Tomáš Horváth

RECOMMENDER SYSTEMS

Tutorial at the conference

Znalosti 2012

October 14-16, 2012, Mikulov, Czech Republic

Institute of Computer Science, Faculty od Science
Pavol Jozef Šafárik University in Košice, Slovak Republic

Information Systems and Machine Learning Lab
University of Hildesheim, Germany

Contents

- Introduction
- Basic concepts
- Knowledge-based techniques
- Content-based techniques
- Collaborative-filtering
- Matrix factorization
- Issues worth to mention
- The MyMedialite library
- Summary

  . . . and, if still alive,
- Questions & Answers

Introduction

What is a RS?


Spongecell
Hula
kiko
Trumba
eskobo
mayomi
Pageflakes
vimeo
skobee
shadows
groovee
YouTube
Zimbra
LookSmart FURL | Your Personal Web File
smugmug
newsgator
Blogniscient
TinyFinger
shutterfly
PodDater
Feedster
favoor
Planzo.com
ZAZZLE
Tailrank
TagWorld
nuvvo
dogear
yakalike
grouper
ODDPOST
QOOP
iNods
Lulu
blish
flagr
FireAnt
simplyhired
Veoh
theadcloud
rbloc.com
pixagogo
gather
Agatra
browsr
oyogi
cafepress
Renkoo
standpoint
meebo
last.fm
YEDDA
Jotspot
Frappr!
jeteye
dabble
Writeboard
SHOUTWIRE
iKarma
AirSet
techmeme
memeorandum
CalendarHub
Suprglu
pando
zigtag
Findory
backfence
clipmarks
wayfaring
gOFFICE
AllPeers
orb
Rallypoint
Zoozio
blogbeat
Ziggs
zooto
vSocial
Boltfolio
wink
riya
Wordcast
Opinity
reddit
measuremap
gumshoo
bluepulse
BETA
STREAMLOAD
Ta-da Lists
FeedSky
jellyBam.INC
nativetext
CONGOO
PODZINGER
RSS MAD
FeedTier
phanfare
Fruitcast
PubSub
dPolls
flickr
Ning
Ookles
Strongspace
zoominfo
CASTPOST
WIKIPEDIA
yubnub
ourmedia
BLOOP
ProjectSpaces
FeedBurner
Bloglines
purevolume.com
FOTOLOG
Yub.com
Spot Runner
myspace
NewsAlloy
Allmydata.com
gabbr.com
Gcast
blinkx
openomy
riffs
ajchat
jambo
ROLLYO
ClipShack
chatsum
PANDORA
looklater
PROTOPAGE
squishr

Why do we need RS?

Why do we need RS?

A **company** wants to

- sell more (diverse) items
- increase users' satisfaction and fidelity
- better understand users' needs

Why do we need RS?

A **company** wants to

- sell more (diverse) items
- increase users' satisfaction and fidelity
- better understand users' needs

A **user** would like to

- find some (or all, in case of critical domains such as medicine) good items with a relatively small effort
- express herself by providing ratings or opinions
- help others by contribute with information to the community

The Big Bang

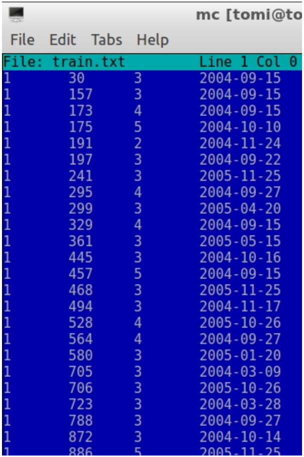
NETFLIX

- Contest begun on October 2, 2006
  - 100M ratings (1-5 stars) from 480K users on 18K movies
  - decrease RMSE of Cinematch (0.9525) at least with 10% ($\leq$0.8572)
- Grand Prize $1.000.000, Annual Progress Prizes $50.000

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace_	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				

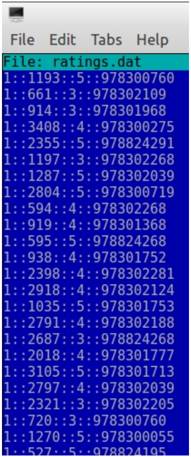
Netflix and Movielens data (1/2)

Netflix



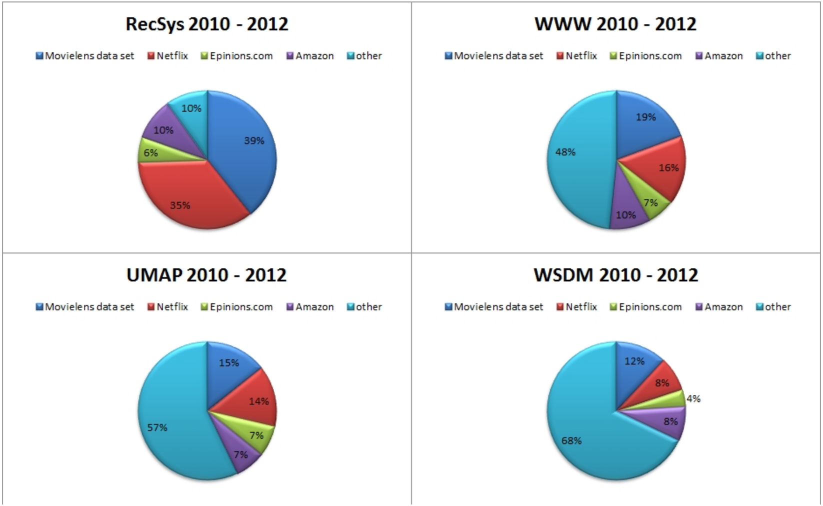
```
mc [tomi@to
File  Edit  Tabs  Help
File: train.txt          Line 1 Col 0
1       30      3       2004-09-15
1       157     3       2004-09-15
1       173     4       2004-09-15
1       175     5       2004-10-10
1       191     2       2004-11-24
1       197     3       2004-09-22
1       241     3       2005-11-25
1       295     4       2004-09-27
1       299     3       2005-04-20
1       329     4       2004-09-15
1       361     3       2005-05-15
1       445     3       2004-10-16
1       457     5       2004-09-15
1       468     3       2005-11-25
1       494     3       2004-11-17
1       528     4       2005-10-26
1       564     4       2004-09-27
1       580     3       2005-01-20
1       705     3       2004-03-09
1       706     3       2005-10-26
1       723     3       2004-03-28
1       788     3       2004-09-27
1       872     3       2004-10-14
1       886     5       2005-11-25
```

Movielens (100K, 1M)



```
File  Edit  Tabs  Help
File: ratings.dat
1::1193::5::978300760
1::661::3::978302109
1::914::3::978301968
1::3408::4::978300275
1::2355::5::978824291
1::1197::3::978302268
1::1287::5::978302039
1::2804::5::978300719
1::594::4::978302268
1::919::4::978301368
1::595::5::978824268
1::938::4::978301752
1::2398::4::978302281
1::2918::4::978302124
1::1035::5::978301753
1::2791::4::978302188
1::2687::3::978824268
1::2018::4::978301777
1::3105::5::978301713
1::2797::4::978302039
1::2321::3::978302205
1::720::3::978300760
1::1270::5::978300055
1::527::5::978824195
```

Netflix and Movielens data (2/2)



Closely related fields

Closely related fields

Information Retrieval

- unstructured data, various topics (IR) vs. repositories focused on a single topic (RS)
- relevant content for the query (IR) vs. relevant content for the user (RS)

Closely related fields

Information Retrieval

- unstructured data, various topics (IR) vs. repositories focused on a single topic (RS)
- relevant content for the query (IR) vs. relevant content for the user (RS)

Data mining & Machine Learning

- hardly measurable, subjective evaluation criteria (RS) besides some classic, objective evaluation measures (ML)

Closely related fields

Information Retrieval

- unstructured data, various topics (IR) vs. repositories focused on a single topic (RS)
- relevant content for the query (IR) vs. relevant content for the user (RS)

Data mining & Machine Learning

- hardly measurable, subjective evaluation criteria (RS) besides some classic, objective evaluation measures (ML)

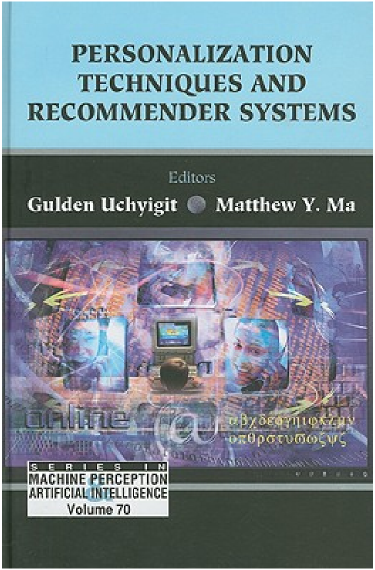
Human-Computer Interaction

- RS should convince the user to try the recommended items
- clear, transparent and trustworthy system logic
- provide details about recommended items and opportunity to refine recommendations

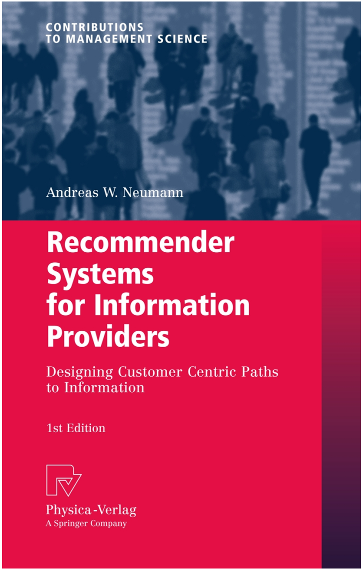
Related conferences

- ACM Recommender Systems (**RecSys**)
- User Modeling, Adaptation, and Personalization (**UMAP**)
- ACM Conference on Human Factors in Computing Systems (**CHI**)
- International World Wide Web Conference (**WWW**)
- ACM International Conference on Web Search and Data mining (**WSDM**)
- International Conference on Research and Development in Information Retrieval (**SIGIR**)
- ACM Conference on Information and Knowledge Management (**CIKM**)
- ...

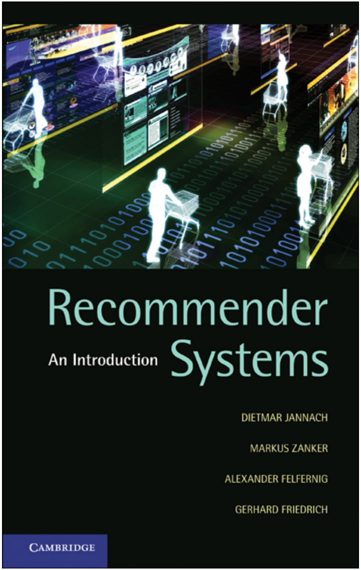
Textbook (2008)



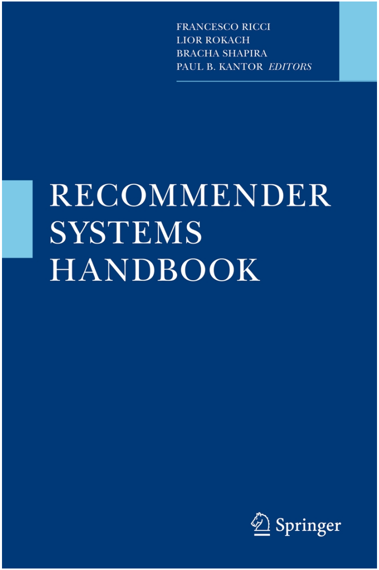
Textbook (2009)



Textbook (2010)



Textbook (2011)



Basic concepts

Users, Items and their characteristics

Users, Items and their characteristics

Users

- set of users $\mathcal{U}$
- user attributes $\mathcal{A}^{user} \subset \mathbb{R}^k$
  - age, income, marital status, education, profession, nationality, ...
  - preferred sport, hobbies, favourite movies, ...
- user characteristics $\chi^{user} : \mathcal{U} \to \mathcal{A}^{user}$
  - *sensitive* information, hard to obtain

Users, Items and their characteristics

Users

- set of users $\mathcal{U}$
- user attributes $\mathcal{A}^{user} \subset \mathbb{R}^k$
  - age, income, marital status, education, profession, nationality, ...
  - preferred sport, hobbies, favourite movies, ...
- user characteristics $\chi^{user} : \mathcal{U} \to \mathcal{A}^{user}$
  - *sensitive* information, hard to obtain

Items

- set of items $\mathcal{I}$
- item attributes $\mathcal{A}^{item} \subset \mathbb{R}^l$
  - movies: title, genre, year, director, actors, budget, nominations, ...
- item characteristics $\chi^{item} : \mathcal{I} \to \mathcal{A}^{item}$
  - quite *costly* to obtain

User feedback

$\phi : \mathcal{D} \to \mathcal{F}$

- feedback values $\mathcal{F} \subset \mathbb{R}$ observed on $\mathcal{D} \subset \mathcal{U} \times \mathcal{I}$

