Matrix Factorization을 이용한 추천 시스템 알고리즘 구현

212STG13 박지원

212STG18 예지혜

218STG01 김지민

목차

1. 서론

- 주제 소개
- 문제 정의
- 데이터 설명

주제 소개

이미지 출처 : http://www.enuri.com/knowcom/detail.jsp?kbno=952175

주제 소개

Netflix Prize Competition

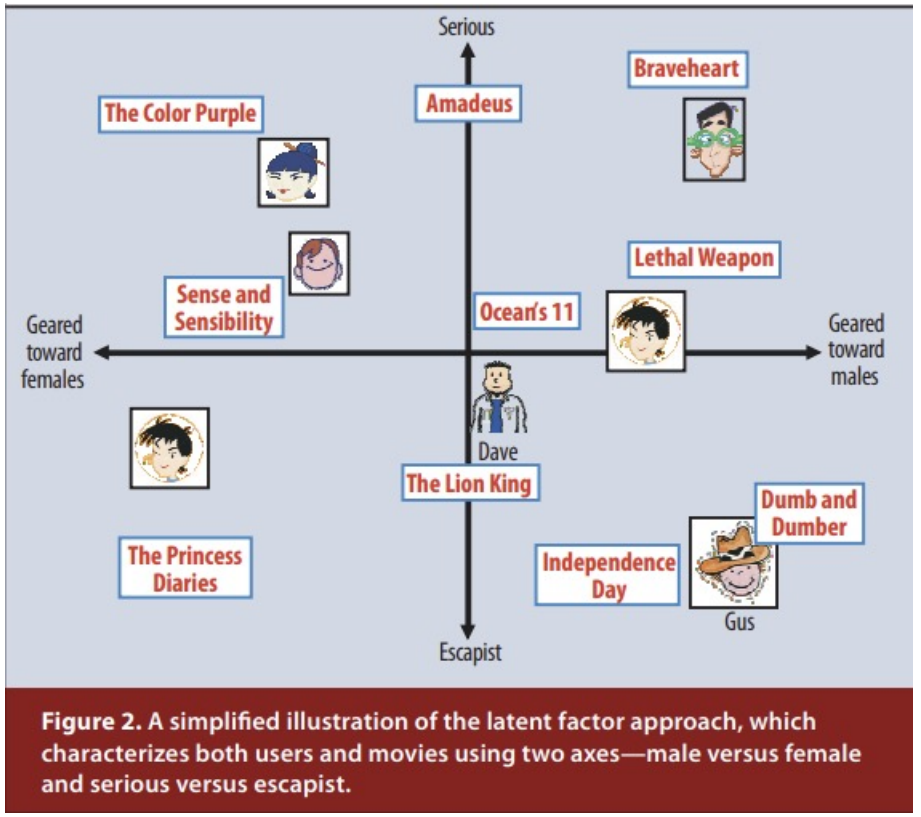


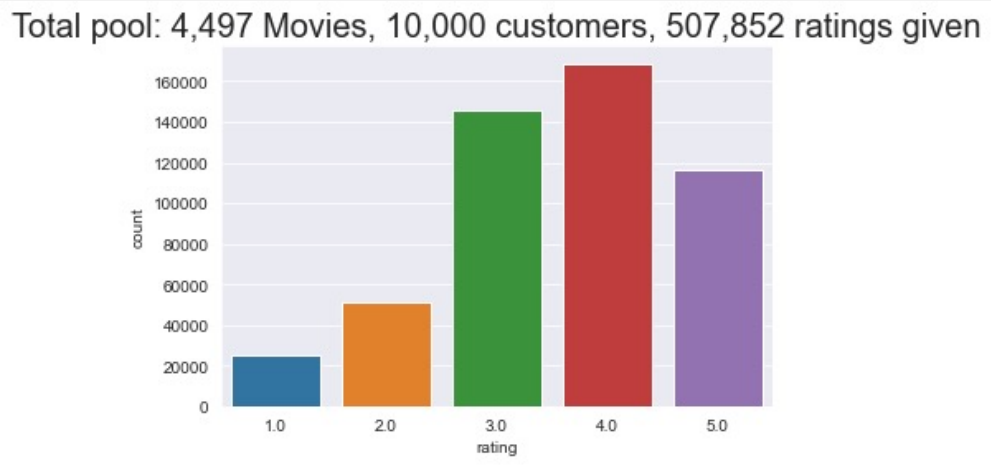
Figure 2. A simplified illustration of the latent factor approach, which characterizes both users and movies using two axes—male versus female and serious versus escapist.

Latent factor model

데이터 설명

Netflix 영화 평점 데이터

- 넷플릭스 데이터
- 출처 : kaggle
- 사용자 10만 명으로 잘라 4497개 영화에 대한 50만개의 평점 데이터 사용
- 성능 지표 : RMSE (train 75%, test 25%)

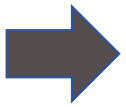


	user_id	movie_id	rating
0	2442	1	3.0
1	1719610	1	2.0
2	1011918	1	4.0
3	479924	1	5.0
4	2389367	1	1.0
5	563962	1	5.0
6	369646	1	5.0
7	1430587	1	4.0
8	305344	1	1.0
9	389872	1	2.0

데이터 설명

Netflix 영화 평점 데이터

	user_id	movie_id	rating
0	2442	1	3.0
1	1719610	1	2.0
2	1011918	1	4.0
3	479924	1	5.0
4	2389367	1	1.0
5	563962	1	5.0
6	369646	1	5.0
7	1430587	1	4.0
8	305344	1	1.0
9	389872	1	2.0



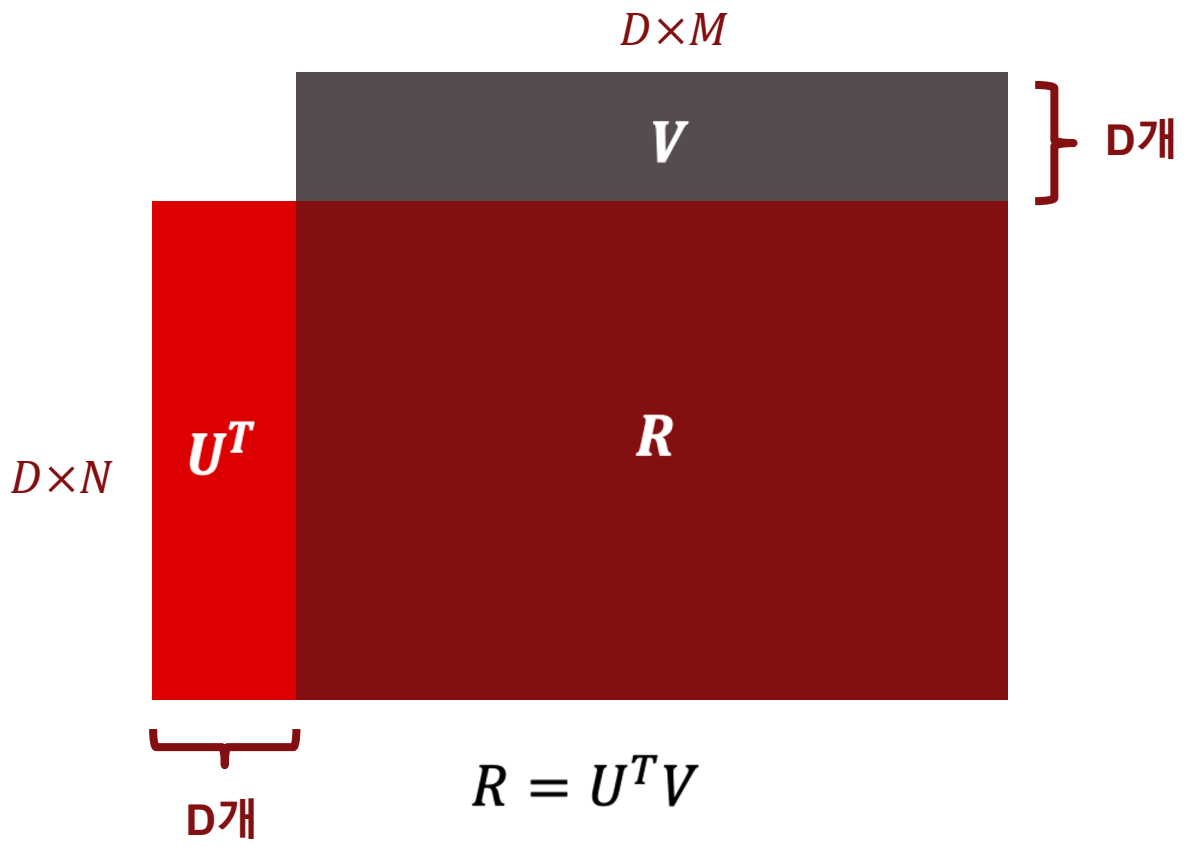
movie_id	1	2	3	4	5	6	7	8	9	10	...	4490	4491	4492	4493	4494	4495	4496	4497	4498	4499
user_id																					
97	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
201	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
379	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
781	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
933	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2648312	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2648719	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2648927	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649110	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649328	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

(number of users) X (number of movies)

Matrix Factorization

movie_id	1	2	3	4	5	6	7	8	9	10	...	4490	4491	4492	4493	4494	4495	4496	4497	4498	4499
user_id																					
97	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
201	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
379	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
781	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
933	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2648312	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2648719	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2648927	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649110	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649328	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

