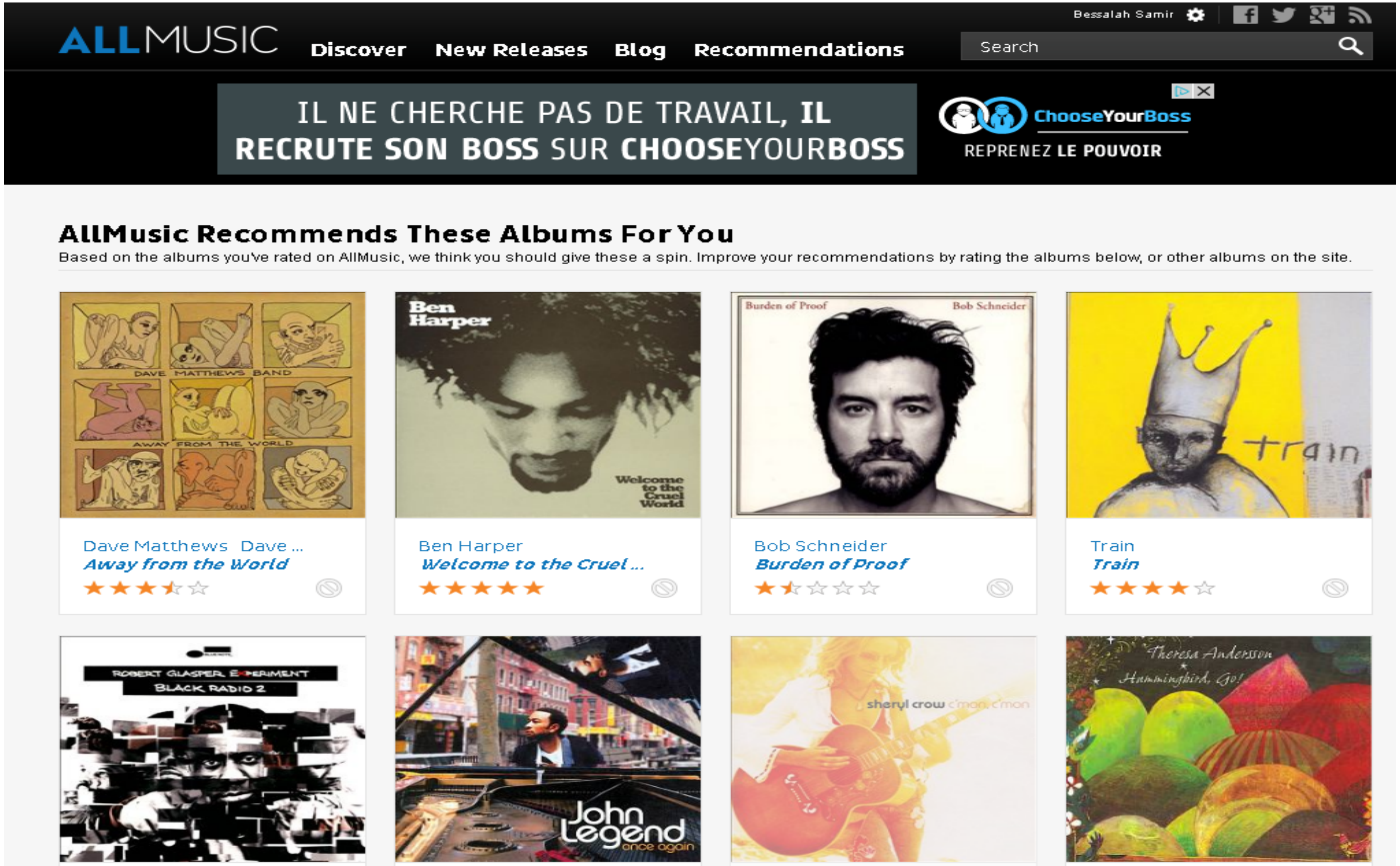
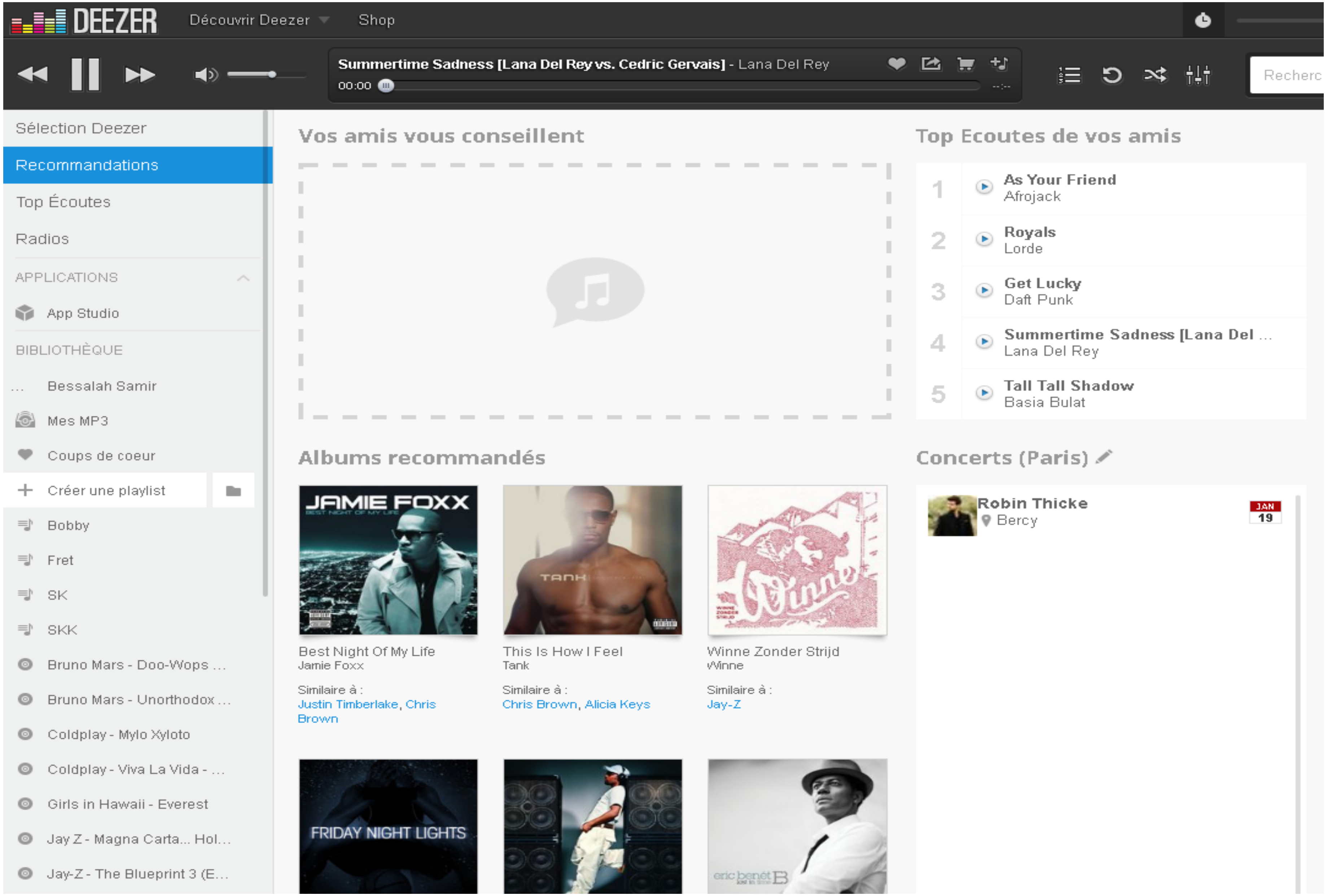
RECOMMENDER SYSTEMS

