

Sistemi di Raccomandazione

Sommario

- Introduzione
- Una panoramica sulle tecniche
 - Content-based
 - Collaborative Filtering
 - Raccomandazione Graph-based
 - Metodi Ibridi
 - Latent Factor Models

Parte I

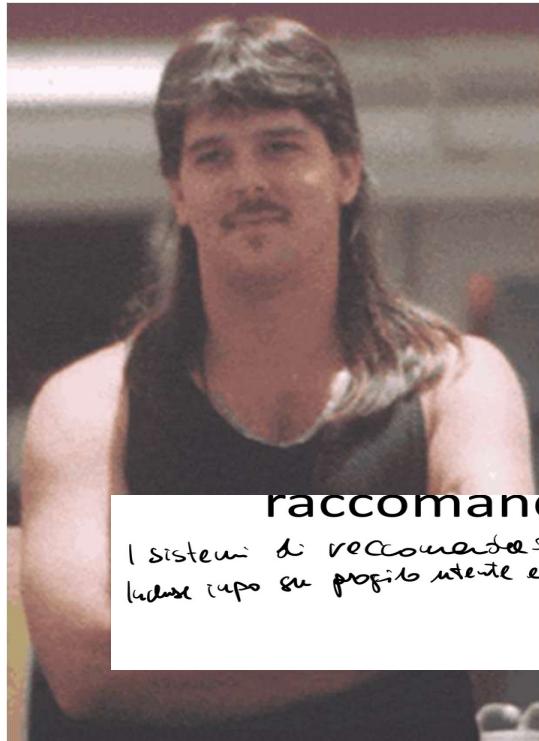
Introduzione

Introduzione

- Un **sistema di raccomandazione** è una classe di applicazioni (comunemente web-based) che predicono le **risposte** degli **utenti** sulla base delle loro **preferenze**.
- Due esempi:
 - Suggerire articoli ai lettori dei quotidiani on-line;
 - Offrire ai clienti dei siti di e-commerce suggerimenti su cosa potrebbe essere di loro interesse.

Sistema di raccomandazione → Tecnologia web che predice il comportamento degli utenti a partire dalle loro preferenze note e li utilizza per suggerire articoli nei giornali e possibili articoli da acquistare nell'e-commerce

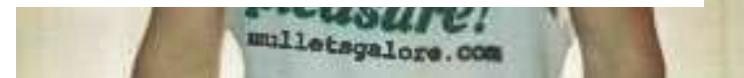
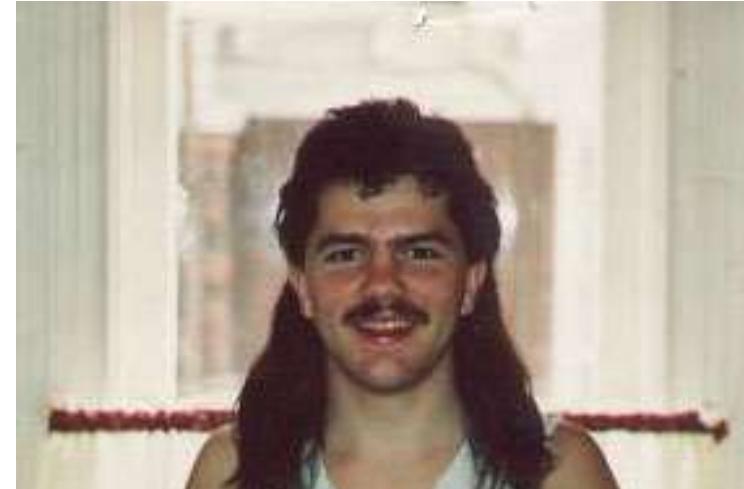
Esempio di sistema di raccomandazione



raccomandazione!

I sistemi di raccomandazione possono utilizzare **tutti i tipi di informazione**.
Incluso i dati sui profili utente e attività compiute sul web. Le rance di Detallece parte questo

↳ Active Recommendation!



• Cliente X

- Acquista CD Metallica
- Acquista CD Megadeth

Utenti con preferenze simili possono essere utilizzati per raccomandare oggetti o utenti
altri utenti con preferenze inizialmente simili

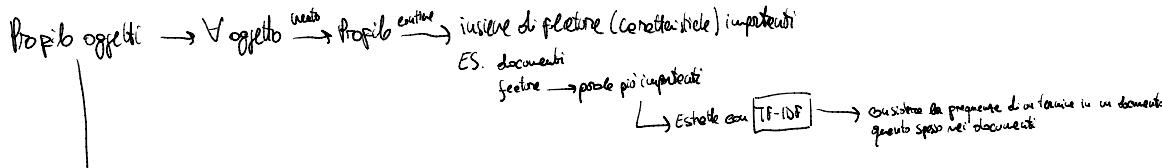
• Cliente Y

- Fa una ricerca sui Metallica
- Il sistema di raccomandazione suggerisce Megadeth in base ai dati collezionati sul cliente X

Le metodologie

- Diverse tecnologie ma due gruppi principali:
 - **Content-based systems:** esaminare le proprietà degli articoli per raccomandarne nuovi.
 - **Collaborative filtering systems:** usare misure di similarità tra utenti e/o prodotti per raccomandare nuovi prodotti (oggetti simili o di proprietà di utenti simili).

Content based system = Si concentra sulle proprietà intrinseche degli oggetti. L'idea fondamentale è che gli oggetti raccomandati a un utente saranno simili agli oggetti che l'utente ha valutato positivamente in passato. Questa similarità viene determinata misurando la somiglianza tra le proprietà degli oggetti stessi.



Proprietà utente \rightarrow Utente \rightarrow oggetti che ha apprezzato.
Potrebbe essere una mappa dei profili di questi oggetti

Euristiche di classificazione \rightarrow Sistema calcola le probabilità che utente apprezzi oggetto.
Sicurezza del sistema

