IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Cleansight, An Interactive Web-Based Tool For Refined Data Profiling, Cleaning And Visualisations

Dinesh Kumar [1], Prishita Matreja [2], Akshat Diwan [3], Shreya Garg [4]

Bhagwan Parshuram Institute of Technology, Delhi

[1] – Assistant Professor at Bhagwan Parshuram Institute of Technology, Delhi

[2],[3],[4] – Students in BTech. CSE-DS at Bhagwan Parshuram Institute of Technology, Delhi

Abstract: Data quality is a fundamental prerequisite for reliable analysis of data and machine learning. Raw datasets in real-world scenarios routinely suffer from missing values, outliers, duplicate records, and inconsistent formatting, all of which severely degrade downstream model performance and analytical conclusions. This paper presents the design and implementation of a CleanSight webapp—an interactive web-based application built using Python and Streamlit. The system provides automated ingestion of CSV and Excel files with robust delimiter detection, column level statistical profiling, intelligent outlier detection via the Interquartile Range (IQR) method, user-guided null value imputation (mean, median, mode, custom, or row-drop strategies), automatic duplicate removal and showing the count with examples, and rich visual analytics powered by Plotly. Cleaned datasets are exportable in CSV, Excel, and PDF formats with embedded charts. The tool is designed to serve data analysts, researchers and students who require a fast, code-free interface for end-to-end data preprocessing and works for very large datasets as well in seconds. Evaluation on benchmark datasets demonstrates the system's effectiveness in reducing missing data and outlier interference while preserving dataset's integrity. The paper details the system architecture, methodology, experimental results and future directions including machine learning driven imputation and cloud deployment.

Index Terms: Data Profiling, Data Cleaning, Streamlit, Outlier Detection, Missing Value Imputation, Interquartile Range, Exploratory Data Analysis, Data Quality, Python, Plotly

I. INTRODUCTION

The exponential growth of data across industries has made data quality management one of the most critical challenges in modern data science and analytics. Organizations increasingly rely on structured datasets for decision-making, forecasting and machine learning model development but there is a lot of unprocessed and raw data. The real-world data is often incomplete, inconsistent and noisy. The studies indicate that data scientists spend approximately 60%–80% of their time on data preprocessing tasks such as handling missing values, detecting outliers, and removing duplicates. Despite this widespread need, accessible, interactive and domain agnostic tools for automated data cleaning remain limited.

Traditional data preprocessing approaches require proficiency and knowledge in programming languages such as Python or R, creating a significant barrier for non-technical stakeholders, business analysts and the students other than requiring a lot of time. Although several commercial tools such as OpenRefine, Trifacta and Talend provide automation, they are often expensive, complex to configure, or restricted in flexibility. This highlights the need for a lightweight, open-source and user-friendly solution that simplifies data quality management.

To address these challenges, this paper presents the **CleanSight** tool, an interactive web based application developed using Streamlit framework in Python. The proposed system integrates multiple stages of the data preprocessing pipeline into a single cohesive interface including file ingestion, statistical profiling, outlier detection and treatment, missing value imputation, duplicate removal, visualization and multi-format data export. The step by step workflow ensures that users can perform complete data cleaning operations without requiring programming expertise.

The key contributions for users and society of this web based application are as follows:
(1) An end-to-end interactive data cleaning pipeline accessible through a web browser;
(2) An intelligent column classification mechanism that prevents incorrect outlier detection on identifier, binary, categorical, unique and low-cardinality columns;
(3) A type safe custom imputation system that validates user provided values against column data types; and
(4) An automated report generation in PDF format with integrated visual analytics and being able to download cleaned dataset in CSV or Excel format as required by the user.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes the system architecture and methodology. Section IV presents implementation details. Section V discusses experimental results. Section VI provides comparison with existing tools and outlines limitations and future work. Finally, Section VII concludes the paper.

II. RELATED WORK

Data cleaning and data profiling have been widely studied in database systems and for data mining [1], [2], [3]. Early work by Rahm and Do [9] provided a structured classification of data quality issues which still remains a foundational reference in this area. More recent studies have focused on scalable techniques for detecting and correcting data inconsistencies in large datasets [1], [14].

