PAC 2 Regresión Lineal

Maria Lucas

2023-06-22

Table of Contents

# Ejercicio 1

# Set viasulization number  
options(scipen = 999)  
  
# Set the file path  
file\_path <- "D:/Antiguos estudios/MASTER2/Sem2/Regresion/PAC2/P2/pancreas\_biomarkers.txt"  
  
# Load the data into a data frame  
data <- read.table(file\_path, header = TRUE, sep = "\t")  
  
# Display the first few rows of the data frame  
head(data)

## sample\_id sample\_origin age age\_cat sex diagnosis stage  
## 1 S1 BPTB 33 26-35 F 1   
## 2 S10 BPTB 81 75+ F 1   
## 3 S100 BPTB 51 46-55 M 1   
## 4 S101 BPTB 61 56-65 M 1   
## 5 S102 BPTB 62 56-65 M 1   
## 6 S103 BPTB 53 46-55 M 1   
## benign\_sample\_diagnosis creatinine LYVE1 REG1B TFF1  
## 1 1.83222 0.89321920 52.94884 654.2822  
## 2 0.97266 2.03758500 94.46703 209.4882  
## 3 0.78039 0.14558890 102.36600 461.1410  
## 4 0.70122 0.00280488 60.57900 142.9500  
## 5 0.21489 0.00085956 65.54000 41.0880  
## 6 0.84825 0.00339300 62.12600 59.7930

# Saving variables as factors  
data$diagnosis <- factor(data$diagnosis, levels = 1:3)  
data$age\_cat <- factor(data$age\_cat)

### (a) Modelo de regresión logística

#### Diagnóstico de todos los casos

model <- glm(diagnosis ~ age\_cat + creatinine + LYVE1 + REG1B + TFF1, data = data, family = binomial)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

summary(model)

##   
## Call:  
## glm(formula = diagnosis ~ age\_cat + creatinine + LYVE1 + REG1B +   
## TFF1, family = binomial, data = data)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.27174931 0.47650815 0.570 0.568479   
## age\_cat36-45 0.27524151 0.52389015 0.525 0.599319   
## age\_cat46-55 -0.17244569 0.49309008 -0.350 0.726545   
## age\_cat56-65 -0.57047219 0.49120108 -1.161 0.245487   
## age\_cat66-75 0.34683938 0.51381181 0.675 0.499655   
## age\_cat75+ 0.00001974 0.59791940 0.000 0.999974   
## creatinine -0.84736599 0.21311812 -3.976 0.000070073 \*\*\*  
## LYVE1 0.21942153 0.06138360 3.575 0.000351 \*\*\*  
## REG1B 0.00069921 0.00143069 0.489 0.625041   
## TFF1 0.00251348 0.00050567 4.971 0.000000668 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 730.70 on 589 degrees of freedom  
## Residual deviance: 567.16 on 580 degrees of freedom  
## AIC: 587.16  
##   
## Number of Fisher Scoring iterations: 7

Tal y como podemos ver en los resultados, tan solo la creatinina, la LYVE1 y el TFF1 son variables que permiten predecir el riesgo de adenocarcinoma ductal pancreático. Se puede observar porque en todas ellas el pvalor es menos a 0.01. En el contexto de regresión logística un pvalor de 0.01 permite rechazar la hipótesis nula de no relación entre la variable predictora y la variable respuesta. En otras palabras, la variable tiene un impacto significativo sobre la clase.

#### Sólo adenocarcinoma y otro

# Subset the data to include only levels 2 and 3 of the "diagnosis" variable  
subset\_data <- subset(data, diagnosis != 1)  
  
# Recode values 2 and 3 to 0 and 1, respectively  
subset\_data$diagnosis <- ifelse(subset\_data$diagnosis == 2, 0, 1)  
  
# Fit the logistic regression model using the subsetted data  
model <- glm(diagnosis ~ age\_cat + creatinine + LYVE1 + REG1B + TFF1, data = subset\_data, family = binomial)  
  
# View the model summary  
summary(model)

##   
## Call:  
## glm(formula = diagnosis ~ age\_cat + creatinine + LYVE1 + REG1B +   
## TFF1, family = binomial, data = subset\_data)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.66048523 1.18806983 -3.081 0.00206 \*\*   
## age\_cat36-45 1.15688820 1.22969696 0.941 0.34681   
## age\_cat46-55 1.66227729 1.18000260 1.409 0.15892   
## age\_cat56-65 3.03244805 1.16788522 2.597 0.00942 \*\*   
## age\_cat66-75 2.70033855 1.16861309 2.311 0.02085 \*   
## age\_cat75+ 3.44294729 1.21523972 2.833 0.00461 \*\*   
## creatinine -0.30213039 0.22625232 -1.335 0.18176   
## LYVE1 0.31400141 0.05280717 5.946 0.00000000274 \*\*\*  
## REG1B 0.00280367 0.00109942 2.550 0.01077 \*   
## TFF1 -0.00005978 0.00021201 -0.282 0.77798   
## ---  
## 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

Tal y como podemos ver en los resultados, la edad a partir de 56 años es un indicador significativo. La LYVE1 y el REG1B también son variables que permiten predecir el riesgo de adenocarcinoma ductal pancreático. Se puede observar porque en todas ellas el pvalor es menos a 0.05. En el contexto de regresión logística un pvalor de 0.01 permite rechazar la hipótesis nula de no relación entre la variable predictora y la variable respuesta. En otras palabras, la variable tiene un impacto significativo sobre la clase.

