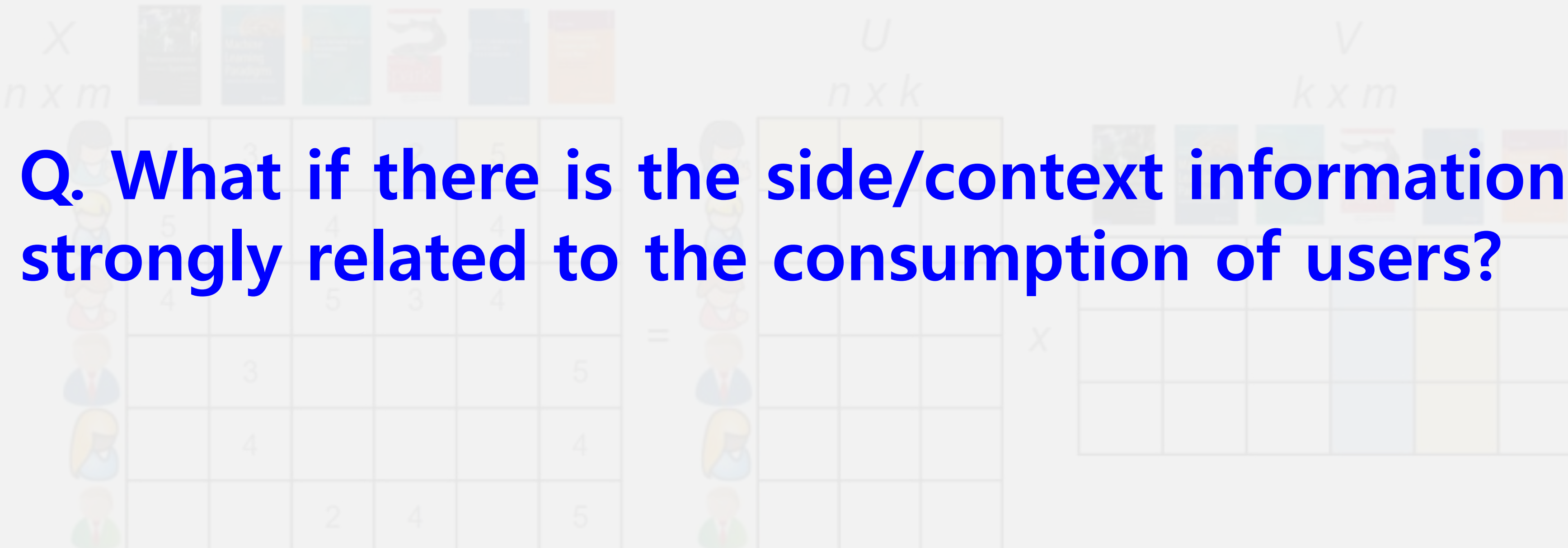
# Service Intelligence Week 2.

# [Recommender Systems for Services]

Chiehyeon Lim

2022. 9. 7

# Use of the Bag-of-Words-Form Transactions Matrix

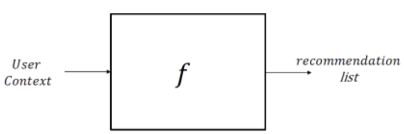



**Q. What if there is the side/context information strongly related to the consumption of users?**

# Approaches of Recommender Systems: A Data Perspective

- Recommender systems use analytic techniques to compute the value that a user will purchase one of the items; the techniques vary according to the purposes and data


$$y = f(x)$$




| | Feature 1 | Feature 2 | Feature 3 | … | Feature m-1 | Feature m |
|---|---|---|---|---|---|---|
| Transaction 1 | … | … | … | … | … | … |
| Transaction 2 | … | … | … | … | … | … |
| Transaction 3 | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| Transaction n-1 | … | … | … | … | … | … |
| Transaction n | … | … | … | … | … | … |

# Approaches of Recommender Systems: A Data Perspective

- Recommender systems use analytic techniques to compute the value that a user will purchase one of the items; the techniques vary according to the purposes and data

Matrix Factorization
Training Data

| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $u_1$ | 2 | 4 | |
| $u_2$ | | 1 | |
| $u_3$ | 3 | | 5 |

**vs.**

Factorization Machine
Training Data

| | $u_1$ | $u_2$ | $u_3$ | $i_1$ | $i_2$ | $i_3$ | $a_1$ | $a_2$ | $y$ |
|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 1 | 0 | 0 | 1 | 0 | 0 | 2.0 | 0.0 | 2 |
| $x_2$ | 1 | 0 | 0 | 0 | 1 | 0 | 1.5 | 0.5 | 4 |
| $x_3$ | 0 | 1 | 0 | 0 | 1 | 0 | 0.0 | 1.0 | 1 |
| $x_4$ | 0 | 0 | 1 | 1 | 0 | 0 | 0.3 | 0.7 | 3 |
| $x_5$ | 0 | 0 | 1 | 0 | 0 | 1 | 3.2 | 1.7 | 5 |

Users | Items | Auxiliary Features | Observed Ratings



Image reference: https://towardsdatascience.com/factorization-machines-for-item-recommendation-with-implicit-feedback-data-5655a7c749db

