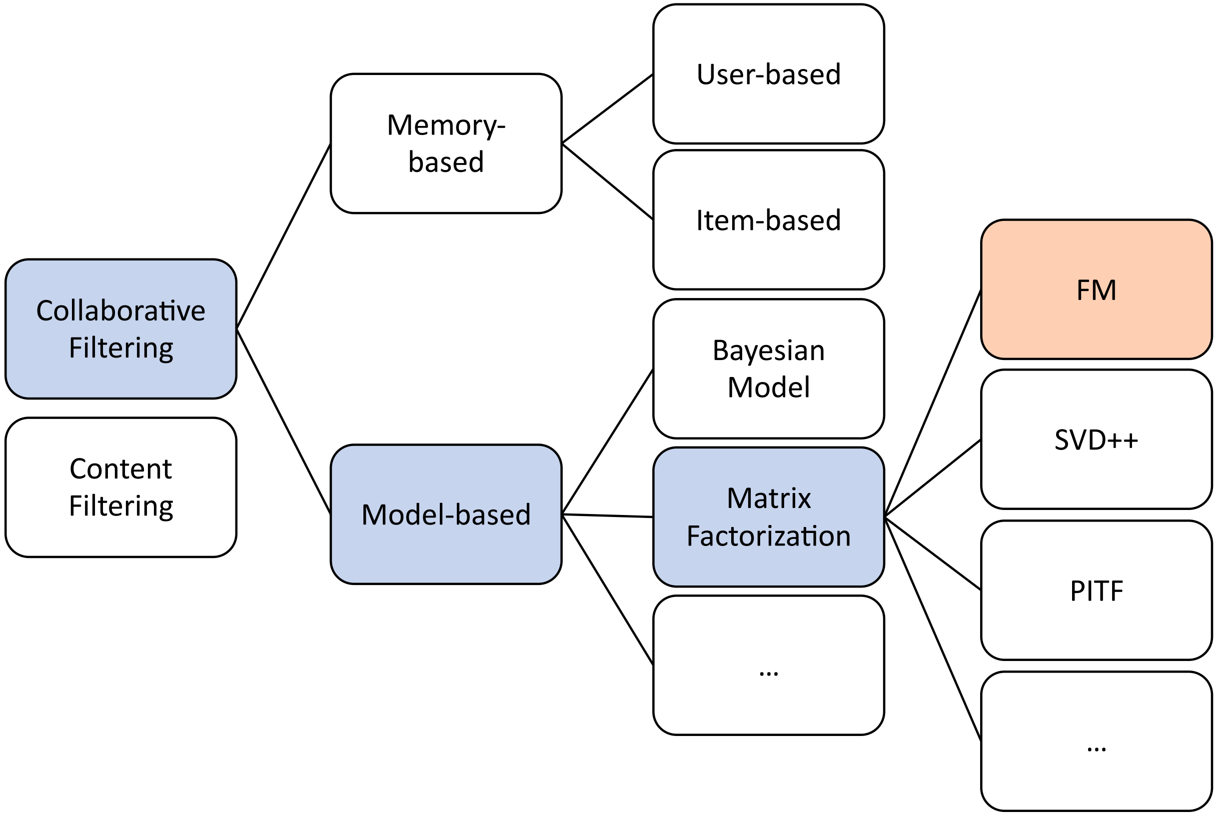
FACTORIZATION MACHINES

by
Steffen Rendle

Presented by Sungwon Kim

Contents

Background



Collaborative Filtering
Content Filtering
Memory-based
Model-based
User-based
Item-based
Bayesian Model
Matrix Factorization
...
FM
SVD++
PITF
...

Introduction

What did FMs solve?

1. **Sparsity Problem** (where SVMs fail)

2. **General Prediction Tasks** (drawback of MF, SVD++, PITF or FPMC model)

Introduction

Dataset

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	$y^{(6)}$

$\mathbf{X} \subset \mathbb{R}^{m \times n}, n = |U| + |I| + |T| + \cdots$

➡ feature data

$\mathrm{x}_i \subset \mathbb{R}^n \in D, i \in \{1,2,\cdots,m\}$

➡ feature vector

$y_i \in \mathbb{R}, i \in \{1,2,3,4,5\}$

➡ target value(rating)