$N \times M$



추천 시스템 알고리즘 소개

Basic Matrix Factorization

- 행렬 분해
- 분포 가정 없음
- SGD, ALS 등으로 최적화

Probabilistic Matrix Factorization

- U와 V에 가우시안 사전 분포 가정
- MAP 방법으로 최적화

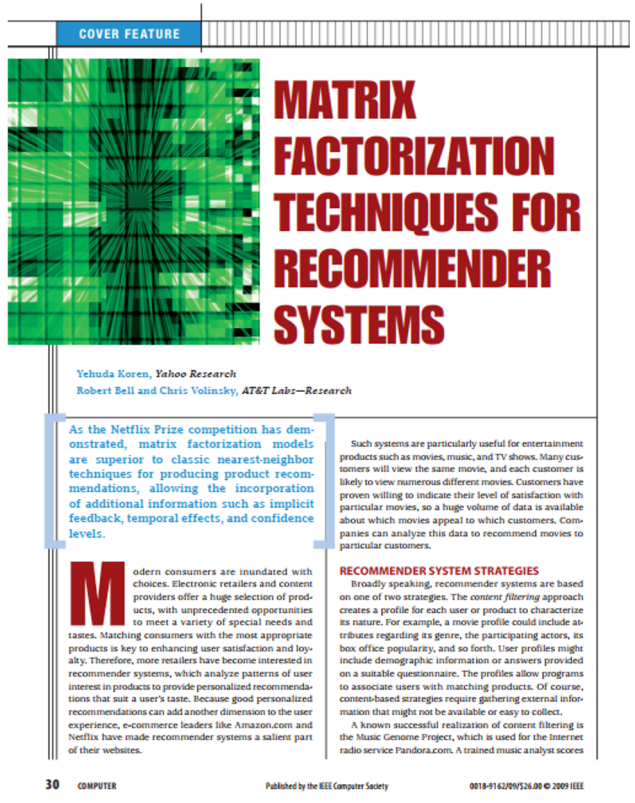
Bayesian PMF

- 사전분포의 하이퍼파라미터에 대해서도 분포 가정
- MCMC 방법을 이용한 추론

2. 알고리즘 구현

- Basic Matrix Factorization
- Probabilistic Matrix Factorization
- Bayesian Probabilistic Matrix Factorization

Basic Matrix Factorization



COVER FEATURE

MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS

Yehuda Koren, *Yahoo Research*
Robert Bell and Chris Volinsky, *AT&T Labs—Research*

As the Netflix Prize competition has demonstrated, matrix factorization models are superior to classic nearest-neighbor techniques for producing product recommendations, allowing the incorporation of additional information such as implicit feedback, temporal effects, and confidence levels.

Modern consumers are inundated with choices. Electronic retailers and content providers offer a huge selection of products, with unprecedented opportunities to meet a variety of special needs and tastes. Matching consumers with the most appropriate products is key to enhancing user satisfaction and loyalty. Therefore, more retailers have become interested in recommender systems, which analyze patterns of user interest in products to provide personalized recommendations that suit a user's taste. Because good personalized recommendations can add another dimension to the user experience, e-commerce leaders like Amazon.com and Netflix have made recommender systems a salient part of their websites.

Such systems are particularly useful for entertainment products such as movies, music, and TV shows. Many customers will view the same movie, and each customer is likely to view numerous different movies. Customers have proven willing to indicate their level of satisfaction with particular movies, so a huge volume of data is available about which movies appeal to which customers. Companies can analyze this data to recommend movies to particular customers.

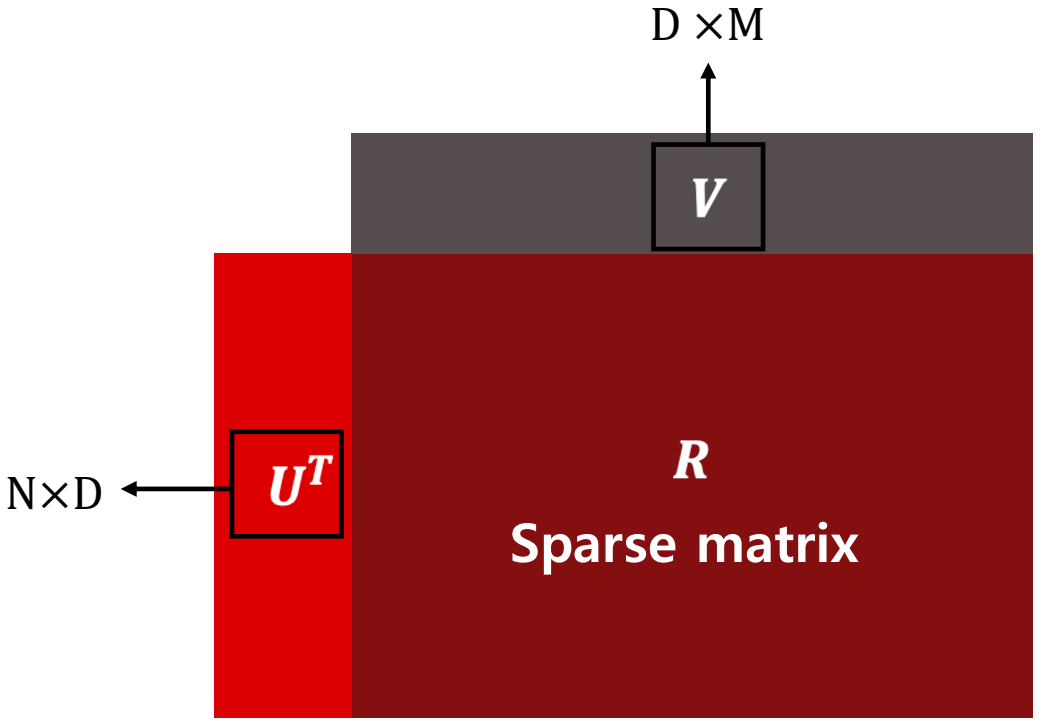
RECOMMENDER SYSTEM STRATEGIES

Broadly speaking, recommender systems are based on one of two strategies. The *content filtering* approach creates a profile for each user or product to characterize its nature. For example, a movie profile could include attributes regarding its genre, the participating actors, its box office popularity, and so forth. User profiles might include demographic information or answers provided on a suitable questionnaire. The profiles allow programs to associate users with matching products. Of course, content-based strategies require gathering external information that might not be available or easy to collect.

A known successful realization of content filtering is the Music Genome Project, which is used for the Internet radio service Pandora.com. A trained music analyst scores

Published by the IEEE Computer Society 0018-9162/09/$26.00 © 2009 IEEE

Y. Koren, R. Bell, and C. Volinsky, **Matrix factorization techniques for recommender systems**, *IEEE Computing*, 2009.



- 전통적인 SVD 적용 불가
- Missing imputation시 계산량 매우 증가, 데이터 왜곡

➔ 알고 있는 rating에 대해서만 학습

Basic Matrix Factorization

target function

$$minimize \sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}[(R_{ij} - U_i^T V_j)^2 + \lambda(||U_i||^2 + ||V_j||^2)]$$

별점이 존재하는 경우만 학습

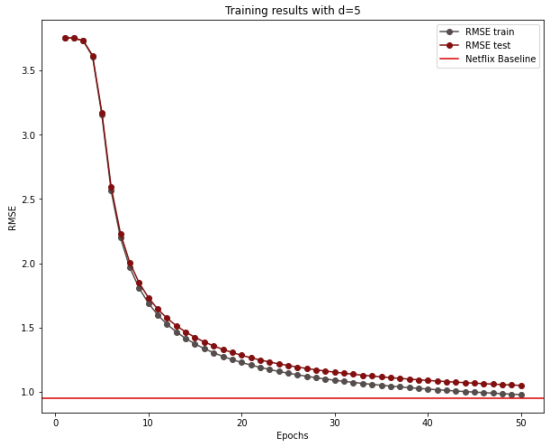
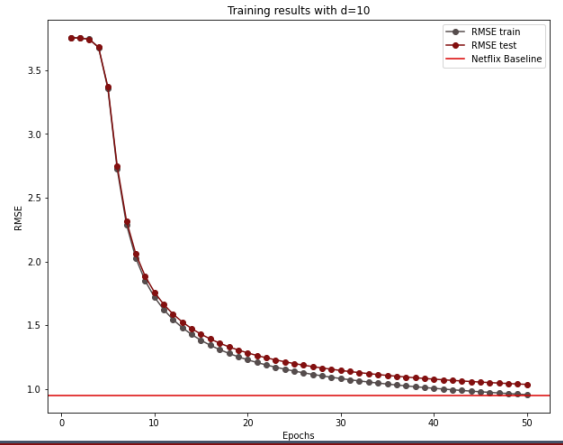
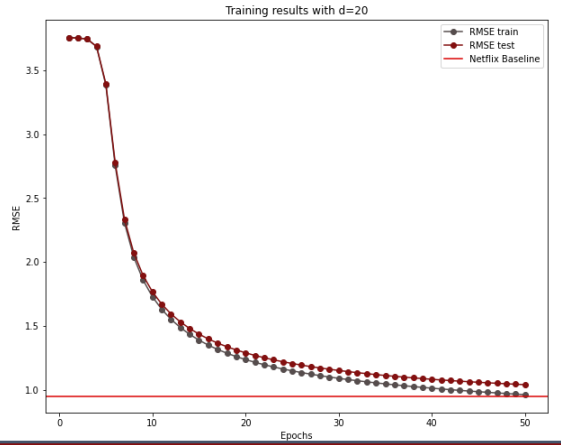
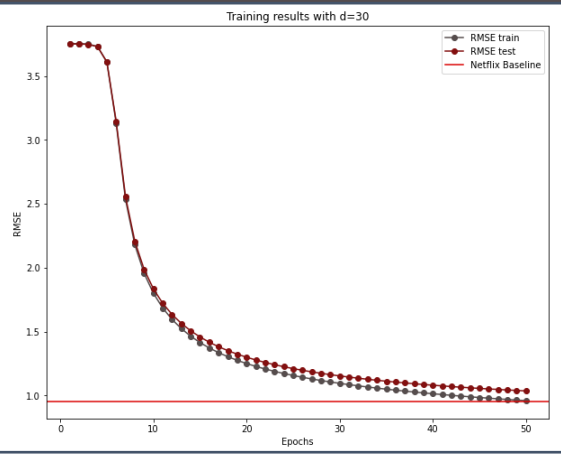
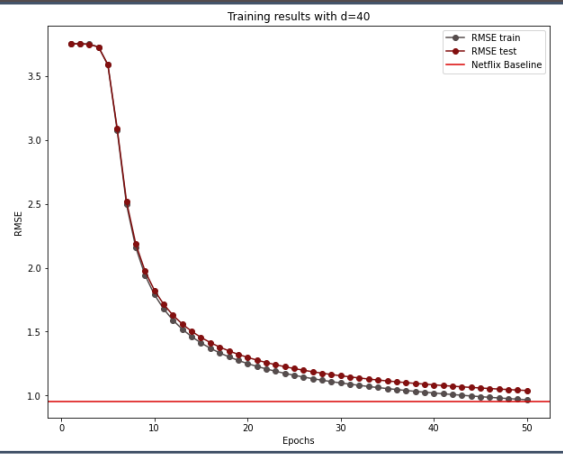
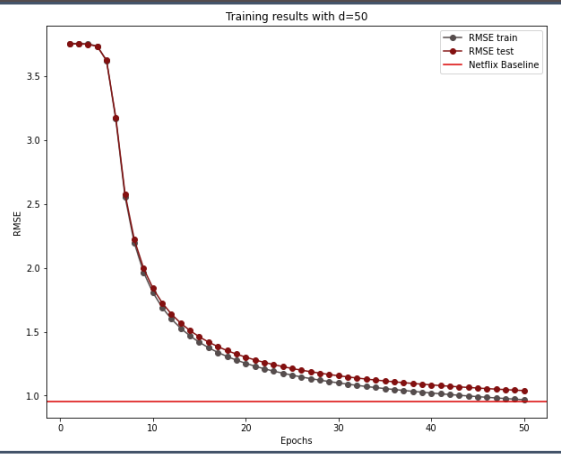
Optimization - SGD