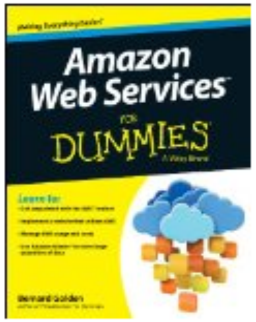
Sam BESSALAH (@samklr)
Software Engineer, Convexity Capital Mngt. (ex Next Capital)

What does a recommender system looks like ?

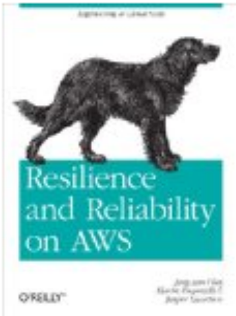

ALLMUSIC
Discover
New Releases
Blog
Recommendations
Bessalah Samir
Search
IL NE CHERCHE PAS DE TRAVAIL, IL RECRUTE SON BOSS SUR CHOOSEYOURBOSS
ChooseYourBoss
REPRENEZ LE POUVOIR
AllMusic Recommends These Albums For You
Based on the albums you've rated on AllMusic, we think you should give these a spin. Improve your recommendations by rating the albums below, or other albums on the site.
Dave Matthews Dave ...
Away from the World
Ben Harper
Welcome to the Cruel ...
Bob Schneider
Burden of Proof
Train
Train


DEEZER
Découvrir Deezer
Shop
Summertime Sadness [Lana Del Rey vs. Cedric Gervais] - Lana Del Rey
00:00
Recherc
Sélection Deezer
Recommandations
Top Écoutes
Radios
APPLICATIONS
App Studio
BIBLIOTHÈQUE
Bessalah Samir
Mes MP3
Coups de coeur
Créer une playlist
Bobby
Fret
SK
SKK
Bruno Mars - Doo-Wops ...
Bruno Mars - Unorthodox ...
Coldplay - Mylo Xyloto
Coldplay - Viva La Vida - ...
Girls in Hawaii - Everest
Jay Z - Magna Carta... Hol...
Jay-Z - The Blueprint 3 (E...
Vos amis vous conseillent
Albums recommandés
JAMIE FOXX
Best Night Of My Life
Jamie Foxx
Similaire à :
Justin Timberlake, Chris Brown
This Is How I Feel
Tank
Similaire à :
Chris Brown, Alicia Keys
Winne Zonder Strijd
Winne
Similaire à :
Jay-Z
FRIDAY NIGHT LIGHTS
Top Ecoutes de vos amis
1 As Your Friend
Afrojack
2 Royals
Lorde
3 Get Lucky
Daft Punk
4 Summertime Sadness [Lana Del ...
Lana Del Rey
5 Tall Tall Shadow
Basia Bulat
Concerts (Paris)
Robin Thicke
Bercy
JAN
19

Recommendations for You in Kindle Store



Amazon Web Services For Dummies
› Bernard Golden
Kindle Edition
(14)
$23.33
Why recommended?



Resilience and Reliability on AWS
Jurg van Vliet, Flavia Paganelli, ...
Kindle Edition
(14)
$17.51
Why recommended?

Cloud Architecture Patterns: Using...
› Bill Wilder
Kindle Edition
(7)
$10.09
Why recommended?

Cloud Computing for Programmers
› Daniele Casal
Kindle Edition
(13)
$5.67
Why recommended?

Big Data and Hadoop
WAGmob
Kindle Edition
(2)
$1.20
Why recommended?

› See more recommendations

Frequently Bought Together

Price for all three: $228.28

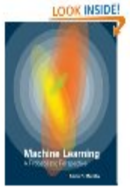
Add all three to Cart | Add all three to Wish List

Show availability and shipping details

- ☑ **This item:** Pattern Recognition and Machine Learning (Information Science and Statistics) by Christopher M. Bishop Hardcover $71.73
- ☑ The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer ... by Trevor Hastie Hardcover $75.55
- ☑ Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series) by Kevin P. Murphy Hardcover $81.00