Que el intercepto también sea significativo sugiere que parte de la variable respuesta no es explicada por las variables independientes estudiadas. En regresión logística el intercepto captura la probabilidad de ocurrencia de un suceso cuando todas las variables predictoras están en su nivel de referencia. En otras palabras, un intercepto significativo implica que aunque no tengamos ningun predictor, hay diferencias significativas entre las clases.

### (b) Interpretación de coeficientes

Los coeficientes en un modelo de regresión logística indican el cambio estimado en el log-odds del evento ocurriendo (codeado como 1 = adenocarcinoma ductal pancreático) asociado a una unidad de cambio en la variable predictora, sin variar el resto de variables.

Un estimador positivo sugiere que el incremento de la variable está asociado a un incremento de la probabilidad (log-odds) del evento ocurriendo. Si el estimador es 0.5 significa que al aumentar en 1 el valor del predictor, esto se asopcia a un 0.5 aumento del log-odds del evento ocurriendo. Este es el caso del LYVE1, al aumentar LYVE1 es más probable tener adenocarcinoma (3.140e-01 = 0.3140). Lo mismo sucede con la edad, parece ser que tener más de 56 años aumenta el log-odds de tener adenocacinoma. Lo hace de forma distinta dependiendo del rango de edad, de 56-65 (3.032e+00 = 3.032), de 66-75 (2.700e+00 = 2.700) y a partir de 75 años (3.443e+00 = 3.443).

Contrariamente, un estimador negativo significa que un aumento de la variable disminuye el log-odds del evento ocurriendo (tener adenocarcinoma).

### (c) Modelo reducido

Para comparar ambos modelos podemos realizar un anova aplicando el test Chi cuadrado. Con esto estamos comparando el ajuste del modelo bajo las siguientes hipótesis:

* **H0:** Desviación del modelo reducido = Desviación del modelo completo. No hay diferencia de ajuste entre los modelos.
* **H1:** Desviación del modelo reducido > Desviación del modelo completo. El modelo reducido se ajusta peor que el modelo completo.

La desviación del modelo se refiere a la diferencia entre los valores reales de los datos y los valores predichos por el modelo.

# Fit the logistic regression model using the subsetted data  
model\_simple <- glm(diagnosis ~ age\_cat + LYVE1 + REG1B, data = subset\_data, family = binomial)  
  
summary(model\_simple)

##   
## Call:  
## glm(formula = diagnosis ~ age\_cat + LYVE1 + REG1B, family = binomial,   
## data = subset\_data)  
##   
## 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 0.00000000359 \*\*\*  
## 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

# Compare the two models  
anova(model, model\_simple, test = "Chi")

## Analysis of Deviance Table  
##   
## Model 1: diagnosis ~ age\_cat + creatinine + LYVE1 + REG1B + TFF1  
## Model 2: diagnosis ~ age\_cat + LYVE1 + REG1B  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
## 1 397 382.52   
## 2 399 384.83 -2 -2.312 0.3147

# Install and load the "lmtest" package  
# install.packages("lmtest")  
library(lmtest)

## Warning: package 'lmtest' was built under R version 4.3.1

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 4.3.1

##   
## Attaching package: 'zoo'

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

# Perform the likelihood ratio test  
lrtest(model, model\_simple)

## Likelihood ratio test  
##   
## Model 1: diagnosis ~ age\_cat + creatinine + LYVE1 + REG1B + TFF1  
## Model 2: diagnosis ~ age\_cat + LYVE1 + REG1B  
## #Df LogLik Df Chisq Pr(>Chisq)  
## 1 10 -191.26   
## 2 8 -192.41 -2 2.312 0.3147

Dado que el pvalor no es significativo (0.31) aceptamos la hipótesis nula, el modelo reducido (sin creatinina y sin TFF1) es igual de bueno que el complejo. Adicionalmente, vemos que el AIC del modelo reducido es 2 unidades menor que el del modelo complejo, esto sugiere que el modelo reducido tiene un ajuste mejor teniendo en cuenta la complejidad de ambos modelos.

### (d) Funcion cuadrática

# Cuadratic LYVE1  
model\_simple\_LYVE1 = glm(diagnosis ~ age\_cat + LYVE1 + I(LYVE1^2) + REG1B, data = subset\_data, family = binomial)  
  
summary(model\_simple\_LYVE1)

##   
## Call:  
## glm(formula = diagnosis ~ age\_cat + LYVE1 + I(LYVE1^2) + REG1B,   
## family = binomial, data = subset\_data)  
##   
## 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 \*\*\*  
## I(LYVE1^2) -0.0104317 0.0090294 -1.155 0.247965   
## REG1B 0.0025524 0.0009032 2.826 0.004715 \*\*   
## ---  
## 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

# Compare the two models  
anova(model\_simple, model\_simple\_LYVE1, test = "Chi")

## Analysis of Deviance Table  
##   
## Model 1: diagnosis ~ age\_cat + LYVE1 + REG1B  
## Model 2: diagnosis ~ age\_cat + LYVE1 + I(LYVE1^2) + REG1B  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
## 1 399 384.83   
## 2 398 383.93 1 0.89906 0.343

