

ToBig's 8기 조양규

# 추천 시스템

# Contents

---

Unit 01 | 추천알고리즘 종류

---

Unit 02 | CF

---

Unit 03 | Matrix Factorization

---

Unit 04 | 참고

---

Unit 05 | 실습

---

## Unit 01 | 추천 시스템의 종류

Recommendation systems  
come in

two types

Unit 01 | 추천 시스템의 종류

Content-based

vs

Collaborative  
Filtering

## Unit 01 | 추천 시스템의 종류

## 컨텐츠 기반 추천은

- 사전의 도메인 지식이 있어야하고..
- 개발컨텐츠의 메타데이터가 있어야하고..
- 상품에서 정보를 뽑아내기도 어렵다.
- 다양한 방법이 있지만 기본적인 CF를 먼저 알아보도록 한다.

## Unit 02 | 협업 필터링

## 협업 필터링

;collaborative filtering

많은 사용자들로부터 얻은 기호정보에 따라 사용자들의  
관심사들을 자동적으로 예측하게 해주는 방법



User-based 추천

(Neighborhood Models)



Item-based 추천

추천에 앞서 **사용자** 집단 / **아이템** 목록 / **Rating** 을 먼저 확인한다.  
Rating이 없다면 binary (항목이 있다 :1, 없다:0) 로 계산

## Unit 02 | 협업 필터링

## 사용자 기반 협업 필터링

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3.0	1.5	NA	3.0	2	NA
Lady in the Water	NA	3.0	3.0	2.5	3	NA
Snakes on a Plane	3.5	3.5	4.0	3.5	4	4.5
Superman Returns	4.0	5.0	5.0	3.5	3	4.0
The Night Listener	4.5	3.0	3.0	3.0	3	NA
You Me and Dupree	2.5	3.5	3.5	2.5	2	1.0

## 1. item/user matrix 만들기

## Unit 02 | 협업 필터링

## 2. 유저간의 유사도 측정

자주 쓰이는 유사도

- 유클리드 거리
- 코사인 유사도

$$\text{msd}(i, j) = \frac{1}{|U_{ij}|} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2$$

$$\text{msd}(u, v) = \frac{1}{|I_{uv}|} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2$$



## Unit 02 | 협업 필터링

## 2. 유저간의 유사도 측정

자주 쓰이는 유사도

- 유클리드 거리
- 코사인 유사도

$$\text{cosine\_sim}(i, j) = \frac{\sum_{u \in U_{ij}} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \cdot \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}$$

$$\text{cosine\_sim}(u, v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}}$$

## Unit 02 | 협업 필터링

## 2. 유저간의 유사도 측정

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3.0	1.5	NA	3.0	2	NA
Lady in the Water	NA	3.0	3.0	2.5	3	NA
Snakes on a Plane	3.5	3.5	4.0	3.5	4	4.5
Superman Returns	4.0	5.0	5.0	3.5	3	4.0
The Night Listener	4.5	3.0	3.0	3.0	3	NA
You Me and Dupree	2.5	3.5	3.5	2.5	2	1.0



	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Claudia Puig	1.0000000	0.7559289	0.9285714	0.9449112	0.6546537	0.8934051
Gene Seymour	0.7559289	1.0000000	0.9449112	0.5000000	0.0000000	0.3812464
Jack Matthews	0.9285714	0.9449112	1.0000000	0.7559289	0.3273268	0.6628490
Lisa Rose	0.9449112	0.5000000	0.7559289	1.0000000	0.8660254	0.9912407
Mick LaSalle	0.6546537	0.0000000	0.3273268	0.8660254	1.0000000	0.9244735
Toby	0.8934051	0.3812464	0.6628490	0.9912407	0.9244735	1.0000000

✓ Jack Matthews와 다른 사용자 간의 유사도 계산

## Unit 02 | 협업 필터링

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3.0	1.5	NA	3.0	2	NA
Lady in the Water	NA	3.0	3.0	2.5	3	NA
Snakes on a Plane	3.5	3.5	4.0	3.5	4	4.5
Superman Returns	4.0	5.0	5.0	3.5	3	4.0
The Night Listener	4.5	3.0	3.0	3.0	3	NA
You Me and Dupree	2.5	3.5	3.5	2.5	2	1.0

