PRÁCTICA 3A: REGRESIÓN LOGÍSTICA

ANÁLISIS ESTADÍSTICO MULTIVARIANTE

GRADO EN CIENCIA E INGENIERÍA DE DATOS

Sumario: En esta práctica mostramos cómo llevar a cabo un análisis de Regresión Logística (RLogistica) usando R. La estimación de los parámetros del modelo se realizará usando la función *glm()* de R, que sirve para ajustar modelos lineales generalizados (en particular, el modelo RLogistica). Al igual que en el caso del análisis RLM, la práctica contempla la selección del modelo (o selección de regresores), la validación del modelo y la obtención de predicciones. Además, se muestra cómo convertir el análisis RLogistica (variable respuesta binaria) en un problema de clasificación en dos grupos, introduciendo indicadores de la bondad del ajuste como la matriz de confusión o la curva ROC.

1. Conjunto de datos y análisis descriptivo previo

Los datos que usaremos en esta práctica han sido descargados de la web de la Universidad de UCLA, cuya práctica sobre Regresión Logística ha servido de base para el desarrollo de la nuestra. El conjunto de datos `binary.csv` contiene datos simulados del proceso de admisión de 400 alumnos en una universidad estadounidense. Las variables del fichero son: `gre` (puntuación obtenida en un test de admisión, de 0 a 1000), `gpa` (puntuación media obtenida en la etapa educativa anterior, equivalente a nuestro bachillerato, medida de 0 a 4), `rank` (prestigio del centro educativo de procedencia, con niveles del 1 al 4, siendo el 1 el más prestigioso) y `admit` (variable binaria que toma el valor 1 si el alumno es admitido en la universidad y 0 en caso contrario).

Objetivo del estudio: analizar cómo afectan las variables `gre`, `gpa` y `rank` en la admisión de los alumnos. Obsérvese que disponemos de tres predictores, 2 de ellos de tipo continuo (`gre` y `gpa`) y 1 categórico (`rank`), y que la variable respuesta (`admit`) es binaria, de manera que plantearemos un análisis de Regresión Logística.

Comenzamos cargando los datos y haciendo un resumen de los mismos.

```
library("tidyverse")
mydata <- read.csv("../data/binary.csv")
summary(mydata)
```

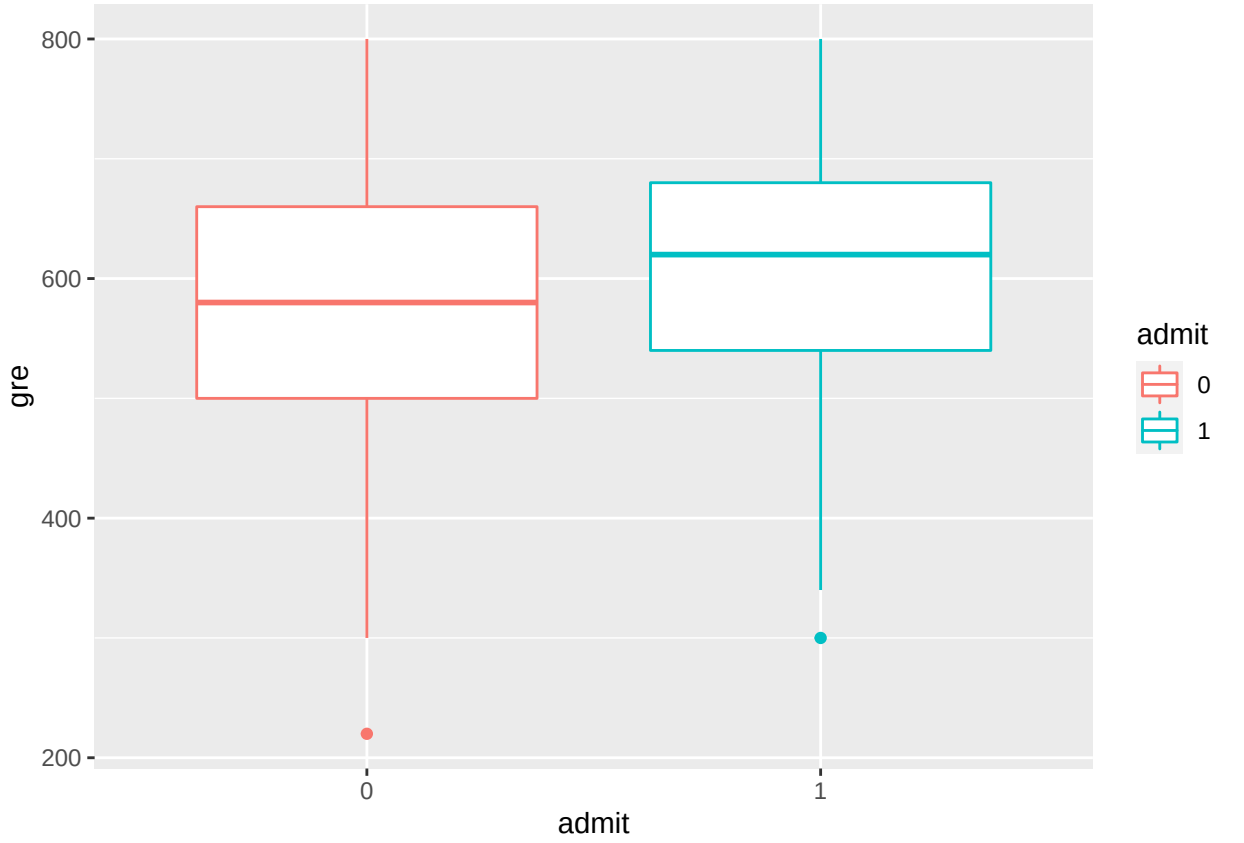
```
##      admit             gre             gpa             rank
##  Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000
##  Median :0.0000   Median :580.0   Median :3.395   Median :2.000
##  Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485
##  3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000
##  Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000
```

Pasamos las variables **admit** y **rank** a tipo factor, aunque no sería necesario para algunos análisis.

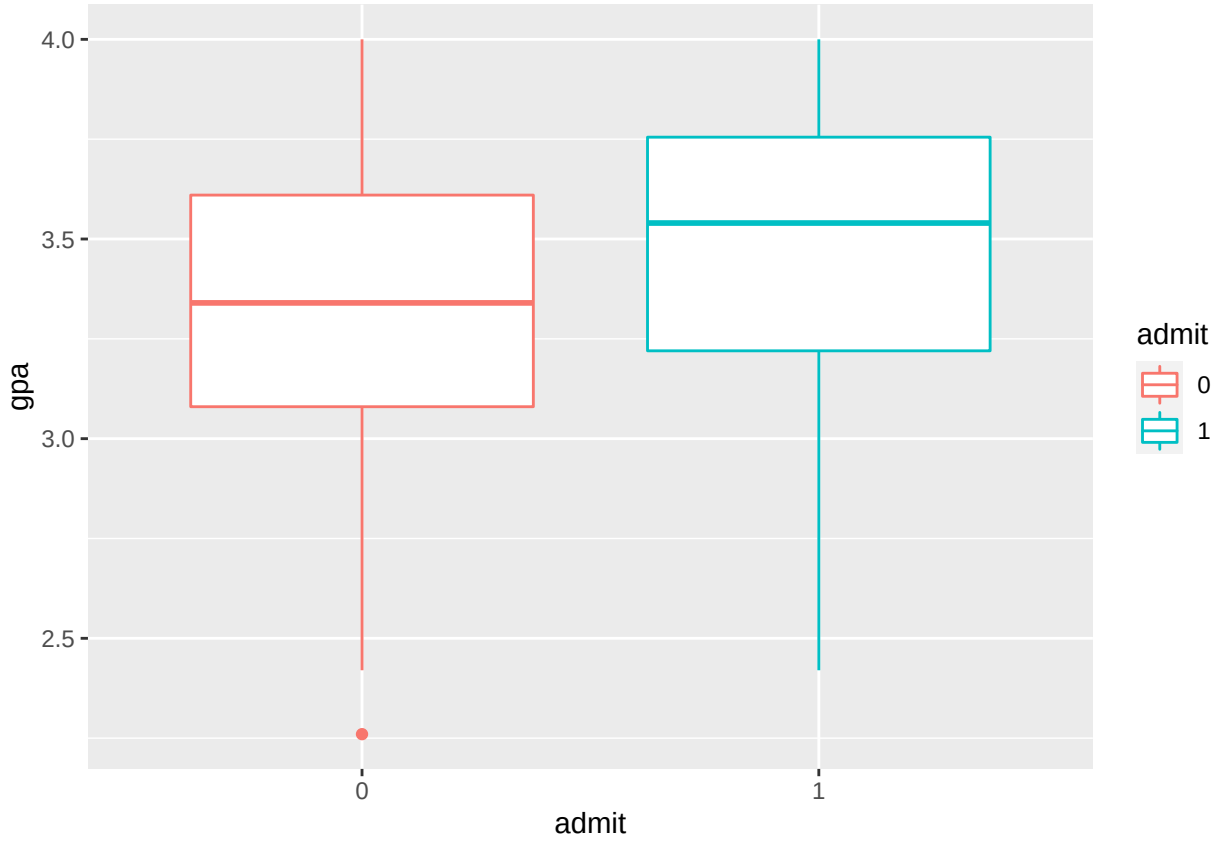
```
mydata$rank <- factor(mydata$rank)
mydata$admit <- factor(mydata$admit)
# summary(mydata)
```

Realizamos un análisis descriptivo inicial.

```
#Diagramas de caja comparativos para cada grupo de la variable respuesta
mydata %>%
  ggplot(aes(x = admit, y = gre)) +
  geom_boxplot(aes(color = admit))
```

