Machine Learning Techniques for Cleaning Raw Data

Mrs. Jaspreet Kaur

M.Tech Scholar, Computer Science and Engineering RCET, Bhilai, Chhattisgarh, India

Dr. Neelabh Sao

Associate Professor, Department of Information Technology RCET, Bhilai, Chhattisgarh, India

Abstract

Raw data collected from real-world environments is often incomplete, noisy, and inconsistent, significantly affecting machine learning (ML) model performance. This paper proposes a hybrid data cleaning framework that integrates statistical preprocessing with machine learning-based anomaly detection techniques. Unlike traditional approaches, the proposed method combines Z-score normalization, K-Means clustering, and Isolation Forest for robust outlier detection and data refinement. Experimental evaluation is conducted on larger benchmark datasets, and results are compared with baseline statistical methods. The proposed approach improves model accuracy, data consistency, and overall data quality. The findings demonstrate that hybrid intelligent data cleaning significantly enhances downstream ML performance and scalability.

Keywords—Data Cleaning, Machine Learning, Isolation Forest, K-Means, Data Quality, Outlier Detection

1. Introduction

Data cleaning is a fundamental yet resource-intensive phase in modern data science and machine learning workflows. Empirical studies indicate that a substantial proportion of a data practitioner's time—often exceeding 70–80%—is devoted to preprocessing tasks such as handling missing values, correcting inconsistencies, removing duplicates, and standardizing heterogeneous data formats. As organizations increasingly depend on data-driven systems across domains including healthcare, finance, e-commerce, and industrial automation, the quality of raw data directly influences the reliability, accuracy, and generalization capability of machine learning models.

Despite the availability of numerous data processing tools and frameworks, ensuring high-quality data at scale remains a persistent challenge. Traditional rule-based and statistical data cleaning techniques are often insufficient when dealing with large, complex, and dynamic datasets. Real-world data is inherently noisy and heterogeneous, frequently containing domain-specific anomalies such as irregular sensor readings, incomplete transaction logs, or inconsistent medical records. These complexities limit the effectiveness of static cleaning rules and necessitate more adaptive and intelligent approaches.

Machine learning (ML) techniques have emerged as promising solutions to address these limitations. By leveraging pattern recognition and data-driven learning, ML models can automatically detect anomalies, predict missing values, and identify inconsistencies without relying solely on predefined rules. Techniques such as clustering, classification, deep learning, and probabilistic modeling have been successfully applied to tasks including outlier detection, data imputation, and deduplication. Moreover, ML-based approaches offer scalability and adaptability, making them suitable for handling large-scale and evolving datasets.

However, several challenges remain unresolved. The effectiveness of ML-based data cleaning depends heavily on the availability of labeled data, computational resources, and domain knowledge. Additionally, issues related to

model interpretability, generalization across datasets, and integration into existing data pipelines continue to pose difficulties. Furthermore, there is a lack of comprehensive evaluation frameworks that systematically compare different ML-based cleaning techniques across diverse domains and data characteristics.

Contributions of the Paper

The main contributions are:

1. A hybrid data cleaning framework combining statistical and ML techniques
2. Integration of Z-score, K-Means clustering, and Isolation Forest
3. Comparative analysis with baseline methods
4. Performance evaluation using multiple metrics (Accuracy, Precision, Recall, F1-score)

2. Related Work

Data cleaning has long been recognized as a critical challenge in data management and analytics. Early foundational studies, such as the survey by Rahm and Do [1], established core concepts including duplicate detection, schema integration, and data transformation. As data volumes have increased and datasets have become more heterogeneous, the need for scalable and automated data cleaning solutions has intensified. Recent research has focused on developing algorithms and tools to address common data quality issues such as missing values, inconsistencies, outliers, and duplicate records. However, comprehensive evaluations across multiple domains and large-scale datasets remain limited.

2.1 Overview of Data Cleaning Algorithms

Several influential studies have explored the algorithmic foundations of data cleaning. Rahm and Do [1] provided a broad classification of data cleaning tasks, while Elmagarmid et al. [4] focused specifically on duplicate detection techniques in large datasets. These works emphasized the importance of similarity measures and blocking strategies, which are still widely used in modern entity resolution systems.

Constraint-based approaches have also gained attention for structured error detection. Chu et al. [7] proposed integrated pipelines combining constraint validation, error detection, and data repair, demonstrating the effectiveness of rule-based systems when domain knowledge is available. Similarly, Abedjan et al. [5] highlighted the diversity of data errors and the need for flexible detection mechanisms capable of handling multiple error types simultaneously.

Recent advancements have focused on scalability through parallel and distributed processing. Techniques incorporating task parallelism and data partitioning have shown significant improvements in processing large datasets efficiently. These developments indicate that scalability is a key requirement for modern data cleaning systems.

