DDA4210/AIR6002 Advanced Machine Learning
Lecture 04 Advanced Applications

Tongxin Li

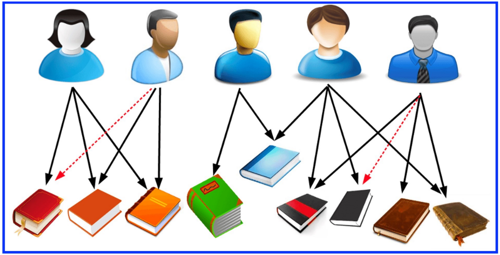
School of Data Science, CUHK-Shenzhen

Spring 2024

Overview

1. Recommendation System
   - Introduction
   - Collaborative Filtering Methods
   - Content-Based Methods
   - Hybrid Methods
   - Evaluation Metrics for RS
   - Examples
2. Learning to Rank
   - Introduction
   - Point-wise/Pair-wise/List-wise Modeling
   - Evaluation for L2R
   - Examples

Recommendation systems are anywhere!



Recommendation System: Methods

Methods for recommendation system

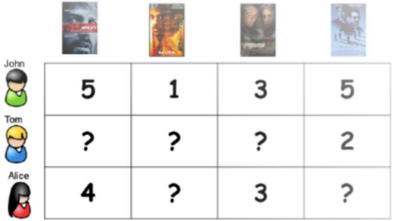
- Collaborative filtering methods
- Content based methods
- Hybrid methods

Collaborative Filtering: User-Item Interaction

- User-item interaction/utility
  - explicit feedback (e.g., purchase or not purchase, rating)
  - implicit feedback (e.g., click or not click, time spent)

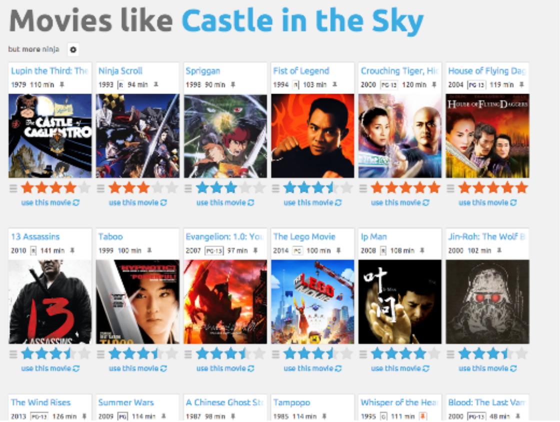
Collaborative Filtering: User-Item Interaction

- User-item interaction/utility
  - explicit feedback (e.g., purchase or not purchase, rating)
  - implicit feedback (e.g., click or not click, time spent)
- User-item rating matrix
  - highly incomplete (why?)
  - very large (why?)



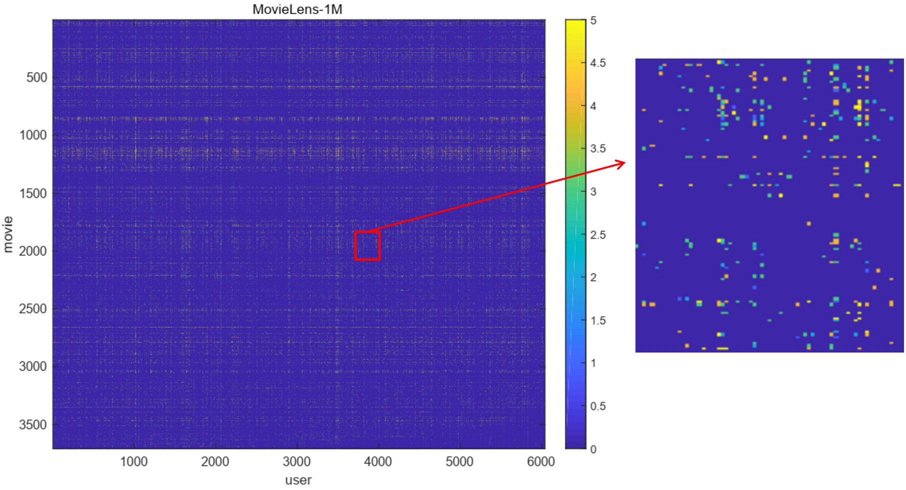
Collaborative Filtering: Examples of Benchmark

- MovieLens-1M: 4000$\times$6000
- MovieLens-20M: 27,000$\times$138,000
- Netflix 2009: 18,000$\times$480,000
- Doban: 58,000$\times$129,000

