Matrix Factorization Techniques For Recommender Systems

Reporter: Lei Guo

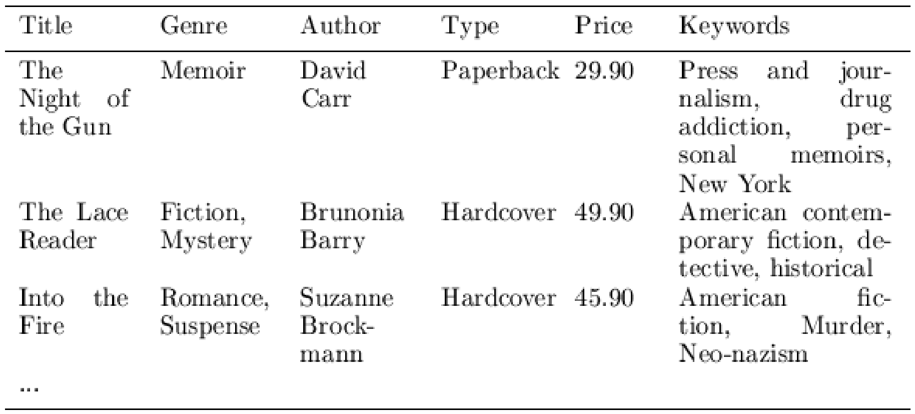
http://ir.sdu.edu.cn/index.htm

Agenda

- Recommender System Strategies
- Matrix Factorization Methods
- Basic Factorization Model
- Learning Algorithms
- Adding Biases
- Additional Input Sources
- Temporal Dynamics
- Inputs With Varing Confidence Level

Recommender System Strategies

- **Contented-based recommendation**
  - Creates a profile for each user or product to characterize its nature
    - Eg. Movie profile include attributes regarding its **genre, the paticipating actors, its box office popularity**…
    - Eg. User profile might include **demographic** information or **answers** provided on a suitable questionnaire
  - Programs use these profiles associate users with mathing products



Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, Murder, Neo-nazism
...					

- **Simple approach**
  - Compute the similarity of an unseen item with the user profile based on the keyword overlap (e.g. using the Dice coefficient)
  - sim(b_i, b_j) = $\frac{2 * |keywords(b_i) \cap keywords(b_j)|}{|keywords(b_i)| + keywords(b_j)|}$
  - Or combine multiple metrics in a weighted approach
  - Require gathering external information that might not be available
- **Collaborative-based recommendation**
  - Relies only on past user behavior (eg. Previous transactions or puduct ratings)
  - Also known as collaborative filtering (CF)
- **Basic assumption and idea**
  - Users give ratings to catalog items (implicitly or explicitly)
  - Customers who had similar tastes in the past, will have similar tastes in the future

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

- The *goal is to estimate Alice's rating for this item*

- **Neighborhood methods**
- **User-oriented approach**
  - Centered on computing the relationships between users

Pearson correlation

$$sim(a,b) = \frac{\sum_{p\in P}(r_{a,p}-\bar{r}_a)(r_{b,p}-\bar{r}_b)}{\sqrt{\sum_{p\in P}(r_{a,p}-\bar{r}_a)^2}\sqrt{\sum_{p\in P}(r_{b,p}-\bar{r}_b)^2}}$$

- **Item-oriented approach**
  - Evaluates a user's preference for an item based on ratings of "neighboring" item by the same user
  - Look for items that are similar to Item5
  - Take Alice's ratings for these items to predict the rating for Item5

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

- **Latent factor models**
  - An alternative approach that tries to explain the ratings by charactering both items and users with the more holistic goal to uncover the latent features
  - For movies, these latent factors might measure obvious dimensions such as comedy versus drama, amount of action, or orientation to children
  - For users, each factor measures how much the users likes movies that score high on the corresponding movie factor
  - PLAS
  - Neural networks
  - Latent Dirichlet Allocation
  - **Matrix factorization (eg. SVD)**
  - …

Matrix Factorization Methods

- **Characteristic**
  - Characterizes both items and users by vectors of factors infered from item rating patterns
  - High correspondence between item and user factors leads to a recommendation
- **Rely on matrix types of input data**
  - One dimention representing user
  - The other representing items
- **Two data types**
  - High-quality explicit feedfack
    - Includs explicit input by users regarding their interest in products
    - We refer to explicit user feedback as ratings
    - Usually sparse matrix, since any single user is likely to have rated only a small percentage of possible items

