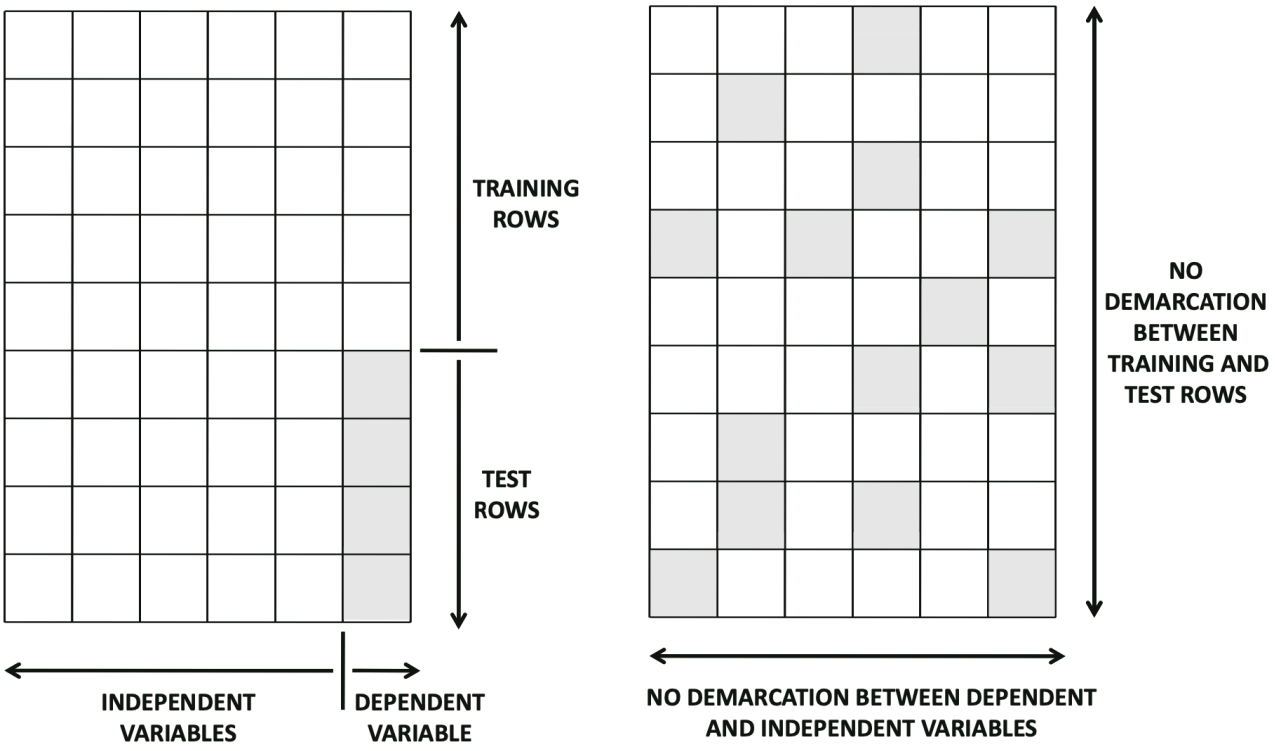
Ensemble-Based and Hybrid Recommender Systems

6.2 Ensemble Methods from the classification Perspective

- Ensemble methods는 일반적으로 data classification에서 learning algorithms의 robustness를 강화하기 위해서 사용됩니다.
- Collaborative filtering은 data classification 문제의 일반화된 형태로 볼 수 있습니다.



(a) Classification (b) Collaborative filtering

Figure 6.3: Revisiting Figure 1.4 of Chapter 1. Comparing the traditional classification problem with collaborative filtering. Shaded entries are missing and need to be predicted.

