

Máquinas de Factorización { Factorization Machines (FM) }

Denis Parra

Sistemas Recomendadores

IIC 3633

Máquinas de Factorización (2010)

- Inspiradas en SVM, permiten agregar un número arbitrario de features (user, item, contexto) pero funcionan bien con “sparse data” al incorporar variables latentes factorizadas (inspiradas en Factorización Matricial). No se necesitan vectores de soporte para optimizar el modelo.
- Generalizan diversos métodos de factorización matricial.
- Disminuyen la complejidad de aprendizaje del modelo de predicción respecto de métodos anteriores.

Motivación de FM

- Cada tarea de recomendación (implicit feedback, agregar tiempo, incorporar contexto) requiere rediseño del modelo de optimización y re-implementación del algoritmo de inferencia
- Lo ideal sería usar alguna herramienta como libSVM, Weka, ... agregar los vectores de features
- Pero para manejar datos tan dispersos, se podrían mantener las factorizaciones!

Ejemplo

- Supongamos los siguientes usuarios, items y transacciones

$$U = \{\text{Alice (A), Bob (B), Charlie (C), ...}\}$$

$$\begin{aligned} I = \{ &\text{Titanic (TI), Notting Hill (NH), Star Wars (SW),} \\ &\text{Star Trek (ST), ...} \end{aligned}$$

Let the observed data S be:

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

Representación Tradicional

Example for data:

		Movie				
		TI	NH	SW	ST	...
User	A	5	3	1	?	...
	B	?	?	4	5	...
	C	1	?	5	?	...

Matrix Factorization:

$$\hat{Y} := W H^t, \quad W \in \mathbb{R}^{|U| \times k}, H \in \mathbb{R}^{|I| \times k}$$

$$\hat{y}(u, i) = \hat{y}_{u,i} = \sum_{f=1}^k w_{u,f} h_{i,f} = \langle \mathbf{w}_u, \mathbf{h}_i \rangle$$

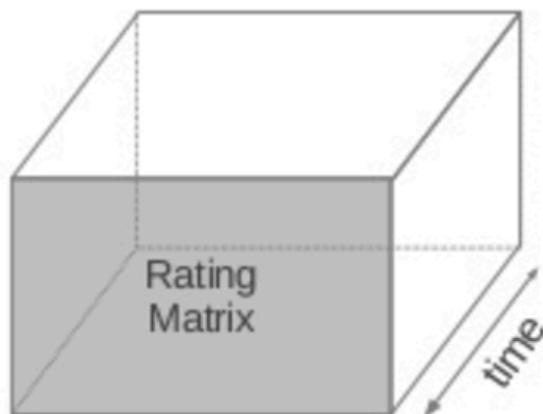
k is the rank of the reconstruction.

$$\min_{q^*, p^*} \sum_{(u,i) \in K} (r_{ui} - q_i^T \cdot p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

Otros Modelos

Ejemplos de Datos:

		Movie				
		TI	NH	SW	ST	...
User		5	3	1	?	...
A		5	3	1	?	...
B		?	?	4	5	...
C		1	?	5	?	...
...	



Ejemplos de Modelos:

$$\hat{y}^{\text{MF}}(u, i) := \sum_{f=1}^k v_{u,f} v_{i,f} = \langle \mathbf{v}_u, \mathbf{v}_i \rangle$$

$$\hat{y}^{\text{SVD++}}(u, i) := \left\langle \mathbf{v}_u + \sum_{j \in N(u)} \mathbf{v}_j, \mathbf{v}_i \right\rangle$$

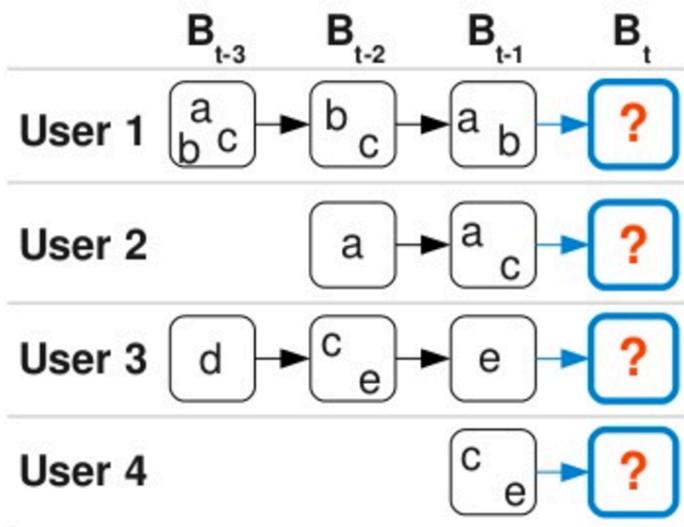
$$\hat{y}^{\text{Fact-KNN}}(u, i) := \frac{1}{|R(u)|} \sum_{j \in R(u)} r_{u,j} \langle \mathbf{v}_i, \mathbf{v}_j \rangle$$

$$\hat{y}^{\text{timeSVD}}(u, i, t) := \langle \mathbf{v}_u + \mathbf{v}_{u,t}, \mathbf{v}_i \rangle$$

$$\hat{y}^{\text{timeTF}}(u, i, t) := \sum_{f=1}^k v_{u,f} v_{i,f} v_{t,f}$$

...

Modelos de Factorización Secuencial



$$\hat{y}^{\text{FMC}}(u, i, t) := \sum_{I \in B_{t-1}} \langle \mathbf{v}_i, \mathbf{v}_I \rangle$$

$$\hat{y}^{\text{FPMC}}(u, i, t) := \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \sum_{I \in B_{t-1}} \langle \mathbf{v}_i, \mathbf{v}_I \rangle$$

...

Modelos de Factorización

- Ventaja:
 - Permiten estimar interacciones entre dos (o más) variables incluso si la interacción no es observada explícitamente.
- Desventajas:
 - Modelos específicos para cada problema
 - Algoritmos de aprendizaje e implementaciones están diseñados para modelos individuales

Datos y Representación de Variables

- Muchos modelos de ML usan vectores de valores reales como input, lo que permite representar, por ejemplo:
 - Cualquier número de variables
 - Variables categóricas -> dummy coding
- Con este modelo estándar podemos usar regresión, SVMs, etc.

Modelo de Regresión Lineal

- Equivale a un polinomio de grado 1
- Queremos aprender w_0 y los p parámetros w_j
- No logra capturar interacciones latentes como la factorización matricial

$$\check{y}(x) = w_0 + \sum_{j=1}^p w_j x_j$$

- $O(p)$ parámetros en el modelo.