$$e_{ij} = R_{ij} - U_i^T V_j$$

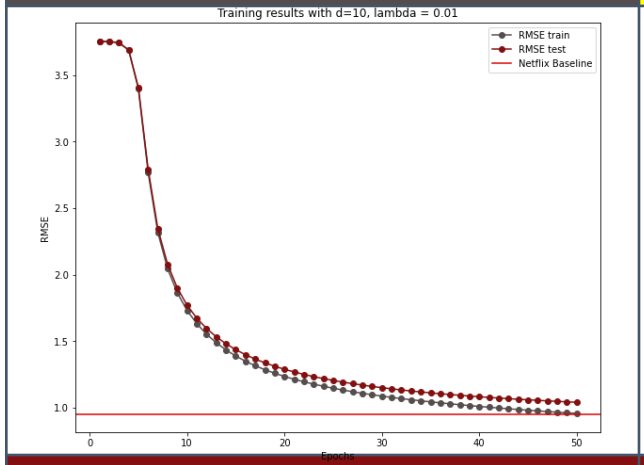
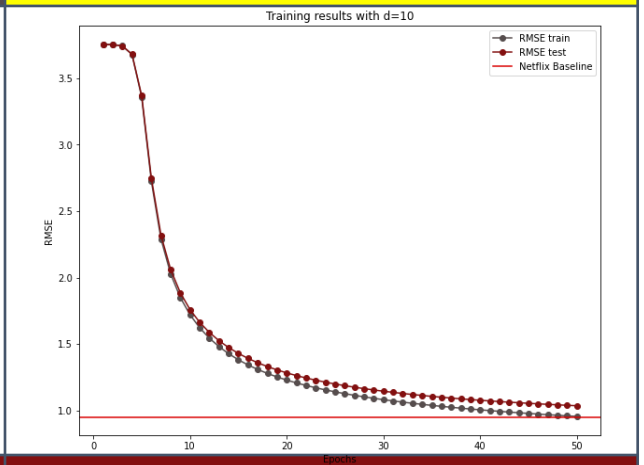
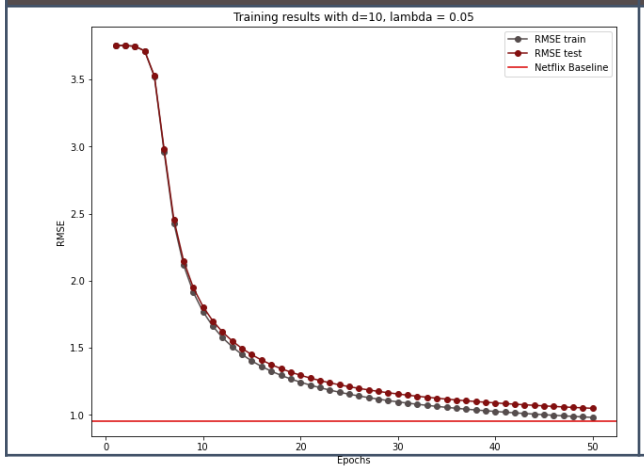
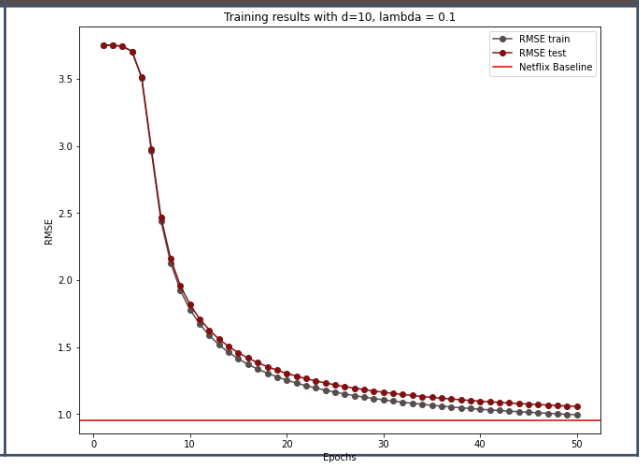
$$U_i \leftarrow U_i + \alpha(e_{ij} V_j - \lambda U_i)$$

$$V_j \leftarrow V_j + \alpha(e_{ij} U_i - \lambda V_j)$$

Basic Matrix Factorization

D = 5	D = 10	D = 20
Test RMSE : 1.0518	Test RMSE : 1.0366	Test RMSE : 1.0403
		
D = 30	**D = 40**	**D = 50**
Test RMSE : 1.0358	Test RMSE : 1.0391	Test RMSE : 1.0391
		

Basic Matrix Factorization

$\lambda = 0.01$	$\lambda = 0.02$
Test RMSE : 1.0408	Test RMSE : 1.0366
	
$\lambda = 0.05$	$\lambda = 0.1$
Test RMSE : 1.0499	Test RMSE : 1.0579
	

Basic Matrix Factorization – bias 추가

user bias

$$b_{ij} = \mu + b_i + b_j$$

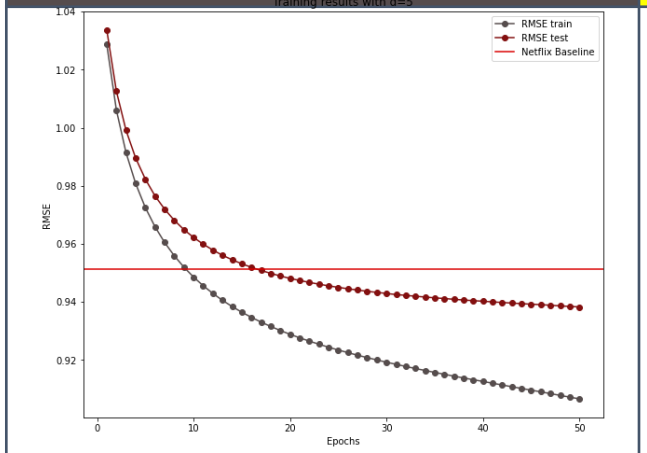
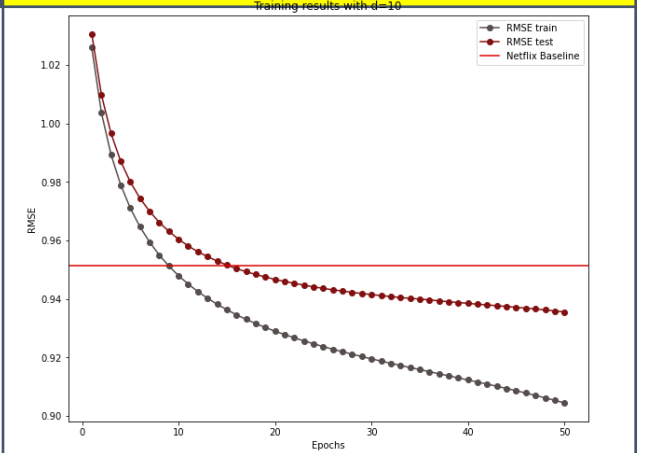
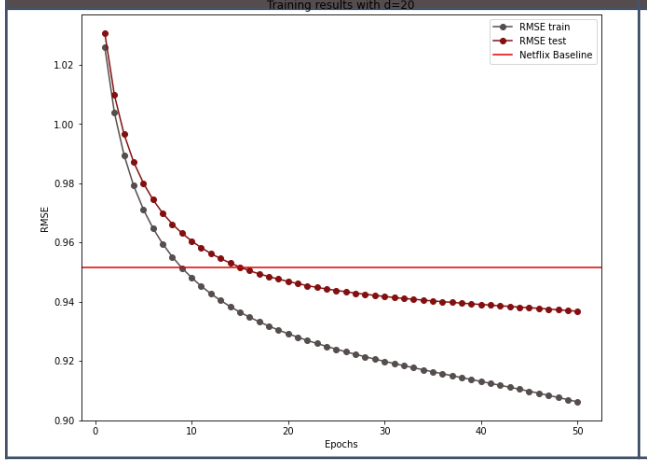
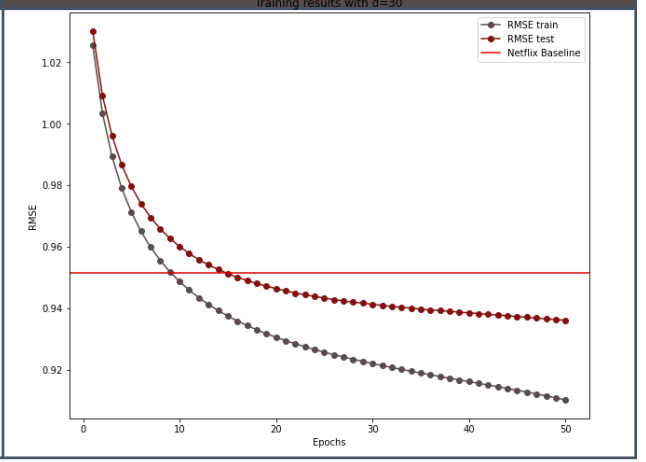
movie bias