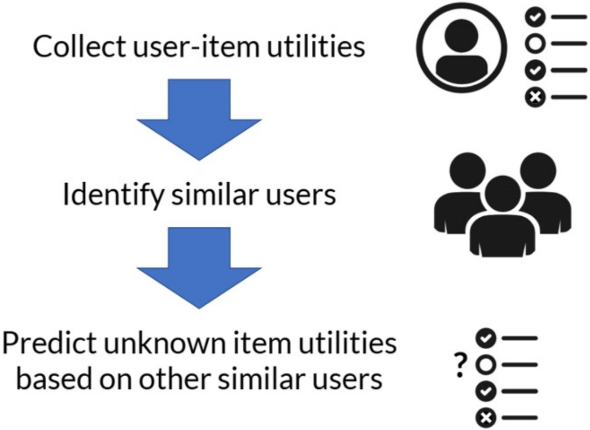


Collaborative Filtering: Examples of Benchmark

MovieLens-1M: 4000×6000, missing rate > 0.95



Collaborative Filtering: Classic Method



Rating matrix is a special case of user-item utility

Image from Eric Eaton

User-Item Utilities

	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace	4	5	4	1	5	3	...
Eric	1	4	5	1	5	3	...
Haren	5	5	5	1	3	4	...
Sai	1	2	5	4	3	5	...
Siyan	3	1	1	3	4	5	...
Nikhil	2	3	4	2	2	2	...
Felix	1	1	1	5	2	2	...

Image from Eric Eaton

Identify Similar Users

Similar users

	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace	4	5	4	1	5	3	...
Eric	1	4	5	1	5	3	...
Haren	5	5	5	1	3	4	...
Sai	1	2	5	4	3	5	...
Siyan	3	1	1	3	4	5	...
Nikhil	2	3	4	2	2	2	...
Felix	1	1	1	5	2	2	...

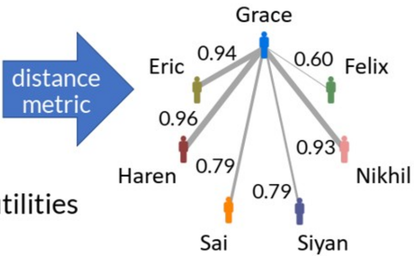
Image from Eric Eaton

Identify Similar Users

User-Item Utility Matrix

	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace	4	5	4	1	5	3	...
Eric	1	4	5	1	5	3	...
Haren	5	5	5	1	3	4	...
Sai	1	2	5	4	3	5	...
Siyan	3	1	1	3	4	5	...
Nikhil	2	3	4	2	2	2	...
Felix	1	1	1	5	2	2	...

User Similarities

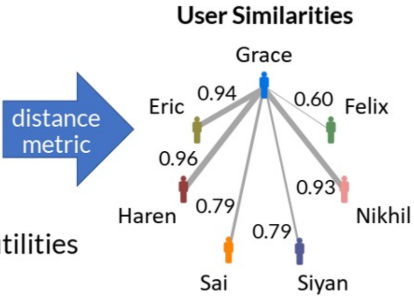


We could then predict unknown item utilities for Grace based on other similar users

Identify Similar Users

User-Item Utility Matrix

	Gossip Girl	The Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace	4	5	4	1	5	3	...
Eric	1	4	5	1	5	3	...
Haren	5	5	5	1	3	4	...
Sai	1	2	5	4	3	5	...
Siyan	3	1	1	3	4	5	...
Nikhil	2	3	4	2	2	2	...
Felix	1	1	1	5	2	2	...



We could then predict unknown item utilities for Grace based on other similar users

Open issues

- Choice of distance metric
- Dealing with sparse data
- How to combine known user utilities to do the prediction

Image from Eric Eaton

Distance/Similarity Measurement

- Euclidean distance:

$$\text{similarity}(user_i, user_j) = \frac{1}{1 + \left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2}$$

- Cosine similarity:

$$\text{similarity}(user_i, user_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\left\|\mathbf{x}_i\right\| \left\|\mathbf{x}_j\right\|}$$

- Pearson correlation coefficient

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

- Spearman's rank correlation coefficient

$$r_s = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

  - $d_i = \mathrm{R}(X_i) - \mathrm{R}(Y_i)$: difference between the two ranks of each observation

Distance/Similarity Measurement

Can only measure similarity between users using their overlapping items

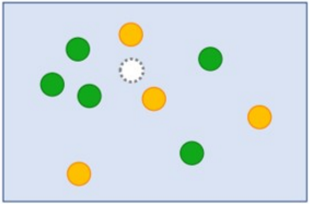
	G	Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace		5		1	5		...
Eric		4	5		5	3	...
Haren	5		5		3	4	...
Sai		2					...
Siyan	3	1		3		5	...
Nikhil				2	2		...
Felix	1		1		2		...

?

Image from Eric Eaton

Nearest-Neighbor Collaborative Filtering

	G[illegible]	[illegible] Office	The Mandalorian	Criminal Minds	The Good Place	Grey's Anatomy	...
Grace		5		1	5		...
Eric		4	5		5	3	...
Haren	5		5		3	4	...
Sai		2					...
Siyan	3	1		3		5	...
Nikhil				2	2		...
Felix	1		1		2		...



