A Lightweight Transformer for Next-Item Recommendation

RecSys 2022
Jeffrey Mei, Cole Zuber, Yasaman Khazaeni

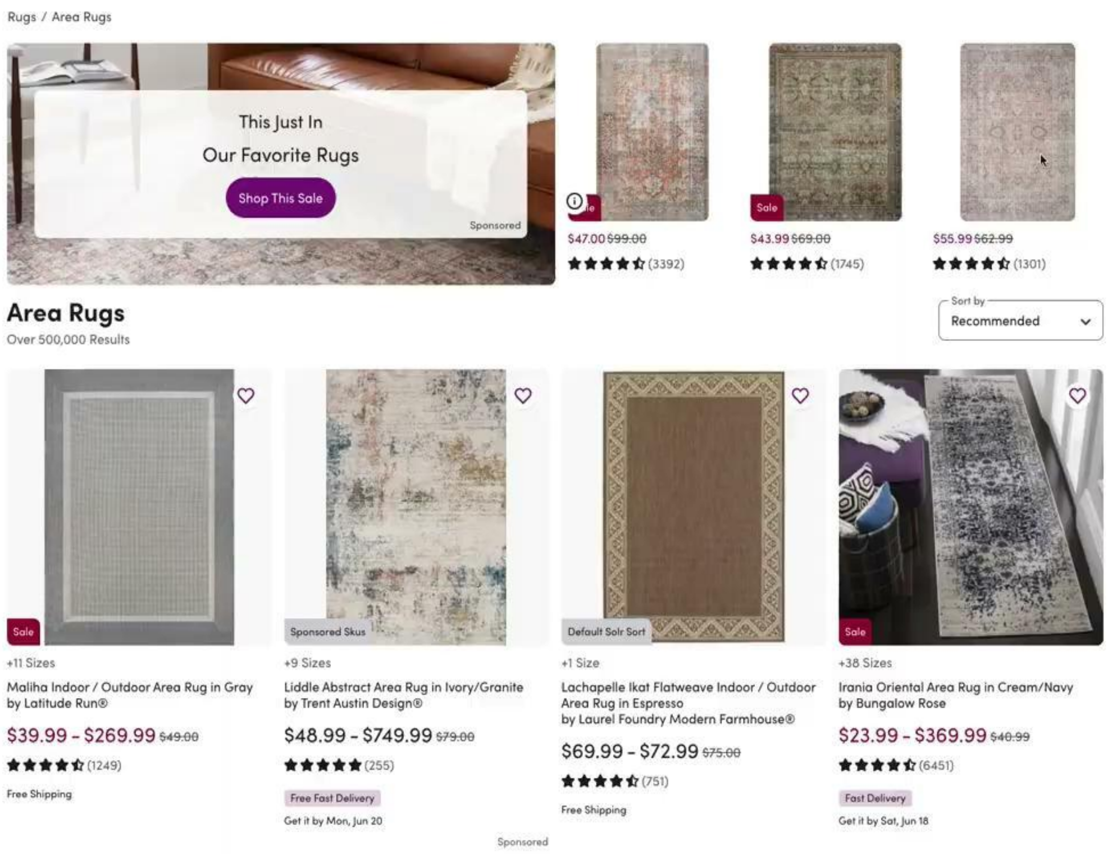
Multi-headed Attention Recommender System

MARS: A lightweight transformer model powering recommendations using **only customer browse** for inference.

Based off SASRec (Kang & McAuley 2018)

It is trained on **customer-item interactions only** - i.e. viewed items and ATC/ordered items (if they exist for a customer), **in the order** they were interacted with.

(link for video)



Offline evaluation

For our purposes - one right answer (**next order**), everything else is wrong

In theory you can use next-ATC, next-view, even all future orders etc.

Standard metrics:

- Recall (position-agnostic)
- nDCG (logarithmic positional discount)

(mean) Recall@6: $$\frac{1+1+1+0}{4} = 0.75$$

$$\textbf{DCG} = \sum_{k}^{N} \frac{2^{rel_k} - 1}{\log_2(k+1)} = \sum_{k}^{4} \frac{1 \text{ if right else } 0}{\log_2(k+1)}$$

(mean) nDCG@6: $$\frac{\frac{1}{\log_2(3+1)} + \frac{1}{\log_2(5+1)} + \frac{1}{\log_2(1+1)} + 0}{4} = 0.47$$

Offline evaluation looks good...