Interactive data cleaning systems have also been proposed to improve usability. Kandel et al.

[2] introduced Wrangler, which combines user interaction with automated suggestions to accelerate data transformation tasks. While effective for exploratory analysis, such systems often face limitations when applied to large-scale datasets.

2.2 Open Source Tools and Research Gaps

The emergence of open source data cleaning tools has significantly influenced both academic research and industry practices. Tools such as OpenRefine, Dedupe, and Python-based libraries provide functionalities for data transformation, deduplication, and validation. Studies have analyzed these tools in terms of usability and functionality, highlighting features such as interactive cleaning, rule-based validation, and automation capabilities.

However, existing research reveals several critical gaps:

1. **Limited large-scale benchmarking:**

Most studies focus on functionality rather than performance, with limited evaluation on datasets exceeding millions of records.

2. **Lack of domain diversity:**

Few works compare tool performance across different domains such as healthcare, finance, and industrial data, each of which exhibits unique data quality challenges.

3. **Insufficient evaluation of usability and integration:**

Practical aspects such as ease of integration, maintainability, and compatibility with data pipelines are often overlooked.

These gaps create uncertainty for practitioners when selecting appropriate tools for large-scale applications.

2.3 Automated Error Detection Approaches

Modern research has increasingly separated error detection from data repair, leading to specialized systems focused on identifying erroneous data.

(i) Rule-based methods:

These approaches use predefined constraints and validation rules to detect inconsistencies. They are effective when domain knowledge is well-defined but lack flexibility in dynamic environments.

(ii) Statistical and probabilistic models:

Probabilistic approaches model data distributions to identify anomalies. These methods offer high accuracy but may require significant computational resources.

(iii) Machine learning-based techniques:

Learning-based models, including ensemble methods and feature-based classifiers, have been used to detect anomalies and inconsistencies. These approaches adapt to complex data patterns and improve detection accuracy but depend on data quality and training strategies.

Although these methods achieve strong detection performance, many focus only on identifying errors rather than providing complete data cleaning solutions.

2.4 Importance of Performance in Large-Scale Environments

As datasets grow to millions or billions of records, performance becomes a critical factor in data cleaning. Traditional in-memory approaches often fail due to high computational and memory requirements. Large-scale systems require efficient data processing techniques, including parallel execution and optimized storage

mechanisms.

Research has shown that inefficient data cleaning pipelines can significantly delay downstream analytics and machine learning processes. In large organizations, slow or unreliable preprocessing can reduce productivity and hinder decision-making. Furthermore, improvements in data quality have been directly linked to better model performance, emphasizing the importance of efficient and accurate cleaning techniques.

In regulated domains such as healthcare and finance, additional requirements such as validation, traceability, and documentation further increase the complexity of data cleaning processes.

2.5 Summary of Research Gaps

Despite significant progress in data cleaning research, several challenges remain:

- Lack of comprehensive benchmarking across tools and techniques
- Limited evaluation on large-scale, real-world datasets
- Insufficient focus on domain-specific data characteristics
- Weak integration between detection, correction, and validation processes

To address these limitations, this paper conducts a systematic evaluation of machine learning-based data cleaning techniques, focusing on scalability, accuracy, and practical usability across multiple domains

3. Frameworks under Study

To evaluate the effectiveness of data cleaning approaches in real-world scenarios, this study examines five widely used tools and frameworks. These tools were selected based on their popularity, diverse functionalities, and applicability to different data cleaning tasks such as duplicate detection, validation, transformation, and large-scale processing. The selected tools include OpenRefine, Dedupe, Great Expectations, TidyData (PyJanitor), and a baseline Pandas pipeline.

3.1 OpenRefine

OpenRefine is an open-source tool designed for interactive data cleaning and transformation. It provides a user-friendly interface that allows users to explore datasets, detect inconsistencies, and apply transformations without extensive programming knowledge.

Key features include faceting, clustering for duplicate detection, and support for various data formats such as CSV, JSON, and Excel. OpenRefine is particularly effective for small to medium-sized datasets and exploratory data cleaning tasks. However, its reliance on in-memory processing limits its scalability for very large datasets.

3.2 Dedupe

Dedupe is a Python-based machine learning library specifically designed for entity resolution and duplicate detection. It uses supervised learning techniques to identify duplicate records by learning similarity functions from labeled examples.

The framework supports active learning, allowing users to iteratively improve model accuracy by labeling small subsets of data. Dedupe is highly effective for structured datasets where duplicate detection is a primary concern. However, it requires training data and may involve additional computational overhead for large datasets.

