Упражнение №5

Галиченко Евгений

23 03 2020

Постановка задачи:

1. Оценить стандартную ошибку модели для линейных регрессионных моделей из упражнения 4 (варианты ниже): а) со всеми объясняющими переменными; б) только с непрерывными объясняющими переменными:

* методом проверочной выборки с долей обучающей 50%;
* методом LOOCV;
* k-кратной кросс-валидацией с и .

Выбрать лучшую модель по минимуму ошибки. Все ли методы кросс-валидации сходятся на одной и той же модели?

1. Оценить стандартные ошибки параметров лучшей модели регрессии методом бутстрепа. Сравнить с оценками стандартных ошибок параметров по МНК.

Прислать на почту преподавателя ссылки:

* на html-отчёт с видимыми блоками кода (блоки кода с параметром echo = T), размещённый на <rpubs.com>.
* на код, генерирующий отчёт, в репозитории на <github.com>.

В текст отчёта включить постановку задачи и ответы на вопросы задания.

## Вариант 4

*Модели*: Линейная регрессия.  
*Данные*: `Auto {ISLR}’.

Набор данных Auto содержит переменные:

* mpg - миль на галлон;
* weight – вес автомобиля (кг.);
* acceleration – время ускорения от 0 до 60 миль в час (сек.);
* year – модельный год (по модулю 100)
* cylinders - количество цилиндров от 4 до 8

# Пакеты  
library('knitr') # Генерация отчёта  
library('ISLR') # Набор данных Auto  
library('GGally') # Матричные графики  
library('boot') # Расчёт ошибки с кросс-валидацией  
  
my.seed <- 1 # Константа для ядра  
  
# Загрузка данных Auto  
data('Auto')  
# Отбор необходимых данных для построения моделей  
Auto <- Auto[,c('mpg', 'weight', 'acceleration', 'year', 'cylinders'), drop = F]

Рассмотрим данные с характеристиками автомобилей auto из пакета ISLR. Скопируем таблицу во фрейм DF.carseats для дальнейших манипуляций.

# Записываем данные во фрейм  
DF.auto <- Auto  
  
# Отобразим первые записи  
head(DF.auto)

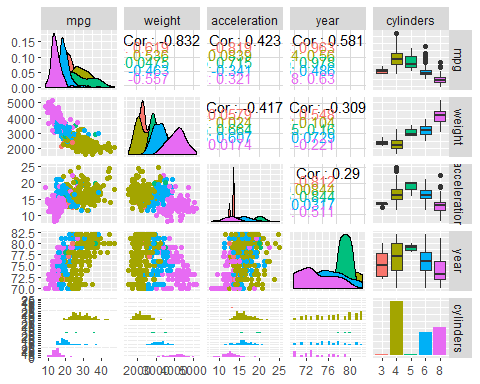
## mpg weight acceleration year cylinders  
## 1 18 3504 12.0 70 8  
## 2 15 3693 11.5 70 8  
## 3 18 3436 11.0 70 8  
## 4 16 3433 12.0 70 8  
## 5 17 3449 10.5 70 8  
## 6 15 4341 10.0 70 8

# Описательные статистики  
summary(DF.auto)

## mpg weight acceleration year cylinders   
## Min. : 9.00 Min. :1613 Min. : 8.00 Min. :70.00 Min. :3.000   
## 1st Qu.:17.00 1st Qu.:2225 1st Qu.:13.78 1st Qu.:73.00 1st Qu.:4.000   
## Median :22.75 Median :2804 Median :15.50 Median :76.00 Median :4.000   
## Mean :23.45 Mean :2978 Mean :15.54 Mean :75.98 Mean :5.472   
## 3rd Qu.:29.00 3rd Qu.:3615 3rd Qu.:17.02 3rd Qu.:79.00 3rd Qu.:8.000   
## Max. :46.60 Max. :5140 Max. :24.80 Max. :82.00 Max. :8.000

В таблице данных 392 наблюдений и 5 переменных, среди которых есть непрерывные количественные и одна дискретная (cylinders, количество цилиндров в атомобиле (от 4 до 8)). Построим графики разброса, показав фактор cylinders цветом. Зависимой переменной модели будет mpg, её покажем в первой строке / столбце матричного графика.

