Improving Recommendation Speed Using Association Rules

Eyad Kannout and Hung Son Nguyen

University of Warsaw

-

29th international Workshop on Concurrency, Specification and Programming

(CS&P'21)

Agenda

- Goal/Motivation of this research
- Overview for Factorization Machines and Association Rules
- FMAR Recommender System
- Experiments/Evaluation methodology
- Results
- Future work

Introduction / Problem Definition

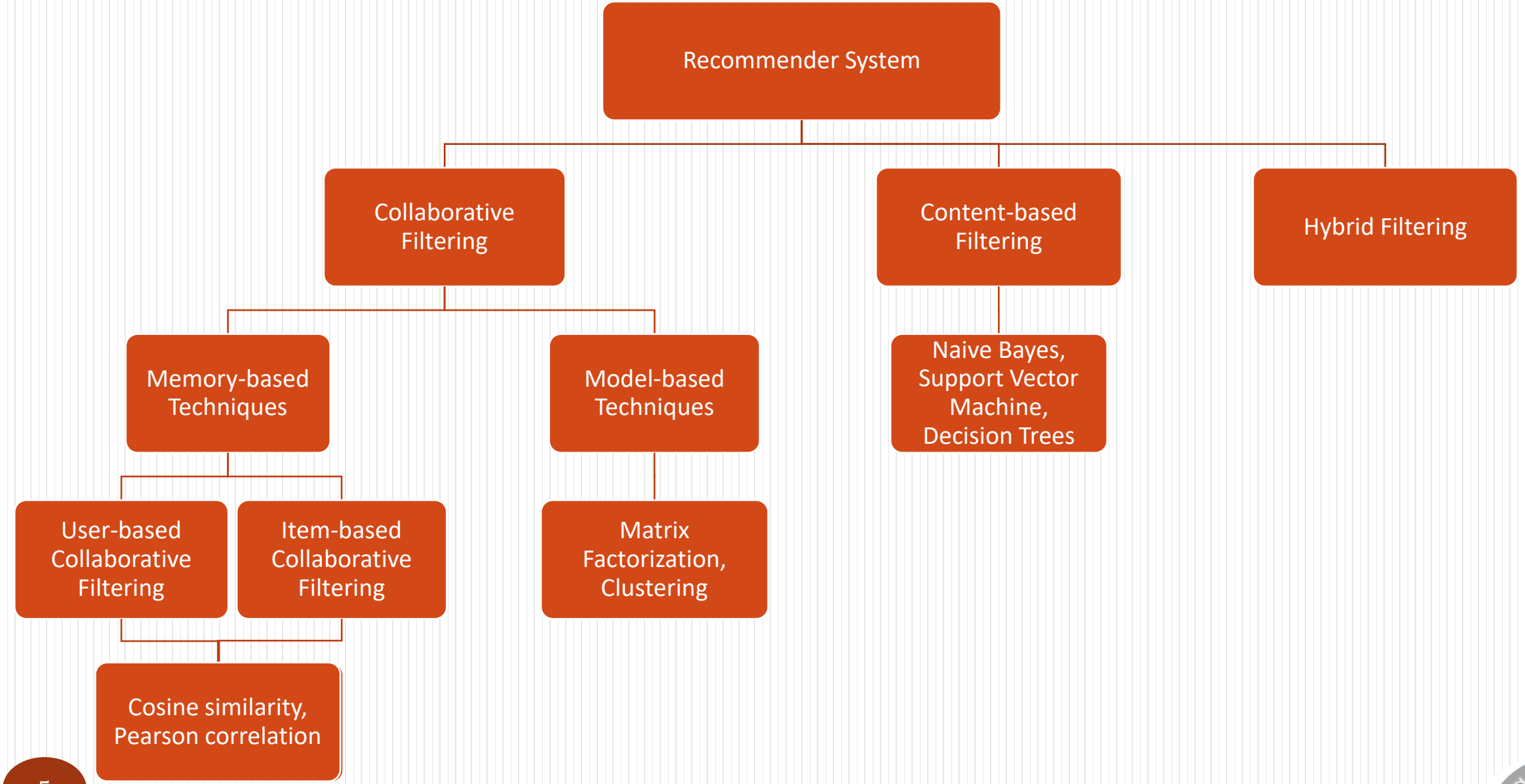
- The demand for finding more efficient techniques to generate more useful recommendations becomes an urgent
- Many recommendations become useless if there is a delay in showing them to the user.
- When it comes to the speed of recommendation we find a research gap in this area.

Goal of the paper

- In this research we focus on speeding up the process of generating the recommendations without impacting the accuracy.

