Machine Learning

Ch5. Resampling methods (CV)

김광수

2024년 2학기

Cross-Validation (교차타당검증)

- training error와 test error는 종종 매우 다른 양태를 보임 .
- test error에 대한 정확한 추정을 위해 모형훈련 및 적합을 위한 자료와 평가를 위한 자료를 분리하는 것이 타당.
- 자료의 특성, 크기, 분석자의 여건 등에 따라 다른 방식이 적용될 수 있음.

Validation set approach

- 전체 자료를 랜덤하게 두 그룹으로 분할하여 각각 훈련/평가 자료로 사용.
- 정해진 기준은 없으나 보통 훈련:평가 자료의 비율을 5:5 또는 7:3 정도로 할당.
- test error를 추정하기 위한 가장 간단한 방법.
- 평가 자료는 validation set, hold-out set 등으로 불림.
- 참고로 validation set과 test set은 훈련 과정의 초매개변수 설정이나 모형 선택에서의 검증과 최종 평가를 위한 것으로 구분되기도 함.

Validation set approach

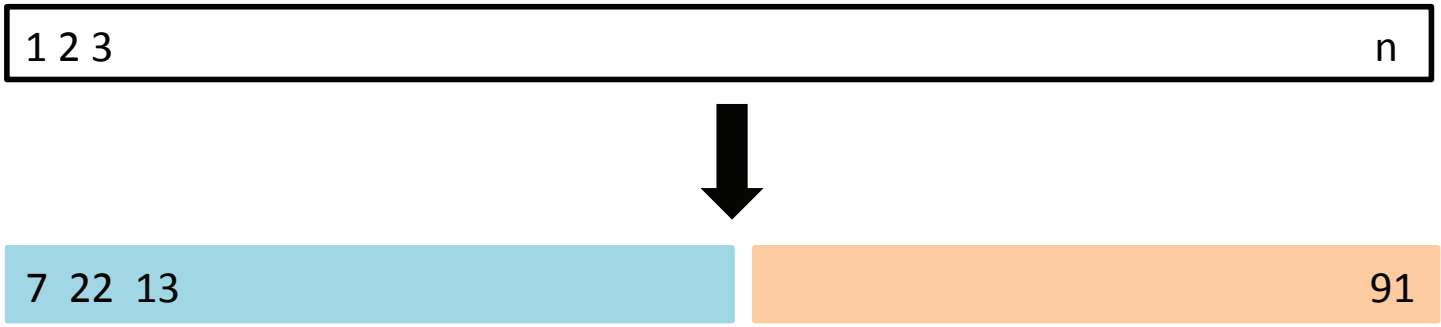

Figure 1: 훈련/평가 자료의 분할

단점

- 랜덤하게 자료를 분할하기 때문에 분할결과에 따라 추정의 변동성이 클 수 있음. 특히, 자료의 크기가 작거나 이상/영향치들이 포함되어 있는 경우에는 더욱 그러함.

- 원 자료의 크기보다 작은 집합의 훈련자료가 모형적합에 사용되기 때문에 test error가 과대추정될 수 있음.

- 훈련 데이터를 쪼개서 훈련/평가로 나누고, 실제 적합에서는 훈련 데이터 전체를 사용함.

test error의 추정

- 다양한 차수의 다항회귀모형에 대한 test error 추정치.
- 오른쪽 그림: 자료의 분할을 10번 반복하여 추정된 test error들의 결과.