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Claudia Puig	1.0000000	0.7559289	0.9285714	0.9449112	0.6546537	0.8934051
Gene Seymour	0.7559289	1.0000000	0.9449112	0.5000000	0.0000000	0.3812464
Jack Matthews	0.9285714	0.9449112	1.0000000	0.7559289	0.3273268	0.6628490
Lisa Rose	0.9449112	0.5000000	0.7559289	1.0000000	0.8660254	0.9912407
Mick LaSalle	0.6546537	0.0000000	0.3273268	0.8660254	1.0000000	0.9244735
Toby	0.8934051	0.3812464	0.6628490	0.9912407	0.9244735	1.0000000

✓ Jack이 아직 평가하지 않은 영화(Just My Luck)의 등급 계산

$$\triangleright (3 \times 0.9286 + 1.5 \times 0.9449 + 3 \times 0.7559 + 2 \times 0.3273) / (0.9286 + 0.9449 + 0.7559 + 0.3273)$$

$$= 2.4099$$

## Unit 02 | 협업 필터링

## 아이템 기반 협업 필터링

사용자가 과거에 아이템 A를 좋아했다면 A와 유사한 B도 좋아할 것이다

→ 아이템 간의 유사도를 이용한 추천

## Unit 02 | 협업 필터링

## 아이템 기반 협업 필터링

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3.0	1.5	NA	3.0	2	NA
Lady in the Water	NA	3.0	3.0	2.5	3	NA
Snakes on a Plane	3.5	3.5	4.0	3.5	4	4.5
Superman Returns	4.0	5.0	5.0	3.5	3	4.0
The Night Listener	4.5	3.0	3.0	3.0	3	NA
You Me and Dupree	2.5	3.5	3.5	2.5	2	1.0

## 1. item/user matrix 만들기

## Unit 02 | 협업 필터링

## 2. Item간의 유사도 측정

자주 쓰이는 유사도

- 유클리드 거리
- 코사인 유사도

$$\text{msd}(i, j) = \frac{1}{|U_{ij}|} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2$$

$$\text{msd}(u, v) = \frac{1}{|I_{uv}|} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2$$

## Unit 02 | 협업 필터링

## 2. Item간의 유사도 측정

자주 쓰이는 유사도

- 유클리드 거리
- 코사인 유사도

$$\text{cosine\_sim}(i, j) = \frac{\sum_{u \in U_{ij}} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \cdot \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}$$

$$\text{cosine\_sim}(u, v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}}$$

## Unit 02 | 협업 필터링

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3.0	1.5	NA	3.0	2	NA
Lady in the Water	NA	3.0	3.0	2.5	3	NA
Snakes on a Plane	3.5	3.5	4.0	3.5	4	4.5
Superman Returns	4.0	5.0	5.0	3.5	3	4.0
The Night Listener	4.5	3.0	3.0	3.0	3	NA
You Me and Dupree	2.5	3.5	3.5	2.5	2	1.0



	Just My Luck	Lady in the water	Snakes on a Plane	Superman Returns	The Night Listener	You Me and Dupree
Just My Luck	1	0.633	0.73	0.71	0.89	0.75
Lady in the water	0.63	1	0.79	0.81	0.79	0.88
Snakes on a Plane	0.73	0.79	1	0.97	0.85	0.92
Superman Returns	0.71	0.81	0.97	1	0.88	0.96
The Night Listener	0.89	0.79	0.85	0.88	1	0.94
You Me and Dupree	0.75	0.89	0.92	0.96	0.94	1

✓ Lady in the water에 대한 Toby의 등급 예측

- ▷ Lady in the water는 You, me and Dupree와 가장 유사(0.8897)
- ▷ Toby가 평가한 각 영화에 대한 Lady in the water의 유사도 점수와 그에 해당되는 등급을 곱한 후 모두 합한다.  
그리고 최종합을 Lady in the Water의 유사도 점수의 합으로 나눈다.



