그래프를 이용한 기계 학습

#8 그래프를 추천시스템에 어떻게 활용할까? (심화)

신기정
(KAIST AI대학원)

boostcamp AI Tech

1. 추천시스템 기본 복습

2. 넷플릭스 챌린지 소개

3. 기본 잠재 인수 모형

4. 고급 잠재 인수 모형

5. 넷플릭스 챌린지의 결과

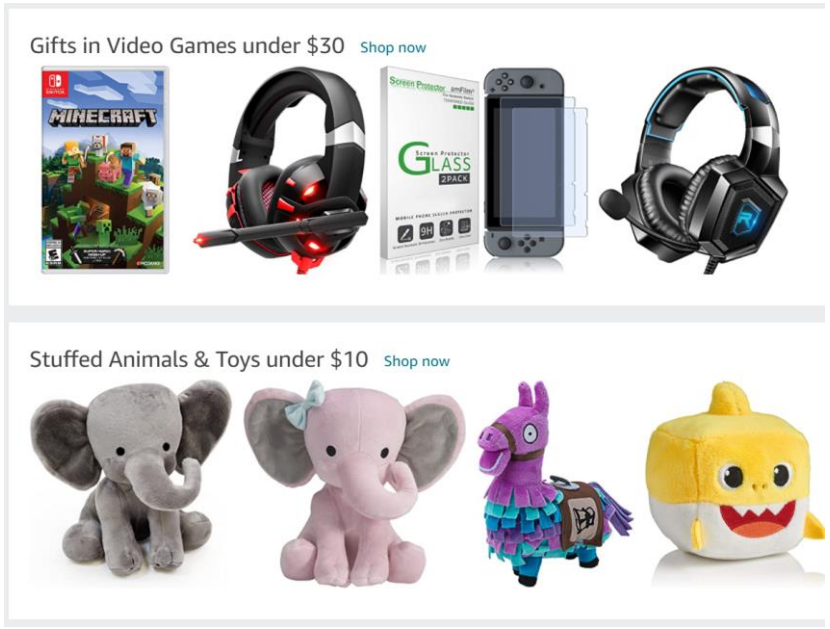
6. 실습: Surprise 라이브러리와 잠재 인수 모형의 활용

1. 추천시스템 기본 복습

1.1 추천시스템 예시
1.2 추천시스템과 그래프
1.3 내용 기반 추천시스템
1.4 협업 필터링
1.5 추천시스템의 평가

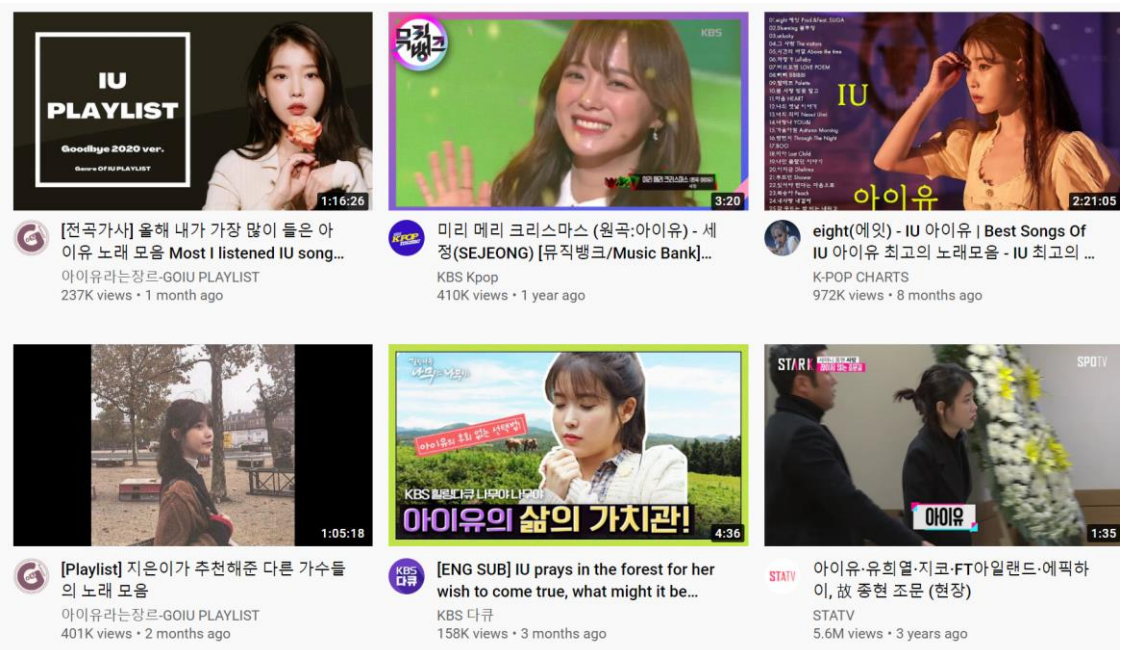
1.1 추천 시스템 예시

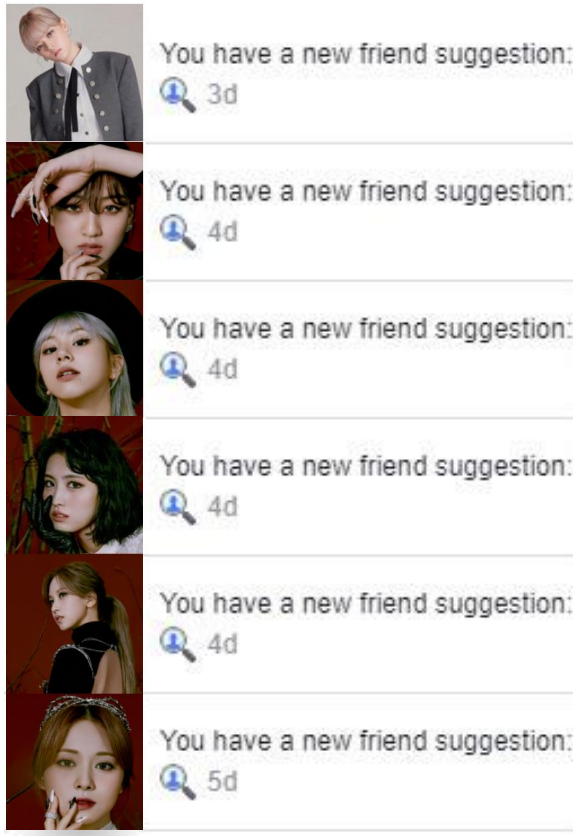
Amazon.com (상품), 넷플릭스 (영화), 유튜브 (영상), 페이스북 (친구)



1.1 추천 시스템 예시

Amazon.com (상품), 넷플릭스 (영화), 유튜브 (영상), 페이스북 (친구)





1.2 추천 시스템과 그래프

추천 시스템은 사용자 각각이 구매할 만한 혹은 선호할 만한 상품/영화/영상을 추천합니다

추천 시스템의 핵심은 사용자별 구매를 예측하거나 선호를 추정하는 것입니다

그래프 관점에서 추천 시스템은

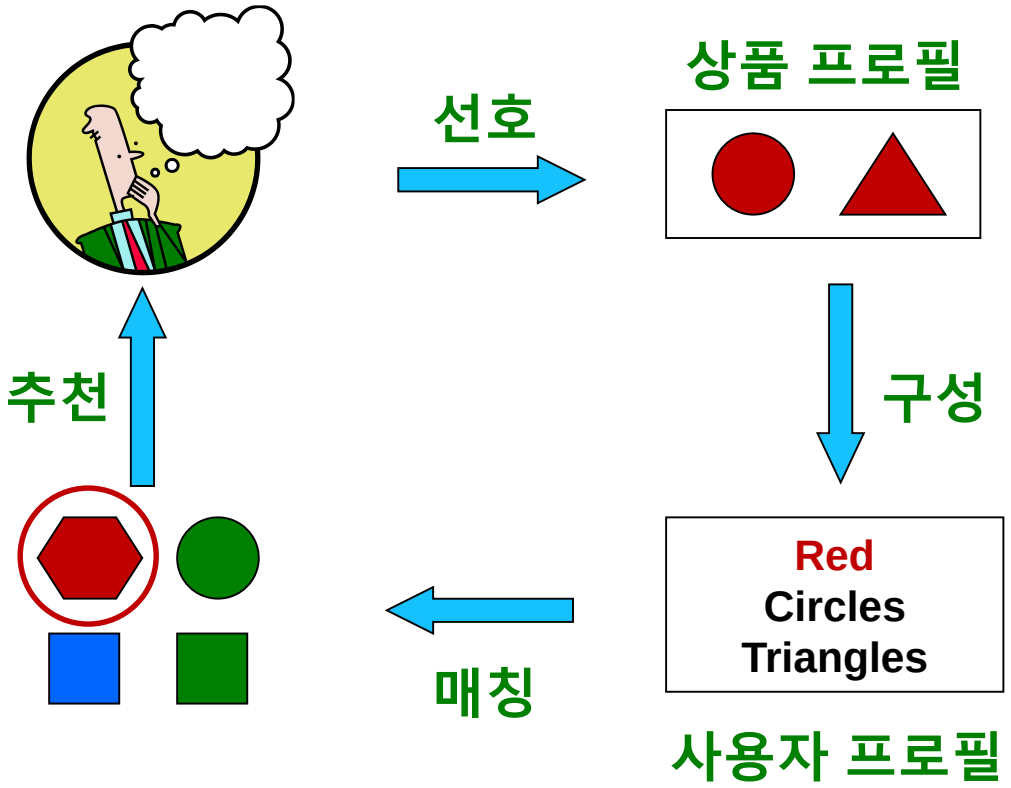
"미래의 간선을 예측하는 문제" 혹은

"누락된 간선의 가중치를 추정하는 문제"로

해석할 수 있습니다

1.3 내용 기반 추천시스템

내용 기반 추천은 각 사용자가 구매/만족했던 상품과 유사한 것을 추천하는 방법입니다



- 동일한 장르의 영화를 추천하는 것
- 동일한 감독의 영화 혹은 동일 배우가 출현한 영화를 추천하는 것
- 동일한 카테고리의 상품을 추천하는 것
- 동갑의 같은 학교를 졸업한 사람을 친구로 추천하는 것

