A Feasibility Study of the Python Package `splink` for Probabilistic Record Linkage (2023 Monograph)

Anders Alexandersson

06/30/2023

This 2023 monograph describes a feasibility study at the Florida Cancer Data System (FCDS) of the Python package `splink` for probabilistic record linkage (PRL). `splink` is tested on 1 million artificial records. The results suggest that it is feasible to use `splink` in linkage data requests, because `splink` was 50 times faster and more accurate than the R package `fastLink`. However, for now, `fastLink` remains better overall than `splink` by being easier to use.

Introduction

The Florida Cancer Data System (FCDS) usually uses the R package fastLink (Enamorado, Fifield, and Imai 2019; Enamorado 2021) for probabilistic record linkage (PRL). The FCDS has developed two `fastLink` templates which are R Markdown document templates on how to use `fastLink` within the RStudio IDE. The `fastLink` templates provide more automated reports in PDF. This 2023 monograph describes an FCDS feasibility study of the Python package splink (Linacre et al. 2022) for PRL. `splink` is used together with Quarto Markdown and RStudio IDE for easily reproducible results in HTML and PDF. The monograph consists of two components:

- The main text, this document: a high-level non-technical overview (5 pages)

- A supplement: a low-level technical showcase using 1 million artificial records (25 pages)

Many cancer registries use the Java-based Match*Pro for PRL. The FCDS prefers `fastLink` over `Match*Pro` because `fastLink` was 1-2% more accurate in previous testing, and `fastLink` is free (open source) and easier to use for automating reports. The current `Match*Pro` version 2.4 performed worse than `fastLink` and `splink` on the new test data, especially in terms of speed. Therefore, the FCDS has little incentive to use `Match*Pro`. Below, we will only compare `fastLink` and `splink`.

Feasibility is the possibility and ability for something to be done. Viability is that something's ability to survive. We used the TELOS framework which has five components: Technological, Economic, Legal, Organizational, and Scheduling.

PRL is usually described as one step in a data cleaning pipeline with these four steps (e.g., Binette and Steorts 2022): 1) attribute alignment, 2) blocking, 3) PRL, and 4) canonicalization. The sections below will discuss the four steps, the feasibility for three use cases, and it will conclude with recommendations.

Step 1: Attribute alignment

Step 1, attribute alignment, is the pre-processing before step 2, blocking. An attribute is an elementary feature of an entity such as address, date of birth, gender or name. The `splink` data prerequisites are aligned attributes. Some useful packages in Python for data pre-processing are metaphone, probablepeople, usaddress and the Python Record Linkage Toolkit for data preprocessing.

Step 1 is the most difficult to automate of the four steps. Work will be done to test the current FCDS automated standardization, and if acceptable will be added to the data extracted from the FCDS database.

Step 2: Blocking

`splink` requires one of three link type settings: `dedupe_only`, `link_only`, and `link_and_dedupe`. The setting names are self-explanatory except that `splink` requires extra code to remove duplicates. Step 2, blocking, is optional in both `splink` and `fastLink`. `splink` has two types of blocking rules. For step 2, blocking, `splink` is better than `fastLink` mostly by offering `OR (disjunctive)` blocking which reduces the risk of missed matches (also known as "False Negatives", FN). The `fastLink` developers are working on "probabilistic blocking" (Enamorado and Steorts 2020) where PRL is used also for blocking. However, probabilistic blocking will likely not be available in the next release of `fastLink`.

Step 3: Record linkage

In `splink`, the record linkage step consists of two sub-steps: 1) estimate model parameters, and 2) predict results. Having two sub-steps enables more complicated models which often are more accurate. Estimate model parameters means to estimate:

- λ (`lambda`): The probability that two random records (with no blocking) match.
- `u`: The probability that wrong matches match (also known as "False Positives", FP).
- `m`: The probability that true matches match (also known as "True Positives", TP).

`fastLink` and `splink` use an iterative optimization algorithm known as Expectation Maximization (EM). Two ways to get faster and more accurate results in `splink` than using the default settings are to set `lambda` manually, and to estimate `lambda` and `u` directly.

Predict results mostly means to calculate the overall match probability. A waterfall chart visually shows the calculation of the overall match probability.

In the `fastLink` templates, Table 3 is frequencies of linkage pattern and Table 4 is the confusion table. A similar frequency table is less useful for `splink` because `splink` allows more comparison values. A confusion table is very difficult to create for `splink` because it requires available labeled data. Best practice in `splink` to visualize the results seems to be to use a waterfall chart.

Step 4: Canonicalization

Step 4, canonicalization, is the post-processing after the PRL. For the `fastLink` template, the most time consuming part is the clerical review. `splink` likely will require less clerical review than the current version of `fastLink`. The showcase could not test clerical review since the data were artificial. Similar to step 1, attribute alignment, `splink` has no feature for step 4.

For the clerical review, for now the easiest is to temporarily use R as shown in the monograph supplement. The `splink` main developer, Robin Linacre, is working on a separate graphical user interface (GUI) tool for easier clerical review.

Three possible uses of `splink` at the FCDS

`splink` is much faster than `fastLink` by using the SQL back-end database DuckDB. `fastLink` completed the linkage in 100 minutes (1 hour and 40 minutes) whereas `splink` completed the linkage in 2 minutes, which is 50 times faster. `splink` is also more accurate than `fastLink` by enabling more complex models. On the test data, `fastLink` found 999,564 matches (missed 436 matches) whereas `splink` found 999,812 matches (missed 188 matches).

