Data Mining

HW#5: Cross-Validation and the Bootstrap

학번	182STG18
이름	이하경
제출일	2019.04.17

Description

Cross-Validation

적합한 모형에서 추가적인 정보를 얻기 위해 사용되는 Resampling Method 중 하나인 Cross-Validation 은 일반적으로 test set 에 대한 예측오차를 추정하기 위해 사용한다. 모형 적합에 사용된 관측치들에 대해 예측된 값과 실제 값 사이의 평균 오차를 Training Error 라고 하며 Test Error 란 모형 적합에 사용되지 않은 새로운 관측치에 대한 실제 값과 예측 값들의 평균 오차이다. 일반적으로 Training Error 는 모형이 복잡해질수록 지속적으로 감소하지만 훈련 데이터에만 과도하게 적합될 경우 Test Error 는 오히려 매우 크게 나타날 수 있다.

데이터를 랜덤하게 training set 과 validation set (hold-out)으로 분할한 후 training set 에 대해 모형을 적합하고, validation set 에 대한 예측오차를 계산한다. 이 예측오차는 새로운 관측치들에 대한 예측오차의 추정치가 된다. 반응변수가 연속형일 경우 MSE(Mean Square Error)를 이용하며, 범주형일 경우에는 오분류율(Missclassification Rate)를 이용한다.

K-fold CV 란 Cross Validation 의 특수한 경우로 주어진 sample 을 K 개의 동일한 사이즈의 subset $C_1, \dots, C_K$로 그룹화하여 각 C_k를 제외한 나머지 K-1 개의 그룹을 훈련 데이터로 사용하여 C_k에 대한 예측오차(MSE 또는 오분류율)을 계산하여 평균 예측오차를 계산하는 방법이다. 일반적으로 K=5 또는 10 을 사용한다.

$$\mathrm{CV}_{(\mathrm{K})} = \sum_{k=1}^{K} \frac{n_k}{n} \cdot MSE_k$$

특히 K=n 으로 전체 데이터의 개수와 동일하게 설정할 경우, 각 i 번째 관측치에 대한 예측오차를 나머지 n-1 개의 데이터를 사용한 모형으로 계산하는 LOOCV(Leave-One-Out CV)라고 할 수 있다.

Bootstrap

Bootstrap 은 관심 통계량에 대한 추정을 효과적으로 제공하는 Resampling 방법이다. 기존의 데이터로부터 복원추출으로 동일한 사이즈의 데이터를 반복적으로 생성해 'bootstrap data set' (BS)을 여러 개 만든다. 각각의 set 을 이용해 관심 통계량을 계산하여 원하는 만큼의 추정치의 값과 분포를 얻을 수 있다. Bootstrap Sample 은 기존의 데이터와 상당히 많은 부분 겹쳐지며 실제 데이터의 약 2/3 이 각각의 BS 에 포함된다. (**Exercise 5.2** 에서는 이것을 n 에 따라 증명한다.)

Exercise 5.5, 5.7, 5.9 에서는 위의 두 가지 Resampling 방법을 직접 실제 데이터 셋에 적용해본다.

Results

Chapter 5 Lab: Cross-Validation and the Bootstrap

Lab에서는 R 코드를 작성해 직접 CV와 Bootstrap을 실행함으로써 두 과정에 대해 유용하게 사용할 수 있는 함수들을 알게 되었다.

먼저 Auto 데이터에 대해 50%의 traininig set과 50%의 validation set으로 나누어 K=2의 K-fold CV를 적용하였다. 선형회귀모형을 적합하고 validation error을 계산하였고 training set이 달라짐에 따라 validation error도 조금씩 달라지는 것을 확인하였다. 또한 같은 데이터에 대해 LOOCV 역시 적용해보았다. K-fold CV를 계산하기 위해서는 직접 데이터를 랜덤 분할하여 회귀모형 추정 및 예측을 할 수도 있지만 **boot** 라이브러리에 내장된 **cv.glm**(data, model, K) 함수를 사용해 쉽게 CV 추정치를 구할 수 있다는 것을 알게 되었다.

두번째로 Bootstrap 방법에서는 Portfolio 데이터에 대해 α 통계량의 bootstrap sample을, Auto 데이터에서 선형회귀모형의 계수에 대한 bootstrap sample을 각각 여러 개 생성해보았다. 이 역시 **boot**(data, function, R) 함수를 사용하여 R개의 bootstrap sample을 생성하고 관심 통계량의 추정치 및 표준 오차를 구할 수 있다는 것을 알게 되었다.

