Final Project: Synthetic Control Shiny App

Minerva University

CS130: Statistical Modeling: Prediction and Causal Inference
Professor: Diamond
December 16th, 2025

Our Online Shiny App:
`https://hakkei.shinyapps.io/synthetic-control-analysis/`

1 Introduction

1.1 Problem Motivation

Synthetic Control is a method in causal inference that estimates what would have happened to a treated unit if it had not received the treatment. It does this by creating a weighted combination of control units to act as a comparison. The method is powerful but often hard to run without coding. A Shiny app makes it easy and accessible by providing a simple interface where users can upload data and run the full analysis without writing any code.

1.2 Project Scope

This project presents a Shiny application that performs the main steps of Synthetic Control: uploading a dataset, selecting the treated unit and treatment year, computing the synthetic control, showing clear visual results, and running optional placebo tests. The app provides all core outputs in an organized and user-friendly way.

1.3 Design Philosophy

The app is built to be simple, intuitive, and statistically correct. The goal is to reduce user effort while keeping the method transparent and reliable. Every feature is designed to minimize friction so that users can complete an analysis quickly and understand the results easily.

2 App Features, Functionality, and Technical Overview

The Synthetic Control Shiny App allows users to run a full synthetic control analysis with no coding. It brings together data upload, variable mapping, predictor selection, synthetic control computation, placebo testing, visualization, and export tools in one interface. This section also includes a technical overview of how the app is implemented internally.

2.1 Data Upload, Structure, and Validation

Users can upload CSV or Excel files. The app accepts any panel style dataset as long as it includes a unit identifier, a time variable, and an outcome variable. After uploading, the available columns appear automatically in dropdown menus.

Validation at upload is intentionally lightweight. The helper function `validate_panel_data()` checks for:

- the presence of data,
- at least three columns,
- empty columns,
- very small datasets.

The app does not check for numeric types, missing required columns, or whether at least two pre treatment years exist. These checks happen later when the actual analysis is executed, at which point the Synth wrappers enforce stricter conditions such as requiring multiple donor units and sufficient pre treatment observations.

2.2 Configuration of Variables and Predictors

After uploading the dataset, users map:

- the unit identifier,
- the time variable,
- the outcome variable.

The treated unit and treatment year are selected from the observed data. All remaining units automatically become the donor pool.

Predictors can be chosen in two ways:

- **Regular predictors**: computed as pre treatment means.
- **Special predictors**: computed over user defined pre treatment year windows.

If no predictors of any kind are chosen, the app automatically creates lag predictors using outcome values from each pre treatment year. This matches standard practice in the Synth package and ensures that the model always has valid predictors.

2.3 Synthetic Control Computation

Internally, all synthetic control calculations rely on the official `Synth` R package. The app prepares the necessary matrices using `dataprep()`, then runs `synth()` to estimate the synthetic unit.

Key implementation details:

- donor weights are non negative and sum to one,
- RMSPE is computed manually for transparency,

- predictor balance tables are generated,
- the synthetic series and gap (treatment effect) series are returned for all time periods.

Multiple helper scripts in the `functions/` directory handle these steps:

- `synth_wrapper.R` for the main model,
- `synthetic_control_simple.R`, `robust`, and `bulletproof` variations,
- `synthetic_control_functions.R` for support utilities.

2.4 Visualization and Downloads

All plots are generated with `ggplot2` using custom plotting functions from `plotting_functions.R`.

The app displays:

- an Actual vs Synthetic plot,
- a Gap (treatment effect) plot,
- donor weight barplots,
- pre treatment balance tables.

Users can download:

- PNG images of individual plots,
- all plots in a ZIP file,
- CSV files for synthetic series, donor weights, and balance tables,
- bundled ZIP files with all data outputs.

2.5 Placebo Tests

Two placebo methods are implemented through `placebo_tests.R`:

- **In space placebos**: treat each donor unit as if it were the treated unit; this produces a distribution of placebo effects.
- **In time placebos**: assign multiple fake treatment years to the treated unit.

Each placebo test returns:

- placebo gap paths,
- RMSPE ratios,
- empirical p value style rankings.

These "p values" are based on how extreme the treated unit's RMSPE ratio is compared to placebo ratios, not formal hypothesis tests.

2.6 Performance and Stability

A standard synthetic control run is fast on small and medium sized datasets. In space placebo tests are more computationally expensive because they re run the optimization for every donor unit. The app shows progress notifications throughout.

Errors are handled with friendly messages using Shiny's `showNotification()` and `req()` wrappers, which ensures the app remains stable even when datasets are messy.

2.7 User Experience and Settings Storage

The app follows a simple workflow:

- Upload,
- Configure,
- Run Analysis,
- Run Placebos,
- Export.

To make the experience smoother, the app stores user settings in the browser using IndexedDB, with a fallback to `localStorage`. This allows the app to restore column mappings, predictors, and parameter settings when the user reopens the same dataset. The app does not save entire analysis histories.

Collapsible panels, warnings, and success messages help guide the user through the workflow.

3 User Guide

This section provides instructions on how to use the Synthetic Control Shiny App for performing causal inference analysis with synthetic control methods.

