CS247: ADVANCED DATA MINING

Recommender Systems I

Instructor: Yizhou Sun

yzsun@cs.ucla.edu

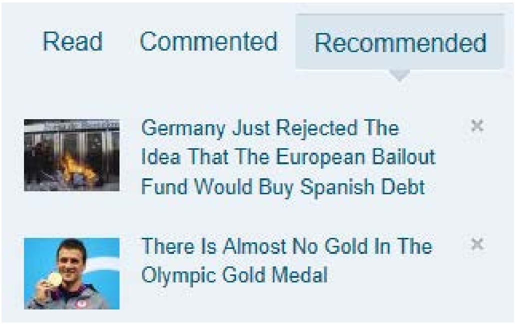
February 27, 2024

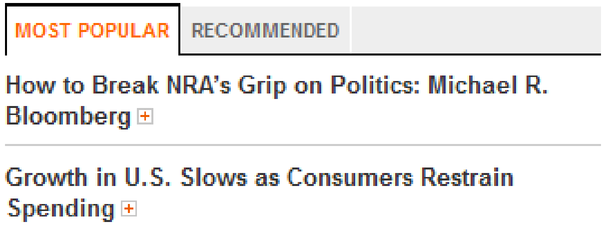
Recommender Systems

- What is Recommender System?
- Collaborative Filtering
- Content-based Recommendation
- Hybrid methods
- Evaluation Metrics
- Summary

