Lightweight Pipelines: Good Enough is Sometimes Better

Camilla Sancricca
Politecnico di Milano
Milan, Italy
camilla.sancricca@polimi.it

Cinzia Cappiello
Politecnico di Milano
Milan, Italy
cinzia.cappiello@polimi.it

ABSTRACT

Data-centric AI prioritizes data-centric strategies for providing high-quality input data, since high data quality is fundamental for ensuring the reliability of AI systems. It recognizes the critical role of data preparation, an often time-consuming task that is a significant challenge for data scientists. While various methods have been proposed to support and automate data preparation, most focus on optimizing model performance without considering the high energy and resource requirements for computing complex pipelines on large-scale datasets.

This paper introduces an approach to designing lightweight data preparation pipelines: it selectively addresses only the data quality issues that significantly impact analysis performance. We present a framework that leverages historical knowledge of previous pipelines to recommend an effective sequence of preparation actions for a specific analysis that favors efficiency while preserving the quality. Our preliminary experiments show that improving only the most impactful data quality errors leads to high performance while significantly conserving time and resources.

VLDB Workshop Reference Format:
Camilla Sancricca and Cinzia Cappiello. Lightweight Pipelines: Good Enough is Sometimes Better. VLDB 2025 Workshop: 2nd International Workshop on Data-Centric AI (DATAI).

VLDB Workshop Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/camillasancricca/diana-repo/tree/main/sustainability.

1 INTRODUCTION

Data-centric Artificial Intelligence (DCAI) recently spotlighted the importance of having high-quality data to be processed in AI pipelines, to ensure the reliability of the final outcomes [4]. This emphasized the crucial role of data preparation, which encompasses a range of techniques to systematically enhance data quality (DQ). Such a phase is very demanding due to the variety of quality errors and available data preparation techniques in the literature. Data scientists experience difficulties in defining effective pipelines, taking about 80% of the total pipeline time in preparing data [2].

Several approaches have been developed in recent years to support this phase: given a new dataset, they aim to automatically generate a sequence of actions to optimize machine learning (ML) results [6]. To achieve that, they adopted various techniques, such as reinforcement learning [1], ML uncertainty estimation [3], or exploiting the knowledge related to pipelines performed in the past [5, 11]. Widespread AutoML approaches also offer the possibility of automatically performing data preparation [7]. These approaches aim to facilitate the design of the preparation pipeline with the sole goal of optimizing the final performance. However, these approaches are often applied to a vast amount of data, employing complex ML-based techniques for preparing datasets.

Consequently, the application of such preparation pipelines can be computationally expensive, requiring a high amount of resources, consuming a significant amount of energy, and contributing to an increase in gas emissions and the carbon footprint of AI systems.

To address this problem, green AI has emerged as a concept that prioritizes energy efficiency and sustainability alongside ML performance. Few contributions have also started to consider data-centric strategies for green AI [8], demonstrating that sustainability can often be achieved with negligible or even absent decline in ML accuracy [10]. Lightweight pipelines can be designed by carefully selecting a subset of the data, choosing an appropriate dataset size, or applying feature selection or dimensionality reduction techniques.

In this paper, we present an approach for supporting the design of lightweight data preparation pipelines. Our approach is based on identifying the DQ errors that mostly affect the final performance, thus prioritizing the data preparation actions that enhance only the most influential DQ aspects, leaving less impactful errors unaddressed.

To achieve this goal, we designed a framework, called DIANA [9], for suggesting users with the most effective sequence of data preparation actions to optimize the ML analysis results in a specific analysis context – the combination of the data profile (the set of data characteristics that can be extracted through data profiling from the dataset – dataset profile – and its columns – columns profiles) and the AI model used.

Pipeline suggestions are given by exploiting a Knowledge Base (KB) that contains historical data about previously designed pipelines. By analyzing the most similar data profiles already stored in the KB, DIANA can identify the most relevant DQ dimensions and the most effective improvement actions for the specific analysis context.

This paper presents preliminary results on the potential of our suggestions in saving energy. We demonstrate that improving only the most influential DQ dimension yields nearly the same final ML performance as completing the entire preparation pipeline, while saving time and resources and ensuring sustainability.

The paper is structured as follows: Section 2 introduces the general approach, Section 3 depicts preliminary results on the effectiveness and sustainability of the presented approach, and Section 4 presents future work and concludes the paper.

Proceedings of the VLDB Endowment. ISSN 2150-8097.