- **Implicit feedback**
  - Which indirectly reflects opion by observing user behavior
    - Purchase history, browsing history, search patterns, mouse movements
  - Usually denotes the presence or absence of an event
  - Typically represented by a densely filled matrix

Basic Matrix Factorization Model

- **Formalization**
  - Map both users and items to a joint latent factor space of dimensionality f
  - User-item interactions are modeled as inner products in that space
  - Each item i is associated with a vector q_i ,and each user u is associated with a a vector p_u,
    - q_i measures the extent to which the item possesses those factors
    - p_u measure the extent of interest the user has in items
    - the resulting dot product $q_i^T p_u$ caputures the interaction between user u and item i – the user's overall interest in the item's characteristics
  - The appoximates user u's rating of item i, which is denoted by r_{ui}, leading to the estimate

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

  - The major chanllenge is computing the mapping of each item and user to factor vecotrs q_i,p_u

- We can caputure the latent relationships between users and items
- We can produce a low-dimensional representation of the original rating matrix
- Factor rating matrix R using SVD obtain Q, S, P
- Reduce the matrix S to dimension k
- Compute two resultant matrices: $Q_k S_k$ (q^T)and $S_k P_k$(p)

- **Predicting task**
  - These resultant matrices can now be used to compute the recommendation score for any user and item
  - We can simply calculte the dot product of the ith row of q and the uth column of p

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

- **Top-N Recommendation Task**
  - We consider customer preference data as binary by treating each non-zero entry of the customer-product matrix as "1"
  - We are only interested in whether a customer consumed a particular product but not how much the user liked
- **First Step (neighborhood formation in the reduced space)**
  - Factor rating matrix R using SVD obtain Q, S, P
  - Reduce the matrix S to dimension k
  - Compute resultant matrices: $P_k S_k$
  - Performed vector similarity (cosine similarity) to form the neighborhood
- **Second Step(Top-N Recommendation)**
  - we scan through the purchase record of each of the k neighbors
  - Compute the frequency count on the products they purchased
  - Return the most frequently purchased N items for the target customer

- **Difficulties**
  - High portion of missing values caused by sparseness in the user-item rating matrix
  - Conventional SVD is undefined when knowledge about the matrix is incomplete
  - Carelessly addressing only the relatively few kown entries is highly prone to overfitting
- **Solutions**
- **Fill missing values**
  - Earlier systems relied on imputation to fill in missing rating and make the rating matrix dense
  - Such as using the average ratings for user and item
  - However, (1)imputation can be very expensive as it significantly increases the amount of data and (2) inaccurate imputation might distort the data.

- **Modeling directly the observed ratings only**
  - Avoiding overfitting through a regularized model
  - The learn the factor vectors(p_u and q_i), the system minimizes the regularized squared error on the set of known ratings:

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

  - The goal is to generalize those previous ratings in a way that predicts future unkown ratings
  - The constant λ controls the extent of regularization

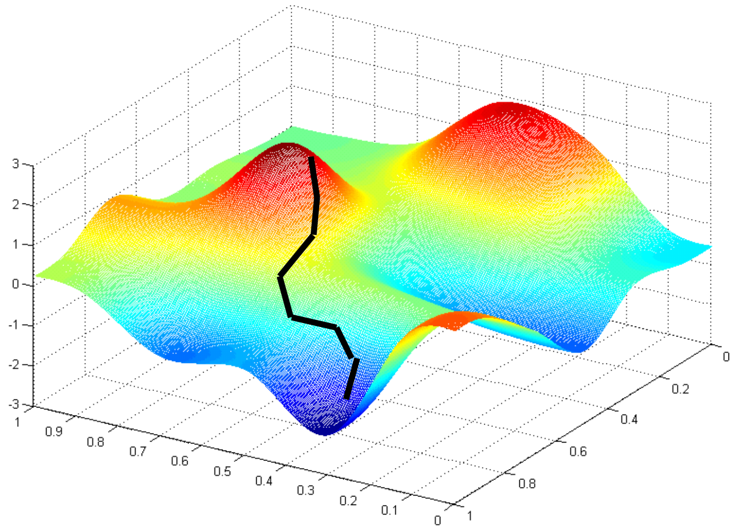
Learning Algorithms

- **Stochastic gradient desent**
  - Also known as incremental learning
  - For each given training case, the system predicts r$_{ui}$ and computes the associated prediction error

$$e_{ui} \overset{def}{=} r_{ui} - q_i^T p_u.$$

  - Learning rule

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$
$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$



