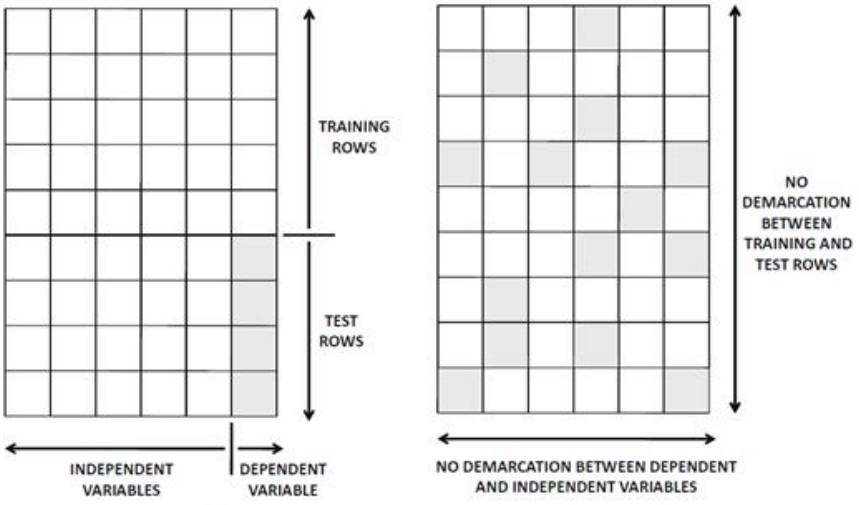
Chapter 3. Model-Based Collaborative Filtering

Chapter 3.1 Introduction, p.71-74

1) "This is because the traditional classification and regression problems are special cases of the matrix completion (or collaborative filtering) problem"
: Neighborhood-based CF 방식이 lazy learning method를 대표하는 KNN을 사용한 비 모델적 방식
↔거의 모든 머신러닝 기반 모델은 CF 시나리오로 일반화 될 수있음. 전통적 분류 및 회귀 모델은 MF(혹은 CF) 문제의 특정한 사례이기 때문이다.

2) 분류(classification)문제와 Matrix Completion문제의 주요 3가지 차이점



(a) Classification (b) Collaborative filtering

Classification	Matrix Completion
독립변수, 종속변수간 명확한 분리 O(column-wise)	X, 어떤 엔트리(entry, 입력값)가 예측 모델상에서 고려되냐에 따라, 각 컬럼은 종속변수이기도 하면서 독립변수이기도 함.
학습과 테스트 데이터의 분할이 명확(row-wise)	X, 특정한 관측된 엔트리를 학습 데이터로, 특정되지 않은 데이터를 테스트 데이터로 간주할 수 있을 뿐 ex) r_{31} 를 예측하고 싶다면, r_{31} 는 테스트, 나머지 유저에 대한 $r_{11}, r_{21}, r_{41}, r_{51}$ 값이 트레이닝데이터가 된다.
컬럼은 피처(feature, 특성)를 나타내고 열은 인스턴스를 나타냄	전치(transpose) 행렬이나 원(original)행렬 모두에 같은 접근법을 사용하는 것이 가능.

3) 이러한 CF 문제의 일반성은 단순 분류 문제에 비해 더욱 풍부한 알고리즘적인 가능성을 제공해 줌.

4) CF 문제와 분류 문제의 유사성은 *CF 문제를 해결하기 위한 학습 알고리즘을 설계할 때 유용*하게 사용될 수 있음.
→ 분류 문제는 상대적으로 잘 연구된 분야로서, 다양한 분류 문제의 해결 방법들은 CF 알고리즘의 디자인에 중요한 힌트를 제공
→ 실제로 대부분의 ML 및 분류 문제 알고리즘은 CF 문제의 맥락에서 직접적인 유사 개념이 있음.
→ 가령 배깅, 부스팅, 모델 결합 등의 meta-algorithm들은 CF 문제 해결에도 확장될 수 있음. 이 방법론들은 실제 Netflix 대회에서 가장 높은 성능을 보이는 것들 중 하나.

5) sparsity 문제가 심각할때는 분류 문제를 일반화 하는 방식이 유용하지 않을 수 있음. 게다가, 다양한 상황따라 다종의 모델들의 상대적 효율성은 달라지기 마련임.
→ 가령 잠재 요소 모델(Latent factor model)은 CF 문제 해결에 특히 효율적이지만, 일반 분류 모델에서는 그렇지 않음.