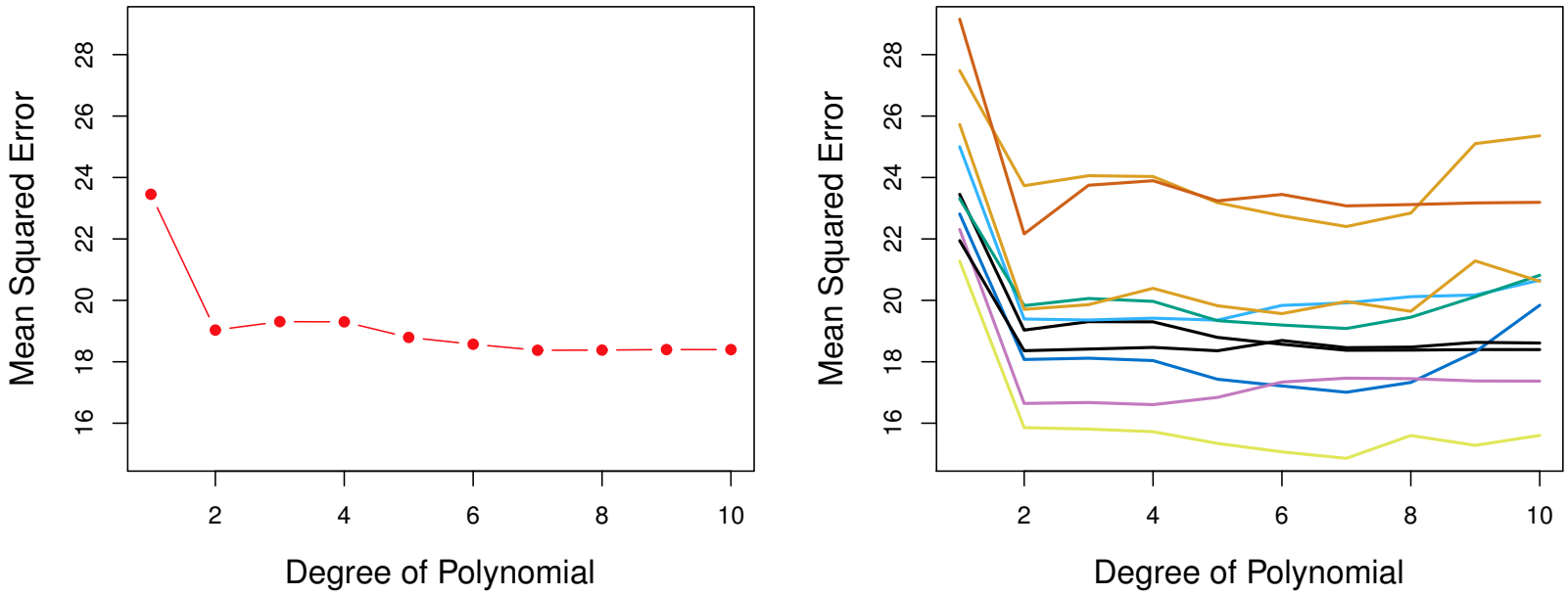

Figure 2: test error의 추정

Leave-One-Out Cross-Validation (LOOCV) (1)

- Validation set approach의 단점을 해소할 수 있는 방법.
- 자료의 크기를 n이라 할 때 $n-1$개의 훈련자료와 1개의 평가자료로 분할.
- (x_i, y_i)가 평가자료라면, (회귀모형의 경우) test error는 다음과 같이 추정.

$$MSE_i = (y_i - \hat{y}_{(i)})^2$$

($\hat{y}_{(i)}$: i번째 관측치를 제외하고 적합된 모형에 의한 예측치)

- 위 과정을 모든 $i = 1, 2, \dots, n$에 대해서 반복.
- test MSE에 대한 LOOCV 추정치는

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} MSE_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_{(i)})^2$$

장단점

- 장점

  - $n-1$개의 자료를 모형적합에 사용하기 때문에 정보량에 손해가 거의 없어 test error의 과대추정에 대한 염려로부터 자유로움.

  - 자료의 분할에 따른 불확실성이 나타나지 않는다 (모든 자료가 똑같이 test set에 한번씩 포함).

- 단점

  - 계산량이 매우 많아 자료의 크기가 매우 크거나 적합모형의 계산에 시간이 많이 소요되는 경우 적용에 제한이 따름.