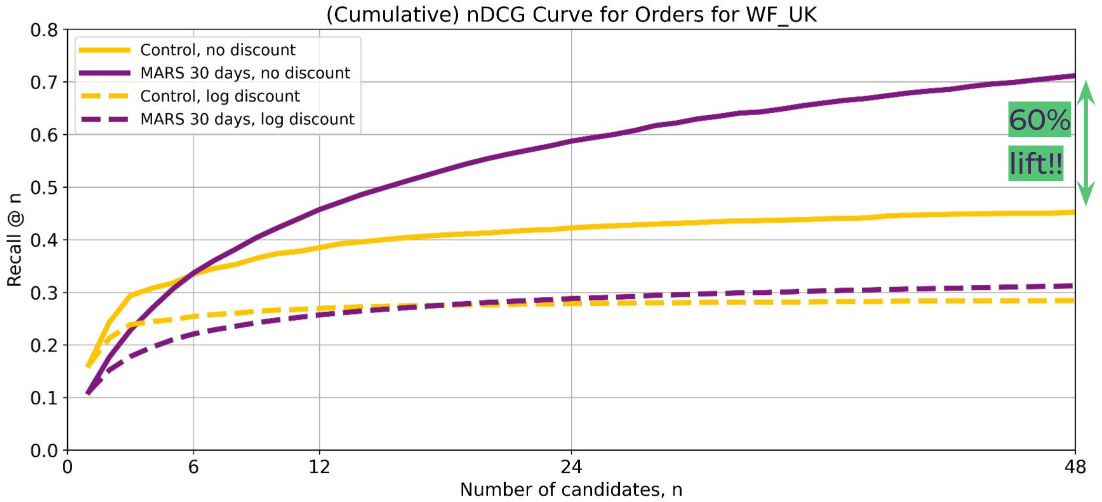
We show 48 results per page, so a very natural metric for us is **Recall@48** or **nDCG@48**.

Recall@48 prediction: **Win**

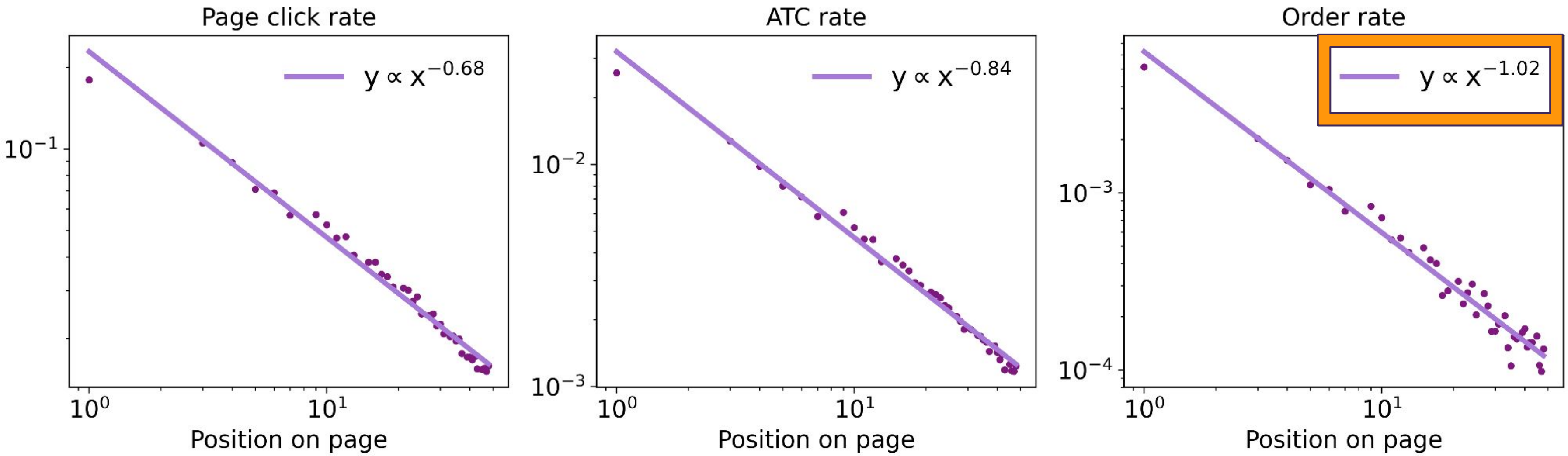
Log_2 nDCG@48 prediction: **Win**

=> Do an A/B test

But the test for WF_UK failed... orders actually **significantly dropped** by >1%.