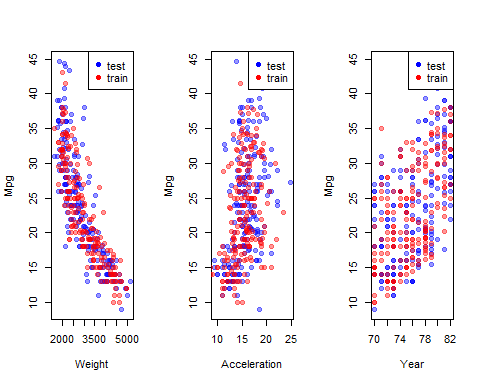
# переведем переменную cylinders в фактор  
DF.auto$cylinders <- as.factor(DF.auto$cylinders)  
  
# Графики разброса, цвет - количество цилиндров  
ggpairs(DF.auto, ggplot2::aes(color = cylinders))



## Метод проверочной выборки

Он состоит в том, что мы отбираем одну тестовую выборку и будем считать на ней ошибку модели.

# Общее число наблюдений  
n <- nrow(DF.auto)  
  
# Доля обучающей выборки  
train.percent <- 0.5  
  
# Выбрать наблюдения в обучающую выборку  
set.seed(my.seed)  
inTrain <- sample(n, n \* train.percent)  
  
# Рисуем разными цветами обучающую и тестовую (для непрерывных переменных)  
  
# Переменная weight  
par(mfrow = c(1, 3))  
plot(DF.auto$weight[inTrain], DF.auto$mpg[inTrain],  
 xlab = 'Weight', ylab = 'Mpg', pch = 21,  
 col = rgb(0, 0, 1, alpha = 0.4), bg = rgb(0, 0, 1, alpha = 0.4))  
points(DF.auto$weight[-inTrain], DF.auto$mpg[-inTrain],  
 pch = 21, col = rgb(1, 0, 0, alpha = 0.4),   
 bg = rgb(1, 0, 0, alpha = 0.4))  
legend('topright',   
 pch = c(16, 16), col = c('blue', 'red'), legend = c('test', 'train'))  
  
# Переменная acceleration  
plot(DF.auto$acceleration[inTrain], DF.auto$mpg[inTrain],  
 xlab = 'Acceleration', ylab = 'Mpg', pch = 21,  
 col = rgb(0, 0, 1, alpha = 0.4), bg = rgb(0, 0, 1, alpha = 0.4))  
points(DF.auto$acceleration[-inTrain], DF.auto$mpg[-inTrain],  
 pch = 21, col = rgb(1, 0, 0, alpha = 0.4),   
 bg = rgb(1, 0, 0, alpha = 0.4))  
legend('topright',   
 pch = c(16, 16), col = c('blue', 'red'), legend = c('test', 'train'))  
  
# Переменная year  
plot(DF.auto$year[inTrain], DF.auto$mpg[inTrain],  
 xlab = 'Year', ylab = 'Mpg', pch = 21,  
 col = rgb(0, 0, 1, alpha = 0.4), bg = rgb(0, 0, 1, alpha = 0.4))  
points(DF.auto$year[-inTrain], DF.auto$mpg[-inTrain],  
 pch = 21, col = rgb(1, 0, 0, alpha = 0.4),   
 bg = rgb(1, 0, 0, alpha = 0.4))  
legend('topright',   
 pch = c(16, 16), col = c('blue', 'red'), legend = c('test', 'train'))



par(mfrow = c(1, 1))

Построим модели для проверки точности. Вид моделей:

а) Со всеми объясняющими переменными

б) Только с непрерывными объясняющими переменными

**Линейная модель 1**: .

# Присоединить таблицу с данными: названия стоблцов будут доступны напрямую  
attach(DF.auto)  
  
# Подгонка линейной модели на обучающей выборке  
fit.lm.1 <- lm(mpg ~ weight + acceleration + year + cylinders, subset = inTrain)  
  
# Считаем MSE на тестовой выборке  
mean((mpg[-inTrain] - predict(fit.lm.1, DF.auto[-inTrain, ]))^2)

## [1] 9.064563

# Отсоединить таблицу с данными  
detach(DF.auto)

# Сохраняем ошибку модели (MSE) на проверочной выборке  
err.test <- mean((DF.auto$mpg[-inTrain] - predict(fit.lm.1,   
 DF.auto[-inTrain, ]))^2)  
  
