ML hw4

Runze Wang

2024-03-8

```r
library(r02pro)     #INSTALL IF NECESSARY
library(tidyverse)  #INSTALL IF NECESSARY
library(MASS)
library(pROC)
library(caret)
library(ISLR)
library(boot)


my_sahp <- sahp %>%
  na.omit() %>%
  mutate(expensive = sale_price > median(sale_price)) %>%
  dplyr::select(gar_car, liv_area, oa_qual, sale_price,
              expensive)
my_sahp$expensive <- as.factor(my_sahp$expensive)
n_all <- nrow(my_sahp)
set.seed(0)
tr_ind <- sample(n_all, round(n_all/2))
my_sahp_train  <- my_sahp[tr_ind, ]
my_sahp_val<- my_sahp[-tr_ind, ]
```

a. Use the validation set approach to divide the data into training (50%) and validation. Compute the average validation error for each model and decide which model is the best.

```r
mod_formula_seq <- c("sale_price ~ gar_car", "sale_price ~ gar_car + liv_area")
 sapply(mod_formula_seq, function(form) {
mod <- as.formula(form)
  fit <- lm(mod, data = my_sahp_train)
  pred_sale <- predict(fit, newdata = my_sahp_val)
  mean((my_sahp_val$sale_price - pred_sale)^2)
})
```

```
##            sale_price ~ gar_car sale_price ~ gar_car + liv_area
##                        3934.446                        2577.296
```

```r
mod_formula_knn <- c("sale_price ~ gar_car + liv_area")
sapply(mod_formula_knn, function(form) {
  mod <- as.formula(form)
  fit <- knn3(mod, data=my_sahp_train, k = 5)
  pred_sale <- predict(fit, newdata = my_sahp_val)
  mean((my_sahp_val$sale_price - pred_sale)^2)
```

```
})
```

```
## sale_price ~ gar_car + liv_area
##                         36814.64
```

```
 sapply(mod_formula_knn, function(form) {
  mod <- as.formula(form)
  fit <- knn3(mod, data=my_sahp_train, k = 50)
  pred_sale <- predict(fit, newdata = my_sahp_val)
  mean((my_sahp_val$sale_price - pred_sale)^2)
})
```

```
## sale_price ~ gar_car + liv_area
##                         36814.63
```

The model2 has the smallest CV error, so it is the best.

b. Use LOOCV approach to compute the CV error for each model and decide which model is the best.

```
mod_formula_seq <- c("sale_price ~ gar_car", "sale_price ~ gar_car + liv_area")
sapply(mod_formula_seq, function(form){
  mod <- as.formula(form)
  mean(sapply(1:nrow(my_sahp), function(j){
    fit <- lm(mod, data = my_sahp[-j, ])
    pred_sale <- predict(fit, newdata = my_sahp[j, ])
   (my_sahp$sale_price[j] - pred_sale)^2
 }))
}
)
```

```
##            sale_price ~ gar_car sale_price ~ gar_car + liv_area
##                        4438.426                        2851.620
```

```
mod_formula_knn <- c("sale_price ~ gar_car + liv_area")
sapply(mod_formula_knn, function(form) {
  mod <- as.formula(form)
  mean(sapply(1:nrow(my_sahp), function(j){
    fit <-  knnreg(mod, data = my_sahp[-j, ],k=5)
    pred_sale <- predict(fit, newdata = my_sahp[j, ])
   (my_sahp$sale_price[j] - pred_sale)^2
 }))
}
)
```

```
## sale_price ~ gar_car + liv_area
##                         4547.453
```

```
sapply(mod_formula_knn, function(form){
  mod <- as.formula(form)
  mean(sapply(1:nrow(my_sahp), function(j){
    fit <-  knnreg(mod, data = my_sahp[-j, ],k=50)
    pred_sale <- predict(fit, newdata = my_sahp[j, ])
     (my_sahp$sale_price[j] - pred_sale)^2
 }))
}
```

```
)
```

```
## sale_price ~ gar_car + liv_area
##                      4429.139
```

The model2 has the smallest CV error, so it is the best.