# Factorization Machine

Feature vector **x** / Target y

| | User | | | | Movie | | | | | Other Movies rated | | | | | Time | Last Movie rated | | | | | Target y | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | ... | TI | NH | SW | ST | ... | TI | NH | SW | ST | ... | Time | TI | NH | SW | ST | ... | | |
| $\mathbf{x}^{(1)}$ | 1 | 0 | 0 | ... | 1 | 0 | 0 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 13 | 0 | 0 | 0 | 0 | ... | 5 | $y^{(1)}$ |
| $\mathbf{x}^{(2)}$ | 1 | 0 | 0 | ... | 0 | 1 | 0 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 14 | 1 | 0 | 0 | 0 | ... | 3 | $y^{(2)}$ |
| $\mathbf{x}^{(3)}$ | 1 | 0 | 0 | ... | 0 | 0 | 1 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 16 | 0 | 1 | 0 | 0 | ... | 1 | $y^{(2)}$ |
| $\mathbf{x}^{(4)}$ | 0 | 1 | 0 | ... | 0 | 0 | 1 | 0 | ... | 0 | 0 | 0.5 | 0.5 | ... | 5 | 0 | 0 | 0 | 0 | ... | 4 | $y^{(3)}$ |
| $\mathbf{x}^{(5)}$ | 0 | 1 | 0 | ... | 0 | 0 | 0 | 1 | ... | 0 | 0 | 0.5 | 0.5 | ... | 8 | 0 | 0 | 1 | 0 | ... | 5 | $y^{(4)}$ |
| $\mathbf{x}^{(6)}$ | 0 | 0 | 1 | ... | 1 | 0 | 0 | 0 | ... | 0.5 | 0 | 0.5 | 0 | ... | 9 | 0 | 0 | 0 | 0 | ... | 1 | $y^{(5)}$ |
| $\mathbf{x}^{(7)}$ | 0 | 0 | 1 | ... | 0 | 0 | 1 | 0 | ... | 0.5 | 0 | 0.5 | 0 | ... | 12 | 1 | 0 | 0 | 0 | ... | 5 | $y^{(6)}$ |

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

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

where the model parameters that have to be estimated are:

$$w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^n, \quad \mathbf{V} \in \mathbb{R}^{n \times k} \qquad (2)$$

And $\langle \cdot, \cdot \rangle$ is the dot product of two vectors of size $k$:

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle := \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f} \qquad (3)$$

A row $\mathbf{v}_i$ within $\mathbf{V}$ describes the $i$-th variable with $k$ factors. $k \in \mathbb{N}_0^+$ is a hyperparameter that defines the dimensionality of the factorization.

If Y is **symmetric**, then it is diagonalizable, its eigenvalues are real, and its eigenvectors are orthogonal. Hence, Y has an eigendecomposition $Y = Q\Lambda Q^\top$, where the columns of Q are the eigenvectors of Y and the diagonal entries of diagonal matrix $\Lambda$ are the eigenvalues of Y.

If Y is also **positive semidefinite**, then all its eigenvalues are nonnegative, which means that we can take their square roots. Hence,

$$Y = Q\Lambda Q^\top = Q\Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}}Q^\top = \underbrace{\left(Q\Lambda^{\frac{1}{2}}\right)}_{=:V}\left(Q\Lambda^{\frac{1}{2}}\right)^\top = V^\top V$$

Note that the rows of V are the eigenvectors of Y multiplied by the square roots of the (nonnegative) eigenvalues of Y.

Service & Knowledge Lab

Reference: https://www.jefkine.com/recsys/2017/03/27/factorization-machines/
https://ieeexplore.ieee.org/abstract/document/5694074?casa_token=CV0m3FJ7U3UAAAAA:1rubmy3hLA6hvfRzAxZV4ykDx4kufMqEsGPX69_eJckG3BP05EJCh7DEkUcSmGQzc6JhEf37dX4
https://math.stackexchange.com/questions/1801403/decomposition-of-a-positive-semidefinite-matrix

# Factorization Machine

| | Feature vector $\mathbf{x}$ | | | | | | | | | | | | | | | | | | | | | | Target y | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | User | | | | Movie | | | | | Other Movies rated | | | | | Time | Last Movie rated | | | | | | | | |
| | A | B | C | ... | TI | NH | SW | ST | ... | TI | NH | SW | ST | ... | Time | TI | NH | SW | ST | ... | | | | |
| $\mathbf{x}^{(1)}$ | 1 | 0 | 0 | ... | 1 | 0 | 0 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 13 | 0 | 0 | 0 | 0 | ... | | | 5 | $y^{(1)}$ |
| $\mathbf{x}^{(2)}$ | 1 | 0 | 0 | ... | 0 | 1 | 0 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 14 | 1 | 0 | 0 | 0 | ... | | | 3 | $y^{(2)}$ |
| $\mathbf{x}^{(3)}$ | 1 | 0 | 0 | ... | 0 | 0 | 1 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 16 | 0 | 1 | 0 | 0 | ... | | | 1 | $y^{(2)}$ |
| $\mathbf{x}^{(4)}$ | 0 | 1 | 0 | ... | 0 | 0 | 1 | 0 | ... | 0 | 0 | 0.5 | 0.5 | ... | 5 | 0 | 0 | 0 | 0 | ... | | | 4 | $y^{(3)}$ |
| $\mathbf{x}^{(5)}$ | 0 | 1 | 0 | ... | 0 | 0 | 0 | 1 | ... | 0 | 0 | 0.5 | 0.5 | ... | 8 | 0 | 0 | 1 | 0 | ... | | | 5 | $y^{(4)}$ |
| $\mathbf{x}^{(6)}$ | 0 | 0 | 1 | ... | 1 | 0 | 0 | 0 | ... | 0.5 | 0 | 0.5 | 0 | ... | 9 | 0 | 0 | 0 | 0 | ... | | | 1 | $y^{(5)}$ |
| $\mathbf{x}^{(7)}$ | 0 | 0 | 1 | ... | 0 | 0 | 1 | 0 | ... | 0.5 | 0 | 0.5 | 0 | ... | 12 | 1 | 0 | 0 | 0 | ... | | | 5 | $y^{(6)}$ |

