

Development of a Python Package for Matching Observational Data

Master's Thesis Presentation

Jack Potrykus

University of Chicago Department of Statistics

May 5, 2022

Contents

Introduction

Literature Review

Measuring Distance

Matching Algorithms

`matching`: a Python Package for Matching Observational Data

Experiments

Caliper vs. Imbalance

Caliper vs. Correlation

Conclusion

Introduction

Introduction

Problem Setting

- ▶ Observational studies frequent in econometrics, psychology, and medical research.
- ▶ This presentation: binary treatment/control variable, \mathbf{z} .
- ▶ Most often, goal is to estimate ATE: $\mathbb{E}[y_{i1} - y_{i0}]$.
- ▶ Solve this problem via *matching* treatment observations to control observations
 - ▶ Produce a matched subset suitable for analysis
- ▶ We break this down into two orthogonal problems:
 - ▶ How is distance measured?
 - ▶ How are matches assigned?
- ▶ `matching`: Python package for matching observational data.

Introduction

Notation

Now, we will introduce some notation, so we can precisely talk about construction of different matching methods.

- ▶ $X \in \mathbb{R}^{(n+m) \times p}$ of features; $X_T \in \mathbb{R}^{n \times p}$, $X_C \in \mathbb{R}^{m \times p}$, $n \leq m$;
- ▶ $\mathbf{z} \in \{0, 1\}^{n+m}$ of binary treatment assignments;
- ▶ $d : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}^+$ is a distance measure between feature vectors, e.g. $d(\mathbf{x}_{T_i}, \mathbf{x}_{C_j})$;
- ▶ $\mathcal{D}_d : \mathbb{R}^{n \times p} \times \mathbb{R}^{m \times p} \rightarrow \mathbb{R}^{n \times m}$ produces *biadjacency matrix*;
- ▶ Some procedures use a preprocessing function f (usually: dimension reduction or coarsening of the data).

Our framework is similar to that of Iacus, King, and Porro (2011):

$$\mathcal{D}_d(f(X)_T, f(X)_C). \quad (1)$$

Literature Review

Measuring Distance

Balancing Scores

Definition (balancing score, Rosenbaum and Rubin (1983))

A function $b(X) : \mathbb{R}^{(n+m) \times p} \mapsto \mathbb{R}^{n+m}$ is a balancing score iff

$$X \perp \mathbf{z} | b(X). \quad (2)$$

- ▶ Idea: balancing score makes observational studies more like randomized studies
- ▶ If $\mathbf{y}_0, \mathbf{y}_1 \perp \mathbf{z} | X$, then $\mathbf{y}_0, \mathbf{y}_1 \perp \mathbf{z} | b(X)$

These methods use L^1 distance by convention:

$$\mathcal{D}_{L^1}(b(X)_T, b(X)_C). \quad (3)$$

Measuring Distance

The Propensity Score

Definition (propensity score, Rosenbaum and Rubin (1983))

The *propensity score* is the balancing score $b(X) = \mathbb{E}[z|X]$.

- ▶ Implicitly “weights” features by heterogeneity wrt \mathbf{z} .
- ▶ Often prune potential matches with a caliper c .
- ▶ Even if not used for matching, a useful diagnostic (Dehejia and Wahba 1999).
- ▶ Continuous scores: Imai and Dyk (2004).

Measuring Distance

The Prognostic Score

Definition (prognostic score, Hansen (2008))

The *prognostic score* is the function $b(X) = \mathbb{E}_C[\mathbf{y}|X]$.

- ▶ Note that the expectation is with respect to control data C .
 - ▶ No information about ATE
 - ▶ Previous attempts to match on outcomes failed, such as Miettinen (1976).
 - ▶ “conditionality principle” Hansen (2008).
- ▶ Implicitly “weights” features by heterogeneity wrt \mathbf{y} .

Measuring Distance

(Almost) Exact Matching

Most notable criticism of balance scores comes from King and Nielsen (2019).

- ▶ “Lower standard”.
- ▶ “Paradox of Propensity Score Matching”.

What do they suggest instead?