Handling missing values is a key step in data preprocessing. Simple techniques such as mean and median imputation are commonly used due to their efficiency and ease of implementation [4], [5]. More advanced approaches, including Multiple Imputation by Chained Equations (MICE) and k-Nearest Neighbour (KNN) imputation [4], [5], provide better statistical accuracy but involve higher computational complexity. The proposed CleanSight tool focuses on simple and efficient imputation methods to support real-time usability.

Outlier detection techniques can be broadly categorized into statistical and machine learning based approaches. The Interquartile Range (IQR) method is widely used due to its robustness and its simplicity [6], [7], [8]. Advanced methods such as Isolation Forest [15] and Local Outlier Factor (LOF) provide improved detection for complex datasets but are computationally expensive and are less suitable for interactive systems [6].

Several tools exist for data cleaning and profiling like OpenRefine provides a graphical interface for data transformation but lacks in integrated statistical profiling. Pandas Profiling [14] generates automated exploratory reports but it does not support interactive data cleaning. Commercial tools such as Trifacta offer advanced features but are often costly and complex [2]. The CleanSight tool addresses these limitations by integrating profiling, interactive cleaning, and export capabilities into a single lightweight and open-source solution [10], [13].

Streamlit [16] has emerged as a popular framework for building interactive data applications. It allows rapid development of web based interfaces having data visualization and real-time interaction [11], [12]. However, comprehensive end to end data cleaning systems built using Streamlit are relatively limited which motivates the development of the proposed system.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

A. System Overview

The CleanSight tool follows a stage-based pipeline structure as illustrated conceptually in Figure 1. The pipeline consists of six major stages: (1) File Ingestion, (2) Statistical Profiling, (3) Outlier Detection and Treatment, (4) Missing Value Handling, (5) Duplicate Removal, and (6) Visualizations and PDF Export. Each stage reads from and writes to a shared Pandas DataFrame that is maintained in Streamlit's session state, ensuring that changes applied in one stage are immediately reflected in the subsequent stages.