Idea: predict utility of item *i* based on the most-similar users who recorded a utility for that item

Image from Eric Eaton

Nearest-Neighbor Collaborative Filtering

Idea: predict utility of item *i* based on the most-similar users who recorded a utility for that item

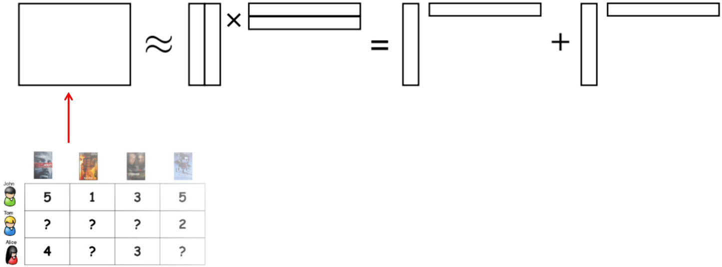
- Let $\mathcal{N}$ be the neighborhood set: the most similar users to user *u* who have rated item *i*
- Let w_{uv} be a weight $\in [0, 1]$ based on the similarity of users *u* and *v*
- Predict user *u*'s utility for item *i* as

$$\hat{x}_{ui} = \bar{x}_u + \sum_{v \in \mathcal{N}} \left((x_{vi} - \bar{x}_v) \times \frac{w_{uv}}{\sum_{v' \in \mathcal{N}} w_{uv'}} \right)$$

  - $\bar{x}_u$: average rating of user *u*
  - $\bar{x}_v$: average rating of user *v*

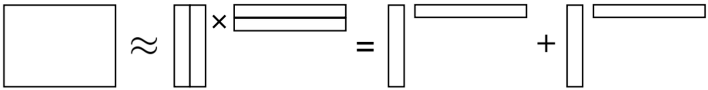
Matrix Factorization Collaborative Filtering

- Low-rank matrix factorization



Matrix Factorization Collaborative Filtering

- Low-rank matrix factorization



John	5	1	3	5
Tom	?	?	?	2
Alice	4	?	3	?

- Low-rank matrix completion

-0.26	-1.84	?	2.16	1.03	0.56	?	1.91
-0.86	?	1.83	0.65	?	0.01	1.17	?
1.23	0.09	?	2.63	0.84	?	-1.56	0.56
?	2.27	-1.48	?	-1.76	-1.26	?	-2.71
-0.39	?	1.12	0.89	0.50	?	0.54	?

-0.26	-1.84	1.54	2.16	1.03	0.56	0.41	1.91
-0.86	-1.74	1.83	0.65	0.52	0.01	1.17	1.47
1.23	0.09	-0.82	2.63	0.84	0.99	-1.56	0.56
-0.35	2.27	-1.48	-4.19	-1.76	-1.26	0.35	-2.71
-0.39	-1.19	1.12	0.89	0.50	0.18	0.54	1.11

Matrix Factorization Collaborative Filtering

John	5	1	3	5
Tom	?	?	?	2
Alice	4	?	3	?

- Notations
  - $R = [r_{ui}] \in \mathbb{R}^{m \times n}$: incomplete user-item rating matrix
  - Ω: the set of indices of observed entries (e.g. known ratings)
  - $P = [p_1, \ldots, p_u, \ldots, p_m] \in \mathbb{R}^{f \times m}, \quad Q = [q_1, \ldots, q_i, \ldots, q_n] \in \mathbb{R}^{f \times n}$

Matrix Factorization Collaborative Filtering

John	5	1	3	5
Tom	?	?	?	2
Alice	4	?	3	?

- Notations
  - $R = [r_{ui}] \in \mathbb{R}^{m \times n}$: incomplete user-item rating matrix
  - Ω: the set of indices of observed entries (e.g. known ratings)
  - $P = [p_1, \ldots, p_u, \ldots, p_m] \in \mathbb{R}^{f \times m}, \quad Q = [q_1, \ldots, q_i, \ldots, q_n] \in \mathbb{R}^{f \times n}$
- SVD-based recommendation ($R \approx P^\top Q$)

$$\underset{P,Q}{\text{minimize}} \sum_{(u,i)\in\Omega} \left\{ \left(r_{ui} - p_u^\top q_i \right)^2 + \lambda \left(\|p_u\|^2 + \|q_i\|^2 \right) \right\} \tag{1}$$

Matrix Factorization Collaborative Filtering

John	5	1	3	5
Tom	?	?	?	2
Alice	4	?	3	?

- Notations
  - $R = [r_{ui}] \in \mathbb{R}^{m \times n}$: incomplete user-item rating matrix
  - Ω: the set of indices of observed entries (e.g. known ratings)
  - $P = [p_1, \ldots, p_u, \ldots, p_m] \in \mathbb{R}^{f \times m}, \quad Q = [q_1, \ldots, q_i, \ldots, q_n] \in \mathbb{R}^{f \times n}$
- SVD-based recommendation ($R \approx P^\top Q$)

$$\underset{P,Q}{\text{minimize}} \sum_{(u,i)\in\Omega} \left\{ \left(r_{ui} - p_u^\top q_i \right)^2 + \lambda \left(\|p_u\|^2 + \|q_i\|^2 \right) \right\} \qquad (1)$$

- Optimization
  - Gradient descent (GD) or SGD
  - Alternating least squares

Matrix Factorization Collaborative Filtering

John	5	1	3	5
Tom	?	?	?	2
Alice	4	?	3	?