- ▶ Coarsened Exact Matching (CEM) (Iacus, King, and Porro 2012). No balance checks required!
- ▶ Observation weights:

$$w_i = \begin{cases} 1, & i \in \text{treatment group} \\ \frac{m_C}{m_T} \frac{m_T^s}{m_C^s}, & i \in \text{control group} \end{cases}. \quad (4)$$

- ▶ ML extensions (Gupta et al. 2021): DAME (Liu et al. 2019), FLAME (Wang et al. 2021)

Measuring Distance

L^P norms and Mahalanobis distance

Norm-based measurements are rarely used without some preprocessing function f . Why? Scaling.

However, Mahalanobis is often used as a *second* step.

- ▶ Initial filter: balancing score with a caliper. Then, match on Mahalanobis distance.
- ▶ “Iterative” approach performs well in practice (Baltar, Sousa, and Westphal 2014).
- ▶ MatchIt has limited support (Ho et al. 2011).

Bipartite Matching Algorithms

Define a matching \mathcal{M} as a set of tuples (i, j) indicating a matching between observations T_i and C_j . “Optimal” matching via the Hungarian algorithm (Munkres 1957).

$$\min_{\mathcal{M}} \sum_{(i,j) \in \mathcal{M}} d(\mathbf{x}_{T_i}, \mathbf{x}_{C_j}), \text{ s.t. } \forall (i, j), (k, l) \in \mathcal{M}, i = k \iff j = l. \quad (5)$$

- ▶ Extendable to $1 : k$ matching via b -matching.
- ▶ Slow: $O((k \cdot n + m)^3)$; approximation: b -SUITOR (Khan et al. 2016).

Greedy matching: sort edges by distance, pop matches off the top.

- ▶ When control observations may not be reused, no optimality assured.
- ▶ Problems when matching $1 : k$ (Rosenbaum 1989).

matching: a Python Package for Matching Observational Data

matching

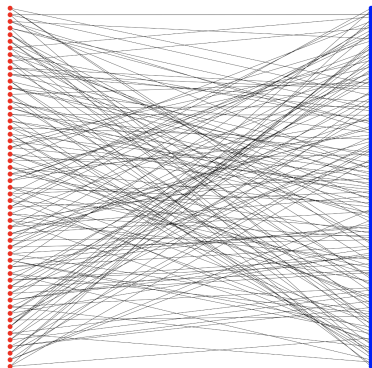
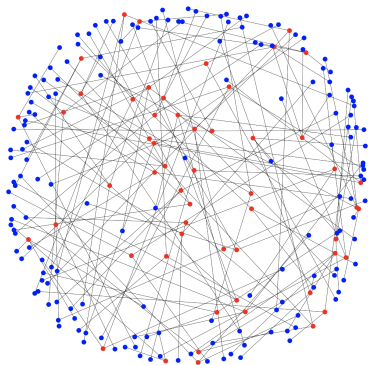
Design Goals

- ▶ The main data structure used is a bipartite graph.
- ▶ It supports iterative distance measures of arbitrary length/complexity.
- ▶ It supports filtering by individual matches (edge), observation (node), and match-group/strata (subgraphs)
- ▶ Includes a preprocessing module for coarsening data and a balance assessment module.

matching

Drawing Graphs

Example graphs after conducting 1 : 3 optimal matching.



Experiments

Experiments

Overview

- ▶ Hyperparameters n , m , p , θ_0 , θ_1 , and ρ .

$$\boldsymbol{\mu}_0 \sim \mathcal{N}(\theta_0, 0.25), \boldsymbol{\mu}_1 \sim \mathcal{N}(\theta_1, 0.25) \quad (6)$$

$$\{\mathbf{x}_C\}_{j=1}^m \sim \mathcal{N}(\boldsymbol{\mu}_0, \Sigma), \{\mathbf{x}_T\}_{i=1}^n \sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma) \quad (7)$$

$$\text{where } \Sigma_{ab} = \begin{cases} 1 & \text{if } a = b \\ \rho & \text{otherwise.} \end{cases} \quad (8)$$

- ▶ Metrics: mean ASMD¹, proportion of treatment observations matched.
 - ▶ In practice: use more robust metrics (Basu, Polsky, and Manning 2008; Zhu, Savage, and Ghosh 2018).
- ▶ Vary calipers $c \in \{0.05, 0.10, 0.15, \dots, 0.50\}$.

