DataLens: ML-Oriented Interactive Tabular Data Quality Dashboard

Mohamed Abdelaal
Software AG, Darmstadt, Germany
Mohamed.Abdelaal@softwareag.com

Samuel Lokadjaja
TU Darmstadt, Darmstadt, Germany
sammyloka@yahoo.com

Arne Kreuz
Software AG, Darmstadt, Germany
Arne.Kreuz@softwareag.com

Harald Schöning
Software AG, Darmstadt, Germany
Harald.Schoening@softwareag.com

ABSTRACT

Maintaining high data quality is crucial for reliable data analysis and machine learning (ML). However, existing data quality management tools often lack automation, interactivity, and integration with ML workflows. This paper introduces DataLens[1], a novel interactive dashboard designed to streamline and automate the data quality management process for tabular data. DataLens integrates a suite of data profiling, error detection, and repair tools, including statistical, rule-based, and ML-based methods. It features a user-in-the-loop module for interactive rule validation, data labeling, and custom rule definition, enabling domain experts to guide the cleaning process. Furthermore, DataLens implements an iterative cleaning module that automatically selects optimal cleaning tools based on downstream ML model performance. To ensure reproducibility, DataLens generates DataSheets capturing essential metadata and integrates with MLflow and Delta Lake for experiment tracking and data version control. Our evaluations showcase DataLens's capabilities in effectively identifying and correcting data errors, improving data quality for downstream tasks, and promoting reproducibility in data cleaning pipelines.

1 INTRODUCTION

In this paper, we introduce an interactive data quality dashboard, referred to as DataLens[2], designed to streamline and automate multiple aspects of the data quality management process. With its modular design, DataLens brings significant enhancements to the field of data science, particularly in data profiling, validation, error detection, and correction. To this end, DataLens integrates several data preparation tools via REST APIs, facilitating the extension of the dashboard's capabilities. For instance, DataLens includes data profiling tools that automatically generate insights and rules, aiding users in exploring the data effectively. Moreover, DataLens integrates statistical, rule-based, and ML-based error detection tools. For the ML-based tools, the dashboard enables users to label data instances, a required step while training detection models. DataLens also integrates advanced algorithms to propose and apply corrections to identified data errors, reducing the manual intervention traditionally required in data cleaning.

DataLens integrates with ML tracking tools like MLflow for seamless tracking of data quality experiments, models, and results, offering a unified view of data quality and ML processes. It also supports Delta Lake integration, enabling version control of data, enhancing robustness, and allowing rollback to previous data versions. Additionally, DataLens generates DataSheets, JSON files that capture critical information including data version tags, hyperparameters, generated rules, data quality metrics, employed cleaning tools, and more. The dashboard offers two important features, including the user-in-the-loop module and the iterative cleaning module. The former module facilitates user interaction with the dashboard to tag dirty values, label data instances, and add custom rules. The latter module automatically selects the optimal data-cleaning tools that improve downstream ML models. Thanks to these features, the role of the data users is streamlined, limited to validating the generated rules and labeling data samples to train ML models used for error detection and/or correction. This approach maximizes the value of domain expert input while minimizing the technical burden on them.

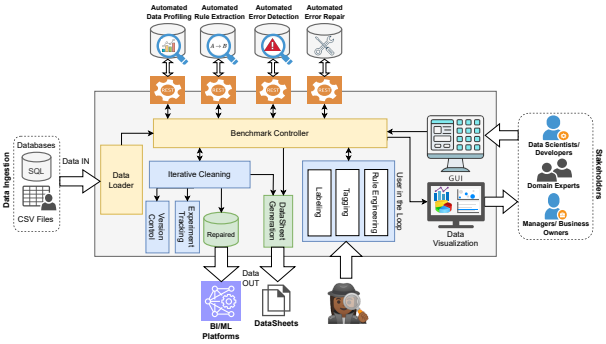


Figure 1: Architecture of DataLens

To sum up, this paper presents several key contributions: (1) a modular design for a data quality dashboard that enhances its extensibility; (2) the formulation of data cleaning tool selection as a hyperparameter tuning problem, allowing for automatic pipeline configuration based on downstream ML model performance; (3) the generation of DataSheets to precisely track the data cleaning process and ensure reproducibility; (4) the implementation of a user-in-the-loop module for interactive data preparation; (5) providing a means to interactively collect user labels for training ML-based error detection tools. This is a significant departure from the norm, where developers of such tools typically rely on ground truth data for labels. By enabling user-driven labeling, DataLens offers a more realistic evaluation framework for ML-based error detection tools; and (5) enabling the execution of multiple error detection tools, with DataLens autonomously integrating and deduplicating results to improve the precision and recall of error detection. To the best of our knowledge, DataLens is the first dashboard to offer interactive, automated, and ML-oriented data quality management.