Exercise 5.2

(a) Probability that the 1st bootstrap observation is not the jth observation from the original sample

Bootstrap Sample에 구성하기 위해 original sample에서 1개씩 n번의 랜덤 복원추출을 한다고 할 때, 첫번째로 추출된 observation이 original sample의 j번째 observation이 아니려면 j를 제외한 다른 n-1개 중 하나여야 하므로 확률은 **(n-1)/n**이다.

(b) Probability that the 2nd bootstrap observation is not the jth observation from the original sample

Bootstrap Sample의 두번째로 복원추출된 observation이 original sample의 j번째 observation이 아닐 확률 역시 (a)와 동일하게 **(n-1)/n**이다.

(c) Probability that the jth observation is not in the the bootstrap sample

총 n번의 복원추출 중 original sample의 j번째 observation이 한번도 포함되지 않으려면 각각의 복원추출이 모두 나머지 n-1개에서만 이루어져야 하므로 확률은 다음과 같다.

$$\left(\frac{n-1}{n}\right)^n = \left(1-\frac{1}{n}\right)^n$$

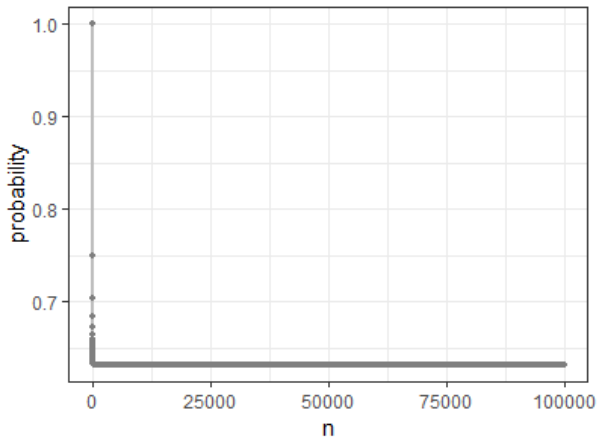
(d)-(f) Probability that the jth observation is in the Bootstrap Sample

j번째 observation이 bootstrap sample에 포함되려면 n번의 복원추출 중 적어도 1번 이상 j번째 observation이 뽑혀야 하므로 (c)의 반대의 경우이며 확률은 다음과 같다.

$$1-\left(1-\frac{1}{n}\right)^n$$

n=5	n=100	n=10000
0.67232	0.63397	0.63214

(g) Plot of the Probability that the jth observation is in the BS (n=1, ..., 100000)



n=100000	n=75000	n=50000	n=25000	n=1
Min	Q1	Median	Q3	Max
0.63212	0.63212	0.63212	0.63213	1

n이 커짐에 따라 확률이 기하급수적으로 감소하며 $1-e^{-1} \cong 0.63212$으로 수렴하였다. 이것은 임의의 j번째 관측치가 Bootstrap Sample에 포함될 확률이 각각 약 0.632으로, 다시 말해 Bootstrap Sample은 전체 Original Sample의 63%, 즉 약 2/3을 포함한다고 할 수 있다. $cf.\ \lim_{n\to\infty}\left(1-\frac{1}{n}\right)^{n} = e^{-1}$

(h) Numerical Investigation of the Probability using the Bootstrap Sample of n=100

주어진 방법에 따라 1, ..., 100으로 이루어진 n=100의 sample에 대해 총 10000번 bootstrap sample을 생성하여 4가 포함되어 있으면 1, 포함되어 있지 않으면 0을 입력하였다.

10000개 중 4를 포함한 경우의 수는 6309번으로 j번째 observation (j=4)가 Bootstrap Sample에 포함될 확률은 0.6309이다. 이것은 수식을 이용해 구한 $1-\left(1-\frac{1}{100}\right)^{100} = 0.63397$과 차이가 크지 않다.