Determining positional effects empirically



nDCG

$$\frac{\frac{1}{\log_2(3+1)} + \frac{1}{\log_2(5+1)} + \frac{1}{\log_2(1+1)} + 0}{4} = 0.47$$

nDCG$_e$

$$\frac{\frac{1}{3^{1.02}} + \frac{1}{5^{1.02}} + \frac{1}{1^{1.02}} + 0}{4} = 0.38$$

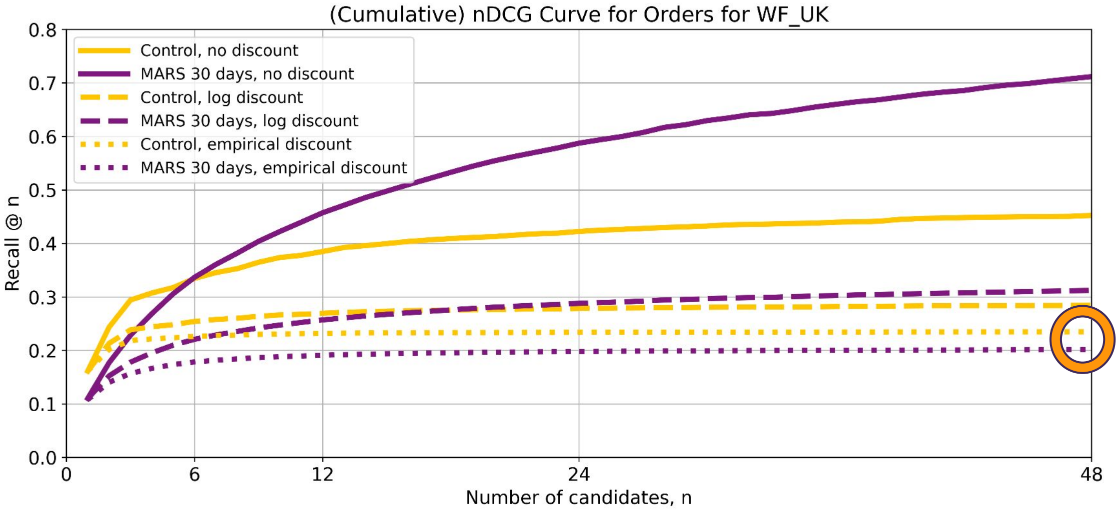
Applying empirical discount

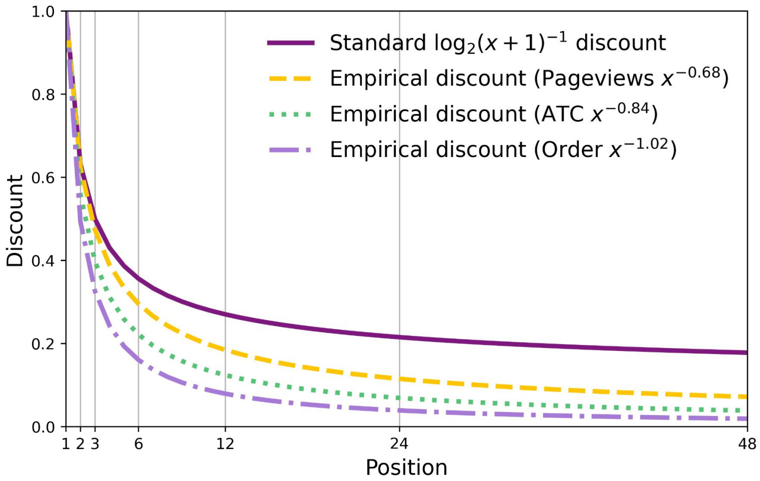
Now we can see that the empirical discount rate of $x^{-1.02}$ actually shows control beating MARS.