Customers Who Bought This Item Also Bought

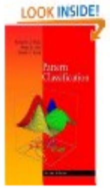
Pa



Machine Learning: A Probabilistic ...
› Kevin P. Murphy
(22)
Hardcover
$81.00

The Elements of Statistical Learning: ...
Trevor Hastie
(28)
Hardcover
$75.55

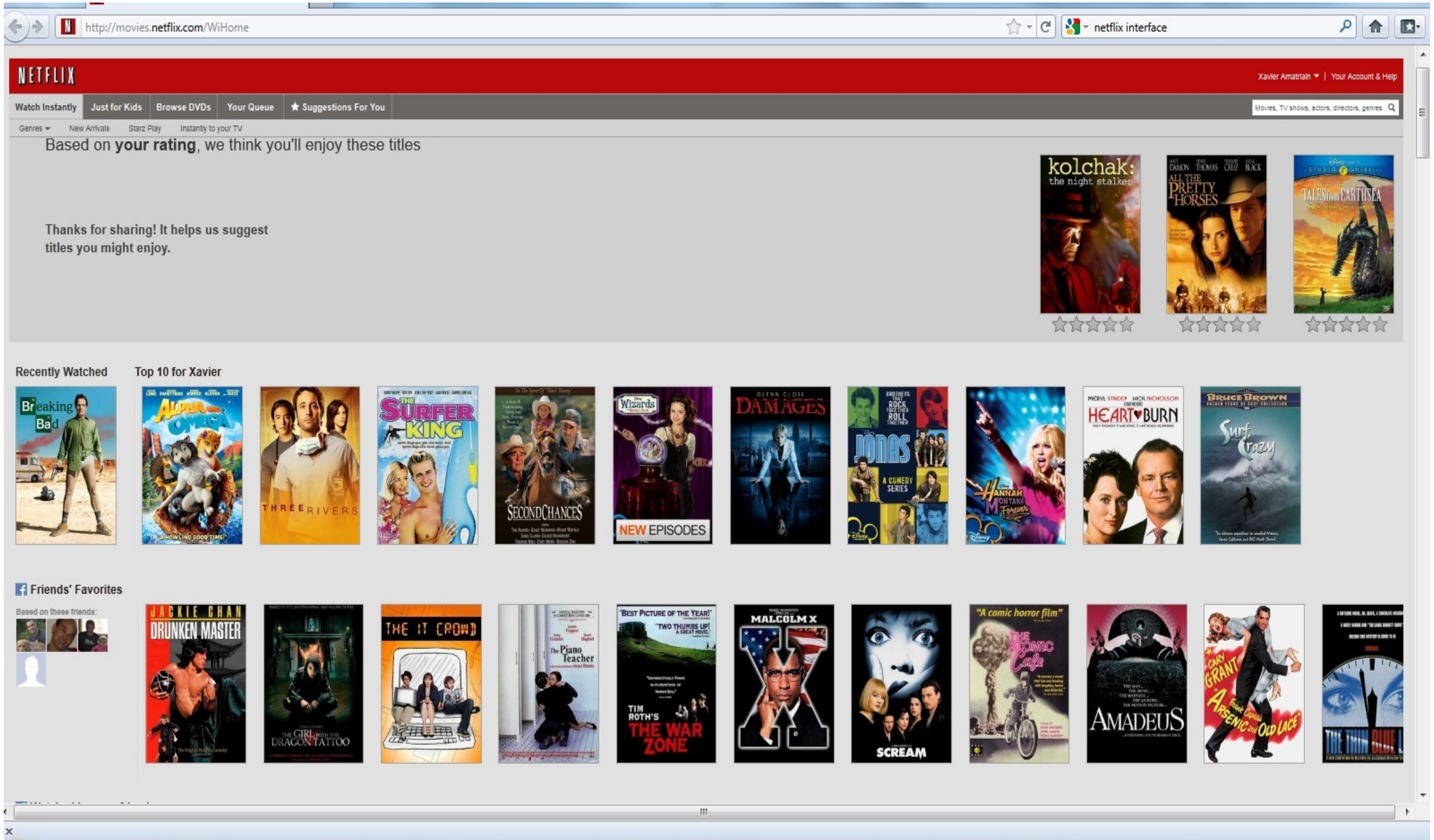
Machine Learning
› Tom M. Mitchell
(45)
Hardcover
$75.98



Pattern Classification (Pt.1)
› Richard O. Duda
(35)
Hardcover
$73.48

Probabilistic Graphical Models: Principles ...
› Daphne Koller
(23)
Hardcover
$92.71

Bayesian Reasoning and Machine Learning
› David Barber
(8)
Hardcover
$59.44



http://movies.netflix.com/WiHome
netflix interface
NETFLIX
Xavier Amatriain
Your Account & Help
Watch Instantly
Just for Kids
Browse DVDs
Your Queue
Suggestions For You
Genres
New Arrivals
Starz Play
Instantly to your TV
Based on your rating, we think you'll enjoy these titles
Thanks for sharing! It helps us suggest
titles you might enjoy.
Recently Watched
Top 10 for Xavier
Friends' Favorites
Based on these friends:

Why a recommender system?

- Help choose among huge choiceof data
- Reduce cognitive load on users
- Drive business revenue

  -**Netflix : 2/3 of the movies watched are recommended**

  -Amazon: 35% sales generated via recommendations

  -Google News : 38% more clicks (CTR) via recommender

BUT HOW IS IT DIFFERENT FROM SEARCH?

Search Engine vs Recommender System

*“ The Web is leaving the era of **search** and entering one of **discovery**. What's the difference?*

***Search** is what you do when you're looking for something.*

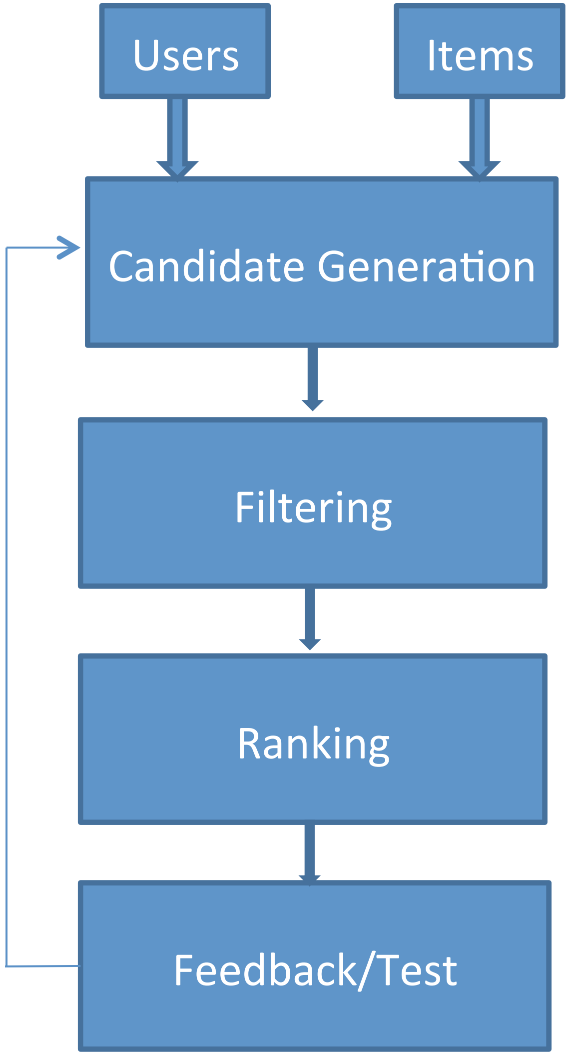
***Discovery** is when something wonderful that you didn't know existed, or didn't know how to ask for, finds you. ”*

CNN Money, “The race to create a 'smart' Google” 2007

http://money.cnn.com/magazines/fortune/fortune_archive/2006/11/27/8394347

How does it work?