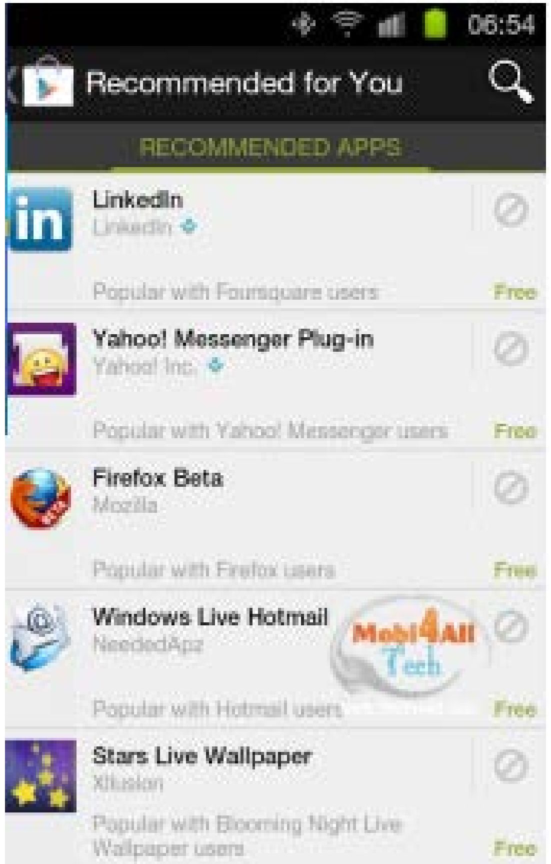
Recommender Systems

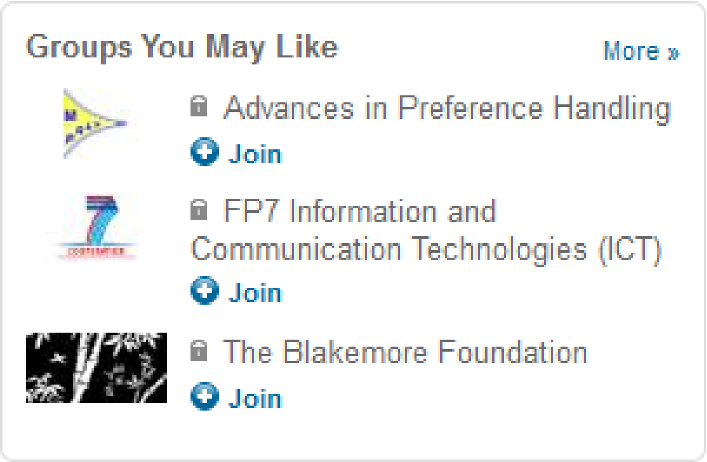
- Application areas

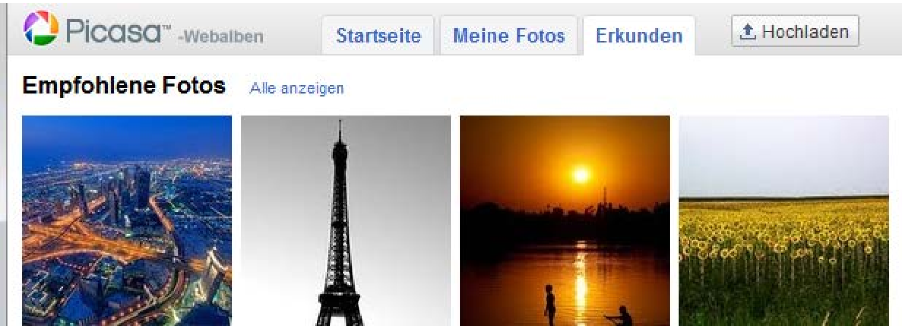




In the Social Web







Why using Recommender Systems?

- Value for the customer
  - Find things that are interesting
  - Narrow down the set of choices
  - Help me explore the space of options
  - Discover new things
  - Entertainment
  - …
- Value for the provider
  - Additional and probably unique personalized service for the customer
  - Increase trust and customer loyalty
  - Increase sales, click trough rates, conversion etc.
  - Opportunities for promotion, persuasion
  - Obtain more knowledge about customers
  - …

Recommendation as Matrix Completion

- Matrix Representation

Users	Movie1	Movie2	Movie3	Movie4	Movie5	Movie6	⋯
User1	?	?	4	?	1	?	⋯
User2	2	5	2	?	?	2	⋯
User3	?	?	5	3	2	4	⋯
User4	1	?	?	4	?	?	⋯
User5	2	3	?	?	?	?	⋯
⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯

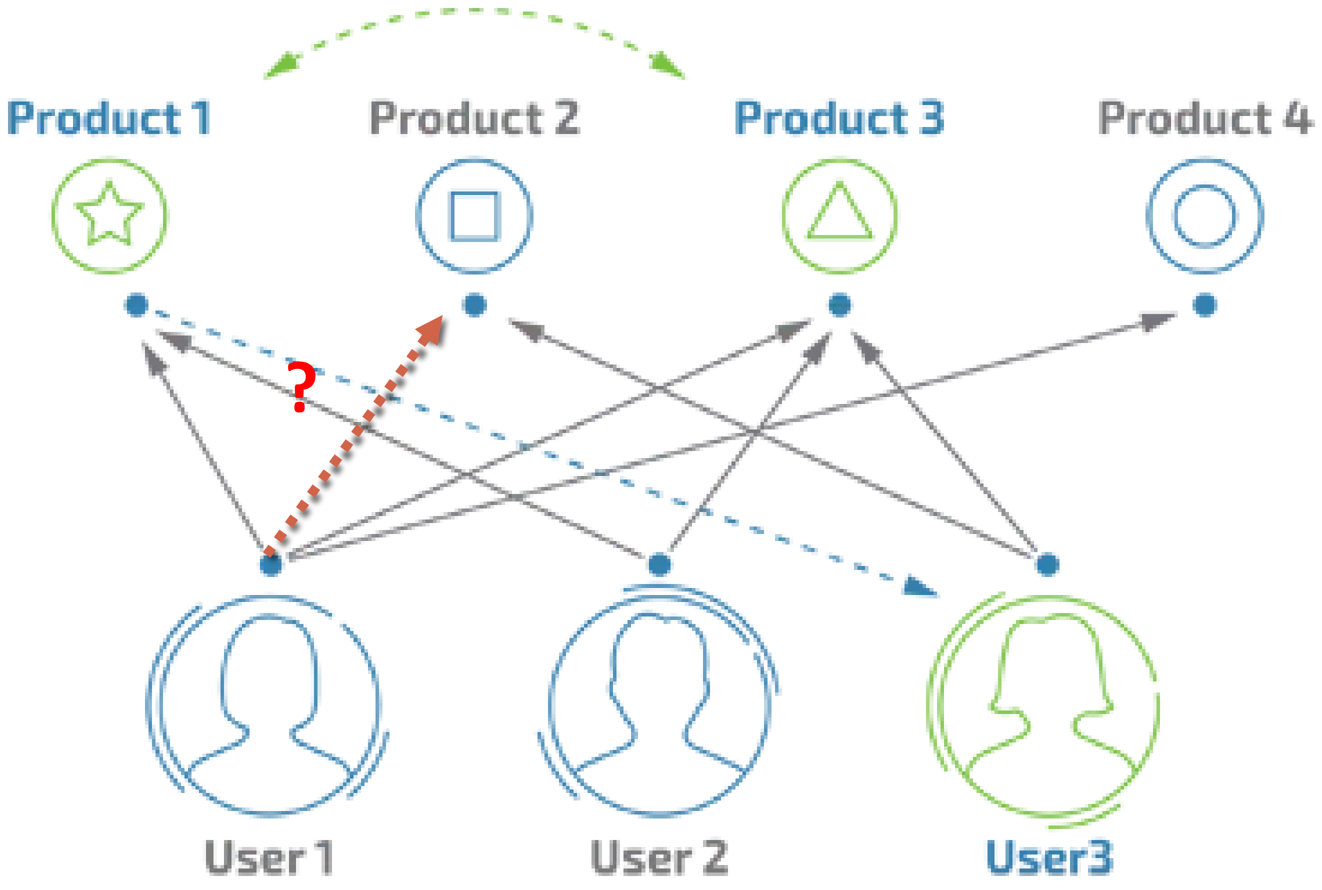
Explicit Feedback vs. Implicit Feedback

- Explicit Feedback
  - Know the ratings
- Implicit Feedback
  - only know whether user and item has interacted
  - Like (1) vs. unknown (0)

		Items				
						...
Users	Alice	1	1	0	0	
	Bob	0	0	1	1	
	Corey	1	0	1	0	
	...					

Recommendation as Link Prediction

- Network view
  - Link prediction problem



Methods

- Collaborative filtering
- Content-based recommendation
- Hybrid methods

Recommender Systems

- What is Recommender System?
- Collaborative Filtering
- Content-based Recommendation
- Hybrid methods
- Evaluation Metrics
- Summary

Collaborative Filtering (CF)

- The most prominent approach to generate recommendations
  - used by large, commercial e-commerce sites
  - well-understood, various algorithms and variations exist
  - applicable in many domains (book, movies, DVDs, ..)
- Approach
  - use the "wisdom of the crowd" to recommend items

- Basic assumption and idea
  - Users give ratings to catalog items (implicitly or explicitly)
  - Customers who had similar tastes in the past, will have similar tastes in the future

Major Methods for CF

- Memory-based Collaborative Filtering
  - User-based CF
    - Compute similarity between users and active users, and use similar users' ratings as prediction
  - Item-based CF
    - Compute similarity between items, and predict similar rating to similar items that the active user has rated before
- Model-based Collaborative Filtering

User-based Collaborative Filtering

1. Define similarity between users according to the history matrix
2. Decide how many "peers" to consider
3. Use peers' ratings to predict the rating between an active user and an item

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

(1) Define Similarities between Users

- Pearson correlation between user a and b

$$sim(a,b) = \frac{\sum_{p \in P}(r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P}(r_{a,p} - \bar{r}_a)^2}\sqrt{\sum_{p \in P}(r_{b,p} - \bar{r}_b)^2}}$$

  - $r_{a,p}$: $rating\ of\ user\ a\ to\ item\ p$
  - P: $a\ set\ of\ items\ that\ are\ rated\ by\ both\ a\ and\ b$
  - $\overline{r_a}, \overline{r_b}$: $average\ rating\ of\ user\ a\ and\ b\ for\ all\ items$

Example

- $sim(Alice, User1)$
  - $\overline{r_{Alice}} = \frac{5+3+4+4}{4} = 4$
  - $\overline{r_{User1}} = \frac{3+1+2+3+3}{5} = 2.4$
  - $\boldsymbol{=>sim(Alice, User1) = 0.8391}$

