2528-9705

Journal of Organizational Behavior Research

Vol.: 10, Is.: S1, Year: 2025, ID: 25S0-871

Available online
https://odad.org/

A Proposed Approach for Managing Noise and Uncertainty in Big Data

Mohsen Hosseinalizadeh[1]*

[1] Department of Computer Science, Faculty of Engineering, Azad Islamic University, Hamedan, Iran.

***Corresponding Author**
E-mail: mohsen.h.alizadeh@outlook.com

ABSTRACT

Sorting and clustering in big data increase search speed because searching within large datasets is both critical and time-consuming. Extensive research has been conducted in recent years on detecting outliers by employing classification through engineering methods such as artificial intelligence and machine learning. One of the significant challenges in applying these methods is the presence of noise in the data. Considering that big data exhibits imbalanced distributions across its classes, it is possible to train these data by creating transformed datasets and perform the necessary classification based on the specific problem. In this study, data clustering is initially performed using the K-means method and hierarchical analysis. After clustering, a hybrid approach composed of neural networks, decision trees, and boosting is applied to manage noise in the dataset. Subsequently, the proposed approach is compared with two other methods, including structural feature methods and data mining techniques such as support vector machines, based on criteria like accuracy, precision, and recall. The results demonstrate an improvement of 2 to 4 percent across all metrics for our proposed method.

Keywords: Clustering, Noise, Neural network, Decision tree, Accuracy

Introduction

The digital and communication technology revolution generates an enormous volume of data. Consequently, the nature of classical data shifts to big data, and extraction techniques must confront challenges related to computational costs, performance, and high scalability. The most common sorting methods are numerical orders and dictionaries. Efficient sorting is crucial in the development of algorithms such as search and join that require sorted lists (Pandey *et al*., 2022).

Sorting and clustering in big data increase search speed since searching in such datasets is both important and time-consuming. The most common sorting methods include numerical order and dictionaries. Efficient sorting plays a vital role in developing algorithms that rely on sorted lists, such as search and join. From the inception of computer science, sequencing problems have attracted considerable research interest because, despite their apparent simplicity, solving them is practically complex (Ghosal *et al*., 2020).

The goal of clustering is to extract models from data. Current research domains in this field, including numerous applications of fuzzy and hard clustering in data analysis and pattern recognition, as well as their use in solving routing, allocation, and scheduling problems, indicate the need to review, improve, and refine existing algorithms. Data clustering is one of the important topics in data mining and machine learning (Figueiredo *et al*., 2019). Clustering refers to grouping similar objects together such that similar objects belong to one cluster and dissimilar objects belong to different clusters. Data clustering is based on similarity and difference indices between data points, which make the cluster analysis meaningful.

The biggest challenge in clustering is the lack of information about known data groups. Moreover, the selection of input parameters such as the number of clusters, number of nearest neighbors, and other algorithms makes clustering a contentious topic. Therefore, incorrect selection of any of these parameters results in undesirable clustering

outcomes. Sorting using processors increases computational speed. Research on processors and memory behavior indicates that significant time is wasted in databases and application searches. Most algorithms introduced for performance improvement are based on multiprocessor parallelism, including parallel data algorithms, buffer training algorithms, better data storage models, and efficient data generation algorithms. Considering that big data exhibits imbalanced distributions among its classes, it is possible to train transformed datasets and perform the necessary classification according to the specific problem. Studies also show that most fundamental classifiers perform better on balanced data. These methods employ a set of mechanisms to manipulate imbalanced datasets to balance the data (Mittal *et al*., 2019).

Clustering with specific algorithms requires prior knowledge, which can enhance its importance. This prior understanding aids in selecting the best algorithm, determining the number of clusters, and interpreting results. Furthermore, clustering's significance lies in recognizing different patterns within data, analyzing them, and improving data-driven decision-making across various domains. Developing better patterns for data access is a necessity of this research because sorting large volumes of data facilitates faster searching, and high-speed access to big data is crucial. **Table 1** presents a comparison of previous studies that have addressed big data clustering using different algorithms.

Table 1. Comparison of Previous Approaches

Methodology	Advantages	Limitations	Citation
Perceptron & Regression	Real-time and online model application	High operational accuracy on the data.	(Chandra & Sharma, 2014)
Genetic Algorithm	Reduction of processing cost and time	Requires testing under real-world conditions and redefining thresholds for greater accuracy in predicting fraudulent records	(Ajunwa, 2017)
Multilayer Perceptron Neural Network	The proposed model has been tested on three datasets	The neural network functions as a black box, making the internal algorithm analysis incomprehensible	(Ramchoun *et al*, 2017)
Cost Minimization	Unlike other methods that detect fraud at the time of occurrence, this approach identifies fraud during account creation	Although the model is capable of predicting fraud, expert verification is still required to confirm with confidence	(Fan *et al*, 2018)
Decision Tree and Genetic Algorithm	High accuracy and high classification cost	Only capable of detecting fraud in personal-use applications	(Hadi & Al-Furat, 2018)
Cost-Sensitive Decision Tree	10% performance improvement. Unknown signal error detection	If a fraudulent record is detected, expert intervention is always required to ensure confirmation	(Suthaharan & Suthaharan, 2016)
Neural Network with Variable Threshold	Ability to detect fraud in imbalanced datasets	Due to threshold setting, some customers who fall outside the defined range may be mistakenly identified as fraudulent	(Jung *et al*, 2016)
Profit-Sensitive Neural Network	Modeling with a supervised approach has proven to be more effective than an unsupervised one	Most of the time is spent on feature extraction, leaving insufficient time for model development.	(Scardapane *et al*, 2016)

