1 Funneling clicks through the widget

1.1 Introduction

The aim of the task is to understand the "funneling" of clicks through the size selection widget, in particular the difference between **Partner A** and **Partner B**. These events are encoded in the `analytics.events` table in the `event_name` field. With the SQL query

```
select
    event_name,
    sum((partner_key = 'Partner A')::int) as "A",
    sum((partner_key = 'Partner B')::int) as "B"
from analytics.events
group by event_name;
```

we find the events shown in Table 1.

Playing with the widget, we infer the sequence of events we want to evestigate in detail as

1. `opened_editor`
2. `opened_brand_list`
3. `selected_brand`
4. `selected_category`
5. `selected_size`
6. `completed_profiling`

Let's break down the data by `product_domain` using the SQL query

```
select product_domain, partner_key, count(event_name)
from analytics.events
group by product_domain, partner_key
order by product_domain, partner_key;
```

The results are given in Table 2.

We see that Partner A and Partner B have significant portions of disjoin product domains. In addition, the product domain "Dresses" does have any `selected_category` events (cf. Table 1 and §1.2). Based on these observations, it makes more sense to "funnel" the events **for each product domain separately** rather than in aggregate, which was requested in the assignment.

event_name	Partner A	Partner B
viewed_product	3492288	1842017
added_variant_to_cart	286695	130564
ordered_variant	44962	23923
opened_editor	31073	6608
opened_brand_list	29478	8745
selected_brand	16086	3928
selected_category	3477	3488
selected_size	12396	3302
completed_profiling	12367	3291

Table 1: Summary of events by Partner.

product_domain	Partner A	Partner B
Dresses	2982850	51771
Female Shoes	369043	409761
Female Tops	333421	134270
Female Bottoms	116141	45007
Male Bottoms	–	44183
Male Shoes	–	105529
Male Tops	–	140946
Swimwear Bottoms	18036	–
Swimwear Tops	64358	–
NONE	44973	1094399

Table 2: Summary of products by Partner.

1.2 Funnel by product domain

To get the click rates by product domain we query as

```
select event_name, count(event_name) as n
from analytics.events
where
    (partner_key = '{partner_key}') and
    (product_domain LIKE '{product_domain}')
group by event_name;
```

For brevity, we only compare within the product domains where both Partner A and Partner B are active:

- "Dresses" (Figure 1),
- "Female Shoes" (Figure 2),
- "Female Tops" (Figure 3),
- "Female Bottoms" (Figure 4).

Three out of four show clearly higher rates for Partner B over Partner A.

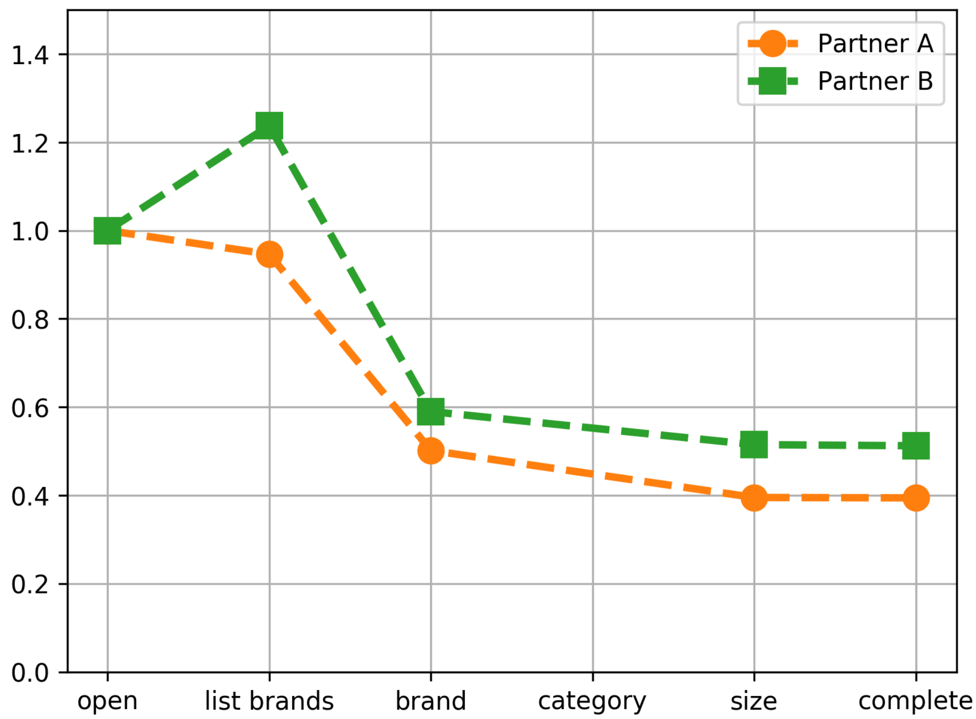


Figure 1: Funneling for product domain “Dresses”.