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

1.3 내용 기반 추천시스템

내용 기반 추천시스템은 다음 장/단점을 같습니다

(+) 다른 사용자의 구매 기록이 필요하지 않습니다

(+) 독특한 취향의 사용자에게도 추천이 가능합니다

(+) 새 상품에 대해서도 추천이 가능합니다

(+) 추천의 이유를 제공할 수 있습니다

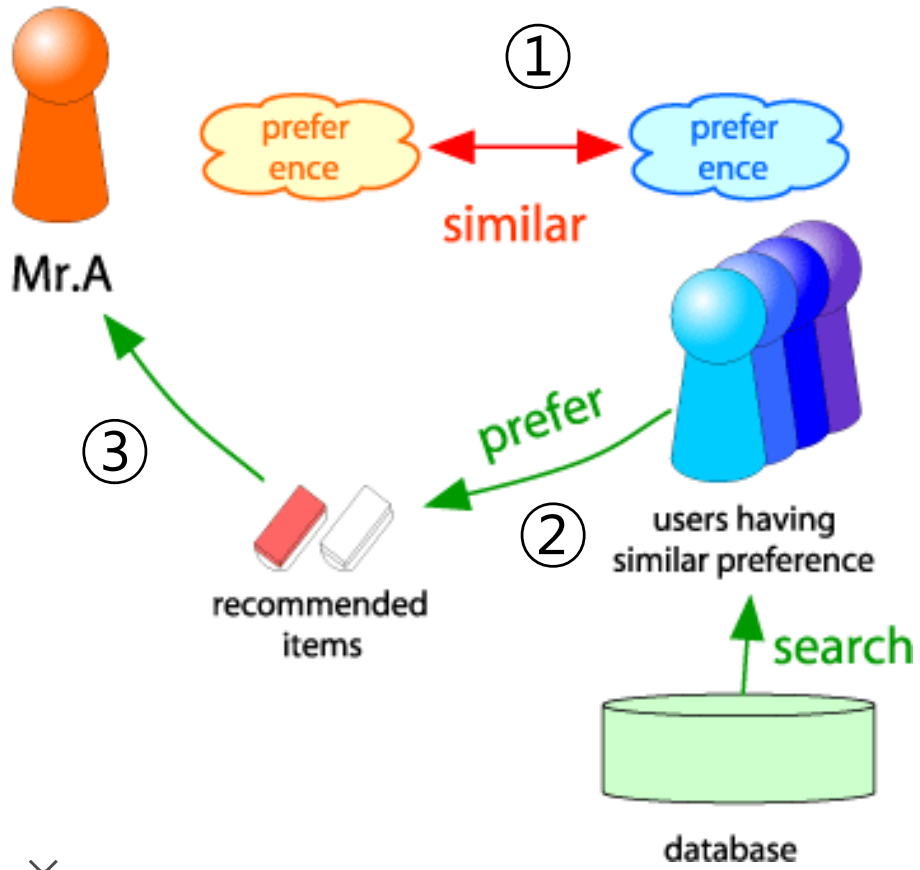
(-) 상품에 대한 부가 정보가 없는 경우에는 사용할 수 없습니다

(-) 구매 기록이 없는 사용자에게는 사용할 수 없습니다

(-) 과적합(Overfitting)으로 지나치게 협소한 추천을 할 위험이 있습니다

1.4 협업 필터링

협업 필터링은 유사한 취향의 사용자들이 선호/구매한 상품을 추천하는 방법입니다



추천의 대상 사용자를 x라고 합시다

우선 x 와 유사한 취향의 사용자들을 찾습니다

다음 단계로 유사한 취향의 사용자들이
선호한 상품을 찾습니다

마지막으로 이 상품들을 x 에게 추천합니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

1.4 협업 필터링

협업 필터링은 다음의 장/단점을 갖습니다

(+) 상품에 대한 부가 정보가 없는 경우에도 사용할 수 있습니다

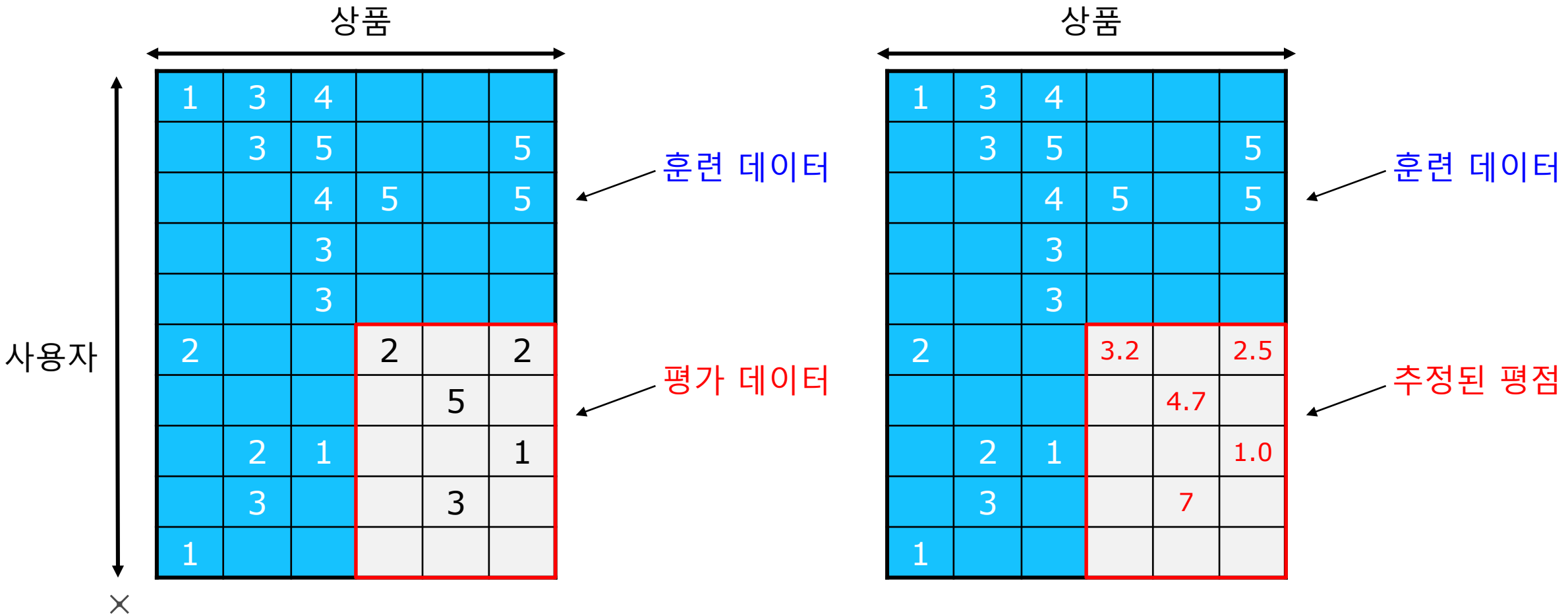
(–) 충분한 수의 평점 데이터가 누적되어야 효과적입니다

(–) 새 상품, 새로운 사용자에 대한 추천이 불가능합니다

(–) 독특한 취향의 사용자에게 추천이 어렵습니다

1.5 추천시스템의 평가

훈련 데이터를 이용하여 추정한 점수를 평가 데이터와 비교하여 정확도를 측정합니다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

1.5 추천시스템의 평가

훈련 데이터를 이용하여 추정한 점수를 평가 데이터와 비교하여 정확도를 측정합니다

오차를 측정하는 지표로는 평균 제곱 오차(Mean Squared Error, MSE)가 많이 사용됩니다
평가 데이터 내의 평점들을 집합을 T라고 합시다
평균 제곱 오차는 아래 수식으로 계산합니다