		It has only been tested on available data and requires further testing in future research	
Structural Properties	Customization, Interpretability, and Integration of Domain Knowledge	Assumption of known structure, overfitting, limited applicability, and risk of information loss	(Amin *et al*, 2017)
Support Vector Machine (SVM)	Effective in class separation, flexible in kernel selection	Computational complexity, sensitivity to imbalanced data, limited to binary classification	(Quellec *et al*, 2016)
Decision Tree and Boosting	Suitable accuracy and improved generalization	Requires better preprocessing and presents challenges in hyperparameter optimization	(Feng *et al*, 2018)

A review of related research on anomaly management in big data shows that the goal of detection and management is to maximize correct predictions while maintaining false predictions at the same level. Given the continual changes in factors such as data generation speed, increasing data volume, data storage methods, organizational policies regarding big data, and the behavior of customers and fraudsters, previous models based on transaction and data analysis may fail to identify new forms of uncertainty in data and fraud in transactions.

None of the existing fraud and data anomaly detection systems alone can identify and cover all forms of fraud and anomalies; therefore, a hybrid approach combining noise and uncertainty detection models can enhance fraud detection and system security. Since the effectiveness of these models in fraud detection is critical, attention must be given to the imbalanced ratio between legitimate and fraudulent transactions, and appropriate solutions must be devised.

Based on the comparison presented in **Table 1**, it can be stated that in the area of clustering and sorting data, an algorithm capable of classifying big data without the need for prior knowledge is desirable. Not using prior knowledge implies employing a predefined framework, which in this study is defined according to user requirements for list sorting. In all previous research, this prior knowledge was required and utilized. However, the proposed algorithm in this research offers an innovative solution for data types lacking prior knowledge and manages noise and uncertainty in big data. Therefore, this study aims to propose a method, by examining machine learning solutions and combining nearest neighbor methods with clustering, that can be applied for clustering and noise reduction in big data.

Materials and Methods

As shown in **Figure 1**, the proposed method is divided into several distinct sections, each complementing the subsequent one. The stages of this approach are carried out sequentially, and the details of each step are presented separately. The data used in this study are obtained from the University of California, Irvine (UCI) website and can be downloaded from the following address:

http://archive.ics.uci.edu/ml/datasets/Bank+Marketing

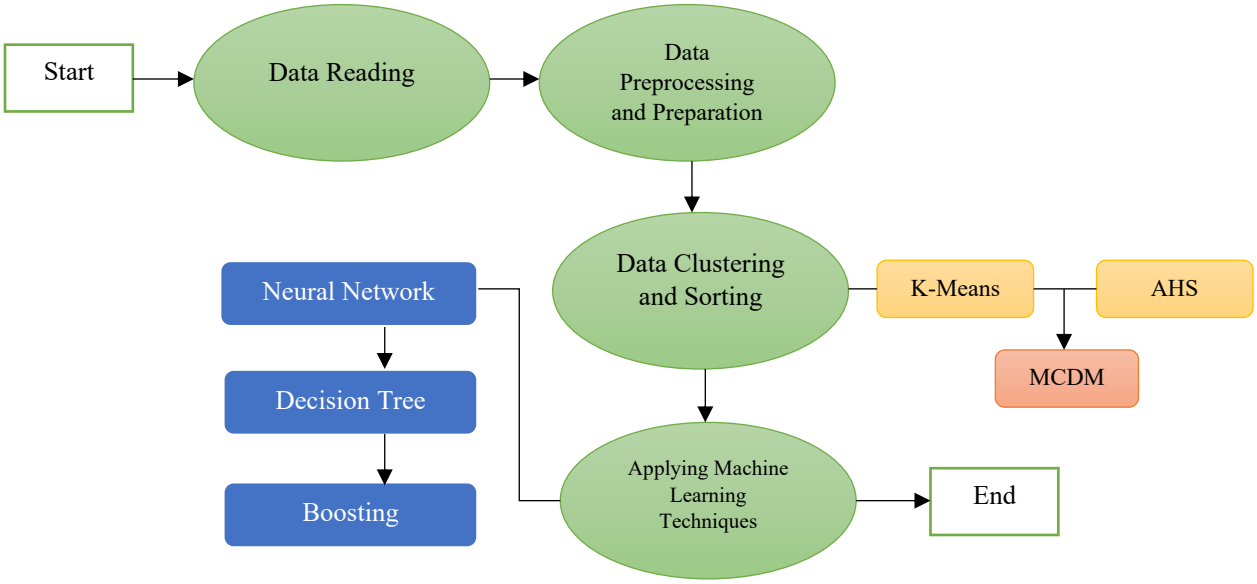

Figure 1. Steps of Proposed Method

The statistical population of this study consists of a dataset of bank customers compiled by Moro *et al*. (2014) using a data segmentation approach aimed at predicting the success of telephone banking campaigns, and applicable in decision support systems. The selected dataset contains over 45,000 records with 17 attributes. In this research, this volume of information is utilized, and a random subset of the data is selected. It is necessary to normalize these data prior to analysis.