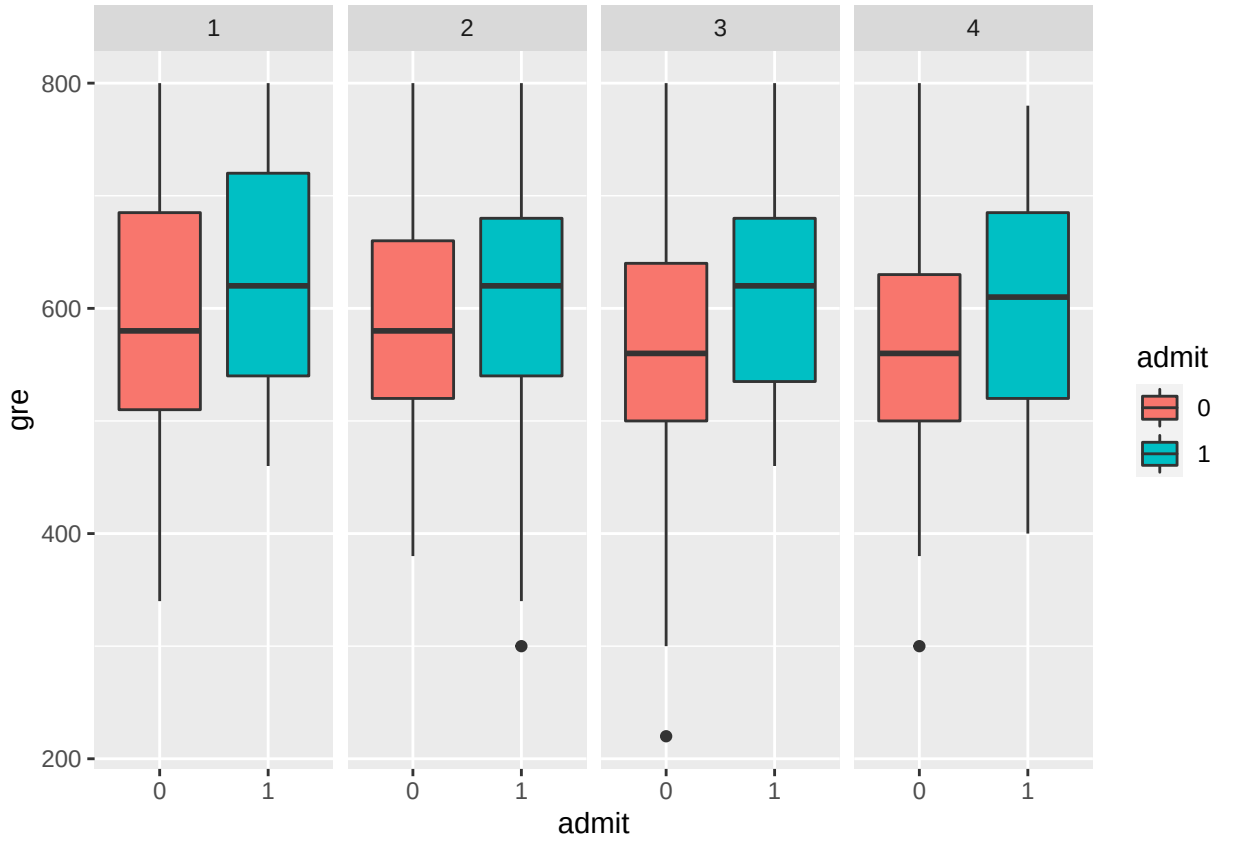



```
mydata %>%
  ggplot(aes(x = admit, y = gpa)) +
  geom_boxplot(aes(color = admit))
```

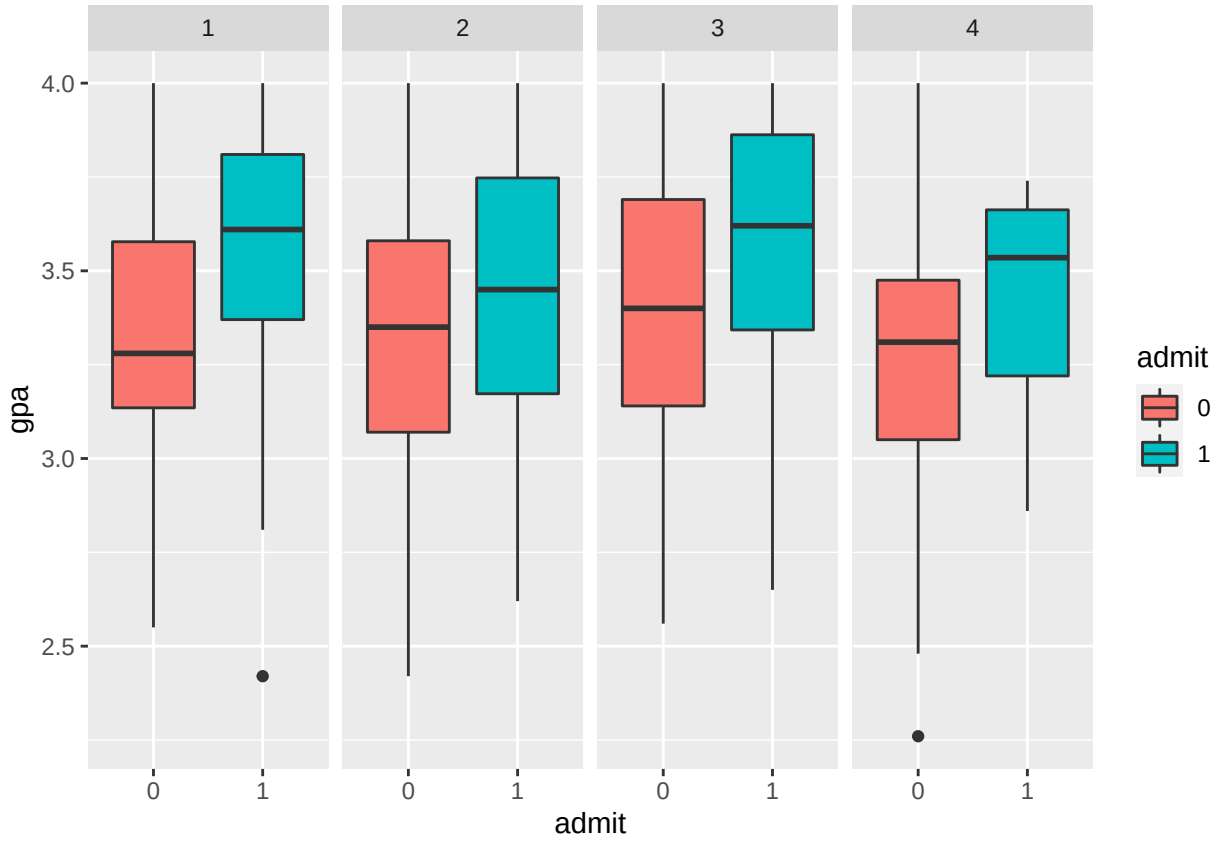



```
#También se podrían haber realizado esos diagramas con:
# boxplot(mydata$gre ~ mydata$admit)
# boxplot(mydata$gpa ~ mydata$admit)
```

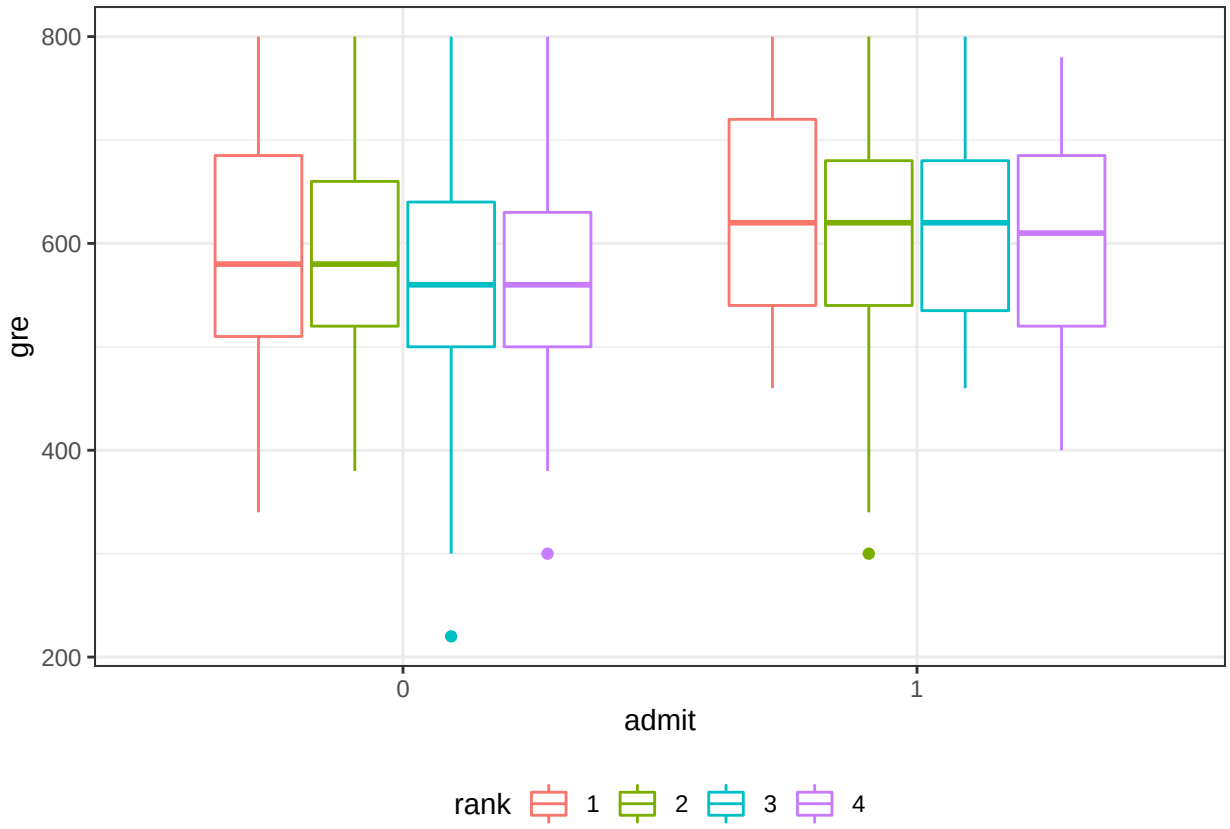
```
#Ahora separamos los boxplot para cada nivel de la categoria rank
mydata %>%
  ggplot(aes(x = admit, y = gre)) +
  geom_boxplot(aes(fill = admit)) +
  facet_grid(.~ rank)
```



