51	0.51	0.51	0.00	0.00	0.02	0.03	0.03	0.02
0.5	0.51	0.51	0.51	0.51	0.51	0.51	0.03	0.03	0.03	0.03	0.03	0.03
0.6	0.56	0.55	0.51	0.51	0.51	0.51	0.12	0.11	0.03	0.03	0.03	0.03
0.7	0.69	0.60	0.58	0.51	0.49	0.51	0.37	0.21	0.17	0.02	0.00	0.03
0.8	0.79	0.70	0.60	0.55	0.50	0.51	0.58	0.40	0.19	0.11	0.01	0.03
0.9	0.85	0.75	0.59	0.56	0.49	0.51	0.69	0.51	0.19	0.12	0.00	0.03
Classical K-Mean Result												
RI = 0.49 ARI = 0.004												

5.3. Comparative Evaluation of Clustering Performance: BPW K-Means vs. Classical K-Means Across Different Cluster Groupings

Further analysis of accuracy performance was conducted to investigate and compare the clustering performance of the proposed algorithm against the classical *K*-means across different cluster groupings (set of clusters). We examined groupings ranging from 2 clusters to 11 clusters. As shown in Figure 10, the BPW *K*-means algorithm, with β values ranging from 0.1 to 0.9 and feature bi-partition numbers from 1 to 6, exhibits varying performance trajectories in terms of average accuracy across different cluster groupings compared to the classical *K*-means algorithm. In the following two subsections, we fix our method parameters to a $\beta = 0.9$ and a bi-partition number of 1—one of the optimal pairs identified in Section 5.1—for the analysis of sub-cluster groupings.



(**a**) Varying β values (**b**) Varying bi-partition

Figure 10. (**a**) Plots of average accuracies for the trajectories of BPW *K*-means for different β values across various cluster splits, while keeping the bi-partition number equal to 1. (**b**) Plots of average accuracies for the trajectories of the BPW *K*-means for different bi-partition numbers across various cluster splits, while keeping the β value equal to 0.9.

5.3.1. Analysis of Clustering Performance with Bi-Partition Number Fixed at 1

With the bi-partition number fixed at 1, and starting with two clusters, as shown in Figure 10a, the BPW *K*-means algorithm demonstrated superior performance over the *K*-means algorithm, achieving average accuracies of 92.25%, 79.9%, 81.35%, 60.9%, and 58.35% corresponding, respectively, to $\beta = 0.9$, $\beta = 0.8$, $\beta = 0.7$, $\beta = 0.6$, and $\beta = 0.5$, while the classical *K*-means and the BPW *K*-means algorithm with $\beta = 0.1$, $\beta = 0.2$, $\beta = 0.3$ and $\beta = 0.4$ achieved 50%.

Furthermore, in Figure 10a, we can see that by increasing the number of clusters to three, the BPW *K*-means algorithm continued to perform better than the classical *K*-means with average accuracies of 89.65%, 83%, 79.75%, and 76.45% corresponding, respectively, to $\beta = 0.9$, $\beta = 0.8$, $\beta = 0.7$, $\beta = 0.6$, surpassing the classical *K*-means of 73%. However, the BPW *K*-means algorithm with $\beta = 0.1$, $\beta = 0.2$, $\beta = 0.3$, $\beta = 0.4$, and $\beta = 0.5$ had average accuracies ranging from 55.7% to 66.7%, which were below the *K*-means performance. This trend continued as the number of clusters increased from four and five up to eleven. We observed that four weight values of the BPW *K*-means algorithm, namely $\beta = 0.9, \beta = 0.8, \beta = 0.7$, and $\beta = 0.6$ consistently outperformed the classical *K*-means algorithm across all cluster groupings, ranging from 2 to 11 clusters.

In contrast, the BPW *K*-means algorithm with $\beta = 0.1$ and $\beta = 0.2$ never surpassed the classical *K*-means performance across all 10 different cluster types. This comparison provides a deeper understanding of the performance and behavior of the BPW *K*-means algorithm in diverse scenarios. It highlights the impact of distinct β values on the Insurance Claim dataset. For this specific dataset, the BPW *K*-means algorithm with $\beta = 0.9$ emerges as the optimal weight choice, as it consistently achieved high accuracy and effectively handled the dataset's features across all cluster numbers. This demonstrates its robustness and advantages over other weight settings.

5.3.2. Analysis of Clustering Performance with β Value Fixed at 0.9

Fixing the choice of the β-value at 0.9, we conducted a similar analysis by varying the feature bi-partitions. The BPW *K*-means method was applied to the seven features of the Insurance Fraud Claims dataset previously mentioned. The partitioning of these

features was determined by the bi-partition number. For example, a bi-partition number of 3 implies that the first three feature columns form one group, while the remaining fourth to seventh feature columns form another group. Analyzing the clustering performance of the proposed algorithm with bi-partition numbers ranging from 1 to 6 revealed varied performance across different cluster groupings.

From Figure 10b, we observed that the clustering performance under the bi-partition number 1 matched the previously observed blue trajectory in Figure 10a. In the 2 cluster groupings in Figure 10b, the BPW *K*-means algorithm achieved average accuracy of 92.25%, 82.25%, 60.45%, 68.7%, 68.7%, and 56.4% for bi-partition numbers $1, 2, 3, 4, 5$ and 6, respectively. In contrast, the classical *K*-means method achieved an average accuracy of 50%, which was lower than all corresponding BPW *K*-means results. This comparison reinforces the consistently high accuracy and stability of the BPW *K*-means across different bi-partition numbers.

Analyzing the performance of the BPW *K*-means algorithm for 3-cluster groupings in Figure 10b, we observed that bi-partition numbers 1 and 2 outperformed the classical *K*-means, with average accuracies of 89.65% and 85.8%, respectively. In contrast, the classical *K*-means method yielded an average accuracy of 73%. Bi-partition numbers 3 and 4 showed only minimal improvement compared to the classical *K*-means performance, while bi-partition numbers 5 and 6 performed below the classical *K*-means, achieving an average accuracy of around 63%, which was lower than the classical *K*-means performance.

From Figure 10b, it is evident that for cluster groupings ranging from 4 to 11 clusters, the BPW *K*-means algorithm consistently outperformed the classical *K*-means across all bi-partition numbers. These results highlight the reliability, consistency, and superior clustering capabilities of the BPW *K*-means algorithm. Additionally, the BPW *K*-means demonstrated its ability to effectively adapt to varying cluster groupings and bi-partition numbers, a capability that the traditional *K*-means method lacked.

In the next section, we discuss the implications of our findings for BPW *K*-means, particularly its potential applications for insurance companies to better understand and manage claims.

6. Practical Implications and Applications of the BPW *K*-Means Algorithm

The results of our analysis demonstrate that the BPW *K*-means algorithm offers significant improvements over traditional *K*-means, particularly when β values of 0.7, 0.8, and 0.9, and bi-partition numbers 1, 2, and 3, are used. The trajectories of the BPW *K*-means and *K*-means performances are illustrated in Figure 10a, b. These configurations consistently achieved higher clustering accuracies, highlighting the importance of certain features in enhancing overall clustering performance. Specifically, claim frequencies, latitude, and longitude emerged as key features significantly influencing clustering outcomes. In contrast, features such as average total claim amount, average injury claim amount, average property claim amount, and average vehicle claim amount contributed marginally less to clustering performance. This indicates that insurance companies can leverage customer claim frequencies and claim locations as reliable indicators for detecting clusters of fraudulent claims. By identifying these clusters, insurers can respond more effectively by enhancing monitoring processes, reducing claim approval rates, and adjusting premiums for customers within these clusters.

Additionally, insurers can allocate resources more efficiently by focusing on geographical features (latitude and longitude) to target regions with higher fraud prevalence or fraud hotspots. This approach can improve overall risk management and ensure fairer premium structures for customers.

The flexibility of the BPW *K*-means method enables insurers to fine-tune its parameters to prioritize the most relevant features when analyzing complex datasets. Beyond insurance fraud detections, the BPW *K*-means algorithm demonstrates applicability in other industries such as finance and marketing, which also require clustering large datasets. For example, financial institutions could utilize this method to identify fraudulent transactions more effectively, while marketers could analyze customer data to identify patterns and tailor marketing strategies to specific customer segments. Moreover, the enhanced clustering quality achieved through bi-partitioning establishes the BPW *K*-means algorithm as a valuable tool in unsupervised machine learning and data analysis.

