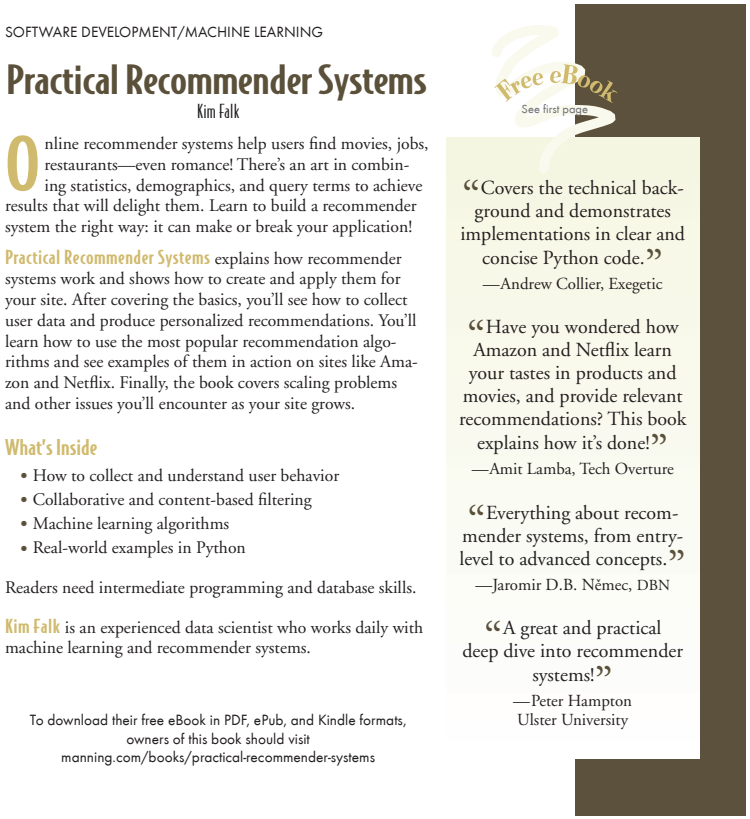
Information Technology Fundamentals

Mohammad Hossein Manshaei
manshaei@gmail.com

Web Systems: Collaborative Filtering Recommendation : Model-Based Module 5: Part 6

Main Reference

Kim Falk. ***Practical Recommender Systems,*** Manning Publication Co., 2019



SOFTWARE DEVELOPMENT/MACHINE LEARNING

Practical Recommender Systems

Kim Falk

Online recommender systems help users find movies, jobs, restaurants—even romance! There's an art in combining statistics, demographics, and query terms to achieve results that will delight them. Learn to build a recommender system the right way: it can make or break your application!

Practical Recommender Systems explains how recommender systems work and shows how to create and apply them for your site. After covering the basics, you'll see how to collect user data and produce personalized recommendations. You'll learn how to use the most popular recommendation algorithms and see examples of them in action on sites like Amazon and Netflix. Finally, the book covers scaling problems and other issues you'll encounter as your site grows.

What's Inside

- How to collect and understand user behavior
- Collaborative and content-based filtering
- Machine learning algorithms
- Real-world examples in Python

Readers need intermediate programming and database skills.

Kim Falk is an experienced data scientist who works daily with machine learning and recommender systems.

To download their free eBook in PDF, ePub, and Kindle formats, owners of this book should visit manning.com/books/practical-recommender-systems

Free eBook

See first page

"Covers the technical background and demonstrates implementations in clear and concise Python code."
—Andrew Collier, Exegetic

"Have you wondered how Amazon and Netflix learn your tastes in products and movies, and provide relevant recommendations? This book explains how it's done!"
—Amit Lamba, Tech Overture

