JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

AutoReport AI: An Intelligent Automated Reporting System Using Data Analytics and Machine Learning

Harshita Tiwari[1], Samriddhi Srivastava[2], Sandhya Sharma[3], Mr. Jay Chand[4]

[1],[2],[3] B.Tech, Computer Science and Engineering, Kamla Nehru Institute of Physical and Social Sciences, Uttar Pradesh, India

[4]Project Guide (Assistant Professor, Computer Science and Engineering), Kamla Nehru Institute of Physical and Social Sciences, Uttar Pradesh, India

Corresponding author: [Insert first author's active email address]

Abstract

Organizations and individuals frequently use spreadsheet files to maintain sales, inventory, employee, and operational records. However, converting raw spreadsheets into reliable summaries and reports can require repetitive manual work and may lead to inconsistent data handling. This paper presents AutoReport AI, a prototype web-based system designed to automate common stages of spreadsheet analysis and reporting. The system accepts CSV and Excel files, profiles uploaded data, identifies missing values and duplicate records, provides configurable cleaning operations, computes descriptive statistics, and generates interactive visualizations. It also incorporates an unsupervised anomaly-detection component based on the Isolation Forest algorithm to flag potentially unusual observations in numeric data. Users can download processed datasets and report outputs in CSV, Excel, and PDF formats. The prototype is implemented in Python using Pandas for tabular data processing, Streamlit for the user interface, Plotly for interactive visualization, scikit-learn for anomaly detection, and ReportLab for PDF generation. The paper describes the system architecture, processing workflow, implementation choices, and a functional demonstration using a small sample spreadsheet. The prototype is intended as a decision-support aid; detected anomalies require human review and should not be treated as proof of error or fraud.

Keywords: Automated Reporting, Data Analytics, Python, Spreadsheet Processing, Data Cleaning, Isolation Forest, Anomaly Detection, Streamlit.

1. Introduction

Tabular data is widely used in routine organizational activities. Small businesses, academic teams, and departments often store information in CSV or Excel files and prepare summaries manually. The process may involve checking missing entries, removing repeated rows, calculating totals and averages, applying filters, preparing charts, and exporting reports. When these steps are repeated across files, they consume time and can introduce avoidable inconsistencies.

AutoReport AI is proposed as a lightweight application that brings these activities into a single workflow. A user uploads a spreadsheet, inspects its structure and quality, chooses suitable cleaning actions, explores summaries and charts, and exports the processed information. A machine-learning component adds an anomaly-screening step for numeric columns. The system focuses on making routine analysis accessible through a browser-based interface rather than requiring users to write Python code.

The term “AI” in the project name refers specifically to the inclusion of an unsupervised anomaly-detection method. The current prototype does not claim to generate natural-language explanations or automatically make business decisions. It presents data-derived indicators that users must interpret in context.

2. Problem Statement and Objectives

The problem addressed is the effort required to repeatedly transform raw spreadsheet files into cleaned, summarized, visual, and shareable reports. A useful tool should combine these tasks while keeping the cleaning choices visible to the user.

- Accept commonly used CSV and Excel input files.

- Display dataset dimensions, column information, and basic data-quality indicators.
- Identify missing values and duplicate rows and provide user-selectable cleaning operations.
- Allow users to filter categorical columns and inspect descriptive statistics for numeric columns.
- Generate interactive charts to support exploratory analysis.
- Use Isolation Forest to flag potentially anomalous records when sufficient numeric data is available.
- Provide downloadable outputs in CSV, Excel, and PDF formats.

3. Related Work

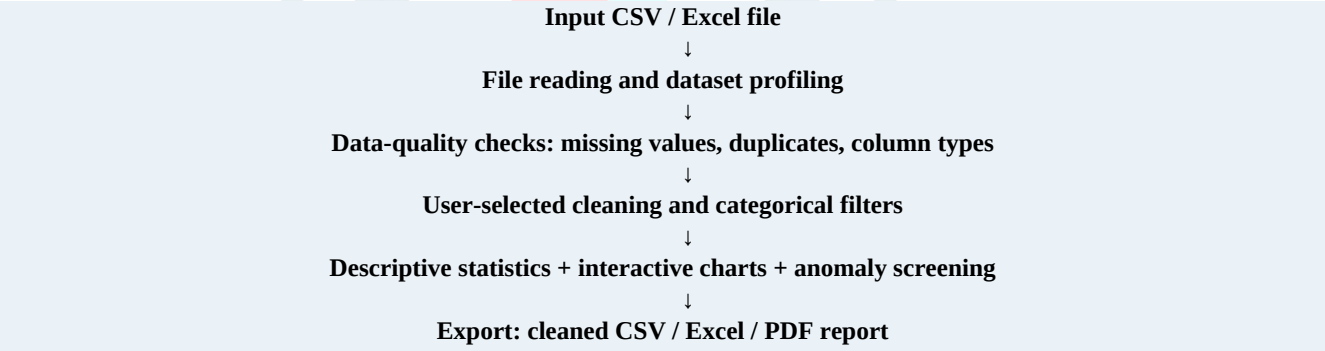
Python has become a common environment for scientific and data-oriented programming. McKinney described Pandas as a set of data structures and tools intended to make practical data analysis in Python more effective [1]. This makes Pandas suitable for operations such as reading tabular files, selecting columns, handling missing values, removing duplicates, and calculating descriptive summaries.