[1]A recording of the dashboard can be found at https://youtu.be/tW5qqFqDFYI.
[2]https://github.com/mohamedyd/Data-quality-Dashboard

2 ARCHITECTURE & OVERVIEW

In this section, we introduce the architecture of DataLens, depicted in Figure 1. The architecture constitutes an automated, interactive data quality dashboard designed to optimize data quality for downstream applications such as Business Intelligence (BI) and ML platforms. The process begins with data ingestion, where data can be ingested into DataLens via one of three methods: (1) using one of the preloaded datasets that come with the dashboard, allowing users to explore its functionalities without needing their data; (2) uploading CSV or Excel files; or (3) establishing a SQL database connection. When a dataset is uploaded as a CSV or Excel file, an automatic backend process is initiated. A dedicated folder, named after the uploaded file, is created to store the dataset as 'dirty.csv'. A subfolder within this directory is also created to house the Delta table associated with this dataset. Additionally, the uploaded dataset is stored in the backend as a pandas DataFrame. For database connections, users can connect the dashboard to MySQL, PostgreSQL, and Microsoft SQL Server databases. They can input their credentials and specify the table and dataset they wish to load. Once loaded, these tables are treated identically to uploaded files.

DataLens incorporates a data loader that feeds the input data into a dashboard controller. Such a controller is responsible for controlling other modules, regulating data flow, and ensuring each step is executed properly. For a modular design, DataLens can integrate with several external tools using a set of standard REST APIs. An automated data profiling module analyzes the ingested data, identifying and recording its characteristics. Concurrently, an automated rule extraction module generates rules used later while detecting and repairing data errors. This automated rule extraction uses advanced algorithms, considering both the statistical properties and domain-specific characteristics of the data. The automated error detection module implements statistical, rule-based, and ML-based tools that scan the data to identify inconsistencies, outliers, or other potential issues. Identified errors are then passed to the automated error repair module, which uses advanced algorithms to propose and apply corrections to the detected errors. In parallel, a version control tracking module maintains a record of each data version throughout the cleaning process. This allows for robust data management and facilitates rollback to previous data versions if necessary.

DataLens is enhanced by two crucial features: the *user-in-the-loop* module and the *iterative cleaning* module. The user-in-the-loop module supports active user involvement, empowering them to validate or adjust the system-generated rules and corrections, as well as to introduce their own custom rules. It also enables users to annotate specific data samples to train ML models utilized for data validation or correction. Furthermore, this module facilitates proactive error management by allowing users to tag data samples known to be corrupted in advance. The iterative cleaning module autonomously identifies the optimal error detection and repair tools for specific input data used in training a downstream ML model. To this end, we conceptualize the selection of error detection and repair tools as a hyperparameter tuning problem. By conducting multiple cleaning iterations with various combinations of these tools, the iterative cleaning module determines the configuration that maximizes predictive accuracy.

The system also generates *DataSheets* capturing critical information such as data version tags, hyperparameters, generated rules, data quality metrics, and employed cleaning tools. Another output of DataLens is the repaired data, which can be utilized in BI or ML platforms. Additionally, the system offers visualizations that allow various stakeholders—including data scientists, developers, domain experts, managers, and business owners—to review and comprehend the data cleaning process and its outcomes. DataLens has been designed with integration capabilities with two common ML tracking tools, such as MLflow. This allows for seamless tracking of data quality experiments, models, and results, providing a unified view of both data quality management and ML processes. DataLens also includes integration with Delta Lake, enabling tracking of different data versions. This feature brings robustness to the data management process, allowing the tracking of changes over time, and facilitating rollback to previous data versions if required.