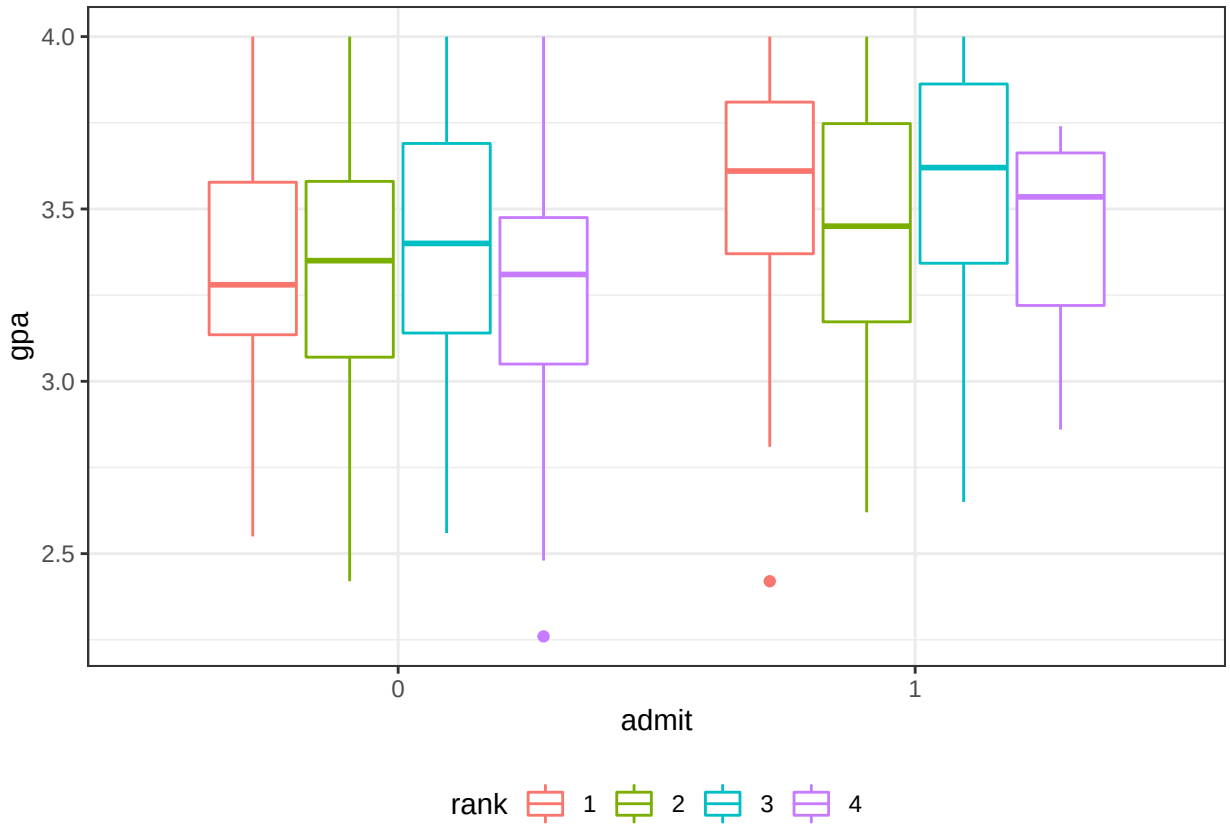

```
mydata %>%
  ggplot(aes(x = admit, y = gpa)) +
  geom_boxplot(aes(fill = admit)) +
  facet_grid (.~ rank)
```




```
ggplot(data = mydata, aes(x = admit, y = gre, color = rank)) +
  geom_boxplot()+
  theme_bw() +
  theme(legend.position = "bottom")
```



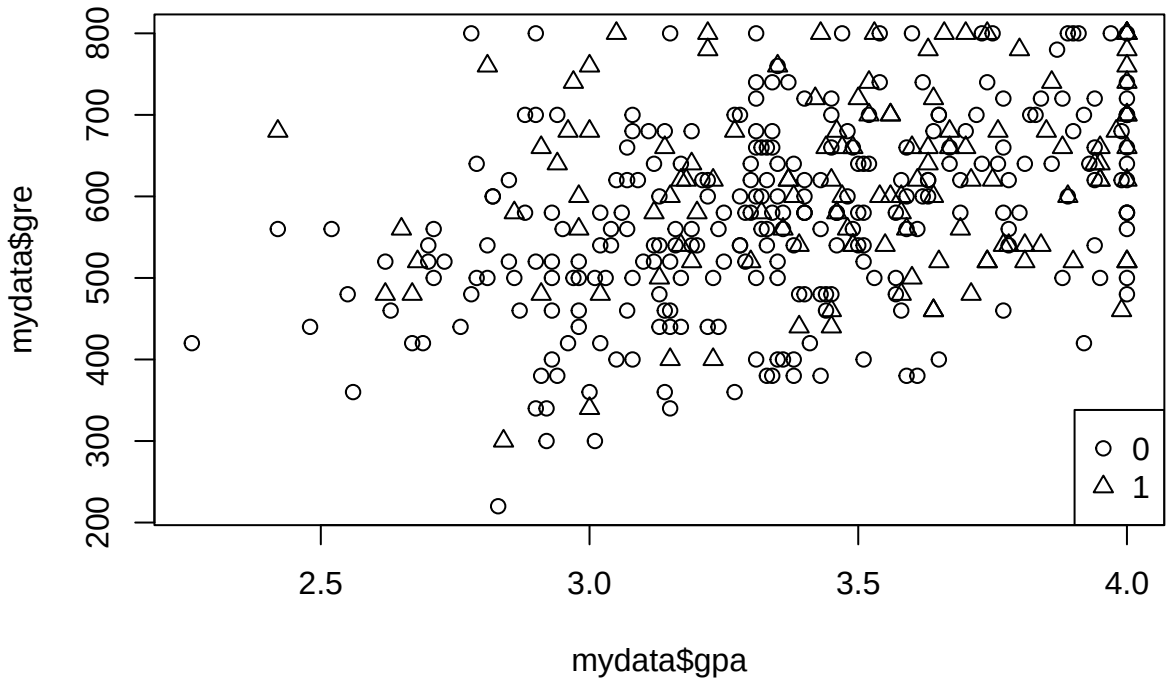

```
ggplot(data = mydata, aes(x = admit, y = gpa, color = rank)) +
  geom_boxplot()+
  theme_bw() +
  theme(legend.position = "bottom")
```



En los gráficos anteriores observamos algunos valores atípicos tanto en gpa como en gre que merecerían una interpretación más detallada. Por ejemplo, en el último gráfico tenemos una observación atípica (individuo de la fila 290) con una nota muy baja en gpa, procedente de un centro tipo 4 y que no ha sido admitido. También tenemos una observación atípica (individuo de la fila 373) con una nota muy baja en gpa, procedente de un centro tipo 1 pero que sí ha sido admitido.

Estos individuos no se detectan como observaciones atípicas si hacemos la nube de puntos de gre y gpa (distinguiendo los subgrupos admitido y no admitido), y tampoco se observa correlación entre dichas variables:

```
plot(mydata$gpa, mydata$gre, pch = as.integer(mydata$admit))
legend('bottomright', legend=c('0','1'), pch=1:2)
```



Ahora vamos a comprobar que no hay celdas vacías (o frecuencias muy bajas) en las tablas de contingencia de los predictores categóricos y la variable respuesta. Si hay celdas vacías o con pocos casos, el modelo RLogistica puede ser inestable y poco fiable.

```
addmargins(table(mydata$admit, mydata$rank,
                 dnn = c("admit", "rank")))
```

```
##       rank
## admit   1   2   3   4 Sum
##   0    28  97  93  55 273
##   1    33  54  28  12 127
##   Sum  61 151 121  67 400
```

Tras el estudio descriptivo previo, podemos observar en los gráficos que la tendencia central (mediana) de las puntuaciones en `gre` y `gpa` son más altas para los alumnos admitidos en la universidad que para los no admitidos. Y mirando la tabla de contingencia anterior, observamos que la tasa de admitidos frente a no admitidos es mayor para los centros educativos de procedencia con más prestigio (`rank = 1`).

2. Estimación de los parámetros del modelo RLogistica y métodos de selección de regresores

Recordemos que el modelo teórico de Regresión Logística para estos datos viene dado por:

$$\log\left(\frac{p}{1-p}\right) = \theta_0 + \theta_1 \cdot gre + \theta_2 \cdot gpa + \theta_3 \cdot rank2 + \theta_4 \cdot rank3 + \theta_5 \cdot rank4\,,$$

siendo $p = \Pr(Y = 1)$.

Hay que entender las variables `rank2`, `rank3` y `rank4` como variables binarias (*dummy*) que toman el valor 1 si el valor de `rank` se corresponde con su índice y valen cero en caso contrario. Por ejemplo, si $rank = 3$, entonces $rank2 = 0$, $rank3 = 1$ y $rank4 = 0$.

Estimamos el modelo logístico usando todos los predictores. Para ello, usamos la función *glm()* de R que sirve para estimar los modelos lineales generalizados. En particular, el modelo logístico requiere poner el argumento `family = "binomial"` que lleva por defecto a la función logística como función `link`.

```
mylogit <- glm(admit ~ gre + gpa + rank, data = mydata, family = "binomial")
summary(mylogit)
```

```
##
## Call:
## glm(formula = admit ~ gre + gpa + rank, family = "binomial",
##     data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

De los resultados anteriores, obtenemos que el modelo lineal ajustado (que explica el logaritmo de los *odds*) viene dado por:

$$-3.989979 + 0.002264 \cdot gre + 0.804038 \cdot gpa - 0.675443 \cdot rank2 - 1.340204 \cdot rank3 - 1.551464 \cdot rank4$$

La **interpretación de los coefientes** del modelo RLogistica sería la siguiente:

Por cada unidad adicional en la variable `gre`, el logaritmo de los *odds* de admitido frente a no admitido aumenta en 0.002264.

Por cada unidad adicional en la variable `gpa`, el logaritmo de los *odds* de admitido frente a no admitido aumenta en 0.804038.

Haber asistido a un centro educativo con prestigio 2 frente a prestigio 1, produce un cambio en el logaritmo de los *odds* de -0.675443, es decir, el logaritmo de los *odds* de admitido frente a no admitido disminuye en 0.675443.

Además de la estimación de los coeficientes de regresión, obtenemos sus p-valores correspondientes. Observamos que en este caso son todos significativos (inferiores a 0.05), lo que nos hace sospechar que el modelo no va a ser reducible.

Como en el caso de RLM, podemos aplicar los **métodos de selección de regresores** *backward*, *fordward* y *stepwise*, para ver si el modelo completo es reducible a otro más sencillo (con menos predictores).

```
modelo_backward <- step(mylogit, direction = "backward")
```

```
## Start:  AIC=470.52
## admit ~ gre + gpa + rank
##
##        Df Deviance    AIC
## <none>      458.52 470.52
## - gre   1   462.88 472.88
## - gpa   1   464.53 474.53
## - rank  3   480.34 486.34
```

Los resultados anteriores indican que el modelo no es reducible. Se obtienen resultados similares con los otros métodos, como mostramos a continuación.

```
modelo_nulo <- glm(admit ~ 1, data = mydata, family = "binomial")
modelo_forward <- step(modelo_nulo, scope = formula(mylogit), direction = "forward")
```

```
## Start:  AIC=501.98
## admit ~ 1
##
##        Df Deviance    AIC
## + rank  3   474.97 482.97
## + gre   1   486.06 490.06
## + gpa   1   486.97 490.97
## <none>      499.98 501.98
##
## Step:  AIC=482.97
## admit ~ rank
##
##        Df Deviance    AIC
## + gpa   1   462.88 472.88
## + gre   1   464.53 474.53
## <none>      474.97 482.97
##
## Step:  AIC=472.88
## admit ~ rank + gpa
##
##        Df Deviance    AIC
## + gre   1   458.52 470.52
## <none>      462.88 472.88
##
## Step:  AIC=470.52
## admit ~ rank + gpa + gre
```

```
modelo_stepwise <- step(modelo_nulo, scope = formula(mylogit), direction = "both")
```

```
## Start:  AIC=501.98
## admit ~ 1
##
##        Df Deviance    AIC
```

```
## + rank  3   474.97 482.97
## + gre   1   486.06 490.06
## + gpa   1   486.97 490.97
## <none>      499.98 501.98
##
## Step:  AIC=482.97
## admit ~ rank
##
##        Df Deviance    AIC
## + gpa   1   462.88 472.88
## + gre   1   464.53 474.53
## <none>      474.97 482.97
## - rank  3   499.98 501.98
##
## Step:  AIC=472.88
## admit ~ rank + gpa
##
##        Df Deviance    AIC
## + gre   1   458.52 470.52
## <none>      462.88 472.88
## - gpa   1   474.97 482.97
## - rank  3   486.97 490.97
##
## Step:  AIC=470.52
## admit ~ rank + gpa + gre
##
##        Df Deviance    AIC
## <none>      458.52 470.52
## - gre   1   462.88 472.88
## - gpa   1   464.53 474.53
## - rank  3   480.34 486.34
```

Observamos que los tres métodos conducen al modelo completo con los 3 predictores, así que ese será nuestro modelo de referencia.

La selección del modelo también puede realizarse con la función *stepAIC()* del paquete `MASS`, obteniendo los mismos resultados que en el caso anterior.