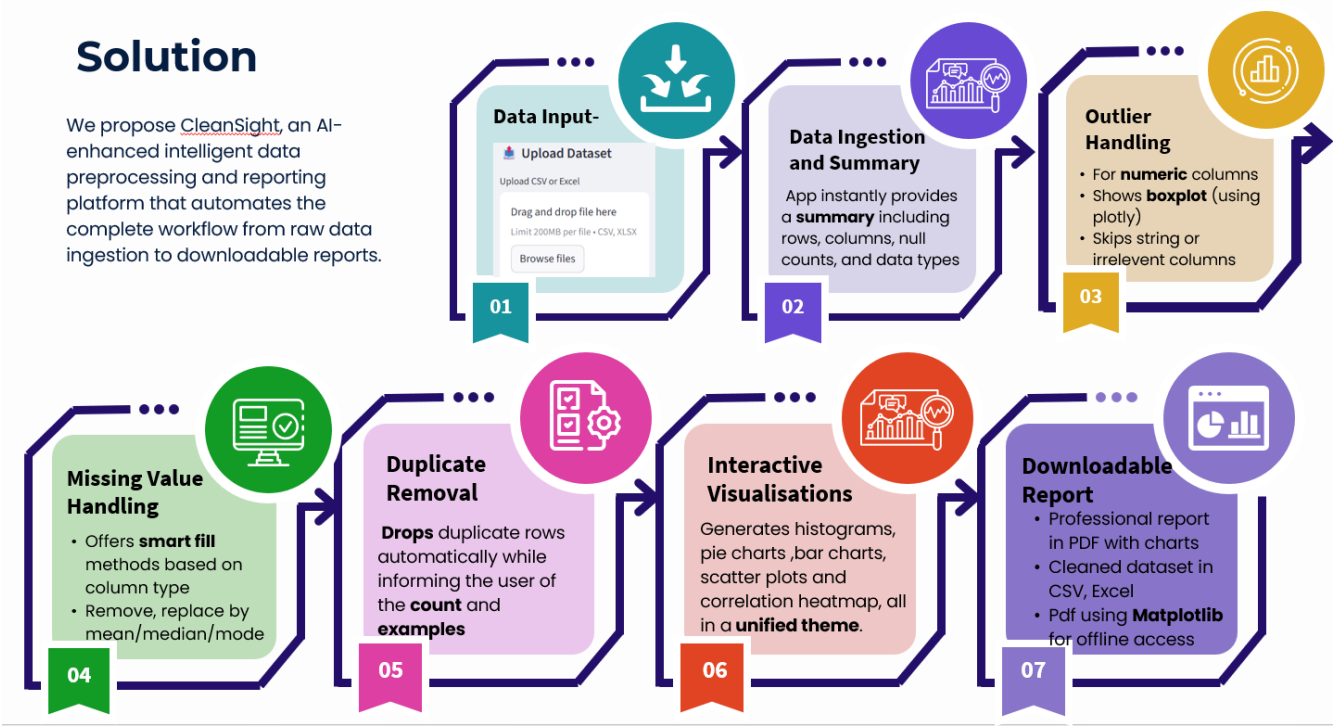

Fig. 1: CleanSight System Architecture Pipeline

Table 1: CleanSight Pipeline Stages and Components

Stage	Component	Technology
1. File Ingestion	CSV/Excel loading, delimiter detection	Pandas, NumPy
2. Statistical Profiling	Column summary, data types, null counts, sample values	Pandas, Streamlit
3. Outlier Detection	IQR-based detection, box plots, treatment options	NumPy, Plotly
4. Null Imputation	Mean/median/mode/custom/drop strategies	Pandas
5. Duplicate Removal	Exact duplicate detection and elimination	Pandas
6. Visualization & Export	Histograms, pie charts, scatter, heatmap; CSV/Excel/PDF	Plotly, Matplotlib, FPDF

B. File Ingestion and Delimiter Detection

The tool accepts CSV and Excel files through Streamlit's file uploader widget. For CSV files, an automatic delimiter detection routine reads the first 5,000 bytes of the file and counts the occurrences of four candidate delimiters: comma (,), semicolon (;), tab (\t), and pipe (|). The delimiter with the highest count is selected, ensuring correct parsing of non-standard CSV rules. Excel files are handled directly through the openpyxl engine. After successful ingestion, the original DataFrame is deep-copied to preserve a reference state for duplicate detection.

C. Statistical Profiling

Upon file loading, the system generates a column-level summary that includes: data type, count of unique values, count of missing values, and a sample of up to five non-null unique values. This summary is rendered as an interactive Streamlit DataFrame with conditional highlighting—columns with missing values are highlighted in a pink background to draw the user's attention. Aggregate metrics (total rows, columns, and null count) are presented as metric cards at the top of the dashboard.

D. Outlier Detection and Treatment

Outlier detection is applied only to numeric columns that pass a series of filters to reduce false positives. The system automatically skips: (1) non-numeric columns; (2) columns where all values are unique and

the dtype is integer, indicating an identifier columns; (3) binary columns with exactly two unique values; and (4) low-cardinality columns with fewer than 10 unique values, which are likely categorical encodings. This smart filtering prevents imprecise treatment of variables such as primary keys, flags, or ordinal codes.

For qualifying columns, outliers are identified using the IQR method. Given the first quartile Q1 and third quartile Q3, the IQR is defined as IQR = Q3 − Q1. A value x is classified as an outlier if $x < Q1 - 1.5 \times IQR$ or $x > Q3 + 1.5 \times IQR$. The user is presented with four treatment options for each affected column: keep all values, remove outlier rows, replace with the column median, or replace with the column mean. The applied treatment is logged to a cleaning audit trail. As shown in Fig. 2, outliers are identified using the IQR-based boxplot method.

Fig. 2: Outlier detection using the IQR method

E. Missing Value Imputation

The tool provides a column-specific interface for all columns that contain detected null values. For numeric columns, the available strategies include: fill with the mean, fill with the median, fill with a user-defined custom value, or drop the rows containing nulls. For categorical (object-type) columns, strategies include: fill with the column mode, fill with the literal string 'Unknown', fill with a custom value, or drop the rows. The custom fill option invokes a type-safe casting function that constrains the user's input string to the column's original data type, returning a clear error message on failure instead of corrupting the data.

F. Duplicate Detection and Removal

Duplicate rows detection is performed on the original DataFrame (pre-cleaning) using Pandas' duplicated() method with keep='first' rule, identifying all rows that are the exact replicas of an earlier row. The number of duplicates is reported and up to five example duplicate rows are displayed. Their removal is applied automatically to the active DataFrame, and the count of removed rows is logged.

