# Provenance Tracking for End-to-End Machine Learning Pipelines

Stefan Grafberger
s.grafberger@uva.nl
AIRLab, University of Amsterdam

Paul Groth
p.groth@uva.nl
University of Amsterdam

Sebastian Schelter
s.schelter@uva.nl
University of Amsterdam

**ACM Reference Format:**
Stefan Grafberger, Paul Groth, and Sebastian Schelter. 2023. Provenance Tracking for End-to-End Machine Learning Pipelines. In *Companion Proceedings of the ACM Web Conference 2023 (WWW '23 Companion), April 30-May 4, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3543873.3587557

Software systems that learn from data are being deployed in increasing numbers in real-world application scenarios. It is a difficult and tedious task to ensure at development time that the end-to-end ML pipelines for such applications adhere to sound experimentation practices, such as the strict isolation of train and test data. Furthermore, there is a need to enforce legal and ethical compliance in automated decision-making with ML. For example, to determine whether a model works equally well for different groups. To enforce privacy rights (such as the 'right to be forgotten'), we must identify which models consumed the user's data for model training to retrain them without this data. Moreover, model predictions can be corrupted due to undetected data distribution shift, e.g., when the train/test data was incorrectly sampled or changed over time (covariate shift) or when the distribution of the target label changed (label shift). Data scientists also require support for pipeline debugging and for uncovering *erroneous data*, e.g., to identify samples that are not helpful for the classifier and potentially dirty or mislabeled or to identify subsets of data for which a model does not work well.

**Towards automated low-effort tooling for ML pipelines**. Most of the listed issues are typically addressed manually in an ad-hoc way and require significant expertise and extra code. In many cases, there is no system support for detecting particular issues. Typically, data has to be integrated first, as common libraries assume the input to be in a single table. Furthermore, specialised solutions are often incompatible with popular libraries. Even simple tasks like computing fairness metrics can be challenging. This situation is in stark contrast to the software engineering world, with established best practices and infrastructure for testing and CI.

**Provenance is all you need**. At the core of these issues is missing provenance tracking in current tooling. For example, we find that can automate the detection of many common correctness issues in ML pipelines with access to (*i*) the materialised artifacts of a pipeline (its input relations, and its outputs, e.g., the feature matrices, labels, and predictions of a classifier) as well as (*ii*) their why-provenance [4] (e.g., which input records were used to compute a particular output). This allows us to design screening techniques with low invasiveness for declaratively written ML pipelines.

**Modeling ML pipelines as dataflow computations**. We base our approach on a recently introduced model of treating ML pipelines for classification tasks as dataflow computations [1–3]. This allows us to reason about the operations and intermediate results in such pipelines. We model a classification pipeline as a dataflow computation from several input relations in a star schema to a set of ML-specific matrices, e.g., for features, labels, and predictions. The data integration stage of a classification pipeline combines the individual input datasets into a single table by conducting a series of joins. The subsequent data cleaning and filtering operations can then be modeled with selections and (extended) projections to remove tuples and remove/recompute attributes. The final feature encoding stage turns the data into matrix form, which we treat as extended projections. Formally, all this can be modeled with SPJU.

**Provenance Granularity**. ML use-cases often require record-level provenance tracking, e.g., using provenance polynomials [4]. For example, applications dealing with user data need provenance that allows for reasoning about data on a user level, e.g., to determine which model was trained on which data and to track the user data through end-to-end ML pipelines. It is often sufficient not to treat feature encoding operations as aggregations, which typically introduces significant performance overhead to provenance tracking.

**Current State & Future Work**. Based on these insights, we developed multiple prototypes. First, we developed machinery to extract intermediate results and provenance (in the form of provenance polynomials [4]) from ML pipelines with MLINSPECT [1]. It works for declaratively written Python pipelines using popular libraries. We see MLINSPECT as a runtime to enable more advanced use-cases. On top of MLINSPECT, we developed ARGUSEYES [2], which enables automatic detection of common issues w.r.t. best practices in ML, and easily hooks into continuous integration workflows. Further, we presented a prototype for Freamon [3], a more general approach to reconstruct and query selected ML pipeline intermediates and their provenance, that works with MLINSPECT, but also other provenance tracking backends, e.g., for SparkML. Our code is open source and we provide demo notebooks for MLINSPECT [5] and FREAMON [7], and a CI integration example for ARGUSEYES [6].

**Acknowledgements.** This work was supported in part by Ahold Delhaize. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

## REFERENCES

[1] Grafberger, et al. Data distribution debugging in ML pipelines. *VLDBJ* (2022).
[2] Schelter, et al. Screening Native ML Pipelines with "ArgusEyes". *CIDR* (2022).
[3] Schelter. Reconstructing and Querying ML Pipeline Intermediates. *CIDR* (2023).
[4] Green, et al.. Provenance semirings. *PODS* (2007).
[5] https://github.com/stefan-grafberger/mlinspect
[6] https://github.com/amsterdata/arguseyes-demo
[7] https://github.com/amsterdata/freamon


*WWW '23 Companion, April 30-May 4, 2023, Austin, TX, USA*

ACM ISBN 978-1-4503-9419-2/23/04.
https://doi.org/10.1145/3543873.3587557