Collaborative Filtering

Nicholas Ruozzi
University of Texas at Dallas

based on the slides of Alex Smola & Narges Razavian

Collaborative Filtering

- Combining information among collaborating entities to make recommendations and predictions
  - Can be viewed as a supervised learning problem (with some caveats)
  - Because of its many, many applications, it gets a special name

UTD

Examples

- Movie/TV recommendation (Netflix, Hulu, iTunes)
- Product recommendation (Amazon)
- Social recommendation (Facebook)
- News content recommendation (Yahoo)
- Priority inbox & spam filtering (Google)
- Online dating (OK Cupid)

Netflix Movie Recommendation

Training Data

user	movie	rating
1	14	3
1	200	4
1	315	1
2	15	5
2	136	1
3	235	3
4	79	3

Test Data

user	movie	rating
1	50	?
1	28	?
2	94	?
2	32	?
3	11	?
4	99	?
4	54	?

Recommender Systems

- Content-based recommendations
  - Recommendations based on a user profile (specific interests) or previously consumed content
- Collaborative filtering
  - Recommendations based on the content preferences of similar users
- Hybrid approaches

Collaborative Filtering

- Most widely-used recommendation approach
  - k-nearest neighbor methods
  - Matrix factorization based methods
- Predict the utility of items for a user based on the items previously rated by other like-minded users

Collaborative Filtering

- Make recommendations based on user/item similarities
  - User similarity
    - Works well if number of items is much smaller than the number of users
    - Works well if the items change frequently
  - Item similarity (recommend new items that were also liked by the same users)
    - Works well if the number of users is small

k-Nearest Neighbor

- Similar to the spectral clustering based approach from the homework
- Create a similarity matrix for pairs of users
- Use k-NN to find the k closest users to a target user
- Use the ratings of the k nearest neighbors to make predictions

UTD

User-User Similarity

- Let $r_{u,i}$ be the rating of the i^{th} item under user u, $\overline{r_u}$ be the average rating of user u, and $com(u,v)$ be the set of items rated by both user u and user v

- The similarity between user u and user v is then given by Pearson's correlation coefficient

$$sim(u,v) = \frac{\sum_{i \in com(u,v)} \left(r_{u,i} - \overline{r_u}\right)\left(r_{v,i} - \overline{r_v}\right)}{\sqrt{\sum_{i \in com(u,v)} \left(r_{u,i} - \overline{r_u}\right)^2} \sqrt{\sum_{i \in com(u,v)} \left(r_{v,i} - \overline{r_v}\right)^2}}$$

User-User Similarity

- Let $r_{u,i}$ be the rating of the i^{th} item under user u, $\overline{r_u}$ be the average rating of user u, and $com(u, v)$ be the set of items rated by both user u and user v

- The similarity between user u and user v is then given by Pearson's correlation coefficient

Empirical covariance of ratings

$$sim(u, v) = \frac{\sum_{i \in com(u,v)} (r_{u,i} - \overline{r_u})(r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i \in com(u,v)} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum_{i \in com(u,v)} (r_{v,i} - \overline{r_v})^2}}$$

User-User Similarity

- Let $r_{u,i}$ be the rating of the i^{th} item under user u, $\overline{r_u}$ be the average rating of user u, and $com(u,v)$ be the set of items rated by both user u and user v

- The similarity between user u and user v is then given by Pearson's correlation coefficient

$$sim(u,v) = \frac{\sum_{i \in com(u,v)} (r_{u,i} - \overline{r_u})(r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i \in com(u,v)} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum_{i \in com(u,v)} (r_{v,i} - \overline{r_v})^2}}$$

Empirical standard deviation of user u's ratings for common items

User-User Similarity

- Let $nn(u)$ denote the set of k-NN to u
- $p_{u,i}$, the predicted rating for the i^{th} item of user u, is given by

$$p_{u,i} = \bar{r_u} + \frac{\sum_{v \in nn(u)} sim(u,v) \cdot (r_{v,i} - \bar{r_v})}{\sum_{v \in nn(u)} |sim(u,v)|}$$

  – This is the average rating of user u plus the weighted average of the ratings of u's k nearest neighbors

User-User Similarity

- Issue: could be expensive to find the k-NN if the number of users is very large
  - Possible solutions?