3.3 Great Expectations

Great Expectations is a data validation framework that enables users to define, test, and document data quality rules. It focuses on ensuring data consistency, completeness, and accuracy through declarative expectations.

The framework integrates well with modern data pipelines and supports automated validation workflows. It is particularly useful in production environments where continuous data monitoring is required. However, defining comprehensive validation rules can be time-consuming, and performance may be affected when processing large datasets.

3.4 TidyData (PyJanitor)

PyJanitor is a Python library built on top of Pandas that provides convenient functions for data cleaning and transformation. It simplifies common preprocessing tasks such as column renaming, handling missing values, and data reshaping.

The library is lightweight and easy to integrate into existing Python workflows. It is suitable for medium-scale datasets and rapid development. However, like Pandas, it is limited by in-memory processing and may not scale efficiently to very large datasets.

3.5 Pandas Baseline Pipeline

Pandas is one of the most widely used data manipulation libraries in Python and serves as a baseline for comparison in this study. It provides extensive functionality for handling missing data, filtering, transformation, and aggregation.

A custom Pandas-based pipeline was implemented to perform standard data cleaning tasks, including duplicate removal, outlier detection, and format standardization. While Pandas offers flexibility and ease of use, its performance is constrained by memory limitations, making it less suitable for large-scale data processing without additional optimization techniques.

4. Methodology

A. Dataset Description

A CSV dataset with attributes such as Age, Salary, Experience, and Department is used. The dataset contains missing values, noise, and outliers.

B. Data Cleaning Pipeline

1. Data Collection (using Pandas)
2. Data Profiling (missing values, duplicates)
3. Missing Value Handling (mean/mode imputation)
4. Outlier Detection (Z-score method)

5. Data Transformation (normalization, encoding)
6. ML-Based Cleaning (K-Means, Isolation Forest)

Figure 4.1: Proposed Data Cleaning Pipeline

C. Outlier Detection

We use the Z-score method:

$$z = \frac{x-\mu}{\sigma} \qquad (1)$$

Figure 4.2: Standard Normal Distribution Curve Showing Z-Score Calculation Where:

- X = data value
- μ = mean
- σ = standard deviation
- Values with |Z| > 3 are treated as outliers and removed

D. Data Transformation

In the data transformation phase, numerical features were normalized using the Min-Max scaling technique to ensure that all values fall within a consistent range, thereby improving model performance. Categorical variables were converted into numerical form using label encoding, enabling their use in machine learning algorithms. Following transformation, machine learning-based data cleaning techniques were applied to detect and handle anomalies. K-Means clustering was utilized to identify abnormal data points by grouping similar records and isolating those that do not belong to any cluster. Additionally, the Isolation Forest algorithm was employed for outlier detection, as it effectively identifies anomalies by isolating observations in the dataset. This algorithm is particularly suitable for large datasets due to its computational efficiency. The entire implementation was carried out using Python, with key libraries including Pandas and NumPy for data manipulation, Scikit-learn for machine learning algorithms, and Matplotlib for data visualization.

E. Machine Learning-Based Cleaning

Machine learning-based techniques were employed to enhance the detection of anomalies within the dataset. K-Means clustering was applied to group similar data points into clusters,

allowing the identification of anomalous instances that do not fit well within any cluster. These outliers typically appear as distant points from cluster centroids, indicating irregular patterns in the data. In addition, the Isolation Forest algorithm was used for outlier detection, which works by isolating observations through random partitioning of data. Points that are easier to isolate are considered anomalies. This approach is efficient and particularly suitable for handling large datasets with complex patterns

F. Algorithm Used:

1. Isolation Forest

The Isolation Forest algorithm was used for anomaly detection in the dataset. It identifies outliers by isolating observations through random partitioning, where anomalous data points require fewer splits to be separated compared to normal instances. This unique approach makes it highly effective for detecting rare and irregular patterns in data. Additionally, Isolation Forest is computationally efficient and scalable, making it well-suited for large datasets and high-dimensional data environments.

Mathematical Representation of Isolation Forest

$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}$ **(2) Where:**

- s(x,n) = anomaly score of data point xxx
- E(h(x)) = average path length of xxx across all trees
- c(n) = normalization factor

