ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Моделі статистичного навчання

Лабораторна робота №4

Виконав:

студент групи ПМІм-11

Скупейко О. В.

2024

Вхідні дані:

Номер групи: 1
Номер студента: 15

Хід роботи:

Встановив змінну variant, що в моєму випадку дорівнює **40** та згенерував значення змінної redundant - це випадкове число з рівномірного розподілу на інтервалі від **6** до **24**, що дорівнює **18**.

```r
variant <- 1 * 25 + 15

set.seed(variant)
cat("Variant:", variant, "\n")

redundant <- round(runif(n = 1, min = 1 + 5, max = 25 - 1))
cat("Redundant: ", redundant, "\n")
```

```
Variant: 40
Redundant:  18
```

Завдання №1

1) Отримав набір даних Boston з бібліотеки MASS та модифікував ці дані.

```r
# Завдання №1
library(MASS)

boston_data <- MASS::Boston

reduction_percent <- redundant / 100

modified_boston_data <- boston_data[sample(nrow(boston_data), size = round(nrow(boston_data) * (1 - reduction_percent))), ] # nolint
```

2) Пристосував моделі логістичної регресії для передбачення у вибраному районі рівня злочинності більшого чи меншого за середній на основі змінних nox та rad. Окрім цього одразу в цикл додав розгляд і за змінних на основі змінних nox, rad та medv.

```r
test_errors1 <- numeric(4)
test_errors2 <- numeric(4)

for (i in 1:4) {
  set.seed(variant + i)

  train_index1 <- sample(seq_len(nrow(modified_boston_data)), size = 0.5 * nrow(modified_boston_data)) # nolint
  train_data1 <- modified_boston_data[train_index1, ]
  test_data1 <- modified_boston_data[-train_index1, ]

  model1 <- glm(high_crime ~ nox + rad, data = train_data1, family = "binomial")
  predictions1 <- predict(model1, newdata = test_data1, type = "response")
  test_errors1[i] <- mean(ifelse(predictions1 > 0.5, 1, 0) != test_data1$high_crime) # nolint

  train_index2 <- sample(seq_len(nrow(modified_boston_data)), size = 0.5 * nrow(modified_boston_data)) # nolint
  train_data2 <- modified_boston_data[train_index2, ]
  test_data2 <- modified_boston_data[-train_index2, ]

  model2 <- glm(high_crime ~ nox + rad + medv, data = train_data2, family = "binomial") # nolint
  predictions2 <- predict(model2, newdata = test_data2, type = "response")
  test_errors2[i] <- mean(ifelse(predictions2 > 0.5, 1, 0) != test_data2$high_crime) # nolint
}
```

```r
cat("Результати для моделі з nox та rad:\n")
for (i in 1:4) {
  cat("Ітерація", i, "- Тестова помилка:", test_errors1[i], "\n")
}

cat("\nРезультати для моделі з nox, rad та medv:\n")
for (i in 1:4) {
  cat("Ітерація", i, "- Тестова помилка:", test_errors2[i], "\n")
}
```

Результати на основі двох змінних nox та rad:

```
Результати для моделі з nox та rad:
Ітерація 1 - Тестова помилка: 0.004807692
Ітерація 2 - Тестова помилка: 0.01442308
Ітерація 3 - Тестова помилка: 0.02884615
Ітерація 4 - Тестова помилка: 0.03365385
```

Результати тестової помилки для моделі з nox та rad показують варіативність між ітераціями: 0.0048, 0.0144, 0.0288, 0.0337, що є очікуваним для методу валідаційного набору, оскільки цей підхід має високу дисперсію. Оцінка тестової помилки в цьому методі є менш точною, тому при кожному розбитті на навчальний та тестовий набори отримуються дещо різні значення.

Результати на основі трьох змінних nox, rad та medv:

```
Результати для моделі з nox, rad та medv:
Ітерація 1 - Тестова помилка: 0.02403846
Ітерація 2 - Тестова помилка: 0.04326923
Ітерація 3 - Тестова помилка: 0.02403846
Ітерація 4 - Тестова помилка: 0.02403846
```

Додавання змінної medv не суттєво зменшило тестову помилку порівняно з моделлю, що включає лише nox та rad. Навпаки, у другій ітерації тестова помилка навіть збільшилася до 0.0433. Це може свідчити про те, що medv не додає істотної прогностичної цінності або навіть вносить певний шум у модель. Загалом, додаткова змінна не призвела до стабільного покращення, а оцінка тестової помилки залишається залежною від випадковості вибірки, що є очікуваним для методу валідаційного набору.