3.1 How to Upload Data

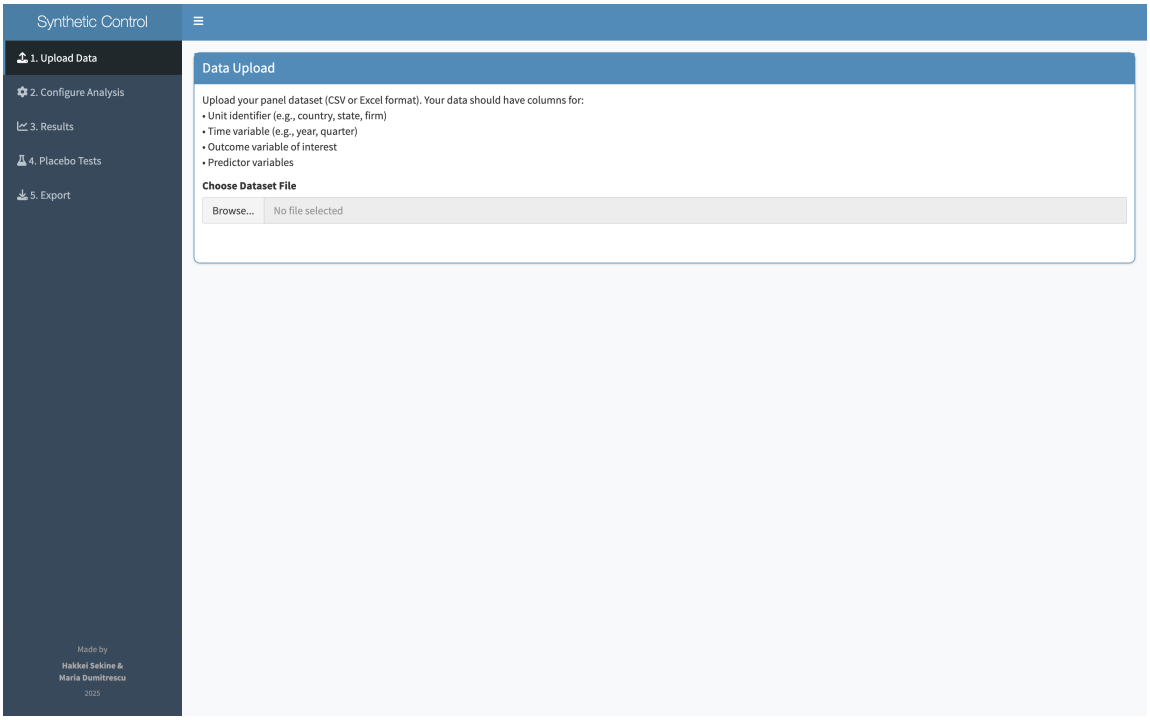

Figure 1: Upload Dataset

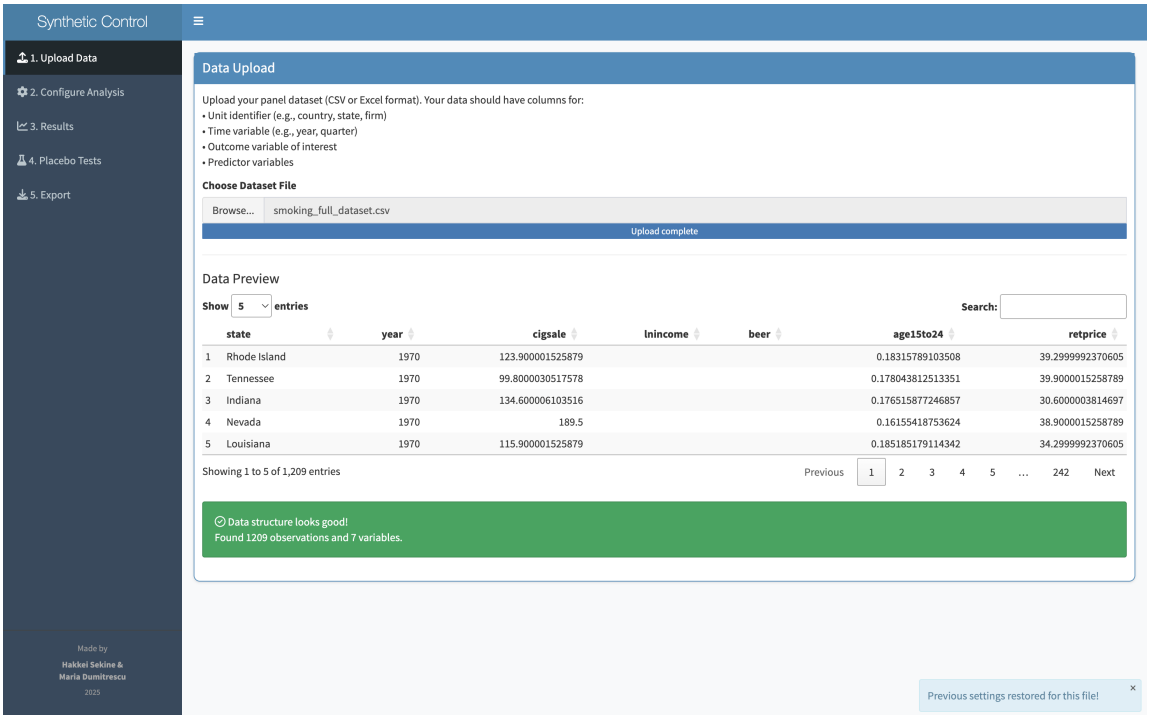

Figure 2: Upload from files

To begin the analysis, you need to upload your panel dataset. The app supports CSV formats. Follow these steps:

1. Click on the `Browse...` button to select your dataset file.
2. Once the file is uploaded, a preview of the first few rows will appear.
3. The app will automatically check the structure of the data and ensure it has at least the required columns: unit identifier, time variable, outcome variable, and optional predictor variables.

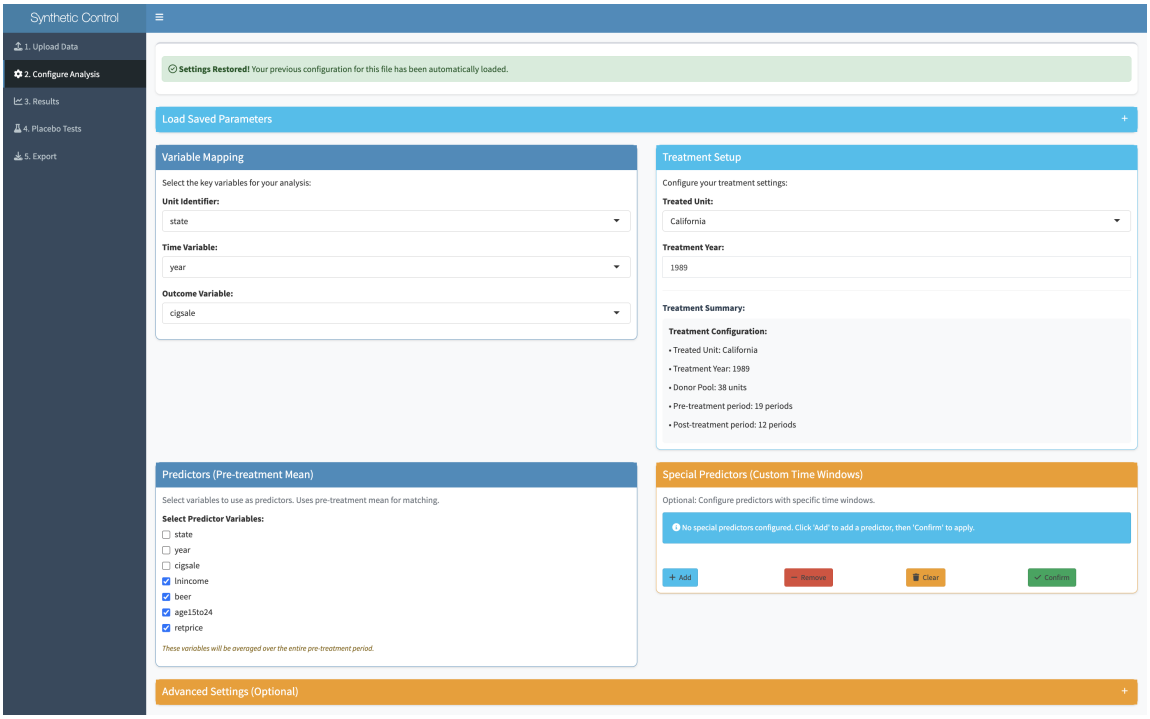

Figure 3: Add variables for the analysis

3.2 How to Map Columns

After uploading the dataset, you need to map the variables for the analysis:

1. Select the **Unit Identifier** (e.g., ‘state‘).
2. Select the **Time Variable** (e.g., ‘year‘).
3. Select the **Outcome Variable** (e.g., ‘cigsale‘).
4. Select any **Predictors** to include in the analysis (e.g., ‘lnincome‘, ‘beer‘, ‘age15to24‘, ‘retprice‘).

3.3 How to Select Treatment and Predictors

Next, set up the treatment:

1. Choose the Treated Unit (e.g., ‘California‘).
2. Set the Treatment Year (e.g., ‘1989‘).
3. Choose predictors for the pre-treatment period (e.g., ‘cigsale‘, ‘lnincome‘).

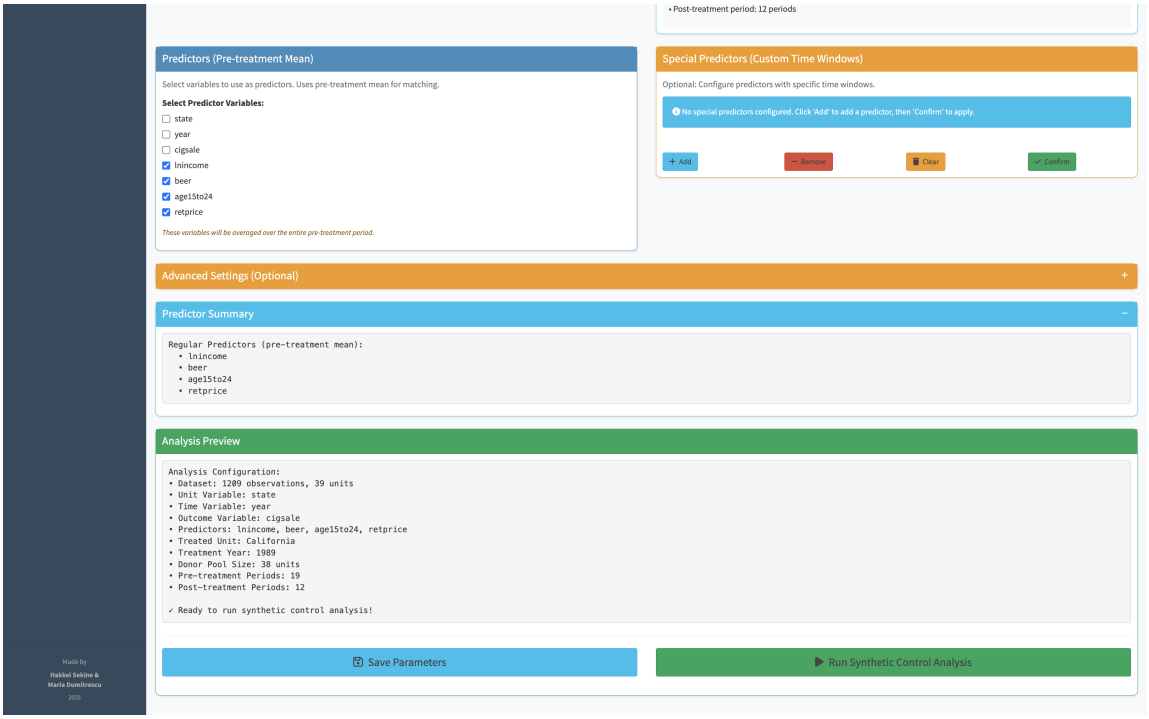

Figure 4: Run the synthetic control analysis

3.4 Optional Settings

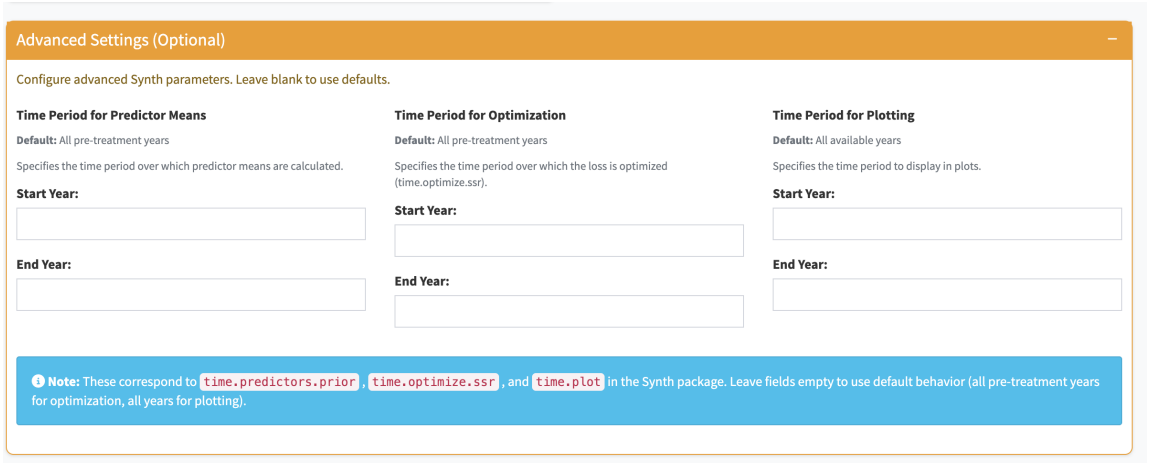

Figure 5: Advance setting for time period

1. Optionally select **Special Predictors**.
2. Optionally define **Time Period** for Predictor Means, Optimization and Plotting.