Data Clustering and Analytic Hierarchy Process (AHP)

The purpose of clustering is to reveal order and patterns by partitioning data into similar groups. The input to a clustering analysis system is a set of sample data along with a criterion for measuring similarity (or dissimilarity) between two samples, while the output is a collection of subsets that represent all or part of the dataset. For data clustering, the k-means clustering algorithm is employed (Blömer *et al*., 2016). Another recommended approach to address the challenges arising from the presence of mixed-type data during clustering is the definition of an appropriate distance metric.

If two data points X= {x_1, x_2,…, x_n} and Y= {y_1, y_2,…, y_n} are considered, the distance metric defined for these two data points—an extension of the Euclidean relation—is expressed as Equation (1).

$$d(X,Y) = \alpha \,.\, distance\big(num(X), num(Y)\big) + \beta . diff(cat(X), cat(Y)) \quad (1)$$

In Equation (1), *distance(num(.))* computes the Euclidean distance between two data points with numerical features, i.e., the data obtained through the function *num(.)*. In fact, the function *num(.)* retains only the numerical attributes of a given data point, which can be represented as Equation (2).

$$num(X) = \{x_i \epsilon feature_{set}(X) | x_i \; is \; numerical \; ; i \; \epsilon [1, |X|\} \quad (2)$$

Similarly, in Equation (1), the function *cat(.)* retains only the categorical attributes of a given data point, which can be defined as Equation (3).

$$cat(X) = \{x_i \epsilon feature_{set}(X) | x_i \; is \; categorical \; ; i \; \epsilon [1 | X|\} \quad (3)$$

Consequently, the function diff(X) computes the distance between categorical attributes. This calculation is performed by counting the number of identical features with different values and dividing the result by the total number of such features. n Equation (6), the parameters α, β∈ [0,1] specify the weight of the numerical and categorical components, respectively, in determining the overall distance between data points (Blömer *et al*., 2016).

After assigning the data to different clusters, the next stage of the proposed system begins. During this stage, unknown or newly arriving data are introduced into the system. The distance of these data points is measured according to each criterion, based on the distance metric previously described. The closest cluster is then identified. Next, the distance

between the new data point and the data points within the identified cluster's center is calculated. Based on these distances, 2n+1 data points are selected as the ones most similar to the unknown instance.

Following the K-means clustering process, the Analytic Hierarchy Process (AHP) is applied to perform multi-criteria decision-making, integrating the clustering results with the classification of evaluation criteria. After determining the closest cluster and focusing on the data points most similar to the new instance, an appropriate decision is made for that instance. The rationale for considering the closest data points as "voters" lies in aggregating their votes to reach a decision. Moreover, selecting more than one similar data point helps reduce the impact of noise or outliers on the final decision. However, stating that a single data point has cast a vote in a multi-criteria decision-making system does not necessarily imply that the recommendation for that data point should be directly adopted. Instead, the process operates as follows: if a decision *d* has been made for data point l_k, a multi-criteria decision-making rule in the form of an AHP decision rule is generated, and its robustness is evaluated by estimating associated probabilities.

These probabilities are not computed over the entire dataset; rather, they are calculated only from the data within the same subset. The probability for the clustering component under rule R=X → is defined as shown in Equation (4) (Fahad & Alam, 2016).

$$P_{Kmeans}(R) = \frac{P(XY)}{|D|} \tag{4}$$

And the probabilities for the k-means clustering component combined with the multi-criteria decision-making Analytic Hierarchy Process are expressed as Equation (5).

$$P_{AssociationRulesKmeans}(R) = \frac{P(XY)}{P(X)} \tag{5}$$

In the above formulas, *P(.)* represents the number of data points in the entire set *D* where both *X* and *Y* occur. n our proposed method, *D* corresponds to the cluster from which the nearest data points have been selected. However, an important aspect that has not yet been addressed regarding these rules is the method of their generation and the calculation of probabilities based on the type of data involved, which will be discussed in the following sections (Blömer *et al*., 2016).

As previously mentioned, for each data point lk in the set of nearest neighbors, a rule in the form of a multi-criteria decision-making Analytic Hierarchy Process (AHP) is generated. If the attributes of l_k are divided into two parts—features and decisions—then the entire feature set can be represented as $I_k = [f_i \ldots f_n, d_i \ldots d_m]$. In this representation, f_i denotes features such as connection type, while d_i indicates the decision made, taking values of either *0* or *1*. It is clear that for each data point, only one decision is made, so among d_i to d_m, only one can be equal to *1*. For these data points, all rules are generated according to Equation (6) (Blömer *et al*., 2016).

$$\begin{aligned} &R_1: f_1, \ldots, f_n \rightarrow S_i \\ &R_2: f_1, \ldots, f_n \rightarrow S_i \\ &. \\ &R_{2n+1}: f_1, \ldots, f_n \rightarrow S_i \\ &\quad i \in [1, m] \end{aligned} \tag{6}$$

As can be understood from Equation (6), decisions may be made for more than one data point. Each rule is evaluated by calculating probability values. In fact, the formulas provided for probability calculation are suitable for categorical data, where numerical data are transformed into a categorical problem.

Neural Network

Considering the dataset, the number of input parameters is 17, and the results of fraudulent and non-fraudulent operations, which are the dependent variables of the problem, are available. The data consist of 9 datasets; each divided into training and testing subsets. The dataset is provided in csv format and must first be converted into an appropriate format for MATLAB. All fields are represented numerically, and for each example, the fraud status is indicated by two characters: s (fraudulent) and n (non-fraudulent). Therefore, the input data should be transformed into a 17-element vector, with corresponding outputs assigned as either *n* or *s*.