Modelo con interacciones (d=2)

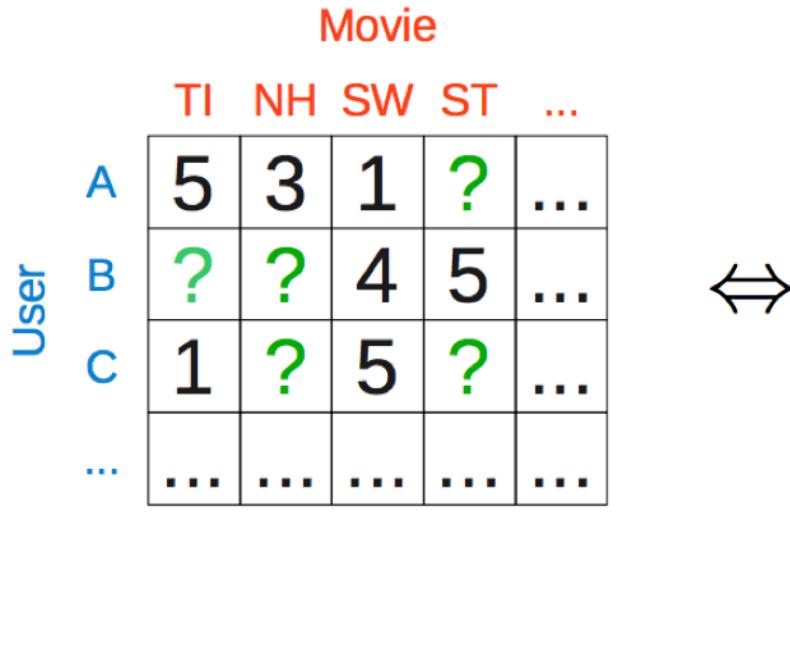
- Regresión Polinomial

$$\check{y}(x) = w_0 + \sum_{j=1}^p w_j x_j + \sum_{j=1}^p \sum_{j'=j+1}^p x_j x_{j'} w_{j,j'}$$

- $O(p^2)$ parámetros en el modelo

$$w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^p, \quad \mathbf{W} \in \mathbb{R}^{p \times p}$$

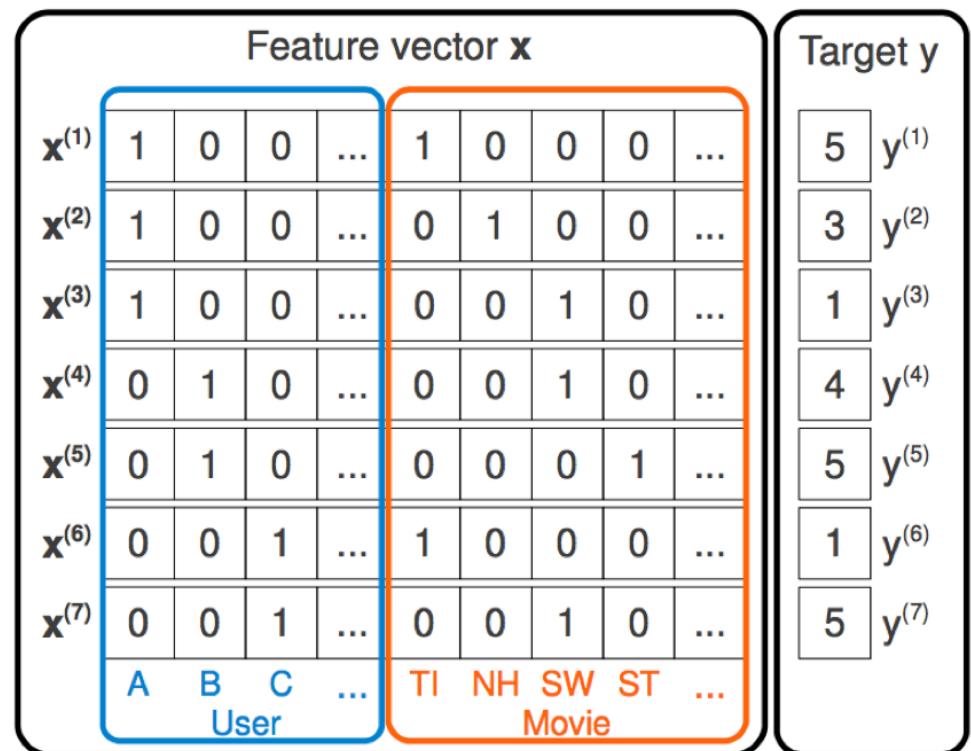
Representación Matricial como Vector de Features



#	User	Movie	Rating
1	Alice	Titanic	5
2	Alice	Notting Hill	3
3	Alice	Star Wars	1
4	Bob	Star Wars	4
5	Bob	Star Trek	5
6	Charlie	Titanic	1
7	Charlie	Star Wars	5
...

Representación Matriz como Vector de Features

#	User	Movie	Rating
1	Alice	Titanic	5
2	Alice	Notting Hill	3
3	Alice	Star Wars	1
4	Bob	Star Wars	4
5	Bob	Star Trek	5
6	Charlie	Titanic	1
7	Charlie	Star Wars	5
...



Aplicación de Regresión

Feature vector \mathbf{x}							Target y				
$\mathbf{x}^{(1)}$	1	0	0	...	$\mathbf{x}^{(1)}$	1	0	0	0	...	$y^{(1)}$
$\mathbf{x}^{(2)}$	1	0	0	...	$\mathbf{x}^{(2)}$	0	1	0	0	...	$y^{(2)}$
$\mathbf{x}^{(3)}$	1	0	0	...	$\mathbf{x}^{(3)}$	0	0	1	0	...	$y^{(3)}$
$\mathbf{x}^{(4)}$	0	1	0	...	$\mathbf{x}^{(4)}$	0	0	1	0	...	$y^{(4)}$
$\mathbf{x}^{(5)}$	0	1	0	...	$\mathbf{x}^{(5)}$	0	0	0	1	...	$y^{(5)}$
$\mathbf{x}^{(6)}$	0	0	1	...	$\mathbf{x}^{(6)}$	1	0	0	0	...	$y^{(6)}$
$\mathbf{x}^{(7)}$	0	0	1	...	$\mathbf{x}^{(7)}$	0	0	1	0	...	$y^{(7)}$
	A	B	C	...		TI	NH	SW	ST	...	
	User					Movie					

- Regresión Lineal: $\hat{y}(\mathbf{x}) = w_0 + w_u + w_i$
- Regresión Polinomial: $\hat{y}(\mathbf{x}) = w_0 + w_u + w_i + w_{u,i}$
- Factorización Matricial: $\hat{y}(u, i) = \langle \mathbf{w}_u, \mathbf{h}_i \rangle$

Problemas con Regresión Tradicional

- Regresión lineal no considera interacciones usuario-item : poder de expresión muy bajo
- Regresión Polinomial incluye interacciones de pares pero no se puede estimar porque
 - $n << p^2$: nro. de casos mucho menor que el número de parámetros.
 - Regresión polinomial no puede generalizar para cualquier efecto de pares de variables.