7. Application of Clustering Methods to Other Datasets

In this section, we evaluate the performance of the proposed BPW K-means method using three publicly available datasets from the Machine Learning Repository at the University of California, Irvine (UCI) [10]. The analysis is structured as follows: the performance of the BPW K-means method is examined on the Iris dataset in Section 7.1, the Sirtuin6 dataset in Section 7.2, and the Wholesale Customers dataset in Section 7.3. The results demonstrate that the BPW K-means method delivers strong performance across all the considered datasets. The method's effectiveness is evaluated using established statistical benchmarks, and its performance is systematically compared to that of classical K-means. Furthermore, the findings presented in this section are complemented by additional results provided in Appendix A.

7.1. Clustering Performance of BPW K-Means vs. Classical K-Means on the Iris Dataset: Ranked vs. Unranked Features

In this section, we analyze the performance of the BPW K-means method by applying it to the well−known Iris dataset. The Iris dataset consists of 150 observations with four features: petal width, petal length, sepal length, and sepal width. These features belong to three distinct flower species namely: setosa, versicolor, and virginica. Two experiments were conducted on the Iris dataset: one with rank features and the other with unranked features.

In Figure 11a, the first row in the left column presents the unranked data type with a β value of 0.1, and bi-partition number 1, where the BPW *K*-means achieved a potential optimal accuracy of 93%. In comparison, the ranked data type in Figure 11d achieved an accuracy of 96%, using the same bi-partition number of 1 but with a β value of 0.9.

In the second row, Figure 11b shows the unranked dataset with a β value of 0.1, while Figure 11e depicts the ranked data type with a β of 0.9. In both cases, the BPW *K*-means achieved a potential accuracy of 95% with the same bi-partition number of 2.

In the last row, Figure 11c illustrates the unranked data type, where the two perpendicular line segments intersect at the pair $\beta = 0.1$ and bi-partition number 3, achieving a potential accuracy of 96%. Similarly, the ranked data type in Figure 11f achieved a potential optimal accuracy of 89% using the pair β value of 0.5 and bi-partition number 3.

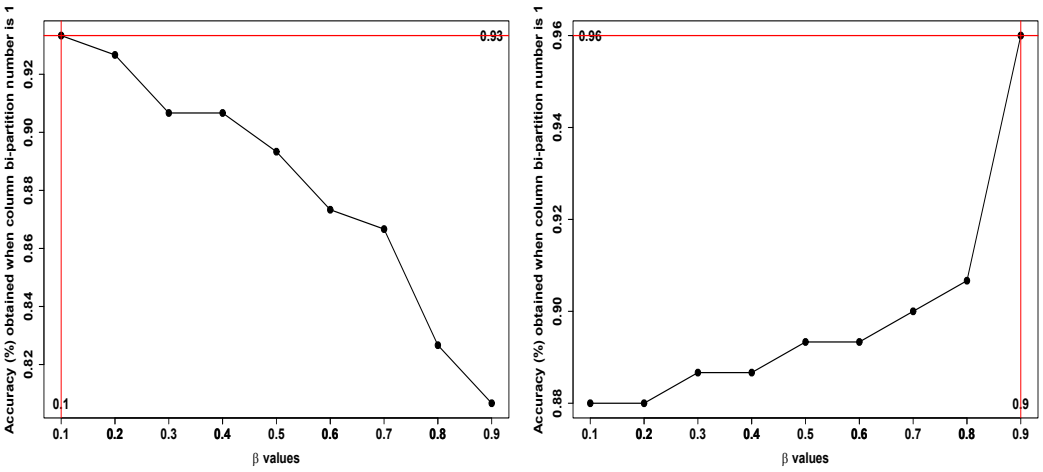
For the ranked data type, we selected the optimal pair $\beta = 0.9$ with a bi-partition number 1, and for the unranked data type, we chose the pair $\beta = 0.1$ with a bi-partition number 1. Next, we performed clustering on the Iris dataset using both BPW *K*-means and classical *K*-means and reported their statistical performance under these parameter settings.

The performance statistics in Output 2 show that the BPW *K*-means algorithm, with $\beta = 0.9$ and a bi-partition number of 1 achieved an accuracy of 96%, with a 95% confidence interval ranging from 92% to 98.5%, compared to the classical *K*-means performance statistics in Output A7, which recorded an accuracy of 89.33%, with a 95% confidence interval ranging from 83.26% to 93.8%. Clearly, the BPW *K*-means algorithm outperforms

the classical *K*-means in most statistical measures. The Kappa statistic, which measures agreement beyond chance, for the proposed method was 94% compared to the classical *K*-means of 84%, a difference of 10%, indicating greater performance. Overall, the BPW *K*-means algorithm performed exceptionally well, with misclassifications primarily occurring in Class 2 and Class 3. Specifically, four virginica flowers were incorrectly classified as versicolor, and two versicolor flowers were misclassified as virginica. In contrast, the classical *K*-means misclassified 16 flowers: 14 virginica and 2 versicolor. In Output A5, when the bi-partition number of the Iris dataset was increased to 2 while keeping $\beta = 0.9$, the BPW *K*-means algorithm achieved a slightly lower accuracy, than its earlier performance, with an accuracy of 94.67%. Eight virginica flowers were misclassified, but the BPW *K*-means algorithm still outperformed the classical *K*-means in most statistical measures, as shown in Output A5. In Output A6, with a bi-partition number of 3 and $\beta = 0.9$, both algorithms achieved equal performance.

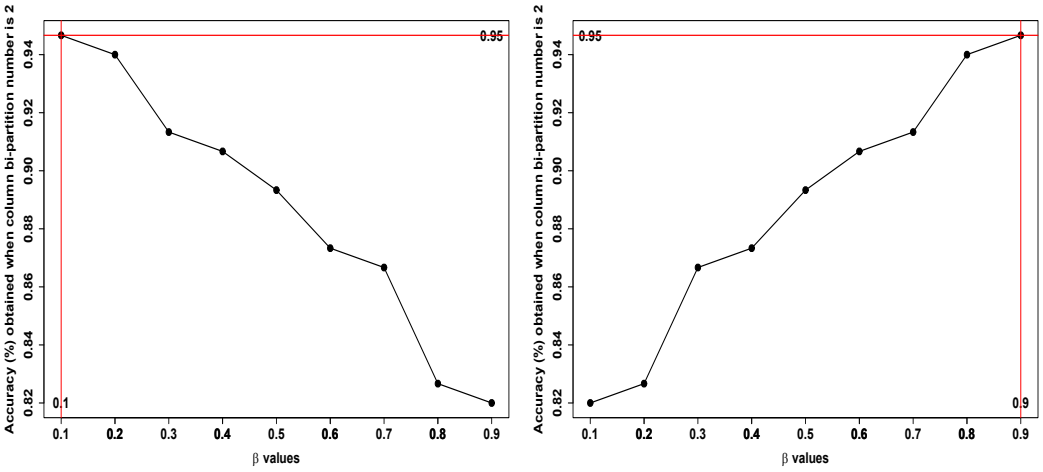
The performance statistics using the classical *K*-means algorithm on the unranked Iris dataset were consistent with those on the ranked dataset. By setting $\beta = 0.1$ and the bi-partition number to 1, and comparing the results from Output A7 with Output A8, the BPW *K*-means algorithm achieved an accuracy of 93.33% and a Kappa value of 90%. The proposed method misclassified 10 virginica flowers as versicolor. In Output A9, the performance of the BPW *K*-means algorithm under unranked settings is similar to the statistics reported in Output A5 (under the ranked feature settings), where an accuracy of 94.67% was recorded. Similarly, Output A10 shows performance statistics comparable to the ranked feature setting results displayed in Output 2.

In Outputs 2 and A10, the F1 scores were recorded as 1, 0.94, and 0.94 for the three classes (setosa, versicolor, and virginica) in the Iris dataset. These results demonstrate that the BPW *K*-means achieved an excellent balance between precision and recall. Similar trends were observed in Outputs A5 and A9, with F1 scores of 1, 0.93, and 0.91. Output A6 also achieved F1 scores of 1, 0.93, and 0.91, while Output A8 recorded slightly lower F1 scores of 1, 0.91, and 0.89. In comparison, Output A7 showed that the classical K-means algorithm achieved overall lower F1 scores of 1, 0.86, and 0.82 for the setosa, versicolor, and virginica classes, respectively.