True A/B test result: **-1% CVR**

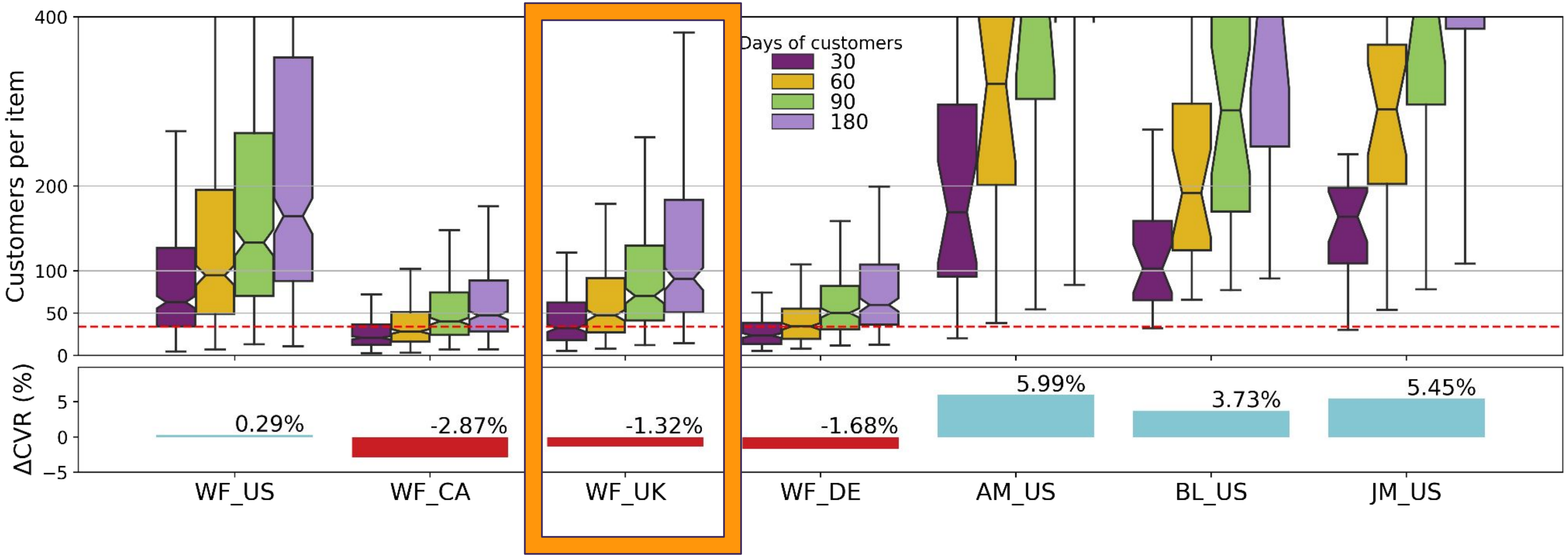
Recall@48 prediction: **Win**
Log_2 nDCG@48 prediction: **Win**
$x^{-1.02}$ nDCG_e@48 prediction: **Loss**

The **standard $\log_2$ discount is too small** to account for our real, observed positional effect.





Data thinness is very correlated with A/B test result



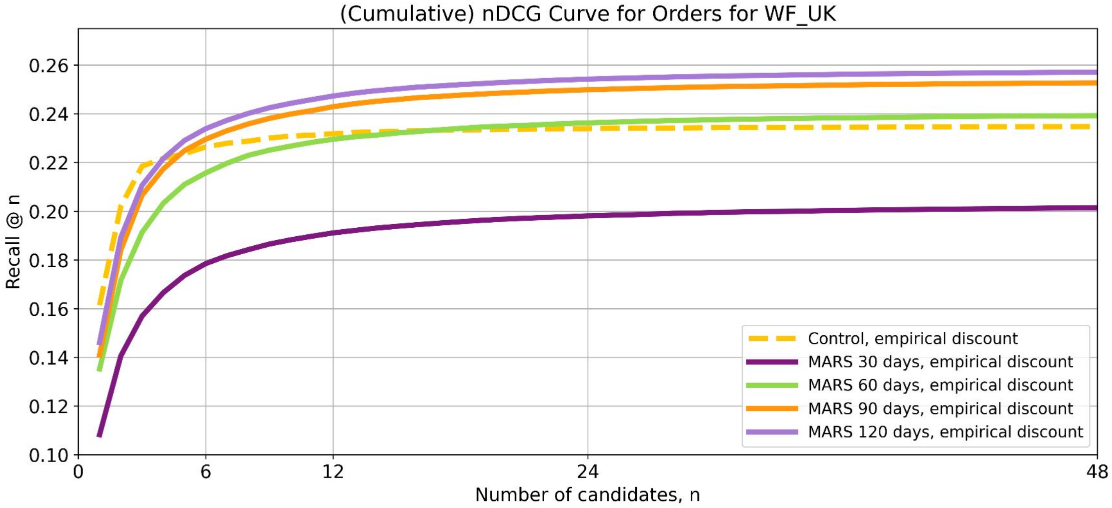
- US stores account for 80% of revenue so the test was an overall win, but we would love to see all stores winning
- Data thinness is different for each store