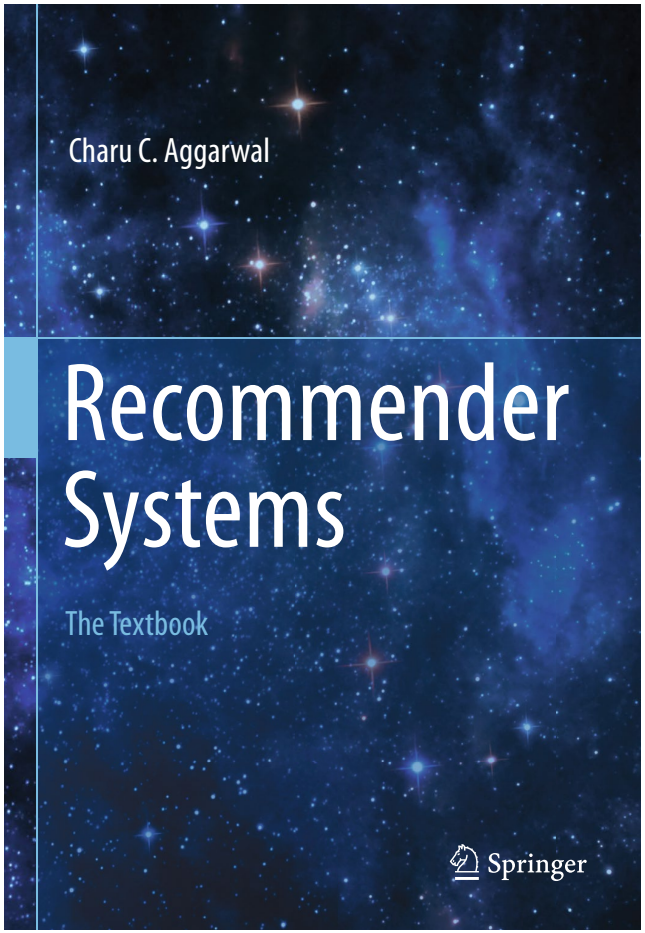
"Everything about recommender systems, from entry-level to advanced concepts."
—Jaromir D.B. Němec, DBN

"A great and practical deep dive into recommender systems!"
—Peter Hampton
Ulster University

Main Reference

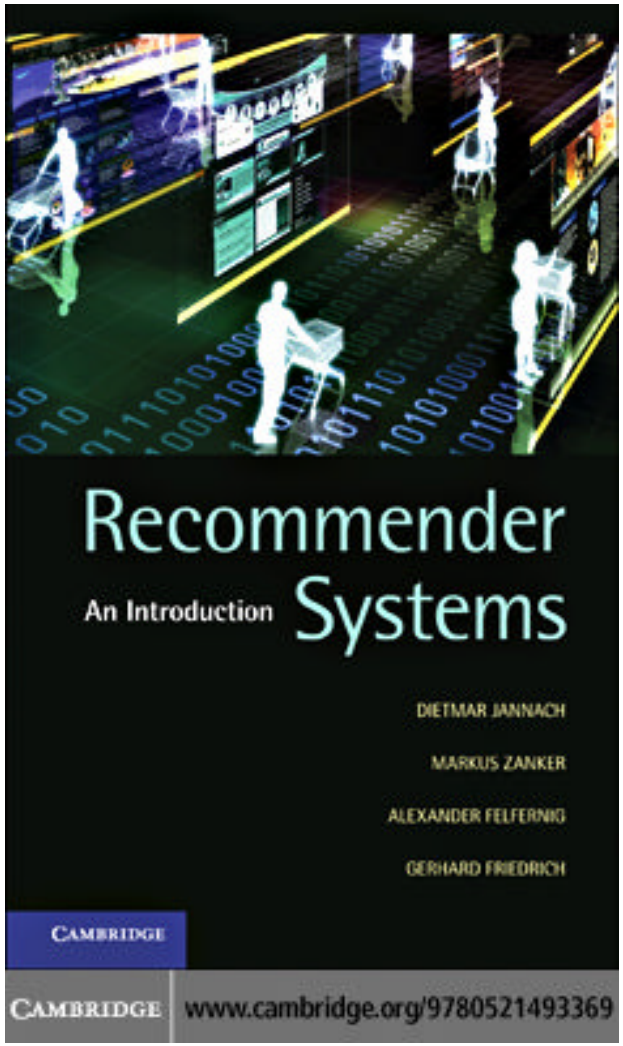
Charu C. Aggarwal. ***Recommender Systems, The Textbook.*** Springer, 2016

