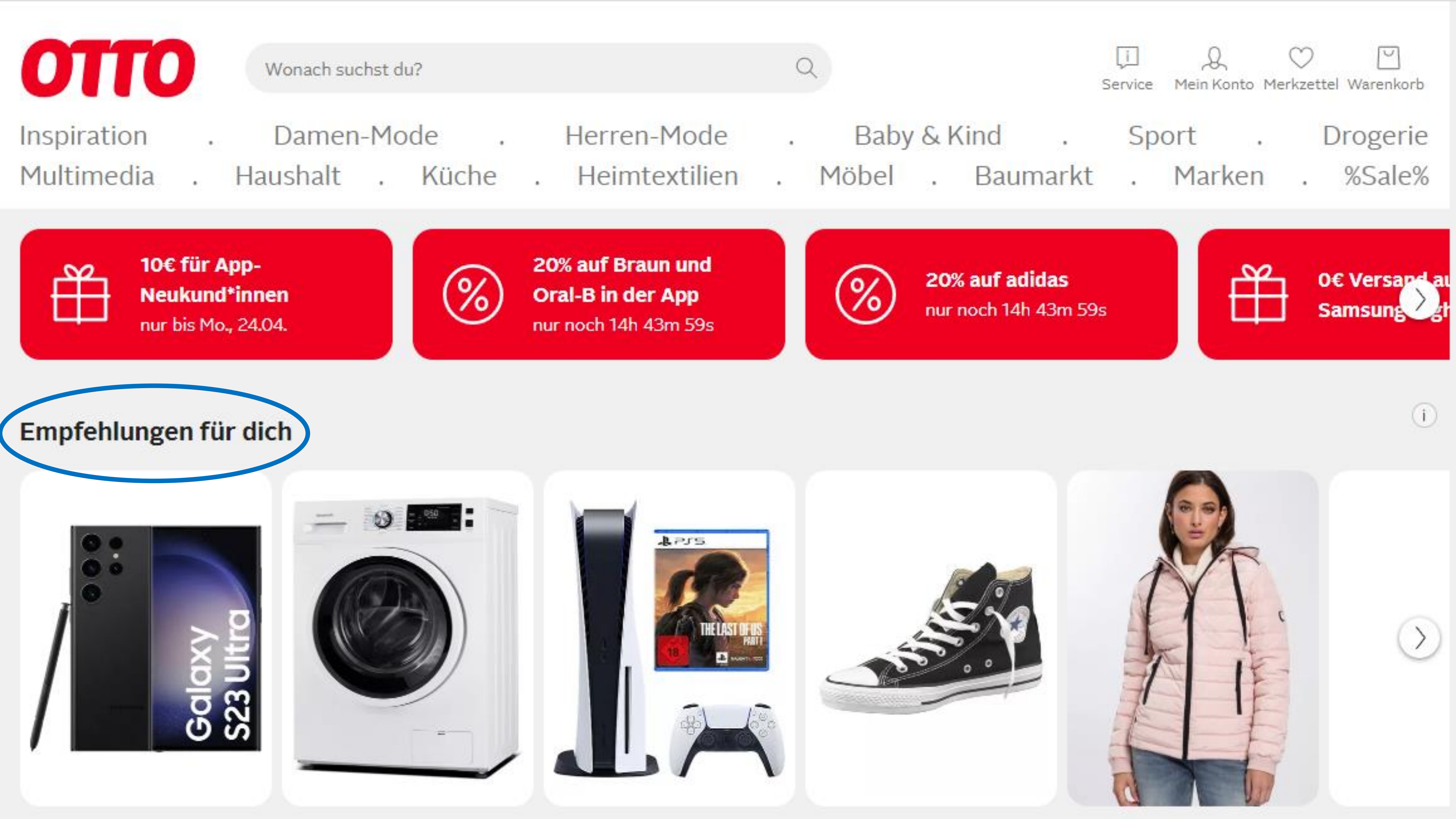
Otto Multi-Objective Recommender System

Timothy Chan

27 Apr 2023

Agenda

- Problem statement
- Evaluation metrics
- Dataset
- User behaviour (EDA and Intuition)
- Approach (models, features, results, key findings)
- Other applications
- Possible improvements
- Conclusion
- Acknowledgement


