Profesor: Héctor Bahamonde, PhD.

e:hector.bahamonde@uoh.cl

w:www.hectorbahamonde.com

Curso: MLE.

TA: Gonzalo Barría.

I. Diagnósticos

Al igual que en el "mundo" OLS, existen formas para evaluar cuán bueno (o malo) es nuestro modelo. Qué tipo de diagnósticos existen en el "mundo" OLS?

Recuerda que en OLS, el residuo ϵ_i es la diferencia entre lo que predecimos y lo que observamos, o $\epsilon_i = y_i - x_{ij}\beta_j$ (donde y_i son los valores de la variable dependiente para observación i, x_{ij} son los j variables dependientes para cada una de las observaciones i, y β_j son los j parámetros estimados).

Si recuerdas bien, $E(\epsilon_i) = 0$ y homoesquedástico (varianza constante). En MLE, es bastante similar, "pero ni tanto":

- En vez de un β_j que se multiplica por cada x_{ij}, hablamos de la probabilidad π_i de que observemos la realización del evento en el sujeto i. O más formalmente, $\pi_i = E(y_i|\mathbf{x}_i) = Pr(y_i = 1|\mathbf{x}_i)$. Nota que $\mathbf{x}_i$ es una matriz (por qué?).
- Ademas, si quisieras comparar diagnósticos entre distintos modelos y con distintas bases de datos, **no puedes**. Esto es una limitante de MLE: al estar diagnosticando modelos (con las herramientas que veremos hoy), **sólo podrás hacerlo entre modelos que se hayan estimado con la *misma* base de datos**.

II. Análisis de Residuos

El primer enfoque para diagnosticar un modelo GLM es mirando sus residuos.

Residuos Pearson Debido a que y_i es una variable bimodal, la distribución de $Pr(y_i = 1|\mathbf{x}_i)$ es sigmoidal. En consecuencia, las desviaciones $y_i - \pi_i$ son heteroesquedásticos (no constantes). Por estas razones, en MLE no trabajamos con residuos del tipo $e_i = y_i - \hat{y}_i$, sino que con un tipo de residuos estandarizados llamados "Residuos Pearson" r_i:

$$r_i = \frac{y_i - \hat{\pi}_i}{\sqrt{\hat{\pi}_i(1-\hat{\pi}_i)}} \tag{1}$$

donde $\sqrt{\hat{\pi}_i(1-\hat{\pi}_i)}$ es la varianza. Cuando r_i es grande, eso indica que existe un mal "fit" (o "ajuste", i.e. nuestra línea de regresión pasa lejos de las observaciones. Si te fijas, cada observación i tiene su contribución al error total del modelo: hay un r_i para cada observación i. Veamos ahora un "*index plot*" donde graficamos todos los r_i de manera ordenada (o "por índice": el primer r_1, despues el segundo r_2, etc.).

Carguemos los datos.

```
dat <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
head(dat)

##   admit gre  gpa rank
## 1     0 380 3.61    3
## 2     1 660 3.67    3
## 3     1 800 4.00    1
## 4     1 640 3.19    4
## 5     0 520 2.93    4
## 6     1 760 3.00    2

summary(dat)

##      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
```

Ahora estimemos el modelo:

```
logit.1 <- glm(admit ~ gre + gpa, data = dat, family = binomial(link = "logit"))
summary(logit.1)

##
## Call:
## glm(formula = admit ~ gre + gpa, family = binomial(link = "logit"),
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2730  -0.8988  -0.7206   1.3013   2.0620
##
## Coefficients:
##              Estimate Std. Error z value   Pr(>|z|)
## (Intercept) -4.949378   1.075093  -4.604 0.00000415 ***
## gre          0.002691   0.001057   2.544     0.0109 *
## gpa          0.754687   0.319586   2.361     0.0182 *
## ---
## 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: 480.34  on 397  degrees of freedom
## AIC: 486.34
##
## Number of Fisher Scoring iterations: 4
```