High level view of a Rec. Sys



Identify items of interest to the user

Find already seen elements, near duplicates, clean …

Order Recommendations, temporal, diversity, pesonalisation, infer business logic

Tracking, CTR, purchase, A/B Test, online Test, Explore/Exploit

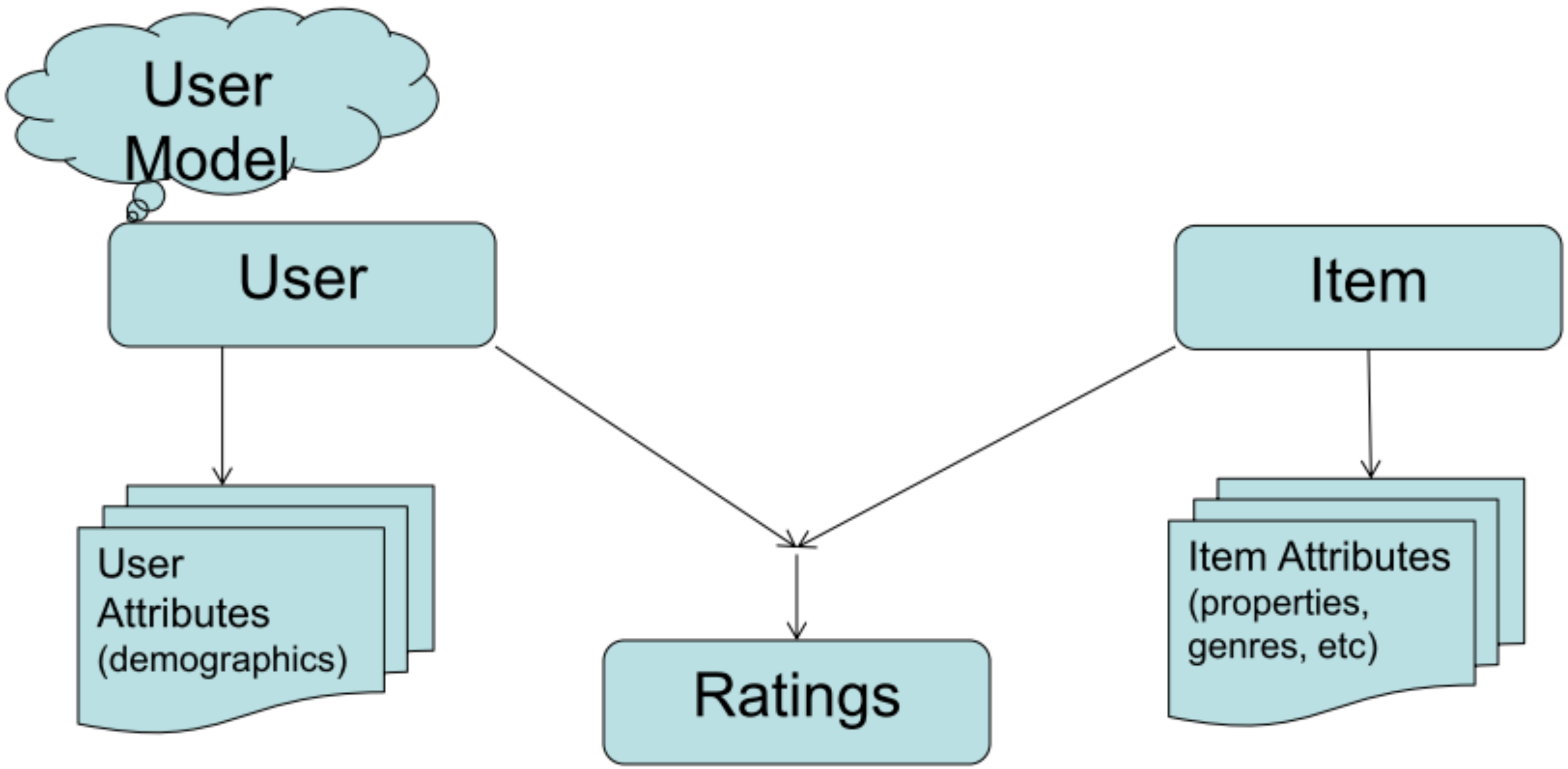
Approaches

- Non Personalized Recommendations
- Content Based Recommendations
- Neigborhood methods, beter kow as **Collaborative Filtering.** (We'll focus on this)

- Hybrid approaches

Collaborative Filtering 101

CONTEXT


User Model
User
Item
User Attributes (demographics)
Ratings
Item Attributes (properties, genres, etc)

- CF algorithms, infer recommendations from historical ***user-item*** interactions, by assuming that « Similar users tend to like similar items ».

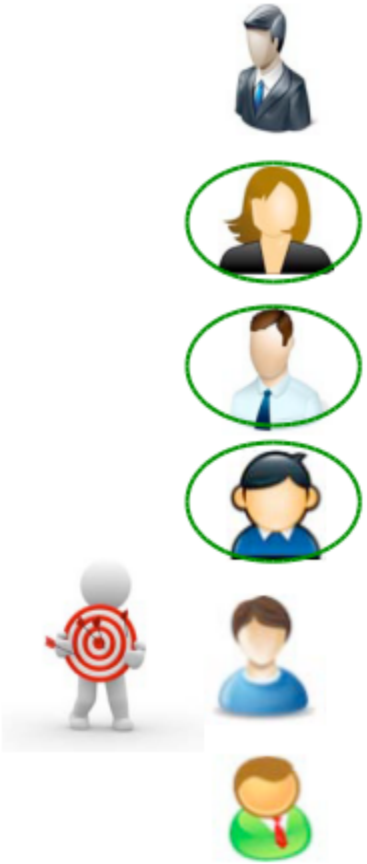
- Two appoaches :
  - <u>Memory based CF</u>
    - *** User based CF**
    - *** Item based CF**
  - <u>Model based CF</u> (Latent factors models)
    - *** Dimensionality Reduction(SVD o PCA)**
    - *** Matrix Factorization**

