Regresion2

Diego Vázquez Zambrano

2023-06-20

#Ejercicio 1

(a) Ajustar un modelo de regresión logística que incluya la edad categorizada y los cuatro potenciales biomarcadores urinarios, con la finalidad de diagnosticar o discriminar pacientes con cáncer frente a pacientes con afecciones pancreáticas no cancerosas. ¿Qué variables nos permiten predecir el riesgo de adenocarcinoma ductal pancreático?

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
pancreas_data <- read.table("pancreas_biomarkers.txt", header = TRUE, sep
= "\t")
disease_filter <- filter(pancreas_data, diagnosis==2 | diagnosis== 3)
#no cancer 2 , cancer 3

disease_filter <- disease_filter %>%
  mutate(diagnosis = ifelse(diagnosis == 2, 0, 1))
#no cancer 0 , cancer 1
```

Modelo de regresión logística

```
log.model <-  glm(diagnosis ~ age_cat + LYVE1 + REG1B + TFF1 +
creatinine, data = disease_filter, family = 'binomial') #family binomial
for logistic regression model
summary(log.model)
```

```
##
## Call:
## glm(formula = diagnosis ~ age_cat + LYVE1 + REG1B + TFF1 + creatinine,
##     family = "binomial", data = disease_filter)
##
```

```
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.660e+00  1.188e+00  -3.081  0.00206 **
## age_cat36-45  1.157e+00  1.230e+00   0.941  0.34681
## age_cat46-55  1.662e+00  1.180e+00   1.409  0.15892
## age_cat56-65  3.032e+00  1.168e+00   2.597  0.00942 **
## age_cat66-75  2.700e+00  1.169e+00   2.311  0.02085 *
## age_cat75+    3.443e+00  1.215e+00   2.833  0.00461 **
## LYVE1         3.140e-01  5.281e-02   5.946 2.74e-09 ***
## REG1B         2.804e-03  1.099e-03   2.550  0.01077 *
## TFF1         -5.978e-05  2.120e-04  -0.282  0.77798
## creatinine   -3.021e-01  2.263e-01  -1.335  0.18176
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 564.02  on 406  degrees of freedom
## Residual deviance: 382.52  on 397  degrees of freedom
## AIC: 402.52
##
## Number of Fisher Scoring iterations: 5
```

Según nuestro modelo, el valor de la intersección ("intercept") exponenciada sería exp(-3.660e+00) o aproximadamente 0.026. Esto significa que cuando todas las demás variables predictoras están en sus niveles de referencia, las probabilidades de tener un diagnóstico de adenocarcinoma de páncreas son aproximadamente 0,026 veces las probabilidades de tener un diagnóstico de afecciones pancreáticas no cancerosas. Luego, fijándonos en las variables vemos varias que nos llaman la atención:

- age_cat56-65: El coeficiente estimado es positivo (3,032), y es estadísticamente significativo (valor p = 0,00942). Esto sugiere que las personas en la categoría de edad de 56 a 65 años tienen mayores probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.

*age_cat66-75: El coeficiente estimado es positivo (2,700), y es estadísticamente significativo (valor p = 0,02085). Esto sugiere que las personas en la categoría de edad de 66 a 75 años tienen mayores probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.

*LYVE1: La estimación del coeficiente es positiva (0,314), y es estadísticamente significativa (p-valor < 0,001). Esto sugiere que un aumento en el biomarcador LYVE1 se asocia con mayores probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.

*REG1B: La estimación del coeficiente es positiva (0,314), y es estadísticamente significativa (p-valor < 0,01), no tanto como el marcador LYVE1. Esto sugiere que un aumento en el biomarcador REG1B se asocia con mayores probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.

El resto de marcadores urinarios fueron correlacionados negativamente y la diferencia no era significativa.

(b) Interpretar los coeficientes que acompañan a edad categorizada y LYVE1.

Es importante tener en cuenta que las siguientes interpretaciones suponen que las demás variables del modelo se mantienen constantes. Los coeficientes proporcionados en el resultado representan los efectos estimados de cada categoría de la variable "age_cat" en la probabilidad LOGARÍTMICA de tener un diagnóstico de 1.

Para interpretar estos coeficientes, los exponenciaremos para obtener la probabilidad (sin logaritmos), que brindan información sobre la diferencia en las probabilidades en comparación con el grupo diagnosticado con una afección no cancerosa.

```
exp(log.model$coefficients)
```

```
##  (Intercept) age_cat36-45 age_cat46-55 age_cat56-65 age_cat66-75
age_cat75+
##   0.02572003   3.18002227   5.27130148  20.74796260  14.88477006
31.27901068
##        LYVE1        REG1B         TFF1   creatinine
##   1.36889167   1.00280761   0.99994023   0.73924167
```

Por tanto, aquí analizaremos los valores exponenciados directamente:

- Por ejemplo, para el primer grupo, calculamos exp(1.157), que es 3.18. Lo cual significa que las personas en el grupo de edad de 36 a 45 años tienen aproximadamente 3.18 veces más probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.
- Las personas en el grupo de edad de 46 a 55 años tienen aproximadamente 5.271 veces más probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.
- Las personas en el grupo de edad de 56 a 65 años tienen aproximadamente 20.748 veces más probabilidades de tener un diagnóstico de adenocarcinoma de páncreas
- Las personas en el grupo de edad de 66 a 75 años tienen aproximadamente 14.885 veces más probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.
- Las personas en el grupo de edad de 75 años o más tienen aproximadamente 31.279 veces más probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.
- Las personas con el marcador LYVE1 en la orina tienen aproximadamente 1.368 veces más probabilidades de tener un diagnóstico de adenocarcinoma de páncreas.

(c) Contrastar si nos podemos quedar con un modelo más reducido que no incluya los biomarcadores creatinina y TFF1. Escribir las hipótesis H0 y H1 de este contraste.

```
#modelo sin cretinina ni TFF1
simple_model <-  glm(diagnosis ~ age_cat + LYVE1 + REG1B, data =
disease_filter, family = 'binomial')

summary(simple_model)

##
## Call:
## glm(formula = diagnosis ~ age_cat + LYVE1 + REG1B, family =
"binomial",
##     data = disease_filter)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0311243  1.1897669  -3.388 0.000704 ***
## age_cat36-45  1.2607495  1.2590860   1.001 0.316672
## age_cat46-55  1.8352456  1.2038156   1.525 0.127378
## age_cat56-65  3.2169977  1.1924837   2.698 0.006981 **
## age_cat66-75  2.8828093  1.1890190   2.425 0.015328 *
## age_cat75+    3.6627647  1.2328682   2.971 0.002969 **
## LYVE1         0.2924045  0.0495417   5.902 3.59e-09 ***
## REG1B         0.0025692  0.0009132   2.813 0.004901 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 564.02  on 406  degrees of freedom
## Residual deviance: 384.83  on 399  degrees of freedom
## AIC: 400.83
##
## Number of Fisher Scoring iterations: 5
```

Como vimos en nuestro log.model, la creatinina y el TFF1 no aumentan significativamente la probabilidad de obtener un diagnóstico de adenocarcinoma de páncreas, por tanto podríamos quedarnos con un modelo reducido como simple_model

Hipótesis nula (H0): Los biomarcadores “creatinina” y “TFF1” no tienen un efecto significativo en la probabilidad de tener un diagnóstico de adenocarcinoma de páncreas, después de considerar el efecto de las demás variables en el modelo.

Hipótesis alternativa (H1): Al menos uno de los biomarcadores (“creatinina” y “TFF1”) tiene un efecto significativo en la probabilidad de tener un diagnóstico de adenocarcinoma de páncreas, después de considerar el efecto de las demás variables en el modelo.

(d) Verificar la suposición de aumento lineal en el log odds del modelo seleccionado en el apartado anterior para los biomarcadores LYVE1 y REG1B. Usar un modelo que agrega el cuadrado LYVE12, y un modelo que agrega el cuadrado REG1B2, uno a la vez. ¿Deberíamos incluir cualquiera de las variables como una función cuadrática? Discutir vuestra conclusión aportando los estadísticos que la soportan.