Exercise 5.5 Default Data

(a) Logistic Regression: using income and balance (Full Data Set)

Call: glm(formula = default ~ income + balance, family = binomial, data = Default)

Coefficients:

	Estimate	Std. Error	z-value	P(>|z|) (p-value)
(Intercept)	-11.5405	0.4348	-26.5447	< 2e-16 ***
income	0.0000	0.0000	4.1742	2.99e-05 ***
balance	0.0056	0.0002	24.8363	< 2e-16 ***

AIC: 1585

(b) Validation Set Approach

i. Default 데이터는 총 10,000 개의 관측치를 포함하고 있으므로 시드를 고정 후 5000 개의 훈련 데이터 셋과 5000 개의 검증 데이터 셋으로 랜덤 분할하였다.
ii. 훈련 데이터 셋에 포함된 5000 개의 관측치를 사용해 Logistic Regression 을 적합하였다.
iii. 모형 적합에 사용하지 않은 5000 개의 검증 데이터 셋에 대해 사후 확률을 예측하고 값이 0.5 이상일 경우 default='Yes'로 분류하였다.
iv. 분류한 값과 실제 default 의 값이 일치하는 지 확인하고 오분류율을 계산하였다.

Call: glm(formula = default ~ student + income + balance, family = binomial, data = train)

Coefficients:

	Estimate	Std. Error	z-value	P(>|z|) (p-value)
(Intercept)	-12.2687	0.6525	-18.8030	< 2e-16 ***
income	0.0000	0.0000	4.1189	3.81e-05 ***
balance	0.0059	0.0003	17.5371	< 2e-16 ***

AIC: 768.19

2.68%	TRUE	
Fitted	No	Yes
No	4816	**115**
Yes	**19**	50

5000 개 중 오분류된 관측치의 개수는 134 개로 검증 데이터 셋에 대한 오분류율은 **2.68%**으로 계산되었다.

(c) Repeat (b) 3-times using different splits

Trial 1	Trial 2	Trial 3
2.38%	2.84%	2.64%

시드를 고정하지 않고 Training/Validation 데이터를 랜덤하게 분할함에 따라 오분류율이 조금씩 달라졌지만 대부분 약 2~3% 이내로 차이가 크지 않았다.

(d) Logistic Regression: using income, balance and student (dummy variable)

Call: glm(formula = default ~ student + income + balance, family = binomial, data = train)

Coefficients:

	Estimate	Std. Error	z-value	P(>|z|) (p-value)
(Intercept)	-11.6007	0.7199	-16.1134	< 2e-16 ***
studentYes	-0.7013	0.3428	-2.0455	0.0408 *
income	0.0000	0.0000	0.9095	0.3631
balance	0.0060	0.0003	17.3563	< 2e-16 ***

AIC: 766.04

(b)에서 사용한 동일한 Training 데이터에 대해 범주형 변수 student 를 포함한 로지스틱 모형을 다시 적합하였다. income 과 balance 의 추정된 회귀계수는 (b)의 모형과 거의 동일하나, student 변수 추가에 따라 income 회귀계수의 유의확률은 매우 커졌다. student 포함 모형으로 사후 확률 및 default 여부를 예측한 값과 실제 값을 비교한 결과는 다음과 같다.

2.72%	TRUE	
Fitted	No	Yes
No	4813	**114**
Yes	**22**	51

오분류율이 2.72%로, student 변수를 포함한 모형의 예측오차가 오히려 더 높았다.

Exercise 5.7 Weekly Data

(a) Logistic Regression: using Lag1 and Lag2 (Full Data Set)

Call: glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)

Coefficients:

	Estimate	Std. Error	z-value	P(>|z|) (p-value)
(Intercept)	0.2212	0.0615	3.599	0.0003 ***
Lag1	-0.0387	0.0262	-1.477	0.1397
Lag2	0.0603	0.0266	2.270	0.0232 *

(b) Logistic Regression: using Lag1 and Lag2 (except the first observation)

Call: glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly, subset = -1)

Coefficients:

	Estimate	Std. Error	z-value	P(>|z|) (p-value)
(Intercept)	0.2232	0.0615	3.630	0.0003 ***
Lag1	-0.0384	0.0262	-1.466	0.1427
Lag2	0.0609	0.0266	2.291	0.0220 *

(c) Prediction & Classification

(b)의 모형을 이용해 첫번째 관측치에 대한 수익률의 상승/하락 여부를 예측하였다. 실제 관측치의 Direction 값은 'Down'이나 예측된 사후확률이 $P(\text{Direction} = \text{'Up'}|\text{Lag1}, \text{Lag2}) = 0.5714 > 0.5$으로, 수익률을 'Up'으로 잘못 분류하였다.

(d) Repeat for i=1 to i=n

1 부터 n=1089 의 관측치에 대해 각각 다음의 과정을 실행해 오분류 여부를 저장하였다.

i. i 번째 관측치를 제외한 나머지 n-1=1088 개의 데이터를 사용해 Logistic Regression 모형을 적합한다.
ii. 사후확률 $P(\text{Direction} = \text{'Up'}|\text{Lag1}, \text{Lag2})$을 계산하여 0.5 이상일 경우 'Up'으로, 그렇지 않으면 'Down'으로 분류한다.
iv. 실제 값과 비교해 오분류되었을 경우 1, 일치할 경우 0 을 입력한다.

