Sliding into Causal Inference, with Python!

euroSciPy Basel 2022

Alon Nir

Sep 1st 2022

https://github.com/alonnir/EuroSciPy-2022-Talk

Introduction	Who am I? What's our goal for today? What is sliding?
The Gold Standard of Causal Inference	The magic of randomisation.
No Gold for You	CI solutions for when randomisation is not possible.
Recap & Where Next	What we've seen, where to go and what to do.

Our principles for today:

Vast
Oversimplification

(exuberant)
Hand Waving

Introduction

About Me

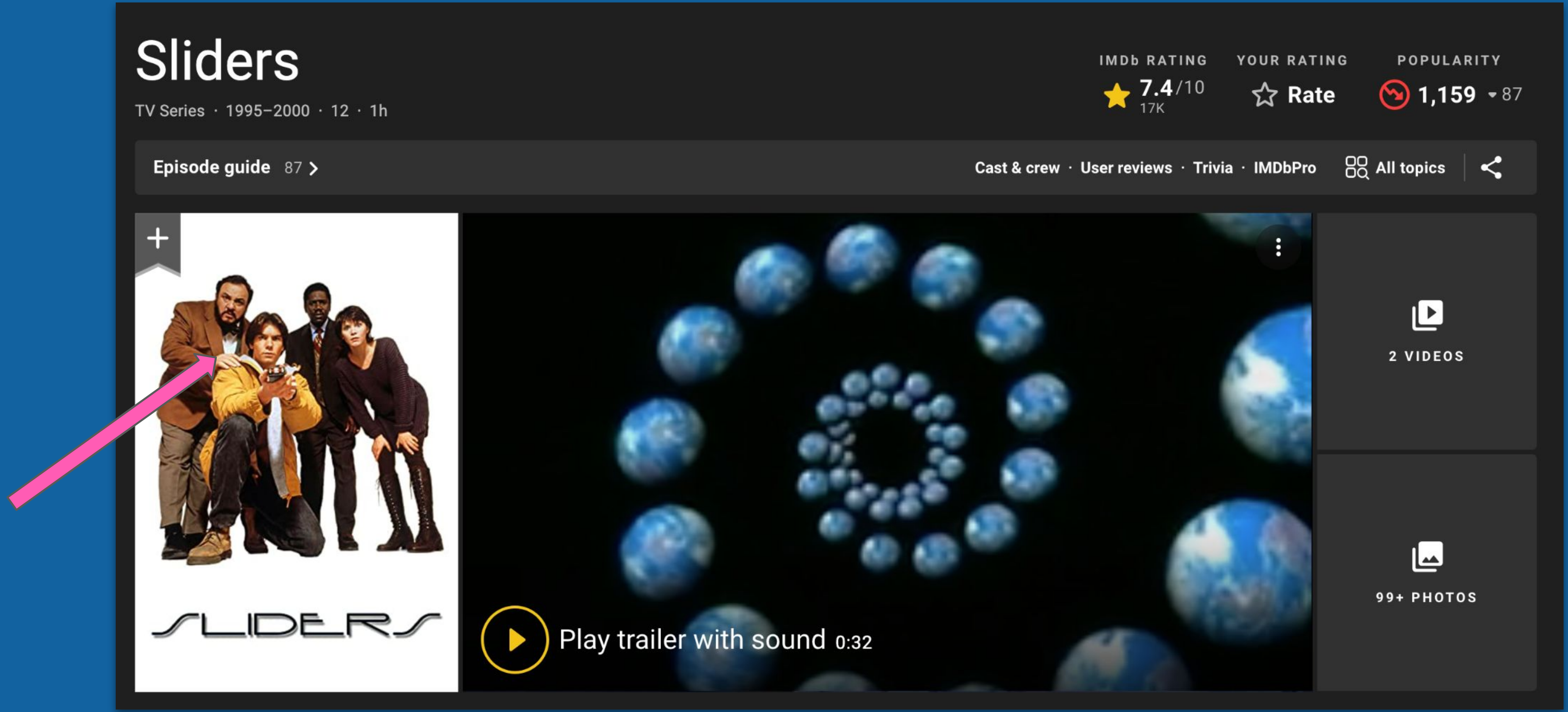
- Data science lead at Spotify. *
- Based in the UK
- @alonnir on Twitter, Linkedin, Github
- I often read “causal inference” as “*casual* inference” 

* But I’m representing myself and will not discuss any projects at Spotify.

About Me

- Also, I’m a big nerd.

About sliding



Source: https://www.imdb.com/title/tt0112167/

Intro - what if...? 🤔

What if you found a portal to a parallel universe?

What if you could Slide into a thousand different worlds?

Where it's the same year, and you're the same person, but everything else is different...

- Sliders (1995-2000) intro

Intro - our own sliding device?

In Academia and industry we already use simulations for many different things.

Can we simulate parallel worlds? Can we build a sliding device with Python?

Before we answer that, let’s start from the basics.

Meet Emma

emmaszone

* nope, not really

Emmaszone

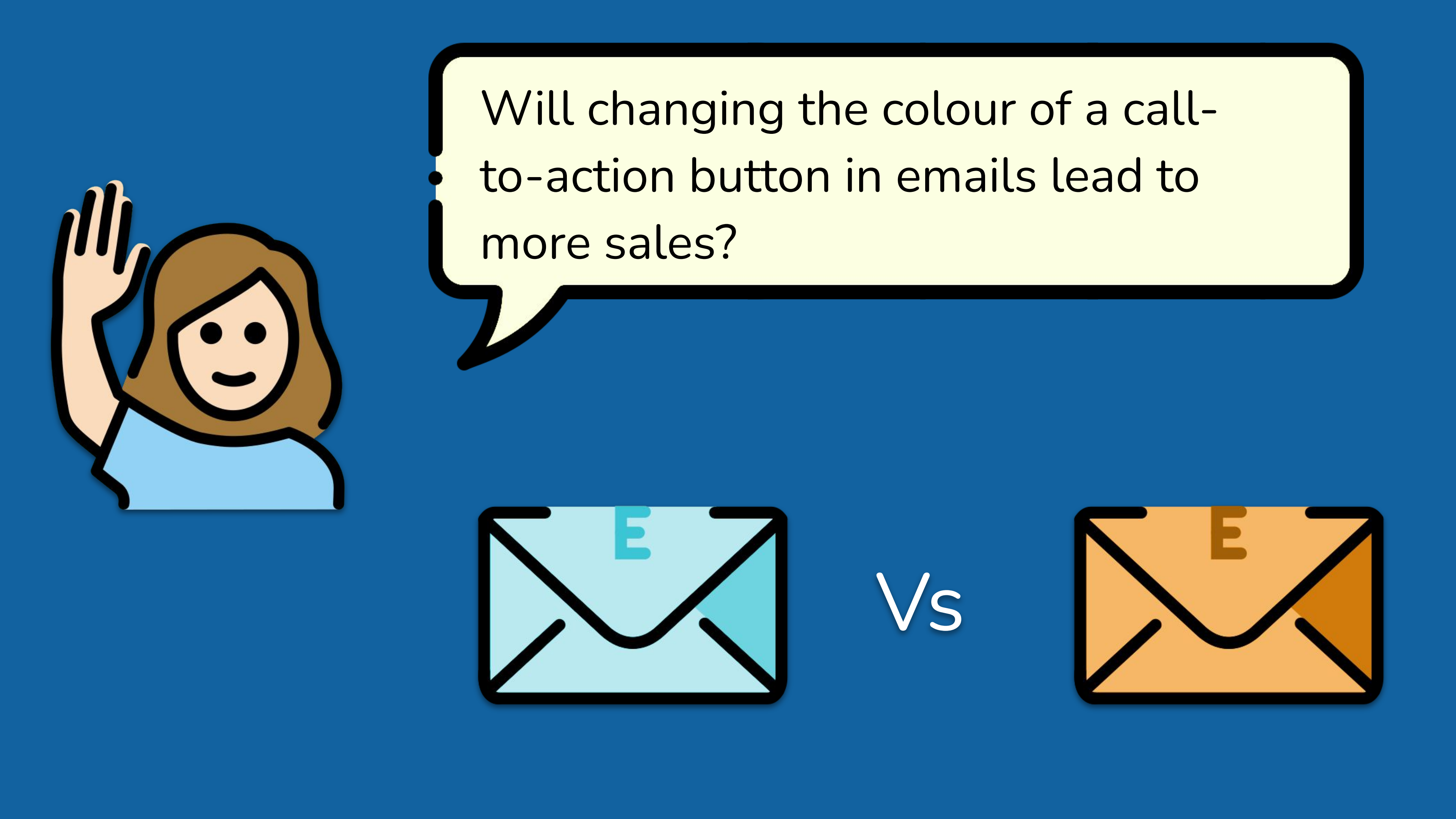
We’ll explore different initiatives to improve the business performance of Emmaszone:

1. Will changing the colour of a button lead to more sales?
2. Will a membership programme increase sales?
3. Will free shipping on every order increase sales?

Let’s get started...

The Gold Standard of Causal Inference

Will changing the colour of a call-to-action button in emails lead to more sales?


Will changing the colour of a call-to-action button in emails lead to more sales?
Vs

In a perfect world...

In a perfect world...

```
For user in registered_users:
    Show email with blue button
    log if user clicks on the button and purchases wool


    Show email with orange button
    log if user clicks on the button and purchases wool
```

In a perfect world…

For user in registered_users:

Show email with blue button

log if user clicks on the button and purchases wool

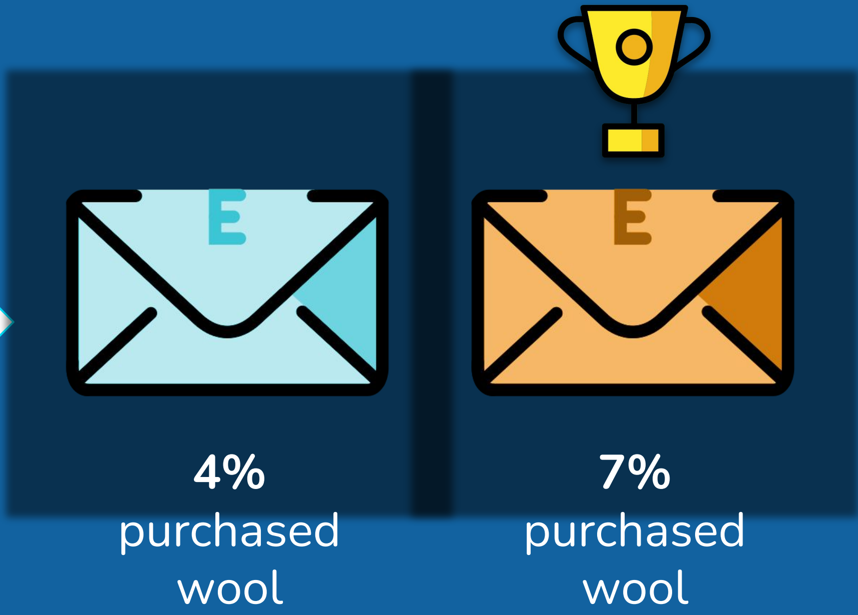
Show email with orange button

log if user clicks on the button and purchases wool

User	Convert on Blue	Convert on Orange
user_1	1	0
user_2	1	1
user_3	0	1
…		
user_n	1	0