User feedback

$\phi : \mathcal{D} \rightarrow \mathcal{F}$

- feedback values $\mathcal{F} \subset \mathbb{R}$ observed on $\mathcal{D} \subset \mathcal{U} \times \mathcal{I}$

Implicit feedback

- information obtained about users by watching their natural *interaction with the system*
  - view, listen, scroll, bookmark, save, purchase, link, copy&paste, ...
- no burden on the user

User feedback

$\phi : \mathcal{D} \rightarrow \mathcal{F}$

- feedback values $\mathcal{F} \subset \mathbb{R}$ observed on $\mathcal{D} \subset \mathcal{U} \times \mathcal{I}$

Implicit feedback

- information obtained about users by watching their natural *interaction with the system*
  - view, listen, scroll, bookmark, save, purchase, link, copy&paste, ...
- no burden on the user

Explicit feedback

- *rating* items on a rating scale (Likert's scale)
- *scoring* items
- *ranking* a collection of items
- *pairwise ranking* of two presented items
- *provide* a list of preferred items

The recommendation task

Given

- $\mathcal{U}, \mathcal{I}$ and ϕ
- χ^{user}, χ^{item}
- some background knowledge κ

The recommendation task

Given

- $\mathcal{U}, \mathcal{I}$ and ϕ
- χ^{user}, χ^{item}
- some background knowledge κ

To learn

- **model** $\hat{\phi} : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal
  - a set of "unseen" (or future) user-item pairs $\mathcal{T} \subseteq (\mathcal{U} \times \mathcal{I}) \setminus \mathcal{D}$
  - acc is the accuracy of $\hat{\phi}$ w.r.t. ϕ measured on $\mathcal{T}$

The recommendation task

Given

- $\mathcal{U}, \mathcal{I}$ and ϕ
- χ^{user}, χ^{item}
- some background knowledge κ

To learn

- **model** $\hat{\phi} : \mathcal{U} \times \mathcal{I} \to \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal
  - a set of “unseen” (or future) user-item pairs $\mathcal{T} \subseteq (\mathcal{U} \times \mathcal{I}) \setminus \mathcal{D}$
  - acc is the accuracy of $\hat{\phi}$ w.r.t. ϕ measured on $\mathcal{T}$

It looks as a simple prediction task, **however**

The recommendation task

Given

- $\mathcal{U}, \mathcal{I}$ and ϕ
- χ^{user}, χ^{item}
- some background knowledge κ

To learn

- **model** $\hat{\phi} : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal
  - a set of "unseen" (or future) user-item pairs $\mathcal{T} \subseteq (\mathcal{U} \times \mathcal{I}) \setminus \mathcal{D}$
  - acc is the accuracy of $\hat{\phi}$ w.r.t. ϕ measured on $\mathcal{T}$

It looks as a simple prediction task, **however**

- χ^{user}, χ^{item} and κ are often unknown

The recommendation task

Given

- $\mathcal{U}, \mathcal{I}$ and ϕ
- χ^{user}, χ^{item}
- some background knowledge κ

To learn

- **model** $\hat{\phi} : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal
  - a set of "unseen" (or future) user-item pairs $\mathcal{T} \subseteq (\mathcal{U} \times \mathcal{I}) \setminus \mathcal{D}$
  - acc is the accuracy of $\hat{\phi}$ w.r.t. ϕ measured on $\mathcal{T}$

It looks as a simple prediction task, **however**

- χ^{user}, χ^{item} and κ are often unknown
- usually, $\mathcal{F} = \{1\}$ in case of implicit feedback

Two distinguished tasks

Two distinguished tasks

Rating prediction from explicit feedback

- How would Steve rate the movie Titanic more likely?

	Titanic	Pulp Fiction	Iron Man	Forrest Gump	The Mummy
Joe	1	4	5		3
Ann	5	1		5	2
Mary	4	1	2	5	
Steve	?	3	4		4

- $\hat{\phi}(u, i)$ – predicted rating of the user u for an item i

Two distinguished tasks

Rating prediction from explicit feedback

- How would Steve rate the movie Titanic more likely?

	Titanic	Pulp Fiction	Iron Man	Forrest Gump	The Mummy
Joe	1	4	5		3
Ann	5	1		5	2
Mary	4	1	2	5	
Steve	?	3	4		4

- $\hat{\phi}(u, i)$ – predicted rating of the user u for an item i

Item recommendation from implicit feedback

- Which movie(s) would does Steve see/buy more likely?

	Titanic	Pulp Fiction	Iron Man	Forrest Gump	The Mummy
Joe	1	1	1		1
Ann	1	1		1	1
Mary	1	1	1	1	
Steve	?	1	1	?	1

- $\hat{\phi}(u, i)$ – predicted likelihood of a "positive" implicit feedback (ranking score) of the user u for an item i

Types of RS

Types of RS

Knowledge-based

- recommendations are based on knowledge about users' needs and preferences
    - χ^{item}, κ, χ^{user}

Types of RS

Knowledge-based

- recommendations are based on knowledge about users' needs and preferences
  - χ^{item}, κ, χ^{user}

Content-based

- learn user's interests based on the features of items previously rated by the user, using supervised machine learning techniques
  - χ^{item}, ϕ

Types of RS

Knowledge-based

- recommendations are based on knowledge about users' needs and preferences
  - χ^{item}, κ, χ^{user}

Content-based

- learn user's interests based on the features of items previously rated by the user, using supervised machine learning techniques
  - χ^{item}, ϕ

Collaborative-filtering

- recognize similarities between users according to their feedbacks and recommend objects preferred by the like-minded users
  - ϕ (also χ^{item} and/or χ^{user} can be utilized)

Types of RS

Knowledge-based

- recommendations are based on knowledge about users' needs and preferences
  - χ^{item}, κ, χ^{user}

Content-based

- learn user's interests based on the features of items previously rated by the user, using supervised machine learning techniques
  - χ^{item}, ϕ

Collaborative-filtering

- recognize similarities between users according to their feedbacks and recommend objects preferred by the like-minded users
  - ϕ (also χ^{item} and/or χ^{user} can be utilized)

Hybrid

Knowledge-based techniques

Knowledge

Knowledge

user requirements

- value ranges
  - *"the maximal accepted price should be lower than 8K EUR"*
- functionality
  - *"the car should be safe and suited for a family"*

Knowledge

user requirements

- value ranges
  - *"the maximal accepted price should be lower than 8K EUR"*
- functionality
  - *"the car should be safe and suited for a family"*
- **interactive** recommendation process needed
  - conversational systems

Knowledge

user requirements

- value ranges
  - *"the maximal accepted price should be lower than 8K EUR"*
- functionality
  - *"the car should be safe and suited for a family"*
- **interactive** recommendation process needed
  - conversational systems

dependencies

- between user requirements and product properties
  - *"a family car should have big trunk size"*
- between different user requirements
  - *"if a safe family car is required the maximal accepted price must be higher than 2000 EUR"*

Representation

possible user requirements V_{user}

- max-price (0,..., 10K), usage (family, ...), safety (small, medium, big)

possible item characteristics V_{item}

- price (0,..., 100K), doors (3, 4, 5), terrain (yes, no), airbags (1, ..., 12)

compatibility constraints κ_C

- allowed instantiations of user properties
  - safety = big $\rightarrow$ max-price $\geq$ 2000

filter conditions κ_F

- item-specific selection criteriae
  - safety = big $\rightarrow$ airbags $>$ 4

item characteristics χ^{item}

- "item constraints"
  - (id=1 $\wedge$ price=4K $\wedge$ doors=3 $\wedge$ terrain=no $\wedge$ airbags=2) $\vee$ ...
    ... $\vee$ (id=100 $\wedge$ price=6K $\wedge$ doors=5 $\wedge$ terrain=no $\wedge$ airbags=6)

Recommendation

identifying products matching **user's requirements** REQ

- can be viewed as a kind of χ^{user}
- REQ = max-price=7000 $\wedge$ usage=family $\wedge$ safety=big

Recommendation

identifying products matching **user's requirements** REQ

- can be viewed as a kind of χ^{user}
- REQ = max-price=7000 $\wedge$ usage=family $\wedge$ safety=big

Constraint-based

- $RES = CSP(V_{user} \cup V_{item}, D, \kappa_C \cup \kappa_F \cup \chi^{item} \cup REQ)$
  - a set D of finite domains for V_{user} and V_{item}
  - RES = {max-price=7000, usage=family, safety=big, id=100, price=6K, doors=5, terrain=no, airbags=6)}

Recommendation

identifying products matching **user's requirements** REQ

- can be viewed as a kind of χ^{user}
- REQ = max-price=7000 $\wedge$ usage=family $\wedge$ safety=big

Constraint-based

- $RES = CSP(V_{user} \cup V_{item}, D, \kappa_C \cup \kappa_F \cup \chi^{item} \cup REQ)$
  - a set D of finite domains for V_{user} and V_{item}
  - RES = {max-price=7000, usage=family, safety=big, id=100, price=6K, doors=5, terrain=no, airbags=6)}

Conjunctive queries

- $\sigma_{[airbags \geq 4 \wedge price \leq 8000]}(\chi^{item})$

Recommendation

identifying products matching **user's requirements** REQ

- can be viewed as a kind of χ^{user}
- REQ = max-price=7000 $\wedge$ usage=family $\wedge$ safety=big

Constraint-based

- $RES = CSP(V_{user} \cup V_{item}, D, \kappa_C \cup \kappa_F \cup \chi^{item} \cup REQ)$
  - a set D of finite domains for V_{user} and V_{item}
  - RES = {max-price=7000, usage=family, safety=big, id=100, price=6K, doors=5, terrain=no, airbags=6)}

Conjunctive queries

- $\sigma_{[airbags \geq 4 \wedge price \leq 8000]}(\chi^{item})$

Case-based

- $similarity(i, REQ) = \sum_{r \in REQ} w_r.sim(i,r) / \sum_{r \in REQ} w_r$
  - weight w_r for requirements r
  - similarity $sim(i,r)$ of items $i \in \chi^{item}$ to requirements $r \in REQ$
    - different types of $sim(i,r)$
    - user might maximize (e.g. safety) or minimize (e.g. price)

Interaction – default requirement values

Interaction – default requirement values

static defaults for each user property

- default(usage)=family

Interaction – default requirement values

static defaults for each user property

- default(usage)=family

dependent defaults on combinations of user requirements

- default(usage=family, max-price=6000)

Interaction – default requirement values

static defaults for each user property

- default(usage)=family

dependent defaults on combinations of user requirements

- default(usage=family, max-price=6000)

derived defaults from user requirements log

- the known requirement of the current user is REQ={price=6000}
- nearest-neighbor
  - 1-NN: REQ={price=6000, doors=5, terrain=no, airbags=6}
  - 3-NN: REQ={price=6000, doors=4, terrain=no, airbags=4}

user id	price	doors	terrain	airbags
1	6000	5	no	6
2	2000	3	yes	2
3	5500	4	yes	4
4	6500	4	no	4

Interaction – unsatisfiable requirements

Which of the requirements should be changed?

[1] D. Jannach (2006). Finding Preferred Query Relaxations in Content-based Recommenders. IEEE Int. Conf. on Intelligent Systems, pp.355-360.

Interaction – unsatisfiable requirements

Which of the requirements should be changed?

- the **MinRelax**[1] algorithm

PQRS = compute the partial query results for all atoms
  a_i of Q for the product catalog P
MinRS = $\emptyset$
forall $p_i \in P$ **do**
  PSX = Compute the product-specific relaxation
    $PSX(Q, p_i)$ by using PQRS
  % Check relaxations that were already found
  SUB = $\{r \in MinRS \mid r$ is subquery of $PSX\}$
  if $SUB \neq \emptyset$
    % Current relaxation is superset of existing
    continue with next p_i
  endif
  SUPER = $\{r \in MinRS \mid$ PSX is subquery of $r\}$
  if $SUPER \neq \emptyset$
    % Remove supersets
    $MinRS = MinRS \setminus SUPER$
  endif
  % Store the new relaxation
  $MinRS = MinRS \cup PSX$
endfor
return MinRS

[1] D. Jannach (2006). Finding Preferred Query Relaxations in Content-based Recommenders. IEEE Int. Conf. on Intelligent Systems, pp.355-360.

Interaction – unsatisfiable requirements

Which of the requirements should be changed?

