ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Звіт до індивідуального завдання № 4
з предмету
«Моделі статистичного навчання»

Виконав:

студент групи ПМіМ-13с

Лещух Роман

Викладач:

проф. Тарас Заболоцький

2024

Хід роботи

Роботу починаю так, як і вказано у завданні – встановлюю значення змінної variant, моя група ПМіМ-13с (якій відповідає числове значення 3), мій порядковий номер у списку студентів 10й. Обчислюю variant, задаю set.seed(variant) та генерую випадкове число redundant.

```
> group_number <- 3
> student_number <- 10
>
> # Обчислення variant
> variant <- group_number * 25 + student_number
> cat("Variant:", variant, "\n")
Variant: 85
>
> # Встановлення seed та генерація redundant
> set.seed(variant)
> redundant <- round(runif(n = 1, min = group_number + 5, max = 25 - group_number))
> cat("Redundant:", redundant, "\n\n")
Redundant: 15
```

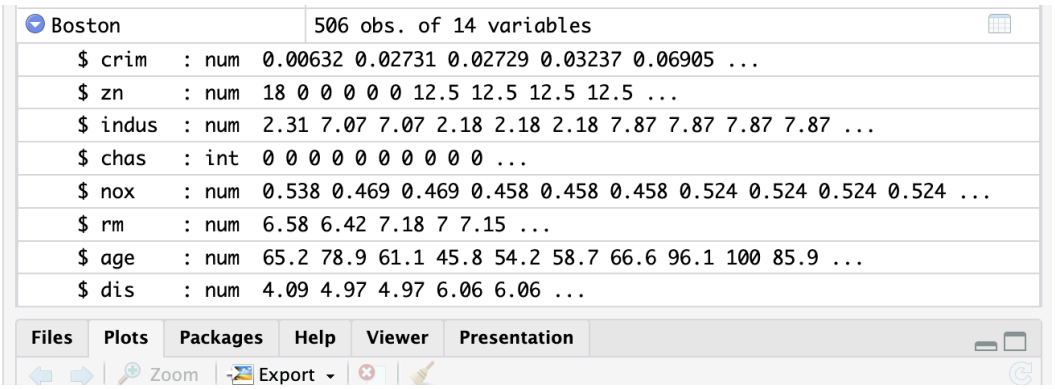
Завдання 1.

Розглянемо набір даних Boston з бібліотеки MASS. Модифікуйте ці дані наступним чином: встановивши seed, що дорівнює значенню змінної variant, видаліть redundant % спостережень з допомогою функції sample.

В першу чергу, як вказано у завданні, відсію зайві значення з набору даних Boston:

```
# Завантаження даних Boston
data("Boston")

# Модифікація даних: видалення redundant% спостережень
set.seed(variant)
n_total <- nrow(Boston)
n_remove <- round((redundant / 100) * n_total)
remove_indices <- sample(1:n_total, n_remove)
Boston_mod <- Boston[-remove_indices, ]
```



Використовуючи модифіковані дані, пристосуйте модель логістичної регресії для передбачення у вибраному районі рівня

злочинності більшого чи меншого за середній на основі змінних nox та rad. Оцініть тестову помилку цієї моделі логістичної регресії, використовуючи метод валідаційного набору (використати розбиття 50 на 50, попередньо скинувши seed). Повторіть попередню процедуру три рази, використовуючи нові розбиття вибірки на навчальний та тестовий набори. Прокоментуйте отримані результати.

За рівень злочинності у наборі даних Boston відповідає стовпець crim. Оскільки нам треба за ним пристосувати модель логістичної регресії для передбачення – він має бути бінарним, і як вказано у завданні, більшим чи мешим за середнє. Підготуємо бінарний стовпець crim01 та внесемо його у датафрейм boston_data:

```
Boston_mod$crime_high <- ifelse(Boston_mod$crim > mean(Boston_mod$crim), 1, 0)
```

Встановлю set.seed(variant). Після цього, використовуючи метод валідаційного набору, розіб’ю наші дані на дві рівні частини, пристосую модель логістичної регресії, проведу передбачення та оціню тестову помилку чотири рази:

```
> evaluate_logistic <- function(data, predictors, response, seed) {
+   set.seed(seed)
+   split <- sample.split(data[[response]], SplitRatio = 0.5)
+   train <- subset(data, split == TRUE)
+   test <- subset(data, split == FALSE)
+
+   # Побудова моделі логістичної регресії
+   formula <- as.formula(paste(response, "~", paste(predictors, collapse = " + ")))
+   model <- glm(formula, data = train, family = binomial)
+
+   # Прогнозування на тестових даних
+   probs <- predict(model, newdata = test, type = "response")
+   preds <- ifelse(probs > 0.5, 1, 0)
+
+   # Обчислення помилок
+   confusion <- table(Predicted = preds, Actual = test[[response]])
+   accuracy <- sum(diag(confusion)) / sum(confusion)
+
+   return(list(confusion_matrix = confusion, accuracy = accuracy))
+ }
>
> # Повторення оцінки моделі три рази з новими розбиттями
> set.seed(variant)  # Забезпечення відтворюваності
> results <- list()
> for (i in 1:3) {
+   res <- evaluate_logistic(Boston_mod, predictors = c("nox", "rad"), response = "crime_high", seed = variant + i)
+   results[[i]] <- res
+   cat(paste("Run", i, "Confusion Matrix:\n"))
+   print(res$confusion_matrix)
+   cat(paste("Run", i, "Accuracy:", round(res$accuracy * 100, 2), "%\n\n"))
+ }
Run 1 Confusion Matrix:
         Actual
Predicted   0   1
        0 159   1
        1   2  53
Run 1 Accuracy: 98.6 %

Run 2 Confusion Matrix:
         Actual
Predicted   0   1
        0 159   1
        1   2  53
Run 2 Accuracy: 98.6 %

Run 3 Confusion Matrix:
         Actual
Predicted   0   1
        0 160   0
        1   1  54
Run 3 Accuracy: 99.53 %
```