$$\frac{1}{|T|}\sum_{r_{xi}\in T}(r_{xi}-\hat{r}_{xi})^2$$

평균 제곱급 오차(Root Mean Squared Error, RMSE)도 많이 사용됩니다

$$\sqrt{\frac{1}{|T|}\sum_{r_{xi}\in T}(r_{xi}-\hat{r}_{xi})^2}$$

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

2. 넷플릭스 챌린지 소개

1.1 넷플릭스 챌린지 데이터셋

1.2 넷플릭스 챌린지 대회 소개

2.1 넷플릭스 챌린지 데이터셋

넷플릭스 챌린지(Netflix Challenge)에서는 사용자별 영화 평점 데이터가 사용되었습니다

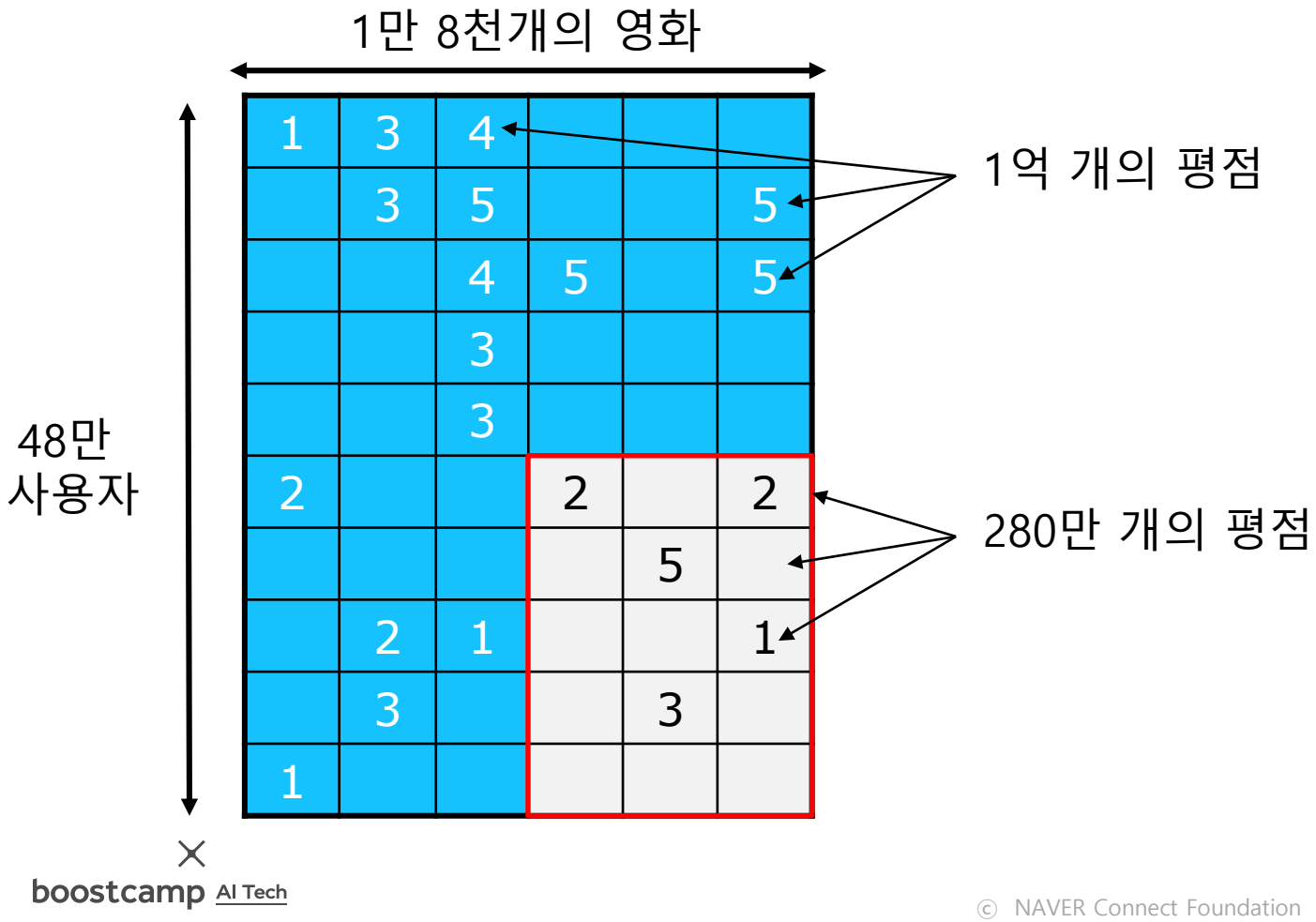
훈련 데이터(Training Data)는 2000년부터 2005년까지 수집한
48만명 사용자의 1만 8천개의 영화에 대한 1억 개의 평점으로 구성되어 있습니다

평가 데이터(Test Data)는 각 사용자의 최신 평점 280만개로 구성되어 있습니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

2.1 넷플릭스 챌린지 데이터셋

넷플릭스 챌린지(Netflix Challenge)에서는 사용자별 영화 평점 데이터가 사용되었습니다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

2.2 넷플릭스 챌린지 대회 소개

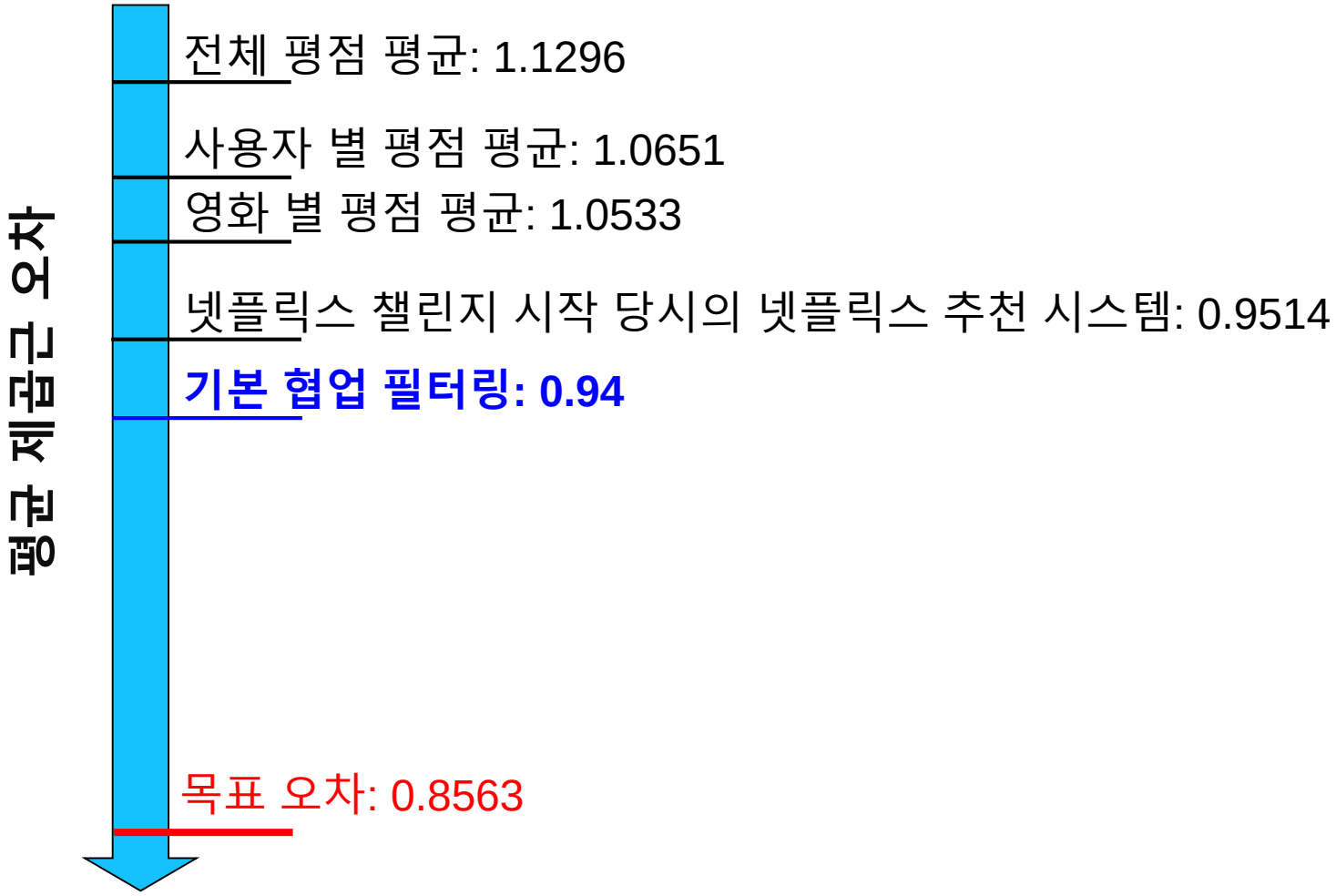
넷플릭스 챌린지의 목표는 추천시스템의 성능을 10%이상 향상시키는 것이었습니다

평균 제곱근 오차 0.9514을 0.8563까지 낮출 경우 100만불의 상금을 받는 조건이었습니다

2006년부터 2009년까지 진행되었으며, 2700개의 팀이 참여하였습니다
넷플릭스 챌린지를 통해 추천시스템의 성능이 비약적으로 발전했습니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

