Class

Recommendation Engines

Topic

Introduction

Collaborative Filtering

There are many ways to create recommendation engines

Classified as an unsupervised or semi-supervised learning algorithm

Collaborative Filtering

- Memory based methods
  - User Based
  - Item Based
- Model based methods
  - SVD
  - NMF

Collaborative filtering is necessarily unsupervised

Predicting User Preferences

Recommender systems attempt to predict user preferences

Use historical user preference data

Popularized by online businesses

Instance of a familiar application of recommendation engines

Based on the browsing history, relevant products are recommended to us

Sell many products

Important to cross sell and up sell their products

Image credit - netflix.com and.amazon.com

Collaborative Filtering

User item rating matrix

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7
User 1	3	5	1	2	4	3	3
User 2	2	4	1	2	?	3	2
User 3	3	?	5	?	4	1	1
User 4	4	5	1	?	?	?	?

Ratings for a user item pair

Collaborative filtering works on the premise

Users who are similar, select similar kind of products

Gauging how similar a product is to a set of products, relevant recommendations are made

Items - movies or TV series

User item rating matrix - user movie rating matrix

Collaborative Filtering

User item rating matrix

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7
User 1	3	5	1	2	4	3	3
User 2	2	4	1	2	?	3	2
User 3	3	?	5	?	4	1	1
User 4	4	5	1	?	?	?	?

Usually a **user item rating matrix is partially populated** as not all users tend to rate all items