Modelo con interacción d=2 y factores latentes vs. Regresión polinomial

- Máquina de Factorización

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

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

- Regresión Polinomial

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^p w_i x_i + \sum_{i=1}^p \sum_{j \geq i}^p w_{i,j} x_i x_j$$

$$w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^p, \quad \mathbf{W} \in \mathbb{R}^{p \times p}$$

F.M. dado un modelo con d=2

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{j=1}^p w_j x_j + \sum_{j=1}^p \sum_{j'=j+1}^p x_j x_{j'} + \sum_{f=1}^k v_{j,f} v_{j',f},$$

Diagram illustrating the components of the model:

- w_0 (Sesgo (bias) global)
- w_j (Coeficientes de regresión de la j-ésima variable)
- $\sum_{j=1}^p \sum_{j'=j+1}^p x_j x_{j'}$ (Interacción de features)
- $\sum_{f=1}^k v_{j,f} v_{j',f}$ (Factorización (variables latentes))

F.M. dado un modelo con d=3

- Modelo

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

$$+ \sum_{i=1}^p \sum_{j>i}^p \sum_{l>j}^p \sum_{f=1}^k v_{i,f}^{(3)} v_{j,f}^{(3)} v_{l,f}^{(3)} x_i x_j x_l$$

- Parámetros

$$w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^p, \quad \mathbf{V} \in \mathbb{R}^{p \times k}, \quad \mathbf{V}^{(3)} \in \mathbb{R}^{p \times k}$$

En suma

- FMs usan como entrada datos numéricos reales
- FMs incluyen interacciones entre variables como la regresión polinomial
- Los parámetros del modelo para las interacciones son factorizados
- Número de parámetros es $O(kp)$ vs. $O(p^2)$ en regresión polinomial.

Ejemplos

- A. Dos variables categóricas
- B. Tres variables categóricas
- C. Dos variables categóricas y tiempo como predictor continuo
- D. Dos variables categóricas y tiempo discretizado en bins
- E. SVD++
- F. Factorized Personalized Markov Chains (FPMC)

A. Dos variables categóricas

Feature vector \mathbf{x}									
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...
	A	B	C	...	TI	NH	SW	ST	...
	User				Movie				

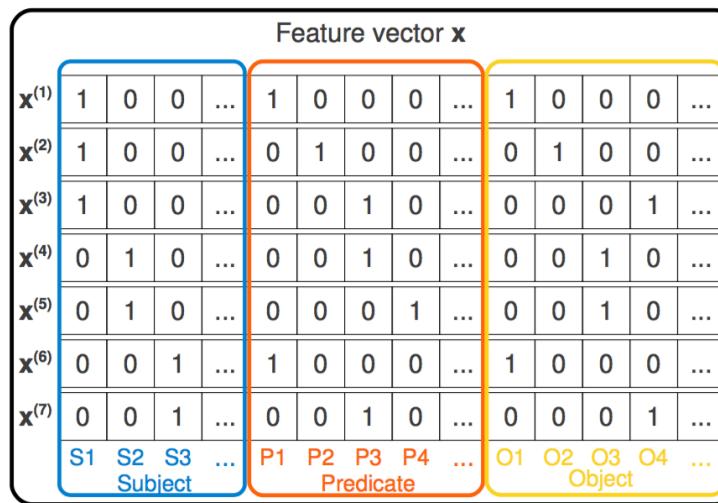
- Así, modelo corresponde a MF con biases

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

libFM, $k = 128$, MCMC inference, Netflix RMSE=0.8937

B. Tres variables categóricas

- Predicción de tripletas RDF con FM



- Equivalente a PITF (recomendación de tags)

$$\hat{y}(\mathbf{x}) := w_0 + w_s + w_p + w_o + \langle \mathbf{v}_s, \mathbf{v}_p \rangle + \langle \mathbf{v}_s, \mathbf{v}_o \rangle + \langle \mathbf{v}_p, \mathbf{v}_o \rangle$$

C. Dos variables categóricas y tiempo como predictor continuo

Feature vector \mathbf{x}										
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...	0.2
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...	0.6
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...	0.61
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...	0.3
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...	0.5
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...	0.1
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...	0.8
	A	B	C	...	TI	NH	SW	ST	...	Time
	User				Movie				Time	

- Modelo corresponde a:

$$\hat{y}(\mathbf{x}) := w_0 + w_i + w_u + t w_{\text{time}} + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + t \langle \mathbf{v}_u, \mathbf{v}_{\text{time}} \rangle + t \langle \mathbf{v}_i, \mathbf{v}_{\text{time}} \rangle$$

D. Dos variables categóricas y tiempo discretizado en bins

Feature vector \mathbf{x}									
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...
	A	B	C	...	TI	NH	SW	ST	...
	User				Movie				
					T1	T2	T3		
					Time				

- Modelo corresponde a:

$$\hat{y}(\mathbf{x}) := w_0 + w_i + w_u + w_{b(t)} + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \langle \mathbf{v}_u, \mathbf{v}_{b(t)} \rangle + \langle \mathbf{v}_i, \mathbf{v}_{b(t)} \rangle$$

libFM, $k = 128$, MCMC inference, Netflix RMSE=0.8873

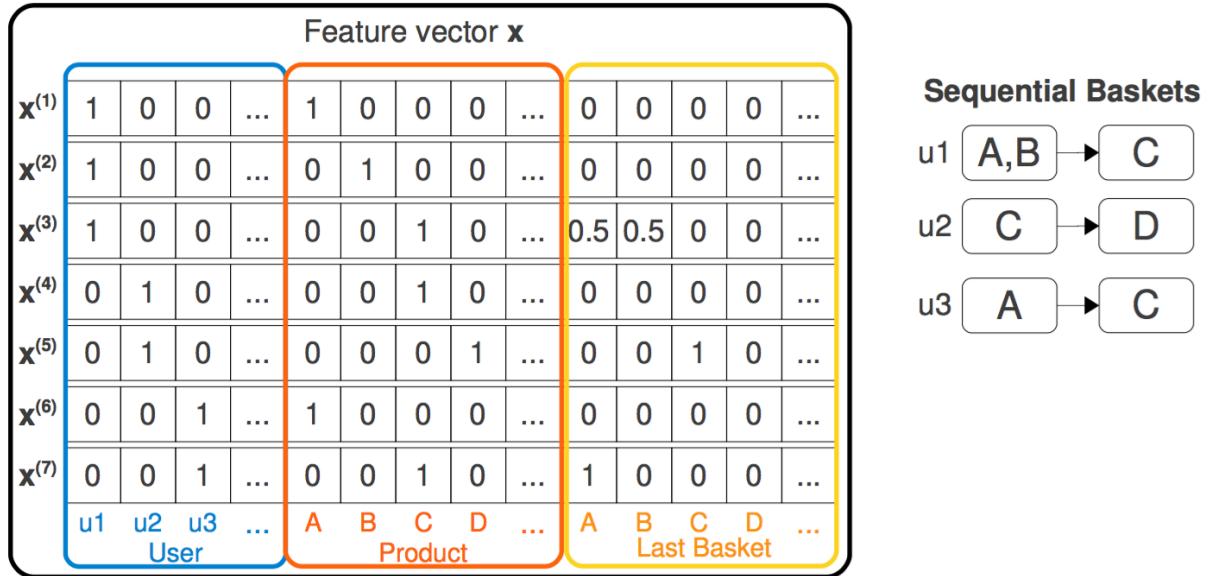
E. SVD++

Feature vector \mathbf{x}										
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...	
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...	
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...	
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...	
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...	
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...	
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...	
A B C User	TI	NH	SW	ST	...	TI	NH	SW	ST	...
Movie	Other Movies rated									