2.2 넷플릭스 챌린지



3. 잠재 인수 모형

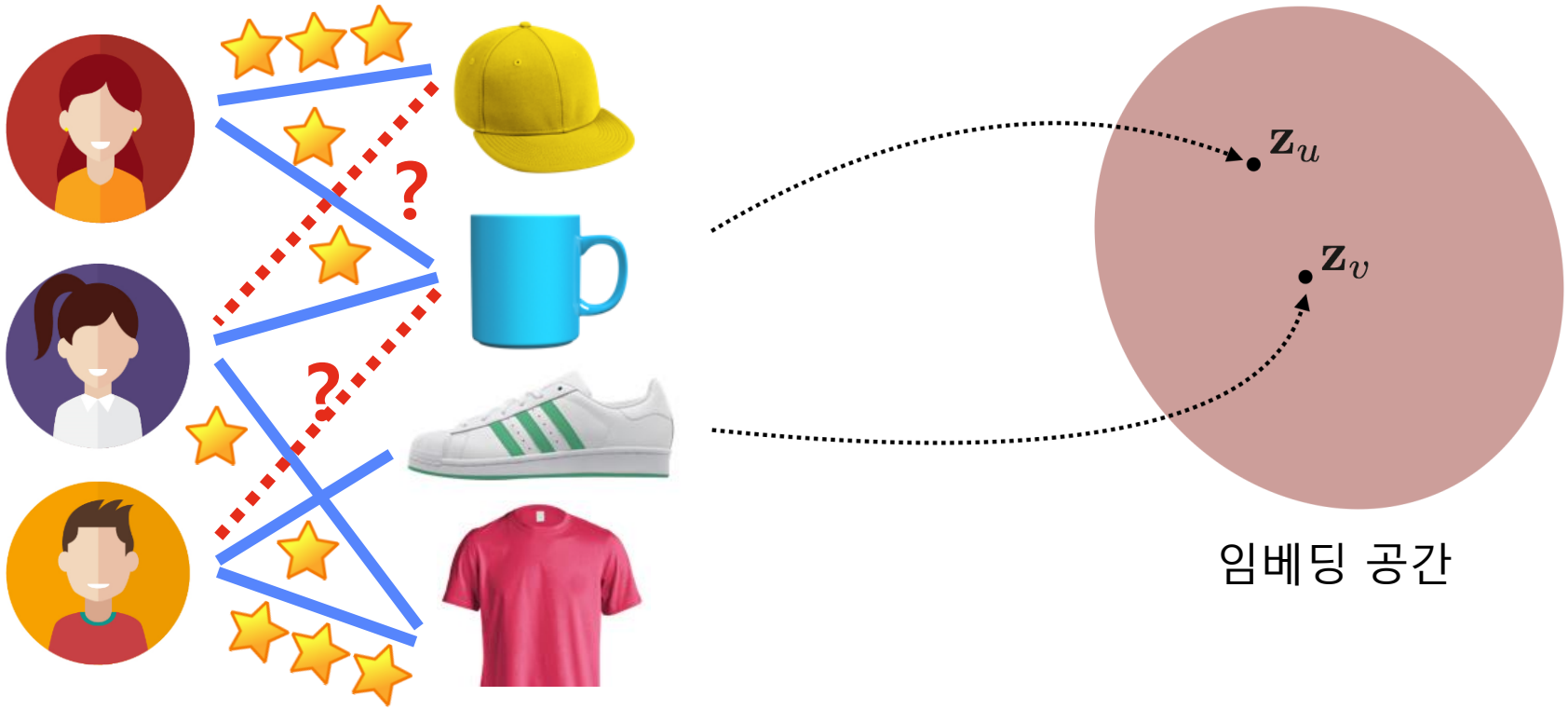
3.1 잠재 인수 모형 개요

3.2 손실 함수

3.3 최적화

3.1 잠재 인수 모형 개요

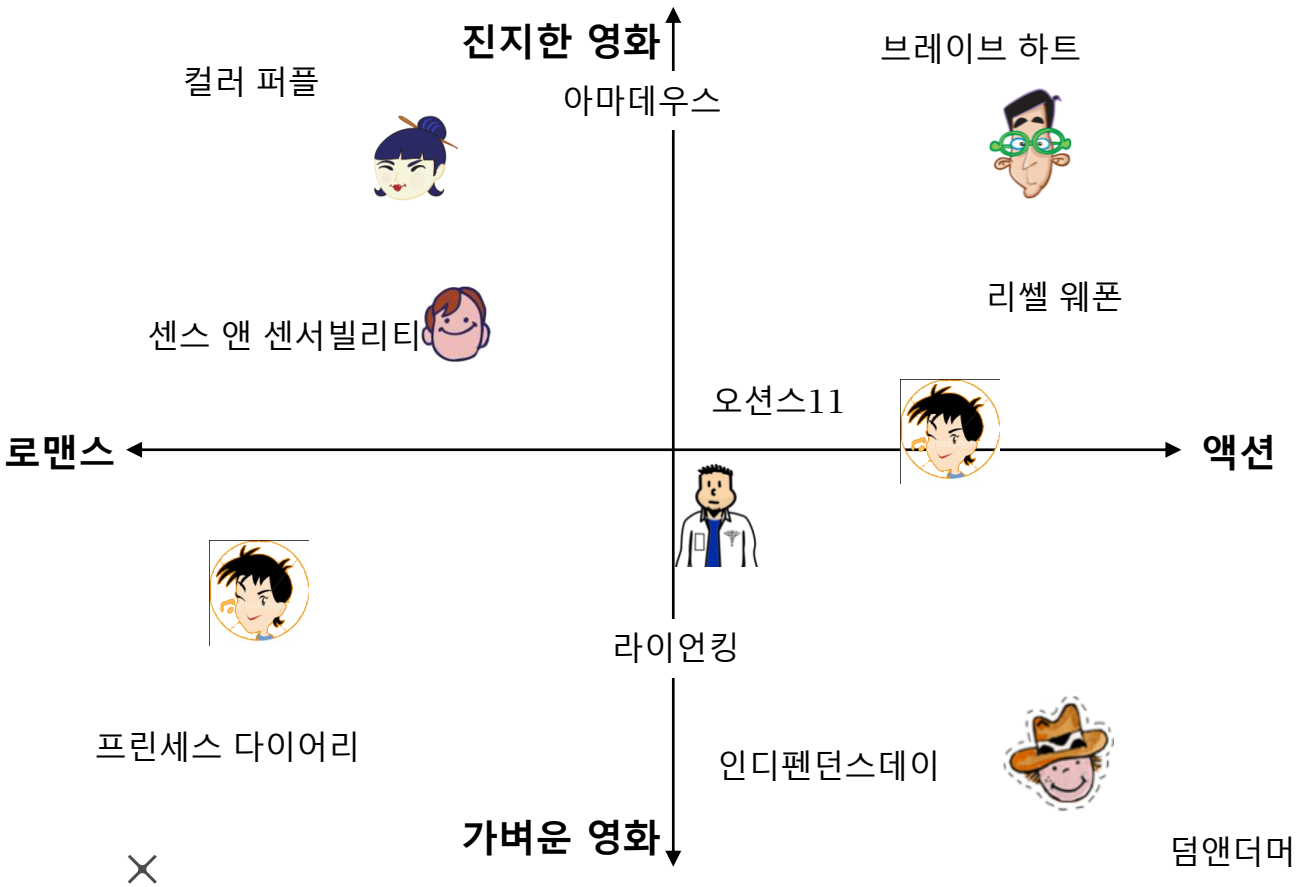
잠재 인수 모형(Latent Factor Model)의 핵심은 **사용자와 상품을 벡터로 표현**하는 것입니다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

3.1 잠재 인수 모형 개요

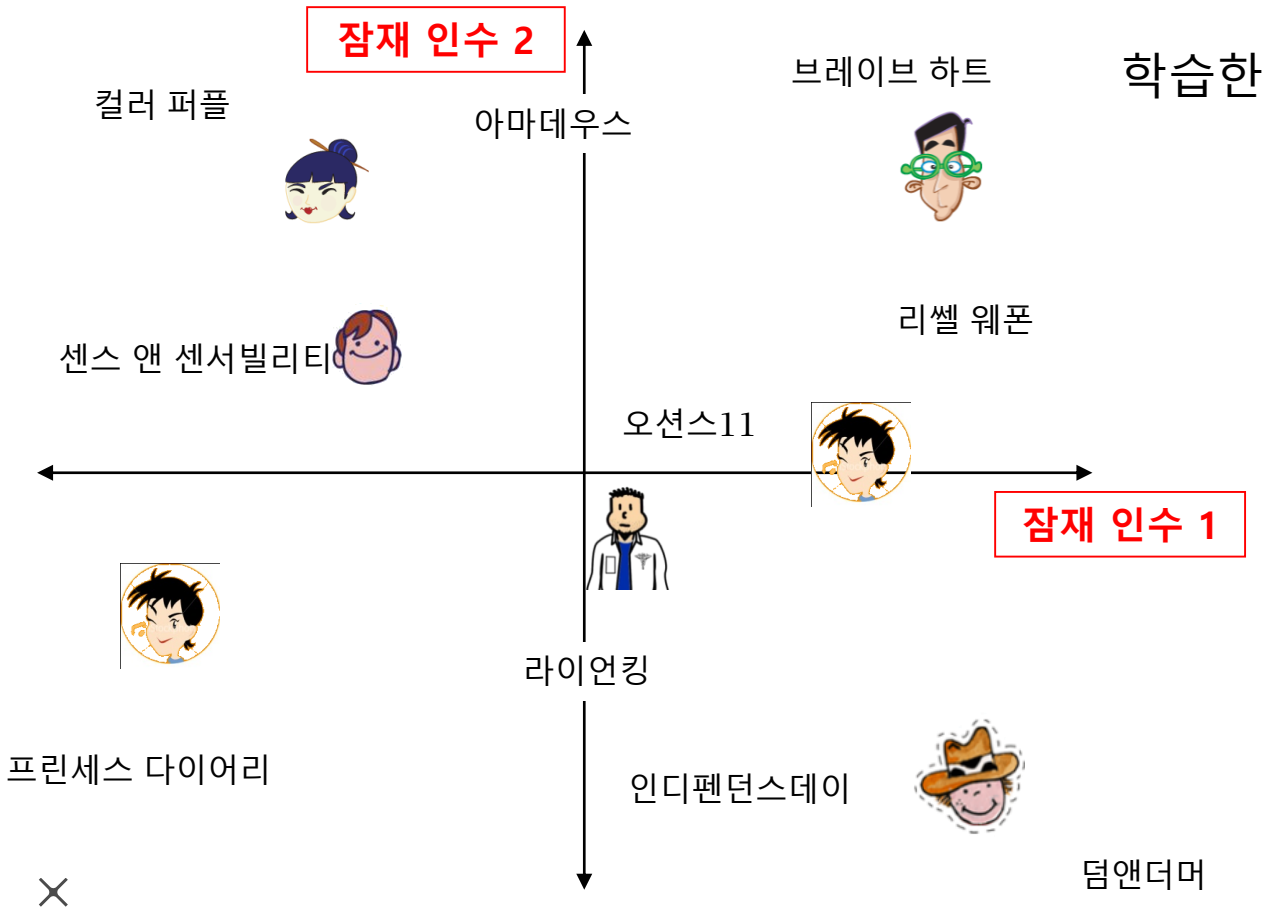
사용자와 영화를 임베딩한 예시입니다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

3.1 잠재 인수 모형 개요

잠재 인수 모형에서는 고정된 인수 대신 효과적인 인수를 학습하는 것을 목표로 합니다



학습한 인수를 잠재 인수(Latent Factor)라 부릅니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

3.2 손실 함수

사용자와 상품을 임베딩하는 기준은 무엇인가요?

사용자와 상품의 임베딩의 내적(Inner Product)이 평점과 최대한 유사하도록 하는 것입니다

사용자 x의 임베딩을 p_x, 상품 i의 임베딩을 q_i라고 합시다

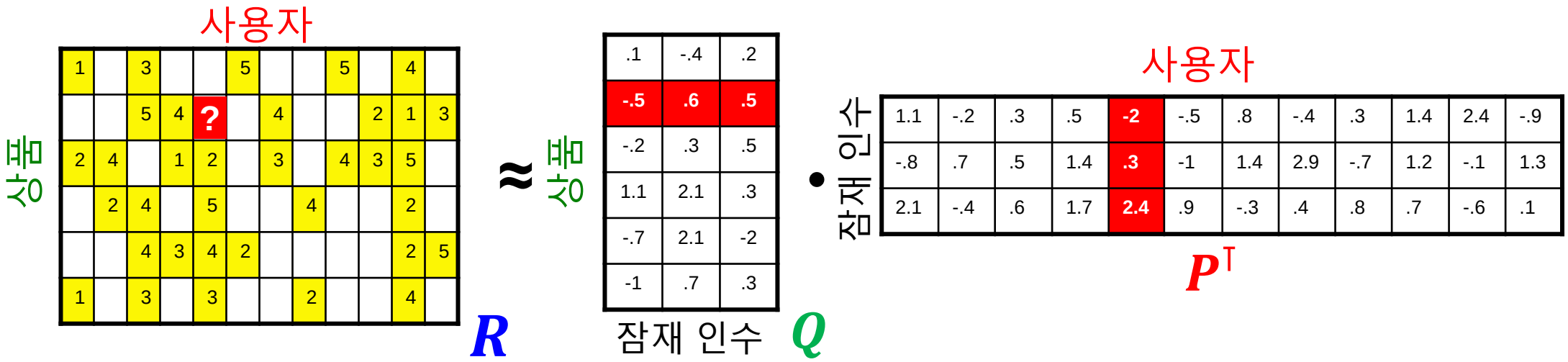
사용자 x의 상품 i에 대한 평점을 r_{xi}라고 합시다

임베딩의 목표는 $p_x^\intercal q_i$이 r_{xi}와 유사하도록 하는 것입니다

3.2 손실 함수

행렬 차원에서 살펴봅시다

사용자 수의 열과 상품 수의 행을 가진 평점 행렬을 R이라고 합시다
사용자들의 임베딩, 즉 벡터를 쌓아서 만든 사용자 행렬을 P라고 합시다
영화들의 임베딩, 즉 벡터를 쌓아서 만든 상품 행렬을 Q라고 합시다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