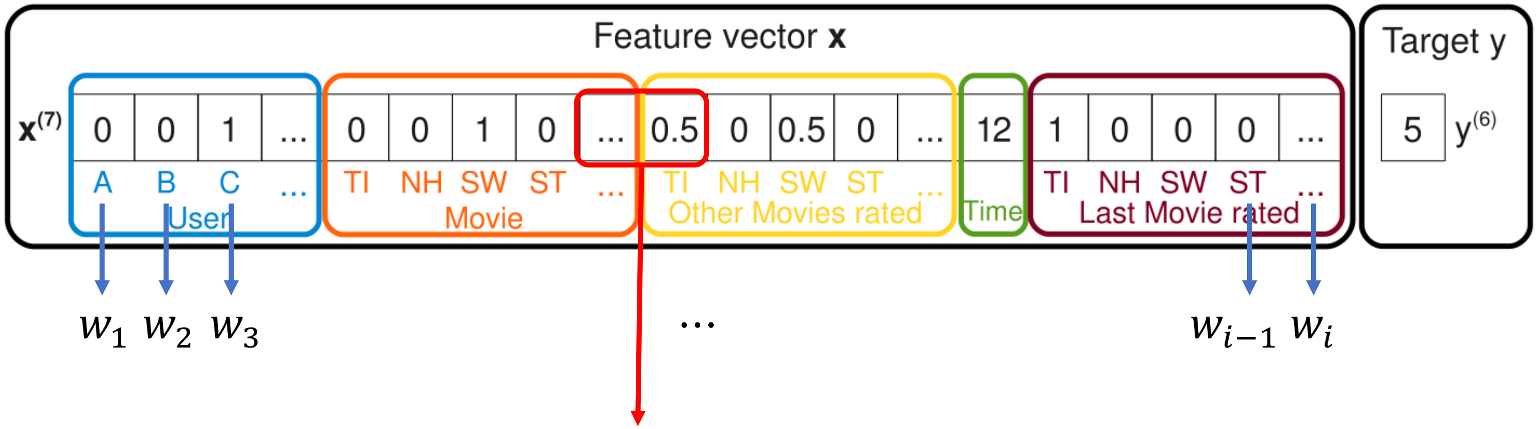
$\hat{y}(\mathrm{x})$

➡ predicted value

Factorization Machines

Linear Regression

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{n} w_i \, x_i, \quad w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^n$$

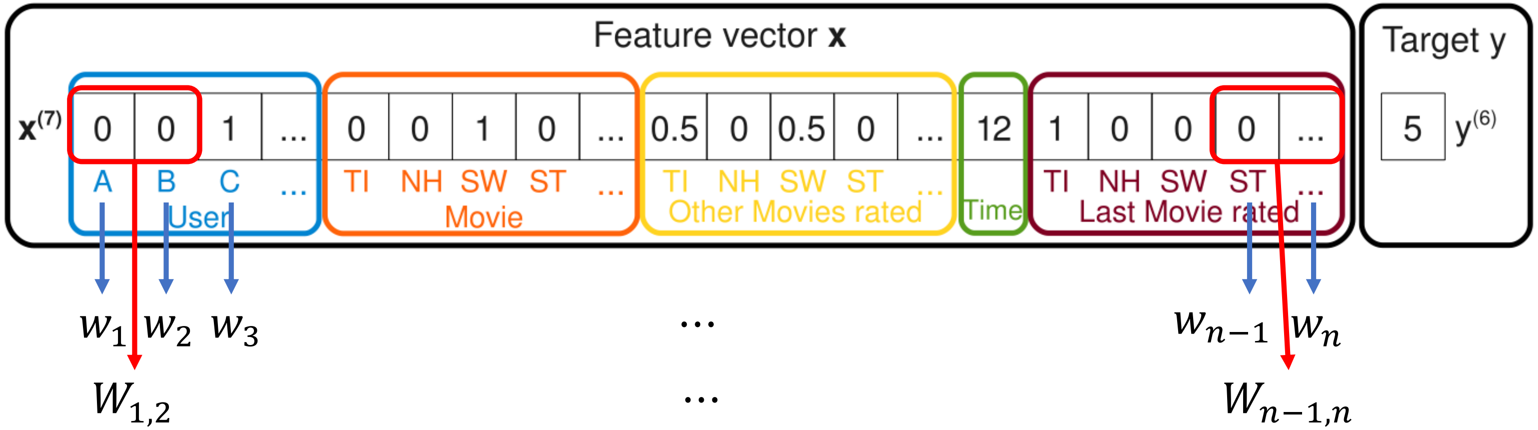


Disadvantage : 1. **No interactions** among the features x_i

2. **Huge sparsity** appears in many real-world data, but hard to be applied

Factorization Machines

Interaction parameter



$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \boxed{W_{ij}} x_i x_j \,,$$

