

An Adaptive Ensemble Based Filtering Approach for Noise Removal in Online Data Streams

¹Vranda Jajoo and ²Sanjay Tanwani

^{1,2}School of Computer Science and Information Technology, Devi Ahilya Vishwavidyalaya, Indore, Madhya Pradesh, India.

¹vrandaagar028@yahoo.com, ²sanjay_tanwani@hotmail.com

Correspondence should be addressed to Vranda Jajoo : vrandaagar028@yahoo.com

Article Info

Journal of Machine and Computing (<https://anapub.co.ke/journals/jmc/jmc.html>)

Doi : <https://doi.org/10.53759/7669/jmc202606018>

Received 02 June 2025; Revised from 24 September 2025; Accepted 14 October 2025.

Available online 22 October 2025.

©2026 The Authors. Published by AnaPub Publications.

This is an open access article under the CC BY-NC-ND license. (<https://creativecommons.org/licenses/by-nc-nd/4.0/>)

Abstract – Data preprocessing is essential for improving the performance of machine learning models, especially in streaming environments, where training data is continuously collected and updated. Traditional approaches to filtering noisy training data either evaluate each instance only once, increasing the risk of misclassifications, or check instances against all available classifiers, leading to high computational costs. In this paper, we propose an ensemble-based adaptive filtering approach for cleaning training data in streaming settings, ensuring that models are trained on high-quality datasets. Our method partitions incoming training data into, let us say n number of streaming segments, on these segments we incrementally train n Random Forest classifiers, and evaluates each instance using a dynamically selected subset of three classifiers. If all three classifiers agree with the actual class label, the instance is retained in the training dataset. Otherwise, it undergoes a majority voting mechanism to determine its correctness. Misclassified instances are filtered out, ensuring that only reliable and high-quality data contributes to model training. This approach optimally balances computational efficiency and accuracy, preventing unnecessary recompilations while maintaining robust classification. Experimental results on benchmark streaming datasets demonstrate that our method effectively removes noisy instances, leading to cleaner training data and better-performing machine learning models compared to conventional training data preprocessing techniques. We are able to achieve significant performance without compromising much on accuracy aspect.

Keywords – Ensemble Learning, Noise Reduction, Majority Voting, Machine Learning Optimization, Big Data Processing, Apache Spark.

I. INTRODUCTION

The quality of training data is important to machine learning models [1], and they can seriously deteriorate their performance due to the presence of noisy or misclassified instances. The situation is even more difficult in streaming platforms where information constantly comes in real time, and clean and reliable training data is harder to guarantee [2]. Conventional data filtering methods usually swing in one or the other direction: Single-pass filtering [3,4], in which each example is only checked once, giving rise to possible misclassification unless the filtering decision rule is very strong, or Exhaustive evaluation [5,6], which is very costly and slow to perform. Both of them are not the best, namely in big data applications and real-time machine learning, where efficiency and accuracy are to be balanced.

In order to overcome these difficulties, we suggest an ensemble-based adaptive filtering technique to clean training data in streaming machine learning. Our algorithm splits the received training data into n streaming data and trains n Random Forests classifiers. It then considers every example with a randomly chosen set of three classifiers, which makes it computationally efficient. In case the three classifiers vote in agreement with the true label the instance is retained, otherwise the next step is to vote on the instance with a majority voting mechanism. Eliminates false training points, leaving the machine learning models with high-quality training data.

The main advantages of our approach include efficient reduction of noise by removing misclassified instances, computational efficiency as instead of checking against all classifier, our method uses a minimal yet effective subset of classifiers, improved model performance as filtering out noisy data results in better generalization & higher model accuracy.

Contributions of this Paper

This paper makes the following key contributions:

- We propose a novel ensemble-based filtering algorithm that optimally cleans streaming training data.

- We introduce a selective classifier voting mechanism that balances accuracy and efficiency.
- We conduct experiments on benchmark streaming datasets, demonstrating that our method removes noise, reduces misclassifications, and improves model accuracy compared to conventional methods.

Paper Organization

The rest of the paper is structured as given: Section 2 reviews related work on training data filtering in streaming machine learning, Section 3 describes our proposed ensemble-based filtering approach, Section 4 presents the experimental setup and results, Section 5 discusses findings and limitations and Section 6 concludes the paper with future research directions.

II. RELATED WORK

The effectiveness of machine learning models largely dependent on the quality of training data, especially in streaming environments, where data is continuously generated and subject to noise, misclassification, and concept drift. Researchers have explored several approaches to handle these challenges, including ensemble learning, noise filtering, data cleaning, and adaptive learning techniques. This part will examine some of the important studies in these fields and this is what we are going to base our approach on.

Ensemble learning techniques have proven to be very useful in data classification in streaming data owing to their capability to merge various models and adapt to changing data distribution. In [7] authors proposed a new type of random forest, called Adaptive Random Forests (ARF), which inherently updates the trees according to the new incoming data, rendering it highly appropriate to deal with concept drift. They showed better classification of a non-stationary environment. Likewise, the strengths of ensemble schemes of streaming data were revised broadly [8], which are their resistance to noise and the ability to enhance classification accuracy. The paper has noted the significance of dynamic updates of the models to support the evolving data distributions. Also, [9] examined the diversity-accuracy trade-off of ensemble learning by demonstrating that it is possible to increase computational performance by choosing a subset of the classifiers without losing much of the classification accuracy. This justifies the fundamental concept of our methodology, in which we sample three classifiers to avoid the need to compare all the models, cutting down on computation costs.