# Cuadratic REG1B  
model\_simple\_REG1B = glm(diagnosis ~ age\_cat + LYVE1 + REG1B + I(REG1B^2), data = subset\_data, family = binomial)  
  
summary(model\_simple\_REG1B)

##   
## Call:  
## glm(formula = diagnosis ~ age\_cat + LYVE1 + REG1B + I(REG1B^2),   
## family = binomial, data = subset\_data)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.956734568 1.152437904 -3.433 0.000596 \*\*\*  
## age\_cat36-45 1.139264885 1.231123197 0.925 0.354765   
## age\_cat46-55 1.740771095 1.169448508 1.489 0.136608   
## age\_cat56-65 3.084592019 1.165415260 2.647 0.008126 \*\*   
## age\_cat66-75 2.757009540 1.160754339 2.375 0.017540 \*   
## age\_cat75+ 3.536822433 1.205876020 2.933 0.003357 \*\*   
## LYVE1 0.285412846 0.050292662 5.675 0.0000000139 \*\*\*  
## REG1B 0.003777035 0.001875858 2.013 0.044062 \*   
## I(REG1B^2) -0.000001654 0.000002161 -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

# Compare the two models  
anova(model\_simple, model\_simple\_REG1B, test = "Chi")

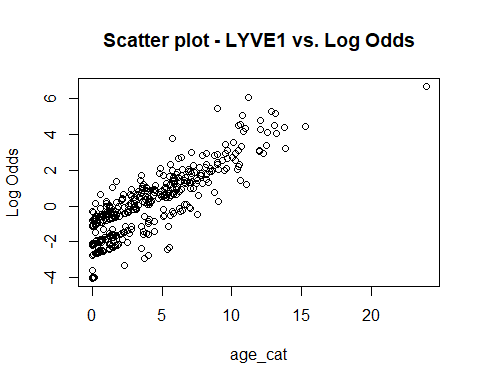
## Analysis of Deviance Table  
##   
## Model 1: diagnosis ~ age\_cat + LYVE1 + REG1B  
## Model 2: diagnosis ~ age\_cat + LYVE1 + REG1B + I(REG1B^2)  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
## 1 399 384.83   
## 2 398 384.31 1 0.52118 0.4703

En ninguno de los dos casos añadir el término cuadrático ha mejorado el ajuste del modelo. Siguiendo la explicación del apartado anterior: pv = 0.3 y 0.4, aceptamos H0 los dos modelos tienen el mismo ajuste.

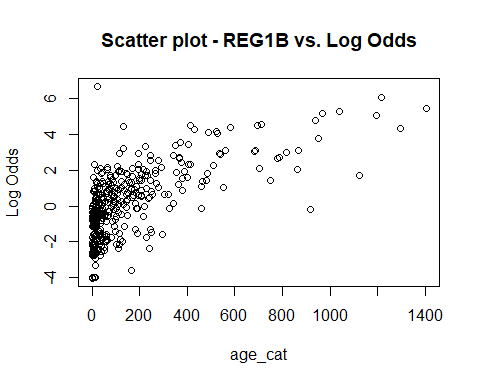
Adicionalmente, si examinamos el AIC veremos que en los modelos con el término cuadrático este es superior. Un valor mayor AIC sugiere que el modelo no se ajusta mejor para la complejidad que presenta.

En conclusión, no deberíamos incluir ninguno de los dos términos cuadráticos. Es importante comentar que, aunque estos tests aparezcan no-significativos, hay que valorarlos siempre junto con el contexto del análisis. En este caso, se ha realizado adicionalmente una inspección visual de la variable vs el log odds de la variable respuesta (disponible en el ANEXO). Estos gráficos nos confirman que efectivamente, existe una relación lineal, dado que se observa un patrón lineal sin observar otros patrones como curvas, forma de U, etc.

# Obtain predicted log odds from the model  
predicted\_logodds <- predict(model\_simple, type = "link")  
  
plot(subset\_data$LYVE1, predicted\_logodds, xlab = "age\_cat", ylab = "Log Odds", main = "Scatter plot - LYVE1 vs. Log Odds")



plot(subset\_data$REG1B, predicted\_logodds, xlab = "age\_cat", ylab = "Log Odds", main = "Scatter plot - REG1B vs. Log Odds")



### (e) Predicción de caso

# Create a new data frame for the new case  
new\_case = data.frame(age\_cat = "66-75", LYVE1 = 6, REG1B = 140)  
  
# Predict the outcome using the model  
prediction = predict(model\_simple, newdata = new\_case, type = "response")  
  
print(prediction)

## 1   
## 0.7242816

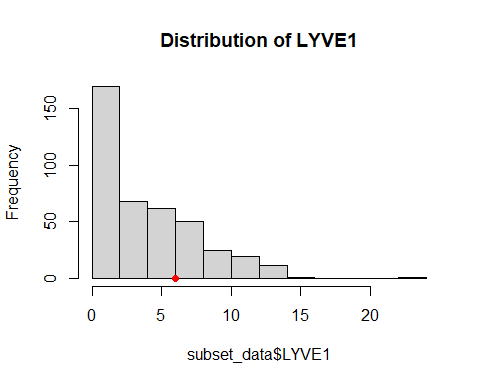
El modelo predice la presencia de adenocarcinoma ductal pancreático con un 72,42% de probabilidad. Recordemos que los valores fueron recodificados a 0 = afecciones pancreáticas no cancerosas y 1 = adenocarcinoma ductal pancreático.