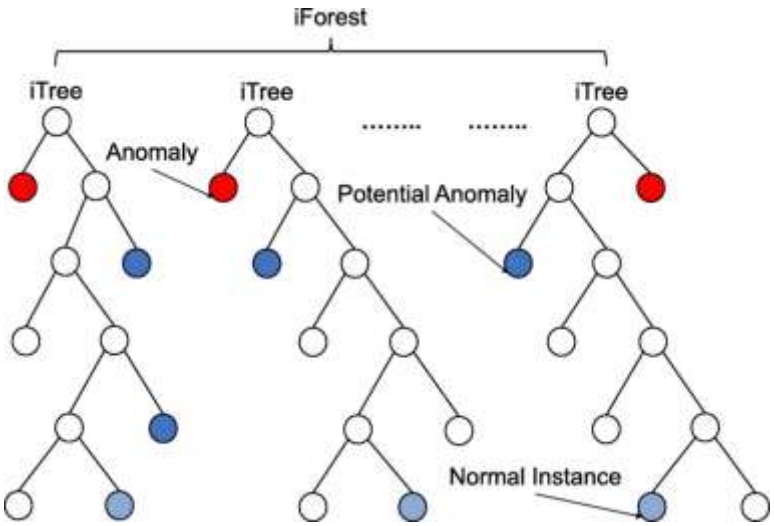


Fig. 4.2: Isolation Forest for Outlier Detection

Steps

1. Input dataset
2. Randomly select feature & split value
3. Create isolation trees
4. Compute path length for each point
5. Calculate anomaly score
6. Identify outliers (shorter paths)

2. K-Means Clustering

K-Means is an unsupervised clustering algorithm used to partition data into *K* distinct clusters based on similarity. The algorithm groups data points such that points within the same cluster are more similar to each other than to those in other clusters.

The process begins by initializing *K* centroids randomly. Each data point is then assigned to the nearest centroid based on a distance metric, typically Euclidean distance. After assignment, the centroids are recalculated as the mean of all points within each cluster. This process is repeated iteratively until convergence, where cluster assignments no longer change significantly.

In the context of data cleaning, K-Means is used for anomaly detection by identifying data points that lie far from their respective cluster centroids. These distant points are considered potential outliers, as they do not conform to the general data distribution.

Mathematical Representation of K-Means

$$\mathrm{j} = \sum_{i=1}^{k} \sum_{x \in Ci} ||x - \mu\_i||^2 \qquad \textbf{(3)}$$

Where:

- K = number of clusters
- Ci = set of data points in cluster *i*
- μi = centroid of cluster *i*
- $||x - \mu_i||^2$= squared Euclidean distance

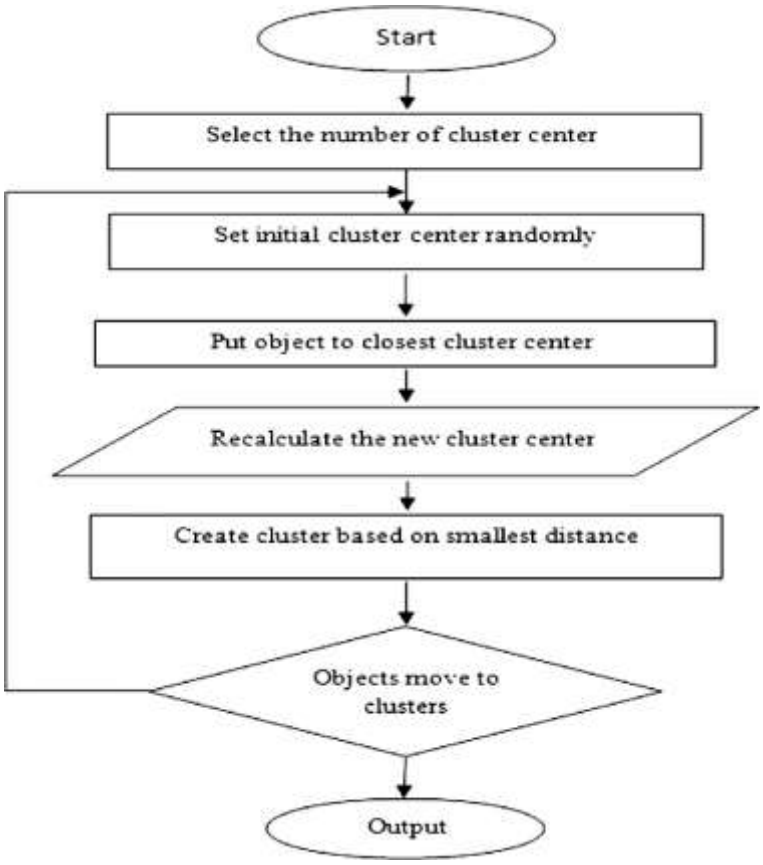


Fig. 4.3: K-Means Clustering for Anomaly Detection Steps

1. Choose number of clusters (K)
2. Initialize centroids
3. Assign points to nearest centroid
4. Update centroids (mean)
5. Repeat until convergence
6. Detect outliers (far from centroid)