- SVD with bias ($b_{ui} = \mu + b_u + b_i$)

$$\underset{P,Q,B}{\text{minimize}} \sum_{(u,i)\in\Omega} \left\{ \left(r_{ui} - \mu - b_u - b_i - p_u^\top q_i\right)^2 + \lambda \left(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2\right) \right\} \tag{2}$$

  - $B = \{\mu, \{b_u\}, \{b_i\}\}$
  - What are the meanings of μ, b_u, and b_i?

Koren, Y., Bell, R., Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.

Collaborative Filtering: Advantage and Disadvantage

Advantages

- No domain knowledge needed
  - Item details are irrelevant, only user behavior matters
- Heterogeneous preferences
  - Capture that users may have diverse preferences

Collaborative Filtering: Advantage and Disadvantage

Advantages

- No domain knowledge needed
  - Item details are irrelevant, only user behavior matters
- Heterogeneous preferences
  - Capture that users may have diverse preferences

Disadvantages

- Cannot handle new items and new users (cold start problem)
  - New items have no user feedback and new users have no rating records
  - So the system cannot make recommendations for them

Content-Based Methods

- Collaborative filtering doesn't consider user or item attributes/content
- Content-based methods do!

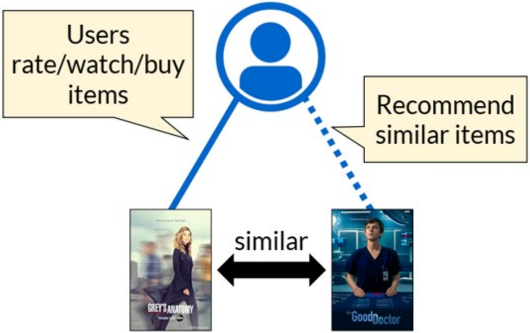


Image from Eric Eaton

Content-Based Methods

- Content analysis: characterize item as feature vector (e.g., TF-IDF features of text description, image features)

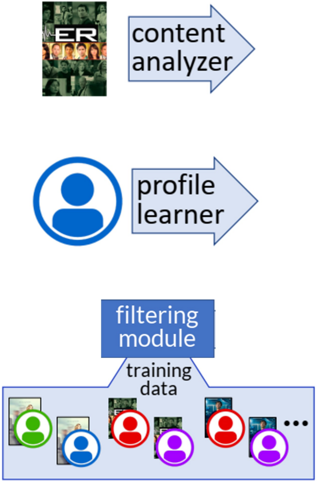


Image from Eric Eaton

- Content analysis: characterize item as feature vector (e.g., TF-IDF features of text description, image features)
- Profile learning: characterize user as feature vector (e.g., age, sex, education)

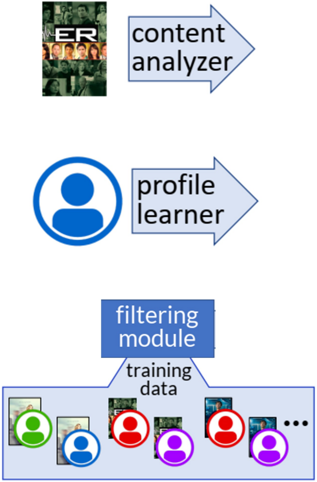


Image from Eric Eaton

Content-Based Methods

- Content analysis: characterize item as feature vector (e.g., TF-IDF features of text description, image features)
- Profile learning: characterize user as feature vector (e.g., age, sex, education)
- Filtering module: train classification/regression model for predicting users utility for an item

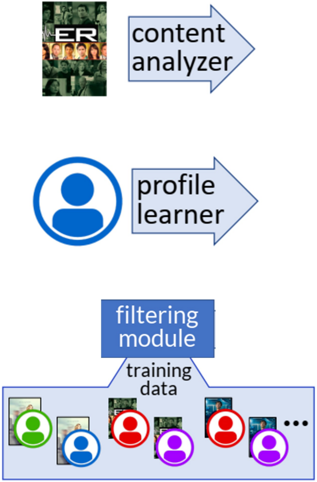


Image from Eric Eaton

Content-Based Methods

- Consider the following notations
  - ℓ: a loss function (e.g. squared loss)
  - $g : \mathbb{R}^{d_I} \to \mathbb{R}^{n_U}$, $h : \mathbb{R}^{d_U} \to \mathbb{R}^{n_I}$
  - d_U (d_I): $\#$ of user (item) features
  - n_U (n_I): $\#$ of users (items)
  - g_u: u-th output of g
- Recommendation for item

$$\underset{g}{\textit{minimize}} \sum_{(u,i)\in\Omega} \ell(r_{ui}, g_u(z_i))$$

- Recommendation for user

$$\underset{h}{\textit{minimize}} \sum_{(u,i)\in\Omega} \ell(r_{ui}, h_i(z_u))$$

Question: How do they handle new users or items?