La extrapolación ocurre cuando se hacen predicciones de para datos de la variable predictora fuera del rango de los datos usados para contruir el modelo.

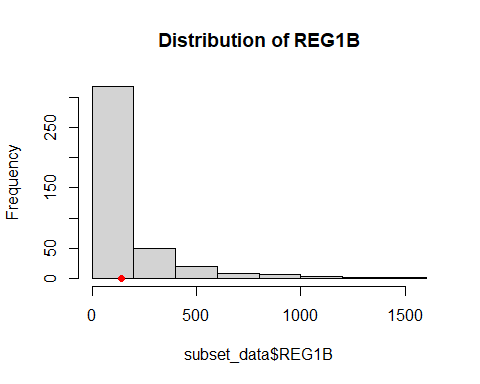
# Check predictor variable ranges  
range\_data <- sapply(subset\_data[, c("LYVE1", "REG1B")], range)  
range\_new\_observation <- c(6, 140) # Replace with the values of the new observation  
  
# Compare new observation values with observed range  
is\_extrapolation <- any(range\_new\_observation < range\_data[1, ] | range\_new\_observation > range\_data[2, ])  
  
# Print results  
cat("Extrapolation:", is\_extrapolation, "\n")

## Extrapolation: FALSE

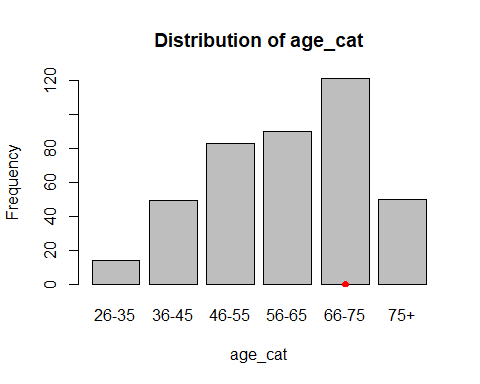
# Assess distribution of predictor variables  
# LYVE1  
hist(subset\_data$LYVE1, main = "Distribution of LYVE1")  
# Add new observation to LYVE1 plot  
points(6, 0, col = "red", pch = 16)



# REGB1  
hist(subset\_data$REG1B, main = "Distribution of REG1B")  
# Add new observation to REGB1 plot  
points(140, 0, col = "red", pch = 16)



# age\_cat  
barplot(table(subset\_data$age\_cat), main = "Distribution of age\_cat", xlab = "age\_cat", ylab = "Frequency")  
# Add new observation to age\_cat plot  
points(5.5, 0, col = "red", pch = 16)



Como podemos ver, los valores del caso a predecir se encuentra entre el rango utilizado para construir el modelo.

# Ejercicio 2

set.seed(123)  
  
# Data import  
# Note: I changed the variable names to avoid problems with symbols  
  
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)  
names(parkinson) <- c("subject#","age","sex","test\_time","motor\_UPDRS","total\_UPDRS",  
"Jitter\_p","Jitter\_Abs","Jitter\_RAP","Jitter\_PPQ5","Jitter\_DDP",  
"Shimmer","Shimmer\_dB","Shimmer\_APQ3","Shimmer\_APQ5","Shimmer\_APQ11", "Shimmer\_DDA","NHR","HNR","RPDE","DFA","PPE")  
  
# install.packages('caTools')  
library(caTools)

## Warning: package 'caTools' was built under R version 4.3.1

# Split the data into training and testing sets  
split = sample.split(parkinson, SplitRatio = 0.8)  
train\_data = parkinson[split, ]  
test\_data = parkinson[!split, ]

### (a) Regresión lineal

Al no utilizar el factor “sujeto” no tenemos en cuenta las posibles variaciones de cada individuo. Las muestras deberían ser apareadas, ya que se ace un seguimiento a lo largo del tiempo. Una estimación. No sé lo que digo ya lo mirare.

# Fit the model  
model\_lineal = lm(total\_UPDRS ~ Jitter\_p + 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 = train\_data)  
  
# Extract R-squared  
r\_squared = summary(model\_lineal)$r.squared  
  
# Extract adjusted R-squared  
adj\_r\_squared = summary(model\_lineal)$adj.r.squared  
  
# Predict on the training data  
predictions\_train = predict(model\_lineal, train\_data)  
  
# Calculate residuals  
residuals\_train = train\_data$total\_UPDRS - predictions\_train  
  
# Calculate RMSE  
rmse\_train = sqrt(mean(residuals\_train^2))  
  
# Predict on the test data  
predictions\_test <- predict(model\_lineal, test\_data)  
  
# Calculate residuals  
residuals\_test <- test\_data$total\_UPDRS - predictions\_test  
  
# Calculate RMSE  
rmse\_test <- sqrt(mean(residuals\_test^2))  
  
# Extract the variable names from the linear regression model  
variables <- names(coef(model\_lineal))[-1]  
  
# Create a data frame for the results  
results\_table <- data.frame(  
 Metric = c("Número de variables", "R-squared", "Adjusted R-squared", "RMSE\_train", "RMSE\_test"),  
 Value = c(length(variables), r\_squared, adj\_r\_squared, rmse\_train, rmse\_test)  
)  
  