## Unit 02 | 협업 필터링

	Claudia Puig	Gene Seymour	Jack Matthews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3.0	1.5	NA	3.0	2	NA
Lady in the Water	NA	3.0	3.0	2.5	3	NA
Snakes on a Plane	3.5	3.5	4.0	3.5	4	4.5
Superman Returns	4.0	5.0	5.0	3.5	3	4.0
The Night Listener	4.5	3.0	3.0	3.0	3	NA
You Me and Dupree	2.5	3.5	3.5	2.5	2	1.0



	Just My Luck	Lady in the water	Snakes on a Plane	Superman Returns	The Night Listener	You Me and Dupree
Just My Luck	1	0.633	0.73	0.71	0.89	0.75
Lady in the water	0.63	1	0.79	0.81	0.79	0.88
Snakes on a Plane	0.73	0.79	1	0.97	0.85	0.92
Superman Returns	0.71	0.81	0.97	1	0.88	0.96
The Night Listener	0.89	0.79	0.85	0.88	1	0.94
You Me and Dupree	0.75	0.89	0.92	0.96	0.94	1

✓ Lady in the water에 대한 Toby의 등급 예측

$$\triangleright (0.795 \cdot 4.5 + 0.814 \cdot 4 + 0.889 \cdot 1) / (0.795 + 0.814 + 0.889) = 3.09$$

## Unit 02 | 협업 필터링

## User-based 추천

데이터 양이 작고,  
데이터 변경이 자주 일어나는 경우

실시간으로 유사도 계산

## Item-based 추천

데이터 양이 크고,  
데이터 변경이 자주 일어나지 않는 경우

항목간의 유사도를 저장하여 사용

## Unit 02 | 협업 필터링

CF의 구현은 쉽다 But

1. 사용자와 아이템에 대한 정보가 전혀 없는 경우 문제  
Cold Start Problem
2. Rating 이외 정보 고려 X

## Unit 03 | Matrix Factorization

## Latent Factor 모형

사용자의 특성 벡터나 상품의 특성 벡터의 길이가 매우 커질 경우

Latent Factor 모형은 행렬 분해를 통해 간략화

PCA를 사용하면 긴 특성 벡터를 소수의 차원으로 축소할 수 있는 것과 비슷

## Unit 03 | Matrix Factorization

## Latent Factor 모형

영화에 대한 평점을 주는 경우, 코미디, 액션, 드라마 장르에서  
사용자는 특정한 장르에 더 점수를 많이 주거나 적게 줄 수 있다.

그리고 영화 자체도 이러한 장르 요인을 가지고 있다면 해당 사용자의 그 영화에 대한  
평점은 사용자의 장르 요인 벡터와 영화의 장르 요인 벡터의 내적으로 표시할 수 있다.

## Unit 03 | Matrix Factorization

## Latent Factor 모형

예를 들어 액션을 싫어하고(-1) 코미디(2)나 드라마(3)를 좋아하는 사용자의 요인 벡터는 다음과 같다.

$$p_u^T = (-1, 2, 3)$$

어떤 영화가 액션 요소가 2이고 코미디 요소가 1이고, 드라마 요소가 1이라면

$$q_i^T = (2, 1, 1)$$

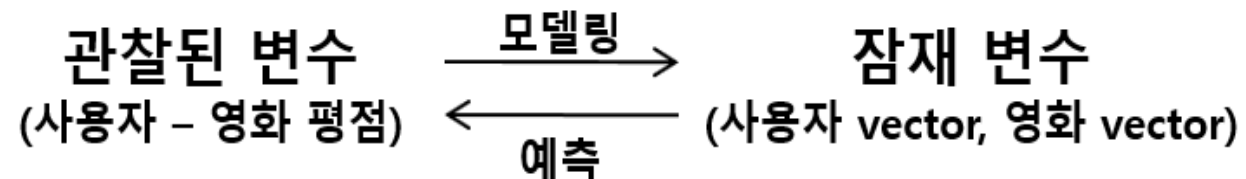
평점은 다음과 같을 것이다.

$$r_{ui} = q_i^T p_u = -1 \cdot 2 + 2 \cdot 1 + 3 \cdot 1 = 3$$

## Unit 03 | Matrix Factorization

## Latent Factor Modeling 방법?

관찰된 변수를 잠재 변수로 어떻게 나타낼 수 있을까?