User based CF example

2			4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

2			4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

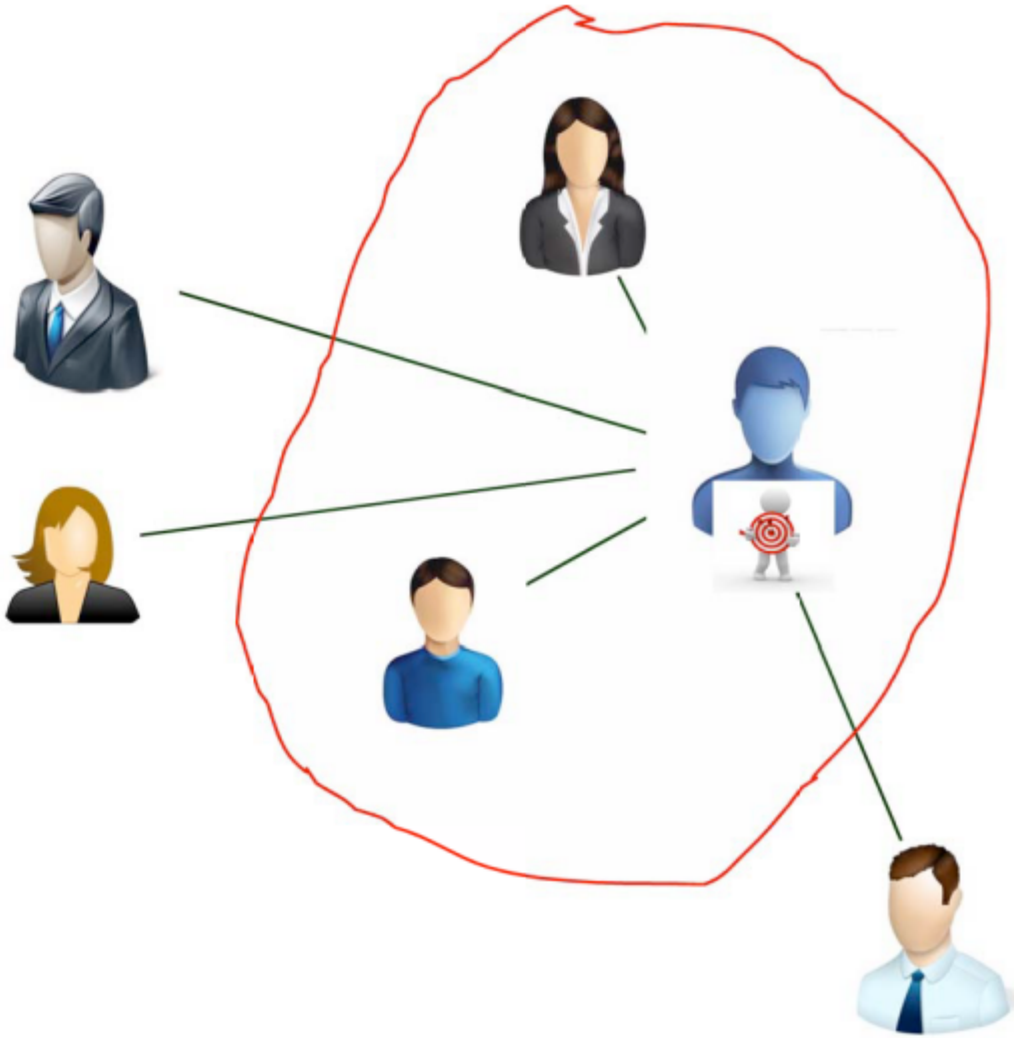
1. Identify items rated by the target user



2			4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

1. Identify items rated by the target user

2. Find other users who rated the same items



1. Identify items rated by the target user

2. Find other users who rated the same items

3. Select the top K most similar neighbors

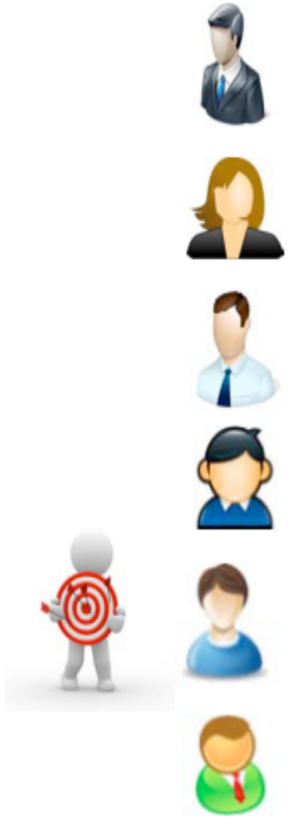
						sim(u,v)
2			4	5		NA
5		4			1	
		5		2		
	1		5		4	
		4			2	
4	5		1			NA

1. Identify items rated by the target user

2. Find other users who rated the same items

3. Select the top K most similar neighbors

Compute Similarities



						sim(u,v)
2			4	5		NA
5		4			1	0.87
		5		2		
	1		5		4	
		4			2	
4	5		1			NA

1. Identify items rated by the target user

2. Find other users who rated the same items

3. Select the top K most similar neighbors

Compute Similarities between neighbors

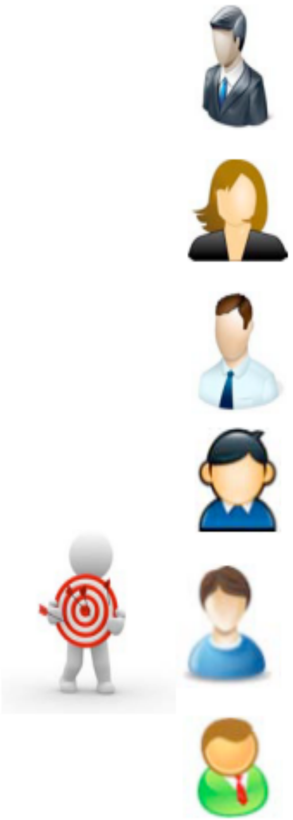
						sim(u,v)
2			4	5		NA
5		4			1	0.87
		5		2		1
	1		5		4	-1
		4			2	
4	5		1			NA

1. Identify items rated by the target user

2. Find other users who rated the same items

3. Select the top K most similar neighbors

Compute Similarities between neighbors



						sim(u,v)
2			4	5		NA
5		4			1	0.87
		5		2		1
	1		5		4	-1
3.51*	3.81*	4	2.42*	2.48*	2	
4	5		1			NA

1. Identify items rated by the target user

2. Find other users who rated the same items

3. Select the top K most similar neighbors

4. Predict Rating of the target user based on unrated items

Target user u, **ratings** matrix Y

Yv,i → rating by user v for item i

Similarity Pearson r correlation sim(u,v) between users u & v

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} (y_{u,i} - \hat{y}_u)(y_{v,i} - \hat{y}_v)}{\sqrt{\sum_{i \in I_{uv}} (y_{u,i} - \hat{y}_u)^2 \sum_{i \in I_{uv}} (y_{v,i} - \hat{y}_v)^2}}$$

Predicted rating $y^*(u,i)$

$$y^*(u,i) = \hat{y}_u + \frac{\sum_{j \in I_{y_{*j} \neq 0}} sim(v_j, u)(y_{v_j,i} - \hat{y}_{v_j})}{\sum_{j \in I_{y_{*j} \neq 0}} |sim(v_j, u)|}$$

