

Estadística y modelos predictivos

Santiago Caño Muñiz

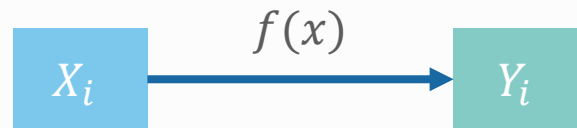


All models are wrong, but some are useful
George Box

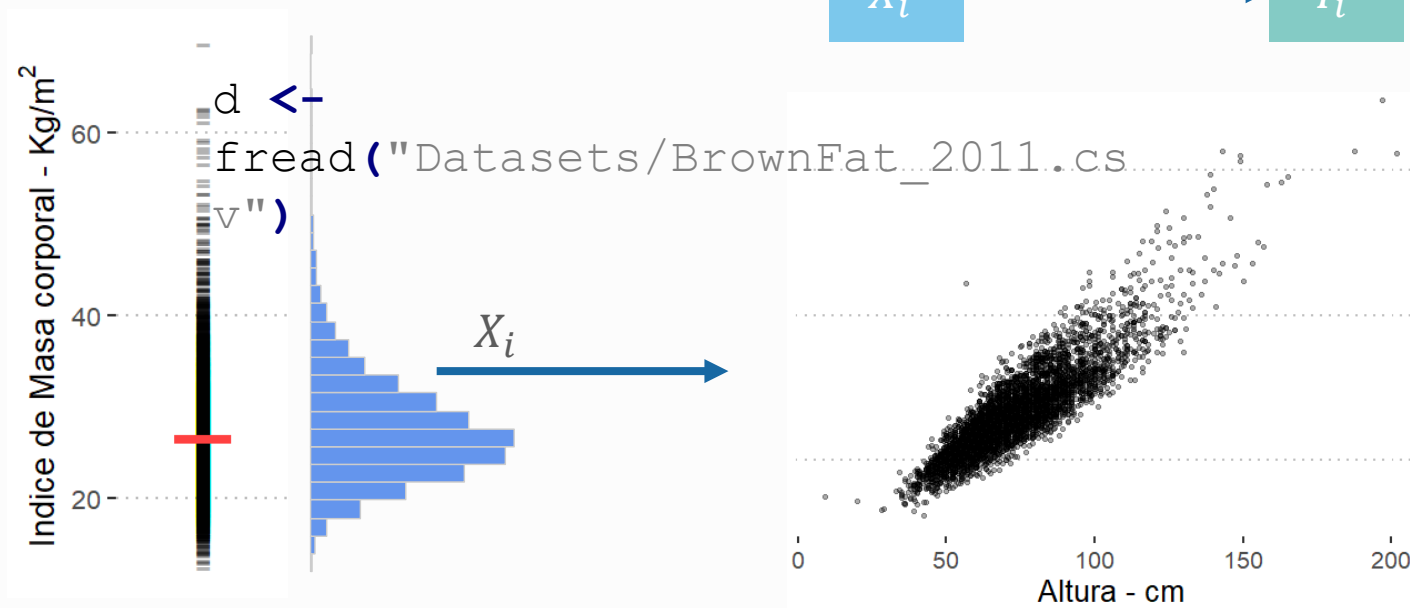
La medida

Data in an uncertain world, perfect knowledge of the uncertainty

$$Y = N(\mu, \sigma)$$



$$Y = [y_i]$$



Presentar resultados

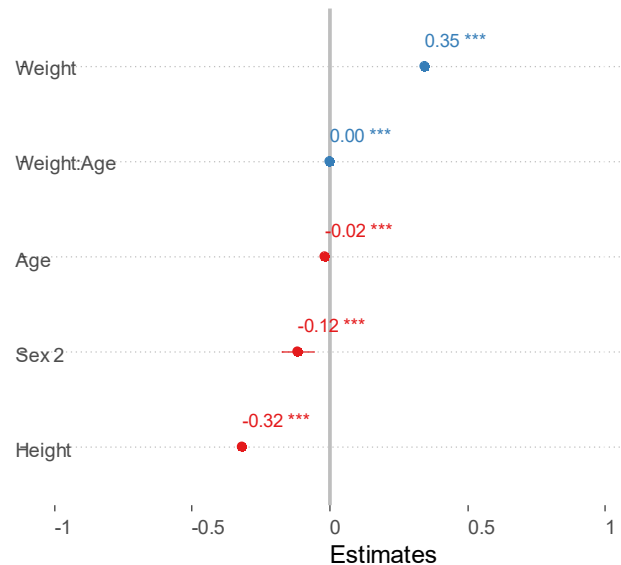
Tablas vs imágenes

| Coef. | 2.50% | 97.50% | Estimate |
|-------------|----------|----------|----------|
| (Intercept) | 52.84821 | 54.08622 | 53.46721 |