```
# library("MASS")
# modelo_backward2 <- stepAIC(mylogit, direction = "backward")
# modelo_forward2 <- stepAIC(modelo_nulo, scope = formula(mylogit), direction = "forward")
# modelo_stepwise2 <- stepAIC(modelo_nulo, scope = formula(mylogit), direction = "both")
```

La **bondad del ajuste** del modelo se puede medir a través del `AIC` (criterio de Akaike), siendo mejor el ajuste cuanto menor sea el AIC. También se puede usar la `devianza residual`, siendo mejor el ajuste cuanto menor sea la `devianza`.

```
mylogit$aic #Valor AIC del modelo
```

```
## [1] 470.5175
```

```
mylogit$deviance #Valor de la devianza
```

```
## [1] 458.5175
```

```
#Comprobamos que la devianza del modelo coincide con -2*log(likelihood)
-2*logLik(mylogit)
```

```
## 'log Lik.' 458.5175 (df=6)
```

3. Inferencias en el modelo RLogistica. Predicciones

Una vez validado el modelo RLogistica, lo cual se realizará en una sección posterior, tenemos garantías para realizar inferencias con dicho modelo. Estas inferencias incluyen la obtención de intervalos de confianza para los parámetros de regresión, la prueba de significación del modelo, pruebas individuales de significación de los predictores y también para un conjunto de predictores (test de Wald).

En primer lugar calculamos los **intervalos de confianza** para los parámetros de regresión.

```
confint(mylogit, level = 0.95)
```

```
## Waiting for profiling to be done...
```

```
##                     2.5 %       97.5 %
## (Intercept) -6.2716202334 -1.792547080
## gre          0.0001375921  0.004435874
## gpa          0.1602959439  1.464142727
## rank2       -1.3008888002 -0.056745722
## rank3       -2.0276713127 -0.670372346
## rank4       -2.4000265384 -0.753542605
```

Recordemos que los coeficientes del modelo miden la variación del logaritmo de los *odds* por unidad de cambio en el correspondiente predictor. Tomando exponenciales sobre los coeficientes y sobre los extremos del intervalo de confianza, medimos las variaciones producidas sobre los *odds* directamente (sin tomar logaritmo).

```
## odds ratios
exp(coef(mylogit))
```

```
## (Intercept)         gre         gpa       rank2       rank3       rank4
##   0.0185001   1.0022670   2.2345448   0.5089310   0.2617923   0.2119375
```

```
## odds ratios con sus Intervalos de Confianza
exp(cbind(OR = coef(mylogit), confint(mylogit, level = 0.95)))
```

```
## Waiting for profiling to be done...
```

```
##                    OR       2.5 %    97.5 %
## (Intercept) 0.0185001 0.001889165 0.1665354
## gre         1.0022670 1.000137602 1.0044457
## gpa         2.2345448 1.173858216 4.3238349
## rank2       0.5089310 0.272289674 0.9448343
## rank3       0.2617923 0.131641717 0.5115181
## rank4       0.2119375 0.090715546 0.4706961
```

Medimos la **significación del modelo** comparando la devianza del modelo completo frente a la devianza del modelo nulo. Esta diferencia sigue una Chi-cuadrado con los grados de libertad dados por la diferencia de grados de ambos modelos.

```
diferencia_devianzas <- mylogit$null.deviance - mylogit$deviance
grados_libertad <- mylogit$df.null - mylogit$df.residual
p_valor <- pchisq(diferencia_devianzas, df = grados_libertad, lower.tail = FALSE)
p_valor
```

```
## [1] 7.578194e-08
```

Obsérvese que el p-valor resultante permite concluir que el modelo completo es significativo.

La signficación individual de cada predictor se obtiene con el test de Wald y el estadístico `Z`, que aparecen directamente al hacer el `summary` del modelo (véase más arriba, donde se indicó que en este caso todos los predictores resultan significativos individualmente).

Podemos hacer un estudio de la significaciόm conjunta de la variable `rank`, pues se trata de un factor con 4 niveles que da lugar a 3 predictores (en el resumen del modelo se puede ver que `rank = 1` es el nivel de referencia). Para ello, podemos usar la función *wald.este()* del paquete `aod`.

```
library("aod")
```

```
## Warning: package 'aod' was built under R version 4.1.3
```

```
wald.test(b = coef(mylogit), Sigma = vcov(mylogit), Terms = 4:6)
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 20.9, df = 3, P(> X2) = 0.00011
```

```
#Observar que los términos 4, 5 y 6 del resumen del modelo son
#los correspondientes a rank2, rank3 y rank4
```

Obsérvese que el p-valor resultante permite concluir que la variable "rank" es significativa.

También podemos comparar si hay diferencia entre los coeficientes de dos niveles de `rank`. Por ejemplo, comparamos si hay diferencia entre los coeficientes de los niveles 2 y 3 de `rank` de la siguiente manera. Hay que tener en cuenta el orden de los parámetros de regresión en el modelo logit, que como muestra la salida del modelo son los correspondientes a la constante, gre, gpa, rank2, rank3 y rank4. Por tanto, la combinación (0,0,0,1,-1,0) se refiere al contraste de igualdad entre los coeficientes de rank2 y rank3:

```
combinacion <- cbind(0, 0, 0, 1, -1, 0)
wald.test(b = coef(mylogit), Sigma = vcov(mylogit), L = combinacion)
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 5.5, df = 1, P(> X2) = 0.019
```

Obsérvese que el p-valor resultante permite concluir que hay diferencia significativa entre los coeficientes de `rank = 2` y `rank = 3`.

Para realizar **predicciones** del modelo RLogistica, usamos la función *predict()*, que en este caso permite hacer predicciones de la variable respuesta (argumento `type = "response"`) o del modelo lineal ajustado (argumento por defecto, `type = "link"`).

Por ejemplo, hagamos la predicción para un alumno con `gre = 750`, `gpa = 3.5` y `rank = 2`.

```
nuevos_1 <- data.frame(gre = 750, gpa = 3.5, rank = as.factor(2))
predict(mylogit, newdata = nuevos_1)
```

```
##          1
## -0.1529712
```

```
predict(mylogit, newdata = nuevos_1, type = "response")
```

```
##         1
## 0.4618316
```

Es decir, la predicción indica que para un alumno de esas características, la probabilidad de ser admitido es

$p = \Pr(Y = 1) = 0.4618316$. Podemos comprobar el logaritmo de los odds en este caso sería log(p/(1-p))=-0.1529712.

Si queremos ver el efecto del prestigio del centro de procedencia en la admisión, podemos fijar `gre` y `gpa` en sus valores medios y variar `rank`.

```
nuevos_2 <- with(mydata, data.frame(gre = mean(gre), gpa = mean(gpa), rank = factor(1:4)))
nuevos_2$rank_predic <- predict(mylogit, newdata = nuevos_2, type = "response")
nuevos_2
```

```
##     gre    gpa rank rank_predic
## 1 587.7 3.3899    1   0.5166016
## 2 587.7 3.3899    2   0.3522846
## 3 587.7 3.3899    3   0.2186120
## 4 587.7 3.3899    4   0.1846684
```

De esta forma comprobamos como la variable `rank` es muy relevante sobre la admisión. Por ejemplo, para un alumno con los valores medios de `gre` y `gpa`, el modelo logit indica que el alumno será admitido si rank=1 (probabilidad superior a 0.5) y no será admitido en los otros casos (probabilidades inferiores a 0.5).

Ahora fijamos `gpa` en su media, movemos `rank` en sus 4 niveles y hacemos un grid con el predictor `gre`. Así vemos el efecto de la nota del examen de acceso (predictor `gre`) en la admisión.

```
nuevos_3 <- with(mydata,
                 data.frame(gre = rep(seq(from = 200, to = 800, length.out = 100),
                                      4),
                            gpa = mean(gpa),
                            rank = factor(rep(1:4, each = 100))))