Increased training data

The benefit of adding more training data eventually saturates

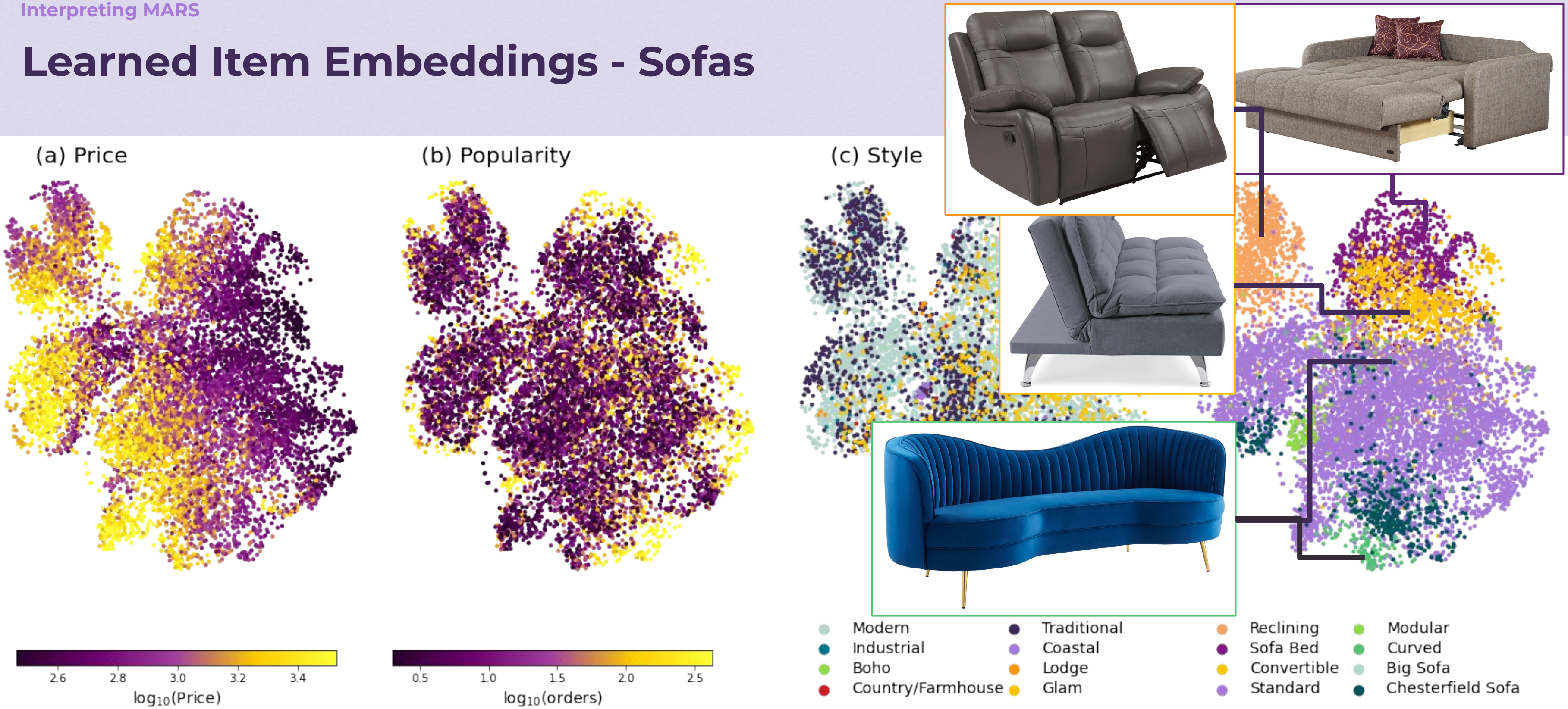
- although we can keep increasing the performance, this also increases training time ~linearly.