c. Use 5-fold CV approach to compute the CV classification error for each model and decide which model is the best.

```
K <- 5
set.seed(0)
f <- ceiling(n_all/K)
fold_ind <- sample(rep(1L:K, f), n_all)
mod_formula_seq <- c("sale_price ~ gar_car", "sale_price ~ gar_car + liv_area")
sapply(mod_formula_seq, function(form){
  mod <- as.formula(form)
  mean(sapply(1:K, function(j){
    fit <- lm(mod, data = my_sahp[fold_ind != j, ])
    pred_sale <- predict(fit, newdata =my_sahp[fold_ind == j, ])
  mean((my_sahp$sale_price[fold_ind == j] - pred_sale)^2)
  }))
}
)
```

```
##            sale_price ~ gar_car sale_price ~ gar_car + liv_area
##                        4365.122                        2871.005
```

```
mod_formula_knn <- c("sale_price ~ gar_car + liv_area")
sapply(mod_formula_knn, function(form){
 mod <- as.formula(form)
 mean(sapply(1:K, function(j){
 fit <- knn3(mod, data = my_sahp[fold_ind != j, ],k=5)
 pred_sale <- predict(fit, newdata = my_sahp[fold_ind == j, ])
 mean((my_sahp$sale_price[fold_ind == j] - pred_sale)^2) }))
}
)
```

```
## sale_price ~ gar_car + liv_area
##                        39355.57
```

```
sapply(mod_formula_knn, function(form){
 mod <- as.formula(form)
 mean(sapply(1:K, function(j){
 fit <- knn3(mod, data = my_sahp[fold_ind != j, ],k=50)
 pred_sale <- predict(fit, newdata = my_sahp[fold_ind == j, ])
 mean((my_sahp$sale_price[fold_ind == j] - pred_sale)^2) }))
}
)
```

```
## sale_price ~ gar_car + liv_area
##                        39355.57
```

The model2 has the smallest CV error, so it is the best.

a. Use the validation set approach to divide the data into training (50%) and validation. Compute the average validation classification error for each model and decide which model is the best.

```
mod_formula_seq2 <- c("expensive ~ gar_car+liv_area")
my_cost <- function(r, pi = 0) {
  pred_label <- pi > 0.5
  mean(pred_label != r)
}
sapply(mod_formula_seq2, function(form){
  mod <- as.formula(form)
  fit <- glm(mod, data = my_sahp, family = "binomial")
  cv.glm(my_sahp, fit, cost = my_cost)$delta[1]
  }
)
```

```
## expensive ~ gar_car+liv_area
##                    0.2283951
```

```
sapply(mod_formula_seq2, function(form) {
 mod <- as.formula(form)
 fit <- lda(mod, data = my_sahp_train)
 lda <- predict(fit, newdata = my_sahp_val)$class
 mean(lda != my_sahp_val$expensive)
})
```

```
## expensive ~ gar_car+liv_area
##                    0.2469136
```

```
sapply(mod_formula_seq2,function(form){
 mod<-as.formula(form)
 fit <-qda(mod, data=my_sahp_train)
 qda <- predict(fit,newdata = my_sahp_val)$class
 mean(qda != my_sahp_val$expensive)
}
)
```

```
## expensive ~ gar_car+liv_area
##                    0.2469136
```

```
sapply(mod_formula_seq2, function(form) {
  mod <- as.formula(form)
  fit <- knn3(mod, data=my_sahp_train, k = 20)
  pred_sale <- predict(fit, newdata = my_sahp_val)
  mean(pred_sale != my_sahp_val$expensive)
}
)
```

```
## expensive ~ gar_car+liv_area
##                            1
```

The model1 has the smallest average validation classification error, so it is the best.

b. Use LOOCV approach to compute the CV classification error for each model and decide which model is the best.

```
sapply(mod_formula_seq2, function(form) {
 mod <- as.formula(form)
 mean(sapply(1:nrow(my_sahp), function(j){
 fit <- glm(mod, data = my_sahp[-j, ], family = "binomial")
 pred_expensive <- predict(fit, newdata = my_sahp[j, ], type = "response") > 0.5
 pred_expensive != my_sahp$expensive[j]
 }))
})
```

```
## expensive ~ gar_car+liv_area
##                     0.2283951
```

```
sapply(mod_formula_seq2, function(form) {
 mod <- as.formula(form)
 mean(sapply(1:nrow(my_sahp),function(j){
 fit <- lda(mod, data = my_sahp[-j,])
 lda<- predict(fit, newdata = my_sahp[j,])$class
 lda != my_sahp$expensive[j]
 }))
})
```

```
## expensive ~ gar_car+liv_area
##                     0.2283951
```

```
sapply(mod_formula_seq2, function(form) {
 mod <- as.formula(form)
 mean(sapply(1:nrow(my_sahp),function(j){
 fit <- qda(mod, data = my_sahp[-j,])
 qda <- predict(fit, newdata = my_sahp[j,])$class
 qda != my_sahp$expensive[j]
 }))
})
```

```
## expensive ~ gar_car+liv_area
##                     0.2098765
```

```
sapply(mod_formula_seq2, function(form){
 mod <- as.formula(form)
 mean(sapply(1:nrow(my_sahp), function(j){
 fit <- knn3(mod, data = my_sahp[-j, ], k = 20)
 pred_expensive <- predict(fit, newdata = my_sahp[j,])
 pred_expensive != my_sahp$expensive[j]
 }))
})
```

```
## expensive ~ gar_car+liv_area
##                             1
```