3.5 How to Run the Analysis

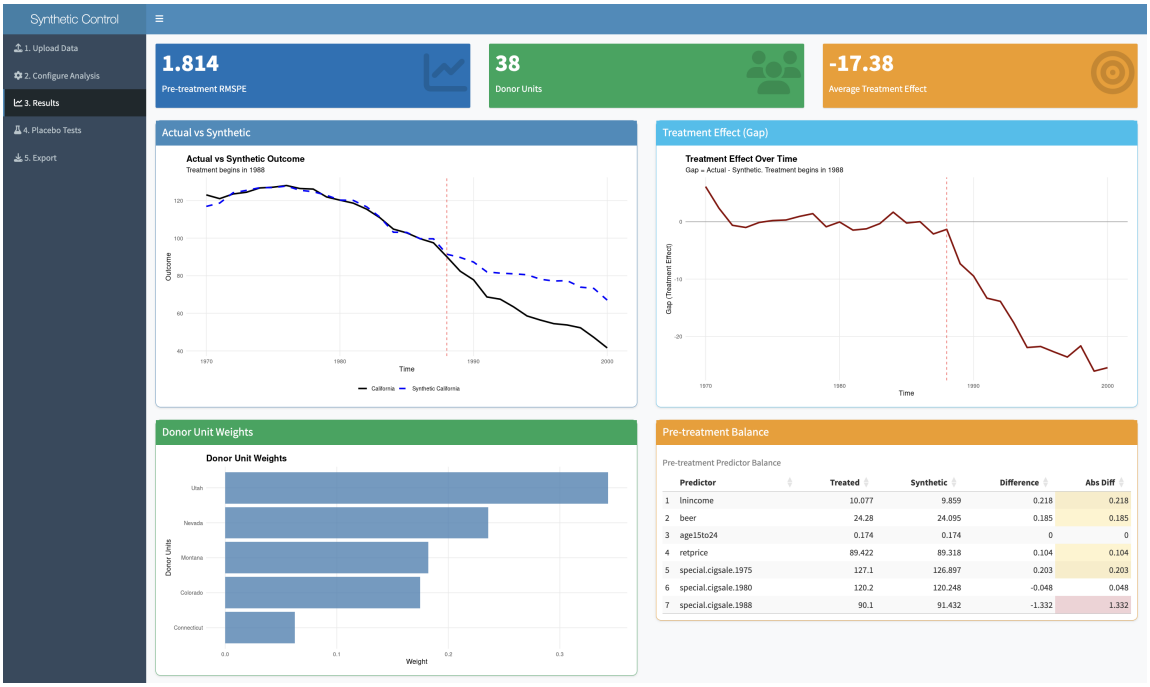

Figure 6: See the analysis results

Once the data and treatment are set up, you can run the synthetic control analysis:

1. Click the `Run Analysis` button.
2. The app will calculate the synthetic control using the selected predictors and treatment setup.
3. Results will include the Actual vs. Synthetic plot, Gap (treatment effect) plot, weights and pre-treatment balance.

