UFMG UNIVERSIDADE FEDERAL DE MINAS GERAIS

Recommender Systems
User-based Collaborative Filtering

Rodrygo Santos
rodrygo@dcc.ufmg.br

Acquiring feedback

UFMG COMPUTER SCIENCE

- We want to know
  - What users consider relevant
- We can observe
  - What users tell us (ratings)
  - What users do (actions)
- These are *noisy measurements*
  - Relevance is a user's prerogative

Feedback model

UFMG COMPUTER SCIENCE

Explicit
- Rate
- Review
- Vote

Implicit
- View
- Click
- Purchase

Feedback

CF formulation

UFMG COMPUTER SCIENCE

- *Predict how much user **u** will like item **i***

			i					
			1	3		2		
	1	2	5		4		1	
	4			3	5		4	
u		2	?		5	4		4
		3	4		5		3	

buys
clicks
views
rates
reviews
…

CF formulation

UFMG COMPUTER SCIENCE

- *Recommend items that user **u** will like*

			1	3		2		
	1	2	5		4		1	
	4			3	5		4	
u	?	2	?	?	5	4	?	4
		3	4		5		3	

buys
clicks
views
rates
reviews
…

CF formulation

UFMG COMPUTER SCIENCE

- *What if we know nothing about user **u**?*

			i					
			1	3		2		
	1	2	5		4		1	
	4			3	5		4	
u			?					
		3	4		5		3	

Non-personalized CF

- *What if we know nothing about user **u**?*

			i					
			1	3		2		
	1	2	5		4		1	
	4			3	5		4	
u			*3.3*					
		3	4		5		3	

We can use the average rating!

$$\hat{r}_{ui} = \frac{1}{|U_i|}\sum_{x \in U_i} r_{xi}$$

$$\hat{r}_{ui} = \frac{1}{3}(1+5+4)$$

$$\hat{r}_{ui} = 3.33$$

Non-personalized CF

- Problems?
  - Predicted utility of *i* will be the same for all users
    - We could compute a segmented average
      - e.g., by age, gender, income, location
  - Predicted utility of *i* ignores context
    - We could compute non-personalized associations
      - e.g., what sauce goes along with ice cream?
- Better, but still not fully personalized
  - Prediction will be the same for all users belonging to a given segment or in a given context

CF formulation

- *What if user **u** has historical preferences?*

			i					
			1	3		2		
	1	2	5		4		1	
	4			3	5		4	
u		2	*?*		5	4		4
		3	4		5		3	

Personalized CF

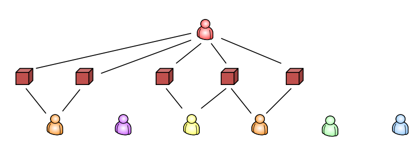
- Memory-based
  - Exploit similarities between users or items
    - Nearest neighbor approaches
    - Graph-based approaches
- Model-based
  - Build a model of user-item interactions
    - Bayesian models
    - Clustering models
    - Latent semantic models
    - ....

Memory-based CF

- How will I like item *i*?
  - User-based CF
    - *How similar users like item i?*
  - Item-based CF
    - *How do I like items similar to i?*

User-based CF

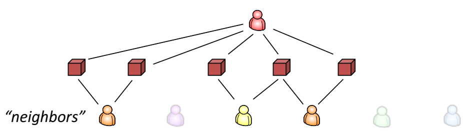
- Look for similar users



User-based CF

UFMG COMPUTER SCIENCE

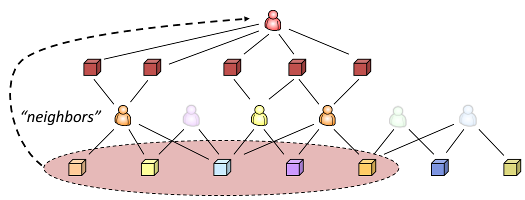
- Look for similar users



User-based CF

UFMG COMPUTER SCIENCE

- Recommend what they like



Breaking it down

UFMG COMPUTER SCIENCE

- Selecting neighborhoods
- Aggregating ratings
- Normalizing data
- Computing similarities
  – Algorithms
  – Tweaks
- Additional options

Selecting neighborhoods

UFMG COMPUTER SCIENCE

- A few options
  – All the neighbors
  – Random neighbors
  – All neighbors above a similarity threshold
  – Top-*k* neighbors ranked by similarity

How to find neighbors?