PRO

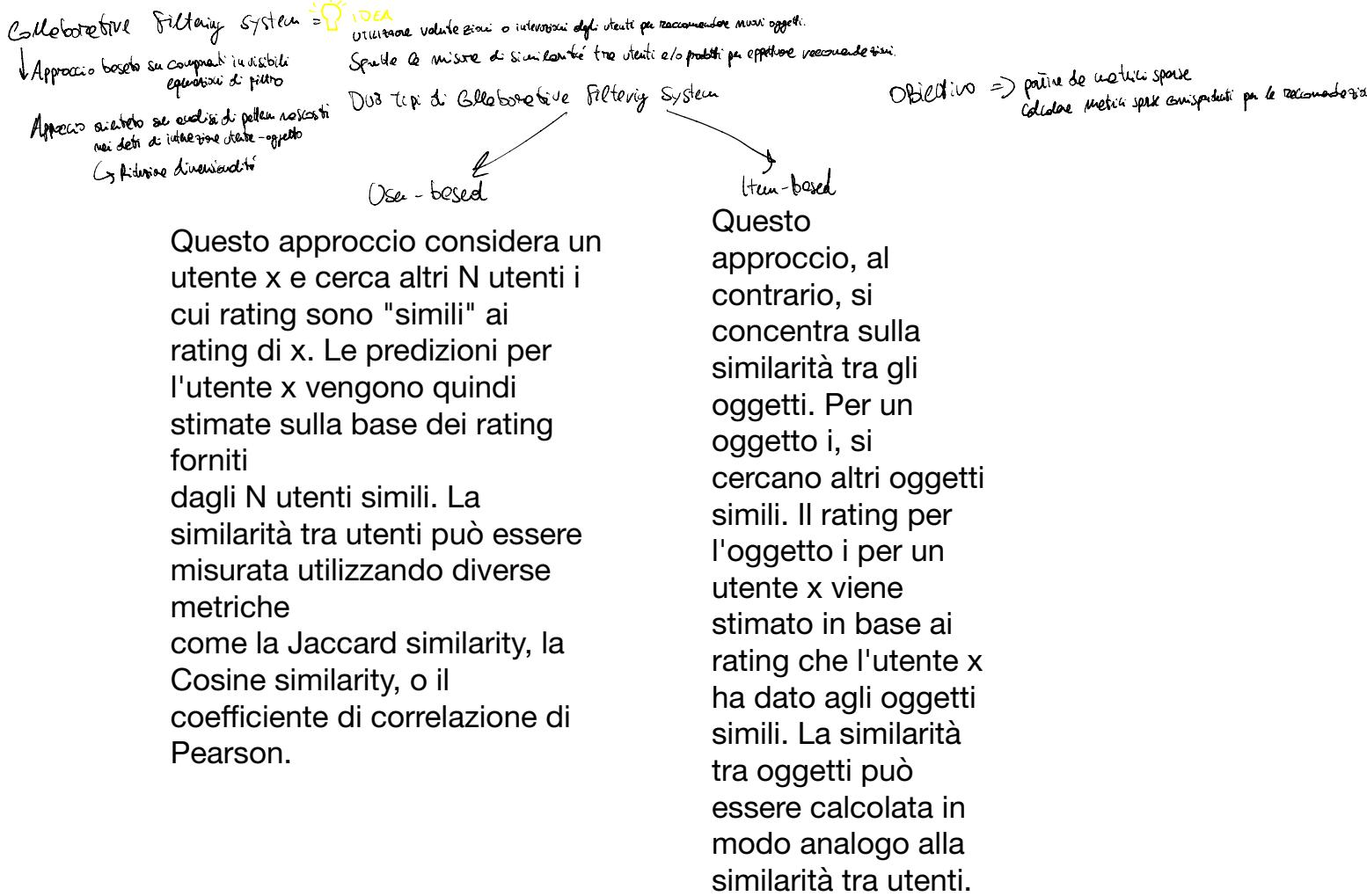
Non necessita di dati su altri utenti

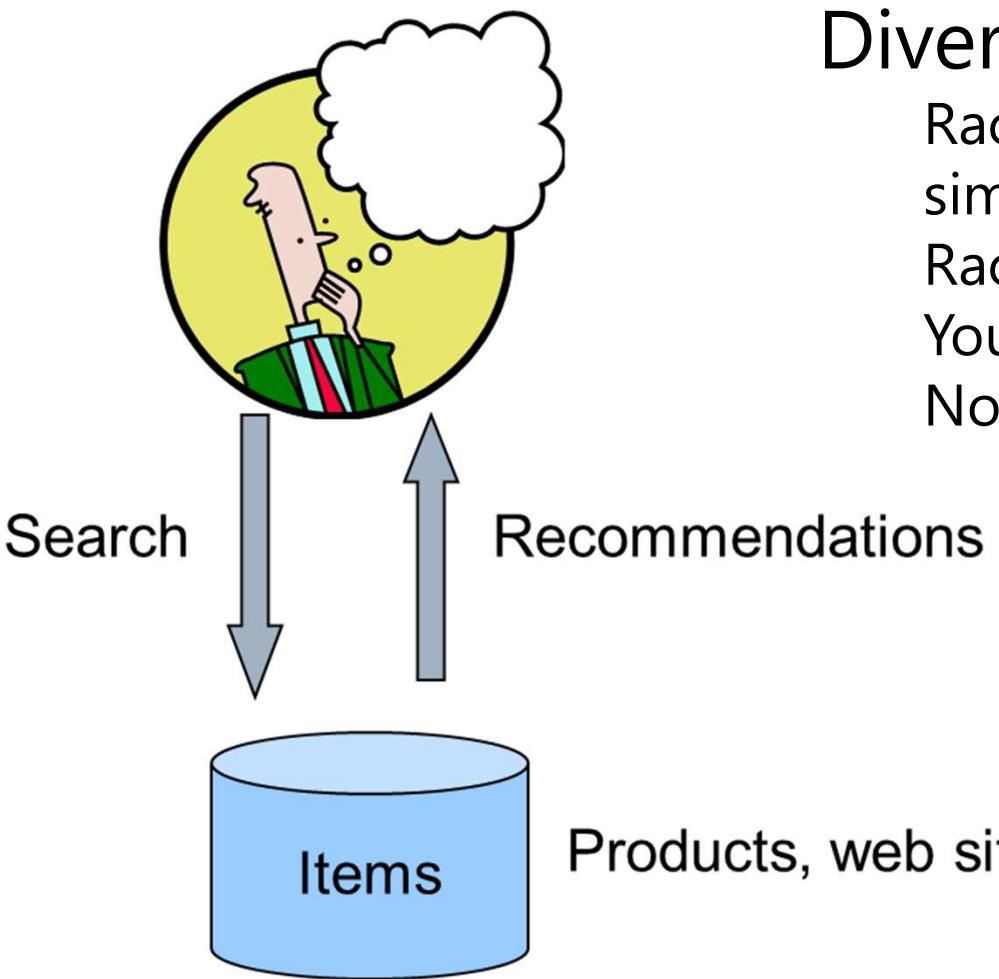
Capacità di raccomandare a utenti con gusti simili \rightarrow raccomandare articoli specifici x interessi specifici

Capacità di raccomandare articoli nuovi e non popolari

CONTRARIO

Difficoltà nel trovare feature giuste \rightarrow Nuovo utente
 \hookrightarrow limitazione
 \hookrightarrow No valutazioni
prese





Diversi campi di applicazione:

Raccomandazione prodotti: Amazon o simili;
Raccomandazione film: Netflix o YouTube;
Notizie: quotidiani o blog.

