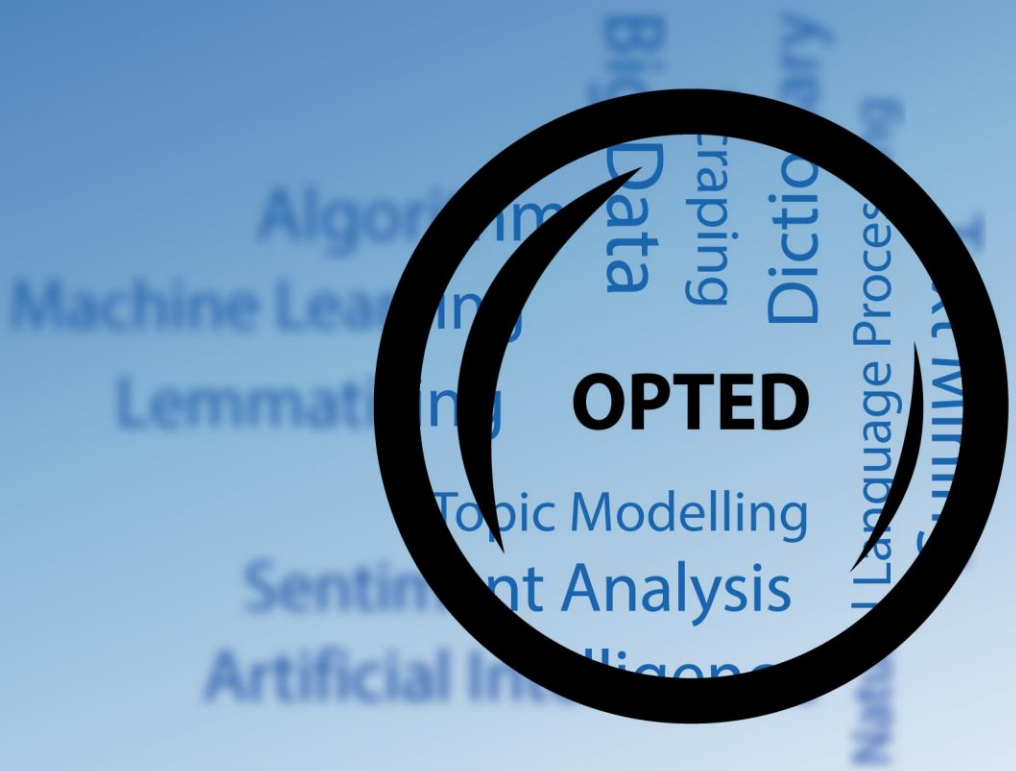
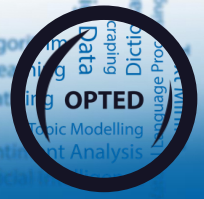
OPTED

A tool that facilitates and partly automatizes data linkages: The Joinery Package

Matt W. Loftis. Anne Kroon, Jeroen Jonkman, Rens Vliegenthart, Christoffer Green-Pedersen, Shaun Bevan




Data
OPTED
Topic Modelling

Disclaimer

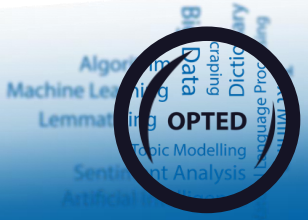
This project has received funding from the European Union's Horizon 2020 research & innovation programme under grant agreement No 951832. The document reflects only the authors' views. The European Union is not liable for any use that may be made of the information contained herein.

