Ch6 Ensemble-based and Hybrid Recommender Systems

6.8 Feature Combination Hybrids

- 이 hybrids의 아이디어는 다양한 sources(e.g., content and collaborative)들을 합쳐서 하나의 representation으로 하여 input data를 만드는 것입니다.

6.8.1 Regression and Matrix Factorization

- R 을 m x n implicit feedback ratings matix 라 하고, C 를 d x n content matrix라 하겠습니다.
- 각각의 item들은 non-negative frequencies of d words로 표현되어집니다.
  ("즉 content에서 item features를 vocab에서 등장한 단어의 수를 vector로 나타내고 있습니다.")
- R은 implicit feedback matrix이므로 missing entries는 0으로 가정하겠습니다.
- W 는 n xn item-item coefficient matix로 rating 은 $\hat{R} = RW$ 로 예측되어집니다.
- 또한 rating은 $\hat{R} = CW$ 로도 예측이 가능합니다. 그러므로 optimization model은 다음과 같이 정의할 수 있습니다.

$$Minimize\ J = ||R - RW||^2 + \beta \cdot ||R - CW||^2 + \lambda||W||^2 + \lambda_1 \cdot ||W||_1$$
$$subject\ to:$$
$$W \geq 0$$
$$Diagonal(W) = 0$$

  - Non-negativity / diagonal 제약을 부여했습니다.
  - Elastic-net regularization을 추가하였습니다.
  - β 는 tuning을 통해 결정됩니다.
- Rating은 $\hat{R} = RW$ 또는 $\hat{R} = CW$ 로 예측 가능하지만, RW 만 사용됩니다.
- 그러므로 $||R - CW||^2$ term은 objective function을 refine해주는 추가적인 regularizer 역할만 하게됩니다.
- 즉 이런 추가적인 term으로 얻고자 하는 것은 아직 알지 못 하는 user의 action을 예측하는데 있어서 모델의 generalization power를 키우는 것입니다.
- 이러한 접근 방법으로 어떤 collaborative filtering model이든 content-based methods와 결합될 수 있습니다.
- 예를 들어 matrix factorization의 경우
  - user factor matrix U $\in \mathbb{R}^{m \times k}$
  - item factor matrix V $\in \mathbb{R}^{n \times k}$
  - content factor matrix Z $\in \mathbb{R}^{d \times k}$

$$minimize\ J = ||R - UV^T||^2 + \beta \cdot ||C - ZV^T||^2 + \lambda(||U||^2 + ||V||^2 + ||Z||^2)$$

  - item factor matrix V 는 factorizations of the ratings matrix와 content matrix에 공유됩니다.

6.8.2 Meta-level Features

- 꼭 content와 collaborative filters에서 feature combination을 해야할 필요는 없습니다.
- 새로운 meta-features는 특정 recommender에서 추출될 수 있고 ensemble model에 결합되어 사용될 수 있습니다.
- 예를 들어, rating matrix에서 uesrs와 items에 의해 주어진 rating 개수는 meta-level features가 될 수 있습니다. 어떤 user가 많은 영화에 rating을 줬다면 또는 어떤 영화가 많은 사람들로 부터 rating을 받았다면, 이 요인은 다양한 알고리즘에 다양한 방식으로 정확도에 영향을 미칠 것입니다.

- meta-level features의 기본적인 idea는 model combination process에서 meta-features를 가지고 entry별 차이점을 설명할 수 있다는 것입니다.
- 이 장에서는 collaborative filtering algoritms에서 meta-level features를 사용하는 방법을 설명합니다.
- 구체적으로는 *feature weighted linear stacking* methods에 대하여 설명합니다. 이는 meta-level features를 stacking methods와 결합하여 사용하는 방법입니다.
- 사용된 meta features의 subset은 다음과 같습니다.

