Untitled

Đinh Quang Dũng

2024-05-07

library(ISLR)

## Warning: package 'ISLR' was built under R version 4.3.3

data(Default)  
#5  
#a  
default\_data = Default  
  
View(default\_data)  
  
logit\_model <- glm(default ~ income + balance, data = default\_data, family = binomial)  
  
summary(logit\_model)

##   
## Call:  
## glm(formula = default ~ income + balance, family = binomial,   
## data = default\_data)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.154e+01 4.348e-01 -26.545 < 2e-16 \*\*\*  
## income 2.081e-05 4.985e-06 4.174 2.99e-05 \*\*\*  
## balance 5.647e-03 2.274e-04 24.836 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2920.6 on 9999 degrees of freedom  
## Residual deviance: 1579.0 on 9997 degrees of freedom  
## AIC: 1585  
##   
## Number of Fisher Scoring iterations: 8

#b  
set.seed(123)  
  
train\_indices <- sample(1:nrow(default\_data), 0.7\*nrow(default\_data))  
train\_set <- default\_data[train\_indices, ]  
valid\_set <- default\_data[-train\_indices, ]  
  
  
  
# Lập mô hình hồi quy logistic trên tập huấn luyện  
logit\_model\_train <- glm(default ~ income + balance, data = train\_set, family = binomial)  
  
# Dự đoán trên tập xác thực  
predictions <- predict(logit\_model\_train, newdata = valid\_set, type = "response")  
  
# Phân loại dự đoán  
predicted\_default <- ifelse(predictions > 0.5, "Yes", "No")  
  
# Tính lỗi của tập xác thực  
validation\_error <- mean(predicted\_default != valid\_set$default)  
validation\_error

## [1] 0.02633333

#c  
  
  
set.seed(123)  
  
errors <- numeric(3)  
for (i in 1:3) {  
 # Chia tập dữ liệu thành tập huấn luyện và tập xác thực  
 train\_indices <- sample(1:nrow(default\_data), 0.7\*nrow(default\_data))  
 train\_set <- default\_data[train\_indices, ]  
 valid\_set <- default\_data[-train\_indices, ]  
   
 # Lập mô hình hồi quy logistic trên tập huấn luyện  
 logit\_model\_train <- glm(default ~ income + balance, data = train\_set, family = binomial)  
   
 # Dự đoán trên tập xác thực  
 predictions <- predict(logit\_model\_train, newdata = valid\_set, type = "response")  
   
 # Phân loại dự đoán  
 predicted\_default <- ifelse(predictions > 0.5, "Yes", "No")  
   
 # Tính lỗi của tập xác thực và lưu vào vector errors  
 errors[i] <- mean(predicted\_default != valid\_set$default)  
}  
  
# In ra các giá trị lỗi  
errors

## [1] 0.02633333 0.02700000 0.02666667

#d  
  
# Lập mô hình hồi quy logistic với biến giả sinh viên  
logit\_model\_student <- glm(default ~ income + balance + student, data = default\_data, family = binomial)  
  
# Tính lỗi của mô hình mới  
predictions\_student <- predict(logit\_model\_student, newdata = valid\_set, type = "response")  
predicted\_default\_student <- ifelse(predictions\_student > 0.5, "Yes", "No")  
validation\_error\_student <- mean(predicted\_default\_student != valid\_set$default)  
validation\_error\_student

## [1] 0.028

#########  
#6  
  
# Load required libraries  
library(ISLR)  
library(boot)  
  
# Set random seed  
set.seed(123)  
  
# (a) Using glm() function to determine estimated standard errors  
model <- glm(default ~ income + balance, data = Default, family = binomial)  
summary(model)

##   
## Call:  
## glm(formula = default ~ income + balance, family = binomial,   
## data = Default)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.154e+01 4.348e-01 -26.545 < 2e-16 \*\*\*  
## income 2.081e-05 4.985e-06 4.174 2.99e-05 \*\*\*  
## balance 5.647e-03 2.274e-04 24.836 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2920.6 on 9999 degrees of freedom  
## Residual deviance: 1579.0 on 9997 degrees of freedom  
## AIC: 1585  
##   
## Number of Fisher Scoring iterations: 8

# (b) Define boot.fn() function  
boot.fn <- function(data, index) {  
 fit <- glm(default ~ income + balance, data = data[index, ], family = binomial)  
 return(coef(fit))  
}

# (c) Use boot() function to estimate standard errors using bootstrap  
boot.results <- boot(Default, boot.fn, R = 1000)  
boot.results

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = Default, statistic = boot.fn, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* -1.154047e+01 -2.754771e-02 4.204817e-01  
## t2\* 2.080898e-05 1.582518e-07 4.729534e-06  
## t3\* 5.647103e-03 1.296980e-05 2.217214e-04

#7

# Load required libraries  
library(ISLR)  
  
# (a) Fit logistic regression model using Lag1 and Lag2 to predict Direction  
model\_a <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial)

# (b) Fit logistic regression model using Lag1 and Lag2, excluding the first observation  
model\_b <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = binomial)

# (c) Predict direction for the first observation using model\_b  
prediction <- predict(model\_b, newdata = Weekly[1, ], type = "response")  
if (prediction > 0.5) {  
 direction\_predicted <- "Up"  
} else {  
 direction\_predicted <- "Down"  
}  
# Check if the prediction matches the actual direction for the first observation  
if (direction\_predicted == Weekly$Direction[1]) {  
 correct\_classification <- TRUE  
} else {  
 correct\_classification <- FALSE  
}