3.6 How to Run Placebo Tests

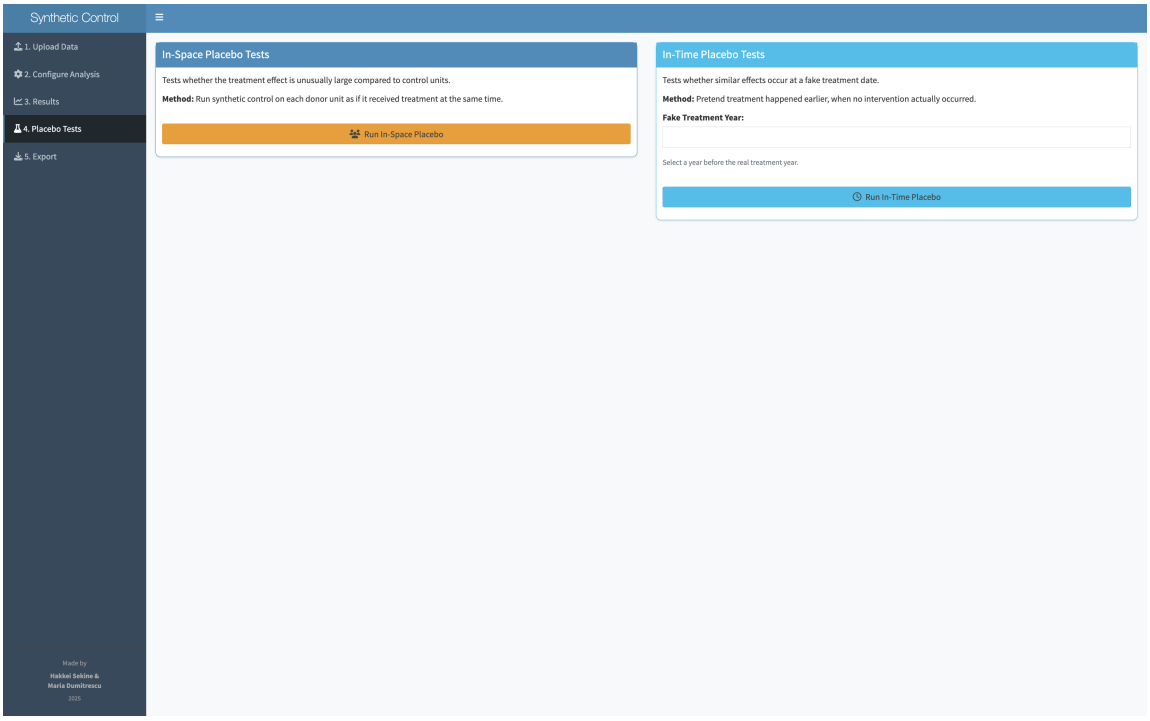

Figure 7: Run placebo tests

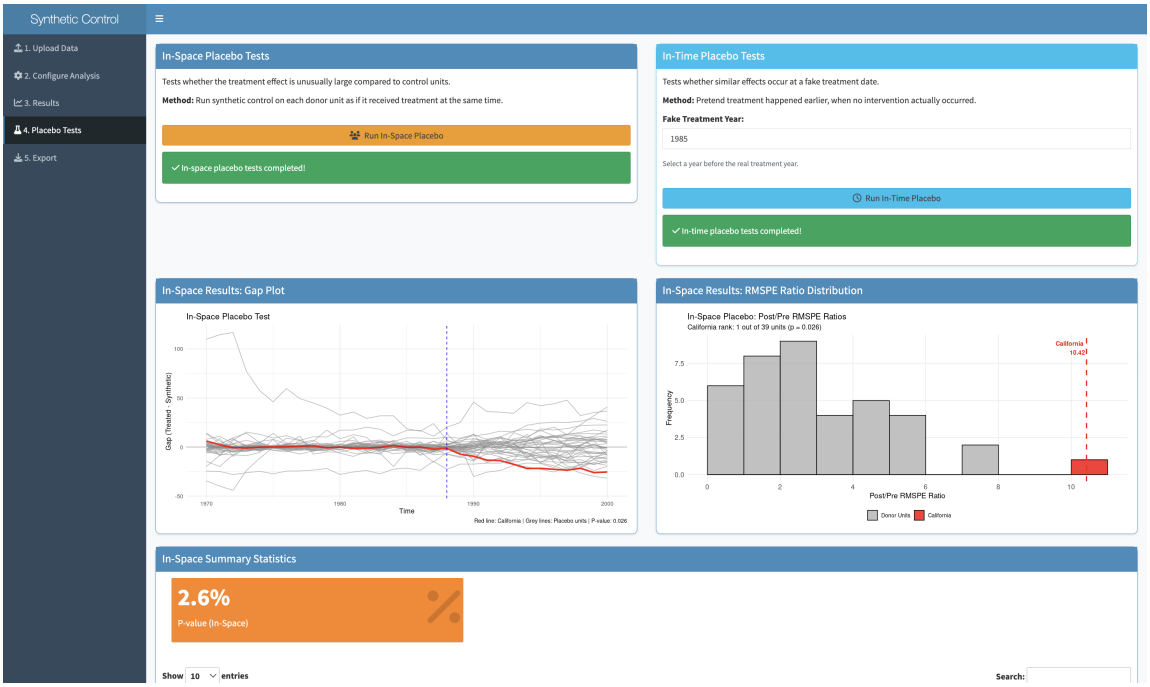

Figure 8: Run placebo tests

The app offers several placebo tests to evaluate the robustness of the treatment effect:

1. Navigate to the `Placebo Tests` tab.
2. You can run:
   - In-Space Placebo Test: Tests whether the treatment effect is unusually large compared to control units.
   - In-Time Placebo Test: Tests whether similar effects would appear with fake treatment dates before the actual treatment.
3. Choose the desired placebo test and click the corresponding button to run it.

3.7 How to Export Results

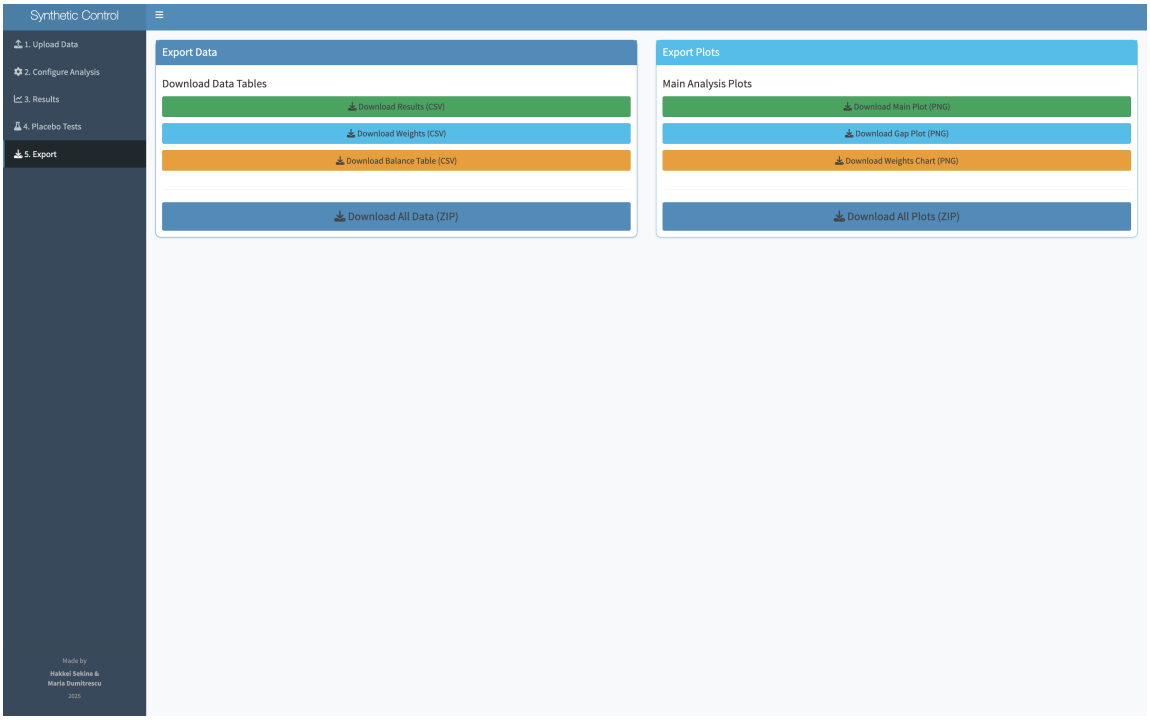

Figure 9: Export results

Once the analysis is complete, you can export the results:

1. Navigate to the `Export` tab.
2. You can export in Zip:
    - Everything as a CSV Document
    - Visualizations (PNG format) for main plots, gap plots, and weights charts.

4 Replication and Checking That the App Works

In this section We compare three things: (1) the original California smoking paper, (2) our own replication using the Synth package in R, and (3) the results produced by our Shiny app.
The goal is simple. If all three line up, then the app is working correctly.

4.1 What the Original Paper Found

The original study (Abadie, Diamond, and Hainmueller, 2010) reported a few main results:

- Before Proposition 99 in 1988, California and its synthetic version almost overlap.
- After 1988, California drops much faster than the synthetic control.
- The average effect after treatment is a large decline in smoking (about 20 to 25 fewer packs per person).
- Placebo tests show that California behaves very differently from the donor states.
- California also has one of the largest post to pre RMSPE ratios, which means the effect is unlikely to be random.

These findings are the standard benchmark.