$$w_0 \in \mathbb{R}, w \in \mathbb{R}, W \in \mathbb{R}^{n \times n}$$

Reflects interactions for each pair

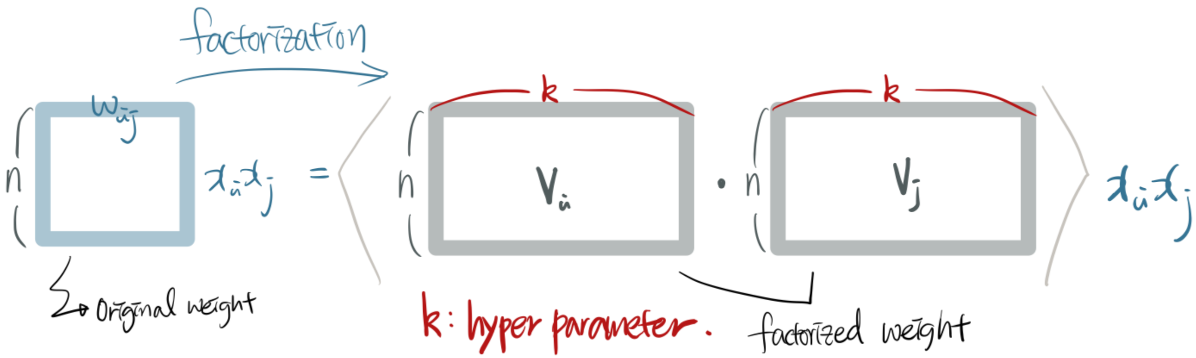
but too expensive! $O(n^2)$

Factorization Machines

Factorization Machines (2-way FM)

$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \boxed{W_{ij}} x_i x_j \, , \qquad w_0 \in \mathbb{R}, w \in \mathbb{R}, W \in \mathbb{R}^{n \times n}$$

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



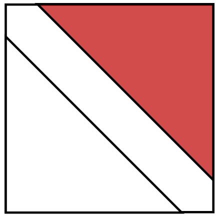
$$\mathbf{W} = \mathbf{Q\Lambda Q^T} = \left(\mathbf{Q\Lambda^{\frac{1}{2}}}\right)\left(\mathbf{Q\Lambda^{\frac{1}{2}}}\right)^{\mathbf{T}} = \mathbf{VV^T} \, , \qquad \mathbf{V} \in \mathbb{R}^{\mathbf{n \times k}}$$

∵ W is symmetric and positive simi-definite ➡ **Factorized!**

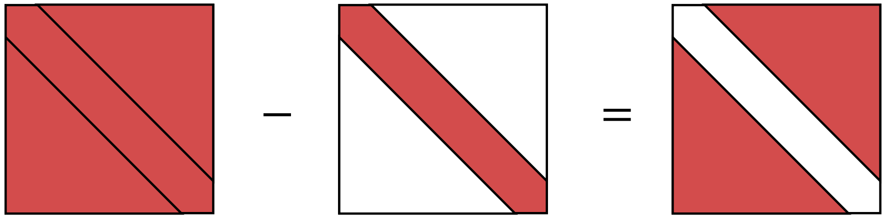
Factorization Machines

Factorization Machines (2-way FM)

$$\boxed{\sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \, x_i \, x_j}$$



$$= \frac{1}{2} \boxed{\sum_{i=1}^{n} \sum_{j=1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \, x_i \, x_j} - \frac{1}{2} \boxed{\sum_{i=1}^{n} \langle \mathbf{v}_i, \mathbf{v}_i \rangle \, x_i \, x_i}$$



$$= \frac{1}{2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{f=1}^{k} v_{i,f} \, v_{j,f} \, x_i \, x_j - \sum_{i=1}^{n} \sum_{f=1}^{k} v_{i,f} \, v_{i,f} \, x_i \, x_i \right)$$

$$= \frac{1}{2} \sum_{f=1}^{k} \left(\left(\sum_{i=1}^{n} v_{i,f} \, x_i \right) \left(\sum_{j=1}^{n} v_{j,f} \, x_j \right) - \sum_{i=1}^{n} v_{i,f}^2 \, x_i^2 \right)$$

$$= \boxed{\frac{1}{2} \sum_{f=1}^{k} \left(\left(\sum_{i=1}^{n} v_{i,f} \, x_i \right)^2 - \sum_{i=1}^{n} v_{i,f}^2 \, x_i^2 \right)} \quad \boldsymbol{O(kn)}$$