Definizione

- Sistema di raccomandazione ha **due classi di entità**:

- Un insieme di **Utenti** ($U=\{u_1, \dots, u_n\}$)
- Un insieme di **Oggetti** ($O=\{o_1, \dots, o_m\}$)

rappresentate all'interno di una **utility matrix** ($U_m = \{d_{uo}\}_{n \times m}$)

- **matrice sparsa** che per ogni coppia utente-oggetto calcola un grado di preferenza dell'utente per l'oggetto.

righe utenti
colonne oggetti

→ valori intascabili

Matrice sparsa nelle entry = 0
=> utenti intere fanno con pochi oggetti

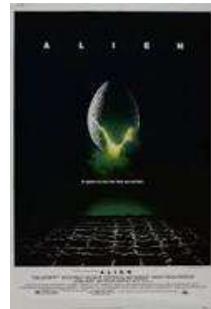
- Obiettivo del sistema di raccomandazione: **predire le entry vuote della matrice** per inferire le preferenze dell'utente.

Partire da Matrici sparse
Calcolare rappresentazioni più dense
per poter effettuare predizioni più efficaci → Scansione gioco buco nelle festine sparse

tecniche Scansione matriciale
Scansione a basso rango

Esempio

- La matrice rappresenta il rating che l'utente da al film visto usando la scala **1-5**, con **5** rating più alto.
- **Zero** indica che l'utente non ha valutato il film.



	MATRIX	ALIEN	STAR WARS	CASABLANCA	TITANIC
Joe	1	1	1	0	0
Jim	3	0	3	0	0
John	4	4	4	0	0
Jack	0	5	5	0	0
Jill	0	0	3	4	4
Jenny	0	0	0	5	5
Jane	3	0	0	Potremmo disegnare un sistema per prendere le proprietà dei film per predire le preferenze degli utenti anche se non hanno visto il film.	

Potremmo disegnare un sistema per prendere le proprietà dei film per predire le preferenze degli utenti anche se non hanno visto il film.

Il fenomeno della Long tail

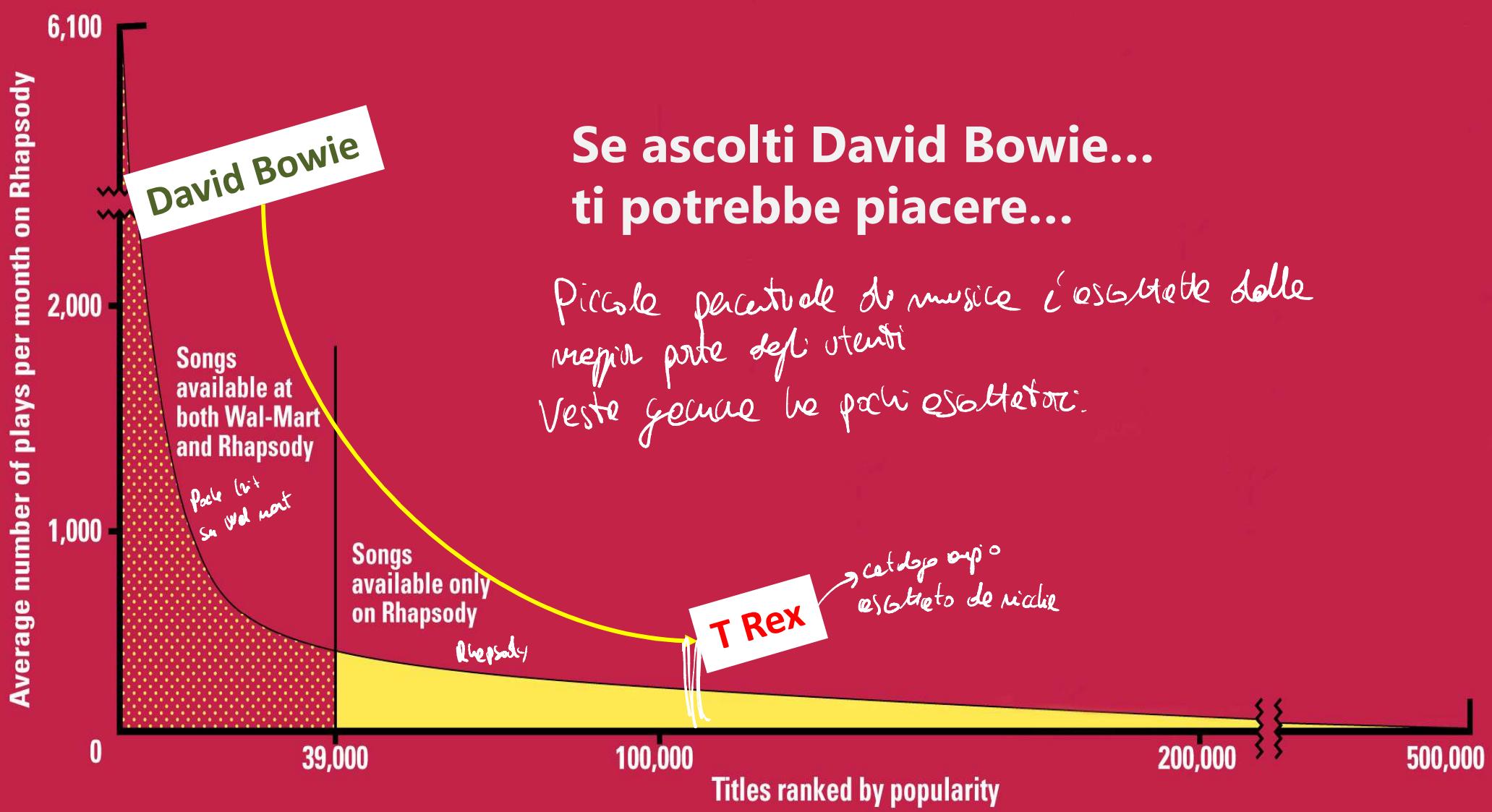
- Negozi fisici: **spazio limitato** sugli scaffali, mostrano piccola frazione di prodotti (i più popolari).
- Web: **non ha questo problema**, costi per pubblicizzare i prodotti vicini allo zero...
- Più scelta per gli utenti, ma... vanno aiutati!
- Ci servono **filtri migliori**... usiamo i sistemi di raccomandazione!

I sistemi di raccomandazione possono utilizzare **tutti i tipi di informazione**.
Individuano su profilo utente e azioni compiute sul web. Le riconoscono e suggeriscono cose che gli piacciono.

↳ Active Recommendation!