4.2 Our Replication Using R

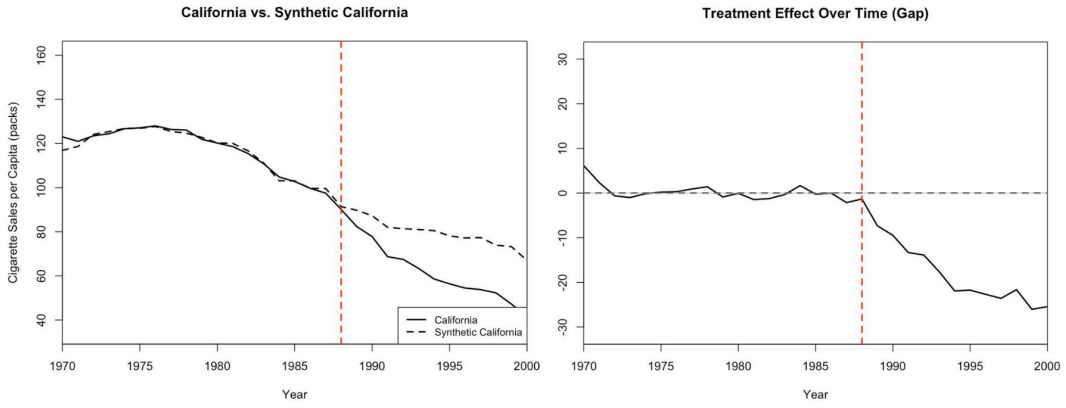

Figure 10: R Studio Calculation Treatment Effect and Synthetic vs Real Results

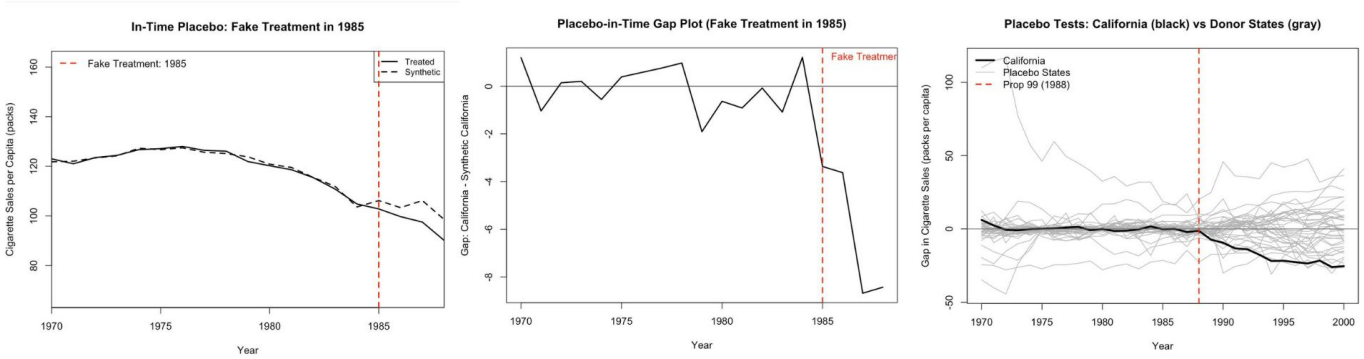

Figure 11: R Studio Calculation Placebo Plots

We used the Synth package for this study in R to replicate the original study. Our R results match what the paper found.

Main numbers from R

- Average treatment effect (1988 to 2000): -17.38 packs.
- Post to pre RMSPE ratio: 9.58.
- P value from RMSPE ratio: 0.026.

On Appendix B you can see the code we run in R studio.

The plots show the same story. California and the synthetic control match closely before 1988, then California drops much more afterward. The placebo tests also show that almost no donor state has a gap as large as California.

We use the exactly same parameter as one we define in the R studio. Overall, the R version behaves exactly as expected.