Main Contributions

- Propose a novel recommender system that combines two approaches in order to speed up the recommendation systems.
- It is based on factorization machines and association rules (FMAR).
- Propose a method that uses Apriori algorithm to generate association rules.
- Use association rules to propose short-listed set of items for every user.
- Employ factorization machines model to predict missing user preferences for the short-listed set of items.
- Evaluate the top-N produced predictions.


Recommender System
Collaborative Filtering
Content-based Filtering
Hybrid Filtering
Memory-based Techniques
Model-based Techniques
Naive Bayes, Support Vector Machine, Decision Trees
User-based Collaborative Filtering
Item-based Collaborative Filtering
Matrix Factorization, Clustering
Cosine similarity, Pearson correlation

Why Factorization Machines

- Factorization Machines can be considered as an extension of linear model that additionally incorporate information about features interactions.
- Factorization machines can be considered as an equivalent to polynomial regression models.
- Factorization machines replace these interactions by factorized interaction parameters (latent vectors).
- Factorization machines is computationally very efficient.
- Factorization machines is not prone to overfitting.
- Factorization machines can generalize the interaction between two variables even if there is no interaction between them, while polynomial regression can't do that.

Why Factorization Machines

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{n} w_i x_i$$

$$\widehat{y}(x) = w_0 + \sum_{i=1}^{N} w_i x_i + \sum_{i=1}^{N} \sum_{i=1+1}^{N} w_{ij} x_i x_j$$

Poly2: $w_{i,j}$

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^{n} w_i\, x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle\, x_i\, x_j$$

Linear Regression — Feature Interactions

- Factorization Machines Trick!
  - It finds latent vectors for each feature and compute the weight of feature interactions as a dot product of those vectors.

Why Association Rules

- The basic idea of association rules is to uncover all relationships between elements.
- Association Rules Vs Collaborative Filtering.
  - Association Rules: all transactions are studied as one group.
  - Collaborative Filtering: transaction are grouped by userID.
- An association rule consists of antecedent and consequent.
  - Antecedent -> Consequent, for example, {x,y,…} -> {z}
- Various metrics exist to identify the most important rules and calculate their strength, such as support, confidence and lift.

Support

- This measure gives an idea of how frequent an itemset is in all the transactions.
- Mathematically, support is the fraction of the total number of transactions in which the itemset occurs.
- Value of support helps us identify the rules worth considering for further analysis

$$Support(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Total\ number\ of\ transactions}$$

$$\text{supp}(A \Rightarrow 0) = P(A \wedge 0) = P(A)P(0 \mid A) = P(0)P(A \mid 0)$$
$$\text{supp}(B \Rightarrow 1) = P(B \wedge 1) = P(B)P(1 \mid B) = P(1)P(B \mid 1)$$

Confidence

- This measure is an indication of how often the rule has been found to be true.
- Confidence (X→Y), with respect to a set of transactions T, is the proportion of the transactions that contain X which also contain Y.
- Mathematically, confidence is the conditional probability of occurrence of consequent given the antecedent.

$$Confidence(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Transactions\ containing\ X}$$

$$\mathrm{conf}(X \Rightarrow Y) = \mathrm{supp}(X \cup Y)/\mathrm{supp}(X)$$

$$\mathrm{conf}(A \Rightarrow 0) = P(0 \mid A)$$
$$\mathrm{conf}(B \Rightarrow 1) = P(1 \mid B)$$

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Lift

- This measure is used to discover (exclude) the weak rules that have high confidence.
- Mathematically, lift can be calculated by dividing the confidence by the unconditional probability of the consequent

$$Lift(\{X\} \rightarrow \{Y\}) = \frac{(Transactions\ containing\ both\ X\ and\ Y)/(Transactions\ containing\ X)}{Fraction\ of\ transactions\ containing\ Y}$$

$$\text{lift}(A \Rightarrow 0) = \frac{P(0 \mid A)}{P(0)} = \frac{P(A \wedge 0)}{P(A)P(0)}$$
$$\text{lift}(B \Rightarrow 1) = \frac{P(1 \mid B)}{P(1)} = \frac{P(B \wedge 1)}{P(B)P(1)}$$

$$\text{lift}(X \Rightarrow Y) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X) \times \text{supp}(Y)}$$

FMAR Model

- A hybrid model that utilizes factorization machines and Apriori algorithm to minimize the prediction latency of recommender system.
- Steps:
  - Use Apriori algorithm to create a set of association rules.
  - Use these rules to create users' profile which recommends a set of items for every user.
  - In prediction time: users' profile is used to filter the items that are passed to recommendation engine.