LOOCV

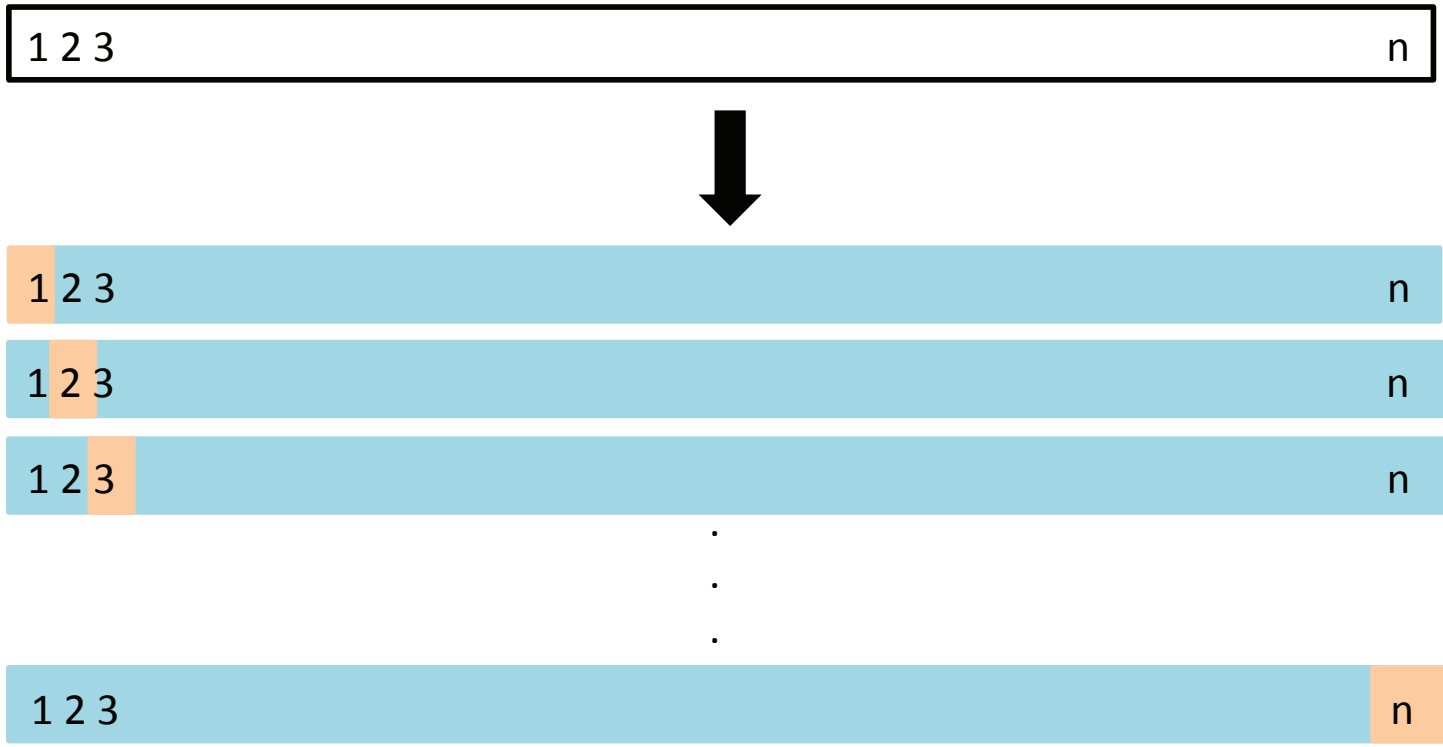

Figure 3: LOOCV

LOOCV의 단순화

- 다항회귀모형의 경우 다음과 같은 단순화가 가능함

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

여기서 h_i는 hat matrix의 i번째 대각원소임

- 위 식을 사용할 경우 한 번의 모형적합을 통해 CV test error를 얻을 수 있음.

- 일반적으로 위와 같은 단순화가 항상 가능하지는 않음. 하지만, LOOCV의 장점을 유지하면서 계산량이 많다는 단점을 해소하기 위해, 위와 비슷한 꼴의 단순화에 대한 연구가 이루어짐. (Generalized Cross-Validation)

k-Fold Cross-Validation (2)

- Validation set approach 대비 LOOCV의 장점은 어느 정도 유지하면서 계산량을 경감하기 위한 시도.

- 전체 자료를 k개의 집합으로 분할한 후 그 중 하나의 집합(i번째)을 평가자료로 설정 ($i = 1, 2, \ldots, k$).

