POLITECNICO MILANO 1863

dressipi

2022 ACM RecSys Challenge

Lightweight Model for Session-Based Recommender Systems with Seasonality Information in the Fashion Domain

POLITECNICO
MILANO 1863

Boston Team Party

Nicola
Della Volpe

Lorenzo
Mainetti

Alessio
Martignetti

Andrea
Menta

Riccardo
Pala

Giacomo
Polvanesi

Francesco
Sammarco

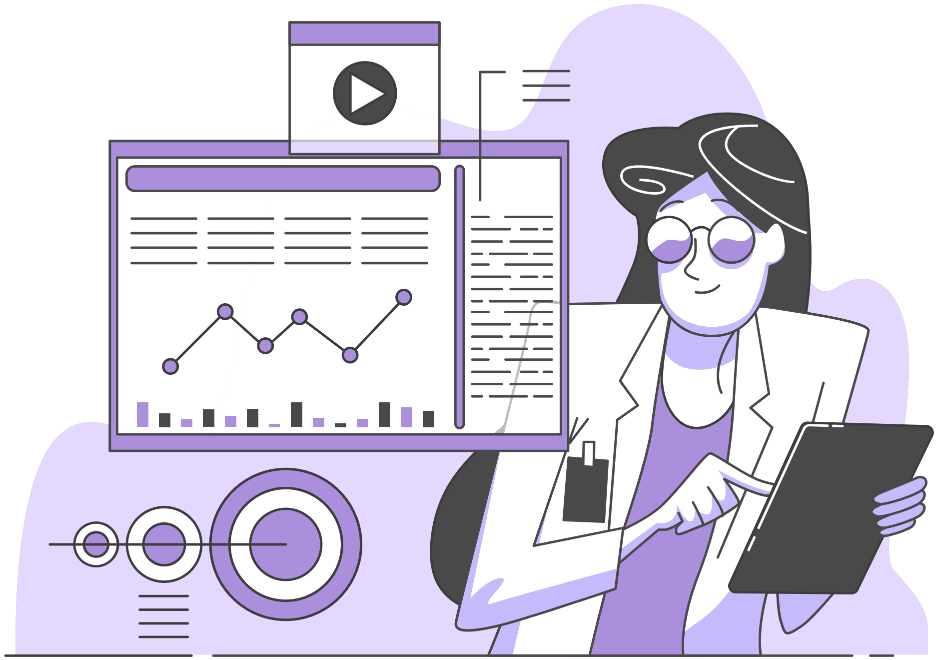
Fernando
B. Pérez Maurera

Cesare
Bernardis

Maurizio
Ferrari Dacrema

Paolo
Cremonesi

Table of Contents



Introduction

Problem formulation and
Dataset description

Problem Formulation

- Top-100 recommendation
- Each recommendation list must contain the purchased item from a subset of possible candidates
- Evaluation metric: MRR

Dataset Description

1.1M

Online retail sessions that resulted in at least a purchase

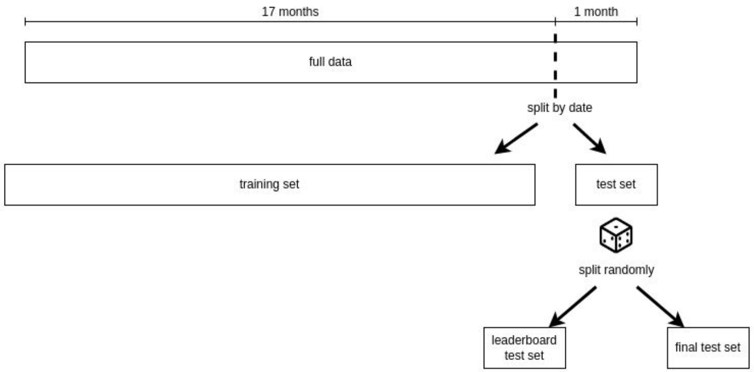
24K

Available items

Available Data

- Sessions
- Purchases
- Item features

Training - Test Split



Data Preparation

Data exploration, splitting,
pre-processing, and
feature engineering

Data Exploration

01 Anomaly Detection

No anomaly detected in the dataset

02 Feature Analysis

Only **0.02%** of the items have no interaction at all in the training dataset

03 Session Analysis

- Average duration of a session (viewed items): around **5**, with min of 1 and max of 100
- "Cold" sessions: around **1.38%** in the test month