nuevos_3$gre_predic <- predict(mylogit, newdata = nuevos_3, type = "response")
# nuevos_3
```

4. Validación del modelo

De forma resumida, para que las inferencias realizadas anteriormente sean válidas, debemos verificar:

1) Linealidad (entre los predictores y el logaritmo de los *odds*).
2) Independencia de las observaciones.
3) No haya multicolinealidad entre predictores.
4) No existan observaciones influyentes.
5) Matriz de datos completamente representativa. Es decir, que se observen los dos niveles de la variable respuesta para cada nivel de los predictores categóricos. Esto se traduce en:

(5.1) No hay celdas vacías en las tablas de contingencia de la respuesta frente a predictores categóricos.

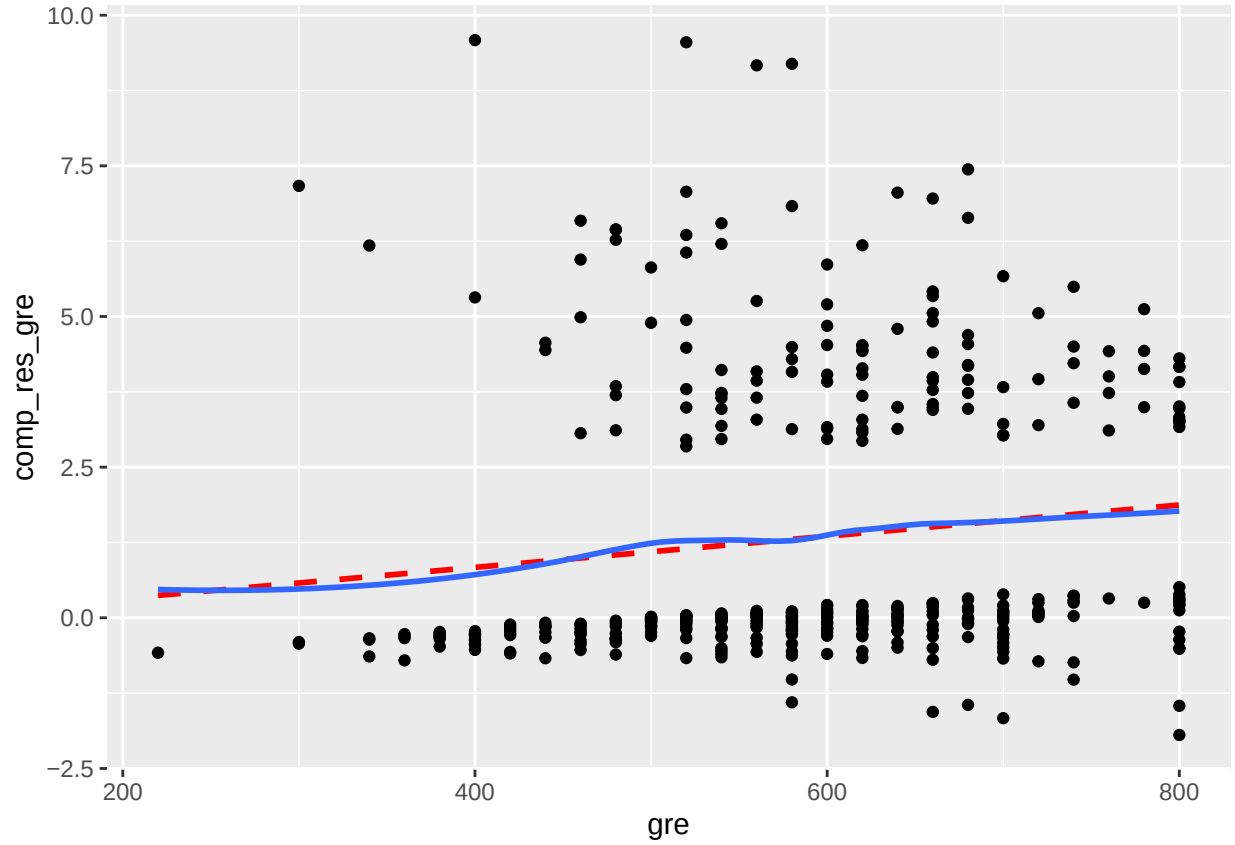
(5.2) No se da el problema de separación perfecta.

Las hipótesis de Normalidad y Homocedasticidad que vimos en RLM no son de aplicación en Regresión Logística.

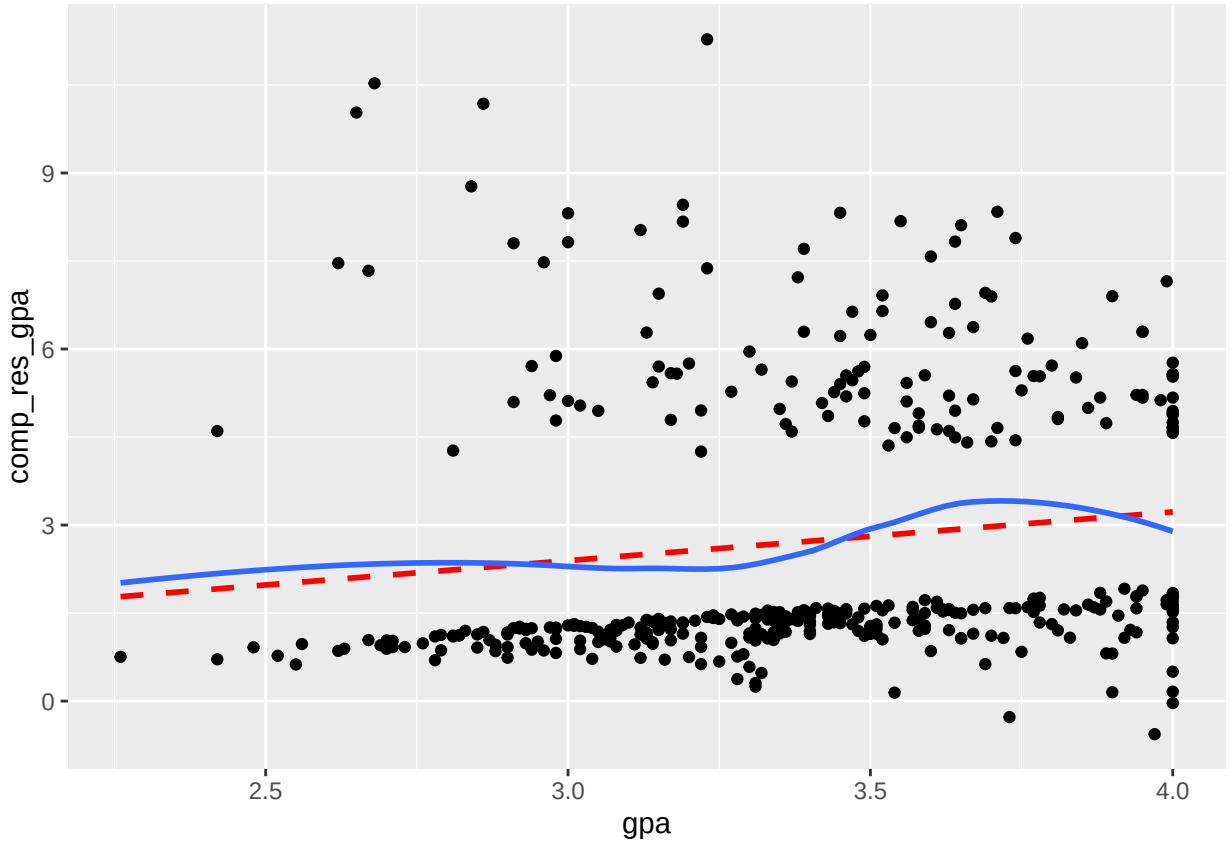
Para diagnosticar violaciones de la **linealidad**, representamos cada predictor frente a su componente + residuos y comprobamos si el ajuste lineal sería adecuado.

```
comp_res_gre <- coef(mylogit)["gre"]*mydata$gre + mylogit$residuals
comp_res_gpa <- coef(mylogit)["gpa"]*mydata$gpa + mylogit$residuals

mydata2 <- mydata
```

```
mydata2$comp_res_gre <- comp_res_gre
mydata2$comp_res_gpa <- comp_res_gpa

ggplot(data = mydata2, aes(x = gre, y = comp_res_gre)) +
  geom_point() +
  geom_smooth(color = "red", method = "lm", linetype = 2, se = F) +
  geom_smooth(se = F)
```

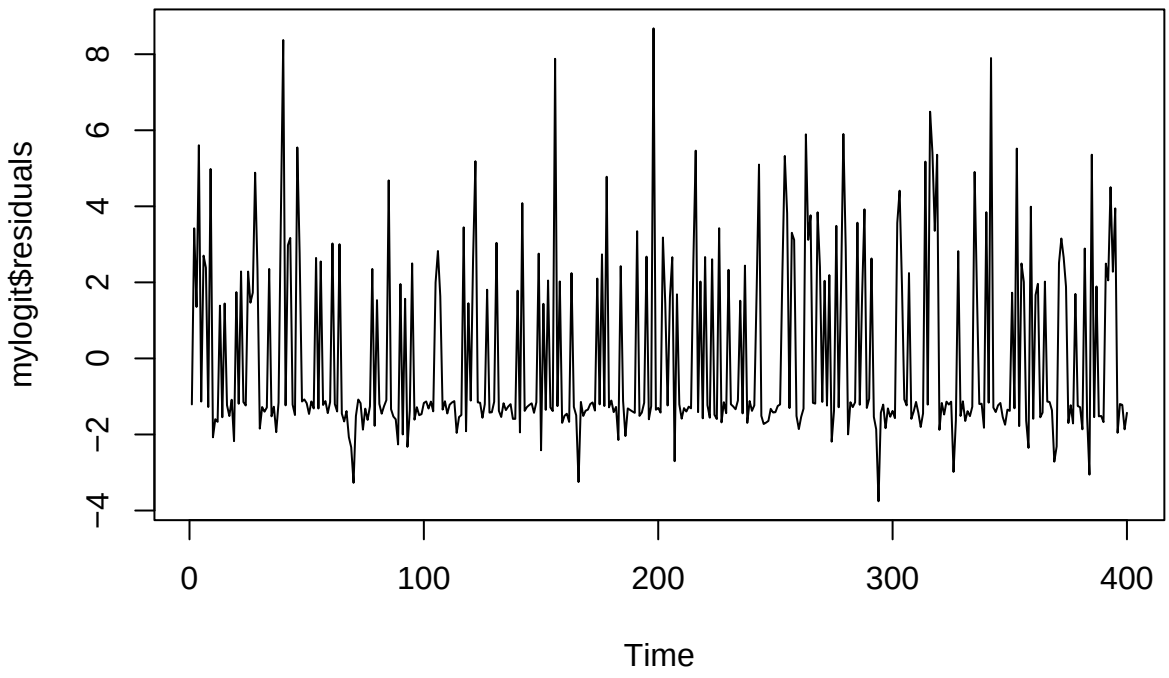



```
ggplot(data = mydata2, aes(x = gpa, y = comp_res_gpa)) +
  geom_point() +
  geom_smooth(color = "red", method = "lm", linetype = 2, se = F) +
  geom_smooth(se = F)
```



Para analizar la **independencia**, podemos hacer un gráfico temporal de los residuos, como se hizo en RLM. No deben observarse patrones ni tendencias.

```
ts.plot(mylogit$residuals)
```



Para la **multicolinealidad**, calculamos los VIFs, como en RLM. En este caso vemos que no existe asociación lineal entre los predictores. Recordemos que valores del VIF superiores a 7 son indicativos de existencia de multicolinealidad.

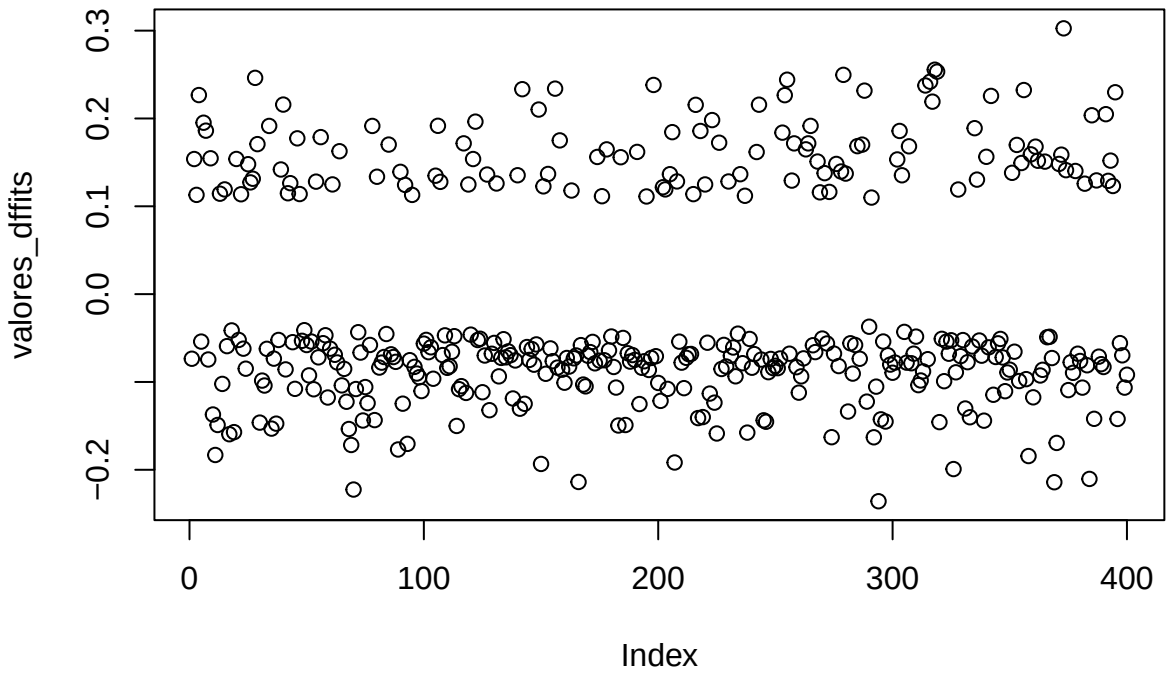
```
rms::vif(mylogit)
```