- **Alternating least squares**
  - Because both q_i and p_u are unknowns, the object function is not convex
  - However, if fix one of the unknowns, the optimization problems quadratic and can be solved optimally
  - Als techniques rotate between fixing the q_i's and fixing the p_u's
  - When all p_u's are fixed, the system recomputes the q_i's by sovling a least-squares problem $\|R - PQ^T\|_F$
  - we can fix the matrix *P* as some matrix $\hat{P}$, such that minimization problem would be equivalent to $R = \hat{P}Q^T$

$$Q^T = \left(\hat{P}^T\hat{P}\right)^{-1}\hat{P}^T R$$

  - Analogously, we can fix *Q* as $\hat{Q}$, $P = R\hat{Q}\left(\hat{Q}^T\hat{Q}\right)^{-1}$

- **Learning rule**

$$
\begin{aligned}
Q^T &\leftarrow \left(P^T P\right)^{-1} P^T R \\
P &\leftarrow RQ\left(Q^T Q\right)^{-1}
\end{aligned}
$$

  - It can be shown that the only possible minimum is the global one
  - so that *P* and *Q* must converge to the true SVD subspace

Adding Biases

- We tries to capture the interactions between users and items that produce the different rating values

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

- However, typical collaborative filtering data exhibits large systematic tendencies for some users to give higher ratings than others
- And for some items to receive higher ratings than others
  - Some products are widely perceived as better(or worse) than others
- Its unwise to explain the full rating value in this form
- We should identify the user and item bias

- **A first-order approximation of the bias is as follows:**

$$b_{ui} = \mu + b_i + b_u$$

  - The bias accounts for the user and item effects, where μ denotes the overall average rating
  - b_u and b_i indicate the observed deviations of user u and item i.
- **For example**
  - Suppose we want to estimate user john’s rating of the movie Titanic
  - And the averagerating over all movies is 3.7 stars
  - Titanic is better than an average movie, so it tends to be rated 0.5 starts above the average movie
  - John is a critical user, who tends to rate 0.3 stars lower than the average
  - Thus, the estimate for Titanic’s rating by John would be (3.7+0.5-0.3)

- **The estimation becomes:**

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$$

- **The system learns by miniming the squared error function**

$$\min_{b_*, q_*, p_*} \sum_{(u,i)\in\mathcal{K}} (r_{ui} - \mu - b_i - b_u - q_i^T p_u)^2 + \lambda_4 (b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$

- **The stochastic gradient descent learning rule becomes:**

$$\begin{aligned} b_u &\leftarrow b_u + \gamma \cdot (e_{ui} - \lambda_4 \cdot b_u) \\ b_i &\leftarrow b_i + \gamma \cdot (e_{ui} - \lambda_4 \cdot b_i) \\ q_i &\leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda_4 \cdot q_i) \\ p_u &\leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda_4 \cdot p_u) \end{aligned}$$

Additional Input Sources

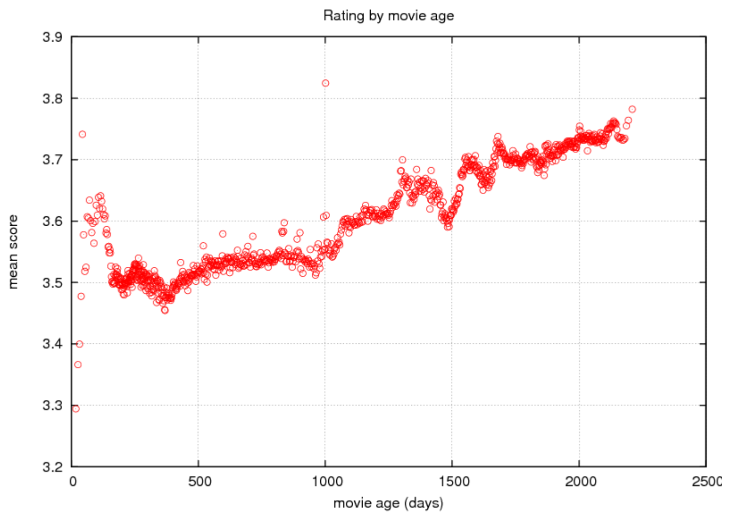
- **A system must deal with cold start problem, wherein many users supply very few ratings**
- **We need to incorporate additional sources to relieve this problem**
- **Use implicit feedback to gain insight into user preferences**
  - They can gather behavior information of user's
  - Eg. A retailer can use its customers' purchases or browsing history to learn their tendencies
  - For simplicity, consider a case with a boolean implicit feedback
  - The exact model is as follows