How Association Rules Are Generated

- Create a dataset of only favorable reviews.
- Find frequent item-sets, which contain movies that have been highly rated together (by same user) more than min_support threshold.
- Extract all possible association rules from frequent item-sets.
- Compute confidence for every rule, and filter out the rules that have less than min_confidence threshold.
- Create users' profile which recommends a set of items for every user.

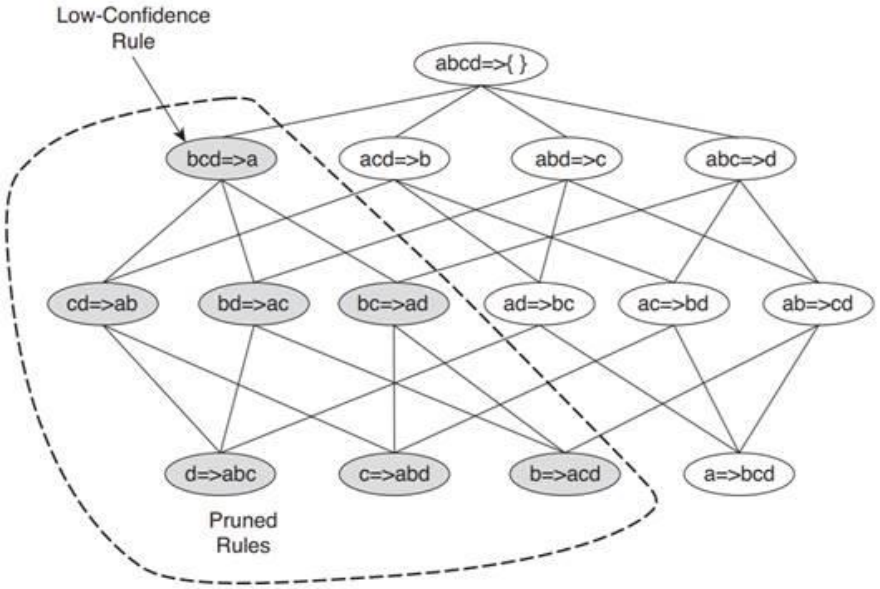
Creating Users' Profile

- For every user, find high rated items based on the rating history.
- Find all association rules that their antecedents are subset of high rated items of that user.
- Recommend all consequences of these rules to the user.

- For example:
  - User1: highly rated {2, 5, 8, 10} items.
  - {5} –> {20}, {2, 5} –> {30}, {8} –> {40}, {2, 5, 8, 10} –> {70}

Anti-Monotone Property

- If we drop out an item from an itemset, support value of new itemset generated will either be the same or will increase.
- All subsets of a frequent itemset must also be frequent.
- This property is considered while calculating support and confidence.



Efficiency of FMAR

- When we try to find new frequent item-set, there is no need to check all items and calculate support for every new combination.
- Assume we need to find new frequent item-set based on {A,B,C}, if item D does not form frequent item-set with {A,B}, then it will not form frequent item-set with {A,B,C}.
- Main idea here is to find intersections for all items that produce frequent item-sets with {A,B}, {A,C}, {B,C}.

Evaluation Methodology

- Dataset: MovieLens 100K dataset is a stable benchmark dataset which consists of 1682 movies and 943 users who provide 100,000 ratings on a scale of 1 to 5.
- It is important to note that in this paper, we are not concerned about users' demographics and contextual information since the association rules are created based only on rating history.
- In order to generate predictions, we employ a factorization machines model which is created using the publicly available software tool libFM.