Anomaly detection is the task of identifying observations that differ significantly from the majority of a dataset. Liu, Ting, and Zhou introduced Isolation Forest, which isolates observations through randomly selected features and split values; observations that are isolated with shorter paths are more likely to be anomalous [2]. The method is useful in settings where labelled examples of anomalies may not be available. The scikit-learn implementation exposes Isolation Forest as an estimator for identifying outliers [3].

AutoReport AI combines these established capabilities into a single end-user workflow. Its contribution is primarily an applied software integration: file ingestion, data-quality inspection, user-controlled cleaning, summary analysis, visualization, anomaly screening, and report export are presented in one browser-accessible prototype.

4. Proposed System and Architecture

The system follows a sequential processing architecture. The uploaded file is first read into a tabular data structure. A profiling stage calculates the number of rows and columns and inspects missing values, duplicates, data types, and numeric columns. The user can then apply selected cleaning operations. The resulting dataset is used by the filtering, statistical-summary, visualization, and anomaly-detection components. Finally, the application prepares downloadable files.



5. Methodology

5.1 File Upload and Data Ingestion

The application provides a file-upload control for CSV and Excel workbooks. After upload, the file is loaded into a Pandas DataFrame, which serves as the main in-memory representation for subsequent operations. Unsupported file types and unreadable files should be handled with a clear error message. For large workbooks, practical deployment should impose file-size and processing limits.

5.2 Data Profiling and Quality Assessment

The profiling stage reports the dataset shape and column-level information. Missing-value counts are calculated for each column, while duplicate detection identifies repeated rows. Numeric columns are separated from categorical or other columns so that appropriate summaries and charts can be offered. These checks describe observable data characteristics; they do not determine whether a value is factually correct.

5.3 Data Cleaning

Cleaning is configurable rather than silently applied. The prototype provides options to remove duplicate rows, replace missing values with zero, or drop rows containing missing values. These operations have different meanings and should be selected according to the dataset. In particular, replacing a missing value with zero is appropriate only when zero is a meaningful substitute; otherwise, it can distort averages, totals, and downstream analysis. The original uploaded file should be retained by the user so that transformations remain reversible.

5.4 Filtering and Descriptive Analysis

Categorical filters allow the user to select values of interest and inspect a subset of the dataset. For numeric columns, the application displays descriptive statistics such as count, mean, standard deviation, minimum, quartiles, and maximum. These summaries provide a compact view of scale, spread, and possible data-entry issues.

5.5 Interactive Visualization

Plotly is used to present interactive charts for numeric and categorical data. Depending on the selected columns and data types, visualizations can help users compare categories, inspect distributions, and explore relationships. A chart should always be interpreted with its axis labels, units, filters, and underlying record count in view.

5.6 Anomaly Detection Using Isolation Forest

The prototype uses the Isolation Forest estimator from scikit-learn on numeric features. Isolation Forest constructs randomized partitioning trees. An observation that is isolated after fewer splits tends to receive a stronger anomaly indication [2, 3]. The current implementation uses a contamination setting of 0.1 and a fixed random state of 42, as configured in the prototype. This setting is a model assumption about the expected proportion of unusual observations, not a measured real-world anomaly rate. The model is run only when the dataset contains at least five records and usable numeric columns.

An anomaly flag is a screening signal, not a verdict. Legitimate high-value transactions, seasonal peaks, rare employee records, or unusual inventory movements may be flagged. Users should review flagged rows against domain knowledge before taking action.

5.7 Report and Data Export

The application provides export options for processed data in CSV and Excel formats and a PDF report generated with ReportLab. Exporting allows users to retain the cleaned dataset or share a compact summary. Before distribution, users should verify that the report includes an appropriate reporting period, units, filters, and any assumptions used during cleaning.