$$\widehat{R_{ij}} = \mu + b_i + b_j + U_i^T V_j$$

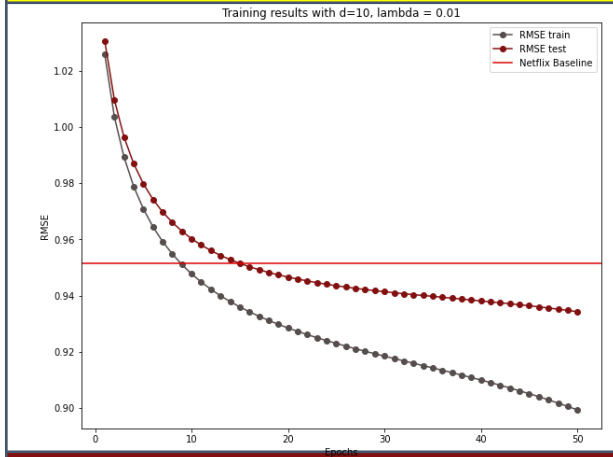
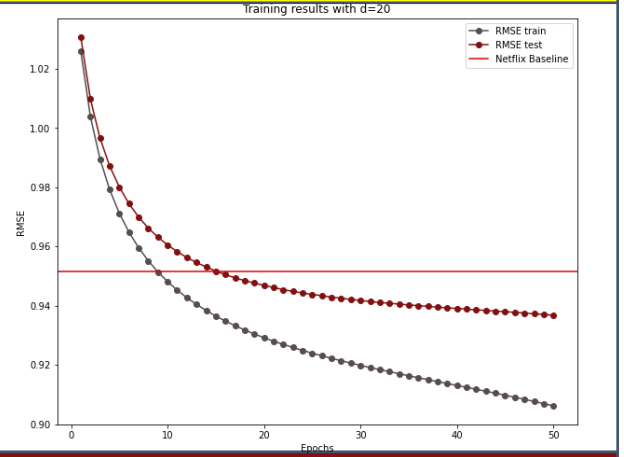
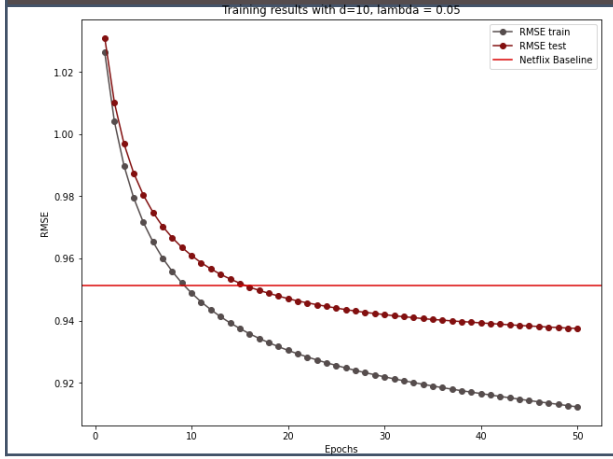
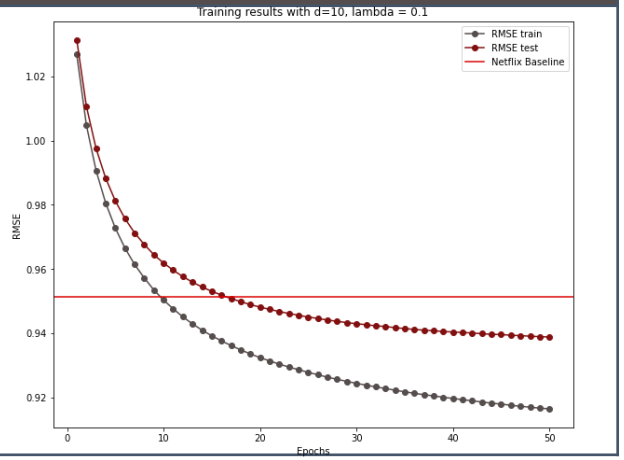
target function

$$minimize \sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}[(R_{ij} - \mu - b_i - b_j - U_i^T V_j)^2 + \lambda(||U_i||^2 + ||V_j||^2 + b_i^2 + b_j^2)]$$

Basic Matrix Factorization – bias 추가

D = 5	D = 10
Test RMSE : 0.9383	Test RMSE : 0.9356
	
D = 20	**D = 30**
Test RMSE : 0.9368	Test RMSE : 0.9360
	

Basic Matrix Factorization – bias 추가

λ = 0.01	λ = 0.02
Test RMSE : 0.9343	Test RMSE : 0.9368
	
λ = 0.05	λ = 0.1
Test RMSE : 0.9375	Test RMSE : 0.9389
	

Basic Matrix Factorization

User A의 취향

	movie_id	title	rating	rating_pred
18	1084	Walking with Prehistoric Beasts	5.0	3.661165
73	2862	The Silence of the Lambs	5.0	3.529433
23	1391	Yanni: Live at the Acropolis	5.0	3.434659
78	3113	Dante's Peak	5.0	3.478117
79	3184	Desert Hearts	5.0	3.413284
26	1470	Bend It Like Beckham	5.0	3.458431
27	1482	Beyond Borders	5.0	3.457147
92	3671	Laughing Matters	5.0	3.394318
57	2452	Lord of the Rings: The Fellowship of the Ring	5.0	3.936750
31	1709	Clash of the Titans	5.0	3.393939

User A가 좋아할만한 영화는?

	movie_id	title	rating	rating_pred
3363	3453	The Man Who Knew Too Much	NaN	4.620457
4307	4424	Elton John: Live in Barcelona	NaN	4.511962
1906	1945	From Hell: Bonus Material	NaN	4.477076
2055	2100	Bliss	NaN	4.474524
2484	2546	The Second Coming	NaN	4.423051
1446	1474	Classic Country Comedy	NaN	4.414985
4236	4350	The Best of the New Scooby-Doo Movies	NaN	4.394918
2965	3044	The Inheritance	NaN	4.360222
2392	2450	Inserts	NaN	4.359082
3351	3441	Kicking & Screaming	NaN	4.353238

Probabilistic Matrix Factorization

Probabilistic Matrix Factorization

Ruslan Salakhutdinov and Andriy Mnih
Department of Computer Science, University of Toronto
6 King's College Rd, M5S 3G4, Canada
{rsalakhu,amnih}@cs.toronto.edu

Abstract

Many existing approaches to collaborative filtering can neither handle very large datasets nor easily deal with users who have very few ratings. In this paper we present the Probabilistic Matrix Factorization (PMF) model which scales linearly with the number of observations and, more importantly, performs well on the large, sparse, and very imbalanced Netflix dataset. We further extend the PMF model to include an adaptive prior on the model parameters and show how the model capacity can be controlled automatically. Finally, we introduce a constrained version of the PMF model that is based on the assumption that users who have rated similar sets of movies are likely to have similar preferences. The resulting model is able to generalize considerably better for users with very few ratings. When the predictions of multiple PMF models are linearly combined with the predictions of Restricted Boltzmann Machines models, we achieve an error rate of 0.8861, that is nearly 7% better than the score of Netflix's own system.

1 Introduction

One of the most popular approaches to collaborative filtering is based on low-dimensional factor models. The idea behind such models is that attitudes or preferences of a user are determined by a small number of unobserved factors. In a linear factor model, a user's preferences are modeled by linearly combining item factor vectors using user-specific coefficients. For example, for N users and M movies, the $N \times M$ preference matrix R is given by the product of an $N \times D$ user coefficient matrix U^T and a $D \times M$ factor matrix V [7]. Training such a model amounts to finding the best rank-D approximation to the observed $N \times M$ target matrix R under the given loss function.

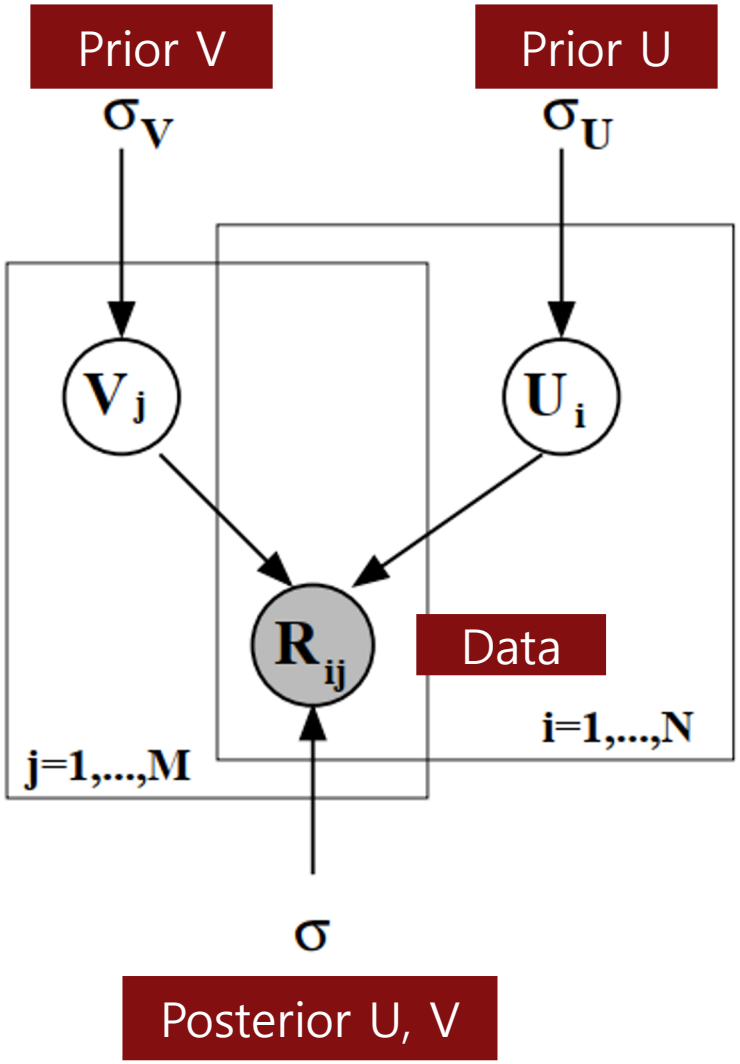
A variety of probabilistic factor-based models has been proposed recently [2, 3, 4]. All these models can be viewed as graphical models in which hidden factor variables have directed connections to variables that represent user ratings. The major drawback of such models is that exact inference is intractable [12], which means that potentially slow or inaccurate approximations are required for computing the posterior distribution over hidden factors in such models.

Low-rank approximations based on minimizing the sum-squared distance can be found using Singular Value Decomposition (SVD). SVD finds the matrix $\hat{R} = U^T V$ of the given rank which minimizes the sum-squared distance to the target matrix R. Since most real-world datasets are sparse, most entries in R will be missing. In those cases, the sum-squared distance is computed only for the observed entries of the target matrix R. As shown by [9], this seemingly minor modification results in a difficult non-convex optimization problem which cannot be solved using standard SVD implementations.