- i번째 평가자료를 이용한 test error 추정치를 MSE_i라 하면, k-fold에 의한 test error 추정치는

$$CV_{(k)} = \sum_{i=1}^{k} MSE_i.$$

- k는 보통 5, 10 등을 사용함.

k-fold CV

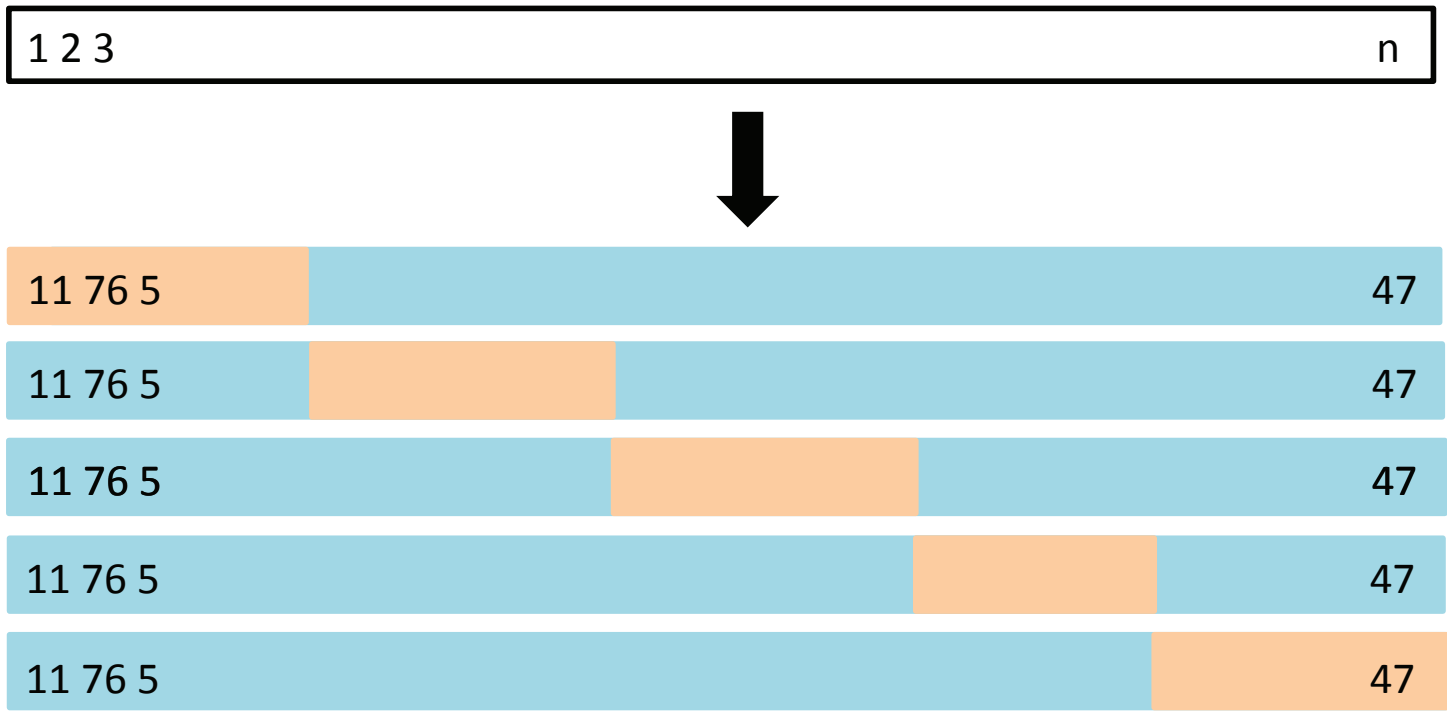

Figure 4: k-fold CV

장단점

- 장점

  - LOOCV에 비해 계산량이 현저히 감소 (k번의 모형적합 필요).

  - LOOCV에 비해 test error의 추정이 오히려 정확한 경우가 종종 발생함 (훈련자료의 overlap 경감효과).

- 단점

  - k개의 집합으로 분할시 randomness가 발생하여 test error 추정시 변동성 발생. 하지만, Validation set approach보다는 훨씬 안정적임.

test error의 추정 (LOOCV vs k-fold CV)