User	Convert on Blue	Convert on Orange
user_1	1	0
user_2	1	1
user_3	0	1
...		
user_n	1	0

Calculate proportions per colour



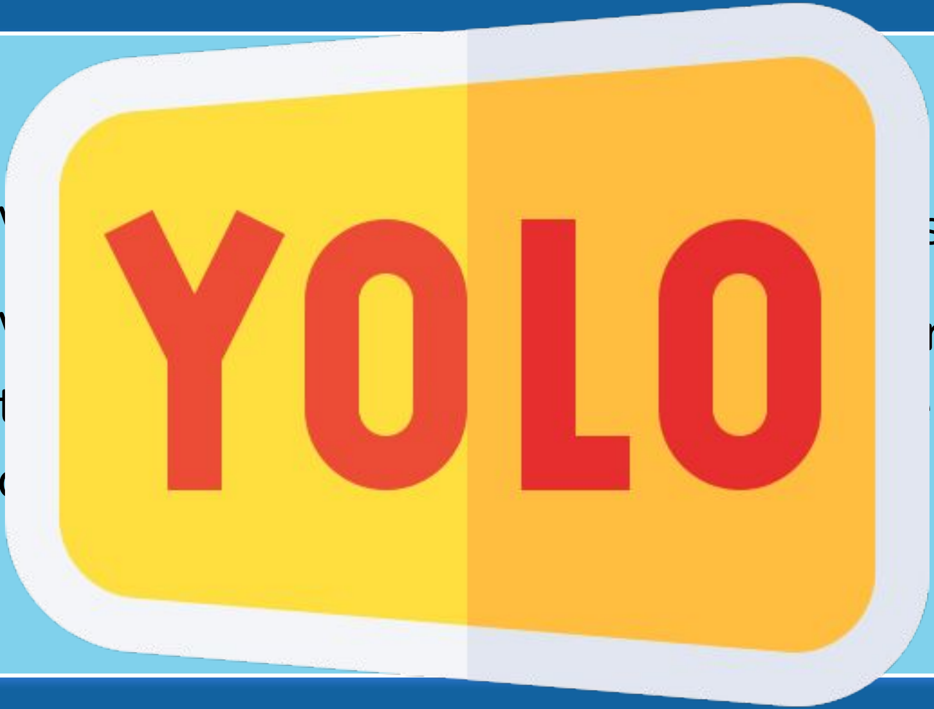
*I conveniently skipped the part with statistical test for proportions.

The Fundamental Problem of Causal Inference

We observe only **one outcome** per person.

We don't know what "**would have been**" the outcome had the person been under different circumstances.

The Fundamental Problem of Causal Inference



Given YOLO, what *can* we do?

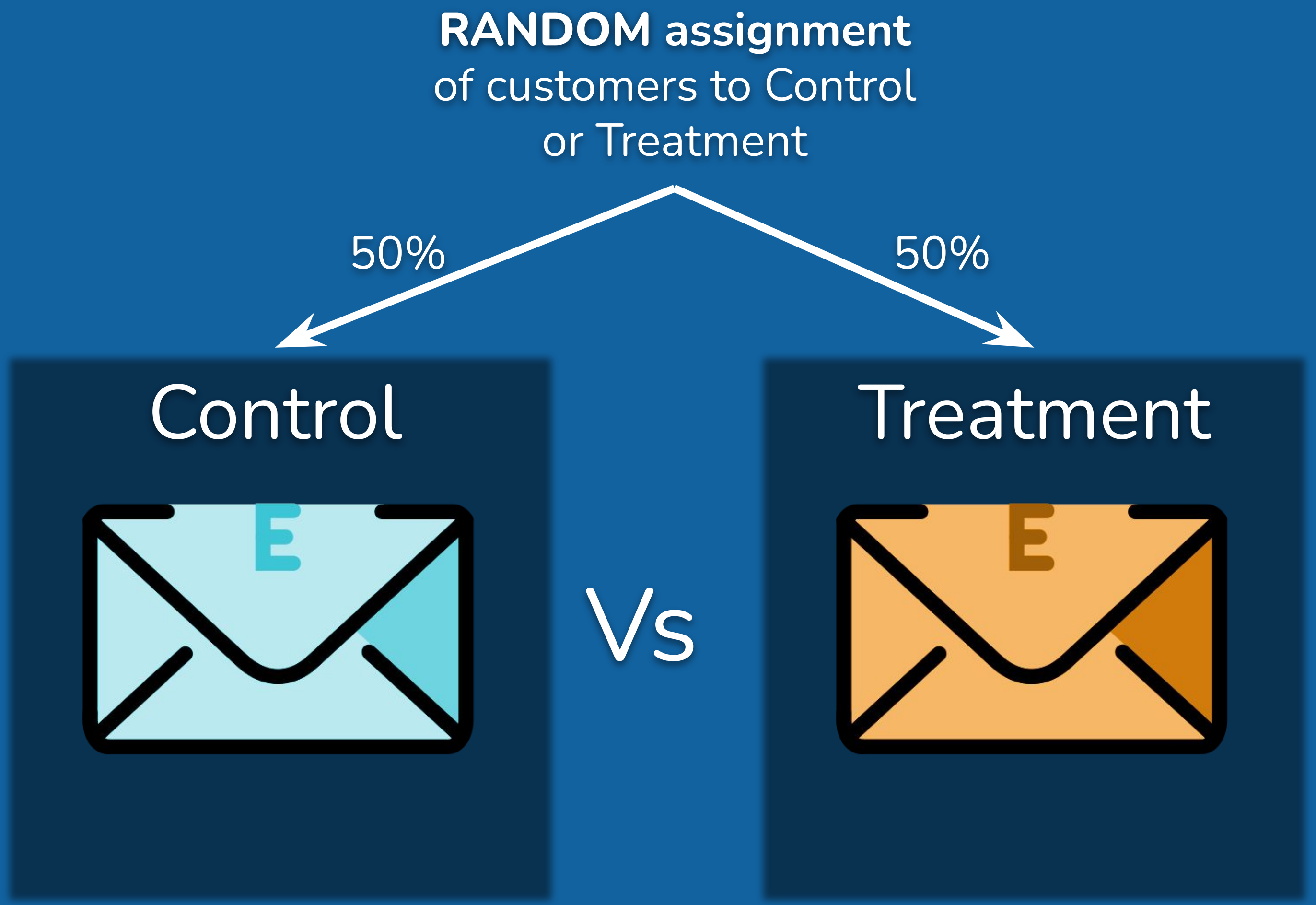
Randomised Controlled Trials*

Control

Vs

Treatment+

* AKA A/B test
\+ AKA Variant


RANDOM assignment
of customers to Control
or Treatment
50%
50%
Control
Vs
Treatment

4% of recipients purchased wool

7% of recipients purchased wool

Control

Vs

Treatment

For an intuitive explanation, [Jake VDP's "Statistics for Hackers"] is superb.

The key words are **random assignment**

The key words are **random assignment**

Unobserved covariates

e.g. colour blindness

Did you know?
Males are 16x more likely to be colour blind than females*.

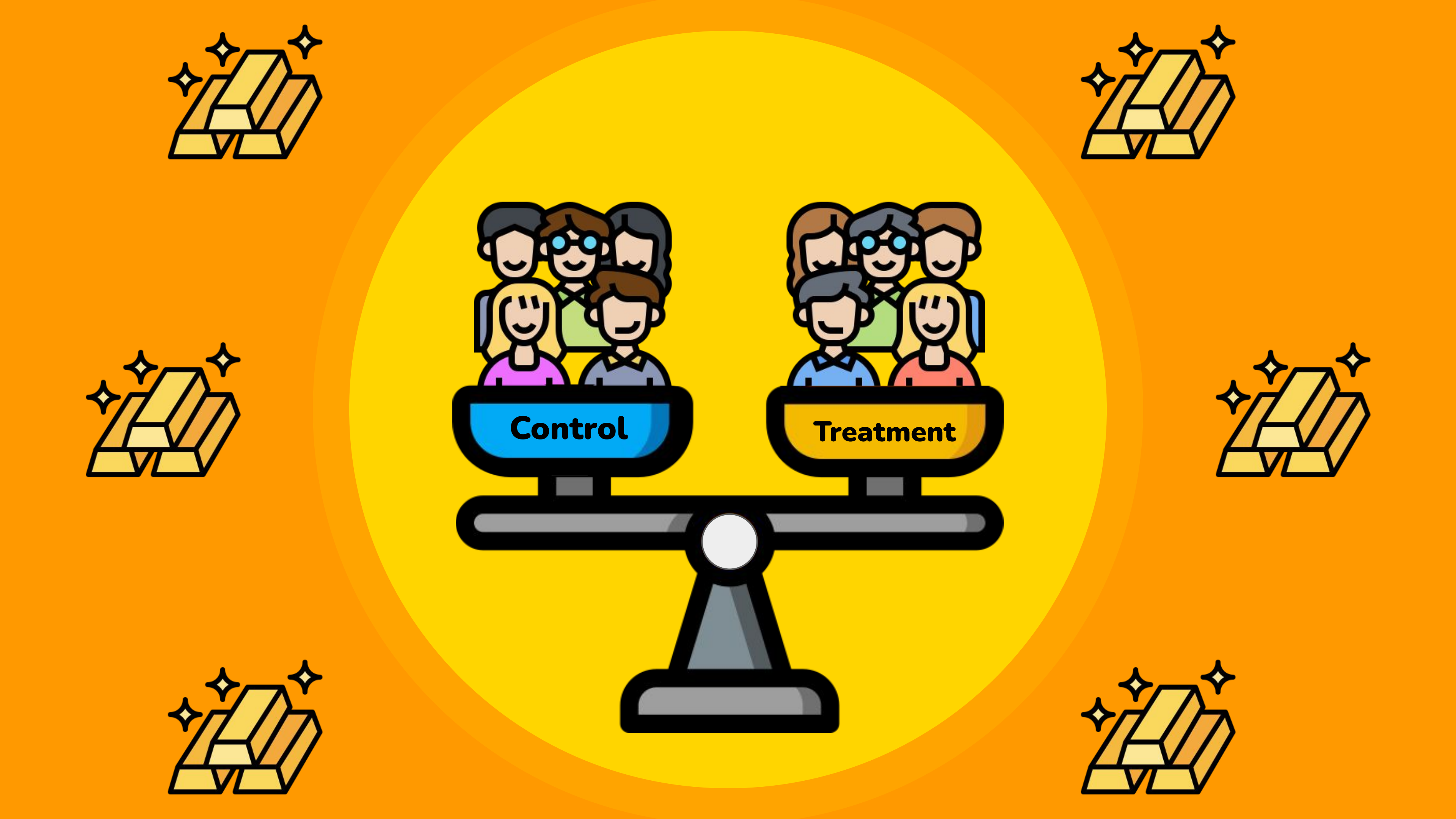
Observed covariates

E.g. smartphone OS, location

Randomisation balances both!