More importantly, we can see 90 days gives a very clear lift of (order-discounted) **$nDCG_e$@48 of ~25%.**

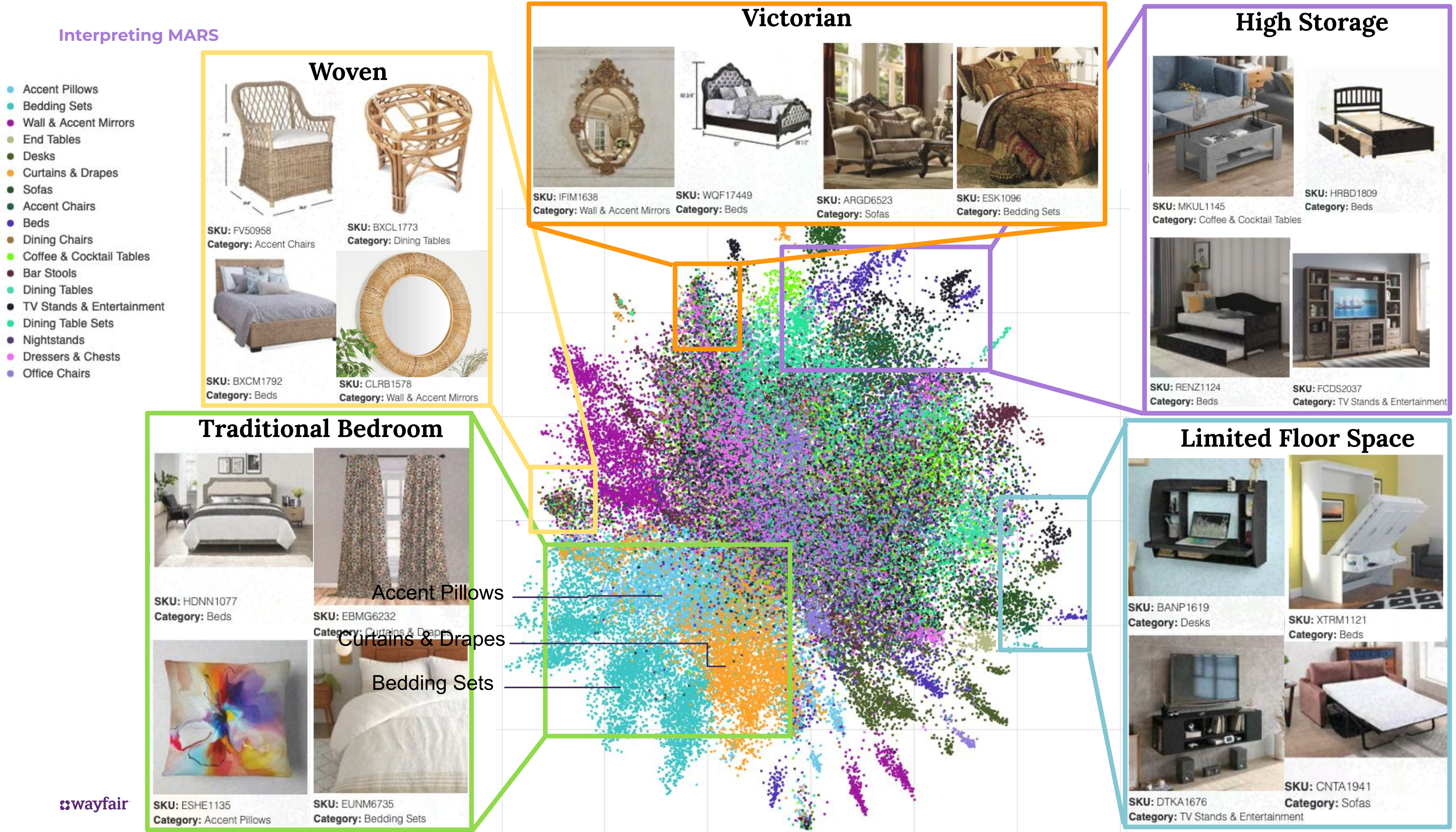


When we A/B tested again using 90 days of training data, CVR for WF_UK **increased 2.4%**!