Predicted value of these ratings of an item is high enough

Recommendation engines usually try to predict these missing ratings

The item is recommended to the user

Memory Based/ Neighbourhood Methods

Popular way of implementing collaborative filtering

Intuitive to understand and relatively straightforward to implement

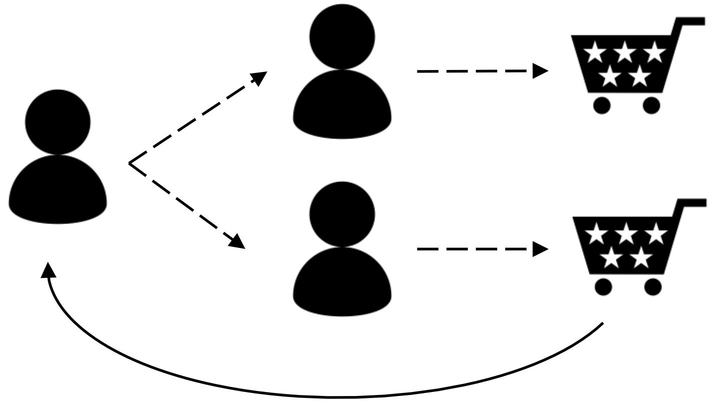
Collaborative Filtering

Memory based methods / Neighborhood methods

Model based methods

User Based

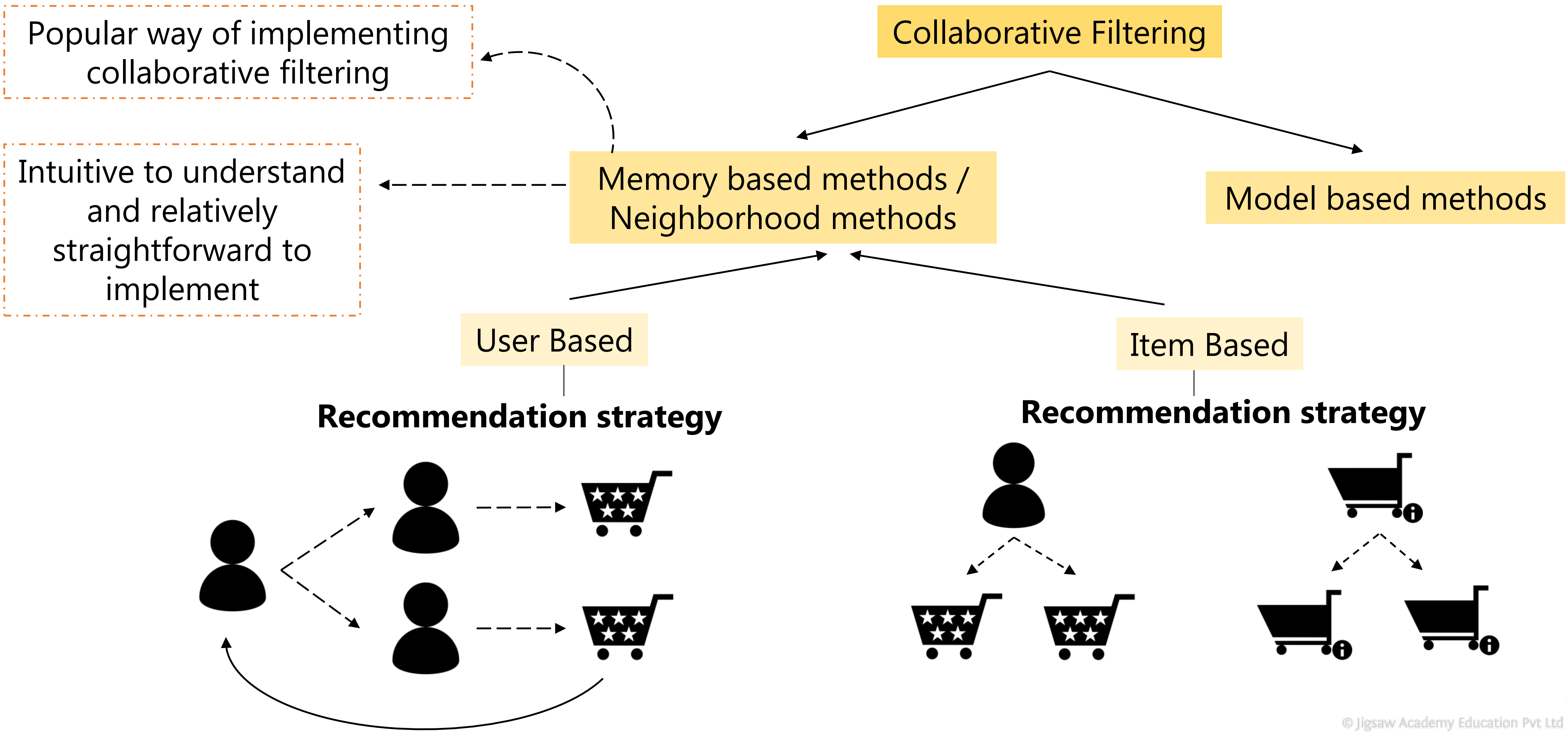
Recommendation strategy



Item Based

Recommendation strategy

Memory Based/ Neighbourhood Methods



User Based Collaborative Filtering

User item rating matrix

	Item 1	Item 2	Item 3	Item 4	Item 5
Alice	5	3	4	4	?
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

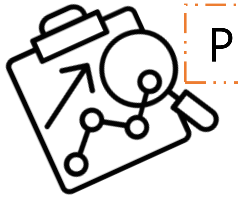
What is the rating for the Item 5 given by Alice?

Find users similar to Alice

Use any measures of similarity such as a **Pearson co-relation** or **Cosine similarity**

Based on the most similar users

Predict the rating for item 5 by Alice



User Based Collaborative Filtering

User item rating matrix

	Item 1	Item 2	Item 3	Item 4	Item 5
Alice	5	3	4	4	?
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

What is the rating for the Item 5 given by Alice?

Find users similar to Alice

Use any measures of similarity such as a **Pearson co-relation** or **Cosine similarity**

User 1 and User 2 are most similar users to Alice

The rating for item 5 by Alice will predicted based on how User 1 and User 2 have rated item 5

Recap

- Predicting user preferences
- Collaborative filtering
- Memory based methods
- User based collaborative filtering

Class

Recommendation Engines

Topic

Item Based Collaborative Filtering

Item Based Collaborative Filtering

User item rating matrix

	Item 1	Item 2	Item 3	Item 4	Item 5
Alice	5	3	4	4	?
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

What is the rating for the Item 5 given by Alice?

Find out the items similar to item 5 that are rated by Alice

Use either Co-relation or Cosine similarity to find the similar items

Based on the items that are more similar to item 5, rating of item 5 by Alice will be computed

Item Based Collaborative Filtering

User item rating matrix

	Item 1	Item 2	Item 3	Item 4	Item 5
Alice	5	3	4	4	?
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

What is the rating for the Item 5 given by Alice?