⇒ Randomised controlled trials are the **gold standard in causal inference.**

* https://www.nhs.uk/conditions/colour-vision-deficiency/ | https://en.wikipedia.org/wiki/Color_blindness



Control
Treatment

No Gold for You

Motivation

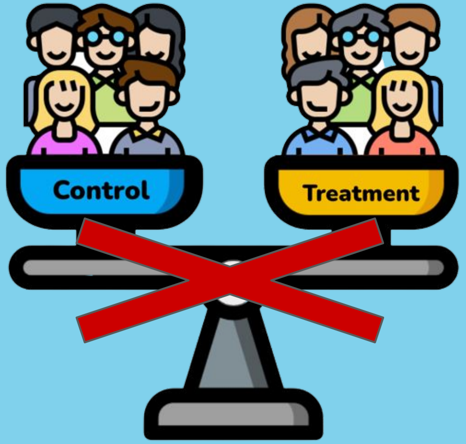
At times a randomised controlled trial cannot be used, because:

- Network effects (SUTVA violation, spillovers)
- Immoral and/or illegal (e.g. drugs, smoking)
- Too expensive or otherwise infeasible, can't randomise on the desired unit
- All we have is observational data, captured some time in the past
- Bad user experience

Two Big Use Cases we'll discuss today

Next

Assignment is not balanced



Later

Assignment is impossible

Two Big Use Cases we'll discuss today

Next

Assignment is not balanced

Later

Assignment is impossible

Will introducing a membership programme - *Emmaszone Sublime* - increase sales?

Emmaszone Part II

For the sake of the exercise, let’s assume:

- Any order from Emmaszone contains exactly *one* ball of wool.
- Delivery is £2 flat *per order*.
- Things are at *a steady state*, i.e. nothing changes, no seasonality, churn, etc.

Emmaszone Part II

For the sake of the exercise, let’s assume:

- Any order from Emmaszone contains exactly one ball of wool.
- Delivery is £2 flat *per order*.
- Things are at *a steady state*, i.e. nothing changes, no seasonality, churn, etc.

Now, Emma introduces a new service: **Emmaszon Sublime**, which offers unlimited **free delivery** for an **£11/month** subscription fee.

How will that affect the business?

Remember what we want is...

Remember what we want is...

Let's compare the "treated" (Sublime subscribers) to the "untreated":

Sublime Subscribers	Non-Subscribers
13 orders placed, on average, per Sublime subscriber	**4.7** orders placed, on average, per non-subscriber

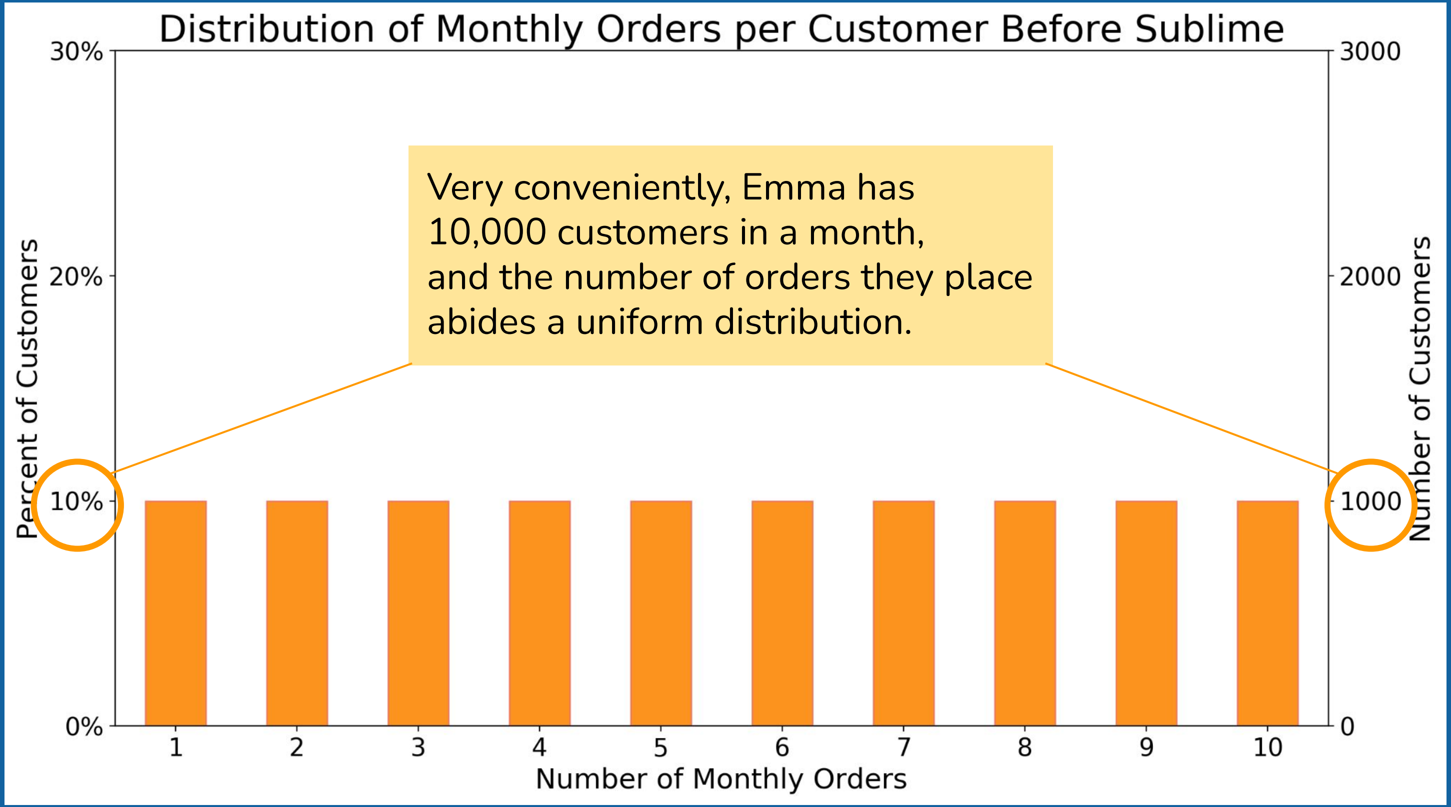
→ Sublime members place **+8.3** orders a month on average, a **179%** uplift!

→ Conclusion: Smashing success! Let's spend a lot of money trying to acquire as many Sublime members as possible!

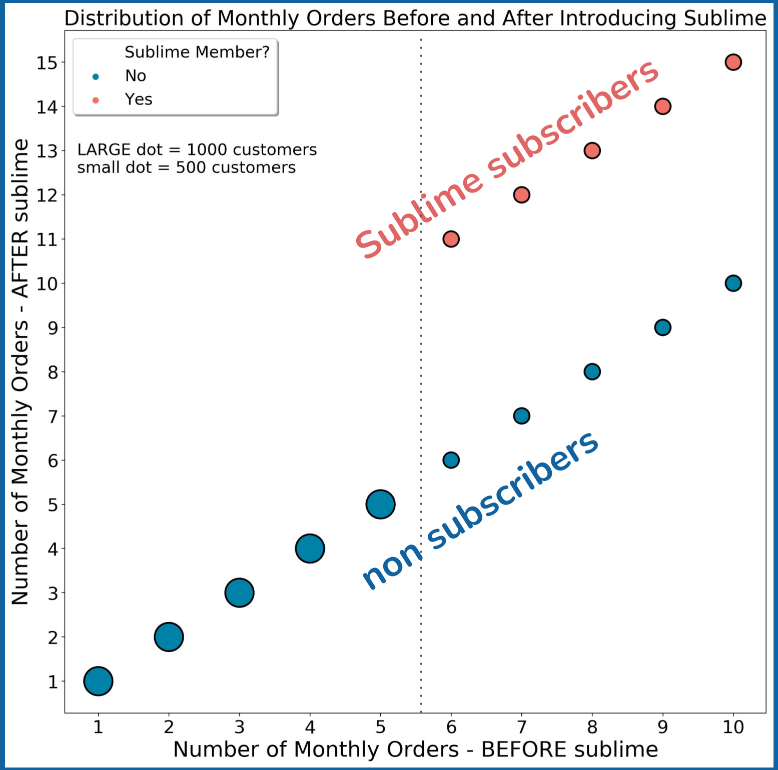
* we'll see why "treated" and "untreated" are in quotes in the next few slides

Or maybe...

Distribution of orders before Sublime was introduced



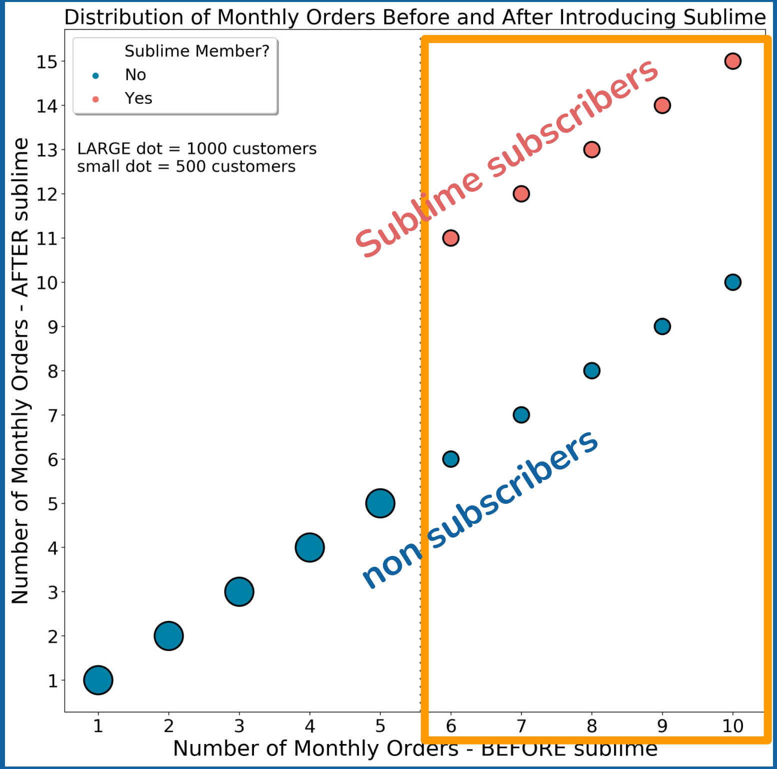
Let's take a closer look..



Red dot: Sublime subscribers
Blue dot: Non-subscribers

BIG dot: 1,000 people
small dot: 500 people

Let's take a closer look..