→ Matrix Factorization !

## Unit 03 | Matrix Factorization

# Matrix Factorization

Matrix Factorization 방법은 모든 사용자와 상품에 대해 다음 오차 함수를 최소화하는 요인 벡터를 찾아낸다. 즉 다음과 같은 행렬  $P, Q$ 를 찾는다.

$$R \approx PQ^T$$

여기에서

- $R \in \mathbf{R}^{m \times n}$  :  $m$  사용자와  $n$  상품의 평점 행렬
- $P \in \mathbf{R}^{m \times k}$  :  $m$  사용자와  $k$  요인의 관계 행렬
- $Q \in \mathbf{R}^{n \times k}$  :  $n$  상품의와  $k$  요인의 관계 행렬



## Unit 03 | Matrix Factorization

## Matrix Factorization

행렬 인수분해는 하나의 행렬을 행렬들의 곱으로 나타내는 것이다.  
행렬 인수분해 방법은 SVD, LU 분해, QR 분해 등이 있다.

출처: wikipedia – matrix decomposition

예시)

1	2
2	4
3	6

=

1
2
3

\*

1	2
---	---

$$R(\text{예상 평점}) = P(\text{user}) * Q(\text{item})$$

## Unit 03 | Matrix Factorization

Matrix Factorization

행렬 인수분해는 하나의 행렬을 행렬들의 곱으로 나타내는 것이다.  
 행렬 인수분해 방법은 SVD, LU 분해, QR 분해 등이 있다.

출처: wikipedia – matrix decomposition

예시)

user \ item	링	미니언즈	주온	알라딘
Gus	2	4.5	2	4
Dave	4	4	3.5	3.5
Jane	3	5	3.5	2
Anna	4.5	3.5	4	1

=

user \ 특징	애니메이션	호러
Gus	1.2	0.6
Dave	1.4	1.2
Jane	1.3	1
Anna	0.8	1.3

\*

특징 \ item	링	미니언즈	주온	알라딘
애니메이션	0.7	1.5	0.4	1.2
호러	1.7	0.6	1.1	0.4

$$R(\text{예상 평점}) = P(\text{user}) * Q(\text{item})$$

행렬 인수분해를 알았으니 우리는 사용자- 영화 평점 데이터를 잠재변수인 사용자 vector와 영화 vector로 나타낼 수 있다.

## Unit 03 | Matrix Factorization

### Matrix Factorization 방법?

Matrix Factorization을 위한 여러 가지 방법이 있다고 하였다. 어떤 것을 선택할까?

선택하기 전에, 우리의 목적이 뭐였더라..

→사용자와 제품들 간의 상호작용(평점, 클릭 여부, 검색 여부..)을 분석해서 **사용자와 상호작용하지 않은 제품에 대한 선호도를 예측해서 추천**해주는 시스템

## Unit 03 | Matrix Factorization

### 목적 함수

Matrix Factorization을 통한 학습의 목적이 있어야 한다. 수식적으로 이를 나타냈을 때 목적 함수라고 정의하며 학습을 통해 함수가 목적을 달성하면 우리의 목적을 달성할 수 있다!