Instead of constraining the rank of the approximation matrix $\hat{R} = U^T V$, i.e. the number of factors, [10] proposed penalizing the norms of U and V. Learning in this model, however, requires solving a sparse semi-definite program (SDP), making this approach infeasible for datasets containing millions of observations.

1

Salakhutdinov, R., & Mnih, A., **Probabilistic matrix factorization,** *Advances in Neural Information Processing Systems 20,* 2008



Probabilistic Matrix Factorization Model

$$p(R|U,V,\sigma^2) = \prod_{i=1}^{N}\prod_{j=1}^{M}[\mathcal{N}\left(R_{ij}\middle|U_i^T V_j,\sigma^2\right)]^{Iij}$$

$$p(U|\sigma_U^2) = \prod_{i=1}^{N}\mathcal{N}(U_i|0,\sigma_U^2),\ p(V|\sigma_v^2) = \prod_{j=1}^{M}\mathcal{N}(V_j|0,\sigma_v^2)$$

Data (likelihood)

X

Prior distribution

$$p(U,V|R,\sigma^2) = p(R|U,V,\sigma^2)\,p(U,V|\sigma_U^2,\sigma_V^2) = p(R|U,V,\sigma^2)\,p(U|\sigma_U^2)\,p(V|\sigma_V^2)$$

$$p(U,V|R,\sigma^2) = \prod_{i=1}^{N}\prod_{j=1}^{M}[\mathcal{N}\left(R_{ij}\middle|U_i^T V_j,\sigma^2\right)]^{Iij}\prod_{i=1}^{N}\mathcal{N}(U_i|0,\sigma_U^2)\prod_{j=1}^{M}\mathcal{N}(V_j|0,\sigma_v^2)$$

Posterior distribution

Probabilistic Matrix Factorization

target function

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}||U_i||^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}||V_j||^2,$$

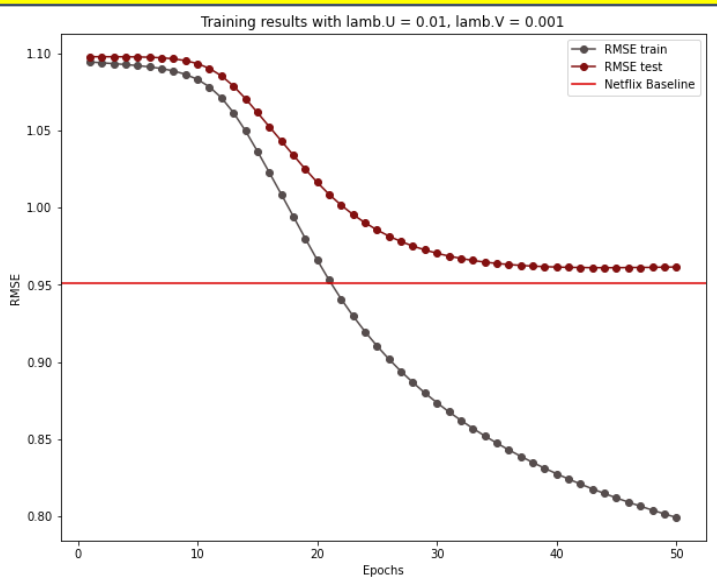
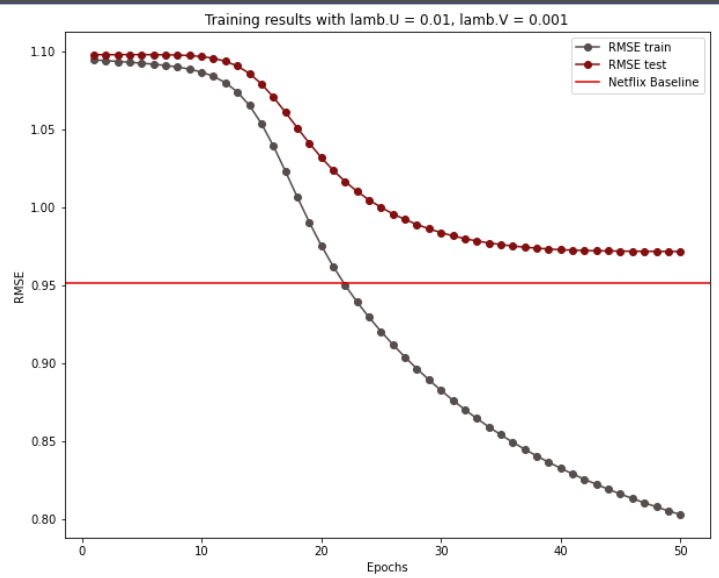
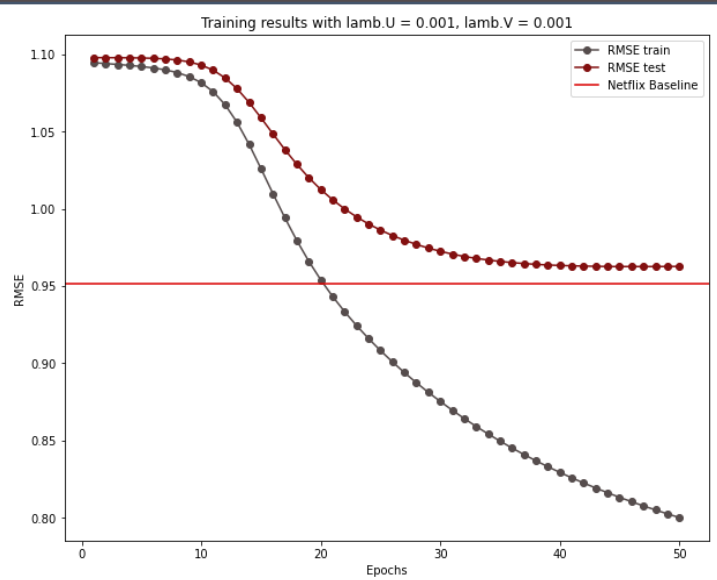
$$\lambda_U = \frac{\sigma^2}{\sigma_U^2},\ \lambda_V = \frac{\sigma^2}{\sigma_V^2}$$

Optimization

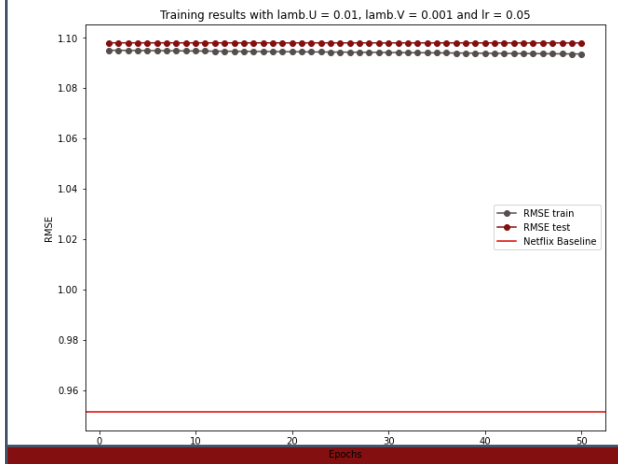
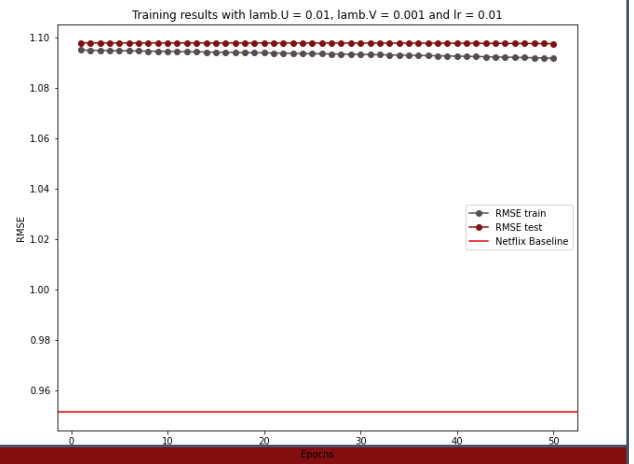
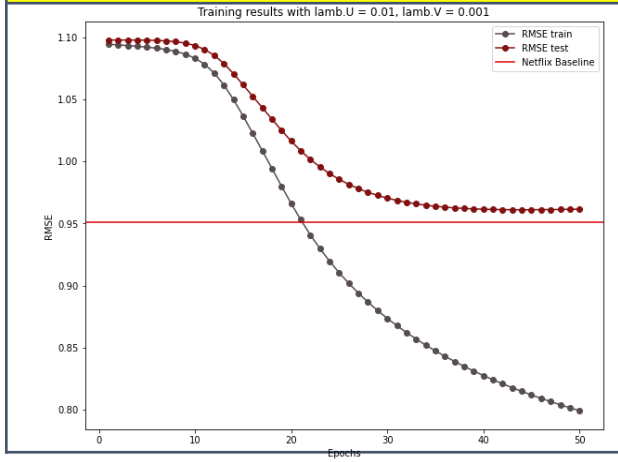
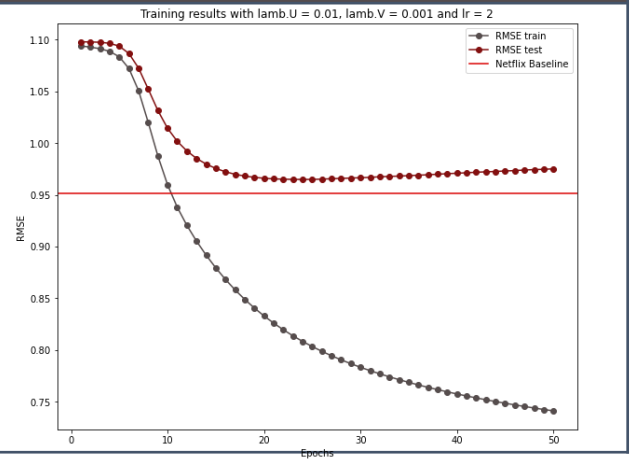
Momentum Gradient Descent