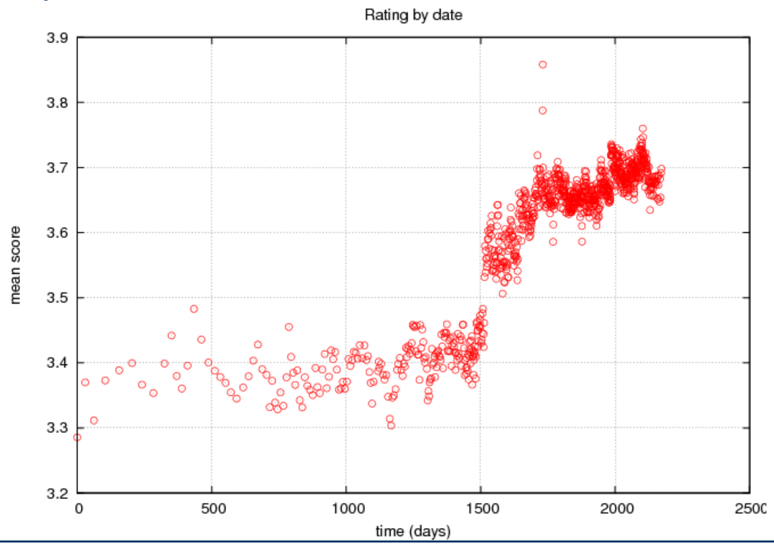
$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T \left(p_u + |\mathrm{R}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{R}(u)} y_j \right)$$

- Several types of implicit feedback can be simultaneously introduced into the model by using extra sets of item factors
- Eg. Demographics, such as gender, age group, zip code, income level
  - Use a distinct factor vector $y_j(2)$ corresponds to each attribute to describe a user
- The matrix factorization model should integrate all signal sources:
- $$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T \left(p_u + |\mathrm{N}^1(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{N}^1(u)} y_j^{(1)} + |\mathrm{N}^2(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{N}^2(u)} y_j^{(2)} \right)$$

Temproal Dynamics

- **Time-drifting nature**
  - So far, the presented models have been static
  - In reality, product perception and popularity constantly change as new selections emerge
  - Similarly, customer's inclinations evolve, leading them to redefine their taste
  - The system should account for the temporal effects





- **Multiple sources of temporal dynamics**
- **Item-side effects:**
  - Product perception and popularity are constantly changing
  - Seasonal patterns influence items' popularity
- **User-side effects:**
  - Customers ever redefine their taste
  - Transient, short-term bias; anchoring
  - Drifting rating scale
  - Change of rater within household
- **Multiple sources: Both items and users are changing over time**
- **Multiple targets: Each user/item forms a unique time series**
- **➔ Scarce data per target**

Addressing temporal dynamics

- **Factor model conveniently allows separately treating different aspects**
- **We observe changes in:**
  1. Rating scale of individual users
  2. Popularity of individual items
  3. User preferences

$$r_{ui}(t) = \mu + b_u(t) + b_i(t) + q_i^T p_u(t)$$

Parameterizing the model

$$r_{ui}(t) = \mu + b_u(t) + b_i(t) + q_i^T p_u(t)$$

- **Use functional forms: $b_u(t)=f(u,t)$, $b_i(t)=g(i,t)$, $p_u(t)=h(u,t)$**
- **Need to find adequate f(), g(), h()**
- **General guidelines:**
  - Items show slower temporal changes
  - Users exhibit frequent and sudden changes
  - Factors –$p_u(t)$– are expensive to model
  - Gain flexibility by heavily parameterizing the functions