Factorization Machines

Factorization Machines

2-way FM

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

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

Computation complexity : $\boldsymbol{O(n^2) \rightarrow O(kn)}$

d-way FM

$$\hat{y}(x) := w_0 + \sum_{i=1}^{n} w_i \, x_i + \sum_{l=2}^{d} \sum_{i_1=1}^{n} \ldots \sum_{i_l=i_{l-1}+1}^{n} \left(\prod_{j=1}^{l} x_{i_j} \right) \left(\sum_{f=1}^{k_l} \prod_{j=1}^{l} v_{i_j,f}^{(l)} \right)$$

Computation complexity : Linear

Factorization Machines

Factorization Machines (2-way FM)

Moreover, FMs can be learned efficiently by Gradient Descent (e.g. SGD)

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

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

independent of i, thus **can be precomputed!**

Factorization Machines

Factorization Machines in sparse applications

FMs break the independence of the interaction parameters by factorizing them.

➡ **Data for one interaction → Estimate the parameters for related interactions**

	User				Movie					Other Movies rated						
	A	B	C	...	TI	NH	SW	ST	...	TI	NH	SW	ST	...		
User A	1	0	0	...	1	0	0	0	...	0.3	0.3	0.3	0	...	5	$y^{(1)}$
	1	0	0	...	0	1	0	0	...	0.3	0.3	0.3	0	...	3	$y^{(2)}$
	1	0	0	...	0	0	1	0	...	0.3	0.3	0.3	0	...	1	$y^{(2)}$
User B	0	1	0	...	0	0	1	0	...	0	0	0.5	0.5	...	4	$y^{(3)}$
	0	1	0	...	0	0	0	1	...	0	0	0.5	0.5	...	5	$y^{(4)}$
User C	0	0	1	...	1	0	0	0	...	0.5	0	0.5	0	...	1	$y^{(5)}$
	0	0	1	...	0	0	1	0	...	0.5	0	0.5	0	...	5	$y^{(6)}$

Direct Estimate

➡ Independence of all the interaction

$W_{A,ST} = 0$ → No Interaction

FMs Estimate

➡ Independence broken

$\mathbf{W} = \mathbf{V}\mathbf{V}^{\mathbf{T}}$, $\mathbf{V} \in \mathbb{R}^{\mathbf{n}\times\mathbf{k}}$

estimate from related interactions !

$e.g.\ V_A \not\approx V_B$,

$V_{SW} \approx V_{ST}$ $then\ V_{A,ST} \approx V_{A,SW}$

Factorization Machines

Factorization Machines in sparse applications

FMs break the independence of the interaction parameters by factorizing them.

➡ **Data for one interaction → Estimate the parameters for related interactions**

$$\mathbf{W} = \mathbf{V}\mathbf{V}^{\mathbf{T}}, \qquad \mathbf{V} \in \mathbb{R}^{\mathbf{n}\times\mathbf{k}}$$

In sparse settings, there's not enough data to estimate complex interactions W

➡ **small $\mathbf{k}$ should be chosen**

Restricting k – leads to better generalization

imporved interaction matrices under sparsity

k	total loss	accuracy	time
5	74.4560	94.3	12.4500
10	72.1391	94.2	13.5061
15	53.5665	**95.8**	14.0932
20	46.1793	95.6	14.8832
25	48.6053	95.1	15.9128
30	45.6153	94.7	16.3316

num_data features : 30, epochs : 100, lr : 0.001, Average of 10 implementations

Interim Check

To sum up

1) Interactions between values can be estimated **even under huge sparsity**. This also makes it **possible to generalize unobserved interactions**.