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle \mathbf{v}_i, \mathbf{v}_j\rangle x_i x_j$$
$$= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\langle \mathbf{v}_i, \mathbf{v}_j\rangle x_i x_j - \frac{1}{2}\sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{v}_i\rangle x_i x_i$$
$$= \frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{f=1}^{k} v_{i,f}\, v_{j,f}\, x_i x_j - \sum_{i=1}^{n}\sum_{f=1}^{k} v_{i,f}\, v_{i,f}\, x_i x_i\right)$$
$$= \frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n} v_{i,f}\, x_i\right)\left(\sum_{j=1}^{n} v_{j,f}\, x_j\right) - \sum_{i=1}^{n} v_{i,f}^2\, x_i^2\right)$$
$$= \frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n} v_{i,f}\, x_i\right)^2 - \sum_{i=1}^{n} v_{i,f}^2\, x_i^2\right)$$

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

where the model parameters that have to be estimated are:

$$w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^n, \quad \mathbf{V} \in \mathbb{R}^{n\times k} \qquad (2)$$

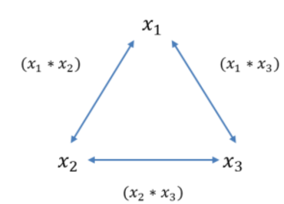
And $\langle\cdot,\cdot\rangle$ is the dot product of two vectors of size $k$:

$$\langle \mathbf{v}_i, \mathbf{v}_j\rangle := \sum_{f=1}^{k} v_{i,f}\cdot v_{j,f} \qquad (3)$$

A row $\mathbf{v}_i$ within $\mathbf{V}$ describes the $i$-th variable with $k$ factors. $k \in \mathbb{N}_0^+$ is a hyperparameter that defines the dimensionality of the factorization.

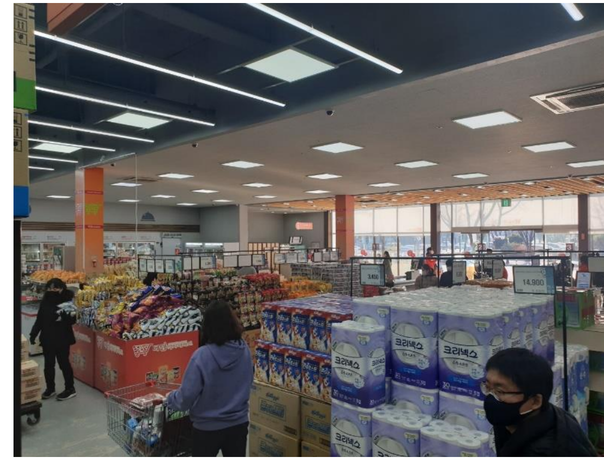
$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^{n} w_i\, x_i + \frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n} v_{i,f}\, x_i\right)^2 - \sum_{i=1}^{n} v_{i,f}^2\, x_i^2\right)$$

$$\frac{\partial}{\partial\theta}\hat{y}(\mathbf{x}) = \begin{cases} 1, & \text{if } \theta \text{ is } w_0 \\ x_i, & \text{if } \theta \text{ is } w_i \\ x_i \sum_{j=1}^{n} v_{j,f} x_j - v_{i,f} x_i^2, & \text{if } \theta \text{ is } v_{i,f} \end{cases}$$




Reference: https://www.jefkine.com/recsys/2017/03/27/factorization-machines/
https://ieeexplore.ieee.org/abstract/document/5694074?casa_token=CV0m3FJ7U3UAAAAA:1rubmy3hLA6hvfRzAxZV4ykDx4kufMqEsGPX69_eJckG3BP05EJCh7DEkUcSmGQzc6JhEf37dX4
https://math.stackexchange.com/questions/1801403/decomposition-of-a-positive-semidefinite-matrix