3.2 손실 함수

잠재 인수 모형은 다음 손실 함수를 최소화하는 P와 Q를 찾는 것을 목표로 합니다

$$\sum_{(i,x)\in R} (r_{xi} - p_x^\intercal q_i)^2$$

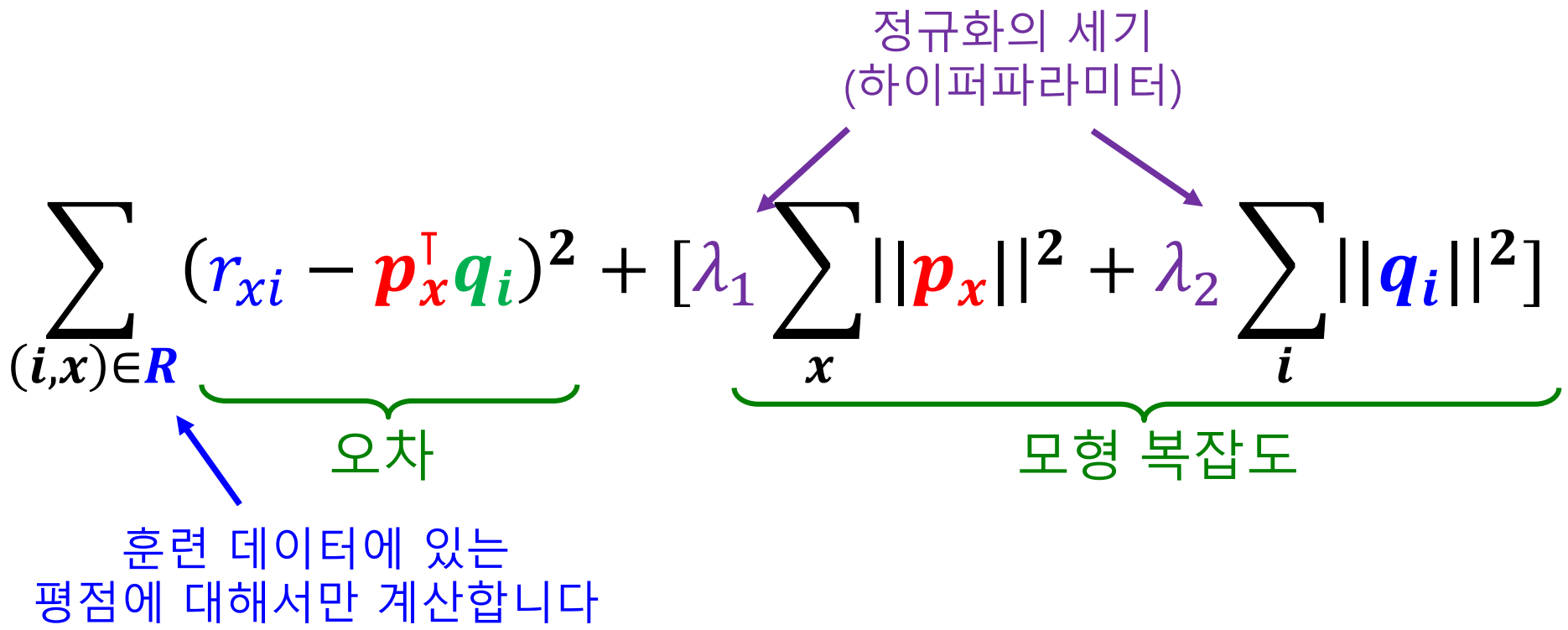
훈련 데이터에 있는
평점에 대해서만 계산합니다

하지만, 위 손실 함수를 사용할 경우 과적합(Overfitting)이 발생할 수 있습니다
과적합이란 기계학습 모형이 훈련 데이터의 잡음(Noise)까지 학습하여,
평가 성능은 오히려 감소하는 현상을 의미합니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

3.2 손실 함수

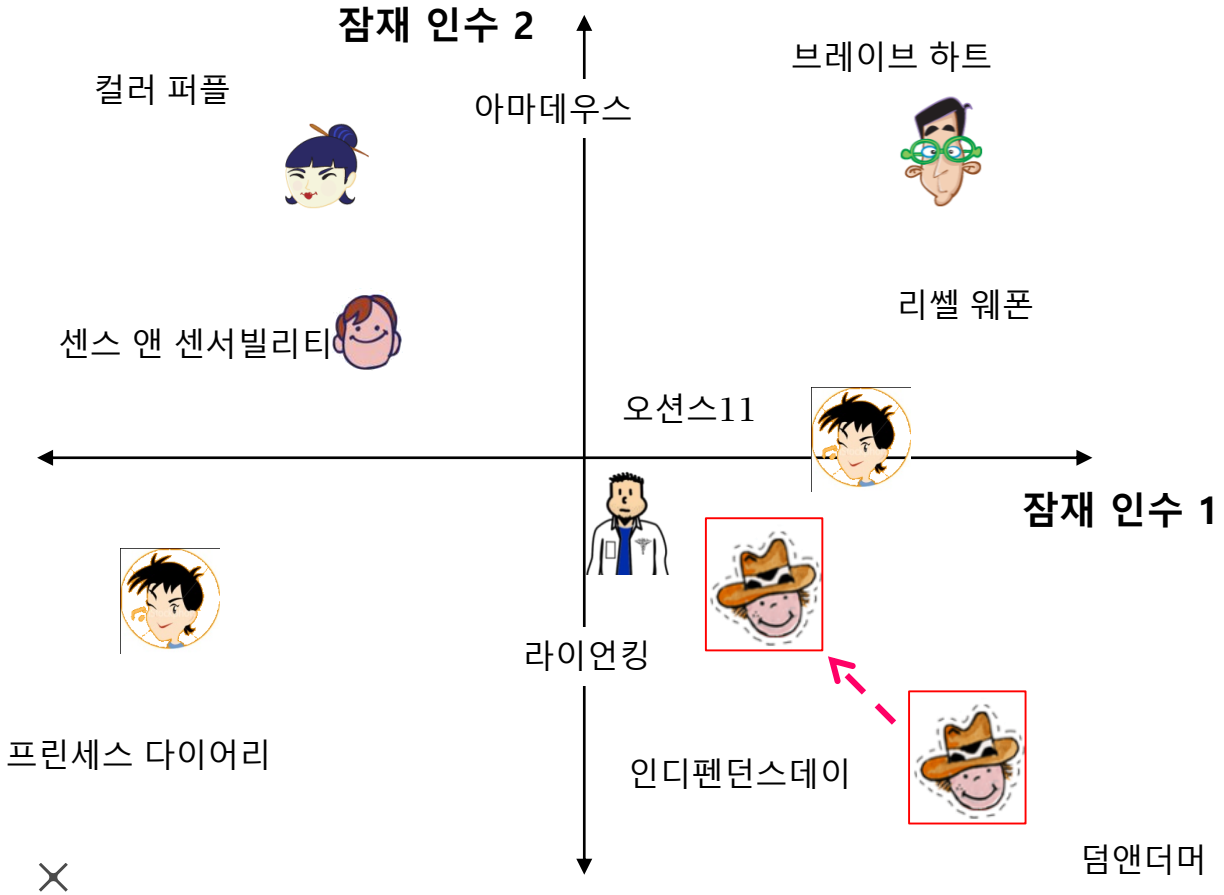
과적합을 방지하기 위하여 정규화 항을 손실 함수에 더해줍니다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys1.pdf

3.3 손실 함수

정규화는 극단적인, 즉 절댓값이 너무 큰 임베딩을 방지하는 효과가 있습니다

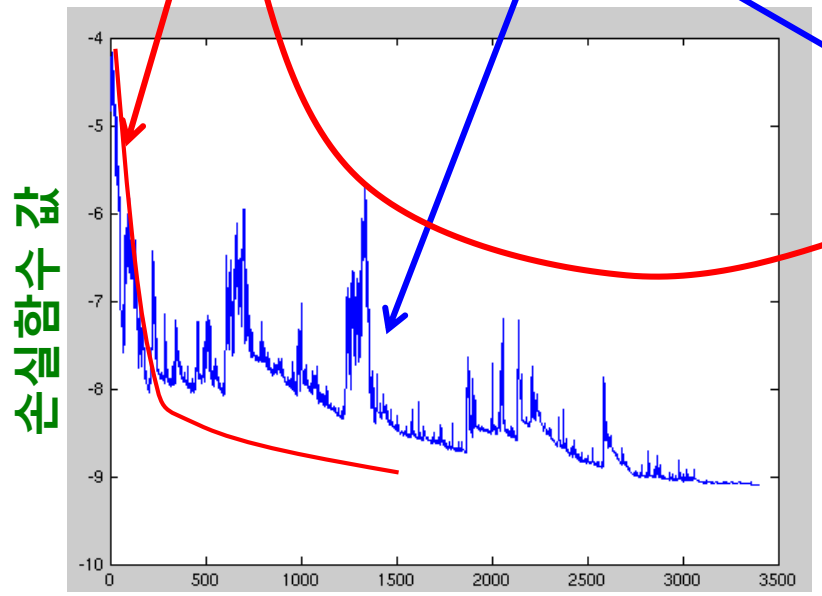


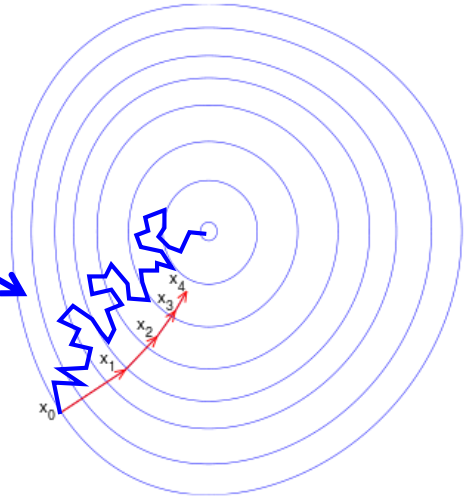
이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