Dissemination level

Public/ Confidential (Please select based on Table 3.1d in the Proposal: PU = Public, CO = Confidential)

Type

Report/Website/Demonstrator (Please select based on Table 3.1d in the Proposal: R = Report, DEC = Websites, Patent, filling, DEM = Demonstrator)



OPTED
Observatory for Political Texts in European Democracies:
A European research infrastructure

A tool that facilitates and partly automatizes data linkages: The Joinery Package

Deliverable 8.4

Authors: Matt W. Loftis[1], Anne Kroon[2], Jeroen Jonkman[2], Rens Vliegenthart[3], Christoffer Green-Pedersen[4], Shaun Bevan[4]

[1]*Department of Political Science, Aarhus University*

[2]*Department of Communication, University of Amsterdam*

[3]*Strategic Communication, Wageningen University*

[4]*School of Social and Political Science, University of Edinburgh*

Due date: September 2023

Executive Summary

We introduce Joinery, a set of tools designed to address linkage challenges encountered by researchers working with textual data in the field of political science research. This tool offers functions for aggregation, standardization, merging, and documentation for political science research. Joinery is accessible in both R and Python programming languages, making it widely available to social scientists. Joinery aims to assist researchers in automating repetitive and time-consuming data analysis pipeline tasks.

1 Introduction

Researchers often encounter challenges when dealing with diverse and unstandardized datasets, particularly when working with textual identifiers like country names, political party names, and institution names. The infrastructure for political text analysis such as developed in the OPTED project this requires systematic procedures to link datasets from different sources. Joinery aims to address common challenges by providing a set of specialized functions to efficiently handle data cleaning, aggregation, and merging tasks, ensuring accuracy, consistency, and reproducibility in political science research.

Joinery is a collection of tools designed to simplify the process of merging; primarily text-based data used in political science research. To facilitate a broad range of social scientific researchers, these tools have been developed and made open-access using the programming language R and Python.

The tool provides facilitates the integration of data from the handling data from sources like the Comparative Agendas Project (CAP) (Baumgartner et al., 2013; Bevan, 2017) and/or the Manifesto Project (CMP) (Lehmann et al., 2023).

2 R-based implementation

The core version of Joinery is written in R, and can be found at https://github.com/mattwloftis/joinery. Figure 1 provides a snapshot of the tool.

Figure 1 JOINERY, R BASED

README.md

joinery

`joinery` is a set of tools to ease the process of merging (primarily text) data used in political science research. `joinery` can help with the following common tasks:

- check and clean common identifiers like country names, political party names, names of institutions, etc.
- aggregate or disaggregate the input datasets
- execute joins
- document each data transformation

See the OPTED project for principles and heuristics applied by joinery and for project updates.

Installation

You can install the development version of joinery from GitHub with:

```
# install.packages("devtools")
devtools::install_github("mattwloftis/joinery")
```

Example: Disambiguating party names

This is a basic example which shows you how to disambiguate party names in a dataset from the *Comparative Agendas Project*:

```
library(joinery)

## basic example of disambiguating party names

# Comparative Agendas Project data on Spanish political party manifestos
spain_cap_manifestos <- read.csv("https://comparativeagendas.s3.amazonaws.com/datasetfiles/Party_Ma

# disambiguate the party names
spain_cap_manifestos <- spain_cap_manifestos %>%
  disamb_party(party_ref = "politicalparty",
               country = "Spain",
               year = "year",
               origin = NULL)
#>
```

3 Python based implementation

A Python adaptation of Joinery has been developed, and can be found at https://github.com/annekroon/joinery-python. See figure 2.

Figure 2 JOINERY, PYTHON BASED

Functions and Methods

The Jnry class provides the following functions and methods:

`get_jnry_year(yr)` : This method parses the provided yr and returns the year as an integer. If yr is not provided, it returns the current year.

`get_jnry_country()` : This method converts the provided country to the ISO3 country code format using the `country_converter`` library. If the conversion fails, it returns the original country name.

`get_unique_country_year_combinations()` : This method returns a dictionary of the unique combinations of years and political parties found in the target DataFrame. These combinations will be matched with the PartyFacts data.

`get_party_facts()` : This method downloads the PartyFacts data using the provided URL and filters it based on the jnry_country.