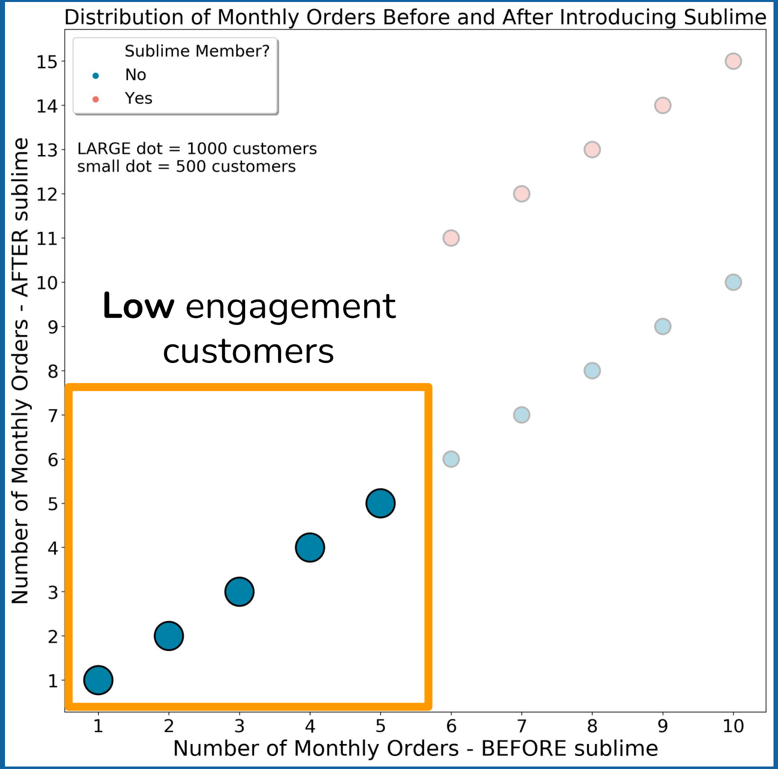


We see that:

- **Only** customers who ordered **6** or more times a month are the only ones to get a Sublime subscription, which makes sense.

  (£11/mo subscription fee < £2*orders)

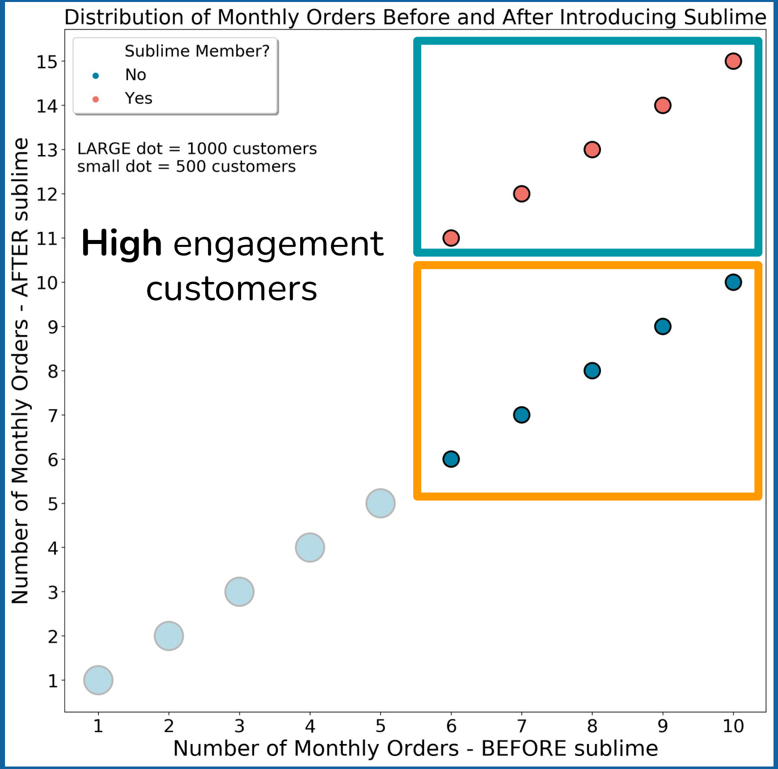
Let’s take a closer look..



We see that:

- Customers that placed 1-5 orders *before* Sublime was introduced, <u>did not</u> subscribe to Sublime and <u>Did not</u> change their behaviour.

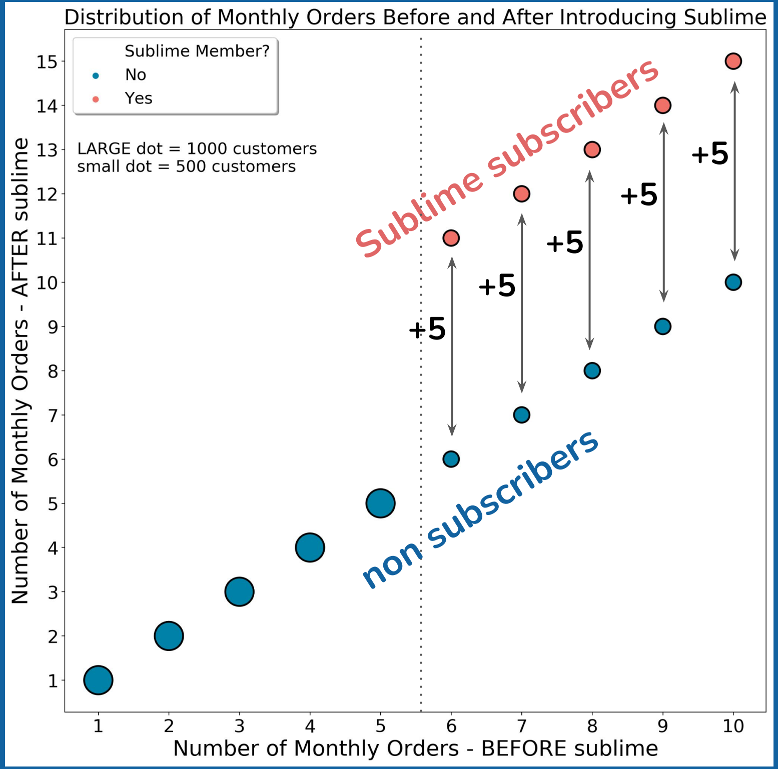
Let’s take a closer look..



We see that:

- <u>50%</u> (n=500) of customers that ordered 6-10 times a month <u>did not</u> subscribe to Sublime and <u>did not</u> change their behaviour.
- The other 50% <u>subscribed</u> to Sublime and <u>increased</u> the number of monthly orders.

Let’s take a closer look..

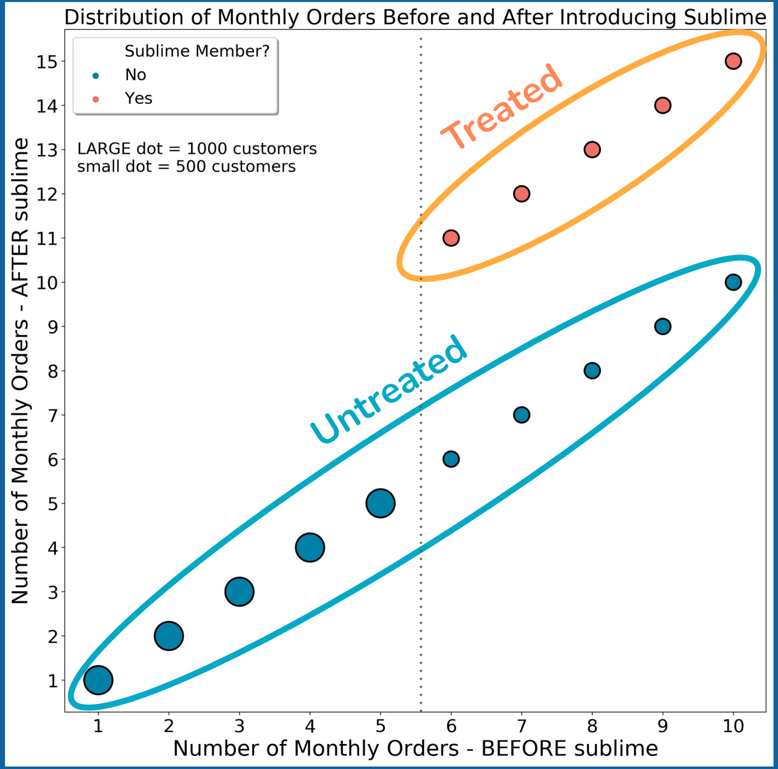


We see that:

- Each Sublime subscriber increased their monthly orders by **exactly 5** additional (or incremental) orders a month.

How did we get to the **+8.3** incremental orders we saw earlier?

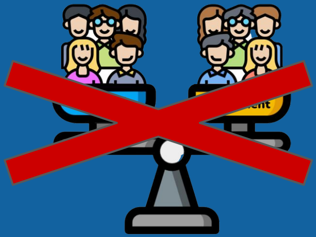
Let’s take a closer look..



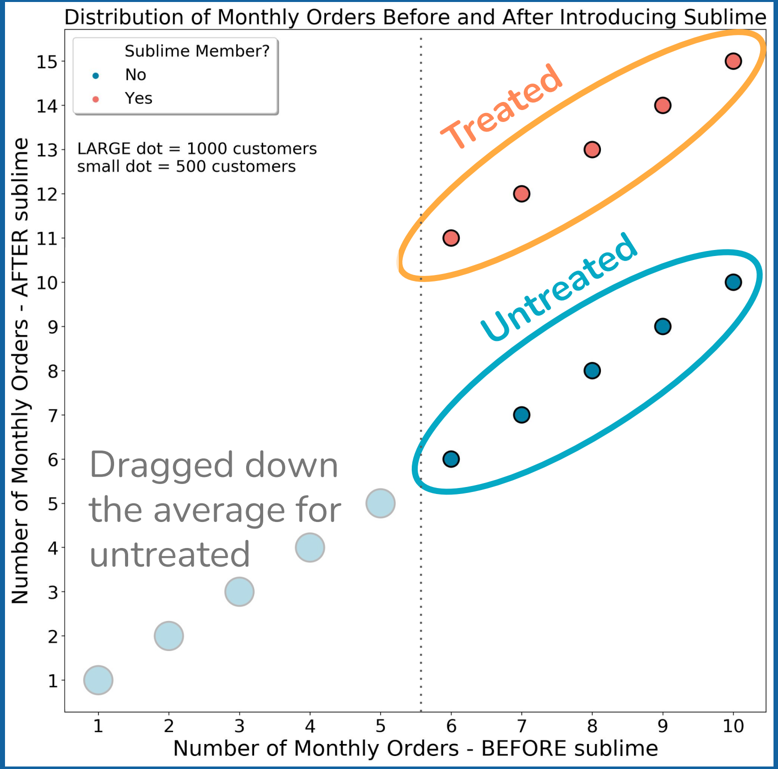
We made an **unfair** comparison.

We included low-engagement customers (<6 orders a month) who were **unaffected** by the introduction of Sublime.

High engagement customers (>=6 orders a month) **self-selected** themselves to become Sublime subscribers.