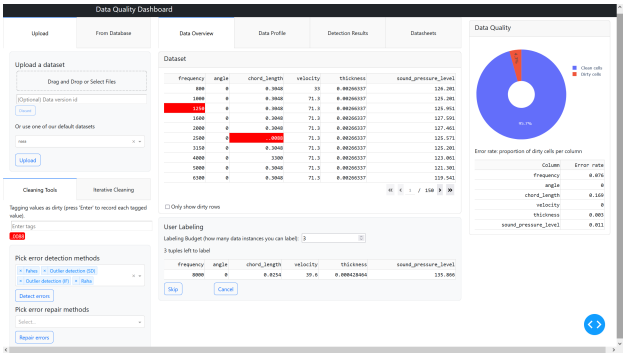


Figure 2: Main window of DataLens

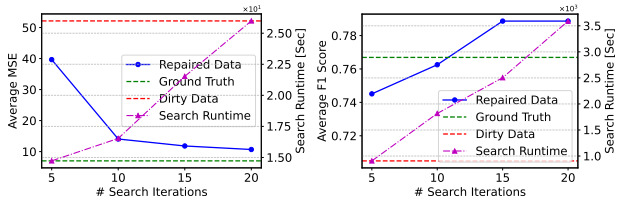
Figure 2 depicts the main window of DataLens. The left segment of the dashboard contains a user interface for data upload. Users can also select from a range of automated error detection and repair tools, enhancing the adaptability of the system to diverse data quality needs. The central part of the dashboard is organized into several tabs: Data Overview, Data Profile, Error Detection Results, and DataSheets. The Data Overview tab displays both the uploaded datasets and any detected errors, providing a quick snapshot of the data and its quality issues. A dedicated user labeling section is included here, where users can label selected data samples as either 'true' (i.e., dirty) or 'false' (i.e., clean). The Data Profile tab presents the findings of the data profiling tools, offering insights into the characteristics and structure of the dataset. The Error Detection Results tab, meanwhile, visualizes the output of the error detection tools, making it easy to understand and manage detected errors. The DataSheets tab displays the generated DataSheet with important metadata and gives users the functionality to download these files for further analysis or record-keeping. Finally, the right segment of the dashboard hosts the Data Quality section, providing visual representations of various data quality metrics.

3 ITERATIVE CLEANING

In this section, we introduce the Iterative Cleaning module, designed to automatically select data cleaning tools in light of the performance of a downstream ML model. By tailoring the cleaning process to the specific needs of the ML model, we can effectively optimize the model's performance. Once a user provides the type of ML model to be trained, the Iterative Cleaning module embarks on a cycle of cleaning processes, employing a variety of cleaning tools. The cleaning tools that contribute to the most significant performance enhancement are selected and applied. In this way, data-cleaning tools are conceptualized as hyperparameters that can be optimized jointly with the typical parameters in ML pipelines, e.g., number of layers, and number of neurons.

We commence the optimization process by delineating the search space, encompassing all potential permutations of error detection and repair tools. Subsequently, a scoring function is defined to measure the performance of a given detector and repair tool on the repaired dataset. Such a function can be defined as the Mean Square Error (MSE) of the target ML model in case of regression and the F1 score of the ML model in case of classification. An iterative process continues for a predetermined number of iterations, or until the accuracy of the ML model reaches a desired threshold. In each iteration, the module first trains the ML model on a repaired version of the dataset, and calculates the model accuracy. DataLens leverages a Bayesian hyperparameter optimization algorithm, referred to as Optuna [2], that idenitifies of optimal hyperparameters for ML models. Optuna systematically navigates this search space to ascertain the amalgamation of tools that yield the highest performance according to the predefined scoring function. A sequential model-based optimization approach drives this process. Optuna iteratively selects the most promising hyperparameters to evaluate, based on the past trial outcomes, to converge on the optimal configuration efficiently.

Figure 3 presents a proof-of-concept evaluation of our iterative cleaning approach. We assess the impact of increasing search iterations on predictive performance, visualizing the results against baseline performances achieved using ground truth and dirty data. As an example, Figure 3a depicts the MSE of a decision tree model trained on data cleaned using the best tools identified by DataLens at each iteration count (ranging from 5 to 20). With fewer iterations, DataLens explores a limited subset of cleaning tools. However, as the iteration count increases, DataLens can evaluate a wider range of tool combinations, leading to the selection of effective cleaning tools. With 20 iterations, the model trained on data cleaned with the identified tools (Raha and ML Imputer in this instance) achieves an MSE of 10.7, closely approaching the performance achieved with the ground truth data.



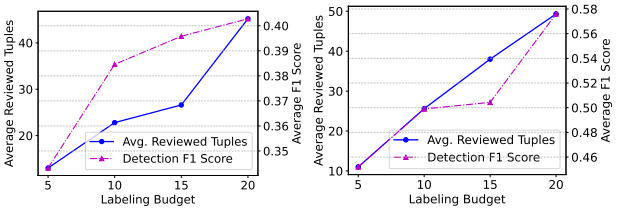
(a) NASA (b) Beers

Figure 3: Impact of the number of search iterations

While achieving these results comes with a slight increase in search runtime (i.e., the time required for hyperparameter optimization), the iterative nature of our approach provides users with a powerful trade-off: they can directly control the balance between accuracy and runtime by adjusting the number of search iterations. Figure 3b further illustrates the effectiveness of our approach on the Beers dataset for a multi-class classification task, demonstrating consistent performance improvements with increasing iterations. It is important to highlight that existing comparable methods like ActiveClean, BoostClean, and CPClean [1] are restricted to binary classification tasks. In contrast, our iterative cleaning approach is broadly applicable to diverse ML tasks and model types, underscoring its potential impact.