# Print the result table  
print(results\_table)

## Metric Value  
## 1 Número de variables 16.0000000  
## 2 R-squared 0.1048411  
## 3 Adjusted R-squared 0.1016745  
## 4 RMSE\_train 10.1276844  
## 5 RMSE\_test 10.2516307

cat("Variables usadas en el modelo: ")

## Variables usadas en el modelo:

cat(variables, sep=", ")

## Jitter\_p, Jitter\_Abs, Jitter\_RAP, Jitter\_PPQ5, Jitter\_DDP, Shimmer, Shimmer\_dB, Shimmer\_APQ3, Shimmer\_APQ5, Shimmer\_APQ11, Shimmer\_DDA, NHR, HNR, RPDE, DFA, PPE

Al no considerar el “sujeto” se viola la suposición de independencia. En este tipo de modelos, se asume que todas las observaciones son independientes. Al no serlo, se construirá un modelo que incluirá métricas erróneas. De manera similar, se pierde precisión al no tener en cuenta la correlación entre los datos. El error estándar de los coeficientes estimados puede subestimarse, dando intervalos de confianza más estrechos.

Adicionalmente, se aumenta el error de Tipo 1 (rechazar incorrectamente H0 haciendo que una variable sea significativa cuando no lo es). Esto sucede porque los datos de un mismo individuo tienden a ser más similares, inflando así la significación de los resultados.

Finalmente, se hace más complicado detectar las variaciones entre en un mismo individuo.

### (b) Regresión lineal con AIC

library(MASS)  
  
# Perform stepwise variable selection based on AIC  
model\_stepwise <- stepAIC(model\_lineal, direction = "both", trace = FALSE)  
  
# Extract R-squared  
r\_squared = summary(model\_stepwise)$r.squared  
  
# Extract adjusted R-squared  
adj\_r\_squared = summary(model\_stepwise)$adj.r.squared  
  
# Predict on the training data  
predictions\_train = predict(model\_stepwise, train\_data)  
  
# Calculate residuals  
residuals\_train = train\_data$total\_UPDRS - predictions\_train  
  
# Calculate RMSE  
rmse\_train = sqrt(mean(residuals\_train^2))  
  
# Predict on the test data  
predictions\_test <- predict(model\_stepwise, test\_data)  
  
# Calculate residuals  
residuals\_test <- test\_data$total\_UPDRS - predictions\_test  
  
# Calculate RMSE  
rmse\_test <- sqrt(mean(residuals\_test^2))  
  
# Extract the variable names from the linear regression model  
variables <- names(coef(model\_stepwise))[-1]  
  
# Create a data frame for the results  
results\_table <- data.frame(  
 Metric = c("Número de variables", "R-squared", "Adjusted R-squared", "RMSE\_train", "RMSE\_test"),  
 Value = c(length(variables), r\_squared, adj\_r\_squared, rmse\_train, rmse\_test)  
)  
  
# Print the result table  
print(results\_table)

## Metric Value  
## 1 Número de variables 12.0000000  
## 2 R-squared 0.1045613  
## 3 Adjusted R-squared 0.1021877  
## 4 RMSE\_train 10.1292671  
## 5 RMSE\_test 10.2510700

cat("Variables usadas en el modelo: ")

## Variables usadas en el modelo:

cat(variables, sep=", ")

## Jitter\_Abs, Jitter\_DDP, Shimmer, Shimmer\_dB, Shimmer\_APQ5, Shimmer\_APQ11, Shimmer\_DDA, NHR, HNR, RPDE, DFA, PPE

Removing a variable solely based on its lack of significance may affect the overall fit of the model and the relationships between other variables. A variable that is not individually significant may contribute to the model’s overall predictive power when combined with other variables or interacted with other predictors. Therefore, it’s crucial to assess the model’s overall performance, such as through measures like R-squared or adjusted R-squared, and consider the context and theoretical significance of the variables

The significance of a variable can be influenced by multicollinearity, which occurs when predictor variables are highly correlated with each other. In the presence of multicollinearity, the individual coefficients and their significance can be unstable or misleading. It’s important to check for multicollinearity among the variables and consider its potential impact on the significance of individual predictors.

In this example, model\_linear is the initial linear regression model you built using all the variables of interest. The stepAIC() function from the MASS package is used to perform the stepwise variable selection based on AIC.

The direction argument specifies the direction of the stepwise procedure. “both” allows variables to be added or removed from the model. Other options include “backward” for variable removal only and “forward” for variable addition only.

### (c) Regresión por componentes principales

Performing regression using principal components involves transforming the predictor variables into a set of principal components and then using these components as predictors in the regression model.