| Weight | 0.341727 | 0.35174 | 0.346734 |
| Age | -0.02391 | -0.01199 | -0.01795 |
| Height | -0.31998 | -0.31354 | -0.31676 |
| Sex2 | -0.17377 | -0.05999 | -0.11688 |
| Weight:Age | 0.000225 | 0.000387 | 0.000306 |

```
result <- confint(m5) %>%
  data.table(., keep.rownames = T)
result[, Estimate := coef(m5)]

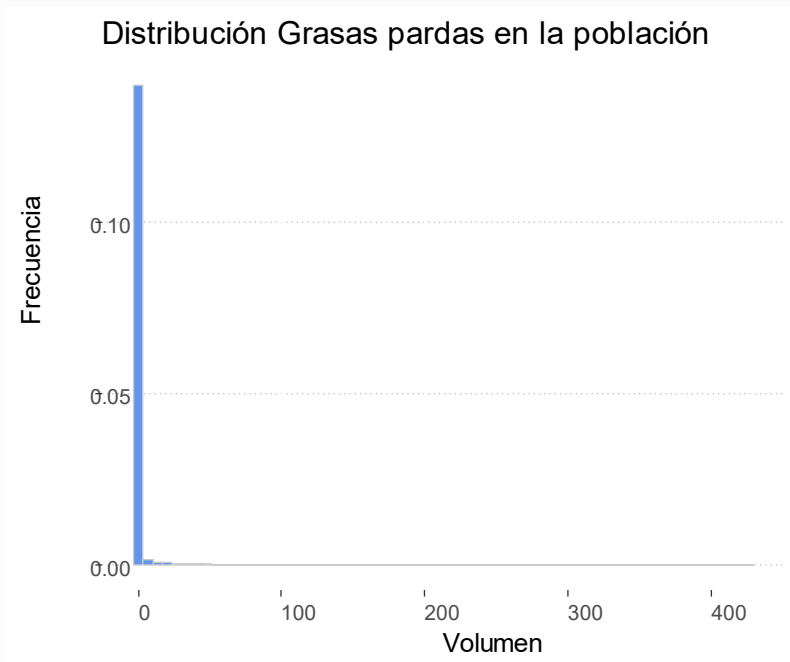
library(sjPlot)
plot_model(m5, show.values = TRUE, sort.est =
  TRUE, value.offset = .3)
```

BMI



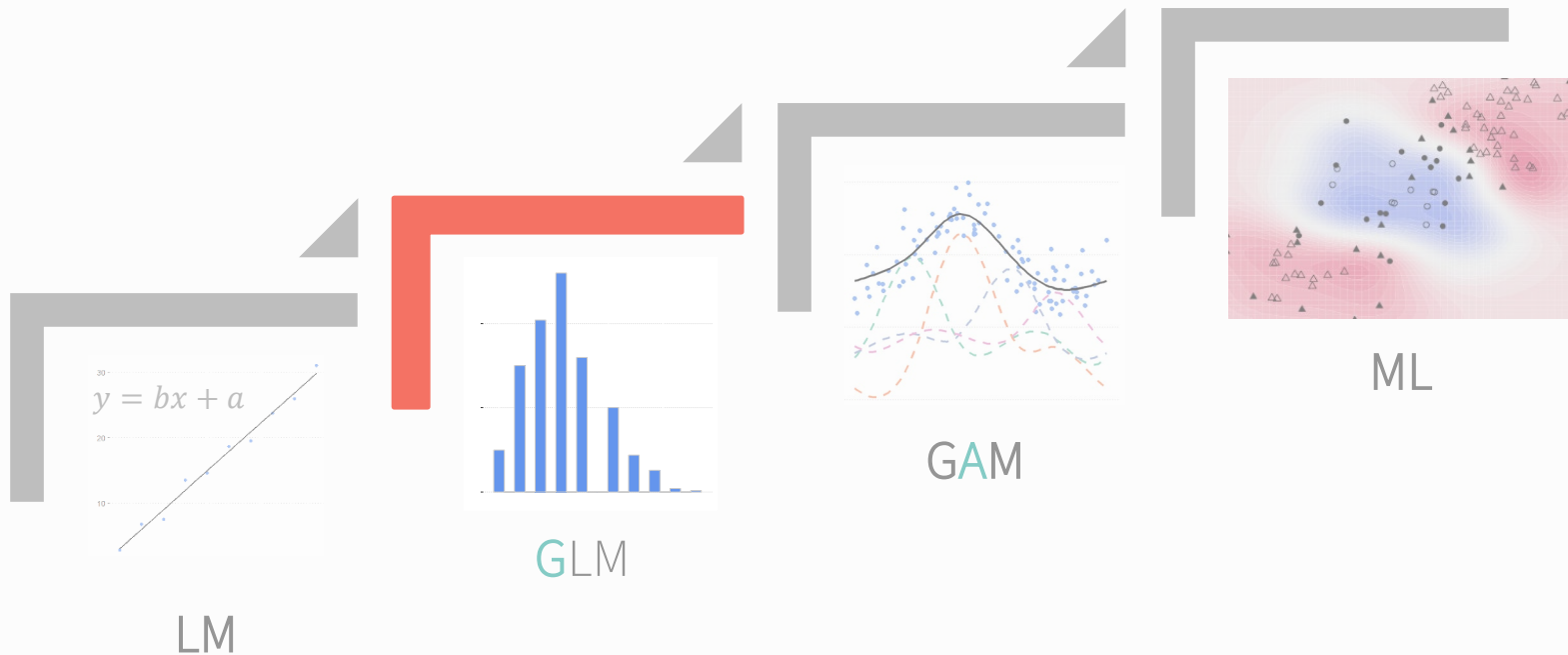
Datos paranormales

¿Qué hacemos cuando los datos no son normales?