Y usando la función `rstandard` calcularemos r_i

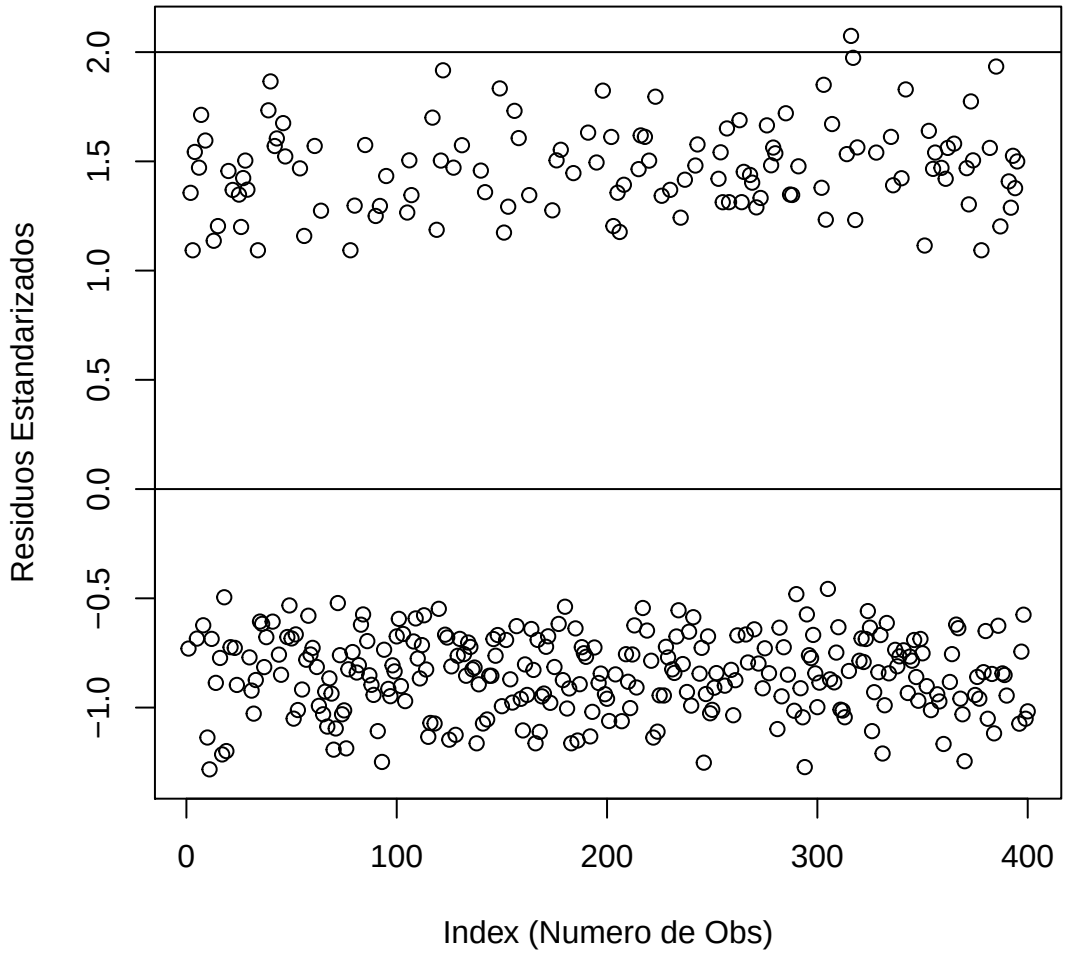
```
r = rstandard(logit.1)
head(r)

##          1          2          3          4          5          6
## -0.7300511  1.3559165  1.0930693  1.5430303 -0.6844243  1.4711335
```

Como ves, cada observación i tiene su propio error (la distancia dada por $y_i - \hat{\pi}_i$).

