HW 4

Fan Bi (fb2234)

You can run the following code to prepare the analysis.

```
library(r02pro)     #INSTALL IF NECESSARY
library(tidyverse)  #INSTALL IF NECESSARY
library(MASS)
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)
```

Please answer the following questions.

1. Use the training data `my_sahp` to fit the following four models of `sale_price`.

- Model 1: linear model on variable `gar_car`.
- Model 2: linear model on variables `gar_car` and `liv_area`.

- Model 3: KNN with $K = 5$ on variables `gar_car` and `liv_area`.
- Model 4: KNN with $K = 50$ on variables `gar_car` and `liv_area`.

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.

Answer:

```
library(caret)
set.seed(1)
flds <- createFolds(1:dim(my_sahp)[1], k = 2, list = TRUE, returnTrain = FALSE)
names(flds)[1:2] <- c("train","test")

mod1 <- lm(sale_price ~ gar_car, data = my_sahp[flds$train,])
mod2 <- lm(sale_price ~ gar_car + liv_area, data = my_sahp[flds$train,])
mod3 <- knnreg(sale_price ~ gar_car + liv_area, data = my_sahp[flds$train,],k = 5)
mod4 <- knnreg(sale_price ~ gar_car + liv_area, data = my_sahp[flds$train,],k = 50)

mod_list <- paste("mod",1:4,sep = "")
err_list <- rep(0,4)
names(err_list) <- mod_list
for(i in 1:4){
  pred <- predict(get(mod_list[i]),newdata = my_sahp[flds$test,])
```

```
  err_list[i] <- mean((pred - my_sahp[["sale_price"]][flds$test])^2)
}

err_list
```

```
##     mod1     mod2     mod3     mod4
## 3594.605 2465.353 4053.862 2862.637
```

According to the average validation errors, `mod2` is the best.

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

Answer:

In order to adjust the loop structure, function `my_fit` is created.

```
my_fit <- function(formula,method,data,k = 1){
  if(method == "lm"){
    return(lm(formula,data))
  }
  if(method == "knn_r"){
    return(knnreg(formula,data,k = k))
  }
  if(method == "knn_c"){
    return(knn3(formula,data,k = k))
  }
  if(method == "logi"){
    return(glm(formula,data,family = "binomial"))
  }
  if(method == "lda"){
    return(lda(formula,data))
  }
  if(method == "qda"){
    return(qda(formula,data))
  }
  warning("no such method found")
}
```

Then we can do LOOCV by loop as below :

```
formula <- c(formula(sale_price ~ gar_car),
             formula(sale_price ~ gar_car + liv_area),
             formula(sale_price ~ gar_car + liv_area),
             formula(sale_price ~ gar_car + liv_area))
type <- c(rep("lm",2),rep("knn_r",2))
k_seq <- c(0,0,5,50)

LOOCV_score <- sapply(1:length(mod_list),function(i){
  mean(sapply(1:dim(my_sahp)[1],function(j){
    set.seed(j)
    md_temp <- my_fit(formula[[i]],type[i],my_sahp[-j,],k_seq[i])
    (predict(md_temp,my_sahp[j,]) - my_sahp[["sale_price"]][j])^2
```

```
  }))
})

names(LOOCV_score) <- mod_list
LOOCV_score
```

```
##     mod1     mod2     mod3     mod4
## 4438.426 2851.620 4547.453 4429.139
```

According to LOOCV scores, `mod2` is the best.

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

Answer:

```
set.seed(1)
flds <- createFolds(1:dim(my_sahp)[1], k = 5, list = TRUE, returnTrain = FALSE)

kfold_score5 <- sapply(1:length(mod_list),function(i){
  mean(sapply(1:5,function(j){
    set.seed(j)
    md_temp <- my_fit(formula[[i]],type[i],my_sahp[-flds[[i]],],k_seq[i])
    mean((predict(md_temp,my_sahp[flds[[i]],]) - my_sahp[["sale_price"]][j])^2)
  }))
})
names(kfold_score5) <- mod_list
kfold_score5
```

```
##     mod1     mod2     mod3     mod4
## 6089.372 4284.820 6176.911 2816.110
```

According to 5-Folds CV score, `mod4` is the best.

2. Use the data `my_sahp` to fit the following four models to predict `expensive`.

- Model 1: logistic regression on variables `gar_car` and `liv_area`.
- Model 2: LDA on variables `gar_car` and `liv_area`.
- Model 3: QDA on variables `gar_car` and `liv_area`.
- Model 4: KNN with $K = 20$ on variables `gar_car` and `liv_area`.

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.