- Modelo idéntico a:

$$\begin{aligned}
 \hat{y}(\mathbf{x}) = & \underbrace{w_0 + w_u + w_i + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \frac{1}{\sqrt{|N_u|}} \sum_{l \in N_u} \langle \mathbf{v}_i, \mathbf{v}_l \rangle}_{\text{SVD++}} \\
 & + \frac{1}{\sqrt{|N_u|}} \sum_{l \in N_u} \left(w_l + \langle \mathbf{v}_u, \mathbf{v}_l \rangle + \frac{1}{\sqrt{|N_u|}} \sum_{l' \in N_u, l' > l} \langle \mathbf{v}_l, \mathbf{v}'_{l'} \rangle \right)
 \end{aligned}$$

F. FPMC



- Equivalente a:

$$\hat{y}(\mathbf{x}) := w_0 + w_u + \mathbf{w}_i + \frac{1}{|B_{t-1}|} \sum_{j \in B_{t-1}} w_j + \langle \mathbf{v}_u, \mathbf{v}_i \rangle + \frac{1}{|B_{t-1}|} \sum_{j \in B_{t-1}} \langle \mathbf{v}_i, \mathbf{v}_j \rangle + \dots$$

[Rendle et al. 2010, WWW Best Paper]

Comparación con otros modelos

- En el paper Rendle, S. (2010, December). **Factorization machines**, se muestra como desde FM se puede derivar:
 - Matrix Factorization
 - SVD++
 - Pair-wise Interaction Tag-Factorization (PITF)
 - Factorized Personalized Markov Chains (FPMC)

Propiedades

- Expresividad* (cualquier matrix semi-definida positiva)

- Multilinearidad**

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

- Complexity

$O(kn^2) \rightarrow O(kn)$

Y debido a dispersión de los datos, $O(km_D)$

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

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

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

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

* , ** ver detalles en Rendle, S. (2010, December). **Factorization machines.**

Complejidad

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{j=1}^p w_j x_j + \sum_{j=1}^p \sum_{j'=j+1}^p x_j x_{j'} \sum_{f=1}^k v_{j,f} v_{j',f},$$

Número de parámetros :

$$1 + p + k*p$$

lineal respecto al tamaño del input y el tamaño de los factores latentes

Reducción del modelo

1) *Model Equation:* The model equation for a factorization machine of degree $d = 2$ is defined as:

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

where the model parameters that have to be estimated are:

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

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

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

Aprendizaje

- Regularización L2 para regresión y clasificación
 - SGD
 - ALS
 - MCMC
- Ranking regularizado L2

Todos los algoritmos tienen tiempo de ejecución $O(k N_z(x) i)$ donde i : iteraciones, $N_z(X)$: elementos no-cero, y k : nro. de factores latentes.

Software: LibFM

- LibFM implementa FMs
 - Modelos: FMs de 2do orden
 - Aprendizaje: SGD, ALS, MCMC
 - Clasificación y regresión
 - Formato de datos: sparse (LIBSVM, LIBLINEAR, SVMlight, etc.)
 - Soporta agrupación de variables
 - Open Source: GPLv3

www.libfm.org

Open Screenshot

[Source Code](#) [Latest Release](#) [Usage](#) [References](#)

libFM: Factorization Machine Library

Author: Steffen Rendle

Factorization machines (FM) are a generic approach that allows to mimic most factorization models by feature engineering. This way, factorization machines combine the generality of feature engineering with the superiority of factorization models in estimating interactions between categorical variables of large domain. libFM is a software implementation for factorization machines that features stochastic gradient descent (SGD) and alternating least squares (ALS) optimization as well as Bayesian inference using Markov Chain Monte Carlo (MCMC).

Source code

- **github** repository: <https://github.com/srendle/libfm>.
- Please acknowledge the software (i.e. cite the paper *Factorization Machines with libFM*) if you publish results produced with this software.

Latest release

- **Source code (C++)** [libfm-1.42.src.tar.gz \(2014-09-14\)](#), GPL v3 license
- **Windows Executable** [libfm-1.40.windows.zip \(2013-07-12\)](#)
- The license is included in the archive -- please see the file `license.txt` for details.
- Please acknowledge the software (i.e. cite the paper *Factorization Machines with libFM*) if you publish results produced with this software.

Usage

Please see the [libFM 1.4.2 manual](#) for details about how to use libFM. This manual is also included in the tar.gz archive of the source code.

References

If you use libFM please cite the following paper:

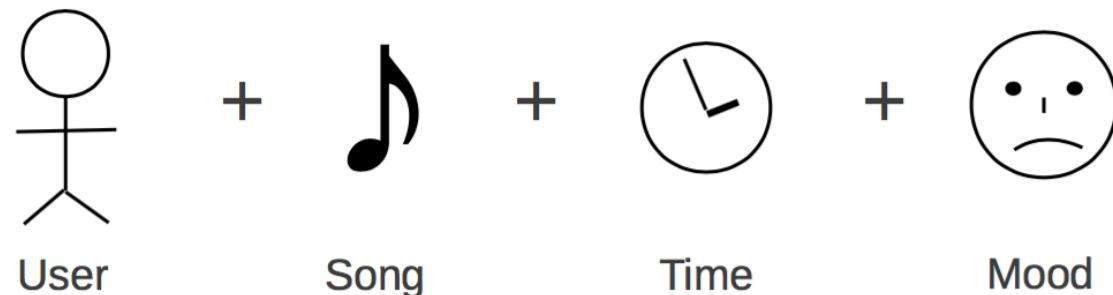
Steffen Rendle (2012): *Factorization Machines with libFM*, in ACM Trans. Intell. Syst. Technol., 3(3), May. [\[PDF\]](#)

BibTeX:

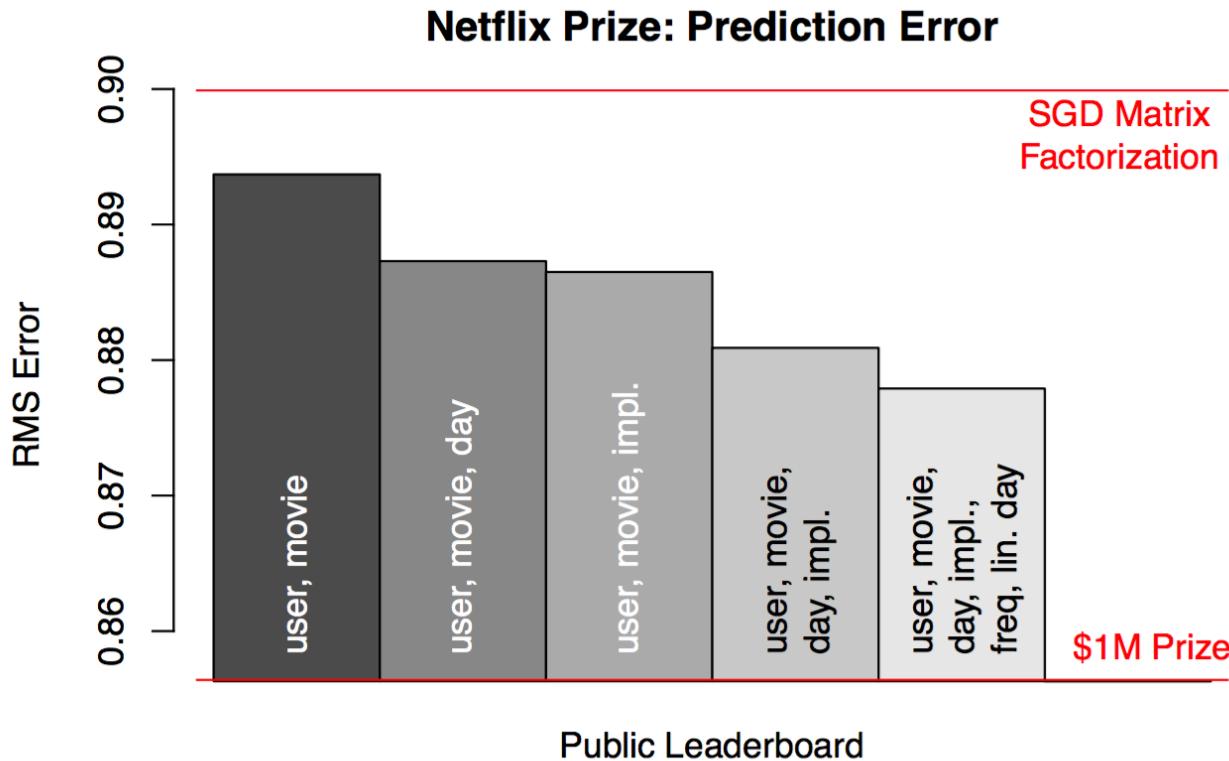
```
@article{rendle:tist2012,
    author = {Rendle, Steffen},
    title = {Factorization Machines with {libFM}},
    journal = {ACM Trans. Intell. Syst. Technol.},
    issue_date = {May 2012},
    volume = {3},
    number = {3},
    month = May,
    year = {2012},
    issn = {2157-6904},
    pages = {57:1--57:22},
    articleno = {57},
    numpages = {22},
    publisher = {ACM},
    address = {New York, NY, USA}.
```

Predictión de ratings (Context-aware)

- ▶ Main variables:
 - ▶ User ID (categorical)
 - ▶ Item ID (categorical)
- ▶ Additional variables:
 - ▶ time
 - ▶ mood
 - ▶ user profile
 - ▶ item meta data
 - ▶ ...
- ▶ Examples: Netflix prize, MovieLens, KDDCup 2011



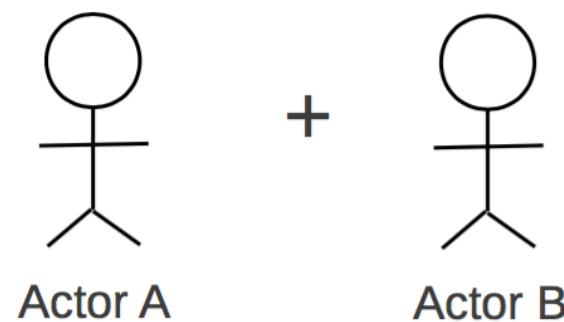
Netflix Prize



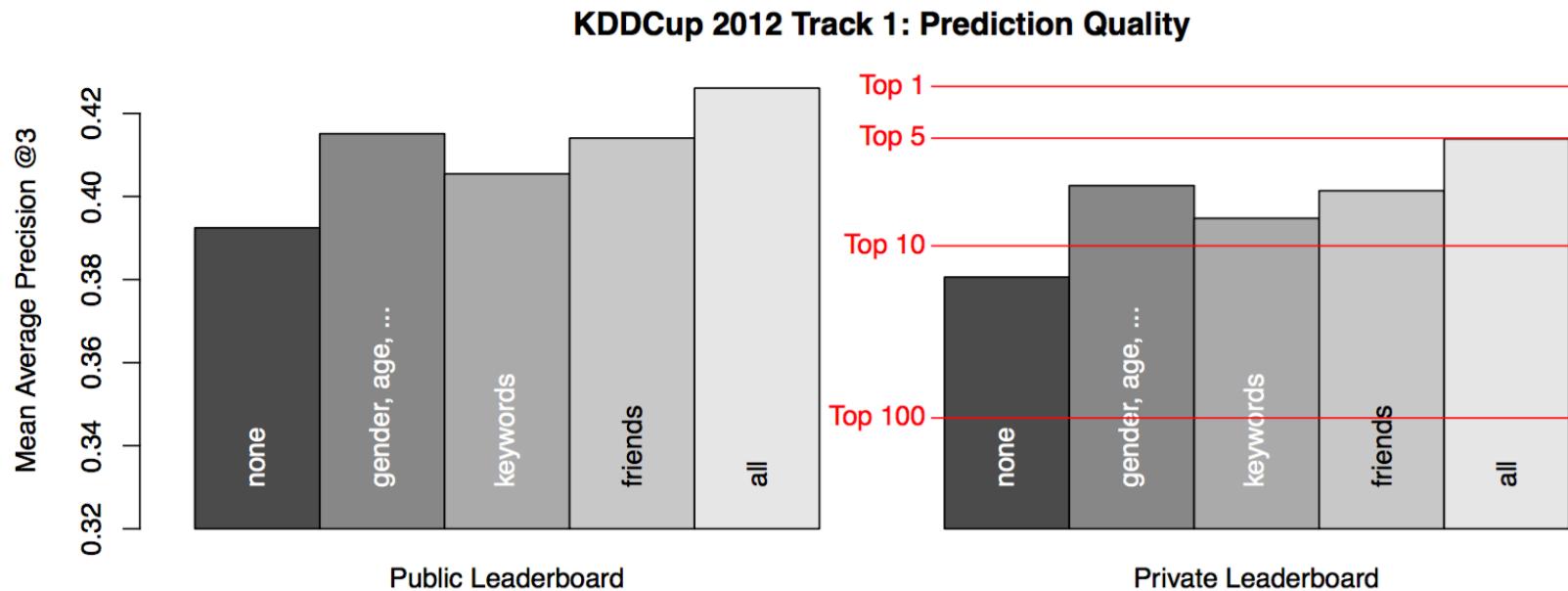
- ▶ $k = 128$ factors, 512 MCMC samples (no burnin phase, initialization from random)
- ▶ MCMC inference (no hyperparameters (learning rate, regularization) to specify)

Predicción de relaciones en Redes

- ▶ Main variables:
 - ▶ Actor A ID
 - ▶ Actor B ID
- ▶ Additional variables:
 - ▶ profiles
 - ▶ actions
 - ▶ ...