# Select predictor variables  
predictor\_vars = train\_data[7:22]  
  
# Initialize variables for best model  
best\_rmse <- Inf  
best\_components <- 0  
best\_model <- NULL  
  
# Perform principal component analysis (PCA) on the predictor variables  
pca <- prcomp(predictor\_vars, scale = TRUE)  
  
# Extract the principal components  
pcs <- pca$x  
  
# Identify the number of components with minimum RMSE  
num\_components <- 1:ncol(pcs)  
rmse\_values <- numeric(length(num\_components))  
  
for (i in num\_components) {  
 # Select the first i principal components  
 pcs\_selected <- pcs[, 1:i]  
   
 # Create a regression model using the selected components  
 model\_pca <- lm(total\_UPDRS ~ pcs\_selected, data = train\_data)  
   
 # Predict on the test data using the model  
 predictions <- predict(model\_pca, test\_data)  
   
 # Calculate residuals  
 residuals <- test\_data$total\_UPDRS - predictions  
   
 # Calculate RMSE  
 rmse\_values[i] <- sqrt(mean(residuals^2))  
   
 # Check if current model has lower RMSE  
 if (rmse\_values[i] < best\_rmse) {  
 best\_rmse <- rmse\_values[i]  
 best\_components <- i  
 best\_model <- model\_pca  
 }  
}

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

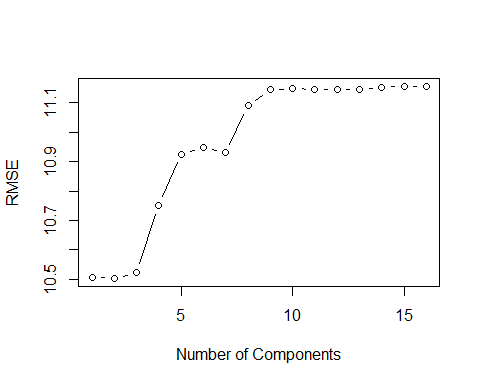
## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

## Warning in test\_data$total\_UPDRS - predictions: longitud de objeto mayor no es  
## múltiplo de la longitud de uno menor

# Plot the RMSE values against the number of components  
plot(num\_components, rmse\_values, type = "b", xlab = "Number of Components", ylab = "RMSE")



# Identify the number of components with the minimum RMSE  
min\_rmse <- min(rmse\_values)  
optimal\_components <- num\_components[which.min(rmse\_values)]  
  
# Print the results  
print(paste("Minimum RMSE:", min\_rmse))

## [1] "Minimum RMSE: 10.503522251986"

print(paste("Optimal Number of Components:", optimal\_components))

## [1] "Optimal Number of Components: 2"

The code performs principal component analysis (PCA) on the predictor variables using the prcomp() function. The resulting principal components (pcs) are then used to construct regression models with different numbers of components (ranging from 1 to the total number of components).

For each model, the code predicts the outcome variable on the test data, calculates the residuals, and computes the root mean squared error (RMSE). The RMSE values are stored in the rmse\_values vector.

The code then plots the RMSE values against the number of components to visualize the relationship. The number of components that yields the minimum RMSE is identified, and the corresponding results are printed.

Yes, in general, a lower RMSE indicates a better-fitting model. RMSE (Root Mean Squared Error) is a commonly used measure of the average prediction error of a regression model. It represents the square root of the average squared differences between the predicted values and the actual values of the outcome variable.

Since RMSE measures the magnitude of the prediction errors, a smaller RMSE implies that the model’s predictions are, on average, closer to the actual values.

# Predict on the training data using the model  
predictions\_train <- predict(model\_pca, train\_data)  
   
# Predict on the test data using the model  
predictions\_test <- predict(model\_pca, test\_data)

## Warning: 'newdata' had 1335 rows but variables found have 4540 rows

# Calculate residuals for training and test data  
residuals\_train <- train\_data$total\_UPDRS - predictions\_train  
residuals\_test <- test\_data$total\_UPDRS - predictions\_test

## Warning in test\_data$total\_UPDRS - predictions\_test: longitud de objeto mayor  
## no es múltiplo de la longitud de uno menor

# Calculate R-squared for training data  
R\_squared\_train <- summary(model\_pca)$r.squared  
   
# Calculate adjusted R-squared for training data  
num\_predictors <- length(coefficients(model\_pca)) - 1  
n <- nrow(train\_data)  
adjusted\_R\_squared\_train <- 1 - (1 - R\_squared\_train) \* ((n - 1) / (n - num\_predictors - 1))  
   
# Calculate RMSE for training data  
RMSE\_train <- sqrt(mean(residuals\_train^2))  
   
# Calculate RMSE for test data  
RMSE\_test <- sqrt(mean(residuals\_test^2))  
  
# Extract the variable names from the linear regression model  
variables <- names(coef(best\_model))[-1]  
  
# Create a data frame for the results  
results\_table <- data.frame(  
 Metric = c("Número de variables", "R-squared", "Adjusted R-squared", "RMSE\_train", "RMSE\_test"),  
 Value = c(length(variables), R\_squared\_train, adjusted\_R\_squared\_train, RMSE\_train, RMSE\_test)  
)  
  
# Print the result table  
print(results\_table)

## Metric Value  
## 1 Número de variables 2.0000000  
## 2 R-squared 0.1048411  
## 3 Adjusted R-squared 0.1016745  
## 4 RMSE\_train 10.1276844  
## 5 RMSE\_test 11.1553273

cat("Variables usadas en el modelo: ")

## Variables usadas en el modelo:

cat(variables, sep=", ")

## pcs\_selectedPC1, pcs\_selectedPC2

Nota: el mejor modelo según el RMSE test no el train.

### (d) Ridge regression

Ridge regression is a model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization. When the issue of multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values being far away from the actual values.