UFMG COMPUTER SCIENCE

- *Who are **u**'s nearest neighbors (who rated **i**)?*

			i						
x			1	3		2			$sim(\vec{u},\vec{x}) = 0.27$
y	1	2	5		4		1		$sim(\vec{u},\vec{y}) = 0.45$
	4			3	5		4		
u		2	?		5	4		4	
z		3	4		5		3		$sim(\vec{u},\vec{z}) = 0.52$

17

How to find neighbors?

UFMG COMPUTER SCIENCE

- *Who are **u**'s nearest neighbors (who rated **i**)?*

			i						
			1	3		2			
y	1	2	5		4		1		$sim(\vec{u},\vec{y}) = 0.45$
	4			3	5		4		**2-NN**
u		2	?		5	4		4	
z		3	4		5		3		$sim(\vec{u},\vec{z}) = 0.52$

18

How to find neighbors?

- *What if we wanted to rank **i** and **j**?*

			i	*j*				
			1	3		2		
y	1	2	5	✗	4		1	
	4			3	5		4	
u		2	?	?	5	4		4
z		3	4	✗	5		3	

Global nearest neighbors don't necessarily cover all items that could be recommended

Selecting neighborhoods

- A few options
  - All the neighbors
  - Random neighbors
  - All neighbors above a similarity threshold
  - Top-*k* neighbors ranked by similarity
    - Single set with *k* neighbors → may lack coverage
    - One set per recommendable item → may be expensive

How many neighbors?

- In theory, the more the better…
  - … if you have a good similarity metric
  - Computational cost is also higher
- In practice
  - More neighbors → more noise
  - Fewer neighbors → lower coverage
- Common practice
  - 25-100 is often used
  - 30-50 often good for movies

Aggregating ratings

- A few options
  - Min / max / average / median rating
  - Weighted average (by similarity)
  - Supervised aggregation
- Common practice
  - Weighted average: simple and effective

How to find neighbors?

- *Who are **u**'s nearest neighbors (who rated **i**)?*

			i					
			1	3		2		
y	1	2	5		4		1	
	4			3	5		4	
u		2	?		5	4		4
z		3	4		5		3	

$sim(\vec{u},\vec{y}) = 0.45$

2-NN

$sim(\vec{u},\vec{z}) = 0.52$

How to predict?

- *How will user **u** like item **i**?*

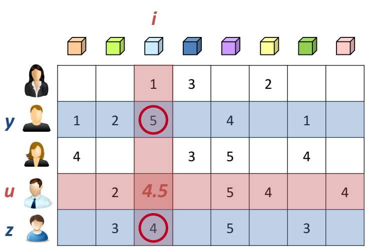
			i					
			1	3		2		
y	1	2	5		4		1	
	4			3	5		4	
u		2	?		5	4		4
z		3	4		5		3	

$sim(\vec{u},\vec{y}) = 0.45$ $r_{yi} = 5$

$sim(\vec{u},\vec{z}) = 0.52$ $r_{zi} = 4$

How to predict?

- *How will user **u** like item **i**?*



$$\hat{r}_{ui} = \frac{\sum_{v \in N} sim(\vec{u},\vec{v}) \times r_{vi}}{\sum_{v \in N} |sim(\vec{u},\vec{v})|}$$

$$\hat{r}_{ui} = \frac{0.45 \times 5 + 0.52 \times 4}{0.45 + 0.52}$$

$$\hat{r}_{ui} = 4.5$$

25

Normalizing ratings

- Users rate differently
  - Some rate high, others low
  - Some use more of the scale than others
- Averaging ignores these differences
  - Normalization compensates for them
- Again, a few options
  - Mean-center normalization
  - Z-score normalization