Long tail

Maggior parte utenti interagisce con una piccola percentuale degli articoli più popolari.
resto numero di articoli riceve percentuale minore.
Compito sistema raccomandazione → Soprattutto articoli meno popolari ma potenzialmente likavati per specifici utenti.



Problemi chiave

- **Collezionare** valutazioni “conosciute” per la utility matrix.
- **Predire** valutazioni **nuove ed elevate** a partire dalle valutazioni conosciute.

Come ottenere le valutazioni (rating)

- Costruire una utility matrix è un **task complicato**, due approcci per scoprire il valore che un utente dà ad un prodotto:
 - **Esplicito:**
 - Chiedere di valutare un item (es. YouTube, ecc.).
 - **Implicito:**
 - Apprendere dalle azioni dell'utente.

Estrapolare la utility matrix e raccomandare

- Descriviamo gli approcci principali:
 - Content-based
 - Collaborative Filtering
 - Graph-based
 - Hybrid Approach
 - Latent factor models

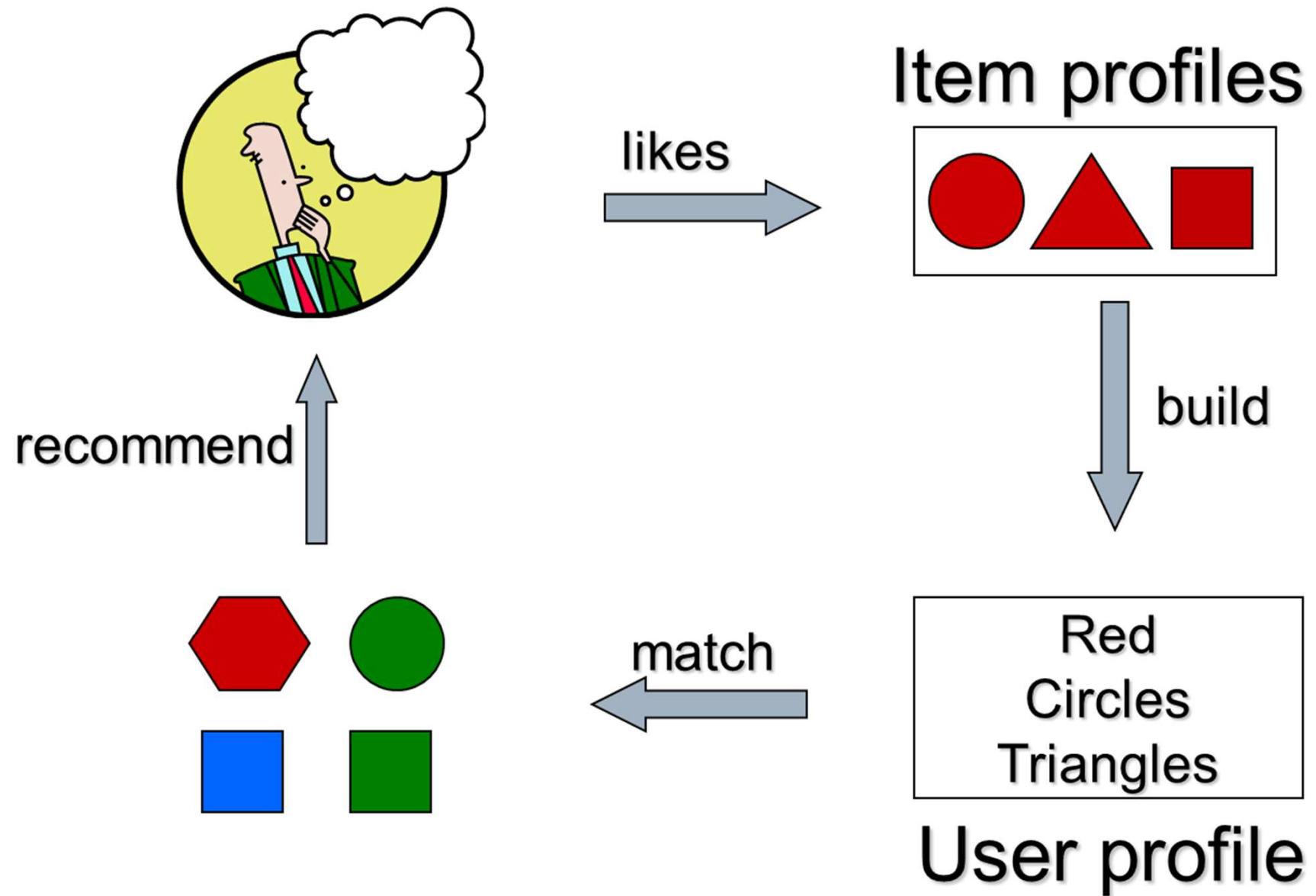
Parte II

Content-based recommendation

Raccomandazioni Content-based

- **Idea principale:** gli oggetti raccomandati all’utente U sono **simili** agli oggetti valutati positivamente da U.
- Il sistema focalizza sulle **proprietà degli oggetti.**
 - La **similarità** tra due oggetti è determinata misurando la *similarità delle loro proprietà.*

Schema del metodo



Il profilo degli oggetti

- Per ogni oggetto si crea un profilo;
- Un **Profilo** è un **insieme di feature** che rappresentano caratteristiche importanti.

- **Esempio**

- I **documenti** sono una classe di oggetti per i quali non è semplice definire quali debbano essere le feature.
- Usare **le parole più importanti** di un documento.
- Come scegliamo le parole importanti?
 - Tipica euristica **TF.IDF (Term Frequency times Inverse Doc Frequency)**

TF.IDF

- “ f_{ij} ” frequenza di del termine “i” nel documento “ d_j ”

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

- “ n_i ” numero di documenti che menzionano il termine “i”,
- N il numero totale di documenti,

$$IDF_i = \log \frac{N}{n_i}$$

$$\text{TF.IDF: } w_{ij} = TF_{ij} \cdot IDF_i$$

- **Profilo di un documento:** insieme delle parole che hanno il più alto **TF.IDF score**, assieme ai loro score.

Profili utente e predizione

- **User profile, diverse possibilità:**

- Media ponderata dei profili degli oggetti;
- Pesare rispetto alla differenza dal rating medio dato dall'utente
- ...