- the **MinRelax**[1] algorithm

```
PQRS = compute the partial query results for all atoms
    a_i of Q for the product catalog P
MinRS = ∅
forall p_i ∈ P do
    PSX = Compute the product-specific relaxation
            PSX(Q, p_i) by using PQRS
    % Check relaxations that were already found
    SUB = {r ∈ MinRS | r is subquery of PSX}
    if SUB ≠ ∅
        % Current relaxation is superset of existing
        continue with next p_i
    endif
    SUPER = {r ∈ MinRS | PSX is subquery of r}
    if SUPER ≠ ∅
        % Remove supersets
        MinRS = MinRS \ SUPER
    endif
    % Store the new relaxation
    MinRS = MinRS ∪ PSX
endfor
return MinRS
```

REQ={r_1:price≤6000, r_2:doors=5, r_3:terrain=no, r_4:airbags≥6}

- $\sigma_{[r_1 \wedge r_2 \wedge r_3 \wedge r_4]}(\chi^{item}) = \emptyset$
- partial query results $PQRS$

req	i_1	i_2	i_3	i_4
1	1	0	1	0
2	0	1	0	1
3	0	0	1	0
4	1	1	0	1

- product-specific relaxation
  - PSX(REQ,i_1)={r_2,r_3}

[1] D. Jannach (2006). Finding Preferred Query Relaxations in Content-based Recommenders. IEEE Int. Conf. on Intelligent Systems, pp.355-360.

Interaction – repairs for unsatisfiable requirements

How should the unsatisfiable requirements be changed?

Interaction – repairs for unsatisfiable requirements

How should the unsatisfiable requirements be changed?

- derive **repair actions** using $MinRS$

  for each $r \in MinRS$ derive $\pi_{[attributes(r)]}\sigma_{[REQ-r]}(\chi^{item})$

Interaction – repairs for unsatisfiable requirements

How should the unsatisfiable requirements be changed?

- derive **repair actions** using $MinRS$

  for each $r \in MinRS$ derive $\pi_{[attributes(r)]}\sigma_{[REQ-r]}(\chi^{item})$

REQ=\{r_1:price$\leq$3000, r_2:doors=5, r_3:terrain=yes, r_4:airbags$\geq$6\}

- $MinRS = \{\{r_2, r_4\}, \{r_2, r_3\}\}$
  - $\pi_{[doors,airbags]}\sigma_{[r_1,r_3]}(\chi^{item}) =$
    $\{(doors = 3, airbags = 4), (doors = 4, airbags = 2)\}$
  - $\pi_{[doors,terrain]}\sigma_{[r_1,r_4]}(\chi^{item}) =$
    $\{(doors = 4, terrain = no)\}$

Interaction – repairs for unsatisfiable requirements

How should the unsatisfiable requirements be changed?

- derive **repair actions** using $MinRS$

  for each $r \in MinRS$ derive $\pi_{[attributes(r)]}\sigma_{[REQ-r]}(\chi^{item})$

REQ={r_1:price≤3000, r_2:doors=5, r_3:terrain=yes, r_4:airbags≥6}

- $MinRS = \{\{r_2, r_4\}, \{r_2, r_3\}\}$
  - $\pi_{[doors,airbags]}\sigma_{[r_1,r_3]}(\chi^{item}) =$
    $\{(doors = 3, airbags = 4), (doors = 4, airbags = 2)\}$
  - $\pi_{[doors,terrain]}\sigma_{[r_1,r_4]}(\chi^{item}) =$
    $\{(doors = 4, terrain = no)\}$

- repair alternatives
  - REQ={r_1:price≤3000, r_2:doors=3, r_3:terrain=yes, r_4:airbags=4}
  - REQ={r_1:price≤3000, r_2:doors=4, r_3:terrain=yes, r_4:airbags=2}
  - REQ={r_1:price≤3000, r_2:doors=4, r_3:terrain=no, r_4:airbags=6}

Interaction – ranking the retrieved items (1/2)

Contributions

- pre-defined set of dimensions

	value	**quality**	**economy**	**safety**
price	$\langle 0, 3000)$	2	3	3
	$\langle 3000, 7000)$	3	2	4
	≥ 7000	5	1	5
terrain	yes	3	2	3
	no	2	4	2
airbags	0	1	5	1
	2	2	4	2
	...		...	
doors	3	3	5	2
	...		...	

Interaction – ranking the retrieved items (1/2)

Contributions

- pre-defined set of dimensions

	value	**quality**	**economy**	**safety**
price	$\langle 0, 3000)$	2	3	3
	$\langle 3000, 7000)$	3	2	4
	≥ 7000	5	1	5
terrain	yes	3	2	3
	no	2	4	2
airbags	0	1	5	1
	2	2	4	2
	...		...	
doors	3	3	5	2
	...		...	

- $contribution(item, dimension)$
  - i = (price=4000 ∧ terrain=no ∧ airbags=2 ∧ doors=3)
  - contribution(i,quality) = 3+2+2+3 = 10, ...

Interaction – ranking the retrieved items (2/2)

Interest of the user in pre-defined dimensions

- user-defined
  - interest(quality) = 0.3
  - interest(economy) = 0.6
  - interest(safety) = 0.1

Interaction – ranking the retrieved items (2/2)

Interest of the user in pre-defined dimensions

- user-defined
  - interest(quality) = 0.3
  - interest(economy) = 0.6
  - interest(safety) = 0.1
- derived from requirements
  - REQ = $\{$price=4000 $\wedge$ airbags=2$\}$
    - contribution(req,quality) = 3+2 = 5
    - contribution(req,economy) = 2+4 = 6
    - contribution(req,safety) = 4+2 = 6
  - interest(quality) = 5/(5+6+6) = 5/17 = 0.3
  - interest(economy) = interest(safety) = 6/17 = 0.35
- other approaches

Interaction – ranking the retrieved items (2/2)

Interest of the user in pre-defined dimensions

- user-defined
  - interest(quality) = 0.3
  - interest(economy) = 0.6
  - interest(safety) = 0.1
- derived from requirements
  - REQ = {price=4000 $\wedge$ airbags=2}
    - contribution(req,quality) = 3+2 = 5
    - contribution(req,economy) = 2+4 = 6
    - contribution(req,safety) = 4+2 = 6
  - interest(quality) = 5/(5+6+6) = 5/17 = 0.3
  - interest(economy) = interest(safety) = 6/17 = 0.35
- other approaches

$$utility(i) = \sum_{d \in dimensions} interest(d).contribution(i, d)$$

Interaction – Critiquing

Interaction – Critiquing

a **browsing-based** approach used in case-based systems

- requirements refined w.r.t. the recommended item
  - *"Show me cheaper cars"* ... *"cars with more airbags"* ...

Interaction – Critiquing

a **browsing-based** approach used in case-based systems

- requirements refined w.r.t. the recommended item
    - *"Show me cheaper cars"* ... *"cars with more airbags"* ...
- unit vs. compound critiques
    - static (user wants more airbags but there are no such cars)

Interaction – Critiquing

a **browsing-based** approach used in case-based systems

- requirements refined w.r.t. the recommended item
  - *"Show me cheaper cars"* … *"cars with more airbags"* …
- unit vs. compound critiques
  - static (user wants more airbags but there are no such cars)

Dynamic critiquing

- suggests **critique patterns** according to the candidate items
  - association rules ($>_{price} \rightarrow <_{doors}$)
  - compound critique patterns ($>_{price} \wedge <_{doors}$)

	price	doors	terrain	airbags
entry item	3600	5	no	4
candidate item 1	4500	3	no	4
candidate item 2	5600	4	yes	6
…		…		
critique pattern 1	$>$	$<$	$\neq$	$=$
critique pattern 2	$>$	$<$	$=$	$>$
…		…		

Content-based techniques

Content

Item features/characteristics (χ^{item})

- explicitly **defined**
  - attributes (price, airbags, doors, ...)

Content

Item features/characteristics (χ^{item})

- explicitly **defined**
  - attributes (price, airbags, doors, ...)
- implicitly **computed** from the document $d \in \mathcal{D}$
  - keywords w, boolean vector space model ...

Content

Item features/characteristics (χ^{item})

- explicitly **defined**
  - attributes (price, airbags, doors, ...)
- implicitly **computed** from the document $d \in \mathcal{D}$
  - keywords w, boolean vector space model ...

$$TF-IDF(w,d) = TF(w,d) \,.\, IDF(w,\mathcal{D})$$

- term frequency

$$TF(w,d) = \frac{freq(w,d)}{max\{freq(w',d)|w' \neq w\}}$$

- inverse document frequency

$$IDF(w,\mathcal{D}) = log\frac{|\mathcal{D}|}{|\{d \in \mathcal{D}|w \in d\}|}$$

Content

Item features/characteristics (χ^{item})

- explicitly **defined**
  - attributes (price, airbags, doors, ...)
- implicitly **computed** from the document $d \in \mathcal{D}$
  - keywords w, boolean vector space model ...

$$TF-IDF(w,d) = TF(w,d)\,.\,IDF(w,\mathcal{D})$$

- term frequency

$$TF(w,d) = \frac{freq(w,d)}{max\{freq(w',d)|w' \neq w\}}$$

- inverse document frequency

$$IDF(w,\mathcal{D}) = log\frac{|\mathcal{D}|}{|\{d \in \mathcal{D}|w \in d\}|}$$

- $\chi^{item} = (TF-IDF(w_1, item), \ldots, TF-IDF(w_k, item))$

Similarity-based recommendation

How to check if a user would like an item?

Similarity-based recommendation

How to check if a user would like an item?

- If she **liked similar** items in the past...
  - feedback and similarity measures needed

cosine vector similarity

$$sim_{cv}(\chi^i, \chi^j) = \frac{\chi^i \cdot \chi^j}{\|\chi^i\| . \|\chi^j\|} = \frac{\sum_{k=1}^{n} \chi_k^i \, \chi_k^j}{\sqrt{\sum_{k=1}^{n} {\chi_k^i}^2} \sqrt{\sum_{k=1}^{n} {\chi_k^j}^2}}$$

Similarity-based recommendation

How to check if a user would like an item?

- If she **liked similar** items in the past...
    - feedback and similarity measures needed

cosine vector similarity

$$sim_{cv}(\chi^i, \chi^j) = \frac{\chi^i \cdot \chi^j}{\|\chi^i\|.\|\chi^j\|} = \frac{\sum_{k=1}^{n} \chi_k^i \, \chi_k^j}{\sqrt{\sum_{k=1}^{n} {\chi_k^i}^2} \sqrt{\sum_{k=1}^{n} {\chi_k^j}^2}}$$

k-nearest-neighbors

- k most similar items user has got feedback on
    - recommend an item according to majority vote/average/etc.

Similarity-based recommendation

How to check if a user would like an item?

- If she **liked similar** items in the past...
  - feedback and similarity measures needed

cosine vector similarity

$$sim_{cv}(\chi^i, \chi^j) = \frac{\chi^i \cdot \chi^j}{\|\chi^i\| . \|\chi^j\|} = \frac{\sum_{k=1}^{n} \chi_k^i \, \chi_k^j}{\sqrt{\sum_{k=1}^{n} {\chi_k^i}^2} \sqrt{\sum_{k=1}^{n} {\chi_k^j}^2}}$$

k-nearest-neighbors

- k most similar items user has got feedback on
  - recommend an item according to majority vote/average/etc.
- reflects on short-term preferences
  - considering only recent feedbacks
- simple to implement, small number of feedbacks is enough

Relevance feedback

Rocchio's method

- find a **prototype** of "user's ideal item"
- user-defined queries refined **iteratively**
  - good results already after the first iteration
- vector space model and similarity measure

Relevance feedback

Rocchio's method

- find a **prototype** of "user's ideal item"
- user-defined queries refined **iteratively**
  - good results already after the first iteration
- vector space model and similarity measure

input for $i+1$-th iteration

- $\mathcal{D}^-$ – documents with negative user feedback
- $\mathcal{D}^+$ – documents with positive user feedback
- Q_i – actual query (**vector**) in the iteration i
- α, β, γ – parameters

Relevance feedback

Rocchio's method

- find a **prototype** of "user's ideal item"
- user-defined queries refined **iteratively**
  - good results already after the first iteration
- vector space model and similarity measure

input for $i+1$-th iteration

- $\mathcal{D}^-$ – documents with negative user feedback
- $\mathcal{D}^+$ – documents with positive user feedback
- Q_i – actual query (**vector**) in the iteration i
- α, β, γ – parameters

$$Q_{i+1} = \alpha Q_i + \beta\Big(\frac{1}{|\mathcal{D}^+|}\sum_{d^+ \in \mathcal{D}^+} d^+\Big) + \gamma\Big(\frac{1}{|\mathcal{D}^-|}\sum_{d^- \in \mathcal{D}^-} d^-\Big)$$

Machine learning

learn a mapping $\hat{\phi} : \mathcal{A}^{item} \to \mathbb{R}$ from

- item features/characteristics χ^{item}
- user's feedback ϕ

with approppriate **classification/regression** techniques

- nearest-neighbor
- probabilistic methods
- decision trees, SVM
- …

item	$\mathcal{A}^i$	$\phi(u, item)$
i_1	$\chi^{item}(i_1)$	$\phi(u, i_1)$
i_2	$\chi^{item}(i_2)$	$\phi(u, i_2)$
$\vdots$	$\vdots$	$\vdots$
i_n	$\chi^{item}(i_n)$	$\phi(u, i_n)$

A little commercial ;)

A fuzzy recommender system

First prototype developed during the NAZOU[1] project (2006 – 2008)

- 2009 – 2012, developed without funding (BSc, MSc theses)
- 2012 – now, development within the CEZIS project

[1] http://nazou.fiit.stuba.sk/

[2] P. Gurský et al. (2008). Knowledge Processing for Web Search – An Integrated Model and Experiments. SCALABLE COMPUTING: PRACTICE AND EXPERIMENTS Vol. 9 (1).