The neural network consists of 12 neurons in the first hidden layer, 6 neurons in the second hidden layer, and 2 neurons in the output layer. The activation functions in both the hidden and output layers are sigmoid functions. Specifically, the network has two hidden layers with 12 and 6 neurons, respectively, and sigmoid logarithmic activation functions

are used throughout all layers. The network initialization commands involve random weights, with the number of training epochs set to 1000, a learning rate of 0.4, and a momentum of 0.7. **Figure 2** illustrates the overall architecture of the proposed neural network. For training, the dataset was applied multiple times to the training algorithm. The test results for the 9 datasets are presented in the Results section.

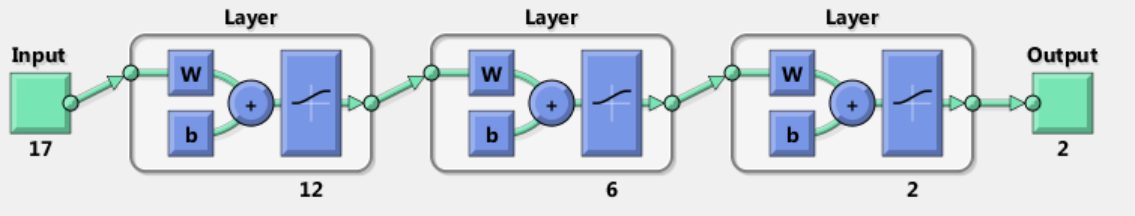

Figure 2. Overall structure of the proposed neural network

Decision Tree

There are two main types of decision trees, classification trees and regression trees. Classification trees are used when the predicted outcome is a class to which the data belong, and they are applied to categorize data into distinct groups. Regression trees, on the other hand, are used when the predicted outcome can be considered a real number, such as the price of a house or the length of a patient's stay in a hospital. In the proposed method, a classification decision tree is employed, which is implemented using the *fitctree* function.

Boosting

While boosting is not itself a single algorithm, most boosting-based algorithms—such as AdaBoost and Gradient Boosting—iteratively train weak learners and incrementally add them to the existing ensemble to form a strong classifier. Weak learners added to the ensemble are typically weighted based on their classification performance on the samples.

After each learner is added, the samples (data points) are also re-weighted. This re-weighting process decreases the weights of correctly classified samples and increases the weights of misclassified samples, ensuring that in subsequent iterations, new weak learners focus more on the data points that the system failed to classify correctly in earlier stages. As a result, the newly added weak learners become more relevant and improve classification accuracy. Thus, in each iteration, the new weak learners place greater emphasis on difficult-to-classify samples, enhancing the model's overall predictive performance (Ma *et al*., 2019).

To implement boosting, the AdaBoost algorithm is employed. This algorithm constructs an AdaBoost classifier and trains the weak learners using the results of the decision tree model. The training process is executed in the simulation environment through the *fitensemble* function, which iteratively combines the weak learners to form a strong and accurate classifier.

Comparison of Different Parameters in Fraud Detection Systems

For validating classification methods, various evaluation metrics are available, each assessing the effectiveness of the methods from a particular perspective. However, given the specific characteristics of this domain, additional standards are required to account for these unique aspects. When the system labels an incoming transaction, one of the following four scenarios occurs:

- True Positive (TP): The incoming transaction is fraudulent, and the system correctly identifies it as fraud.
- False Positive (FP): The incoming transaction is legitimate, but the system incorrectly classifies it as fraud.
- True Negative (TN): The incoming transaction is legitimate, and the system correctly classifies it as legitimate.
- False Negative (FN): The incoming transaction is fraudulent, but the system incorrectly classifies it as legitimate.

In optimal conditions, the system should minimize the number of FN cases while maximizing the number of TP cases. Generally, transactions that are incorrectly classified as legitimate are far more dangerous than those incorrectly flagged as fraudulent. In the first case, the organization may incur significant financial losses depending on the

transaction amount. In the second case, however, the primary consequences are the additional cost of reviewing the transaction and potential customer dissatisfaction if their card is temporarily blocked (Kyriienko & Magnusson, 2022). The metrics used in this study to evaluate the performance of the proposed method include accuracy, precision, recall (sensitivity), F-measure, and cost. Equations (7) through (10) present the formulas for calculating these metrics.

$$Accuracy(A) = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \quad (7)$$

$$Precision(P) = \frac{TP}{FP + TP} \quad (8)$$

$$Recall(R) = \frac{TP}{FN + TP} \quad (9)$$

$$F - measure(F) = \frac{TP}{FN + TP} \quad (10)$$

The three specified metrics—accuracy, precision, and fraud detection rate—represent the percentage of correct predictions. However, none of these alone can fully determine the effectiveness of an algorithm. A higher accuracy does not necessarily imply better performance, as the cost of misclassifying positive and negative cases often differs. High accuracy may result in a low fraud detection rate, and conversely, a high fraud detection rate may not yield satisfactory results if it comes at the expense of precision. This indicates the need to maintain a balance between the TP rate and the FN rate.

Results and Discussion

Model Training with Default Parameters: Using default parameters, the neural network method—without incorporating decision trees or boosting—was applied to the input datasets for model training, followed by the proposed method. The obtained results were compared at the end. This process is used to train the neural network to recognize patterns, with the objective of minimizing the difference between the predicted output and the actual output. Once trained, the network can be used to predict outcomes for new data. **Table 2** presents the output of the neural network method. As can be observed, the average detection accuracy of this method is 86%.