OTTO
Wonach suchst du?
Service
Mein Konto
Merkzettel
Warenkorb
Inspiration . Damen-Mode . Herren-Mode . Baby & Kind . Sport . Drogerie
Multimedia . Haushalt . Küche . Heimtextilien . Möbel . Baumarkt . Marken . %Sale%
10€ für App-Neukund*innen
nur bis Mo., 24.04.
20% auf Braun und Oral-B in der App
nur noch 14h 43m 59s
20% auf adidas
nur noch 14h 43m 59s
0€ Versand au
Samsung
Empfehlungen für dich
Galaxy S23 Ultra
THE LAST OF US

Without recommender

1,855,603 items

Search / Scroll

What you want

With recommender

20 items

What you want

- Enhance buyer experience
- Increase user engagement and spending

Problem Statement

Based on users' past clicks, carts and orders, I will predict the next

20 clicks, 20 carts, 20 orders

to recommend items for users on the Otto ecommerce platform.

What are clicks, carts, orders?

Evaluation metrics

$$\text{Recall@20} = (0.10 \cdot R_{clicks}) + (0.30 \cdot R_{carts}) + (0.60 \cdot R_{orders})$$

where

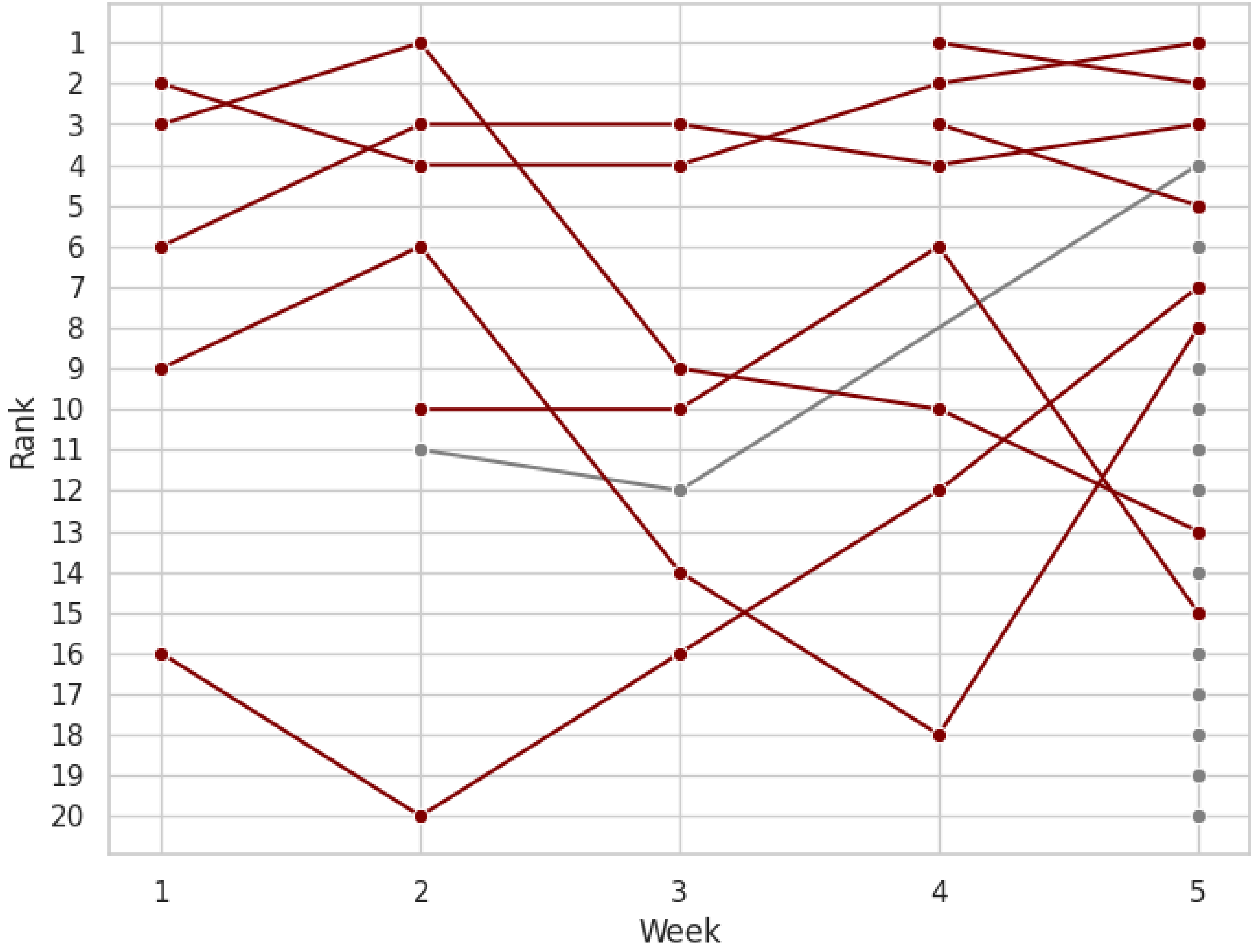
R = TP / P for all 20 predicted items per user