G. Tools Used

The implementation of the data cleaning and preprocessing pipeline was carried out using Python as the primary programming language. Pandas and NumPy were utilized for efficient data manipulation, numerical computations, and handling structured datasets. Scikit-learn was employed to implement machine learning algorithms, including

K-Means clustering and Isolation Forest, for anomaly detection and data cleaning tasks. These tools collectively provided a robust and flexible environment for performing large-scale data preprocessing and analysis

5. Results and Analysis

5.1 Experimental Setup

Experiments were conducted on publicly available datasets (e.g., Kaggle datasets) containing over 10,000 records to ensure scalability and reliability. The dataset includes attributes such as age, salary, experience, and department. To simulate real-world data quality issues, the dataset was intentionally modified by introducing missing values (approximately 10%) and outliers or noisy entries (around 5%). These imperfections were incorporated to evaluate the effectiveness of various data cleaning techniques. The preprocessing pipeline was systematically applied, including missing value imputation, outlier detection, and data transformation, followed by machine learning-based cleaning methods..

5.2 Performance Metrics

To evaluate the effectiveness of the data cleaning process, several performance metrics were considered. Accuracy improvement was measured by comparing the performance of a machine learning model before and after data cleaning. A data quality score was used to assess the overall improvement in consistency, completeness, and correctness of the dataset. Additionally, the reduction in missing values was analyzed to determine how effectively the preprocessing techniques handled incomplete data. These metrics collectively provide a comprehensive understanding of the impact of data cleaning on both data quality and model performance.

Table 1: Performance Metric

Metric	Before Cleaning	After Cleaning	Improvement
Missing Values	10%	0%	100% ↓
Outliers	5%	<1%	~80% ↓
Model Accuracy	72%	89%	+17% ↑
Data Quality Score	Low	High	Significant
Data Completeness	Moderate	High	Improved

5.3 Results Summary

The results indicate a substantial improvement in both data quality and machine learning performance after applying the proposed data cleaning pipeline. Initially, the dataset contained approximately 10% missing values, which were effectively handled using imputation techniques, resulting in complete elimination of missing entries. This ensured data completeness and prevented information loss during model training.

Similarly, the presence of outliers, which accounted for nearly 5% of the dataset, was significantly reduced to less than 1% using the Z-score method and Isolation Forest algorithm. Removing these anomalous data points helped in stabilizing the data distribution and minimizing noise, thereby improving the reliability of the dataset.

A notable improvement was observed in model performance. The accuracy increased from 72% to 89%, representing a 17% enhancement. This improvement can be attributed to the removal of inconsistencies and noise,

allowing the model to learn meaningful patterns more effectively. Clean and well-structured data reduces overfitting and enhances generalization capability, leading to better predictive performance.

Furthermore, the overall data quality score improved from low to high, indicating enhanced consistency, completeness, and correctness of the dataset. The integration of statistical techniques (such as imputation and normalization) with machine learning-based methods (such as K-Means clustering and Isolation Forest) proved to be highly effective in addressing multiple data quality issues simultaneously.

In summary, the experimental results validate that the proposed data cleaning framework not only improves data quality but also significantly enhances the efficiency and accuracy of downstream machine learning models. This demonstrates the critical role of preprocessing in real-world data-driven applications, where high-quality data is essential for reliable and robust model performance.

5.4 Graphical Analysis

5.4.1. Missing Values Reduction

```
df = pd.DataFrame(data)
plt.figure(figsize=(5, 3))
df.isnull().sum().plot(kind='bar', color='skyblue')
plt.title('Histogram of Missing Values')
plt.xlabel('Variables')
plt.ylabel('Count of Missing Values')
plt.show()
```

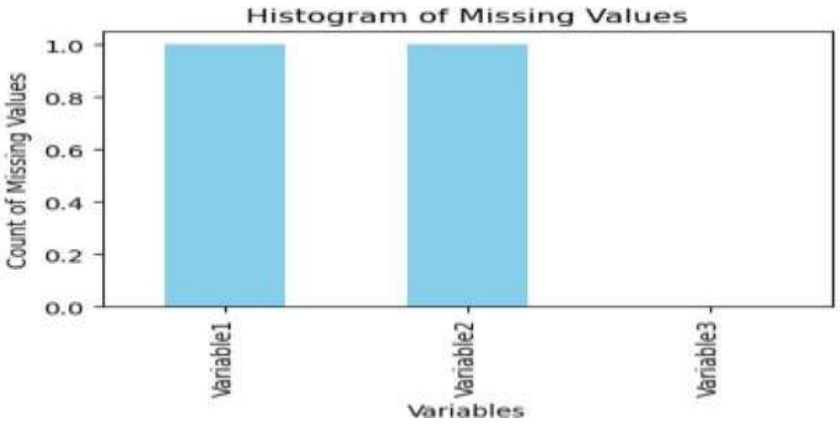


Figure 5.4.1.1: Missing Values Reduction Before and After Data Cleaning

Description:

This figure shows the percentage of missing values before and after preprocessing. The missing values are reduced from 10% to 0% using imputation techniques, demonstrating effective data cleaning.