Learned Item Embeddings - Sofas



Interpreting MARS



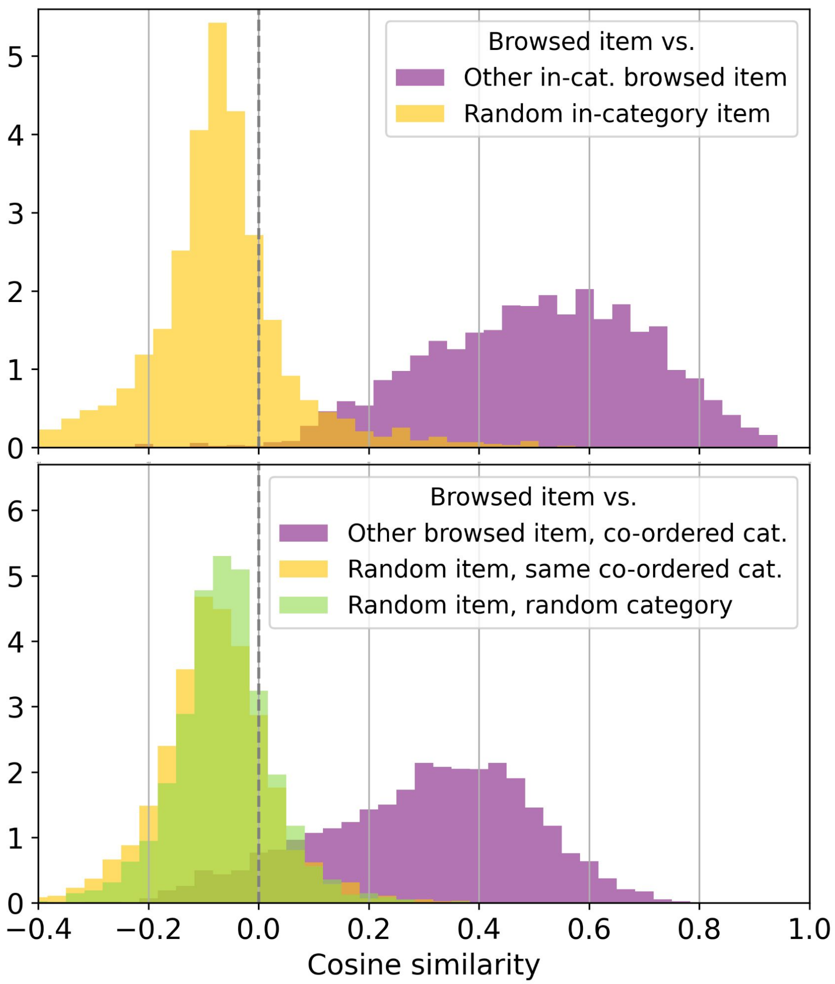
Do these attributes carry across furniture types?

We see high similarity between browsed items for both in-category and cross-category* browse.

Here, we remove the category signal from the embeddings as we want to exclude the effect of categories themselves being overall similar.

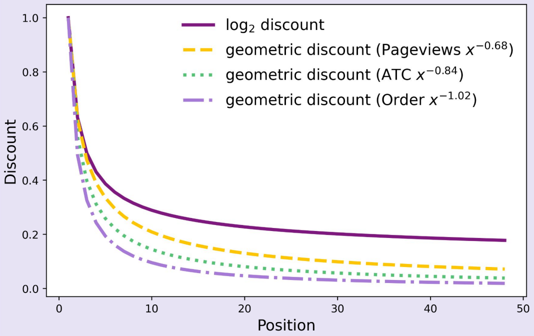
*Commonly co-ordered category e.g.

- Nightstands and beds
- Sofas and coffee tables



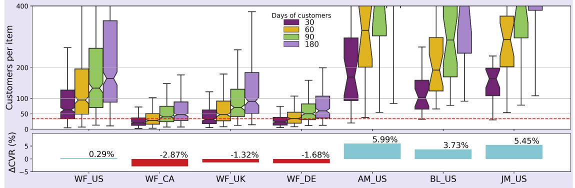
Summary of learnings

Empirically find discount rate



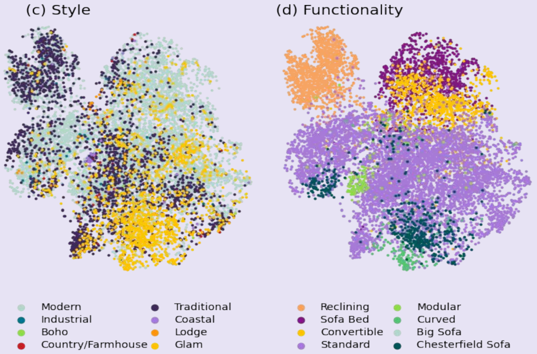
… using historic interaction data by position

Account for data thinness



… by tracking average number of customers viewing each item

Simplicity



… putting the **implicit** in **simplicit**y wayfair®