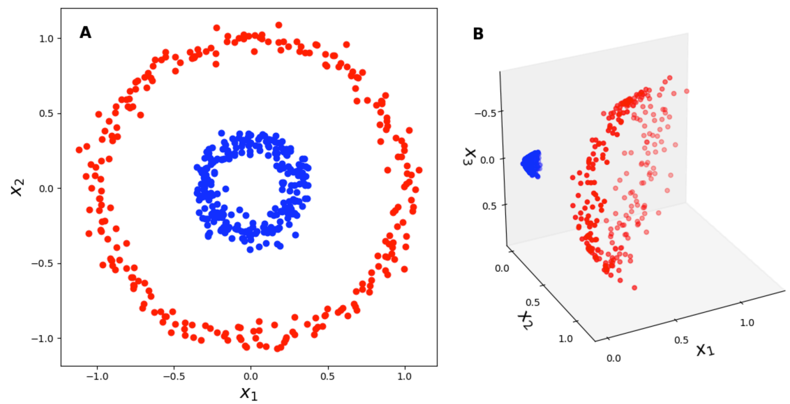
2) The time required for learning and prediction is **linear**, and thus the number of parameters is linear. – able to be applied to a variety of prediction tasks using **SGD**.

FMs vs. SVMs

Support Vector Machines

- Enlarge the feature space then solves!
- generalize the inner product then efficiently solves!

Linear SVM : $f(\boldsymbol{x}) = \beta_0 + \sum_{i \in S} \alpha_i \boxed{\langle \boldsymbol{x}, \boldsymbol{x}_i \rangle} \longrightarrow K(\boldsymbol{x}, \boldsymbol{x}_i)$



Kernels

- **Linear Kernel**

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

- **Polynomial kernel** of degree d :

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

FMs vs. SVMs

Relationships of FMs and SVMs

SVM

Linear Kernel

$$K_l(\mathbf{x}, \mathbf{z}) := 1 + \langle \mathbf{x}, \mathbf{z} \rangle$$

Model Equation

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

d-way FMs

Degree of d

$$Degree\ d = 1$$

Model Equation

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

FMs vs. SVMs

Relationships of FMs and SVMs

SVM

Polynomial Kernel (d=2)

$$K(\mathbf{x}, \mathbf{z}) := (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^d$$

Model Equation

$$\hat{y}(\mathbf{x}) = w_0 + \sqrt{2} \sum_{i=1}^{n} w_i \, x_i + \sum_{i=1}^{n} w_{i,i}^{(2)} x_i^2 + \sqrt{2} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \boxed{w_{i,j}^{(2)}} x_i \, x_j$$

All interaction parameters are completely independent

d-way FMs

Degree of d

Degree $d = 2$

Model Equation

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

The interaction parameters are factorized, share parameters

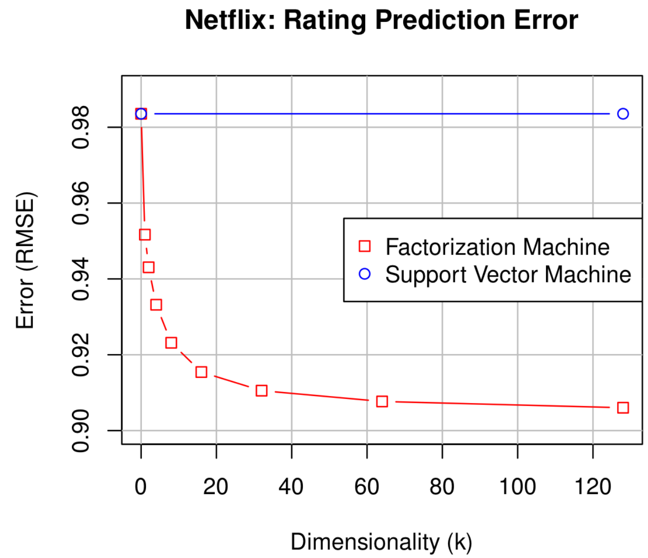
FMs vs. SVMs

Summary

1. Parameters of FMs can be estimated well even under sparsity, where SVMs fail.

2. Unlike nonlinear SVMs, FMs can be calculated in linear time and optimized directly.

3. FMs is independent of the training data. Prediction with SVMs depends on parts of the training data (the support vectors)



FMs vs. Other Factorization Models

General Prediction Tasks

FM

A. Matrix and Tensor Factorization

B. SVD++

C. PITF for Tag Recommendation

D. Factorized Personalized Markov Chains

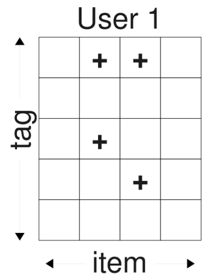
FMs can mimic these models just by specifying the input data (i.e. the feature vectors)

FMs vs. Other Factorization Models

FMs vs PITF for Tag Recommendation

PITF for Tag Recommendation

Factorization model for tag recommendation, that explicitly models the pairwise interactions between users, items, and tags.