A fuzzy recommender system

First prototype developed during the NAZOU[1] project (2006 – 2008)

- 2009 – 2012, developed without funding (BSc, MSc theses)
- 2012 – now, development within the CEZIS project

Main characteristics of the UPRE recommender module[2]

- **fuzzy preference models**
  - on attributes (local)
  - aggregated (global)
  - top-k item retrieval
- explicit user feedback
- conversational

[1] http://nazou.fiit.stuba.sk/

[2] P. Gurský et al. (2008). Knowledge Processing for Web Search – An Integrated Model and Experiments. SCALABLE COMPUTING: PRACTICE AND EXPERIMENTS Vol. 9 (1).

A fuzzy recommender system

First prototype developed during the NAZOU[1] project (2006 – 2008)

- 2009 – 2012, developed without funding (BSc, MSc theses)
- 2012 – now, development within the CEZIS project

Main characteristics of the UPRE recommender module[2]

- **fuzzy preference models**
  - on attributes (local)
  - aggregated (global)
  - top-k item retrieval
- explicit user feedback
- conversational

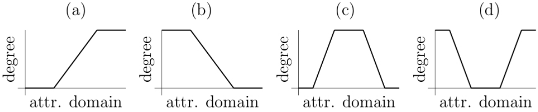
A **hybrid** of content-based and knowledge-based techniques...

- collaborative-filtering is planned

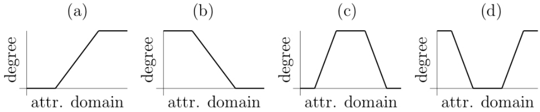
[1] http://nazou.fiit.stuba.sk/

[2] P. Gurský et al. (2008). Knowledge Processing for Web Search – An Integrated Model and Experiments. SCALABLE COMPUTING: PRACTICE AND EXPERIMENTS Vol. 9 (1).

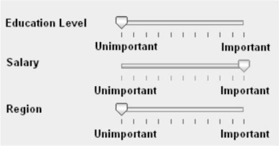
Preferences on attributes

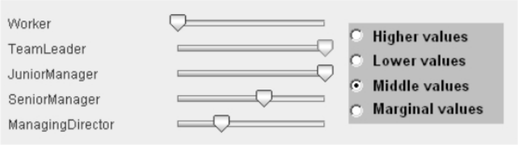


Preferences on attributes

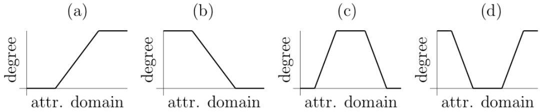


defined by the user

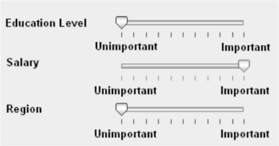




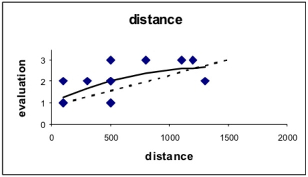
Preferences on attributes

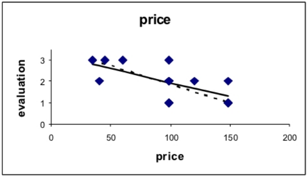


defined by the user



computed from explicit feedback





Aggregated preferences

computed[1] with **monotone prediction** techniques

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\mathbb{D} \arrow[r, "\preceq^u_{\mathbb{D}} \equiv p^u_G"] \arrow[d, "data_{\mathbb{R}}"'] & {[0,1]} \\
\prod\langle \mathbb{D}_i, \preceq^u_{\mathbb{D}_i} \rangle \arrow[r, "p^u_{L_1}, \ldots, p^u_{L_n} \approx \preceq^u_{\mathbb{D}_1}, \ldots, \preceq^u_{\mathbb{D}_n}"] \arrow[ur, dash, "\#1", "\#2"', "monotone\ dataset\ \mathbb{R}" description] & {[0,1]^n} \arrow[u, "@ \approx p^u_G"']
\end{tikzcd}
$$

[1] T. Horváth(2009). A Model of User Preference Learning for Content-Based Recommender Systems. COMPUTING AND INFORMATICS Vol. 28 (4).

[2] Gurský et al. (2008). Fuzzy User Preference Model for Top-k Search. IEEE World Congress on Computational Intelligence.

Aggregated preferences

computed[1] with **monotone prediction** techniques

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\mathbb{D} \arrow[rr, "\preceq^u_{\mathbb{D}} \equiv p^u_G"] \arrow[d, "data_{\mathbb{R}}"'] \arrow[dr, phantom, "\#1"] & & {[0,1]} \\
\prod \langle \mathbb{D}_i, \preceq^u_{\mathbb{D}_i} \rangle \arrow[urr, dash, "\textit{monotone dataset } \mathbb{R}" description] \arrow[rr, "p^u_{L_1}, \ldots, p^u_{L_n} \approx \preceq^u_{\mathbb{D}_1}, \ldots, \preceq^u_{\mathbb{D}_n}"'] & \arrow[ur, phantom, "\#2"] & {[0,1]^n} \arrow[u, "@ \approx p^u_G"']
\end{tikzcd}
$$

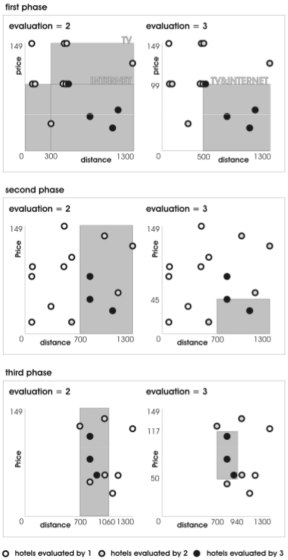
preference rules integrated[2] with **top-k search**

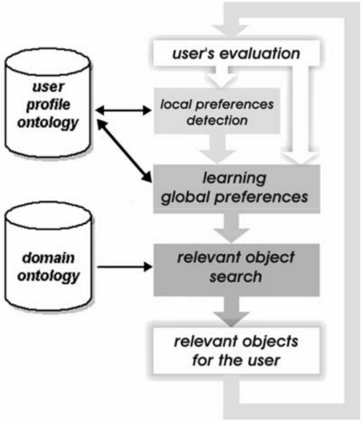
- fast computation of pareto-optimal values
    - implicit ranking of items in the resulting list

[1] T. Horváth(2009). A Model of User Preference Learning for Content-Based Recommender Systems. COMPUTING AND INFORMATICS Vol. 28 (4).

[2] Gurský et al. (2008). Fuzzy User Preference Model for Top-k Search. IEEE World Congress on Computational Intelligence.

Iterative recommendation





Collaborative filtering

Neighborhood-based CF

Recommendation $\hat{\phi}(u, i)$ for user u on item i using ϕ

Neighborhood-based CF

Recommendation $\hat{\phi}(u,i)$ for user u on item i using ϕ

- user-based
  - $\hat{\phi}(u,i)$ computed using feedback given by k **most similar users**

$$\mathcal{N}_i^{u,k} = \arg\max_{\mathcal{U}'} \sum_{\substack{v \in \mathcal{U}', v \neq u \\ \mathcal{U}' \subseteq \mathcal{U}_i, |\mathcal{U}'| = k}} sim(u,v)$$

- $\mathcal{U}_i = \{v \in \mathcal{U} \mid \phi(v,i) \text{ is defined on } \mathcal{D}\}$

Neighborhood-based CF

Recommendation $\hat{\phi}(u,i)$ for user u on item i using ϕ

- user-based
  - $\hat{\phi}(u,i)$ computed using feedback given by k **most similar users**

$$\mathcal{N}_i^{u,k} = \arg\max_{\mathcal{U}'} \sum_{\substack{v \in \mathcal{U}', v \neq u \\ \mathcal{U}' \subseteq \mathcal{U}_i, |\mathcal{U}'| = k}} sim(u,v)$$

    - $\mathcal{U}_i = \{v \in \mathcal{U} \mid \phi(v,i) \text{ is defined on } \mathcal{D}\}$

- item-based
  - $\hat{\phi}(u,i)$ computed using feedback given by k **most similar items**

$$\mathcal{N}_u^{i,k} = \arg\max_{\mathcal{I}'} \sum_{\substack{j \in \mathcal{I}', j \neq i \\ \mathcal{I}' \subseteq \mathcal{I}_u, |\mathcal{I}'| = k}} sim(i,j)$$

    - $\mathcal{I}_u = \{j \in I \mid \phi(u,j) \text{ is defined on } \mathcal{D}\}$

Item recommendation

What is the likelihood of an item i being liked by the user u?

[1] Simplified notation: $\phi(u,i) \rightsquigarrow \phi_{ui}$, $\mathcal{I}_u \cap \mathcal{I}_v \rightsquigarrow \mathcal{I}_{uv}$, $\mathcal{U}_i \cap \mathcal{U}_j \rightsquigarrow \mathcal{U}_{ij}$

Item recommendation

What is the likelihood of an item i being liked by the user u?

- a simple **k-nearest-neighbor** approach[1]

[1] Simplified notation: $\phi(u,i) \rightsquigarrow \phi_{ui}$, $\mathcal{I}_u \cap \mathcal{I}_v \rightsquigarrow \mathcal{I}_{uv}$, $\mathcal{U}_i \cap \mathcal{U}_j \rightsquigarrow \mathcal{U}_{ij}$

Item recommendation

What is the likelihood of an item i being liked by the user u?

- a simple **k-nearest-neighbor** approach[1]
  - user-based
    - an average similarity of most similar users which liked the item i

$$\hat{\phi}_{ui} = \frac{\sum_{v \in \mathcal{N}_i^{u,k}} sim(u,v)}{k}$$

[1] Simplified notation: $\phi(u,i) \rightsquigarrow \phi_{ui}$, $\mathcal{I}_u \cap \mathcal{I}_v \rightsquigarrow \mathcal{I}_{uv}$, $\mathcal{U}_i \cap \mathcal{U}_j \rightsquigarrow \mathcal{U}_{ij}$

Item recommendation

What is the likelihood of an item i being liked by the user u?

- a simple **k-nearest-neighbor** approach[1]
  - user-based
    - an average similarity of most similar users which liked the item i

$$\hat{\phi}_{ui} = \frac{\sum_{v \in \mathcal{N}_i^{u,k}} sim(u,v)}{k}$$

  - item-based
    - an average similarity of most similar items liked by the user u

$$\hat{\phi}_{ui} = \frac{\sum_{j \in \mathcal{N}_u^{i,k}} sim(i,j)}{k}$$

[1] Simplified notation: $\phi(u,i) \rightsquigarrow \phi_{ui}$, $\mathcal{I}_u \cap \mathcal{I}_v \rightsquigarrow \mathcal{I}_{uv}$, $\mathcal{U}_i \cap \mathcal{U}_j \rightsquigarrow \mathcal{U}_{ij}$

Item recommendation

What is the likelihood of an item i being liked by the user u?