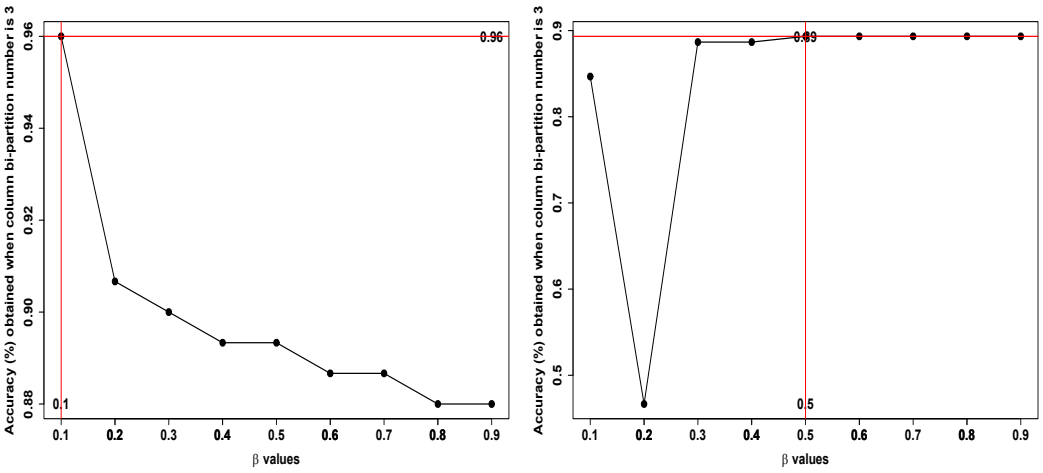


(**a**) Unranked type (**d**) Ranked type

Figure 11. *Cont.*



(**b**) Unranked type (**e**) Ranked type



(**c**) Unranked type (**f**) Ranked type

Figure 11. Plots of accuracies for optimal β and bi-partition number search under the BPW *K*-means algorithm. The first row shows bi-partition number 1 against $\beta \in (0,1)$, the second row shows bi-partition number 2 against $\beta \in (0,1)$, and the third row shows bi-partition number 3 against $\beta \in (0,1)$. The left column represents the unranked Iris dataset, while the right column represents the ranked Iris dataset, both obtained from UCI [43].

```
Output 2: Clustering Performance of the BPW K-Means Algorithm on
          the Iris Dataset---Ranked Features

Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 1


  setosa versicolor virginica
1 50         0         0
2  0        48         2
3  0         4        46
```

```
Overall Statistics

               Accuracy : 0.96
                 95% CI : (0.915, 0.9852)
    No Information Rate : 0.3467
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.94

 McNemar's Test P-Value : NA

Statistics by Class:

                     Class:1 Class:2 Class:3
Sensitivity           1.0000  0.9231  0.9583
Specificity           1.0000  0.9796  0.9608
Pos Pred Value        1.0000  0.9600  0.9200
Neg Pred Value        1.0000  0.9600  0.9800
Precision             1.0000  0.9600  0.9200
Recall                1.0000  0.9231  0.9583
F1                    1.0000  0.9412  0.9388
Prevalence            0.3333  0.3467  0.3200
Detection Rate        0.3333  0.3200  0.3067
Detection Prevalence  0.3333  0.3333  0.3333
Balanced Accuracy     1.0000  0.9513  0.9596
```

The superior clustering performance demonstrated by the BPW *K*-means, compared to the traditional *K*-means on both ranked and unranked feature types of the Iris dataset, highlights the proposed algorithm's versatility and effectiveness. This performance underscores the BPW *K*-means algorithm capability to handle diverse datasets and confirms that its applicability extends beyond insurance fraud claim datasets.

7.2. Clustering Performance of the BPW K-Means vs. Classical K-Means on the Sirtuin6 Dataset: Ranked vs. Unranked Features

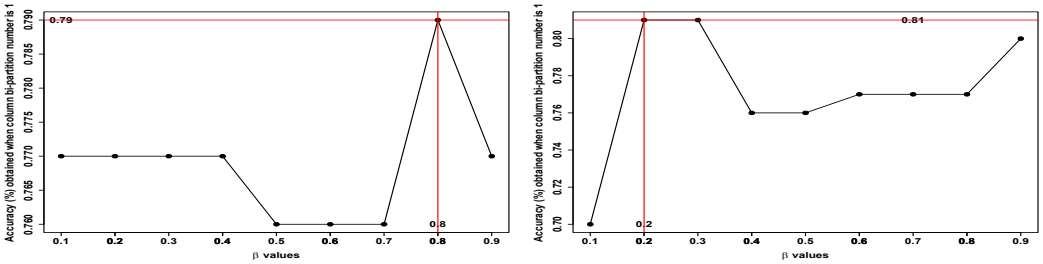
In this section, we once again demonstrate the capabilities of the BPW K-means algorithm and describe the process of selecting β values and bi-partition numbers. The dataset used for this analysis is the Sirtuin6 dataset [44], a biological classification dataset containing 100 observations and 6 features that describe small protein molecules. The dataset was obtained from the University of California Irvine (UCI) database [10]. The features include SC-5, SP-6, SHBd, minHaaCH, maxwHBa, and FMF. The dataset is classified into two classes: low- and high-binding free energies (BFEs), to cluster proteins into these categories. From Figure 12, the first column group (Figure 12a–e) represents the unranked data type, while the second column group (Figure 12f–j) corresponds to the ranked data type. The rows in this figure display the following information:

- First row: In Figure 12a, the BPW K-means achieved an optimal accuracy of 79% with the pair β = 0.8 and bi-partition number = 1 for the unranked data type (left column). Similarly, in Figure 12f, the ranked data type (right column) achieved the highest accuracies with β values of 0.2 and 0.3, paired with the same bi-partition number 1. These pairs achieved a potential optimal accuracy of 81%.
- Second row: In Figure 12b, the unranked data type achieved an optimal pair with β = 0.1 and bi-partition number = 2, leading to a potential accuracy of 82%. Likewise,

in Figure 12g, the ranked data type achieved optimal accuracies of 81% using β values of 0.2 and 0.3 paired with bi-partition number = 2.

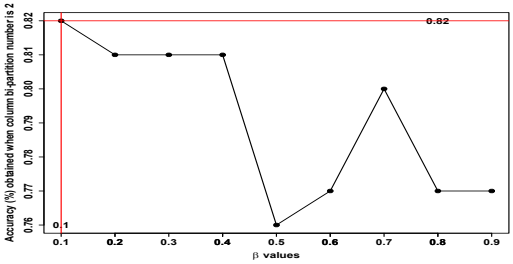
- Third row: In Figure 12c, the unranked data type achieved a potential optimal accuracy of 82% with the pair β = 0.1 and bi-partition number = 3. Similarly, the ranked data type in Figure 12h achieved the same potential optimal accuracy of 82% with β = 0.1 and bi-partition number = 3.
- Fourth row: In Figure 12d, the BPW K-means achieved a potential optimal accuracy of 82% for the unranked data type with the pair β = 0.1 and bi-partition number = 4. For the ranked data type (Figure 12i), β values of 0.2, 0.3, and 0.4 paired with bi-partition number = 4 achieved a slightly lower accuracy of 81%.
- Fifth row: Figure 12e,j exhibit similar patterns, where the BPW K-means algorithm achieved potential optimal accuracies of 79% and 78%, respectively, using β = 0.1 and bi-partition number = 5 for both ranked and unranked data types.

Based on the analysis above, for the ranked Sirtuin6 dataset [44], the optimal pair is β = 0.1 and bi-partition number = 3, as this pair achieved the best accuracy compared to others. For the unranked data type, β = 0.1 consistently performed well, achieving optimal solutions when paired with bi-partition numbers 2, 3, and 4. Using these optimal parameter choices, we conducted further analysis employing statistical measures for deeper insights. The results are presented in Outputs 3, A11, and A12. In Outputs 3 and A11, BPW K-means achieved an F1 score of 0.82 on the Sirtuin6 dataset. In contrast, the classical K-means algorithm (Output A12) recorded a lower F1 score of 0.79, further highlighting the superiority of the BPW K-means algorithm.