```
##      gre      gpa    rank2    rank3    rank4
## 1.134377 1.155902 1.908139 1.808731 1.467641
```

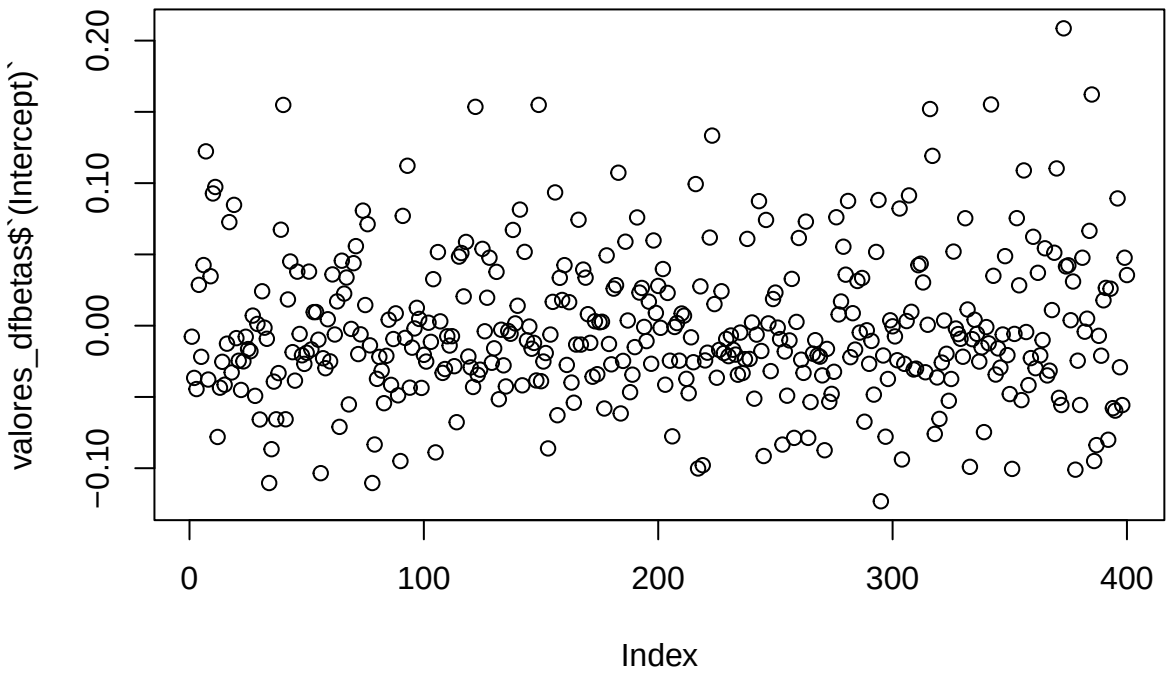
Podemos usar los valores `DFFITS` y `DFBETAS` para detectar **valores influyentes**. Se trata de medidas (similares a la distancia de Cook en RLM) para cuantificar cómo cambiaría nuestro modelo estimado si excluyéramos una observación. Concretamente, a cada observación le corresponde un `DFFITS` midiendo cuánto cambiaría la predicción para dicho punto, y varios `DFBETAS` (uno para cada predictor) midiendo cuánto cambiaría su coeficiente de regresión.

Lo recomendable es representar los valores `DFFITS` y `DFBETAS` para identificar observaciones o grupos de observaciones que se desvían de la mayoría.

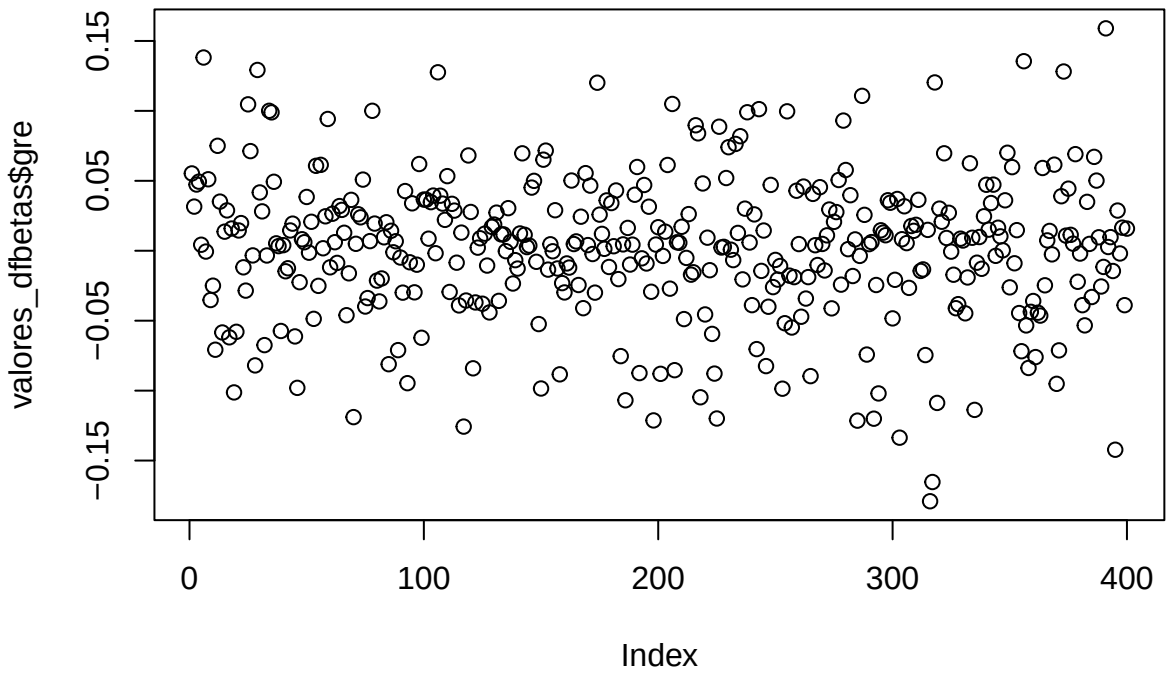
```
valores_dffits <- dffits(mylogit)
valores_dfbetas <- as.data.frame(dfbetas(mylogit))

