Profesor: Héctor Bahamonde, PhD.

e:hector.bahamonde@uoh.cl

w:www.hectorbahamonde.com

Curso: MLE.

TA: Gonzalo Barría.

I. Inferencia e Interpretación

Los GLMs no se pueden interpretar usando la tabla. Los coeficientes no significan lo que significaban en nuestro mundo OLS. Para interpretarlos, debemos usar otras herramientas.

Intervalos de confianza Ya sabemos lo que un intervalo de confianza significa: si nuestra confianza es al 95%, sabemos que el "true value" de la estimación caerá el 95% de las veces dentro del margen de los intervalos de confianza. Es en base a esto que antes hablábamos de "significancia estadística".

Carguemos los datos.

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

##   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(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
```

Estimemos el primer modelo

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

##
## Call:
## glm(formula = admit ~ gre + gpa, family = binomial(link = "logit"),
##     data = mydata)
##
## 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 ahora, estimemos los intervalos de confianza. Los intervalos de confianza son sencillos de calcular. Si queremos un 95% de confianza, miramos la tabla con los valores Z. Para ese porcentaje es 1.960 (para un 90% es 1.645).

$$\hat{\theta} \pm 1.960 \times \text{SE}_{\hat{\theta}} \tag{1}$$

donde $\hat{\theta}$ es la estimación (el parámetro), 1.960 es el valor z para el 95% y $\text{SE}_{\hat{\theta}}$ es el "standard error" del parámetro estimado $\hat{\theta}$. Si te fijas, tenemos el signo $\pm$ que implica que debemos restar para obtener el rango mínimo del intervalo de confianza, y sumar para obtener el rango máximo del intervalo de confianza asociado a $\hat{\theta}$.

Por ejemplo, el coeficiente de "gre" es $\hat{\theta}_{\text{GRE}} = 0.0026907$, el $SE_{\hat{\text{GRE}}} = 0.0010575$. Entonces, $0.0026907 + 1.96 * 0.0010575 = 0.0048$ para el intervalo superior, y $0.0026907 - 1.96 * 0.0010575 = 0.0006$ para el intervalo inferior.

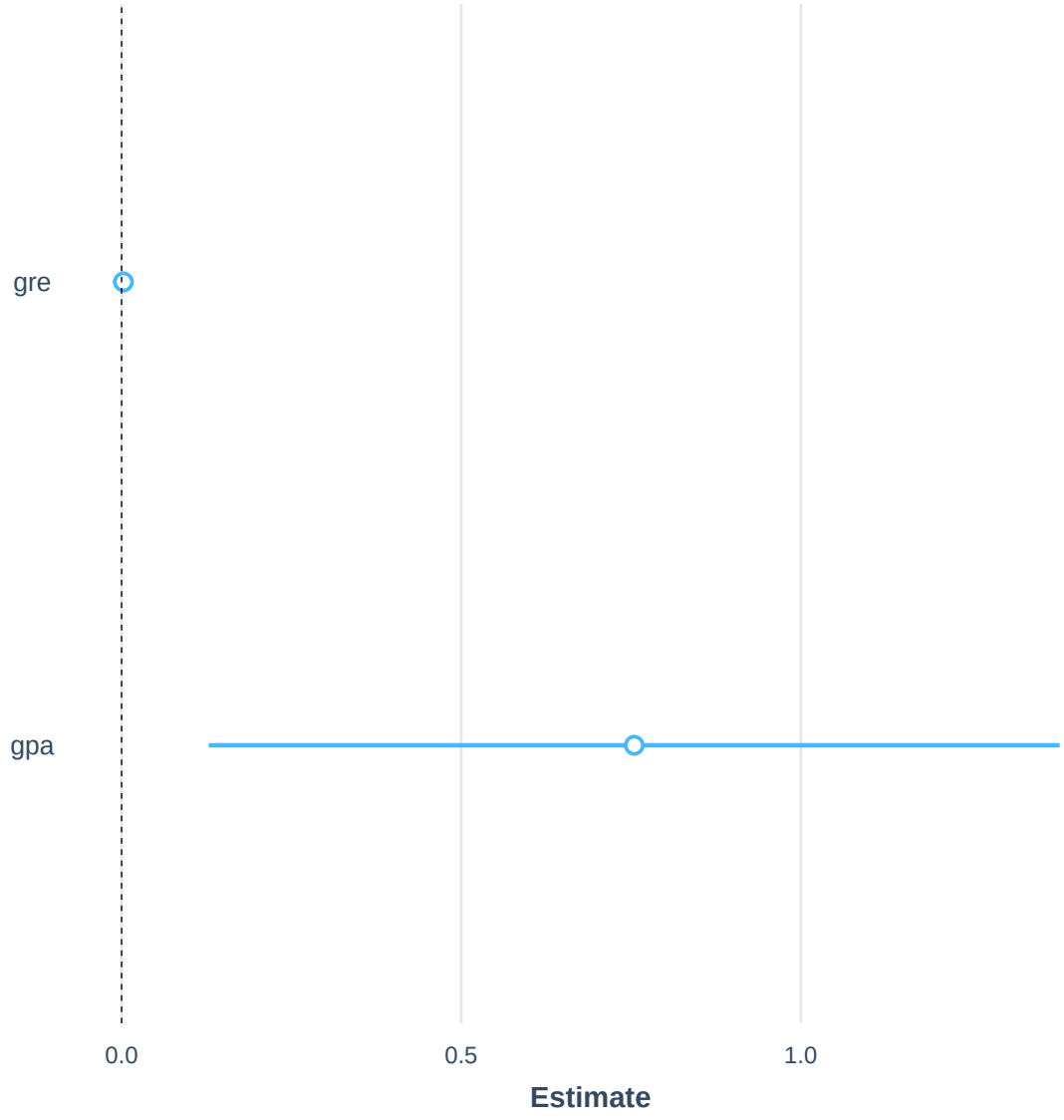
Usemos un paquete.