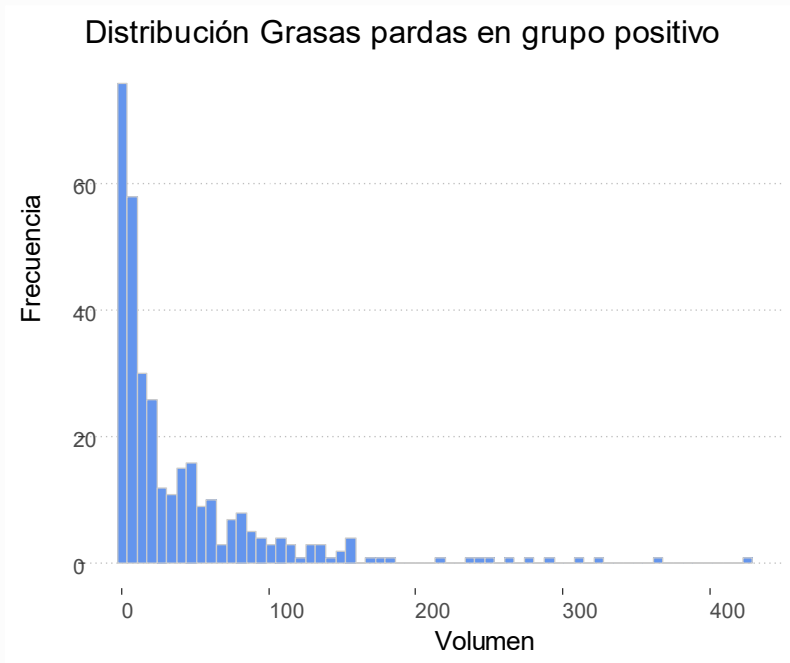
El modelo

Generalización de modelos lineares



Datos paranormales

GLM



Modelando la distribución

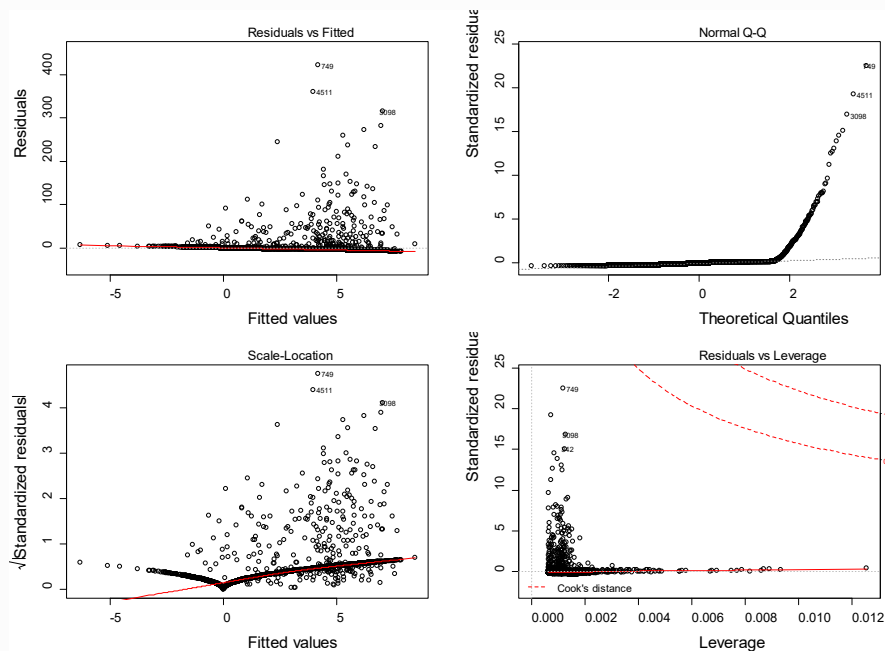
- Las Grasa parda no sigue una distribución normal

$$\begin{cases} Y \sim N(\mu, \sigma^2) \\ \mu \sim \beta_1 X_i + \beta_0 + e_{ij} \end{cases}$$

```
# El modelo lineal simple
m6.1 <- lm(Total_vol ~ Age + LBW +
           Sex + Ext_Temp,
           data = d)
```

Datos parnormales

GLM



Modelando la distribución

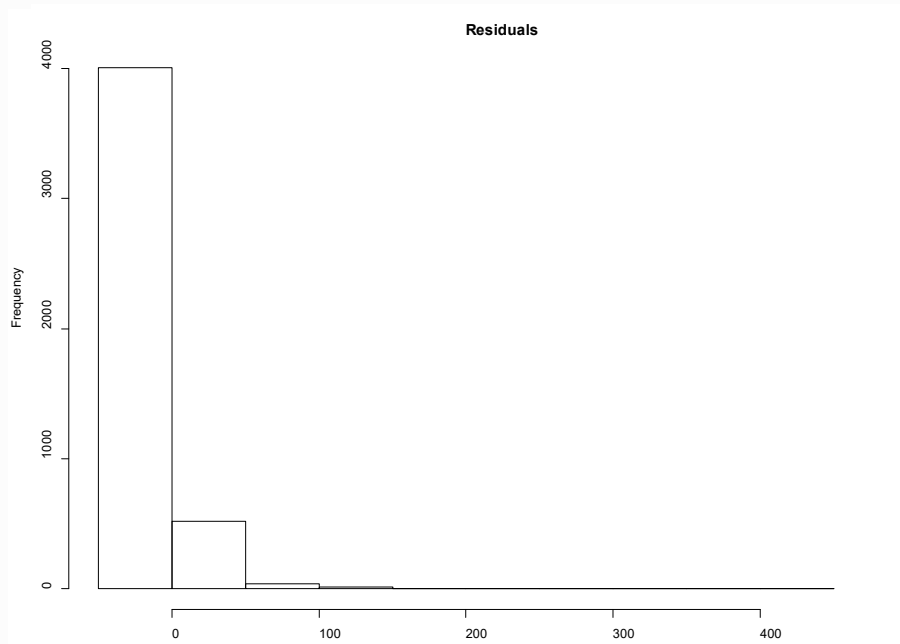
- Las Grasa parda no sigue una distribución normal

$$\begin{cases} Y \not\sim N(\mu, \sigma^2) \\ \mu \sim \beta_1 X_i + \beta_0 + e_{ij} \end{cases}$$

$$\begin{cases} Y \sim g(\mathbf{E}(\mathbf{y}), \mathbf{Var}(\mathbf{y}), \mathbb{P}(\mathbf{0})) \\ \mathbf{E}(\mathbf{y}) = \mu \sim \beta_1 X_i + \beta_0 + e_{ij} \end{cases}$$

Datos paranormales

GLM



Modelando la distribución

- Las Grasa parda no sigue una distribución normal

$$\begin{cases} Y \not\sim N(\mu, \sigma^2) \\ \mu \sim \beta_1 X_i + \beta_0 + e_{ij} \end{cases}$$

$$\begin{cases} Y \sim g(\mathbf{E}(\mathbf{y}), \mathbf{Var}(\mathbf{y}), \mathbf{P}(\mathbf{0})) \\ \mathbf{E}(\mathbf{y}) = \beta_1 X_i + \beta_0 + e_{ij} \end{cases}$$

¿Puede un tratamiento X curar la tuberculosis?