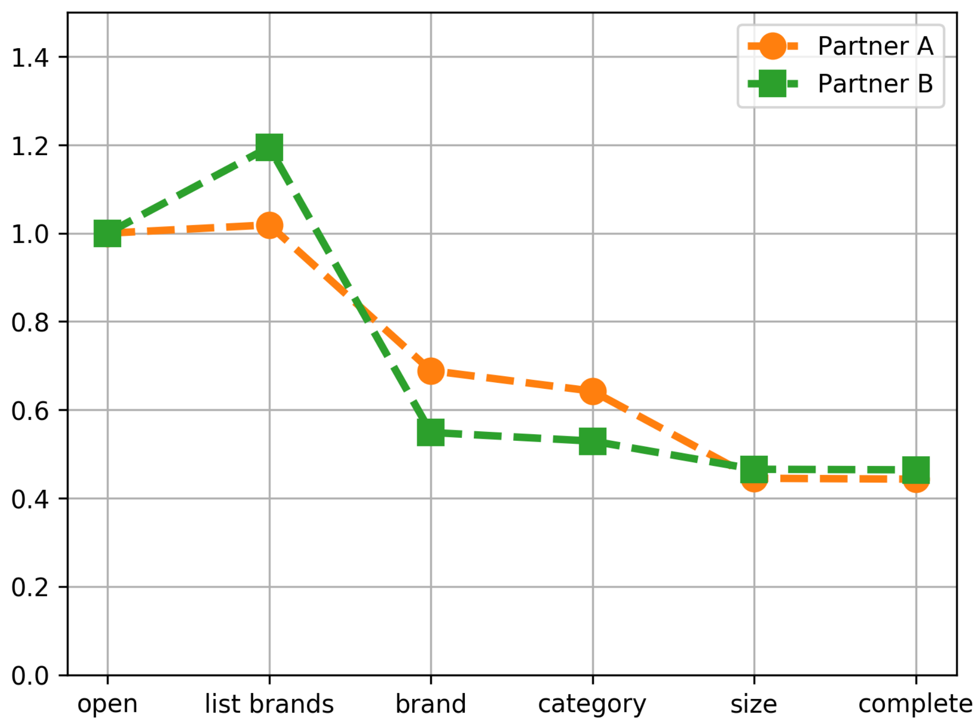


Figure 2: Funneling for product domain “Female Shoes”.

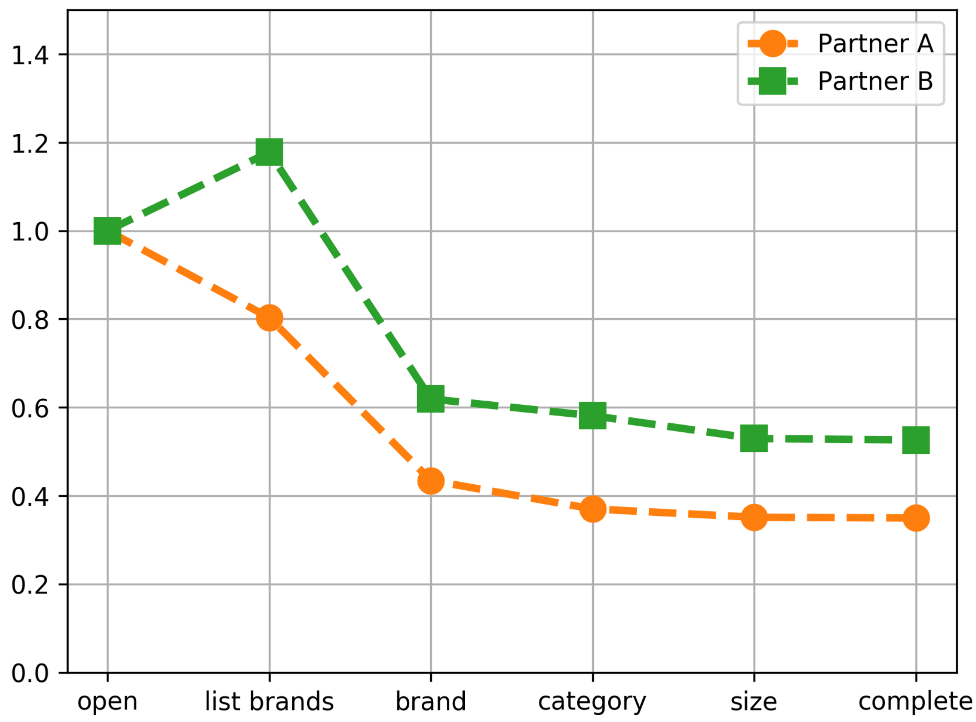


Figure 3: Funneling for product domain “Female Tops”.