Mean-centering

$$\hat{r}_{ui} = \frac{\sum_{v \in N} sim(\vec{u},\vec{v}) \times r_{vi}}{\sum_{v \in N} |sim(\vec{u},\vec{v})|}$$

$$\hat{r}_{ui} = \frac{\sum_{v \in N} sim(\vec{u},\vec{v}) \times (r_{vi} - \boxed{\bar{r}_v})}{\sum_{v \in N} |sim(\vec{u},\vec{v})|}$$

(subtract neighbor's mean)

$$\hat{r}_{ui} = \boxed{\bar{r}_u} + \frac{\sum_{v \in N} sim(\vec{u},\vec{v}) \times (r_{vi} - \bar{r}_v)}{\sum_{v \in N} |sim(\vec{u},\vec{v})|}$$

(add target user's mean)

Z-score normalization

- Two steps
  - Mean-center each rating
  - Divide by standard deviation
- Normalizes for the spread across the scale
  - Slightly better than mean-centering

Computing similarities

- Pearson correlation

$$sim(\vec{u},\vec{v}) = \frac{\mathrm{cov}(u,v)}{\sigma_u \sigma_v} \approx \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \bar{r}_v)^2}}$$

  - Usually only over ratings in common (I_{uv})
  - Built-in user-mean normalization ($\bar{r}_u, \bar{r}_v$)
- Spearman = Pearson applied to *ranks*
  - Hasn't been found to work as well

Significance weighing

- What about little data?
  - Two users with one common rating → *sim* = 1
    - Are they really similar?
- Solution: weight similarity by confidence
  - Simple approach: multiply by ***min*(c, 50) / 50** (*c:* number of common ratings)
    - *c* < 50: neighbor's contribution scaled down by ***c* / 50**
    - *c* ≥ 50: neighbor's contribution unscaled

Cosine similarity

UFMG COMPUTER SCIENCE

- Cosine of the angle between user vectors

$$sim(\vec{u},\vec{v}) = \cos(\vec{u},\vec{v}) = \frac{\vec{u}\cdot\vec{v}}{|\vec{u}|\,|\vec{v}|} = \frac{\sum_{i\in I} r_{ui} r_{vi}}{\sqrt{\sum_{i\in I} r_{ui}^2}\sqrt{\sum_{i\in I} r_{vi}^2}}$$

  - In general: *sim* $\in$ [-1, 1]
  - With non-negative ratings: *sim* $\in$ [0, 1]
  - With user-mean norm (aka ***adjusted cosine***)
    - Equivalent to Pearson… ***almost!***

Self-weighting significance

UFMG COMPUTER SCIENCE

- Cosine has built-in significance weighting
  - Weights proportionally to ratio

$$\frac{|R_u \cap R_v|}{|R_u|\,|R_v|}$$

  - Similar effect can be obtained by using ***overall*** $\boldsymbol{\sigma_u}$ instead of just $\boldsymbol{\sigma_u}$ ***over common ratings*** in Pearson

Additional options

UFMG COMPUTER SCIENCE

- Clustering
  - Cluster users, pick user's cluster for prediction
  - Doesn't work particularly well
- Similarity pre-computation
  - Can be expensive
  - Often unstable
    - Users move as their ratings change

Suggested configuration

UFMG COMPUTER SCIENCE

- Top-*k* neighbors (*k* ≈ 30)
- Weighted averaging of scores
- User-mean normalization
- Cosine similarity

Optimal configuration is application-dependent

Implementation issues

UFMG COMPUTER SCIENCE

- Given ***m*** users and ***n*** items:
  - Computation can be a bottleneck
    - Correlation between two users is O(*n*)
    - All correlations for a user is O(*mn*)
    - All pairwise user correlations is O(m^2n)
    - Recommendations at least O(*mn*)
  - Lots of ways to make more practical
    - More persistent neighborhoods (*m*→*k*)
    - Cached or incremental correlations

Summary

UFMG COMPUTER SCIENCE

- User-based CF is simple and effective
  - The oldest CF approach
- Lots of configuration knobs
  - Similarity functions, neighborhood selection, aggregation functions, rating normalization
- Neighborhood selection is a bottleneck
  - Expensive to compute online
  - Unstable to pre-compute offline