Pensar en la **pregunta** adecuada

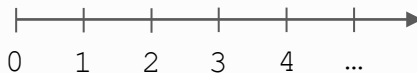
Resultado del tratamiento en los pacientes

Positivo | Negativo

Mejora | Empeora |
Transplante | Defunción

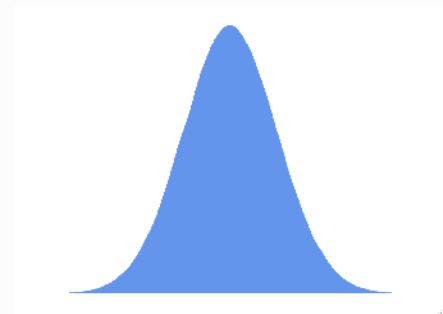
Binomial
Multinomial
Logística

Número de bacterias en una biopsia del paciente



Poisson
N-binomial

Concentración de antibiótico para inhibir el crecimiento de la bacteria



Gaussian
T-Student
Gamma
Tweedie

Datos paranormales

GLM

Modelando la distribución

- Las Grasa parda no sigue una distribución normal

```
glmmTMB (formula = ...,
         ziformula = ...,
         dispformula = ...,
         family = ...
         data = ...
         )
```

$$\left\{ \begin{array}{l} Y \sim \mathbf{EF}(\mathbf{E}(\mathbf{y}), \mathbf{Var}(\mathbf{y}), \mathbb{P}(\mathbf{y} = \mathbf{0})) \\ \mathbf{g}(\mathbf{E}(\mathbf{y})) = \beta_1 X_i + \beta_0 + e_{ij} \end{array} \right.$$

Datos parnormales

Modelar la distribución con glmmTMB

```
library(glmmTMB)

# Igual que m6.1 pero con notación GLM explícita

m6.1.1 <- glmmTMB(Total_vol ~ BMI + Cancer_Status + Sex + Season,
                  data = d, family = gaussian)

m6.2 <- glmmTMB(Total_vol ~ BMI + Cancer_Status + Sex + Season,
                data = d, family = tweedie(link = "log")) # Modelando la distribución

AIC(m6.1, m6.1.1, m6.2)

>      df      AIC
m6.1   6 39960.32
m6.1.1 6 39960.32
m6.2   7 5083.34
```

Datos paranormales

Entender el resultado

```

Family: tweedie ( log )
Formula:      Total_vol ~ Cancer_Status + Sex + Season
Data: d[, `:=`(Cancer_Status, as.factor(Cancer_Status))]

      AIC      BIC   logLik deviance df.resid
5104.6  5143.2  -2546.3  5092.6     4586

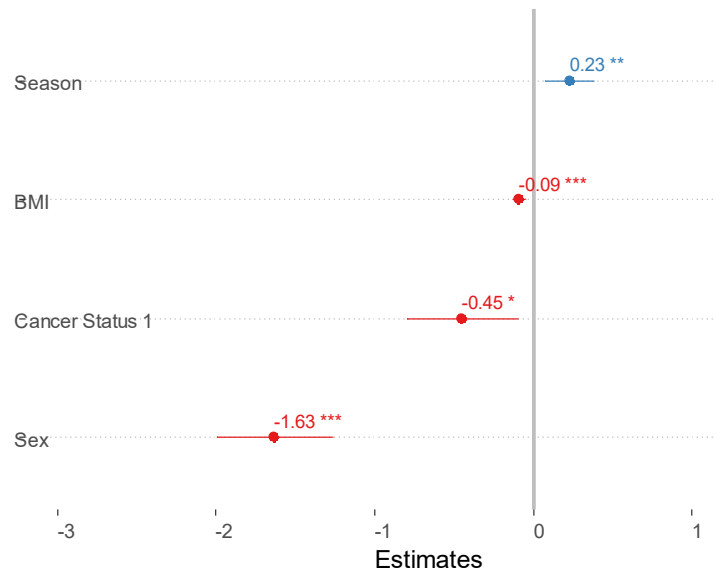
Overdispersion parameter for tweedie family (): 49.3

Conditional model:
      Estimate Std. Error z value Pr(>|z|)
(Intercept)   5.08051    0.58791   8.642 < 2e-16 ***
BMI           -0.08965    0.01883  -4.761 1.93e-06 ***
Cancer_Status1 -0.44755    0.17739  -2.523 0.01164 *
Sex           -1.63311    0.18545  -8.806 < 2e-16 ***
Season        0.22783    0.07683   2.966 0.00302 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# Representar el resultado
plot_model(m6.2, sort.est = T, show.values = TRUE)

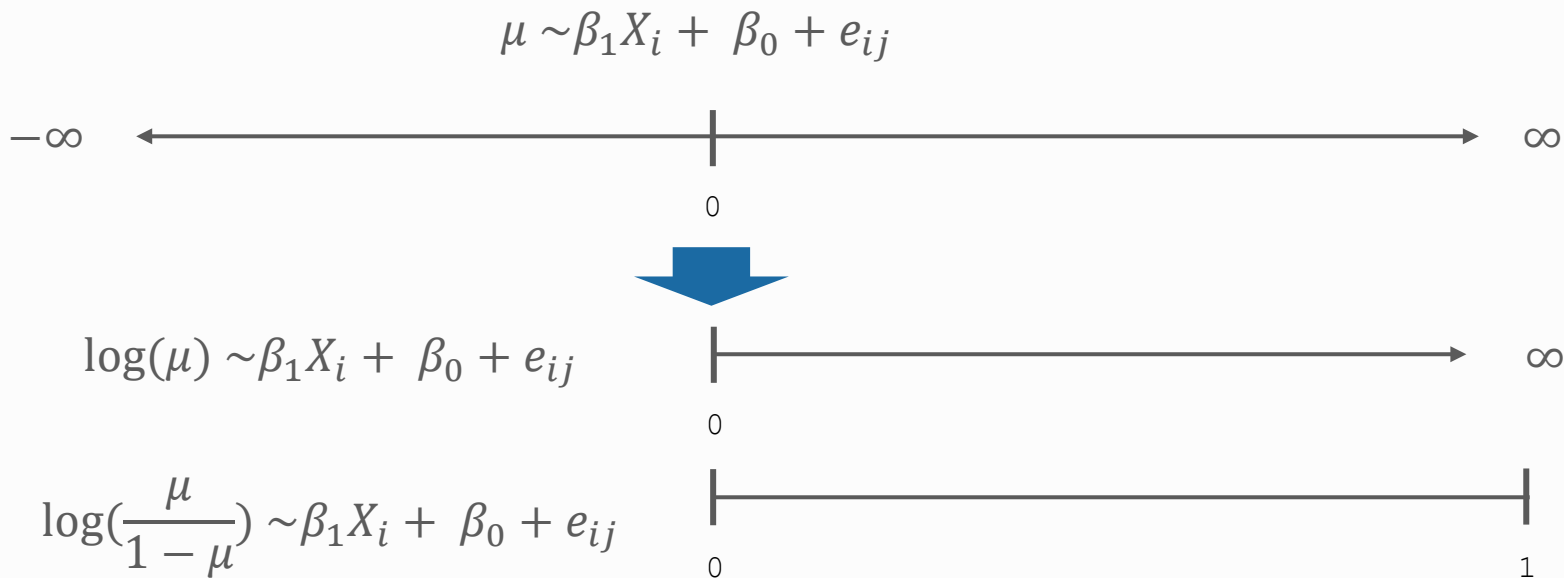
```