Parameters Selection

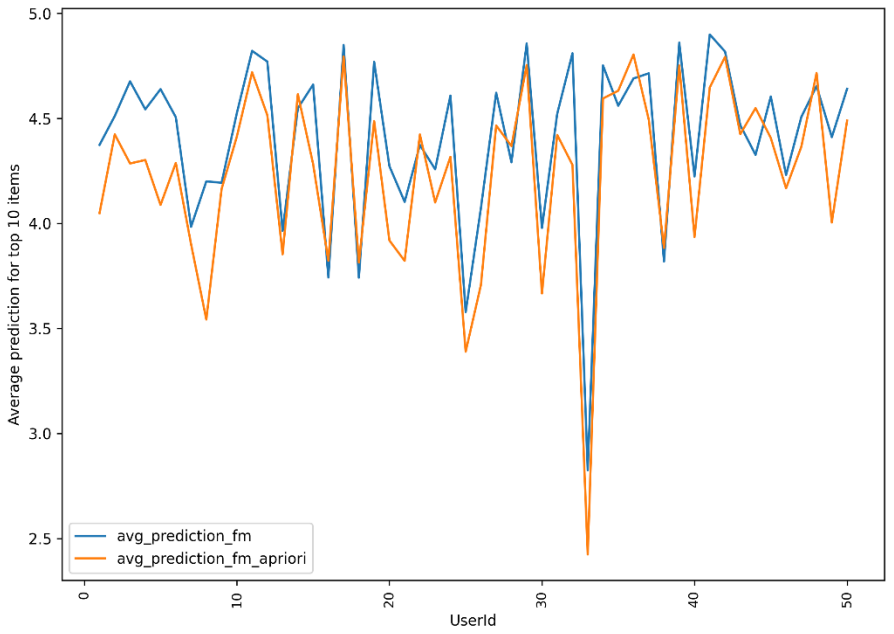
- Several experiments are conducted in order to select the appropriate values of parameters in previous algorithm, such as min_support and min_confidence.
- Multiple factors are taken into consideration while selecting those values, including accuracy, number of generated rules, and memory consumption.

Performance Comparison and Analysis

- Several methods are used to compare between FMAR and FM recommender system.
- In every method, we create two sets of items for every user:
  - Original set which contains all items that are not rated before by the user.
  - Short-listed set which is created by filtering the original set using the association rules.
  - We pass both sets to factorization machines model to generate predictions.
  - We arrange the results in descending order based on the predicted values, and find the top 10 items for every set.
  - Compare the results.

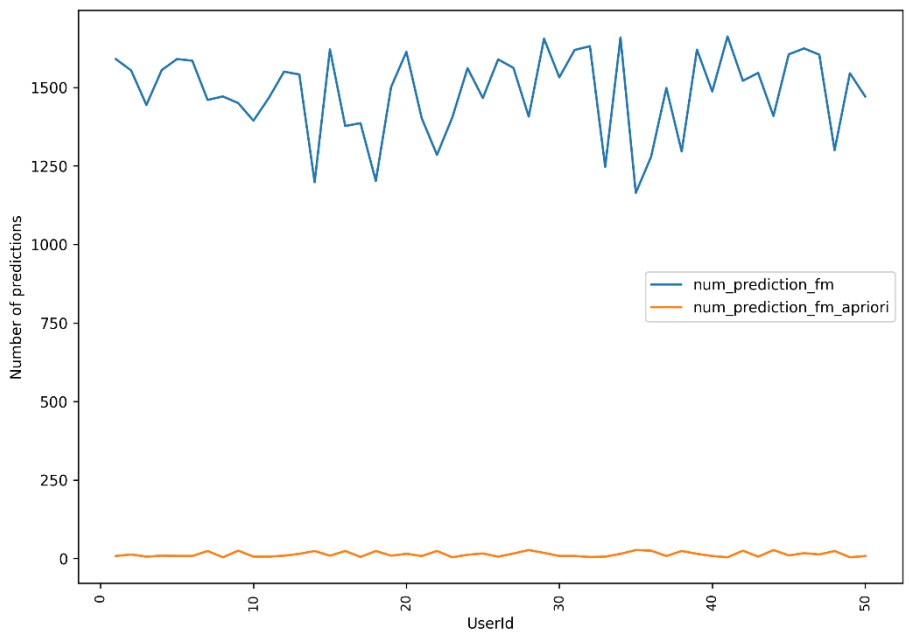
First Method

- We calculate the average of prediction for the top 10 items which are generated in both recommendation engines.
- The main goal of this approach is to make sure that accuracy of predicted items is not highly impacted after filtering the items using the association rules.