# Сохранять все ошибки будем в один вектор, присваиваем имя первому элементу  
# (имя - степень объясняющей переменной)  
names(err.test) <- 1

**Линейная модель 2**:

# Присоединить таблицу с данными: названия стоблцов будут доступны напрямую  
attach(DF.auto)  
  
# Подгонка линейной модели на обучающей выборке  
fit.lm.2 <- lm(mpg ~ weight + acceleration + year, subset = inTrain)  
  
# Считаем MSE на тестовой выборке  
mean((mpg[-inTrain] - predict(fit.lm.2, DF.auto[-inTrain, ]))^2)

## [1] 10.34423

# Отсоединить таблицу с данными  
detach(DF.auto)

# Сохраняем ошибку модели (MSE) на проверочной выборке  
err.test <- c(err.test,  
 mean((DF.auto$mpg[-inTrain] - predict(fit.lm.2,  
 DF.auto[-inTrain, ]))^2))  
  
# Имя второго элемента вектора  
names(err.test)[length(err.test)] <- 2

### Перекрёстная проверка по отдельным наблюдениям (LOOCV)

Это самый затратный в вычислительном плане метод, но и самый надёжный в плане оценки ошибки вне выборки. Попробуем применить его к линейной модели 1.

# Подгонка линейной модели на обучающей выборке  
fit.glm1 <- glm(mpg ~ weight + acceleration + year + cylinders, data = DF.auto)  
  
# Считаем LOOCV-ошибку  
cv.err.loocv <- cv.glm(DF.auto, fit.glm1)$delta[1]  
  
# Сохранять все ошибки будем в один вектор, присваиваем имя первому элементу  
# (имя -- степень объясняющей переменной)  
names(cv.err.loocv) <- 1

Теперь оценим точность линейной модели 2.

# Подгонка линейной модели на обучающей выборке  
fit.glm2 <- glm(mpg ~ weight + acceleration + year, data = DF.auto)  
  
# Считаем LOOCV-ошибку  
cv.err.loocv <- c(cv.err.loocv, cv.glm(DF.auto, fit.glm2)$delta[1])  
  
# Сохранять все ошибки будем в один вектор, присваиваем имя второму элементу  
names(cv.err.loocv)[length(cv.err.loocv)] <- 2  
  
# результат  
cv.err.loocv

## 1 2   
## 10.73184 11.90440

### k-кратная перекрёстная проверка

K-кратная кросс-валидация - компромисс между методом проверочной выборки и LOOCV. Оценка ошибки вне выборки ближе к правде, по сравнению с проверочной выборкой, а объём вычислений меньше, чем при LOOCV. Проведём 5-кратную кросс-валидацию моделей 1 и 2.

# Оценим точность линейных моделей 1 и 2  
# Вектор с ошибками по 5-кратной кросс-валидации  
cv.err.k.fold5 <- rep(0, 2)  
  
# Имена элементов вектора  
names(cv.err.k.fold5) <- 1:2  
  
# Оценка модели 1  
fit.glm <- glm(mpg ~ weight + acceleration + year + cylinders, data = DF.auto)  
# Расчёт ошибки  
cv.err.k.fold5[1] <- cv.glm(DF.auto, fit.glm, K = 5)$delta[1]  
  
# Оценка модели 2  
fit.glm <- glm(mpg ~ weight + acceleration + year, data = DF.auto)  
# Расчёт ошибки  
cv.err.k.fold5[2] <- cv.glm(DF.auto, fit.glm, K = 5)$delta[1]  
  
# Результат  
cv.err.k.fold5

## 1 2   
## 10.77136 11.92647

Теперь проведём 10-кратную кросс-валидацию моделей 1 и 2.