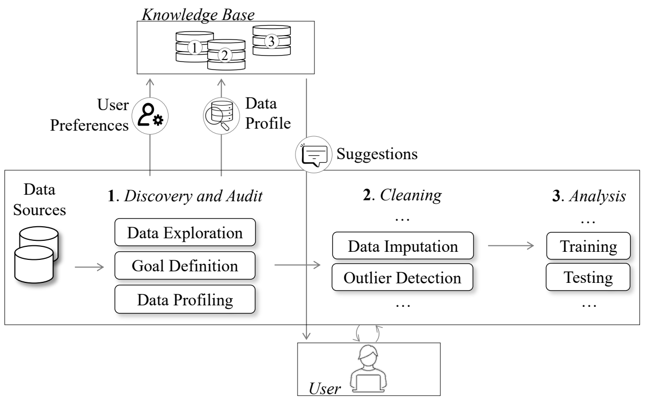

Figure 1: DIANA's general architecture

2 THE DIANA APPROACH

This section presents our approach for a**d**apt**i**ve d**a**ta-ce**n**tric **AI**.

Figure 1 shows DIANA's general functioning, including the main steps of the data science pipeline. DIANA can explore and profile datasets, such as providing metadata, basic statistics and visualizations on the datasets' content, and assess their quality revealing potential issues that might compromise the analysis results in the *discovery and audit* phase; moreover, it allows the user to enter the type of analysis to be performed in the *goal definition*.

The targeted analysis and the computed *data profile* (the new user's analysis context) are provided as input to the KB, which is responsible for generating an optimal sequence of data preparation actions (along with their suggested execution order) to maximize the quality of the results.

The DIANA's KB includes results of previous pipelines, computed on several datasets, related to the impact of DQ errors and the effectiveness of data preparation techniques on ML models. The KB exploits the information on the collected data profiles that are most similar to the user's dataset and provides (i) the order in which the DQ dimensions should be improved and, for each DQ dimension, (ii) a suggestion about the best improvement technique to apply for each column.

The data *cleaning* phase (a) outputs the most appropriate techniques to perform for designing the pipeline, and (b) executes the DQ improvement tasks. At the end of the cleaning phase, the user can proceed with the *analysis*, i.e., all the steps needed for executing the ML model.

2.1 Knowledge Base content

To feed the content of the KB, we performed experiments on a large number of analysis contexts.

For several combinations of dataset-ML algorithms, we extracted three types of results: [KB_1] impact of DQ errors (related to a DQ dimension, e.g., missing values for completeness, outliers for accuracy) on ML performance; [KB_2] effectiveness of applying several data imputation techniques on ML performance; [KB_3] effectiveness of several outlier detection techniques in detecting outliers.

The pipeline for enriching the KB is reported in Figure 2. Note that phases 1 and 4 were performed only for enriching the KB_1.

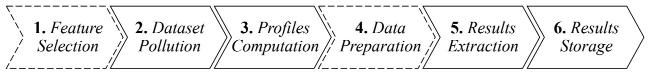

Figure 2: KB's enrichment pipeline

1. *Feature Selection* We kept only the three most influential columns employing a univariate statistical test to identify features most correlated to the target column.
2. *Dataset Pollution* We created different versions of the original dataset, injecting a different percentage (from 10% to 50%, with an increasing step of 10%) of one type of DQ error. To inject the errors, we followed a uniform distribution.
3. *Profiles Extraction* We computed the dataset and column profiles of the considered dataset's version.
4. *Data Preparation* For each polluted version, we applied different data preparation techniques to improve the injected errors.
5. *Results Extraction* We computed the selected ML algorithms on the final dataset version and extracted the performance. The polluted/cleaned version is split into train and test sets (70% and 30%), the train set is used to fit the model, while the test set is employed for the prediction task through 4-fold cross-validation. We tuned the hyperparameters on the original dataset, and the ones achieving higher performance were used for all computations.
6. *Results Storage* We stored the experiment results as follows: [KB_1] ratio between ML performance achieved by the polluted version and the original dataset; [KB_2] ratio between ML performance achieved by the final cleaned version and the original dataset; [KB_3] percentage of detected outliers.

Setup. We performed the experiments considering two DQ dimensions, Completeness and Accuracy, thus injecting missing values and outliers as DQ errors and applying 12 data imputation and 6 outlier detection techniques as data preparation techniques (both standard and ML-based techniques were considered). We employ a total of 31 datasets with heterogeneous profiling characteristics taken from the UCI Machine Learning and Kaggle repositories.