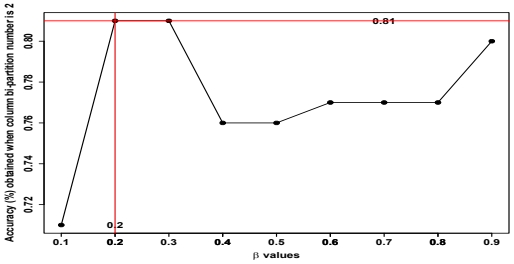


(**a**) Unranked type

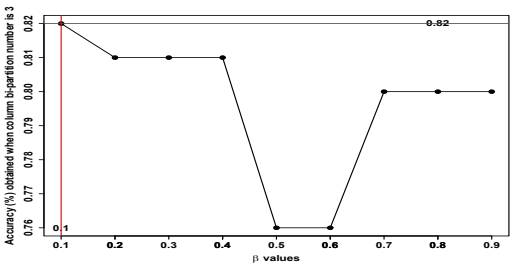
(**f**) Ranked type



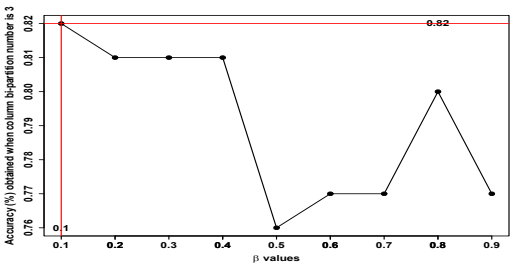
(**b**) Unranked type



(**g**) Ranked type

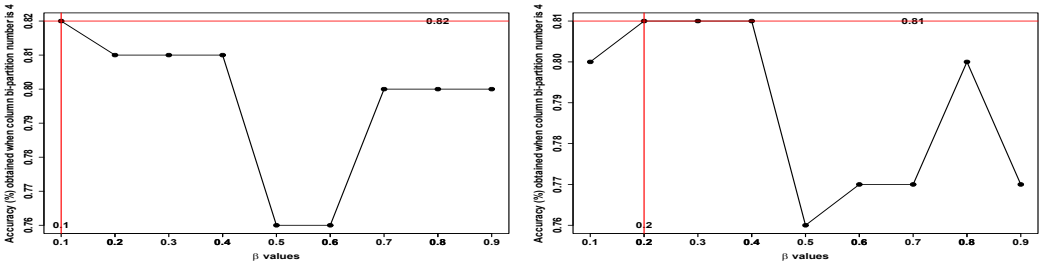


(**c**) Unranked type

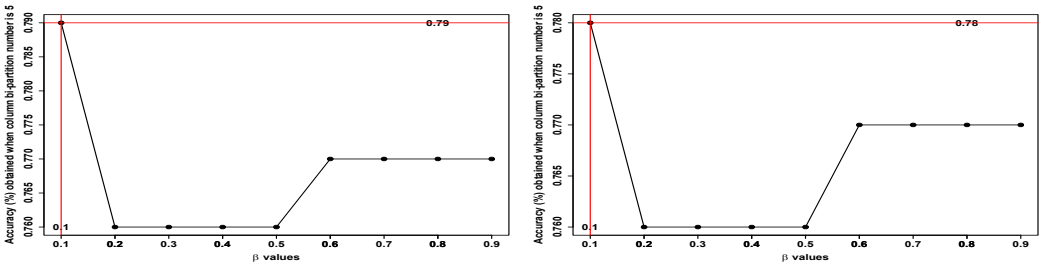


(**h**) Ranked type

Figure 12. *Cont.*



(**d**) Unranked type (**i**) Ranked type



(**e**) Unranked type (**j**) Ranked type

Figure 12. Plots of accuracies for optimal β and bi-partition number pair searches under the BPW *K*-means algorithm. The first row shows bi-partition number 1 against $\beta \in (0,1)$, the second row shows bi-partition number 2 against $\beta \in (0,1)$, the third row shows bi-partition number 3 against $\beta \in (0,1)$, the fourth row shows bi-partition number 4 against $\beta \in (0,1)$, and the fifth row shows bi-partition number 5 against $\beta \in (0,1)$. The left column represents the unranked Sirtuin6 dataset, while the right column represents the ranked Sirtuin6 dataset, both obtained from UCI [44].

```
Output 3: Clustering Performance of the BPW K-Means Algorithm on
          the Sirtuin6 Dataset---Ranked Features

Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.1, Bi-Partition Number = 3

  Low BFE High BFE
1  41       9
2  9  41

 Overall Statistics

          Accuracy: 0.82
            95% CI: (0.7305, 0.8897)
 No Information Rate: 0.5
 P-Value [Acc > NIR]: 3.074e-11

           Kappa: 0.64

 McNemar's Test P-Value: 1

       Sensitivity: 0.82
       Specificity: 0.82
     Pos Pred Value: 0.82
```

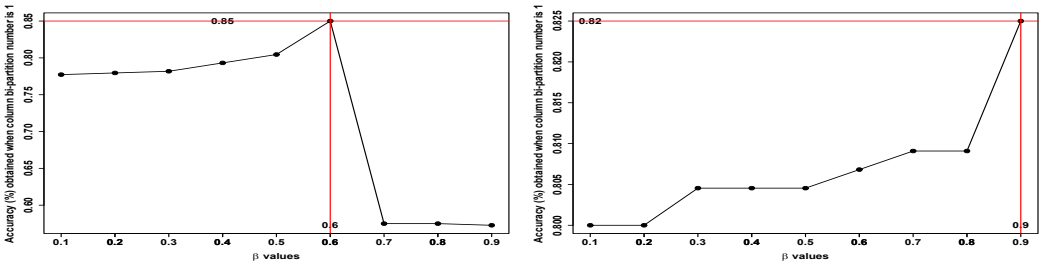
```
Neg Pred Value: 0.82
Precision: 0.82
Recall: 0.82
F1: 0.82
Prevalence: 0.50
Detection Rate: 0.41
Detection Prevalence: 0.50
Balanced Accuracy: 0.82
```

7.3. Clustering Performance of the BPW K-Means vs. Classical K-Means on the Wholesale Customers Dataset: Ranked vs. Unranked Features

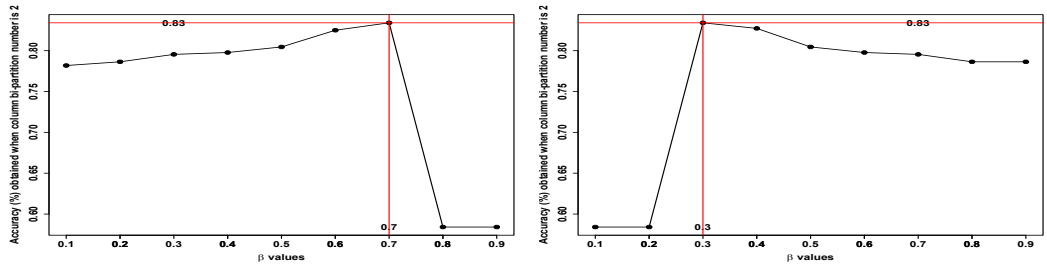
In this section, we further demonstrate the capabilities of the BPW K-means algorithm and once again outline the process for selecting β values and bi-partition numbers using the Wholesale Customers dataset from UCI [10]. This dataset represents clients of a wholesale distributor, comprising 440 observations and six features: fresh, milk, grocery, frozen, detergent paper, and delicatessen. The dataset has two categorical target variables: Channel and Region. The channel variable has two classes—"Horeca" (hotel/restaurant/café) and "Retail"—while the region variable includes three categories: Lisbon, Oporto, and Other. For this study, we used 'Channel' as the dependent variable and clustered the customers into two groups: Horeca and Retail. Figure 13 illustrates the results, with the first column (Figure 13a–e) representing the unranked data type and the second column (Figure 13f–j) representing the ranked data type. The rows of this figure display the following information.