```
confint(logit.1)
```

```
##                     2.5 %       97.5 %
## (Intercept) -7.1092205752 -2.886726963
## gre          0.0006373958  0.004791712
## gpa          0.1347509288  1.390131131
```

También podemos graficar:

```
p_load(jtools,ggstance,broom.mixed)
plot_summs(logit.1)
```



Odds Ratios ("Radios de Posibilidad") Esta manera de interpretar solo funciona con los *link function* tipo logit (logit, multinomial logit, etc.), no probit links (i.e. probit, multinomial probit). Formalmente,

$$
\begin{aligned}
ln\Omega(\mathrm{x}) &= \mathrm{x}\beta \\
\Omega(\mathrm{x}) &= \frac{\Pr(\mathrm{y}=1|\mathrm{x})}{\Pr(\mathrm{y}=0|\mathrm{x})}
\end{aligned}
\tag{2}
$$

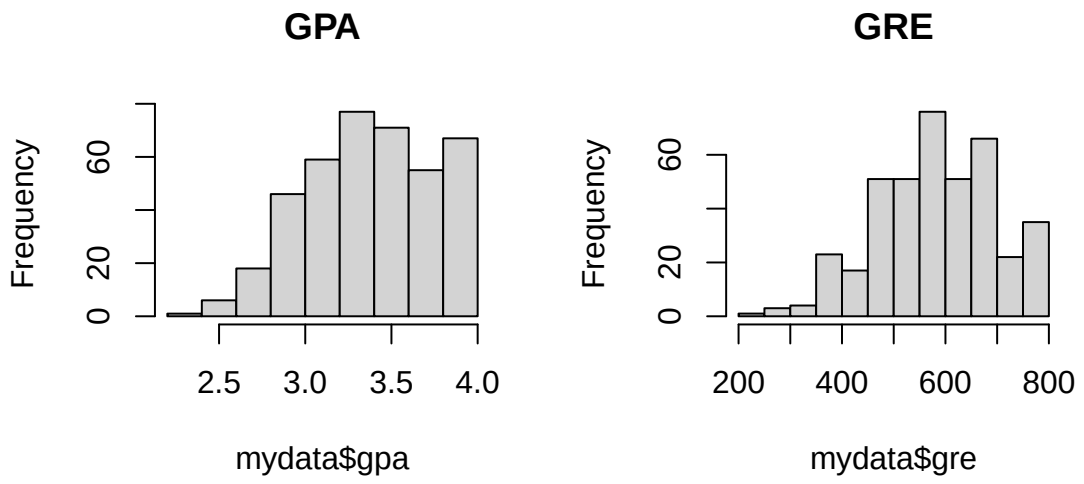
donde Equation 2 es el *odd ratio* de que y sea 1 dado x, relativo a que y sea 0 dado x. Es un ratio, una fracción. Su interpretación es intuitiva: "cuando x cambia, cuánto cambia el logit estimado ($\hat{\theta}$) manteniendo los otros parámetros constantes"?

Lo bueno de esta interpretación es que podemos observar el cambio de la variable dependiente. Ten en cuenta que los *odds ratios* son comparables sólo con la misma variable. **Lo malo** es que seguimos en unidades de la escala logit (que sigue siendo poco interpretable).