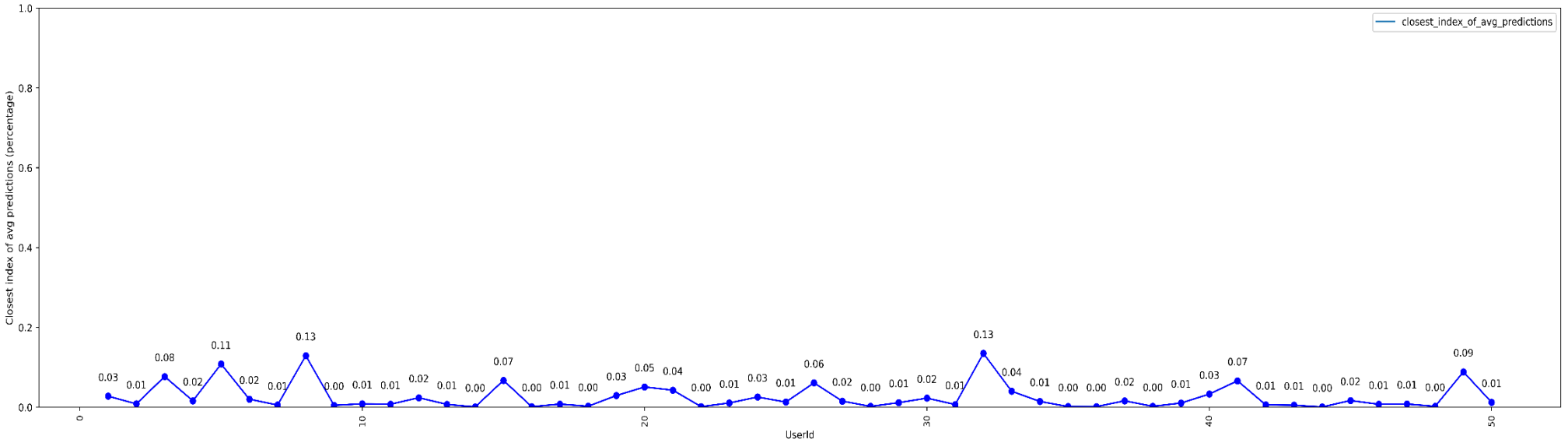
Second Method

- We compare between the number of items in original and short-listed sets.
- The main idea here is to show how many items we have to pass to factorization machines model before and after using the association rules.



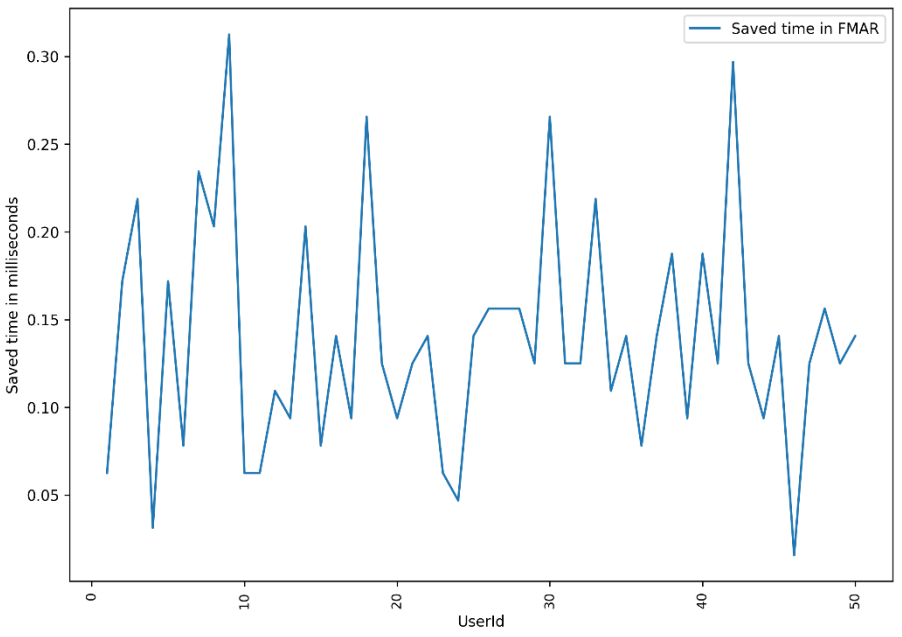
Third Method

- We calculate the average of predictions for the top 10 items in the short-listed set. Then, we find the closest index of this average value in the original set.
- The original set here is sorted in descending order based on the predicted value for each item.



Fourth Method

- We compare between FMAR and classical recommender system in terms of the elapsed time necessary to make a prediction.
- The saved time can be increased based the length of original set of items. So, we are expecting to save more time when we use larger dataset



Future Work

- Incorporate more information in the process of producing the association rules , such as users' demographics, items' characteristics and contextual information.
- Creating a web interface where FMAR is used to generate recommendations for users to evaluate FMAR using different settings:
  - Employing more advanced algorithms to generate association rules, such as SETM and FP-growth (Frequent pattern).
  - Selecting different set of datasets with different sizes.
- Take into account the dynamics of evolving the association rules by periodically updating them based on recent changes in rating history.

Future Work

- Utilize the generated association rules to solve the cold-start problem in recommendation systems.
- Use distributed stream processing engines, like Apache Flink, to examine parallel implementations of FMAR, where the process of extracting the rules and generating the recommendations is scalable to infinite streams or large-scale datasets.

Thank you

Q&A

eyad.kannout@mimuw.edu.pl