¹Absolute Standardized Mean Difference

Experiments

Caliper vs. Imbalance

This will consider the effects of increased heterogeneity between T and C .

- ▶ $m = 750$

- ▶ $n = 250$

- ▶ $p = 5$

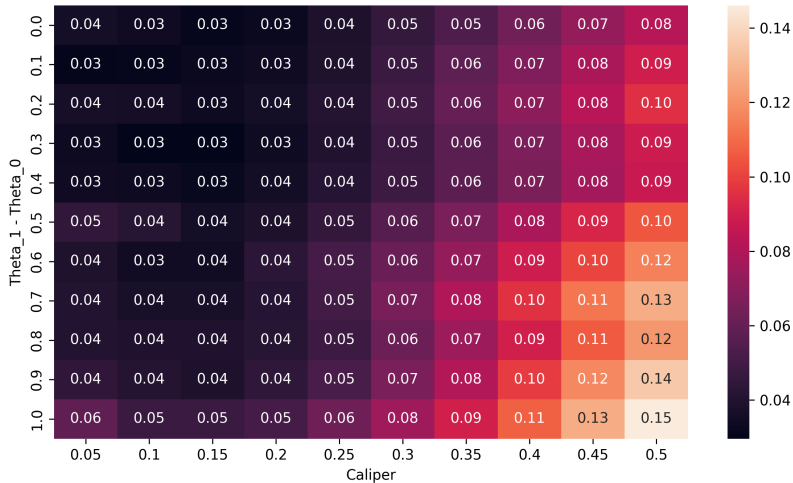
- ▶ $\rho = 0$

- ▶ $\theta_0 = 0$

We vary $\theta_1 \in \{0, 0.1, 0.2, \dots, 1\}$.

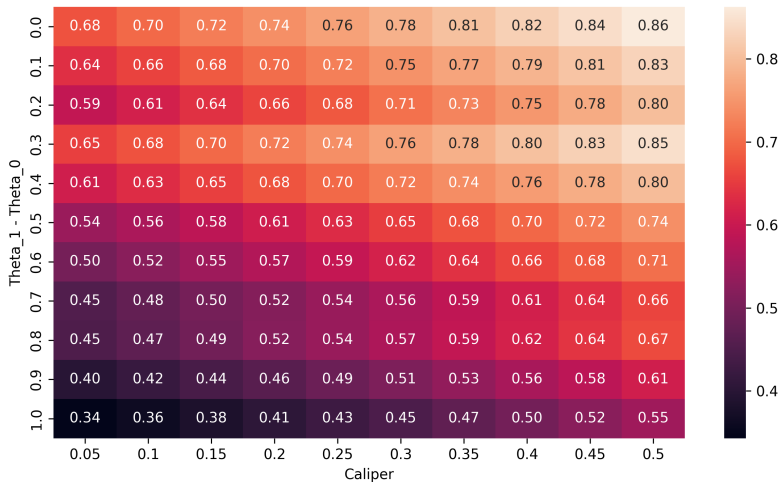
Experiments

Caliper vs. Imbalance: Mean Absolute Standardized Mean Difference



Experiments

Caliper vs. Imbalance: Proportion of Treatment Observations Matched



Experiments

Caliper vs. Correlation

This will consider the effects of increased correlation between the features of X