6) 모델 기반 추천 시스템은 보통 이웃 기반 방식에 비해 많은 장점
   - **Space-efficiency**
     : 보통은 학습된 모델은 원본 평점 행렬(ratings matrix)보다 훨씬 작다. 즉, 요구되는 공간이 상대적으로 적다.
     ↔ user-based Neighborhood-based CF: O(m^2)의 공간 복잡도
     item-based Neighborhood-based CF: O(n^2)의 공간 복잡도

   - **Training speed and prediction speed**
     : 이웃 기반 방식의 단점은 전처리 단계가 유저 혹은 아이템 수의 제곱이라는 점이다.
     ↔ 모델 기반 시스템은 학습된 모델을 만들 때 전처리 단계가 훨씬 빠르다. 대부분의 경우에, 압축되고 요약된 모델이 예측을 효율적으로 수행하는 데 사용된다.

   - **Avoiding overfitting**
     : 분류나 회귀 문제에서 심각한 문제인 과적합 문제를 회피하는 데 효율적이다. 또한 이러한 모델을 더욱 강건하게 만들기 위한 정규화(regularization) 방법이 적용될 수 있다.

Chapter 3.4.1 Naive Bayes CF_ Handling Overfitting, p.84

ratings matirx는 여전히 sparse하고, observed ratings는 적기때문에, 데이터에 기반한 예측모델($P(r_{uj} = v_s)$)이 not robust.
ex) j번째 item을 평가한 user가 한 명도없다면, $P(r_{uj} = v_s)$ = 0 / 0.

또한 sparse하기 때문에, $\prod_{k \in I_u} P(r_{uk} | r_{uj} = v_s)$ 도 less robust해진다.

이러한 문제를 해결하기 위해, 나이브베이즈 모델에서는 smoothing방법을 활용하는데, 책에서는 Laplacian smoothing을 소개한다. 즉, 확률값이 곱해지면서 0에 수렴하거나, 0으로 단정되어 학습이되지않는 문제를 방지하기위해 보정값을 활용하는 것이다.

Chapter 3.6 Latent Factor Models, p.90-128

1) 2장에서 차원축소 방식을 사용한 matrix completion에 대해 배웠음. 이웃 알고리즘을 사용한 여러 휴리스틱한 알고리즘이 제시되었음. 이러한 차원축소 기법들은 다른 모델 기반 방법을 가능케 하는데도 사용되는데, 이 방법에서는 분류 알고리즘을 서브루틴으로서 활용함.
   → 따라서 차원축소는 모델 기반 방법을 위한 더 간편한 data representation을 만들어 내는 데 조력하는 역할을 했음.
   →3.6절에서는 차원 축소를 사용해 원행렬을 한방에 근사할만한 좀 더 정교한 방법론이 소개될 것임.

2) “The basic idea is to exploit the fact that significant portions of the rows and columns of data matrices are highly correlated.”
   : 데이터 행렬의 열과 행의 중요 부분들은 높은 상관관계를 갖는다는 점을 활용한다는 것이 기존의 matrix completion을 위한 latent factor model의 basic idea였음.
   → 결과적으로, 데이터는 내재적인 잉여 정보를 지니고 있으며, 결과로 나오는 data matrix는 low-rank matrix로 상당히 잘 근사되고는 함.
   → 이러한 내재적인 잉여성때문에, 완전 특정된 저수준 랭크 근사(fully specified low-rank approximation)는 원 입력 정보의 부분집합만으로도 결정될 수도 있다.
   → 이 fully specified low-rank approximation은 결측 엔트리에 강건한(robust) 근사를 제공하기도 한다.
   → 차원 축소와 EM(expectation-maximization) 알고리즘을 결합해 불완전한 data matirx의 엔트리를 재생성하는 접근을 취한 논문도 있다.

3) 잠재 요인 모델은 (글이 쓰인 당시)SOTA recsys로, 차원축소를 사용해 결측 엔트리를 채워 넣는다.
   차원 축소 방식의 기본적인 원리는, 차원간의 쌍 방식(pairwise)상관관계가 제거되도록 axis system을 회전시키는 것이다.
   차원 축소의 핵심 아이디어는 축소되고, 회전되고, 완전 특정된 표현이 불완전한 data matrix에서 강건하게 추정될 수 있다는 것이다.
   온전히 특정된 표현(completely specified representation)이 얻어지기만 하면, 원래 axis system으로 돌리면 fully specified representation을 얻을 수 있음.

4) 이러한 방식은 행간, 열간의 상관관계를 사용해 fully specified and reduced representation을 얻는 것. 이 방법은 여러 추천시스템 방법론에 사용됨.