Rank tags for a given user and item combination

$$\hat{y}_{u,i,t} = \sum_f \hat{u}_{u,f} \cdot \hat{t}^U_{t,f} + \sum_f \hat{i}_{i,f} \cdot \hat{t}^I_{t,f}$$

FMs

Specified Input Data : $n := |U \cup I \cup T|, \quad x_j := \delta\,(j = i \vee j = u \vee j = t)$

Binary indicator variables for the active user u, item i and tag t

$$\hat{y}(\mathbf{x}) = w_0 + w_u + w_i + w_t + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \langle \mathbf{v}_u, \mathbf{v}_t \rangle + \langle \mathbf{v}_i, \mathbf{v}_t \rangle$$

PITF is used for **ranking between two tags** within the **same user-item** combination

$$\hat{y}(\mathbf{x}) := w_t + \langle \mathbf{v}_u, \mathbf{v}_t \rangle + \langle \mathbf{v}_i, \mathbf{v}_t \rangle$$

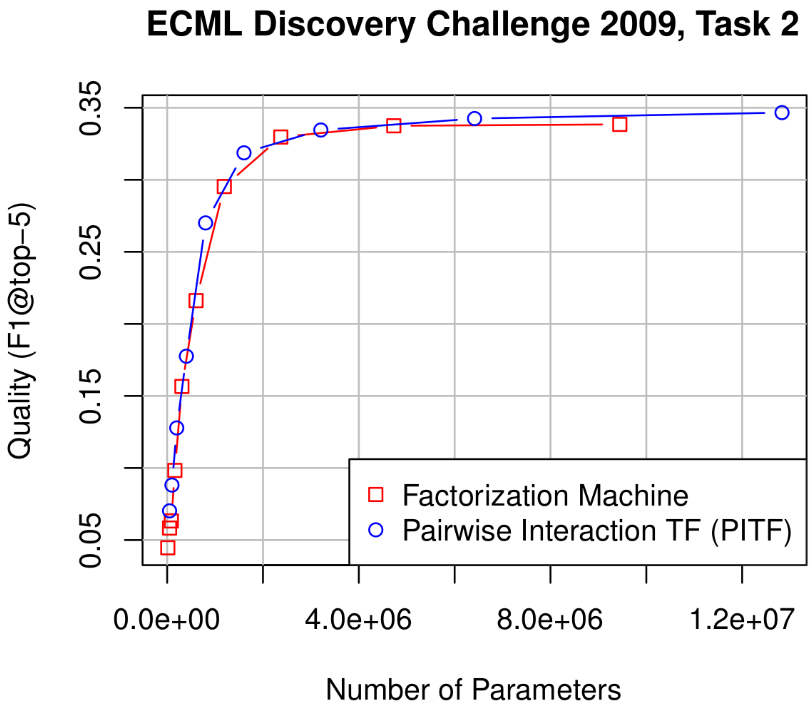
Now, PITF and FM with binary indicators are almost identical

but 1) bias term w_t,

2) interaction independency

FMs vs. Other Factorization Models

FMs vs PITF for Tag Recommendation



Conclusion and Future Work

In contrast to SVMs

1) FMs are able to estimate parameters under **huge sparsity**
2) **The model equation is linear** and depends only on the model parameters
3) Parameters can be **optimized directly** in the primal

Moreover,

1) Simply **by using the right indicators** in the input feature vector, **FMs are identical or very similar to many of the specialized models** that are applicable only for a specific task.

