Recommender Systems

Boston University CS506 - Lance Galletti

An Example: Movie Recommendations

Given

- Users: $\mathbf{U_1}, \ldots, \mathbf{U_n}$
- Movies: $\mathbf{M_1}, \ldots, \mathbf{M_m}$
- Ratings: $\mathbf{R_{ij}}$

Goal: Recommend movies to users

Challenges:

- Scale (millions of users, millions of movies)
- Cold Start (change in user base, change in content)
- Sparse Data (Not many users rank movies)

An Example: Movie Recommendations

	M_1	M_2	M_3	M_4
U_1	R_{11}	R_{12}	R_{13}	R_{14}
U_2	R_{21}	R_{22}	R_{23}	R_{24}
U_3	R_{31}	R_{32}	R_{33}	R_{34}

Use Rating prediction as proxy for recommendation!

An Example: Movie Recommendations

	M_1	M_2	M_3	M_4
U_1	5	?	0	0
U_2	?	4	0	0
U_3	0	?	4	?

An Example: Movie Recommendations

	M_1	M_2	M_3	M_4
U_1	5	5	0	0
U_2	5	4	0	0
U_3	0	0	4	5

Neighborhood Methods

- (user, user) similarity measure
  - i.e. recommend same movies to similar users (requires info about users)
- (item, item) similarity measure
  - i.e. recommend movies that are similar (requires info about movies)

Pros:

- Intuitive / easy to explain
- No training
- Handles new users/items

Challenges:

- Users rate differently (bias)
- Ratings change over time (bias)

Feature Extraction - Content-Based

Realistically:

- It’s difficult to characterize movies and users with the right features
- Characterization of users and movies may not be accurate
    - If you are using genres for example, movies with varying degree of “comedy” will get the tag “comedy”.

Goal:

- Discover the best features in an automated way

Content-Based: assume you have features for movies - want to learn features for users

Collaborative filtering: want to learn features for both users and movies

Feature Extraction - Content-Based

Suppose we have a set of features that characterizes each movie (ex: category, genre...), we could obtain the following **feature-to-movie** similarity matrix:

	M_1	M_2	M_3	M_4
F_1 (Romance)	.9	1	.1	0
F_2 (Action)	0	.01	1	.9

Feature Extraction - Content-Based

Given this **feature-to-movie** similarity matrix, how can we predict rating for User 2 or Movie 1 (i.e. R_{12})?

If we had a **user-to-feature** similarity matrix, we could multiply:

user-to-feature x feature-to-movie = user-to-movie = $\mathbf{R_{ij}}$

Feature Extraction - Content-Based

	F_1 (Romance)	F_2 (Action)
U_1	5	0
U_2	5	0
U_3	0	5

X

	M_1	M_2	M_3	M_4
F_1 (Romance)	.9	1	.1	0
F_2 (Action)	0	.01	1	.9

Feature Extraction - Content-Based

$$P^{(2)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$

$$Q^{(1)} = \begin{bmatrix} .9 \\ 0 \end{bmatrix}$$

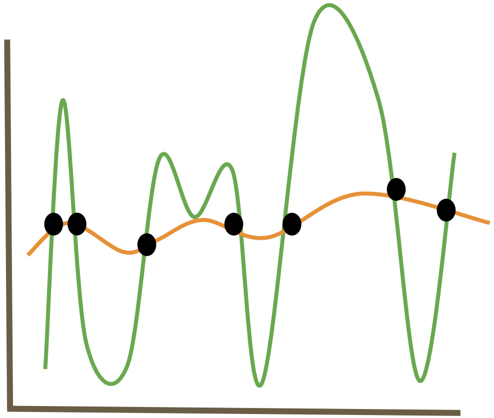
$$\begin{aligned} R_{21} &= P^{(2)T} \cdot Q^{(1)} \\ &= \begin{bmatrix} 5 & 0 \end{bmatrix} \cdot \begin{bmatrix} .9 \\ 0 \end{bmatrix} \\ &= 4.5 \end{aligned}$$

But, how to we find $p^{(1)}$, … , $p^{(n)}$?

Feature Extraction - Content-Based

$$P^{(j)} = \arg\min_{P} \frac{1}{\|M^{(j)}\|} \sum_{i \in M^{(j)}} \left(P^T Q^{(i)} - r_{ij}\right)^2 + \lambda \|p\|^2$$

Regularization Term: a penalty on the size of the parameter p



Feature Extraction - Collaborative Filtering

Challenge with content-based:

How to get the right features $f_1, \ldots, f_k$ **and** $p^{(1)}, \ldots, p^{(n)}$?

Can we learn these features?

$$\mathbf{R = PQ}$$

Feature Extraction - Collaborative Filtering

Can't use SVD because R is sparse... BUT, we can formulate an optimization problem to solve:

$$\min_{p,q} \sum_{i,j \in R} (r_{ij} - p_i^T q_j)^2 + \lambda(||p||_F^2 \, ||p||_F^2)$$

To solve, take derivatives wrt P & Q. Then, just like Expectation-Maximization Algorithm from GMM:

1. Start with random Q
2. Get P
3. Improve Q
4. Repeat 2 & 3