# Factorization Machine for Offline Contexts

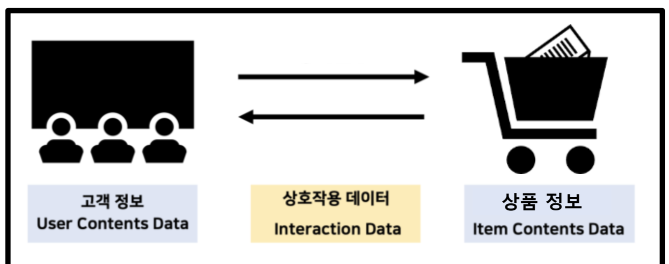
**user**

- Customer # →
- Gender, Age, … →
- Business type, … →

**item**

- Item # →
- Location, Position →
- Seasonal info →
- Repurchase info →

## Factorization Machine




→ **Recommendation**

# Assignment 2 (by 9.16 11:59 pm)

- There are two types of practice demonstrated by TAs. One approach is to use the user-item matrix and the other is to use the transaction-feature matrix. By yourself, (1) identify several (i.e., top k) recommendations using one of the two approaches with the given datasets. Of course you can try both.
- (2) Then, evaluate and interpret the recommendation outcomes quantitatively (e.g., calculate the recall, calculate the similarities between the recommended items) and qualitatively (e.g., interpret the factorization outcome, identify the characteristics of the top k recommended items). Do it all by yourself, and describe the analysis/interpretation process and outcome in detail.
- (3) Assume you need to use your recommender system for real-world service (i.e., streaming service or hypermarket service). How can you improve your recommender system to be used for the service effectively? For example, what kinds of data should you use further? How would you design a method for using/learning the data? Think beyond these examples in your own creative, unique way!
- (4) You must have your own interested or favorite service WITHOUT a recommender system (i.e., it should be different with the intelligent services discussed in the class). Discuss the requirements of original recommender system development for the service. Describe the requirements in detail.
- (5) If you would conduct a study on the recommender system development for the service, how would you conduct the research in your own creative, unique way? What kinds of data and methods are you going to collect, analyze, and learn? Describe your service intelligence development plan in detail. If possible, visualize your plan clearly (e.g., draw an image, construct a mathematical model). To facilitate your thinking, you may want to identify and review a recommender system paper related to the service you are interested or concerned.
- Upload your code and a several paragraph essay on the tasks (1)~(5) in the Blackboard.

# Matrix Factorization Practice

## Demonstrated by TA Seo

# Factorization Machine Practice

## Demonstrated by TA Shin

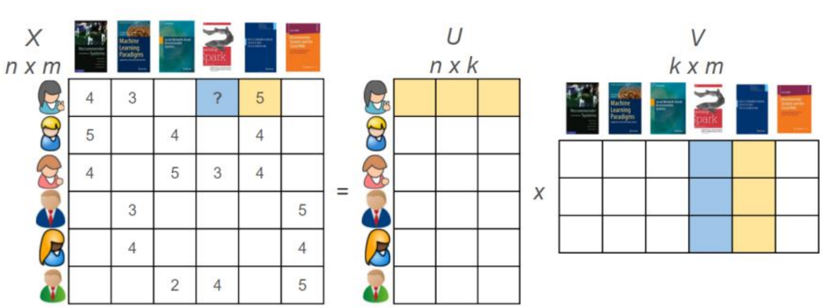
# Assignment 2 (by 9.16 11:59 pm)

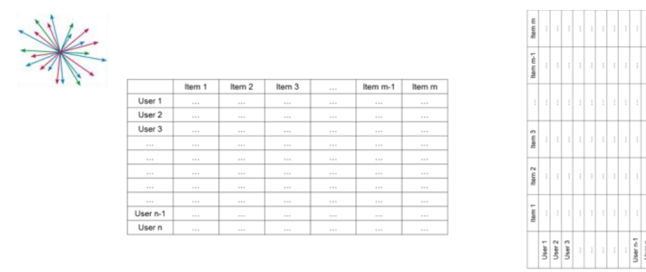
- There are two types of practice demonstrated by TAs. One approach is to use the user-item matrix and the other is to use the transaction-feature matrix. By yourself, (1) identify several (i.e., top k) recommendations using the two approaches with the provided datasets.
- (2) In each practice, evaluate and interpret the recommendation outcomes quantitatively (e.g., calculate the recall, calculate the similarities between the recommended items) and qualitatively (e.g., interpret the factorization outcome, identify the characteristics of the top k recommended items). Do it all by yourself, and describe the analysis/interpretation process and outcome in detail.
- (3) Assume you need to use your recommender system for real-world service (i.e., streaming service or hypermarket service). How can you improve your recommender system to be used for the service effectively? For example, what kinds of data should you use further? How would you design a method for using/learning the data? Think beyond these examples in your own creative, unique way!
- (4) You must have your own interested or favorite service WITHOUT a recommender system (i.e., it should be different with the intelligent services discussed in the class). Discuss the requirements of original recommender system development for the service. Describe the requirements in detail.
- (5) If you would conduct a study on the recommender system development for the service, how would you conduct the research in your own creative, unique way? What kinds of data and methods are you going to collect, analyze, and learn? Describe your service intelligence development plan in detail. If possible, visualize your plan clearly (e.g., draw an image, construct a mathematical model). To facilitate your thinking, you may want to identify and review a recommender system paper related to the service you are interested or concerned.
- Upload your code and a several paragraph essay on the tasks (1)~(5) in the Blackboard.

# Concluding Remarks

# Approaches of Collaborative Filtering: Emergence of Deep Learning

- Recommender systems use analytic techniques to compute the value that a user would purchase one of the items; the techniques vary according to the purposes and data
  - Collaborative filtering
    - Uses an information filtering technique based on the user's previous evaluation of items or history of previous purchases




- Euclidean distance
- Cosine similarity
- Jaccard coefficient
- Pearson correlation coefficient
- …

$$\mathrm{Sim}(u_a, u_b) = \frac{\sum_{i=1}^{i'} (r_{u_{a,i}})(r_{u_{b,i}})}{\sqrt{\sum_{i=1}^{n} (r_{u_{a,i}})}\sqrt{\sum_{i=1}^{n} (r_{u_{b,i}})}}$$

$$\mathrm{Sim}(u_a, u_b) = \frac{|I_{u_a}|\cap|I_{u_b}|}{|I_{u_a}|\cup|I_{u_b}|}$$

$$\mathrm{Sim}(u_a, u_b) = \frac{\sum_{i=1}^{i'} (r_{u_{a,i}} - r_{\overline{u_a}})(r_{u_{b,i}} - r_{\overline{u_b}})}{\sqrt{\sum_{i=1}^{i'} (r_{u_{a,i}} - r_{\overline{u_a}})^2} \cdot \sqrt{\sum_{i=1}^{i'} (r_{\overline{u_{b,i}}} - r_{\overline{u_b}})^2}}$$