Table 2. Results after applying Neural Network method with default parameters

Dataset	TP	TN	FP	FN	Cost	Precision	Recall	Accuracy
1	3780	7358	1029	1235	137559	0.72	0.74	0.8661
2	3868	7365	1038	1228	136480	0.74	0.76	0.8658
3	3477	7361	1135	1434	139745	0.73	0.75	0.8619
4	3781	7357	1041	1231	137426	0.75	0.77	0.8642
5	3573	7463	1132	1330	134564	0.71	0.73	0.8631
6	3769	7364	1033	1236	139823	0.72	0.74	0.8663
7	3975	7259	1036	1129	137841	0.74	0.76	0.8701
8	3776	7362	1240	1233	137965	0.73	0.75	0.8646
9	3874	7260	1030	1237	137412	0.75	0.77	0.8684
		Average			137924	0.73	0.75	0.8651

Furthermore, the error rate of the neural network in its default configuration—used alone without decision trees or boosting—was tested with different learning rates. The results are presented below. As can be seen, the error rate in the default state is not uniform and exhibits low stability.

P=0.5

Layer 1: Neurons = 12, Min Error = 0.31569

Layer 2: Neurons = 6, Min Error = 0.30346

Layer 3: Neurons = 2, Min Error = 0.29546

P=0.6

Layer 1: Neurons = 12, Min Error = 0.1789

Layer 2: Neurons = 6, Min Error = 0.3353

Layer 3: Neurons = 2, Min Error = 0.4238
P=0.7
Layer 1: Neurons = 12, Min Error = 0.36042
Layer 2: Neurons = 6, Min Error = 0.4556
Layer 3: Neurons = 2, Min Error = 0.4426
P=0.8
Layer 1: Neurons = 12, Min Error = 0.4067
Layer 2: Neurons = 6, Min Error = 0.5052
Layer 3: Neurons = 2, Min Error = 0.5843
P=0.9
Layer 1: Neurons = 12, Min Error = 0.53667
Layer 2: Neurons = 6, Min Error = 0.49777
Layer 3: Neurons = 2, Min Error = 0.381012

This section presents four charts illustrating the values of cost, accuracy, false negatives (FN), and true positives (TP). In these charts, both methods are compared: the neural network without applying the proposed approach, and the neural network with the proposed approach implemented.

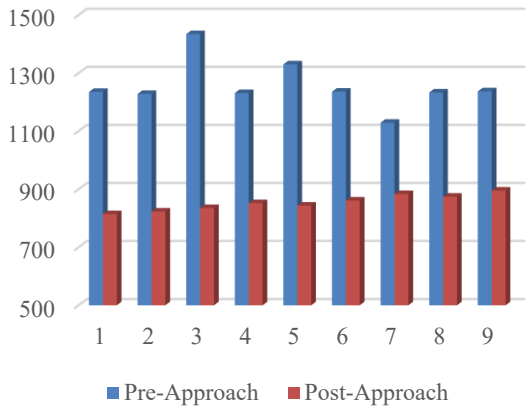

Figure 3. FN

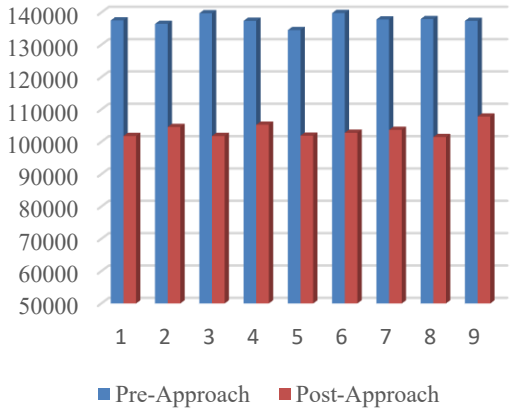

Figure 4. Cost

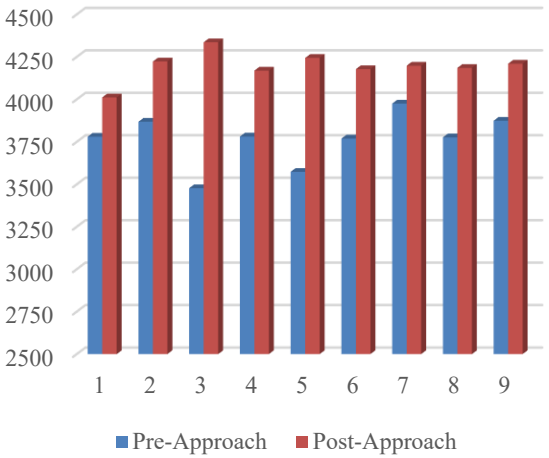

Figure 5. TP

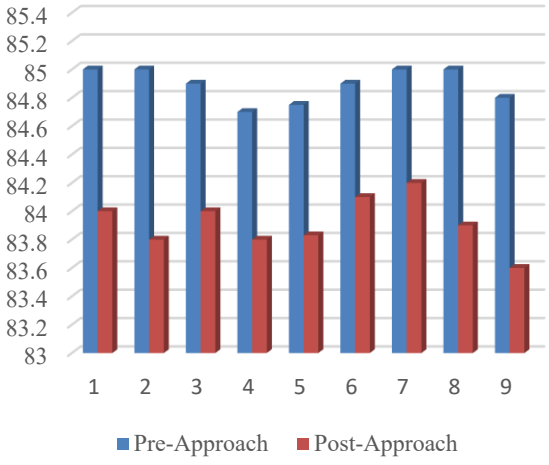

Figure 6. Accuracy

As observed in figures 3 to 6, in all charts—except for accuracy—the proposed method demonstrates superior performance. Regarding accuracy, it should be noted that the change is approximately one percent, and this adjustment was made to improve the cost metric. In other words, within the context of this study, accuracy was considered less important than cost.