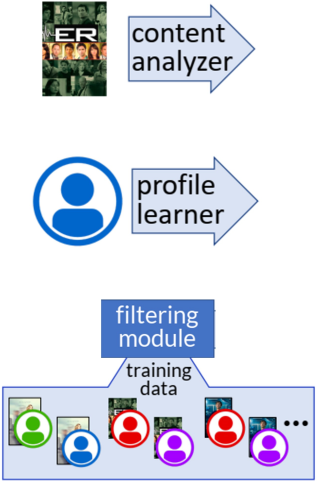


Image from Eric Eaton

Content-Based Methods

Advantages

- User-independent: only relies on user's profile to make recommendations
- Explainable: recommendations are based on concrete interacting features
- Handles new items well: item features are from content
- Handles new users well: user features are from content

Content-Based Methods

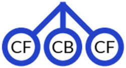
Advantages

- User-independent: only relies on user's profile to make recommendations
- Explainable: recommendations are based on concrete interacting features
- Handles new items well: item features are from content
- Handles new users well: user features are from content

Disadvantages

- Content-analysis is limited: relies on discrete features, often needs domain knowledge
- Narrow recommendations: often recommends similar items to a user, since those have highest scores

Hybrid Methods



Combining separate recommenders

- Can use any ensemble technique: linear weighting, stacking, etc.
- Recall – the Netflix prize winner was a blend of over 800+ recommenders

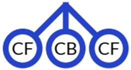
Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	The Ensemble	0.8553	10.10	2009-07-26 18:38:22
2	BellKor's Pragmatic Chaos	0.8554	10.09	2009-07-26 18:18:28
Grand Prize - RMSE <= 0.8563				
3	Grand Prize Team	0.8571	9.91	2009-07-24 13:07:49
4	Opera Solutions and Vandelay United	0.8573	9.89	2009-07-25 20:05:52
5	Vandelay Industries !	0.8579	9.83	2009-07-26 02:49:53
6	PragmaticTheory	0.8582	9.80	2009-07-12 15:09:53
7	BellKor in BigChaos	0.8590	9.71	2009-07-26 12:57:25
8	Dace_	0.8603	9.58	2009-07-24 17:18:43
9	Opera Solutions	0.8611	9.49	2009-07-26 18:02:08
10	BellKor	0.8612	9.48	2009-07-26 17:19:11
11	BigChaos	0.8613	9.47	2009-06-23 23:06:52
12	Feeds2	0.8613	9.47	2009-07-24 20:06:46
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8633	9.26	2009-07-21 02:04:40
14	Gravity	0.8634	9.25	2009-07-26 15:58:34

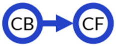
Image from Eric Eaton

Hybrid Methods



Combining separate recommenders

- Can use any ensemble technique: linear weighting, stacking, etc.
- Recall – the Netflix prize winner was a blend of over 800+ recommenders



Adding content-based aspects to collaborative models

- e.g., content-based user profiles, add content agents as collaborators

Example: matrix factorization with side information

Most systems that we use nowadays are hybrid recommenders.

Evaluation Metric for Recommendation System

Predictive metrics

- RMSE (root mean square error)

$$\mathrm{RMSE} = \sqrt{\frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} (r_{ui} - \hat{r}_{ui})^2}$$

  - $\hat{R}$: the set of ratings we predicted

- MAE (mean absolute error)

$$\text{MAE} = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} |r_{ui} - \hat{r}_{ui}|$$

Question: Given the rating matrix, how to define/construct training data and test data?

Recommender Systems Handbook

https://link.springer.com/book/10.1007/978-0-387-85820-3

Evaluation Metric for Recommendation System

Ranking-based metrics

- Precision@k (fraction of top k recommended items that are relevant to the user)

$$\text{Prec}(R)_k = \frac{|\{r \in R : r \leq k\}|}{k}$$

  - R: the set of relevant items; r: a recommended item
  - k: $\#$ of recommended items; Precision $= TP/(TP + FP)$

Evaluation Metric for Recommendation System

Ranking-based metrics

- Precision@k (fraction of top k recommended items that are relevant to the user)

$$\mathrm{Prec}(R)_k = \frac{|\{r \in R : r \leq k\}|}{k}$$

  - R: the set of relevant items; r: a recommended item
  - k: # of recommended items; Precision $= TP/(TP + FP)$

- Recall@k (fraction of top k recommended items that are in a set of items relevant to the user; also known as HitRatio@k)

$$\mathrm{Recall}(R)_k = \frac{|\{r \in R : r \leq k\}|}{|R|}. \tag{3}$$

  - Recall $= TP/(TP + FN)$

Evaluation Metric for Recommendation System

Ranking-based metrics

- Average Precision

$$\mathrm{AP@N} = \frac{1}{m}\sum_{k=1}^{N}\Big(P(k) \text{ if } k^{th} \text{ item was relevant }\Big) = \frac{1}{m}\sum_{k=1}^{N} P(k)\cdot \mathrm{rel}(k)$$

  - $P(k)$: precision at k
  - N: number of recommended items
  - m: total number of relevant items in the full space of items
  - $\mathrm{rel}(k)$: an indicator function