Discussion

1) Constrain the parameter (k) ➡ Underfit Problem
2) Degree of sparsity to perform better than SVM

Advanced Model

Field-aware Factorization Machine (FFM)

Clicked	Publisher(P)	Advertisor(A)	Gender(G)
Yes	ESPN	Nike	Male

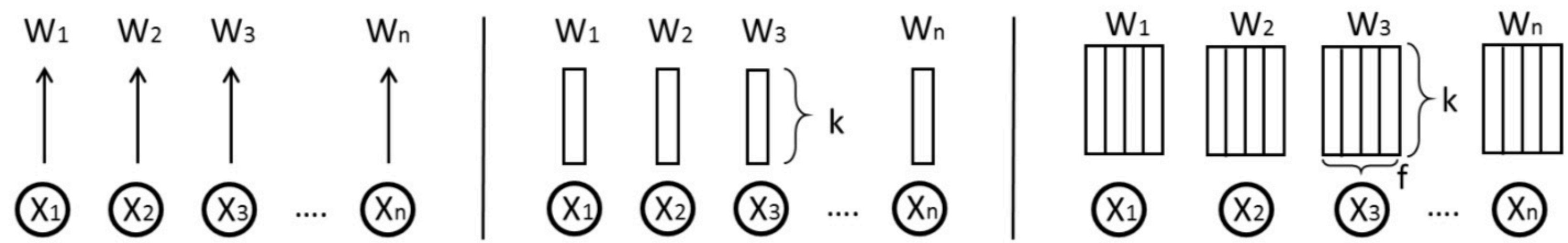


Figure 2: **Comparison of the conventional linear model, FM, and FFM.** Notice that we omit the bias term in linear model and omit the bias and linear term in FM and FFM.

References

Guo, Huifeng, et al. "DeepFM: a factorization-machine based neural network for CTR prediction." arXiv preprint arXiv:1703.04247 (2017).

Juan, Yuchin, et al. "Field-aware factorization machines for CTR prediction." Proceedings of the 10th ACM conference on recommender systems. 2016.

https://greeksharifa.github.io/machine_learning/2019/12/21/FM/

https://hyunlee103.tistory.com/69

Building a Social Network Content Recommendation Service Using Factorisation Machines - Conor Duke (https://youtu.be/9lifSPf1Y5o)

머신러닝, 딥러닝 예측모델 구현 어떻게 할까? Factorization Machine (https://www.youtube.com/watch?v=96vMbEz7nK8)

APPENDIX

FMs vs. Matrix Factorization (MF) / PARAFAC

Matrix Factorization

MF <u>factorizes a relationship</u> between two categorical variables (e.g. U and I)

$$\boxed{\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u}$$

Using binary indicator variable for each level of U and I

→ FMs can mimic MF

FMs

Specified Input Data :

User				Movie						
A	B	C	...	TI	NH	SW	ST	...		
1	0	0	...	1	0	0	0	...	5	$y^{(1)}$
1	0	0	...	0	1	0	0	...	3	$y^{(2)}$
1	0	0	...	0	0	1	0	...	1	$y^{(2)}$
0	1	0	...	0	0	1	0	...	4	$y^{(3)}$

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

x_i is only non−zero for u and i, so all other biases and interactions drop,

$$\boxed{\hat{y}(\mathbf{x}) = w_0 + w_u + w_i + \langle \mathbf{v}_u, \mathbf{v}_i \rangle}$$

APPENDIX

FMs vs. Matrix Factorization (MF) / PARAFAC

PARAFAC

A decomposition of the data is made into triads or trilinear components
→ one score vector + two loading vectors

$$x_{ijk} = \Sigma^{R}_{r=1} \boxed{a_{ir} b_{jr} c_{kr}}$$

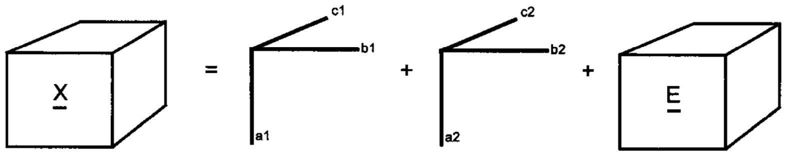


Fig. 1. A graphical representation of a two-component PARAFAC model of the data array $\underline{\mathbf{X}}$.