Dataset

- Info given: User ID, Item ID, Timestamp, Type (clicks, carts, orders)
  - Uses implicit data → real and consistent; less sparse
- 5 weeks: Train (1 Aug 22 to 28 Aug 22) and Test (29 Aug 22 to 4 Sep 22)
- Test users are different from train users → cold start problem
- **1 GB size → memory error problem**

User behaviour

Top 20 items by clicks



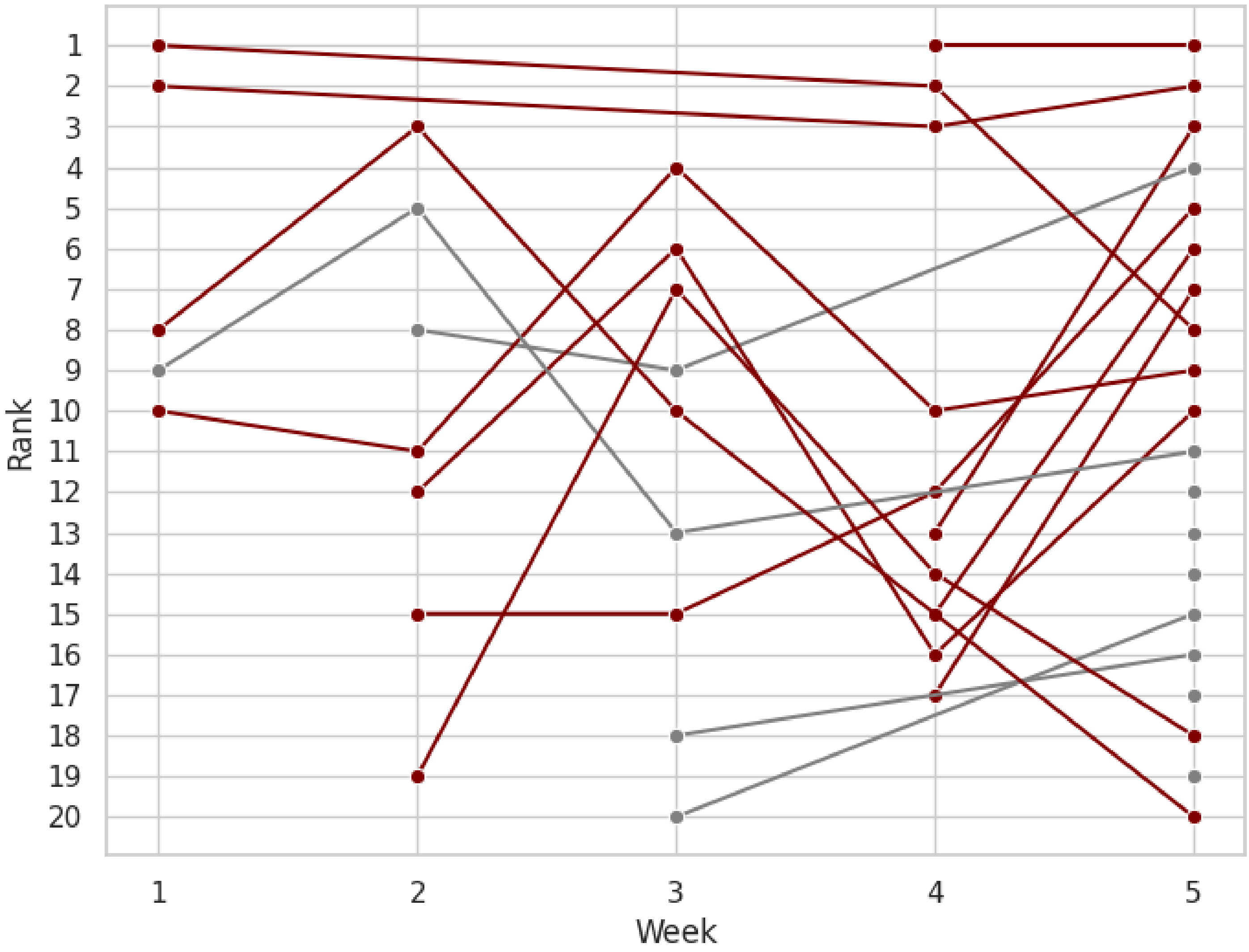
High level EDA due to size of data:
analysed top 20 by weeks

