Exercise 4: CV and Bootstrap problems

Thai pham- T00727094

2024-03-25

QUESTION 1

• Write your own R-code for cross-validation technique (10-fold, LOOCV) in linear regression problem from scratch and use it to a Auto data from ISLR package in R. – Load the Auto data and remove the last variable (name) to create a new data and use it to do the above task. Use the code to load data: data(Auto, package="ISLR") • In Auto data, mpg is your outcome variable • Select the significant variables first by fitting the linear regression model and use these predictors in cross-validation • Evaluate the model performance (compute the CV error)

```
# Load the data
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.3.2
```

```
data(Auto, package="ISLR")
head(Auto)
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
##                       name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
## 5               ford torino
## 6          ford galaxie 500
```

```
names(Auto)
```

```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
## [6] "acceleration" "year"         "origin"       "name"
```

```
# Remove the last variable (name)
Auto_new <- Auto[, -ncol(Auto)]
```

```
# Fitting the linear regression model with the outcome is mpg
lm_model <- lm(mpg ~ ., data = Auto_new)
summary(lm_model)
```

```
##
## Call:
```

```
## lm(formula = mpg ~ ., data = Auto_new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

```r
# Choose significant variables
significant_predictors <- c("displacement", "weight", "year", "origin")

# Use only significant predictors for cross-validation
Auto_subset <- Auto_new[, c("mpg", significant_predictors)]
```

a. 10- fold CV techinique (applied on the Auto_subset data)

```r
# Function to perform linear regression
perform_linear_regression <- function(train_data, test_data) {
  lm_model <- lm(mpg ~ ., data = train_data)
  predicted <- predict(lm_model, newdata = test_data)
  return(predicted)
}

# Function to calculate Mean Squared Error (MSE)
calculate_mse <- function(actual, predicted) {
  mse <- mean((actual - predicted)^2)
  return(mse)
}

# Function to perform 10-fold cross-validation
cross_validation_10fold <- function(data) {
  set.seed(123)  # For reproducibility
  folds <- cut(seq(1, nrow(data)), breaks = 10, labels = FALSE)
  mse_scores <- numeric(10)

  for (i in 1:10) {
    test_indices <- which(folds == i)
    train_data <- data[-test_indices, ]
    test_data <- data[test_indices, ]
```

```
    predicted <- perform_linear_regression(train_data, test_data)
    mse_scores[i] <- calculate_mse(test_data$mpg, predicted)
  }

  avg_mse <- mean(mse_scores)
  return(avg_mse)
}

# Perform 10-fold CV
mse_10fold <- cross_validation_10fold(Auto_subset)
cat("Average MSE (10-fold cross-validation):", mse_10fold, "\n")
```

```
## Average MSE (10-fold cross-validation): 12.2719
```

b. LOOCV technique (applied on the Auto_subset data)

```
# Function to perform linear regression
perform_linear_regression <- function(train_data, test_data) {
  lm_model <- lm(mpg ~ ., data = train_data)
  predicted <- predict(lm_model, newdata = test_data)
  return(predicted)
}

# Function to calculate Mean Squared Error (MSE)
calculate_mse <- function(actual, predicted) {
  mse <- mean((actual - predicted)^2)
  return(mse)
}

# Function to perform Leave-One-Out Cross-Validation (LOOCV)
loocv <- function(data) {
  n <- nrow(data)
  mse_scores <- numeric(n)

  for (i in 1:n) {
    train_data <- data[-i, ]
    test_data <- data[i, ]

    predicted <- perform_linear_regression(train_data, test_data)
    mse_scores[i] <- calculate_mse(test_data$mpg, predicted)
  }

  avg_mse <- mean(mse_scores)
  return(avg_mse)
}

# Perform LOOCV
mse_loocv <- loocv(Auto_subset)
cat("Average MSE (LOOCV):", mse_loocv, "\n")
```

```
## Average MSE (LOOCV): 11.35661
```

QUESTION 2

Suppose we have a population which is generated from a Poisson ($\lambda = 2.3$) distribution with pdf

$$f(x) = \frac{e^{-\lambda}\lambda^x}{x!}$$

We know that the MLE of λ in Poisson distribution is the sample mean $\frac{1}{n}\Sigma_{i=1}^{n} x_i$ of n observations in the sample. Do bootstraping (B = 1000) to see the variability of the MLE as an estimator of the Poisson parameter ($\lambda = 2.3$) for n = 100 by writing you own function. Calculate the bootstrap bias and bootstrap standard error of the bootstrap estimates of MLE. Also calculate the 95% bootstrap percentile confidence interval (you can use quantile function).

```
pkg_list <- c("dplyr", "caret", "boot","calibrate")
# Install packages if needed
for (pkg in pkg_list)
{
  # Try loading the library.
  if ( ! library(pkg, logical.return=TRUE, character.only=TRUE) )
    {
         # If the library cannot be loaded, install it; then load.
        install.packages(pkg)
        library(pkg, character.only=TRUE)
  }
}
```

```
## Warning: package 'dplyr' was built under R version 4.3.2
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
## Warning: package 'calibrate' was built under R version 4.3.3
```

```
# Define the Statistic function to calculate sample mean
sample_mean <- function(Population) {
  return(mean(Population))
}

# Define the boot.approx function
boot.approx <- function(Population, Statistic, B, n) {
  out <- numeric(B)
  for (b in 1:B) {
    sample_data <- sample(Population, n, replace = TRUE)
    out[b] <- Statistic(sample_data)
  }
  return(out)
}

# Generate Population from a Poisson distribution with lambda = 2.3
set.seed(1) # For reproducibility
Population <- rpois(10000, lambda = 2.3)

# Perform bootstrap resampling
B <- 1000
n <- 100
bootstrap_means <- boot.approx(Population, sample_mean, B, n)
```

```r
# Calculate bootstrap bias and standard error
bootstrap_bias <- mean(bootstrap_means) - 2.3
bootstrap_standard_error <- sd(bootstrap_means)

# Calculate 95% bootstrap percentile confidence interval
bootstrap_conf_interval <- quantile(bootstrap_means, c(0.025, 0.975))

# Print results
cat("Bootstrap bias:", bootstrap_bias, "\n")
```

```
## Bootstrap bias: 0.00689
```

```r
cat("Bootstrap standard error:", bootstrap_standard_error, "\n")
```

```
## Bootstrap standard error: 0.1523261
```

```r
cat("95% Bootstrap percentile confidence interval:",
    bootstrap_conf_interval[1], "-", bootstrap_conf_interval[2], "\n")
```

```
## 95% Bootstrap percentile confidence interval: 2.01 - 2.6
```