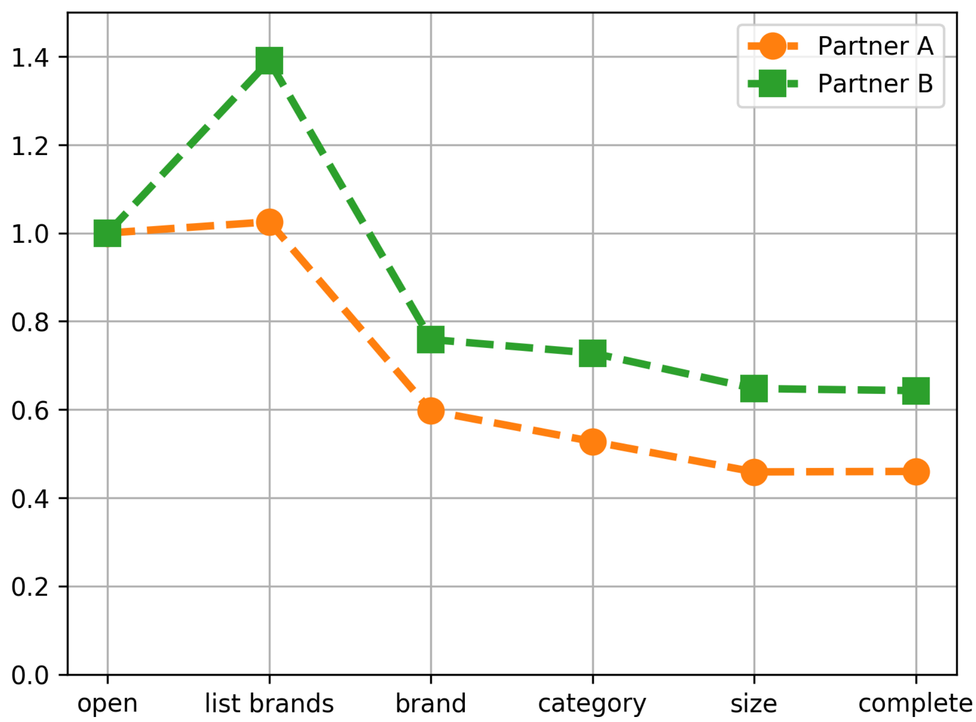


Figure 4: Funneling for product domain “Female Bottoms”.

2 Observations

2.1 Higher funneling for B over A

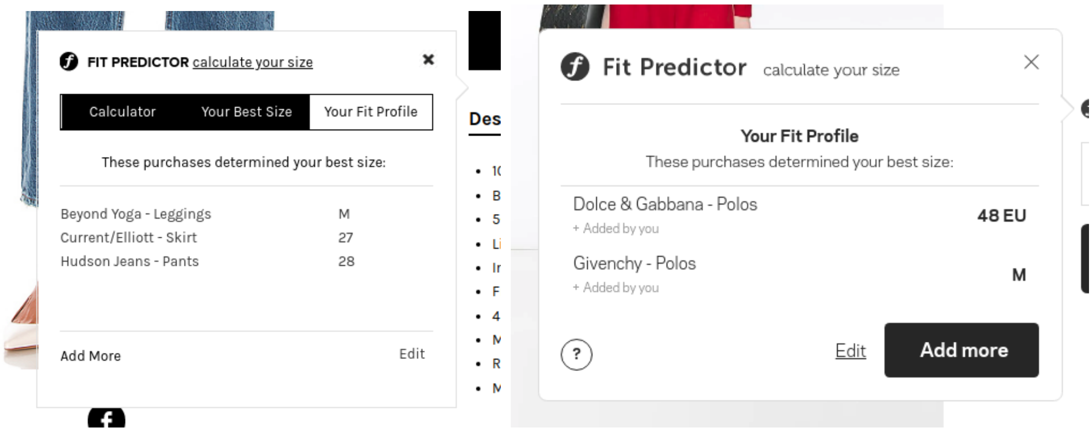
The highlighted ones among product domains

Dresses, Female Shoes, **Female Tops**, **Female Bottoms**

show clearly higher "funneling" rates for Partner B over Partner A.

Possible reasons are:

- the customer enters several purchased items via the "add more" button on website B because the meaning and appearance of the button is clearer (assuming there is no `opened_editor` event). Compare, for example **revolve.com** and **farfetch.com**:



- it is easier to interact with the drop-down list of website B resulting in fewer misclicks.

- the widget implementation is more robust on website B, i.e. the widget is more likely to stall after the `opened_editor` event on website A in some browsers.