Total vol



Datos paranormales

¿Qué es una función de enlace?



Datos paranormales

Entender el resultado

```

Family: tweedie ( log )
Formula:      Total_vol ~ Cancer_Status + Sex + Season
Data: d[, `:=`(Cancer_Status, as.factor(Cancer_Status))]

      AIC      BIC   logLik deviance df.resid
5104.6  5143.2  -2546.3  5092.6    4586

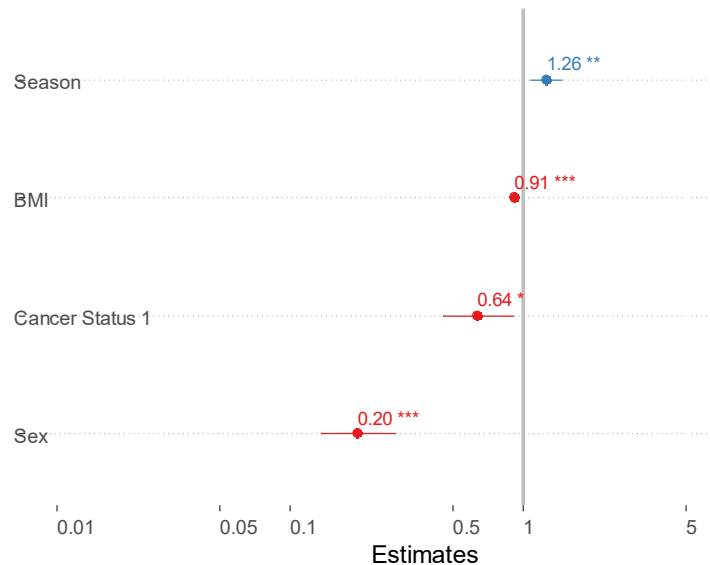
Overdispersion parameter for tweedie family (): 49.3

Conditional model:
      Estimate Std. Error z value Pr(>|z|)
(Intercept)   2.80629    0.34430   8.151 3.62e-16 ***
Cancer_Status1 -0.37062    0.17799  -2.082 0.03731 *
Sex           -1.68022    0.18626  -9.021 < 2e-16 ***
Season        0.23416    0.07764   3.016 0.00256 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# Representar el resultado
plot_model(m6.2, sort.est = TRUE, show.values = TRUE,
           transform = "exp")