Findings:

- Recency huge factor in recurring top products
- Preceding week has more recurring top

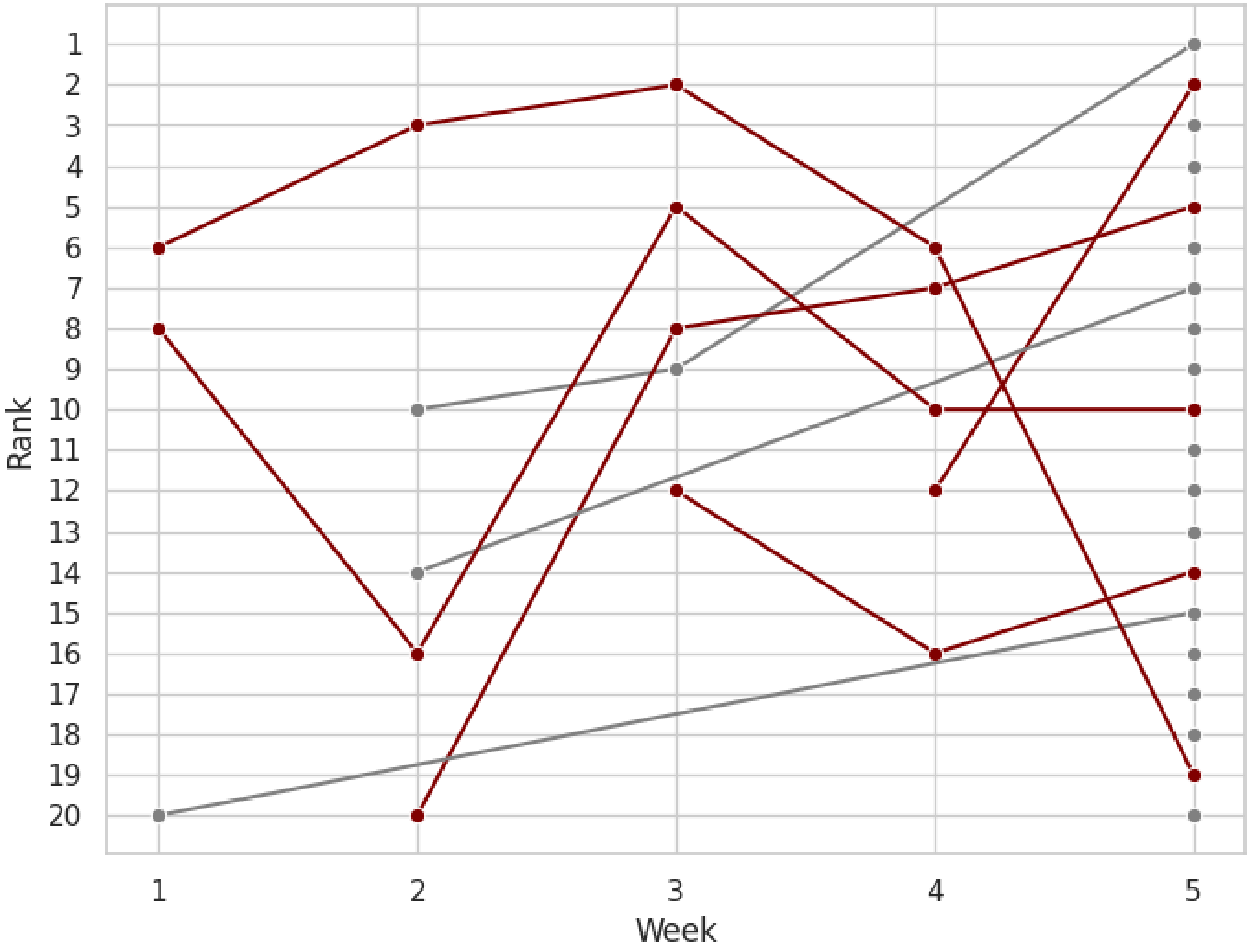
If look at shorter time periods e.g. (minutes / hours / days)
and across all products,
it should have similar trend

