Matrix Factorization Techniques for Recommender System

Status | Papers

https://datajobs.com/data-science-repo/Recommender-Systems-%5BNetflix%5D.pdf

MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS 리뷰

Yehuda Koren, ***Yahoo Research***

Robert Bell and Chris Volinsky, ***AT&T Labs-Research***

Abstract

Netflix Prize competition에서 입증되었듯이 matrix factorization 모델은 제품 추천에 있어 고전적인 nearest-neighbor 기법보다 뛰어난 성능을 보입니다. **Matrix factorization**은 암묵적 피드백, 시간적 효과, 유의 수준과 같은 추가적인 정보의 결합을 허용합니다.

기업 입장에서 소비자에게 가장 적합한 제품을 추천한다면 만족도와 충성도에 큰 향상을 불러오기 때문에 retailers는 recommender system에 대한 관심이 나날이 커져왔습니다.

추천 시스템은 특히 영화, 음악, 티비쇼와 같은 엔터테인먼트 상품에 유용합니다. 소비자들은 특정 상품에 만족도를 표시하고자 하는 것으로 입증되었기 때문에 어떤 상품이 어떤 고객에게 appeal하는지에 대한 방대한 양의 data가 있습니다. 기업은 이 data를 분석해 특정 소비자에게 특정 상품을 추천할 수 있습니다.

Recommender System Strategies

추천 시스템은 주로 두 가지 방법론으로 설명되어 왔습니다.

첫째는 **Content filtering approach**이며, 둘째는 **Collaborative filtering approach**입니다.

Content filtering은 user나 Item에 대한 profile을 먼저 생성합니다. 예를 들어, user의 인구통계학적 사실이나 item의 genre나 category를 이야기합니다. 그리고 작성된 profile을 기반으로 고객에게 추천을 진행합니다.

Collaborative filtering은 사전 생성된 profile이 없다는 전제로, user와 item에 관해 주어진 data로 추천을 진행합니다. User 간의 관계와 item 간의 상호 의존성을 분석하며 새로운 user - item 연관을 식별합니다. 이때, **Neighborhood method**와 **Latent factor models**가 이용됩니다.

Neighborhood method는 이름에서 알 수 있듯, user 혹은 item 사이의 관계를 계산하고 거리로 환산해 서로 가까운 것을 추천하는 방법론입니다. Latent factor method는 user가 item에 부여하는 rating에 pattern이 있다는 것을 전제로 그 pattern을 찾아 하나의 factor로 추출해 facator 점수를 중심으로 거리를 추정해 추천하는 방법론입니다.

Matrix Factorization methods

몇몇 가장 성공적인 latent factor model은 matrix factorization에 기반하고 있습니다.

추천 시스템은 보통 ‘user’와 ‘items of interest’ 두 차원의 행렬을 기초로 작용합니다. 가장 품질이 좋은 interest data는 user가 item에 직접 평가를 남긴 explicit feedback이지만 이 경우, sparse matrix가 구성된다는 희소성 문제가 존재합니다.

Matrix Factorization의 큰 강점은 explicit feedback에 추가적인 정보의 결합을 허용한다는 점입니다. 구매 이력, 검색 기록, 검색 패턴 등 implicit feedback을 추론하며 기존의 sparsity issue를 해결하도록 도와줍니다.

A Basic Matrix Factorization Model

Basic matrix factorization은 SVD(특이값 분해)와 아주 깊은 관련이 있습니다. 하지만 평점 행렬처럼 missing value가 많은 경우, 예측에 실패하거나 overfitting하기 때문에 위험합니다. 기존 방식은 빈 칸에 imputation을 적용하지만 data가 많아지면 매우 비효율적인 방법이 될 수 있고 imputation이 부정확할 경우, data가 왜곡될 가능성이 있습니다.

따라서, 최근 대부분의 경우 정규화된 모델을 통해 overfitting을 피하곤 합니다.