Find out the items similar to item 5 that are rated by Alice

Item 1 and Item 4 turn out to be the most similar items to Item 5

Based on Alice's rating of Item 1 and Item 4, the prediction about the rating for Item 5 would be made

Code Demo

Prediction

Prediction in a Recommender system, particularly the memory based Recommender systems, can be done in 2 ways:

1. Predict the rating based on just the ratings and similarities of items or users

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} \text{sim}(u,v) \cdot r_{vi}}{\sum_{v \in N_i^k(u)} \text{sim}(u,v)}$$

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} \text{sim}(i,j) \cdot r_{uj}}{\sum_{j \in N_u^k(j)} \text{sim}(i,j)}$$

2. Predict the rating based on the average effects of ratings of items and users as well

$$\hat{r}_{ui} = \mu_i + \frac{\sum_{j \in N_u^k(i)} \text{sim}(i,j) \cdot (r_{uj} - \mu_j)}{\sum_{j \in N_u^k(i)} \text{sim}(i,j)}$$

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} \text{sim}(u,v) \cdot (r_{vi} - \mu_v)}{\sum_{v \in N_i^k(u)} \text{sim}(u,v)}$$

Baseline Effects

Recommender system implementations

Baseline effects

User based – systematically, some users on an average rate items high, others rate items low

Item based – systematically, some items on an average, receive a higher rating while others receive a lower rating

Baseline estimates → Account for these systematic tendencies

Not all software libraries have the ability to take into account the baseline effects

Accuracy

The accuracy of a Recommender system can be measured by estimating the metrics

Root Mean Squared Error (RMSE)

Mean Absolute Error (MAE)

Computation of these quantities is quite straightforward since the actual ratings for many user and item pairs are already known

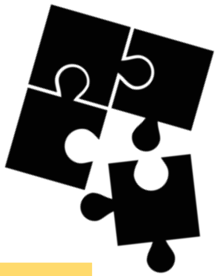
Hyperparameters

What would be the hyperparameters for neighborhood based recommendation systems?

1. Number of neighbors

2. Similarity metric

3. Prediction method



Grid Search using K-fold Cross Validation to figure out which values of these hyperparameters will be most suitable

Recap

- Item based collaborative filtering
- Code demo of item based collaborative filtering
- Prediction
- Baseline effects
- Accuracy
- Hyperparameters

Class

Recommendation Engines

Topic

Model Based

Matrix Factorization

Build recommendation engines using - user based and item based collaborative filtering

Another class of collaborative filtering recommendation engines that rely on the use

↓

Matrix factorization algorithms

Singular Value Decomposition

Non Negative Matrix Factorization

Matrix Factorization

User item rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	4	1	1
User 2	5	5	2	2
User 3	1	1	4	4
User 4	2	2	5	5

Assume –
Ratings are on a scale of 1 to 5

In context of recommendation engines, matrix factorization indicates finding 2 matrices :

User factor matrix

Item factor matrix

Matrix Factorization

User item rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	4	1	1
User 2	5	5	2	2
User 3	1	1	4	4
User 4	2	2	5	5

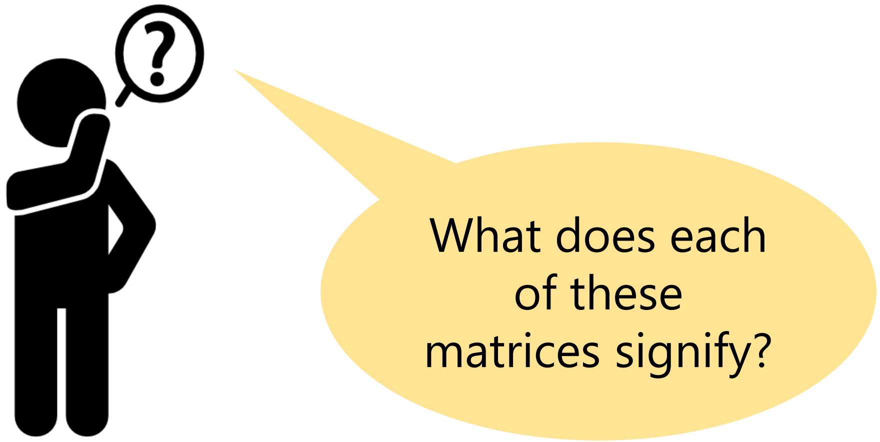
Factorizing

User factor matrix

Dim 1	Dim 2
0.5	-0.41
0.5	-0.57
-0.5	0.41
-0.5	0.57

Item factor matrix

	Star Trek	Avatar	Spiderman	Hulk
Dim 1	0.5	0.5	-0.5	-0.5
Dim 2	-0.2	-0.2	0.7	0.7



Matrix Factorization

User item rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	4	1	1
User 2	5	5	2	2
User 3	1	1	4	4
User 4	2	2	5	5