Local minimum에 빠지는 것을 막기 위해 관성을 주는 방법

Probabilistic Matrix Factorization

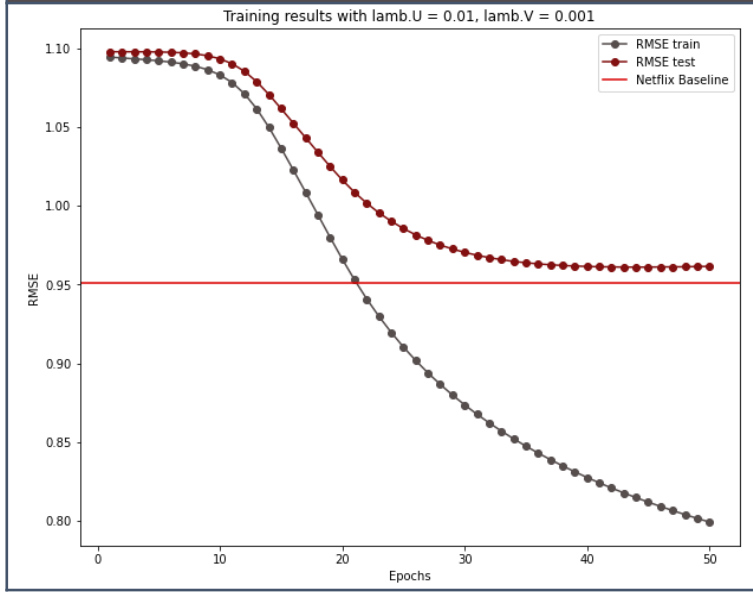
$\lambda_U = 0.01, \lambda_V = 0.001$	$\lambda_U = 0.001, \lambda_V = 0.0001$	$\lambda_U = 0.001, \lambda_V = 0.001$
Test RMSE : 0.9610	Test RMSE : 0.9716	Test RMSE : 0.9626
		

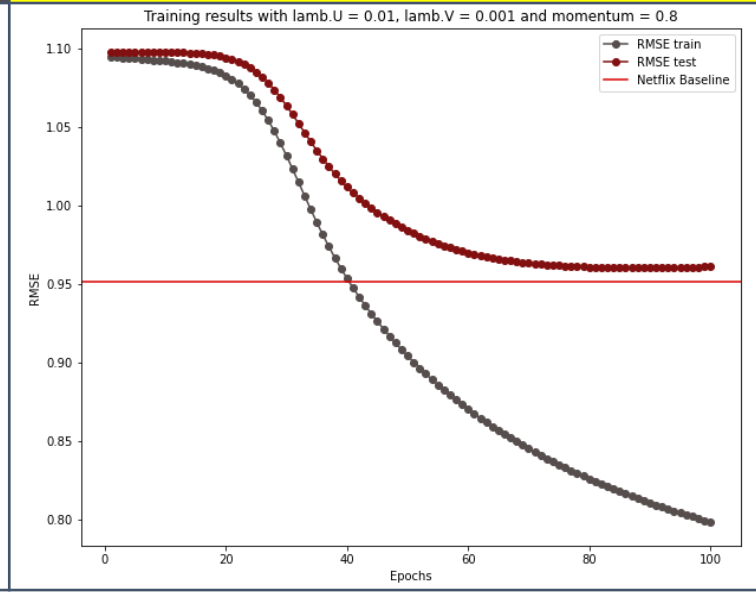
Probabilistic Matrix Factorization

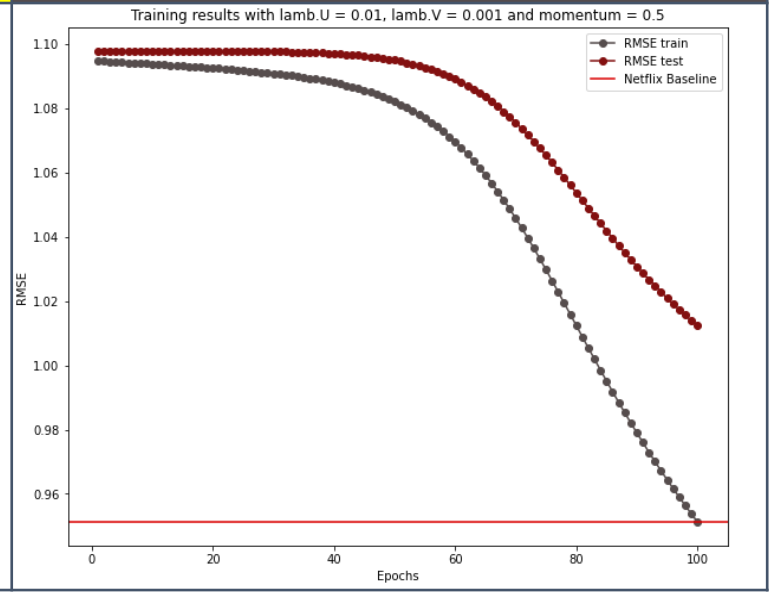
Learning rate = 0.05	Learning rate = 0.1
Test RMSE : 1.0980	Test RMSE : 1.0976
	
Learning rate = 1	**Learning rate = 2**
Test RMSE : 0.9610	Test RMSE : 0.9650
	

Probabilistic Matrix Factorization

momentum = 0.9	momentum = 0.8	momentum = 0.5
Test RMSE : 0.9610	Test RMSE : 0.9605	Test RMSE : 1.0124







Probabilistic Matrix Factorization

User A의 취향

	movie_id	title	rating	rating_pred
18	1082	Walking with Prehistoric Beasts	5.0	3.832549
73	2860	The Silence of the Lambs	5.0	4.207895
23	1389	Yanni: Live at the Acropolis	5.0	3.579948
78	3110	Dante's Peak	5.0	3.291114
79	3181	Desert Hearts	5.0	3.580797
26	1468	Bend It Like Beckham	5.0	3.705519
27	1480	Beyond Borders	5.0	3.492514
92	3668	Laughing Matters	5.0	3.733208
57	2450	Lord of the Rings: The Fellowship of the Ring	5.0	4.484694
31	1707	Clash of the Titans	5.0	3.726379

User A가 좋아할만한 영화는?

	movie_id	title	rating	rating_pred
2115	2160	CSI: Season 1	NaN	4.631397
4193	4303	The Sixth Sense	NaN	4.477036
2708	2780	Braveheart	NaN	4.430554
2486	2546	Gilmore Girls: Season 1	NaN	4.411368
1448	1474	Six Feet Under: Season 4	NaN	4.360109
265	269	Sex and the City: Season 4	NaN	4.336044
3027	3103	Ghost	NaN	4.332624
2068	2112	Firefly	NaN	4.318385
1765	1796	Lethal Weapon	NaN	4.307304
3000	3077	The Lion King: Special Edition	NaN	4.303823

Bayesian Probabilistic Matrix Factorization

Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo

Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Andriy Mnih AMNIH@CS.TORONTO.EDU
Department of Computer Science, University of Toronto, Toronto, Ontario M5S 3G4, Canada

Abstract

Low-rank matrix approximation methods provide one of the simplest and most effective approaches to collaborative filtering. Such models are usually fitted to data by finding a MAP estimate of the model parameters, a procedure that can be performed efficiently even on very large datasets. However, unless the regularization parameters are tuned carefully, this approach is prone to overfitting because it finds a single point estimate of the parameters. In this paper we present a fully Bayesian treatment of the Probabilistic Matrix Factorization (PMF) model in which model capacity is controlled automatically by integrating over all model parameters and hyperparameters. We show that Bayesian PMF models can be efficiently trained using Markov chain Monte Carlo methods by applying them to the Netflix dataset, which consists of over 100 million movie ratings. The resulting models achieve significantly higher prediction accuracy than PMF models trained using MAP estimation.

1. Introduction

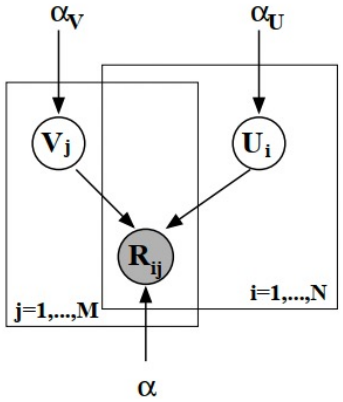
Factor-based models have been used extensively in the domain of collaborative filtering for modelling user preferences. The idea behind such models is that preferences of a user are determined by a small number of unobserved factors. In a linear factor model, a user's rating of an item is modelled by the inner product of an item factor vector and a user factor vector. This means that the $N \times M$ preference matrix of ratings that N users assign to M movies is modeled by the product of an $D \times N$ user coefficient matrix U and a $D \times M$ factor matrix V (Rennie & Srebro, 2005; Srebro & Jaakkola, 2003). Training such a model amounts to finding the best rank-D approximation to the observed $N \times M$ target matrix R under the given loss function.

A variety of probabilistic factor-based models have been proposed (Hofmann, 1999; Marlin, 2004; Marlin & Zemel, 2004; Salakhutdinov & Mnih, 2008). In these models factor variables are assumed to be marginally independent while rating variables are assumed to be conditionally independent given the factor variables. The main drawback of such models is that inferring the posterior distribution over the factors given the ratings is intractable. Many of the existing methods resort to performing MAP estimation of the model parameters. Training such models amounts to maximizing the log-posterior over model parameters and can be done very efficiently even on very large datasets.

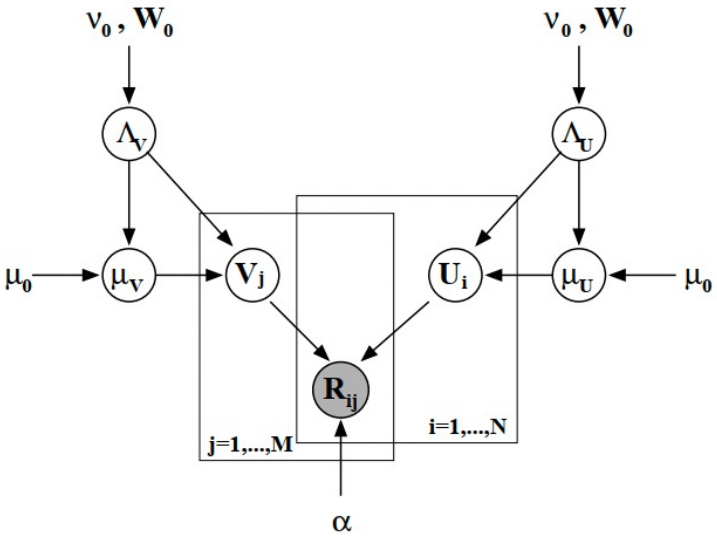
In practice, we are usually interested in predicting ratings for new user/movie pairs rather than in estimating model parameters. This view suggests taking a Bayesian approach to the problem which involves integrating out the model parameters. In this paper, we describe a fully Bayesian treatment of the Probabilistic Matrix Factorization (PMF) model which has been recently applied to collaborative filtering (Salakhutdinov & Mnih, 2008). The distinguishing feature of our work is the use of Markov chain Monte Carlo (MCMC) methods for approximate inference in this model. In practice, MCMC methods are rarely used on large-scale problems because they are perceived to be very slow by practitioners. In this paper we show that MCMC can be successfully applied to the large, sparse, and very imbalanced Netflix dataset, containing over 100 million user/movie ratings. We also show that it significantly increases the model's predictive accuracy, especially for the infrequent users, compared to the standard PMF models trained using MAP with regularization parameters that have been carefully tuned on the validation set.