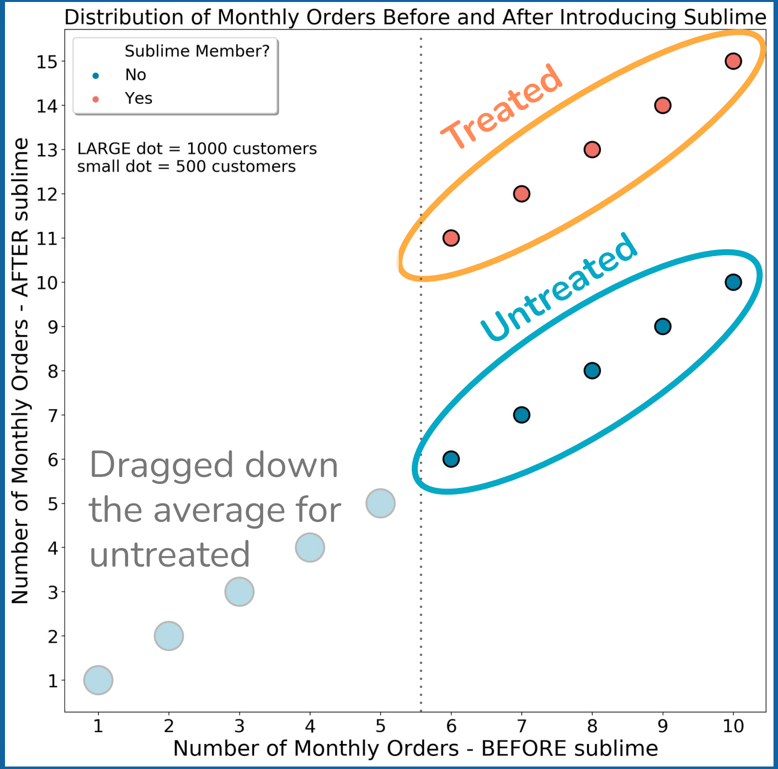
Let's take a closer look..



In order to get the **true** effect of Sublime, *controlling for the selection bias*, we need to compare apples to apples (or oranges to oranges, your pick).

In other words, we need to "fix" the assignment to treatment/control to resemble *random* assignment.

Let's take a closer look..



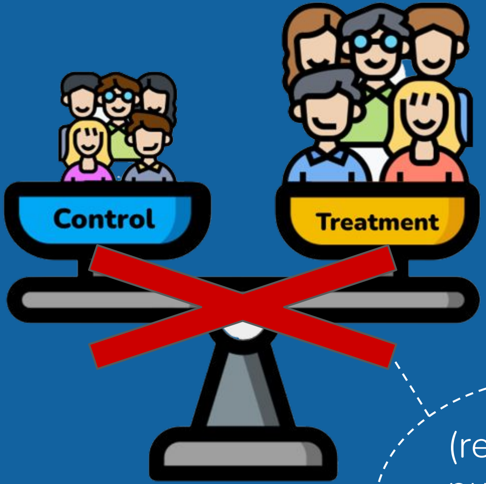
In order to get the **true** effect of Sublime, *controlling for the selection bias*, we need to compare apples to apples (or oranges to oranges, your pick).

In other words, we need to "fix" the assignment to treatment/control to resemble *random* assignment.

The way we do that is with **Propensity Score Matching**.

What we
WANT

What we
HAVE



(really pushing my artistic skills to the limit)

Propensity Score Matching

Method:

1. Learn a model to predict the propensity a user will get the treatment

2. Match every *treated* user to an *untreated* user with a similar propensity

3. For each pair find the difference in number of orders (or any other metric)

4. The average of these differences would be a pretty good approximation of the effect

Step 0: Simulate some data

```
# Generate some dummy data
import pandas as pd
import numpy as np

# Set the params
n_low_engagement = 5000          # 1-5 orders a month
n_high_engagement = 5000        # 6-10 orders a month
p = 0.5                          # probability a high-engagement customer would convert to Sublime
abs_uplift = 5                  # Number of additional orders each Sublime subscriber makes

# Generate user_ids
user_ids = [10000+x for x in range(n_low_engagement+n_high_engagement)]
```

Step 0: Simulate some data (continued)

```
# Establish baseline orders (pre-intervention)
low_engagement = np.random.randint(low=1, high=6, size=n_low_engagement)
high_engagement = np.random.randint(low=6, high=11, size=n_high_engagement)
engagement_baseline = np.concatenate((low_engagement, high_engagement), axis=0)

# Conversions to Sublime subscription
low_engagement_conversions = np.zeros(n_low_engagement)    # no low-engagement customer converts
high_engagement_conversions = np.random.choice(a=[False, True],
                                               size=n_high_engagement, p=[p, 1-p])  # 50% convert
conversions = np.concatenate((low_engagement_conversions, high_engagement_conversions), axis=0)

# Put it all together
df = pd.DataFrame({'orders_baseline':engagement_baseline, 'converted':conversions}, index=user_ids)
```

Step 0: Simulate some data (continued)

```
df.groupby('orders_baseline')['converted'].mean()
```

Orders Baseline	% Converted
1	0.0%
2	0.0%
3	0.0%
4	0.0%
5	0.0%
6	51.7%
7	51.2%
8	50.5%
9	50.9%
10	51.3%

Step 0: Simulate some data (continued)

```
# Finally, add the impact of the treatment
df['uplift'] = df['converted']*abs_uplift
df['orders_post_treatment'] = df['orders_baseline']+df['uplift']
```

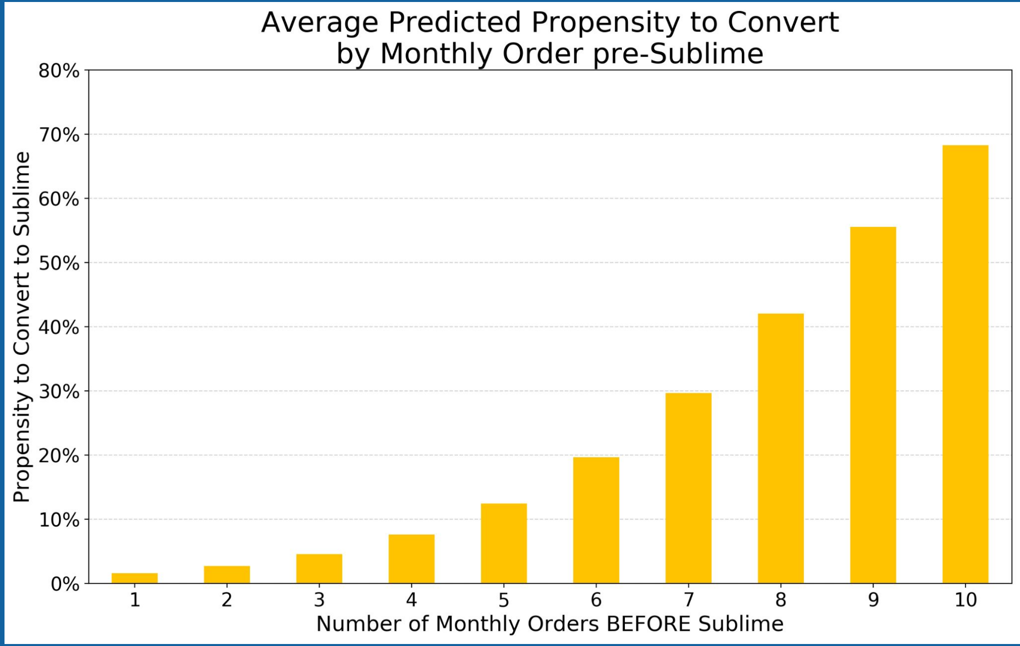
user_id	Orders Baseline	Converted	Uplift	Orders Post Treatment
10000	4	0	0	4
10001	5	0	0	5
10002	2	0	0	2
10003	5	0	0	5
:	:	:	:	:

Step 1. Predict the propensity to get the treatment

```
# Calculate the propensity scores
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

X = engagement_baseline.reshape(-1,1)
y = conversions

logreg = LogisticRegression()

logreg.fit(X,y)  #  we're not overly concerned with train/test splits (descriptive Vs predictive model)

y_pred = logreg.predict(X)

print(accuracy_score(y, y_pred))
>>>0.7486    # pretty much what we would expect
```

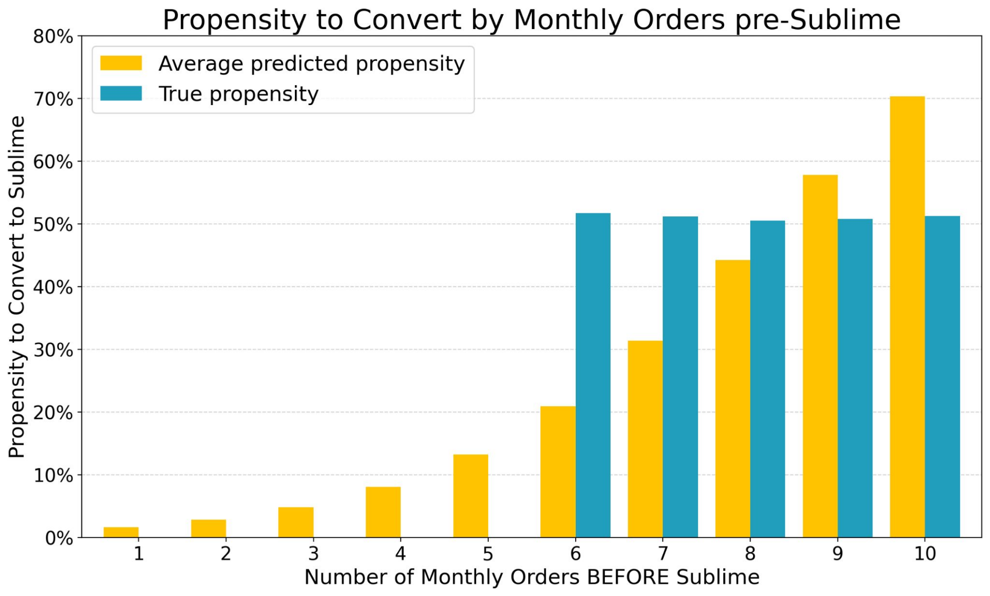
Step 1. Predict the propensity to get the treatment

```
# Get the propensity to get the treatment
y_pred_proba = logreg.predict_proba(X)

# assign to a column in the dataframe
df['propensity_from_proba'] = y_pred_proba[:,1]
```



Step 1. Predict the propensity to get the treatment

```python
# Get the propensity to get the treatment
y_pred_proba = logreg.predict_proba(X)

# assign to a column in the dataframe
df['propensity_from_proba'] = y_pred_proba[:,1]
```



Step 2. Match users with similar propensity scores

```python
treated = df[df['converted']==1].copy(deep=True)          # Sublime subscribers
untreated = df[df['converted']==0].copy(deep=True)        # non-subscribers

def matcher(score, untreated_data=untreated):
    """
    Returns the user_id of the user in the untreated data with the closest score.
    Don't do this at home - there are better ways.
    """
    untreated['delta'] = abs(score - untreated['propensity_from_proba'])
    return untreated.loc[:, 'delta'].idxmin()


treated['match_user_id'] = treated['propensity_from_proba'].apply(matcher)
```