우리의 목적을 풀어서 쓰면 예상 평점과 실제 평점의 차이의 모든 데이터에 대한 총합을 최소화하는 것이다.

$$L = \sum_{(u,i) \in \kappa} (r_{ui} - q_i^T p_u)^2 + \lambda_q \sum_i \|q_i\|^2 + \lambda_p \sum_u \|p_u\|^2 \quad : \text{목적 함수}$$

## Unit 03 | Matrix Factorization

## 참고) 학습 방법

$$L = \sum_{(u,i) \in K} (r_{ui} - q_i^T p_u)^2 + \lambda_q \sum_i \|q_i\|^2 + \lambda_p \sum_u \|p_u\|^2$$

## - SGD(Stochastic Gradient Descent)

1. 특정 지점에서 어디로 가야 최소로 가는 방향인지 확인한다. (gradient)

$$\frac{\partial L}{\partial q_i} = (r_{ui} - q_i^T p_u) p_u - \lambda_q q_i$$

$$\frac{\partial L}{\partial p_u} = (r_{ui} - q_i^T p_u) q_i - \lambda_p p_u$$

2. Gradient를 일정 부분 사용자 vector, 제품 vector에 각각 update를 해준다.

$$q_i \leftarrow q_i + \eta ((r_{ui} - q_i^T p_u) p_u - \lambda_q q_i)$$

$$p_u \leftarrow p_u + \eta ((r_{ui} - q_i^T p_u) q_i - \lambda_p p_u)$$

※ 1, 2의 과정을 수렴할 때까지 반복한다.

## - ALS(Alternating Least Square)

1. 사용자 vector를 상수라고 가정하자.
2. L의 제품 vector에 대한 미분 값이 0(최소값)일 때의 제품 vector를 구한다.
3. 2의 vector를 제품 vector에 업데이트 한다.

$$\frac{\partial L}{\partial q_i} = -2 \sum_i (r_{ui} - q_i^T p_u) p_u^T + 2\lambda_q q_i^T$$

$$0 = -(r_i - q_i^T P^T) P + \lambda_q q_i^T$$

$$q_i^T (P^T P + \lambda_q I) = r_i P$$

$$q_i^T = r_i P (P^T P + \lambda_q I)^{-1}$$

4. 제품 vector를 상수라고 가정하자.
5. L의 사용자 vector에 대한 미분 값이 0 (최소값)일 때의 사용자 vector를 구한다.
6. 5의 vector를 사용자 vector에 업데이트 한다.

$$\frac{\partial L}{\partial p_u} = -2 \sum_i (r_{ui} - p_u^T q_i) q_i^T + 2\lambda_p p_u^T$$

$$0 = -(r_u - p_u^T Q^T) Q + \lambda_p p_u^T$$

$$p_u^T (Q^T Q + \lambda_p I) = r_u Q$$

$$p_u^T = r_u Q (Q^T Q + \lambda_p I)^{-1}$$

※ 1~6의 과정을 수렴할 때까지 반복한다.  
(\*수렴 여부에 대한 증명은 생략)

## Unit 03 | Matrix Factorization

## 참고) 목적함수 변경

## - Bias(편향):

- **user**: 좀 더 너그러운 사용자가 있는 반면 비판적인 사용자가 있다.
- **item**: 인기가 많은 제품 군이 있는 반면 항상 인기가 없는 제품 군이 있다.

## - 목적 함수 변경:

$$L = \sum_{u,i \in K} (r_{ui} - (\mu + b_u + b_i + q_i^T p_u))^2 + \lambda_{pb} \sum_u \|b_u\|^2 + \lambda_{qb} \sum_i \|b_i\|^2 + \lambda_{pf} \sum_u \|p_u\|^2 + \lambda_{qf} \sum_i \|q_i\|^2$$

전체 평점  
평균user의  
biasitem의  
bias

→ 전체 평점의 평균을 기준으로 잡고 item, user의 치우친 정도를 고려했을 때 예상 평점과 실제 평점과의 차이의 총합을 최소화 하겠다.

## Unit 03 | Matrix Factorization

## 참고) 목적함수 변경

## - Implicit feedback

- **preference(선호도)**: user의 관심 여부를 나타내며 1, 0의 값으로 나타낸다.
- **confidence(신뢰도)**: user의 관심을 얼마나 신뢰할 수 있을 것인지를 나타내며 관심을 보여준 정도(*implicit feedback*: 마우스 클릭 등)에 비례한다.