Top 20 items by carts



Similarly, more recurring top 20 items in preceding week than before that

Top 20 items by orders



Similarly, more recurring top 20 items in preceding week than before that

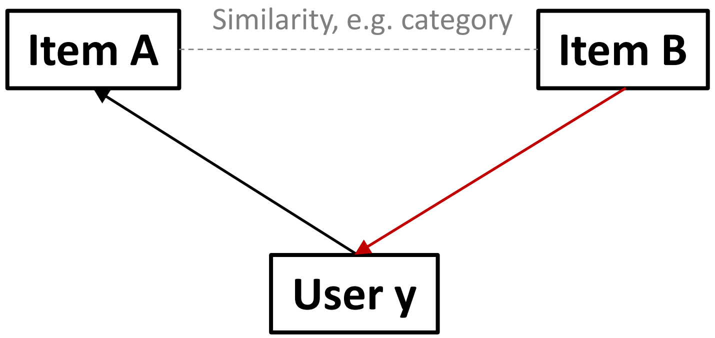
Intuition and quick validation:

Intuition	Validation (Findings)	Action
Recent carts have bigger impact on next orders	Validated to be true	Heavy weight on carts
Open carts have impact on next orders	Validated to be true, but later discovered to overlap with recent carts	None
Repeated orders have impact on next orders	Validated to be true, but small impact	None

APPROACH

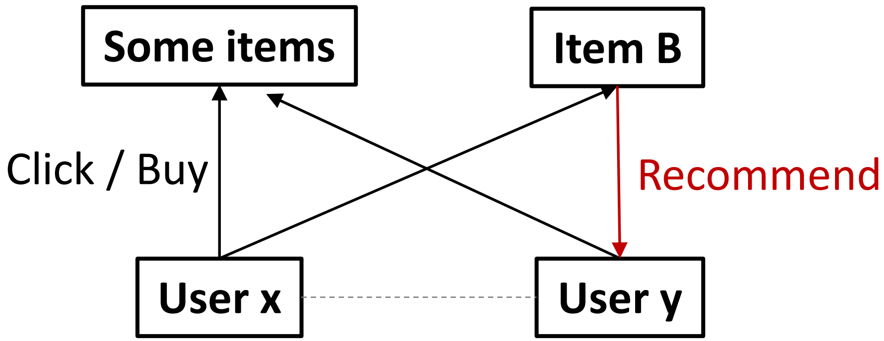
Handling large dataset

Content based filtering



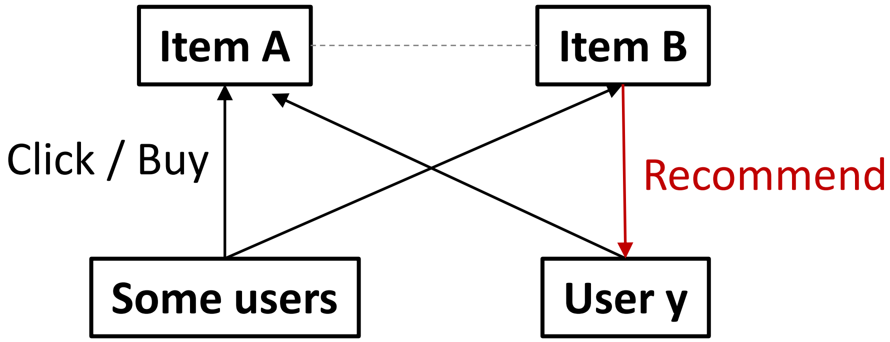
No such similarity features

User based collaborative filtering



Cold start problem: how similar to other users?