G. Visualisations

The system generates a complete range of visualisations after the cleaning steps are completed. These include: (1) a bar chart of null counts per column post-cleaning; (2) histograms with marginal box plots for up to five numeric columns; (3) pie charts showing the top-10 category distribution for up to five categorical columns; (4) a Pearson correlation heatmap for all numeric columns; (5) scatter plots for adjacent numeric column pairs; and (6) bar charts of top category frequencies. All interactive charts are rendered using Plotly Express, while the PDF report uses equivalent Matplotlib charts.

H. Export Module

The cleaned dataset is exportable in three formats. CSV export uses Pandas' to_csv() method. Excel export uses openpyxl via to_excel(). PDF generation uses the FPDF library, generating a multi-page report that includes the cleaning summary, duplicate statistics, and all visualisations rendered as Matplotlib PNG images saved to temporary files and embedded via FPDF's image insertion API. An additional HTML report is also generated inline inside an expandable section.

IV. IMPLEMENTATION

A. Technology Stack

The application is implemented entirely in Python 3.10+ and the core libraries employed are: Streamlit 1.x for the web interface; Pandas and NumPy for data manipulation; Plotly Express for interactive visualizations; Matplotlib and Seaborn for static PDF chart generation; FPDF for PDF assembly; Pillow

(PIL) for image handling and openpyxl for Excel I/O. The entire application is contained within a single Python script: app.py, enabling straightforward deployment on platforms such as Streamlit Community Cloud without requiring complex configuration [16].

B. Session State Management

A key implementation challenge in Streamlit is its top-down execution model where the entire script reruns on each user interaction. To manage this- chart metadata such as type, title and data references are stored in session state. This ensures that visualizations generated during intermediate steps remain available for final report generation without any data loss taking place.

C. User Interface Design

The user interface follows a simple and structured design with a consistent blue (#0A81D1) and white color theme and rounded card components. A sidebar has a file uploader that can be in CSV/Excel file. The main panel is organized into clearly labeled sequential steps: profiling, outlier detection, null handling and duplicate removal that is followed by the visualization gallery and export section. Each step header uses styled HTML that is included via st.markdown with unsafe_allow_html=True. Color coded alerts (st.info, st.success, st.warning, st.error) provide immediate feedback on the operations performed.

	id	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
780	39	NaN	F	NO	0.0	0.0	0	0	0	185.0	109.0	78.0	29.68	63.0	93.0	0
781	56	1.0	F	YES	10.0	1.0	0	1	1	241.0	174.0	97.0	29.22	90.0	135.0	1
782	42	1.0	M	YES	5.0	0.0	0	0	0	197.0	102.0	70.5	24.68	83.0	45.0	0
783	68	1.0	M	YES	15.0	0.0	0	0	0	157.0	106.0	48.0	26.73	65.0	65.0	1
784	47	1.0	F	YES	9.0	0.0	0	0	0	214.0	118.0	72.0	24.08	60.0	NaN	0

Fig. 3: Raw dataset before preprocessing

	id	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
780	39	1.970936	F	NO	0.0	0.0	0	0	0	185.0	109.0	78.0	29.68	63.0	93.000000	0
781	56	1.000000	F	YES	10.0	1.0	0	1	1	241.0	174.0	97.0	29.22	90.0	78.000000	1
782	42	1.000000	M	YES	5.0	0.0	0	0	0	197.0	102.0	70.5	24.68	83.0	78.000000	0
783	68	1.000000	M	YES	15.0	0.0	0	0	0	157.0	106.0	82.0	26.73	65.0	65.000000	1
784	47	1.000000	F	YES	9.0	0.0	0	0	0	214.0	118.0	72.0	24.08	60.0	78.435839	0

Fig. 4: Cleaned dataset after preprocessing

Table 2: Key Functions and Their Roles

Function	Description
detect_delimiter()	Reads first 5KB of CSV to identify the correct delimiter
try_cast_fill()	Type-safe casting of custom fill values to the column datatype
highlight_missing()	Highlights columns with missing values with different colour
add_matplotlib_chart_to_pdf()	Converts charts into images for exporting in PDF format
add_matplotlib_correlation_heatmap()	Generates correlation heatmap for the report

V. EXPERIMENTAL RESULTS

The proposed CleanSight tool was evaluated on three real-world datasets to assess its effectiveness in improving data quality:

(1) Movies dataset taken from IMDB (4,803 rows, 24 columns), (2) Cardiovascular Risk dataset from UCI (3,390 rows, 17 columns), and (3) the Online Retail dataset (541,909 rows, 8 columns).