Genre: **Science Fiction (Sci-Fi)**

Genre: **Fantasy movies**

User factor matrix

Dim 1	Dim 2
0.5	-0.41
0.5	-0.57
-0.5	0.41
-0.5	0.57

2 columns

Representing factors

Movie genres

Item factor matrix

	Star Trek	Avatar	Spiderman	Hulk
Dim 1	0.5	0.5	-0.5	-0.5
Dim 2	-0.2	-0.2	0.7	0.7

Matrix Factorization

User item rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	4	1	1
User 2	5	5	2	2
User 3	1	1	4	4
User 4	2	2	5	5

High values corresponding to Sci-Fi genre

User factor matrix

Sci-Fi	Dim 2
0.5	-0.41
0.5	-0.57
-0.5	0.41
-0.5	0.57

High rating to Sci-Fi movies in the original matrix

Numbers represent how much each factor, each of the items have

Item factor matrix

	Star Trek	Avatar	Spiderman	Hulk
Dim 1	0.5	0.5	-0.5	-0.5
Dim 2	-0.2	-0.2	0.7	0.7

Matrix Factorization

User item rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	4	1	1
User 2	5	5	2	2
User 3	1	1	4	4
User 4	2	2	5	5

User factor matrix

Sci-Fi	Dim 2
0.5	-0.41
0.5	-0.57
-0.5	0.41
-0.5	0.57

Item factor matrix

	Star Trek	Avatar	Spiderman	Hulk
Sci-Fi	0.5	0.5	-0.5	-0.5
Dim 2	-0.2	-0.2	0.7	0.7

Matrix Factorization

User item rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	4	1	1
User 2	5	5	2	2
User 3	1	1	4	4
User 4	2	2	5	5

User factor matrix

Sci-Fi	Fantasy
0.5	-0.41
0.5	-0.57
-0.5	0.41
-0.5	0.57

Item factor matrix

	Star Trek	Avatar	Spiderman	Hulk
Sci-Fi	0.5	0.5	-0.5	-0.5
Fantasy	-0.2	-0.2	0.7	0.7

Matrix Factorization

Technical details of matrix factorization

User Rating Matrix
$m \times n$

=

User Factor Matrix
$m \times k$

$\underbrace{}_{p}$

Item Factor Matrix
$k \times n$

$\underbrace{}_{q^T}$

k is determined by the user

Number of factors to be considered is always less than the number of rows, or columns or original rating matrix, whichever is minimum

Matrix Factorization

Technical details of matrix factorization

User Rating Matrix
$m \times n$

$$\text{User Rating Matrix } (m \times n) = \underbrace{\text{User Factor Matrix } (m \times k)}_{p} \; \underbrace{\text{Item Factor Matrix } (k \times n)}_{q^T}$$