(e) LOOCV estimate for the test error

1	n	mean
490	**1089**	**0.44995**

1089 개 중 오분류된 경우는 490 으로 Test Error 에 대한 LOOCV 추정치는 약 **45%**이다.

각 관측치 하나에 대해 나머지 n-1 개의 데이터를 사용한 모형으로 예측할 경우 오분류되는 경우가 많으므로 주어진 데이터에 포함되어 있지 않은 새로운 test 데이터에 대해서도 예측력이 좋지 못할 수 있으며 데이터가 전체 모집단을 대표하지 못하는 것으로 보인다.

Exercise 5.9 Boston Housing Data

(a) Estimate the population mean of medv $\hat{\mu}$

$$\hat{\mu} = \bar{x} = 22.5328$$

(b) Estimate the standard error of $\hat{\mu}$

$$SE(\hat{\mu}) = s/\sqrt{n} = 9.1971/506 = 0.4089$$

모집단으로부터 추출된 n=506 개의 관측치로 계산된 표본 평균은 실제 모집단의 평균 μ와 약 0.4089 만큼의 오차를 가진다고 할 수 있다.

(c) Estimate the standard error of $\hat{\mu}$ using the Bootstrap

Bootstrap Statistics:

original	bias	std. error
22.5328	0.0013	**0.4060**

1000 개의 Bootstrap Sample 을 생성하여 1000 개의 $\hat{\mu}$의 추정치를 구하고, SE($\hat{\mu}$)을 추정하였다. Bootstrap 을 이용해 추정한 결과가 (b)에서와 거의 동일하다.

(d) 95% CI using the Bootstrap & t-test

	2.5%	97.5%
Bootstrap	21.7208	23.3448
t-test	21.7295	23.3361

Bootstrap 신뢰구간 $[\hat{\mu}_{BS} - 2SE(\hat{\mu}_{BS}),\ \hat{\mu}_{BS} - 2SE(\hat{\mu}_{BS})]$이 one sample t-test 를 이용해 구한 신뢰구간과 소수 둘째자리에서 거의 동일하다.

(e) Estimate the population median of medv $\hat{\mu}_{med}$ based on the Data Set

$$\hat{\mu}_{med} = \mathbf{quantile}(\mathbf{sample},\ \mathbf{0.5}) = \mathbf{21.2}$$

(f) Estimate the standard error of $\hat{\mu}_{med}$ using the Bootstrap

Bootstrap Statistics:

original	bias	std. error
21.2	0.0108	**0.3776**

1000 개의 Bootstrap Sample 을 생성해 각각 중앙값을 뽑고 표준오차 SE($\hat{\mu}_{med}$)를 추정하였다. 평균과 다르게 정해진 form 으로 구할 수 없는 중앙값의 표준오차를 Bootstrap 방법을 이용해 쉽게 추정할 수 있고, 신뢰구간 또한 구할 수 있다.

(g) Estimate the 10th percentile of medv $\hat{\mu}_{0.1}$ based on the Data Set

$$\hat{\mu}_{0.1} = \mathbf{quantile}(\mathbf{sample},\ \mathbf{0.1}) = \mathbf{12.75}$$

(h) Estimate the standard error of $\hat{\mu}_{0.1}$ using the Bootstrap

Bootstrap Statistics:

original	bias	std. error
12.75	0.0057	**0.5080**

10% quantile 에 해당하는 값 역시 Bootstrap 방법으로 표준오차를 추정하고 sample 의 추정치 12.75 에 대한 신뢰구간을 구할 수 있다.

Discussion

Exercise 2에서는 Bootstrap Sample이 기존 데이터의 각 관측치를 포함할 확률을 정해진 형태의 수식으로 표현하고 n에 따른 확률의 값과 그래프를 통해 0.63 정도에 가까워지는 것을 확인하였다. 또한 시뮬레이션으로 데이터를 직접 생성해 결과가 동일한지 확인하였다.

Exercise 5와 7에서는 Validation Approach를 적용하여 실제 데이터에 대한 로지스틱 회귀 모형을 적합하고 K=2, K=n (LOOCV)의 K-fold CV Error을 계산하였다. 여러 개의 Test Error를 구함으로써 데이터에 포함되지 않은 새로운 관측치에 대한 예측력을 판단할 수 있다.