4 EVALUATION OF ML-BASED TOOLS

In this section, we introduce another feature of DataLens related to the realistic evaluation of ML-based error detectors. RAHA, a state-of-the-art ML-based detector, employs a user-dependent operational modality, unlike other detection methods that operate independently of user input. While RAHA is initiated simultaneously with other detection methods, its results computation and visualization occur asynchronously, contingent upon the user completing the required tuple labeling. This design ensures that RAHA's output accurately reflects the user's input, preventing the premature display of results before data labeling is complete. To facilitate **user labeling**, DataLens prompts the user to define a labeling budget (N), representing the number of tuples they are willing to label. Subsequently, the dashboard presents N tuples sequentially for labeling. The user examines each tuple, marking any dirty instances. If a tuple contains no dirty instances, the user can skip it, prompting DataLens to display the next recommended tuple. As a tuple selection strategy, RAHA leverages clustering with label propagation. Figure 4 presents an evaluation of the data labeling process for RAHA, highlighting a key advantage of DataLens in facilitating realistic assessments of ML-based cleaning tools. Unlike evaluations presented in the original publications of such tools, DataLens allows us to quantify the actual labeling effort required. As depicted in Figure 4a, the average number of tuples reviewed by users consistently exceeds twice the user-defined budget. For instance, with a budget of 20 tuples, users reviewed an average of 45.2 tuples. This discrepancy arises because the tuple selection strategy, while designed to prioritize potentially erroneous data, often selects clean tuples for review. The figure also demonstrates that increasing the budget from 5 to 20 tuples yields a marginal improvement in the average F1 score for error detection, rising from 0.34 to 0.4. Similar trends are observed for the Beers dataset, as shown in Figure 4b.



(a) NASA (b) Beers

Figure 4: Evaluation of labeling ML-based tools

5 REPRODUCIBLE DATA QUALITY

In this section, we present three features to facilitate the reproducibility of data quality experiments, including DataSheets, integration with mlflow, and integration with Delta Lake. DataLens allows users to generate DataSheets once error detection and repair tools have been executed on the dataset. These DataSheets compile an array of details about the dataset, including the dataset's name, locations for both the input dirty dataset and the repaired dataset, the shape (number of rows and columns) of the dataset, the detection tools applied, the number of erroneous cells identified in the dataset, the repair tools executed, and the configurations of such tools. It is important to mention that DataLens enables users to download the DataSheets and upload them later to reproduce the same data preparation steps.

Alongside experiment tracking, DataLens also monitors various versions of a dataset with the assistance of the Delta Lake library [3]. This library is built upon the delta-rs Rust library, providing a robust foundation for dataset versioning. We opted for

[3]https://delta-io.github.io/delta-rs/python/

Delta Lake due to its simplicity in both setup and usage, particularly through its Python API. Unlike other data versioning libraries, Delta Lake does not necessitate preliminary setup requirements such as establishing an SQL connection, configuring a Kubernetes cluster, or initializing a Git repository. This straightforward approach reduces complexity and enhances user experience, making it an optimal choice for our dataset versioning needs. Upon the initial upload of a dataset, a Delta Lake is instantiated. This Delta Lake essentially serves as a repository for the dataset, housing all the versions and transformations the dataset undergoes. The uploaded dataset is stored within this Delta Lake as a DeltaTable. A DeltaTable is a high-performance, format-agnostic, and schema-enforced collection of data, providing a structured and scalable framework for data storage. One of the key advantages of DeltaTable is its seamless interoperability with pandas DataFrames, facilitated by the methods provided by the Delta Lake library. It can be easily converted to a DataFrame for analysis and manipulation, and conversely, a DataFrame can be readily stored as a DeltaTable post-processing.