5) 이 방법의 원리에 대해 기하적, 의미적 해석으로 설명할 예정임.

Chapter 3.6.1 Geometric Intuition for Latent Factor Models

1) it means that the original data matrix has a rank of approximately 1 after removing the noisy variations.

2) Note that dimensionality reduction methods such as Principal Component Analysis (PCA) and (mean-centered) Singular Value Decomposition (SVD) typically represent the projection of the data along this line an approximation

3) When the m × n ratings matrix has a rank of p<< min{m, n} (after removing noisy variations), the data can be approximately represented on a p-dimensional hyperplane. In such cases, the missing ratings of a user can often be robustly estimated with as few as p specified entries as long as the p-dimensional hyperplane is known.

4) For example, if the rating of Spartacus is fixed at 0.5, then the ratings of Nero and Gladiator can be estimated as the intersection of the 1-dimensional latent vector with the axis-parallel hyperplane, in which the rating of Spartacus is fixed to 0.5. This hyperplane is illustrated in Figure 3.6. Therefore, dimensionality reduction methods such as SVD leverage the inter-attribute correlations and redundancies in order to infer unspecified entries.
   → 노이즈 제거 후 1-rank로 approx되는 원래 3차원 데이터의 예를 보자. 이 경우 1차원 하이퍼플레인이 얻어지는데, 이 하이퍼플레인으로도 나머지 생략된 엔트리를 (강건하게) 추정해낼 수 있다는 것이다.
   만일 스파르타쿠스의 평점이 0.5라는 값에 고정되면, 방금의 1차원 latent vector(i.e. 1차원 하이퍼플레인)이 축-평행 하이퍼플레인(spartacus=.5)과 교차하는 지점의 주변에 존재하는 데이터 포인트가 우리의 추정 값이라는 것 (적어도 그 주변에 있으리라 기대하는 것)

5) 위의 사례에서 relevant latent vector를 추정하기 위해서는 fully specified data matrix가 있다고 가정되었지만, 'dominant'한 latent vector를 추정하기 위해서는 반드시 그럴 필요는 없음(결측 엔트리가 존재해도 무관하다는 의미).
   결측치의 유무에 관계 없이 latent vector를 추정할 수 있는 능력이야말로 latent factor approach의 성공의 핵심.

6) 이러한 방법론의 기본 아이디어는 다음과 같음:
   latent vector들에 의해 정의되는 하이퍼플레인과 데이터 포인트(개인의 rating을 나타내는)들간의 평균 제곱 거리가 가능한 한 가까워지는 latent vector의 집합을 찾는 것 (선형 회귀모델의 OLS와 유사한 개념인듯 함)

7) 따라서, 데이터가 근사적으로 놓여 있을 저차원 하이퍼플레인을 찾기 위해서는, 일부분만 특정된(partially specified) 데이터셋을 사용해야함.
   이렇게 함으로써 데이터 안의 상관관계의 구조에 있는 잠재된 잉여성을 implicit하게 잡아내어, 모든 결측값을 한번에 재생성 할 수 있음.

8) 데이터가 어떠한 상관관계나 잉여성을 지니고 있지 않다면, 잠재 요소 모델은 작동하지 않음.

Chapter 3.6.2 Low-Rank Intuition for Latent Factor Models

1) 앞에서 살펴본 기하적인 직관은 모든 잠재 벡터들이 상호 직교(mutually orthogonal)할 때의 영향을 이해하는 데 유용함.
그러나 늘 상호 직교하는 것은 아님. 이러한 경우에는, 선형대수의 직관을 얻는 것이 유용함. 한 방법으로는 이러한 행렬에서 factorization의 역할을 살펴보는 것이 도움이 됨.

2) Factorization: 행간 혹은 열간의 상관관계로 인해 차원 축소에 용이할 때 이 행렬을 근사하는 좀 더 일반적인 방법론임. 대부분의 차원축소 방법은 Matrix Factorization으로도 표현될 수 있음.

3) rank k의 matrix R (m*n)이 있을 때, rank-k factor의 곱 형태로 나타낼 수 있음 (R=UV^T)
여기서 U는 m*k, V는 n*k임. 여기서 열공간과 행공간의 rank는 k임을 주목하라. (기저벡터로 span 될 수 있는 vector space)
U의 각 col은 R의 k차원 열공간의 기저를 이루는 k개 기저벡터의 하나로 간주되며, V의 j번째 row는 이러한 기저 벡터들과 결합하여 R의 j번째 col을 이루도록 하는, 그에 상응하는 계수를 포함한다.