# Approaches of Collaborative Filtering: Emergence of Deep Learning

- Deep-learning-based nonlinearity consideration complements the traditional approaches

## Neural Collaborative Filtering*


Xiangnan He, National University of Singapore, Singapore, xiangnanhe@gmail.com
Lizi Liao, National University of Singapore, Singapore, liaolizi.llz@gmail.com
Hanwang Zhang, Columbia University, USA, hanwangzhang@gmail.com
Liqiang Nie, Shandong University, China, nieliqiang@gmail.com
Xia Hu, Texas A&M University, USA, hu@cse.tamu.edu
Tat-Seng Chua, National University of Singapore, Singapore, dcscts@nus.edu.sg



### ABSTRACT

In recent years, deep neural networks have yielded immense success on speech recognition, computer vision and natural language processing. However, the exploration of deep neural networks on recommender systems has received relatively less scrutiny. In this work, we strive to develop techniques based on neural networks to tackle the key problem in recommendation — collaborative filtering — on the basis of implicit feedback.

Although some recent work has employed deep learning for recommendation, they primarily used it to model auxiliary information, such as textual descriptions of items and acoustic features of musics. When it comes to model the key factor in collaborative filtering — the interaction between user and item features, they still resorted to matrix factorization and applied an inner product on the latent features of users and items.

By replacing the inner product with a neural architecture that can learn an arbitrary function from data, we present a general framework named NCF, short for *Neural network-based Collaborative Filtering*. NCF is generic and can express and generalize matrix factorization under its framework. To supercharge NCF modelling with non-linearities, we propose to leverage a multi-layer perceptron to learn the user–item interaction function. Extensive experiments on two real-world datasets show significant improvements of our proposed NCF framework over the state-of-the-art methods. Empirical evidence shows that using deeper layers of neural networks offers better recommendation performance.



### Keywords

Collaborative Filtering, Neural Networks, Deep Learning, Matrix Factorization, Implicit Feedback

*NExT research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@SG Funding Initiative.




*WWW 2017*, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052569

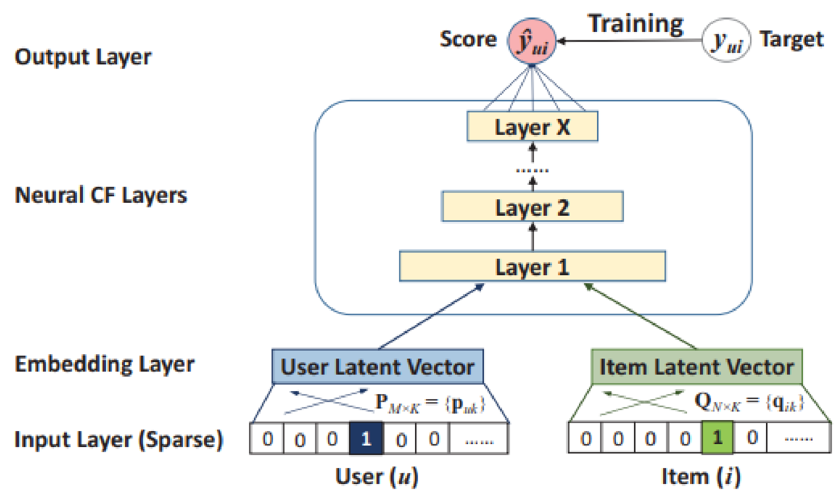

### 1. INTRODUCTION

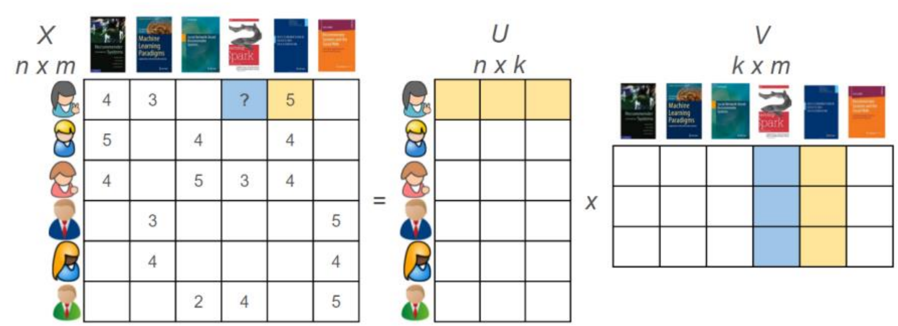
In the era of information explosion, recommender systems play a pivotal role in alleviating information overload, having been widely adopted by many online services, including E-commerce, online news and social media sites. The key to a personalized recommender system is in modelling users' preference on items based on their past interactions (*e.g.*, ratings and clicks), known as collaborative filtering [31, 46]. Among the various collaborative filtering techniques, matrix factorization (MF) [14, 21] is the most popular one, which projects users and items into a shared latent space, using a vector of latent features to represent a user or an item. Thereafter a user's interaction on an item is modelled as the inner product of their latent vectors.

Popularized by the Netflix Prize, MF has become the *de facto* approach to latent factor model-based recommendation. Much research effort has been devoted to enhancing MF, such as integrating it with neighbor-based models [21], combining it with topic models of item content [38], and extending it to factorization machines [26] for a generic modelling of features. Despite the effectiveness of MF for collaborative filtering, it is well-known that its performance can be hindered by the simple choice of the interaction function — inner product. For example, for the task of rating prediction on explicit feedback, it is well known that the performance of the MF model can be improved by incorporating user and item bias terms into the interaction function[1]. While it seems to be just a trivial tweak for the inner product operator [14], it points to the positive effect of designing a better, dedicated interaction function for modelling the latent feature interactions between users and items. The inner product, which simply combines the multiplication of latent features linearly, may not be sufficient to capture the complex structure of user interaction data.