Six classification algorithms were performed using the F1 score as a performance indicator: decision tree (DT), logistic regression (LR), k-nearest Neighbors (KNN), random forest (RF), adaboost (AB), and support vector machine (SVC) implemented by the *scikit-learn* library.

Details on the datasets used to enrich the three KBs, the entire set of profiling metadata extracted for dataset and column profiles, and the complete list of data preparation tasks are recorded in our repository.

2.2 Pipeline suggestions

The KB based its suggestions on three ML-based predictors. The are regression models (*KNNRegressor* models, with different hyperparameters depending on the prediction task) with the following characteristics:

P1 *Trained on*: [KB_1]. *Input*: a data profile and an ML model. *Output*: expected impact of a DQ dimension on the analysis performance.

P2 *Trained on*: [KB$_2$]. *Input*: a column profile and an ML model. *Output*: expected effectiveness (ML performance) of applying a data imputation task on that column.

P3 *Trained on*: [KB$_3$]. *Input*: a column profile. *Output*: expected effectiveness of applying an outlier detection task on that column.

Our predictors had an average root mean squared error of at most 13.44% (for the first predictor), while the other errors were even smaller. Moreover, we were able to predict the correct ranking for 74,13% of the tested analysis contexts. We were always able to predict the most relevant methods to apply for each column.

By sorting the results of the predictors, we can suggest an optimal data preparation pipeline for a specific analysis context.

3 PRELIMINARY RESULTS

In this section, we present the experiments performed to validate that (i) the proposed pipeline suggestions can lead to better results for a specific analysis and (ii) *good enough is sometimes better.*

Improving only the most influential DQ errors in the analysis context, while leaving other errors unaddressed, can lead to shorter and more efficient pipelines. We demonstrate that this approach still achieves high performance, saving computational time and resources required for executing the entire data preparation pipeline.

3.1 Experimental pipeline

We employed 6 new datasets from Kaggle[1] with different dimensionalities (number of tuples) and data types (numerical/categorical columns) to validate our approach: *weather*, *consumer*, *pet*, *character*, *galaxy*, and *heart*.

We kept only the three most influential columns on classification based on the mutual information score between the columns and the target one. More details are available in our public repository. For each dataset, we perform the following pipeline.

We polluted the dataset, creating different versions of the original one but injecting a different percentage (from 10% to 50%) of errors equally distributed. For each polluted version-ML model:

(1) We computed the dataset/column profiles and extract the suggested pipeline from our predictors.
(2) We computed the suggested pipeline and all the other possible pipelines on the polluted dataset.
(3) The ML model is executed on the cleaned dataset: it is split into train and test sets (70% and 30%), the train set is used to fit the model, while the test set is employed for the prediction task through 4-fold cross-validation. The model hyperparameters were tuned on the original dataset and used for all computations. The performance of the ML model is extracted *each time a DQ dimension is improved.*

All the possible pipelines were applied to the polluted dataset. First, the DQ dimensions were improved in different orders: completeness → accuracy, and accuracy → completeness. Then, all possible combinations of data preparation tasks were applied to the dataset's columns in all column orders.

For the completeness → accuracy pipeline: (a) a specific combination of imputation techniques is applied to the dataset's columns; (b) a combination of outlier detection techniques is applied to the numerical columns and the identified outliers are set to null; (c) the same imputation techniques applied in (a) are executed again to fill the null values. For the accuracy → completeness pipeline, only (b) and (c) are executed.

We performed a total of 176, 580 pipelines for datasets with three numerical columns, 89, 211 pipelines for datasets with two numerical columns, and 22, 440 pipelines for datasets with one numerical column.

3.2 Results and discussion

Preliminary results obtained by applying the validation pipeline for all tested dataset–ML model combinations are presented as follows. The goal is to demonstrate that our approach is:

- *effective*, thus, the suggested pipeline for that analysis context achieves better results than applying all the other combinations;
- *lightweight*, thus, improving only the first DQ dimension of the suggested ranking still achieves good performance and, therefore, the other less influential DQ dimensions can be neglected and not addressed.

To achieve this goal, we need a metric for comparing the achieved performance p of an ML model, executed on the dataset version d_j, after the improvement of the first dimension d_1 of the ranking, and after all dimensions, d_1 and d_2, are improved. This metric, called ΔQ, is computed as follows:

$$\Delta Q_i = \frac{p_i}{p_o}$$

Where p_o is the performance achieved by computing an ML model on the original dataset d_o, and p_i is the performance achieved by the same ML model at step i: $i = 1$ when only the first DQ dimension is improved; $i = 2$ when all considered dimensions are improved.