- **Euristica di predizione:**

- Dato il profilo x *dell'utente* e il profilo i *del prodotto*,

stimiamo $u(x, i) = \cos(x, i) = \frac{x \cdot i}{\|x\| \cdot \|i\|}$

Pro e contro

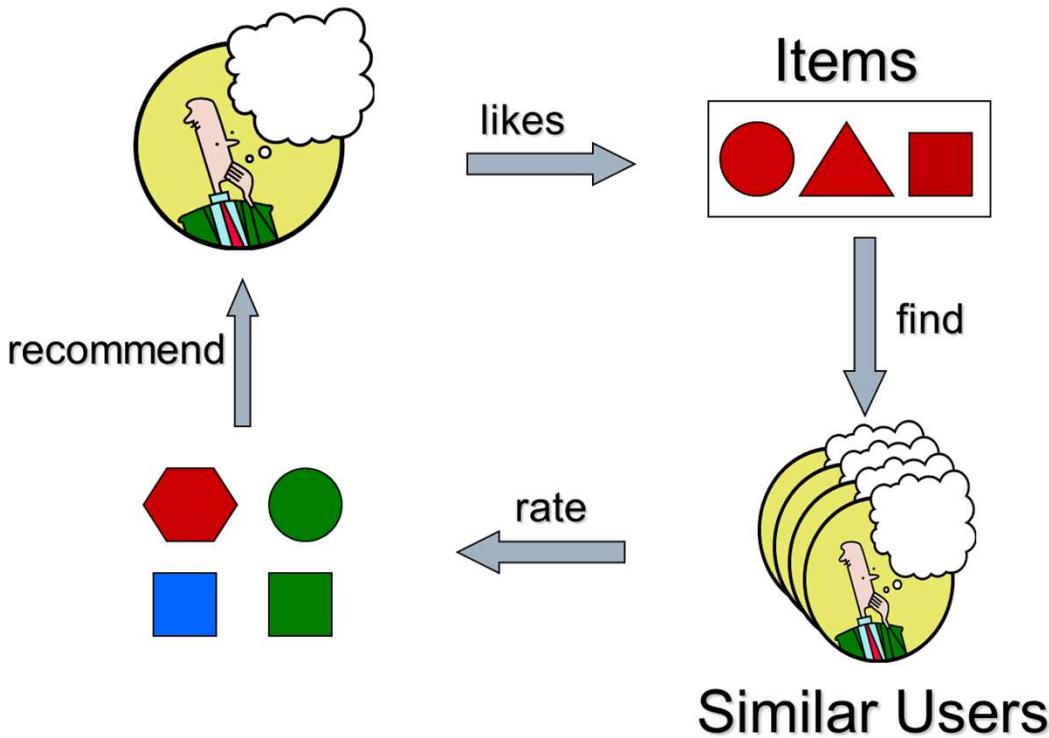
- +: Non c'è bisogno di dati su altri utenti**
- +: In grado di consigliare agli utenti con gusti unici**
- +: In grado di raccomandare articoli nuovi e non popolari**
- +: facile da interpretare**

- : Trovare le feature giuste è difficile**
- : Raccomandazione per nuovi utenti**

Parte III

Collaborative Filtering

Schema del metodo



Consideriamo l'utente
x

Troviamo **N** altri utenti
i cui rating sono
"simili" ai rating di x

Stimiamo i rating di **x**
sulla base degli **N**
rating degli utenti

Come misuriamo la similarità

$$\begin{aligned} \mathbf{r}_x &= [* , _, _, *, **] \\ \mathbf{r}_y &= [* , _, **, **, _] \end{aligned}$$

- **Jaccard similarity:**

$$\text{sim}(x,y) = |\mathbf{r}_x \cap \mathbf{r}_y| / |\mathbf{r}_x \cup \mathbf{r}_y|$$

- **Cosine similarity:**

$$\text{sim}(x,y) = \cos(\mathbf{r}_x, \mathbf{r}_y)$$

- **Pearson correlation coefficient:**

$$\text{sim}(x,y) = \frac{\sum_{s \in S_{xy}} (r_x[s] - \bar{r}_x)(r_y[s] - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_x[s] - \bar{r}_x)^2 (r_y[s] - \bar{r}_y)^2}}$$

Predizioni

Dalla similarità alla raccomandazione:

- Sia r_x il vettore di rating dell'utente x
- Sia N l'insieme dei k utenti più simili a x che hanno valutato l'oggetto i
- **Predizione dell'oggetto i per l'utente x :**
 - $r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$
 - $r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$ $s_{xy} = sim(x, y)$
 - Altre opzioni?
 - Molte..

Oggetto-Oggetto Collaborative Filtering

- **Vista: oggetto-oggetto**

- Troviamo gli oggetti simili a i
- Stimiamo il rating per i in base ai rating degli oggetti simili

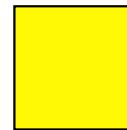
$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij} ... similarità degli oggetti i e j
 r_{xj} ... rating dell'utente x sull'oggetto j
 $N(i;x)$... set di oggetti simili a i valutati da x

Oggetto-Oggetto CF ($|N|=2$)

	utenti											
	1	2	3	4	5	6	7	8	9	10	11	12
film	1	1		3			5			5		4
	2			5	4			4			2	1
	3	2	4		1	2		3		4	3	5
	4		2	4		5			4			2
	5			4	3	4	2					2
	6	1		3		3			2			4

 - Rating sconosciuto

 - rating tra 1 e 5

Oggetto-Oggetto CF ($|N|=2$)

	utenti											
	1	2	3	4	5	6	7	8	9	10	11	12
film	1	1		3		?	5			5		4
	2			5	4			4			2	1
	3	2	4		1	2		3		4	3	5
	4		2	4		5			4			2
	5			4	3	4	2				2	5
	6	1		3		3			2			4



- Stimare il rating del film 1 per l'utente 5

Oggetto-Oggetto CF ($|N|=2$)

	utenti												
	1	2	3	4	5	6	7	8	9	10	11	12	
film	1	1		3		?	5			5		4	
	2			5	4			4			2	1	3
	3	2	4		1	2		3		4	3	5	
	4		2	4		5			4			2	
	5			4	3	4	2					2	5
	6	1		3		3			2			4	

sim(1,m)
 1.00
 -0.18
0.41
 -0.10
 -0.31
0.59

Neighbor selection:

Identificare film simili al film 1,
valutati dall'utente 5

Usiamo la Pearson correlation come :

- 1) Sottraiamo il rating medio m_i da ogni film i
 $m_1 = (1+3+5+5+4)/5 = 3.6$
row 1: [-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]
- 2) Calcoliamo la cosine similarity

Oggetto-Oggetto CF ($|N|=2$)

	utenti												
	1	2	3	4	5	6	7	8	9	10	11	12	
film	1	1		3		?	5			5		4	
	2			5	4			4			2	1	3
	3	2	4		1	2		3		4	3	5	
	4		2	4		5			4			2	
	5			4	3	4	2					2	5
	6	1		3		3			2			4	

$\text{sim}(1,m)$

1.00

-0.18

0.41

-0.10

-0.31

0.59

Pesi similarità:

$s_{1,3}=0.41, s_{1,6}=0.59$

Oggetto-Oggetto CF ($|N|=2$)

	utenti											
	1	2	3	4	5	6	7	8	9	10	11	12
film	1	1		3		2.6	5			5		4
	2			5	4			4			2	1
	3	2	4		1	2		3		4	3	5
	4		2	4		5			4			2
	5			4	3	4	2					2
	6	1		3		3			2			4

Predizione tramite media pesata :

$$r_{1.5} = (0.41 \cdot 2 + 0.59 \cdot 3) / (0.41 + 0.59) = 2.6$$

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

CF: Approccio comune

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

- Definiamo la **similarità** s_{ij} tra gli oggetti i e j
- Selezioniamo k nearest neighbor $N(i; x)$
 - Oggetti più simili a i , valutati dall'utente x
- Stimiamo il rating r_{xi} pesato con la media:

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$



Stima baseline per r_{xi}

$$b_{xi} = \mu + b_x + b_i$$

- μ = media generale di tutti i film
- b_x = deviazione del rating dalla media dell'utente x = $(\text{avg. rating utente } x) - \mu$
- b_i = deviazione del rating per l'oggetto i

Pro e contro

- +: Lavora con tutti i tipi di oggetti**
- +: Non è necessaria feature selection**
- : New user problem**
- : New item problem**
- : Matrice dei rating sparsa**

Dimensionality Reduction

- Possibile soluzione per il problema dovuto alla **“sparsità” della matrice?**
 - Dimensionality Reduction
- **Latent Semantic Indexing (LSI)**
 - Tecnica algoritmica sviluppata tra la fine degli anni ‘80 primi anni ‘90
 - Risolve problemi di sinonima, “sparsità” e scalabilità per grandi dataset
 - Riduce la dimensionalità e cattura le relazioni latenti
- **Si può mappare facilmente nel Collaborative Filtering!**

- Tecnica di indicizzazione che usa la **Singular Value Decomposition** per identificare pattern nella relazione tra termini e concetti contenuti nel testo.
- Si basa sul principio che parole che sono usate nello stesso contesto tendono ad avere significato simile.

- Matrice Termini-Documenti
- Spazio dei concetti
- Mapping tra:
 - **Termini ↔ Concetti**
 - **Documenti ↔ Concetti**



- Matrice Utenti-Oggetti
- Spazio delle categorie
- Mapping tra:
 - **Oggetti ↔ Categorie**
 - **Utenti ↔ Categorie**