KDDCup 2012: track 1

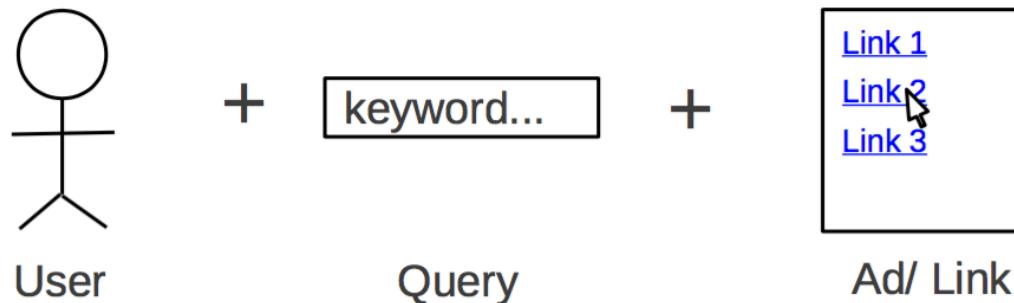


- ▶ $k = 22$ factors, 512 MCMC samples (no burnin phase, initialization from random)
- ▶ MCMC inference (no hyperparameters (learning rate, regularization) to specify)

[Awarded 2nd place (out of 658 teams)]

Predicción de Clicks

- ▶ Main variables:
 - ▶ User ID
 - ▶ Query ID
 - ▶ Ad/ Link ID
- ▶ Additional variables:
 - ▶ query tokens
 - ▶ user profile
 - ▶ ...



KDDCup 2012: Track 2

Model	Inference	wAUC (public)	wAUC (private)
ID-based model ($k = 0$)	SGD	0.78050	0.78086
Attribute-based model ($k = 8$)	MCMC	0.77409	0.77555
Mixed model ($k = 8$)	SGD	0.79011	0.79321
Final ensemble	n/a	0.79857	0.80178

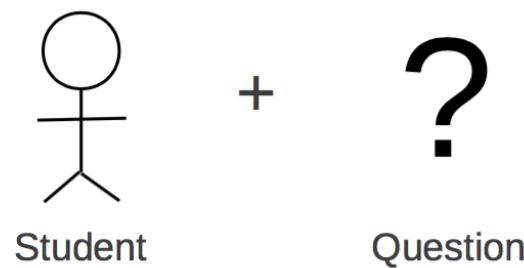
Ensemble

- ▶ Rank positions (not predicted clickthrough rates) are used.
- ▶ The MCMC attribute-based model and different variations of the SGD models are included.

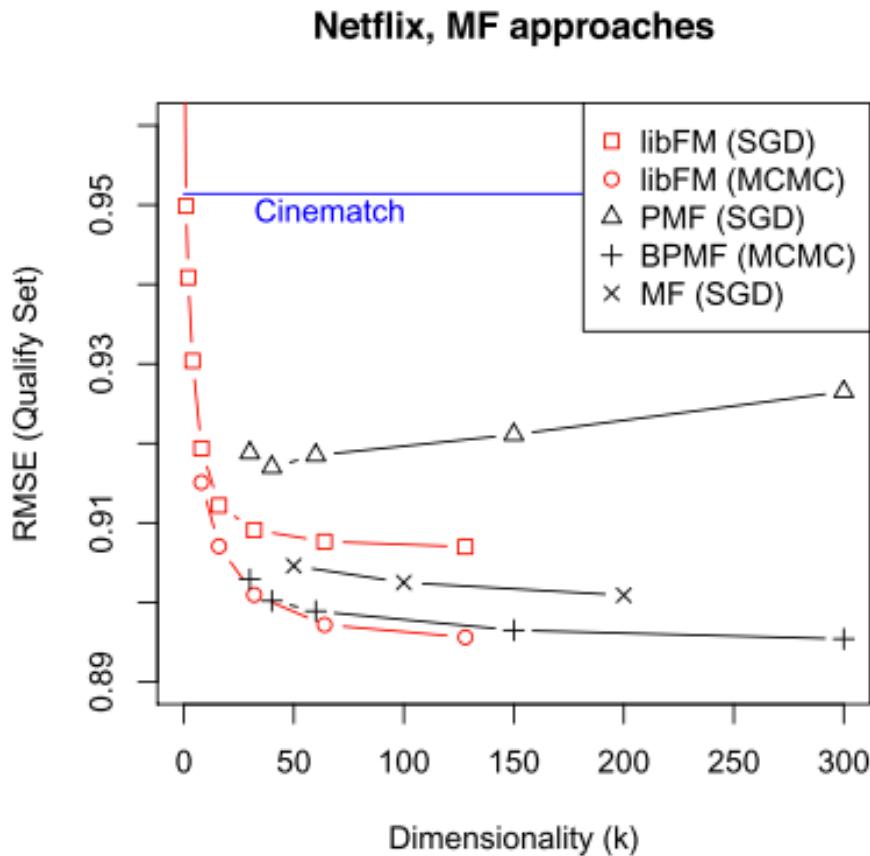
[Awarded 3rd place (out of 171 teams)]