4.3 Results From the Shiny App

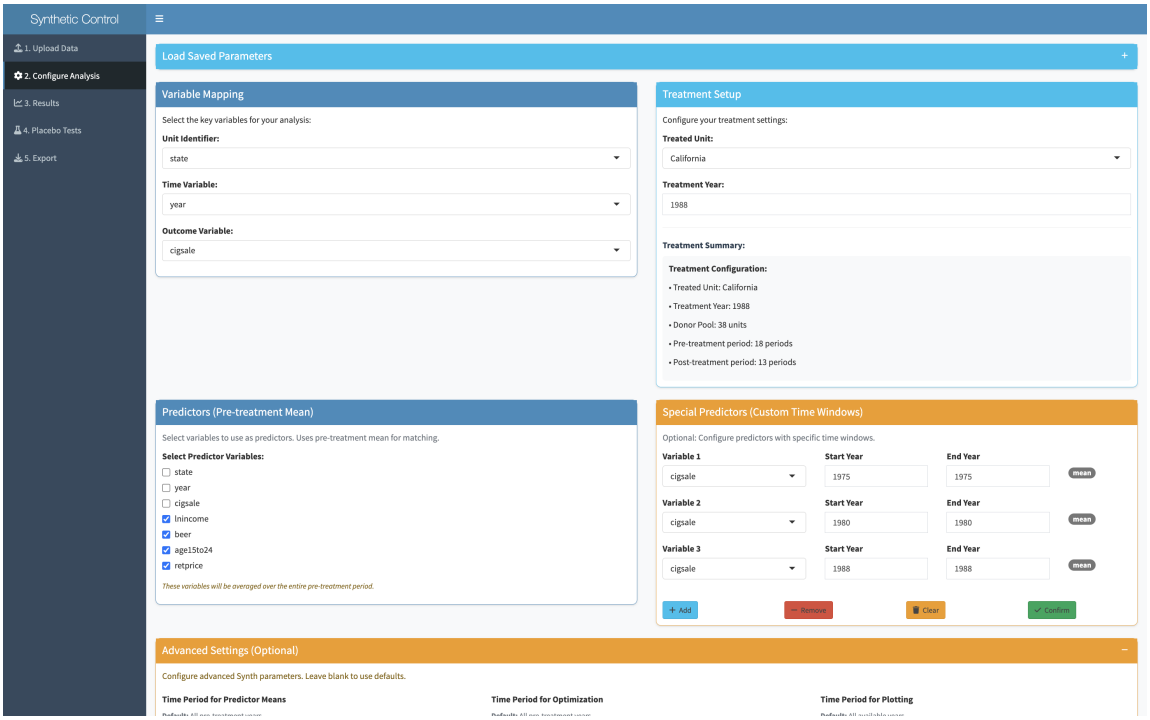

Figure 12: App Synthetic control parameters

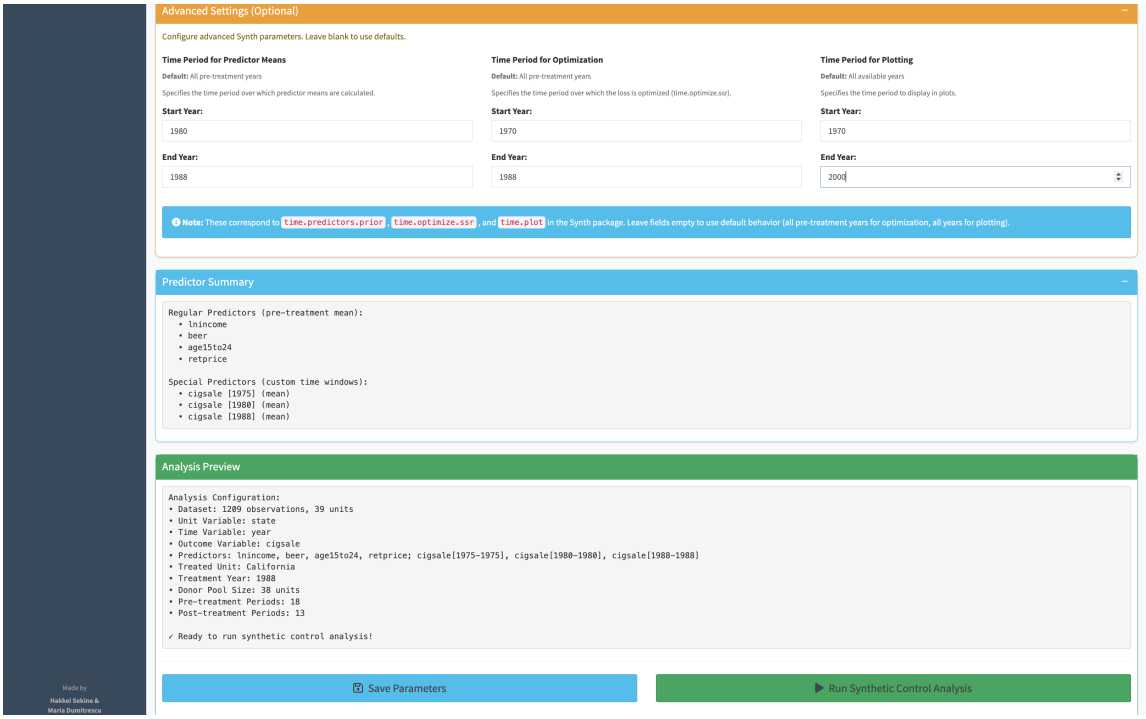

Figure 13: App Synthetic control special parameters

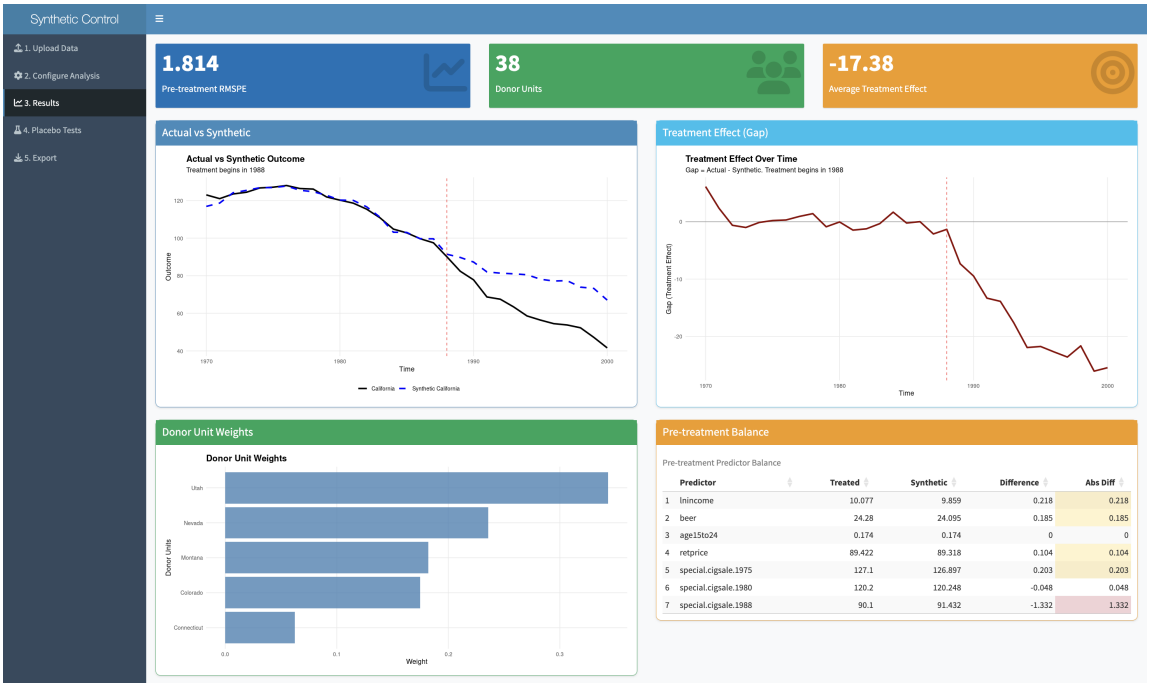

Figure 14: App Synthetic control calculation

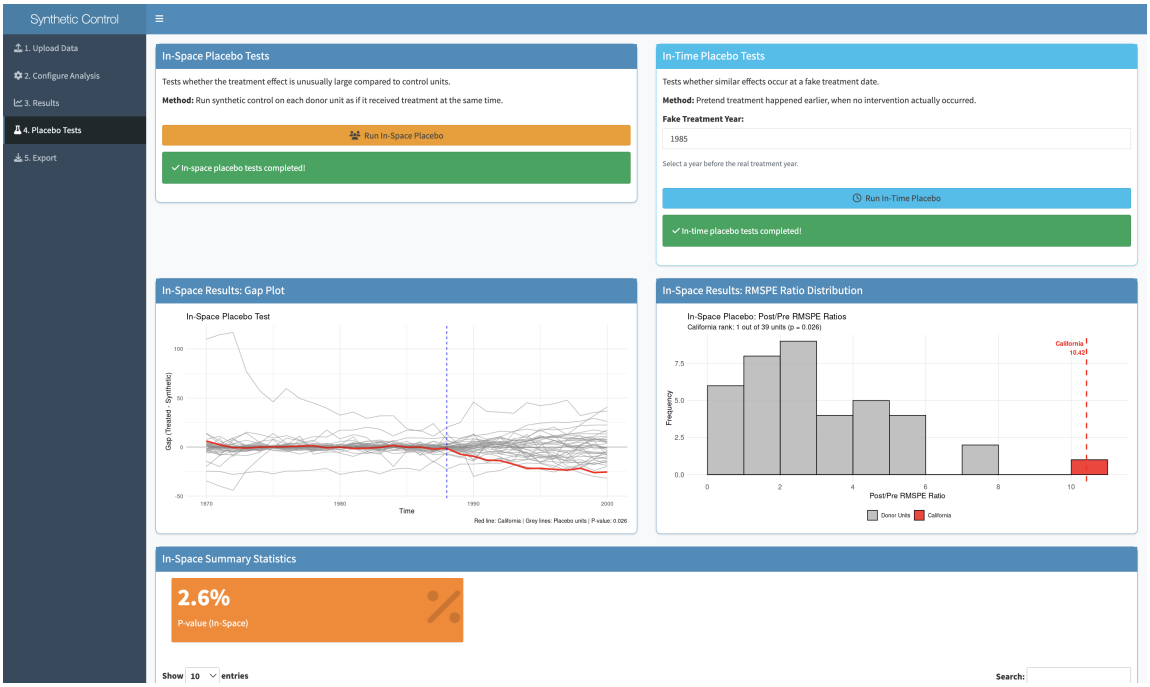

Figure 15: App Calculation Placebo In Space Plots

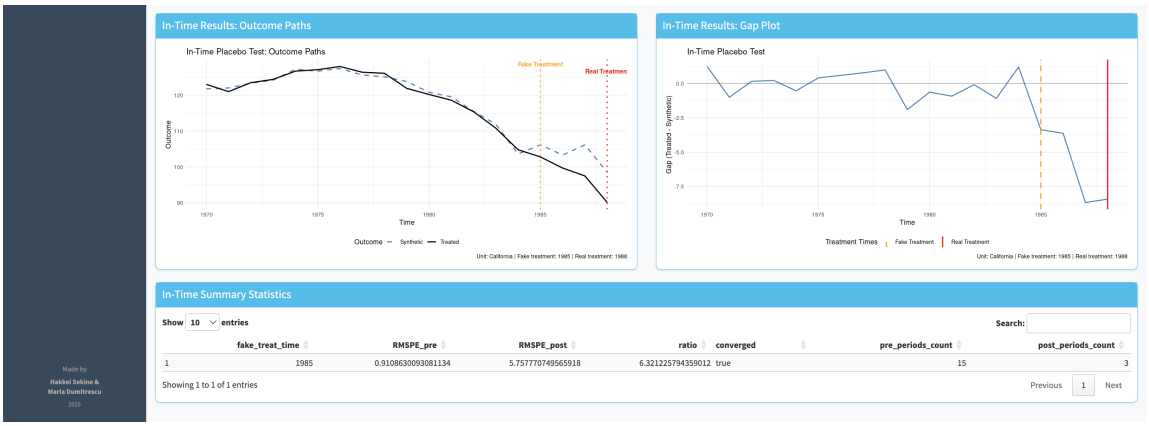

Figure 16: App Calculation Placebo In Time Plots

Our Shiny app produces almost the exact same results as the R code:

- Pre treatment RMSPE: 1.814.
- Average treatment effect: -17.38 packs.

- RMSPE ratio: 9.58.
- P value: 0.026.
- Placebo tests: California is again the clear outlier.

The app gives the same patterns in the graphs too. The pre treatment match is good, the post treatment drop is strong, and the placebo curves look almost identical to the R version.

4.4 Side by Side Comparison

Feature	Original Paper	R Replication	Shiny App
Pre treatment fit	Good	Good	Good
Drop after 1988	Large	Large	Large
Average effect	About -20 to -25	-17.38	-17.38
RMSPE ratio	Very large	9.58	9.58
P value	Small	0.026	0.026
Placebo pattern	CA stands out	CA stands out	CA stands out
In time placebo	No false effect	Same	Same

Everything lines up: the direction, the size of the effect, the p value, the placebo results, and even the shape of the graphs.

4.5 Final Conclusion

We should note that the original California smoking paper is not identical to the R studio or ShinyApp version. This is because the original study may have used slightly different special predictors, time ranges, or even a slightly different cleaned dataset that we do not have the exact access to. Because of this, the exact numbers in published figures can vary across different replications. However, we confirmed that our app is accurate by taking the exact same predictors in R-Studio and running them in our ShinyApp. The app produced the same synthetic control path, the same treatment effect, and the same placebo patterns. This confirms that the synthetic control engine inside our app works correctly. Even if different datasets or predictor choices change the exact numerical values, our tests show that the app applies the method properly and replicates results when the inputs match.