$$\boxed{\boldsymbol{k < \min(m, n)}}$$

Matrix Factorization

Technical details of matrix factorization

User Rating Matrix
$1000 \times 10{,}000$

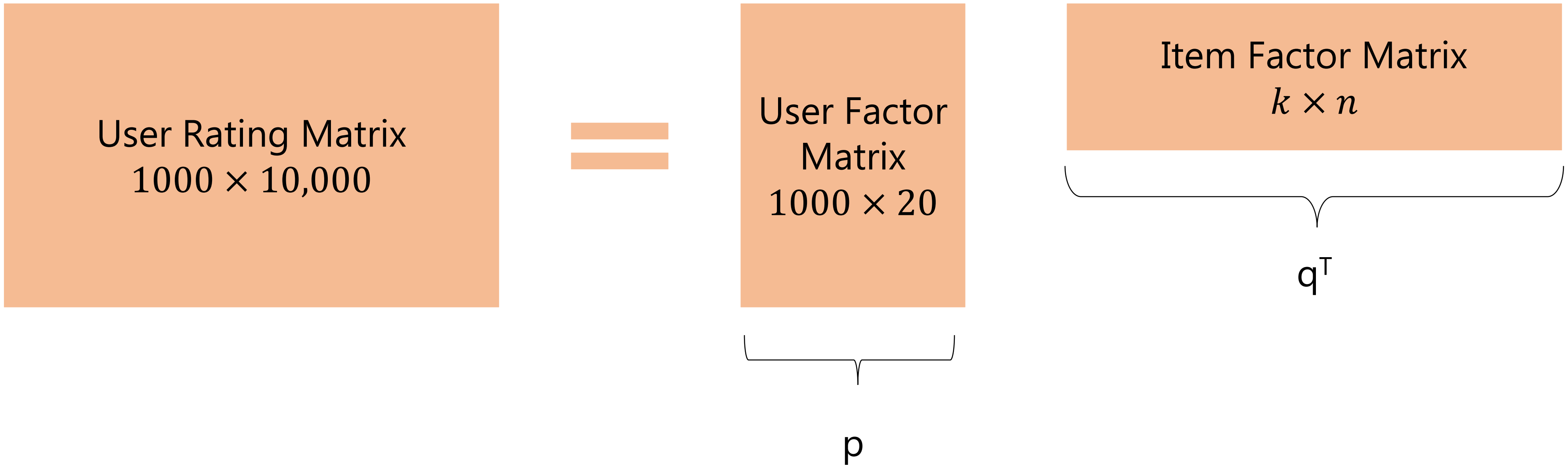
=

$$\underbrace{\text{User Factor Matrix } m \times k}_{p} \quad \underbrace{\text{Item Factor Matrix } k \times n}_{q^T}$$

$$\boxed{\mathbf{20 < \min(1000, 10000)}}$$

Matrix Factorization

Technical details of matrix factorization



$$\mathbf{20 < \min(1000, 10000)}$$

Matrix Factorization

Technical details of matrix factorization

User Rating Matrix
$1000 \times 10,000$

$=$

$\underbrace{\text{User Factor Matrix } 1000 \times 20}_{p}$

$\underbrace{\text{Item Factor Matrix } 20 \times 10000}_{q^T}$

$$\boxed{\mathbf{20 < \min(1000, 10000)}}$$

Recap

- Matrix factorization

Class

Recommendation Engines

Topic

Model Based Methods: Prediction and Estimation Using SVD and NMF

Matrix Factorization

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

$\boldsymbol{\mu}$ – Average of all the ratings in the data
$\boldsymbol{b_\mu}$ – Estimate of average ratings considering each user
$\boldsymbol{b_i}$ – Estimate of average ratings considering each item
$\boldsymbol{p_u}$ – User factor matrix obtained from matrix factorization
$\boldsymbol{q_i^T}$ – Item factor matrix

Matrix Factorization

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

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	?	3	?
User 2	4	?	2	1
User 3	3	?	4	3
User 4	?	?	?	?

What will be the form of User factor and Item factor matrix?

If a given item or user doesn't have any rating history, then that item or user is dropped before doing matrix factorization

p

Dim 1	Dim 2
0.8	0.4
0.6	-0.5
0.2	0.1

3×2

Each entry in this matrix talks about user preference for each of the 2 factors

Matrix Factorization

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

	Star Trek	Avatar	Spiderman	Hulk	b_u
User 1	4	?	3	?	1.75
User 2	4	?	2	1	1.75
User 3	3	?	4	3	2.5
User 4	?	?	?	?	0
b_i	2.75	0	2.75	1	

b_i - Computed the column averages

b_u- Computed the row averages

p

Dim 1	Dim 2
0.8	0.4
0.6	-0.5
0.2	0.1

3×2

q^T

	Star Trek	Spiderman	Hulk
Dim 1	0.6	0.8	-0.4
Dim 2	0.2	0.3	0.1

2×3

Matrix Factorization

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

	Star Trek	Avatar	Spiderman	Hulk	b_u
User 1	4	?	3	?	1.75
User 2	4	?	2	1	1.75
User 3	3	?	4	3	2.5
User 4	?	?	?	?	0
b_i	2.75	0	2.75	1	

p

Dim 1	Dim 2
0.8	0.4
0.6	-0.5
0.2	0.1

3×2

q^T

	Star Trek	Spiderman	Hulk
Dim 1	0.6	0.8	-0.4
Dim 2	0.2	0.3	0.1

2×3

Global average of ratings

$\mu = 1.5$

$\hat{r}_{14} = 1.5 + 1.75 + 1 + p_1 q_4^T$

$p_1 q_4^T = [0.8, 0.4] \times [-0.4, 0.1]$

$p_1 q_4^T = [-0.32 + 0.8 - 0.16 + 0.4]$

$p_1 q_4^T = -0.36$

$\hat{r}_{14} = 1.5 + 1.75 + 1 - 0.36$

$\hat{r}_{14} = 3.89$

Matrix Factorization

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

	Star Trek	Avatar	Spiderman	Hulk	b_u
User 1	4	?	3	?	1.75
User 2	4	?	2	1	1.75
User 3	3	?	4	3	2.5
User 4	?	?	?	?	0
b_i	2.75	0	2.75	1	