# Оценим точность линейных моделей 1 и 2  
# Вектор с ошибками по 10-кратной кросс-валидации  
cv.err.k.fold10 <- rep(0, 2)  
  
# Имена элементов вектора  
names(cv.err.k.fold10) <- 1:2  
  
# Оценка модели 1  
fit.glm <- glm(mpg ~ weight + acceleration + year + cylinders, data = DF.auto)  
# Расчёт ошибки  
cv.err.k.fold10[1] <- cv.glm(DF.auto, fit.glm, K = 10)$delta[1]  
  
# Оценка модели 2  
fit.glm <- glm(mpg ~ weight + acceleration + year, data = DF.auto)  
# Расчёт ошибки  
cv.err.k.fold10[2] <- cv.glm(DF.auto, fit.glm, K = 10)$delta[1]  
  
# Результат  
cv.err.k.fold10

## 1 2   
## 10.53472 11.88837

Для определения лучшей модели по стандартной ошибке MSE объединим все полученные результаты в таблицу.

MSE.tbl <- rbind(err.test, cv.err.loocv, cv.err.k.fold5, cv.err.k.fold10)  
colnames(MSE.tbl) <- c('Модель 1', 'Модель 2')  
row.names(MSE.tbl) <- c('Проверочная выборка', 'LOOCV', '5-кратная кросс-валидация', '10-кратная кросс-валидация')  
kable(MSE.tbl)

|  |  |  |
| --- | --- | --- |
|  | Модель 1 | Модель 2 |
| Проверочная выборка | 9.064563 | 10.34423 |
| LOOCV | 10.731843 | 11.90440 |
| 5-кратная кросс-валидация | 10.771358 | 11.92647 |
| 10-кратная кросс-валидация | 10.534719 | 11.88837 |

Опираясь на результаты расчётов с проверочной выборкой, LOOCV и кросс-валидацией ( и ), можно заключить, что стандартная ошибка MSE линейной модели 1 (со всеми объясняющими переменными) оказалась меньше по всем методам кросс-валидации, чем MSE линейной модели 2 (только с непрерывными объясняющими переменными). Таким образом, линейную модель 1 можно считать лучшей: .

## Бутстреп

### Точность оценки параметра регрессии

При построении модели регрессии проблемы в остатках приводят к неверной оценке ошибок параметров. Обойти эту проблему можно, применив для расчёта этих ошибок бутстреп.

# Оценивание точности лучшей линейной регрессионной модели  
  
# Оценить стандартные ошибки параметров модели   
# mpg = beta\_0 + beta\_1 \* weight + beta\_2 \* acceleration + beta\_3 \* year + beta\_4 \* cylinders с помощью бутстрепа,  
# Сравнить с оценками ошибок по МНК  
  
# функция для расчёта коэффициентов ЛР по выборке из данных  
boot.fn <- function(data, index){  
 coef(lm(mpg ~ weight + acceleration + year, data = data, subset = index))  
}  
boot.fn(DF.auto, 1:n)

## (Intercept) weight acceleration year   
## -14.936555253 -0.006554126 0.066359316 0.748446024

# Пример применения функции к бутстреп-выборке  
set.seed(my.seed)  
boot.fn(DF.auto, sample(n, n, replace = T))

## (Intercept) weight acceleration year   
## -17.491016213 -0.006586511 0.178885468 0.760795919

# Применяем функцию boot для вычисления стандартных ошибок параметров  
# (1000 выборок с повторами)  
boot(DF.auto, boot.fn, 1000)

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = DF.auto, statistic = boot.fn, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* -14.936555253 1.014126e-01 4.04929416  
## t2\* -0.006554126 -1.048877e-05 0.00021444  
## t3\* 0.066359316 1.572497e-03 0.08749440  
## t4\* 0.748446024 -1.315632e-03 0.04871285

# Сравним с МНК  
attach(DF.auto)  
summary(lm(mpg ~ weight + acceleration + year))$coef

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -14.936555253 4.055512064 -3.6830257 2.631750e-04  
## weight -0.006554126 0.000229953 -28.5020269 3.451654e-97  
## acceleration 0.066359316 0.070361131 0.9431246 3.462040e-01  
## year 0.748446024 0.050366056 14.8601276 7.371722e-40

detach(DF.auto)

В модели регрессии, для которой проводился расчёт, похоже, не нарушаются требования к остаткам, и оценки стандартных ошибок параметров, рассчитанные по МНК, очень близки к ошибкам этих же параметров, полученных бутстрепом.