```
LYVE1_squared <- disease_filter$LYVE1^2
disease_filter$LYVE1_squared <- LYVE1_squared

LYVE1_model <- glm(diagnosis ~ age_cat + LYVE1 + REG1B + LYVE1_squared,
data = disease_filter, family = 'binomial')

REG1B_squared <- disease_filter$REG1B^2
disease_filter$REG1B_squared <- REG1B_squared

REG1B_model <- glm(diagnosis ~ age_cat + LYVE1 + REG1B + REG1B_squared,
data = disease_filter, family = 'binomial')

summary(simple_model)

##
## Call:
## glm(formula = diagnosis ~ age_cat + LYVE1 + REG1B, family =
"binomial",
##     data = disease_filter)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0311243  1.1897669  -3.388 0.000704 ***
## age_cat36-45  1.2607495  1.2590860   1.001 0.316672
## age_cat46-55  1.8352456  1.2038156   1.525 0.127378
## age_cat56-65  3.2169977  1.1924837   2.698 0.006981 **
## age_cat66-75  2.8828093  1.1890190   2.425 0.015328 *
## age_cat75+    3.6627647  1.2328682   2.971 0.002969 **
## LYVE1         0.2924045  0.0495417   5.902 3.59e-09 ***
## REG1B         0.0025692  0.0009132   2.813 0.004901 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 564.02  on 406  degrees of freedom
## Residual deviance: 384.83  on 399  degrees of freedom
## AIC: 400.83
##
## Number of Fisher Scoring iterations: 5

summary(LYVE1_model)
```

```
##
## Call:
## glm(formula = diagnosis ~ age_cat + LYVE1 + REG1B + LYVE1_squared,
##     family = "binomial", data = disease_filter)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -4.2017396  1.2069541  -3.481 0.000499 ***
## age_cat36-45   1.3193891  1.2661132   1.042 0.297375
## age_cat46-55   1.8948671  1.2104349   1.565 0.117479
## age_cat56-65   3.2671843  1.1997369   2.723 0.006464 **
## age_cat66-75   2.9251655  1.1953280   2.447 0.014398 *
## age_cat75+     3.7143104  1.2402500   2.995 0.002746 **
## LYVE1          0.3878762  0.1016950   3.814 0.000137 ***
## REG1B          0.0025524  0.0009032   2.826 0.004715 **
## LYVE1_squared -0.0104317  0.0090294  -1.155 0.247965
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 564.02  on 406  degrees of freedom
## Residual deviance: 383.93  on 398  degrees of freedom
## AIC: 401.93
##
## Number of Fisher Scoring iterations: 5

#El coeficiente para LYVE1_squared no es significativo (valor p = 0.248),
lo que sugiere que incluir el término al cuadrado no mejora
significativamente el modelo.
summary(REG1B_model)

##
## Call:
## glm(formula = diagnosis ~ age_cat + LYVE1 + REG1B + REG1B_squared,
##     family = "binomial", data = disease_filter)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.957e+00  1.152e+00  -3.433 0.000596 ***
## age_cat36-45   1.139e+00  1.231e+00   0.925 0.354765
## age_cat46-55   1.741e+00  1.169e+00   1.489 0.136608
## age_cat56-65   3.085e+00  1.165e+00   2.647 0.008126 **
## age_cat66-75   2.757e+00  1.161e+00   2.375 0.017540 *
## age_cat75+     3.537e+00  1.206e+00   2.933 0.003357 **
## LYVE1          2.854e-01  5.029e-02   5.675 1.39e-08 ***
## REG1B          3.777e-03  1.876e-03   2.013 0.044062 *
## REG1B_squared -1.654e-06  2.161e-06  -0.765 0.444018
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 564.02  on 406  degrees of freedom
## Residual deviance: 384.31  on 398  degrees of freedom
## AIC: 402.31
##
## Number of Fisher Scoring iterations: 5

#El coeficiente para REG1B_squared no es significativo (valor p = 0,444),
lo que sugiere que incluir el término al cuadrado no mejora
significativamente el modelo.
```

(e) Suponemos que tenemos un paciente con afección pancreática, aunque no sabemos si es cancerosa o no, con las siguientes características: edad=68, creatinine=0.5, LYVE1=6, REG1B=140, TFF1=400. Según el modelo seleccionado en el apartado (c) para el diagnóstico de cáncer de páncreas, ¿lo clasificaríamos como afección cancerosa o no cancerosa? Argumentar la respuesta evaluando la posible extrapolación de la observación.

Calculamos la probabilidad logarítmica utilizando la fórmula del modelo simple_model:

log_odds = Intercept + coef(age_cat66-75) * age_cat66-75 + coef(LYVE1) * LYVE1 + coef(REG1B) * REG1B

```
log_odds = -4.0311243 + 2.8828093 * 1 + 0.2924045 * 6 + 0.0025692 * 140

probabilidad <- exp(log_odds)

print(probabilidad)

## [1] 2.626888
```

#Ejercicio 2

Parkinson

```
import.data <-
"http://archive.ics.uci.edu/ml/machine-learning-
databases/parkinsons/telemonitoring/parkinsons_updrs.data"

parkinson <- read.table(url(import.data), sep=",", skip=1)
parkinson <- na.omit(parkinson) #omitimos valores perdidos
names(parkinson) <-
c("subject","age","sex","test_time","motor_UPDRS","total_UPDRS",
"Jitter(%)","Jitter(Abs)","Jitter:RAP","Jitter:PPQ5","Jitter:DDP",
"Shimmer","Shimmer(dB)","Shimmer:APQ3","Shimmer:APQ5","Shimmer:APQ11",
"Shimmer:DDA","NHR","HNR","RPDE","DFA","PPE")
set.seed(123)
```

Separación de datos: training data 80%

```
train_size <- floor(0.8 * nrow(parkinson))
train_indices <- sample(seq_len(nrow(parkinson)), size = train_size)
data.train <- parkinson[train_indices, ]
data.test <- parkinson[-train_indices, ]
# Eliminar observaciones con valores perdidos
data.train <- na.omit(data.train)
data.test <- na.omit(data.test)
```

##(a) Regresión lineal con todas las 16 variables predictoras. Indicar los posibles problemas, que no consideramos, al prescindir del factor "sujeto".