Chapter 1: An Introduction to Recommender Systems



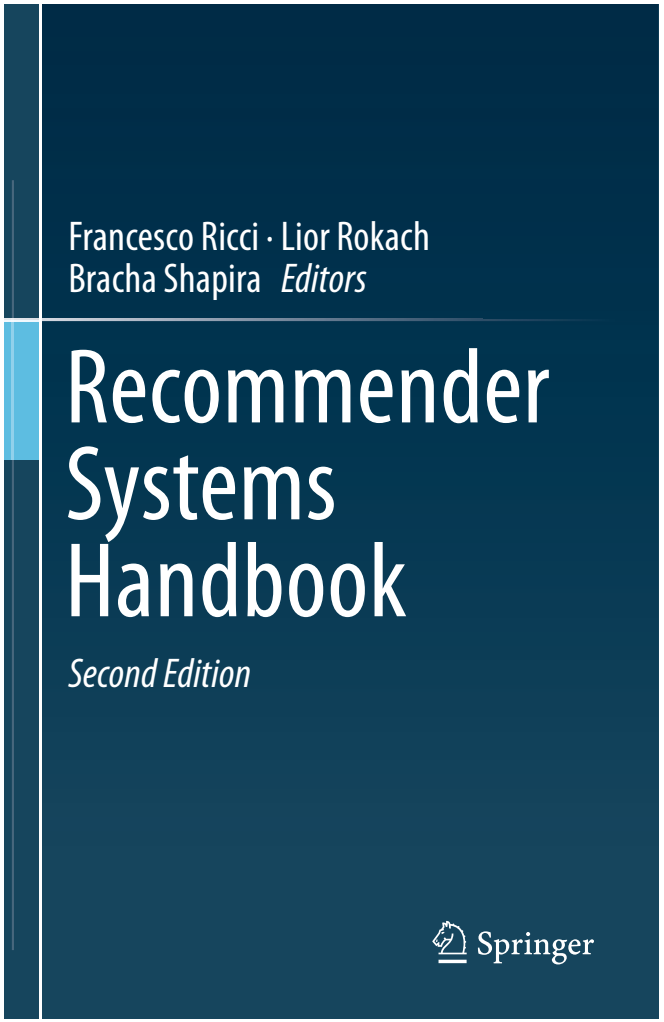
Main Reference

D. Jannach, M. Zanker, A. Felfering, G. Friedrich. ***Recommender Systems: An Introduction.*** Cambridge University Press, 2011



Main Reference

F. Ricci,L. Rokach, B. Shapira. ***Recommender Systems Handbook.*** Springer, 2015



Contents

- Collaborative Filtering Recommendation
  - ✓ Memory-Based:
    - User-Based and Item-Based
  - ✓ Model-Based
    - Matrix Factorization/latent factor models and Association Rules
- Content-Based Recommendation
- Knowledge Based Recommendation
- Demographic Recommendation
- Hybrid Recommendation Systems
- RS Evaluation
  - ✓ Online and Offline Evaluations
  - ✓ Design Issues and RS Properties