```
p_load(oddsratio)
#
odds = or_glm(data = mydata,
       model = logit.1,
       incr = list(gre = mean(mydata$gre), gpa = mean(mydata$gpa) + sd(mydata$gpa)))
odds

##   predictor oddsratio ci_low (2.5) ci_high (97.5)  increment
## 1       gre     4.861        1.454         16.711 587.700000
## 2       gpa    17.211        1.662        188.943   3.770467
```

Aquí vemos que la variable dependiente (`admit`) es 17.211 veces más "*posible*" de ocurrir cuando la variable independiente `gpa` incrementa una desviación estandard, es decir, se mueve desde su promedio (3.3899) en 0.3805668 puntos de `gpa`.

```
par(mfrow=c(1,2))
hist(mydata$gpa, main = "GPA")
hist(mydata$gre, main = "GRE")
```



Cómo interpretamos `GRE`?

```
mean(mydata$gpa)

## [1] 3.3899

mean(mydata$gre)

## [1] 587.7
```

Lo malo Si bien es cierto que ganamos interpretación del modelo, lo malo es seguimos hablando con poca precisión. Decir que algo es 17.211 veces más "*posible*" es interesante, pero no sé si tan "científico".

***Partial/Marginal Changes* in y** Los "cambios parciales" son muy parecidos a los *odds ratios*. Son parecidos porque nos muestra "cuánto cambia $\hat{y}$ cuando cambia x". Formalmente,

$$\frac{\partial \hat{y}}{\partial x_k} = \beta_k \tag{3}$$

lo que significa "por un cambio en x_k, se espera que $\hat{y}$ cambie β_k" (manteniendo todas las otras variables constantes en su media). Sin embargo, debido a que la varianza de $\hat{y}$ es desconocida

(la varianza es un *population parameter*), entonces, esto complica la interpretación de β_k. Para resolver esto, lo que hacemos es pensar en coeficientes β_k estandarizados. Siguiendo Equation 3, lo que pensamos es en "por un cambio en x_k, se espera que $\hat{y}$ cambie β_k **desviaciones estándard**". Formalmente,

$$\beta_k^S = \frac{\sigma_k \beta_k}{\sigma_{\hat{y}}} = \sigma_k \beta_k^{S_{\hat{y}}} \tag{4}$$

Nota que Equation 4 también estandariza $\hat{y}$. Debido a que $\hat{\sigma}_{\hat{y}} = \sigma_{\hat{y}}$ (i.e. podemos estimar la varianza estandarizada de los datos),

$$\hat{\sigma}_{\hat{y}} = \beta^\top \hat{\sigma}(x)\beta + \sigma(\epsilon). \tag{5}$$

Calculemos dos escenarios. Uno donde el estudiante le iba muy bien en el colegio (`max(mydata$gpa)`), pero tuvo un pésimo puntaje GRE que es la prueba de admisión de doctorado (`min(mydata$gre)`).