This paper explores the use of deep neural networks for learning the interaction function from data, rather than a handcraft that has been done by many previous work [18, 21]. The neural network has been proven to be capable of approximating any continuous function [17], and more recently deep neural networks (DNNs) have been found to be effective in several domains, ranging from computer vision, speech recognition, to text processing [5, 10, 15, 47]. However, there is relatively little work on employing DNNs for recommendation in contrast to the vast amount of literature

[1] http://alex.smola.org/teaching/berkeley2012/slides/8_Recommender.pdf



# Approaches of Content-based Filtering: Emergence of Deep Learning

- Recommender systems use analytic techniques to compute the value that a user would purchase one of the items; the techniques vary according to the purposes and data
  - Content-based filtering
    - Analyzes a set of documents (of the items in question) rated by an individual user and uses the contents of the items, as well as the provided ratings, to infer a user profile that can be used to recommend additional items of interest

| | Feature 1 | Feature 2 | Feature 3 | … | Feature m-1 | Feature m |
|---|---|---|---|---|---|---|
| Item 1 | … | … | … | … | … | … |
| Item 2 | … | … | … | … | … | … |
| Item 3 | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| Item n-1 | … | … | … | … | … | … |
| Item n | … | … | … | … | … | … |

# Approaches of Collaborative Filtering: Emergence of Deep Learning

- Deep-learning-based representation/embedding complements the traditional approaches

**ITEM2VEC: NEURAL ITEM EMBEDDING FOR COLLABORATIVE FILTERING**

*Oren Barkan^* and Noam Koenigstein**

^Tel Aviv University
*Microsoft

**ABSTRACT**

Many Collaborative Filtering (CF) algorithms are item-based in the sense that they analyze item-item relations in order to produce item similarities. Recently, several works in the field of Natural Language Processing (NLP) suggested to learn a latent representation of words using neural embedding algorithms. Among them, the Skip-gram with Negative Sampling (SGNS), also known as word2vec, was shown to provide state-of-the-art results on various linguistics tasks. In this paper, we show that item-based CF can be cast in the same framework of neural word embedding. Inspired by SGNS, we describe a method we name item2vec for item-based CF that produces embedding for items in a latent space. The method is capable of inferring item-item relations even when user information is not available. We present experimental results that demonstrate the effectiveness of the item2vec method and show it is competitive with SVD.

***Index terms*** – skip-gram, word2vec, neural word embedding, collaborative filtering, item similarity, recommender systems, market basket analysis, item-item collaborative filtering, item recommendations.

People also like

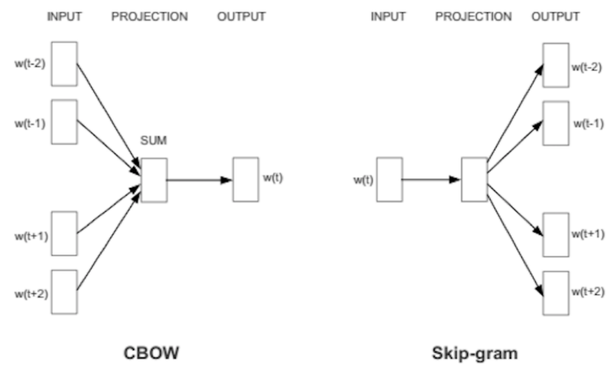
**Fig. 1.** Recommendations in Windows 10 Store based on similar items to Need For Speed.

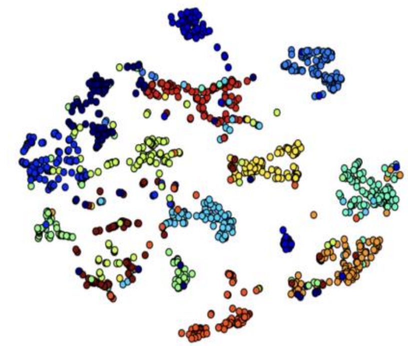
**1. INTRODUCTION AND RELATED WORK**

Computing item similarities is a key building block in modern recommender systems. While many recommendation algorithms are focused on learning a low dimensional embedding of users and items simultaneously [1, 2, 3], computing item similarities is an end in itself. Item similarities are extensively used by online retailers for many different recommendation tasks. This paper deals with the overlooked task of learning item similarities by embedding items in a low dimensional space.

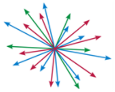
Item-based similarities are used by online retailers for recommendations based on a single item. For example, in the Windows 10 App Store, the details page of each app or game includes a list of other similar apps titled "People also like". This list can be extended to a full page recommendation list of items similar to the original app as shown in Fig. 1. Similar recommendation lists which are based merely on similarities to a single item exist in most online stores e.g., Amazon, Netflix, Google Play, iTunes store and many others.