↔ V를 R의 행공간의 기저 벡터들로 볼 수 있고, U의 row를 상응하는 계수로 볼 수 있다.

4) 이러한 형태의 어떠한 rank k의 행렬도 factorize할 수 있는 능력은 선형대수의 근본적 사실임. 그리고 무한한 수의, 다양한 기저벡터의 집합에 상응하는 이런 형태의 factorization이 있음.
SVD는 이런 factorization의 한 예시로, 기저 벡터들이 U의 칼럼, 그리고 V의 칼럼으로 표현되며 각각에 대해 직교한다.

5) R이 k보다 큰 rank를 갖는다 하더라도, 근사적으로 rank-k factor의 곱으로 표현될 수 있음.
전 예시와 같이, U는 m*k, V는 n*k. 근사의 오차는 ||R-UV^T||^2로 표현되는데, 이는 residual matrix의 Sum of Squares임. 이 값을 residual matrix의 (squared) Forbenius norm이라고 부른다. 이러한 residual matrix는 잠재된 평점 행렬안의 노이즈를 나타내는데, 이는 low-rank factor에 의해 모델링 될 수 없다.

6) factorization process의 함의, 그리고 높이 상관된 행들과 열들을 가진 matrix에 대한 영향
유저 6명과 영화 7개가 있을 때, rank가 2인 matrix factorization을 하면 역사/로맨스라는 두개의 잠재 변수로 표현할 수 있음. 여기서 latent vectors는 SVD와는 다르게 상호 직교하지 않음.
이 예시처럼 sementic하게 해석될 수는 있으나 항상 그렇지는 않다. 만약 factorized 된 각 matrix U/V에 임의의 실수가 곱해진 경우, 해석이 어려워짐.
그럼에도 불구하고, U와 V의 k개 열들은 유저와 아이템간의 핵심적인 상호작용을 각각 표현해낸다.
이 k개 열들은, 의미적으로 해석 가능하든 그렇지 않든 추상적으로 잠재 개념(latent concept)으로 볼 수 있다.

7) 결측치가 많은 행렬에서도 위처럼 잠재 특성의 모든 입력값(all entries of latent factors U and V)을 강건하게 추정할 수 있음.
rank가 낮은 경우에서도(for low values of the rank), sparsely specified data로부터 추정하는 것은 여전히 가능함.
“This is because one does not need too many observed entries to estimate the latent factors from inherently redundant data.”
→ 너무 많은 값이 필요하지 않기 때문!
U와 V행렬이 추정되기만 하면, 전체 평가 matrix는 UV^T로 한번에 추정될 수 있고, 이는 한번에 결측치를 채워낼 수 있음을 의미함.

Chapter 3.6.3 Basic Matrix Factorization Principles

1) R~UV^T 에서, U의 i번째 행 u_i는 유저 특성, V의 j번째 행 v_j는 아이템 특성을 의미함.
이 특성 벡터들은 k개의 컨셉에 대한 연관성을 나타냄.

2) 이 두 벡터를 내적하면, 해당 유저의 해당 아이템에 대한 평가가 근사적으로 표현됨.
r_ij ~ u_i'v_j

3) 이 잠재 특성 벡터는 의미적인 해석을 찾기 어려운 경우가 왕왕 있음. 그러나, 이들은 평점 행렬 내의 우세한 상관관계 패턴을 표현해냄.
+ NMF와 같은 방법론은, 잠재 벡터의 해석가능성을 높이기 위해 명시적으로 고안됨.

4) MF 알고리즘간의 차이점은,
1. U와 V에 대한 제약 ((잠재 벡터들의?) 직교성 혹은 잠재 벡터들의 비음수성)
2. 목적 함수의 특성 (Frobenius norm 최소화 혹은 생성모델의 MLE)으로 나뉘어짐.

Chapter 3.6.4 Unconstrained Matrix Factorization

1) 제약없는 MF 모델이라 하면 보통 SVD를 떠올리지만, 엄밀히는 틀린 표현임. SVD는 U와 V가 직교해야 함.
따라서 제약없는 MF 모델의 예시를 하나 들고, SVD를 따로 살펴보려고 함.