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

sim = 0.84
sim = 0.61
sim = -0.77

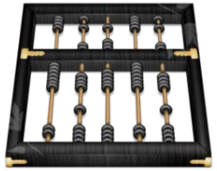
(2) Decide how many peers to use

- Usually only use top K most similar users for prediction
  - i.e., based on top-K most similar users' rating for an item

(3) Predict the rating

- A common prediction function:

$$pred(a,p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a,b)}$$



  - N denotes the k-nearest neighbor of a
- Calculate, whether the neighbors' ratings for the unseen item *p* are higher or lower than their average
- Combine the rating differences – use the similarity as a weight
- Add/subtract the neighbors' bias from the active user's average and use this as a prediction

Example

- Use top-2 neighbor for prediction
  - Alice's top-2 neighbor are User1 and User2
  - $pred(Alice, Item5) = \overline{r_{Alice}} + \frac{sim(Alice,User1)(r_{User1,Item5}-\overline{r_{User1}})+sim(Alice,User2)(r_{User2,Item5}-\overline{r_{User2}})}{sim(Alice,User1)+sim(Alice,User2)}$

$$= 4 + \frac{0.84*(3-2.4)+0.61*(5-3.8)}{0.84+0.61} = 4.85$$

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

sim = 0.84
sim = 0.61

Model-based Collaborative Filtering

- User-based CF is said to be "memory-based"
  - the rating matrix is directly used to find neighbors / make predictions
  - does not scale for most real-world scenarios
  - large e-commerce sites have tens of millions of customers and millions of items
- Model-based approaches
  - based on an offline pre-processing or "model-learning" phase
  - at run-time, only the learned model is used to make predictions
  - models are updated / re-trained periodically
  - large variety of techniques used
  - model-building and updating can be computationally expensive

