Ajuste de modelos lineales

Juliana Quirós, Alberto (ajuliana@ucm.es)

```
rm(list = ls())
load("E:\\GDrive1\\Uni\\Master\\simulacion\\ejercicios\\ajuste_modelo\\auto-mpg.Rda")

## Codificación
data_train$horsepower <- as.integer(data_train$horsepower)
```

```
## Warning: NAs introducidos por coerción
```

```
data_test$horsepower <- as.integer(data_test$horsepower)

miFit <- function(data, dependiente) {
    formulas <- list()
    modelos <- list()
    formulas_int <- list()
    modelos_int <- list()
    mses <- list()
    mses_int <- list()
    preds <- names(data[, 2:7])  ## Obviamos el nombre del coche por colinealidad
    comb_preds <- list()

    ## Genera todas las combinaciones de 1 a 5 predictores
    for (i in 1:5) {
        comb_preds[[i]] <- combn(preds, i)
    }

    ## Inicializa las listas
    for (i in 1:length(comb_preds)) {
        formulas[[i]] <- vector("list", ncol(comb_preds[[i]]))
        modelos[[i]] <- vector("list", ncol(comb_preds[[i]]))
        formulas_int[[i]] <- vector("list", ncol(comb_preds[[i]]))
        modelos_int[[i]] <- vector("list", ncol(comb_preds[[i]]))
        mses <- vector("list", ncol(comb_preds[[i]]))
        mses_int <- vector("list", ncol(comb_preds[[i]]))
    }
    ## Genera Lms con todas las combos de predictores
    for (i in 1:length(comb_preds)) {
        for (j in 1:ncol(comb_preds[[i]])) {
            formulas[[i]][[j]] <- formula(paste(dependiente,
                paste(comb_preds[[i]][, j], collapse = " + "),
                sep = " ~ "))
            modelos[[i]][[j]] <- lm(formulas[[i]][[j]], data)
            mses[[i]][[j]] <- mean(modelos[[i]][[j]]$residuals^2)
        }
    }
    ## Genera Lms de todas las combos con todas las
    ## interacciones No quiero que explore con un
    ## coeficiente (no hay interaccion)
    for (i in 2:length(comb_preds)) {
        for (j in 1:ncol(comb_preds[[i]])) {
            formulas_int[[i]][[j]] <- formula(paste(dependiente,
                "~", paste(paste0("(", paste(comb_preds[[i]][,
                  j], collapse = " + "), ")", sep = ""), "^ 5")))
            modelos_int[[i]][[j]] <- lm(formulas_int[[i]][[j]],
```

```
                data)
            mses_int[[i]][[j]] <- mean(modelos_int[[i]][[j]]$residuals^2)
        }
    }
    ## Obtener las fórmulas con el mínimo MSE.  Sin
    ## interacciones
    pos_min_mse <- which.min(unlist(mses))
    formulas <- unlist(formulas)
    ## Con interacciones
    pos_min_mse_int <- which.min(unlist(mses_int))
    formulas_int <- unlist(formulas_int)
    cat(sprintf("El MSE mínimo sin interacciones es: %5.2f . Con interacciones es: %5.2f",
        min(unlist(mses)), min(unlist(mses_int))))
    return(c("Sus respectivas fórmulas son:", formulas[pos_min_mse],
        formulas_int[pos_min_mse_int]))  ## mismos indices en todas las listas
}

miFit(data_train, "mpg")
```

```
## El MSE mínimo sin interacciones es: 11.74 . Con interacciones es:  6.14
```

```
## [[1]]
## [1] "Sus respectivas fórmulas son:"
##
## [[2]]
## mpg ~ cylinders + displacement + horsepower + weight + model.year
## <environment: 0x0000026c3380fd78>
##
## [[3]]
## mpg ~ (displacement + horsepower + weight + acceleration + model.year)^5
## <environment: 0x0000026c3380fd78>
```

Aplico el mejor modelo con interacciones al conjunto de tests:

```
mejor_modelo <- lm(mpg ~ (displacement + horsepower + weight +
    acceleration + model.year)^5, data_train)
data_predicts <- predict(mejor_modelo, data_test)

# Calculo el MSE
mean((data_test$mpg - data_predicts)^2)
```

```
## [1] 8.062546
```