```

Total vol



A programar

Por ejemplo

```
library(mgcv)

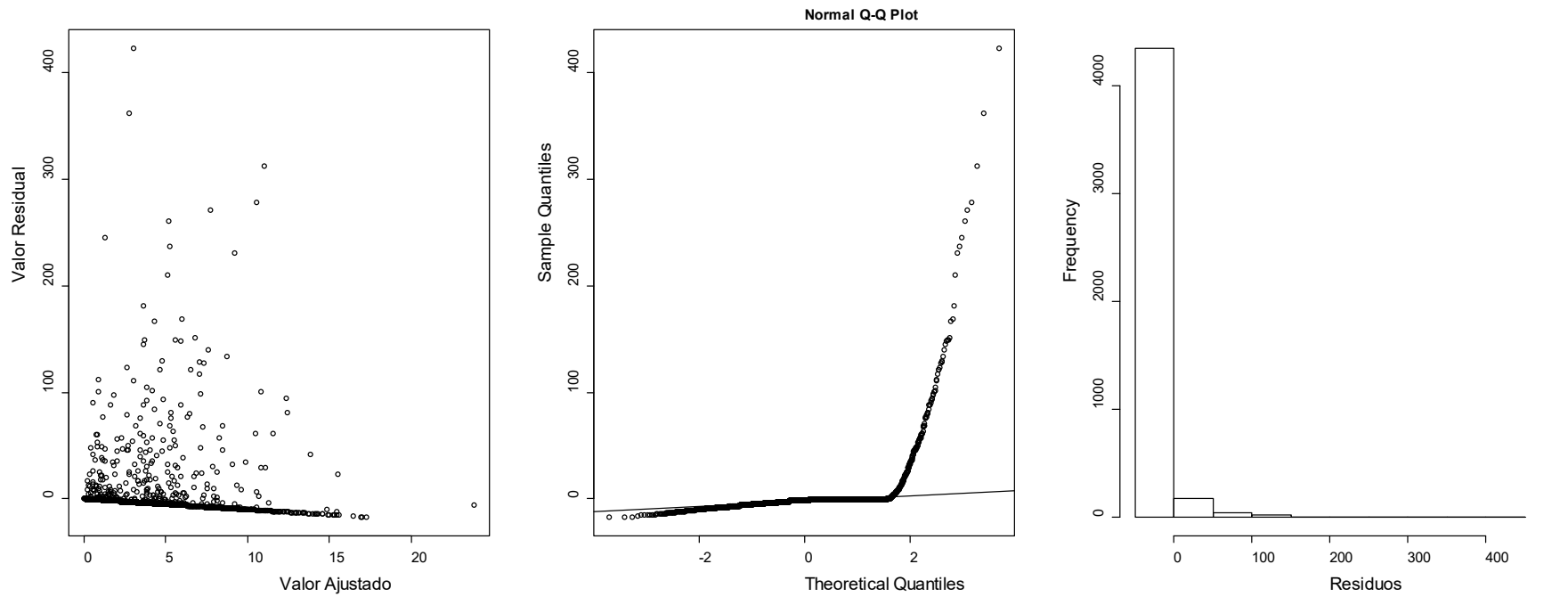
m <- lm(BMI ~ Weigth, # Formula Y ~ X
        data = d, )

summary(m)

plot(m)
```

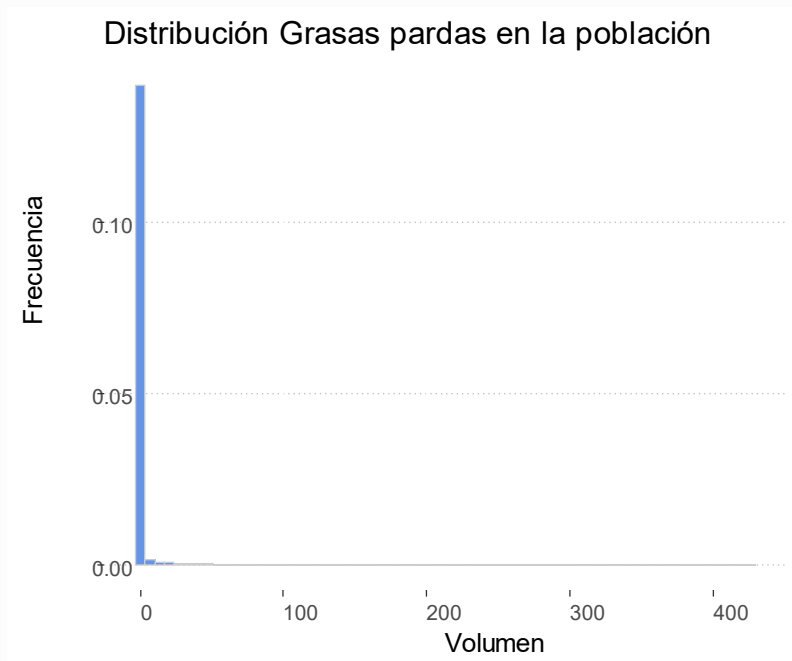
Datos paranormales

Mucho por mejorar



Datos parnormales

Inspeccionar la dispersion y exceso de 0

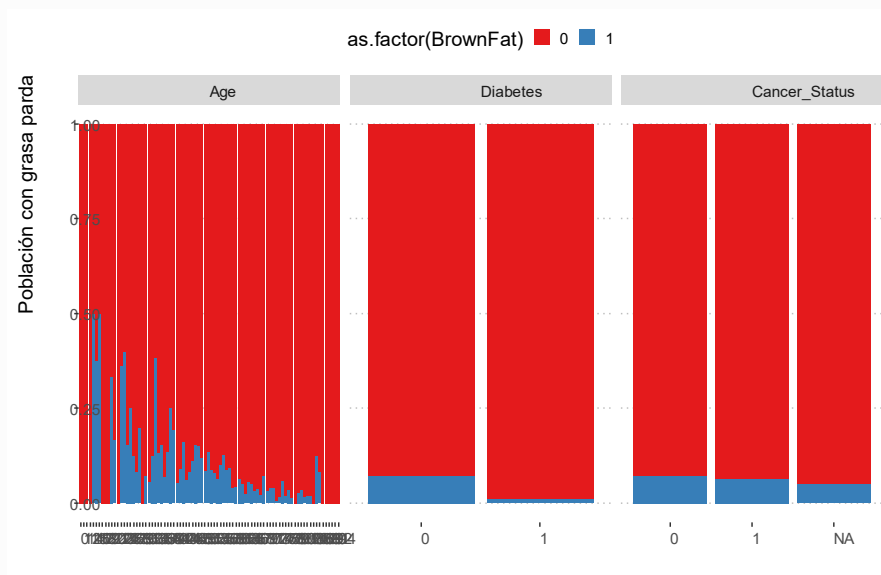


Modelando los ceros

- La mayor parte de la población carece de grasa parda
 - Podría existir un proceso que “activa” la presencia de grasa parda
 - Una vez activada la presencia de grasa parda, la masa empieza a crecer
- La inspección visual de variables separadas nos puede dar una pista de que factores participan en la generación de 0