→ Implicit feedback은 explicit과 다르게 user의 의증을 정확하게 파악하기가 어렵다. 그래서 관심을 준 item에 대해서는 '선호함' 이라고 가정하고 이에 대한 가중치를 feedback의 값에 비례해서 '신뢰한다' 고 가정한다.

## - 목적 함수 변경:

$$L = \sum_{u,i \in \kappa} \underbrace{c_{ui}}_{\text{신뢰도}} \left( \underbrace{r_{ui}}_{\text{선호도}} - q_i^T p_u \right)^2 + \lambda_p \sum_u \|p_u\|^2 + \lambda_q \sum_i \|q_i\|^2$$

→ user의 관심 여부를 예측해서 실제와의 차이를 최소화한다. 대신 신뢰도만큼 가중치를 뒀서 가중치가 높은 데이터는 더 주의 깊게 본다.

## Unit 03 | Matrix Factorization

참고) SVD

# SVD (Singular Value Decomposition)


SVD (Singular Value Decomposition) 는 Matrix Factorization 문제를 푸는 방법 중 하나이다.

$m \times n$  크기의 행렬  $R$ 은 다음과 같이 세 행렬의 곱으로 나타낼 수 있다. 이를 특이치 분해(Singular Value Decomposition) 라고 한다.

$$R = U\Sigma V^T$$

이 식에서

- $U$  는  $m \times m$  크기의 행렬로 역행렬이 대칭 행렬
- $\Sigma$  는  $m \times n$  크기의 행렬로 비대각 성분이 0
- $V$  는  $n \times n$  크기의 행렬로 역행렬이 대칭 행렬



[https://ko.wikipedia.org/wiki/%EA%B3%A0%EC%9C%A0%EA%B0%92\\_%EB%B6%84%ED%95%B4](https://ko.wikipedia.org/wiki/%EA%B3%A0%EC%9C%A0%EA%B0%92_%EB%B6%84%ED%95%B4)  
(고유값 분해 하는법)



## Unit 03 | Matrix Factorization

## 참고) SVD

$\Sigma$ 의 대각 성분은 특이치라고 하며 전체 특이치 중에서 가장 값이 큰  $k$ 개의 특이치만을 사용하여 (Truncated SVD), 다음과 같은 행렬을 만들 수 있다.

- $\hat{U}$  는  $U$ 에서 가장 값이 큰  $k$ 개의 특이치에 대응하는  $k$ 개의 성분만을 남긴  $m \times k$  크기의 행렬
- $\hat{\Sigma}$  는 가장 값이 큰  $k$ 개의 특이치에 대응하는  $k$ 개의 성분만을 남긴  $k \times k$  크기의 대각 행렬
- $\hat{V}$  는  $V$ 에서 가장 값이 큰  $k$ 개의 특이치에 대응하는  $k$ 개의 성분만을 남긴  $k \times n$  크기의 행렬

이 행렬을 다시 조합하면 원래의 행렬과 같은 크기를 가지고 유사한 원소를 가지는 행렬을 만들 수 있다.

$$\hat{U}\hat{\Sigma}\hat{V}^T = \hat{R} \approx R$$

하지만 실제로 평점 행렬은 빈 원소가 많은 sparse 행렬로서 SVD를 바로 적용하기 힘들기 때문에 행렬  $P$ ,  $Q$ 는 다음과 같은 모형에 대해 오차 함수를 최소화하여 구한다.

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

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda (b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$

## Unit 04 | 참고

## 참고링크

SVD / MF

<http://darkpgmr.tistory.com/106>

<http://sanghyukchun.github.io/73/>

코세라 추천시스템 강의

<https://www.coursera.org/specializations/recommender-systems>

파이썬 Surprise 패키지

<https://datascienceschool.net/view-notebook/fcd3550f11ac4537acec8d18136f2066/>



들어주셔서 감사합니다.