```
p_load(margins)
summary(margins(logit.1,
        at = list(
          gre = min(mydata$gre),
          gpa = max(mydata$gpa))
        ))
```

```
##  factor      gre    gpa    AME     SE      z      p   lower  upper
##     gpa 220.0000 4.0000 0.1242 0.0808 1.5364 0.1244 -0.0342 0.2827
##     gre 220.0000 4.0000 0.0004 0.0001 5.9659 0.0000  0.0003 0.0006
```

Ahora calculemos un escenario donde el estudiante era regular en el colegio (`mean(mydata$gpa)`) pero le fue súper bien en el GRE (`max(mydata$gre)`):

```
summary(margins(logit.1,
        at = list(
          gre = max(mydata$gre),
          gpa = mean(mydata$gpa))
          ))
```

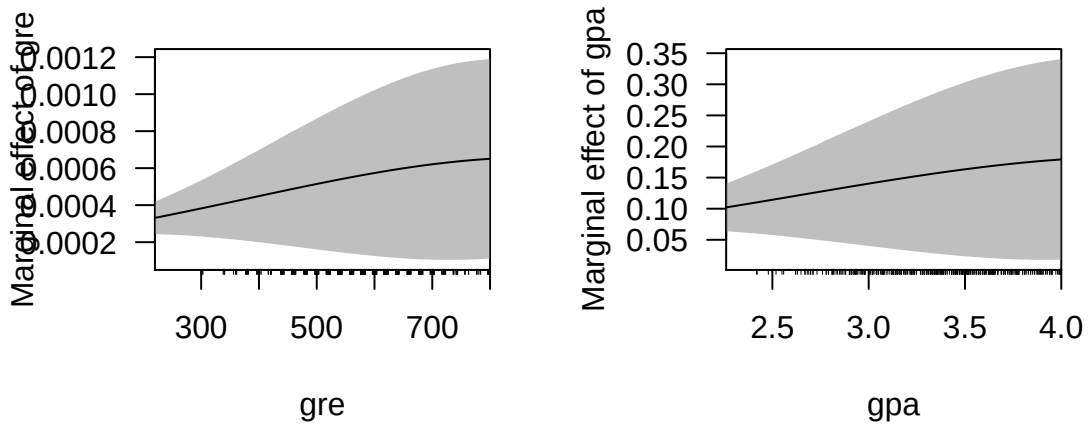
```
##  factor      gre    gpa    AME     SE      z      p  lower  upper
##     gpa 800.0000 3.3899 0.1860 0.0770 2.4157 0.0157 0.0351 0.3369
##     gre 800.0000 3.3899 0.0007 0.0003 2.3899 0.0169 0.0001 0.0012
```

Aquí vemos en `factor` cuánto cambia (en unidades de log-likelihood, segun sale en `lower` y `upper`) $\hat{y}$, y si es significativo o no (`p`).

Es muy útil establecer valores según ciertos perfiles típicos o "escenarios". Sin embargo, esto asume que conocemos estos perfiles (lo que puede que no sea cierto todas las veces). Para compensar por esto, podemos graficar los cambios marginales para todo el rango de x e y.

Grafiquemos:

```
par(mfrow=c(1,2))
cplot(logit.1, "gre", what = "effect") # effect para mostrar cambios marginales
cplot(logit.1, "gpa", what = "effect") # effect para mostrar cambios marginales
```



Predicted Probabilities: Gráficos Quizás esta sea la manera más usada para poder interpretar: graficando (o mostrando en una tabla) "la probabilidad de que y sea 1 a distintos valores de x", o de manera más formal,

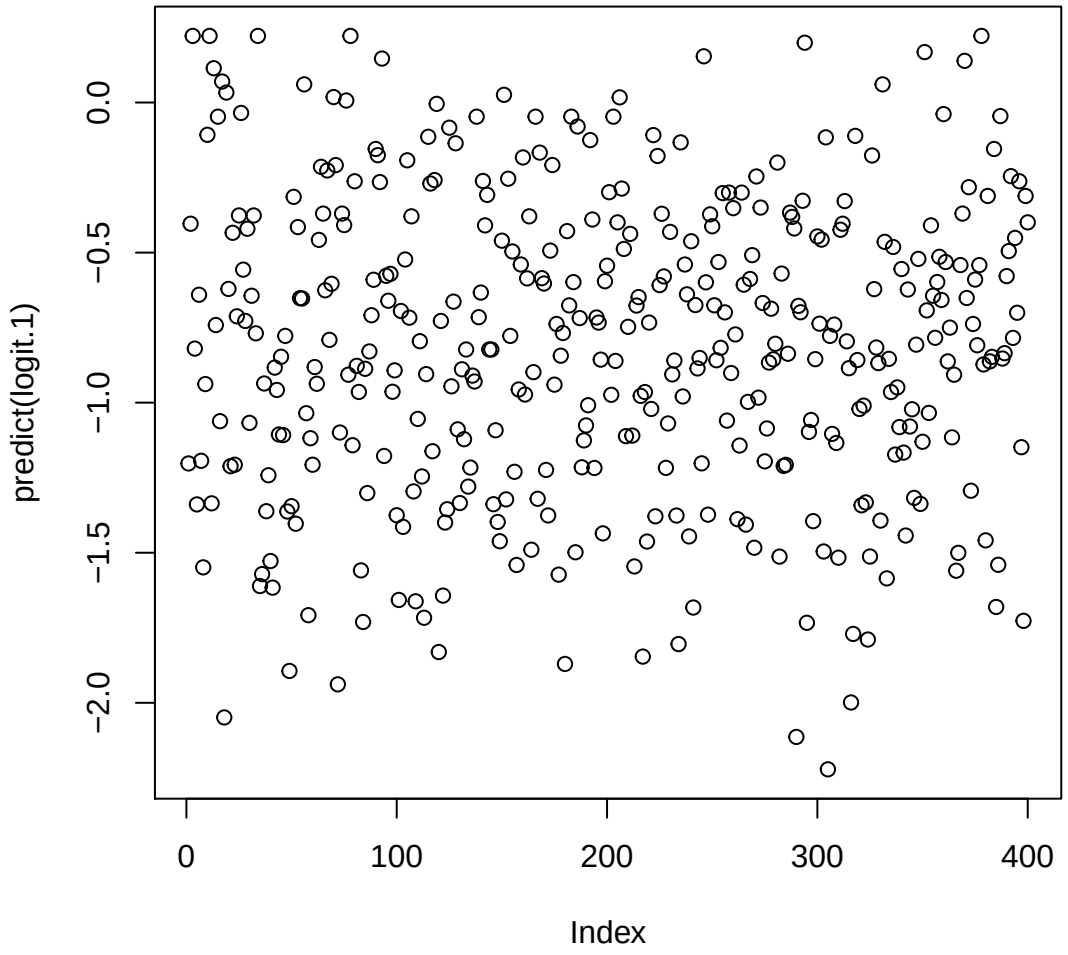
$$\hat{\Pr}(y_i = 1|\mathbf{x}_i) = f(\mathbf{x}_i\hat{\boldsymbol{\beta}}) \tag{6}$$

Fíjate que $\mathbf{x}$ y $\hat{\boldsymbol{\beta}}$ son matrices. Por que?

Aquí lo que queremos saber es cómo se comporta la probabilidad de que $y_i = 1$ a medida que nos movemos en una de las x's. **Lo bueno**, es que tenemos un lenguaje fácil de entender: todos entienden probabilidades, y además, estas varían entre 0 y 1. Pero qué hacemos con el resto de las x's?

Lo que estamos tratando de comunicar es $\hat{y}_i$ que está en escala logit. Veámos cómo se ve $\hat{y}_i$ en escala logit usando el comando `predict`:

```
head(predict(logit.1))

##          1          2          3          4          5          6
## -1.2024987 -0.4038261  0.2219162 -0.8198895 -1.3389901 -0.6403980

summary(predict(logit.1))

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.2217 -1.1517 -0.7954 -0.8098 -0.4333  0.2219