Yes, you can adjust the model using Ridge regression. Ridge regression is a regularization technique that introduces a penalty term to the least squares objective function, helping to reduce the impact of multicollinearity and potentially improve the model’s performance.

# install.packages('glmnet')  
library(glmnet)

## Warning: package 'glmnet' was built under R version 4.3.1

## Loading required package: Matrix

## Loaded glmnet 4.1-7

# Select predictor variables as matrix  
x\_train = data.matrix(train\_data[7:22])  
y\_train = train\_data$total\_UPDRS  
  
x\_test = data.matrix(test\_data[7:22])  
y\_test = test\_data$total\_UPDRS  
  
# Fit model  
ridge\_model = glmnet(x\_train, y\_train, alpha = 0, lambda = 1)  
  
# Make predictions  
predictions\_train <- predict(ridge\_model, newx = x\_train)  
predictions\_test <- predict(ridge\_model, newx = x\_test)  
  
# R squared  
SSR <- sum((predictions\_train - y\_train)^2) # Sum of Squares of Residuals  
SST <- sum((y\_train - mean(y\_train))^2) # Total Sum of Squares  
r\_squared <- 1 - (SSR / SST)  
  
# Adjusted R squared  
n <- length(y\_train) # Number of observations  
p <- ncol(x\_train) # Number of predictors (excluding intercept)  
adj\_r\_squared <- 1 - (1 - r\_squared) \* ((n - 1) / (n - p - 1))  
  
# RMSE\_train  
rmse\_train <- sqrt(mean((predictions\_train - y\_train)^2))  
  
# RMSE\_test  
rmse\_test <- sqrt(mean((predictions\_test - y\_test)^2))  
  
# Extract the coefficients  
coefficients <- coef(ridge\_model)  
  
# Extract the variable names  
variables <- colnames(x\_train)  
  
# Create a data frame for the results  
results\_table <- data.frame(  
 Metric = c("Número de variables", "R-squared", "Adjusted R-squared", "RMSE\_train", "RMSE\_test"),  
 Value = c(length(variables), r\_squared, adj\_r\_squared, rmse\_train, rmse\_test)  
)  
  
# Print the result table  
print(results\_table)

## Metric Value  
## 1 Número de variables 16.00000000  
## 2 R-squared 0.09706003  
## 3 Adjusted R-squared 0.09386590  
## 4 RMSE\_train 10.17160591  
## 5 RMSE\_test 10.20596170

cat("Variables usadas en el modelo: ")

## Variables usadas en el modelo:

cat(variables, sep=", ")

## Jitter\_p, Jitter\_Abs, Jitter\_RAP, Jitter\_PPQ5, Jitter\_DDP, Shimmer, Shimmer\_dB, Shimmer\_APQ3, Shimmer\_APQ5, Shimmer\_APQ11, Shimmer\_DDA, NHR, HNR, RPDE, DFA, PPE

alpha = 0 specifies Ridge regression (since alpha = 0 represents the L2 penalty), and lambda = 1 controls the amount of regularization. You can adjust the value of lambda to control the amount of shrinkage applied to the coefficients.

### (e) motor\_UPDRS como respuesta

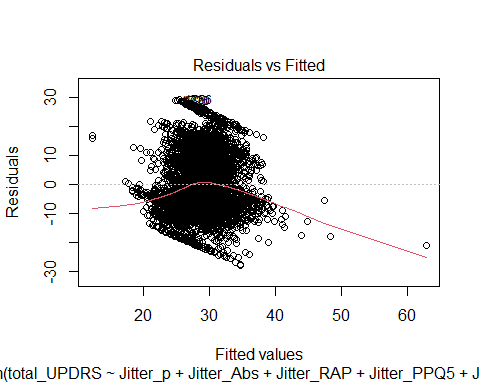
A value of “0.1” for both R2 and R2 adjusted means that the predictors included in the model explain approximately 10% of the variance in the dependent variable. This indicates a relatively weak relationship between the predictors and the response variable. Keep in mind that the interpretation of the R2 and R2 adjusted values depends on the specific context and the nature of the data being modeled.

Yo diria que si porque es puta mierda. No cumpliria el objetivo principal: El principal objetivo es predecir el UPDRS total a partir de las 16 medidas de voz. Pero esque este tampoco lo hace porque es basssurrra.

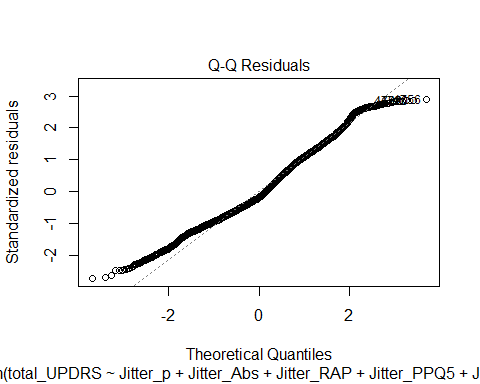
### (f) Análisis de residuos

In statistics, ordinary least squares (OLS) is a type of linear least squares method for choosing the unknown parameters in a linear regression model (with fixed level-one effects of a linear function of a set of explanatory variables) by the principle of least squares: minimizing the sum of the squares of the differences between the observed dependent variable (values of the variable being observed) in the input dataset and the output of the (linear) function of the independent variable.