If a DeltaTable already exists for a dataset from previous uploads, the user has the option to specify a version number during the dataset upload process. If the specified version number exists within the Delta Lake, the corresponding version of the dataset will be loaded for use. In scenarios where the indicated version does not exist, or the user does not provide a version number, the uploaded dataset will be stored as a new version within Delta Lake. Notably, this process does not overwrite or erase previous versions. Each iteration of the dataset is preserved within Delta Lake, maintaining a comprehensive record of dataset versions. This allows for historical tracking, comparison across versions, and the ability to revert to earlier versions if needed, thereby enhancing the robustness and flexibility of the data management system. Once a user executes an error repair method, the resultant repaired dataset is stored within Delta Lake as a new, distinct version. This ensures that the dataset's progression through each error repair operation is precisely tracked, thereby maintaining a comprehensive record of the dataset's evolution. If a DataSheet is generated by the user, it will contain the version number of the dataset that has been used for error detection, as well as the version number of the dataset post-repair. This information provides a clear reference of the dataset's status at various stages of the error detection and repair process. By incorporating these version numbers into the datasheet, we enhance its utility as a comprehensive report of the data operations.

6 RELATED WORK

In this section, we report on the state-of-the-art techniques and tools relevant to DataLens. There exist also a few research papers that tackle the data quality problem. For instance, Ramneesh et al. [5] presents a dashboard for linked data quality assessment. The dashboard leverages AI methods to evaluate and monitor the quality of linked data. While it offers novel ways to handle linked data, its scope may be limited to this specific data type and may not provide comprehensive solutions for broader data quality issues. Similarly, Blacketer et al. [3] show the application of a data quality dashboard to ensure conformance to model specifications within the European Health Data and Evidence Network (EHDEN). While the dashboard's application in a network like EHDEN underlines its potential, it might be heavily tailored to specific requirements of such a network and may not be fully applicable to other domains or data types. Gitzel et al. [4] introduces a data quality dashboard for Computerized Maintenance Management System (CMMS) data, where all equipment failures are reported. The focus on CMMS data indicates a high level of specialization, which could limit its broader applications. Additionally, being published in 2018, it might lack some of the more recent advancements in data quality management, such as deep integration with machine learning tools and processes.

In the realm of contemporary tools, a plethora of shortcomings persist that impede their seamless integration into practical applications. Notably, these include limited data quality management, wherein existing tools predominantly concentrate on data visualization while neglecting crucial dimensions of data quality management such as automated data profiling, validation, and correction. Furthermore, an overreliance on outdated approaches is evident, with many tools failing to incorporate recent advancements in data quality management. Additionally, there is a pronounced dependence on the user's comprehension and delineation of data quality, which can be highly variable and insufficiently comprehensive, thereby undermining the holistic assessment of data quality. Another limitation lies in the restricted scope of some research methodologies, which are often confined to specific data types or domains, thus constraining their applicability across diverse contexts. Lastly, commercial data quality dashboards frequently exhibit complexity and impose a significant burden on users, necessitating sophisticated configuration and integration efforts, along with skilled personnel. Collectively, these factors elucidate the exigency for innovative solutions that address these multifaceted challenges.

7 CONCLUSION & FUTURE WORK

This paper introduces DataLens, an interactive and ML-oriented dashboard aimed at streamlining and automating the management of tabular data quality. We elaborated on integrating a diverse set of data profiling, error detection, and repair tools through REST APIs, enhancing DataLens's extensibility and functionality. Additionally, we described our iterative cleaning approach, which automatically selects the most effective cleaning tools based on the performance of downstream ML models, thus optimizing the data-cleaning pipeline. Moreover, we introduced how DataLens enables domain experts to contribute their knowledge for improved cleaning accuracy through supporting interactive rule validation, data labeling, and custom rule definition. While DataLens offers a comprehensive approach to data quality management, several avenues for future work exist: (1) Exploring more intuitive and user-friendly ways to interact with DataLens, such as natural language processing for rule definition and visual analytics for data exploration, can further enhance user experience. (2) Integrating explainability techniques into the error detection and repair process would give users insights into why specific errors were flagged and how corrections were made, fostering trust and understanding.

REFERENCES

[1] M. Abdelaal, C. Hammacher, and H. Schoening. 2023. REIN: A Comprehensive Benchmark Framework for Data Cleaning Methods in ML Pipelines. In *EDBT*.

[2] T. Akiba, S. Sano, T. Yanase, and M. Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[3] Clair Blacketer, Erica A Voss, Frank DeFalco, Nigel Hughes, Martijn J Schuemie, Maxim Moinat, and Peter R Rijnbeek. 2021. Using the data quality dashboard to improve the EHDEN network. *Applied Sciences* 11, 24 (2021), 11920.

[4] Ralf Gitzel, Subanatarajan Subbiah, and Christopher Ganz. 2018. A Data Quality Dashboard for CMMS Data.. In *ICORES*. 170–177.

[5] Ramneesh Vaidyambath, Jeremy Debattista, Neha Srivatsa, and Rob Brennan. 2019. An intelligent linked data quality dashboard. (2019).