- (a) 는 classification에서 feature variables와 class variables가 깔끔하게 분리되어 있는 것을 보여주고 있습니다.
- Collaborative filtering이 Classification과 다른 가장 큰 이유는 feature variables와 class variables의 구분이 있지 않습니다. 그리고 missing value가 어떤 columns와 rows에서 나타날 수 있습니다.
- Classification에서의 bias-variance theory가 recommender systems에 적용될 수 있을까요?
- multiple collaborative recommender systems를 결합하면 더 정확한 결과를 얻을 수도 있습니다. 하지만 missing entires는 어떤 row 또는 columns에서 나타날 수 있기 때문에, collaborative filtering에서 ensemble algorithms의 성능을 일반화 시키는 것은 쉽지 않은 문제입니다.
- Classifier의 error는 다음 세 가지 components로 나누어 볼 수 있습니다.
    1. Bias : bias가 클 수록 실제 데이터 분포에 대한 결정 경계를 찾지 못 하고 모델 알고리즘이 가지고 있는 임의의 결정 경계로 분류를 하게 됩니다. 예를 들어 선형 분류기가 있다고 하면 이 선형 분류기는 선형의 decision boundary를 가집니다. 하지만 대부분의 실제 두 클래스를 가지는 데이터는 선형으로 구분되지 않을 것 입니다. 즉 이 선형 분류기는 bias를 가지고 있다고 할 수 있습니다.
    2. Variance : 모델은 학습 데이터에 의해 달라질 수 있습니다. 모델의 variance 는 overfitting과 연관이 되어 있습니

다. classificer가 overfitting의 경향을 보인다는 것은 다른 training data sets으로 학습을 하면 동일한 test instance에 대해서 일관되지 않는 예측을 하는 것입니다.

3. Noise : noise는 target class labeling이 가지고 있는 본연의 errors 입니다. 이 error를 줄일 수 있는 방법은 거의 없습니다. 때문에 ensemble methods는 bias나 variacne를 줄이는데에 포커스를 맞추고 있습니다.

- test instances에 대한 MSE는 다음과 같이 표현할 수 있습니다.

  $$Error = Bias^2 + Variance + Noise$$

  즉 bias와 variance를 줄이는 것이 전체 error를 줄이는 것과 같습니다. 예를 들어, ensemble methods 중 *bagging* 은 variacne를 줄이고, *boosting* 은 bias를 줄여줍니다.

6.3 Weighted Hybrids

- R = [r_{uj}] 는 m x n rating matrix라 하겠습니다.
- Weighted hybrids에서는 다양한 recommender systems의 예측 결과를 weight를 사용해서 combine하여 최종 output를 만들게 됩니다.
- $\hat{R}_1, \cdots, \hat{R}_q$ 를 *completely specified* rating matrices라고 하겠습니다.

$$\hat{R} = \sum_{i=1}^{R} \alpha_i \hat{R}_i$$

- 모든 α를 1/q 로 단순하게 할 수 도 있지만, 좀 더 정교한 output을 위해서 α 를 조정해야할 것입니다.
- 개별 rating은 아래와 같이 계산할 수 있습니다.

$$\hat{r}_{uj} = \sum_{i=1}^{q} \alpha_i \hat{r}_{uj}^i$$

- 최적의 weighte를 찾기 위해서 특정 combination이 얼마나 효과적인지 평가할 수 있어야합니다. 자세한 내용은 다음 챕터에서 소개한다고 합니다. 여기서는 간단한 방법을 소개합니다.
  - 먼저 주어진 rating matrix(R)에서 작은 비율(25%)을 hold out(*H*)합니다. 그리고 R에 남아있는 entries(75%)에 대하여 q 개의 다른 알고리즘으로 예측을 하여 $\hat{R}_1, \ldots, \hat{R}_q$ matrices를 얻습니다. 이 matrices들로 위 식을 이용해서 최종 ensembel-based prediction $\hat{R}$ 을 계산하게 됩니다.
  - mse나 mae를 이용하여 특정 weight combination을 평가할 수 있습니다.

$$MSE(\overline{\alpha}) = \frac{\sum_{u,j \in H} (\hat{r}_{uj} - r_{uj})^2}{|H|}$$

$$MAE(\overline{\alpha}) = \frac{\sum_{u,j \in H} |\hat{r}_{uj} - r_{uj}|}{|H|}$$

  - 어떻게 하면 위 식을 최소화 하는 최적의 α 값들을 찾을 수 있을까요? MSE에서 가장 간단한 방법은 linear regression을 사용하는 것입니다.
  - Held-out set *H* 의 ratings이 dependent variable의 ground truth values를 제공하고 파라미터 $\alpha_1, \ldots, \alpha_q$ 는 independent variables로 가정하겠습니다.
  - 기본적인 idea는 held-out set에서 주어진 rating에 대하여 MSE가 최소가 되는 independent variables를 찾는 것입니다.