Завдання №2

1) Отримав набір даних Auto з бібліотеки ISLR та модифікував ці дані.

```r
# Завдання №2
library(ISLR)
library(boot)

auto_data <- ISLR::Auto

reduction_percent <- redundant / 100
modified_auto_data <- auto_data[sample(nrow(auto_data), size = round(nrow(auto_data) * (1 - reduction_percent))), ]
```

2) Обчислив оцінку середнього змінної mpg та її стандартну похибку

```r
mpg_mean <- mean(modified_auto_data$mpg)
cat("Оцінка середнього значення mpg:", mpg_mean, "\n")

std_error_mean <- sd(modified_auto_data$mpg) / sqrt(nrow(modified_auto_data)) #
cat("Стандартна похибка середнього (традиційний метод):", std_error_mean, "\n")
```

```
--------
Оцінка середнього значення mpg: 23.47539
Стандартна похибка середнього (традиційний метод): 0.4413198
```

3) Обчислив стандартну похибку за допомогою бутсрапу

```r
cat("\n-------\n")
bootstrap_mean <- function(data, indices) {
  return(mean(data[indices]))
}
set.seed(variant)
boot_mean <- boot(data = modified_auto_data$mpg, statistic = bootstrap_mean, R = 1000)
cat("Стандартна похибка середнього (бутстрап):", sd(boot_mean$t), "\n")
```

```
-------
Стандартна похибка середнього (бутстрап): 0.430894
```

Ці значення дуже близькі, що свідчить про узгодженість обох методів для цього набору даних. У випадках, де дані розподілені нормально і мають достатній розмір вибірки, традиційна оцінка стандартної похибки часто дає близькі результати до бутстрапу.

4) Обчислив оцінку для медіани та десятого процентиля змінної mpg.

```r
cat("\n-------\n")
bootstrap_median <- function(data, indices) {
  return(median(data[indices]))
}
set.seed(variant)
boot_median <- boot(data = modified_auto_data$mpg, statistic = bootstrap_median, R = 1000) # nolint
cat("Оцінка медіани mpg:", median(modified_auto_data$mpg), "\n")
cat("Стандартна похибка медіани (бутстрап):", sd(boot_median$t), "\n")

bootstrap_percentile_10 <- function(data, indices) {
  return(quantile(data[indices], probs = 0.1))
}

set.seed(variant)
boot_percentile_10 <- boot(data = modified_auto_data$mpg, statistic = bootstrap_percentile_10, R = 1000)
cat("Оцінка десятого процентиля mpg:", quantile(modified_auto_data$mpg, probs = 0.1), "\n") # nolint
cat("Стандартна похибка десятого процентиля (бутстрап):", sd(boot_percentile_10$t), "\n") # nolint
```

```
-------
Оцінка медіани mpg: 22.5
Стандартна похибка медіани (бутстрап): 0.7932495
Оцінка десятого процентиля mpg: 14
Стандартна похибка десятого процентиля (бутстрап): 0.3512469
```

Завдання №3

1) Створив змодельований набір даних

```r
# Завдання №3
library(boot)

x <- rnorm(100)
y <- variant * x - ((redundant * 40) / variant) * x^2 + rnorm(100)
```

2) Написав функцію для обчислення тестової помилки для кожної моделі

```r
calculate_loocv <- function(formula, data) {
  model <- glm(formula, data = data)
  cv_result <- cv.glm(data, model, K = nrow(data))
  return(cv_result$delta[1])
}
```

3) Задав 4 моделі й обрахував для них тестову помилку методом LOOCV

```r
data <- data.frame(x = x, y = y)

loocv_error_1 <- calculate_loocv(y ~ poly(x, 1), data)
loocv_error_2 <- calculate_loocv(y ~ poly(x, 2), data)
loocv_error_3 <- calculate_loocv(y ~ poly(x, 3), data)
loocv_error_4 <- calculate_loocv(y ~ poly(x, 4), data)

cat("Тестова помилка LOOCV для моделі з x:", loocv_error_1, "\n")
cat("Тестова помилка LOOCV для моделі з x та x^2:", loocv_error_2, "\n")
cat("Тестова помилка LOOCV для моделі з x, x^2 та x^3:", loocv_error_3, "\n")
cat("Тестова помилка LOOCV для моделі з x, x^2, x^3 та x^4:", loocv_error_4, "\n")
```

Результати:

```
Тестова помилка LOOCV для моделі з x: 411.2865
Тестова помилка LOOCV для моделі з x та x^2: 0.9296603
Тестова помилка LOOCV для моделі з x, x^2 та x^3: 0.9607402
Тестова помилка LOOCV для моделі з x, x^2, x^3 та x^4: 1.010574
```

Найменша тестова помилка у моделі №2 = 0.9296603.

Це результат, який можна очікувати, виходячи з форми змодельованих даних.

Перша модель: Лінійна модель (тільки зі змінною x) має дуже високу тестову помилку (411.29), оскільки вона не враховує квадратичний компонент. Через це її здатність відтворити структуру даних є низькою.

Друга модель: Модель із змінними x та x^2 добре підходить до даних, оскільки вона враховує квадратичну залежність. Ця модель найкраще відтворює залежність y від x, що відображається у найменшій тестовій помилці LOOCV.

Третя та четверта моделі: Хоча ці моделі включають додаткові змінні x^3 і x^4, вони не покращують узгодженість із даними та навіть трохи збільшують тестову помилку. Це може бути результатом додаткового шуму від вищих степенів, який призводить до перенавчання. Через це більш складні моделі не додають значної користі, а навпаки, збільшують помилку.

Отже, модель із x та x^2 є оптимальною для відображення структури даних.