Matrix Factorization for Recommendation

- Map users and items into the same latent space

Item

User	W	X	Y	Z
A		4.5	2.0	
B	4.0		3.5	
C		5.0		2.0
D		3.5	4.0	1.0

Rating Matrix

=

A	1.2	0.8
B	1.4	0.9
C	1.5	1.0
D	1.2	0.8

User Matrix

X

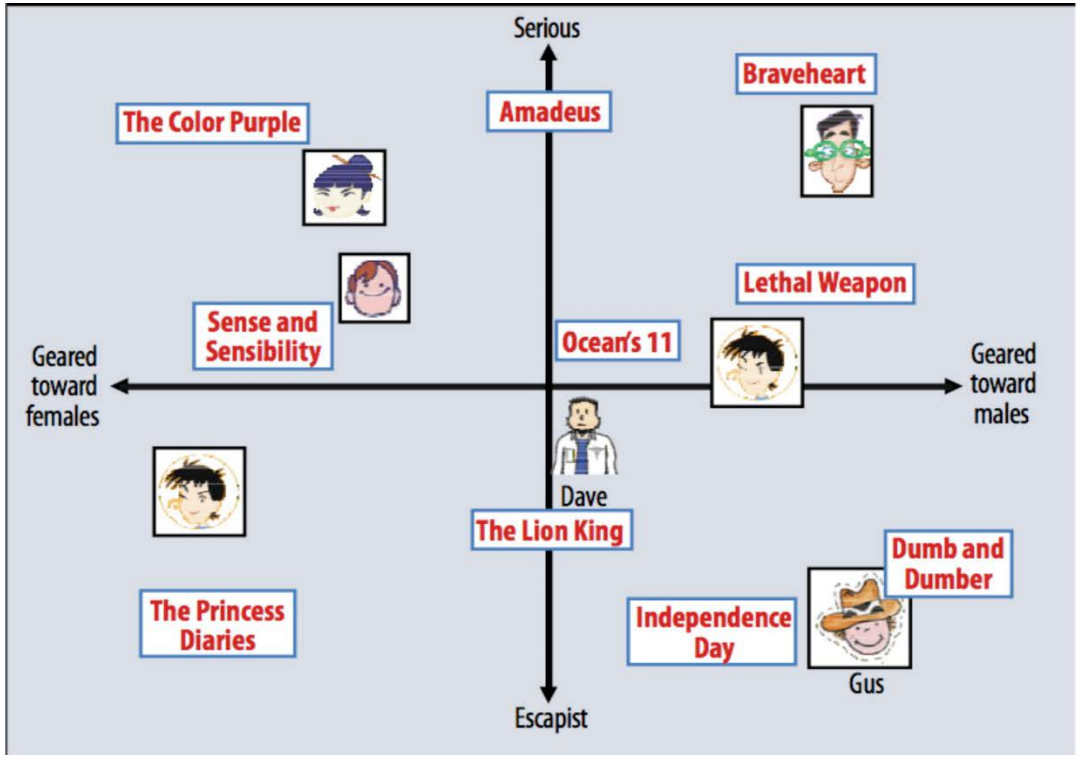
W	X	Y	Z
1.5	1.2	1.0	0.8
1.7	0.6	1.1	0.4

Item Matrix

Reference: Koren et al., "Matrix Factorization Techniques for Recommender System", Computer (Volume: 42, Issue: 8), 2009

Now users and items are comparable

- Recommendation: find items that are close to users in the new space



https://towardsdatascience.com/paper-summary-matrix-factorization-techniques-for-recommender-systems-82d1a7ace74

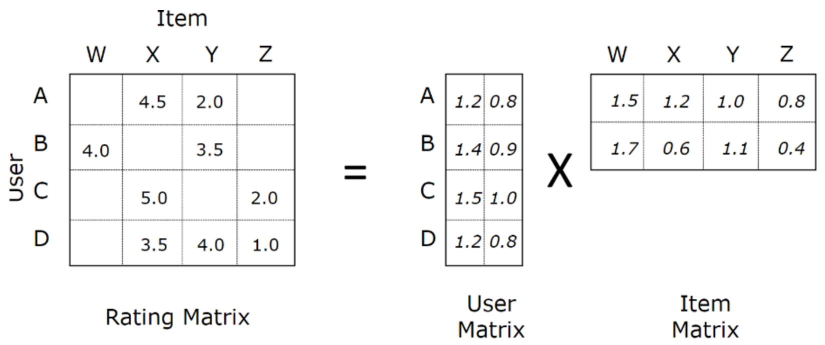
Procedure

- Training stage
  - Use existing matrix to learn the latent feature vector for both users and items by matrix factorization
- Recommendation stage
  - Predict the score for unknown (user, item) pairs

Training Stage

- r_{ui}: the rating from u to i
- p_u: the latent feature vector for user u
- q_i: the latent feature vector for item i
- $\hat{r}_{ui}$: score function for (u,i), $\hat{r}_{ui} = q_i^T p_u$
- Objective function:

$$\min_{p^*,q^*} \sum_{(u,i)\in D} \left(r_{ui} - q_i^T p_u\right)^2 + \lambda\left(\sum_i \left|\left|q_i\right|\right|^2 + \sum_u \left|\left|p_u\right|\right|^2\right)$$



Learning Algorithm

- Stochastic gradient descent
- For each rating (u, i):
  - $update\ p_u: p_u \leftarrow p_u + \eta \cdot ((r_{ui} - \hat{r}_{ui})q_i - \lambda p_u)$
  - $update\ q_i: q_i \leftarrow q_i + \eta \cdot \left((r_{ui} - \hat{r}_{ui})p_u - \lambda q_i\right)$

  - Where η is the learning rate

Prediction Stage

- For an unseen pair (u, i)
  - $\hat{r}_{ui} = q_i^T p_u = p_u^T q_i$
- Example:
  - $r_{AW} = p_A^T q_W = 1.2 * 1.5 + 0.8 * 1.7 = 3.16$