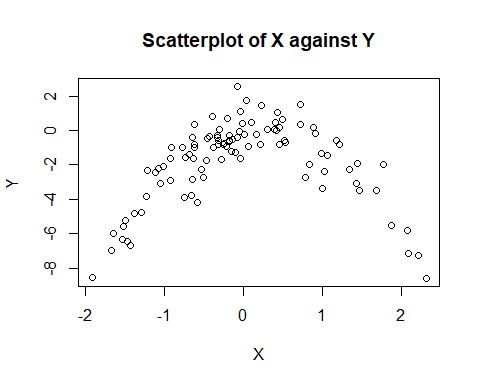
# (d) Implement LOOCV using a for loop  
n <- nrow(Weekly)  
errors <- numeric(n)  
for (i in 1:n) {  
 # Fit logistic regression model using all but the ith observation  
 model <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ], family = binomial)  
   
 # Predict the direction for the ith observation  
 prediction <- predict(model, newdata = Weekly[i, ], type = "response")  
   
 # Determine if an error was made  
 if ((prediction > 0.5 & Weekly$Direction[i] == "Down") | (prediction <= 0.5 & Weekly$Direction[i] == "Up")) {  
 errors[i] <- 1  
 } else {  
 errors[i] <- 0  
 }  
}

# (e) Compute LOOCV estimate for the test error  
LOOCV\_error <- mean(errors)  
LOOCV\_error

## [1] 0.4499541

#######  
#8  
  
#a  
set.seed(1)  
y <- rnorm(100)  
x <- rnorm(100)  
y <- x - 2 \* x^2 + rnorm(100)

#b  
  
plot(x, y, main = "Scatterplot of X against Y", xlab = "X", ylab = "Y")



#c  
# Khởi tạo vector lưu trữ LOOCV errors  
LOOCV\_errors <- numeric(4)  
  
# Mô hình 1: Y = β0 + β1X  
model1 <- lm(y ~ x)  
LOOCV\_errors[1] <- sum((residuals(model1) / (1 - hatvalues(model1)))^2) / 100  
  
# Mô hình 2: Y = β0 + β1X + β2X^2  
model2 <- lm(y ~ x + I(x^2))  
LOOCV\_errors[2] <- sum((residuals(model2) / (1 - hatvalues(model2)))^2) / 100  
  
# Mô hình 3: Y = β0 + β1X + β2X^2 + β3X^3  
model3 <- lm(y ~ x + I(x^2) + I(x^3))  
LOOCV\_errors[3] <- sum((residuals(model3) / (1 - hatvalues(model3)))^2) / 100  
  
# Mô hình 4: Y = β0 + β1X + β2X^2 + β3X^3 + β4X^4  
model4 <- lm(y ~ x + I(x^2) + I(x^3) + I(x^4))  
LOOCV\_errors[4] <- sum((residuals(model4) / (1 - hatvalues(model4)))^2) / 100  
  
# In kết quả  
print(LOOCV\_errors)

## [1] 5.890979 1.086596 1.102585 1.114772

###########  
#9  
  
#a  
# Load the MASS package if not already loaded  
  
library(MASS)

## Warning: package 'MASS' was built under R version 4.3.2

# Load the Boston housing data set  
data(Boston)  
  
# Estimate for the population mean of medv  
mu\_hat <- mean(Boston$medv)  
mu\_hat

## [1] 22.53281

#b  
# Estimate of the standard error of mu\_hat  
se\_mu\_hat <- sd(Boston$medv) / sqrt(length(Boston$medv))  
se\_mu\_hat

## [1] 0.4088611

#c  
  
# Bootstrap estimation of the standard error of mu\_hat  
boot\_mu <- function(data, index) {  
 mean(data[index])  
}  
set.seed(1)  
bootstrap\_mu <- replicate(1000, boot\_mu(Boston$medv, sample(length(Boston$medv), replace = TRUE)))  
se\_boot\_mu <- sd(bootstrap\_mu)  
se\_boot\_mu

## [1] 0.3974788

#d  
# 95% confidence interval for the mean of medv  
lower\_bound <- mu\_hat - 2 \* se\_boot\_mu  
upper\_bound <- mu\_hat + 2 \* se\_boot\_mu  
confidence\_interval <- c(lower\_bound, upper\_bound)  
confidence\_interval

## [1] 21.73785 23.32776

# Comparison with t.test  
t\_test\_result <- t.test(Boston$medv)  
t\_test\_result$conf.int

## [1] 21.72953 23.33608  
## attr(,"conf.level")  
## [1] 0.95

#e  
  
# Estimate for the median value of medv  
mu\_med\_hat <- median(Boston$medv)  
mu\_med\_hat

## [1] 21.2

#f  
  
# Bootstrap estimation of the standard error of mu\_med\_hat  
bootstrap\_med <- function(data, index) {  
 median(data[index])  
}  
set.seed(1)  
bootstrap\_median <- replicate(1000, bootstrap\_med(Boston$medv, sample(length(Boston$medv), replace = TRUE)))  
se\_boot\_med <- sd(bootstrap\_median)  
se\_boot\_med

## [1] 0.3693316

#g  
# Estimate for the tenth percentile of medv  
mu\_01\_hat <- quantile(Boston$medv, 0.1)  
mu\_01\_hat

## 10%   
## 12.75

#h  
  
# Define the bootstrap function to estimate the standard error of mu\_0.1\_hat  
bootstrap\_01 <- function(data, index) {  
 quantile(data[index], 0.1)  
}  
  
# Set random seed for reproducibility  
set.seed(1)  
  
# Perform bootstrap to estimate the 10th percentile  
bootstrap\_quantile <- replicate(1000, bootstrap\_01(Boston$medv, sample(length(Boston$medv), replace = TRUE)))  
  
# Standard error of mu\_0.1\_hat  
se\_01 <- sd(bootstrap\_quantile)  
se\_01

## [1] 0.4793555