```
model <- lm(total_UPDRS ~ motor_UPDRS + `Jitter(Abs)` + `Jitter:RAP` +
`Jitter:PPQ5` + `Jitter:DDP` + Shimmer + `Shimmer(dB)` +
`Shimmer:APQ3` + `Shimmer:APQ5` + `Shimmer:APQ11` +
`Shimmer:DDA` + NHR + HNR + RPDE + DFA + PPE,
data = data.train)
summary(model)
```

```
##
## Call:
## lm(formula = total_UPDRS ~ motor_UPDRS + `Jitter(Abs)` + `Jitter:RAP`
+
##     `Jitter:PPQ5` + `Jitter:DDP` + Shimmer + `Shimmer(dB)` +
##     `Shimmer:APQ3` + `Shimmer:APQ5` + `Shimmer:APQ11` + `Shimmer:DDA`
+
##     NHR + HNR + RPDE + DFA + PPE, data = data.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.1876 -2.1166 -0.3131  1.4190 12.0978
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.554e+00  1.145e+00   4.848 1.28e-06 ***
## motor_UPDRS      1.254e+00  6.320e-03 198.363  < 2e-16 ***
## `Jitter(Abs)`    2.964e+04  3.375e+03   8.783  < 2e-16 ***
## `Jitter:RAP`    -2.460e+03  1.790e+04  -0.137 0.890690
## `Jitter:PPQ5`   -3.275e+01  6.162e+01  -0.532 0.595090
## `Jitter:DDP`     7.808e+02  5.968e+03   0.131 0.895910
## Shimmer         -4.236e+01  2.444e+01  -1.733 0.083173 .
## `Shimmer(dB)`    7.813e-01  1.881e+00   0.415 0.677856
## `Shimmer:APQ3`  -3.523e+03  1.789e+04  -0.197 0.843905
## `Shimmer:APQ5`   1.016e+02  2.200e+01   4.617 4.00e-06 ***
## `Shimmer:APQ11` -4.513e+01  1.034e+01  -4.363 1.31e-05 ***
## `Shimmer:DDA`    1.176e+03  5.964e+03   0.197 0.843665
## NHR             -1.201e+01  2.332e+00  -5.148 2.74e-07 ***
## HNR             -8.389e-02  2.605e-02  -3.220 0.001289 **
## RPDE             3.559e+00  6.880e-01   5.173 2.40e-07 ***
## DFA             -3.988e+00  8.814e-01  -4.525 6.19e-06 ***
```

```
## PPE             -3.844e+00  1.079e+00  -3.562 0.000372 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.33 on 4683 degrees of freedom
## Multiple R-squared:  0.9045, Adjusted R-squared:  0.9042
## F-statistic:  2772 on 16 and 4683 DF,  p-value: < 2.2e-16

coefficients <- coef(model)

adj_r_squared <- summary(model)$adj.r.squared

train_predictions <- predict(model, newdata = data.train)
train_rmse <- sqrt(mean((data.train$total_UPDRS - train_predictions)^2))

test_predictions <- predict(model, newdata = data.test)
test_rmse <- sqrt(mean((data.test$total_UPDRS - test_predictions)^2))

cat("Número de variables predictoras:", length(coefficients) - 1, "\n")

## Número de variables predictoras: 16

cat("R^2 ajustado:", adj_r_squared, "\n")

## R^2 ajustado: 0.9041695

cat("RMSE para el grupo de ajuste:", train_rmse, "\n")

## RMSE para el grupo de ajuste: 3.323669

cat("RMSE para el grupo de prueba:", test_rmse, "\n")

## RMSE para el grupo de prueba: 3.471948

model <- lm(total_UPDRS ~ motor_UPDRS + `Jitter(Abs)` + `Jitter:RAP` +
`Jitter:PPQ5` + `Jitter:DDP` + Shimmer + `Shimmer(dB)` +
`Shimmer:APQ3` + `Shimmer:APQ5` + `Shimmer:APQ11` +
`Shimmer:DDA` + NHR + HNR + RPDE + DFA + PPE + subject,
data = data.train)
coefficients <- coef(model)

adj_r_squared <- summary(model)$adj.r.squared

train_predictions <- predict(model, newdata = data.train)
train_rmse <- sqrt(mean((data.train$total_UPDRS - train_predictions)^2))

test_predictions <- predict(model, newdata = data.test)
test_rmse <- sqrt(mean((data.test$total_UPDRS - test_predictions)^2))

cat("Número de variables predictoras:", length(coefficients) - 1, "\n")
```

```
## Número de variables predictoras: 17
```

```
cat("R^2 ajustado:", adj_r_squared, "\n")
```

```
## R^2 ajustado: 0.9047628
```

```
cat("RMSE para el grupo de ajuste:", train_rmse, "\n")
```

```
## RMSE para el grupo de ajuste: 3.313011
```

```
cat("RMSE para el grupo de prueba:", test_rmse, "\n")
```

```
## RMSE para el grupo de prueba: 3.464818
```

```
#Cuando quitamos el sujeto vemos que el R2 se reduce a 0.9041695 de
0.9047628, es decir, que añadiendo el sujeto obtenemos un modelo con
mejor ajuste. Sin embargo, esto puede ser debido a un sobreajuste del
modelo.  Dado que cada sujeto tiene múltiples observaciones, es posible
que exista una dependencia entre las observaciones dentro del mismo
sujeto. Ignorar esto puede conducir a resultados incorrectos y como se ha
mencionado anteriormente, un sobreajuste del modelo.
```

##(b) Regresión lineal con las variables seleccionadas paso a paso por AIC. Nota: Para no mostrar todos los pasos, utilizar el parámetro trace = F.

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
model_step <- stepAIC(lm(total_UPDRS ~ motor_UPDRS + `Jitter(Abs)` +
`Jitter:RAP` +
`Jitter:PPQ5` + `Jitter:DDP` + Shimmer + `Shimmer(dB)` +
`Shimmer:APQ3` + `Shimmer:APQ5` + `Shimmer:APQ11` +
`Shimmer:DDA` + NHR + HNR + RPDE + DFA + PPE,
data = data.train), direction = "both", trace = FALSE)

coefficients_step <- coef(model_step)
adj_r_squared_step <- summary(model_step)$adj.r.squared

train_predictions_step <- predict(model_step, newdata = data.train)
train_rmse_step <- sqrt(mean((data.train$total_UPDRS -
train_predictions_step)^2))

test_predictions_step <- predict(model_step, newdata = data.test)
test_rmse_step <- sqrt(mean((data.test$total_UPDRS -
test_predictions_step)^2))
```

```r
cat("Número de variables predictoras:", length(coefficients_step) - 1,
"\n")
```

```
## Número de variables predictoras: 11
```

```r
cat("R^2 ajustado:", adj_r_squared_step, "\n")
```

```
## R^2 ajustado: 0.9042567
```

```r
cat("RMSE para el grupo de ajuste:", train_rmse_step, "\n")
```

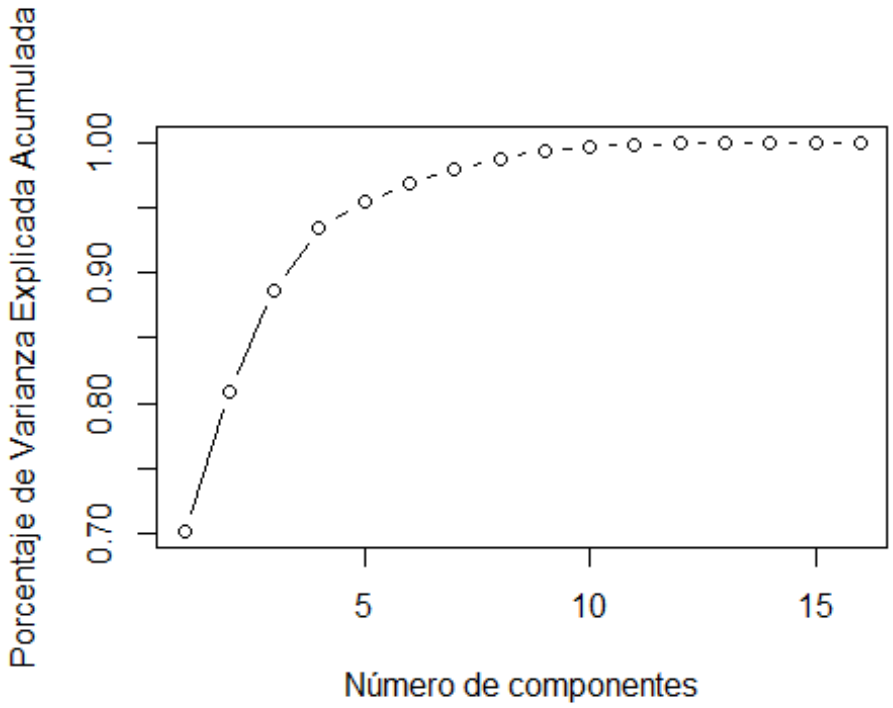
```
## RMSE para el grupo de ajuste: 3.32393
```

```r
cat("RMSE para el grupo de prueba:", test_rmse_step, "\n")
```

```
## RMSE para el grupo de prueba: 3.468669
```