The largest technical issue is that steps 1 and 4 require Python programming skills for `splink` or R skills for `fastLink`. The following are feasible uses for `splink` at the FCDS, in order of suggested priority:

1. Replace `fastLink` for linkage data requests.

The largest advantage of replacing `fastLink` with `splink` for linkage data requests is much shorter completion time for steps 2 and 3, and possibly shorter completion time for step 4 by having to manually review fewer records. Replacing `fastLink` for linkage data requests with `splink` can save about 33% or 1 of 3 weeks for an average linkage data request. It would be helpful to have a graphical user interface (GUI) for `fastLink` or `splink`. ShinyLink is a web-based GUI for `fastLink` which the company Nelson Scientific Labs is developing. However, `ShinyLink` is not usable yet because it does not have blocking.

The `fastLink` developers are working on a new release for the fall which will be more accurate and faster. The improvement in accuracy will mostly come from "Active Learning" (Enamorado 2019) which is a semi-automated way of classifying record pairs that otherwise would require clerical review ("possibles"). The author expects about 1-2% improved accuracy and 5-10 times less clerical review thanks to active learning in `fastLink`. The next `fastLink` version is expected to have the Damerau-Levenshtein string distance comparator from the package `stringdist` for better partial matching of Social Security Number. `splink` uses Damerau-Levenshtein from `DuckDB`. The next version of `fastLink` will not use `DuckDB` because `DuckDB` does not perform well with input datasets over 2 million records. Instead, `fastLink` is expected to be 50% faster than now by using sampling to estimate model parameters. That would make the expected new run time on the test data about 50 minutes or 25 times slower than `splink`; see the monograph supplement for the details.

2. Reduce the amount of manual review for `Match*Pro` de-duplication.

NAACCR requires the use of `Match*Pro` for de-duplication of the annually submitted data. The much faster and more accurate `splink` could be useful. A `splink` template for linkage data requests will make `splink` easier to use for de-duplication. However, a `splink` template for de-duplication is better if de-duplication is the priority.

3. Replace real-time deterministic record linkages with real-time PRL.

`splink` is fast enough to replace the FCDS real-time (also known as "spine-based") deterministic record linkages with real-time PRL. The `splink` Github webpage has an example of real-time PRL. In general, this third use case is the most high-risk and high-reward because it affects operational processing. The FCDS relies on the consulting expertise of Advanced Consulting Enterprises for the current (SQL) code. The Julia package SpineBasedRecordLinkage is possible for real-time *deterministic* record linkages but to review it is outside of the scope here.

Recommendation

It is feasible to use `splink` at the FCDS, especially for replacing `fastLink` in linkage data requests, because `splink` was 50 times faster and more accurate on the new test data. However, for now, `fastLink` remains better overall than `splink` for the FCDS by being easier to use. The recommendation is to continue to improve PRL at the FCDS as the 2024 monograph. For now, the proposed topic of the 2024 monograph is titled "More Accurate Probabilistic Record Linkage using `fastLink` with Active Learning". In order of priority, these are the specific recommendations:

- Improve the accuracy and speed of `fastLink` by using the expected new release.
- Improve the user-friendliness of `fastLink` by updating the FCDS templates from R Markdown to Quarto Markdown. It will enable new useful features such as multi-format support and code annotation.
- Improve the user-friendliness of `fastLink` and `splink` by reducing the need for data cleaning. Decide if and how "standardized" variables in the FCDS database can be used or created for the linkage data requests.
- Work with Abraham Flaxman to improve and make public the artificial test data. The test data need to have more noise to test how accurate PRL is in terms of wrong matches (FP).

References

Binette, Olivier, and Rebecca C. Steorts. 2022. "(Almost) All of Entity Resolution." *Science Advances* 8 (12): 1–14. https://doi.org/10.1126/sciadv.abi8021.

Enamorado, Ted. 2019. "Active Learning for Probabilistic Record Linkage." *SSRN e-Print*, 1–37. http://dx.doi.org/10.2139/ssrn.3257638.

———. 2021. "A Primer on Probabilistic Record Linkage." In *Handbook of Computational Social Science, Volume 2*, edited by Uwe Engel, Anabel Quan-Haase, Sunny Xun Liu, and Lars Lyberg, 95–107. Routledge. https://doi.org/10.4324/9781003025245-8.

Enamorado, Ted, Bejamin Fifield, and Kosuke Imai. 2019. "Using a Probabilistic Model to Assist Merging of Large-scale Administrative Records." *American Political Science Review* 113 (2): 353–71. https://doi.org/10.1017/S0003055418000783.

Enamorado, Ted, and Rebecca C. Steorts. 2020. "Probabilistic Blocking and Distributed Bayesian Entity Resolution." In *Privacy in Statistical Databases*, edited by Domingo-Ferrer J. and Muralidhar K., 224–39. Cham, Switzerland: Springer Lecture Notes in Computer Science. Volume 12276. https://doi.org/10.1007/978-3-030-57521-2_16.

Linacre, Robin, Sam Lindsay, Theodore Manassis, Zoe Slade, and Tom Hepworth. 2022. "Splink: Free Software for Probabilistic Record Linkage at Scale." *International Journal of Population Data Science* 7 (3): 23. https://doi.org/10.23889/ijpds.v7i3.1794.