Recommender Systems

Collaborative Filtering

1. User-based Recommendation[1]

input:

$$\text{rating matrix: } R \in \mathbb{R}^{N \times M}$$

where r_{ui} is the rating of user u for item i.

example:

	The Matrix	Titanic	Die Hard	Forrest Gump	Wall-E
John	5	1	?	2	2
Lucy	1	5	2	5	5
Eric	2	?	3	5	4
Diane	4	3	5	3	?

hypothesis:

$$\hat{r}_{ui} = \frac{1}{|N_i(u)|} \sum_{v \in N_i(u)} r_{vi}$$

where $N_i(u)$ is the set of k users most similar to u that have rated i.

weights:

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i(u)} w_{uv} r_{vi}}{\sum_{v \in N_i(u)} |w_{uv}|}$$

cosine similarity:

$$w_{uv} = \cos(\mathbf{x}_u, \mathbf{x}_v) = \frac{\sum_{i \in I_{uv}} r_{ui} r_{vi}}{\sqrt{\sum_{i \in I_u} r_{ui}^2 \sum_{i \in I_v} r_{vi}^2}}$$

Pearson correlation:

$$w_{uv} = \mathrm{PC}(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)(r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)^2} \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \mu_v)^2}}$$

issues with Pearson correlation[2].

normalization:

$$\hat{r}_{ui} = h^{-1}\left(\frac{\sum_{v \in N_i(u)} w_{uv} h(r_{vi})}{\sum_{v \in N_i(u)} |w_{uv}|}\right)$$

mean-centering:

$$h(r_{ui}) = r_{ui} - \mu_u$$

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i(u)} w_{uv} (r_{vi} - \mu_v)}{\sum_{v \in N_i(u)} |w_{uv}|}$$

2. Item-based Recommendation[1]

input:

$$\text{rating matrix: } R \in \mathbb{R}^{N \times M}$$

where r_{ui} is the rating of user u for item i.

hypothesis:

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u(i)} w_{ij} r_{uj}}{\sum_{j \in N_u(i)} |w_{ij}|}$$

normalization:

$$\hat{r}_{ui} = h^{-1}\left(\frac{\sum_{j \in N_u(i)} w_{ij} h(r_{uj})}{\sum_{j \in N_u(i)} |w_{ij}|}\right)$$

3. Matrix Factorization(SVD)[4,5]

input:

$$\text{rating matrix: } R \in \mathbb{R}^{N \times M}$$

where r_{ui} is the rating of user u for item i.

hypothesis:

$$\hat{R} = PQ$$
$$\hat{r}_{ui} = p_u^T q_i$$

cost function:

$$\min_{q,\ p} \sum_{(u,i) \in K} (r_{ui} - p_u^T q_i)^2$$

normalization:

$$\min_{q,\ p} \sum_{(u,i) \in K} (r_{ui} - \mu - p_u^T q_i)^2$$

add bias:

$$\min_{q,\ p} \sum_{(u,i) \in K} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2$$

regularization:

$$\min_{q,\ p} \sum_{(u,i) \in K} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(||p_u||^2 + ||q_i||^2 + b_u^2 + b_i^2)$$

stochastic gradient descent[4]:

$$
\begin{aligned}
b_u &\leftarrow b_u + \eta \cdot [(r_{ui} - \mu - b_u - b_i - p_u^T q_i) - \lambda \cdot b_u] \\
b_i &\leftarrow b_i + \eta \cdot [(r_{ui} - \mu - b_u - b_i - p_u^T q_i) - \lambda \cdot b_i] \\
p_u &\leftarrow p_u + \eta \cdot [(r_{ui} - \mu - b_u - b_i - p_u^T q_i) \cdot q_i - \lambda \cdot p_u] \\
q_i &\leftarrow q_i + \eta \cdot [(r_{ui} - \mu - b_u - b_i - p_u^T q_i) \cdot p_u - \lambda \cdot q_i]
\end{aligned}
$$

cost function 2:

$$
C = \sum_{u,i} c_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda(\sum_u ||x_u||^2 + \sum_i ||y_i||^2)
$$

where

$$
p_{ui} = \begin{cases} 1, & r_{ui} > 0 \\ 0, & r_{ui} = 0 \end{cases} \quad \text{and} \quad c_{ui} = 1 + \alpha r_{ui}
$$

alternating least squares[5,6]:

$$
\begin{aligned}
\frac{\partial C}{\partial x_u} &= -2 \sum_i c_{ui}(p_{ui} - x_u^T y_i) y_i + 2\lambda x_u \\
&= -2 \sum_i c_{ui}(p_{ui} - y_i^T x_u) y_i + 2\lambda x_u \\
&= -2 Y^T C^u p(u) + 2 Y^T C^u Y x_u + 2\lambda x_u
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial C}{\partial x_u} = 0 &\Rightarrow (Y^T C^u Y + \lambda I) x_u = Y^T C^u p(u) \\
&\Rightarrow x_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u p(u)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial C}{\partial y_i} &= -2 \sum_u c_{ui}(p_{ui} - x_u^T y_i) x_u + 2\lambda y_i \\
&= -2 X^T C^i p(i) + 2 X^T C^i X y_i + 2\lambda y_i
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial C}{\partial y_i} = 0 &\Rightarrow (X^T C^i X + \lambda I) y_i = X^T C^i p(i) \\
&\Rightarrow y_i = (X^T C^i X + \lambda I)^{-1} X^T C^i p(i)
\end{aligned}
$$

Reference

1. **Recommender Systems Handbook:**

http://www.cs.bme.hu/nagyadat/Recommender_systems_handbook.pdf

2. **Similarity Functions for User-User Collaborative Filtering**: http://grouplens.org/blog/similarity-functions-for-user-user-collaborative-filtering/

3. **Matrix Factorization: A Simple Tutorial and Implementation in Python**: http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/

4. **Matrix Factorization and Neighbor Based Algorithms for the Netflix Prize Problem**: http://primeranks.net/yeti/Temp/reco/nmf_nn_netflix.pdf

5. **Collaborative Filtering for Implicit Feedback Datasets**: http://labs.yahoo.com/files/HuKorenVolinsky-ICDM08.pdf

6. **Analytic solution for matrix factorization using alternating least squares**: http://math.stackexchange.com/questions/1072451/analytic-solution-for-matrix-factorization-using-alternating-least-squares

7. **Recommender Systems: Collaborative Filtering and other approaches**: https://www.youtube.com/watch?v=bLhq63ygoU8

8. **Netflix Prize**: http://www.netflixprize.com/