6. Technology Stack and Implementation

Technology	Role in the prototype
Python	Core programming language
Streamlit	Browser-based interactive application interface
Pandas	CSV/Excel reading, tabular transformations, profiling, and summaries
NumPy	Numerical operations where required
Plotly	Interactive data visualizations
scikit-learn	Isolation Forest anomaly-detection model
ReportLab	PDF report generation

The prototype is organized around a Streamlit application script. The user interface collects the uploaded file and cleaning preferences, while Pandas performs the data operations. Plotly charts are rendered within the application, and scikit-learn is invoked for anomaly screening when the numeric-data conditions are met. Export functions serialize the processed data and create a PDF report. The implementation is designed for local or appropriately configured hosted execution; production deployment would require additional security, resource, and access-control measures.

7. Functional Demonstration and Preliminary Observations

A small sample spreadsheet used during prototype development contained 10 records and 7 columns, including numeric fields such as Quantity, Price, and Revenue. In the recorded sample overview, the dataset contained zero missing values and zero duplicate rows. The numeric totals displayed for the sample were 95 for Quantity, 194100 for Price, and 697100 for Revenue. These figures describe that particular demonstration file only; they are not benchmark results and should not be generalized to other datasets.

Demonstration item	Observed value	Interpretation
Rows and columns	10 × 7	Small prototype sample
Missing values	0	No missing entries detected in this sample
Duplicate rows	0	No duplicate rows detected in this sample
Quantity total	95	Sample-specific numeric aggregate
Price total	194100	Sample-specific numeric aggregate
Revenue total	697100	Sample-specific numeric aggregate

The demonstration indicates that the prototype can present dataset-level quality indicators and numeric aggregates for the sample file. A formal evaluation has not yet been conducted. In particular, detection precision, recall, false-positive rate, processing time, scalability, and user task-completion time have not been measured. These measures should be evaluated using labelled test data and datasets of different sizes before making performance claims.

8. Discussion

AutoReport AI brings several routine reporting activities into one interface and can reduce the need to repeat common spreadsheet operations manually. User-selectable cleaning is important because missing values and duplicates should not always be treated in the same way. Interactive charts and export formats make the output easier to inspect and reuse. The anomaly-detection feature adds a further review mechanism for numeric data, but it remains dependent on feature selection, data quality, and the contamination parameter.

The current prototype has limitations. The zero-fill option may be unsuitable for many variables; automatic type inference may misclassify some columns; Isolation Forest does not explain the business reason for an anomaly; and the present demonstration is too small to establish general performance. Sensitive business data should not be uploaded to an unsecured deployment. Authentication, encryption, access control, audit logging, and secure file handling would be necessary for organizational use.

9. Future Scope

- Add domain-aware missing-value imputation and configurable validation rules.
- Support larger files through chunked processing, background jobs, and database connectivity.
- Provide explainable anomaly summaries, configurable thresholds, and user feedback on flagged records.
- Add scheduled reports, reusable report templates, and role-based access.
- Evaluate the system on multiple real-world datasets using precision, recall, F1-score, false-positive rate, and execution time.
- Introduce privacy safeguards and deployment controls before handling confidential organizational data.

10. Conclusion

This paper presented AutoReport AI, a Python-based prototype for automated spreadsheet analysis and report generation. The system combines CSV/Excel ingestion, data-quality inspection, configurable cleaning, filtering, descriptive statistics, interactive visualizations, Isolation Forest-based anomaly screening, and CSV/Excel/PDF export. A small development sample demonstrated the presentation of data-quality indicators and numeric summaries. The work provides a foundation for a more comprehensive reporting tool, while further testing is required to quantify accuracy, scalability, usability, and practical value. The system is intended to assist human analysis rather than replace domain expertise or independent verification.

References

[1] W. McKinney, “Data Structures for Statistical Computing in Python,” in Proceedings of the 9th Python in Science Conference, 2010, pp. 56–61. doi: 10.25080/Majora-92bf1922-00a.

[2] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation Forest,” in 2008 Eighth IEEE International Conference on Data Mining, 2008, pp. 413–422. doi: 10.1109/ICDM.2008.17.

[3] Scikit-learn Developers, “IsolationForest,” scikit-learn documentation. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html (accessed: 20 Nov. 2026).

[4] The pandas development team, “pandas: Python Data Analysis Library,” documentation. Available: https://pandas.pydata.org/ (accessed: 20 Nov. 2026).

[5] Plotly Technologies Inc., “Plotly Python Graphing Library,” documentation. Available: https://plotly.com/python/ (accessed: 20 Nov. 2026).

[6] Streamlit, Inc., “Streamlit Documentation,” documentation. Available: https://docs.streamlit.io/ (accessed: 20 Nov. 2026).

[7] ReportLab, “ReportLab PDF Generation User Guide,” documentation. Available: https://www.reportlab.com/docs/reportlab-userguide.pdf (accessed: 20 Nov. 2026).