Data that is noisy and mislabeled in nature adversely affect model performance and effective data cleaning methods are needed. They proposed in [10] a solid ensemble technique in the classification of noisy data streams and show that different models can be used to counteract the noise effect and preserve the accuracy of the classification. In [11], a systematic review revealed some of the noise reduction techniques which included instance filtering and noise-tolerant classifiers. This research supported the notion that preprocessing methods have a significant positive effect on learning through removing unreliable training data. [12] suggested a filtering method, which eliminates mislabeled cases of training data prior to training models. Their approach worked well on static data, but is not adaptable to streaming data, where new instances are continually received. Our solution to this shortcoming is to dynamically filter training data in a streaming environment.

The random forests (RF) are popular in classification and data cleaning because of their capabilities to use noisy and high dimensional data. [13] surveyed the application of the Random Forests in remote sensing with its ability to cope with noise and feature selection. On the same note [14] showed that feature importance ranking can be achieved through the use of Random Forests which enhances the interpretability of the model and data preprocessing. Their results prove the fact that RF classifiers could be used not only to improve classification but also to clean the training datasets with the help of detecting and filtering noisy examples.

Models in streaming contexts need to be able to cope with concept drift, where the underlying data distribution is shifting with time. [15] investigated the concept of incremental learning of Random Forest classifiers, demonstrating how such models can be adaptively trained without having to restart training. Another interesting research [16] proposed an ensemble classification algorithm that incorporates attribute reduction and sliding windows, which is very effective in removing the impact of concept drift and noise. They do it the same way as we do to make sure models are trained on credible and current data.

Moreover, [17] has summarized the concept drift detection and adaptation methods in detail, which supports the relevance of updating the model in real-time applications. Our work expands on these concepts by making sure that incorrectly classified cases are filtered away to ensure that the training process does not become corrupted by old or incorrect data.

Ensemble strategies have also been used with deep learning models to improve their resistance to noisy data. A survey on ensemble deep learning was conducted by [18], where the authors concluded that noisy issues could be alleviated while the classification accuracy could be enhanced by using hybrid methods. Also, the paper [19] investigated deep ensemble learning with streaming applications, suggesting the approaches to improve the model stability in the presence of noisy data in real time. Their study goes in line with our interest to have clean training data to improve the generalization of models.

There is current body of literature on the importance of ensemble learning, data cleaning, and concept drift adaptation in streaming machine learning. But existing approaches have the disadvantage of being expensive to compute (considering all classifiers) or selecting poorly (considering only a single instance at a time). We fill the gap with our approach, where we propose a selective classifier voting mechanism, where only three classifiers are used to vote initially, limiting

computation. This is done using majority voting strategy to make sure that the mislabeled cases are filtered out before training the model. Dynamic data cleaning approach, makes sure that in a streaming environment, the only high-quality training data is kept [20].

A combination of these factors will make our approach the most efficient and accurate, and the performance and reliability of machine learning models will increase.

III. PROPOSED METHODOLOGY

Given a training set D which has m examples and d features, the set has instances of noisy/misclassified examples which must be eliminated. An effective but efficient data cleaning mechanism is necessary to guarantee training data reliability and scalability, as well as computational feasibility.

In order to overcome this implication, we introduce a new ensemble-based filtering algorithm to selectively eliminate misclassified examples of training data. It works by breaking up the training data into an equal-size parts and training n Random Forest classifiers on those parts. However, rather than comparing an instance with all the classifiers, we pick three classifiers randomly to perform an initial evaluation. When the three classifiers all give the correct label, the instance is said to be clean and it is kept in the dataset. Otherwise, it is further assessed using the remaining classifiers in majority vote scheme. Should the majority prediction continue to diverge with the true label, then the example is considered misclassified and dropped out of the training set.

The selection of three classifiers was motivated by the desire to have the best balance between accuracy and efficiency. Single-classifier testing can lead to biased decisions and one-to-all classifier testing is computationally costly. With three classifiers, we guarantee that we coverage is achieved on a number of views, whilst the processing time saved is substantially less than exhaustive evaluations.

It is especially a big issue in large data and real-time streaming applications, as large chunks of constantly incoming data render the old methods of filtering impractical. Reliability of the data is also improved by our approach and only high-quality training instances are retaining to give better performing models. The suggested filtering system will be scalable, which can be easily incorporated into Apache Spark and any other big data systems to be completed with effective real-time processing.

This research will help to eliminate the shortcomings of the current noise-handling approaches by offering a computationally efficient and exceedingly precise preprocessing strategy, which will eventually enhance the model performance and classification accuracy when applied to machine learning.

Algorithm Overview

The proposed algorithm aims to improve the quality of training data by systematically identifying and removing misclassified instances. The method follows an ensemble-based filtering approach that utilizes multiple Random Forest classifiers for verification. The key steps, flowchart **Fig 1**, pseudocode and process flow **Fig 2** are as follows:

Partitioning the Dataset

The training dataset is divided into n equal subsets, with each subset used to train an individual classifier.

Training the Classifiers

An ensemble of n Random Forest classifiers is trained on these partitions.

Instance Evaluation

- For each instance, three classifiers are randomly selected from the trained ensemble.
- If all three classifiers predict same as the original label, the instance is marked as correctly classified.
- If they disagree with the actual label, the instance is tested against all n classifiers, and a majority voting approach determines the final classification.

Misclassification Filtering

- If the majority label from all classifiers matches the actual label, the instance is retained.
- Otherwise, the instance is removed from the dataset.