3.3 최적화

손실함수를 최소화하는 P와 Q를 찾기 위해서는 (확률적) 경사하강법을 사용합니다

경사하강법 vs 확률적 경사하강법





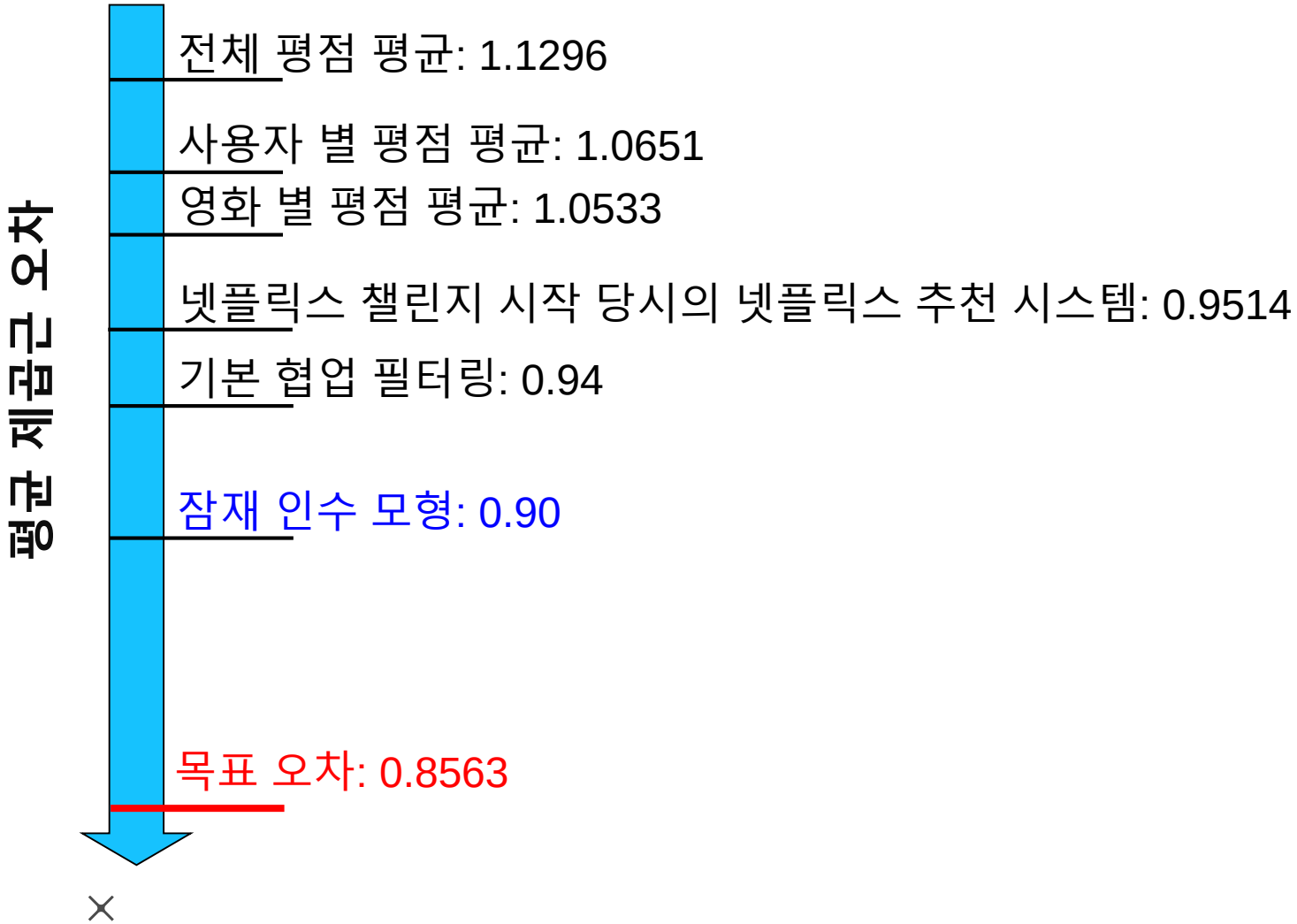
경사하강법은 손실함수를 안정적으로 하지만 느리게 감소시킵니다

확률적 경사하강법은 손실함수를 불안정하지만 빠르게 감소시킵니다

실제로는 확률적 경사하강법이 더 많이 사용됩니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

3.3 최적화



4. 고급 잠재 인수 모형

4.1 사용자와 상품의 편향을 고려한 잠재 인수 모형

4.2 시간에 따른 편향을 고려한 잠재 인수 모형

4.1 사용자와 상품의 편향을 고려한 잠재 인수 모형

각 사용자의 편향은 해당 사용자의 평점 평균과 전체 평점 평균의 차입니다

나연이 매긴 평점의 평균이 4.0개의 별,
다현이 매긴 평점의 평균이 3.5개의 별이라고 합시다

전체 평점 평균이 3.7개의 별인 경우,
나연의 사용자 편향은 4.0 - 3.7 = 0.3개의 별입니다
다현의 사용자 편향은 3.5 - 3.7 = -0.2개의 별입니다

4.1 사용자와 상품의 편향을 고려한 잠재 인수 모형

각 상품의 편향은 해당 상품에 대한 평점 평균과 전체 평점 평균의 차입니다

영화 식스센스에 대한 평점의 평균이 4.5개의 별,
영화 클레멘타인이 매긴 평점의 평균이 3.0개의 별이라고 합시다

전체 평점 평균이 3.7개의 별인 경우,
식스센스의 상품 편향은 4.5 – 3.7 = 0.8개의 별입니다
클레멘타인의 상품 편향은 3.0 - 3.7 = -0.7개의 별입니다

4.1 사용자와 상품의 편향을 고려한 잠재 인수 모형

개선된 잠재 인수 모형에서는 평점을 전체 평균, 사용자 편향, 상품 편향, 상호작용으로 분리합니다

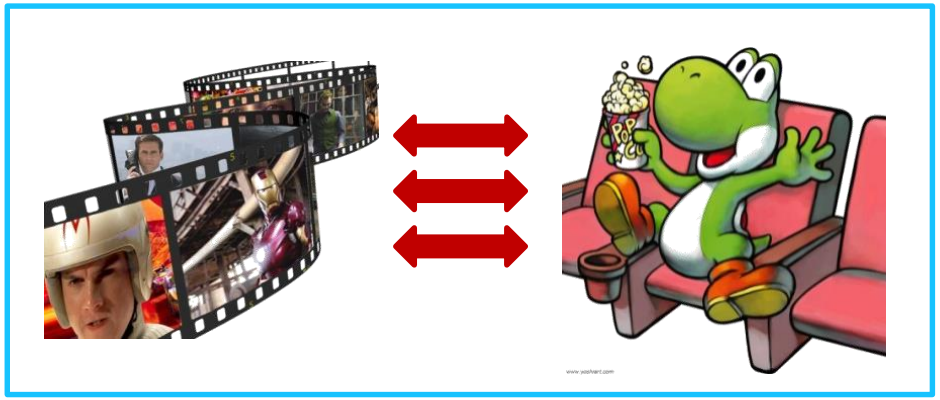
$$r_{xi} = \mu + b_x + b_i + p_x^T q_i$$

평점 전체 평균 사용자 편향 상품 편향 상호작용

사용자 편향

상품 편향

사용자-상품 상호작용



×

boostcamp AI Tech

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

4.1 사용자와 상품의 편향을 고려한 잠재 인수 모형

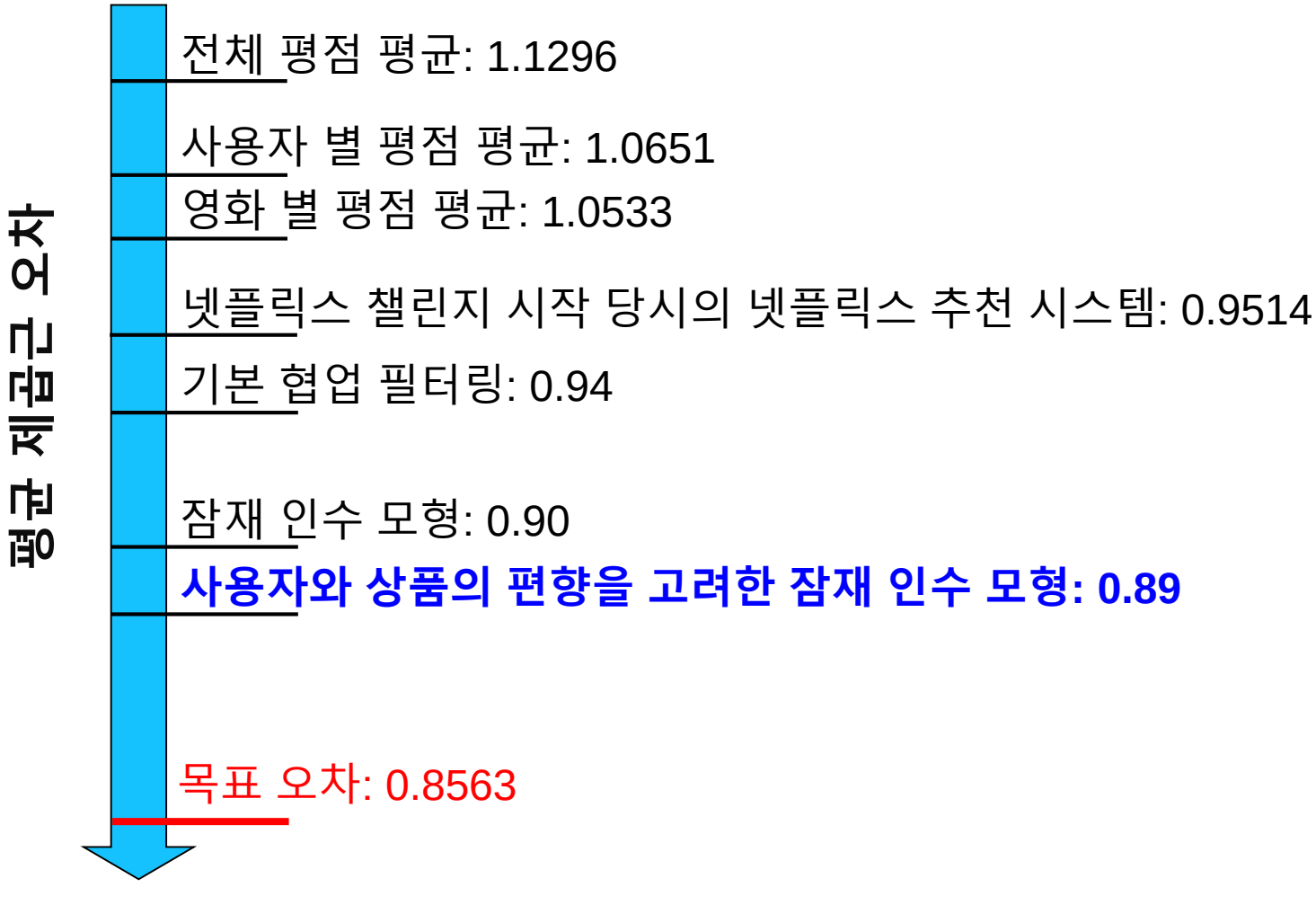
개선된 잠재 인수 모형의 손실 함수는 아래와 같습니다

$$\sum_{(i,x)\in R}(r_{xi}-(\mu+b_x+b_i+p_x^\intercal q_i)^2$$

$$+[\lambda_1\sum_x||p_x||^2+\lambda_2\sum_i||q_i||^2+\lambda_3\sum_x b_x^2+\lambda_4\sum_i b_i^2]$$