`get_party_facts_ids()` : This method performs the matching between the target DataFrame and PartyFacts data. It returns a list of dictionaries, each containing information about the matched political party, including the year, jnry_year, politicalparty, partyfactors_id, and wikipedia.

`merge_party_facts_with_target()` : This method merges the PartyFacts data with the target DataFrame based on the year and politicalparty columns. The resulting DataFrame contains the original data from CMP or CAP, along with additional columns containing PartyFacts information for each unique combination of political parties and years.

Installation

First, make sure you have Python installed on your system. To install Joinery and its dependencies, run the following command:

4 Key functionalities

The key functionalities of Joinery include:

1. **Check and clean identifiers.** Joinery offers functionality to check and clean common identifiers. In this way, the tools helps to ensure that data across different datasets are matched in similar ways, and errors and inaccuracies during merging are

avoided. In particular, joinery performs sanity checks for year and country variables. In case missing values are present, it will identify them. Additionally, it will transform countries to ISO country codes.

2. **Data aggregation and disaggregation.** Researchers often need to aggregate data to different granularities or disaggregate it for in-depth analysis. Joinery provides tools to aggregate data over various time scales, such as months, weeks, or quarters. It allows you to convert data from more granular time scales to less granular ones.
3. **Joins and merges.** Joinery facilitates the execution of joins between datasets. Researchers can merge datasets based on common fields. This facilitates the merging of different sources into a unified dataset for further analysis.
4. **Documentation of each data transformation.** Joinery generates detailed documentation regarding the transformation steps, with the aim to ensure transparency and reproducibility. Information about the type of matching (such as *exact*, *heuristic* or *fuzzy* matching) is provided. In the case of fuzzy matching, a url to relevant Wikipedia pages is provided, so users can manually validate ambiguous matches.
5. **Integration of Party Facts data.** Joinery allows to integrate data with the Party Facts dataset (Döring & Regel, 2019).Party Facts offers standardized linked data on political parties. Consequently, it provides users with unique numeric identifiers for recognized political parties. These identifiers facilitate the linkage of user data with the Party Facts dataset and, in turn, open doors to various political science datasets for further enrichment and analysis. It can take data from the comparative manifesto project as well as the comparative agendas project, and link this data with Party Facts.
6. **Handling missing data.** Joinery includes functionalities to handle missing data and fill in gaps for time points not represented in the input datasets. Herewith, Joinery hopes to ensure a more complete and consistent dataset for analysis.

5 Differences between the R-based and Python-based version

In the Python-based version of Joinery, less emphasis is placed on aggregation since Python provides straightforward mechanisms for handling such tasks (e.g., using panda's .groupby() methods). On the contrary, the Python-based implementation includes code to access the API of the Comparative Manifesto Project and fetch data from the Comparative Agendas Project. This feature enables researchers to integrate data from these projects into their analysis pipelines (see also OPTED Deliverable 8.5).

6 Conclusion

Joinery is developed for researchers in the field of political/ communication science, where textual data from various sources is commonly used. Joinery aims to assist researchers in automating repetitive and time-consuming data analysis pipeline tasks. The current scripts in R and Python offer solid procedures to deal with most of the common challenges in linking political (text) datasets and can be readily extended in the future for more advanced applications and linkage strategies.

References

Baumgartner, F. R., Green-Pedersen, C., & Jones, B. D. (2013). *Comparative studies of policy agendas*. Routledge.

Bevan, S. (2017). Gone fishing: The creation of the comparative agendas project master codebook. version 0.9. 1 beta. *URL: http://sbevan. com/cap-master-codebook. html*.

Döring, H., & Regel, S. (2019). Party facts: A database of political parties worldwide. *Party politics*, *25* (2), 97–109.

Lehmann, P., Franzmann, S., Burst, T., Regel, S., RiethmÃ¼ller, F., Volkens, A., WeÃŸels, B., & Zehnter, L. (2023). The manifesto data collection. manifesto project (mrg/cmp/marpor). version 2023a. https://doi.org/10.25522/manifesto.mpds.2023a