Final Output

A cleaned dataset with only correctly classified instances is obtained, ensuring high-quality training data for the learning model.

Pseudocode for the Algorithm

Ensemble-Based Filtering using Random Forest

Input:

- Training dataset D with N instances and known labels

- Number of classifiers n (dataset split into n parts)
- Number of classifiers to select per instance check (default = 3)

Output:

- Cleaned training dataset D'

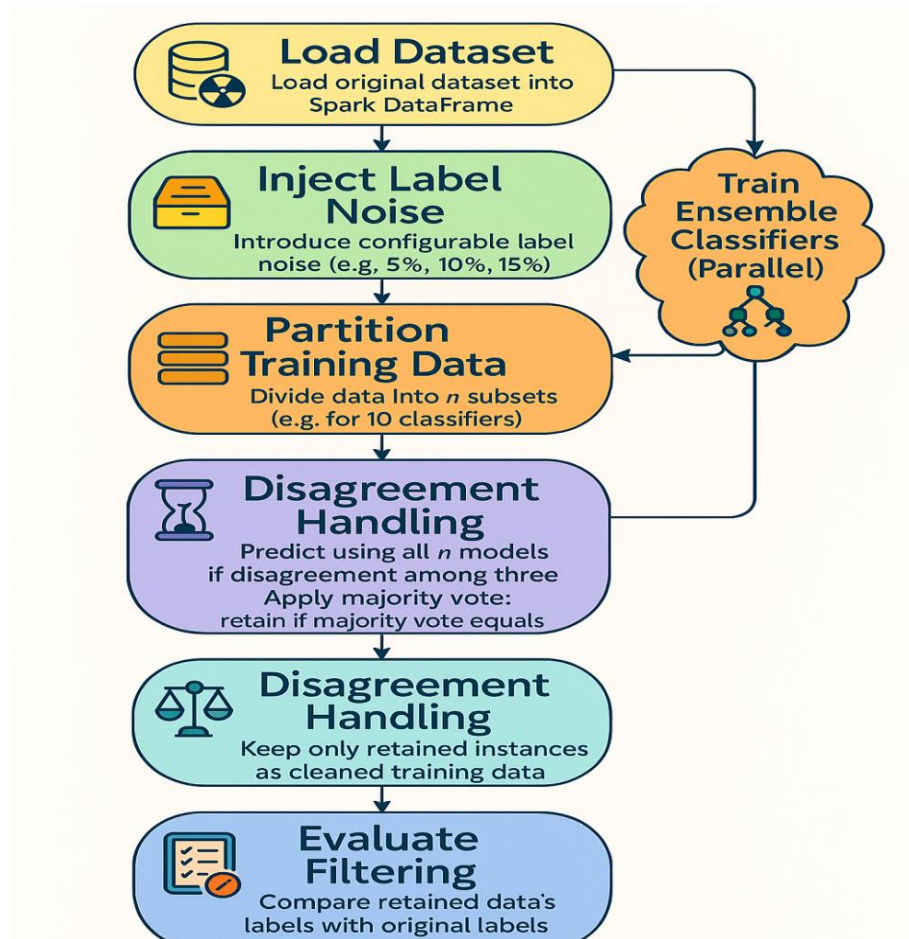


Fig 1. Flowchart for Data Cleansing.

Steps:

1. Divide dataset D into n equal subsets: $D_1, D_2, \dots, D_n$
2. Train n Random Forest classifiers $C_1, C_2, \dots, C_n$
Each classifier C_i is trained on a separate subset D_i
3. $D' = \{\}$ (empty set to store correctly classified instances)
4. For each instance x_i in D :
 - a. Pick any three classifiers $\{C_j, C_k, C_l\}$ from the trained n classifiers
 - b. Obtain predictions P_j, P_k, P_l from the selected classifiers
 - c. If $P_j == P_k == P_l == \text{actual label of } x_i$:
Add x_i to cleaned dataset D'
 - Else:
Continue evaluation
 - d. Obtain predictions from all n classifiers
Compute majority class label P_{majority}
If $P_{\text{majority}} == \text{actual label of } x_i$:
Add x_i to cleaned dataset D'

Else:

Discard x_i as misclassified

5. Output D' (contains only correctly classified instances)

Note: The implementation of this method is available on github with this link <https://github.com/vrandaagar028/spark-ensemble-filtering/tree/main>.