Table 1 shows the median results (aggregated for all percentages of injected errors) of ΔQ_1 (improvement of the first dimension) and ΔQ_2 (all pipeline improvement), for all combinations of dataset–ML models. Moreover, ΔQ values obtained by computing the ***suggested*** pipeline, and executing ***all*** the other pipeline combinations are shown separately. The median results, aggregated for all datasets, are shown at the bottom of the table. Moreover, Figure 3 reports the ΔQ_1 and ΔQ_2 value distributions over the different models for the suggested pipeline and all the other pipelines.

From Table 1, we can notice that the suggested pipeline *always achieve higher* ΔQ (in both steps 1 and 2) than trying all the other pipeline combinations (except for ΔQ_1 in AB, which still obtains a higher value for ΔQ_2), validating that our approach is *effective*. This is also highlighted by Figure 3, in which the ΔQ values obtained by running the suggested pipelines are more concentrated at the top of the violin plots.

Moreover, we can appreciate that the ΔQ_1 values are *always very close and even higher*, i.e., a better performance is achieved after step 1 than after completing step 2, than the ΔQ_2 values. This is also evident by looking at Figure 3, in which the value distribution of ΔQ_1 (both for all and suggested pipelines) is almost the same as the ΔQ_2 distribution.

[1]https://www.kaggle.com

Table 1: Median ΔQ_1 and ΔQ_2 results for different combinations of dataset and algorithms

		DT		*LR*		*KNN*		*RF*		*AB*		*SVC*	
		ΔQ_1	ΔQ_2	ΔQ_1	ΔQ_2	ΔQ_1	ΔQ_2	ΔQ_1	ΔQ_2	ΔQ_1	ΔQ_2	ΔQ_1	ΔQ_2
weather	*suggested*	0.8388	0.836	0.8552	0.8698	0.8396	0.8405	0.8438	0.841	0.8561	0.8531	0.4609	0.5865
	all	0.815	0.78	0.5592	0.657	0.7651	0.7819	0.8385	0.8124	0.8493	0.8316	0.3143	0.3549
galaxy	*suggested*	0.9802	0.9776	1.1107	1.1104	0.9696	0.9715	0.9735	0.9805	0.9627	0.9735	1.2328	1.2403
	all	0.9284	0.9011	0.7415	0.9142	0.8545	0.869	0.9391	0.9207	0.9423	0.918	1.0045	1.0723
character	*suggested*	0.7926	0.813	0.9476	0.9456	0.8234	0.8283	0.7984	0.7997	0.8514	0.8449	0.98	0.9805
	all	0.8039	0.7936	0.8714	0.8752	0.8195	0.8076	0.7962	0.792	0.8511	0.8305	0.9466	0.9588
consumer	*suggested*	0.9441	0.9385	1.0049	0.9977	0.9361	0.9325	0.921	0.9239	0.9348	0.9292	0.6352	0.6315
	all	0.9268	0.9113	0.9213	0.9171	0.9229	0.9161	0.908	0.8993	0.9427	0.9342	0.5804	0.5685
pet	*suggested*	0.8223	0.8389	0.994	0.9924	0.8761	0.8853	0.8279	0.8258	0.8275	0.8279	1.0143	0.9912
	all	0.8178	0.8095	0.9761	0.9572	0.882	0.8567	0.8172	0.8006	0.8768	0.862	1.0146	0.9902
heart	*suggested*	0.815	0.8089	0.9427	0.9269	0.8488	0.8107	0.8224	0.7884	0.8215	0.7671	1.0343	1.0334
	all	0.8187	0.8047	0.9363	0.9504	0.8494	0.83	0.8165	0.7894	0.8293	0.8349	1.0173	1.0188
MEDIAN (all data)	*suggested*	0.8306	0.8375	0.9708	0.969	0.8625	0.8629	0.8359	0.8334	0.8538	0.849	0.9972	0.9859
	all	0.8183	0.8071	0.8964	0.9157	0.852	0.8434	0.8279	0.8065	0.864	0.8485	0.9756	0.9745

This clearly suggests that improving the most influential DQ dimension results in a significant gain in performance, while subsequent improvements of less influential dimensions result in a very small performance gain.

We also noted that improving less impactful errors can worsen the final performance: since the algorithm is not sensitive to those types of errors, correcting them introduces approximate values that can compromise the final results.

This highlights that *"good enough is sometimes better"*: improving the first dimension suggested by our approach always leads to performance that is at most the same as the performance obtained by running the total pipeline. This performance can be achieved even without completing the pipeline execution, saving the energy and resources needed for all subsequent quality improvement operations.