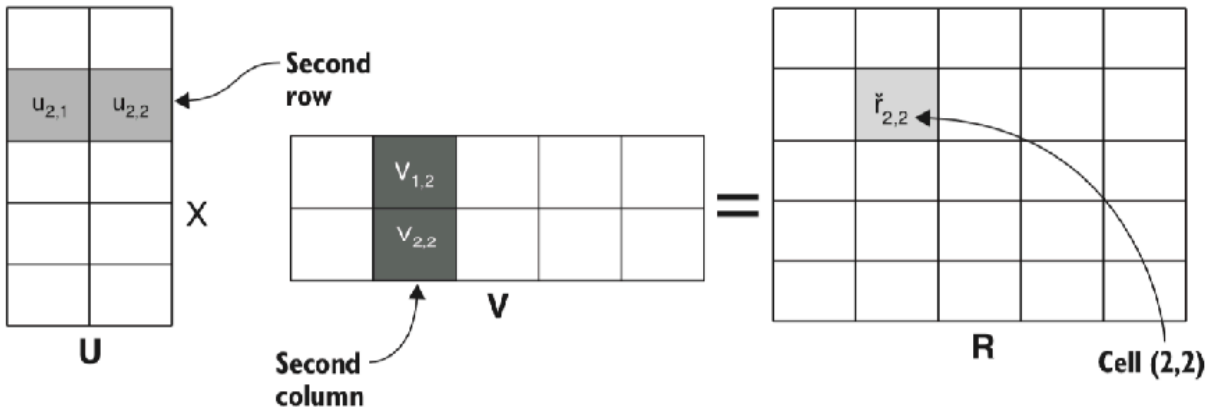
Netflix Prize

- A competition launched by Netflix in 2006
- Aimed to improve the accuracy of their recommendation algorithm, Cinematch.
- Participants were challenged to create a recommendation system that could beat Cinematch by **at least 10%.**
- **The prize: $1 million**
- Participants employed advanced machine learning and data mining techniques
- **Winner:** In 2009, BellKor's Pragmatic Chaos won the prize (achieved the required improvement in accuracy)
- The Netflix Prize significantly influenced the field of machine learning and data science

Matrix Factorization

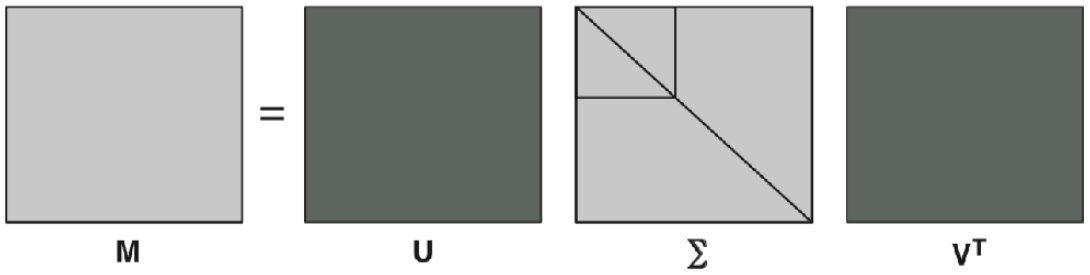
- To do the factorization, you need to somehow insert values in the U and V matrixes in such a way that UV is as close to R as possible.
- Let's start simple and say, for example, you want to fit the cells in the second row, second column from the upper left corner of R.

$$\hat{r}_{2,2} = u_2 \times v_2 = u_{2,1} v_{1,2} + u_{2,2} v_{2,2}$$



Constructing the Factorization Using Singular Value Decomposition (SVD)

- Matrix of Factorization:
  - M—A matrix you want to decompose; in your case, it's the rating matrix.
  - U—User feature matrix.
  - Σ—Weights diagonal.
  - V^T—Item feature matrix.
- When using the SVD algorithm this $\sum$ will always be a diagonal matrix.