Item

User	W	X	Y	Z
A		4.5	2.0	
B	4.0		3.5	
C		5.0		2.0
D		3.5	4.0	1.0

Rating Matrix

=

A	1.2	0.8
B	1.4	0.9
C	1.5	1.0
D	1.2	0.8

User Matrix

X

W	X	Y	Z
1.5	1.2	1.0	0.8
1.7	0.6	1.1	0.4

Item Matrix

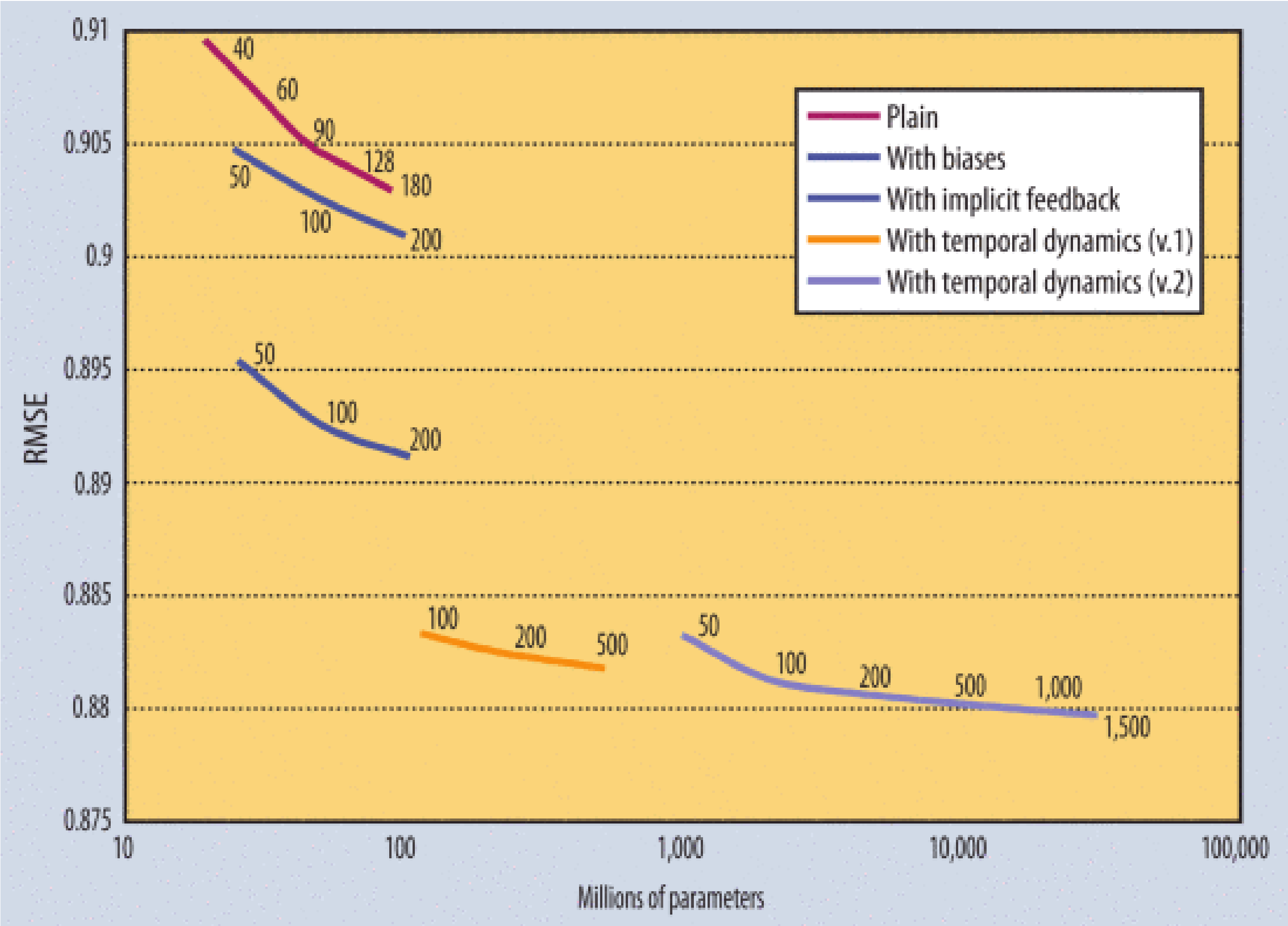
Variations

- Adding biases
  - $b_{ui} = \mu + b_i + b_u$
  - $\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$
  - Objective function:

$$\min_{p^*,q^*,b^*} \sum_{(u,i)\in D} \left(r_{ui} - \mu - b_i - b_u - q_i^T p_u\right)^2 + \lambda(\left||q_i|\right|^2 + \left||p_u|\right|^2 + \sum_u b_u^2 + \sum_i b_i^2)$$