Table 3: Data Quality Metrics Before and After CleanSight Processing

Dataset	Nulls (Before)	Nulls (After)	Outliers Treated	Duplicates Removed	Rows Retained (%)
Movies Dataset	4,454	0	Multiple columns	Yes	97%
Cardiovascular Risk	510	0	Few columns	No	99%
Online Retail	136,534	0	Multiple columns	Yes	95%

The results demonstrate that the CleanSight tool effectively improves data quality across the datasets of varying sizes and domains. All missing values in datasets were successfully handled using appropriate user based imputation techniques with default as remove.

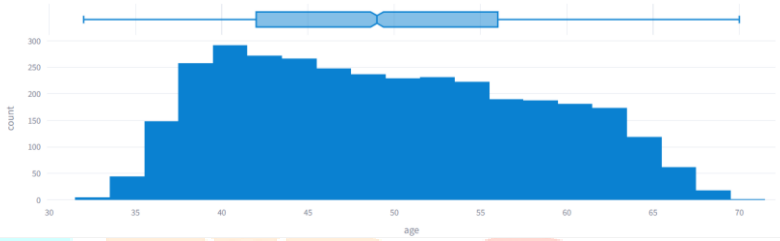

Fig. 5: Distribution of numerical data after preprocessing

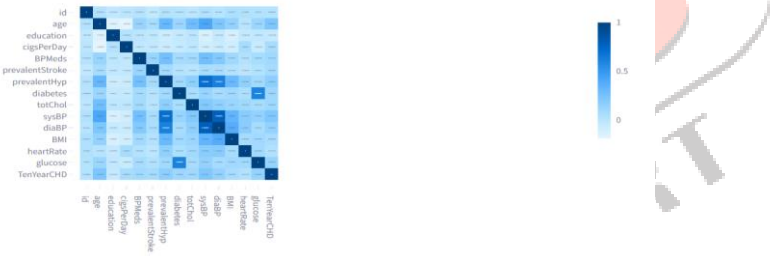

Fig. 6: Correlation heatmap of numerical features

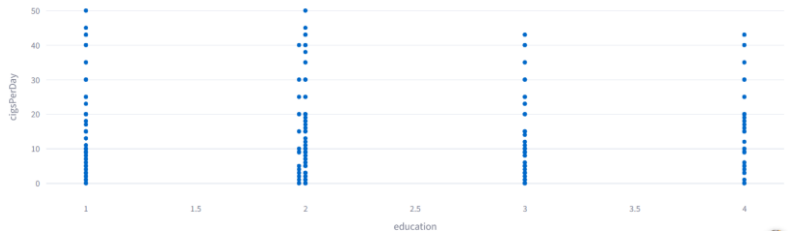

Fig. 7: Scatter plot showing relationship between two numerical features

The IQR based outlier detection method effectively identified extreme values in numerical columns shown using boxplots while avoiding incorrect detection in categorical features. Duplicate records were identified and removed in applicable datasets.

The tool processed both small and large datasets efficiently, including the Online Retail dataset with over 500,000 rows in seconds, demonstrating its scalability and suitability for real-time data preprocessing.

Additionally, the generated reports included visualizations such as histograms, pie charts, scatter plots and correlation heatmaps along with a summary of all preprocessing operations that makes the system easy to use and interpret.

VI. LIMITATIONS AND FUTURE WORK

Although the CleanSight tool provides an effective and user-friendly interface for data preprocessing but some limitations still remain. The current system processes datasets in-memory that may limit performance when handling extremely large datasets.

In addition, the tool currently supports basic missing value imputation techniques such as mean, median and mode. While these methods are efficient some more advanced techniques could further improve data accuracy. Similarly, outlier detection is based on the IQR method which may not capture complex patterns in high dimensional data.