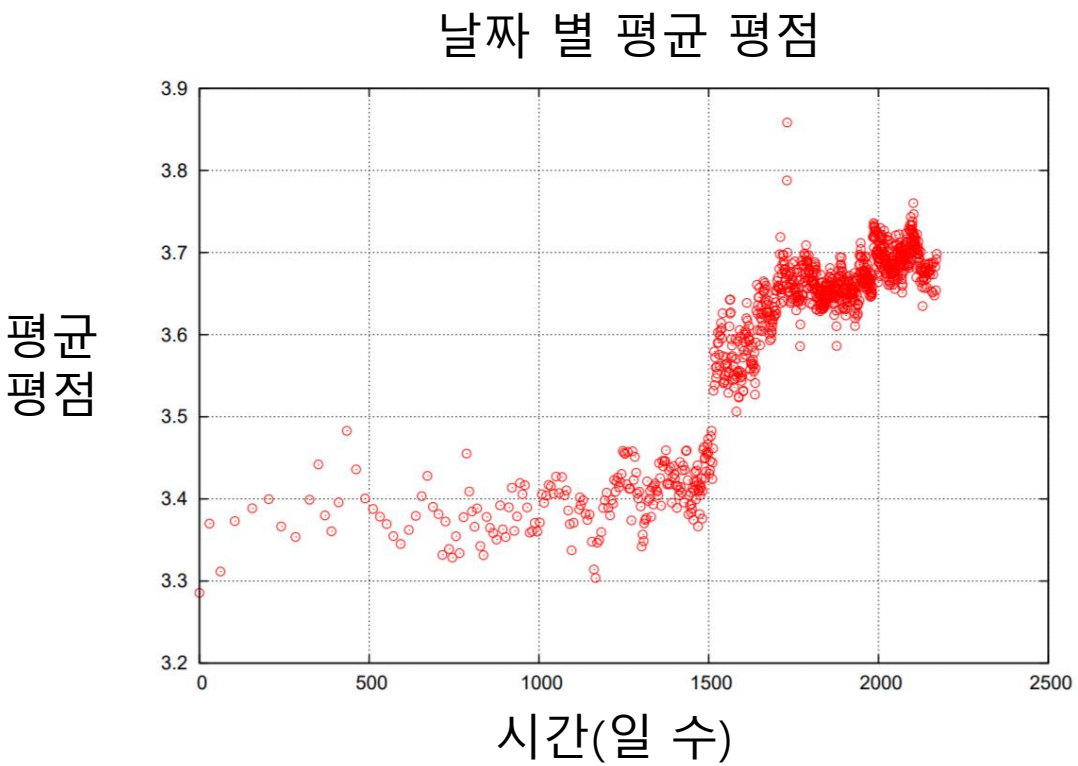
(확률적) 경사하강법을 통해 손실 함수를 최소화하는 잠재 인수와 편향을 찾아냅니다