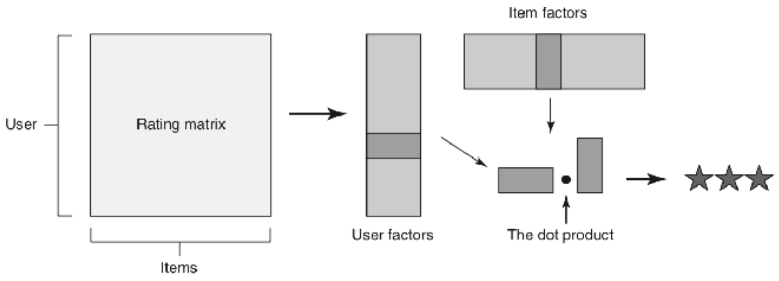
Example for SVD-based Recommendation

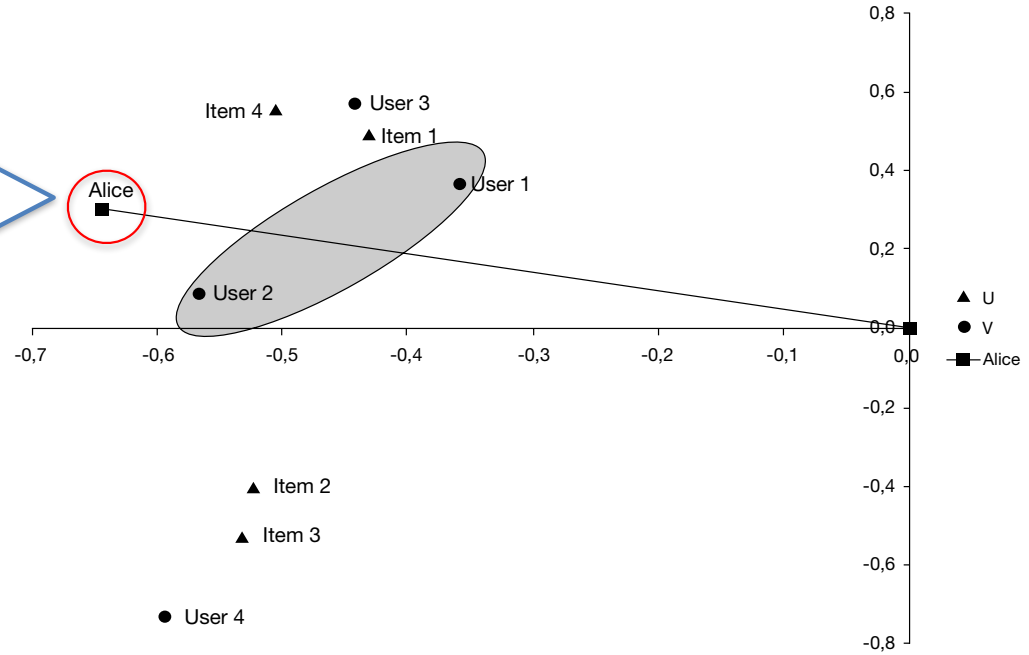
- The **SVD** theorem states that a given matrix M can be decomposed into a product of three matrices as follows, where U and V are called left and right singular vectors and the values of the diagonal of $\sum$ are called the singular values.

$$M = U \Sigma V^T$$

	User1	User2	User3	User4
Item1	3	4	3	1
Item2	1	3	2	6
Item3	2	4	1	5
Item4	3	3	5	2

SVD-based projection in two-dimensional space.





U_2		V_2		Σ_2	
−0.4312452	0.4931501	−0.3593326	0.36767659	12.2215	0
−0.5327375	−0.5305257	−0.5675075	0.08799758	0	4.9282
−0.5237456	−0.4052007	−0.4428526	0.56862492		
−0.5058743	0.5578152	−0.5938829	−0.73057242		

First two columns of decomposed matrix and singular values.

How to Predict Ratings Using the Factors

- Because our goal is to make a prediction for Alice's ratings, we must first find out where Alice would be positioned in this two-dimensional space.
- To find out Alice's datapoint, multiply Alice's rating vector [5, 3, 4, 4] by the two-column subset of U and the inverse of the two-column singular value matrix Σ

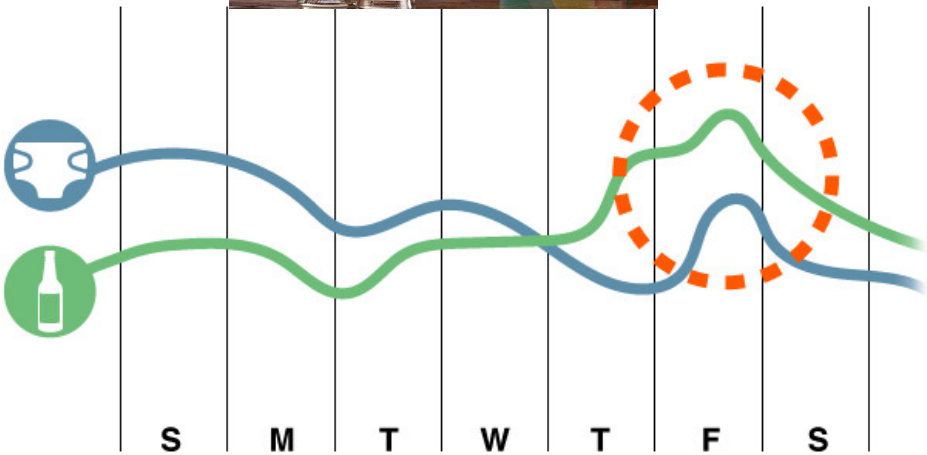
$$Alice_{2D} = Alice \times U_2 \times \Sigma_2^{-1} = [-0.64, 0.30]$$