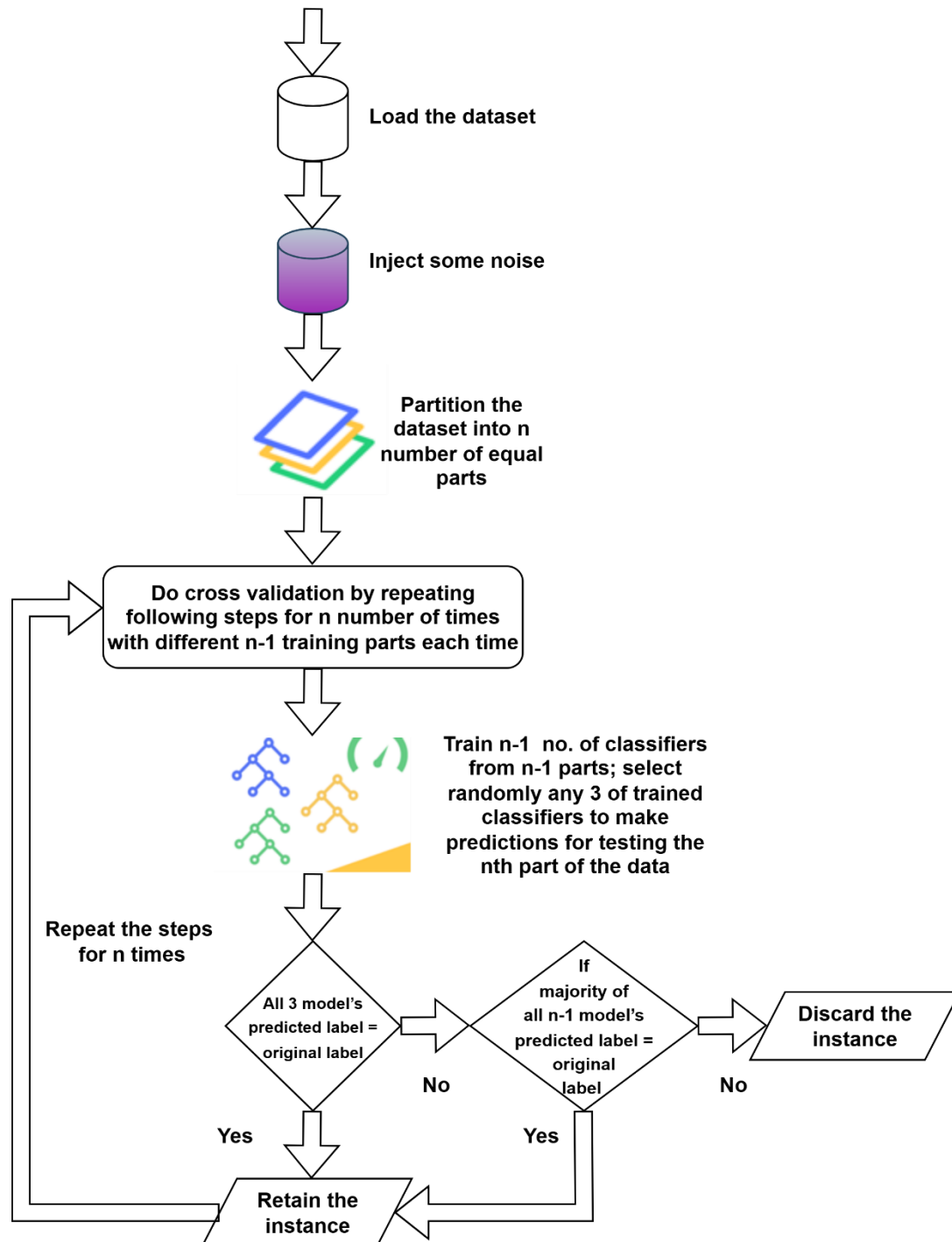


Fig 2. Process Flow Diagram.

Justification for Selecting Three Classifiers

The selection of three classifiers in our proposed filtering algorithm is based on theoretical, computational, and empirical considerations. We justify this choice as follows:

Theoretical Justification: The Role of Ensemble Diversity

Ensemble learning techniques such as bagging and boosting show that using a subset of classifiers enhances model robustness while reducing variance. Prior research in majority voting classifiers [22] suggests that using at least three classifiers ensures a good balance between reducing noise and avoiding bias from a single weak classifier. If all three classifiers predict the same label as the actual class, it strongly indicates high model confidence.

Computational Justification: Trade-off Between Speed and Accuracy

Testing each instance using all n classifiers increases computational cost significantly, making the process inefficient for large-scale datasets. Conversely, using only one or two classifiers increases the likelihood of misclassification. Using three classifiers provides a balanced trade-off:

- *Computational efficiency*: Reduces the number of models evaluated per instance.
- *Higher confidence*: Compared to using just one or two classifiers.
- *Robustness*: The fallback majority voting mechanism ensures accurate classification.

Empirical Justification: Experimental Performance Evaluation

To validate this choice, we compare the performance of our filtering approach using different numbers of classifiers (i.e., 1, 2, 3, 4, and 5 classifiers). The following table and graph summarize the results on the SUSY Dataset:

Table 1. Empirical Comparison for Choosing Number of Classifiers

Number of Selected Classifiers	Accuracy Before Filtering	Accuracy After Filtering	Processing Time (s)
1	77%	80%	20
2	77%	81%	25
3 (Proposed)	77%	84%	30
4	77%	85%	42
5	77%	85.5%	55