- First row: From Figure 13a,f, the unranked dataset achieved a potential optimal accuracy of 85% with the pair (β = 0.6, bi-partition number = 1), while the ranked dataset achieved a potential accuracy of 82% with the pair (β = 0.9, bi-partition number = 1).
- Second row: In Figure 13b,g, the BPW K-means achieved an accuracy of 83% for both the unranked and ranked datasets, with the optimal pair (β = 0.7, bi-partition number = 2) for the unranked dataset and (β = 0.3, bi-partition number = 2) for the ranked dataset.
- Third row: In this row, the BPW K-means achieved a potential accuracy of 82% for the unranked dataset in Figure 13c with the pair (β = 0.1, bi-partition number = 3). For the ranked dataset in Figure 13h, the optimal pair (β = 0.4, bi-partition number = 3) achieved a potential accuracy of 85%.
- Fourth row: Figure 13d shows that the unranked dataset achieved an accuracy of 81% with the pair (β = 0.3, bi-partition number = 4). In Figure 13i, the ranked dataset achieved a potential accuracy of 85% with the pair (β = 0.4, bi-partition number = 4).
- Fifth row: Figure 13e demonstrates that the unranked Wholesale Customers dataset achieved a potential accuracy of 80% with the pair (β = 0.4, bi-partition number = 5). Similarly, Figure 13j shows that the ranked dataset achieved the same accuracy with the same pair.

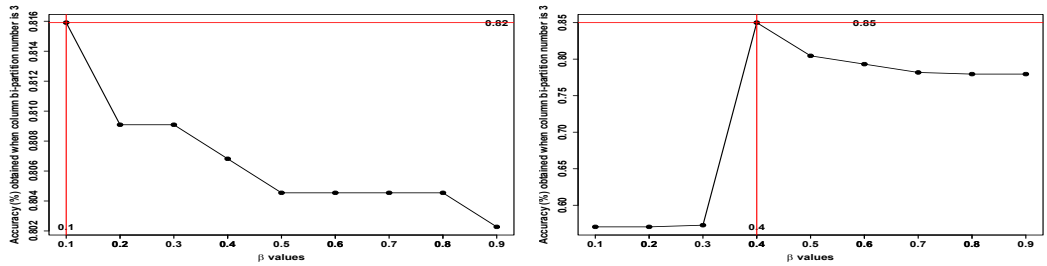
Based on the above analysis, for the ranked Wholesale Customers dataset, the overall optimal pairs were identified as (β = 0.4, bi-partition number = 3) and (β = 0.4, bi-partition number = 4), as these combinations achieved the highest accuracy of 85% under the ranked settings. For the unranked dataset, the optimal pair was (β = 0.6, bi-partition number = 2), which achieved an accuracy of 83%. With these parameter choices, further statistical analysis was conducted to gain deeper insights. From Output 4, the BPW K-means algorithm achieved an approximate F1 score of 0.90 on the Wholesale Customers dataset. Similarly, Output A13 recorded an F1 score of 0.88. In contrast, the classical K-means algorithm recorded a significantly lower F1 score of 0.73 in Output A14. These results highlight the reliability and superiority of the BPW K-means algorithm compared to the classical K-means approach.



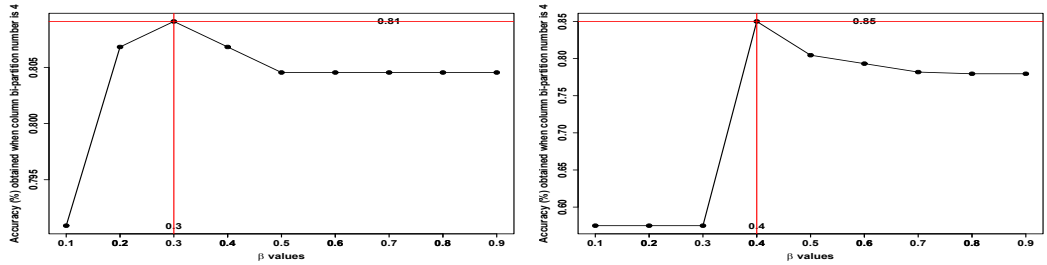
(**a**) Unranked type (**f**) Ranked type



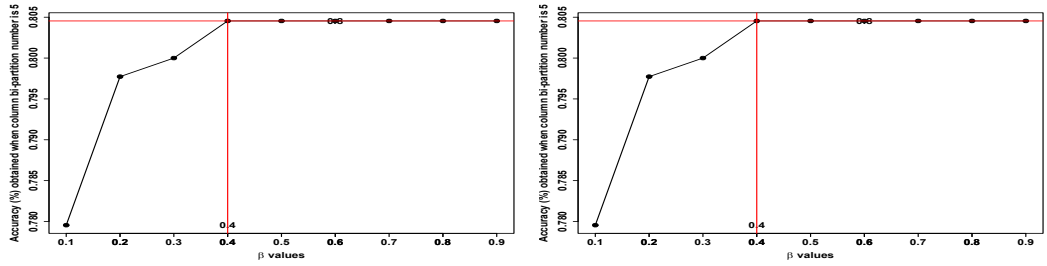
(**b**) Unranked type (**g**) Ranked type



(**c**) Unranked type (**h**) Ranked type



(**d**) Unranked type (**i**) Ranked type



(**e**) Unranked type (**j**) Ranked type

Figure 13. Plots of accuracies for optimal β and bi-partition number pair searches under the BPW K-means algorithm. The first row shows bi-partition number 1 against $\beta \in (0,1)$, the second row shows bi-partition number 2 against $\beta \in (0,1)$, the third row shows bi-partition number 3 against $\beta \in (0,1)$, the fourth row shows bi-partition number 4 against $\beta \in (0,1)$, and the fifth row shows bi-partition number 5 against $\beta \in (0,1)$. The left column represents the unranked Wholesale Customers dataset, while the right column represents the ranked Wholesale Customers dataset, both obtained from UCI [45].

```
Output 4: Clustering Performance of the BPW K–Means Algorithm on
       the Wholesale Customers Dataset---Ranked Features

Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.4, Bi-Partition Number = 3


   Horeca   Retail
1 288      10
2  56 86


 Overall Statistics

       Accuracy: 0.85
         95% CI: (0.8132, 0.8821)
 No Information Rate: 0.7818
 P-Value [Acc > NIR]: 0.0001987

        Kappa: 0.6251

 McNemar's Test P-Value: 3.04e-08

      Sensitivity: 0.8372
      Specificity: 0.8958
    Pos Pred Value: 0.9664
    Neg Pred Value: 0.6056
       Precision: 0.9664
          Recall: 0.8372
              F1: 0.8972
      Prevalence: 0.7818
   Detection Rate: 0.6545
 Detection Prevalence: 0.6773
  Balanced Accuracy: 0.8665
```

Note: The clustering performance of bi-partition numbers 3 and 4 was identical for the ranked features on the Wholesale Customers dataset. Therefore, we provide only one output of the optimal pair solutions.

8. Conclusions and Future Work

In this study, we introduced a novel clustering algorithm called BPW *K*-means, which enhances the classical *K*-means method. We derived and presented the mathematical objective function for the BPW *K*-means algorithm and evaluated its performance on a vehicle Insurance Fraud Claims dataset using various metrics and visualizations. The proposed method demonstrated superior clustering performance compared to the classical *K*-means, particularly in terms of average accuracy. Furthermore, we found that the selection of appropriate weight values (β values) and feature bi-partitioning are crucial parameters for achieving optimal performance with the BPW *K*-means algorithm. Moreover, we applied and evaluated the clustering performance of the BPW *K*-means algorithm on

three additional datasets obtained from the Machine Learning Repository at the University of California, Irvine (UCI) [10]. The proposed algorithm consistently demonstrated superior clustering performance compared to the classical *K*-means algorithm, delivering promising results across these datasets.

The success of the BPW *K*-means algorithm has potentially transformative implications across multiple domains. By effectively addressing the challenge of fraudulent activities using this advanced clustering approach, the method can substantially mitigate financial losses and save considerable resources for both insurance companies and policyholders.

Future research could explore the applicability of the bi-partition weighted *K*-means (BPW *K*-means) method to other clustering algorithms, such as hierarchical clustering. Additionally, given the improved clustering quality achieved with bi-partitioning, the method could be extended to support multi-feature partitioning, broadening its usability for more complex datasets.