The model 3 has the smallest average validation classification error, so it is the best.

c. Use 5-fold CV approach to compute the CV classification error for each model and decide which model is the best.

```
K <- 5
n_all <- nrow(my_sahp)
fold_ind <- sample(1:K, n_all, replace = TRUE)
sapply(mod_formula_seq2, function(form){
 mod <- as.formula(form)
 mean(sapply(1:K, function(j){
 fit <- glm(mod, data = my_sahp[fold_ind != j, ],family = "binomial")
 pred_prob <- predict(fit, newdata = my_sahp[fold_ind == j, ], type = "response")
 pred_label <- ifelse(pred_prob > 0.5, "TRUE", "FALSE")
 mean(my_sahp$expensive[fold_ind == j] != pred_label)
 }))
}
)
```

```
## expensive ~ gar_car+liv_area
##                       0.2158097
```

```
sapply(mod_formula_seq2, function(form){
 mod <- as.formula(form)
 mean(sapply(1:K, function(j){
 fit <- lda(mod, data = my_sahp[fold_ind != j, ])
 pred_prob <- predict(fit, newdata = my_sahp[fold_ind == j, ])
    pred_label <- pred_prob$class
 mean(my_sahp$expensive[fold_ind == j] != pred_label)
 }))
})
```

```
## expensive ~ gar_car+liv_area
##                       0.2220597
```

```
sapply(mod_formula_seq2, function(form){
 mod <- as.formula(form)
 mean(sapply(1:K, function(j){
 fit <- qda(mod, data = my_sahp[fold_ind != j, ])
 pred_prob <- predict(fit, newdata = my_sahp[fold_ind == j, ])
    pred_label <- pred_prob$class
 mean(my_sahp$expensive[fold_ind == j] != pred_label)
 }))
}
)
```

```
## expensive ~ gar_car+liv_area
##                       0.2310131
```

```
sapply(mod_formula_seq2, function(form){
 mod <- as.formula(form)
 mean(sapply(1:K, function(j){
 fit <- knn3(mod, data = my_sahp[fold_ind != j, ],k=20)
 pred_prob <- predict(fit, newdata = my_sahp[fold_ind == j, ])
 pred_label <- ifelse(pred_prob[,2] > 0.5, "TRUE", "FALSE")
 mean(my_sahp$expensive[fold_ind == j] != pred_label)
 }))
})
```

```
## expensive ~ gar_car+liv_area
##                     0.2781984
```

The model 1 has the smallest average validation classification error, so it is the best.

3. Q2 from Chapter 5, Page 219, ISLRv2.

We will now derive the probability that a given observation is part of a bootstrap sample. Suppose that we obtain a bootstrap sample from a set of n observations. (a) What is the probability that the first bootstrap observation is not the jth observation from the original sample? Justify your answer.

1-1/n. There is total n observation, and the probability of jth the observation has been selected is 1/n. so the probability of not being selected is 1-1/n.

(b) What is the probability that the second bootstrap observation is not the jth observation from the original sample?

1-1/n. There is total n observation , and the probability of jth the observation has been selected is 1/n. so the probability of not being selected is 1-1/n due to with replacement.

Argue that the probability that the jth observation is not in the bootstrap sample is (1 - 1/n)^n.

The probability that the jth observation has been selected is 1/n, so the probability for the jth observation not to be selected is 1-1/n. Each time we selected the jth observation was random and independent. so probabilities that each bootstrap observation is not the jth observation is (1-1/n)* (1-1/n)… (1-1/n) = (1-1/n)^n.