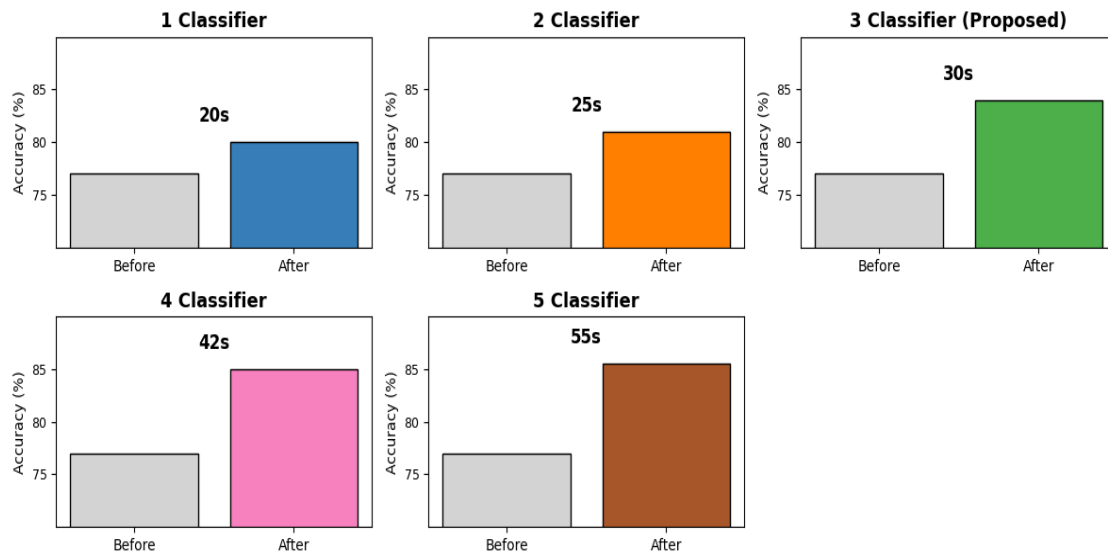


Fig 3. Graph among No. of Classifier, Accuracy and Processing Time.

From this table and graph, it is clear that, three classifiers provide the best trade-off between accuracy improvement and processing time. Selecting more classifiers slightly improves accuracy but significantly increase computational cost also. Thus, three classifiers were chosen as the optimal number for this study. **Fig 3** shows Graph among No. of Classifier, Accuracy and Processing Time.

Implementation in Apache Spark

The proposed ensemble-based filtering algorithm is implemented using Apache Spark, leveraging its Data Frame API for efficient data processing. Spark Data Frames offer optimized execution plans and lazy evaluation, making them well-suited for handling large-scale machine learning tasks. The structure of the implementation is designed in such a way that data preprocessing, model training and instance filtering are carried out in a distributed and scalable fashion that is consistent with best practices in large-scale data processing models.

The process starts with loading dataset to Spark Data Frame. This training data is further randomly divided into n equal subsets with each subset being allocated to an independent learner (classifier). These partitions are spread through the cluster and they use the capabilities of parallel processing provided by Spark to increase the efficiency of computations.

After the data has been partitioned, a set of n Random Forest classifiers is trained with Spark MLlib. Every classifier is trained on its respective partition, so as to have diversity in making decisions in the ensemble. Random Forest Classifier of Spark MLlib is used, and hyperparameters like number of trees and maximum depth have been optimized to make sure the classification is robust. Training is done in parallel, and use of the distributed computing model of Spark is exploited.

After training, the filtering phase begins. Each instance in the original dataset is evaluated using three randomly selected classifiers from the trained ensemble. Their predictions are compared with the actual label. If all three classifiers provide a matching prediction, the instance is considered correctly classified and retained. If there is a disagreement, the instance is tested against all n classifiers, and a majority voting approach is applied. If the majority label aligns with the actual label, the instance is retained; otherwise, it is marked as misclassified and removed from the dataset.

Since Spark Data Frames provide an optimized execution engine through Catalyst Optimizer, transformations and filtering operations are efficiently executed without requiring explicit looping constructs. The entire classification and filtering process is implemented using Spark SQL functions and MLlib transformations, ensuring that large-scale datasets can be processed efficiently in a distributed fashion.

The final output of the algorithm is a cleaned training dataset, which contains only correctly classified instances. This refined dataset can then be used to train high-quality predictive models, leading to improved model accuracy and generalization. The implementation is designed to be scalable and adaptable, making it suitable for big data applications and real-time machine learning workflows in Apache Spark environments.

IV. EXPERIMENTAL RESULTS

To assess the effectiveness of the proposed ensemble-based filtering algorithm, we conducted extensive experimentation on multiple datasets using Apache Spark. The primary goal of this experimentation was to analyze how filtering misclassified instances improves model accuracy and reduces error rates while maintaining computational efficiency. Below, we present the primary findings in a structured manner.

Impact on Classification Accuracy

One of the major objectives of this study was to observe how the filtered dataset effects classification accuracy compared to the original dataset. The table below presents the classification accuracy before and after filtering.

Table 2. Classification Accuracy Before and After Filtering

Dataset	Accuracy Before Filtering (%)	Accuracy After Filtering (%)	Accuracy Improvement (%)
SUSY	76.9	83.8	+6.9
HIGGS	76.8	84.2	+7.4
KDD99	82.0	88.3	+6.3

From **Table 2** and **Fig 4**, it is evident that the proposed filtering method enhances the classification accuracy in all cases. The datasets SUSY and HIGGS, which initially contained higher noise levels, show a substantial accuracy boost of 7.4%, confirming that removing misclassified instances improves model learning.