Previous applications of Bayesian matrix factorization methods to collaborative filtering (Lim & Teh, 2007; Raiko et al., 2007) have used variational approxima-

Appearing in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008.



PMF



Bayesian PMF

Salakhutdinov, R., & Mnih, A.
Bayesian probabilistic matrix factorization using MCMC. ICML'08.

Bayesian Probabilistic Matrix Factorization

Data (Likelihood)

$$R\,|U,V,\sigma^2 \sim \prod_{i=1}^{N}\prod_{j=1}^{M}\left[\,Normal\left(R_{ij}\middle|U_i^T V_j,\sigma^2\right)\right]^{I_{ij}}$$

Gaussian Prior

$$\mathrm{U}_i \sim N(\mu_U,\Lambda_U^{-1}) \qquad V_j \sim N(\mu_V,\Lambda_V^{-1})$$

Gaussian – Wishart Hyperprior

$$\mu_U|\Lambda_U \sim Normal(\mu_0,(\beta_0\Lambda_U)^{-1})$$

$$\Lambda_U \sim Wishart(W_0,\nu_0)$$

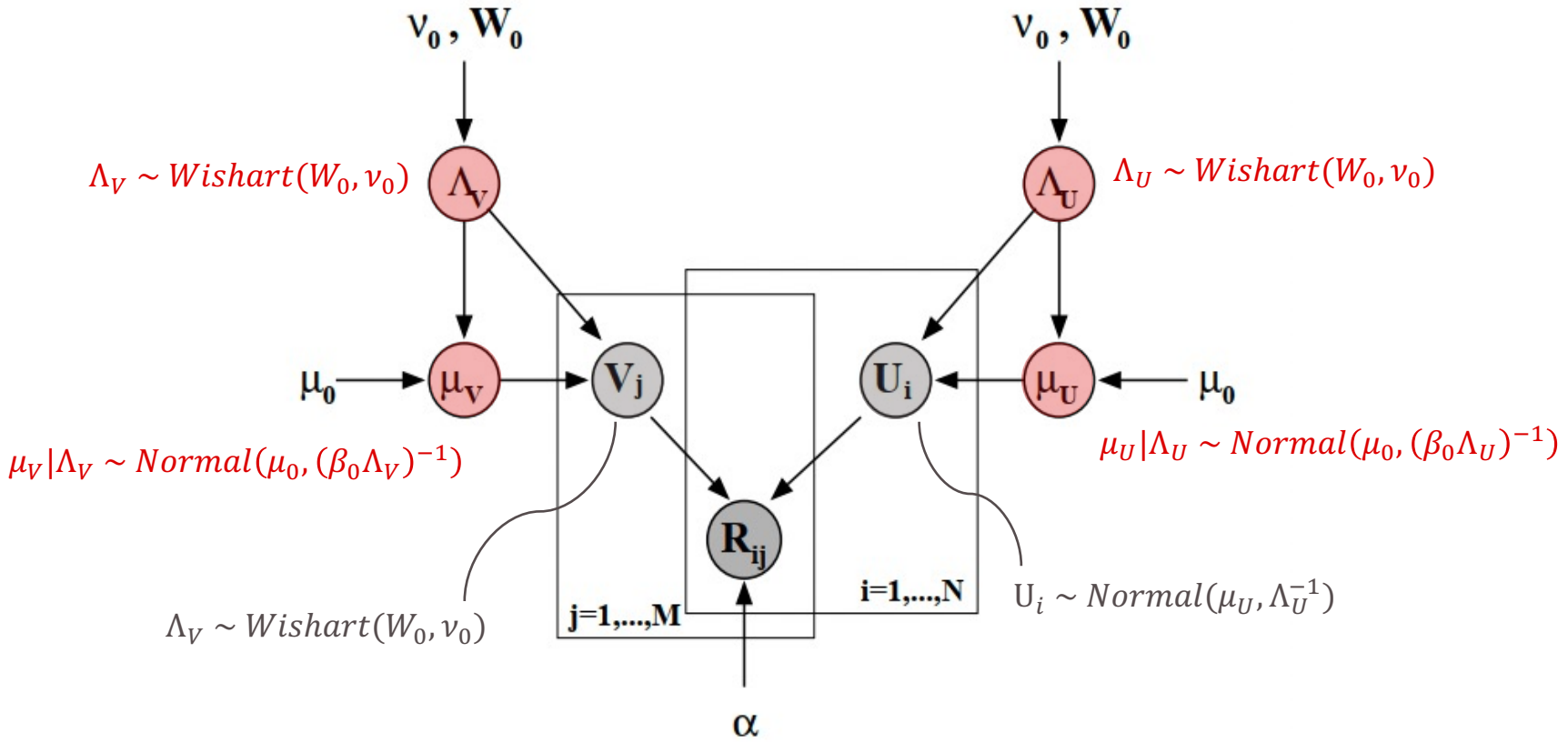
$$\rightarrow \{\mu_U,\Lambda_U\} \sim N(\mu_0,(\beta_0\Lambda_U)^{-1})\,W(W_0,\nu_0)$$

$$\mu_V|\Lambda_V \sim Normal(\mu_0,(\beta_0\Lambda_V)^{-1})$$

$$\Lambda_V \sim Wishart(W_0,\nu_0)$$

$$\rightarrow \{\mu_V,\Lambda_V\} \sim N(\mu_0,(\beta_0\Lambda_V)^{-1})\,W(W_0,\nu_0)$$

Bayesian Probabilistic Matrix Factorization



Bayesian PMF

Bayesian Probabilistic Matrix Factorization

Predictive distribution of R_{ij}^*

Posterior Expectation of $p(R_{ij}^*|U_i, V_j)$:

$$p(R_{ij}^*|R, \Theta_0) = \iint p(R_{ij}^*|U_i, V_j)\, p(U, V|R, \Theta_U, \Theta_V)\, p(\Theta_U, \Theta_V|\Theta_0)\, d\{U, V\}\, d\{\Theta_U, \Theta_V\}$$

$p(U, V, \Theta_U, \Theta_V|R, \Theta_0)$: joint posterior

$$\approx \frac{1}{K} \sum_{k=1}^{K} p(R_{ij}^*|U_i^{(k)}, V_j^{(k)}) \quad \text{by Monte Carlo approximation}$$

$\{U_i^{(k)}, V_j^{(k)}\}$: MCMC samples

Bayesian Probabilistic Matrix Factorization

objective function

$$\iint p\left(R_{ij}^{*} \middle| U_i, V_j\right) p(U, V | R, \Theta_U, \Theta_V)\, p(\Theta_U, \Theta_V | \Theta_0)\, d\{U, V\}\, d\{\Theta_U, \Theta_V\}$$

approximation

$$\frac{1}{K} \sum_{k=1}^{K} p(R_{ij}^{*} | U_i^{(k)}, V_j^{(k)}) \qquad \{U_i^{(k)}, V_j^{(k)}\} : \text{MCMC samples}$$

Bayesian Probabilistic Matrix Factorization

conditional posterior of $\{U, V\}$

$$p(U_i|R, V, \Theta_U, \alpha) \sim N(U_i \mid \mu_i^*, [\Lambda_i^*]^{-1})$$

$$p(V_j|R, U, \Theta_V, \alpha) \sim N\left(V_j \mid \mu_j^*, [\Lambda_j^*]^{-1}\right)$$

conditional posterior of $\{\Theta_U, \Theta_V\}$

$$\Theta_U = \{\mu_U, \Lambda_U\}$$

$$p(\mu_U, \Lambda_U|U, \Theta_0) = N(\mu_U|\mu_0^*, (\beta_0^*\Lambda_U)^{-1})\, W(\Lambda_U|W_0^*, \nu_0^*)$$

$$\Theta_V = \{\mu_V, \Lambda_V\}$$

$$p(\mu_V, \Lambda_V|V, \Theta_0) = N(\mu_V|\mu_0^*, (\beta_0^*\Lambda_V)^{-1})\, W(\Lambda_V|W_0^*, \nu_0^*)$$

Bayesian PMF using MCMC

Gibbs sampling Algorithm

$\Theta_0 = \{\mu_0, \nu_0, W_0\}$

Step 1. Initialize parameters: $\{U^1, V^1\}$

Step 2. For t = 1,...,T

sample the hyperparameters :

$$\Theta_U^t = \{\mu_U, \Lambda_U\}^t \sim p(\Theta_U | U^t, \Theta_0)$$

$$\Theta_V^t = \{\mu_V, \Lambda_V\}^t \sim p(\Theta_V | V^t, \Theta_0)$$

For each i = 1, ... ,N user features :

$$U_i^{t+1} \sim \mathrm{p}(\mathrm{U_i} | \mathrm{R}, \mathrm{V^t}, \Theta_U^t)$$

For each j = 1, ..., M movie features:

$$V_j^{t+1} \sim \mathrm{p}(\mathrm{V_i} | \mathrm{R}, \mathrm{U^{t+1}}, \Theta_V^t)$$