Contents

- Collaborative Filtering Recommendation
  - ✓ Memory-Based:
    - User-Based and Item-Based
  - ✓ Model-Based
    - Matrix Factorization/latent factor models and Association Rules
- Content-Based Recommendation
- Knowledge Based Recommendation
- Demographic Recommendation
- Hybrid Recommendation Systems
- RS Evaluation
  - ✓ Online and Offline Evaluations
  - ✓ Design Issues and RS Properties

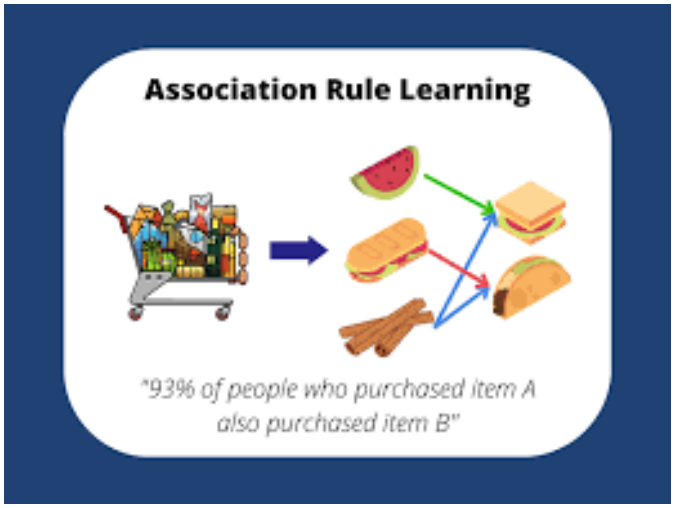
Association Rule

- **History:** In the 1990s, the story of Beer & Diapers were very famous @ Walmart
-



Association Rule

- A common technique used to identify rule like relationship patterns in large-scale sales transactions
- A typical application is the detection of pairs or groups of products in a supermarket that are often purchased together
- The goal of rule-mining algorithms is to automatically detect rules and calculate a measure of quality for those rules



Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$	2/5	2/4	5/6
$A \Rightarrow C$	2/5	2/3	5/6
$B \& C \Rightarrow D$	1/5	1/3	5/9

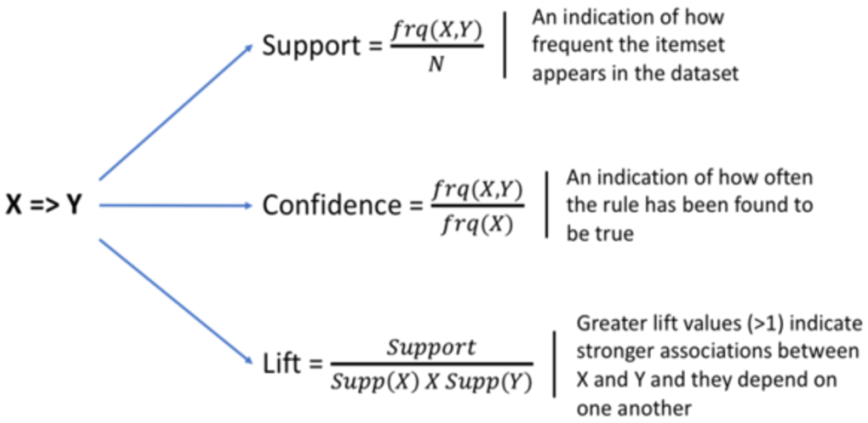
Association Rule

- The goal will be to automatically detect rules such as "*If user X liked both item1 and item2, then X will most probably also like item5.*"
- Recommendations for the active user can be made by evaluating which of the detected rules apply – in the example, checking whether the user liked item1 and item2 – and then generating a ranked list of proposed items based on statistics about the co-occurrence of items in the sales transactions.

