MATRIX FACTORIZATION

Overview—

Matrix factorization is a class of collaborative filtering algorithms used in recommender systems. Matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. The idea behind matrix factorization is to represent users and items in a lower dimensional latent space.

Funk SVD—

The original algorithm proposed by Simon Funk in his blog post factorized the user-item rating matrix as the product of two lower dimensional matrices, the first one has a row for each user, while the second has a column for each item. The row or column associated to a specific user or item is referred to as **latent factors**. Note that, despite its name, in Funk SVD, no singular value decomposition is applied. The predicted ratings can be computed as $\tilde{R} = HW$, where $\tilde{R} \in R^{users \times items}$ is the user-item rating matrix, $H \in R^{users \times latent\ factors}$ contains the user's latent factors and $W \in R^{latent\ factors \times items}$ the item's latent factors. Specifically, the predicted rating user u will give to item i is computed as $\tilde{r}_{ui} = \sum_{f=0}^{n\ factors} H_{u,f} W_{f,i}$.

It is possible to tune the expressive power of the model by changing the number of latent factors. It has been demonstrated that a matrix factorization with one latent factor is equivalent to a most popular or top popular recommender (e.g. recommends the items with the most interactions without any personalization). Increasing the number of latent factor will improve personalization, therefore recommendation quality, until the number of factors becomes too high, at which point the model starts to overfit and the recommendation quality will decrease. A common strategy to avoid overfitting is to add regularization terms to the objective function. Funk SVD was developed as a rating prediction problem, therefore it uses explicit numerical ratings as user-item interactions.

All things considered, Funk SVD minimizes the following objective function:

$$arg\ min_{H,W} ||R - \tilde{R}||_F + \alpha ||H|| + \beta ||W||$$

Where $||.||_F$ is defined to be the frobenius norm where as the other norms might be either frobenius or another norm depending on the specific recommending problem.

SVD plus plus—

While Funk SVD is able to provide very good recommendation quality, its ability is to use only explicit numerical ratings as user-items interactions constitutes a limitation. Modern day recommender systems should exploit all available interactions both explicit and implicit. To this end SVD++ was designed to take into account implicit interactions as well. Compared to Funk SVD, SVD++ takes also into account user and item bias. The predicted rating user u will give to item i is computed as

$$\tilde{r}_{ui} = \mu + b_i + b_u + \sum_{f=0}^{n\ factors} H_{u,f} W_{f,i}$$

Asymmetric SVD—

Asymmetric SVD aims at combining the advantages of SVD++ while being a model based algorithm, therefore being able to consider new users with a few ratings without needing to retrain the whole model. As opposed to the model-based SVD here the user latent factor matrix H is replaced by Q, which learns the user's preferences as function of their ratings.

The predicted rating user u will given to item i is computed as:

$$\tilde{r}_{ui} = \mu + b_i + b_u + \sum_{f=0}^{n\ factors} \sum_{j=0}^{n\ items} r_{uj} Q_{j,f} W_{f,i}$$