Author Contributions: Conceptualization, S.X.; Methodology, F.K.C. and S.X.; Software, F.K.C.; Validation, F.K.C.; Formal analysis, F.K.C., S.X. and A.T.L.; Investigation, S.X. and A.T.L.; Writing—original draft, F.K.C. and S.X.; Writing—review & editing, S.X. and A.T.L.; Visualization, F.K.C.; Supervision, S.X. and A.T.L.; Project administration, S.X. and A.T.L.; Funding acquisition, A.T.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by A.T. Lawniczak's The Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant.

Data Availability Statement: The original datasets used in this study are publicly available on UCI (https://archive.ics.uci.edu/datasets) and Kaggle (https://www.kaggle.com/code/arpan129/eda-on-insurance-claim-fraud-detection/input) or (https://www.kaggle.com/datasets/aashishjhamtani/automobile-insurance/data) repositories.

Acknowledgments: A.T. Lawniczak and Francis K. Combert acknowledge financial support from The Natural Sciences and Engineering Research Council of Canada (NSERC).

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A

Output A1: Clustering Performance of the BPW K–Means Algorithm on the Insurance Fraud Claims Dataset–––Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 1

  Fraudulent Non-Fraudulent
1 44         3
2 5 42

 Overall Statistics

        Accuracy: 0.9149
          95% CI: (0.8392, 0.9625)
 No Information Rate: 0.5213
```

```
P-Value [Acc > NIR]: <2e-16

        Kappa: 0.8298

McNemar's Test P-Value: 0.7237

     Sensitivity: 0.8980
     Specificity: 0.9333
   Pos Pred Value: 0.9362
   Neg Pred Value: 0.8936
     Precision: 0.9362
       Recall: 0.8980
          F1: 0.9167
     Prevalence: 0.5213
   Detection Rate: 0.4681
Detection Prevalence: 0.5000
 Balanced Accuracy: 0.9156
```

Output A2: Clustering Performance of the BPW K-Means Algorithm on the Insurance Fraud Claims Dataset---Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 2

  Fraudulent Non-Fraudulent
1 44         3
2 4 43

 Overall Statistics

        Accuracy: 0.9255
         95% CI: (0.8526, 0.9695)
 No Information Rate: 0.5106
 P-Value [Acc > NIR]: <2e-16

        Kappa: 0.8511

McNemar's Test P-Value: 1

     Sensitivity: 0.9167
     Specificity: 0.9348
   Pos Pred Value: 0.9362
   Neg Pred Value: 0.9149
     Precision: 0.9362
       Recall: 0.9167
          F1: 0.9263
     Prevalence: 0.5106
   Detection Rate: 0.4681
Detection Prevalence: 0.5000
```

```
    Balanced Accuracy: 0.9257
```

Output A3: Clustering Performance of the BPW K–Means Algorithm on the Insurance Fraud Claims Dataset–––Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 3

  Fraudulent Non-Fraudulent
1 45         2
2  5        42

 Overall Statistics

         Accuracy: 0.9255
           95% CI: (0.8526, 0.9695)
 No Information Rate: 0.5319
 P-Value [Acc > NIR]: <2e-16

          Kappa: 0.8511

 McNemar's Test P-Value: 0.4497

      Sensitivity: 0.9000
      Specificity: 0.9545
    Pos Pred Value: 0.9574
    Neg Pred Value: 0.8936
       Precision: 0.9574
          Recall: 0.9000
              F1: 0.9278
       Prevalence: 0.5319
    Detection Rate: 0.4787
  Detection Prevalence: 0.5000
    Balanced Accuracy: 0.9273
```

Output A4: Clustering Performance of the Classical K–Means Algorithm on the Insurance Fraud Claims Dataset–––Ranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = NA, Bi-Partition Number = NA


  Fraudulent Non-Fraudulent
1  5        42
2  5        42
```

```
Overall Statistics

        Accuracy: 0.5
        95% CI: (0.3951, 0.6049)
 No Information Rate: 0.8936
 P-Value [Acc > NIR]: 1

        Kappa: 0

McNemar's Test P-Value: 1.512e-07

      Sensitivity: 0.50000
      Specificity: 0.50000
    Pos Pred Value: 0.10638
    Neg Pred Value: 0.89362
       Precision: 0.10638
          Recall: 0.50000
            F1: 0.17544
      Prevalence: 0.10638
    Detection Rate: 0.05319
 Detection Prevalence: 0.50000
  Balanced Accuracy: 0.50000
```

Note: The clustering performance of the classical K-means algorithm was identical for both ranked and unranked features on the Insurance Fraud Claims dataset. Therefore, we provide the output solely for the ranked features.

Output A5: Clustering Performance of the BPW K-Means Algorithm on the Iris Dataset---Ranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 2


  setosa versicolor virginica
1 50      0      0
2 0     50      0
3 0      8 42

Overall Statistics

        Accuracy: 0.9467
        95% CI: (0.8976, 0.9767)
 No Information Rate: 0.3867
 P-Value [Acc > NIR]: < 2.2e-16

        Kappa: 0.92

McNemar's Test P-Value: NA
```

```
Statistics by Class:

                     Class:1 Class:2 Class:3
Sensitivity           1.0000  0.8621  1.0000
Specificity           1.0000  1.0000  0.9259
Pos Pred Value        1.0000  1.0000  0.8400
Neg Pred Value        1.0000  0.9200  1.0000
Precision             1.0000  1.0000  0.8400
Recall                1.0000  0.8621  1.0000
F1                    1.0000  0.9259  0.9130
Prevalence            0.3333  0.3867  0.2800
Detection Rate        0.3333  0.3333  0.2800
Detection Prevalence  0.3333  0.3333  0.3333
Balanced Accuracy     1.0000  0.9310  0.9630
```

Output A6: Clustering Performance of the BPW K–Means Algorithm on the Iris Dataset---Ranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.9, Bi-Partition Number = 3


  setosa versicolor virginica
1 50      0       0
2 0      48       2
3 0      14 36

Overall Statistics

               Accuracy: 0.8933
                 95% CI: (0.8326, 0.9378)
    No Information Rate: 0.4133
    P-Value [Acc > NIR]: < 2.2e-16

                  Kappa: 0.84

 McNemar's Test P-Value: NA

Statistics by Class:

                     Class:1 Class:2 Class:3
Sensitivity           1.0000  0.7742  0.9474
Specificity           1.0000  0.9773  0.8750
Pos Pred Value        1.0000  0.9600  0.7200
Neg Pred Value        1.0000  0.8600  0.9800
Precision             1.0000  0.9600  0.7200
Recall                1.0000  0.7742  0.9474
F1                    1.0000  0.8571  0.8182
```

```
Prevalence          0.3333   0.4133   0.2533
Detection Rate      0.3333   0.3200   0.2400
Detection Prevalence 0.3333  0.3333   0.3333
Balanced Accuracy   1.0000   0.8757   0.9112
```

Output A7: Clustering Performance of the Classical K–Means Algorithm on the Iris Dataset---Ranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = NA, Bi-Partition Number = NA


  setosa versicolor virginica
1 50      0         0
2 0       48        2
3 0       14        36

Overall Statistics

        Accuracy: 0.8933
          95% CI: (0.8326, 0.9378)
  No Information Rate: 0.4133
  P-Value [Acc > NIR]: < 2.2e-16

         Kappa: 0.84

 McNemar's Test P-Value: NA

Statistics by Class:

                     Class:1  Class:2  Class:3
Sensitivity          1.0000   0.7742   0.9474
Specificity          1.0000   0.9773   0.8750
Pos Pred Value       1.0000   0.9600   0.7200
Neg Pred Value       1.0000   0.8600   0.9800
Precision            1.0000   0.9600   0.7200
Recall               1.0000   0.7742   0.9474
F1                   1.0000   0.8571   0.8182
Prevalence           0.3333   0.4133   0.2533
Detection Rate       0.3333   0.3200   0.2400
Detection Prevalence 0.3333   0.3333   0.3333
Balanced Accuracy    1.0000   0.8757   0.9112
```

Note: The clustering performance of the classical K-means algorithm was identical for both ranked and unranked features on the Iris dataset. Therefore, we provide the output solely for the ranked features.

Output A8: Clustering Performance of the BPW K–Means Algorithm on the Iris Dataset---Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.1, Bi-Partition Number = 1


  setosa versicolor virginica
1 50      0       0
2 0      50       0
3 0      10      40

Overall Statistics

               Accuracy: 0.9333
                 95% CI: (0.8808, 0.9676)
    No Information Rate: 0.4
    P-Value [Acc > NIR]: < 2.2e-16

                  Kappa: 0.9

 McNemar's Test P-Value: NA

Statistics by Class:

                     Class:1 Class:2 Class:3
Sensitivity           1.0000  0.8333  1.0000
Specificity           1.0000  1.0000  0.9091
Pos Pred Value        1.0000  1.0000  0.8000
Neg Pred Value        1.0000  0.9000  1.0000
Precision             1.0000  1.0000  0.8000
Recall                1.0000  0.8333  1.0000
F1                    1.0000  0.9091  0.8889
Prevalence            0.3333  0.4000  0.2667
Detection Rate        0.3333  0.3333  0.2667
Detection Prevalence  0.3333  0.3333  0.3333
Balanced Accuracy     1.0000  0.9167  0.9545
```