2) 제약없는 MF 모델에서 U와 V를 찾을 때, ||R-UV^T||^2와 같은 Frobenius norm을 최소화 시키는 방법을 떠올릴 수 있음.
그러나 우리의 R은 일부분의 엔트리만 알려져 있음. 따라서 목적 함수 역시 정의되지 않는다.결측치를 포함한 행렬의 Frobenius norm을 구할수는 없으므로, 값이 존재하는 entry만을 사용.
관측된 엔트리를 가지고 u_is와 v_js를 내적한 뒤, 이 값을 원래 평가 r_ij에서 빼준 것을 error로 사용, 이를 minimize. (u_is와 v_js는 unknown variables.)

3) 이런 최적화 과정을 SGD로 구현할 수 있음.
목적함수 J를 결정변수 u_is와 v_js 기준으로 편미분하면 됨. 이 편미분 값은 결정변수의 벡터 관점에서의 기울기를 제공하고, 이 기울기에 일정한 alpha(learning

rate)를 곱한 만큼을 업데이트해, 수렴할 때까지 반복하는 것을 Gradient Descent라고 부름.
* 이때, U와 V 내의 모든 엔트리에서 업데이트가 동시에 이루어 짐.

Chapter 3.6.4.1 Stochastic Gradient Descent

위와 같은 방식을 batch update method라고 한다(전체 batch를 업데이트 하는 방식이라)

1) batch update method 대신 **각각의 엔트리에 대해** 업데이트 하는 방식이 있다. 이러한 업데이트는 stochastically(확률적으로) 근사될 수 있다. 무작위적으로 선택한 개별 엔트리의 에러에 기반해 기울기가 근사되는 방식.

2) batch update method가 좀 더 smooth하게 converge하지만, SGD 방식이 더 빠르게 수렴함.

3) 이 둘의 장단을 절충하기 위해 mini-batch를 사용하는 방법도 있다.

4) bold driver algorithm: 매 iter마다 lr을 선택하는 방식.

5) 수렴에 너무 지나치게 엄격한 기준을 사용하는 것은 권장되지 못한다. 너무 많은 iter를 돌리면 관측되지 않은 엔트리에 대해 더 안좋은 성능을 보이기도 하기 때문.

6) 초기값의 설정 역시 중요함. SVD로 근사적인 초기값을 설정하는 방법 등이 있음.

Chapter 3.6.4.2 Regularization

오버피팅을 방지하는 그 정규화

Chapter 3.6.4.3 Incremental Latent Component Training

먼저 잠재변수 1개부터 시작해서, U와 V를 구하고 이 두 벡터를 외적해 원행렬 R의 크기로 복원한다. 이를 R에서 빼어 줌. 그 다음 잠재변수의 수를 2개로 늘리고, 2번째 잠재변수 벡터의 외적을 R에서 빼어 준다. 이 과정은 전체 잠재변수의 수인 k까지 k번 반복됨.
(SVD에서 dominant component 1개부터 계속 중첩해 나가는 식으로 계산하는 방식과 유사해 보임)

Chapter 3.6.4.4 Alternating Least Squares (ALS) and Coordinate Descent

일반적으로, lr과 init 값에 크게 좌우받는 SGD 방식에 비해 좀 더 안정적으로 알려진 방법.
이 방법 역시 초기에 U와 V 매트릭스를 가지고 시작하는 반복적 방법.

1) 순서 설명

1. U를 고정시키고, V의 n개 행에 대해 Least-squares regression problem을 해결함.
U의 모든 벡터를 상수로 취급하고, V의 모든 벡터를 최적화 변수로 간주함.
총 n개의 LS problem이 실행되어야하고, 각 LS problem은 k개의 변수를 가지고 있음.
각 상품에 대한 LS problem은 독립적이므로, 쉽게 병렬화될 수 있음.

2. V를 고정시키고, U의 m개 행에 대해 1.과 같은 방식을 반복하면 됨.

2) 정규화가 목적함수에서 사용되면 Tikhonov regularization이라고 불림.
정규화 파라미터 람다는 모든 독립적 LS problem에 고정될 수도 있고 다양하게 적용될 수도 있음. 이 때 람다의 값을 cross-validation이나 hold-out과 같은 방식으로 최적치를 찾아 조정할 수 있음.

3) 장점:
행렬의 모든 값이 0값으로 fully specified된 경우, implicit한 feedback 상황에서 특히 weighted ALS가 강점을 보인다고 함.
이런 상황에서 non-zero 엔트리들이 더욱 무겁게 가중하는 경향을 보임.
대부분의 엔트리가 0일 경우, SGC 방식은 너무 비용이 많이 들어감. 이 때 ALS가 특히 효율적.

4) 단점:
large scale, explicit rating 상황에서는 SGD만큼 효율적이지는 않다고 함.