Id.	Description
1	Constant value of 1 (using only this feature amounts to using the global linear regression model of section 6.3)
2	A binary variable indicating whether the user rated more than 3 movies on this particular date
3	The log of the number of times a movie has been rated
4	The log of the number of distinct dates on which a user has rated movies
5	A Bayesian estimate of the mean rating of the movie after having subtracted out the user's Bayesian estimated mean
6	The log of the number of user ratings
16	The standard deviation of the user ratings
17	The standard deviation of the movie ratings
18	The log of (Rating Date − First User Rating Date +1)
19	The log of the number of user ratings on the date +1

  - Netflix Prize data set에 사용된 stacking process에 적용된 features입니다.
- 총 l 개의 (numeric) meta-features가 있다고 가정하겠습니다. => $z_1^{ut}, \ldots, z_l^{ut}$
- Meta-features는 각 entry (u,t)에 따라 구체적인 값을 가집니다. 물론 어떤 feature는 u나 t가 다양한 값을 가지더라도 같은 값을 가질 수도 있습니다. 예를 들어 위 table에서 Id 3.의 경우 u에 따라서는 다양하지 못 하지만 v에 따라서 다양한 값을 가질 것입니다.
- 총 q 개의 recommendation methods가 있다고 해봅시다. $w_1, \ldots, w_q$ 의 weights들이 결합되어 rating을 예측할 것입니다. 즉 entry (u,t) 가 주어졌을때, rating은 다음과 같이 계산됩니다.

$$\hat{r}_{ut} = \sum_{i=1}^{q} w_i \hat{r}_{ut}^i$$

  - methods 별로 entry에 대하여 예측한 값에 methods별 가중치를 곱하여 linear combination 했습니다.
- 앞서 6.3절에서 linear regression model로 일정 비율의 data를 holding out하고 weights를 학습했습니다.

("Held-out entries는 train? Valid?")

- 이제 여기에 meta-features를 사용해서 성능을 더 올려보겠습니다.
- main idea는 "linear regression weights는 개별 entry에 대하여 구체화될 수 있고, 이 가중치들은 meta-features의 linear functions이 된다." 입니다.
- 먼저 각 entry(u,t) 마다 가중치가 적용된 model은 다음과 같습니다.

$$\hat{r}_{ut} = \sum_{i=1}^{q} w_i^{ut} \hat{r}_{ut}^i$$

  - 이 model은 기존의 모델보다 더 정교한 값을 예측할 수 있습니다. 왜냐하면 개별 모델들이 예측한 entry 하나하나에 가중치가 결합되기 때문입니다.

    - 하지만 이 모델의 문제는 파라미터의 수 입니다. 가중치의 총 파라미터 수는 (m x n x q) 가 됩니다. 관측된 ratings(data) 보다 파라미터가 더 많기 때문에 overfitting이 발생할 것입니다.
    - 그래서 weights를 meta-features의 linear combinaitons로 가정합니다.
      (meta-features는 미리 계산된 실수 값이기 때문에 파라미터 수에 포함되지 않습니다.)

- Meta-features는 개별 uesr-item combinations를 위해 다양한 models의 상대적인 중요도를 조절합니다.
- 여기서 파라미터 v_{ij} 를 도입하여 i-th model에서 j-th meta-feature의 중요도를 조절합니다.

$$w_i^{ut} = \sum_{j=1}^{l} v_{ij} z_i^{ut}$$

    - v_{ij} 는 j-th meta-feature가 i-th model에서 상대적으로 얼마나 중요한지를 나타냅니다.
- 위 내용을 결합하여 최종 예측 rating을 구하는 식은 다음과 같습니다.

$$\hat{r}_{ut} = \sum_{i=1}^{q} \sum_{j=1}^{l} v_{ij} z_j^{ut} \hat{r}_{ut}^i$$

    - 여기서 idependent variables는 $z_j^{ut} \hat{r}_{ut}^i$ 입니다.
    - held-out ratings에서 v_{ij} 를 학습할 수 있습니다.
    - 규제항을 넣어 overfitting을 줄일 수 있습니다.
    - lienar regression으로 weights가 학습되고 나면 개별 component models는 전체 데이터셋에 대하여 재학습을 합니다. 이때 held-out entries로 학습된 weights가 q models에 적용되어 사용됩니다.

