Data Quality & AI - Automated Data Validation and Cleaning

Corneliu Cofaru

Brussels, March 25, 2026

AI VET
Artificial Intelligence for VET providers

AI VET
Artificial Intelligence for VET providers

outline

- **intro**: linting in data & code quality
- **vision**: where do we want to get
- **DataLinter**: how does it work?
- **demo**: small demonstration
- **q&a**: discussion

"***Lint*** *is the computer science term for a static code analysis tool used to flag programming errors, bugs, stylistic errors and suspicious constructs. The term originates from a Unix utility that examined C language source code. A program which performs this function is also known as a* ***linter*** *or* ***linting tool***." (Wikipedia)

- At this point, linting is considered a mature technology, done mostly for code
- Data linting is problematic and receives little attention[1];
  - **why?**: context is most of the time difficult to ascertain
- Industry focuses on 'data quality' i.e. correct values, formats, semantics
- Academic interest in complex linting scenarios is just emerging[2]

[1] A data linter from Google brain team
[2] Towards a high level linter for data science (Dolcetti et al. 2024)

Why lint ?

Why lint ?

- reproducibility becomes an issue across datasets, time
- multiple programming languages
- custom policies i.e. *"never less than X samples for models of type Y"*
- with LLMs it is easy to generate code. who/what verifies that ?

Why not code my own checks ?

Why not code my own checks ?

- code is difficult to maintain
  - even more so in more than 2 languages and/or people are involved
- checks may have to be done by other teams
  - no access to the programming language nor modelling skills
- checks may need to be combined and customized by many users
- more complex analysis techniques (AST, CFG, DFG-based) require significant effort

- Our vision is to lint at higher conceptual levels and capture problematic patterns
- Examples:
  - wrong algorithmic parameters for given data
  - wrong operation flow within experiment
  - flaws in software architecture

- Independence from data type, programming language
- Knowledge-driven approach: users can implement their own logic
- Library could be used in:
  - iterative development i.e. online agent
  - production, as a stage in CI pipelines for code+data checks

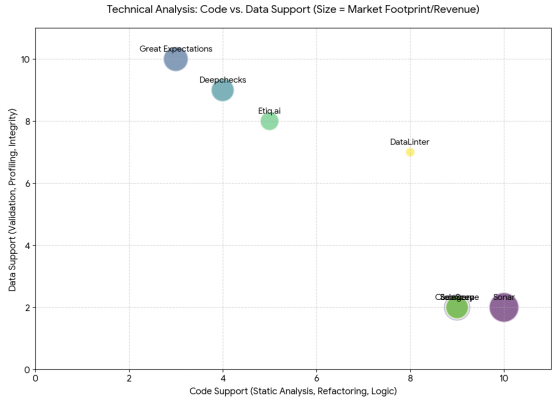


Figure 1: Competitive landscape[3]

- In industry, there is a clear separation between code and data analysis
- **DataLinter** attempts to capture data+code analysis synergies

[3]Graph obtained with Google's Gemini.

AI VET
Artificial Intelligence for VET providers

The linter can be extended to a more complete platform by combining:

- model analysis
- contextual recommendations
- background simulations (guided AutoML style)
- formalized representations[4] of experiments (in the KB)
- integration with LLMs for orchestration and capability expansion

[4] E Patterson et. al. (2018) "Teaching machines to understand data science code by semantic enrichment of dataflow graphs"

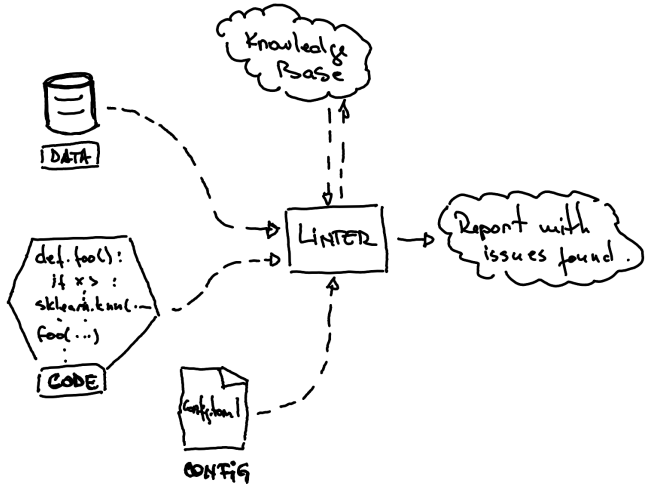
[5]We refer to semantic mistakes here

- the linter is based on a simple idea: code and data are each other's contexts
  - data issues are more apparent if we know the code that uses the data
  - coding mistakes[5] (i.e. parameters, algorithms) can be detected only with access to the data

[5] We refer to semantic mistakes here

DataLinter (dataflow)

DataLinter (dataflow)



DataLinter (elements)

DataLinter (elements)

1. **config**: input configuration, enables linters, sets parameters, context
2. **code**: code that may contribute to the context
3. **data**: the data; can be anything; tabular data is supported now
4. **KB**: knowledge base with linting rules; only a stub at the moment
5. **output**: information regarding the problems found

DataLinter (linting flow)

DataLinter (linting flow)

- the **config** is loaded first
- **data** and **code** are sent together to be linted
- for each enabled linter:
  - if the linter queries code:
    - **code** is parsed and queried
    - if the query fails, the linter is disabled
  - the linting function is applied to the **data** with context information extracted from the code
  - the output of the linter is stored

Let's see how it works!

Thank you for your attention !

You can reach us at:

- corneliu.cofaru@vub.be
- https://aivetproject.eu/