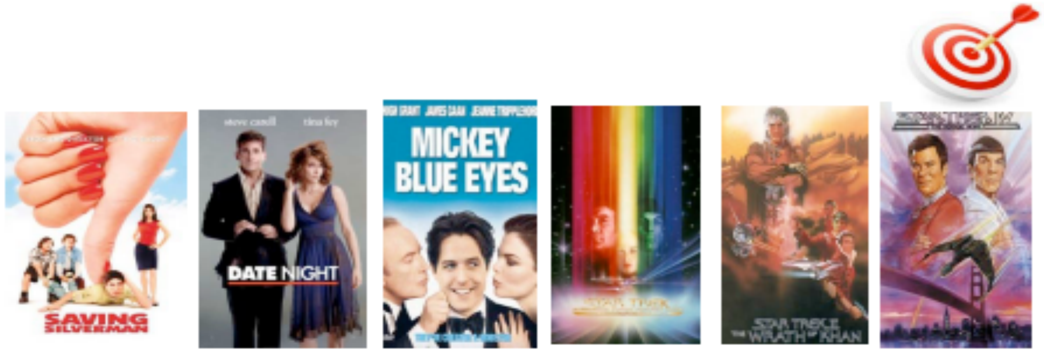
Item based CF example

2			4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

Goal : Predict users rating for an item based on their ratings for other items

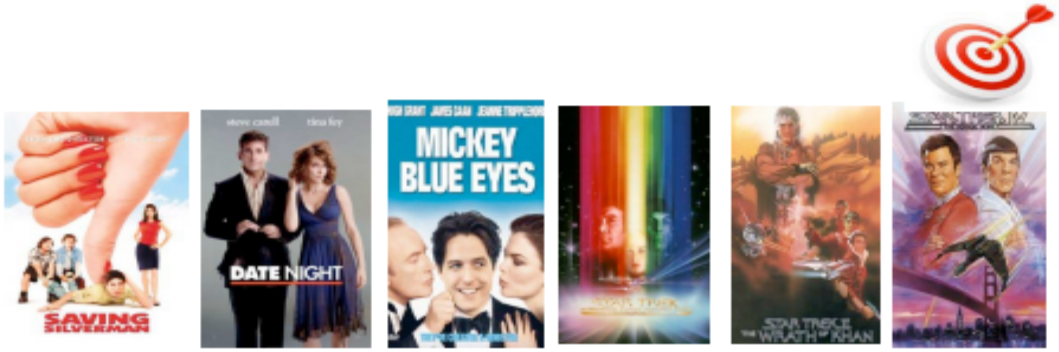
1. Identify the set of users who rated the target item
2. Find neighboring items
3. Compute similarities
4. Select top K similar items (Rank)
5. Predict rating for the target

Detect Neighbors

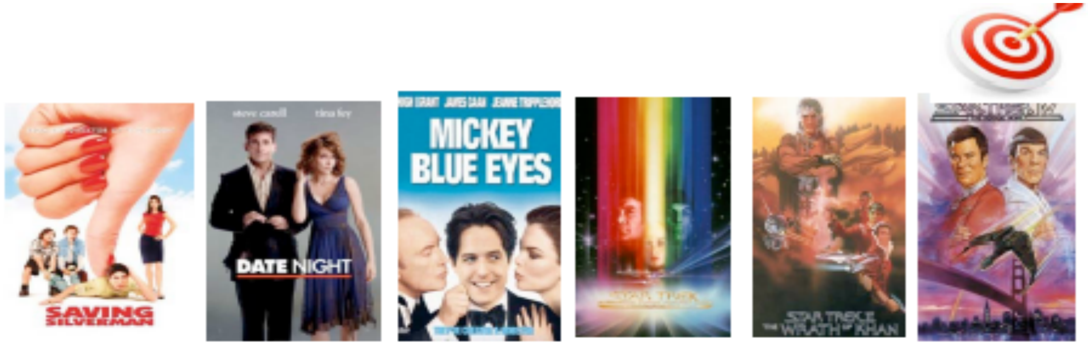


2			4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

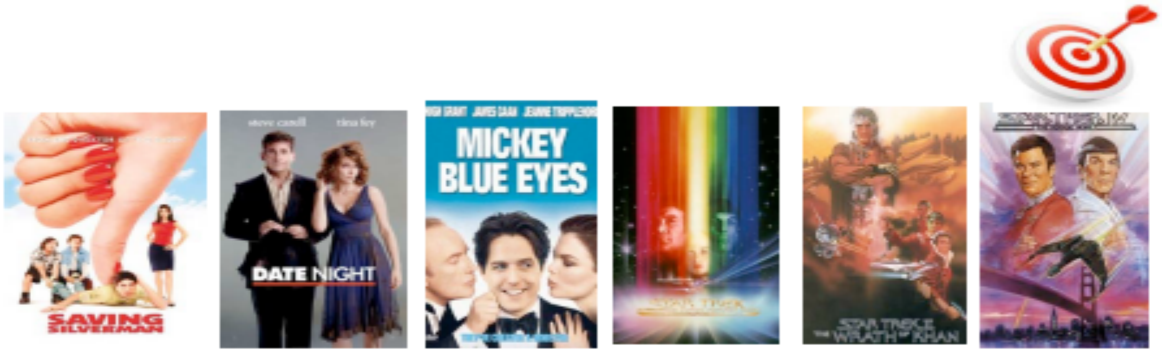
sim(i,j) -1



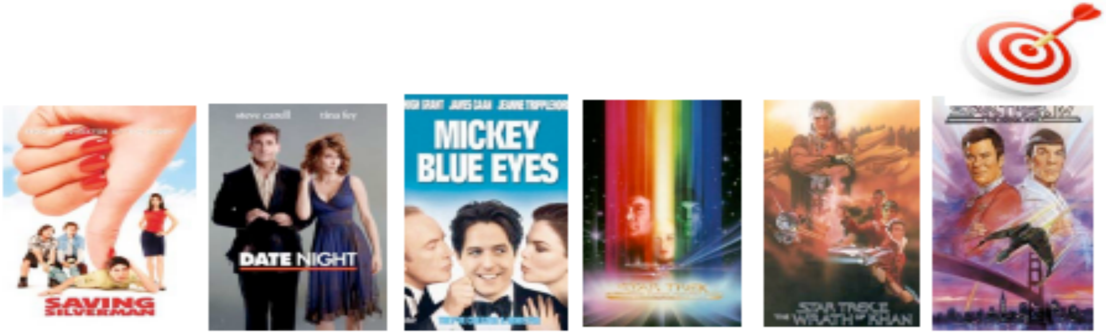
	2			4	5	
	5		4			1
			5		2	
		1		5		4
			4			2
	4	5		1		
sim(i,j)	-1	-1				



	2			4	5	
	5		4			1
			5		2	
		1		5		4
			4			2
	4	5		1		
sim(i,j)	-1	-1	0.86			



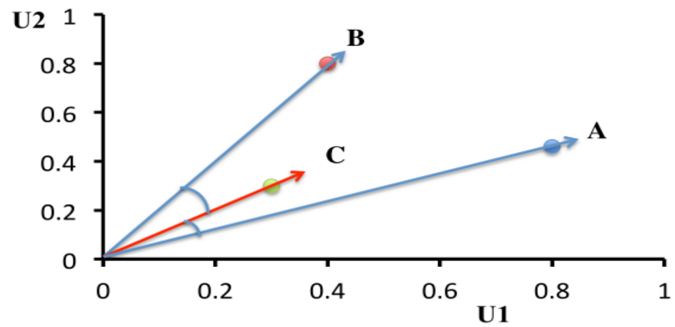
	2			4	5	
	5		4			1
			5		2	
		1		5		4
			4			2
	4	5		1		
sim(i,j)	-1	-1	0.86	1		



	2			4	5	2.94*
	5		4			1
			5		2	2.48*
		1		5		4
			4			2
	4	5		1		1.12*
sim(i,j)	-1	-1	0.86	1	NA	