Output A9: Clustering Performance of the BPW K-Means Algorithm on the Iris Dataset---Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.1, Bi-Partition Number = 2


  setosa versicolor virginica
1 50      0       0
2 0      50       0
3 0       8      42
```

```
Overall Statistics

        Accuracy: 0.9467
          95% CI: (0.8976, 0.9767)
  No Information Rate: 0.3867
  P-Value [Acc > NIR]: < 2.2e-16

           Kappa: 0.92

 McNemar's Test P-Value: NA

Statistics by Class:

                     Class:1 Class:2 Class:3
Sensitivity           1.0000  0.8621  1.0000
Specificity           1.0000  1.0000  0.9259
Pos Pred Value        1.0000  1.0000  0.8400
Neg Pred Value        1.0000  0.9200  1.0000
Precision             1.0000  1.0000  0.8400
Recall                1.0000  0.8621  1.0000
F1                    1.0000  0.9259  0.9130
Prevalence            0.3333  0.3867  0.2800
Detection Rate        0.3333  0.3333  0.2800
Detection Prevalence  0.3333  0.3333  0.3333
Balanced Accuracy     1.0000  0.9310  0.9630
```

Output A10: Clustering Performance of the BPW K-Means Algorithm on the Iris Dataset---Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.1, Bi-Partition Number = 3


  setosa versicolor virginica
1 50      0      0
2 0      48      2
3 0       4 46

Overall Statistics

        Accuracy: 0.96
          95% CI: (0.915, 0.9852)
  No Information Rate: 0.3467
  P-Value [Acc > NIR]: < 2.2e-16

           Kappa: 0.94

 McNemar's Test P-Value: NA
```

```
Statistics by Class:

                     Class:1 Class:2 Class:3
Sensitivity           1.0000  0.9231  0.9583
Specificity           1.0000  0.9796  0.9608
Pos Pred Value        1.0000  0.9600  0.9200
Neg Pred Value        1.0000  0.9600  0.9800
Precision             1.0000  0.9600  0.9200
Recall                1.0000  0.9231  0.9583
F1                    1.0000  0.9412  0.9388
Prevalence            0.3333  0.3467  0.3200
Detection Rate        0.3333  0.3200  0.3067
Detection Prevalence  0.3333  0.3333  0.3333
Balanced Accuracy     1.0000  0.9513  0.9596
```

Output A11: Clustering Performance of the BPW K–Means Algorithm on the Sirtuin6 Dataset---Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.1, Bi-Partition Number = 2


  Low BFE High BFE
1  41       9
2   9      41

 Overall Statistics

               Accuracy: 0.82
                 95% CI: (0.7305, 0.8897)
    No Information Rate: 0.5
    P-Value [Acc > NIR]: 3.074e-11

                  Kappa: 0.64

 McNemar's Test P-Value: 1

            Sensitivity: 0.82
            Specificity: 0.82
         Pos Pred Value: 0.82
         Neg Pred Value: 0.82
              Precision: 0.82
                 Recall: 0.82
                     F1: 0.82
             Prevalence: 0.50
         Detection Rate: 0.41
   Detection Prevalence: 0.50
      Balanced Accuracy: 0.82
```

Note: The clustering performance of bi-partition numbers 2, 3, and 4 was identical for unranked features on the Sirtuin6 dataset. Therefore, we provide only one output of the optimal pair of solutions.

Output A12: Clustering Performance of the Classical K–Means Algorithm on the Sirtuin6 Dataset–––Ranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = NA, Bi-Partition Number = NA


  Low BFE High BFE
1  44       6
2  18  32

 Overall Statistics

        Accuracy: 0.76
         95% CI: (0.6643, 0.8398)
  No Information Rate: 0.62
  P-Value [Acc > NIR]: 0.002122

         Kappa: 0.52

 McNemar's Test P-Value: 0.024745

      Sensitivity: 0.7097
      Specificity: 0.8421
    Pos Pred Value: 0.8800
    Neg Pred Value: 0.6400
       Precision: 0.8800
          Recall: 0.7097
              F1: 0.7857
      Prevalence: 0.6200
    Detection Rate: 0.4400
  Detection Prevalence: 0.5000
   Balanced Accuracy: 0.7759
```

Note: The clustering performance of the classical K-means algorithm was identical for both ranked and unranked features on the Sirtuin6 dataset. Therefore, we provide the output solely for the ranked features.

Output A13: Clustering Performance of the BPW K–Means Algorithm on the Wholesale Customers Dataset–––Unranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = 0.6, Bi-Partition Number = 2
```

```
  Horeca    Retail
1 293       5
2 72  70

Overall Statistics

        Accuracy: 0.825
          95% CI: (0.7862, 0.8594)
 No Information Rate: 0.8295
 P-Value [Acc > NIR]: 0.6291

          Kappa: 0.5433

McNemar's Test P-Value: 5.419e-14

      Sensitivity: 0.8027
      Specificity: 0.9333
    Pos Pred Value: 0.9832
    Neg Pred Value: 0.4930
       Precision: 0.9832
          Recall: 0.8027
            F1: 0.8839
       Prevalence: 0.8295
    Detection Rate: 0.6659
  Detection Prevalence: 0.6773
   Balanced Accuracy: 0.8680
```

Output A14: Clustering Performance of the Classical K-Means Algorithm on the Wholesale Customers Dataset---Ranked Features

```
Confusion Matrix and Statistics

Parameter Values Used:
Beta = NA, Bi-Partition Number = NA


  Horeca    Retail
1 247       51
2 128  14

Overall Statistics

        Accuracy: 0.5932
          95% CI: (0.5456, 0.6395)
 No Information Rate: 0.8523
 P-Value [Acc > NIR]: 1