5 AI Statement

We used AI throughout the development of this project as a support tool for planning, structuring, and implementing the Shiny app. At the beginning, we used AI to help us create a detailed product requirements document and a full task list with subtasks. We provided all the goals, features, and constraints of the app, and the model turned them into a complete PRD written in Markdown. During development, we used Codex inside Visual Studio Code as our main coding assistant. We pasted the PRD into our workspace and used prompt engineering to ask Codex to build the app step by step, generate helper functions, fix errors, and rewrite code blocks. When Codex did not understand something or kept misinterpreting a task, we also used Claude Code, which helped

clarify or rewrite specific pieces of logic more reliably. We manually tested and refined every feature of the app to make sure it worked correctly. We also used AI to explain some of the technical code it produced by asking for summaries of functions and overviews of how different components interacted. Finally, we used AI only for small formatting tasks in LaTeX, not for writing the actual text. Overall, AI acted as a planning and coding assistant, but all design decisions, debugging, testing, and writing were done by us.

6 Collaboration Note

Maria Dumitrescu and Hakkei Sekine collaborated on this assignment.

Appendix A: Resources

Github Repository Link:
`https://github.com/riadumitrescu/cs130-synthetic-control-shiny-app`

Dataset Used for Testing:
`https://docs.google.com/spreadsheets/d/e/2PACX-1vQBlO75ljKmDYid0kT0_uIhovnecaka2zvKgT9ZO_JW3TvyoRKgL6DJzUFFBxL-qisHbPWBhnK4UpLW/pub?output=csv`

Online Shiny App:
`https://hakkei.shinyapps.io/synthetic-control-analysis/`

7 Appendix B: R Studio Code

R studio code to test Shiny App:
`https://colab.research.google.com/drive/1rVu3YU8Uf1vjEmR9Eiu-9nHzNDcKo61u?usp=sharing`