5) coordinate-descent:
SGD와 ALS의 효율성/안정성 trade-off를 효율적으로 처리하는 접근.
U나 V 둘 중 하나의 matrix의 단 한 rating 포인트를 가지고 최적화를 수행하는 접근.
→ 왜 잘되는거지?

Chapter 3.6.4.5 Incorporating User and Item Biases

NN 방식에서 group mean을 빼어 주는 것처럼, global bias를 도입해 학습 대상에 추가하는 것. * o_i는 유저의 편향, p_j는 아이템의 편향을 반영함.

Chapter 3.6.4.6 Incorporating Implicit Feedback

유저가 평가했다는 그 사실 자체를 implicit한 feedback으로 간주하고, 이를 반영하는 방법. SVD++과 asymmetric factor model과 같은 방법론이 제안되었음.

1) 위의 알고리즘들은 item factor의 matrix를 V와 Y로 각각 사용하는데, 이는 명시적/암시적 feedback을 각각 의미함.

2) user latent factor는 암시적 item latent factor matrix Y의 행들의 선형 결합을 통해 일부분 혹은 전체가 유도된다.

3) 핵심적인 아이디어는, user factor는 유저의 선호에 상응하고, 유저의 선호는 그러므로 그들이 평가하기로 선택한 아이템에 의해 영향을 받는다는 것이다.

4) SVD++은 이러한 asymmetric factor model에 factorization framework를 결합.

Chapter 3.6.5 Singular Value Decomposition

truncated SVD의 기하적인 의미에 대해 설명했음.

1) 앞서 설명한 rank-1의 경우의 해석을 참고해서 설명했는데, 완전히 이해하지는 못했음.

2) 3개로 쪼개어 진 SVD는 가운데 singular value의 matrix를 U 또는 V에 결합함으로써 두개로 만든다. 보통 U에 결합한다.

3) 학습 과정은, 기존의 Frobenius norm을 최소화하는 방식에 제약조건으로
U의 칼럼들은 mutually orthogonal 해야하며,
V의 칼럼들도 mutually orthogonal 해야한다는 조건이 걸려있다.

4) 제약조건이 없는 MF는 보통 관측된 결과에서 적은 error를 보인다. 그러나 unseen에는 성능을 알 수 없다.
"because of varying levels of generalizability of different models, it turns out that the optimal value of J is identical in the case of SVD and unconstrained matrix factorization, if the matrix R is fully specified and regularization is not used"

Chapter 3.6.5.1 A Simple Iterative Approach to SVD

이 절에서는 행렬 R이 불완전하게 상술되었을 경우 최적화 문제를 해결하는 방법에 대해 논의함

1) 우선 R의 각 행에 대해 mean-center를 수행함. (각 유저 rating의 평균을 각 행에서 빼어 줌)
이 평균 값들은 보존되어, 결측 엔트리를 복원할 때 재사용됨.

2) R_c를 centering이 끝난 R이라고 하자. 이 R_c의 모든 결측 값은 "0으로 채워진다"
→ 이는 해당 유저의 평균 rating으로(mean-centering 되었으므로) 결측 엔트리를 채워주는 것과 같음.

3) 예측 rating r_ij는 u_i와 v_j의 내적에 mu_i가 더해져 만들어짐

4) 그러나 mean subtraction의 방법은 상당한 편향을 가질 수 있음.
이러한 편향을 줄이는 방법은 여러 가지가 있는데, 그 중 하나는 MLE를 사용하는 것이고, 다른 하나는 반복적으로 결측 엔트리의 추정을 개선해 가며 편향을 줄여가는 방법이다.

    1. MLE 방식
    → 2.5.1 Handling Problems with Bias

: 원래는 rating matrix에서 더욱 상관관계가 깊은 영화 2개가, mean-centering을 한 뒤에는 더 상대적으로 덜 상관이 있게 나와버림.
: 이는 공분산행렬의 값이 더욱 낮게 보정됨으로써 드러남.
: 원 평점 행렬에 결측 엔트리가 많을수록, mean-centering 방식은 더 큰 편향을 가지게 됨.