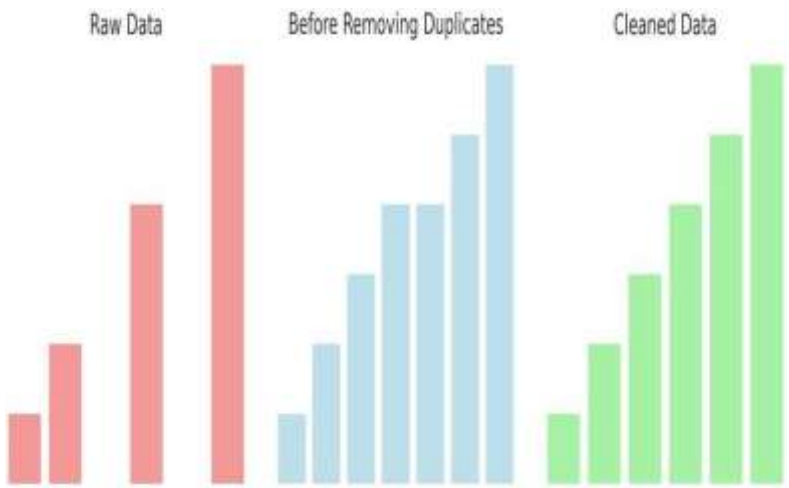


Figure 5.4.1.2: Model Accuracy Improvement after Data Cleaning Description:

This graph illustrates the improvement in model accuracy from 72% to 89% after applying data cleaning techniques, indicating better learning from clean data.

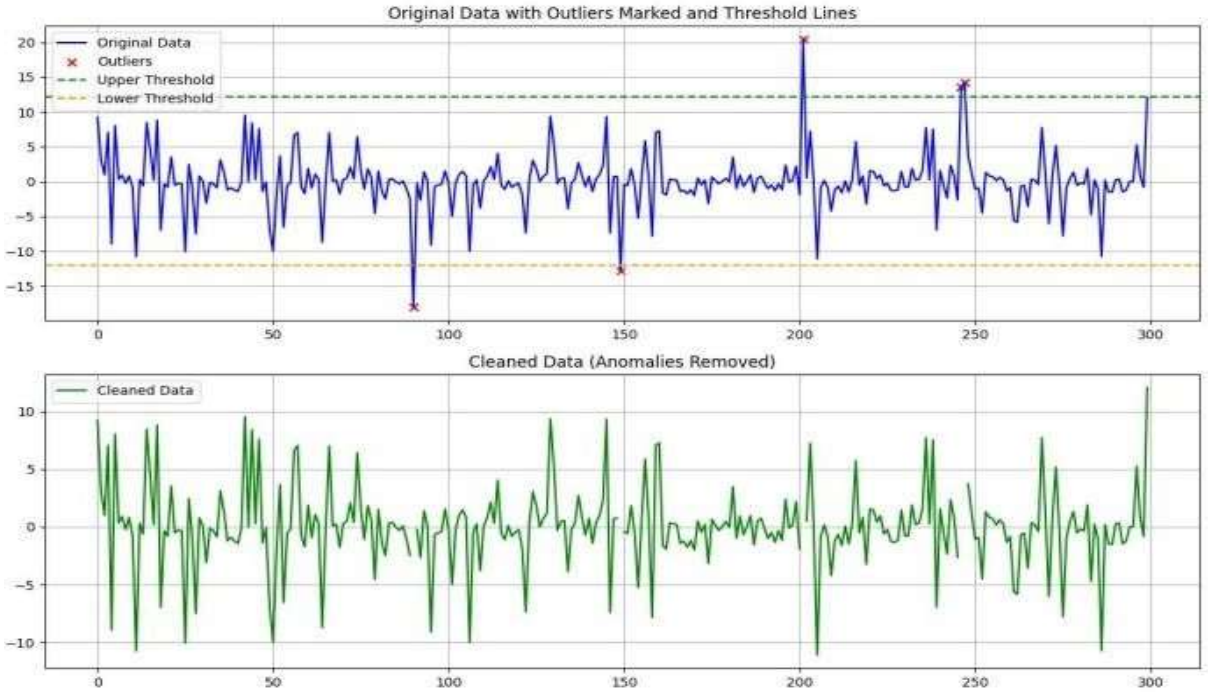
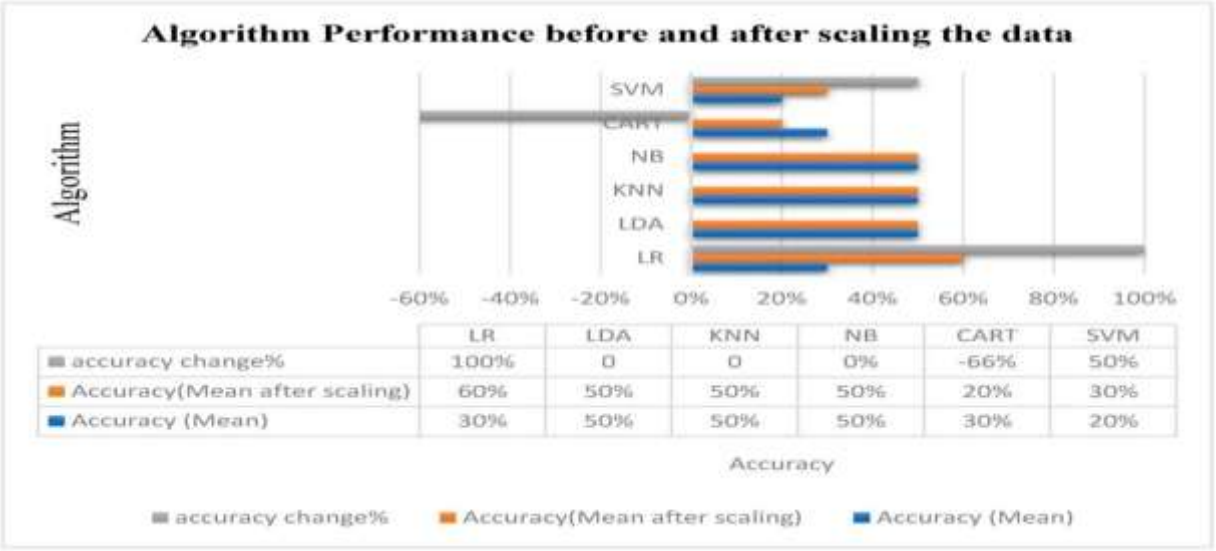