- a simple **k-nearest-neighbor** approach[1]
  - user-based
    - an average similarity of most similar users which liked the item i

$$\hat{\phi}_{ui} = \frac{\sum_{v \in \mathcal{N}_i^{u,k}} sim(u,v)}{k}$$

  - item-based
    - an average similarity of most similar items liked by the user u

$$\hat{\phi}_{ui} = \frac{\sum_{j \in \mathcal{N}_u^{i,k}} sim(i,j)}{k}$$

assume that only (implicit) feedback ϕ is available

- users and items represented by **sparse vectors**
  - cosine-vector similarity sim_{cv}

[1] Simplified notation: $\phi(u,i) \rightsquigarrow \phi_{ui}$, $\mathcal{I}_u \cap \mathcal{I}_v \rightsquigarrow \mathcal{I}_{uv}$, $\mathcal{U}_i \cap \mathcal{U}_j \rightsquigarrow \mathcal{U}_{ij}$

Item recommendation – example

$sim_{cv}(i,j)$	Titanic	Pulp Fiction	Iron Man	Forrest Gump	The Mummy
Titanic	1.0	0.87	0.67	0.82	0.67
Pulp Fiction	–	1.0	0.87	0.71	0.87
Iron Man	–	–	1.0	0.41	0.67
Forrest Gump	–	–	–	1.0	0.41
The Mummy	–	–	–	–	1.0

$sim_{cv}(u,v)$	Joe	Ann	Mary	Steve
Joe	1.0	0.75	0.75	0.87
Ann	–	1.0	0.75	0.58
Mary	–	–	1.0	0.58
Steve	–	–	–	1.0

user-based[1]

- $\mathcal{N}_{Titanic}^{Steve,2} = \{Joe, Ann\}$, $\hat{\phi}_{ST} = \frac{s_{cv}(S,J)+s_{cv}(S,M)}{2} = \frac{0.87+0.58}{2} = 0.725$
- $\mathcal{N}_{ForrestGump}^{Steve,2} = \{Ann, Mary\}$, $\hat{\phi}_{ST} = \frac{s_{cv}(S,A)+s_{cv}(S,M)}{2} = \frac{0.58+0.58}{2} = 0.58$

item-based

- $\mathcal{N}_{Steve}^{Titanic,2} = \{PulpFiction, IronMan\}$, $\hat{\phi}_{ST} = \frac{s_{cv}(T,P)+s_{cv}(T,I)}{2} = \frac{0.87+0.67}{2} = 0.77$
- $\mathcal{N}_{Steve}^{ForrestGump,2} = \{PulpFiction, IronMan\}$, $\hat{\phi}_{ST} = \frac{s_{cv}(F,P)+s_{cv}(F,I)}{2} = \frac{0.71+0.41}{2} = 0.56$

[1] s_{cv} – cosine–vector similarity

Rating prediction

How would the user rate an item?

Rating prediction

How would the user rate an item?

- user's/item's ratings are **biased**
  - optimistic, pessimistic users
  - items rated above or below average

Rating prediction

How would the user rate an item?

- user's/item's ratings are **biased**
  - optimistic, pessimistic users
  - items rated above or below average

mean-centered rating prediction

- user-based

$$\hat{\phi}_{ui} = \overline{\phi}_u + \frac{\sum_{v \in \mathcal{N}_i^{u,k}} sim(u,v) \cdot (\phi_{vi} - \overline{\phi}_v)}{\sum_{v \in \mathcal{N}_i^{u,k}} |sim(u,v)|}$$

  - $\overline{\phi}_u = \frac{\sum_{i \in \mathcal{I}_u} \phi(u,i)}{|\mathcal{I}_u|}$
- item-based

$$\hat{\phi}_{ui} = \overline{\phi}_i + \frac{\sum_{j \in \mathcal{N}_u^{i,k}} sim(i,j) \cdot (\phi_{uj} - \overline{\phi}_j)}{\sum_{v \in \mathcal{N}_u^{i,k}} |sim(i,j)|}$$

  - $\overline{\phi}_i = \frac{\sum_{u \in \mathcal{U}_i} \phi(u,i)}{|\mathcal{U}_i|}$

Pearson-correlation similarity

What similarity measure to use?

- sim_{cv} doesn't take into account the mean and variances of ratings

Pearson-correlation similarity

What similarity measure to use?

- sim_{cv} doesn't take into account the mean and variances of ratings

pearson-correlation similarity

$$sim_{pc}(u,v) = \frac{\sum_{i \in \mathcal{I}_{uv}} (\phi_{ui} - \overline{\phi}_u)(\phi_{vi} - \overline{\phi}_v)}{\sqrt{\sum_{i \in \mathcal{I}_{uv}} (\phi_{ui} - \overline{\phi}_u)^2 \; \sum_{i \in \mathcal{I}_{uv}} (\phi_{vi} - \overline{\phi}_v)^2}}$$

$$sim_{pc}(i,j) = \frac{\sum_{u \in \mathcal{U}_{ij}} (\phi_{ui} - \overline{\phi}_i)(\phi_{uj} - \overline{\phi}_j)}{\sqrt{\sum_{u \in \mathcal{U}_{ij}} (\phi_{ui} - \overline{\phi}_i)^2 \; \sum_{i \in \mathcal{U}_{ij}} (\phi_{uj} - \overline{\phi}_j)^2}}$$

Rating prediction – example

$sim_{pc}(i,j)$	Titanic	Pulp Fiction	Iron Man	Forrest Gump	The Mummy
Titanic	1.0	-0.956	-0.815	NaN	-0.581
Pulp Fiction	–	1.0	0.948	NaN	0.621
Iron Man	–	–	1.0	NaN	0.243
Forrest Gump	–	–	–	1.0	NaN
The Mummy	–	–	–	–	1.0

NaN values are usually converted to zero (rare in case of enough data)

$sim_{pc}(u,v)$	Joe	Ann	Mary	Steve
Joe	1.0	-0.716	-0.762	-0.005
Ann	–	1.0	0.972	0.565
Mary	–	–	1.0	0.6
Steve	–	–	–	1.0

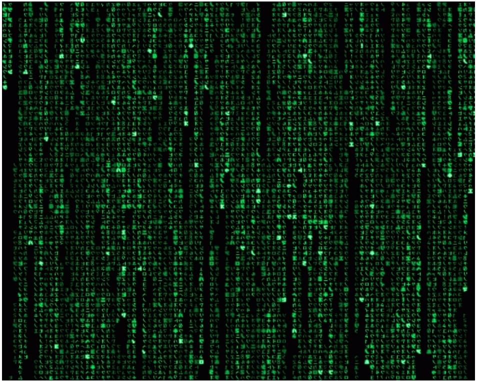
user-based

- $\mathcal{U}_{Titanic} = \{Joe, Ann, Mary\}$, $\mathcal{N}^{Steve,2}_{Titanic} = \{Mary, Ann\}$
- $\overline{\phi}_{Steve} = \frac{11}{3} = 3.67$, $\overline{\phi}_{Mary} = \frac{12}{4} = 3$, $\overline{\phi}_{Ann} = \frac{13}{4} = 3.25$
- $\hat{\phi}_{ST} = \overline{\phi}_S + \frac{s_{pc}(S,M)\cdot(\phi_{MT}-\overline{\phi}_M)+s_{pc}(S,A)\cdot(\phi_{AT}-\overline{\phi}_A)}{|s_{pc}(S,M)|+|s_{pc}(S,A)|} = 3.67 + \frac{0.6\cdot(4-3)+0.565\cdot(5-3.25)}{0.6+0.565} = 1.36$

item-based

- $\mathcal{I}_{\underline{S}teve} = \{\underline{P}ulp\ Fiction, \underline{I}ron\ Man, The\ \underline{M}ummy\}$, $\mathcal{N}^{Titanic,2}_{Steve} = \{\underline{I}ron\ Man, The\ \underline{M}ummy\}$
- $\overline{\phi}_T = \frac{10}{3} = 3.34$, $\overline{\phi}_I = \frac{11}{3} = 3.67$, $\overline{\phi}_M = \frac{9}{3} = 3$
- $\hat{\phi}_{ST} = \overline{\phi}_T + \frac{s_{pc}(T,I)\cdot(\phi_{SI}-\overline{\phi}_I)+s_{pc}(T,M)\cdot(\phi_{SM}-\overline{\phi}_M)}{|s_{pc}(T,I)|+|s_{pc}(T,M)|} = 3.34 + \frac{-.815\cdot(4-3.67)-.581\cdot(4-3)}{0.815+0.581} = 2.73$

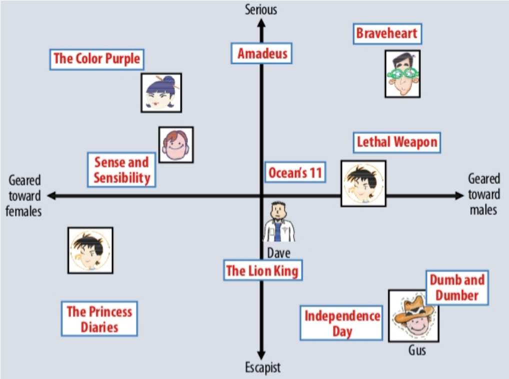
Matrix factorization



A latent space representation

Map users and items to a common latent space

- where dimensions or **factors** represent
  - items' **implicit properties**
  - users' **interest** in items' hidden properties



[1]

[1] The picture is taken from *Y. Koren et al. (2009). Matrix Factorization Techniques for Recommender Systems. Computer 42 (8).*

Known factorization models (1/2)

ϕ represented as a user-item matrix $\Phi^{n \times m}$

- n users, m items

[2] The picture is taken from wikipedia.

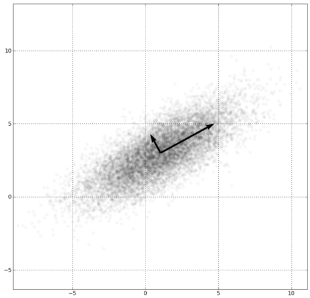
Known factorization models (1/2)

ϕ represented as a user-item matrix $\Phi^{n \times m}$

- n users, m items

Principal Component Analysis (PCA)

- transform data to a new coordinate system
  - variances by any projection of the data lies on coordinates in decreasing order



[2]

[2] The picture is taken from wikipedia.

Known factorization models (2/2)

Singular Value Decomposition (SVD)

$$\Phi = W^{n \times k} \Sigma^{k \times k} H^{n \times k^T}$$

- $W^T W = I$, $H^T H = I$
- column vectors of W are orthonormal eigenvectors of $\Phi\Phi^T$
- column vectors of H are orthonormal eigenvectors of $\Phi^T\Phi$
- Σ contains eigenvallues of W in descending order

[1] T.Raiko et al. (2007). Principal Component Analysis for Sparse High-Dimensional Data. Neural Information Processing, LNCS. 4984.

[2] A.K. Menon and Ch. Elkan (2011). Fast Algorithms for Approximating the Singular Value Decomposition. ACM Trans. Knowl. Discov. Data 5 (2).

Known factorization models (2/2)

Singular Value Decomposition (SVD)

$$\Phi = W^{n \times k} \Sigma^{k \times k} H^{n \times k^T}$$

- $W^T W = I$, $H^T H = I$
- column vectors of W are orthonormal eigenvectors of $\Phi\Phi^T$
- column vectors of H are orthonormal eigenvectors of $\Phi^T \Phi$
- Σ contains eigenvallues of W in descending order

PCA, SVD computed algebraically

- Φ is a **big** and **sparse** matrix
  - approximations of PCA[1], SVD[2]

[1] T.Raiko et al. (2007). Principal Component Analysis for Sparse High-Dimensional Data. Neural Information Processing, LNCS. 4984.

[2] A.K. Menon and Ch. Elkan (2011). Fast Algorithms for Approximating the Singular Value Decomposition. ACM Trans. Knowl. Discov. Data 5 (2).

MF – rating prediction (1/2)

recommendation task

- to find $\hat{\phi} : \mathcal{U} \times \mathcal{I} \to \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal

MF – rating prediction (1/2)

recommendation task

- to find $\hat{\phi} : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal
  - acc is the **expected** accuracy on $\mathcal{T}$
  - training $\hat{\phi}$ on $\mathcal{D}$ such that the **empirical** loss $err(\hat{\phi}, \phi, \mathcal{D})$ is minimal

MF – rating prediction (1/2)

recommendation task

- to find $\hat{\phi} : \mathcal{U} \times \mathcal{I} \to \mathbb{R}$ such that $acc(\hat{\phi}, \phi, \mathcal{T})$ is maximal
  - acc is the **expected** accuracy on $\mathcal{T}$
  - training $\hat{\phi}$ on $\mathcal{D}$ such that the **empirical** loss $err(\hat{\phi}, \phi, \mathcal{D})$ is minimal

a simple, **approximative** MF **model**

- only $W^{n \times k}$ and $H^{m \times k}$
- k – the number of factors

$$\Phi^{n \times m} \approx \hat{\Phi}^{n \times m} = WH^T$$

- predicted rating $\hat{\phi}_{ui}$ of the user u for the item i

$$\hat{\phi}_{ui} = w_u h_i^T$$

MF – rating prediction (2/2)

the **loss** function $err(\hat{\phi}, \phi, \mathcal{D})$

- squared loss

$$err(\hat{\phi}, \phi, \mathcal{D}) = \sum_{(u,i)\in\mathcal{D}} e_{ui}^2 = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - \hat{\phi}_{ui})^2 = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - w_u h_i^T)^2$$

MF – rating prediction (2/2)

the **loss** function $err(\hat{\phi}, \phi, \mathcal{D})$

- squared loss

$$err(\hat{\phi}, \phi, \mathcal{D}) = \sum_{(u,i)\in\mathcal{D}} e_{ui}^2 = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - \hat{\phi}_{ui})^2 = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - w_u h_i^T)^2$$

the **objective function**

- **regularization** term $\lambda \geq 0$ to prevent overfitting
  - penalizing the magnitudes of parameters

$$f(\hat{\phi}, \phi, \mathcal{D}) = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - w_u h_i^T)^2 + \lambda(\|W\|^2 + \|H\|^2)$$

MF – rating prediction (2/2)

the **loss** function $err(\hat{\phi}, \phi, \mathcal{D})$

- squared loss

$$err(\hat{\phi}, \phi, \mathcal{D}) = \sum_{(u,i)\in\mathcal{D}} e_{ui}^2 = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - \hat{\phi}_{ui})^2 = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - w_u h_i^T)^2$$

the **objective function**

- **regularization** term $\lambda \geq 0$ to prevent overfitting
  - penalizing the magnitudes of parameters

$$f(\hat{\phi}, \phi, \mathcal{D}) = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - w_u h_i^T)^2 + \lambda(\|W\|^2 + \|H\|^2)$$

The task is to find parameters W and H such that, given λ, the objective function $f(\hat{\phi}, \phi, \mathcal{D})$ is minimal.

