Regresja liniowa

Semestr letni 2022/23

Dołączanie pakietów

Dołączanie pakietu spoza zestawu standardowego

```
library(MASS)
```

Pakiety zawierają definicje obiektów, w tym funkcji i zbiorów danych.

Niezainstalowane pakiety można pobrać z repozytorium i zainstalować

```
install.packages("ISLR", dependencies = TRUE)
```

Prosta regresja liniowa

Używamy zbioru danych `Boston` z pakietu `MASS`.

```
names(Boston)
dim(Boston)
?Boston
head(Boston)
```

Dopasowanie (uczenie) modelu liniowego wykonuje się przy pomocy funkcji `lm()`. Postać modelu określa się przy pomocy **formuły** (czyli obiektu klasy `formula`). Modelowi

$$Y = \beta_0 + \beta_1 X + \epsilon$$

odpowiada formuła `Y ~ X`. Poniższe instrukcje są równoważne i oznaczają model

$$medv = \beta_0 + \beta_1 \cdot lstat + \epsilon.$$

```
fit_simple <- lm(Boston$medv ~ Boston$lstat)
fit_simple <- lm(medv ~ lstat, data = Boston)
```

Natomiast poniższa ma działanie szersze

```
attach(Boston)
fit_simple <- lm(medv ~ lstat)
```

Wynikiem w każdym przypadku jest obiekt klasy `lm`, który jest też listą

```
fit_simple
class(fit_simple)
is.list(fit_simple)
names(fit_simple)
```

Składowe obiektu modelu liniowego są dostępne przez indeksowanie typu listowego lub przez odpowiednie funkcje/metody akcesorowe (co jest metodą zalecaną), np.

```
fit_simple$coefficients
coef(fit_simple)
```

Dodatkowe informacje można uzyskać przy pomocy funkcji `summary()`

```
?summary.lm
summary(fit_simple)
```

Funkcja `summary()` zwraca listę (składowa `sigma` to RSE)

```
summaryList <- summary(fit_simple)
summaryList$sigma
summaryList$r.squared
summaryList$fstatistic
```

Przedziały ufności dla współczynników regresji oblicza funkcja `confint()`

```
confint(fit_simple)
```

Funkcja `predict()` oblicza przedziały ufności dla predykcji — zarówno dla przewidywania średniej wartości

```
predict(fit_simple, data.frame(lstat = c(5, 10, 15)), interval = "confidence")
```

jak i dla przewidywania przyszłej wartości

```
predict(fit_simple, data.frame(lstat = c(5, 10, 15)), interval = "prediction")
```

Wykresy prostej regresji liniowej

Prosta regresji na tle danych

```
plot(Boston$lstat, Boston$medv)
abline(fit_simple)
```

Wykresy diagnostyczne

```
# Można poprzedzić instrukcją: par(mfrow = c(2, 2))
plot(fit_simple)
```

Alternatywnie

```
plot(predict(fit_simple), residuals(fit_simple))
plot(predict(fit_simple), rstudent(fit_simple))
```

Identyfikacja obserwacji wpływowych (statystyka “dźwigni” [*leverage*])

```
plot(hatvalues(fit_simple))
which.max(hatvalues(fit_simple))
```

Regresja wielokrotna

Model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon$$

reprezentowany jest przez formułę `Y ~ X1 + X2 + X3`, np.

```
fit_la <- lm(medv ~ lstat + age)
summary(fit_la)
```

Jeśli chcemy wykonać regresję pewnej zmiennej względem wszystkich pozostałych stosuje się składnię (parametr `data` jest tu wymagany)

```
fit_all <- lm(medv ~ ., data = Boston)
summary(fit_all)
```

Regresja z jedną zmienną usuniętą

```
fit_no_age <- lm(medv ~ . - age, data = Boston)
summary(fit_no_age)
```

Alternatywnie można skorzystać z funkcji `update()`

```
fit_no_age2 <- update(fit_all, ~ . - age)
summary(fit_no_age2)
```

Zbiór ufności dla dwóch współczynników można obliczyć korzystając np. z funkcji `ellipse()` z pakietu `ellipse`.

```
library(ellipse)
plot(ellipse(fit_la, which = -1), type = "l")
la_coefs <- coef(fit_la)
points(la_coefs[2], la_coefs[3])
```

Interakcje między zmiennymi

Obecność składnika $X_1 \cdot X_2$ zaznacza się w formule przez człon `X1 : X2`. Składnia `X1 * X2` jest skrótem do `X1 + X2 + X1:X2`. Np.

```
summary(lm(medv ~ lstat * age))
```

Nieliniowe transformacje predyktorów

Model z kwadratową zależnością od `lstat`, czyli

$$medv = \beta_0 + \beta_1 \cdot lstat + \beta_2 \cdot lstat^2 + \epsilon$$

dopasowywany jest następująco (funkcja `I()` jest konieczna ze względu na specjalne znaczenie operatora `^` w formułach)

```
fit_l2 <- lm(medv ~ lstat + I(lstat^2))
summary(fit_l2)
```

Dopasowanie modeli `fit_simple` i `fit_l2` można porównać porównując RSE i R^2. Funkcja `anova()` wykonuje test statystyczny, w którym hipotezą zerową jest jednakowe dopasowanie.

```
anova(fit_simple, fit_l2)
```

Regresja wielomianowa wyższego stopnia może wykorzystywać funkcję `poly()`

```
fit_l5 <- lm(medv ~ poly(lstat, 5))
summary(fit_l5)
```

Logarytmiczna transformacja predyktora

```
summary(lm(medv ~ log(rm)))
```

Predyktory jakościowe

Nowy zbiór danych

```
library(ISLR)
names(Carseats)
dim(Carseats)
head(Carseats)
```

Zbiór `Carseats` zawiera zmienne jakościowe (czynniki) `ShelveLoc`, `Urban` i `US`

```
attach(Carseats)
summary(ShelveLoc)
```

Dla czynników generowane są automatycznie zmienne zastępcze, np.

```
sales_all_ia_fit <- lm(Sales ~ . + Income:Advertising, data = Carseats)
summary(sales_all_ia_fit)
```

Funkcja `contrasts()` pokazuje kodowanie używane przez `R` dla zmiennych zastępczych.

```
contrasts(ShelveLoc)
```