Datos paranormales

Inspeccionar el exceso de 0

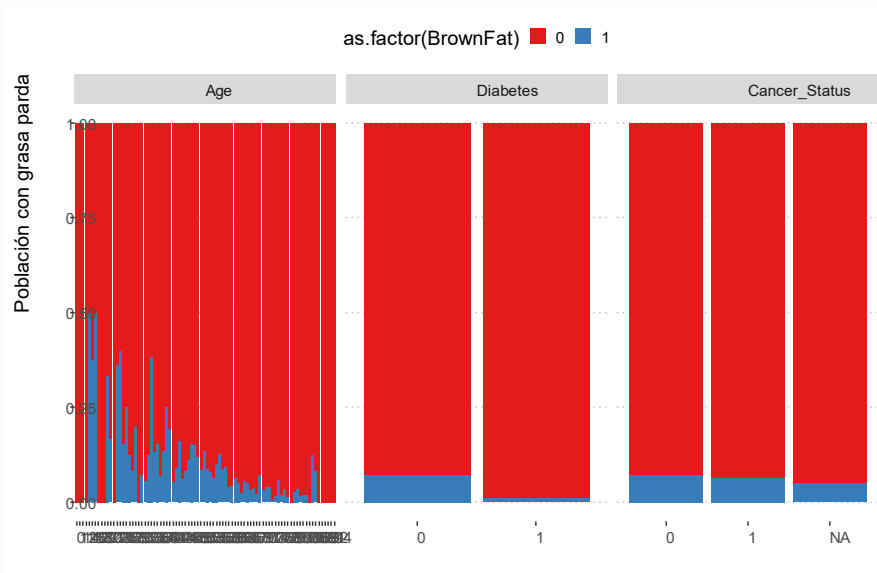


¿Cómo explorar la densidad en 0?

- Buscamos relacionar distintas variables con la probabilidad de $y = 0$
 - Contar $y = 0$
 - Contar $y \neq 0$
 - Comparar

Datos parnormales

Inspeccionar el exceso de 0



```
# Seleccionar variables
d[, .(BrownFat, Age, Diabetes, Cancer_Status)] %>%
# Agregar entorno a la variable de interes

melt.data.table(., id.vars = "BrownFat") %>%
# Contar obs. En cada categoría
.[, .(Count = .N),
  by = .(variable, value, BrownFat)] %>%
# Graficar
ggplot(., aes(x = as.factor(value), y = Count,
  fill = as.factor(BrownFat))) +

# Formato barra. Apilar por color
geom_bar(stat = "identity", position = "fill") +
# Un panel para cada variable
facet_wrap(~ variable,
  scales = "free_x", nrow = 1)
```

Datos paranormales

Modelar el exceso de 0

$$\begin{cases} \mathbb{E}(\mathbf{y}) = \beta_1 X_i + \dots + \beta_0 + e_{ij} \\ \mathbb{P}(\mathbf{y} = \mathbf{0}) = \beta_1 X_i + \dots + \beta_0 + e_{ij} \end{cases}$$

```
m7.2 <- glmmTMB(Total_vol ~ Season +
                 BMI +
                 Cancer_Status,
                 ziformula = ~ Diabetes +
                 Age,
                 data = d, family = tweedie)
```

Modelando los ceros

- El volumen de grasa parda depende de
 - BMI
 - Cancer
 - Sexo
 - Season
- La **PROBABILIDAD** de tener o no grasa parda depende de
 - Diabetes
 - Edad

Datos paranormales

Modelar la dispersión y exceso de 0

```
summary(m7.1)
>...
Conditional model:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   3.73905    0.56212   6.652 2.90e-11 ***
BMI            -0.07472    0.01916  -3.900 9.64e-05 ***
Cancer_Status1 -0.27970    0.17701  -1.580 0.11407
Sex2          -1.53802    0.18670  -8.238 < 2e-16 ***
Season         0.23400    0.07596   3.081 0.00206 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Zero-inflation model:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.62769    1.19632  -3.868 0.00011 ***
Diabetes     2.14340    0.51858   4.133 3.58e-05 ***
Age          0.07682    0.01473   5.216 1.83e-07 ***
```

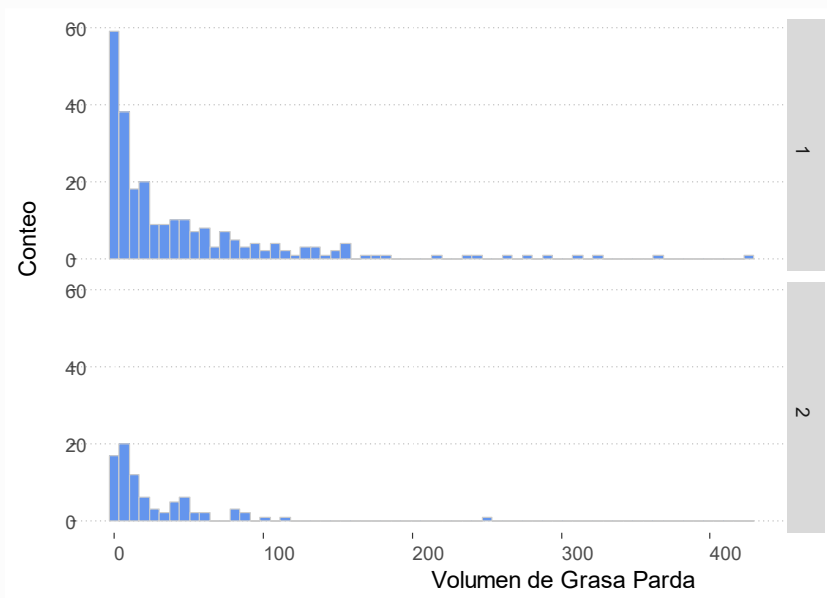
Modelando los ceros

- El volumen de grasa parda depende de
 - BMI
 - Cancer
 - Sexo
 - Season
- La **PROBABILIDAD** de tener o no grasa parda depende de
 - Diabetes
 - Edad