Hagamos el "index plot":

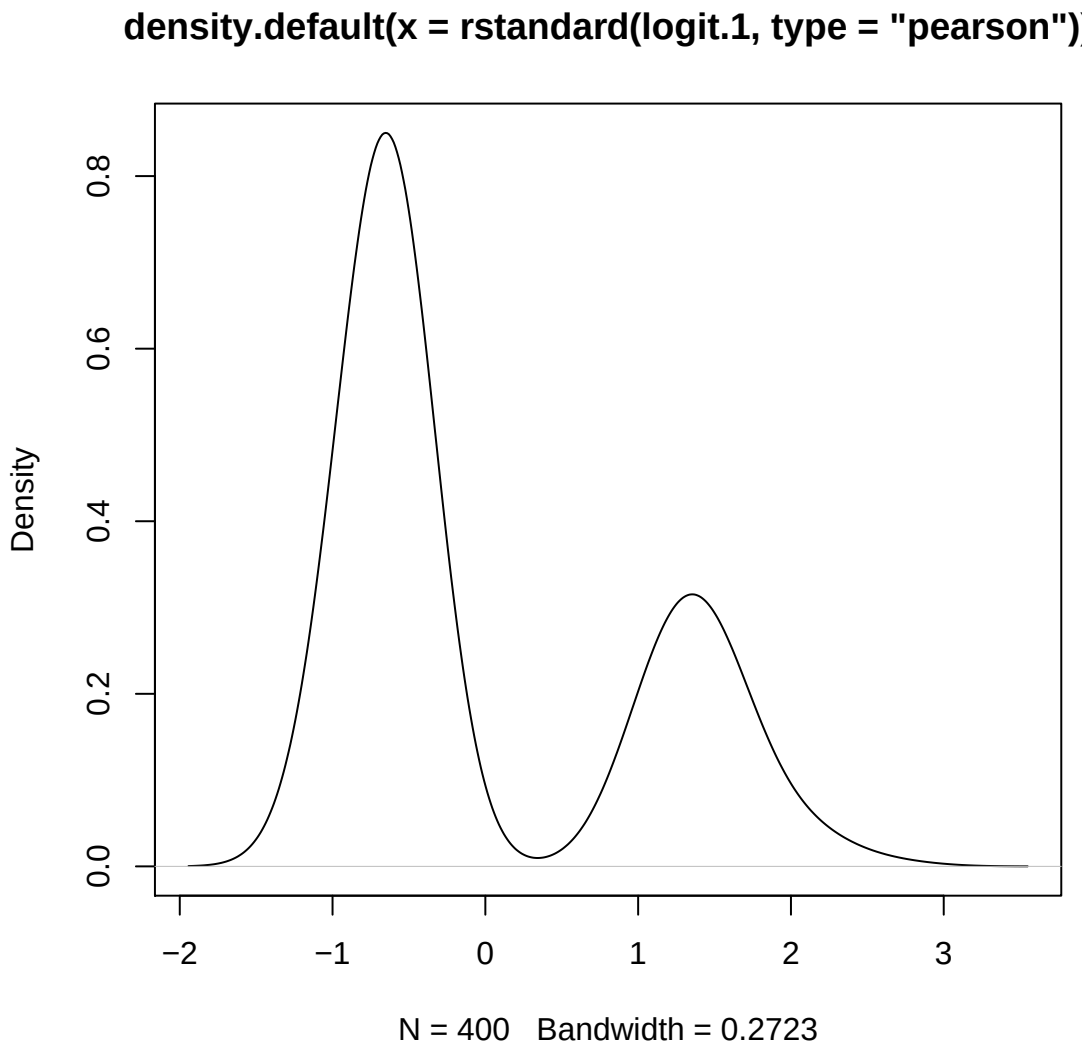
```
plot(1:nrow(dat), # Numero de Obs
     r, # y
     ylab="Residuos Estandarizados",
     xlab="Index (Numero de Obs)")
abline(0, 0)
abline(2, 0)
abline(-2, 0)
```