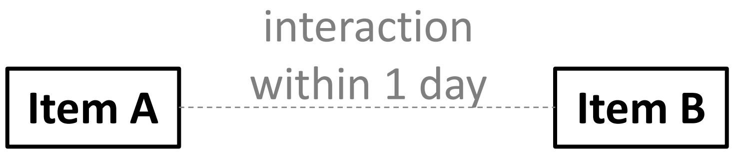
Item based collaborative filtering



Used **covisitation matrix**,
a form of item-item collaborative filtering

- *No need for similarity features*
- *From one item, can find many similar items*
- *Captures users' preference over time*

Covisitation matrix

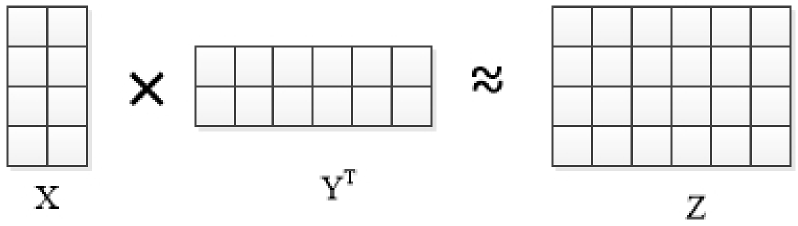


x

n such interactions

Advantages:
- Easy to understand and implement
- Less computational resources comparatively

Matrix factorization (not used here)



Advantages:
- Can handle large datasets by reducing dimensionality of data
- Can capture latent features of users and items, thus improving recommendation score.

Image from: https://www.researchgate.net/figure/Concept-of-Matrix-Factorization_fig3_312480566

Approach 1: Rule-based ranker

2 million items

Weigh and rank:

- Recency
- Type
- Frequency

Consider:

- covisitation pairs

20 items per user

Approach 2: Two-stage re-ranker

30 items ("candidates") per user

(each for train and test)

- Add features to train / infer
- Boosting re-ranker (LGBM / XGB)

20 items per user

Differentiated Covisitation Matrix

#	Using	Weigh by	Used for
1	Clicks, carts, orders	Frequency, Type	Carts, Orders
2	Carts, orders	Frequency; Limit to recent 7 days	Carts, Orders
3	Clicks, carts, orders	Frequency, Recency	Clicks

There are many more possible covisitation matrix, with many different permutations