- ▶ $m = 750$

- ▶ $n = 250$

- ▶ $p = 5$

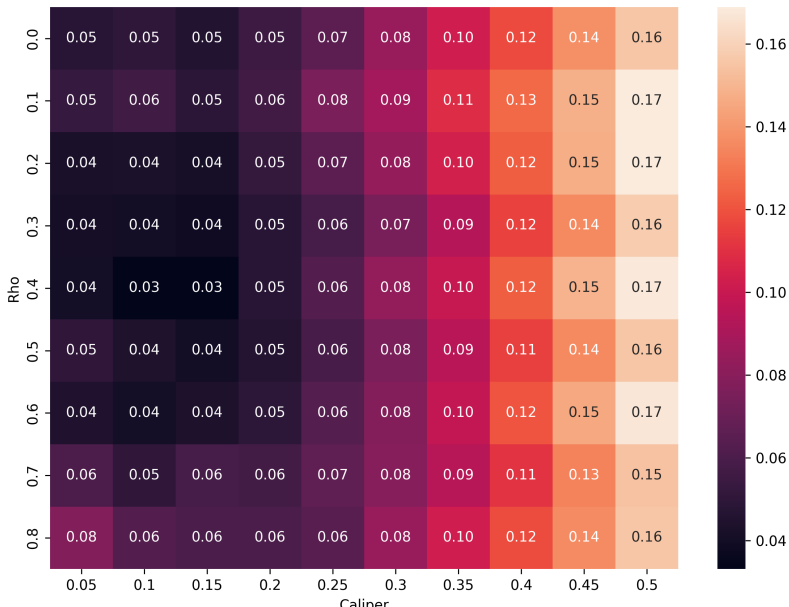
- ▶ $\theta_0 = 0$

- ▶ $\theta_1 = 1$

We vary $\rho \in \{0, 0.1, 0.2, \dots, 0.8\}$.

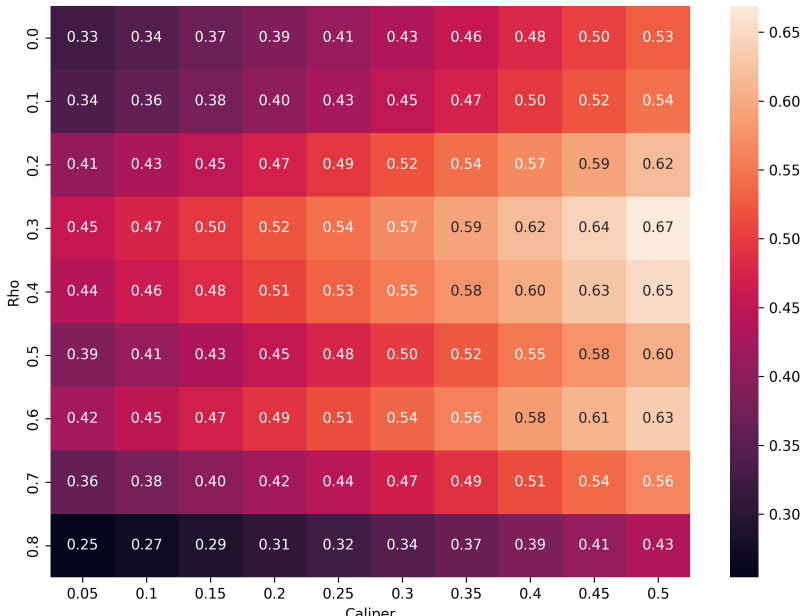
Experiments

Caliper vs. Correlation: Mean Absolute Standardized Mean Difference



Experiments

Caliper vs. Correlation: Proportion of Treatment Observations Matched



Conclusion

Conclusion

- ▶ Common theme of numerical studies: data-dependence.
- ▶ Emphasizes the importance of
 - ▶ quality of balance metrics;
 - ▶ inspectability of matching.
- ▶ `matching` offers extreme flexibility in design of matching process.
 - ▶ The current state of the graph is always inspectable.
 - ▶ Can easily tune matches.

matching Example Code

```
1  from matplotlib import pyplot as plt
2  from matching.distance import Exact, L1Norm
3  from matching.graph import MatchingGraph
4  from matching.preprocessing import propensity_score
5
6  # Take `caliper_width` as given
7  # `X` has columns "score" and "is_nice", `z` is treatment assignments
8  mg = MatchingGraph(X, z)
9
10 # NOTE: there is also a keyword argument "exclude"
11 mg.set_edges(distance=L1Norm(max_distance=caliper_width), include=["score"])
12 mg.filter_edges(distance=Exact(), include=["is_nice"])
13
14 # Conduct optimal matching
15 mg.match(n_match=3, min_match=3, method="optimal")
16
17 # Get the matched data as a `pandas.DataFrame`
18 match_df = mg.match_data.frame
19
20 # Draw the graph of matches
21 mg.draw()
22 plt.show()
```