(c) Regresión por componentes principales. Nota: Utilizar el mínimo de componentes razonable a la vista del gráfico de RMSE.

```r
# Realizar el análisis de componentes principales (PCA)
pca <- prcomp(data.train[, 7:22], scale. = TRUE)

# Obtener el porcentaje de varianza explicada acumulada
variance_explained <- cumsum(pca$sdev^2) / sum(pca$sdev^2)

# Graficar el porcentaje de varianza explicada acumulada
plot(variance_explained, type = "b", xlab = "Número de componentes", ylab
= "Porcentaje de Varianza Explicada Acumulada")
```



```r
# El número mínimo de componentes razonable a partir del gráfico es de 5-7
library(pls)
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```r
num_components <- 5
model_pca <- plsr(total_UPDRS ~ ., data = data.train[, 6:22], scale = TRUE, ncomp = num_components)


# Test-MSE
predicciones <- predict(model_pca, newdata = data.test, ncomp = 4)
test_mse <- mean((predicciones - data.test$total_UPDRS)^2)
test_mse
```

```
## [1] 99.54143
```

```r
# Calcular el R^2 ajustado del modelo de regresión lineal utilizando las componentes principales seleccionadas
adj_r_squared_pca <- summary(model_pca)$adj.r.squared
```

```
## Data:    X dimension: 4700 16
##  Y dimension: 4700 1
## Fit method: kernelpls
## Number of components considered: 5
## TRAINING: % variance explained
##              1 comps  2 comps  3 comps  4 comps  5 comps
## X             68.421   75.681   83.957   87.590   94.276
## total_UPDRS    2.007    7.666    8.757    9.769    9.922
```

```r
#Standardize the training dataset
scaled_data.train <- scale(data.train[, 5:22])

#Perform principal component analysis
pca <- prcomp(scaled_data.train)

#Calculate cumulative variance explained by principal components
cumulative_var <- cumsum(pca$sdev^2) / sum(pca$sdev^2)

model_pca <- plsr(total_UPDRS ~ ., data = data.train[, 6:22], scale = TRUE, ncomp = num_components)


#use model to make predictions on a test set
y_test <- data.test$motor_UPDRS
```

```
model <- pcr(motor_UPDRS ~ total_UPDRS + `Jitter(%)` + `Jitter(Abs)` +
`Jitter:RAP` + `Jitter:PPQ5` + `Jitter:DDP` + `Shimmer` + `Shimmer(dB)` +
`Shimmer:APQ3` + `Shimmer:APQ5` + `Shimmer:APQ11` + `Shimmer:DDA` + `NHR`
+ `HNR` + `RPDE` + `DFA` + `PPE`, data=data.train, scale=TRUE,
validation="CV")
pcr_pred <- predict(model, data.test, ncomp=5)

#adj_r_squared_pca <- summary(model)$adj.r.squared no funciona

#calculate RMSE
rmse_test <- sqrt(mean((pcr_pred - y_test)^2))
cat("RMSE on test dataset:", rmse_test, "\n")
```

```
## RMSE on test dataset: 2.667638
```

```
cat("Number of components used:", num_components, "\n")
```

```
## Number of components used: 5
```

```
cat("R^2:", adj_r_squared_pca, "\n")
```

```
## R^2:
```

```
#Según estos resultados podemos decir que el error mínimo se obtiene con
8 componentes principales.
```

(d) Ridge regression

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-7
```

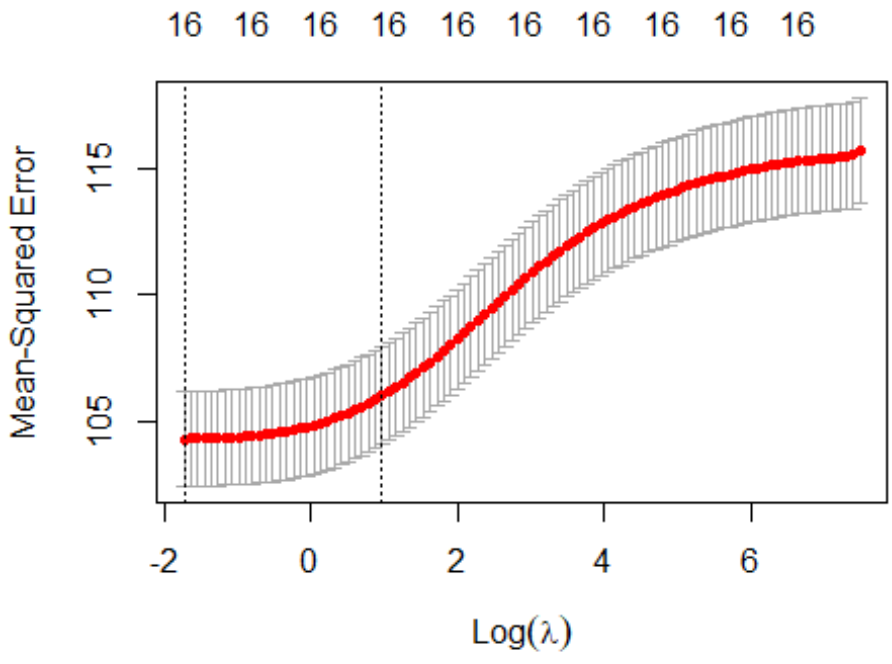
```
x_train <- as.matrix(data.train[, c("Jitter(%)", "Jitter(Abs)",
"Jitter:RAP", "Jitter:PPQ5",
                                    "Jitter:DDP", "Shimmer",
"Shimmer(dB)", "Shimmer:APQ3",
                                    "Shimmer:APQ5", "Shimmer:APQ11",
"Shimmer:DDA", "NHR",
                                    "HNR", "RPDE", "DFA", "PPE")])
y_train <- data.train$total_UPDRS

ridge_model <- cv.glmnet(x_train, y_train, alpha = 0, nfolds = 10)
plot(ridge_model, xvar = "lambda", label = TRUE)
```