The single item recommendations are different than the more "traditional" user-to-item recommendations because they are usually shown in the context of an explicit user interest in a specific item and in the context of an explicit user intent to purchase. Therefore, single item recommendations based on item similarities often have higher Click-Through Rates (CTR) than user-to-item recommendations and consequently responsible for a larger share of sales or revenue.





| | Item 1 | Item 2 | Item 3 | ... | Item m-1 | Item m |
|---|---|---|---|---|---|---|
| User 1 | ... | ... | ... | ... | ... | ... |
| User 2 | ... | ... | ... | ... | ... | ... |
| User 3 | ... | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... |
| User n-1 | ... | ... | ... | ... | ... | ... |
| User n | ... | ... | ... | ... | ... | ... |

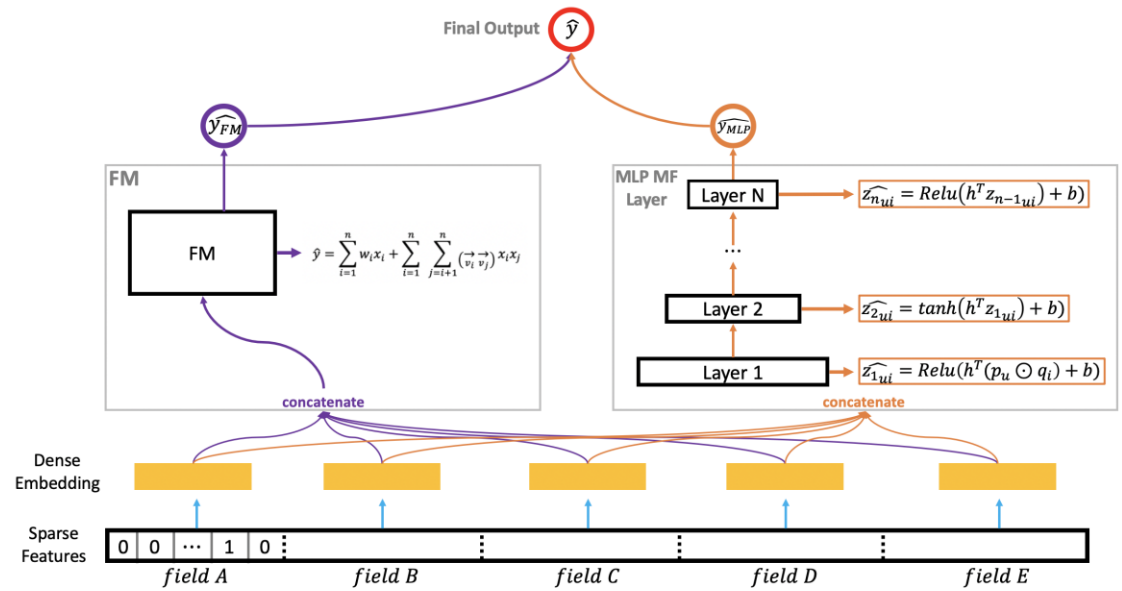
- Euclidean distance
- Cosine similarity
- Jaccard coefficient
- Pearson correlation coefficient
- …

# Approaches of Collaborative Filtering: Emergence of Deep Learning

- Deep-learning-based nonlinearity consideration complements the traditional approaches
- Deep-learning-based representation/embedding complements the traditional approaches

Feature vector **x**

| | User | | | | Movie | | | | | Other Movies rated | | | | | Time | Last Movie rated | | | | | Target y | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | ... | TI | NH | SW | ST | ... | TI | NH | SW | ST | ... | | TI | NH | SW | ST | ... | | |
| $x^{(1)}$ | 1 | 0 | 0 | ... | 1 | 0 | 0 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 13 | 0 | 0 | 0 | 0 | ... | 5 | $y^{(1)}$ |
| $x^{(2)}$ | 1 | 0 | 0 | ... | 0 | 1 | 0 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 14 | 1 | 0 | 0 | 0 | ... | 3 | $y^{(2)}$ |
| $x^{(3)}$ | 1 | 0 | 0 | ... | 0 | 0 | 1 | 0 | ... | 0.3 | 0.3 | 0.3 | 0 | ... | 16 | 0 | 1 | 0 | 0 | ... | 1 | $y^{(2)}$ |
| $x^{(4)}$ | 0 | 1 | 0 | ... | 0 | 0 | 1 | 0 | ... | 0 | 0 | 0.5 | 0.5 | ... | 5 | 0 | 0 | 0 | 0 | ... | 4 | $y^{(3)}$ |
| $x^{(5)}$ | 0 | 1 | 0 | ... | 0 | 0 | 0 | 1 | ... | 0 | 0 | 0.5 | 0.5 | ... | 8 | 0 | 0 | 1 | 0 | ... | 5 | $y^{(4)}$ |
| $x^{(6)}$ | 0 | 0 | 1 | ... | 1 | 0 | 0 | 0 | ... | 0.5 | 0 | 0.5 | 0 | ... | 9 | 0 | 0 | 0 | 0 | ... | 1 | $y^{(5)}$ |
| $x^{(7)}$ | 0 | 0 | 1 | ... | 0 | 0 | 1 | 0 | ... | 0.5 | 0 | 0.5 | 0 | ... | 12 | 1 | 0 | 0 | 0 | ... | 5 | $y^{(6)}$ |