plot(valores_dffits)
```



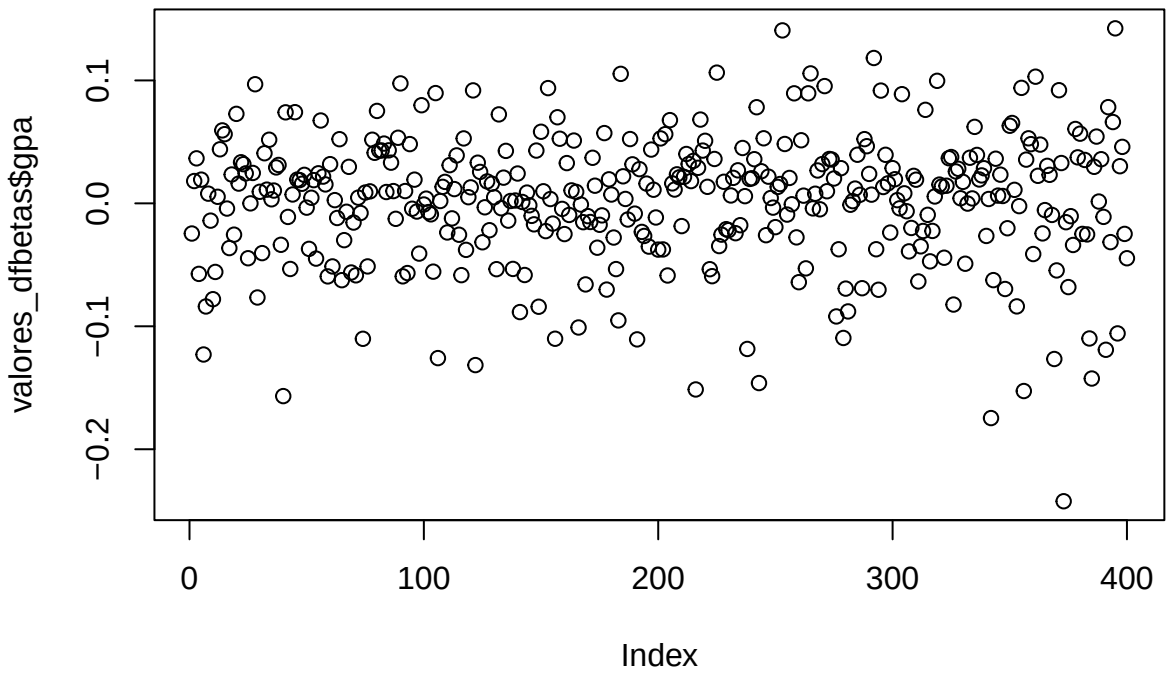

```
plot(valores_dfbetas$`(Intercept)`)
```



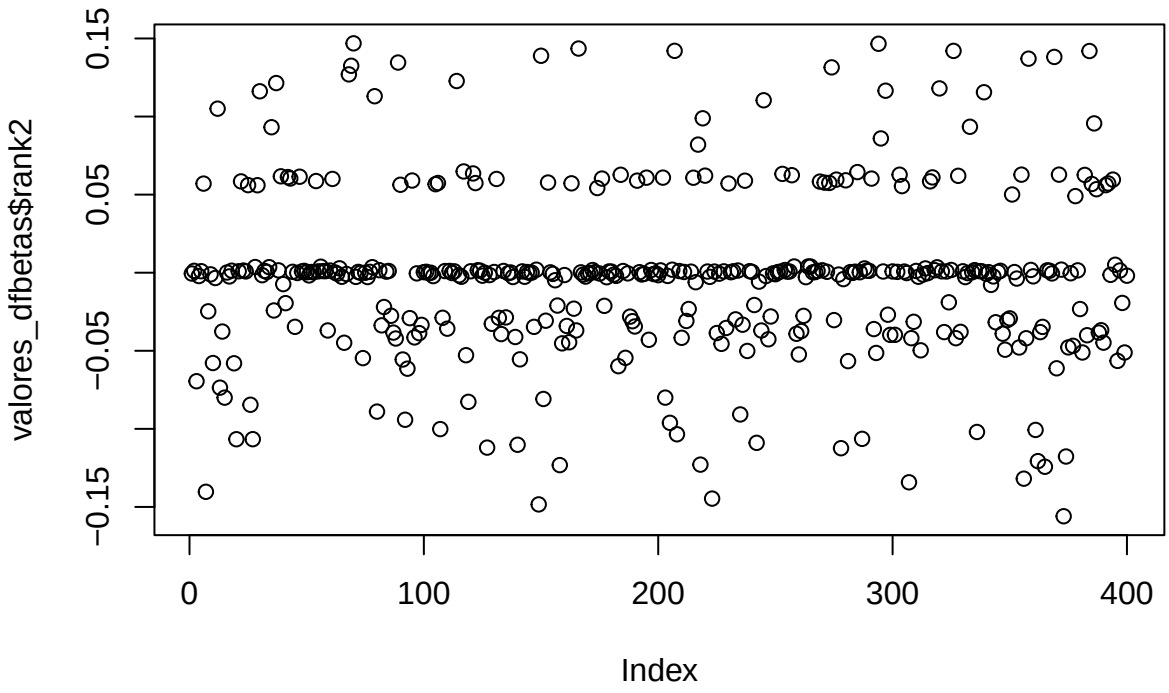

```
plot(valores_dfbetas$gre)
```



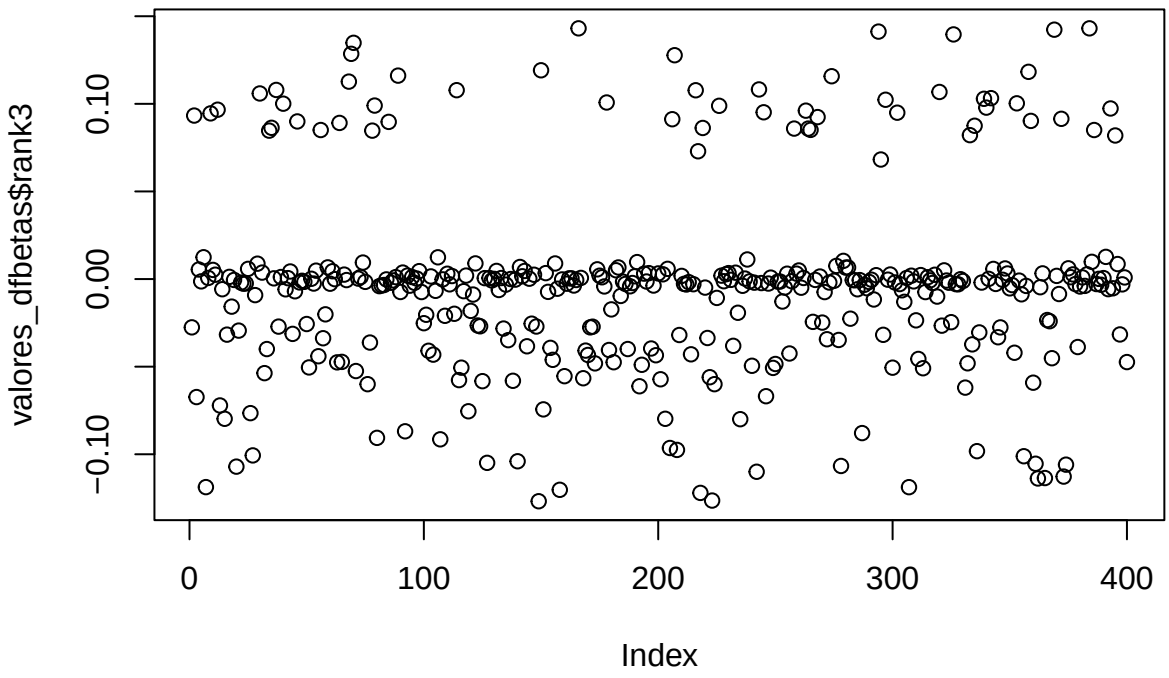

```
plot(valores_dfbetas$gpa)
```



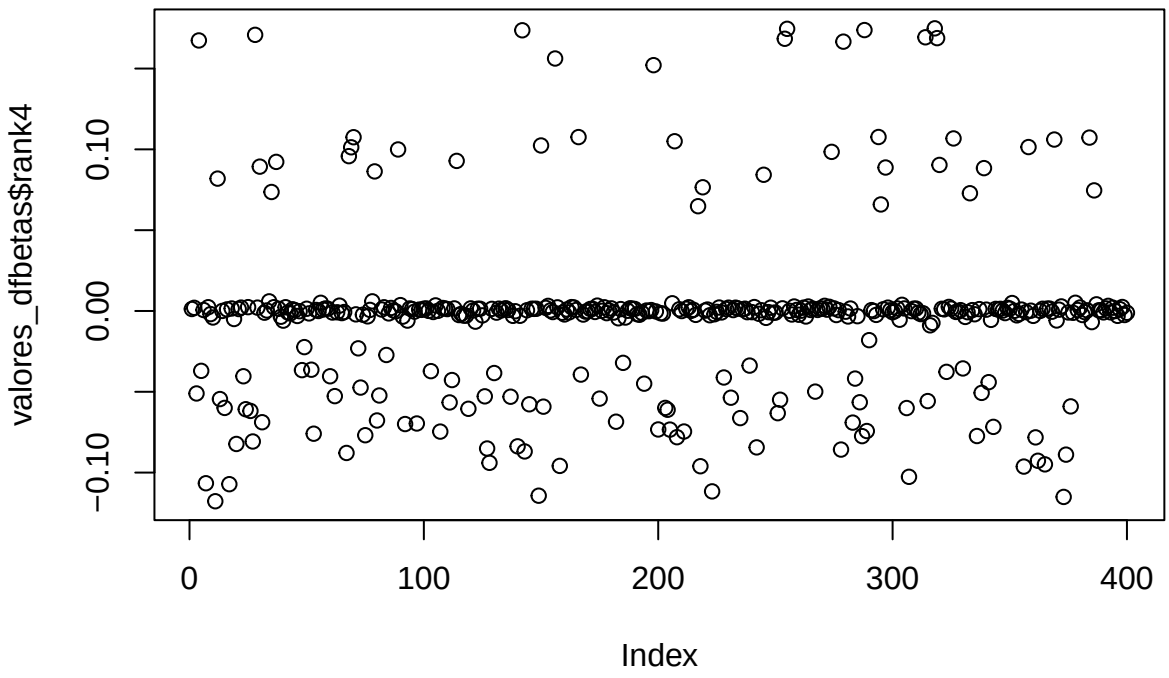

```
plot(valores_dfbetas$rank2)
```




```
plot(valores_dfbetas$rank3)
```




```
plot(valores_dfbetas$rank4)
```



Obsérvese que en los gráficos anteriores no se identifica ninguna observación especialmente alejada del resto, de manera que concluimos que no existen observaciones influyentes.

5. La Regresión Logística como un problema de clasificación

Como en RLogistica la variable respuesta es binaria, podemos entenderlo como un problema de clasificación con dos grupos: el grupo con $Y = 0$ (en nuestro caso, alumnos no admitidos en la universidad) y el grupo con $Y = 1$ (en nuestro caso, alumnos sí admitidos en la universidad).

Al realizar predicciones con el modelo logístico, la predicción resultante representa la probabilidad de que Y valga 1 para dicha observación. Podemos usar como criterio razonable que si dicha probabilidad es de al menos 0.5, entonces clasificaremos la observación en el grupo $Y = 1$, mientras que si la probabilidad es menor de 0.5 clasificaremos la observación en el grupo $Y = 0$.

```
predic_grupos <- if_else(condition = mylogit$fitted.values >= 0.5,
                         true = 1, false = 0)
```

La matriz de confusión se construye comparando los valores observados de la muestra con las predicciones del modelo.

```
matriz_confusion <- table(mydata$admit, predic_grupos,
                          dnn = c("observado", "predicciones"))
matriz_confusion
```

```
##          predicciones
## observado   0   1
##         0 254  19
##         1  97  30
```

Si el grupo $Y = 1$ es el grupo de interés (en nuestro caso alumnos admitidos), se pueden definir:

VP = verdaderos positivos = número de individuos del grupo $Y = 1$ clasificados correctamente.

FN = falsos negativos = número de individuos del grupo $Y = 1$ clasificados erróneamente.

VN = verdaderos negativos = número de individuos del grupo $Y = 0$ clasificados correctamente.

FP = falsos positivos = número de individuos del grupo $Y = 0$ clasificados erróneamente.

Y relacionados con estos valores se pueden calcular las siguientes medidas:

$$Sensibilidad = \frac{VP}{VP + FN}$$
$$Especificidad = \frac{VN}{VN + FP}$$
$$Accuracy = \frac{VP + VN}{VP + FP + VN + FN}$$

En nuestro caso, fijado el umbral en 0.5, podemos obtener dichos valores a partir de la matriz de confusión.

```
VP <- matriz_confusion[2, 2]
FN <- matriz_confusion[2, 1]
VN <- matriz_confusion[1, 1]
FP <- matriz_confusion[1, 2]

sensibilidad <- VP/(VP+FN)
sensibilidad
```

```
## [1] 0.2362205
```

```
especificidad <- VN/(VN+FP)
especificidad
```

```
## [1] 0.9304029
```

```
accuracy <- (VP+VN)/(VP+FP+VN+FN)
accuracy
```

```
## [1] 0.71
```

```
# Otra forma de calcular el accuracy es dividiendo aciertos entre total de datos
sum(diag(matriz_confusion)) / sum(matriz_confusion)
```

```
## [1] 0.71
```

Si queremos detectar más individuos del grupo $Y = 1$ (aumentar la sensibilidad), debemos bajar el umbral. Por ejemplo, fijado el umbral en 0.3, recalculamos los valores anteriores.

```
predic_grupos2 <- if_else(condition = mylogit$fitted.values >= 0.3,
                          true = 1, false = 0)
matriz_confusion2 <- table(mydata$admit, predic_grupos2,
                           dnn = c("observado", "predicciones"))
matriz_confusion2
```

```
##          predicciones
## observado   0   1
##         0 161 112
##         1  42  85
```

Obsérvese que en este caso 112 predicciones de admisión serían erróneas frente a 85 correctas.