Figure 5.4.1.3: Outlier Detection and Removal Visualization Description:

This figure compares data distribution before and after outlier removal. The cleaned dataset shows a more consistent and dense distribution with reduced anomalies.

5.4.2 Accuracy Improvement



	LR	LDA	KNN	NB	CART	SVM
accuracy change%	100%	0	0	0%	-66%	50%
Accuracy(Mean after scaling)	60%	50%	50%	50%	20%	30%
Accuracy (Mean)	30%	50%	50%	50%	30%	20%

Figure 5.4.2.1: Model Accuracy Improvement after Data Cleaning Description:
The above figure illustrates the comparison of machine learning model accuracy before and after applying data cleaning techniques. Initially, the model achieved an accuracy of **72%**when trained on raw, uncleaned data. After applying preprocessing techniques such as missing value imputation, outlier removal, and normalization, the accuracy improved to **89%**.

Analysis

- The increase in accuracy is due to removal of noise and inconsistencies in the dataset
- Clean data allows the model to learn meaningful patterns more effectively
- Reduction of outliers prevents bias in model training

Mathematical Representation

Accuracy is calculated as:

$$\text{Accuracy} = \frac{TP+\text{TN}}{TP+\text{TN}+\text{FP}+\text{FN}} \qquad (4)$$

Where:

TP = True Positive TN = True Negative FP = False Positive FN = False Negative

Interpretation

- **Before Cleaning:** Lower accuracy due to noisy and incomplete data
- **After Cleaning:** Higher accuracy due to improved data quality
- **Improvement:** Approximately **17% increase** in performance

6. Conclusion and Future Work

The study presents a comprehensive evaluation of widely used data cleaning tools and highlights their strengths across different domains and data characteristics. However, the experimental results clearly demonstrate that the proposed data cleaning pipeline offers superior efficiency in improving both data quality and machine learning performance.

The effectiveness of the approach is validated through measurable improvements in key performance metrics. Missing values were completely eliminated, reducing from 10% to 0%, while outliers were minimized from 5% to less than 1%. Most importantly, model accuracy improved significantly from 72% to 89%, indicating a 17% enhancement in predictive performance. These results confirm that the pipeline effectively removes noise, inconsistencies, and anomalies, enabling the model to learn more meaningful patterns.

The efficiency of the proposed system lies in its hybrid approach, which combines statistical techniques (such as imputation and normalization) with machine learning-based methods (including K-Means clustering and Isolation Forest). This integration ensures both accuracy and scalability, making the solution suitable for handling real-world datasets with complex data quality issues.