(d) When n = 5, what is the probability that the jth observation is in the bootstrap sample?

```r
1 - (1-1/5)^5
```

```
## [1] 0.67232
```

The probability that the jth observation is in the bootstrap sample is 0.67232.

(e) When n = 100, what is the probability that the jth observation is in the bootstrap sample?

```r
1 - (1-1/10)^10
```

```
## [1] 0.6513216
```

The probability that the jth observation is in the bootstrap sample is 0.6513216.

(f) When n = 10, 000, what is the probability that the jth observation is in the bootstrap sample?

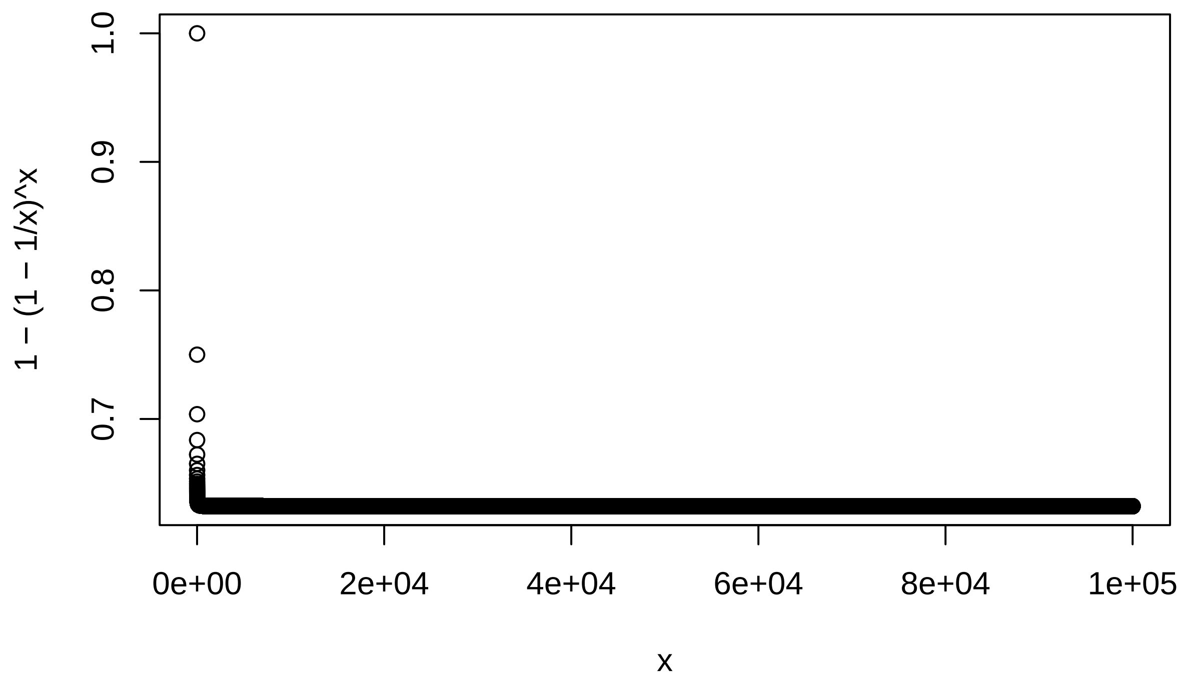
```r
1 - (1-1/10000)^10000
```

```
## [1] 0.632139
```

The probability that the jth observation is in the bootstrap sample is 0.632139.

(g) Create a plot that displays, for each integer value of n from 1 to 100, 000, the probability that the jth observation is in the bootstrap sample. Comment on what you observe.

```
x <- 1:100000
plot(x, 1 - (1 - 1/x)^x)
```



As the n becomes large, the probability tend to becomes constants.

$$\lim_{n\to\infty}\left(1-\left(1-\frac{1}{n}\right)^n\right)=1-\frac{1}{e}\approx 0.632,$$

.

The probability tends to becomes 0.632.

We will now investigate numerically the probability that a bootstrap sample of size n = 100 contains the jth observation. Here j = 4. We repeatedly create bootstrap samples, and each time we record whether or not the fourth observation is contained in the bootstrap sample.

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

```
## [1] 0.6346
```

we create 1000 observations with a bootstrap sample of size n = 100 each time, the probability that will contain 4th observation is 0.6351.