Reranker features to train / infer

User features

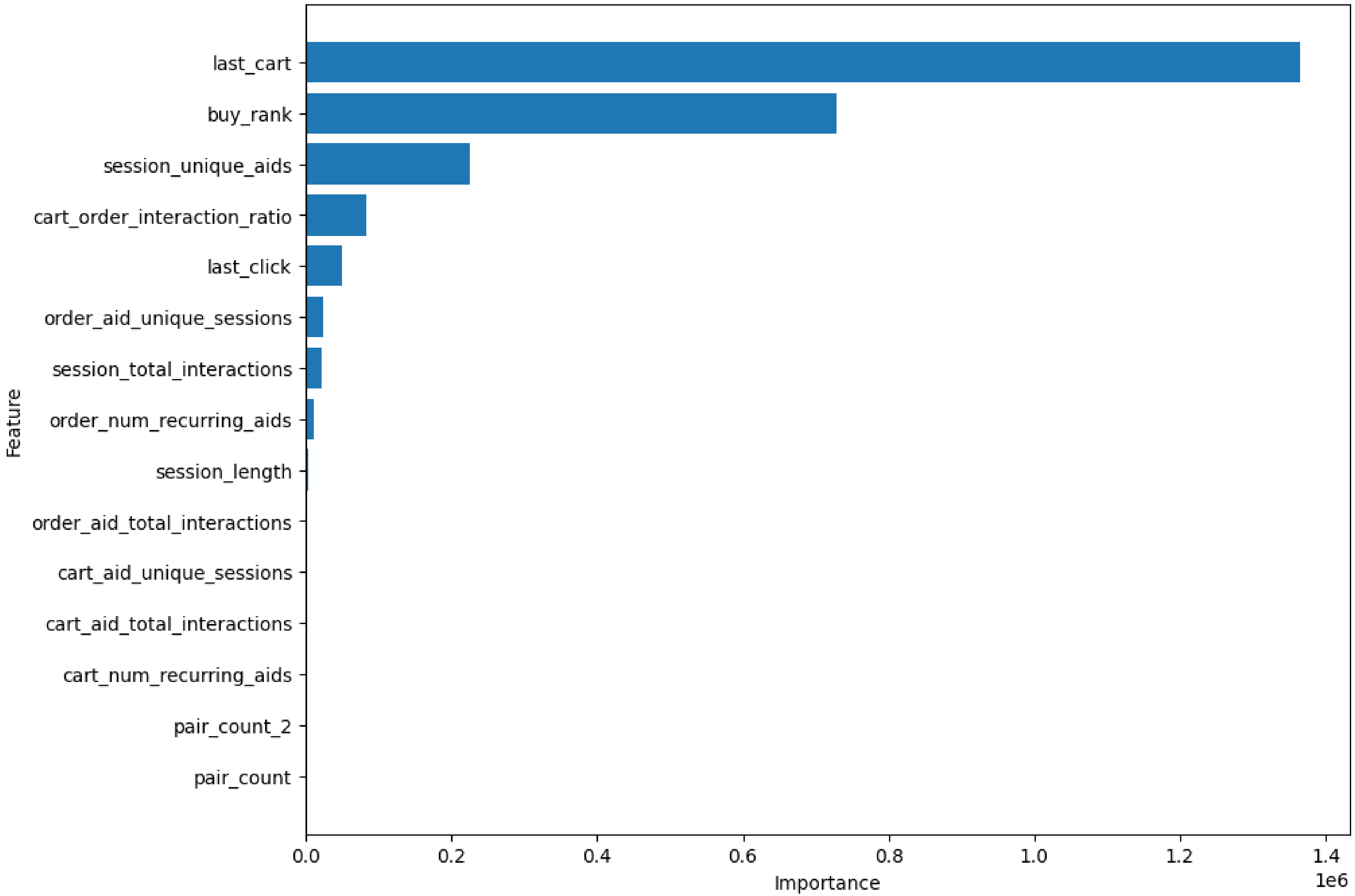
Item features

User-item features

Model Results

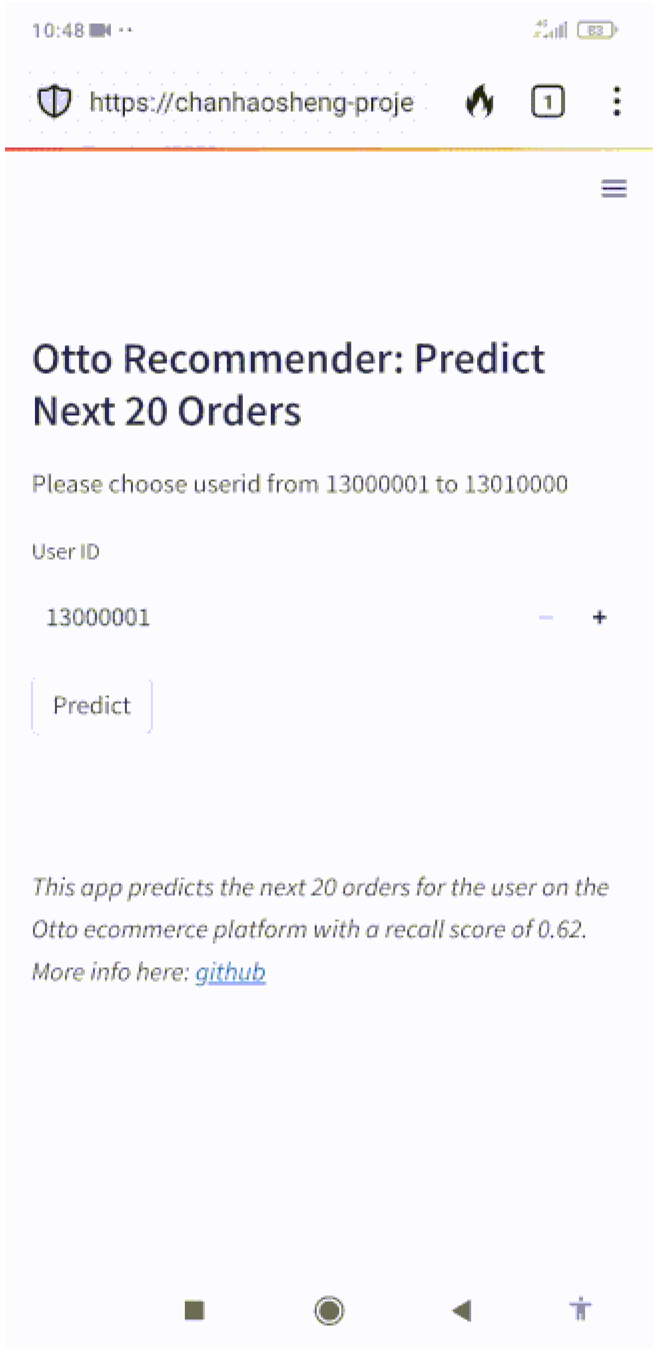
Model	**Classifier**	**Brief Description**	**Recall@20**
1 (baseline)	Rule-based	Most recent items for each user	0.46682
2	Rule-based	Weighted + 3x Covisitation matrix	**0.57648**
3	LGBM Ranker	Reranked Orders from 30 ranked candidates	0.55088
4	XGB Ranker	Reranked Orders from 30 ranked candidates	0.45848