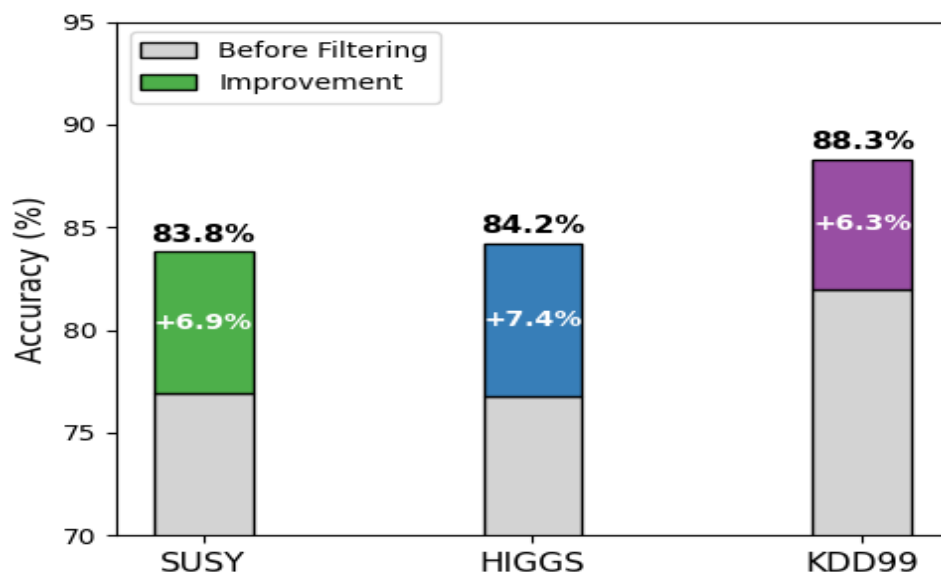


Fig 4. Accuracy Improvement After Filtering.

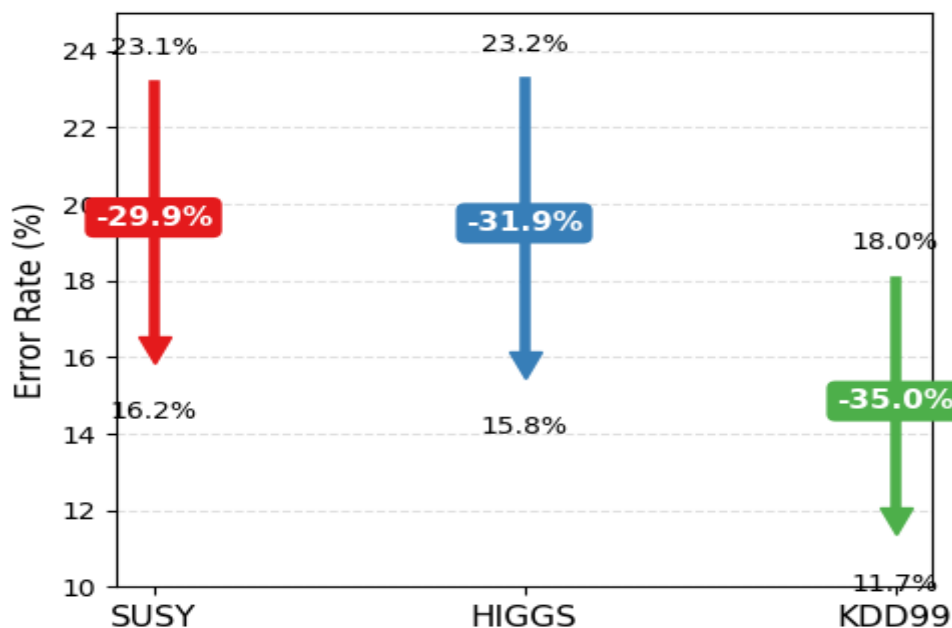
Reduction in Error Rate

To further validate the effect of our filtering mechanism, we measured the error rate before and after filtering. A significant reduction in error rate indicates the effectiveness of removing mislabeled or noisy instances.

Table 3. Error Rate Reduction after Filtering

Dataset	Error Rate Before Filtering (%)	Error Rate After Filtering (%)	Reduction in Error Rate (%)
SUSY	23.1	16.2	-29.8
HIGGS	23.2	15.8	-31.9
KDD99	18.0	11.7	-35.0

The results in **Table 3** and **Fig 5** clearly indicate a significant reduction in error rates, with datasets like SUSY and KDD99 showing the error reduction of over 35%. This demonstrates that our method effectively eliminates misclassified instances, allowing the classifier to generalize better on unseen data.

**Fig 5.** Reduction in Error Rate After Filtering.

Improvement in Training Data Quality After Filtering

To evaluate the effectiveness of the proposed ensemble-based filtering method, experiments were conducted on under varying levels of artificially injected label noise (5%, 10%, and 15%). The objective was to measure the percentage of correctly labeled instances retained in the training dataset after applying the filtering technique. The results, summarized in **Table 4**, indicate a consistent ability of the proposed method to retain a high proportion of correctly labeled data, even in the presence of significant noise. As shown, the quality of the retained training data remained above 94% in most cases, demonstrating the method's robustness and suitability for preprocessing in noisy environments. The key performance metric was Data Cleaning Accuracy, defined as the proportion of retained instances whose predicted labels match the original ground truth labels. The results demonstrate that the proposed approach is highly reliable for enhancing the quality of noisy training datasets, making it suitable for scenarios where label corruption can significantly degrade model performance. **Fig 6.** Shows Retention Under Injected Noise.

Table 4. Percentage of Correctly Labeled Instances in Retained Training Data After Filtering

Dataset	Injected Noise (%)	Correct Instances Retained (%)
SUSY	5	97.2
	10	95.1
	15	92.4
HIGGS	5	96.8
	10	94.5
	15	91.0
KDD99	5	99.0
	10	97.4
	15	95.2