Step 3. Find the difference between matched customers

```
# Put it together

# merge (join) the tables together
merged = treated.merge(untreated.loc[:,['orders_post_treatment', 'propensity_from_proba']],
                left_on='match_user_id', right_index=True, suffixes=("_treated", "_untreated"))


# calculate the effect size
merged['uplift_abs'] = merged['orders_post_treatment_treated']-merged['orders_post_treatment_untreated']
```

Step 4. Get the ATT

```
print(merged['uplift_abs'].mean())
```

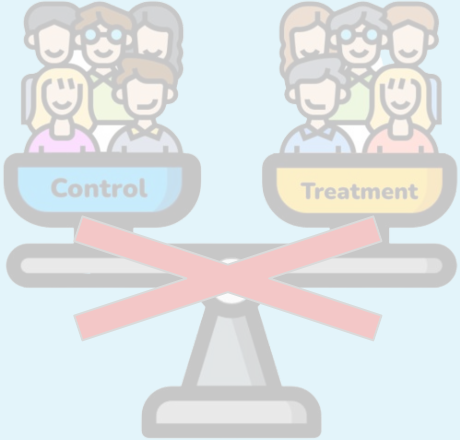
→ 5.0

Propensity Score Matching - Summary

- Since the split to Treated/Untreated **wasn’t random**, the effect we saw was *skewed and inflated*.
- PSM proved useful in fixing the split to emulate random assignment.
- In our simple example, where there was only one covariate, PSM is superfluous. We could have just matched on # of orders before Sublime.
- In a real world setting you would have **many** covariates, and the propensity score is a method to get **one** number to balance on, instead of many dimensions (think about *the curse of dimensionality*).

Just saw

Assignment is not balanced



Next

Assignment is impossible

What will giving users free shipping on **every order** (no membership needed) do to wool sales?

Emmaszone Part III

5 different types of wool

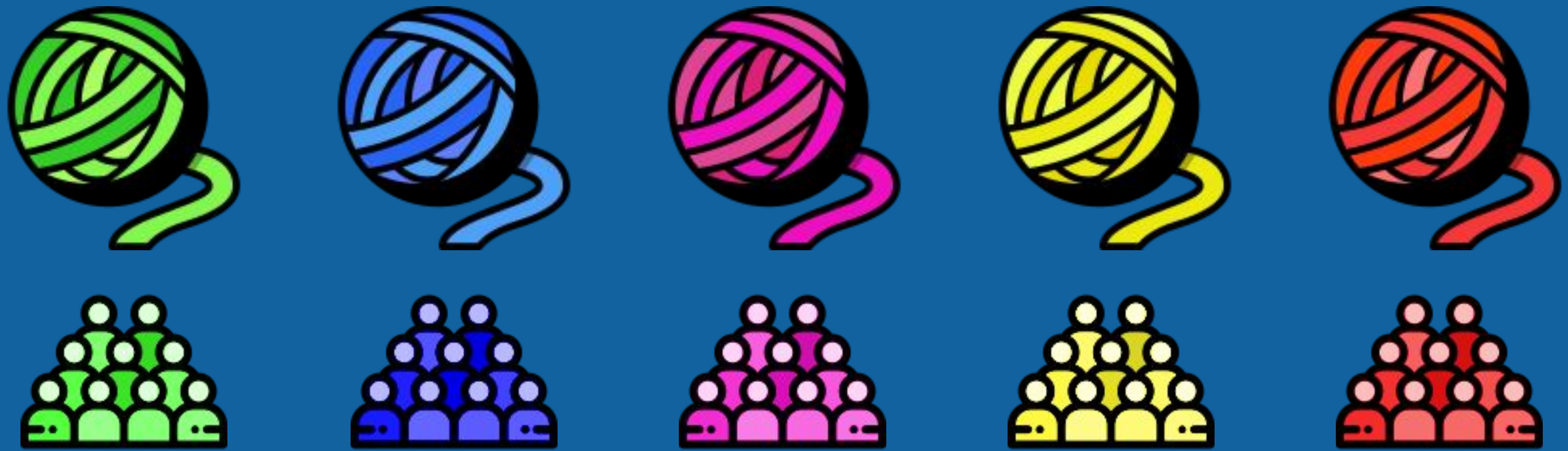
icons designed by [Freepik]() from [Flaticon]()

Emmaszone Part III

5 different groups of customers

icons designed by Freepik from Flaticon

Emmaszone Part III



avg = 5 | avg = 3 | avg = 1 | avg = 7 | avg = 8

5 different purchasing behaviours

Emmaszone Part III

avg =

5

Assumptions:

- Imagine the Sublime subscription programme never happened.
- Instead, Emma decided to offer free delivery **for everyone** buying green wool!
- Customers of other wool colours don’t change their behaviour.

Emmaszone Part III

Assumptions:

- Imagine the Sublime subscription programme never happened.
- Instead, Emma decided to offer free delivery **for everyone** buying green wool!
- Customers of other wool colours don't change their behaviour.

avg =
5

?

How would free delivery affect the number of orders green wool customers make in a month?

Emmaszone Part III

Remember, what we want is this:

Emmaszone Part III

Remember, what we want is this:

But **everyone** is exposed to the treatment. There is **no** control group.

So what can we do?

Depends on what other data you have you can do several things:

So what can we do?

Depends on what other data you have you can do several things:

- Pre/post analysis
- Diff-in-diff
- Synthetic controls (or: build a sliding device!)

So what can we do?

Depends on what other data you have you can do several things:

- Pre/post analysis
- Diff-in-diff

See Appendix

- Synthetic controls (or: build a sliding device!)

It’s a slide on a slide about sliding! 🤯

Synthetic controls

What if we could build our own sliding device?

Can we build a parallel world where the treated weren’t treated? 🌍

Synthetic controls

Synthetic controls method generates a **counterfactual** time series by combining (weighting) control time series that are **unaffected** by the treatment.

Synthetic controls

Green ≈ 0.5*Blue + 0*Magenta + 0.5*Yellow + 0*Red

Synthetic controls

Green ≈ 0.5*Blue + 0*Magenta + 0.5*Yellow + 0*Red

5 3 1 7 8

5 = 0.5*3 +0.5*7

Let's see it in code

```
# Generate some dummy data
# ---------------------------------------
periods_before = 18
periods_after = 6

# Monthly customers (remains constant - net churn/new customers = 0)
num_green = num_blue = num_yellow = 500

ef = 2  # <--- absolute effect size

green_before = np.random.randint(low=4, high=7, size=(num_green, periods_before))
green_after = np.random.randint(low=4+ef, high=7+ef, size=(num_green, periods_after))

blue = np.random.randint(low=5, high=10, size=(num_blue, periods_before+periods_after))
yellow = np.random.randint(low=1, high=6, size=(num_yellow, periods_before+periods_after))

# put it all together
g = np.concatenate((np.mean(green_before, axis=0), np.mean(green_after, axis=0)))
b = np.mean(blue, axis=0)
y = np.mean(yellow, axis=0)

avg_orders = pd.DataFrame({'green':g, 'blue':b, 'yellow':y}).round(2)
```

Let’s see it in code

```python
# Generate some dummy data
# -------------------------------------
periods_before = 18
periods_after = 6

# Monthly customers (remains constant - net churn/new customers = 0)
num_green = num_blue = num_yellow = 500

ef = 2  # <--- absolute effect size

green_before = np.random.randint(low=4, high=7, size=(num_green, periods_before))
green_after = np.random.randint(low=4+ef, high=7+ef, size=(num_green, periods_after))

blue = np.random.randint(low=5, high=10, size=(num_blue, periods_before+periods_after))
yellow = np.random.randint(low=1, high=6, size=(num_yellow, periods_before+periods_after))

# put it all together
g = np.concatenate((np.mean(green_before, axis=0), np.mean(green_after, axis=0)))
b = np.mean(blue, axis=0)
y = np.mean(yellow, axis=0)

avg_orders = pd.DataFrame({'green':g, 'blue':b, 'yellow':y}).round(2)
```

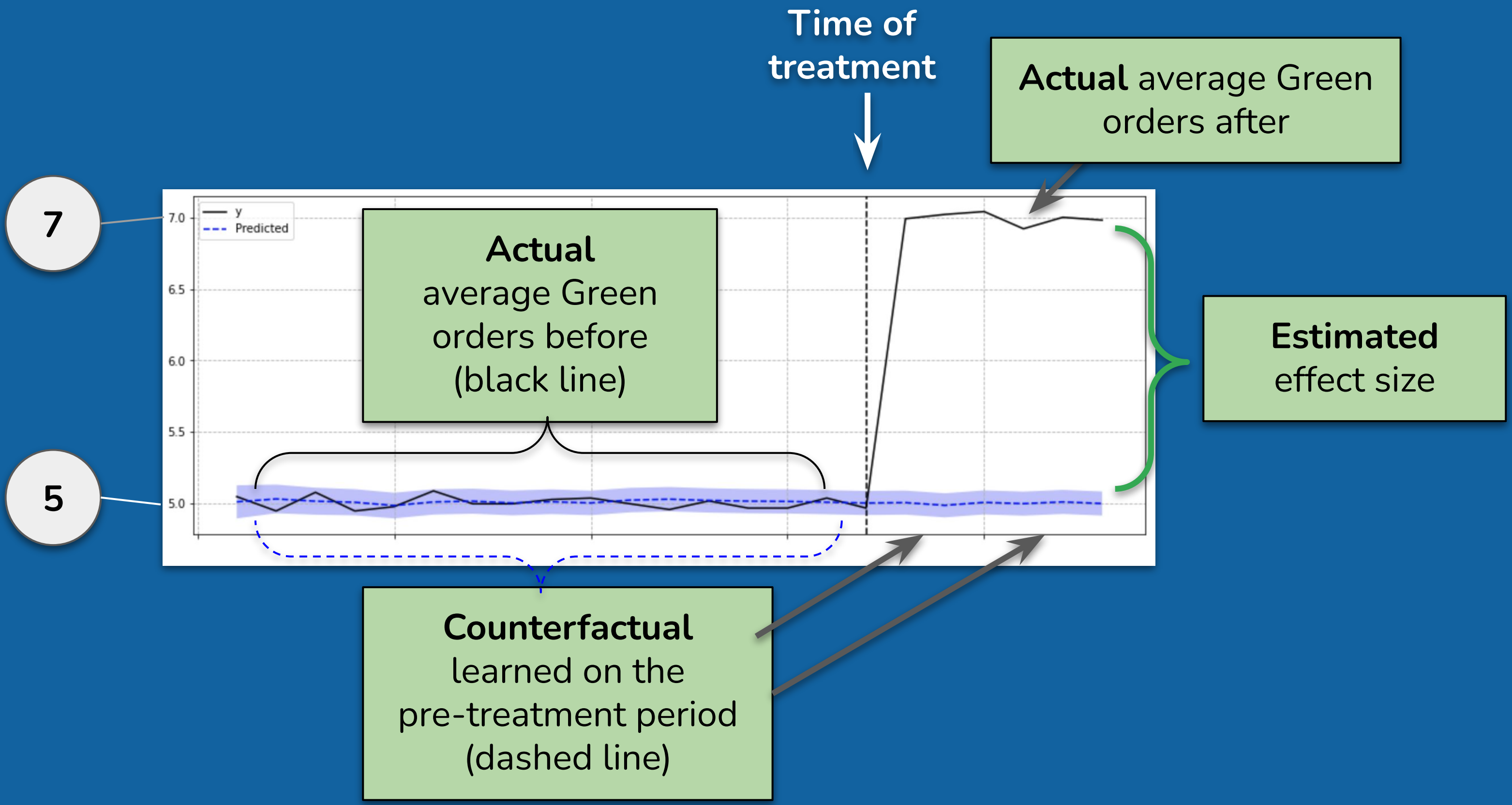
Pre-Treatment Descriptive Statistics

	green	blue	yellow
count	18.00	18.00	18.00
mean	5.01	6.98	3.03
std	0.03	0.04	0.04
min	4.95	6.88	2.97
25%	4.99	6.96	2.99
50%	5.00	6.98	3.02
75%	5.02	7.01	3.06
max	5.09	7.05	3.11