```
VP2 <- matriz_confusion2[2, 2]
FN2 <- matriz_confusion2[2, 1]
VN2 <- matriz_confusion2[1, 1]
FP2 <- matriz_confusion2[1, 2]

sensibilidad2 <- VP2/(VP2+FN2)
sensibilidad2
```

```
## [1] 0.6692913
```

```
especificidad2 <- VN2/(VN2+FP2)
especificidad2
```

```
## [1] 0.5897436
```

```
accuracy2 <- (VP2+VN2)/(VP2+FP2+VN2+FN2)
accuracy2
```

```
## [1] 0.615
```

La curva `ROC` representa la sensibilidad (eje y) frente a 1-especificidad (eje x), es decir, representa la proporción de verdaderos positivos frente a la proporción de falsas alarmas. La curva `ROC` se obtiene modificando el umbral del método clasificador, de manera que cambian los valores de sensibilidad y especificidad.

También se puede calcular el área bajo la curva, la cual se denota por `AUC` (*area under the curve*). El área `AUC` sirve para cuantificar la eficiencia del método de clasifición. Si el `AUC` supera 0.5 (el 50%), el método de clasificación representa mejora respecto al azar.

```
coordenada_ROC <- c(1 - especificidad, sensibilidad)
coordenada_ROC
```

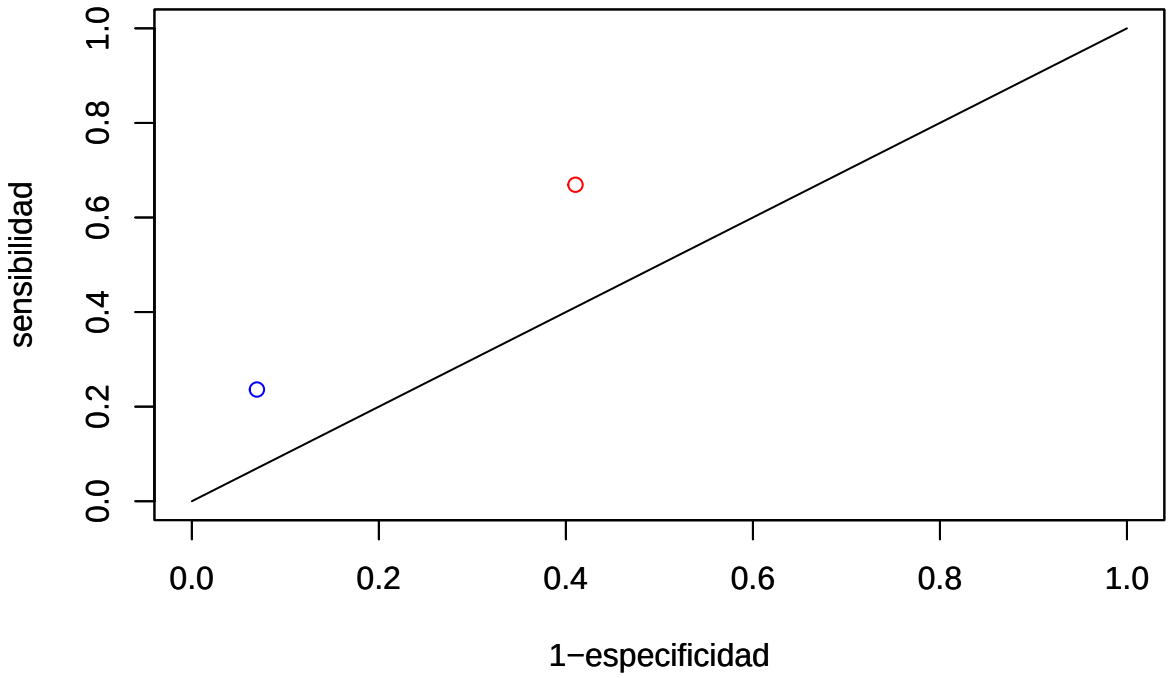
```
## [1] 0.06959707 0.23622047
```

```
coordenada_ROC2 <- c(1 - especificidad2, sensibilidad2)
coordenada_ROC2
```

```
## [1] 0.4102564 0.6692913
```

Representamos estos dos puntos de la curva ROC en el plano:

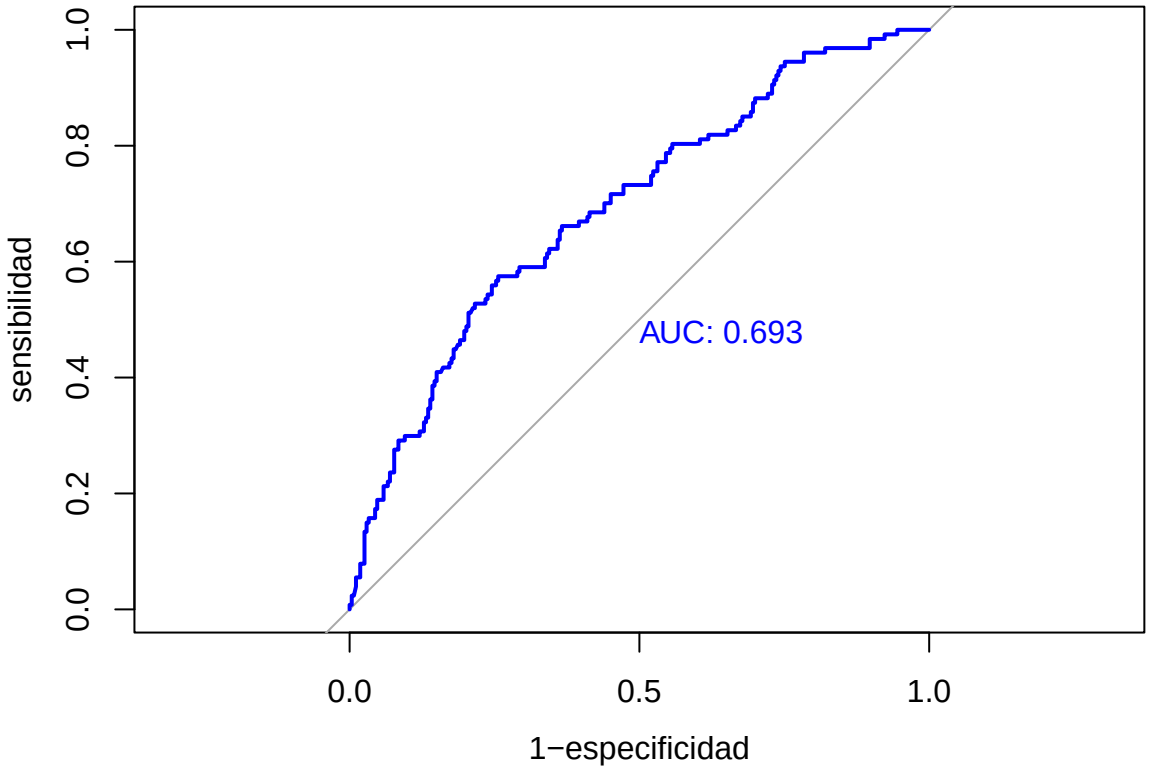
```
plot(coordenada_ROC[1], coordenada_ROC[2], xlim = c(0, 1), ylim = c(0, 1),
     col = "blue", xlab = "1-especificidad", ylab = "sensibilidad")
par(new = TRUE)
plot(coordenada_ROC2[1], coordenada_ROC2[2], xlim = c(0, 1), ylim = c(0, 1),
     col = "red", xlab = "1-especificidad", ylab = "sensibilidad")
curve(1*x, 0, 1, add = TRUE)
```



El punto azul de la gráfica anterior representa la coordenada de la curva ROC para el umbral o punto de corte en la probabilidad igual a 0.50, mientras que el punto rojo representa la coordenada de la curva ROC para el umbral o punto de corte en la probabilidad igual a 0.30. En ambos casos los puntos están por encima de la recta (bisecctriz del primer cuadrante), lo que indica que en ambos casos un criterio clasificador mejor que el azar.

Para obtener la curva `ROC` completa, usaremos la función *roc()* del paquete `pROC`.

```
library("pROC")
roc(mydata$admit, mylogit$fitted.values, plot = TRUE,
    legacy.axes = TRUE, percent = FALSE,
    xlab = "1-especificidad", ylab = "sensibilidad",
    col = "blue", lwd = 2, print.auc = TRUE)
```




```
##
## Call:
## roc.default(response = mydata$admit, predictor = mylogit$fitted.values,     percent = FALSE, plot = T
##
## Data: mylogit$fitted.values in 273 controls (mydata$admit 0) < 127 cases (mydata$admit 1).
## Area under the curve: 0.6928
```

Obsérvese que en este caso obtenemos un `AUC` cercano a 0.7, indicativo de que el método clasificador (en este caso la Regresión Logística) tiene un rendimiento de aproximadamente el 70% y por tanto es mejor que el azar. Un clasificador perfecto daría un valor de `AUC` igual a 1.

IMPORTANTE: En este ejemplo no hemos dividido el conjunto de datos en `entrenamiento` y `test`. Por tanto, se puede dar sobreajuste (*overfitting*) al calcular la matriz de confusión y las medidas de bondad del ajuste puesto que las propias observaciones se han usado para calibrar el modelo y realizar predicciones.

6. Consideraciones adicionales

En ocasiones, necesitamos ajustar modelos con alguna o varias variables predictoras elevadas a una potencia, así como considerar interacciones entre las variables predictoras.

A continuación se muestra un ejemplo de este tipo de ajustes:

```
mylogit_nolineal <- glm(admit ~ gre + gpa + rank + I(gre^2) + I(gpa^2) + I(gre*gpa),
                        data = mydata, family = "binomial")
summary(mylogit_nolineal)
```

```
##
## Call:
```

```
## glm(formula = admit ~ gre + gpa + rank + I(gre^2) + I(gpa^2) +
##     I(gre * gpa), family = "binomial", data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5502  -0.8754  -0.6297   1.1187   2.1888
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -7.325e+00  9.065e+00  -0.808 0.419012
## gre           1.860e-02  1.184e-02   1.571 0.116136
## gpa          -1.777e-01  4.952e+00  -0.036 0.971371
## rank2        -7.130e-01  3.202e-01  -2.227 0.025958 *
## rank3        -1.341e+00  3.474e-01  -3.861 0.000113 ***
## rank4        -1.595e+00  4.221e-01  -3.780 0.000157 ***
## I(gre^2)      3.070e-06  8.216e-06   0.374 0.708624
## I(gpa^2)      6.699e-01  7.625e-01   0.878 0.379690
## I(gre * gpa) -5.888e-03  3.196e-03  -1.842 0.065475 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 454.90  on 391  degrees of freedom
## AIC: 472.9
##
## Number of Fisher Scoring iterations: 4
```

Obsérvese que en caso anterior, el modelo resultante es reducible al haber p-valores muy superiores a 0.10.

Por último, comentar que la Regresión Logística también puede realizarse con la función *lrm()* del paquete `rms`.

```
library("rms")
mylogit_new <- lrm(admit ~ gre + gpa + rank, data = mydata)
summary(mylogit)
```

```
##
## Call:
## glm(formula = admit ~ gre + gpa + rank, family = "binomial",
##     data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```