FMs

Specified Input Data :

User: A	B	C	...	Movie: TI	NH	SW	ST	...	Last Movie rated: TI	NH	SW	ST	...		
1	0	0	...	1	0	0	0	...	0	0	0	0	...	5	$y^{(1)}$
1	0	0	...	0	1	0	0	...	1	0	0	0	...	3	$y^{(2)}$
1	0	0	...	0	0	1	0	...	0	1	0	0	...	1	$y^{(2)}$
0	1	0	...	0	0	1	0	...	0	0	0	0	...	4	$y^{(3)}$

$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{f=1}^{k} v_{if}^{(2)} v_{jf}^{(2)} x_i x_j + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=j+1}^{n} \sum_{f=1}^{l} \boxed{v_{if}^{(3)} v_{jf}^{(3)} v_{kf}^{(3)}} x_i x_j x_k$$

Regarding to problems with more than two categorical variables,
FMs includes a nested parallel factor analysis model.

APPENDIX

FMs vs SVD++

SVD++

$$\hat{r}_{ui} = b_{ui} + q_i^T \left(p_u + |\mathrm{N}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{N}(u)} y_j \right)$$

FMs

Specified Input Data :

$$n := |U \cup I \cup L|, \quad x_j := \begin{cases} 1, & \text{if } j = i \vee j = u \\ \frac{1}{\sqrt{|N_u|}}, & \text{if } j \in N_u \\ 0, & \text{else} \end{cases}$$

U: User A	B	C	...	I: Movie TI	NH	SW	ST	...	L: Other Movies rated TI	NH	SW	ST	...		
1	0	0	...	1	0	0	0	...	0.3	0.3	0.3	0	...	5	$y^{(1)}$
1	0	0	...	0	1	0	0	...	0.3	0.3	0.3	0	...	3	$y^{(2)}$
1	0	0	...	0	0	1	0	...	0.3	0.3	0.3	0	...	1	$y^{(2)}$
0	1	0	...	0	0	1	0	...	0	0	0.5	0.5	...	4	$y^{(3)}$

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

$$\hat{y}(\mathbf{x}) = \overbrace{w_0 + w_u + w_i + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \frac{1}{\sqrt{|N_u|}} \sum_{l \in N_u} \langle \mathbf{v}_i, \mathbf{v}_l \rangle}^{\text{SVD++}}$$

$$\boxed{+ \frac{1}{\sqrt{|N_u|}} \sum_{l \in N_u} \left(w_l + \langle \mathbf{v}_u, \mathbf{v}_l \rangle + \frac{1}{\sqrt{|N_u|}} \sum_{l' \in N_u, l' > l} \langle \mathbf{v}_l, \mathbf{v}'_l \rangle \right)}$$

Additional interactions between **users vs. movies**
movie vs. movie

A FM approximately can mimic SVD++

APPENDIX

FMs vs Factorized Personalized Markov Chains (FPMC)

FPMC

Rank products in an online shop based on last purchases (at time $t-1$) of the user u

$$\hat{p}(i \in B_t^u | B_{t-1}^u) = \langle v_u^{U,I}, v_i^{I,U} \rangle + \frac{1}{|B_{t-1}^u|} \sum_{l \in B_{t-1}^u} \left(\langle v_i^{I,L}, v_l^{L,I} \rangle + \langle v_u^{U,L}, v_l^{L,U} \rangle \right)$$

FMs

Specified Input Data :

$$n := |U \cup I \cup L|, \quad x_j := \begin{cases} 1, & \text{if } j = i \vee j = u \\ \frac{1}{|B_{t-1}^u|}, & \text{if } j \in B_{t-1}^u \\ 0, & \text{else} \end{cases}$$

where $B_t^u \subseteq L$ is the set ('basket') of all items a user u has purchased at time t

$$\hat{y}(\mathbf{x}) = w_0 + w_u + w_i + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \frac{1}{|B_{t-1}^u|} \sum_{l \in B_{t-1}^u} \langle \mathbf{v}_i, \mathbf{v}_l \rangle + \frac{1}{|B_{t-1}^u|} \sum_{l \in B_{t-1}^u} \left(w_l + \langle \mathbf{v}_u, \mathbf{v}_l \rangle + \frac{1}{|B_{t-1}^u|} \sum_{l' \in B_{t-1}^u, l' > l} \langle \mathbf{v}_l, \mathbf{v}'_l \rangle \right)$$

Rank between (u, i_A, t) and (u, i_B, t)

$$\hat{y}(\mathbf{x}) = w_i + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \frac{1}{|B_{t-1}^u|} \sum_{l \in B_{t-1}^u} \langle \mathbf{v}_i, \mathbf{v}_l \rangle$$

Now, FPMC and FM are almost identical

but 1) bias term $\mathbf{w_t}$,

2) interaction independency