	Item1	Item2	Item3	Item4	Item5
Alice	1	0	0	0	?
User1	1	0	0	1	1
User2	1	0	1	0	1
User3	0	0	0	1	1
User4	0	1	1	0	0

Association Rule

- Standard rule-mining algorithms can be used to analyze this database and calculate a list of association rules and their corresponding confidence and support values.
- The standard measures for association rules are support, confidence, and lift:



- If lift is greater than 1 then Y will be bought with X

Association Rule

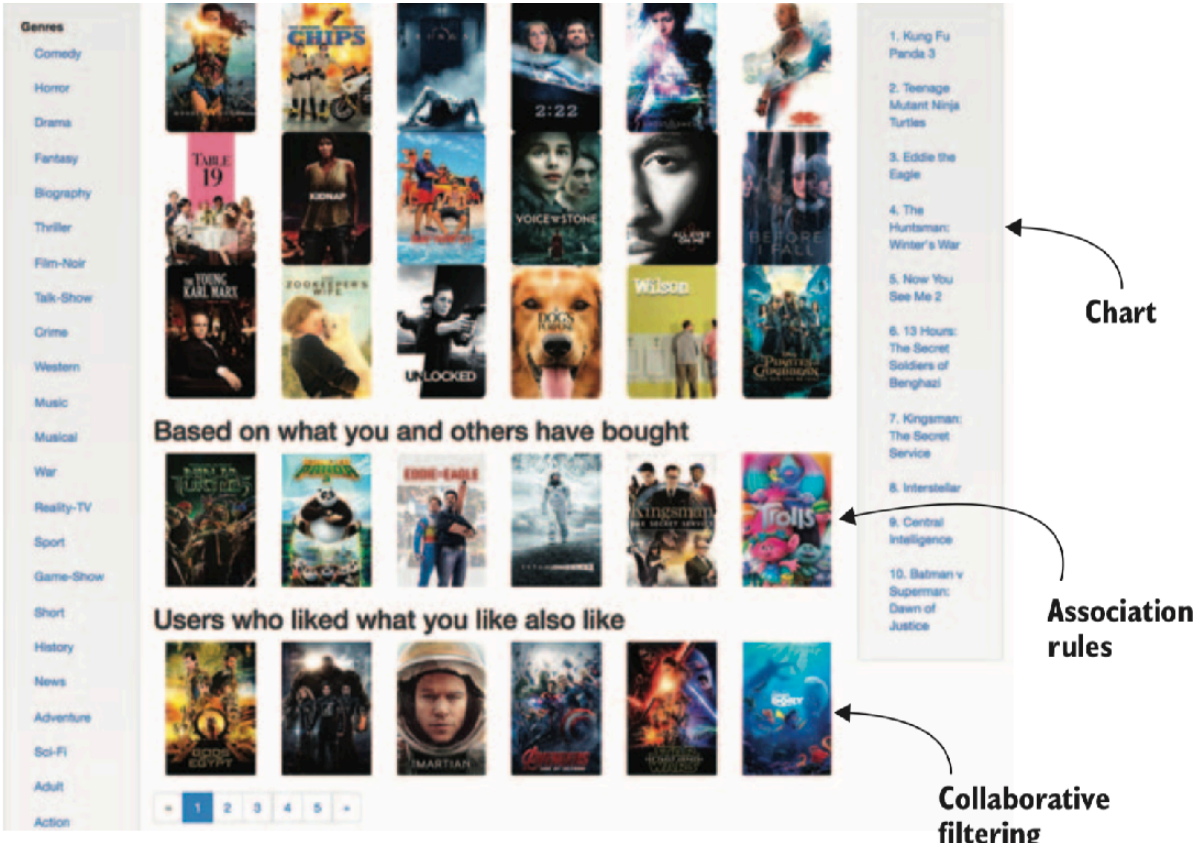
- The calculation of the set of interesting association rules with a sufficiently high value for confidence and support can be performed offline (**Apriori with Support Threshold** or FP-Growth)
- At run time, recommendations for user Alice can be efficiently computed based on the following scheme:

1. Determine the set of $X \Rightarrow Y$ association rules that are relevant for Alice – Where Alice has bought (or liked) all elements from X
2. Compute the union of items appearing in the consequent Y of these association rules that have not been purchased by Alice.
3. Sort the products according to the confidence of the rule that predicted them. If multiple rules suggested one product, take the rule with the highest confidence.
4. Return the first N elements of this ordered list as a recommendation.

Association Rule

- To determine whether an item will be liked by Alice, we can check, for each item, whether the rule "fires" for the item (if User1 liked it and User2 disliked it)
- Based on confidence and support values of these rules, an overall score can be computed for each item as follows:

$$score_{item_i} = \sum_{rules\ recommending\ item_i} (support_{rule} * confidence_{rule})$$



Examples of Collaborative learning and Association Rules