Let's see it in code

```
# pip install pycausalimpact
from causalimpact import CausalImpact

pre_period = [0, 17]
post_period = [18, 23]

# note the column with the treatment needs to be the leftmost
column in the DF
ci = CausalImpact(avg_orders, pre_period, post_period)

# bish bash bosh
ci.plot(panels=['original', 'pointwise'])
```

CauslImpact uses *Bayesian structural time series*, which is well beyond the scope of this talk.

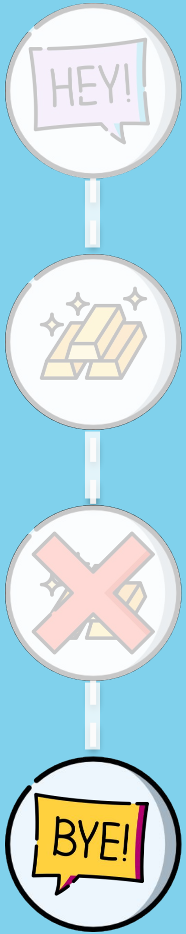


Time of treatment
Actual average Green orders after
7
5
7.0
6.5
6.0
5.5
5.0
y
Predicted
Actual average Green orders before (black line)
Estimated effect size
Counterfactual learned on the pre-treatment period (dashed line)

Synthetic controls - Discussion

- This was a super simple example to convey the idea you can use a *mix* of controls to generate a counterfactual, even if none of the other groups is a good control group by itself.

- In fact, the model specification was *wrong* and we actually did not need *any* of the controls here. (hint: `ci.trained_model.summary()`)



Recap & Where Next

Quick Recap

The fundamental problem of causal inference.

Control/Variant Assignment	Method	Conclusion
Control and *variant* are balanced due to random assignment	Randomised Controlled Trials (A/B testing)	🤩
Split to *control* and *variant* is not random	Propensity score matching	🤓
No *control* group, everyone is exposed to the treatment	Before/After (Appendix)	🤨
	Diff-in-diff (Appendix)	😏
	Synthetic controls	😎

Next up

Breadth and depth

Breadth

What we didn't talk about...

- Pre/Post (see Appendix)
- Diff-in-diff (see Appendix)
- DAGs (!)
- Instrumental variables
- Fixed effects regression
- Regression discontinuity design
- Meta-learners
- Cluster randomised trials
- Simpson's paradox
- Switchbacks
- Sensitivity analysis
- AND MORE

Depth - extensions and things we didn’t cover

Take PSMs as an example:

- How do we check the groups are balanced post matching?
- How should we match? (greedy matching? NN matching? Genetic matching? Optimal matching?...)
- Should we match with or without replacement? What would we do if 90% get the treatment and 10% don’t?
- What if the closest match isn’t that close? Where do we draw the line for a match to be good enough?
- What if the treatment is not one-and-done, but a multi stage process?
- And more and more...

Final thoughts

We discussed the fundamental problem of causal inference.

We have fancy math and shiny tools to address it, and as we've seen we can do a pretty good job with these tools.

We must not forget we will always live only on one Earth, hence we will never *truly* know the size of the effect an intervention had.

According to Chaos theory the outcomes of a system can vary massively even with slight differences for the starting conditions.

Would you like to know more?

- Short term: slides and code will be available here:
  https://github.com/alonnir/EuroSciPy-2022-Talk

- Long term: curated curriculum for causal inference at:
  https://github.com/alonnir/just-cause
  (WIP)

Thank you

and happy sliding :)

https://twitter.com/alonnir | https://www.linkedin.com/in/alonnir

https://github.com/alonnir/EuroSciPy-2022-Talk

Appendix

Emoji and Icon Credits

Icons from flaticon.com :

Ball of wool icons created by Freepik
https://www.flaticon.com/free-icons/ball-of-wool

People icons created by Eucalyp
https://www.flaticon.com/free-icons/people

Business and finance icons created by Smashicons
https://www.flaticon.com/free-icons/business-and-finance
(yes, I'm confused about the name too)

Circle icons created by Freepik
https://www.flaticon.com/free-icons/circle

Gold Ingots icons created by surang
https://www.flaticon.com/free-icons/gold-ingots

You only live once icons created by Freepik
https://www.flaticon.com/free-icons/you-only-live-once

Cross icons created by Freepik
https://www.flaticon.com/free-icons/cross

People icons created by Freepik
https://www.flaticon.com/free-icons/people

Playground icons created by mangsaabguru
https://www.flaticon.com/free-icons/playground

Thumbs down icons created by Freepik
https://www.flaticon.com/free-icons/thumbs-down

Speech bubble icons created by Freepik
https://www.flaticon.com/free-icons/speech-bubble

Hey icons created by Freepik
https://www.flaticon.com/free-icons/hey

Bye icons created by Freepik
https://www.flaticon.com/free-icons/bye

https://openmoji.org/library/emoji-1F468/
https://openmoji.org/library/emoji-1F3D6/
https://openmoji.org/library/emoji-1F9F6/
https://openmoji.org/library/emoji-1F44B/
https://openmoji.org/library/emoji-1F431-200D-1F4BB/
https://openmoji.org/library/emoji-1F9D1-200D-1F9B3/
https://openmoji.org/library/emoji-1F64B-200D-2640-FE0F/

Emojis designed by OpenMoji – the open-source emoji and icon project.
License: CC BY-SA 4.0

https://openmoji.org/library/emoji-1F913/
https://openmoji.org/library/emoji-1F929/
https://openmoji.org/library/emoji-1F60E/
https://openmoji.org/library/emoji-1F60F/
https://openmoji.org/library/emoji-1F928/
https://openmoji.org/library/emoji-1F4E7/
https://openmoji.org/library/emoji-1F379/
https://openmoji.org/library/emoji-1F53A
https://openmoji.org/library/emoji-1F3C6/
https://openmoji.org/library/emoji-1F973/
https://openmoji.org/library/emoji-1F30D

Assumptions for PSM

- There are *observable* covariates such that after controlling for these covariates the assignment to treatment/control is as good as random.

- There is sufficient overlap in the covariates of the treated and untreated so units can be matched (common support).

1. Before/After analysis (Pre/Post)

If we have historical data *on our customers*, we can compare similar time periods *before* and *after* the treatment. E.g.:

Average number of orders per customer in the month **after** the treatment was introduced	-	Average number of orders per customer in the month **before** the treatment was introduced	=	Effect size?

1. Before/After analysis (Pre/Post)

Pros:

- Quick and easy

Cons:

- Assumes the treatment is the only thing affecting the target variable.
- Sensitive to trends, seasonality, special events (superbowl, earthquakes, deadly virus spreading)

Bottom line: 🤨

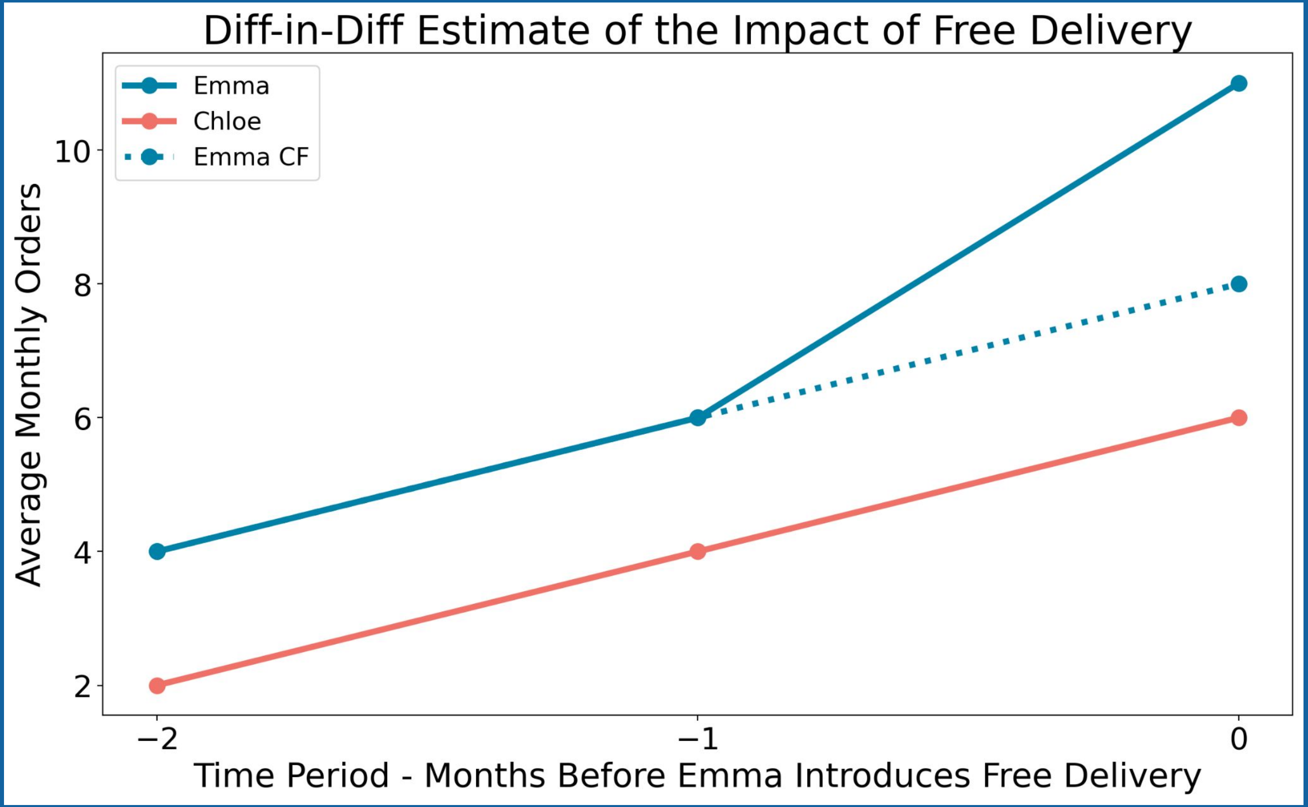
2. Diff-in-diff

Emma’s sister, Chloe, sells *yarn* online.