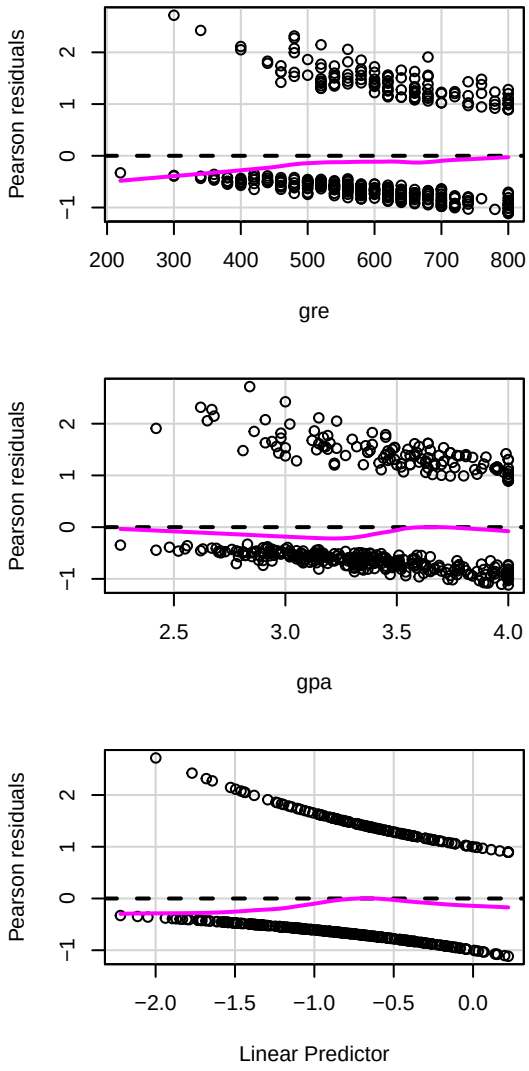
En la figura vemos que hay un *outlier* (fuera de las dos desviaciones estándar).

Veamos ahora cómo se ven los residuos Pearson r_i:

```
plot(density(rstandard(logit.1, type='pearson')))
```




```
p_load(car)
residualPlots(logit.1, layout=c(3,1))
```




```
##     Test stat Pr(>|Test stat|)
## gre    0.1197           0.7293
## gpa    0.1613           0.6880
```

Cook's Distance Una de las ventajas del análisis de residuos en GLM via MLE, es que podemos extraer casi todo lo que hemos aprendido en OLS. Podemos aproximar el *Cook's Distance* (D_i) para

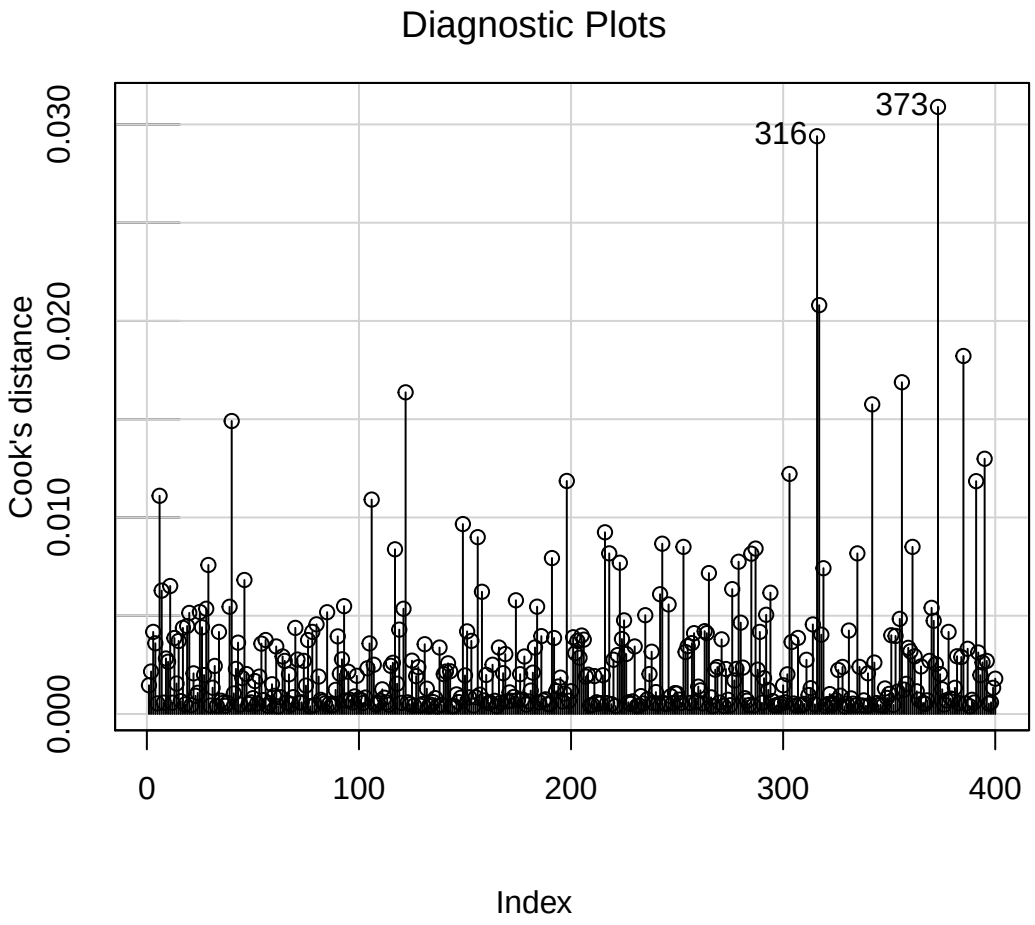
GLMs de la siguiente manera:

$$D_i = (\frac{r_i}{1-\mathcal{H}})^2 \times \frac{\mathcal{H}}{\sigma \times K} \tag{2}$$

donde r_i está definido en Equation 1, $\mathcal{H}$ es una "hat matrix" (función que mapea $y_i \to \hat{y}_i$), K el número de parámetros, y σ es la dispersión del modelo (varianza). En los modelos logísticos y Poisson $\sigma = 1$.

Grafiquemos D_i:

```
p_load(car)
influenceIndexPlot(logit.1, vars=c("Cook"))
```



DFFIT Otra manera de pensar la influencia que tiene cada observación es haciendo una regresión secuencial en la que vamos sacando esa observación, una a la vez. Define *DFFIT* como,

$$DFFIT_i = \hat{y}_i - \hat{y}_{i_{(i)}} \tag{3}$$

donde *DFFIT* es el cambio en el valor predicho para la observación i (o "*fit*") cuando i es dejada

afuera de la regresión. La versión más usada del *DFFIT* es la versión estandarizada (es decir, que podemos usar para comparar) llamada *DFFITS* (con una "s" al final de "standarized"). La única diferencia es que ésta está dividida por por el error estándar que aporta la observación SE_i (y multiplicado por la raíz cuadrada del "*leverage*" o influencia de la observación i), $\sqrt{h_i}$[1]. En otras palabras,

$$DFFITS_i = \frac{DFFIT}{SE_i\sqrt{h_i}} \tag{5}$$

Calculemos el vector de DFFITS, y veamos los primeros diez valores.

```
dffits = dffits(logit.1) # calcula el vector de dffits
as.numeric(dffits)[1:10] # ve los primeros 10.

##  [1] -0.07958515  0.08100147  0.12336778  0.09660158 -0.04878515  0.17588014
##  [7]  0.11753107 -0.05089782  0.08350179 -0.09699958
```

Los *DFFITS* tienen valores críticos (*critical values*, o "cv"). Si la observación supera ese valor crítico, es considerada "*influential*". Los valores críticos están dados por el siguiente cálculo,

$$\text{cv}_{\text{DFFITS}} = 2 \times \sqrt{\frac{k}{n}} \tag{6}$$

donde k es el número de paramétros, y n el número de observaciones. Calculemos $\text{cv}_{\text{DFFITS}}$ en R:

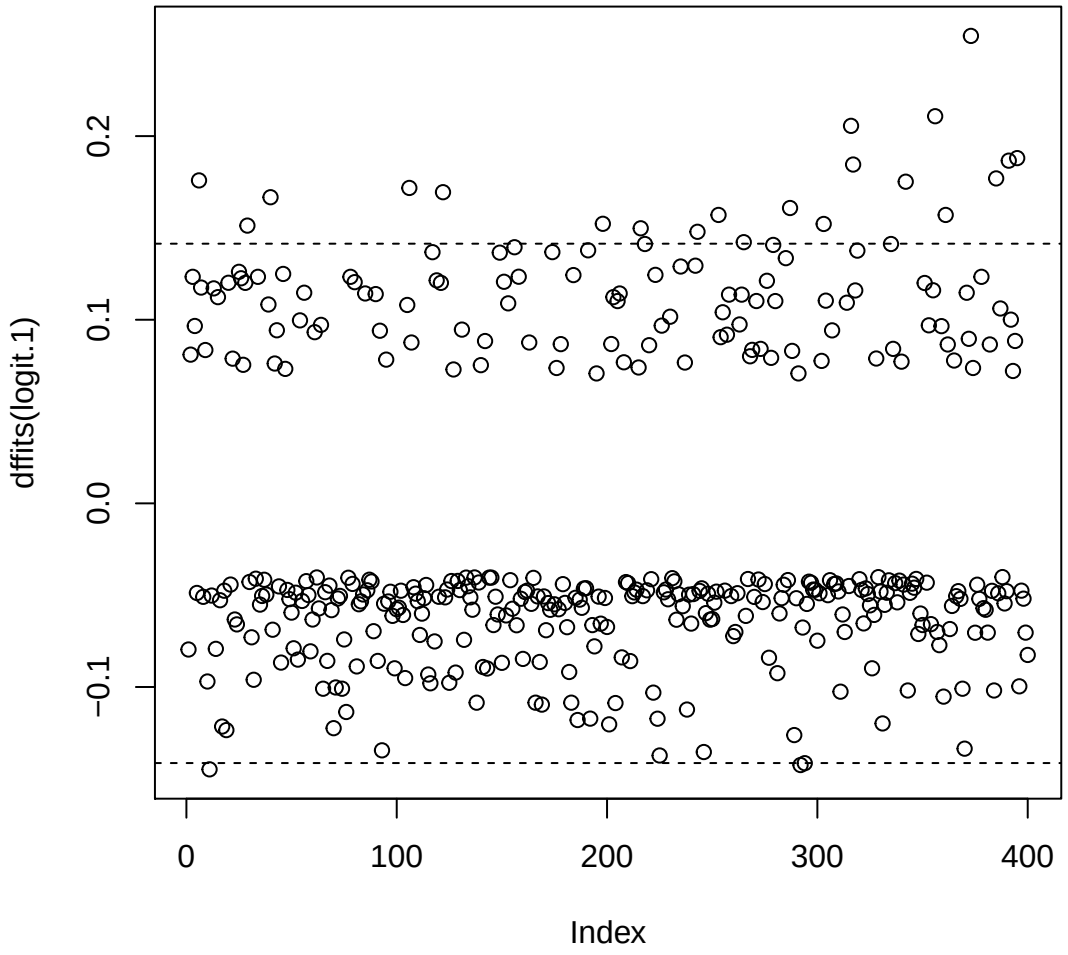
```
n = nrow(dat)
k = length(logit.1$coefficients)-1
cv = 2*sqrt(k/n)
```

Ahora grafiquemos la influencia de cada i, pero siempre tomando en cuenta los $\text{cv}_{\text{DFFITS}}$.

```
plot(dffits(logit.1)) # grafico de dffits
abline(h = cv, lty = 2) # anadir cv hacia arriba
abline(h = -cv, lty = 2) # anadir cv hacia abajo
```

[1] El "leverage" de i está dado por:

$$x(x^T x) - 1x^T \tag{4}$$



Como puedes ver existen varias observacioens "influyentes". Vayamos a `R` y tratemos de identificar cuales son esas observaciones influyentes usando la siguiente línea:

```
identify(row.names(dat), dffits(logit.1), row.names(dat), plot=TRUE)
```

DFBETA Una manera homóloga es ver cuánto cambian los parámetros estimados $\hat{\beta}_k$ al excluir cada i a la vez. Estos son los $DFBETA$ y están dados por el siguiente cálculo,

$$DFBETA_i = \boldsymbol{\beta} - \boldsymbol{\beta}_{(-i)} \tag{7}$$

Donde $\boldsymbol{\beta}$ son todos los parámetros estimados y $\boldsymbol{\beta}_{(-i)}$ son todos los parámetros estimados quitando la observación i.