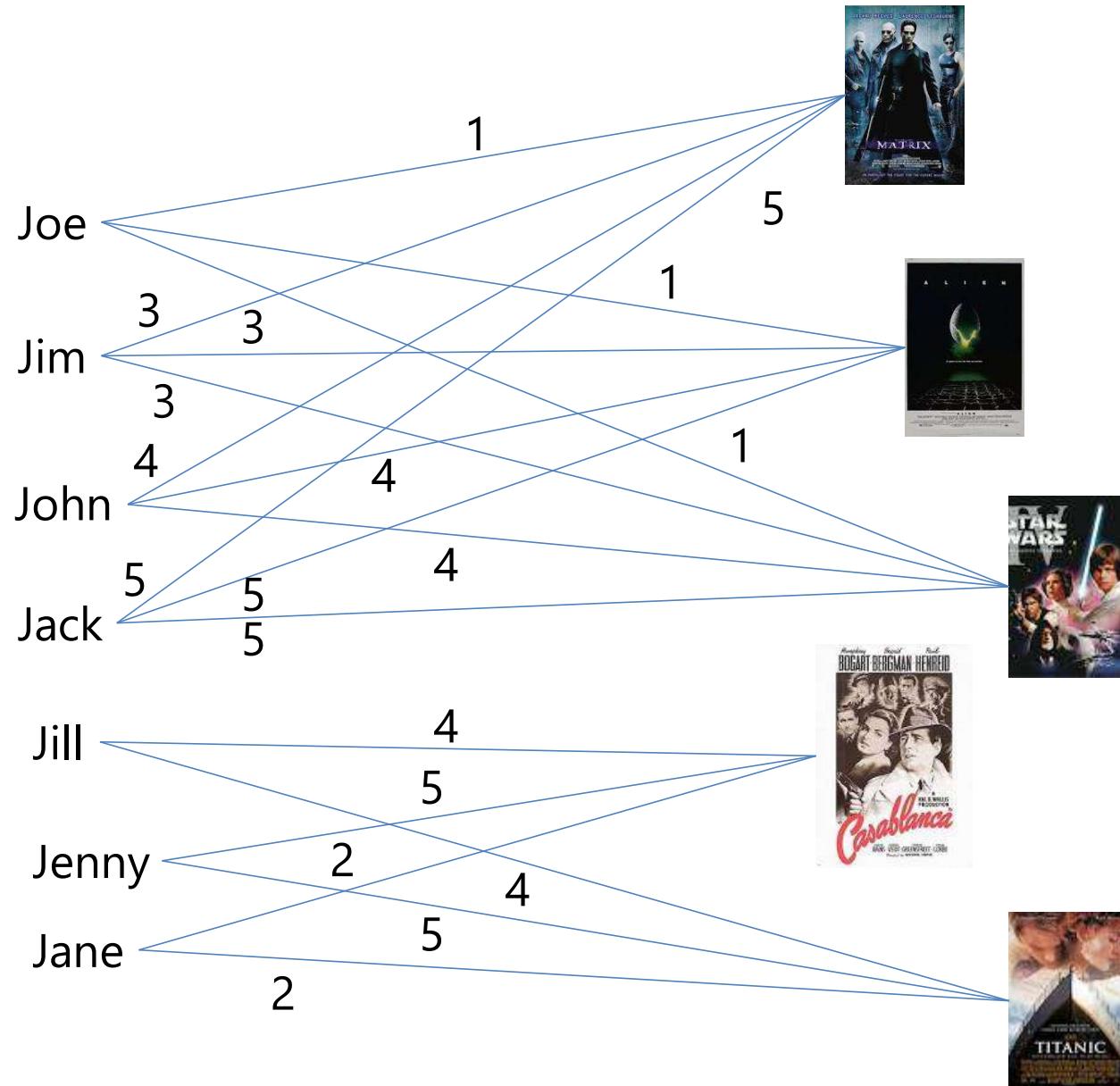
Part IV

Metodi Graph-based

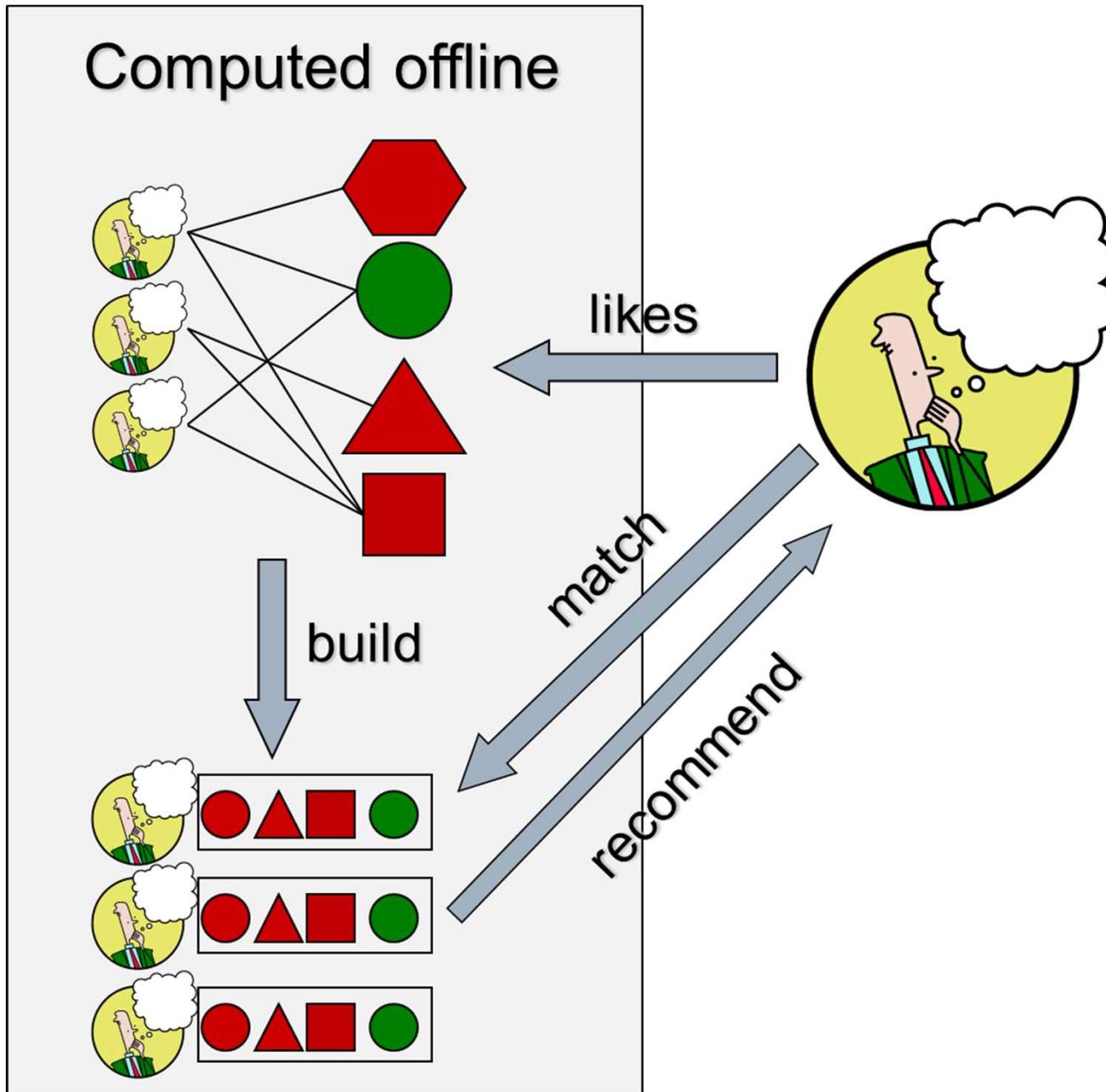
Metodi Graph-based

- Simile al Collaborative Filtering ma usa un **grafo bipartito** per immagazzinare le informazioni.
- Le raccomandazioni sono ottenute a partire dalla struttura della rete bipartita.
- Due classi di identità: **Utenti (U)** e **Oggetti (O)**.
- Definiamo il grafo bipartito come segue:
 - $G(U, O, E, W)$
 - $E = \{e_{ij} : u_i \text{ likes } o_j\}$
 - $W : E \rightarrow \mathbb{R}$
 - “ E ” insieme degli archi e “ W ” è una funzione che rappresenta il degree di preferenza che un utente ha per il particolare oggetto.