Bayesian PMF using MCMC

Gibbs sampling Algorithm

$\Theta_0 = \{\mu_0, \nu_0, W_0\}$

Step 1. Initialize parameters: $\{U^1, V^1\}$

Step 2. For t = 1,...,T

sample the hyperparameters :

$$\Theta_U^t = \{\mu_U, \Lambda_U\}^t \sim p(\Theta_U | U^t, \Theta_0)$$

$$\Theta_V^t = \{\mu_V, \Lambda_V\}^t \sim p(\Theta_V | V^t, \Theta_0)$$

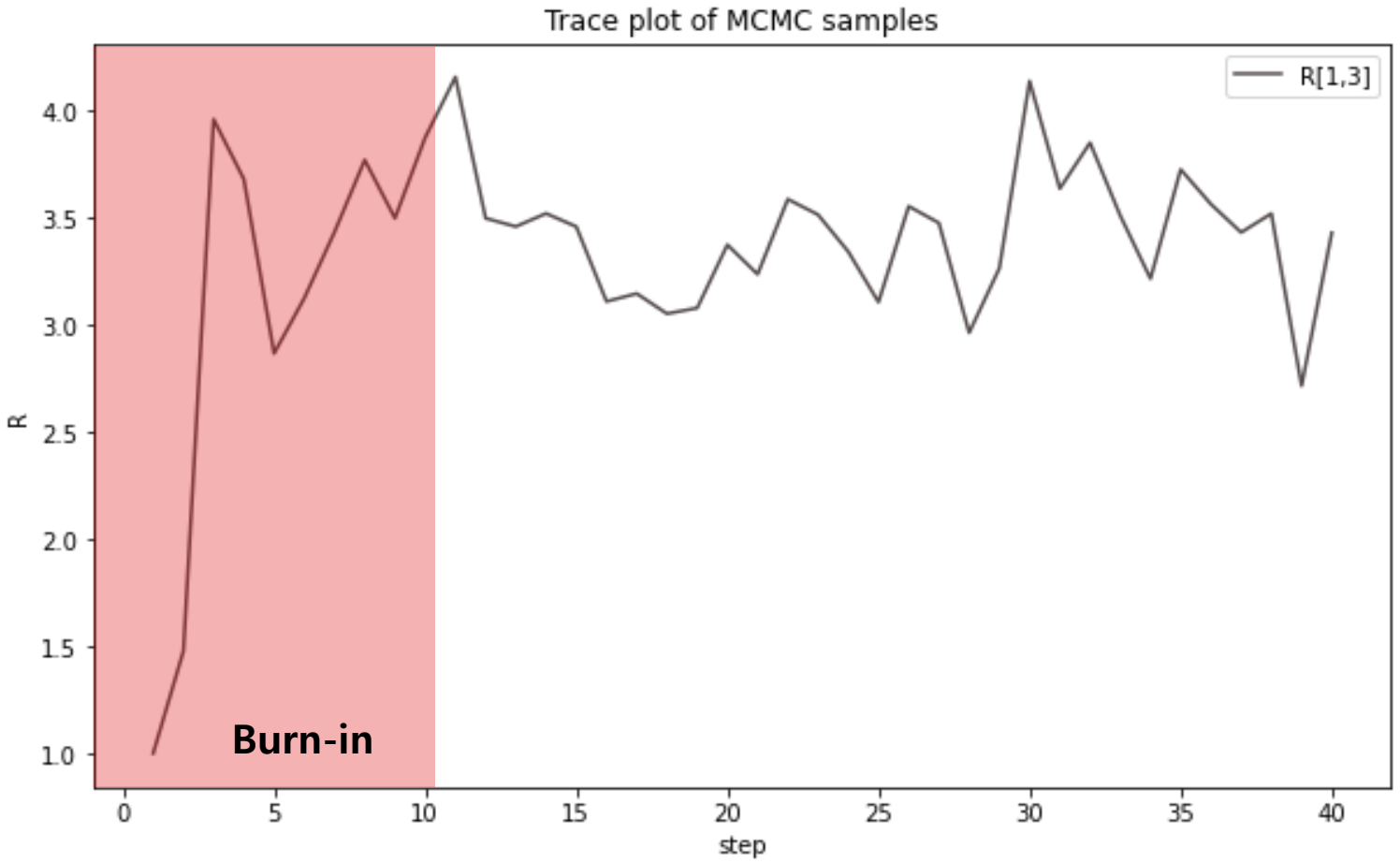
For each i = 1, ... ,N user features :

$$U_i^{t+1} \sim \mathrm{p}(\mathrm{U_i} | \mathrm{R}, \mathrm{V^t}, \Theta_U^t)$$

For each j = 1, ..., M movie features:

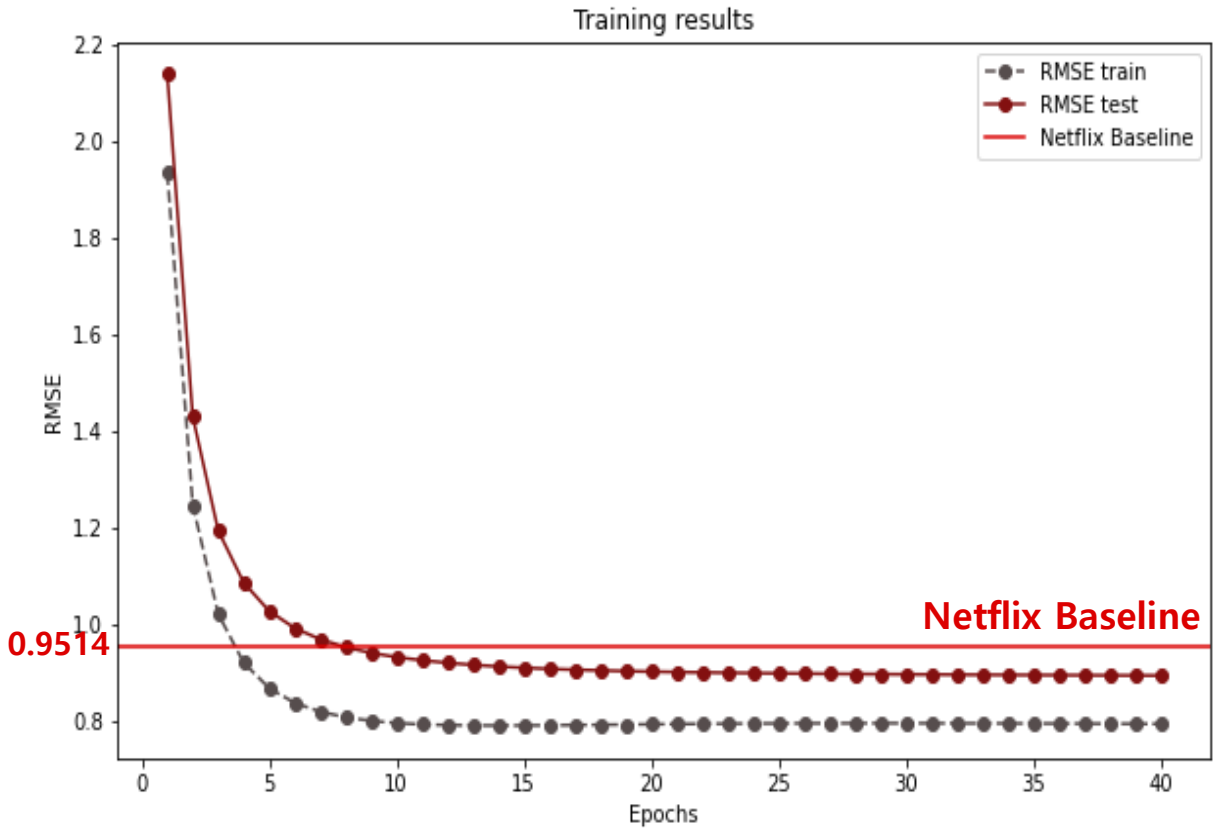
$$V_j^{t+1} \sim \mathrm{p}(\mathrm{V_i} | \mathrm{R}, \mathrm{U^{t+1}}, \Theta_V^t)$$

Bayesian Probabilistic Matrix Factorization



Gibbs sampling Result

Bayesian Probabilistic Matrix Factorization



D = 10
Test RMSE : 0.8942

Bayesian Probabilistic Matrix Factorization

User A가 좋아할만한 영화는?

	movie_id	title	rating	rating_pred
2115	2160	CSI: Season 1	NaN	4.631397
4193	4303	The Sixth Sense	NaN	4.477036
2708	2780	Braveheart	NaN	4.430554
2486	2546	Gilmore Girls: Season 1	NaN	4.411368
1448	1474	Six Feet Under: Season 4	NaN	4.360109
265	269	Sex and the City: Season 4	NaN	4.336044
3027	3103	Ghost	NaN	4.332624
2068	2112	Firefly	NaN	4.318385
1765	1796	Lethal Weapon	NaN	4.307304
3000	3077	The Lion King: Special Edition	NaN	4.303823

PMF recommendation

	movie_id	title	rating	rating_pred
3857	3960	Pollyanna	NaN	4.731766
3364	3454	Kenny Chesney: Greatest Hits	NaN	4.692884
4308	4425	Miranda (Tinto Brass)	NaN	4.654913
1465	1494	Appalachian Journey: Yo-Yo Ma/Edgar Meyer/Mark...	NaN	4.638162
2393	2451	Stealing Candy	NaN	4.620511
1907	1946	Spartacus	NaN	4.573866
265	270	Sex and the City: Season 4	NaN	4.551571
2485	2547	Green Legend Ran	NaN	4.545802
3853	3956	Soul in the Hole	NaN	4.534116
1761	1794	The Rage: Carrie 2	NaN	4.530854

BPMF recommendation

3. 결론

- 알고리즘 구현 결과 비교
- 한계점 및 제언

알고리즘 구현 결과 비교

	Basic MF	PMF	BPMF
Train RMSE	0.9045	0.7984	0.7968
Test RMSE	0.9356	0.9605	0.8942
epoch	50	100	40
time	25m 51s	15m 5s	55m 35s

한계점 및 제언

- 시간, 컴퓨터 자원의 부족 → 원 데이터를 전부 사용하지 못하고 줄여서 사용

Basic MF

- 규제항에 대한 여러 번의 실험이 필요

PMF

- 사전분포에 대한 가정이 현실과 다를 위험성
- 과적합

BPMF

- Markov Chain 수렴 진단의 명확한 기준 존재 X
- Computaionally expensive → 학습시간↑

참고 문헌

1. Y. Koren, R. Bell, and C. Volinsky, Matrix factorization techniques for recommender systems, IEEE Computing, 2009.
2. Salakhutdinov, R., & Mnih, A., Probabilistic matrix factorization. *Advances in Neural Information Processing Systems 20,* 2008
3. Salakhutdinov, R., & Mnih, A. Bayesian probabilistic matrix factorization using MCMC. *ICML'08*.
4. https://github.com/LoryPack/BPMF
5. https://towardsdatascience.com/pmf-for-recommender-systems-cbaf20f102f0

감사합니다