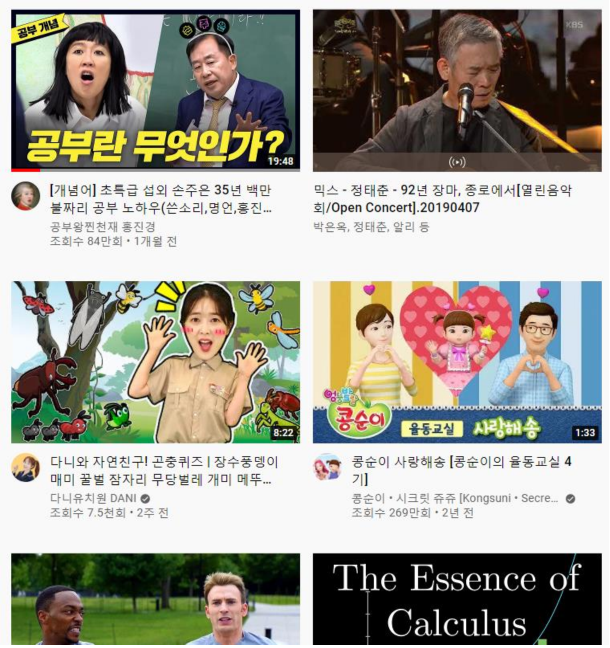





Reference: https://techblog-history-younghunjo1.tistory.com/208

# Approaches of Recommender Systems: A Categorization

- Recommender systems use analytic techniques to compute the value that a user would purchase one of the items; the techniques vary according to the purposes and data
  - Content-based filtering
  - Collaborative filtering

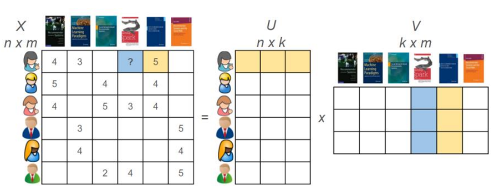



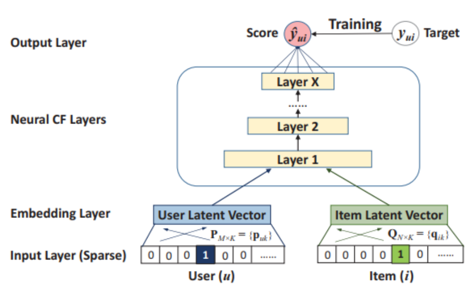
# Approaches of Recommender Systems: Our Focus

- Recommender systems use analytic techniques to compute the value that a user would purchase one of the items; the techniques vary according to the purposes and data

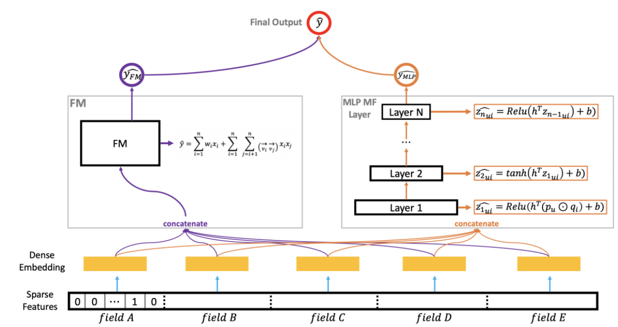
| | Item 1 | Item 2 | Item 3 | … | Item m-1 | Item m |
|---|---|---|---|---|---|---|
| User 1 | … | … | … | … | … | … |
| User 2 | … | … | … | … | … | … |
| User 3 | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| User n-1 | … | … | … | … | … | … |
| User n | … | … | … | … | … | … |

| | Feature 1 | Feature 2 | Feature 3 | … | Feature m-1 | Feature m |
|---|---|---|---|---|---|---|
| Transaction 1 | … | … | … | … | … | … |
| Transaction 2 | … | … | … | … | … | … |
| Transaction 3 | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| … | … | … | … | … | … | … |
| Transaction n-1 | … | … | … | … | … | … |
| Transaction n | … | … | … | … | … | … |




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



# Assignment 2 (by 9.16 11:59 pm)

- There are two types of practice demonstrated by TAs. One approach is to use the user-item matrix and the other is to use the transaction-feature matrix. By yourself, (1) identify several (i.e., top k) recommendations using one of the two approaches with the given datasets. Of course you can try both.
- (2) Then, evaluate and interpret the recommendation outcomes quantitatively (e.g., calculate the recall, calculate the similarities between the recommended items) and qualitatively (e.g., interpret the factorization outcome, identify the characteristics of the top k recommended items). Do it all by yourself, and describe the analysis/interpretation process and outcome in detail.
- (3) Assume you need to use your recommender system for real-world service (i.e., streaming service or hypermarket service). How can you improve your recommender system to be used for the service effectively? For example, what kinds of data should you use further? How would you design a method for using/learning the data? Think beyond these examples in your own creative, unique way!
- (4) You must have your own interested or favorite service WITHOUT a recommender system (i.e., it should be different with the intelligent services discussed in the class). Discuss the requirements of original recommender system development for the service. Describe the requirements in detail.
- (5) If you would conduct a study on the recommender system development for the service, how would you conduct the research in your own creative, unique way? What kinds of data and methods are you going to collect, analyze, and learn? Describe your service intelligence development plan in detail. If possible, visualize your plan clearly (e.g., draw an image, construct a mathematical model). To facilitate your thinking, you may want to identify and review a recommender system paper related to the service you are interested or concerned.
- Upload your code and a several paragraph essay on the tasks (1)~(5) in the Blackboard.

# Potential Discussion Points

- Relation and gap between the recommender system and service quality
- Considerations of recommender system development for real-world services:

  (1) data, (2) model, (3) service speed, (4) service UI, (5) model and service evaluation …
- Beyond the user-item matrix
  - Consideration of the side information (details of items and user contexts) is also required
  - What other customer variabilities should be considered in recommendation?
- Objectives of recommendation from the customer vs. operations perspectives
- Dealing with the customer's cognitive processes unknown
- Knowledge discovery (customer understanding) for recommender system development
  - Performance of model + Explainability of model + Interpretability of result
- Ethics around the recommender systems