Evaluation Metric for Recommendation System

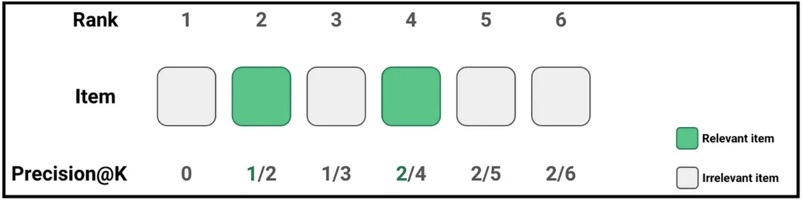
Ranking-based metrics

- Average Precision

$$\text{AP@N} = \frac{1}{m}\sum_{k=1}^{N}\Big(P(k) \text{ if } k^{th} \text{ item was relevant}\Big) = \frac{1}{m}\sum_{k=1}^{N} P(k)\cdot \text{rel}(k)$$

  - $P(k)$: precision at k
  - N: number of recommended items
  - m: total number of relevant items in the full space of items
  - $\text{rel}(k)$: an indicator function

Example:



$$\text{AP@6} = \frac{1}{2} \times (0\cdot 0 + 0.5\cdot 1 + 0.33\cdot 0 + 0.5\cdot 1 + 0.4\cdot 0 + 0.33\cdot 0) = 0.5$$

Image from https://towardsdatascience.com/mean-average-precision-at-k-map-k-clearly-explained-538d8e032d2

Ranking-based metrics

- Mean Average Precision (MAP, mean of APs over Q users):

$$\mathrm{MAP} = \frac{\sum_{q=1}^{Q} \mathrm{AP}(\mathrm{q})}{Q}$$

Evaluation Metric for Recommendation System

Ranking-based metrics

- Mean Average Precision (MAP, mean of APs over Q users):

$$\mathrm{MAP} = \frac{\sum_{q=1}^{Q} \mathrm{AP(q)}}{Q}$$

- Mean Reciprocal Rank (MRR, used when there is only one relevant item or only the first recommended item is the essential one; over Q users)

$$\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}$$

- Normalized Discounted Cummulative Gain (NDCG, optional in this course)

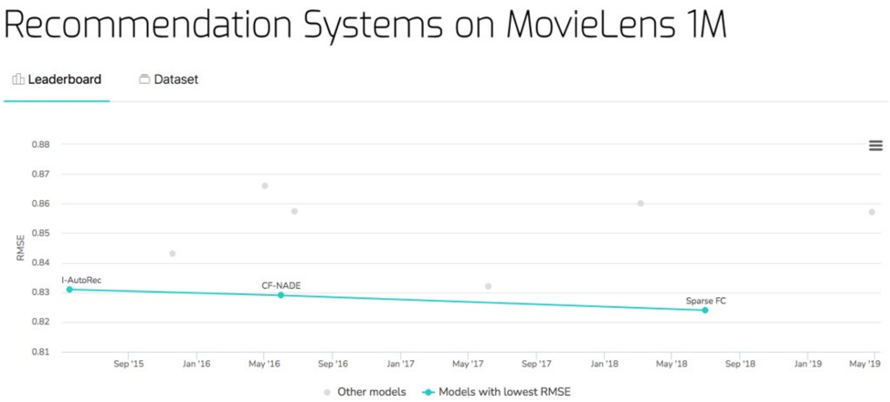
Examples

Benchmarks

Add a Result

TREND	DATASET	BEST METHOD	PAPER TITLE	PAPER	CODE	COMPARE
	MovieLens 1M	Bayesian timeSVD++ flipped	On the Difficulty of Evaluating Baselines: A Study on Recommender Systems			See all
	MovieLens 100K	Bayesian timeSVD++ flipped + Feat w/ Ordered Probit Regression	On the Difficulty of Evaluating Baselines: A Study on Recommender Systems			See all
	MovieLens 10M	Bayesian timeSVD++ flipped	On the Difficulty of Evaluating Baselines: A Study on Recommender Systems			See all
	MovieLens 20M	H+Vamp Gated	Enhancing VAEs for Collaborative Filtering: Flexible Priors & Gating Mechanisms			See all
	Netflix	H+Vamp Gated	Enhancing VAEs for Collaborative Filtering: Flexible Priors & Gating Mechanisms			See all
	Douban Monti	IGMC	Inductive Matrix Completion Based on Graph Neural Networks			See all
	Flixster Monti	IGMC	Inductive Matrix Completion Based on Graph Neural Networks			See all
	Million Song Dataset	EASE	Embarrassingly Shallow Autoencoders for Sparse Data			See all

Examples


Recommendation Systems on MovieLens 1M
Leaderboard
Dataset
0.88
0.87
0.86
0.85
0.84
0.83
0.82
0.81
RMSE
I-AutoRec
CF-NADE
Sparse FC
Sep '15
Jan '16
May '16
Sep '16
Jan '17
May '17
Sep '17
Jan '18
May '18
Sep '18
Jan '19
May '19
Other models
Models with lowest RMSE