Answer:

```r
set.seed(1)
flds <- createFolds(1:dim(my_sahp)[1], k = 2, list = TRUE, returnTrain = FALSE)
names(flds)[1:2] <- c("train","test")

formula <- expensive ~ gar_car + liv_area
traindata <- my_sahp[flds$train,]
testdata <- my_sahp[flds$test,]
k <- 20

mod1 <- my_fit(formula,"logi",traindata)
mod2 <- my_fit(formula,"lda",traindata)
mod3 <- my_fit(formula,"qda",traindata)
mod4 <- my_fit(formula,"knn_c",traindata,k)

mod_list <- paste("mod",1:4,sep = "")
err_list <- rep(0,4)
names(err_list) <- mod_list

OR1 <- predict(get(mod_list[1]),newdata = my_sahp[flds$test,])
pred1 <- ifelse(exp(OR1)/(exp(OR1) + 1) >= 0.5,T,F)
err_list[1] <- mean((as.factor(pred1) != my_sahp[["expensive"]][flds$test]))

pred2 <- predict(get(mod_list[2]),newdata = my_sahp[flds$test,])$class
err_list[2] <- mean(pred2 != my_sahp[["expensive"]][flds$test])

pred3 <- predict(get(mod_list[3]),newdata = my_sahp[flds$test,])$class
err_list[3] <- mean(pred3 != my_sahp[["expensive"]][flds$test])

pred4 <- predict(get(mod_list[4]),newdata = my_sahp[flds$test,])[,2] >= 0.5
err_list[4] <- mean((as.factor(pred4) != my_sahp[["expensive"]][flds$test]))

err_list
```

```
##      mod1      mod2      mod3      mod4
## 0.1975309 0.2345679 0.2839506 0.2592593
```

According to the average validation error rate, `mod1` is the best.

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

Answer:

```r
mod_type <- c("logi","lda","qda","knn_c")
LOOCV_score <- sapply(1:length(mod_type),function(i){
  mean(sapply(1:dim(my_sahp)[1],function(j){
    md_temp <- my_fit(formula,mod_type[i],my_sahp[-j,],k)
    if(mod_type[i] == "logi"){
      OR <- predict(md_temp,newdata = my_sahp[j,])
      pred <- ifelse(exp(OR)/(exp(OR) + 1) >= 0.5,T,F)
      return(as.factor(pred))
    }
```

```
    else if(mod_type[i] == "lda" | mod_type[i] == "qda"){
      return(predict(md_temp,newdata = my_sahp[j,])$class)}
    else if(mod_type[i] == "knn_c"){
      set.seed(j)
      pred <- predict(md_temp,newdata = my_sahp[j,])[,2] >= 0.5
      return(as.factor(pred))}
    else {warning("no such method found")}
  }) != my_sahp[["expensive"]])
})
names(LOOCV_score) <- mod_list
LOOCV_score
```

```
##      mod1      mod2      mod3      mod4
## 0.2283951 0.2283951 0.2098765 0.2654321
```

According to LOOCV error rate, `mod3` 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.

Answer:

```
set.seed(1)
flds <- createFolds(1:dim(my_sahp)[1], k = 5, list = TRUE, returnTrain = FALSE)

kfold_score5 <- sapply(1:length(mod_type),function(i){
  mean(sapply(1:5,function(j){
    md_temp <- my_fit(formula,mod_type[i],my_sahp[-flds[[j]],],k)
    if(mod_type[i] == "logi"){
      OR <- predict(md_temp,newdata = my_sahp[flds[[j]],])
      pred <- ifelse(exp(OR)/(exp(OR) + 1) >= 0.5,T,F)
      return(mean(as.factor(pred) != my_sahp[["expensive"]][flds[[j]]]))
    }
    else if(mod_type[i] == "lda" | mod_type[i] == "qda"){
      return(mean(predict(md_temp,newdata = my_sahp[flds[[j]],])$class
             != my_sahp[["expensive"]][flds[[j]]]))}
    else if(mod_type[i] == "knn_c"){
      set.seed(j)
      pred <- predict(md_temp,newdata = my_sahp[flds[[j]],])[,2] >= 0.5
      return(mean(as.factor(pred) != my_sahp[["expensive"]][flds[[j]]]))}
    else {warning("no such method found")}
  }))
})
names(kfold_score5) <- mod_list
kfold_score5
```

```
##      mod1      mod2      mod3      mod4
## 0.2280303 0.2217803 0.2090909 0.2952652
```

According to 5-Folds CV error rate, `mod3` is the best.

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

(a):

Denote original observations as $\{x_i\}_n$ and bootstrap samples as $\{x_{[i]}\}_n$.

$\forall j \in \{1, 2, \ldots, n\}$,

$$P(x_{[1]} \neq x_j) = 1 - P(x_{[1]} = x_j) = \frac{n-1}{n}$$

(b):

Because at any time we do bootstrap sampling with replacement, so

$\forall i, j \in \{1, 2, \ldots, n\}$,

$$\begin{aligned} P(x_{[i]} = x_j) &= P(x_{[1]} = x_j) \\ 1 - P(x_{[i]} = x_j) &= 1 - P(x_{[1]} = x_j) \\ \Rightarrow P(x_{[i]} \neq x_j) &= P(x_{[1]} \neq x_j) \end{aligned}$$

When $i = 2$, $\forall j \in \{1, 2, \ldots, n\}$

$$P(x_{[2]} \neq x_j) = P(x_{[1]} \neq x_j) = \frac{n-1}{n}$$

(c):