Although the proposed method reduces the FN metric, the classifiers may not be fully optimized, and if accuracy is not sufficiently high, the FP metric may increase. While a higher FP rate may be of less concern to the organization, it can increase customer dissatisfaction. However, such dissatisfaction can be justified under the pretext of ensuring system security. One of the main objectives of this combined approach—minimizing the cost function—was successfully achieved.

Table 3 presents the complete values of the performance evaluation metrics in this study after applying the proposed method to the datasets. As shown, the average detection accuracy of the proposed method is 84%. It should be noted that the values are reported to two decimal places.

Table 3. Results after Applying the proposed approach

Dataset	TP	TN	FP	FN	Cost	Precision	Recall	Accuracy
1	4011	6384	1201	814	101789	0.78	0.86	0.8404
2	4223	6391	1214	823	104586	0.77	0.82	0.8373
3	4336	6405	1227	835	101788	0.69	0.84	0.8420
4	4169	6427	1243	852	105285	0.76	0.79	0.8381
5	4244	6312	1233	843	101890	0.81	0.87	0.8421
6	4178	6443	1254	861	102784	0.78	0.81	0.8386
7	4198	6468	1278	883	103691	0.72	0.85	0.8419
8	4185	6456	1261	874	101483	0.77	0.80	0.8535
9	4210	6488	1295	895	107792	0.79	0.88	0.8468
		Average			101698	0.77	0.83	0.8423

Results Comparison

To compare and evaluate the proposed method against existing approaches, we employed the technique introduced by Quellec *et al*. (2016), which leverages structural feature extraction and data mining methods—such as Support Vector Machines (SVM)—to effectively handle noise in large-scale datasets. For comparison, the accuracy test was repeated multiple times using different dataset sizes, and the accuracy was calculated separately for each run. The improvement ratio of the accuracy metric over other approaches was then computed and used as the accuracy measure of the proposed algorithm. The results of comparing the three methods are shown in **Figure 7**.

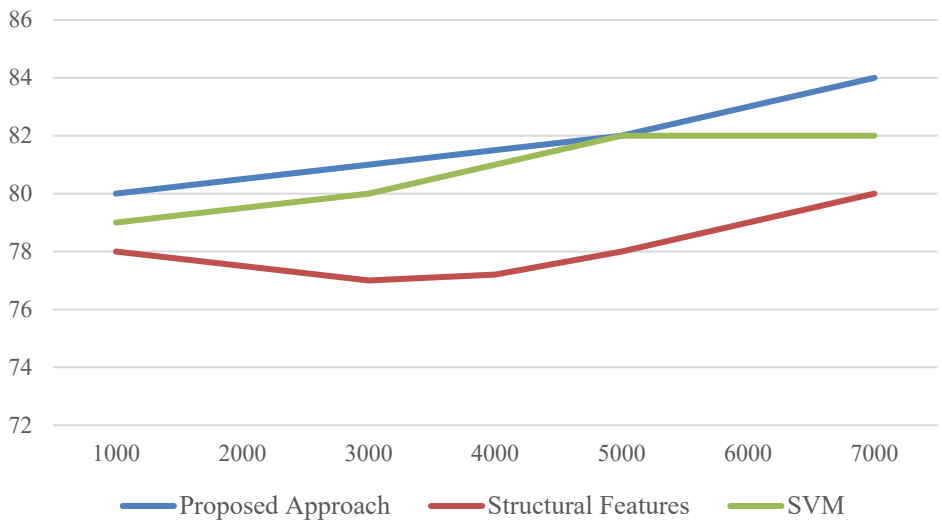

Figure 7. Detection and classification accuracy

In the proposed method, the noise management accuracy for 7,000 data instances is approximately 84.0%, and when this value is compared with that obtained from other methods, an improvement is observed.

On the other hand, the presence of high-quality data, proper preprocessing, and an appropriate data mining method yields favorable results in this context. The more accurate the input data, the more accurate the output will be; naturally, incorrect input will produce incorrect output. Having better information about the data enables the incorporation of correct data into the model structure, thereby producing correct results. According to this metric, the proposed method outperformed the two previous methods in managing noise in large datasets. The average F-measure of each method is shown in **Figure 8**.