4 CONCLUSIONS

We presented an approach for suggesting lightweight preparation pipelines that favor sustainability while preserving an acceptable results' quality. Future work includes actually measuring the consumption of the collected pipelines, in terms of energy, gas emissions, and carbon footprint, if carried out on large amounts of data. Moreover, the current framework can be extended with more DQ dimensions, ML algorithms, and data preparation techniques.

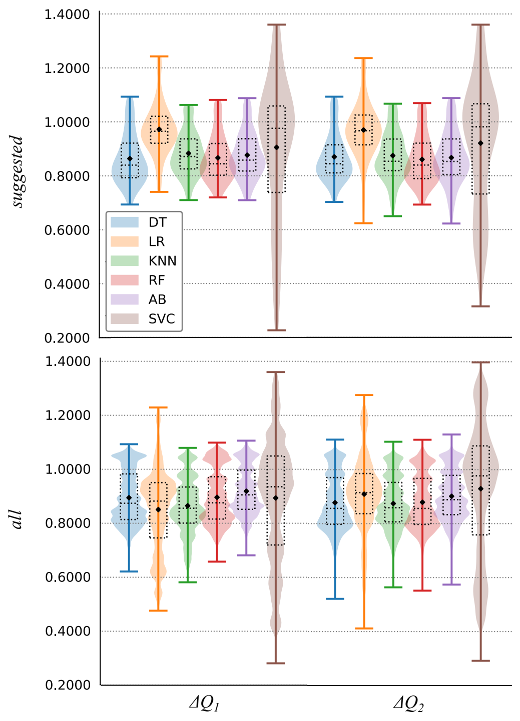

Figure 3: ΔQ_1 and ΔQ_2 distributions for several algorithms

ACKNOWLEDGMENTS

This research was supported by EU Horizon Framework grant agreement 101069543 (CS-AWARE-NEXT).

REFERENCES

[1] Laure Berti-Équille. 2019. Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation. In *The World Wide Web Conference, WWW 2019*. ACM, 2580–2586.

[2] Mazhar Hameed and Felix Naumann. 2020. Data Preparation: A Survey of Commercial Tools. *SIGMOD Rec.* 49, 3 (2020), 18–29.

[3] Sebastian Jäger and Felix Biessmann. 2024. From Data Imputation to Data Cleaning - Automated Cleaning of Tabular Data Improves Downstream Predictive Performance. In *AISTATS (Proceedings of Machine Learning Research)*, Vol. 238. PMLR, 3394–3402.

[4] Mohammad Hossein Jarrahi, Ali Memariani, and Shion Guha. 2023. The Principles of Data-Centric AI. *Commun. ACM* 66, 8 (2023), 84–92.

[5] Mohammad Mahdavi and Ziawasch Abedjan. 2021. Semi-Supervised Data Cleaning with Raha and Baran. In *11th Conference on Innovative Data Systems Research, CIDR 2021*. www.cidrdb.org.

[6] Leonel Aguilar Melgar, David Dao, et al. 2021. Ease.ML: A Lifecycle Management System for Machine Learning. In *CIDR*. www.cidrdb.org.

[7] Felix Neutatz, Binger Chen, Yazan Alkhatib, Jingwen Ye, and Ziawasch Abedjan. 2022. Data Cleaning and AutoML: Would an Optimizer Choose to Clean? *Datenbank-Spektrum* 22, 2 (2022), 121–130.

[8] Shirin Salehi and Anke Schmeink. 2024. Data-Centric Green Artificial Intelligence: A Survey. *IEEE Trans. Artif. Intell.* 5, 5 (2024), 1973–1989.

[9] Camilla Sancricca, Giovanni Siracusa, and Cinzia Cappiello. 2024. Enhancing data preparation: insights from a time series case study. *Journal of Intelligent Information Systems* (2024), 1–28.

[10] Roberto Verdecchia, Luís Cruz, June Sallou, Michelle Lin, James Wickenden, and Estelle Hotellier. 2022. Data-centric green AI an exploratory empirical study. In *International Conference on ICT for Sustainability (ICT4S)*. IEEE, 35–45.

[11] Fernando Rezende Zagatti, Lucas Cardoso Silva, Lucas Nildaimon dos Santos Silva, Bruno Silva Sette, Helena de Medeiros Caseli, Daniel Lucrédio, and Diego Furtado Silva. 2021. MetaPrep: Data preparation pipelines recommendation via meta-learning. In *ICMLA*. IEEE, Piscataway, NJ, 1197–1202.