- **Take item biases and user biases as an example**
- **It is more of a challenge on the users side**
  - we would like a finer resolution for users to detect very short lived temporal effects
  - we do not expect enough ratings per user to produce reliable estimates for isolated bins
- **One simple modeling choice**
  - uses a linear function to capture a possible gradual drift of user bias

$$\text{dev}_u(t) = \text{sign}(t - t_u) \cdot |t - t_u|^\beta .$$

  - Timedependent user-bias

$$b_u^{(1)}(t) = b_u + \alpha_u \cdot \text{dev}_u(t)$$

- **item biases $b_i(t)$**
- **split the item biases into time-based bins and how?**
  - The decision of how to split the timeline into bins should balance the desire to achieve finer resolution (hence, smaller bins) with the need for enough ratings per bin (hence, larger bins).
- **The movie bias is split into a stationary part and a time changing part**

$$b_i(t) = b_i + b_{i,\mathrm{Bin}(t)}$$

$$\min \sum_{(u,i,t)\in\mathcal{K}} \left(r_{ui}(t) - \mu - b_u - \alpha_u \mathrm{dev}_u(t) - b_{u,t} - b_i - b_{i,\mathrm{Bin}(t)}\right)^2 + \lambda\left(b_u^2 + \alpha_u^2 + b_{u,t}^2 + b_i^2 + b_{i,\mathrm{Bin}(t)}^2\right)$$

Inputs With Varying Confidence Levels

- **Are all observed ratings deserve the same weight or confidence?**
  - For example, massive advertising might influence votes for certain items, which do not aptly reflect longer-term characteristics
  - Adversarial users might try to tilt the ratings of certain items
- **The same in implicit feedback**
  - The system works with a cruder binary representation (like or not like)
  - A zero value may not means not liking
    - The user might be unaware of the existence of the item, or unable to consume it due to its price
  - Consuming an item can also be the result of factors different from preferring it
    - a user may watch a TV show just because she is staying on the channel of the previously watched show
    - a consumer may buy an item as gift for someone else, despite not liking the item for himself

- **The notion of confidence**
  - Confidence can stem from available number values that describe the frequency of actions
    - How much time the user watched a certain show
    - How frequently a user bought a certain item
  - Various factors that have nothing to do with user preferences might cause a one-time event
  - A recurring event is more likely to reflect user opinion
- **We introduce a set of variables c_{ui} , which measure our confidence in observing p_{ui}**
  - In general, as r_{ui} grows, we have a stronger
  - Indication that the user indeed likes the item

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases}$$

- A plausible choice for c_{ui} would be

$$c_{ui} = 1 + \alpha r_{ui}$$

  - This way, we have some minimal confidence in p_{ui} for every user-item pair, but as we observe more evidence for positive preference, our confidence in $p_{ui} = 1$ increases accordingly.
  - The rate of increase is controlled by the constant α
- Our goal is to find a vector pu for each user u and a vector of item q_i that will factor user preferences

$$\mathrm{Min}_{p*,q*,b*} \sum_{\text{known } r_{ui}} c_{ui}\left(r_{ui} - (\mu + b_u + b_i + p_u^T q_i)\right)^2 + \lambda\left(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2\right)$$

Conclusions

- Basic Factorization Model
- Adding Biases
- Additional Input Sources
- Temporal Dynamics
- Inputs With Varing Confidence Level

References

[1] Matrix factorization techniques for recommender systerms, Yehuda Koren,2009.

[2]Application of Dimensionality Reduction in Recommender System-A case study,B.M. Sarwar, KDD, 2000.

[3] A Guide to Singular Value Decomposition for collaborative filtering. Chih-Chao Ma

[4] Factorization meets the neighborhood: a multifacedted collaborative fitering model. Yehuda Koren, KDD'08.

[5]Collaborative filtering for implicit feedback datasets. Yehuda Koren, ICDM'08.

[6]Collaborative filtering with temporal dynamics. Yehuda Koren, KDD'09.

[7] Recommender systems –An introduction, Dietmar Jannach, Cambridge university press,2011.

[8] Recommender Systems Handbook, Francesco Ricci, Springer , 2011.

- Thanks