Gradient descent

How to find a minimum of an "objective" function $f(\Theta)$?

- in case of MF, $\Theta = W \cup H$, and
- $f(\Theta)$ refers to the error of approximation of Φ by WH^T

Gradient descent

How to find a minimum of an "objective" function $f(\Theta)$?

- in case of MF, $\Theta = W \cup H$, and
- $f(\Theta)$ refers to the error of approximation of Φ by WH^T

<u>Gradient descent</u>

input: $f, \alpha, \Sigma^2,$ *stopping criteria*
initialize $\Theta \sim \mathcal{N}(0, \Sigma^2)$
repeat
    $\Theta \leftarrow \Theta - \alpha \frac{\partial f}{\partial \Theta}(\Theta)$
until approximate minimum is reached
return Θ

Gradient descent

How to find a minimum of an "objective" function $f(\Theta)$?

- in case of MF, $\Theta = W \cup H$, and
- $f(\Theta)$ refers to the error of approximation of Φ by WH^T

Gradient descent

input: f, α, Σ^2, *stopping criteria*
initialize $\Theta \sim \mathcal{N}(0, \Sigma^2)$
repeat
$\Theta \leftarrow \Theta - \alpha \frac{\partial f}{\partial \Theta}(\Theta)$
until approximate minimum is reached
return Θ

stopping criteria

- $|\Theta^{old} - \Theta| < \epsilon$
- maximum number of iterations reached
- a combination of both

Stochastic gradient descent

if f can be written as

$$f(\Theta) = \sum_{i=1}^{n} f_i(\Theta)$$

Stochastic gradient descent

if f can be written as

$$f(\Theta) = \sum_{i=1}^{n} f_i(\Theta)$$

Stochastic gradient descent (SGD)

input: $f_i, \alpha, \Sigma^2,$ *stopping criteria*
initialize $\Theta \sim \mathcal{N}(0, \Sigma^2)$
repeat
  for all i in random order **do**
    $\Theta \leftarrow \Theta - \alpha \frac{\partial f_i}{\partial \Theta}(\Theta)$
  end for
until approximate minimum is reached
return Θ

MF with SGD

updating parameters **iteratively** for each data point ϕ_{ui} in the opposite direction of the **gradient** of the objective function at the given point until a **convergence** criterion is fulfilled.

- updating the vectors w_u and h_i for the data point $(u, i) \in D$

MF with SGD

updating parameters **iteratively** for each data point ϕ_{ui} in the opposite direction of the **gradient** of the objective function at the given point until a **convergence** criterion is fulfilled.

- updating the vectors w_u and h_i for the data point $(u, i) \in D$

$$\frac{\partial f}{\partial w_u}(u,i) = -2(e_{ui}h_i - \lambda w_u)$$

$$\frac{\partial f}{\partial h_i}(u,i) = -2(e_{ui}w_u - 2\lambda h_i)$$

$$w_u(u,i) \leftarrow w_u - \alpha \frac{\partial f}{\partial w_u}(u,i) = w_u + \alpha(e_{ui}h_i - \lambda w_u)$$

$$h_i(u,i) \leftarrow h_i - \alpha \frac{\partial f}{\partial h_i}(u,i) = h_i + \alpha(e_{ui}w_u - \lambda h_i)$$

where $\alpha > 0$ is a **learning rate**.

MF with SGD – Algorithm

Hyper-parameters: $k, iters$ *(the max number of iteration)*$, \alpha, \lambda, \Sigma^2$

$W \leftarrow \mathcal{N}(0, \Sigma^2)$

$H \leftarrow \mathcal{N}(0, \Sigma^2)$

for $iter \leftarrow 1, \ldots, iters \cdot |\mathcal{D}|$ **do**

  draw randomly (u, i) from $\mathcal{D}$

  $\hat{\phi}_{ui} \leftarrow 0$

  for $j \leftarrow 1, \ldots, k$ **do**

    $\hat{\phi}_{ui} \leftarrow \hat{\phi}_{ui} + W[u][j] \cdot H[i][j]$

  end for

  $e_{ui} = \phi_{ui} - \hat{\phi}_{ui}$

  for $j \leftarrow 1, \ldots, k$ **do**

    $W[u][j] \leftarrow W[u][j] + \alpha * (e_{ui} * H[i][j] - \lambda * W[u][j])$

    $H[i][j] \leftarrow H[i][j] + \alpha * (e_{ui} * W[u][j] - \lambda * H[i][j])$

  end for

end for

return $\{W, H\}$

MF with SGD – Example[2]

Let's have the following hyper-parameters:

$K = 2,\ \alpha = 0.1,\ \lambda = 0.15,\ iter = 150,\ \sigma^2 = 0.01$

$\Phi =$

1	4	5		3
5	1		5	2
4	1	2	5	
	3	4		4

Results are:

$W =$

1.1995242	1.1637173
1.8714619	-0.02266505
2.3267753	0.27602595
2.033842	0.539499

$H^T =$

1.6261001	1.1259034	2.131041	2.2285593	1.6074764
-0.40649664	0.7055319	1.0405376	0.39400166	0.49699315

Results[1] are:

$\hat{\Phi} =$

1.477499	2.171588	3.767126	3.131717	2.506566
3.052397	2.091094	3.964578	4.161733	2.997066
3.671365	2.814469	5.245668	5.294111	3.877419
3.087926	2.670543	4.895569	4.745101	3.537480

[1] Note, that these hyper-parameters are just picked up in an ad-hoc manner. One should search for the "best" hyper-parameter combinations using e.g. grid-search (a brute-force approach).

[2] Thanks to my colleague Thai-Nghe Nguyen for computing an example.

Biased MF

baseline estimate

- user-item bias

$$b_{ui} = \mu + b'_u + b''_i$$

  - μ – average rating across the whole $\mathcal{D}$
  - b', b'' – vectors of user and item biases, respectively

Biased MF

baseline estimate

- user-item bias

$$b_{ui} = \mu + b_u^{'} + b_i^{''}$$

  - μ – average rating across the whole $\mathcal{D}$
  - $b^{'}, b^{''}$ – vectors of user and item biases, respectively

prediction

$$\hat{\phi}_{ui} = \mu + b_u^{'} + b_i^{''} + w_u h_i$$

Biased MF

baseline estimate

- user-item bias

$$b_{ui} = \mu + b^{'}_{u} + b^{''}_{i}$$

  - μ – average rating across the whole $\mathcal{D}$
  - $b^{'}, b^{''}$ – vectors of user and item biases, respectively

prediction

$$\hat{\phi}_{ui} = \mu + b^{'}_{u} + b^{''}_{i} + w_u h_i$$

objective function to minimize

$$f(\phi, \hat{\phi}, \mathcal{D}) = \sum_{(u,i)\in\mathcal{D}} (\phi_{ui} - \mu - b^{'}_{u} - b^{''}_{i} - w_u h_i)^2 + \lambda(\|W\|^2 + \|H\|^2 + b^{'2} + b^{''2})$$

Biased MF with SGD

similar to unbiased MF

- initialize average and biases

$$\mu = \frac{\sum_{(u,i)\in\mathcal{D}}}{|\mathcal{D}|}$$

$$b' \leftarrow (\overline{\phi}_{u_1}, \ldots, \overline{\phi}_{u_n})$$

$$b'' \leftarrow (\overline{\phi}_{i_1}, \ldots, \overline{\phi}_{i_m})$$

Biased MF with SGD

similar to unbiased MF

- initialize average and biases

$$\mu = \frac{\sum_{(u,i)\in\mathcal{D}}}{|\mathcal{D}|}$$

$$b' \leftarrow (\overline{\phi}_{u_1}, \ldots, \overline{\phi}_{u_n})$$

$$b'' \leftarrow (\overline{\phi}_{i_1}, \ldots, \overline{\phi}_{i_m})$$

- update average and biases

$$\mu \leftarrow \mu - \frac{\partial f}{\partial \mu}(u,i) = \mu + \alpha e_{ui}$$

$$b' \leftarrow b' - \frac{\partial f}{\partial b'}(u,i) = b' + \alpha(e_{ui} - \lambda b')$$

$$b'' \leftarrow b'' - \frac{\partial f}{\partial b''}(u,i) = b'' + \alpha(e_{ui} - \lambda b'')$$

MF – item recommendation

to predict a personalized **ranking score**[1] $\hat{\phi}_{ui}$

- how the item i is preferred to other items for the user u
- to find W and H such that $\hat{\Phi} = WH^T$

$$\hat{\phi}_{ui} = w_u h_i^T$$

[1] S. Rendle et al. (2009). BPR: Bayesian Personalized Ranking from Implicit Feedback. 25th Conference on Uncertainty in Artificial Intelligence.

MF – item recommendation

to predict a personalized **ranking score**[1] $\hat{\phi}_{ui}$

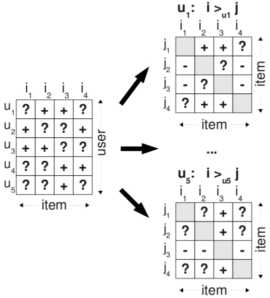
- how the item i is preferred to other items for the user u
- to find W and H such that $\hat{\Phi} = WH^T$

$$\hat{\phi}_{ui} = w_u h_i^T$$

problem: positive feedback only

- **pairwise ranking** data

$$\mathcal{D}_p = \{(u, i, j) \in \mathcal{D} | i \in \mathcal{I}_u \wedge j \in \mathcal{I} \setminus \mathcal{I}_u\}$$



[1] S. Rendle et al. (2009). BPR: Bayesian Personalized Ranking from Implicit Feedback. 25th Conference on Uncertainty in Artificial Intelligence.

MF – Bayesian Personalized Ranking (1/3)

Bayesian formulation of the problem

- $\succ$ – the unknown preference structure (ordering)
  - we use the derived pairwise ranking data $\mathcal{D}_p$
- Θ – parameters of an arbitrary prediction model
  - in case of MF, $\Theta = W \cup H$

$$p(\Theta|\succ) \propto p(\succ|\Theta)p(\Theta)$$

MF – Bayesian Personalized Ranking (1/3)

Bayesian formulation of the problem

- $\succ$ – the unknown preference structure (ordering)
  - we use the derived pairwise ranking data $\mathcal{D}_p$
- Θ – parameters of an arbitrary prediction model
  - in case of MF, $\Theta = W \cup H$

$$p(\Theta | \succ) \propto p(\succ | \Theta) p(\Theta)$$

prior probability

- assume independence of parameters
- assume, $\Theta \sim N(0, \frac{1}{\lambda} I)$

$$p(\Theta) = \prod_{\theta \in \Theta} \sqrt{\frac{\lambda}{2\pi}} \, e^{-\frac{1}{2}\lambda\theta^2}$$

MF – Bayesian Personalized Ranking (2/3)

likelihood

- assume users' feedbacks are independent
- assume, ordering of each pair is independent

$$p(\succ|\Theta) = \prod_{u \in \mathcal{U}} p(\succ_u|\Theta) = \prod_{(u,i,j) \in \mathcal{D}_p} p(i \succ_u j|\Theta)$$

MF – Bayesian Personalized Ranking (2/3)

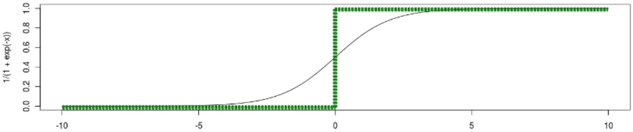
likelihood

- assume users' feedbacks are independent
- assume, ordering of each pair is independent

$$p(\succ|\Theta) = \prod_{u \in \mathcal{U}} p(\succ_u |\Theta) = \prod_{(u,i,j) \in \mathcal{D}_p} p(i \succ_u j|\Theta)$$

- using the ranking scores $\hat{\phi}$

$$p(i \succ_u j|\Theta) = p(\hat{\phi}_{ui} - \hat{\phi}_{uj} > 0) = \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) = \frac{1}{1 + e^{-(\hat{\phi}_{ui} - \hat{\phi}_{uj})}}$$