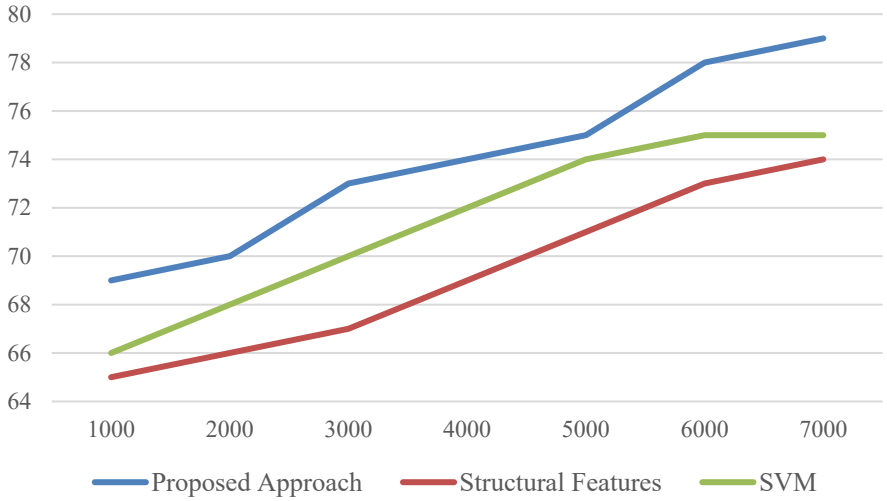

Figure 8. Comparing f-measure values for each method

The specificity metric refers to the proportion of negative instances that the test correctly identifies as negative. Mathematically, specificity is calculated as the ratio of true negatives to the sum of true negatives and false positives. The simulation results based on the aforementioned metrics are presented in **Figure 9**.

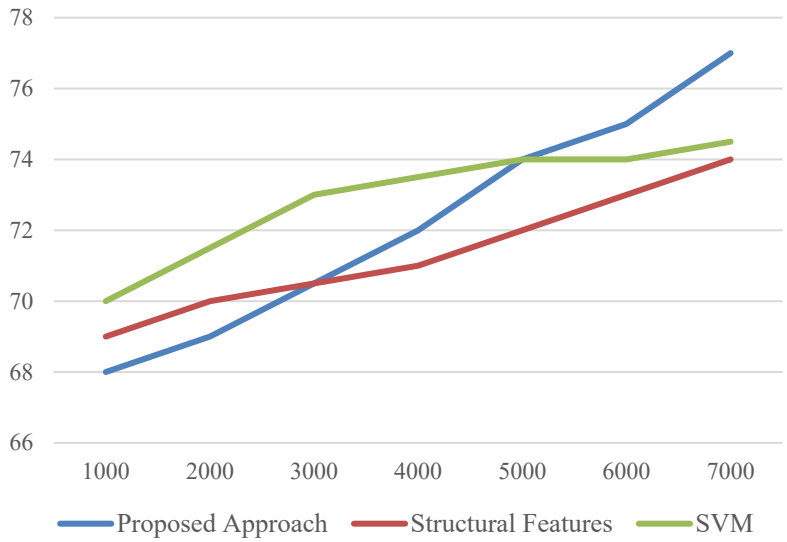

Figure 9. Comparing Precision of each method

In the first two simulation stages, the proposed method performs weaker than the other two methods; however, as the data volume increases, it demonstrates strong performance in detection and classification within big data environments. According to the chart, the proposed method achieves approximately 77% effectiveness in terms of specificity. Its main limitation lies in handling datasets with fewer than 3,000 records, where its specificity is lower compared to the other two methods.

The next criterion used for comparing results in this study is sensitivity. In other words, sensitivity is calculated as the ratio of true positives to the sum of true positives and false negatives. Some cases are presented in brief. The simulation results for this criterion are illustrated in **Figure 10**.

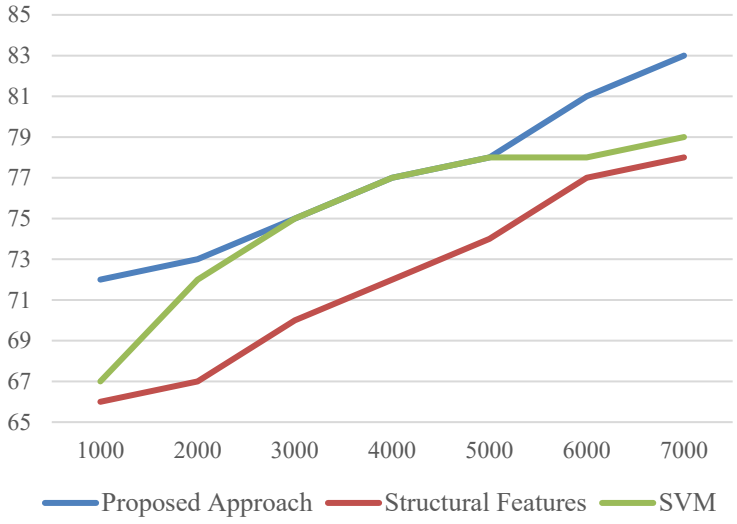

Figure 10. Comparing Recall value of each method

In **Figure 10**, for transaction counts ranging from 1,000 to 7,000, the proposed method achieved sensitivities of 72%, 75%, 79%, and 83%, respectively. Based on these results, the proposed method outperformed the two comparable approaches. Moreover, as the volume of big data and the number of inputs increased, the proposed method demonstrated high sensitivity in detecting and classifying its patterns and indicators within large-scale datasets.

Conclusion

In this study, clustering techniques and well-known supervised algorithms such as neural networks were employed to improve results. While neural networks require longer training and model construction times, their testing phase is comparatively faster. The imbalance of data in neural networks significantly affects the learning process and the accuracy of results, with smaller data samples typically leading to lower learning rates.