```
## Warning in plot.window(...): "xvar" is not a graphical parameter
```

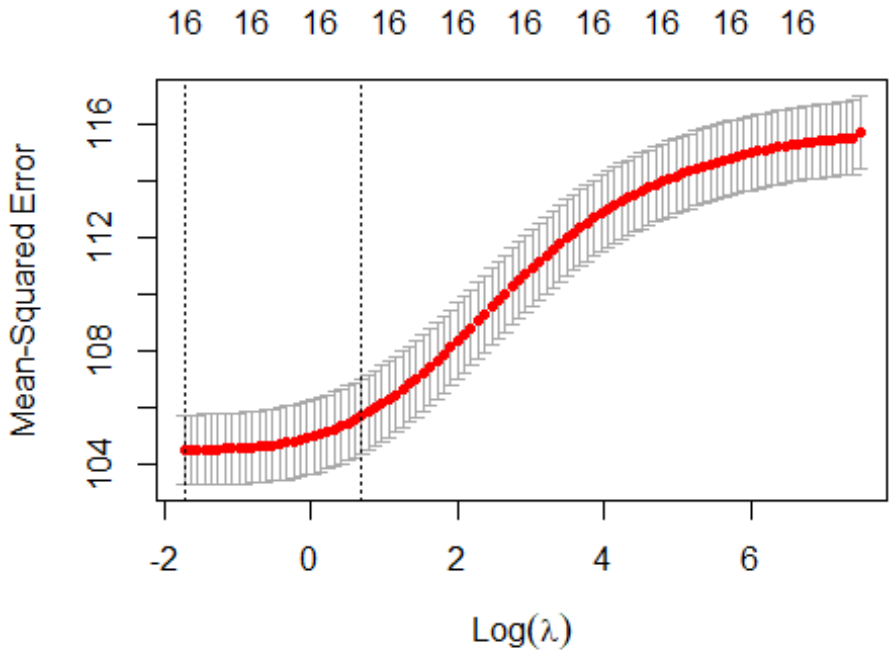
```
## Warning in plot.window(...): "label" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "xvar" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "label" is not a graphical
parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "xvar" is
not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "label"
is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "xvar" is
not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "label"
is not a
## graphical parameter

## Warning in box(...): "xvar" is not a graphical parameter

## Warning in box(...): "label" is not a graphical parameter

## Warning in title(...): "xvar" is not a graphical parameter

## Warning in title(...): "label" is not a graphical parameter
```



```r
cv_error_ridge <- cv.glmnet(x = x_train, y = y_train, alpha = 0, nfolds =
10,type.measure = "mse")
plot(cv_error_ridge)
```




```r
best_lambda <- ridge_model$lambda.min #menor error
best_lambda
```

```
## [1] 0.1774049
```

```r
optimo_lambda <- ridge_model$lambda.1se #valor óptimo
optimo_lambda
```

```
## [1] 2.634406
```

```r
final_model <- glmnet(x_train, y_train, alpha = 0, lambda =
optimo_lambda)
coef(final_model)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                              s0
## (Intercept)    4.472208e+01
## Jitter(%)      1.888409e+01
## Jitter(Abs)   -1.093545e+04
## Jitter:RAP     9.500189e+00
## Jitter:PPQ5   -3.669523e+01
## Jitter:DDP     3.960141e+00
## Shimmer       -2.553678e+00
## Shimmer(dB)    1.894329e-01
```

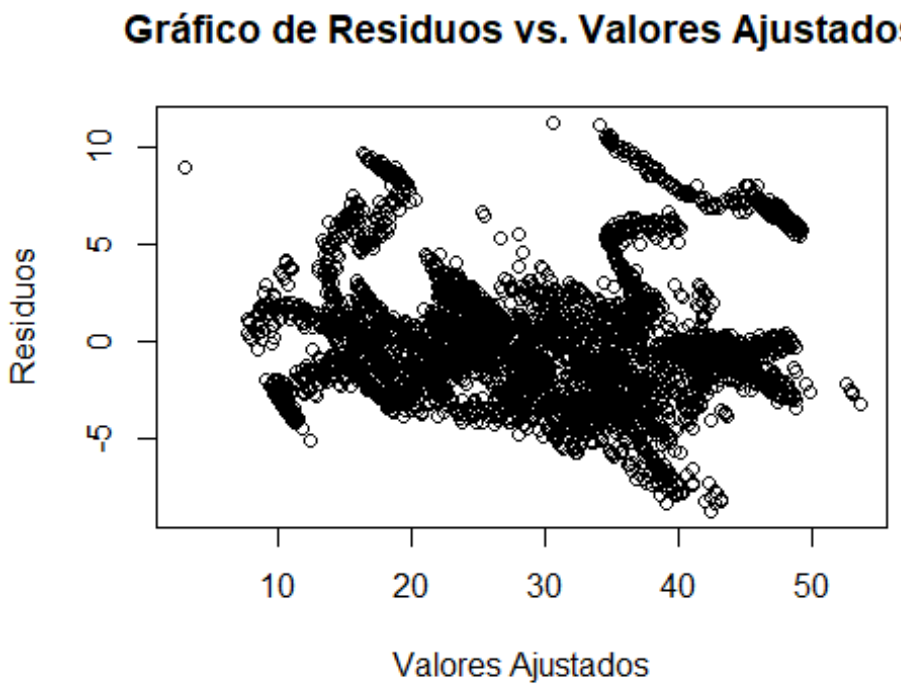
```
## Shimmer:APQ3  -2.339831e+01
## Shimmer:APQ5  -1.601147e+01
## Shimmer:APQ11  3.855922e+01
## Shimmer:DDA   -8.185134e+00
## NHR           -1.360275e+01
## HNR           -2.996927e-01
## RPDE           8.110887e+00
## DFA           -2.448521e+01
## PPE            1.539364e+01

#Predictions
x_test <- as.matrix(data.test[, c("Jitter(%)", "Jitter(Abs)",
"Jitter:RAP", "Jitter:PPQ5",
                                  "Jitter:DDP", "Shimmer", "Shimmer(dB)",
"Shimmer:APQ3",
                                  "Shimmer:APQ5", "Shimmer:APQ11",
"Shimmer:DDA", "NHR",
                                  "HNR", "RPDE", "DFA", "PPE")])
y_test <- data.test$total_UPDRS
predictions <- predict(final_model, newx = x_test)

rmse_test <- sqrt(mean((y_test - predictions)^2))
rmse_train <- sqrt(mean((y_train - predict(final_model, newx =
x_train))^2))

num_variables <- sum(final_model$beta != 0)
r_squared <- final_model$dev.ratio[length(final_model$dev.ratio)]

cat("Number of variables used:", num_variables, "\n")

## Number of variables used: 16

cat("R-squared coefficient:", r_squared, "\n")

## R-squared coefficient: 0.08496117

cat("RMSE for the test group:", rmse_test, "\n")

## RMSE for the test group: 10.05166

cat("RMSE for the train group:", rmse_train, "\n")