Predecir Resultados de Estudiantes

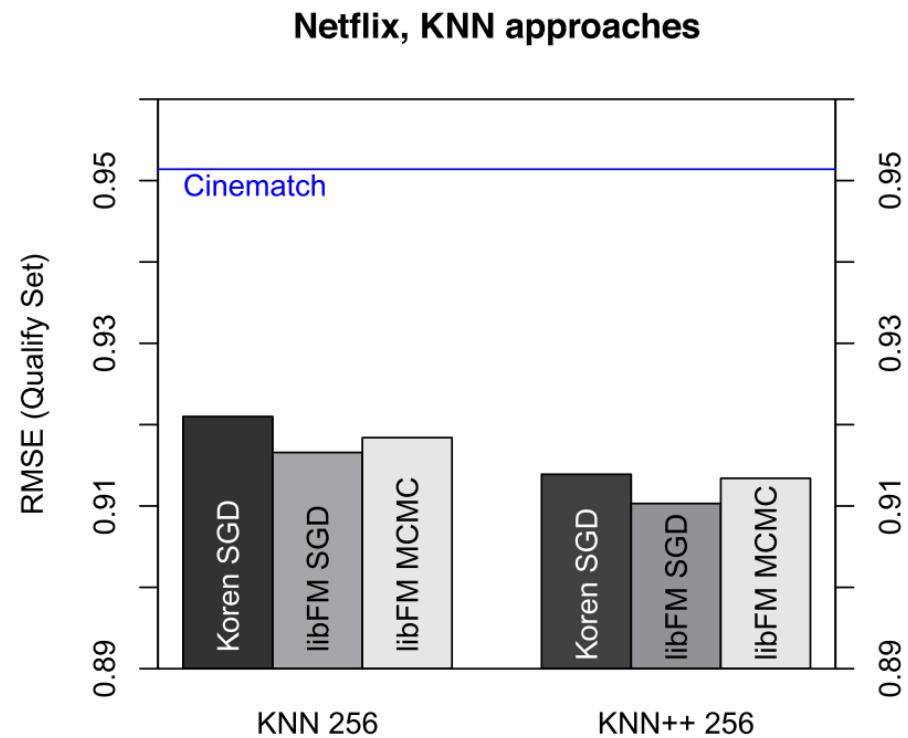
- ▶ Main variables:
 - ▶ Student ID
 - ▶ Question ID
- ▶ Additional variables:
 - ▶ question hierarchy
 - ▶ sequence of questions
 - ▶ skills required
 - ▶ ...
- ▶ Examples: KDDCup 2010, Grockit Challenge⁴ (FM placed 1st/241)



Algunos Resultados

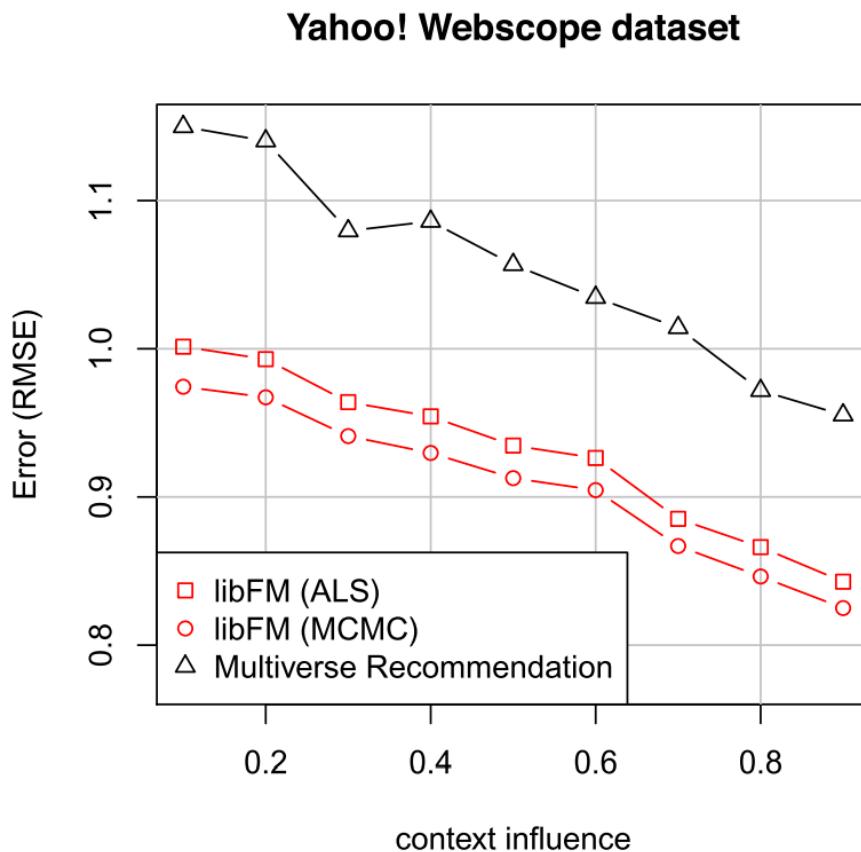


(a) Matrix factorization (MF).

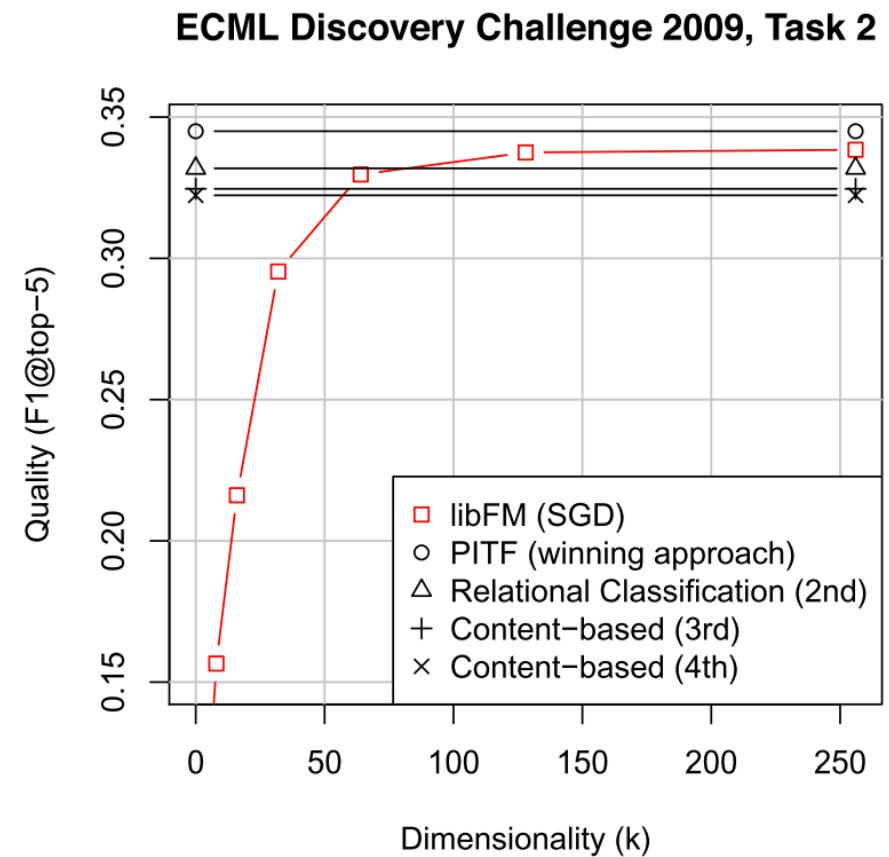


(b) Nearest neighborhood (KNN).

Algunos Resultados II



(a) Context-aware recommendation.



(b) Tag recommendation.