Exercise 9에서는 Boston Housing 데이터에서 반응변수 medv의 분포를 대표하는 통계량들에 대해 Bootstrap 방법을 적용해 통계량의 분포 및 표준오차를 추정하고 original sample에서의 추정 통계량과 비교해보았다. 간단한 Resampling으로 구하고자 하는 통계량에 대한 분포 및 표준오차, 신뢰구간을 구할 수 있음을 알게 되었다. 평균에 대한 t-test 신뢰구간과 bootstrap 신뢰구간이 거의 동일하였으므로 Bootstrap을 이용한 추정이 합리적임을 알 수 있다.

[Appendix] R code

Exercise 5.2

```
# (d)-(f)
prob <- function(n) 1 - (1-1/n)^n
prob(5)
prob(100)
prob(10000)

# (g)
n <- 1:100000
ggplot() + theme_bw() + labs(y = 'probability') +
  geom_line(aes(n, prob(n)), color = 'gray', size = 1) +
  geom_point(aes(n, prob(n)), color = 'gray50', size = 1)
summary(prob(n))

# (h)
store <- rep(0, 10000)
for (i in 1:10000) store[i] <- sum(sample(1:100, replace = TRUE) == 4) > 0
mean(store)
```

Exercise 5.5

```
# (a) Logistic Regression
summary(glm5fit <- glm(default ~ income + balance, Default, family = binomial))

# (b) Validation Approach
# i. split
set.seed(10)
train.i <- sample(1:10000, 5000)
train <- Default[train.i,]
val <- Default[-train.i,]
# ii. fit
summary(glm5fit <- glm(default ~ income + balance, train, family = binomial))
# iii. CV
glm5prob <- predict(glm5fit, val, type = 'response')
glm5pred <- ifelse(glm5prob > 0.5, 'Yes', 'No')
table(glm5pred, val$default)
mean(glm5pred != val$default)

# (c) Repeat 3 times
repeatCV <- function() {
  train.i <- sample(1:10000, 5000)
  train <- Default[train.i,]
  val <- Default[-train.i,]
  glm5fit <- glm(default ~ income + balance, train, family = binomial)
  glm5prob <- predict(glm5fit, val, type = 'response')
  glm5pred <- ifelse(glm5prob > 0.5, 'Yes', 'No')
  return(mean(glm5pred != val$default))
}
repeatCV()

# (d)
summary(glm5fit.d <- glm(default ~ student + income + balance, train, family = binomial))
glm5prob.d <- predict(glm5fit.d, val, type = 'response')
glm5pred.d <- ifelse(glm5prob.d > 0.5, 'Yes', 'No')
mean(glm5pred.d != val$default)
table(glm5pred.d, val$default)
```

Exercise 5.7

```
# (a)
summary(glm7fit <- glm(Direction ~ Lag1 + Lag2, Weekly, family = binomial))
# (b)
summary(glm7fit.b <- glm(Direction ~ Lag1 + Lag2, Weekly, family = binomial, subset = -1))
predict(glm7fit.b, Weekly[1,], type = 'response')
Weekly[1,'Direction']
# (c)
store <- rep(0, nrow(Weekly))
for (i in 1:nrow(Weekly)) {
  fit <- glm(Direction ~ Lag1 + Lag2, Weekly, family = binomial, subset = -i)
  pred <- ifelse(predict(fit, Weekly[i,], type = 'response') > 0.5, 'Up', 'Down')
  store[i] <- as.numeric(pred != Weekly[i,'Direction'])
}
# (d)
mean(store)
myloocv <- cv.glm(Weekly, glm7fit)
myloocv$delta
```

Exercise 5.9

```
# (a)-(b)
attach(Boston)
mean(medv)
sd(medv) / sqrt(length(medv))

# (c)
boot9c <- function(data,index) mean(data[index])
(result9c <- boot(medv, boot9c, R = 1000))

# (d) 95% CI
c(result9c$t0 - 2*sd(result9c$t), result9c$t0 + 2*sd(result9c$t))
t.test(medv)

# (e)
median(medv)
# (f)
boot9f <- function(data,index) median(data[index])
(result9f <- boot(medv, boot9f, R = 1000))

# (g)
quantile(medv, 0.1)
# (h)
boot9h <- function(data, index) quantile(data[index], 0.1)
(result9h <- boot(medv, boot9h, R = 1000))
```