Report date 09/01/2025

Paper report:

**Overview:**

The paper presents a system to rewrite ML pipelines with declarative APIs so that they can be more accurately and easily analyzed to highlight issues related to data management. The rewriting phase is assisted by LLMs.

ML pipelines often contain problems regarding data management (privacy issues, data leaks, data lineage, right to be forgotten enforcement…). According to the paper many of these problems are hard to identify and easy to make in imperative pipelines and in all of those pipelines which code is written quickly and just to get the job done (common in data science).

It is unrealistic to expect developers and scientists to increase the abstraction level of their pipelines and there's the need to find a way to automate this process.

In order to fill the “code abstraction gap” the paper proposes the usage of Lester, a tool that relies on an underlying LLM (namely GPT-4) and more formal pipeline abstraction tools such as mlinspect.

Pipelines are represented as datasets on which positive algebra operations and a sequence of transformer/estimator functions are applied to, in order to create the final pipeline dataset.

Via mlinspect they also consider: how-provenance, column provenance, row provenance and matrix column provenance for the feature encoding operations (shows which column dimensions in the feature matrix are used by a given column's feature encoder).

Lester uses a dataframe API built on top of Pandas where all the needed abstraction used for the tracking is implemented. For the feature encoding stage scikit-learn operators are used (estimator/transformer).

Then GPT is prompted with a standard query and a schema dependent query that contains details about the required output schema and the pipeline code itself.

Same thing is done for the feature encoding part.

The artifacts generated by this process (provenance, matrix column provenance, column provenance) also support IVM since they help us to see the pipelines as materialized views. The paper cites [Machine Unlearning of Features and Labels](https://drive.google.com/drive/folders/1FSABbXitEUto3Q1uyacYrZ4wSJ090CXa?usp=drive_link) and its influence function to implement the unlearning/IVM.

**Experimental metrics and scenarios:**

Two example scenarios are presented:

* Training data preparation pipeline that leaks credit card information, contains: hard coded schema, manual data filtering, user defined functions usage, manual join
* Feature encoding pipeline, contains: imperative numpy code usage, user defined functions usage, manual embedding via external function, manual one hot encoding

They are run on the same synthetically generated customer and mail data.

THey run up to 100,000 mail entries and 10,000 customer entries. Each run is run 7 times (except for the biggest one, only run 3 times due to long runtime).

Then some data is unlearned from the pipeline (to simulate compromised emails) and the re-execution time is compared with the unlearning time via IVM with Lester.

A user study is also conducted. The participants received a 10 minutes intro about the tool and were requested to: assess group fairness of product reviews and to track row provenance for the products and product rating relations.

4 out of 9 people completed the first task and only 2 out of 9 completed the second task in an hour.

The findings are:

* Manual code extension in non-trivial
* Manual code extension is error prone
* Manual code extension introduces high code complexity (in terms of LOC and operation complexity)

This experiment shows that manual code extension is non trivial and that data scientist may benefit from systems to support this task.

**Benchmarked against:**

Re-execution of the whole pipeline with the updated data.

**Limitations:**

Being a prototype the dataframe API used to build on top of Pandas may not support all the functions. The code will need to be reworked after it's been improved by the LLM. Issues related to LLM accuracy are reflected on Lester. Unlearning as implemented in the example is only available on differentiable model (because of the limitations of [Machine Unlearning of Features and Labels](https://drive.google.com/drive/folders/1FSABbXitEUto3Q1uyacYrZ4wSJ090CXa?usp=drive_link)).

**Repo:**

<https://github.com/deem-data/lester>

**Authors:**

**BIFOLD**

Sebastian and Stefan

**Year:**

2024