          Kappa: -0.0845
```

```
McNemar's Test P-Value: 1.343e-08

          Sensitivity: 0.65867
          Specificity: 0.21538
       Pos Pred Value: 0.82886
       Neg Pred Value: 0.09859
            Precision: 0.82886
               Recall: 0.65867
                   F1: 0.73403
           Prevalence: 0.85227
       Detection Rate: 0.56136
 Detection Prevalence: 0.67727
    Balanced Accuracy: 0.43703
```

Note: The clustering performance of the classical K-means algorithm was identical for both ranked and unranked features on the Wholesale Customers dataset. Therefore, we provide the output solely for the ranked features.

References

1. McCaffery, K. Financial Services Regulatory Authority of Ontario. Automobile Insurance. 2023. Available online: https://insurance-portal.ca/article/large-number-of-ontario-drivers-believe-auto-insurance-fraud-is-prevalent/ (accessed on 10 November 2023).
2. Lekha, K.C.; Prakasam, S. Data mining techniques in detecting and predicting cyber crimes in banking sector. In Proceedings of the 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), Chennai, India, 1–2 August 2017; pp. 1639–1643.
3. Nassar, O.A.; Al Saiyd, N.A. The integrating between web usage mining and data mining techniques. In Proceedings of the 2013 5th International Conference on Computer Science and Information Technology, Amman, Jordan, 27–28 March 2013; pp. 243–247.
4. Kowshalya, G.; Nandhini, M. Predicting fraudulent claims in automobile insurance. In Proceedings of the 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, India, 20–21 April 2018; pp. 1338–1343.
5. Patel, D.K.; Subudhi, S. Application of extreme learning machine in detecting auto insurance fraud. In Proceedings of the 2019 International Conference on Applied Machine Learning (ICAML), Bhubaneswar, India, 25–26 May 2019; pp. 78–81.
6. Óskarsdóttir, M.; Ahmed, W.; Antonio, K.; Baesens, B.; Dendievel, R.; Donas, T.; Reynkens, T. Social network analytics for supervised fraud detection in insurance. *Risk Anal.* **2022**, *42*, 1872–1890. [CrossRef] [PubMed]
7. Bodaghi, A.; Teimourpour, B. Automobile insurance fraud detection using social network analysis. In *Applications of Data Management and Analysis: Case Studies in Social Networks and Beyond*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 11–16.
8. Ho, T.K. Random decision forests. In Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, Canada, 14–16 August 1995; Volume 1, pp. 278–282.
9. Kaggle Dataset. Vehicle Insurance Claim Fraud Detection. 2021. Available online: https://www.kaggle.com/datasets/aashishjhamtani/automobile-insurance (accessed on 17 May 2023).
10. Kelly, M.; Longjohn, R.; Nottingham, K. The UCI Machine Learning Repository. 2023. Available online: https://archive.ics.uci.edu (accessed on 18 December 2024).
11. Nian, K.; Zhang, H.; Tayal, A.; Coleman, T.; Li, Y. Auto insurance fraud detection using unsupervised spectral ranking for anomaly. *J. Financ. Data Sci.* **2016**, *2*, 58–75. [CrossRef]
12. Yang, J.; Chen, K.; Ding, K.; Na, C.; Wang, M. Auto insurance fraud detection with multimodal learning. *Data Intell.* **2023**, *5*, 388–412. [CrossRef]
13. Ming, R.; Abdelrahman, O.; Innab, N.; Ibrahim, M.H.K. Enhancing fraud detection in auto insurance and credit card transactions: A novel approach integrating CNNs and machine learning algorithms. *PeerJ Comput. Sci.* **2024**, *10*, e2088. [CrossRef]
14. Wongpanti, R.; Vittayakorn, S. Enhancing Auto Insurance Fraud Detection Using Convolutional Neural Networks. In Proceedings of the 2024 21st International Joint Conference on Computer Science and Software Engineering (JCSSE), Phuket, Thailand, 19–22 June 2024; pp. 294–301.
15. Nti, I.K.; Adu, K.; Nimbe, P.; Nyarko-Boateng, O.; Adekoya, A.F.; Appiahene, P. Robust and resourceful automobile insurance fraud detection with multi-stacked LSTM network and adaptive synthetic oversampling. *Int. J. Appl. Decis. Sci.* **2024**, *17*, 230–249. [CrossRef]
16. Wei, S.; Lee, S. Financial anti-fraud based on dual-channel graph attention network. *J. Theor. Appl. Electron. Commer. Res.* **2024**, *19*, 297–314. [CrossRef]

17. Van Driel, H. Financial fraud, scandals, and regulation: A conceptual framework and literature review. In *Business History*; Taylor and Francis Group: Abingdon, UK, 2019.
18. Schrijver, G.; Sarmah, D.K.; El-Hajj, M. Automobile Insurance Fraud Detection Using Data Mining: A Systematic Literature Review. In *Intelligent Systems with Applications*; Elsevier: Amsterdam, The Netherlands, 2024; p. 200340.
19. Government of Ontario. Ontario Automobile Insurance Anti-Fraud Task Force: Groupe de Travail Antifraude de L'Assurance-Automobile de L'Ontario, Canadian Electronic Library. Canada. Business History. 2012. Available online: https://canadacommons.ca/artifacts/1201133/ontario-automobile-insurance-anti-fraud-task-force/1754253/ (accessed on 8 August 2024).
20. Nobel, S.N.; Sultana, S.; Singha, S.P.; Chaki, S.; Mahi, M.J.N.; Jan, T.; Barros, A.; Whaiduzzaman, M. Unmasking Banking Fraud: Unleashing the Power of Machine Learning and Explainable AI (XAI) on Imbalanced Data. *Information* **2024**, *15*, 298. [CrossRef]
21. Urunkar, A.; Khot, A.; Bhat, R.; Mudegol, N. Fraud Detection and Analysis for Insurance Claim using Machine Learning. In Proceedings of the 2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), Phuket, Thailand, 19–22 June 2022; Volume 1, pp. 406–411.
22. Soua, M.; Kachouri, R.; Akil, M. Improved Hybrid Binarization based on Kmeans for Heterogeneous document processing. In Proceedings of the 2015 9th International Symposium on Image and Signal Processing and Analysis (ISPA), Zagreb, Croatia, 7–9 September 2015; pp. 210–215.
23. Thiprungsri, S.; Vasarhelyi, M.A. Cluster Analysis for Anomaly Detection in Accounting Data: An Audit Approach. *Int. J. Digit. Account. Res.* **2011**, *11*, 69–84. [CrossRef] [PubMed]
24. Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *London Edinb. Dublin Philos. Mag. J. Sci.* **1901**, *2*, 559–572. [CrossRef]
25. Hotelling, H. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* **1933**, *24*, 417. [CrossRef]
26. Comon, P. Independent component analysis, a new concept? *Signal Process.* **1994**, *36*, 287–314. [CrossRef]
27. Hyvärinen, A.; Oja, E. Independent component analysis: Algorithms and applications. *Neural Netw.* **2000**, *13*, 411–430. [CrossRef] [PubMed]
28. He, X.; Cai, D.; Niyogi, P. Laplacian score for feature selection. *Adv. Neural Inf. Process. Syst.* **2005**, *18*, 1–8.
29. Peng, H.; Long, F.; Ding, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 1226–1238. [CrossRef] [PubMed]
30. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
31. Kira, K.; Rendell, L.A. The feature selection problem: Traditional methods and a new algorithm. In Proceedings of the 10th National Conference on Artificial Intelligence, San Jose, CA, USA, 12–16 July 1992; pp. 129–134.
32. Kononenko, I. Estimating attributes: Analysis and extensions of RELIEF. In Proceedings of the European Conference on Machine Learning, Catania, Italy, 6–8 April 1994; pp. 171–182.
33. Robnik-Šikonja, M.; Kononenko, I. Theoretical and empirical analysis of ReliefF and RReliefF. *Mach. Learn.* **2003**, *53*, 23–69. [CrossRef]
34. Hartigan, J.A.; Wong, M.A. Algorithm AS 136: A k-means clustering algorithm. *J. R. Stat. Society Ser. C (Appl. Stat.)* **1979**, *28*, 100–108. [CrossRef]
35. Lloyd, S. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137. [CrossRef]
36. MacQueen, J. Some methods for classification and analysis of multivariate observations. *Berkeley Symp. Math. Statist. Prob.* **1967**, *1*, 281–297
37. Wang, S.; Sun, Y.; Bao, Z. On the efficiency of k-means clustering: Evaluation, optimization, and algorithm selection. *arXiv* **2020**, arXiv:2010.06654. [CrossRef]
38. D'urso, P.; Massari, R. Fuzzy clustering of mixed data. *Inf. Sci.* **2019**, *505*, 513–534. [CrossRef]
39. Qian, Y.; Yao, S.; Wu, T.; Huang, Y.; Zeng, L. Improved Selective Deep-Learning-Based Clustering Ensemble. *Appl. Sci.* **2024**, *14*, 719. [CrossRef]
40. Gan, L.; Allen, G.I. Fast and interpretable consensus clustering via minipatch learning. *PLoS Comput. Biol.* **2022**, *18*, e1010577. [CrossRef] [PubMed]
41. Rand, W.M. Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850. [CrossRef]
42. Hubert, L.; Arabie, P. Comparing partitions. *J. Classif.* **1985**, *2*, 193–218. [CrossRef]
43. Fisher, R.A. Iris. UCI Machine Learning Repository. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume II, Part II: Probability Theory*; University of California Press: Berkeley, CA, USA, 1936. [CrossRef]
44. Tardu, M.; Rahim, F. Sirtuin6 Small Molecules. UCI Machine Learning Repository. *RAIRO Oper. Res.* **2016**. [CrossRef]
45. Cardoso, M. Wholesale Customers. UCI Machine Learning Repository. 2013. Available online: https://archive.ics.uci.edu/dataset/292/wholesale+customers (accessed on 1 January 2025). [CrossRef]