De manera analoga, existe un correlato estandarizado $DFBETAS$ dado por,

$$DFBETAS_i = \frac{DFBETA}{\sqrt{\hat{\sigma^2}_i (x^T x)^{-1}}} \tag{8}$$

Esta estandarización nos permite establecer un parámetro de comparación: toda observación i que tenga un $DFBETAS_i$ mayor a 1 es considerada "sospechosa" de tener influencia.

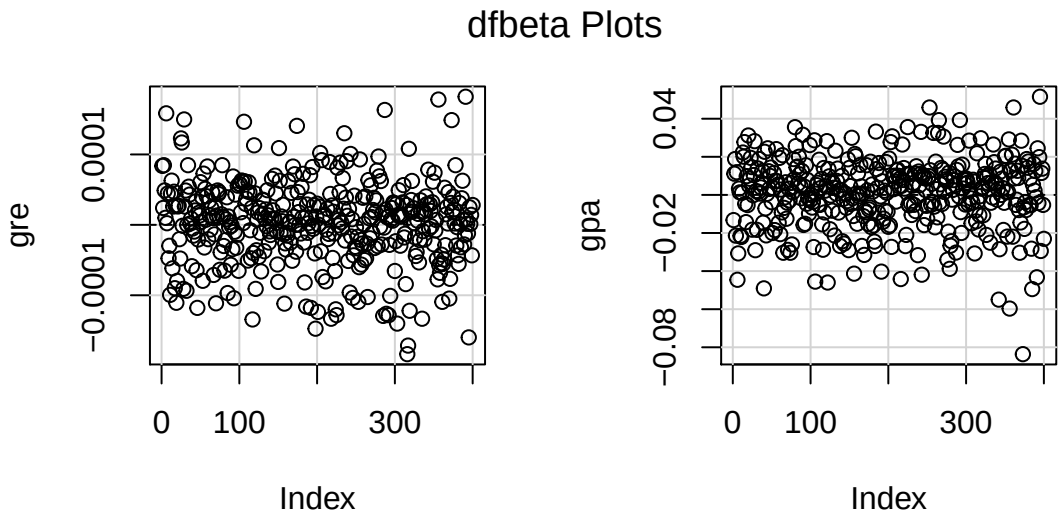
Extraigamos los $DFBETAS_i$ para cada uno de los k parametros (en nuestro caso, 3). Por simplicidad veamos los primeros.

```
dfbetas = dfbetas(logit.1)
head(dfbetas)

##    (Intercept)         gre         gpa
## 1 -0.007503563 0.071946453 -0.03756627
## 2 -0.037877366 0.020634219  0.03167665
## 3 -0.092543108 0.072844646  0.05354764
## 4  0.044668602 0.042084994 -0.06093219
## 5 -0.042668687 0.008568481  0.03391040
## 6  0.056065572 0.136316436 -0.12687735
```

Ahora usemos la función `dfbetaPlots` para graficar con un sólo comando los dos graficos que necesitamos (`gre` y `gpa`),

```
p_load(car)
dfbetaPlots(logit.1)
```



En este caso no vemos grandes alejamientos del parámetro.

III. Goodness of Fit

Otra de las preguntas que siempre debemos hacer es cuán bien (o mal) nuestro modelo ($\boldsymbol{x}_i\boldsymbol{\beta}_K$) se ajusta a la variable dependiente y_i. Esto se llama *goodness of fit*.

pseudo-r^2 El primer indicador de ajuste es el pseudo$-r^2$. En MLE los coeficientes no están diseñados para minimizar varianza (sino que para maximizar (log)likelihood). Entonces, el enfoque clasico del r^2 no nos sirve aquí. Además, recuerda que si bien el r^2 podía ser comparado entre modelos estimados con distintas bases de datos, en MLE el pseudo-r^2 **sólo puede ser comparado con modelos estimados con la misma base de datos** (esto es porque en MLE todo es relativo no absoluto). Recuerda: el r^2 en OLS significa "el porcentaje de varianza explicada"—pero ve King (1986). Y el r^2 podía (en OLS) ser comparado entre distintos modelos estimados con distintas bases de datos.