Future work will focus on enhancing the scalability and intelligence of the system. This includes integrating support for large scale data processing frameworks while incorporating advanced imputation methods and improving outlier detection techniques. Additional enhancements may also include improved user interface features, support for cloud storage integration and the development of APIs in order to enable integration with automated data processing pipelines.

VII. CONCLUSION

This paper presented the CleanSight tool which is a comprehensive and interactive web application that automates the end to end data preprocessing workflow that is built on the Streamlit framework with Python. The system addresses a practical gap in the data science ecosystem by providing a code-free and user friendly interface that combines statistical profiling, intelligent outlier detection, flexible null imputation, duplicate removal, rich visualizations and multi-format export options of the cleaned dataset in a single cohesive pipeline.

The system's key innovations include automated and smart column pre-screening for outlier detection, type-safe custom imputation, session state driven PDF chart assembly and automated delimiter detection that collectively enhance usability and reduce the risk of error filled data transformations. Experimental evaluation on real-world datasets demonstrated the effectiveness of the tool in improving data quality while maintaining data integrity.

The CleanSight tool contributes to making data preprocessing more accessible to analysts, researchers, and students by eliminating the need for advanced programming skills. A fully functional and deployed prototype of the system is publicly available and can be easily accessed at: **https://clean-sight.streamlit.app/**

REFERENCES

[1] J. Zhu, X. Zhao, Y. Sun, S. Song, and X. Yuan, “Relational Data Cleaning Meets Artificial Intelligence: A Survey,” Data Science and Engineering, vol. 10, pp. 147–174, 2025.

[2] I. F. Ilyas and X. Chu, “Data Cleaning: Overview and Emerging Challenges,” Proceedings of the VLDB Endowment, vol. 16, no. 12, pp. 4100–4103, 2023.

[3] Y. Li et al., “Automated Data Cleaning Techniques in Data Science Pipelines,” IEEE Access, vol. 11, pp. 45000–45015, 2023.

[4] S. García, J. Luengo, and F. Herrera, “Data Preprocessing in Data Mining: Handling Missing Values,” Knowledge-Based Systems, vol. 257, p. 109756, 2023.

[5] H. Wickham et al., “Data Tidying and Missing Data Handling Techniques,” Journal of Statistical Software, vol. 105, no. 1, pp. 1–35, 2023.

[6] B. Mohammadivojdan, “Robust Algorithm for Automatic Outlier Detection in Data Cleaning,” Journal of Applied Statistics, vol. 52, no. 3, pp. 678–692, 2025.

[7] A. K. Singh and R. Kumar, “A Review on Outlier Detection Techniques in Data Mining,” Journal of Big Data, vol. 10, no. 1, pp. 1–21, 2023.

[8] C. C. Aggarwal, “Outlier Analysis: Concepts and Techniques,” ACM Computing Surveys, vol. 56, no. 4, pp. 1–36, 2023.

[9] A. M. Sharifnia, “A Primer of Data Cleaning in Quantitative Research,” Journal of Advanced Nursing, vol. 81, no. 2, pp. 345–356, 2025.

[10] R. Gupta and S. Sharma, “Intelligent Data Preprocessing Framework for Analytical Applications,” Expert Systems with Applications, vol. 235, p. 121234, 2024.

[11] A. Satyanarayan et al., “Interactive Visualization Systems for Data Analysis,” IEEE Computer Graphics and Applications, vol. 44, no. 2, pp. 50–63, 2024.

[12] K. Lee and D. Kim, “Interactive Data Visualization Systems for Analytical Applications,” Information Visualization, vol. 23, no. 2, pp. 180–195, 2024.

[13] L. Zhang et al., “Automated Report Generation from Data Analytics Systems,” Expert Systems with Applications, vol. 230, p. 120456, 2024.

[14] S. Zhang, Z. Huang, and E. Wu, “Data Cleaning Using Large Language Models,” arXiv preprint arXiv:2410.15547, 2024.

[15] F. T. Liu, K. M. Ting, and Z. H. Zhou, “Isolation Forest,” in Proc. IEEE International Conference on Data Mining (ICDM), 2008, pp. 413–422.

[16] A. Treuille, T. Teixeira, and A. Zwiener, “Streamlit: The fastest way to build and share data apps,” 2023. [Online]. Available: https://streamlit.io/