- Adding temporal dynamics
  - $\hat{r}_{ui}^{(t)} = \mu + b_i(t) + b_u(t) + q_i^T(t) p_u(t)$

Results



Plain
With biases
With implicit feedback
With temporal dynamics (v.1)
With temporal dynamics (v.2)
RMSE
0.91
0.905
0.9
0.895
0.89
0.885
0.88
0.875
10
100
1,000
10,000
100,000
Millions of parameters
40
60
90
128
180
50
100
200
50
100
200
100
200
500
50
100
200
500
1,000
1,500

Question

- Can we make recommendation for new users or items using the above models?

Implicit Feedback Models

- Only implicit signals are received
  - E.g., click though, music streaming play

- Methods:
  - Turn it into binary classification problem: Logistic Matrix Factorization
    - Johnson, Logistic Matrix Factorization for Implicit Feedback Data, NIPS workshop 2014
  - Turn it into ranking problem: BPR: Bayesian Personalized Ranking
    - Rendel et al., BPR: Bayesian Personalized Ranking from Implicit Feedback, UAI’09

Logistic MF

- Model:
  - $r_{ui} = \{0,1\}$
  - $p(r_{ui} = 1 | p_u, q_i, b_i, b_u) = \frac{\exp(b_i + b_u + q_i^T p_u)}{1 + \exp(b_i + b_u + q_i^T p_u)}$
- Loss function
  - For each user-item pair (cross entropy loss):
    - $l_{ui} = -\mathbf{1}_{(\mathrm{r_{ui}}=1)} \log p(r_{\mathrm{ui}} = 1) - \mathbf{1}_{(r_{\mathrm{ui}}=0)} \log p(\mathrm{r_{ui}} = 0)$
  - Total loss:
    - $l = \sum_{u,i} l_{ui}$
    - Too many negative pairs! => solution: negative sampling

Bayesian Ranking

- Data re-arrangement:
  - $D_s = \{(u,i,j) | i \in I_u^+ \; and \; j \in I \backslash \mathrm{I}_u^+\}$
    - For user u, s/he ranks item i higher than j,
- Model:

$$p(i >_u j|\Theta) = \sigma(\hat{x}_{uij}(\Theta))$$

where $\hat{x}_{uij} = \hat{x}_{ui} - \hat{x}_{uj}$ and $\hat{x}_{ui} =< \boldsymbol{p}_u, \boldsymbol{q}_\mathrm{i} >$

- Loss Function

$$\prod_{u \in U} p(>_u |\Theta) = \prod_{(u,i,j) \in U \times I \times I} p(i >_u j|\Theta)^{\delta((u,i,j) \in D_S)} \cdot (1 - p(i >_u j|\Theta))^{\delta((u,j,i) \notin D_S)}$$

Discussion

- How to further improve the sampling strategy? (Hint: consider negative sampling in Word2Vec)

Issues of CF

- **Cold Start**: There needs to be enough other users already in the system to find a match.
- **Sparsity**: If there are many items to be recommended, even if there are many users, the user/ratings matrix is sparse, and it is hard to find users that have rated the same items.
- **First Rater**: Cannot recommend an item that has not been previously rated.
  - New items
  - Esoteric items
- **Popularity Bias**: Cannot recommend items to someone with unique tastes.
  - Tends to recommend popular items.

Recommender Systems

- What is Recommender System?
- Collaborative Filtering
- Content-based Recommendation
- Hybrid methods
- Evaluation Metrics
- Summary

Content-based recommendation

- Collaborative filtering does NOT require any information about content,
  - However, it might be reasonable to exploit such information
  - E.g. recommend fantasy novels to people who liked fantasy novels in the past
- What do we need:
  - Information about the available items such as the genre ("content")
  - *user profile* describing what the user likes (the preferences)
- The task:
  - Learn user preferences
  - Locate/recommend items that are "similar" to the user preferences

Content representation and item similarities

User profile

Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, Murder, Neo-nazism
...					