We proposed a learning method combining neural networks, decision trees, and boosted boosting algorithms that enables faster and more cost-effective detection of fraud and noise. The results demonstrate that this hybrid approach reduces the cost function in managing anomalies within big data by more than 30,000 units compared to similar methods. Additionally, the proposed method improved accuracy, specificity, and sensitivity by 2%, 3%, and 4%, respectively, compared to structural feature-based methods and data mining techniques such as Support Vector Machines. Based on the findings from integrating neural networks with decision trees and boosting, it is recommended that other classification and clustering methods be evaluated on big data volumes exceeding 100 gigabytes in terms of cost and accuracy, with their parameters thoroughly assessed. In the boosting phase, implementing gradient boosting algorithms could be considered. Given the diversity of neural network models, using artificial neural networks may yield different outcomes.

As mentioned previously, an effective noise management system must be fast, accurate, and exhibit low false positive rates. This study highlights the importance of noise detection and uncertainty management algorithms in big data for better identification of patterns and useful information from raw data. The combined approach proves to be highly effective and demonstrates the potential to achieve optimal performance by integrating diverse techniques. Through continuous implementation and refinement of such approaches, big data management systems can provide more reliable and secure services to organizations and financial institutions.

Acknowledgments: The author gratefully acknowledges the valuable guidance and support of Professor Dr. Mansour Esmaeilpour throughout the completion of the MSc dissertation from which this paper was derived.

Conflict of Interest: The author declares that there is no conflict of interest regarding the publication of this paper.

Financial Support: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Ethics Statement: This research does not involve human participants, animal subjects, or sensitive personal data. All data used in this study were obtained from publicly available sources or generated synthetically, and no ethical approval was required.

References

Ajunwa, I. (2017). Genetic testing meets big data: Tort and Contract Law issues. *Ohio St. LJ*, *75*, 1225.

Amin, J., Sharif, M., Yasmin, M., Ali, H., & Fernandes, S. L. (2017). A method for the detection and classification of diabetic retinopathy using structural predictors of bright lesions. Journal of Computational Science, 19, 153-164.

Blömer, J., Lammersen, C., Schmidt, M., & Sohler, C. (2016). Theoretical analysis of the k-means algorithm–a survey. In *Algorithm Engineering* (pp. 81-116). Springer, Cham.

Chandra, B., & Sharma, R. K. (2014, October). Fast learning for big data applications using parameterized multilayer perceptron. In *2014 IEEE International Conference on Big Data (Big Data)* (pp. 17-22). IEEE.

Fahad, S. A., & Alam, M. M. (2016). A modified K-means algorithm for big data clustering. *International Journal of Science, Engineering and Computer Technology*, *6*(4), 129.

Fan, J., Sun, Q., Zhou, W. X., & Zhu, Z. (2018). Principal component analysis for big data. *arXiv preprint arXiv:1801.01602*.

Feng, J., Yu, Y., & Zhou, Z. H. (2018). Multi-layered gradient boosting decision trees. Advances in neural information processing systems, 31.

Figueiredo, E., Macedo, M., Siqueira, H. V., Santana Jr, C. J., Gokhale, A., & Bastos-Filho, C. J. (2019). Swarm intelligence for clustering—A systematic review with new perspectives on data mining. Engineering Applications of Artificial Intelligence, 82, 313-329.

Ghosal, A., Nandy, A., Das, A. K., Goswami, S., & Panday, M. (2020). A short review on different clustering techniques and their applications. Emerging technology in modelling and graphics, 69-83.

Hadi, A. A. A., & Al-Furat, A. A. (2018). Performance analysis of big data intrusion detection system over random Forest algorithm. *International Journal of Applied Engineering Research*, *13*(2), 1520-1527.

Jung, S. H., Kim, J. C., & Sim, C. B. (2016). Prediction Data Processing Scheme using an Artificial Neural Network and Data Clustering for Big Data. *International Journal of Electrical & Computer Engineering (2088-8708)*, *6*(1).

Kyriienko, O., & Magnusson, E. B. (2022). Unsupervised machine learning for fraud detection. *arXiv preprint arXiv:2208.01203*.

Ma, J., Ding, Y., Cheng, J. C., Tan, Y., Gan, V. J., & Zhang, J. (2019). Analyzing the leading causes of traffic fatalities using XGBoost and grid-based analysis: a city management perspective. *IEEE Access*, *7*, 148059-148072.

Mittal, M., Goyal, L. M., Hemanth, D. J., & Sethi, J. K. (2019). Clustering approaches for high-dimensional databases: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9(3), e1300.

Pandey, K. K., & Shukla, D. (2022). Maxmin distance sort heuristic-based initial centroid method of partitional clustering for big data mining. Pattern Analysis and Applications, 25(1), 139-156.

Quellec, G., Lamard, M., Cozic, M., Coatrieux, G., & Cazuguel, G. (2016). Multiple-instance learning for anomaly detection in digital bank. *Ieee transactions on medical imaging*, *35*(7), 1604-1614.

Ramchoun, H., Idrissi, M. J., Ghanou, Y., & Ettaouil, M. (2017, March). Multilayer Perceptron: Architecture Optimization and training with mixed activation functions. In *Proceedings of the 2nd international Conference on Big Data, Cloud and Applications* (pp. 1-6).

Scardapane, S., Wang, D., & Panella, M. (2016). A decentralized training algorithm for echo state networks in distributed big data applications. *Neural Networks*, *78*, 65-74.

Suthaharan, S., & Suthaharan, S. (2016). Decision tree learning. *Machine learning models and algorithms for big data classification: Thinking with examples for effective learning*, 237-269.