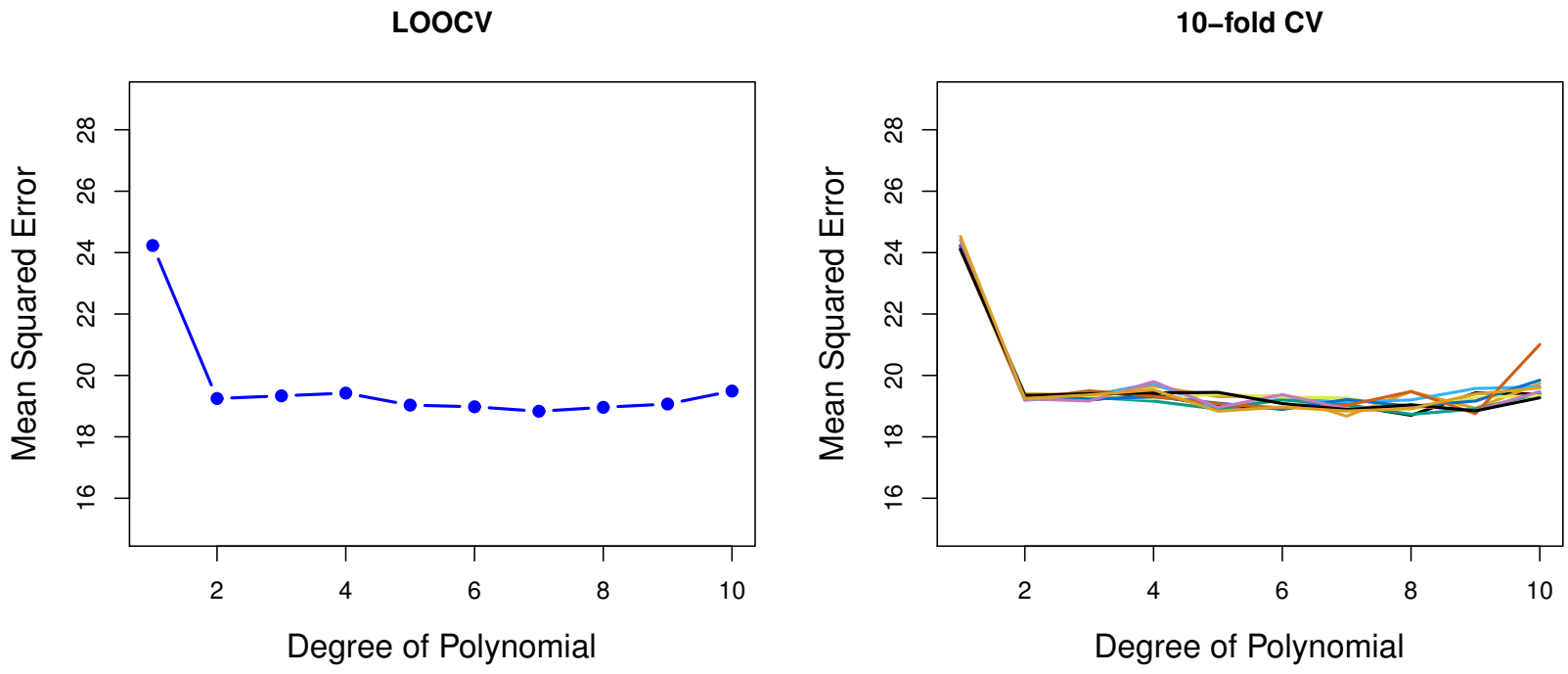

Figure 5: LOOCV vs k-fold CV

분류문제에서의 CV

- 교차타당검증법은 자료를 어떻게 분할할 것인가에 대한 문제이므로 평가측도와는 큰 관련이 없.
- 평가자료에서의 오분류율, 민감도, 특이도, AUC등을 분류기에 대한 지표로 사용할 수 있음.
- 예: 분류문제에서의 LOOCV에 의한 오분류율.

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_{(i)})$$

test error의 minimizer

- test error의 추정 자체에 관심 있는 경우가 있음.
- 언제 test error가 최소가 되는지에 관심 있는 경우도 많음.
- 후자의 경우 test error의 과대/과소추정에는 보다 관대할 수 있음.
- true test MSE (blue), LOOCV estimate (black), 10-fold estimate (orange).