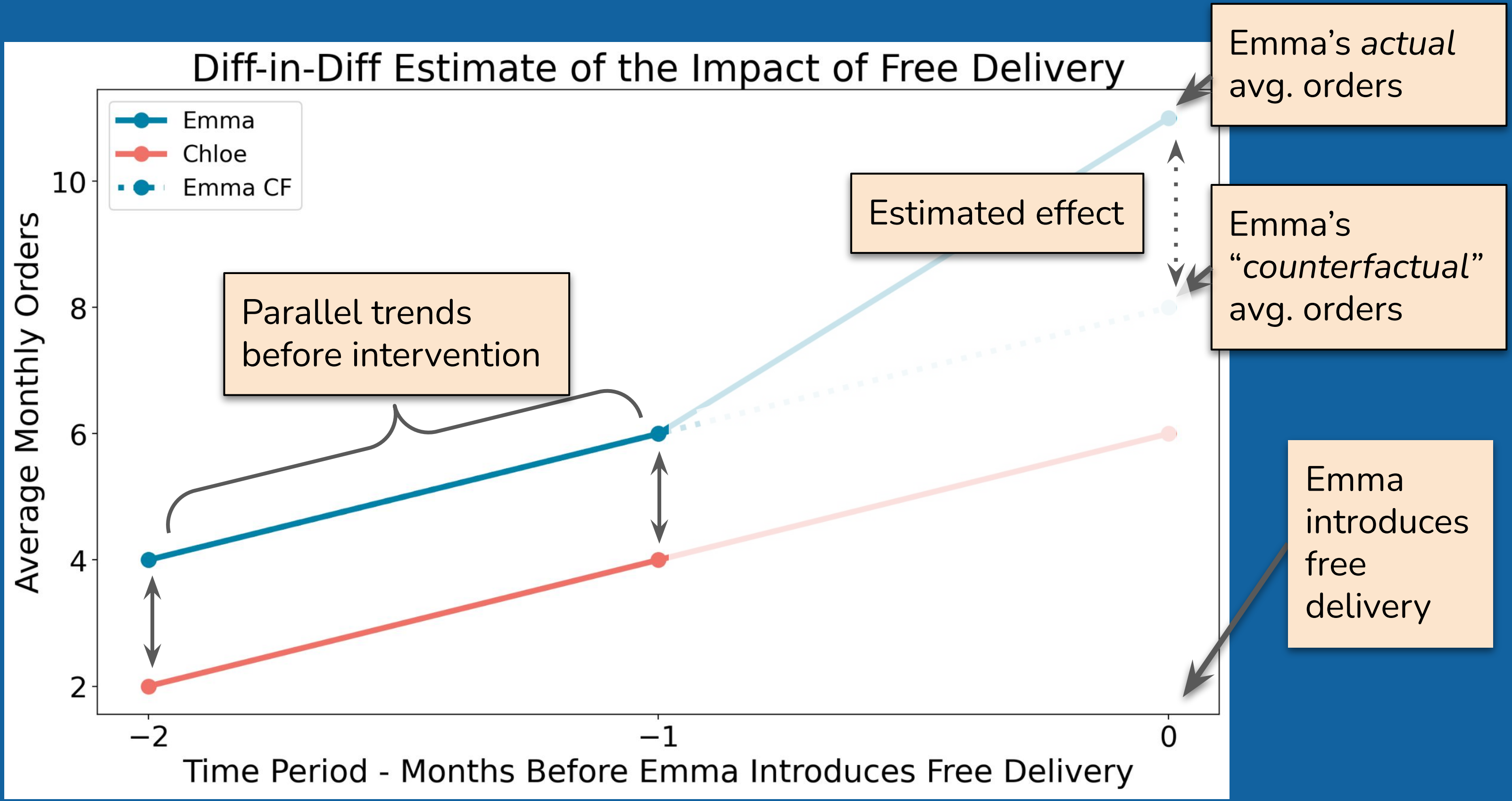
For both Emma’s and Chloe’s stores, we have a few data points of average monthly orders customers made before the introduction of free shipping.

Luckily, it looks like wool and yarn orders have similar **trends** (but they are independent).

2. Diff-in-diff



2. Diff-in-diff



2. Diff-in-diff

Assumptions:

- Common trend assumption
- No other changes over time to treated and untreated
- Trends persist over time

Pros:

- Intuitive
- Easy to implement

Cons:

- Usually only 2 data points per group (or very few)

Conclusion: OK but not great (very popular though). 😏

Synthetic controls - Discussion

- For synthetic controls to work well, i.e. produce good estimates of the effect size, the following conditions need to be met:
  - there are time series sets that can be used to synthetise a control.
  - these are **unaffected** by the treatment.
  - the relationship between these sets and the treated set remains stable over time (post treatment).
  - The treatment is the only thing that affects the exposed group, and no other effects impact the treatment and/or the control groups.

Quick Recap

We explored some of the tools at our disposal when randomised controlled trials are not an option.

The tools we saw try to create a proper *control* group so we could estimate the effect a *treatment* had.

Network Effects

Randomised controlled trials rest on the assumption of **SUTVA**: Stable Unit Treatment Value Assumption.

Network Effects

Randomised controlled trials rest on the assumption of **SUTVA**: Stable Unit Treatment Value Assumption.

tl;dr: the response of a person* to a treatment depends only on their own treatment allocation, and is not affected by the treatment allocation of others.

* or any unit/subject/participate of interest in an experiment

Network Effects

Randomised controlled trials rest on the assumption of **SUTVA**: Stable Unit Treatment Value Assumption.

tl;dr: the response of a person* to a treatment depends only on their own treatment allocation, and is not affected by the treatment allocation of others.

e.g.: if Emma's email campaign is so successful that people receiving the orange email buy so much wool there's nothing left for the people receiving the blue email, SUTVA is violated.

* or any unit/subject/participate of interest in an experiment

Network Effects

Randomised controlled trials rest on the assumption of **SUTVA**: Stable Unit Treatment Value Assumption.

tl;dr: the response of a person* to a treatment depends only on their own treatment allocation, and is not affected by the treatment allocation of others.

e.g.: if Emma's email campaign is so successful that people receiving the orange email buy so much wool there's nothing left for the people receiving the blue email, SUTVA is violated.

There are tools in the data scientist's toolbox for cases where network effects violate SUTVA.

* or any unit/subject/participate of interest in an experiment

Immoral and/or Illegal

Classic example: does smoking at a young age cause lower university acceptance rates?

We can't *legally* let a group of teens smoke (and keep a control group).

Even if it *was* legal, we *shouldn't*.

Too expensive or otherwise infeasible

For example, if we want to test the performance of ads posted in Tube platforms. We can’t have only 50% of the commuters see the ad.

We can’t tell who saw it and who didn’t.

Also..

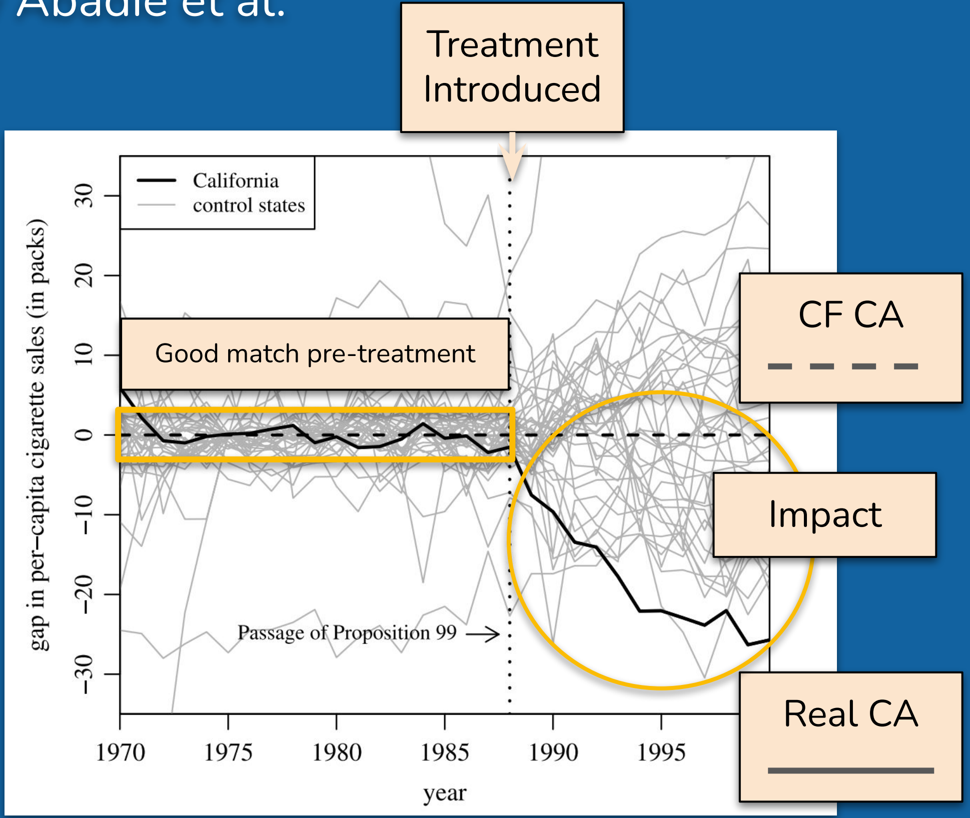
- All we have is observational data, captured some time in the past

- Bad user experience

Synthetic Control Methods for Comparative Case Studies: Estimating the Effect of California's Tobacco Control Program by Abadie et al.

Table 2. State weights in the synthetic California

State	Weight	State	Weight
Alabama	0	Montana	0.199
Alaska	–	Nebraska	0
Arizona	–	Nevada	0.234
Arkansas	0	New Hampshire	0
Colorado	0.164	New Jersey	–
Connecticut	0.069	New Mexico	0
Delaware	0	New York	–
District of Columbia	–	North Carolina	0
Florida	–	North Dakota	0
Georgia	0	Ohio	0
Hawaii	–	Oklahoma	0
Idaho	0	Oregon	–
Illinois	0	Pennsylvania	0
Indiana	0	Rhode Island	0
Iowa	0	South Carolina	0
Kansas	0	South Dakota	0
Kentucky	0	Tennessee	0
Louisiana	0	Texas	0
Maine	0	Utah	0.334
Maryland	–	Vermont	0
Massachusetts	–	Virginia	0
Michigan	–	Washington	–
Minnesota	0	West Virginia	0
Mississippi	0	Wisconsin	0
Missouri	0	Wyoming	0



Journal of the American Statistical Association June 2010, Vol. 105, No. 490, Applications and Case Studies

JakeVDP did it better

Jake Vanderplas - Statistics for Hackers - PyCon 2016 (link)