p

Dim 1	Dim 2
0.8	0.4
0.6	-0.5
0.2	0.1

3×2

q^T

	Star Trek	Spiderman	Hulk
Dim 1	0.6	0.8	-0.4
Dim 2	0.2	0.3	0.1

2×3

$$\mu = 1.5$$

$$\hat{r}_{22} = 1.5 + 1.75 + 0 + p_2 q_2^T$$

$$p_2 q_2^T = [0.6, -0.5] \times [0,0]$$

$$p_2 q_2^T = 0$$

$$\hat{r}_{22} = 1.5 + 1.75 + 0 + 0$$

$$\hat{r}_{22} = 3.25$$

Matrix factorization

How is the matrix factorization done?

How does Singular Value Decomposition work?

It will be discussed in the context of recommendation engines with an emphasis on providing an intuition on how things work

2 popular algorithms

Singular Value Decomposition (SVD)

Non Negative Matrix Factorization (NMF)

Singular Value Decomposition

Rating matrix

	Star Trek	Avatar	Spiderman	Hulk
User 1	4	?	3	?
User 2	4	?	2	1
User 3	3	?	4	3
User 4	?	?	?	?

Singular Value Decomposition

Rating matrix

	Star Trek	Spiderman	Hulk
User 1	4	3	?
User 2	4	2	1
User 3	3	4	3

It will be imputed with either the mean or 0 value

Depending on which implementation is being used to do the matrix factorization

Some implementations may not even impute the values

Singular Value Decomposition

Rating matrix

	Star Trek	Spiderman	Hulk
User 1	4	3	3.5
User 2	4	2	1
User 3	3	4	3

User Factor Matrix
$m \times k$

p

Item Factor Matrix
$k \times n$

q^T

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

Predictions

	Star Trek	Spiderman	Hulk
User 1	3.2	2.9	3.3
User 2	3.8	2.5	1.2
User 3	2.9	3.8	3.4

$$Error = (4 - 3.2)^2 + (3 - 2.9)^2 + \cdots + (3 - 3.4)^2$$

$$Error = \sum (r_{ui} - \hat{r}_{ui})^2$$

Singular Value Decomposition

Minimizing the error can lead to overfit

Regularization is mostly used to guard against overfit

Commonly used cost function -

$$\boldsymbol{Error} = \sum(\boldsymbol{r}_{\boldsymbol{ui}} - \hat{\boldsymbol{r}}_{\boldsymbol{ui}})^2 + \boldsymbol{\lambda}(\boldsymbol{b}_{\boldsymbol{u}}^2 + \boldsymbol{b}_{\boldsymbol{i}}^2 + ||\boldsymbol{q}_{\boldsymbol{i}}||^2 + ||\boldsymbol{p}_{\boldsymbol{u}}||^2)$$

Hyperparameter - λ

Parameters of SVD - $\boldsymbol{p}_{\boldsymbol{u}}$, $\boldsymbol{q}_{\boldsymbol{i}}$, $\boldsymbol{b}_{\boldsymbol{\mu}}$, and $\boldsymbol{b}_{\boldsymbol{i}}$

Non Negative Matrix Factorization

In the context of recommendation engines NMF is very similar to SVD

Only difference

Factors estimated in NMF are non negative

The same cost function is minimized as in the case of SVD

$$\boldsymbol{Error} = \sum(r_{ui} - \hat{r}_{ui})^2 + \lambda(b_u^2 + b_i^2 + ||q_i||^2 + ||p_u||^2)$$

With constraints that, p_u *and* q_i are positive

Cost Function

Minimize the cost functions

- Stochastic Gradient Descent
- Alternating Least Squares

Exact details of implementations can differ depending on the machine learning framework being used

Cost functions discussed here are only indicative of what is normally used by most implementations

Hyperparameters

Hyperparameters

- The number of epochs to be used (Stochastic Gradient Descent)
- The number of factors to be considered
- The learning rates
- The value of regularization term

The optimum values are found out using grid search and cross validation

Recap

- Matrix factorization
- Singular Value Decomposition
- Non Negative Matrix Factorization
- Cost Function
- Hyperparameters