Item

Title	Genre	Author	Type	Price	Keywords
...	Fiction, Suspense	Brunonia Barry, Ken Follet, ..	Paperback	25.65	detective, murder, New York

- Simple approach
  - Compute the similarity of an unseen item with the user profile based on the keyword overlap (e.g. using the Dice coefficient)
  - $\mathrm{sim}(b_i, b_j) = \dfrac{2 * |keywords(b_i) \cap keywords(b_j)|}{|keywords(b_i)| + |keywords(b_j)|}$
- Other advanced similarity measure

Question

- Can content-based recommendation handle new users/items?

Recommender Systems

- What is Recommender System?
- Collaborative Filtering
- Content-based Recommendation
- Hybrid methods
- Evaluation Metrics
- Summary

Hybrid Methods

- Combining both user-item interaction and other external sources of information

- One example:
  - Factorization Machines
  - Steffen Rendle, “Factorization Machines,” in ICDM’10, Sydney, Australia.

Factorization Machines

- Treat each user-item transaction as one data point
  - U = {Alice (A), Bob (B), Charlie (C), …}
  - I = {Titanic (TI), Notting Hill (NH), Star Wars (SW), Star Trek (ST), . . .}
  - S = {(A, TI, 2010-1, 5), (A,NH, 2010-2, 3), (A, SW, 2010-4, 1), (B, SW, 2009-5, 4), (B, ST, 2009-8, 5), (C, TI, 2009-9, 1), (C, SW, 2009-12, 5)}

FM: Feature Preparation

- Each data point has a feature vector ***x***, and a target value (e.g., rating score)

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	$y^{(6)}$

The Model

- Model second-order interaction to overcome the sparsity

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

- $w_0 : global\ bias$
- $w_i : strength\ of\ ith\ variable$
- $\widehat{w}_{ij} =< \boldsymbol{v}_i, \boldsymbol{v}_j >$
  $: strength\ of\ the\ interaction\ of\ ith\ and\ jth\ variable$
  - E.g., interaction between Alice and Titanic, or Alice and Bob

Apply to Recommendation

- Explicit Feedback:
  - Treat it as a prediction task, with mean square error loss
- Implicit Feedback:
  - Treat it as a binary classification or ranking task, with logistic loss or pairwise logistic loss
- Learning:
  - Stochastic gradient descent

Recommender Systems

- What is Recommender System?
- Collaborative Filtering
- Content-based Recommendation
- Hybrid methods
- Evaluation Metrics
- Summary

Accuracy measures: Explicit Feedback

- Datasets with items rated by users
  - MovieLens datasets 100K-10M ratings
  - Netflix 100M ratings
- Historic user ratings constitute ground truth
- Metrics measure error rate
  - Mean Absolute Error (*MAE*) computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - r_i|$$

  - Root Mean Square Error (*RMSE*) is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2}$$

Implicit Feedback: Precision and Recall

- **Precision:** a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved
  - E.g. the proportion of recommended movies that are actually good

$$Precision = \frac{tp}{tp + fp} = \frac{|good\ movies\ recommended|}{|\text{all recommendations}|}$$

- **Recall:** a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items
  - E.g. the proportion of all good movies recommended

$$Recall = \frac{tp}{tp + fn} = \frac{|good\ movies\ recommended|}{|all\ good\ movies|}$$

More Implicit Feedback Measures

- Precision@k; recall@k
- AUC:
  - Area under ROC curve
- Area under Precision-Recall Curve
- MRR:
  - Mean reciprocal rank over a set of queries Q
  - $MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$, $rank_i$ is the rank position of the first relevant item for the ith query

Recommender Systems

- What is Recommender System?
- Collaborative Filtering
- Content-based Recommendation
- Hybrid methods
- Evaluation Metrics
- Summary

Summary

- Recommendation
  - User-based CF, matrix factorization-based CF
  - Explicit feedback, implicit feedback
  - Content-based recommendation
  - Hybrid methods
  - Evaluation

References

- http://ijcai13.org/files/tutorial_slides/td3.pdf
- http://research.microsoft.com/pubs/115396/EvaluationMetrics.TR.pdf
- https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf
- http://www.librec.net/tutorial.html