## RMSE for the train group: 10.28789
```

(e) ¿Cree necesario repetir estos métodos tomando la variable motor_UPDRS como respuesta?

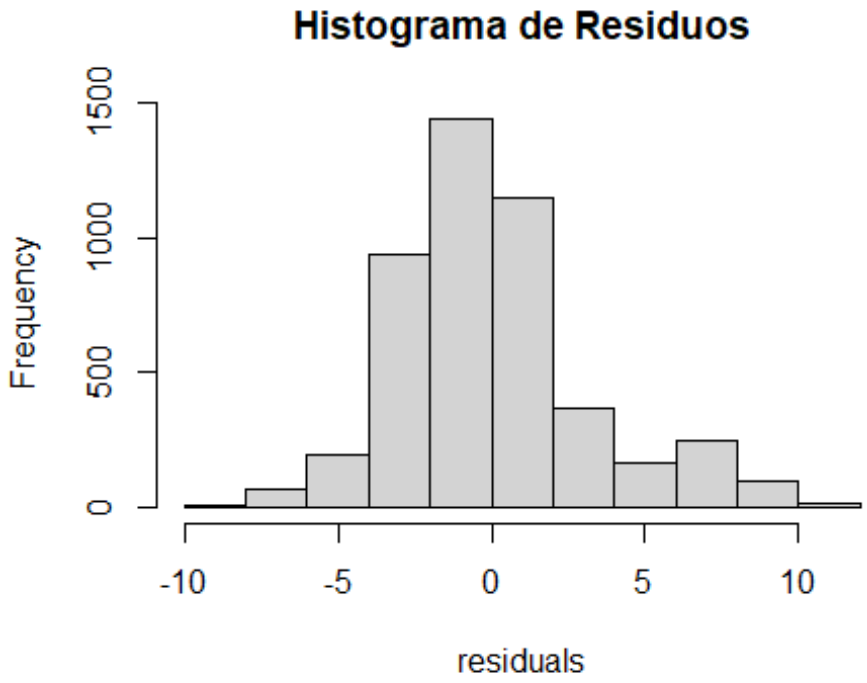
Depende del contexto del estudio. Si lo que se quiere predecir es específicamente la variable motor_UPDRS, entonces debería de tomarse como respuesta.

(f) Con la base de datos data.train y el modelo OLS hacer un rápido análisis de los residuos para detectar incumplimientos de las condiciones de un modelo lineal. Estudiar especialmente si hay problemas de multicolinealidad.

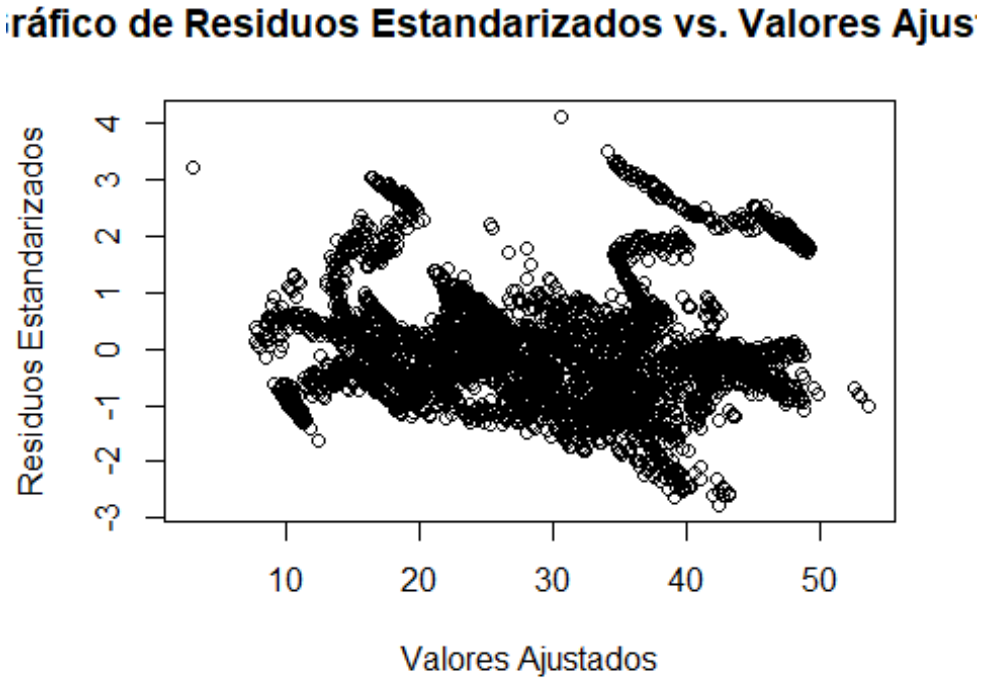
```
model <- lm(total_UPDRS ~ ., data = data.train)
residuals <- resid(model)
plot(predict(model), residuals, xlab = "Valores Ajustados", ylab =
"Residuos", main = "Gráfico de Residuos vs. Valores Ajustados")
```



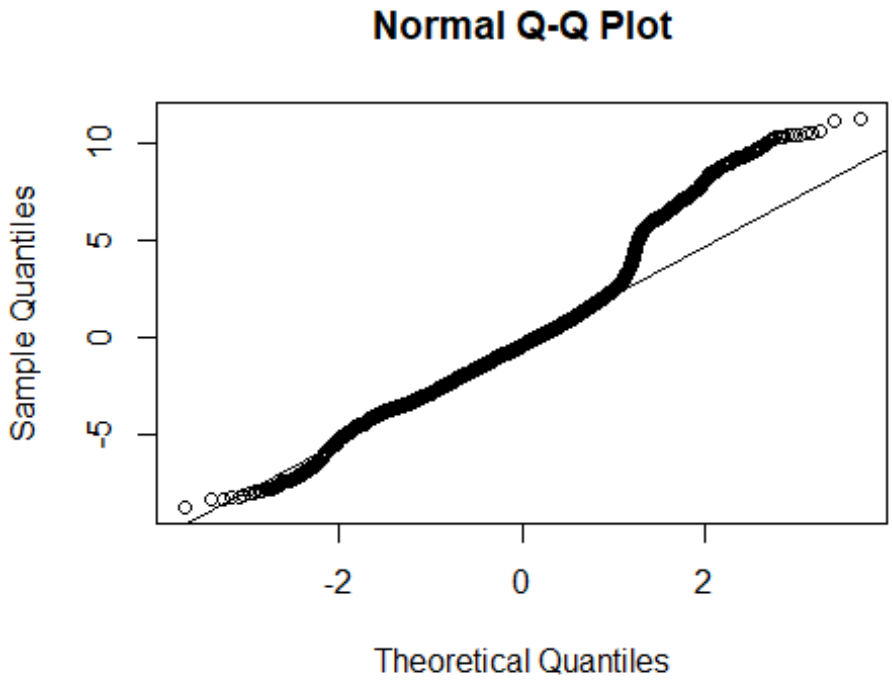

```
hist(residuals, main = "Histograma de Residuos")
```




```
plot(predict(model), rstandard(model), xlab = "Valores Ajustados", ylab =
"Residuos Estandarizados", main = "Gráfico de Residuos Estandarizados vs.
Valores Ajustados")
```



```r
qqnorm(residuals)
qqline(residuals)
```




```r
# Prueba de normalidad de los residuos
shapiro.test(residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals
## W = 0.94928, p-value < 2.2e-16
```

```r
#Valor p mucho menor de 0.05, lo que supone no normalidad de los
residuos.
# Prueba de homocedasticidad (varianza constante de los residuos)
library(lmtest)
```