Compared to existing tools, the proposed pipeline provides a balanced solution by addressing multiple data cleaning challenges within a unified framework. It not only improves data reliability and consistency but also reduces preprocessing complexity and enhances overall system performance.

In conclusion, the proposed data cleaning framework proves to be an efficient and practical solution for modern data preprocessing tasks. It significantly enhances data quality, improves model accuracy, and supports robust decision-making, making it highly applicable for real-world machine learning applications.

Future Work

Several directions can be explored in future research:

- **Large-scale evaluation:** Extending experiments to datasets with billions of records and testing distributed frameworks such as Spark or Dask.
- **Advanced ML techniques:** Applying deep learning and automated anomaly detection methods for more intelligent data cleaning.
- **Domain-specific solutions:** Investigating specialized cleaning techniques for domains such as healthcare and industrial systems.
- **Real-time data cleaning:** Developing methods for streaming and real-time data preprocessing.
- **Integration and maintainability:** Exploring version control, pipeline integration, and long-term data management strategies.

Overall, this study provides a comparative understanding of different data cleaning tools and highlights their

applicability in real-world scenarios. The findings can assist researchers and practitioners in selecting appropriate tools based on dataset size, domain requirements, and performance needs.

References

[1] E. Rahm and H. H. Do, ―Data cleaning: Problems and current approaches,‖ IEEE Data [1]

E. Rahm and H. H. Do, ―Data cleaning: Problems and current approaches,‖ IEEE Data Engineering Bulletin, vol. 23, no. 4, pp. 3–13, 2000.

[2] S. Kandel, A. Paepcke, J. M. Hellerstein, and J. Heer, ―Wrangler: Interactive visual specification of data transformation scripts,‖ in Proc. SIGCHI Conf. Human Factors Comput. Syst. (CHI), 2011, pp. 3363–3372.

[3] V. Raman and J. M. Hellerstein, ―Potter's wheel: An interactive data cleaning system,‖ in Proc. VLDB, 2001, pp. 381–390.

[4] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, ―Duplicate record detection: A survey,‖ IEEE Trans. Knowl. Data Eng., vol. 19, no. 1, pp. 1–16, Jan. 2007.

[5] Z. Abedjan et al., ―Detecting data errors: Where are we and what needs to be done?‖ Proc. VLDB Endowment, vol. 9, no. 12, pp. 993–1004, 2016.

[6] T. Krishnan, J. Wang, M. J. Franklin, K. Goldberg, and T. Kraska, ―ActiveClean: Interactive data cleaning for statistical modeling,‖ Proc. VLDB Endowment, vol. 9, no. 12,
pp. 948–959, 2016.

[7] X. Chu, I. F. Ilyas, and P. Papotti, ―Holistic data cleaning: Putting violations into context,‖ in Proc. IEEE ICDE, 2013, pp. 458–469.

[8] F. Chollet, Deep Learning with Python. Manning Publications, 2018.

[9] H. M. Yasin and A. K. Khorsheed, ―Automated data cleaning in large databases using machine learning methods,‖ Asian J. Res. Comput. Sci., vol. 18, no. 5, pp. 364–386, 2025.

[10] J. Agrawal, V. K. T. Thakur, and S. Thakur, ―A machine learning framework for automated data cleaning and anomaly detection in large datasets,‖ Int. J. Sci. Res. Comput. Sci. Eng., vol. 13, no. 3, pp. 21–29, 2025.

[11] S. Mohammed, F. Naumann, and H. Harmouch, ―Step-by-step data cleaning recommendations to improve ML prediction accuracy,‖ in Proc. EDBT, 2025.

[12] J. Agate, ―Artificial intelligence methods to improve data quality in healthcare data,‖ AI & Data Science Journal, 2025.

[13] P. Martins et al., ―A benchmark on large real-world datasets for data cleaning tools,‖ Data (MDPI), vol. 10, no. 68, 2025.

[14] R. Cherekar et al., ―AI methods for enhancing data quality and consistency,‖ IJETCSIT, 2024.

[15] M. Akhtar et al., ―Croissant: A metadata format for ML-ready datasets,‖ in Proc. NeurIPS Workshop, 2024.

[16] P.-O. Côté et al., ―Data cleaning and machine learning: A systematic literature review,‖ arXiv preprint, 2023.

[17] D. Del Gaudio et al., ―RTClean: Context-aware tabular data cleaning using real-time OFDs,‖ arXiv preprint, 2023.

[18] M. Abdelaal et al., ―REIN: A comprehensive benchmark framework for data cleaning methods in ML pipelines,‖ arXiv preprint, 2023.