4.1 사용자와 상품의 편향을 고려한 잠재 인수 모형



4.2 시간적 편향을 고려한 잠재 인수 모형

넷플릭스 시스템의 변화로 평균 평점이 크게 상승하는 사건이 있었습니다

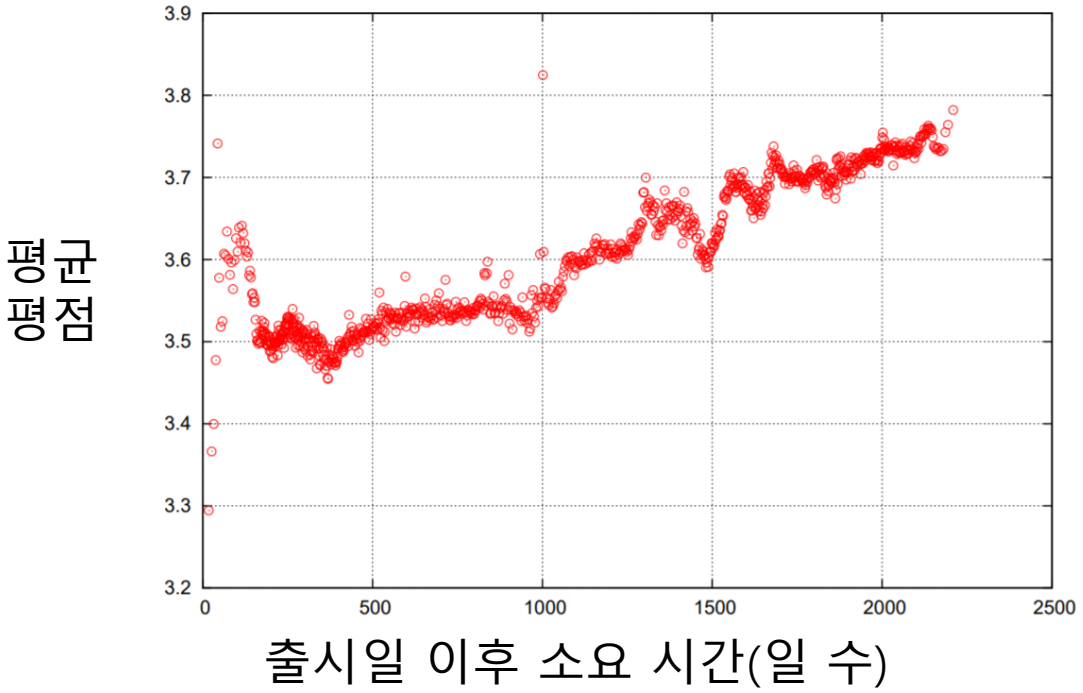


이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

4.2 시간적 편향을 고려한 잠재 인수 모형

영화의 평점은 출시일 이후 시간이 지남에 따라 상승하는 경향을 갖습니다

영화 출시일 이후 소요 시간에 따른 평점 변화



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

4.2 시간적 편향을 고려한 잠재 인수 모형

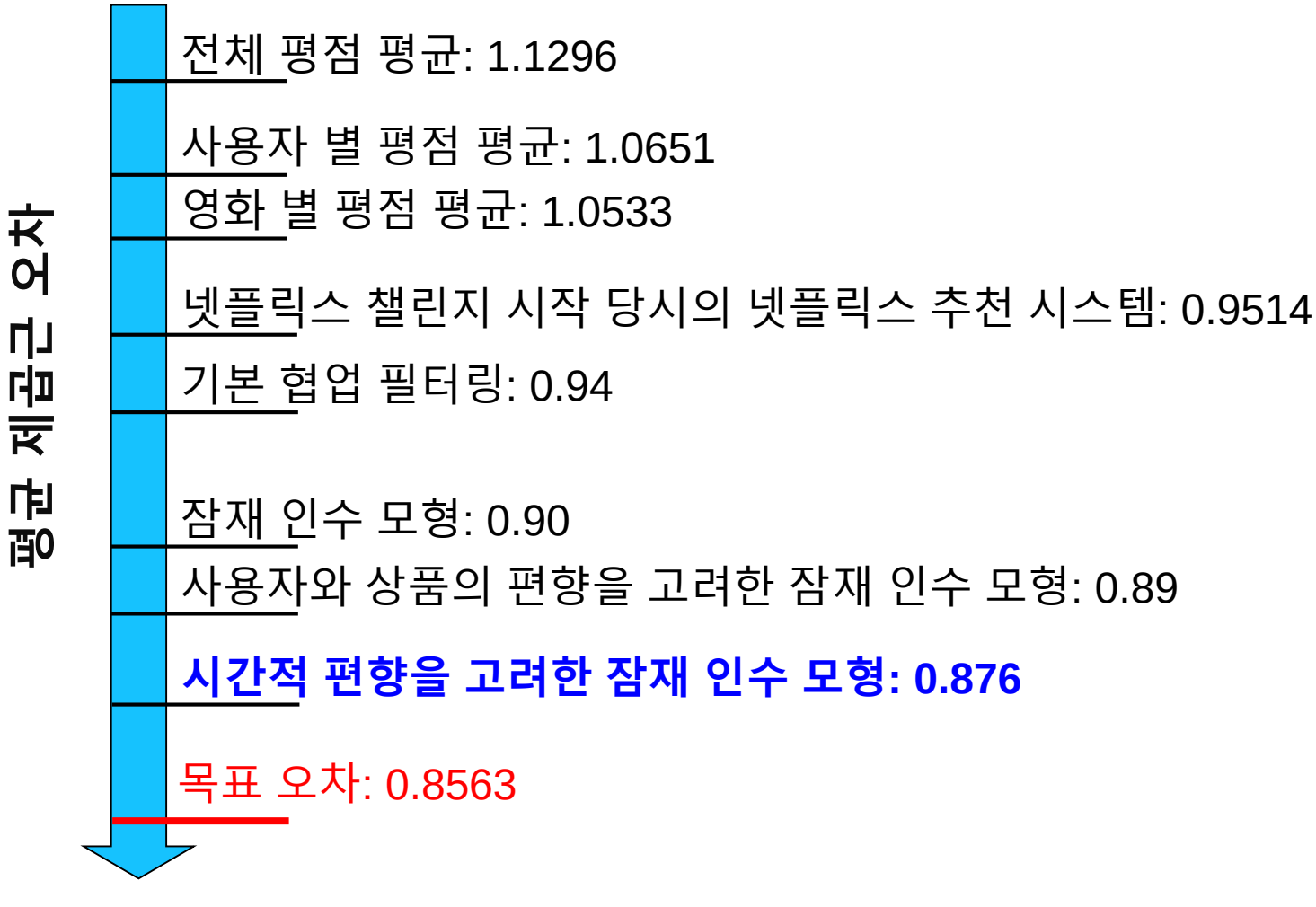
개선된 잠재 인수 모형에서는 이러한 시간적 편향을 고려합니다

구체적으로 사용자 편향과 상품 편향을 시간에 따른 함수로 가정합니다

$$r_{xi} = \mu + b_x(t) + b_i(t) + p_x^T q_i$$

평점 전체 평균 사용자 편향 상품 편향 상호작용

4.2 시간적 편향을 고려한 잠재 인수 모형



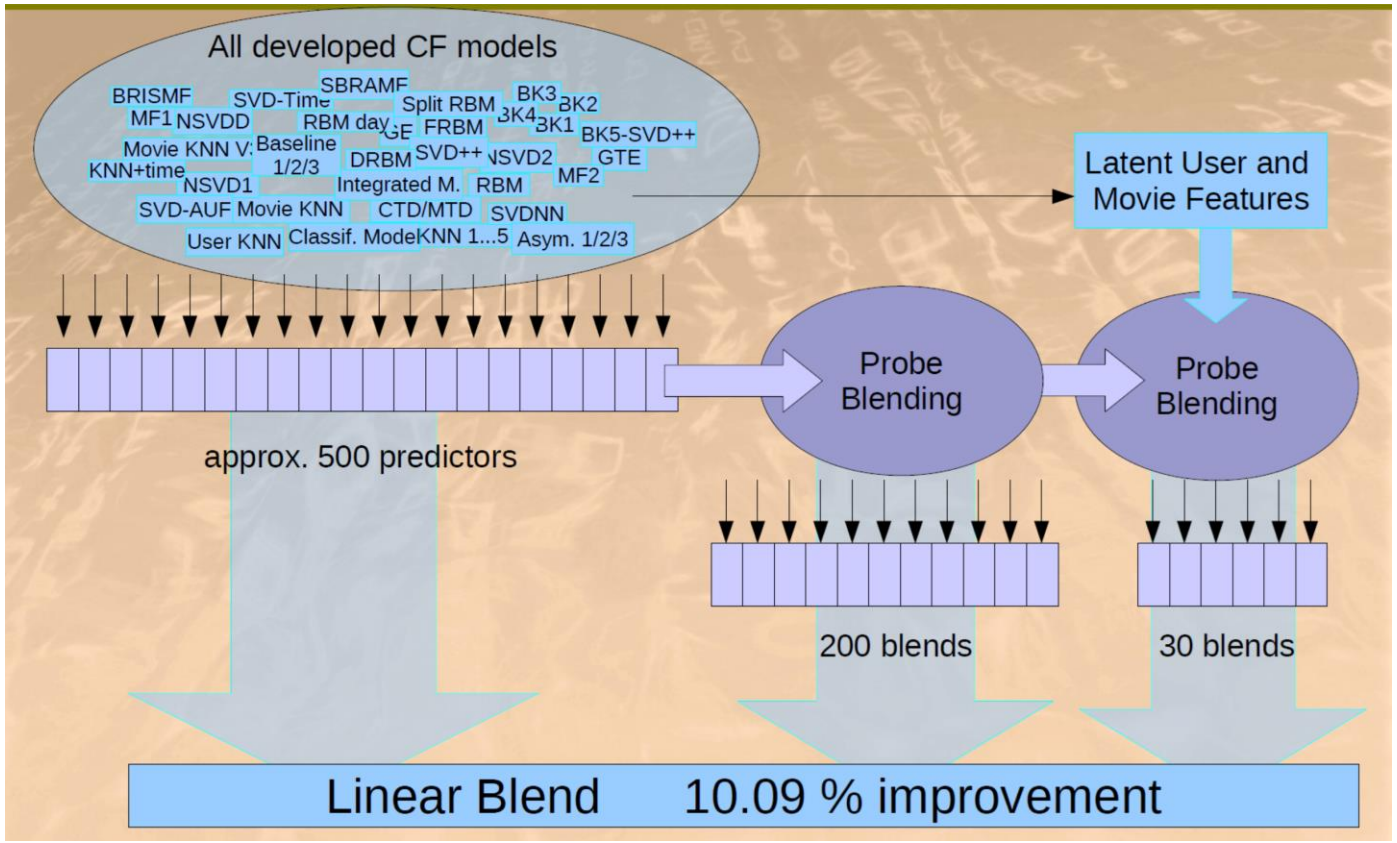
5. 넷플릭스 챌린지의 결과

5.1 앙상블 학습

5.2 넷플릭스 챌린지의 우승팀

5.1 앙상블 학습

BellKor 팀은 앙상블 학습을 사용하여 처음으로 목표 성능에 도달하였습니다



이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

5.1 앙상블 학습

BellKor 팀의 독주에 위기감을 느낀 다른 팀들은 연합팀 Ensemble을 만들었습니다

Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	BellKor's Pragmatic Chaos	0.8558	10.05	2009-06-26 18:42:37
Grand Prize - RMSE <= 0.8563				
2	PragmaticTheory	0.8582	9.80	2009-06-25 22:15:51
3	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
4	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
5	Dace	0.8604	9.56	2009-04-22 05:57:03
6	BigChaos	0.8613	9.47	2009-06-23 23:06:52
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				

그 결과 Ensemble 팀 역시 목표 성능에 도달하였습니다

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

5.2. 넷플릭스 챌린지의 우승팀

넷플릭스 챌린지 종료 시점에 BellKor 팀 Ensemble 팀의 오차는 정확히 동일했습니다
하지만 BellKor 팀의 제출이 20분 빨랐습니다. BellKor 팀의 우승입니다!

Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09

이미지 출처: http://www.mmds.org/mmds/v2.1/ch09-recsys2.pdf

6. 실습: Surprise 라이브러리와 잠재 인수 모형의 활용

6.1 데이터 불러오기 및 전처리

6.2 잠재 인수 모형 학습

6.3 점수 추정

6.4 영화 추천

6.1 데이터 불러오기 및 전처리

먼저 실습에서 사용하는 파이썬 라이브러리를 불러옵니다

```
import numpy as np
import pandas as pd
from surprise import SVD
from surprise.model_selection import train_test_split
from surprise.model_selection import cross_validate
from surprise import Dataset, Reader
from surprise import accuracy
from surprise.model_selection import GridSearchCV
```

5.1 데이터 불러오기 및 전처리

본 실습에서는 100,000개의 평점으로 구성된 MovieLens 데이터셋을 사용합니다

```
### Rating Dataset Format ###

   userId  movieId  rating  timestamp
0       1        1     4.0  964982703
1       1        3     4.0  964981247
2       1        6     4.0  964982224
3       1       47     5.0  964983815
4       1       50     5.0  964982931
```

```
### Rating Dataset - Timestamp Removed ###

        userId  movieId  rating
0            1        1     4.0
1            1        3     4.0
2            1        6     4.0
3            1       47     5.0
4            1       50     5.0
...        ...      ...     ...
100831     610   166534     4.0
100832     610   168248     5.0
100833     610   168250     5.0
100834     610   168252     5.0
100835     610   170875     3.0

[100836 rows x 3 columns]
```

6.1 데이터 불러오기 및 전처리

데이터를 파일에서 읽어옵니다

```
df_ratings = pd.read_csv('./ratings.csv')
```

6.1 데이터 불러오기 및 전처리

데이터에 포함된 사용자 수와 영화 수를 확인합니다

```
n_users = df_ratings.userId.unique().shape[0]
n_items = df_ratings.movieId.unique().shape[0]
print("num users: {}, num items:{}".format(n_users, n_items))
```

```
num users: 611, num items:9725
```

6.1 데이터 불러오기 및 전처리

훈련 데이터와 평가 데이터를 분리합니다

```
reader = Reader(rating_scale=(0, 5))
data = Dataset.load_from_df(df_ratings[['userId','movieId','rating']], reader=reader)
train, test = train_test_split(data, test_size = 0.2, shuffle = True)
```

6.2 잠재 인수 모형 학습

하이퍼파라미터를 탐색한 뒤, 잠재 인수 모형을 학습합니다

```
param_grid = {'n_factors': [50,100,150,200]}
grid = GridSearchCV(SVD, param_grid, measures = ['rmse'], cv=4)
grid.fit(train)
algorithm = SVD(grid.best_params['rmse']['n_factors'])
algorithm.fit(train)
```

6.3 점수 추정

학습된 잠재 인수 모형을 활용하여 평점을 추정합니다

```
uid = 800
iid = 8368
prediction_user_item = algorithm.predict(uid, iid)
```

6.4 영화 추천

시청하지 않은 영화 중에 추정 평점이 높은 것들을 추정 평점 역순으로 추천합니다

```
unseen_movies = get_unseen_movies(train, user_id)
prediction = [algorithm.predict(user_id, movie_id) for movie_id in unseen_movies]

prediction.sort(key=lambda x:x.est, reverse=True)
for _, movie, _, pred, _ in prediction[:top_k]:
    print("movid id: {}, movie genre: {},predicted rating: {}".format(
        movie_id_to_name[movie], movie_id_to_genre[movie], pred))
```

8강 정리

1. 추천시스템 기본 복습
- 내용 기반 추천, 협업 필터링 등

2. 넷플릭스 챌린지 소개

3. 기본 잠재 인수 모형
- 사용자 임베딩, 상품 임베딩을 학습
- 임베딩의 내적으로 평점을 근사

4. 고급 잠재 인수 모형
- 사용자 편향, 상품 편향, 시간적 편향을 고려

5. 넷플릭스 챌린지의 결과

6. 실습: Surprise 라이브러리와 잠재 인수 모형의 활용