```
## Loading required package: zoo
```

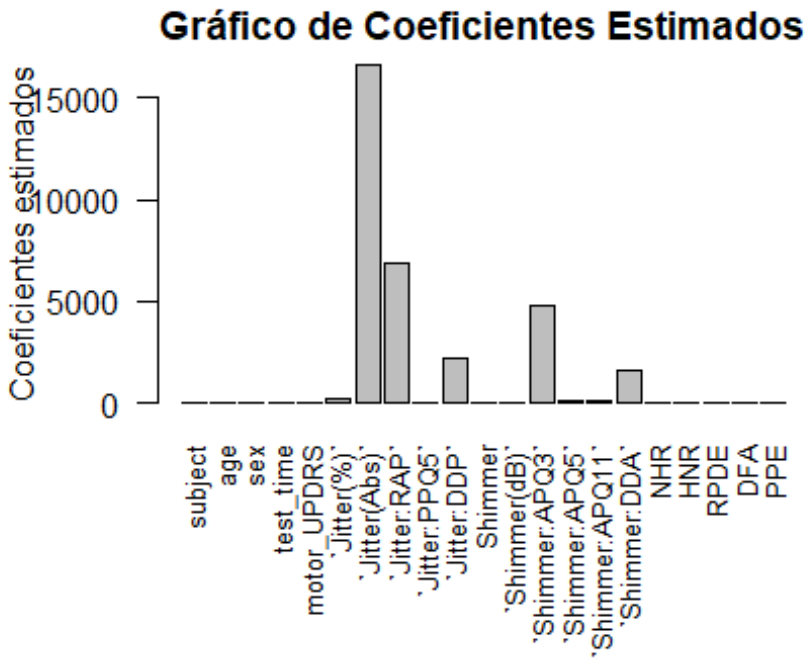
```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
bptest(model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model
## BP = 467.94, df = 21, p-value < 2.2e-16

#Valor mucho menor de 0.05 por lo que suponemos que no hay
homocedasticidad.
# Calcula el VIF para cada variable predictora
library(car)

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

vif_values <- vif(model)
vif_values

##         subject             age             sex       test_time
motor_UPDRS
##    1.315203e+00    1.164634e+00    1.479957e+00    1.020464e+00
1.280816e+00
##     `Jitter(%)`    `Jitter(Abs)`     `Jitter:RAP`    `Jitter:PPQ5`
`Jitter:DDP`
##    9.335471e+01    7.834522e+00    1.334783e+06    3.310913e+01
1.334974e+06
##         Shimmer    `Shimmer(dB)`   `Shimmer:APQ3`   `Shimmer:APQ5`
`Shimmer:APQ11`
##    1.681839e+02    7.982619e+01    2.360292e+07    5.733279e+01
1.786401e+01
##    `Shimmer:DDA`             NHR             HNR            RPDE
DFA
##    2.360207e+07    8.871374e+00    5.465189e+00    2.123608e+00
1.725192e+00
##             PPE
##    4.457099e+00

barplot(abs(model$coefficients[-1]), names.arg =
names(model$coefficients[-1]), xlab = "", ylab = "Coeficientes
estimados", main = "Gráfico de Coeficientes Estimados", las = 2,
cex.names = 0.8)
 # Ajustar los márgenes para crear más espacio
 par(mar = c(9, 7, 2, 1) + 0.1)

 # Graficar el gráfico de barras con los nombres girados en el eje X
 barplot(abs(model$coefficients[-1]), names.arg =
```

```r
names(model$coefficients[-1]), xlab = "", ylab = "Coeficientes
estimados", main = "Gráfico de Coeficientes Estimados", las = 2,
cex.names = 0.8)
```




```r
#La mayoría de las variables tienen valores de VIF por debajo de 10, lo
cual generalmente se considera aceptable. Esto indica que no existe un
problema grave de multicolinealidad entre estas variables.

#Sin embargo, hay algunas variables con valores de VIF extremadamente
altos, como Jitter:RAP, Jitter:PPQ5, Shimmer:APQ3, Shimmer:APQ11 y
Shimmer:DDA. Estos valores de VIF altos sugieren que estas variables
pueden tener un alto grado de multicolinealidad con otras variables en el
modelo.

#Las variables con valores de VIF cercanos o superiores a 10 indican
posibles problemas de multicolinealidad. En este caso, Jitter(Abs),
Shimmer(dB) y PPE entran en esta categoría.
```

Ejercicio 3

Seguimos con la base de datos del ejercicio anterior. En ese ejercicio hemos visto que el modelo OLS presenta algunas dificultades. Para superarlas podemos estudiar otros métodos.

(a) Eliminar de la base de datos los 3 puntos más influyentes y volver a calcular los modelos del ejercicio 2. Mostrar en una tabla los RMSE del grupo de prueba (test) para cada uno de los modelos con y sin los puntos influyentes. Nota: En este apartado sólo hay que mostrar la tabla.

```
# Calculate Cook's distance
cooksd <- cooks.distance(model)

# Identify influential points
influential_indices <- which(cooksd > 4/nrow(data.train))  # Adjust the
threshold as needed

# Remove influential points from the dataset
data.train_filtered <- data.train[-influential_indices, ]
data.test_filtered <- data.test[-influential_indices, ]

# Scale the filtered training and test data
scaled_data.train_filtered <- scale(data.train_filtered[, -1])
scaled_data.test_filtered <- scale(data.test_filtered[, -1])

num_components_filtered <- 5
x_train_filtered <- as.matrix(data.train_filtered[, c("Jitter(%)",
"Jitter(Abs)", "Jitter:RAP", "Jitter:PPQ5",
                                    "Jitter:DDP", "Shimmer",
"Shimmer(dB)", "Shimmer:APQ3",
                                    "Shimmer:APQ5", "Shimmer:APQ11",
"Shimmer:DDA", "NHR",
                                    "HNR", "RPDE", "DFA", "PPE")])
y_train_filtered <- data.train_filtered$total_UPDRS


# Recalculate the models without influential points
model_filtered <- lm(total_UPDRS ~ ., data = data.train_filtered)
aic_model_filtered <- stepAIC(model, direction = "both", trace = FALSE)
pca_train_filtered <- predict(pca, newdata =
scaled_data.train_filtered)[, 1:num_components_filtered]
plsr_model_filtered <- plsr(total_UPDRS ~ ., data = data.train_filtered,
ncomp = 5)
ridge_model_filtered <- glmnet(x_train_filtered, y_train_filtered, alpha
= 0, lambda = best_lambda)

# Calculate RMSE for each model with and without influential points
```

```r
rmse_ols_train <- sqrt(mean((y_train - predict(model, newdata =
data.train))^2))
rmse_aic_train <- sqrt(mean((y_train_filtered -
predict(aic_model_filtered, newdata = data.train_filtered))^2))
rmse_pca_train <- sqrt(mean((data.train_filtered$total_UPDRS -
predict(plsr_model_filtered, newdata = data.train_filtered))^2))
rmse_ridge_train <- sqrt(mean((y_train - predict(ridge_model, newx =
x_train))^2))

# Calculate RMSE for each model without influential points (test group)
rmse_ols_without_influential <- sqrt(mean((y_test[-influential_indices] -
predict(model, newdata = data.test[-influential_indices, ]))^2))
rmse_aic_without_influential <- sqrt(mean((y_test[-influential_indices] -
predict(aic_model_filtered, newdata = data.test[-influential_indices,
]))^2))
rmse_pca_without_influential <- sqrt(mean((data.test$total_UPDRS[-
influential_indices] - predict(plsr_model_filtered, newdata = data.test[-
influential_indices, ]))^2))
#rmse_ridge_without_influential <- sqrt(mean((y_test[-
influential_indices] - predict(ridge_model, newx = x_test[-
influential_indices, ]))^2))
rmse_ridge_without_influential <- sqrt(mean((y_test -
predict(ridge_model_filtered, newx = x_test))^2))

# Create a table of RMSE results
results_table <- tibble(
  Model = c("OLS", "OLS (without influential)", "AIC", "AIC (without
influential)",
            "PCA", "PCA (without influential)", "Ridge", "Ridge (without
influential)"),
  RMSE_Test = c(rmse_ols_train, rmse_ols_without_influential,
rmse_aic_train,
                rmse_aic_without_influential, rmse_pca_train,
rmse_pca_without_influential,
                rmse_ridge_train, rmse_ridge_without_influential)
)

results_table
```

```
## # A tibble: 8 × 2
##   Model                          RMSE_Test
##   <chr>                              <dbl>
## 1 OLS                                 3.19
## 2 OLS (without influential)           3.30
## 3 AIC                                 2.87
## 4 AIC (without influential)           3.29
## 5 PCA                                 5.59
## 6 PCA (without influential)           5.72
## 7 Ridge                              10.3
## 8 Ridge (without influential)        10.1
```

(b) Dado que el grupo de prueba (test) puede contener outliers, calcular un RMSE robusto para cadamodelo utilizando la media recortada (trimmed) al 10 %. Añadir esta información a la tabla del apartado anterior..