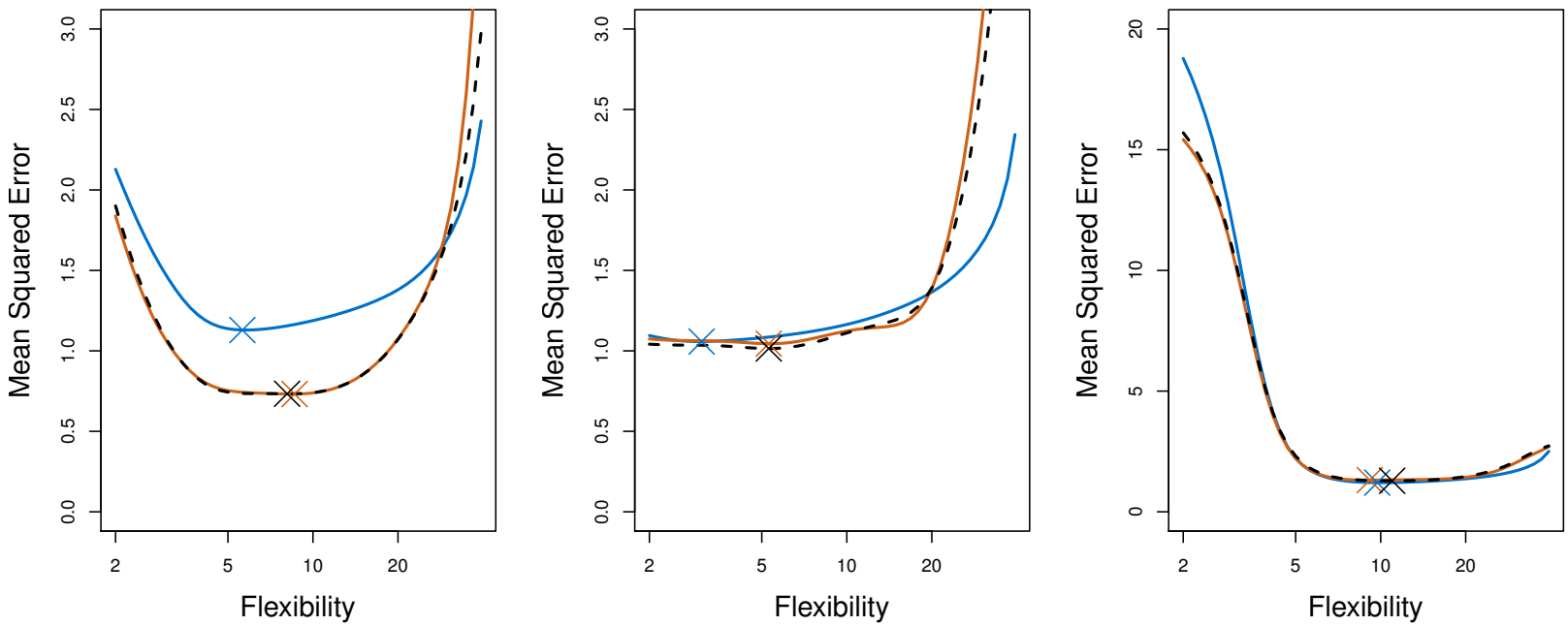

Figure 6: 여러 데이터에 대한 test error 추정

Bootstrap

- 주어진 데이터의 크기가 n일 때, 각각의 데이터 포인트에 대해서 $1/n$의 확률을 주고 샘플링을 함.

- 주로 추정량의 분산을 정확히 알기 힘들 때, 이를 알아내는 방법으로 사용함.

$$Z_*^1, Z_*^2, \cdots, Z_*^B \rightarrow \hat{\alpha}^1, \hat{\alpha}^2, \cdots, \hat{\alpha}^B$$

$$SE(\hat{\alpha}) = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\alpha}^b - \bar{\hat{\alpha}}_B)^2}$$