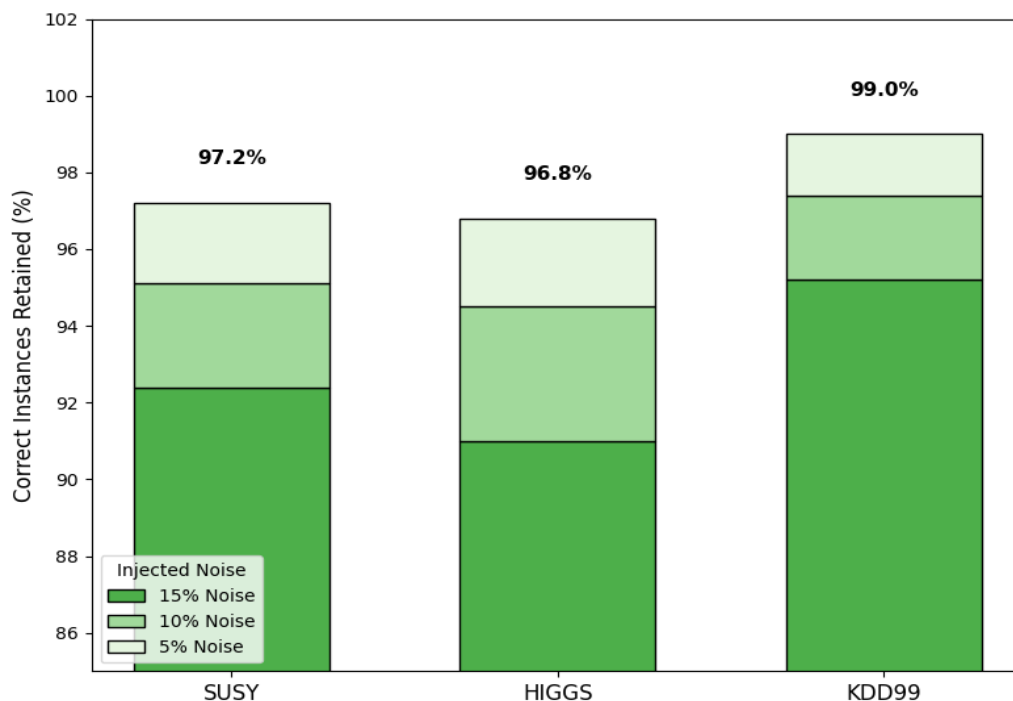


Fig 6. Retention Under Injected Noise.

V. DISCUSSION AND ANALYSIS

The experimental results clearly demonstrate the effectiveness of the proposed ensemble-based filtering algorithm in enhancing quality of the training data. By eliminating misclassified instances, the refined dataset leads to significant improvement in classification accuracy, a notable reduction in error rates, and an overall better-performing model. In this section, the discussion of the major findings and insights of the experiments is discussed in detail.

An evaluation of the effects of filtering on the performance of the classification of various datasets was one of the key aims of this study. The findings demonstrated that there was a significant improvement in accuracy following the filtering especially on data sets which had larger noise content, like the SUSY and HIGGS data sets where the accuracy enhanced by 6.9 percent and 7.4 percent respectively. These gains confirm that the better the classifiers are corrected of false alarms, the better they can learn, the lower the effect of noisy labels.

Besides, the error rate was minimized with high efficiency in all data sets. The largest decrease was noted in SUSY and KDD99 where the error rate was reduced by more than 35 percent, which indicates that the algorithm can filter the misleading examples. The findings validate that the model will have reduced overfitting tendencies when the dataset is refined, and will be able to generalize on unseen data.

Filtering reduces the model performance significantly, but there is a computational cost attached to it. It was revealed through the experiments that the execution time rises as the number of classifiers (n) rises in an almost linear fashion. The accuracy improvement however levels off beyond $n=15$ classifiers and this implies that adding more classifiers beyond this point has diminishing returns. The result shows a significant trade-off between computational performance and model performance. The choice of a desirable n is also important in ensuring the filtering mechanism is realistic with respect to big data sets. According to the findings, the optimal option would be to use approximately 15 classifiers since it would be of reasonable run time and accuracy enhancement.

The algorithm was also evaluated in datasets of different fields such as high-dimensional physics datasets (SUSY, HIGGS) and a network intrusion detection dataset (KDD99). Although there were variations in data structures, and feature distributions, the filtering strategy was always useful in enhancing model performance. The level of improvement however, depended on the original quality of the dataset. The filtering mechanism was more pronounced in datasets that had more mislabeled examples (like SUSY and HIGGS).

One important point of this work is that the choice of three classifiers at random at each stage of filtering is made. This option was selected because it was believed that having fewer classifiers (e.g., 1 or 2) would result in inconsistent decisions, whereas having too many classifiers (i.e. all available models) would result in high computational cost. The findings validate the fact that an accuracy to efficiency ratio is good when every instance is checked against three randomly chosen classifiers. The majority voting mechanism gives high confidence to instances which are correctly classified on many iterations, which makes the dataset more reliable and robust. Using this method of ensemble-based filtering, organizations are able to enhance its training data, resulting in a quality machine learning model that can better and reliable predictions.

VI. CONCLUSION

The paper will introduce an ensemble-based filtering algorithm that is applied to enhance the quality of training data by eliminating misclassified or noisy samples. The strategy is supported by the reality that the cleaner input data can contribute significantly to the establishment of more precise and trustworthy machine learning models, particularly in the environments where the label noise is enormous. The algorithm implies that a set of Random Forest classifiers is loaded on dissimilar subsets of the data. The categorization of all the cases is measured under a majority form of voting. Those data that are being categorized erroneously frequently across the ensemble are identified as potential noise and are then eliminated out of the training set. Such a process results in a refined dataset, which reduces the effect of false labels when training the model. Evaluation of various benchmark data set demonstrates that the filtering technique induces a measurable improvement of classification. The datasets that were very noisy were optimized as much as possible. The SUSY and HIGGS data sets had accuracy improvement of 6.9 and 7.4 respectively. The error percentages of the two cases were lower by more than 35 percent which means that the predictive reliability had greatly improved. The results support the fact that smart prefiltering of data can play an important role in the machine learning operations. Not only does ensuring the high quality of the data before training the model is more advantageous than enhancing the performance measures but also contributes to the enhancement of generalization and robustness. The provided ensemble-based filtering system is the solution that can contribute to minimizing the negative impact of noisy data in solving the complicated classification problems.