Similarities Computations

- Pearson Similarity : Doesn'take into account user ratings bias
- Cosine Similarity : Item are represeted vector ove user space. Similarity is the cosine of angle betwee two vectors : -*1<=Sim(i,j) <=1*

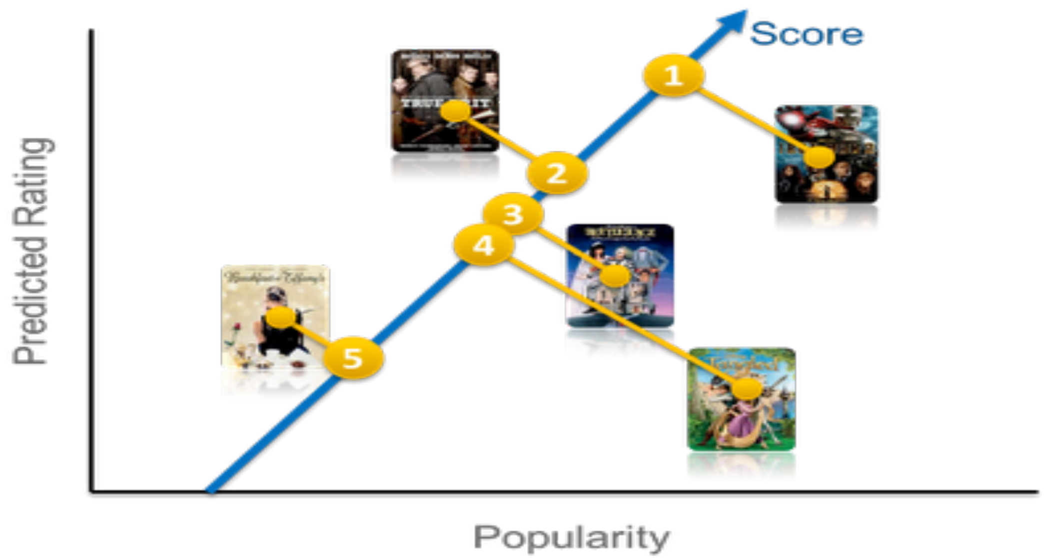


- Other similarities measures : Jaccard index, Magnitude aware measure …

Ranking

- Balance between popularity and predicted ranking.
- Predicted ranking : « Learning to Rank »
- Use a ranking function

$$f_{rank}(u,v) = w_1\, p(v) + w_2\, r(u,v) + b$$



Challenges

- ***Data sparsity*** : Users rarely clicks, rate or buy
- ***Cold Start Poblem***
- Harry Potter problem : correlations can be odious
- Long tail recommendations : lesser known items

Model based recommenders

- Learn models from latent factors (underlying poperties of data) rather from heuristics
- Try to identify inter-relationships between between variables
- *Clustering*
- *Dimensionality reduction (SVD)*
- *Matrix Factorization*

Dimensionality Reduction

- Generalize movies into latent semantics characteristics :
- Reduces dimensions and improve scalability
- Reduce Data sparsity and improves prediction accuracy

  e.g User who likes « Star Trek» also likes

  « Star Gate » …

Latent factor : Sci-fi, novel based …

Singular Value Decomposition

Single value decomposition takes a (m x n) matrix and produces three matrices:

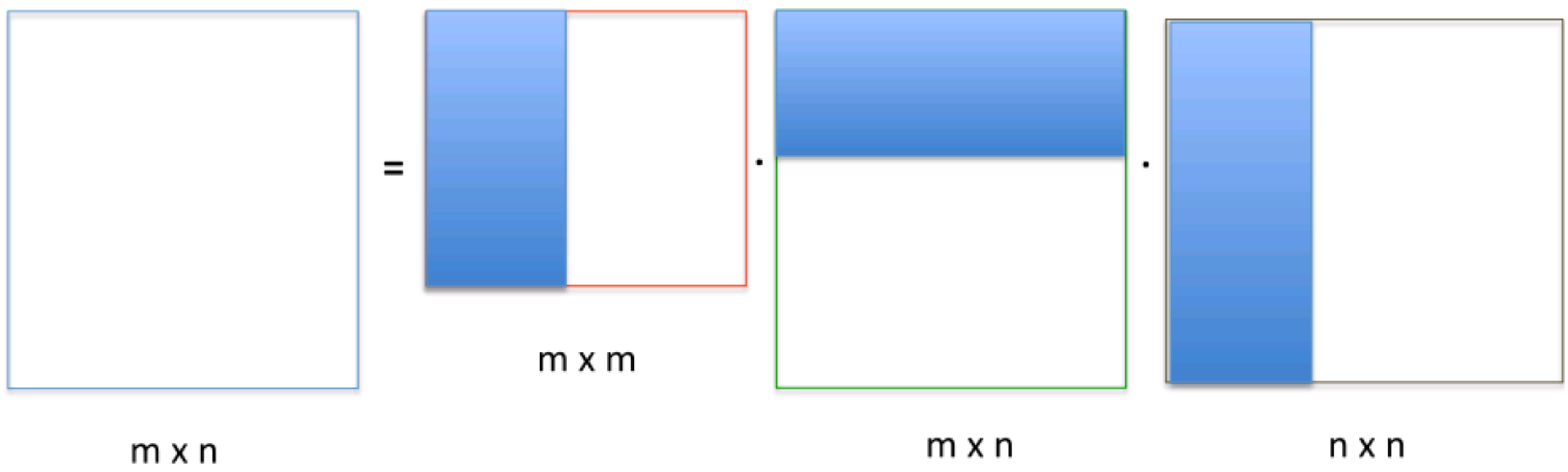
S : a (m x n) diagonal matrix with non-negative numbers

U : a (m x m) matrix

V : a (n x n) matrix

$$M = U \cdot S \cdot V^T$$

Matrix Decomposition



Apply SVD – example contd.

- SVD collapses the matrix to a smaller matrix retaining important features
- Pick *k* dimensions and chop off the matrixes

U	
-0.59	0.37
-0.17	0.13
-0.36	0.016
-0.31	0.51
-0.5	-0.03
-0.47	-0.75

S	
12.65	0
0	5.77

V^T		
-0.59	0.40	**Alice**
-0.39	0.49	
-0.20	0.53	
-0.42	0.62	
-0.53	0.44	**Ed**

We have reduced our dataset to a 2-dimensional space!

