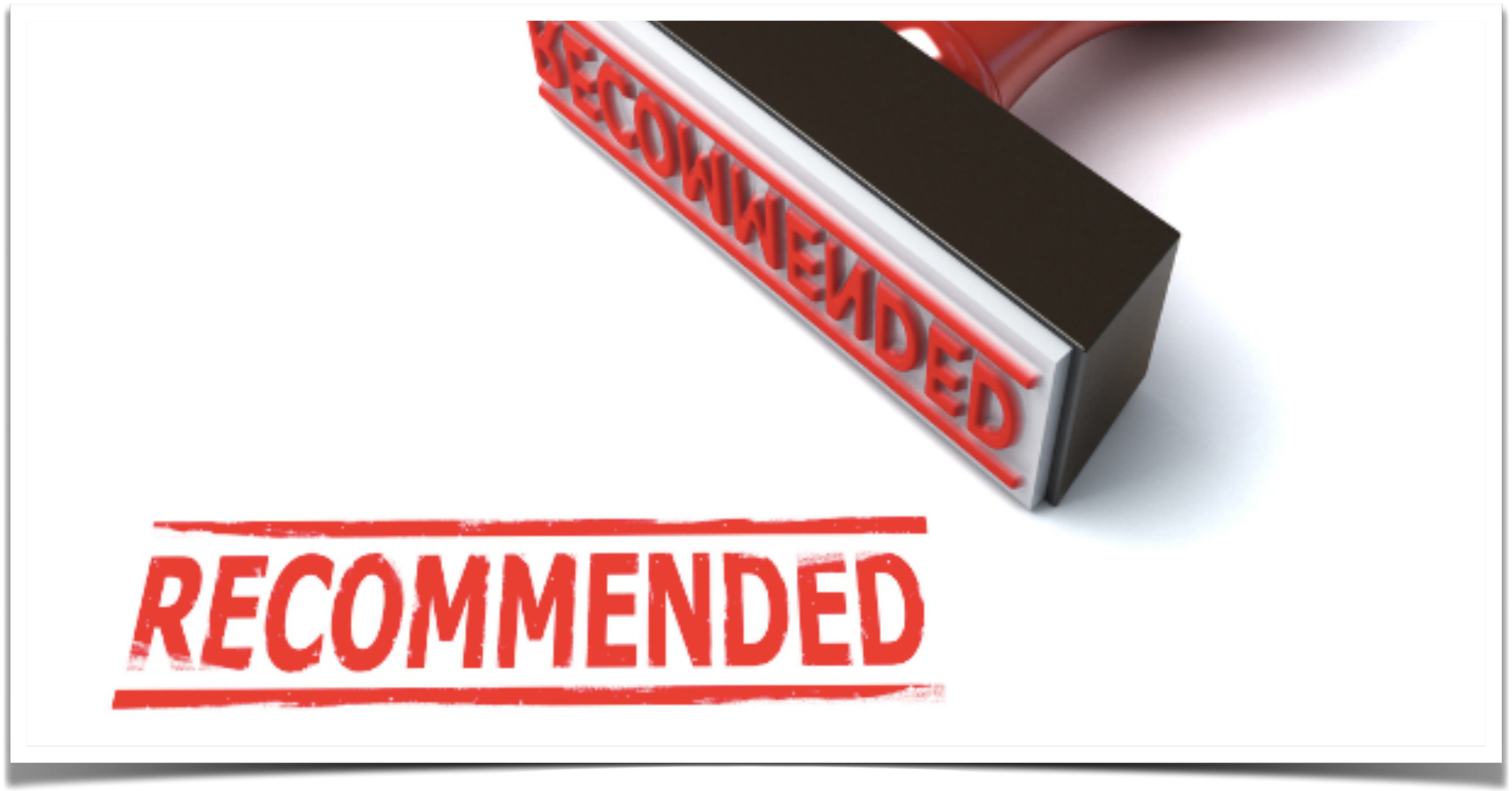
Master on Foundations of Data Science



Recommender Systems

Collaborative Recommender Systems: Factorization Models meets Factorization Machines

Santi Seguí | **2022-2023**

Matrix Factorization
Hybrid Models

Matrix Factorization with side features

Side (or content) features can be useful for 1) cold-start problem and 2) extra information about items/users

Side features can be attributes (e.g. demographics) or implicit feedback.