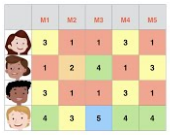
$$\min_{q^*,p^*} \sum_{(u,i)\in K} (r_{ui} - q_i^T p_u)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2)$$

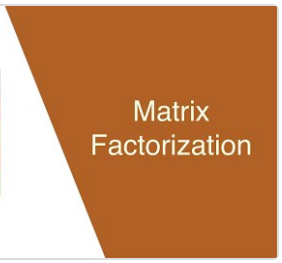
λ 는 regularization의 정도를 조절하는 상수이며, 해당 모델에서 factor vector을 학습시키기 위해 error를 최소화하는 것을 목표로 합니다.

How does Netflix recommend movies? Matrix Factorization

Announcement: New Book by Luis Serrano! Grokking Machine Learning. bit.ly/grokkingML40% discount code: serranoytA friendly introduction to recommender system...

https://www.youtube.com/watch?v=ZspR5PZemcs





Learning Algorithms

Stochastic gradient descent (확률적 경사하강법)

$$e_{ui} \overset{def}{=} r_{ui} - q_i^T p_u$$

- $q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$
- $p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$

r_{ui}를 예측하고 예측 error를 계산합니다. 그리고 parameters를 γ 만큼 반대 방향으로 수정합니다. 확률적 경사하강법을 통해 비교적 빠르고 효율적인 계산을 합니다. 하지만, 가끔의 경우, altering least squares가 더 효율적이기도 합니다.

Altering least squares

위 식에서 q_i와 p_u는 둘 다 모르는 parameter이기 때문에 하나를 fix하면 이 optimization problem은 2차식으로 해결할 수 있습니다. Altering least squares는 q_i와 p_u를 번갈아가며 fix하고 modify하는 방법으로 보통은 stochastic gradient descent가 더 시간효율적이지만 다음과 같은 경우에는 altering least squares가 우월합니다.

Adding Biases

Collaborative filtering(협업 필터링)과 비교했을 때, matrix factorization의 장점은 다양한 data와 다른 application 특화 요구사항을 반영한다는 유연성이 있다는 점입니다.

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

대부분 관찰되는 변량은 biases 또는 intercepts와 관련이 있습니다. 예로, 어떤 user는 다른 users보다 더 높은 점수를 주는 경향이 있고 어떤 item은 다른 items보다 좋은 rating을 받는 경향이 있습니다만 이는 평향을 만들어내는 결과를 불러왔습니다. 따라서, $q_i{}^T p_u$로만 전체 rating을 설명하는 것은 현명한 방법이라고 볼 수 없습니다. 위 equation은 global average, item bias, user bias와 user-item interaction으로 쪼개어졌으며 이는 Squared error function minimizing을 통해 학습합니다.

$$\min_{p^*,q^*,b^*} \sum_{(u,i)\in\kappa} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda$$
$$(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2) \qquad (5)$$

Biases는 observed signal을 포착하기 때문에 정확한 모델링은 필수적이고 중요합니다.

Additional Input Sources

추천 시스템은 간혹 cold start problem에 처합니다. 이를 완화하기 위해 user와 관련한 additional sources를 결합할 수 있습니다. 추천 시스템 설계자는 user의 explicit ratings와 관계 없이 구매 이력, 탐색 기록 등 implicit feedback을 이용해 cold start problem을 해결할 수 있으며 이때, user 표현이 더 강화되어 정확해진 것을 알 수 있습니다.

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T \left[p_u + |N(u)|^{-0.5} \sum_{i \in N(u)} x_i + \sum_{a \in A(u)} y_a \right] \quad (6)$$

Temporal Dynamics

지금까지, 위의 equation은 모두 정적인 환경을 가정하고 있습니다. 현실에서는 제품에 대한 인식과 인기도는 항상 변하며 새로운 선택지도 등장하곤 합니다.

$$\hat{r}_{ui}(t) = \mu + b_i(t) + b_u(t) + q_i^T p_u(t) \quad (7)$$

따라서, 위와 같이 (t)를 적용해 시간 함수를 표현합니다.