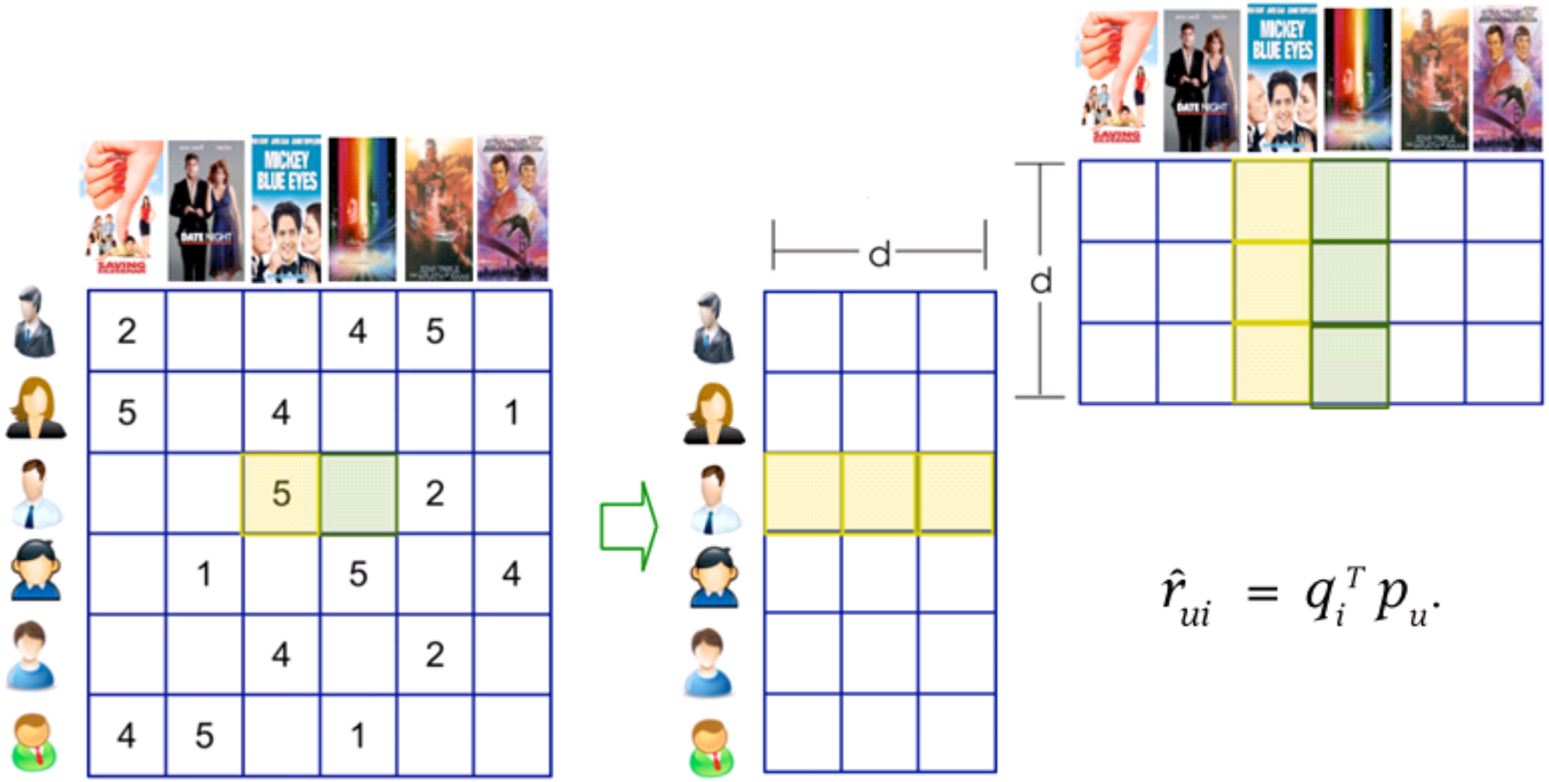
Finding Recommendations

- A user X comes in with some ratings in the original feature space : [0, 0, 2, 0, 4, 1]
- Map it to *k*-dimensional vector

$$\boxed{B = B^T \cdot U \cdot S^{-1}}$$

- X → [-0.25, -0.15]
- Note this is similar to the problem we solved earlier using cosine similarity

Matrix Factorization



$$\hat{r}_{ui} = q_i^T p_u.$$

For a given user u, p measure the extent of interest the user has in items that are high on the corresponding factors. R captures the interaction user-item .

Mahout Recommenders

- Two types of recommenders:

  - Single Machine Recommenders :

    Based on the Taste Framework , focus mostly on neighborhood methods :

    Recommender encapsulates algorithms, and ***DataModel*** handle interaction with data.

    E.g : SVDPlusPlusFactorizer, ALSWRFactorizer, …

  - Parallel Recommenders : ***RowSimilarityJob, ItemSimilarityJob, RecommenderJob***, strongly tied to hadoop

Exemple :

```
DataModel dataModel = new FileDataModel(new File(‘’file.csv’’));

UserSimilarity userSimilarity = new PearsonCorrelationSimilarity
                                                 (datamodel)

UserNeighborHood neighborhood = new NearestNUserNeighborhood(25,
                     userSimilarity, dataModel)

RecommenderBuilder recommenders = new GenericUserBasedRecommender
                                          (dataModel, neighborhood, userSimilarity)
```

Run it

- User Id: 1001
- Recommened Item Id 9010. Strength of the preference: 8.699270
- Recommened Item Id 9012. Strength of the preference: 8.659677
- Recommened Item Id 9011. Strength of the preference: 8.377571
- Recommened Item Id 9004. Strength of the preference: 1.000000
- User Id: 1002
- Recommened Item Id 9012. Strength of the preference: 8.721395
- Recommened Item Id 9010. Strength of the preference: 8.523443
- Recommened Item Id 9011. Strength of the preference: 8.211071
- User Id: 1003
- Recommened Item Id 9012. Strength of the preference: 8.692321
- Recommened Item Id 9010. Strength of the preference: 8.613442
- Recommened Item Id 9011. Strength of the preference: 8.303847
- User Id: 1004
- No recommendations for this user.
- User Id: 1005
- No recommendations for this user.
- User Id: 1006
- No recommendations for this user.

On Hadoop

```
hadoop  - jar  mahout-core-0.8-job.jar
 org.apache.mahout.cf.taste.hadoop.cf.item.RecommenderJob
       -- booleanData
       -- similarityClassname SIMILARITY_LOGLIKELIHOOD
       -- output output
       -- input input/data.dat
```

Evaluate a Recommender

- How to know if a recommender is good?
  -Compare implementations, play with similarity measures
  - Test your recommenders : A/B Testing, Multi Armed Bandits

- Business metrics
  - Does your recommender leads to increase value (CTR, sales, ..)

- Leave one out
  - Remove one preferences, rebuild the model, see if recommended
  - Cross validation, …
- Precision / Recall
  - Precision : Ratio of recommended items that are relevant
  - Recall : Ratio of relevant items actually recommended

Diversity / Serendipity

- Increase Diversity / Novelty
  - As items comes in remove the ones too similar to prior recommendation
  - Play with ranking to randomize Top K

- Increase Serendipity
  - Downgrade too popular items …