Every sample in bootstrap sampling is independent, which means

$$P(x_{[i]} = x_j | x_{[1]} = x_1^*, x_{[2]} = x_2^*, \ldots, x_{[i-1]} = x_{i-1}^*) = P(x_{[i]} = x_j)$$

So

$$\begin{aligned} P(x_j \notin \{x_{[i]}\}_n) &= \prod_{k=1}^{n} P(x_j \neq x_{[k]} | x_j \neq x_{[1]}, x_j \neq x_{[2]}, \ldots, x_j \neq x_{[k-1]}) \\ &= \prod_{k=1}^{n} P(x_j \neq x_{[k]}) \\ &= \prod_{k=1}^{n} P(x_{[k]} \neq x_j) \\ &= \prod_{k=1}^{n} P(x_{[1]} \neq x_j) \\ &= \prod_{k=1}^{n} \frac{n-1}{n} = \left(1 - \frac{1}{n}\right)^n \end{aligned}$$

(d):

When $n = 5$,$\forall j \in \{1, 2, \ldots, n\}$,

$$P(x_j \in \{x_{[i]}\}_n) = 1 - P(x_j \notin \{x_{[i]}\}_n) = 1 - \left(1 - \frac{1}{5}\right)^5 = 0.67232$$

(e):

When $n = 100$,$\forall j \in \{1, 2, \ldots, n\}$,

$$P(x_j \in \{x_{[i]}\}_n) = 1 - P(x_j \notin \{x_{[i]}\}_n) = 1 - \left(1 - \frac{1}{100}\right)^{100} = 0.6339677$$

(f):

When $n = 10,000, \forall j \in \{1, 2, \ldots, n\}$,

$$P(x_j \in \{x_{[i]}\}_n) = 1 - P(x_j \notin \{x_{[i]}\}_n) = 1 - \left(1 - \frac{1}{10,000}\right)^{10,000} = 0.632139$$

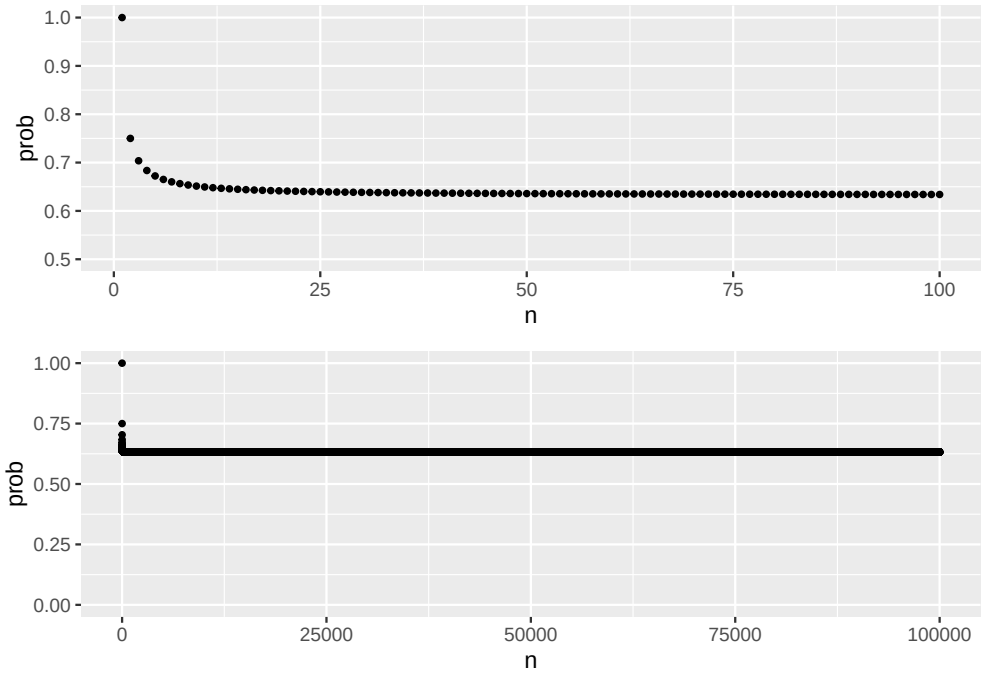
(g):

```
library(ggplot2)
library(egg)
p <- NULL
p[1] <- 1
for(i in 2:100000){p[i] <- 1-((i-1)/i)^i}
sml <- data.frame(n = 1:100000,prob = p)

p1 <-
ggplot(data = sml[1:100,]) +
  geom_point(aes(x = n,y = prob),size = 1) +
  coord_cartesian(ylim = c(0.5, 1))

p2 <-
ggplot(data = sml) +
  geom_point(aes(x = n,y = prob),size = 1) +
  coord_cartesian(ylim = c(0, 1))

ggarrange(p1,p2,nrow = 2)
```



When n increases from 1 to 100,000, the probability of any observation in original set included by bootstrap samples quickly converges to 0.63, which means bootstrap method is stable even sample size is extremely large.

(h):

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

```
## [1] 0.6258
```

We can find the numeric simulation result follows the plot in (g). The probability of the 4th observation included by 100 bootstrap samples is close to 0.63.