그림을 통한 bootstrap의 이해

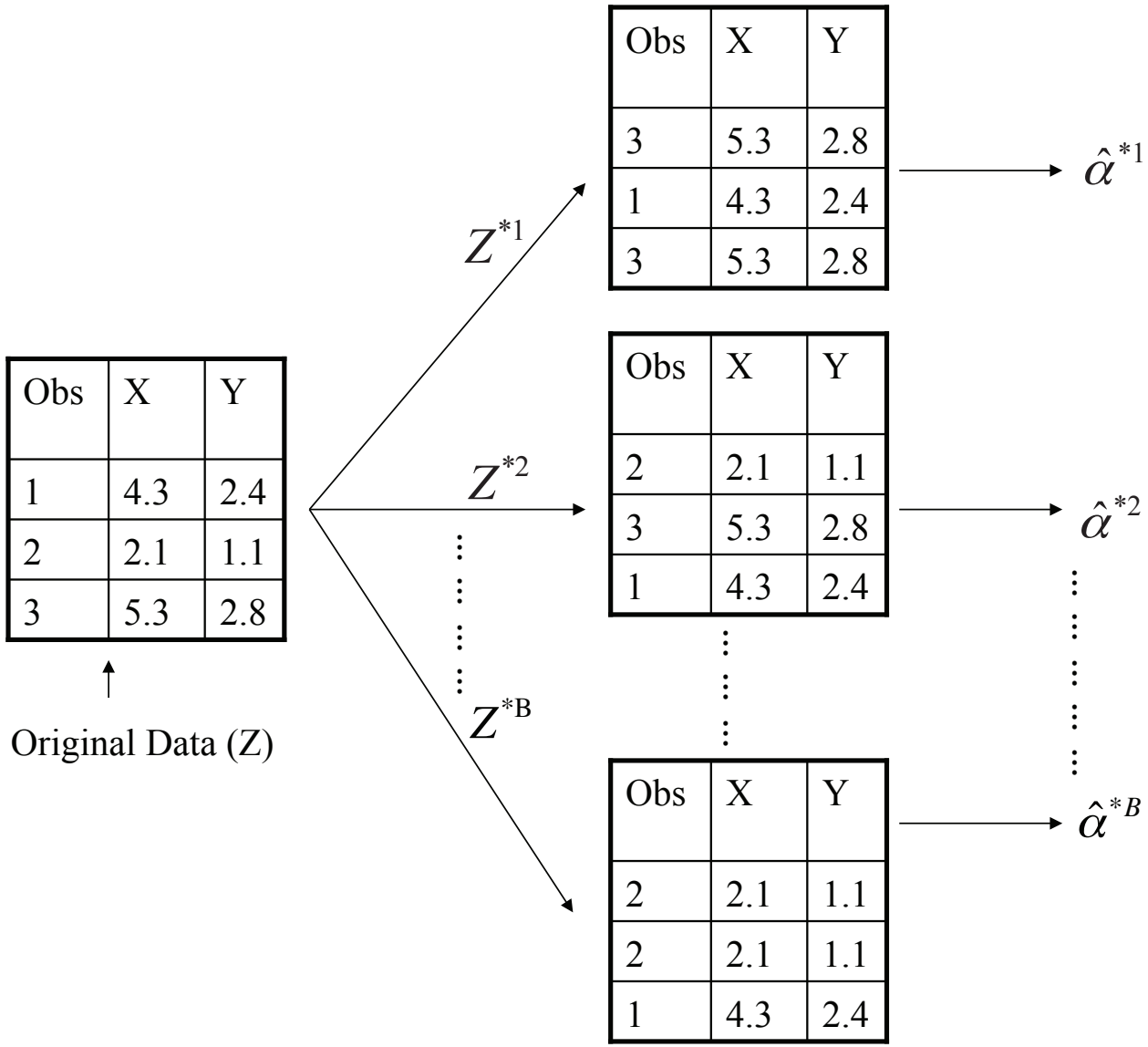

Figure 7: Bootstrap의 그래프를 통한 이해

Bootstrap R 예제

```r
set.seed(20)
x <- rnorm(100)

 set.seed(20)
 num_bootstrap = 2000
 size_bootstrap= 20
 bootstrap_samples <- lapply(1:num_bootstrap,
             FUN = function(i)
             sample(x, size =size_bootstrap, replace = T))
print(round(bootstrap_samples[[1]],3))
```

```
##  [1]  1.246 -0.385 -0.586  1.101 -2.475 -0.217 -0.069 -1.047 -0.818
## [11]  0.059  1.323 -0.660 -0.385 -0.031  0.992  0.189 -0.241  1.329
```

```r
rs = c()
for ( i in 1:num_bootstrap)
{ rs = c(rs, mean(bootstrap_samples[[i]])) }
c(mean(x), quantile(rs, 0.975), qt(0.975, df=19)*sd(x)/sqrt(20))
```

```
##                      97.5%
## 0.004908104 0.426921919 0.463287783
```