Examples

View RMSE

Edit

RANK	MODEL	RMSE	NDCG@10	HR@10	NDCG	PAPER	CODE	RESULT	YEAR
1	Sparse FC	0.824				Kernelized Synaptic Weight Matrices			2018
2	CF-NADE	0.829				A Neural Autoregressive Approach to Collaborative Filtering			2016
3	I-AutoRec	0.831				AutoRec: Autoencoders Meet Collaborative Filtering			2015
4	GC-MC	0.832				Graph Convolutional Matrix Completion			2017
5	I-CFN	0.8321				Hybrid Recommender System based on Autoencoders			2016
6	NNMF	0.843				Neural Network Matrix Factorization			2015
7	IGMC	0.857				Inductive Matrix Completion Based on Graph Neural Networks			2019
8	U-CFN	0.8574				Hybrid Recommender System based on Autoencoders			2016
9	Factorized EAE	0.860				Deep Models of Interactions Across Sets			2018
10	Factorization with	0.866				Dictionary Learning for Massive Matrix Factorization			2016

Information Retrieval and Ranking

- Information Retrieval (IR)
  - Query: formal statement of information needs (e.g., search strings in web search engines)
  - In IR, a query does not uniquely identify a single object in the collection. Instead, several objects may match the query, perhaps with different degrees of relevance (ranking).

Information Retrieval and Ranking

- Information Retrieval (IR)
  - Query: formal statement of information needs (e.g., search strings in web search engines)
  - In IR, a query does not uniquely identify a single object in the collection. Instead, several objects may match the query, perhaps with different degrees of relevance (ranking).
- Application of ranking: Search Engine (Google, Baidu, Bing, etc)
  - Big data
  - Many available features: anchor texts, PageRank score, click through data
  - Using machine learning to rank queries is effective and popular
- Other applications: collaborative filtering, key term extraction, sentiment analysis, etc

Learning to Rank

- Learning to rank (L2R) is a supervised learning problem
- Training data of L2R
  - A set of queries $Q = \{q_1, \ldots, q_m\}$
  - A set of documents D
  - Documents relevant to the i-th query $D_i = \{d_{i,1}, \ldots, d_{i,n_i}\} \subseteq D$
  - A vector of relevance scores $y_i = (y_{i,1}, \ldots, y_{i,n_i})$ for each document relevant to query i

Learning to Rank

- Learning to rank (L2R) is a supervised learning problem
- Training data of L2R
  - A set of queries $Q = \{q_1, \ldots, q_m\}$
  - A set of documents D
  - Documents relevant to the i-th query $D_i = \{d_{i,1}, \ldots, d_{i,n_i}\} \subseteq D$
  - A vector of relevance scores $y_i = (y_{i,1}, \ldots, y_{i,n_i})$ for each document relevant to query i
- Goal of L2R: given a new query q, output a sorted list of (a permutation) of relevant documents

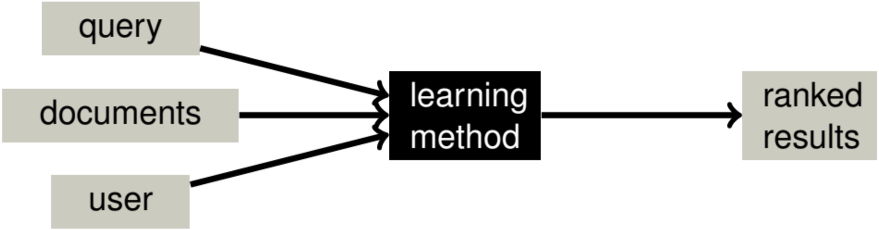
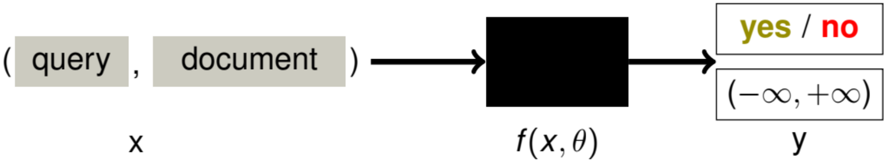


Image from Jannik Strötgen

Pointwise Modeling

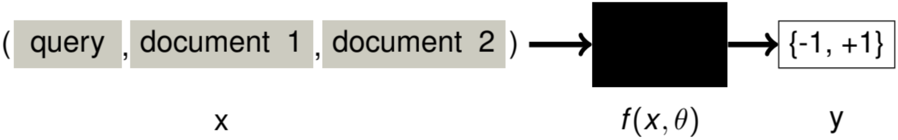


- Predict for every document separately
- x is the feature vector extracted from one query and one document
- y is the response (document goodness, e.g. label or measure of engagement)
- Learn a model f with parameters θ

Disadvantage: as the input is a single document, the relative order between documents cannot be naturally considered in the learning process.