    - 여기서는 independent variablessms entry (u,j) 에 대하여 다양한 모델의 rating 예측 결과에 대응됩니다. 그리고 dependent variable 는 held-out set H에서 ensemble combination의 예측된 rating $\hat{r}_{uj}$ 값에 대응됩니다.

      {α 를 찾는 거에 held-out set H 가 사용된다는 뜻?}
    - regression coefficients는 다양한 component models 의 weights에 대응됩니다.
- Linear regression approach 는 noise나 outliers에 민감합니다.
- 다양한 robust regression methods가 가능합니다.
- MAE가 MSE보다는 더 robust하다고 알려져 있습니다. 왜냐하면, 큰 에러를 더 강조하지 않기 때문입니다.
- 최적의 parameter vector ($\alpha_1, \ldots, \alpha_q$) 를 찾는 방법으로 gradient descent를 사용할 수 있습니다.

$$\frac{\partial MAE(\overline{\alpha})}{\partial \alpha_i} = \frac{\sum_{u,j \in H} \frac{\partial |\hat{r}_{uj} - r_{uj}|}{\partial \alpha_i}}{|H|}$$

$$\frac{\partial MAE(\overline{\alpha})}{\partial \alpha_i} = \frac{\sum_{u,j \in H} sign(\hat{r}_{uj} - r_{uj}) \hat{r}^i_{uj}}{|H|}$$

- 각각의 편미분을 다 모으면,

$$\overline{\nabla MAE} = (\frac{\partial MAE(\overline{\alpha})}{\partial \alpha_1}, \ldots, \frac{\partial MAE(\overline{\alpha})}{\partial \alpha_q})$$

- 이제 iterative gradient descent 을 다음과 같이 수행하며 $\overline{\alpha}$ 를 업데이트 합니다.
  1. Initialize $\overline{\alpha}^{(0)}$ = (1/q, . . . ,1/q) and t = 0
  2. **Iterative Step 1 :** Update $\overline{\alpha}^{(t+1)} \Leftarrow \overline{\alpha}^{(t)} - \gamma \cdot \overline{\nabla MAE}$
  3. **Iterative Step 2 :** Update the iteration index as $t \Leftarrow t + 1$
  4. **Iterative Step 3 (convergence check) :** 만약 MAE가 최소 양 이상 개선되었다면 다시 1단계로 돌아갑니다.
  5. Report $\overline{\alpha}^{(t)}$
- Regularization은 overfitting을 막기 위해 추가 될 수 있습니다. 또한 α_i 에 non-negativity 나 sum to 1 같은 제약을 걸어 줄 수도 있습니다. 이러한 제한은 unseen entries에 대하여 일반화를 더 잘 할 수 있습니다.
- optimal weights가 결정되고 나면, 모든 ensemble models는 전체 rating matirx 에 대하여 재학습을 하게됩니다. 여기서는 held-out entries가 따로 사용되지 않습니다.
- 또 다른 파라미터 searches 방법이 있습니다. 더 간단한 방법은 held-out set에서 선정한 몇 가지 파라미터 조합을 시도하는 것입니다. 예를 들어, 다른 constant는 고정하고 다른 값들을 조정 해볼 수 있습니다.
- 이 방법들은 meta-level content features에서 더 강화됩니다. 6.8.2에서 더 자세히 살펴볼 것입니다.
- 다른 components에 다른 weight를 주는 것은 예측 값의 scales가 다르거나, 몇몇 ensemble compents가 더 정확한 예측을 보일때 특히 중요합니다.