En el mundo MLE existen varios indicadores que se asemejan al r^2, por eso ellos son "pseudo" r^2. El más utilizado es el de McFadden dado por,

$$r^2_{\text{McFadden}} = 1 - \frac{log(L)}{log(L_{null})} \tag{9}$$

De manera análoga, es un radio que varia entre 0 y 1. Pero su significado no está dado en términos de "varianza explicada". Si te fijas, es el radio entre el log-likelihood del modelo entero ("*unrestricted*") comparado modelo con sólo el intercepto—"*restricted*", es decir, cuando todos los x del modelo son 0).

Para calcular el L_{null}, estimemos el modelo,

```
logit.null <- glm(admit ~ 1, data = dat, family = binomial(link = "logit"))
summary(logit.null)

##
## Call:
## glm(formula = admit ~ 1, family = binomial(link = "logit"), data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8741  -0.8741  -0.8741   1.5148   1.5148
##
## Coefficients:
##             Estimate Std. Error z value         Pr(>|z|)
## (Intercept)  -0.7653     0.1074  -7.125 0.00000000000104 ***
## ---
## 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: 499.98  on 399  degrees of freedom
## AIC: 501.98
##
## Number of Fisher Scoring iterations: 4
```

Ahora, capturemos ambos log-likelihoods,

```
logLik(logit.1)

## 'log Lik.' -240.172 (df=3)

logLik(logit.null)

## 'log Lik.' -249.9883 (df=1)
```

y computemos r^2_{McFadden},

```
1-(logLik(logit.1)/logLik(logit.null))

## 'log Lik.' 0.03926692 (df=3)
```

En este caso vemos que el $log(L)$ (modelo con parámetros) no alcanza a superar por tanto más al $log(L_{null})$ (modelo sin parámetros). El r^2_{McFadden} es casi 10%.

Aunque este indicador es intiutivo, conceptualmente es deficiente: **el modelo que explica toda la varianza existente es un modelo que tiene todas las variables posibles de existir**.

Information Criteria Existen mejores maneras de referirse al ajuste del modelo ("*goodness of fit*"). Hay dos tipos básicos de criterios de información.

1. BIC: Bayesian Information Criteria, dado por $k \times ln(n) - 2ln(\hat{l})$.
2. AIC: Akaike Information Criteria, dado por $2k - 2ln(\hat{l})$.

donde k es el número de parámetros, n el número de observaciones, y $\hat{l}$ el likelihood estimado. Nota que ambos son muy parecidos. **En ambos criterios, el número más pequeño sugiere un mejor modelo**.

Ahora comparemos el BIC para el `logit.1` y el `logit.null` (ambos estimados usando la misma base de datos).

```
BIC(logit.1,logit.null)

##            df      BIC
## logit.1     3 498.3184
## logit.null  1 505.9680
```

Ahora comparemos el AIC para el `logit.1` y el `logit.null` (ambos estimados usando la misma base de datos).

```
AIC(logit.1,logit.null)
```

```
##            df      AIC
## logit.1     3 486.3440
## logit.null  1 501.9765
```

```
knitr::purl('Diagnosticos.Rnw')
```

```
## Error in parse_block(g[-1], g[1], params.src, markdown_mode):  Duplicate chunk label
'setup', which has been used for the chunk:
## if (!require("pacman")) install.packages("pacman"); library(pacman)
## p_load(knitr)
## set.seed(2020)
## options(scipen=9999999)
```

```
Stangle('Diagnosticos.Rnw')
```

```
## Writing to file Diagnosticos.R
```