plot(predict(logit.1)) # logit scale
```



Como la escala logit no es facilmente entendible (`predict(logit.1)`), debemos mapear $\hat{y}_i$ en una función que que retorne $\hat{y}_i$ pero en escala de probabilidades (entre 0 y 1).

Asumiendo que $\hat{y}_i$ está en escala logit, la función es la siguiente:

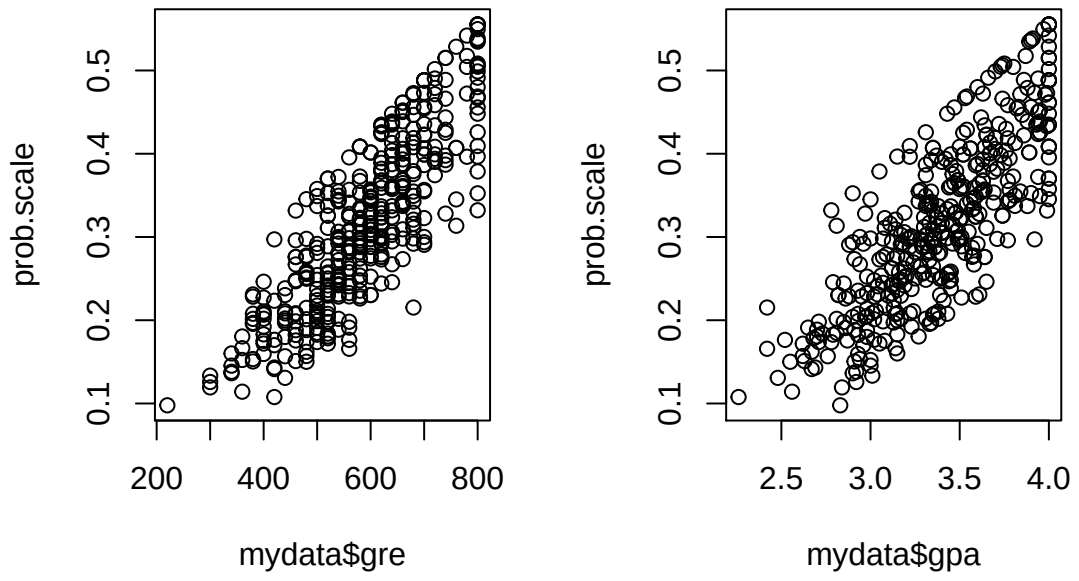
$$Pr(y_i = 1) = \frac{exp(\hat{y}_i)}{1 + exp(\hat{y}_i} \tag{7}$$

y que se logra así:

```
logit.scale = as.numeric(predict(logit.1)) # extraer vector y'
prob.scale = exp(logit.scale)/(1+exp(logit.scale)) # exponenciar
head(prob.scale) # ve como se ve y' en formato prob

## [1] 0.2310310 0.4003934 0.5552525 0.3057871 0.2076762 0.3451566

summary(prob.scale) # resumen de y' en formato prob (bounded)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.09782 0.24018 0.31100 0.31750 0.39334 0.55525
```

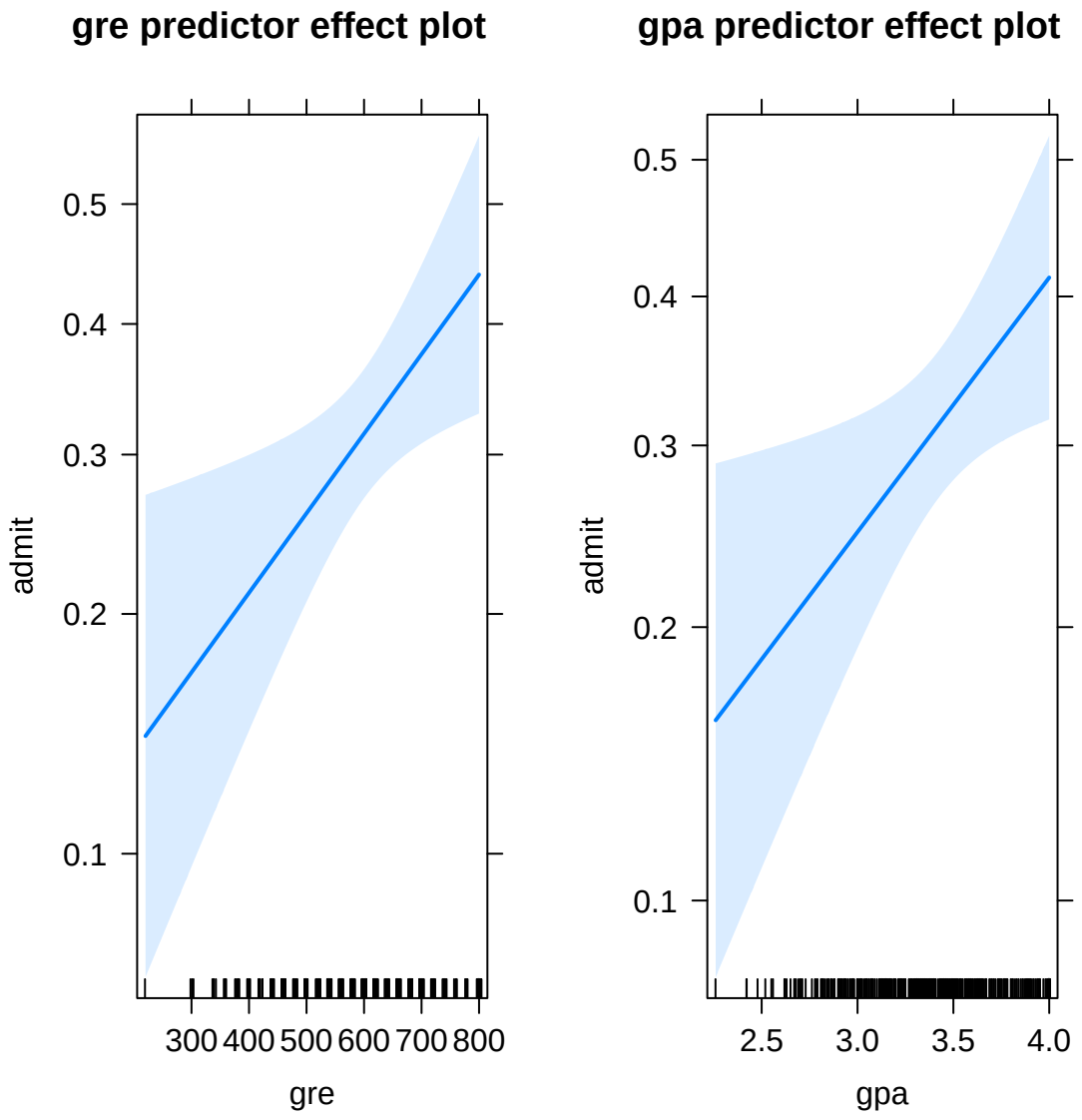
Después de eso, se pueden hacer gráficos:

```
par(mfrow=c(1,2)) # divide el panel de graf's en dos
plot(mydata$gre, prob.scale) # para gre
plot(mydata$gpa, prob.scale) # para gpa
```



Afortunadamente se puede hacer de manera más simple. Usando la librería `effects`, podemos obtener $\hat{y}_i$ expresado en probabilidades. Aquí tomamos $\hat{y}_i$ que está en escala logit (que es también lo que está en la tabla de la regresión logística) y lo mapeamos en la función que retorna probabilidades entre 0 y 1 (Equation 7).

```
p_load("effects")
plot(predictorEffects(logit.1)) # Todo el modelo
```



Nota que incluye automáticamente intervalos de confianza. De qué dependen?

Predicted Probabilities: Tablas La otra alternativa es poder generar tablas con *predicted prob's*. La idea es ir generando "perfiles" que hagan sentido desde un punto de vista substantivo.

```
# generar los rangos de los perfiles deseados
gpa.bajo <- with(mydata,
                 data.frame(
                         gre = c(min(mydata$gre), max(mydata$gre)),
                         gpa = min(mydata$gpa))
                 )
gpa.medio <- with(mydata,
                  data.frame(
                          gre = c(min(mydata$gre), max(mydata$gre)),
                          gpa = mean(mydata$gpa, na.rm = T))
                          )
gpa.alto <- with(mydata,
                       data.frame(
                               gre = c(min(mydata$gre), max(mydata$gre)),
                               gpa = max(mydata$gpa))
                               )
```

```
# usar funcion predict
predict(logit.1, gpa.bajo, type="response") # gpa.bajo

##          1          2
## 0.06587598 0.25138506

predict(logit.1, gpa.medio, type="response") # gpa.medio

##         1         2
## 0.1419589 0.4406515

predict(logit.1, gpa.alto, type="response") # gpa.alto
```

```
##         1         2
## 0.2077272 0.5552525
```

GPA Bajo		GPA Medio		GPA Alto	
Gre Min	*Gre Max*	*Gre Min*	*Gre Max*	*Gre Min*	*Gre Max*
0.07	0.25	0.14	0.44	0.21	0.56

Cuál es el problema?

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

## Writing to file Inferencia_Interpretacion.R
```