Using Libfm

- Llamada 1:

```
./libFM -task r -train ml1m-train -test ml1m-test -dim '1,1,8'
```

- Llamada 2:

```
./libFM -task r -train ml1m-train.libfm -test ml1m-test.libfm -dim '1,1,8' -iter 1000  
-method sgd -learn_rate 0.01 -regular '0,0,0.01' -init_stdev 0.1
```

$$X = \begin{pmatrix} 1.5 & 0.0 & 0.0 & -7.9 & 0.0 & 0.0 & 0.0 \\ 0.0 & 10^{-5} & 0.0 & 2.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix}, \quad y = \begin{pmatrix} 4 \\ 2 \\ -1 \end{pmatrix}$$

Example

```
4 0:1.5 3:-7.9  
2 1:1e-5 3:2  
-1 6:1  
...
```

Ejemplo con libFMexe

- Wrapper de LibFM para R

```
library(libFMexe)

data(movie_lens)

set.seed(1)
train_rows = sample.int(nrow(movie_lens), nrow(movie_lens) * 2 / 3)
train = movie_lens[train_rows, ]
test = movie_lens[-train_rows, ]

predFM = libFM(train, test, Rating ~ User + Movie, task = "r", dim = 10, iter = 300,
exe_loc = "/Users/denisparra/libfm-1.42.src/bin/")

head(predFM)
# How good is RMSE ?
mean((predFM - test$Rating)^2)
```

Conclusiones

- FMs combinan regresión lineal/polinomial con modelos de factorización.
- Interacción entre variables se aprenden vía representación low-rank.
- Es posible la estimación de observaciones no observadas.
- Se pueden calcular eficientemente y tienen una buena calidad de predicción.

Referencias

- **Rendle, S. (2010) “Factorization Machines”**
(<https://www.ismll.uni-hildesheim.de/pub/pdfs/Rendle2010FM.pdf>)
- <http://www.slideshare.net/hongliangjie1/libfm>
- <http://www.slideshare.net/SessionsEvents/steffen-rendle-research-scientist-google-at-mlconf-sf>
- http://www.slideshare.net/0x001/intro-to-factorization-machines?next_slideshow=1

Proyecto Final curso RecSys 2014

- Trade-offs Between Implicit Feedback and Context-Aware Recommendation
 - Santiago Larraín, PUC Chile
 - Nicolás Risso, PUC Chile
- Moviecity Dataset

Proyecto Final curso RecSys 2014

- Moviecity

Columna	Descripción
user_id	Identificador de usuario único
version_id	Identificador único de contenido
user_watchinglist_time_minutes_spent	Consumo acumulado en minutos
DURATION_MINUTES	Largo del contenido
account_country_code	Código de país del usuario
country_description	IdentificadorNombre del país
country_region	Región geográfica
Kids	Marca de si el contenido es para kids o no
Genre	Genero del contenido
Subgenre	Genero primario del contenido

Dataset Moviecity

Mes	Cantidad de usuarios	Cantidad de items	Rmin	Rmax	Ravg
Junio	95013	1679	0	33.94	0.26
Julio	83924	1612	0	38.20	0.34
Agosto	95013	1679	0	33.94	0.26
Total	191657	1918	0	38.20	0.32

Month	Row count	User count	Item count
June	407.078	95.013	1.679
July	482.772	83.924	1.612
August	548.419	95.013	1.668
Total	1.438.269	191.657	1.918

Table 2: Dataset statistics by month

Dataset MovieCity II

Géneros	Subgéneros	Países	Zonas geográficas
Kids	Animation	Mexico	N
Pelicula	Family	Argentina	S
Serie	Thriller	Peru	SA
Movies And Features	Comedy	Colombia	C
Anime	Adventure	Chile	
Documental	Drama	Uruguay	
	Documentary	Venezuela	
	Horror	Rep. Dominicana	
	Action	Honduras	
	Classics	Panama	
	Musical	El Salvador	
	Science Fiction	Guatemala	
	Western	Bolivia	
		Costa Rica	
		Nicaragua	
		Paraguay	

Métodos I

- Hu and Koren ~ Implicit Feedback

$$p_{u,i} = \begin{cases} 1 & \text{if } r_{u,i} > 0 \\ 0 & \text{other case} \end{cases} \quad c_{u,i} = 1 + \alpha r_{u,i}$$

$$\min_{x^*, y^*} \sum_{u, i} c_{u,i} (p_{u,i} - \vec{x}_u^T \vec{y}_i)^2 + \lambda \left(\sum_u \|\vec{x}_u\|^2 + \sum_i \|\vec{y}_i\|^2 \right)$$

$$\vec{x}_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u p(u) \tag{4}$$

$$\vec{y}_u = (X^T C^i X + \lambda I)^{-1} X^T C^i p(i) \tag{5}$$

Where $C^u \in \mathbb{R}^{U \times k} : C_{i,i}^u = c_{u,i}$ and $C^i \in \mathbb{R}^{I \times k} : C_{u,u}^i = c_{u,i}$

Métodos II

- Factorización Tensorial (usando HOSVD)

If we define $\mathbf{P} \in \mathbb{R}^{U \times I \times C_1 \dots \times C_N}$, we can decompose the original data as follows:

$$\mathbf{P} = S \times_U U^{(U)} \times_I U^{(I)} \times_{C_1} \dots \times_N U(C_N) \quad (6)$$

Where $U^{(n)} \in \mathbb{R}^{n \times n}$ is a n dimension tensor and \times_n is a tensor product in dimension n .

Métodos III

- Factorization Machines, Rendle (2010)

$$y(x) := w_0 + \sum_{i=1} w_i x_i + \sum_{i=1} \sum_{j=i+1} (\vec{v}_i \cdot \vec{v}_j) x_i x_j$$

$$y(x) := w_0 + \sum_{i=1} w_i x_i + \sum_{l=1} \sum_{i_l=1} \cdots \sum_{i_l=i_{l-1}+1} \left(\prod_{j=1} x_{i_j} \right) \left(\sum_{f=1} \prod_{j=1} v_{i_j} f \right)$$

Métricas de Evaluación

- RMSE: Diferencia de tiempo entre programa visto y lo predicho

$$1. \ RMSE = \sqrt{\sum_{(u,i) \in Train} (r_{u,i} - \hat{r}_{u,i})^2}$$

$$2. \ MAE = \sum_{(u,i) \in Train} |r_{u,i} - \hat{r}_{u,i}|$$

$$3. \ \overline{rank} = \frac{\sum_{(u,i) \in Train} r_{u,i} rank_{u,i}}{\sum_{(u,i) \in Test} r_{u,i}}$$

Comparación de los Modelos

Model	RMSE	MAE	rank
Matrix factorization	0.7404	0.4820	0.1334
Tensor factorization	1.0024	0.7553	0.5117
Factorization machine	0.6105	0.4148	0.4092

- Matrix Factorization: $k = 40$; $\lambda = 75$; $\alpha = 60$.
- Tensor Factorization: $k = 40$.
- Factorization Machine: $k = 10$.

Conclusiones

- Error de MAE entre 40% y 70%: diferencia promedio entre el tiempo predicho y el tiempo que el usuario realmente vio. Mejor método es Factorization Machines, indicando que para esta tarea el contexto ayuda.
- Ranking: el mejor método es Implicit Feedback recommender. Extrañamente, esto indica que para rankear, el mejor método no requiere contexto.