Slide adapted from Jannik Strötgen

Pair-wise Modeling

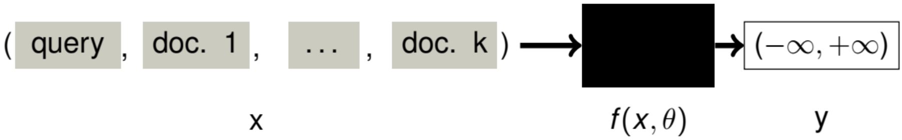


- Predict for every pair of documents jointly
- x is the feature vector extracted from one query and two documents
- y is the user's relative preference regarding the documents ($+1$ shows preference for document 1; -1 for document 2)

Disadvantage: no distinction between excellent-bad and fair-bad

Slide adapted from Jannik Strötgen

List-wise Modeling



- Predict for each ranked list of documents

Advantage: positional information visible to loss function
Disadvantage: high training complexity

Slide adapted from Jannik Strötgen

Evaluation for L2R

Benchmark dataset

- LETOR 2.0, 3.0, 4.0 (2007-2009) by Microsoft Research Asia
  - based on publicly available document collections
  - come with precomputed low-level features and relevance assessments
- Yahoo! Learning to Rank Challenge (2010) by Yahoo! Labs
  - comes with precomputed low-level features and relevance assessments
- Microsoft Learning to Rank Datasets by Microsoft Research U.S.
  - comes with precomputed low-level features and relevance assessments

Evaluation metric: MAP, NDCG, etc

Slide adapted from Jannik Strötgen

Learning to Rank Algorithms

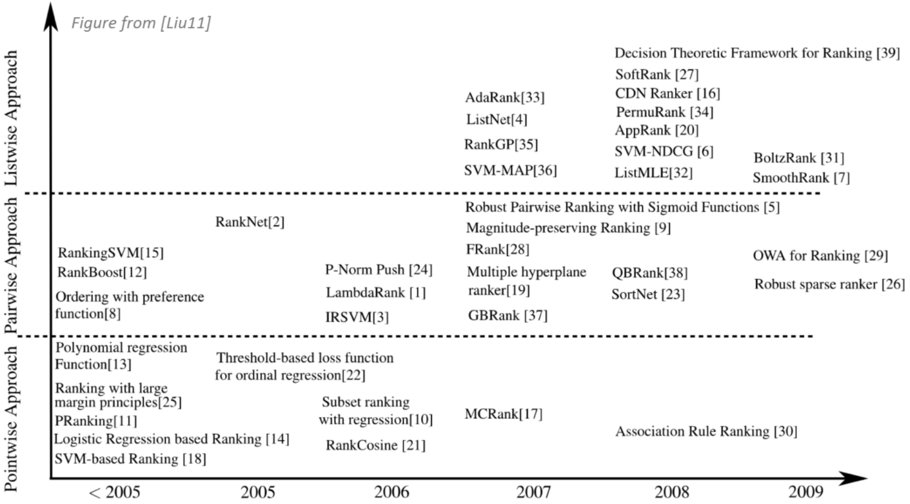


Chart from `http://ltr-tutorial-sigir19.isti.cnr.it/`

Examples

Ranking SVM

- Goal: learn h from $\{(\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, y_{i_1}, y_{i_2}) : (i_1, i_2) \in \mathcal{P}\}$ such that

$$h(\mathbf{x}_i) > h\left(\mathbf{x}_j\right) \iff y_i > y_j$$

Examples

Ranking SVM

- Goal: learn h from $\{(\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, y_{i_1}, y_{i_2}) : (i_1, i_2) \in \mathcal{P}\}$ such that

$$h(\mathbf{x}_i) > h\left(\mathbf{x}_j\right) \Longleftrightarrow y_i > y_j$$

- Optimization problem:

$$\begin{aligned} \underset{\mathbf{w}, \xi_{ij} \geq 0}{\text{minimize}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{m} \sum_{(i,j) \in \mathcal{P}} \xi_{ij} \\ \text{subject to} \quad & \left(\mathbf{w}^T\mathbf{x}_i\right) \geq \left(\mathbf{w}^T\mathbf{x}_j\right) + 1 - \xi_{ij}, \ \forall (i,j) \in \mathcal{P} \end{aligned}$$

More about ranking SVM:

https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html#References

https://arxiv.org/pdf/0704.3359.pdf

Examples

NDCG@10 on public LtR Datasets

Algorithm	MSN10K	Y!S1	Y!S2	Istella-S
RankingSVM	0.4012	0.7238	0.7306	N/A
GBRT	0.4602	**0.7555**	**0.7620**	0.7313
LambdaMART	**0.4618**	0.7529	0.7531	**0.7537**

Table from `http://ltr-tutorial-sigir19.isti.cnr.it/`

Learning Outcomes

- Know the types of recommendation system methods
- Know the two basic methods of collaborative filtering
- Understand the advantages and disadvantages of CF and CB
- Know the evaluation metrics for recommendation systems
- Be able to conduct CF and CB on some benchmark datasets
- Know the main ideas in learning to rank
- Be able to implement an algorithm of learning to rank