Chapter 2.5.1.1. Maximum Likelihood Estimation

: 이 공분산 행렬을 추정하기 위해 EM 알고리즘과 같은 확률적인 방법을 제안한다.
: 데이터에 대해 생성 모델이 가정되고, 특정된 엔트리 값들은 이 생성 모델의 결과물로 간주됨.
: 공분산 행렬은 이 생성 모델의 파라미터를 추정하는 과정의 일부로 추정될 수 있음. (공분산 행렬의 MLE가 추정될 것.)
: 각 아이템 쌍 사이의 공분산의 MLE는 오직 특정된 엔트리들을 간의 공분산으로써 추정됨. 즉, 특정 아이템 쌍에 대한 rating을 특정한 유저만이 공분산을 추정할 때 사용된다는 것.
: 한 쌍의 아이템 사이에 공통 유저가 없을 경우, 공분산은 0으로 추정됨.
: 이러한 방식이 항상 효과적인 것은 아니나 mean-filling 기법보다는 효과적임.
: 이 결과로 나온 공분산 행렬의 top-d 고유벡터를 선택함으로써 축소된 n*d 기저 행렬 P_d가 계산됨.
: 완전 상술된 R을 P_d에 사영하는 것이 아니라, 불완전한 행렬 R을 사영하는 방법이 사용될 수 있는데, 이는 각 관측된 rating이 P_d의 각 잠재 벡터에의 projection에 기여한 정도를 계산하기 위함이다.

2. 반복적 추정 개선 방식
   → 1) row-mean으로 결측치를 채워 init함.
   2) R_f의 rank-k SVD를 수행
   3) 원래 결측치였던 값만 SVD의 estimation 값으로 추정하고, 다시 1단계로.

Chapter 3.6.5.2 An Optimization-Based Approach

1) 반복적 접근법은 비용이 많이 들어감. 이는 fully specified matrices를 사용해 작업하기 때문.

2) 더 작은 행렬에 대해 수행하는 것이 더 간단하지만, large-scale 상황에서는 잘 확장되지 않음.

3) specified entries에 대해서만 연산을 수행하되, GD 방식을 사용할 수 있음. 여기에 직교성 제약을 걸어준다.

4) projected gradient descent
-> U 혹은 V의 특정 칼럼의 모든 요소가 한 회마다 업데이트 되는 방식을 이름

Chapter 3.6.5.3 Out-of-Sample Recommendations

1) MF와 같은 matrix completion 방법은 태생적으로 transductive한데, 이는 학습에 사용된 원 평점 행렬에 포함된 유저와 아이템에 대한 prediction만이 이루어질 수 있음을 의미

2) 직교 기저 벡터의 한 장점은 새로운 유저와 아이템에 대해 out-of-sample 추천을 수행할 수 있다는 점임.
→ 이러한 문제를 귀납적(inductive) matrix completion이라고 함.

3) 잠재 벡터가 구해졌을 경우, 특정된 rating 안의 정보를 그에 상응하는 잠재 벡터에 project할 수 있음
→ 이는 벡터들이 서로 직교할 경우 훨씬 쉬워짐.
→ SVD가 U와 V라는 잠재 요소들을 각각 얻었을 경우를 생각
: V의 열들은 원점을 교차하는 k-차원의 하이퍼플레인을 정의함(H_1).

4) 새로운 유저가 추가됐을 경우의 시나리오
1. 총 h개의 평가를 내린 유저가 시스템에 등장
2. 해당 유저에 대해서는 h개의 값이 고정된 (n-h) 차원의 하이퍼플레인이 이 유저의 rating의 확률 공간임. 이 하이퍼플레인을 H_2로 정의. (여기서 n은 상품의 수)
3. 이제 목표는 H_2 위의 점 중 H_1에 가장 가까운 지점을 결정하는 것임. 여기서 H_2의 점은 다른 상품에 대한 모든 rating을 결정하게 됨.
4. 3가지 가능성이 존재함
1] H_1과 H_2가 교차하지 않는다
: H_1에 가장 가까운 H_2 위의 지점이 반환됨. 한 쌍의 하이퍼플레인 사이의 가장 가까운 거리는 간단한 Sum of Squares 최적화 문제로 표현될 수 있다.
2] H_1과 H_2가 한 지점에서 교차한다
: 이 경우, 교차 지점의 rating 값들이 사용될 수 있다.
3] H_1과 H_2가 t차원 하이퍼플레인에서 교차하며, t>=1이다
: t차원의 하이퍼플레인에 가능한 한 가까운 모든 rating이 발견된다. 각각에 상응하는 유저의 rating의 평균 값이 반환된다.

이 접근은 잠재 요인과 NN 방식을 결합하는 것임을 주목하라. NN 방식과의 주된 차이점은 neighborhood가 잠재 요인 모델에서의 피드백을 사용하여 좀 더 정제된 형태로 발견된다는 것이다.