6.9 Mixed Hybrids

- mixed hybrids의 main 특징은 여러 시스템의 score를 presentation 측면에서 결합한다는 것입니다. 이는 기존에 Rating score를 결합던 것과 차이가 있습니다.
- 대표적인 예로 개인화된 television listing이 있습니다. 이는 *composite* programs를 user에게 보여주는 것입니다. 이 composite program은 다른 시스템들에 의해 추천된 items의 결합으로 만들어 집니다.
- 이 방법은 개별 아이템을 추천하는 것에는 큰 의미는 없고, 상대적으로 복잡한 아이템 구성을 추천하기 위해서 사용됩니다.
- 한 가지 예를 들면, tourism domain에서 사용될 수 있습니다. 이 경우 다양한 여러 아이템 카테고리에서 추천이 이루어집니다.
- tourism recommender systems에는 숙박, 레져 활동, 항공권 등 여러가지 카테고리가 있습니다. 여행객은 여러 카테고리의 items를 한 묶음으로 구매하게 됩니다.
- 각각의 카테고리에 다른 추천 시스템이 적용됩니다. 가장 기본적인 idea는 예를 들어 숙박에 최적화된 추천이 tourism activites에는 적합하지 않을 수 있다는 것입니다.
- 그래서 각각의 카테고리에 따라 다른 관점에 맞는 추천 시스템을 적용해야합니다. 이때 중요한 것은 추천 bunddles가 여러 카테고리에 있어서 서로 모순적이면 안됩니다.

> It is important to recommend bundles in which the items from multiple categories are not mutually inconsistent.

- 예를 들어, 추천된 레져 엑티비티가 추천된 식당에서 매우 멀리 떨어져있으면 전체적인 추천 bundle에 대한 만족도는 떨어질 것입니다.
- 그래서 도메인에따른 제약을 가질 수 있는 knowledge base 시스템이 많이 활용됩니다. 이는 제품간 어울리지 않는 조합들을 걸러줄 수 있습니다.

- 정리하자면 Mixed hybrids는 일반적으로 복잡한 제품 도메인에서 사용되며, knowledge-based recommender systems등 다양한 components 들로 디자인 됩니다.

6.10 Summary

- Hybrid recommender systems는 다양한 데이터 소스의 활용을 높이거나 특정 데이터 양식에 존재하는 추천 시스템의 퍼포먼스를 높이기 위해 개발되었습니다.
- 다양한 추천 시스템들은 각각의 장단점이 있고 hybrid recommender systems은 개별 시스템들의 장점을 잘 활용하고자 합니다.
- 또한 Ensemble methods는 collaborative filtering methodsd의 성능을 높여주었습니다. 이때 다른 components가 적용되는데 사용되는 data가 동일한 ratings matrix입니다.
- Classification에서 bias-variance trade-off에 존재하는 많은 이론적 결과들이 collaborative filtering applicaitons에서도 동일하게 적용되었습니다. 따라서 bagging, boosting과 같은 기술들이 약간의 수정으로 적용될 수 있었습니다.
- Hybrid systems는 monolithic systems, ensemble systems, 또는 mixed systems로 디자인될 수 있습니다.
- Ensemble systems는 구체적으로 recommender의 sequential 또는 parallel arrangement로 디자인될 수 있습니다.
- Monolithic design에서 추천 시스템은 기존의 존재하는 recommenders를 수정하거나, 다양한 데이터 양식의 featrues의 결합으로 만들어진 새로운 recommenders가 있습니다.
- Mixed systems에서 다수의 엔진으로 부터 추천이 동시에 보여집니다.
- 많은 경우에 Meta-features 는 특정 데이터 양식에서 추출될 수 있고, entry-specific 방식으로 다양한 recommenders의 예측을 통합하는 것에 사용될 수 있습니다.
- Hybrids와 ensemble systems의 가장 큰 강점은 다양한 시스템의 장점들을 보완해줄 수 있다는 것입니다.