```
library(robustbase)
```

```
## Warning: package 'robustbase' was built under R version 4.3.1
```

```
# Calculate trimmed mean and RMSE for each model
trimmed_mean <- function(x) mean_trimmed(x, trim = 0.1)
rmse_robust <- function(actual, predicted) sqrt(mean((actual -
predicted)^2))

rmse_ols_robust <- rmse_robust(y_test, predict(model, newdata =
data.test))
rmse_aic_robust <- rmse_robust(y_test, predict(aic_model_filtered,
newdata = data.test))
rmse_pca_robust <- rmse_robust(data.test$total_UPDRS,
predict(plsr_model_filtered, newdata = data.test))
rmse_ridge_robust <- rmse_robust(y_test, predict(ridge_model_filtered,
newx = x_test))

# Add robust RMSE values to the table
results_table$RMSE_Robust <- c(
  rmse_ols_robust,
  rmse_ols_without_influential,
  rmse_aic_robust,
  rmse_aic_without_influential,
  rmse_pca_robust,
  rmse_pca_without_influential,
  rmse_ridge_robust,
  rmse_ridge_without_influential
)

results_table
```

```
## # A tibble: 8 × 3
##   Model                         RMSE_Test RMSE_Robust
##   <chr>                             <dbl>       <dbl>
## 1 OLS                                3.19        3.30
## 2 OLS (without influential)          3.30        3.30
## 3 AIC                                2.87        3.29
## 4 AIC (without influential)          3.29        3.29
## 5 PCA                                5.59        5.73
## 6 PCA (without influential)          5.72        5.72
## 7 Ridge                             10.3        10.1
## 8 Ridge (without influential)       10.1        10.1
```

(c) Dados los problemas observados con los residuos del modelo OLS, podemos probar un método robusto como el de Huber o el Least trimmed squares (LTS).

```
library(dplyr)
# Renombrar columnas problemáticas
data.train <- data.train %>%
rename(Jitter_Abs = `Jitter(Abs)`,
Jitter_RAP = `Jitter:RAP`,
Jitter_PPQ5 = `Jitter:PPQ5`,
Jitter_DDP = `Jitter:DDP`,
Shimmer_dB = `Shimmer(dB)`,
Shimmer_APQ3 = `Shimmer:APQ3`,
Shimmer_APQ5 = `Shimmer:APQ5`,
Shimmer_APQ11 = `Shimmer:APQ11`,
Shimmer_DDA = `Shimmer:DDA`,
Jitter_p = `Jitter(%)`)
# Ajustar el modelo con los nombres de columnas actualizados
model_huber <- rlm(total_UPDRS ~ motor_UPDRS + Jitter_Abs + Jitter_RAP +
Jitter_PPQ5 + Jitter_DDP + Shimmer + Shimmer_dB +
Shimmer_APQ3 + Shimmer_APQ5 + Shimmer_APQ11 +
Shimmer_DDA + NHR + HNR + RPDE + DFA + PPE,
data = data.train)

summary(model_huber)

##
## Call: rlm(formula = total_UPDRS ~ motor_UPDRS + Jitter_Abs +
Jitter_RAP +
##     Jitter_PPQ5 + Jitter_DDP + Shimmer + Shimmer_dB + Shimmer_APQ3 +
##     Shimmer_APQ5 + Shimmer_APQ11 + Shimmer_DDA + NHR + HNR +
##     RPDE + DFA + PPE, data = data.train)
## Residuals:
##      Min       1Q   Median       3Q      Max
## -9.20441 -1.79147  0.02159  1.67197 17.59132
##
## Coefficients:
##               Value      Std. Error t value
## (Intercept)       4.5984     0.9759     4.7118
## motor_UPDRS       1.2284     0.0054   228.1480
## Jitter_Abs    36010.4860  2875.5286    12.5231
## Jitter_RAP    -6744.7440 15251.9680    -0.4422
## Jitter_PPQ5     -41.6151    52.5031    -0.7926
## Jitter_DDP     2173.4071  5084.6930     0.4274
## Shimmer         -63.2040    20.8236    -3.0352
## Shimmer_dB        2.1365     1.6024     1.3333
## Shimmer_APQ3  -1545.7159 15245.1521    -0.1014
## Shimmer_APQ5     75.0780    18.7441     4.0054
## Shimmer_APQ11   -35.6169     8.8131    -4.0414
## Shimmer_DDA     527.0734  5081.6329     0.1037
```

```
## NHR              -5.8440     1.9868    -2.9415
## HNR              -0.0436     0.0222    -1.9633
## RPDE              4.4603     0.5862     7.6096
## DFA              -4.4174     0.7510    -5.8823
## PPE              -4.1212     0.9196    -4.4815
##
## Residual standard error: 2.556 on 4683 degrees of freedom
```

```
#Model_LTS
model_lts <- ltsreg(total_UPDRS ~ motor_UPDRS + Jitter_Abs + Jitter_RAP +
Jitter_PPQ5 + Jitter_DDP + Shimmer + Shimmer_dB +
Shimmer_APQ3 + Shimmer_APQ5 + Shimmer_APQ11 +
Shimmer_DDA + NHR + HNR + RPDE + DFA + PPE,
data = data.train)
model_lts
```

```
## Call:
## lqs.formula(formula = total_UPDRS ~ motor_UPDRS + Jitter_Abs +
##     Jitter_RAP + Jitter_PPQ5 + Jitter_DDP + Shimmer + Shimmer_dB +
##     Shimmer_APQ3 + Shimmer_APQ5 + Shimmer_APQ11 + Shimmer_DDA +
##     NHR + HNR + RPDE + DFA + PPE, data = data.train, method = "lts")
##
## Coefficients:
##   (Intercept)    motor_UPDRS     Jitter_Abs     Jitter_RAP
Jitter_PPQ5
##     1.119e+01      1.164e+00      3.831e+04     -7.320e+04
8.003e+01
##    Jitter_DDP        Shimmer     Shimmer_dB   Shimmer_APQ3
Shimmer_APQ5
##     2.413e+04      2.004e+02      4.925e+00      1.247e+05     -
7.715e+02
## Shimmer_APQ11    Shimmer_DDA            NHR            HNR
RPDE
##     3.113e+01     -4.143e+04      5.948e+01     -1.152e-02
3.386e+00
##           DFA            PPE
##    -1.352e+01     -7.616e+00
##
## Scale estimates 2.937 2.750
```

```
summary(model_lts)
```

```
##               Length Class      Mode
## crit             1   -none-     numeric
## sing             1   -none-     character
## coefficients    17   -none-     numeric
## bestone         17   -none-     numeric
## fitted.values 4700   -none-     numeric
## residuals     4700   -none-     numeric
## scale            2   -none-     numeric
## terms            3   terms      call
```

```
## call             4   -none-     call
## xlevels          0   -none-     list
## model           17   data.frame list
```