MF – Bayesian Personalized Ranking (3/3)

maximum **a posteriori estimation** of Θ

$$\arg\max_{\Theta} p(\Theta, \succ) =$$

MF – Bayesian Personalized Ranking (3/3)

maximum **a posteriori estimation** of Θ

$$\arg\max_{\Theta} p(\Theta, \succ) =$$

$$\arg\max_{\Theta} p(\succ |\Theta) p(\Theta) =$$

MF – Bayesian Personalized Ranking (3/3)

maximum **a posteriori estimation** of Θ

$$\arg\max_{\Theta} p(\Theta, \succ) =$$

$$\arg\max_{\Theta} p(\succ |\Theta) p(\Theta) =$$

$$\arg\max_{\Theta} ln\, p(\succ |\Theta) p(\Theta) =$$

MF – Bayesian Personalized Ranking (3/3)

maximum **a posteriori estimation** of Θ

$$\arg\max_{\Theta} p(\Theta, \succ) =$$

$$\arg\max_{\Theta} p(\succ |\Theta) p(\Theta) =$$

$$\arg\max_{\Theta} ln\, p(\succ |\Theta) p(\Theta) =$$

$$\arg\max_{\Theta} ln \prod_{(u,i,j)\in\mathcal{D}_p} \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) \sqrt{\frac{\lambda}{2\pi}}\, e^{-\frac{1}{2}\lambda\theta^2}$$

MF – Bayesian Personalized Ranking (3/3)

maximum **a posteriori estimation** of Θ

$$\arg\max_{\Theta} p(\Theta, \succ) =$$

$$\arg\max_{\Theta} p(\succ |\Theta) p(\Theta) =$$

$$\arg\max_{\Theta} ln\, p(\succ |\Theta) p(\Theta) =$$

$$\arg\max_{\Theta} ln \prod_{(u,i,j) \in \mathcal{D}_p} \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) \sqrt{\frac{\lambda}{2\pi}}\, e^{-\frac{1}{2}\lambda\theta^2}$$

$$\arg\max_{\Theta} \underbrace{\sum_{(u,i,j) \in \mathcal{D}_p} ln\, \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) \; - \lambda \|\Theta\|^2}_{BPR\text{–}OPT}$$

Finding parameters for BPR-OPT

Stochastic gradient ascent

$$\frac{\partial BPR-OPT}{\partial \Theta} \propto \sum_{(u,i,j)\in \mathcal{D}_p} \frac{e^{-(\hat{\phi}_{ui}-\hat{\phi}_{uj})}}{1+e^{-(\hat{\phi}_{ui}-\hat{\phi}_{uj})}} \frac{\partial}{\partial \Theta}(\hat{\phi}_{ui}-\hat{\phi}_{uj}) - \lambda\Theta$$

Finding parameters for BPR-OPT

Stochastic gradient ascent

$$\frac{\partial BPR-OPT}{\partial \Theta} \propto \sum_{(u,i,j)\in \mathcal{D}_p} \frac{e^{-(\hat{\phi}_{ui}-\hat{\phi}_{uj})}}{1+e^{-(\hat{\phi}_{ui}-\hat{\phi}_{uj})}} \frac{\partial}{\partial \Theta}(\hat{\phi}_{ui}-\hat{\phi}_{uj}) - \lambda\Theta$$

$$\frac{\partial}{\partial \theta}(\hat{\phi}_{ui}-\hat{\phi}_{uj}) = \begin{cases} (h_i - h_j) & if\ \theta = w_u \\ w_u & if\ \theta = h_i \\ -w_u & if\ \theta = h_j \\ 0 & else \end{cases}$$

Finding parameters for BPR-OPT

Stochastic gradient ascent

$$\frac{\partial BPR-OPT}{\partial \Theta} \propto \sum_{(u,i,j) \in \mathcal{D}_p} \frac{e^{-(\hat{\phi}_{ui} - \hat{\phi}_{uj})}}{1 + e^{-(\hat{\phi}_{ui} - \hat{\phi}_{uj})}} \frac{\partial}{\partial \Theta}(\hat{\phi}_{ui} - \hat{\phi}_{uj}) - \lambda \Theta$$

$$\frac{\partial}{\partial \theta}(\hat{\phi}_{ui} - \hat{\phi}_{uj}) = \begin{cases} (h_i - h_j) & if\ \theta = w_u \\ w_u & if\ \theta = h_i \\ -w_u & if\ \theta = h_j \\ 0 & else \end{cases}$$

<u>LearnBPR</u>

input: $f_i, \alpha, \Sigma^2, stopping\ criteria$
initialize $\Theta \sim \mathcal{N}(0, \Sigma^2)$
repeat
  draw $(u, i, j) \in \mathcal{D}_p$ randomly
  $\Theta \leftarrow \Theta + \alpha \frac{\partial BPR-OPT}{\partial \Theta}(\Theta)$
until approximate maximum is reached
return Θ

BPR-OPT vs AUC

Area under the ROC curve (AUC)

- probability that the ranking of a randomly drawn pair is correct

$$AUC = \sum_{u \in \mathcal{U}} AUC(u) = \frac{1}{|\mathcal{U}|} \frac{1}{|\mathcal{I}_u| \, |\mathcal{I} \setminus \mathcal{I}_u|} \sum_{(u,i,j) \in \mathcal{D}_p} \delta(\hat{\phi}_{ui} \succ \hat{\phi}_{uj})$$

  - $\delta(\hat{\phi}_{ui} \succ \hat{\phi}_{uj}) = 1$ if $\hat{\phi}_{ui} \succ \hat{\phi}_{uj}$, and 0, else

BPR-OPT vs AUC

Area under the ROC curve (AUC)

- probability that the ranking of a randomly drawn pair is correct

$$AUC = \sum_{u \in \mathcal{U}} AUC(u) = \frac{1}{|\mathcal{U}|} \frac{1}{|\mathcal{I}_u| \, |\mathcal{I} \setminus \mathcal{I}_u|} \sum_{(u,i,j) \in \mathcal{D}_p} \delta(\hat{\phi}_{ui} \succ \hat{\phi}_{uj})$$

  - $\delta(\hat{\phi}_{ui} \succ \hat{\phi}_{uj}) = 1$ if $\hat{\phi}_{ui} \succ \hat{\phi}_{uj}$, and 0, else

Smoothed AUC objective function with regularization of parameters

$$AUC - OPT = \sum_{(u,i,j) \in \mathcal{D}_p} \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) - \lambda \|\Theta\|^2$$

BPR-OPT vs AUC

Area under the ROC curve (AUC)

- probability that the ranking of a randomly drawn pair is correct

$$AUC = \sum_{u \in \mathcal{U}} AUC(u) = \frac{1}{|\mathcal{U}|} \frac{1}{|\mathcal{I}_u| \; |\mathcal{I} \setminus \mathcal{I}_u|} \sum_{(u,i,j) \in \mathcal{D}_p} \delta(\hat{\phi}_{ui} \succ \hat{\phi}_{uj})$$

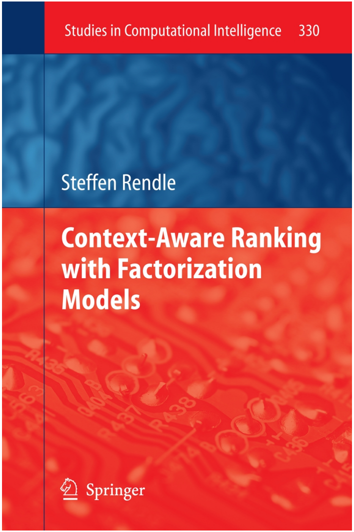
- $\delta(\hat{\phi}_{ui} \succ \hat{\phi}_{uj}) = 1$ if $\hat{\phi}_{ui} \succ \hat{\phi}_{uj}$, and 0, else

Smoothed AUC objective function with regularization of parameters

$$AUC - OPT = \sum_{(u,i,j) \in \mathcal{D}_p} \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) - \lambda \|\Theta\|^2$$

$$BPR - OPT = \sum_{(u,i,j) \in \mathcal{D}_p} ln\, \sigma(\hat{\phi}_{ui} - \hat{\phi}_{uj}) \; - \lambda \|\Theta\|^2$$

More info on ranking with factorization models



Issues worth to mention

The cold-start problem

arises when not enough collaborative information is available

- new user or new item

[1] Z. Gantner et al. (2010). Learning Attribute-to-Feature Mappings for Cold-Start Recommendations. 10th IEEE International Conference on Data Mining.

The cold-start problem

arises when not enough collaborative information is available

- new user or new item

possible solutions

- recommend popular items, “predict” global average, …

[1] Z. Gantner et al. (2010). Learning Attribute-to-Feature Mappings for Cold-Start Recommendations. 10th IEEE International Conference on Data Mining.

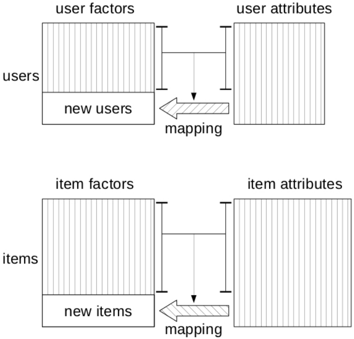
The cold-start problem

arises when not enough collaborative information is available

- new user or new item

possible solutions

- recommend popular items, “predict” global average, ...
- utilize item attributes[1]



[1] Z. Gantner et al. (2010). Learning Attribute-to-Feature Mappings for Cold-Start Recommendations. 10th IEEE International Conference on Data Mining.

Context-aware recommendation

Context is any additional information, besides $\chi^{user}, \chi^{item}, \phi$ and κ, that is relevant for the recommendation[1]

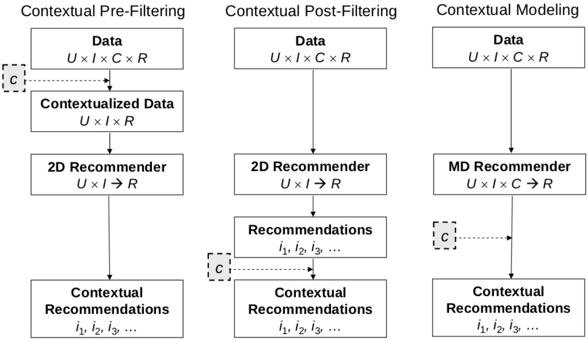
- time, location, companion (when, where and with whom the user wants to watch some movie)

[1] Picture from *G. Adomavicius and A. Tuzhilin: Context-Aware Recommender Systems. Tutorial on the 2nd ACM International Conference on Recommender Systems, 2008.* `http://ids.csom.umn.edu/faculty/gedas/talks/RecSys2008-tutorial.pdf`

Context-aware recommendation

Context is any additional information, besides $\chi^{user}, \chi^{item}, \phi$ and κ, that is relevant for the recommendation[1]

- time, location, companion (when, where and with whom the user wants to watch some movie)



[1] Picture from *G. Adomavicius and A. Tuzhilin: Context-Aware Recommender Systems. Tutorial on the 2nd ACM International Conference on Recommender Systems, 2008.* `http://ids.csom.umn.edu/faculty/gedas/talks/RecSys2008-tutorial.pdf`

Evaluating RS (1/3)

experiments

Evaluating RS (1/3)

experiments

- **offline**
  - no interaction with real users, need to simulate user behaviour
  - low cost, short time
  - answers only a few questions, e.g. the predictive power of techniques

Evaluating RS (1/3)

experiments

- **offline**
  - no interaction with real users, need to simulate user behaviour
  - low cost, short time
  - answers only a few questions, e.g. the predictive power of techniques
- **user studies**
  - observing test subjects' behaviour in the system
  - questionnaries
  - expensive, small scale,

Evaluating RS (1/3)

experiments

- **offline**
    - no interaction with real users, need to simulate user behaviour
    - low cost, short time
    - answers only a few questions, e.g. the predictive power of techniques
- **user studies**
    - observing test subjects' behaviour in the system
    - questionnaries
    - expensive, small scale,
- **online evaluation**
    - redirect a small part of the traffic to an alternative recommendation engine
    - risky – we can loose some customers
    - good to do after an offline testing of an recommendation engine showes good results

Evaluating RS (2/3)

properties of a recommender system

- user preference
  - Which one from different RS users prefer more?

Evaluating RS (2/3)

properties of a recommender system

- user preference
  - Which one from different RS users prefer more?
- prediction accuracy
  - How precise recommendations does a RS provide?