La utility matrix come grafo bipartito

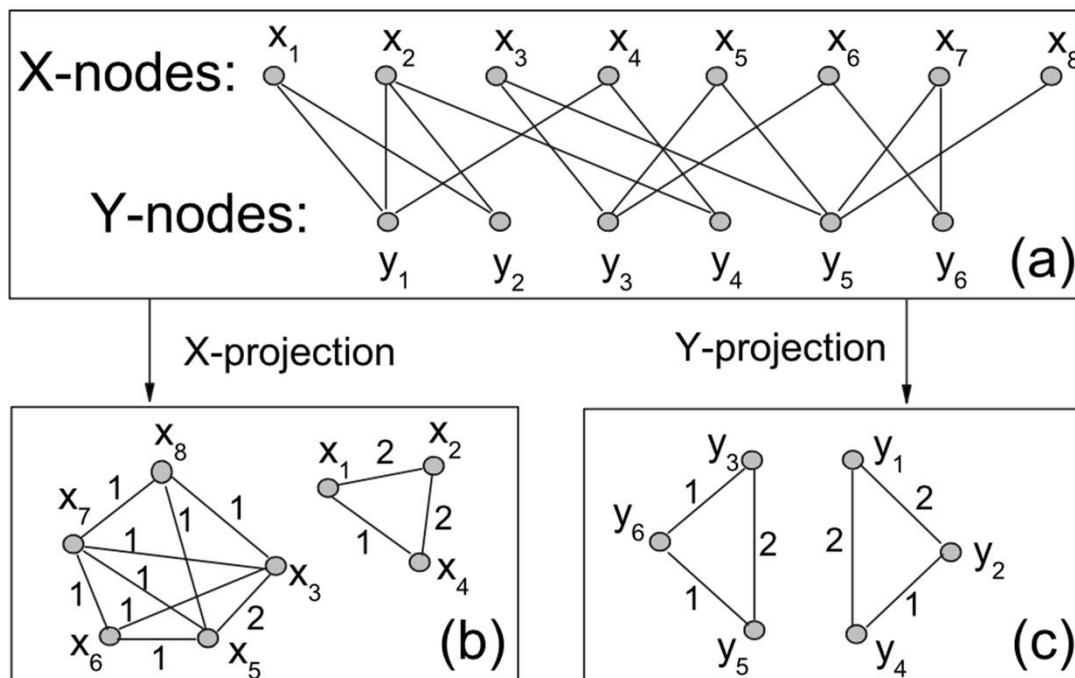


Schema del metodo



NBI

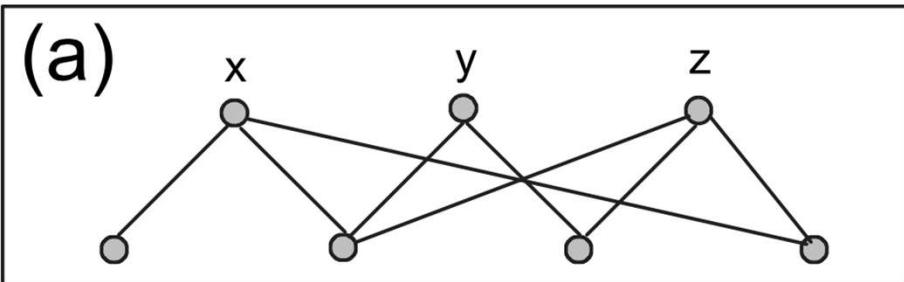
- NBI (Network-based inference) sviluppato nel 2007.
- Bipartite network projection per ottenere informazioni sulle proiezioni.
- Dopo la proiezione:
 - Reti con nodi dello stesso tipo (utenti o oggetti);
 - Due nodi sono connessi se sono collegati da almeno un nodo di altro tipo;



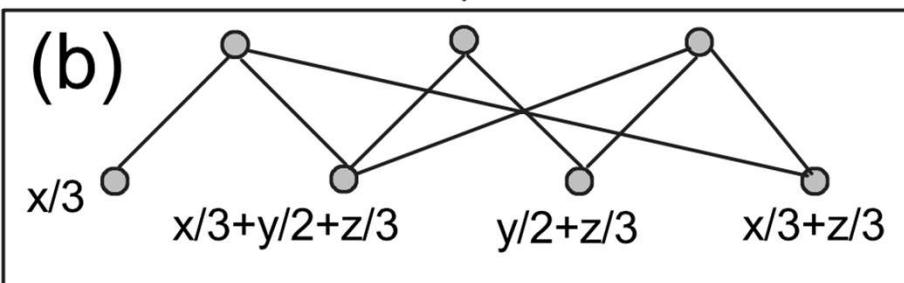
- Idea base dell'algoritmo: flusso delle risorse sulla rete bipartita:

- agli oggetti viene assegnata una quantità iniziale di risorse;
- in un processo a due fasi le risorse sono trasferite dagli oggetti agli utenti e successivamente trasferiti indietro agli oggetti;
- questo processo, con una fase di normalizzazione consente di ottenere degli score per le coppie utenti - oggetti.

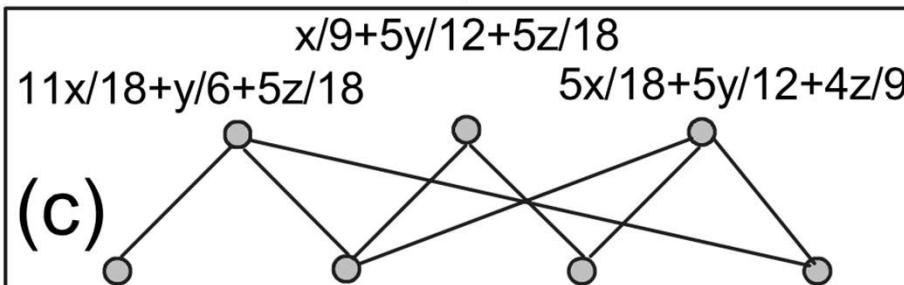
NBI



- Il calcolo dei pesi avviene attraverso la seguente equazione:



- Dove $\Gamma(i, j)$ è definita come :
 - $\Gamma(i, j) = D(o_j)$ per NBI
 - $\Gamma(i, j) = D(o_i)$ per HeatS
- $D(t)$ degree del nodo “t” nella rete bipartita.



Raccomandazioni con NBI

- Il peso “ w_{ij} ” della proiezione corrisponde a quante risorse vengono trasferite dall’oggetto “j” all’oggetto “i”, o quanto piacerà l’oggetto “j” ad un utente a cui piace l’oggetto “i”.
- Data la matrice di adiacenza “A” del grafo bipartito e la matrice “W”, la matrice di raccomandazione “R” per tutti gli utenti può essere calcolata in un unico step:

$$R = W \times A$$

Pro e contro

- +: Funziona su tutti i tipi di oggetti**
- +: Risolve il problema della sparsità della utility matrix.**
- : New user problem**
- : New item problem**
- : Richiede importanti risorse computazionali**

Metodi ibridi

- Usare in combinazione diversi metodi di raccomandazione
- Supponiamo di avere i metodi “X” e “Y” che danno rispettivamente gli score “ x_a ” e “ y_a ”. Lo score di un modello ibrido può essere ottenuto come:

$$z_a = (1 - \lambda) \frac{x_a}{\max_{\beta} x_{\beta}} + \lambda \frac{y_a}{\max_{\beta} y_{\beta}}$$

- $\lambda \in [0;1]$

Pro e contro

- +: Efficace nel migliorare la qualità delle raccomandazioni
- : Computazionalmente pesante

Part VI

Results Evaluation

Evaluating Predictions

- Compare predictions with known ratings

- Root-mean-square error (RMSE)

$$RMSE(\hat{Y}, Y) = \sqrt{\frac{\sum_{i=1}^{|\hat{Y}|} (\hat{Y}_i - Y_i)^2}{|\hat{Y}|}}$$

- Another approach: 0/1 model

- Coverage

- Number of items/users for which system can make predictions

- Precision/Recall

$$\text{Precision} = \frac{tp}{tp + fp}$$

- Accuracy of predictions

$$\text{Recall} = \frac{tp}{tp + fn}$$

- Receiver operating characteristic (ROC)

- Tradeoff curve between false positives and false negatives

The Netflix Prize

- **Training data**

- 100 milioni di rating, 480,000 utenti, 17,770 film
- 6 anni di dati: 2000-2005

- **Test data**

- Ultimi rating degli utenti (2.8 million)

- **Criterio di valutazione:** Root Mean Square Error (RMSE) = $\frac{1}{|R|} \sqrt{\sum_{(i,x) \in R} (\hat{r}_{xi} - r_{xi})^2}$

- **Netflix's system RMSE: 0.9514**

- **Competizione**

- 2,700+ team
- **\$1 million** prize for 10% improvement on Netflix

The Netflix Utility Matrix R

Matrix R

480,000 users					
17,700 movies	1	3	4		
		3	5		5
			4	5	5
				3	
				3	
					2
	2			2	2
					5
		2	1		1
			3		3
	1				

Utility Matrix R : Evaluation