Non-reasons are:

- the visibility of the widget, since the baseline is the `opened_editor` event;

- male vs. female customers, since we are looking at dresses and such.

2.2 Sharp fall after `opened_brand_list` event

It appears that customers click on the brand list inside the widget a lot, but then abandon the widget. This is the case for both Partner A and Partner B.

Possible reasons:

- They don't find the brands they own. This creates a moment of confusion and the reflexive hope that upon a second click the situation will be different. Possibly, Partner B (with the high spike on `opened_brand_list` event) is the fancier of the two (indeed, if it is **farfetch.com**, that would be consistent with the earlier observation).

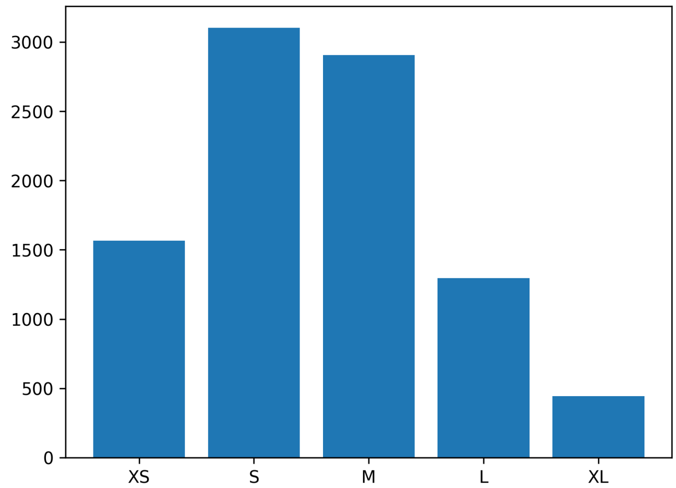
- They don't find *another* brand they own after some successful entries.
- They don't remember the items or the sizes they own and have to look those up first, i.e. click on the brand list, close the list, look up item, open the list again.

3 Predicted size histogram

With the query

```
select prediction_size, count(*)
from analytics.events
where
    (partner_key = 'Partner A') and
    (product_domain = 'Dresses') and
    (event_name = 'completed_profiling')
group by prediction_size;
```

we get the predicted size at the `completed_profiling` event for dresses on Partner A's website. The results are in Figure 5.



prediction_size	#
12	1
2	3
4	6
6	2
8	3
L	1296
M	2905
S	3103
XL	442
XS	1566
XXS	6
NULL	232

Figure 5: Predicted size distribution for dresses of Partner A.

4 A/B testing

Does the fit predictor make more people purchase the products? Presumably, the purchase of a product is indicated by the `ordered_variant` event. We can compare the number of those events to the number of `viewed_product` events (or, alternatively, to `added_variant_to_cart` events, but the results are similar). So we define the conversion rate as

$$\text{conversion rate} = \frac{\#\texttt{ordered\_variant}}{\#\texttt{viewed\_product}}. \tag{1}$$

With the query

```
select
    event_name,
    sum((ab_slot1_variant = 'Control')::int) as "Control",
    sum((ab_slot1_variant = 'Test')::int) as "Test"
from analytics.events
where
    (partner_key = 'Partner A')
    and
    (
        (event_name = 'viewed_product') or
        (event_name = 'ordered_variant')
    )
group by event_name;
```

we get the table

event_name	Control	Test
viewed_product	1749420	1742868
ordered_variant	22156	22806
conversion rate	0.012665	0.013085

of the pertinent events for the Control and Test groups, where we have added the conversion rate by hand. The conversion rate has increased in the Test group. Next, we test the statistical significance of the improvement.

4.1 Binomial test

Our null-hypothesis is: a customer purchases a viewed product with the probability p. This probability is determined from the Control group. Let p' denote the corresponding probability for the Test group. We wish to check if $p' > p$ is significant. Since the number of observations n' in the Test group is rather large (although p' is somewhat small), we resort to the normal approximation and compute the z-score:

$$z = \frac{p' - p}{\sqrt{p(1-p)/n'}} \approx 4.965. \tag{2}$$

This gives an α-value of $\alpha \approx 3.43 \times 10^{-7}$. The observed improvement in the conversion rate is therefore highly statistically significant.

This reasoning is contaminated by multiple visits by the same user in a session to the website, mainly because the number is unequal among users. It would be cleaner to identify those visits as one.

4.2 Fisher test

For the Fisher 2×2 test we rephrase the table as

ordered	Control	Test
no	1727264	1720062
yes	22156	22806

by considering the complement of `ordered_variant` event as the no-order case. Aforegoing remarks on contamination by multiple visits apply. Nevertheless, the Fisher exact test statistic value is 0.05%, again highly statistically significant.

Appendix

The Python codes are deposited under

`https://github.com/numpde/misc/tree/master/ssp`