Evaluating RS (2/3)

properties of a recommender system

- user preference
  - Which one from different RS users prefer more?
- prediction accuracy
  - How precise recommendations does a RS provide?
- coverage
  - What proportion of all items can a RS ever recommend? To what proportion of users can a system recommend? How rich a user profile should be for making recommendation?
  - cold-start as a subproblem ("coldness" of an item)

Evaluating RS (2/3)

properties of a recommender system

- user preference
  - Which one from different RS users prefer more?
- prediction accuracy
  - How precise recommendations does a RS provide?
- coverage
  - What proportion of all items can a RS ever recommend? To what proportion of users can a system recommend? How rich a user profile should be for making recommendation?
  - cold-start as a subproblem ("coldness" of an item)
- confidence
  - How confident the system is with its recommendation? (e.g. depends on amount of data in CF…)

Evaluating RS (2/3)

properties of a recommender system

- user preference
  - Which one from different RS users prefer more?
- prediction accuracy
  - How precise recommendations does a RS provide?
- coverage
  - What proportion of all items can a RS ever recommend? To what proportion of users can a system recommend? How rich a user profile should be for making recommendation?
  - cold-start as a subproblem ("coldness" of an item)
- confidence
  - How confident the system is with its recommendation? (e.g. depends on amount of data in CF...)
- novelty
  - Does the system recommends items the user did not know about?

Evaluating RS (2/3)

properties of a recommender system

- user preference
  - Which one from different RS users prefer more?
- prediction accuracy
  - How precise recommendations does a RS provide?
- coverage
  - What proportion of all items can a RS ever recommend? To what proportion of users can a system recommend? How rich a user profile should be for making recommendation?
  - cold-start as a subproblem ("coldness" of an item)
- confidence
  - How confident the system is with its recommendation? (e.g. depends on amount of data in CF...)
- novelty
  - Does the system recommends items the user did not know about?
- trust
  - What is the users' trust in recommendation?

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)
- diversity
  - How "colorful" the recommendations are?

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)
- diversity
  - How "colorful" the recommendations are?
- utility
  - How useful a RS is for the provider/user? (e.g. generated revenue)

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)
- diversity
  - How "colorful" the recommendations are?
- utility
  - How useful a RS is for the provider/user? (e.g. generated revenue)
- robustness
  - How stable a RS is in presence of fake information?

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)
- diversity
  - How “colorful” the recommendations are?
- utility
  - How useful a RS is for the provider/user? (e.g. generated revenue)
- robustness
  - How stable a RS is in presence of fake information?
- privacy
  - How users' privacy is retained in a RS ?

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)
- diversity
  - How "colorful" the recommendations are?
- utility
  - How useful a RS is for the provider/user? (e.g. generated revenue)
- robustness
  - How stable a RS is in presence of fake information?
- privacy
  - How users' privacy is retained in a RS ?
- adaptivity
  - How does a RS adapt to changes in the item collection?

Evaluating RS (3/3)

- serendipity
  - How surprising the recommendations are? (e.g. a new movie with the user's favourite actor can be novel but not surprising)
- diversity
  - How "colorful" the recommendations are?
- utility
  - How useful a RS is for the provider/user? (e.g. generated revenue)
- robustness
  - How stable a RS is in presence of fake information?
- privacy
  - How users' privacy is retained in a RS ?
- adaptivity
  - How does a RS adapt to changes in the item collection?
- scalability
  - How scalable a RS is?

The MyMediaLite library

MyMediaLite

- is lightweight, multi-purpose library
- is mainly a library, meant to be used by other applications
- is free software (under the terms of the GNU General Public License)
- was developed by Zeno Gantner, Steffen Rendle, and Christoph Freudenthaler at University of Hildesheim

http://ismll.de/mymedialite

MyMediaLite features

major

- **scalable** implementations of many state-of-the-art recommendation methods
- evaluation framework for **reproducible** research
- **ready to be used**: command line tools, not programming necessary

MyMediaLite features

major

- **scalable** implementations of many state-of-the-art recommendation methods
- evaluation framework for **reproducible** research
- **ready to be used**: command line tools, not programming necessary

using for

- rating prediction
- item recommendation
- group recommendation

MyMediaLite features

major

- **scalable** implementations of many state-of-the-art recommendation methods
- evaluation framework for **reproducible** research
- **ready to be used**: command line tools, not programming necessary

using for

- rating prediction
- item recommendation
- group recommendation

next features

- usable from C#, Python, Ruby, F#
- Java ports available
- written in C#, runs on Mono
- regular releases (ca. 1 every 2 months)

Methods in MyMediaLite

State-of-the-art recommendation methods in MyMediaLite:

- kNN variants
- Online-Updating Regularized Kernel Matrix Factorization [Rendle and Schmidt-Thieme, RecSys 2009]
- SocialMF [Jamali and Ester, RecSys 2010] Freudenthaler at University of Hildesheim
- Asymmetric Factor Models (AFM) [Paterek, KDD Cup 2007]
- SVD++ [Koren, KDD 2008]
- Weighted Regularized Matrix Factorization (WR-MF) [Hu and Koren, ICDM 2008], [Pan et al., ICDM 2008]
- BPR-MF [Rendle et al., UAI 2009]

Data

e.g. MovieLens, Netflix

user ID	item ID	rating	timestamp
196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923

Data

e.g. MovieLens, Netflix

user ID	item ID	rating	timestamp
196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923

Remarks

- user and item IDs can be (almost) arbitrary strings
- separator: whitespace, tab, comma, ::
- alternative date and time format: yyyy-mm-dd
- rating and date and time fields are optional
- import script; Unix tools, Perl, Python . . .

Usage: Explicit Feedback I

Getting Help

- `rating prediction --help`

Data sets

- `rating prediction --training-file=u1.base --test-file=u1.test`

Recommender Options

- `rating prediction --training-file=u.data --test-ratio=0.2`

Fixing the Random Seed

- `rating prediction ... --random-seed=1`

Choosing a Recommender (algorithm)

- `rating prediction ... --recommender=UserAverage`
- `rating prediction ... --recommender=UserItemBaseline`

Usage: Explicit Feedback II

Iterative Recommenders

- `rating prediction`
  `... --recommender=BiasedMatrixFactorization --find-iter=1 --max-iter=30`

Recommender Options (Hyperparameters)

- `rating prediction`
  `... --recommender-options=''num factors=5''`
- `rating prediction ... --recommender-options=''num_factors=5 reg=0.05''`

SVD++

- `rating prediction ... --recommender=SVDPlusPlus --recommender-options=''num factors=5 reg=0.1 learn rate=0.01''`

Example: rating prediction

1	1	5
1	2	3
1	3	4
1	4	3
1	5	3
1	7	4

input data

- `user_id item_id rating`

where `user_id` and `item_id` are integers referring to users and items, respectively, and rating is a floating-point number expressing how much a user likes an item

- **separator**: either spaces, tabs, or commas
- only three columns, all additional columns will be ignored

Example: rating prediction

input data

- `user_id item_id rating`

1	1	5
1	2	3
1	3	4
1	4	3
1	5	3
1	7	4

where `user_id` and `item_id` are integers referring to users and items, respectively, and rating is a floating-point number expressing how much a user likes an item

- **separator**: either spaces, tabs, or commas
- only three columns, all additional columns will be ignored

usage of the rating prediction program

```
rating_prediction --training-file=TRAINING_FILE
--test-file=TEST_FILE --recommender=METHOD [OPTIONS]
```

Example: simple and advanced recommender

simple recommender

- run: `rating_prediction --training-file=u1.base --test-file=u1.test --recommender=UserAverage`
- output: `UserAverage training_time 00:00:00.000098 RMSE 1.063 MAE 0.85019 testing_time 00:00:00.032326`

Example: simple and advanced recommender

simple recommender

- run: `rating_prediction --training-file=u1.base --test-file=u1.test --recommender=UserAverage`
- output: `UserAverage training_time 00:00:00.000098 RMSE 1.063 MAE 0.85019 testing_time 00:00:00.032326`

advanced recommender

- run: `rating_prediction --training-file=u1.base --test-file=u1.test --recommender=BiasedMatrixFactorization`
- output: `BiasedMatrixFactorization num_factors=10 regularization=0.015 learn_rate=0.01 num_iter=30 init_mean=0 init_stdev=0.1 training_time 00:00:03.3575780 RMSE 0.96108 MAE 0.75124 testing_time 00:00:00.0159740`

Example: hyperparameter search

- run: `rating_prediction --training-file=u1.base --test-file=u1.test --recommender=BiasedMatrixFactorization --recomender-options="num_factors=20 num_iter=0" --max-iter=25 --num-iter=0`

Example: hyperparameter search

- run: `rating_prediction --training-file=u1.base --test-file=u1.test --recommender=BiasedMatrixFactorization --recomender-options="num_factors=20 num_iter=0" --max-iter=25 --num-iter=0`
- output:

```
...
RMSE 1.17083 MAE 0.96918 iteration 0
RMSE 1.01383 MAE 0.8143 iteration 1
RMSE 0.98742 MAE 0.78742 iteration 2
RMSE 0.97672 MAE 0.77668 iteration 3
RMSE 0.9709 MAE 0.77078 iteration 4
RMSE 0.96723 MAE 0.76702 iteration 5
RMSE 0.96466 MAE 0.76442 iteration 6
RMSE 0.96269 MAE 0.76241 iteration 7
RMSE 0.96104 MAE 0.76069 iteration 8
RMSE 0.95958 MAE 0.75917 iteration 9
RMSE 0.95825 MAE 0.75783 iteration 10
RMSE 0.95711 MAE 0.75667 iteration 11
RMSE 0.95626 MAE 0.75569 iteration 12
RMSE 0.95578 MAE 0.75501 iteration 13
RMSE 0.95573 MAE 0.75467 iteration 14
RMSE 0.95611 MAE 0.75467 iteration 15
RMSE 0.9569 MAE 0.75499 iteration 16
RMSE 0.95802 MAE 0.75551 iteration 17
RMSE 0.95942 MAE 0.75623 iteration 18
RMSE 0.96102 MAE 0.7571 iteration 19
RMSE 0.96277 MAE 0.75806 iteration 20
RMSE 0.96463 MAE 0.75909 iteration 21
RMSE 0.96656 MAE 0.76017 iteration 22
RMSE 0.96852 MAE 0.7613 iteration 23
RMSE 0.9705 MAE 0.76246 iteration 24
RMSE 0.97247 MAE 0.76364 iteration 25
```

Why use MyMediaLite?

- simple
- free
- scalable
- well-documented
- well-tested

Why use MyMediaLite?

- simple
- free
- scalable
- well-documented
- well-tested

possibility of using extra features

- Item Recommendation Tool (very similar usage like `rating_prediction`)
- `--cross-validation=K`
- `--chronological-split=2012-01-01`
- `--online-evaluation`
- `--save-model=FILE --load-model=FILE`
- `--measure=RMSE --epsilon=0.001`
- ...

Summary


Seriously.....
LGF
MINIONS
Do I need to say more?
ICANHASCHEEZBURGER.COM

Types of RS (1/2)

Knowledge-based

- pros: no cold-start, deterministic
- cons: knowledge-engineering needed, static

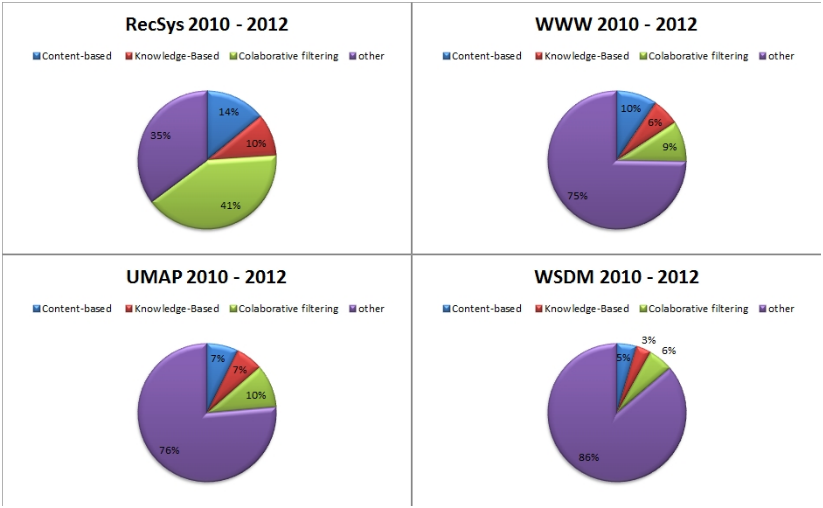
Content-based

- pros: no collaborative information needed
- cons: content is needed, cold-start for new users, no serendipity

Collaborative-filtering

- pros: no user nor item attributes needed, serendipity
- cons: cold-start for new users and items

Types of RS (2/2)




That's all Folks!

Many thanks go to

- **Štefan Pero** for his great help
- **Zeno Gantner** for providing materials and help regarding MyMediaLite
- **Artus Krohn-Grimberghe** for a picture from his PhD defense presentation
- all my colleagues and friends from ICS, UPJŠ and the ISMLL, UHI as well as other institutes for helping me to understand these things ;)

...also,

- all the people providing their materials (funny pictures, graphs, leaderboards, ...) on the web

...and, last but not least

- **YOU for your attention!**

Questions?

Tomas.Horvath@upjs.sk

http://www.ics.upjs.sk/~horvath