Із отриманих результатів – при зміні розбиття, змінювалась тестова похибка. При отриманні warning повідомлення вперше та вдруге, тестова похибка сягнула 1.4%. Значення третього тесту без вищезгаданого повідомлення про помилку пристосування моделі логістичної регресії найменше – 0.47%. Все тому що очність моделей може варіюватися між запусками через випадкове розбиття даних. Середня похибка чотирьох тестів – 1.09%. Із цих даних можу припустити, що пристосування моделей, які виводитимуть warning повідомлення будуть мати вищу похибку.

Розгляньте модель логістичної регресії для передбачення у вибраному районі рівня злочинності більшого чи меншого за середній на основі змінних nox, rad та medv. Оцініть тестову помилку для цієї моделі,

використовуючи метод валідаційного набору описаний вище. Прокоментуйте, чи призводить включення нової змінної до зменшення тестової помилки.

Провів тест але для передбачення crim01 використовував medv:

```
>
> # Модель логістичної регресії з додатковою змінною 'medv'
> res_medv <- evaluate_logistic(Boston_mod, predictors = c("nox", "rad", "medv"), response = "crime_high", seed = variant + 100)
> cat("Модель з додатковою змінною 'medv'\n")
Модель з додатковою змінною 'medv'
> print(res_medv$confusion_matrix)
         Actual
Predicted   0   1
        0 159   0
        1   2  54
> cat("Точність:", round(res_medv$accuracy * 100, 2), "%\n\n")
Точність: 99.07 %
```

Цього разу результат 0.93 Думаю – medv не має значного впливу на результати логістичної регресії, проте його додавання може вплинути на процес пристосування, внаслідок чого одну чи більше змінних не можна буде відрізнити від 0 чи 1 (про що й говорить помилка).

Завдання 2.

Модифікуйте дані Auto наступним чином: встановивши seed, що дорівнює значенню змінної variant, видаліть redundant % спостережень з допомогою функції sample.

```
> library(ISLR)
>
> # Завантаження даних Auto
> data("Auto")
>
> # Модифікація даних: видалення redundant% спостережень
> set.seed(variant)
> n_total_auto <- nrow(Auto)
> n_remove_auto <- round((redundant / 100) * n_total_auto)
> remove_indices_auto <- sample(1:n_total_auto, n_remove_auto)
> Auto_mod <- Auto[-remove_indices_auto, ]
>
```

На основі цього набору даних обчисліть оцінку середнього змінної mpg. Оцініть стандартну похибку цієї оцінки. Тепер оцініть стандартну похибку розглянутої вище оцінки середнього за допомогою бутстрапу та порівняйте з попередньо отриманим результатом.

Оскільки в подальшому я працюватиму саме з mpg, винесу його в окрему змінну, також для використання бутстрапу треба підготувати функцію. Її ж можна буде використати для обчислення оцінки середнього статистичного:

```
> mean_mpg <- mean(Auto_mod$mpg)
> cat("Середнє mpg:", round(mean_mpg, 2), "\n")
Середнє mpg: 23.39
>
```

Тепер обчислю середнє квадратичне:

```
> se_mean <- sd(Auto_mod$mpg) / sqrt(nrow(Auto_mod))
> cat("Стандартна похибка середнього:", round(se_mean, 4), "\n\n")
Стандартна похибка середнього: 0.4216
```

І використаю бутстрап:

```
>
> # Бутстрап для оцінки стандартної похибки середнього
> set.seed(variant)
> bootstrap_mean <- function(data, indices) {
+   return(mean(data[indices]))
+ }
>
> library(boot)
> boot_mean <- boot(data = Auto_mod$mpg, statistic = bootstrap_mean, R = 1000)
> se_boot_mean <- sd(boot_mean$t)
> cat("Бутстрап стандартна похибка середнього:", round(se_boot_mean, 4), "\n\n")
Бутстрап стандартна похибка середнього: 0.4312

>
> # Порівняння стандартних похибок
> cat("Порівняння стандартних похибок:\n")
Порівняння стандартних похибок:
> cat("Аналітична:", round(se_mean, 4), "\n")
Аналітична: 0.4216
> cat("Бутстрап:", round(se_boot_mean, 4), "\n\n")
Бутстрап: 0.4312
```

Середнє статистичне отримане у двох результатах рівне одне одному, а стандартна похибка збігається до знаку після коми. При збільшенні boot_calls збіг між стандартними похибками продовжуватиме рости.