Feature importance (LGBM Ranker)



Deployment (LGBM Ranker)

On streamlit:
https://chanhaosheng-project-submissions-capstonedeployapp-x0mcmu.streamlit.app/



Key findings

Recency

Type (carts)

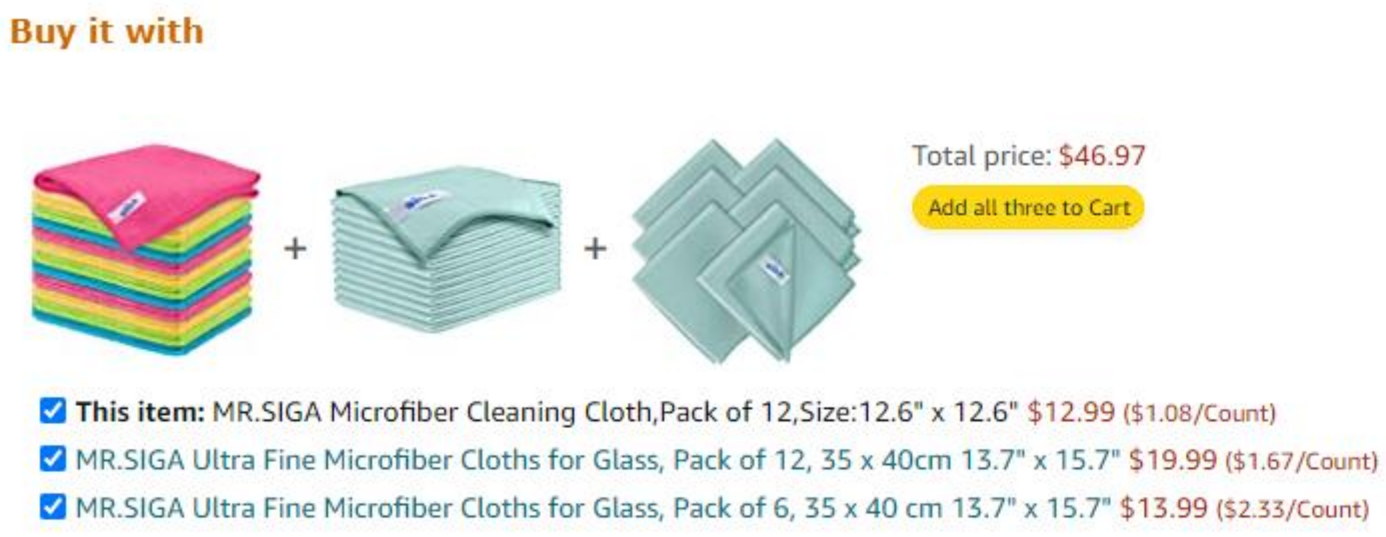
Covisitation pairs

Possible improvements

- Better pipeline including validation for experimentation
- Explore word2vec / matrix factorization techniques to create more candidates
- Explore ensembling. Produce multiple models before aggregating.

Other applications

- New user vouchers to encourage activity: reduce cold start problem
- Identify bundled deals / recommend complementary items



Conclusion

- Respectable recall score of 0.58
- Handling a large dataset requires additional considerations
- Simple information alone can produce so much insight
- Rule based approaches might be more practical than boosting rankers, while also providing reasonable scores
- Boosting rerankers should theoretically improve the scores further, but unfortunately did not do so for this project

Acknowledgement

- Sharings on Kaggle discussion board, especially Chris Deotte and Radek Osmulski
- Instructors Ryan and Ming Jie for their guidance these past 3 months
- Coursemates for their encouragement

All the best and keep in touch!