Item biases $b_i(t)$: Item popularity의 변화를 (t)로 표현

User biases $b_u(t)$: Natural drift in a user's rating을 (t)로 표현

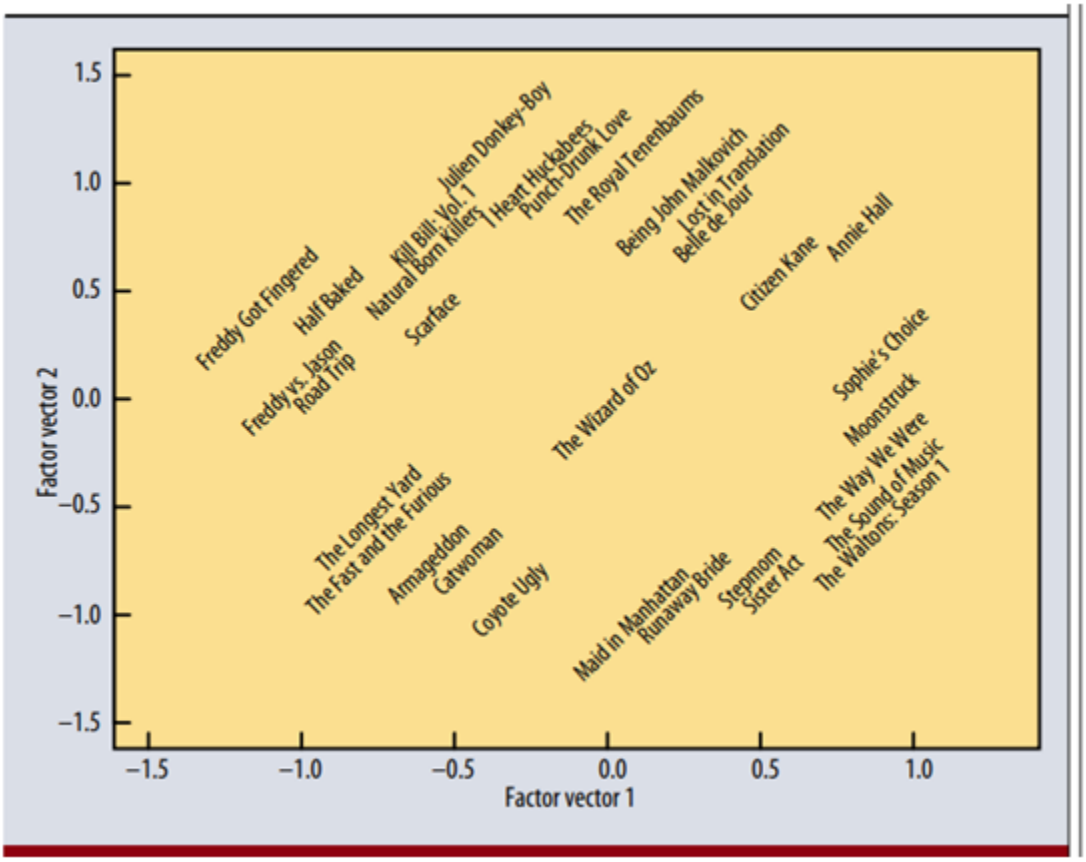
User preferences $p_u(t)$: User와 Item의 상호작용이므로 user preference도 시간에 따라 변화하는 함수로 표현

Inputs with Varying Confidence Levels

시간이 그랬듯 유의 수준도 항상 변화하기 마련입니다. 큰 규모의 광고로 인해 단기적 효과를 누리는 item이 있을 수도 있으며, users가 고의적으로 rating을 높거나 낮게 매기는 문제가 발생할 수도 있습니다. 또한, implicit feedback을 정확도를 올리기 위한 수단으로 활용했으나 그 자체에 문제가 있을 수 있습니다. 따라서, 유의 수준도 모델링에 반영하는 것이 바람직합니다.

$$\min_{p^*,q^*,b^*} \sum_{(u,i)\in\kappa} c_{ui}(r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2) \quad (8)$$

Netflix Prize Competition



2006년에 실시한 Netflix Prize Competiton입니다.