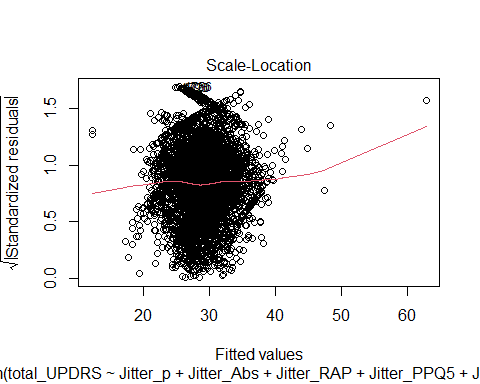
# Residual plot  
plot(model\_lineal, which = 1)



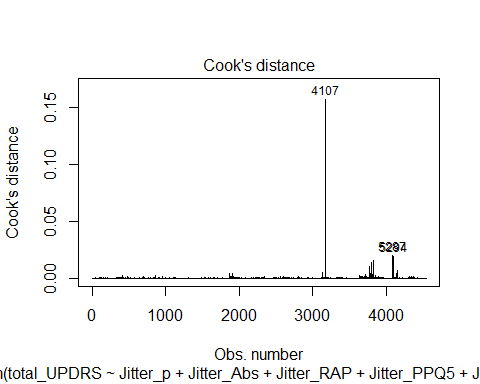
# Normal Q-Q plot  
plot(model\_lineal, which = 2)



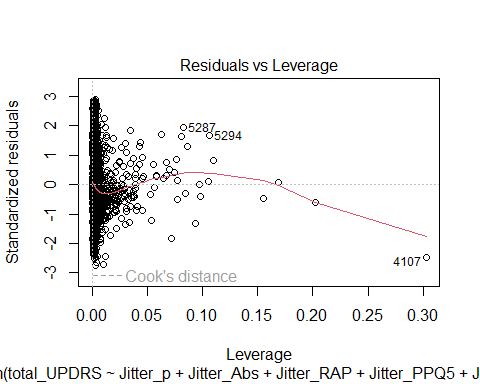
# Scale-location plot (square root of standardized residuals)  
plot(model\_lineal, which = 3)



# Cook's distance  
plot(model\_lineal, which = 4)



# Residuals vs. fitted values plot  
plot(model\_lineal, which = 5)

 which = 1: Residuals vs. Fitted: Alrededor de 0, mas o menos lineal which = 2: Normal Q-Q plot: No son muy normal en linea recta which = 3: Scale-Location plot: homocedasticidad mas o menos, no es cono which = 4: Cook’s distance plot: NO Hay puntos influyentes. Más grande cook más influye. Más 1 es influyente which = 5: Residuals vs. Leverage plot: No Existen puntos influyentes

Multicollinearity refers to a situation where independent variables in a regression model are highly correlated with each other. It can cause issues in the interpretation of coefficients and affect the stability and reliability of the regression model. To study multicollinearity in R, you can use the following approaches:

cor\_matrix <- cor(train\_data[, c("Jitter\_p","Jitter\_Abs","Jitter\_RAP","Jitter\_PPQ5","Jitter\_DDP",  
"Shimmer","Shimmer\_dB","Shimmer\_APQ3","Shimmer\_APQ5","Shimmer\_APQ11", "Shimmer\_DDA","NHR","HNR","RPDE","DFA","PPE")])  
  
# install.packages("corrplot")  
library(corrplot)

## Warning: package 'corrplot' was built under R version 4.3.1

## corrplot 0.92 loaded

corrplot(cor\_matrix, method = "number")



library(car)

## Warning: package 'car' was built under R version 4.3.1

## Loading required package: carData

vif\_values <- vif(model\_lineal)  
  
tolerance\_values <- 1/vif\_values  
  
# Extract the variable names from the linear regression model  
variables <- names(coef(model\_lineal))[-1]  
  
# Create a data frame for the results  
results\_cor <- data.frame(  
 VIF = vif\_values,  
 Tolerance = tolerance\_values  
)  
  
print(results\_cor)

## VIF Tolerance  
## Jitter\_p 97.074438 0.01030137310282  
## Jitter\_Abs 7.567100 0.13215103077251  
## Jitter\_RAP 1278727.253152 0.00000078202760  
## Jitter\_PPQ5 32.845531 0.03044554263565  
## Jitter\_DDP 1279217.387951 0.00000078172796  
## Shimmer 165.617638 0.00603800422140  
## Shimmer\_dB 75.369763 0.01326792023822  
## Shimmer\_APQ3 24471929.793367 0.00000004086314  
## Shimmer\_APQ5 50.119652 0.01995225360210  
## Shimmer\_APQ11 16.306123 0.06132665794355  
## Shimmer\_DDA 24471089.285893 0.00000004086455  
## NHR 7.757467 0.12890805722562  
## HNR 5.363433 0.18644774263020  
## RPDE 2.114996 0.47281408645254  
## DFA 1.600104 0.62495957160617  
## PPE 4.345757 0.23010953113121

Todos los tipos de Jitter están muy relacionados con todos los tipos de Jitter y los Shimmer con los Shimmer.

# Ejercicio 3

### (a) Comparación modelos con y sin puntos influyentes

### (b) Cálculo del RMSE robusto

### (c) LTS o Huber

# ANEXO

### Código