04 Seasonalities

- Majority of purchases made in the evening
- Months with more purchases: November 2020 and May 2021

Data Splitting and Pre-processing

Split

Split	Date Interval	Num. of Sessions
Training	01/01/2020 - 31/04/2021	918382
Validation	01/05/2021 - 31/05/2021	81618
Leaderboard	01/06/2021 - 30/06/2021	50000
Final Test	01/06/2021 - 30/06/2021	50000

Interaction Weighting

Views-purchases distinction

Views are weighted with a value $\alpha \in (0, 1)$.

Cyclic decay

Give bigger weight to interactions that are in a period of the year closer to a reference timestamp

Exponential decay

Reduce weight of the interactions as the distance of these increments with respect to a reference timestamp

Feature Engineering – Item Features

904 tuples
(category, value)

Multi-Label
Encoding

...

Feature Engineering - Booster Features

Item Features Embeddings

Using a Variational Autoencoder with a latent space of size 32 with MLE vector as input

Embeddings Aggregation

Aggregating through a sum the embeddings of the items belonging to the same session

RecVae Embeddings

Sessions as input, output of the encoder used as the additional representation of the sessions

Recommenders Scores

Provided by our basic models

Seasonal Tendencies

Tendency of an item to be purchased or viewed in a specific season or subset of seasons or to be all-seasonal

Our Solution

Proposed solution
and results

Candidate Selection

01

Baseline Models

Top Popular
K-Nearest Neighbors:
- ItemKNN CF CBF
- UserKNN CF

Graph Based:
- RP3Beta

EaseR

02

Deep Learning Models

Autoencoders:
- RecVAE
- MultVAE

Recurrent Neural Networks:
- GRU4Rec

Ranking

Step 1 — **Training Candidate Production**: discarding duplicate recommendations but keeping all scores on separate columns

Step 2 — **Candidate Cleanup** & **Ground truth Insertion**: keep only sessions where the purchased item was produced

Step 3 — **GBDT model Training**: on candidates augmented with booster features

Step 4 — **Test Candidate Production**: basic models trained on training and validation set

Step 5 — **GBDT model Prediction** & **Re-ranking**: on candidates augmented with booster features

LightGBM

dmlc XGBoost

Results

Model	MRR Validation	View-Purchase Weight	Uses Cycling Decay	Exponential Decay Weight
GRU4Rec	0.17953	-	-	-
RP3Beta	0.15768	0.2	No	-
EaseR	0.15518	0.5	Yes	182
UserKNN CF	0.14962	0.2	Yes	182
ItemKNN CF+CBF	0.14886	0.5	Yes	182
RecVAE	0.14748	0.5	Yes	182
MultVAE	0.13004	0.5	No	365

Model	3-Fold CV MAP (Validation)	MRR (Public Leaderboard)
LightGBM	0.49115	0.18800
XGBoost	0.46390	0.18347

Conclusions

Takeaways and
possible improvements

Takeaways

- For this challenge: not insert the true label (purchase) in the booster training dataset
- Recurrent models exploit the concept of sequence
- Interaction weighting exploits the temporal information of the sessions (explicit URM)

Possible Improvements

Multiple Model Instance

Use multiple instances of the same models with different objectives for candidate generation

BERT4Rec

BERT4Rec could be tuned and used as another candidate selector

Segmentation

Using different models to recommend sessions depending on their length

Thanks!

Do you have any questions?

lorenzo.mainetti@mail.polimi.it
+1 (928) 225-7715

Code available at
https://tinyurl.com/4h52ean6