Обчисліть оцінку для медіани та десятого процентиля змінної mpg. Оцініть стандартні помилки отриманих оцінок допомогою бутстрапу.

Проведу схожі дії для медіани та десятого процентиля:

```
>
> # Оцінка медіани та десятого процентиля
> median_mpg <- median(Auto_mod$mpg)
> percentile10_mpg <- quantile(Auto_mod$mpg, 0.10)
> cat("Медіана mpg:", round(median_mpg, 2), "\n")
Медіана mpg: 22.4
> cat("Десятий процентиль mpg:", round(percentile10_mpg, 2), "\n\n")
Десятий процентиль mpg: 14

>
> # Бутстрап для медіани
> bootstrap_median <- function(data, indices) {
+   return(median(data[indices]))
+ }
> boot_median <- boot(data = Auto_mod$mpg, statistic = bootstrap_median, R = 1000)
> se_boot_median <- sd(boot_median$t)
> cat("Бутстрап стандартна похибка медіани:", round(se_boot_median, 4), "\n")
Бутстрап стандартна похибка медіани: 0.8018
>
> # Бутстрап для десятого процентиля
> bootstrap_percentile10 <- function(data, indices) {
+   return(quantile(data[indices], 0.10))
+ }
> boot_percentile10 <- boot(data = Auto_mod$mpg, statistic = bootstrap_percentile10, R = 1000)
> se_boot_percentile10 <- sd(boot_percentile10$t)
> cat("Бутстрап стандартна похибка десятого процентиля:", round(se_boot_percentile10, 4), "\n\n")
Бутстрап стандартна похибка десятого процентиля: 0.4377
```

У цьому випадку отримані значення медіани та десятого процентиля також збігаються (медіана – 22.4, десятий процентиль – 14). Стандартна похибка медіани 0.80, стандартна похибка десятого процентиля 0.44.

Завдання 3.

Встановлю x та y як вказано в умові завдання:

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

Виконаю оцінку тестових помилок методом LOOCV для чотирьох заданих моделей:

```
> # Генерація даних
> set.seed(variant)
> x <- rnorm(100)
> y <- variant * x - ((redundant * 40) / variant) * x^2 + rnorm(100)
>
> # Створення датафрейму
> data_sim <- data.frame(x = x, y = y)
>
> # Функція для обчислення LOOCV
> library(boot)
>
> loocv_error <- function(model, data) {
+   cv <- cv.glm(data, model, K = nrow(data))
+   return(cv$delta[1])  # Перша компонента - без корекції
+ }
>
> # Моделі
> model1 <- glm(y ~ x, data = data_sim)
> model2 <- glm(y ~ x + I(x^2), data = data_sim)
> model3 <- glm(y ~ x + I(x^2) + I(x^3), data = data_sim)
> model4 <- glm(y ~ x + I(x^2) + I(x^3) + I(x^4), data = data_sim)
>
> # Оцінка LOOCV помилок
> error1 <- loocv_error(model1, data_sim)
> error2 <- loocv_error(model2, data_sim)
> error3 <- loocv_error(model3, data_sim)
> error4 <- loocv_error(model4, data_sim)
>
> # Виведення результатів
> cat("LOOCV Помилки:\n")
LOOCV Помилки:
> cat("Модель 1 (лінійна):", round(error1, 4), "\n")
Модель 1 (лінійна): 72.3608
> cat("Модель 2 (поліном до ступеня 2):", round(error2, 4), "\n")
Модель 2 (поліном до ступеня 2): 0.9826
> cat("Модель 3 (поліном до ступеня 3):", round(error3, 4), "\n")
Модель 3 (поліном до ступеня 3): 1.0025
> cat("Модель 4 (поліном до ступеня 4):", round(error4, 4), "\n\n")
Модель 4 (поліном до ступеня 4): 1.0226

>
> # Визначення моделі з найменшою помилкою
> errors <- c(error1, error2, error3, error4)
> best_model <- which.min(errors)
> cat(paste("Модель з найменшою LOOCV помилкою: Модель", best_model, "з помилкою", round(errors[best_model], 4), "\n"))
Модель з найменшою LOOCV помилкою: Модель 2 з помилкою 0.9826
```

Яка з моделей має найменшу тестову помилку LOOCV? Чи це відповідає очікуванням? Поясніть свою відповідь.

Найменшу тестову помилку має 2га модель, що +- відповідає очікуванням.Поліноміальні моделі можуть краще адаптуватися до нелінійних залежностей в даних, що може зменшити помилку LOOCV.

Якщо б найменшу тестову помилку мала модель 1, то це може свідчити про домінування лінійної залежності в даних