5) 직교성은 기하적 해석 가능성의 관점에서 상당한 장점을 갖는다. out-of-sample 추천을 발견하는 능력은 그러한 장점 중 하나임.

Chapter 3.6.5.4 Example of Singular Value Decomposition

1) 먼저 행의 평균으로 결측 엔트리를 채워준다
2) rank-k truncated SVD를 실행하고, 대각행렬 sigma를 유저 요인에 흡수하여 2개의 matrix 곱으로 표현한다.

이 자체로도 수용가능한 추정값이 나옴. 그러나 mean-filled 되어 있으므로, 편향이 존재함을 감안해야 한다.

3) 이러한 한계를 개선하기 위해, SVD 실행 전의 원 행렬에 결측 엔트리의 예측 값들을 채워 넣는다.
4) 수렴할 때까지 이러한 과정을 반복함. 이 과정의 끝에 나온 값을 예측 값으로 사용할 수 있다.
5) 이 과정을 수행하기 전에 열/행 기준 평균을 빼어 주는 과정이 선행되기도 함. 이는 대개 더 나은 예측 성능을 낳는 것으로 알려져 있음.
6) 결측치가 많을 경우 특히 초기값에 의해 많은 영향을 받는다. 초기값에 영향을 받아 global optimum에 도달하지 못하는 경우가 많이 존재함.
   이러한 초기값의 더 나은 설정을 위해 NN 기반 접근법을 사용하기도 한다. 이는 더 빠른 수렴과 더 나은 예측 성능을 돕는다.

7) “Furthermore, one could easily apply this entire process with regularized singular value decomposition of the filled-in matrices. The main difference is that each iteration uses regularized singular value decomposition of the current matrix, which is filled in with the estimated values. The work in [541] may be used as the relevant subroutine for regularized singular value decomposition.”

Chapter 3.6.6 Non-negative Matrix Factorization

1) rating 행렬이 non-negative일 때 사용함.
2) 이 방식의 주된 이점은 유저-아이템 interaction에 대한 높은 해석 가능성이고, 그 다음으로 정확도가 따라옴.
3) 가장 주된 차이점은 U와 V가 non-negative라는 점임. Frobenius norm을 손실함수로 갖는데, 여기에 U>=0, V>=0이라는 제약조건이 주어짐.
   특히 상품의 선호(긍정 평가)를 반영할 방법은 있으나 좋아하지 않음을 수집할 방법이 없는 상황에서 더 탁월한 해석가능성을 제공함.
   → implicit feedback dataset이라고 부름.
4) 상품 구매 정보와 같은 데이터가 대표적인 예시임. 상품의 구매를 선호로 어느 정도 간주할 수 있으나 그 반대의 경우는 잡아내지 못함.
   implicit feedback의 숫자는 confidence를 의미하고,
   explicit feedback의 숫자는 preference를 의미한다고 해석할 수 있음.

5) implicit feedback 상황은 분류 혹은 회귀문제에서의 positive-unlabeled (PU) learning problem으로 간주할 수 있음.
   → 일반적으로 분류 및 회귀 문제에서는 negative class로 간주함으로써 해결할 수 있음.
   → 따라서 Matrix Completion 상황에서도 특정되지 않은 엔트리를 결측값 대신 0으로 간주할 수 있음.
   → 이러한 상황을 one class Collaborative Filtering이라고 부르기도 함.

6) 최근 0을 채워 넣는 접근 방식으로 인한 편향에 대해 부정 의견이 있으나, 충분히 많은 연구에서 이 방식의 강건성을 증명했음.

"Although some recent works argue that the missing values should not be set to 0 in such cases [260, 457, 467, 468] to reduce bias, a considerable amount of work in the literature shows that reasonably robust solutions can be obtained bytreating the missing entries as 0 in the modeling process."
→ 특히, 엔트리가 0이 될 사전확률이 몹시 높을 때 타당함.
→ 예를 들어 구매하지 않는 상품이 대부분일 슈퍼마켓의 경우를 생각해보면, 결측값을 0으로 설정하는 접근은 적은 편향을 낳게될 것임.
그러나 결측치로 간주한다면 풀이의 복잡도를 몹시 크게 만들 것임.
불필요한 복잡도는 항상 과적합으로 귀결되고, 이러한 효과는 특히 데이터셋이 작을 때 더 큰 영향을 끼친다.

7) NMF의 제약된 최적화 식은, 라그랑지안 완화법과 같은 표준적인 해법을 통해 해결될 수 있음.

8) 학습 과정에서, U와 V는 동시에 업데이트됨.