Item-Item Similarity

- Use Pearson's correlation coefficient to compute the similarity between pairs of items
- Let $com(i, j)$ be the set of users common to items i and j
- The similarity between items i and j is given by

$$sim(i,j) = \frac{\sum_{u \in com(i,j)} \left(r_{u,i} - \bar{r_u}\right)\left(r_{u,j} - \bar{r_u}\right)}{\sqrt{\sum_{u \in com(i,j)} \left(r_{u,i} - \bar{r_u}\right)^2} \sqrt{\sum_{u \in com(i,j)} \left(r_{u,j} - \bar{r_u}\right)^2}}$$

Item-Item Similarity

- Let $nn(i)$ denote the set of k-NN to i
- $p_{u,i}$, the predicted rating for the i^{th} item of user u, is given by

$$p_{u,i} = \frac{\sum_{j \in nn(i)} sim(i,j) \cdot \left(r_{u,j}\right)}{\sum_{j \in nn(i)} |sim(i,j)|}$$

  – This is the weighted average of the ratings of i's k nearest neighbors

k-Nearest Neighbor

- Easy to train
- Easily adapts to new users/items
- Can be difficult to scale (finding closest pairs requires forming the similarity matrix)
  – Less of a problem for item-item assuming number of items is much smaller than the number of users
- Not sure how to choose k
  – Can lead to poor accuracy

k-Nearest Neighbor

- Tough to use without any ratings information to start with
  - "Cold Start"
    - New users should rate some initial items to have personalized recommendations
      - Could also have new users describe tastes, etc.
    - New Item/Movie may require content analysis or a non-CF based approach

Matrix Factorization

- There could be a number of latent factors that affect the recommendation
  - Style of movie: serious vs. funny vs. escapist
  - Demographic: is it preferred more by men or women
- Alternative approach: view CF as a matrix factorization problem

Matrix Factorization

- Express a matrix $M \in \mathbb{R}^{m \times n}$ approximately as a product of factors $A \in \mathbb{R}^{m \times p}$ and $B \in \mathbb{R}^{p \times n}$

$$M \sim A \cdot B$$

- Approximate the user $\times$ items matrix as a product of matrices in this way
  – Similar to SVD decompositions that we saw earlier (SVD can't be used for a matrix with missing entries)
  – Think of the entries of M as corresponding to an inner product of latent factors

Matrix Factorization

users

items

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

~

~ items

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

●

users

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

[from slides of Alex Smola]

UTD

Matrix Factorization

users

items

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

~

items

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

●

users

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

[from slides of Alex Smola]

Matrix Factorization

users

items

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

~

~ items

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

●

users

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

[from slides of Alex Smola]

UTD

Matrix Factorization

- We can express finding the "closest" matrix as an optimization problem

$$\min_{A,B} \sum_{(u,i)\ observed} \left(M_{u,i} - \langle A_{u,:}, B_{:,i} \rangle\right)^2 + \lambda(\|A\|_F^2 + \|B\|_F^2)$$

Matrix Factorization

- We can express finding the "closest" matrix as an optimization problem

$$\min_{A,B} \sum_{(u,i)\ observed} \left(M_{u,i} - \langle A_{u,:}, B_{:,i} \rangle\right)^2 + \lambda(\|A\|_F^2 + \|B\|_F^2)$$

Computes the error in the approximation of the observed matrix entries

Matrix Factorization

- We can express finding the "closest" matrix as an optimization problem

$$\min_{A,B} \sum_{(u,i)\ observed} \left(M_{u,i} - \langle A_{u,:}, B_{:,i} \rangle\right)^2 + \lambda(\|A\|_F^2 + \|B\|_F^2)$$

Regularization preferences matrices with small Frobenius norm

UTD

Matrix Factorization

- We can express finding the "closest" matrix as an optimization problem

$$\min_{A,B} \sum_{(u,i)\ observed} \left(M_{u,i} - \langle A_{u,:}, B_{:,i} \rangle\right)^2 + \lambda(\|A\|_F^2 + \|B\|_F^2)$$

- How to optimize this objective?

Matrix Factorization

- We can express finding the “closest” matrix as an optimization problem

$$\min_{A,B} \sum_{(u,i)\ observed} \left(M_{u,i} - \langle A_{u,:}, B_{:,i} \rangle\right)^2 + \lambda(\|A\|_F^2 + \|B\|_F^2)$$

- How to optimize this objective?
  - (Stochastic) gradient descent!

UTD

Extensions

- The basic matrix factorization approach doesn't take into account the observation that some people are tougher reviewers than others and that some movies are over-hyped
  - Can correct for this by introducing a bias term for each user and a global bias

$$\min_{A,B,\mu,b} \sum_{(u,i)\ observed} \left(M_{u,i} - \mu - b_i - b_u - \langle A_{u,:}, B_{:,i} \rangle\right)^2$$

$$+\lambda(\|A\|_F^2 + \|B\|_F^2) + \nu\left(\sum_i b_i^2 + \sum_u b_u^2\right)$$