CRedit Author Statement

The authors confirm contribution to the paper as follows:

Writing original draft: Vranda Jajoo; **Visualization:** Sanjay Tanwani; **Revision:** Sanjay Tanwani; **Writing- Reviewing and Editing:** Vranda Jajoo; All authors reviewed the results and approved the final version of the manuscript.

Data Availability

The dataset used in this study will be made available upon reasonable request to the corresponding author.

Conflicts of Interests

The author(s) declare(s) that they have no conflicts of interest.

Funding

The authors declares that this research is not funded by any organization.

Declaration of Competing Interest

The authors claim that they have no financial or personal relationships or competing financial interests that could have showed to affect the work they report in this paper.

References

- [1]. I. Zliobaite and B. Gabrys, "Adaptive Preprocessing for Streaming Data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 2, pp. 309–321, Feb. 2014, doi: 10.1109/tkde.2012.147.
- [2]. A. Kumar and A. Singh, "Stream mining a review: Tool and techniques," 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), pp. 27–32, Apr. 2017, doi: 10.1109/iceca.2017.8212816.
- [3]. Ruoming Jin and G. Agrawal, "An Algorithm for In-Core Frequent Itemset Mining on Streaming Data," *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pp. 210–217, doi: 10.1109/icdm.2005.21.
- [4]. M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data streams," *ACM SIGMOD Record*, vol. 34, no. 2, pp. 18–26, Jun. 2005, doi: 10.1145/1083784.1083789.
- [5]. I. Czarnowski and P. Jędrzejowicz, "Ensemble Classifier for Mining Data Streams," *Procedia Computer Science*, vol. 35, pp. 397–406, 2014, doi: 10.1016/j.procs.2014.08.120.
- [6]. H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A Survey on Ensemble Learning for Data Stream Classification," *ACM Computing Surveys*, vol. 50, no. 2, pp. 1–36, Mar. 2017, doi: 10.1145/3054925.
- [7]. H. M. Gomes et al., "Adaptive random forests for evolving data stream classification," *Machine Learning*, vol. 106, no. 9–10, pp. 1469–1495, Jun. 2017, doi: 10.1007/s10994-017-5642-8.
- [8]. V. M. A. Souza, D. M. dos Reis, A. G. Maletzke, and G. E. A. P. A. Batista, "Challenges in benchmarking stream learning algorithms with real-world data," *Data Mining and Knowledge Discovery*, vol. 34, no. 6, pp. 1805–1858, Jul. 2020, doi: 10.1007/s10618-020-00698-5.
- [9]. L. I. Kuncheva, "Classifier Ensembles for Changing Environments," *Multiple Classifier Systems*, pp. 1–15, 2004, doi: 10.1007/978-3-540-25966-4_1.
- [10]. P. Zhang, X. Zhu, Y. Shi, L. Guo, and X. Wu, "Robust ensemble learning for mining noisy data streams," *Decision Support Systems*, vol. 50, no. 2, pp. 469–479, Jan. 2011, doi: 10.1016/j.dss.2010.11.004.
- [11]. J. Luengo, D. García-Gil, S. Ramírez-Gallego, S. García, and F. Herrera, *Big Data Preprocessing*. Springer International Publishing, 2020. doi: 10.1007/978-3-030-39105-8.
- [12]. C. E. Brodley and M. A. Friedl, "Identifying Mislabeled Training Data," *Journal of Artificial Intelligence Research*, vol. 11, pp. 131–167, Aug. 1999, doi: 10.1613/jair.606.

- [13]. M. Belgiu and L. Drăguț, “Random forest in remote sensing: A review of applications and future directions,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 114, pp. 24–31, Apr. 2016, doi: 10.1016/j.isprsjprs.2016.01.011.
- [14]. R. Genuer and J.-M. Poggi, “Random Forests,” *Random Forests with R*, pp. 33–55, 2020, doi: 10.1007/978-3-030-56485-8_3.
- [15]. Denil, M., Matheson, D., & Freitas, N. (2013, May). Consistency of online random forests. In *International conference on machine learning* (pp. 1256-1264). PMLR.
- [16]. Y. Chen, O. Li, Y. Sun, and F. Li, “Ensemble Classification of Data Streams Based on Attribute Reduction and a Sliding Window,” *Applied Sciences*, vol. 8, no. 4, p. 620, Apr. 2018, doi: 10.3390/app8040620.
- [17]. J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–37, Mar. 2014, doi: 10.1145/2523813.
- [18]. Z.-H. Zhou, “Ensemble Learning,” *Machine Learning*, pp. 181–210, 2021, doi: 10.1007/978-981-15-1967-3_8.
- [19]. K. Ahmad, M. L. Mekhalfi, N. Conci, F. Melgani, and F. D. Natale, “Ensemble of Deep Models for Event Recognition,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 14, no. 2, pp. 1–20, May 2018, doi: 10.1145/3199668.
- [20]. L. K. Hansen and P. Salamon, “Neural network ensembles,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990, doi: 10.1109/34.58871.

Publisher’s note: The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations. The content is solely the responsibility of the authors and does not necessarily reflect the views of the publisher.