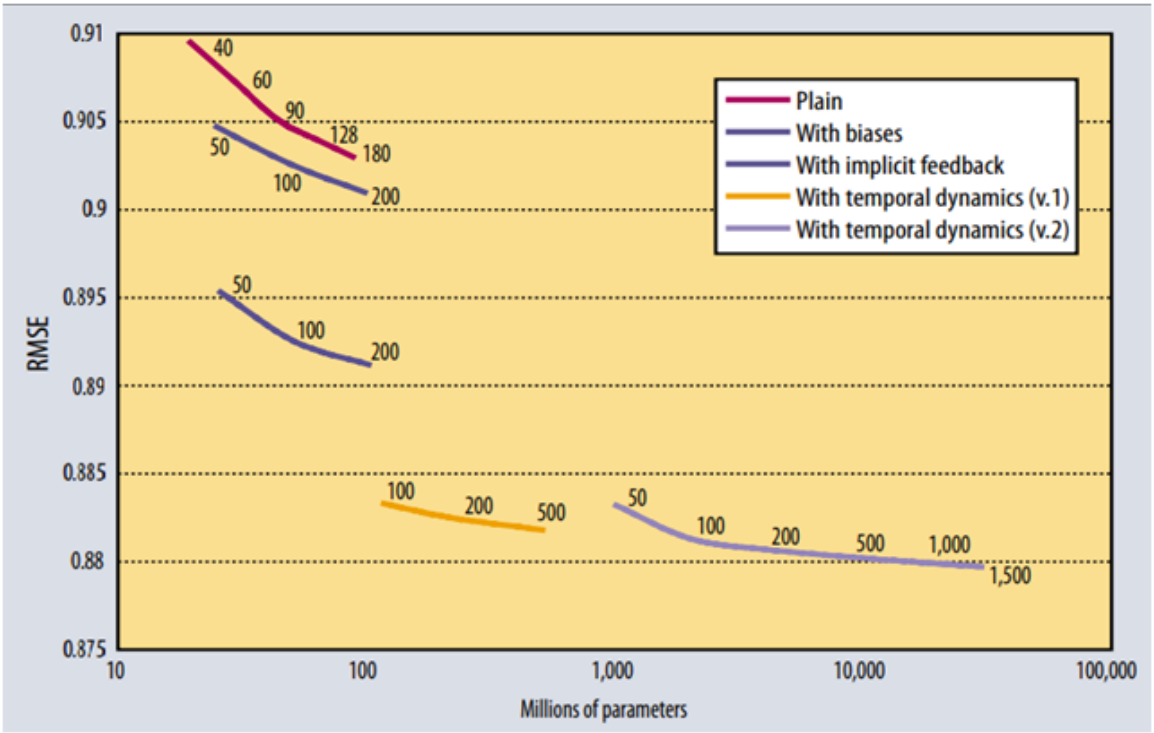
참가 팀은 500,000명 이상의 고객이 17,000개 영화에 ratings을 부여한 1억 건 이상의 dataset을 이용해 test set 300만 건에 대한 rating 예측치를 제출하면 Netflix는 이의 RMSE를 계산합니다.

10% 이상 성능 개선을 이뤄내면 $ 1 million을, 아무도 10%에 달하는 성과를 내지 못하면 1등에게 $50,000 상금을 수여한다는 조건으로 실시합니다.

본 competiton은 182개 국가에서 28,000 팀이 참가했습니다.

본 논문의 저자는 matrix factorization으로 1등을 했는데, 이때 factor vector를 활용했습니다.

Matrix의 x축은 관객 선호을 기준으로, y축은 영화의 성질을 기준으로 분류했습니다. 그 결과, lowbrow 영화와 independent 영화가 가깝게 positioning 됐고, serious한 영화와 mainstream 영화가 가깝게 positioning 됐습니다.



Modeling 성과를 분석하자면, parameters가 많으면 많고, 복잡하면 복잡할수록 accuracy가 증가했습니다. 특히, temporal component가 중요하게 나타났습니다.

Summarize

Matrix factorization은 collaborative filtering에서 dominant한 방법론으로 자리잡았습니다. Netflix Prize dataset에 대한 MF의 성과는 고전적인 kNN 방법보다 우수한 정확도를 제공한다는 것을 입증했습니다. 동시에 추천 시스템이 비교적 쉽게 학습할 수 있는 작은 메모리 효율적인 모델을 제공하며 이러한 기술을 더욱 편리하게 만드는 것은 모델이 여러 형태의 feedback을 자연스럽게 통합할 수 있기 때문입니다.