Bias term for occupation

$$\hat{r}_{ui} = b + b_i + b_u + p_u q_i^T + q_i t_o + b_o$$

Side term for occupation

Matrix Factorization with temporal features

Matrix factorization models have been static. However, in reality, **item popularity** and **user preferences** change constantly.
We should account for the temporal effects reflecting the dynamic nature of user-item interactions

We can add a temporal term that affects user preferences and, therefore, the interaction between users and item

User factor a a function of time

$$\hat{r}_{ui} = b + b_i + b_u + p_u q_i^T + p_u t_o + p_u(t)$$

Factorization Machines

Factorization machines

S Rendle - 2010 IEEE International conference on data mining, 2010 - ieeexplore.ieee.org

… **factorization machines** using such feature vectors as input data are related to specialized state-of-the-art **factorization** … between **factorization machines** and support vector **machines** as …

☆ Save 99 Cite Cited by 2075 Related articles All 17 versions

Our Data

Items		Items			
		i_1	i_2	i_3	i_4
	u_1	?	2	?	2
	u_2	?	3	2	?
	u_3	?	?	?	2
	u_4	1	4	2	2

1	u1	i2	2
2	u1	i4	2
3	u2	i2	3
4	u2	i3	2
5	u3	i4	2
6	u4	i1	1
7	u4	i2	4
8	u4	i3	2
9	u4	i4	2

One-hot-encoding

1	1	0	0	0	0	1	0	0	2
2	1	0	0	0	0	0	0	1	2
3	0	1	0	0	0	1	0	0	3
4	0	1	0	0	0	0	1	0	2
5	0	0	1	0	0	0	0	1	2
6	0	0	0	1	1	0	0	0	1
7	0	0	0	1	0	1	0	0	4
8	0	0	0	1	0	0	1	0	2
9	0	0	0	1	0	0	0	1	2

Linear Models

$$\hat{y} = w_0 + \sum_{j=1}^{N} w_j x_j$$

Items	i_1	i_2	i_3	i_4
u_1	?	2	?	2
u_2	?	3	2	?
u_3	?	?	?	2
u_4	1	4	2	2

1	u1	i2	2
2	u1	i4	2
3	u2	i2	3
4	u2	i3	2
5	u3	i4	2
6	u4	i1	1
7	u4	i2	4
8	u4	i3	2
9	u4	i4	2

One-hot-encoding

1	1	0	0	0	0	1	0	0	2
2	1	0	0	0	0	0	0	1	2
3	0	1	0	0	0	1	0	0	3
4	0	1	0	0	0	0	1	0	2
5	0	0	1	0	0	0	0	1	2
6	0	0	0	1	1	0	0	0	1
7	0	0	0	1	0	1	0	0	4
8	0	0	0	1	0	0	1	0	2
9	0	0	0	1	0	0	0	1	2

Polynomial Models

$$\hat{y} = w_0 + \sum_{j=1}^{N} w_j x_j + \sum_{j=1} \sum_{k=j+1} x_j x_k v_{jk}$$

Model parameters w_0, $\mathbf{w} \in \mathbb{R}^n$, $\mathbf{V} \in \mathbb{R}^{n \times n}$

Factorization Machines

$$\hat{y} = w_0 + \sum_{j=1}^{N} w_j x_j + \sum_{j=1} \sum_{k=j+1} x_j x_k < \mathbf{v}_j, \mathbf{v}_k >$$

Model parameters w_0, $\mathbf{w} \in \mathbb{R}^n$, $\mathbf{V} \in \mathbb{R}^{n \times k}$ and $< \cdot , \cdot >$ is the dot product of two vectors of size k

$$\hat{y} = w_0 + \sum_{j=1}^{N} w_j x_j + \sum_{j=1} \sum_{k=j+1} x_j x_k \sum_{f=1}^{l} v_{fj} v_{fk}$$

Factorization Machines

TRICK: Pairwise interactions can be reformulated:

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

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

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

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

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

This equation has only linear complexity in both k and n

Factorization Machines

The model is:

$$\hat{y} = w_0 + \sum_{j=1}^{N} w_j x_j + \frac{1}{2}\sum_{f=1}^{k}((\sum_{i=1}^{n} v_{i,f}x_i)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2)$$

The gradient of the **FM model** is:

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

Matrix Factorization vs. **Factorization Machines**

Items				
	i_1	i_2	i_3	i_4
u_1	?	2	?	2
u_2	?	3	2	?
u_3	?	?	?	2
u_4	1	4	2	2

Items

1	1	0	0	0	0	1	0	0	2
2	1	0	0	0	0	0	0	1	2
3	0	1	0	0	0	1	0	0	3
4	0	1	0	0	0	0	1	0	2
5	0	0	1	0	0	0	0	1	2
6	0	0	0	1	1	0	0	0	1
7	0	0	0	1	0	1	0	0	4
8	0	0	0	1	0	0	1	0	2
9	0	0	0	1	0	0	0	1	2

$$\mathbf{x} = (\underbrace{0,\ldots,0,1,0,\ldots,0}_{|U|},\underbrace{0,\ldots,0,1,0,\ldots,0}_{|I|})$$

If this is your data

(Biased) Matrix Factorization == Factorization Machines

$$\hat{y}(\mathbf{x}) = \hat{y}(u,i) = w_0 + w_u + w_i + \sum_{j=1}^{k} v_{u,j} v_{i,j}$$

FM and SVM

- FM combines the advantages of SVM and factorization models
- Good estimates interaction model with huge sparsity where SVM fail.
- Comparable to polynomial kernel in SVM, but works for very sparse data and much faster

Factorization Machines

- Example:

$$
\begin{aligned}
U &= \{\text{Alice (A)}, \text{Bob (B)}, \text{Charlie (C)}, \ldots\} \\
I &= \{\text{Titanic (TI)}, \text{Notting Hill (NH)}, \text{Star Wars (SW)}, \\
&\quad \text{Star Trek (ST)}, \ldots\}
\end{aligned}
$$

- The observed data:

$$
\begin{aligned}
S = \{&(\text{A}, \text{TI}, \text{2010-1}, 5), (\text{A}, \text{NH}, \text{2010-2}, 3), (\text{A}, \text{SW}, \text{2010-4}, 1), \\
&(\text{B}, \text{SW}, \text{2009-5}, 4), (\text{B}, \text{ST}, \text{2009-8}, 5), \\
&(\text{C}, \text{TI}, \text{2009-9}, 1), (\text{C}, \text{SW}, \text{2009-12}, 5)\}
\end{aligned}
$$

Factorization Machies

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	...	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	$y^{(6)}$

Factorization machines

S Rendle - 2010 IEEE International conference on data mining, 2010 - ieeexplore.ieee.org

… **factorization machines** using such feature vectors as input data are related to specialized state-of-the-art **factorization** … between **factorization machines** and support vector **machines** as …

FM and SVD++

- **Explicit** (e.g. numerical rantings) + **Implicit** information (e.g. likes, purchases, skipped, bookmarked,…)

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

  - *N(u)* is the set of items for which the user *u* has implicit information

FM and SVD++

- **Explicit** (e.g. numerical rantings) + **Implicit** information (e.g. likes, purchases, skipped, bookmarked,…)

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

$$(u, i, \{l_1, \ldots, l_m\}) \rightarrow \mathbf{x} = (\underbrace{0, \ldots, 1, 0, \ldots}_{|U|}, \underbrace{0, \ldots, 1, 0, \ldots}_{|I|}, \underbrace{0, \ldots, 1/m, 0, \ldots, 1/m, 0, \ldots}_{|L|}),$$

$$\hat{y}(\mathbf{x}) = \hat{y}(u, i, \{l_1, \ldots, l_m\}) = \overbrace{w_0 + w_u + w_i + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \frac{1}{m} \sum_{j=1}^{m} \langle \mathbf{v}_i, \mathbf{v}_{l_j} \rangle}^{SVD++}$$

Factorization Machines

- Offers combination of regression and factorization models
- Low rank approximation ranking enables estimation of unobserved interactions
- Effective for sparse and extremely sparse data sets
- Flexible through feature engineering