Datos paranormales

Inspeccionar la dispersion

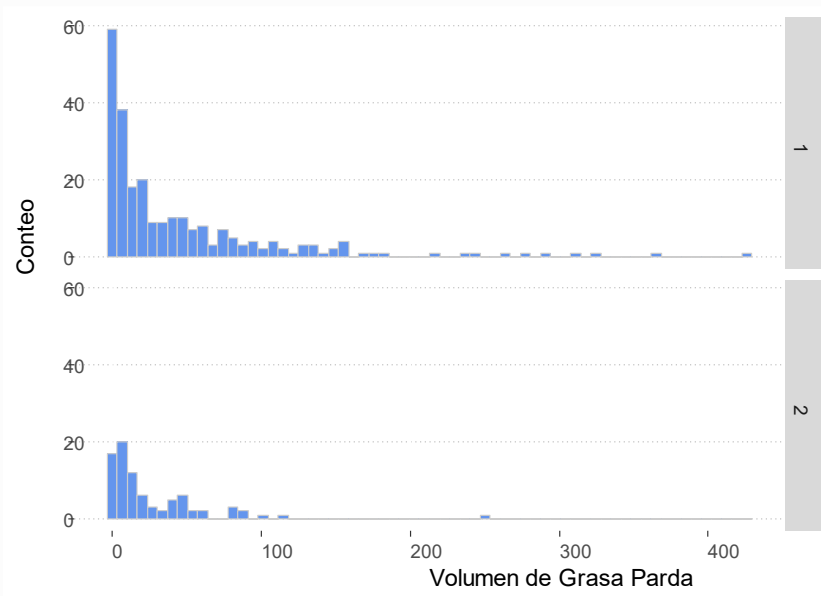


¿Cómo explorar la densidad en 0?

- Buscamos explorar la extensión de los datos por subgrupos
 - Histogramas
 - Diagramas de cajas
 - Diagramas de violín
 - ...

Datos paranormales

Inspeccionar la dispersión



¿Cómo explorar la densidad en 0?

```
# Subset casos > 0
d[Total_vol > 0] %>%
# Graficar
ggplot(., aes(x = Total_vol)) +

  # Representar como histograma
  geom_histogram(fill = "cornflowerblue",
                 col = "gray80", bins = 64) +

  # Dividir panels por sexo
  facet_grid(Sex ~ .) +

  labs(x = "Volumen de Grasa Parda",
       y = "Conteo")
```

Datos paranormales

Modelar el exceso de 0

$$\begin{cases} \mathbb{E}(\mathbf{y}) = \beta_1 X_i + \dots + \beta_0 + e_{ij} \\ \mathbf{Var}(\mathbf{y}) = \beta_1 X_i + \dots + \beta_0 + e_{ij} \end{cases}$$

```
m7.2 <- glmmTMB(Total_vol ~ Season +  
                 BMI +  
                 Cancer_Status,  
                 dispformula = ~ Sex,  
                 data = d, family = tweedie)
```

Modelando los ceros

- El volumen de grasa parda depende de
 - Season
 - BMI
 - Cancer
- La variabilidad del volumen de grasa parda depende de
 - Sexo

Datos paranormales

Modelar el exceso de 0

```
summary(m7.2)
>...
Conditional model:
      Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.54772    0.56428   6.287 3.23e-10 ***
BMI          -0.09128    0.01934  -4.720 2.36e-06 ***
Cancer_Status1 -0.45346    0.18075  -2.509  0.0121 *
Sex2        -1.63022    0.20829  -7.827 5.01e-15 ***
Season       0.20645    0.07833   2.636  0.0084 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Dispersion model:
      Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.73872    0.05503  67.94 < 2e-16 ***
Sex2         0.50576    0.10654   4.75 2.06e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

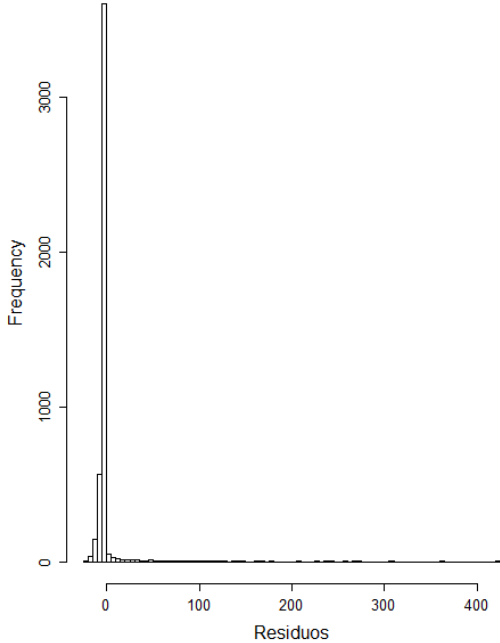
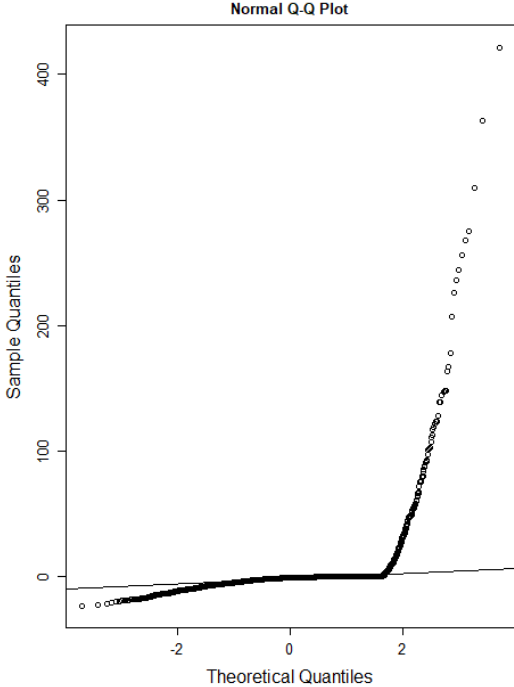
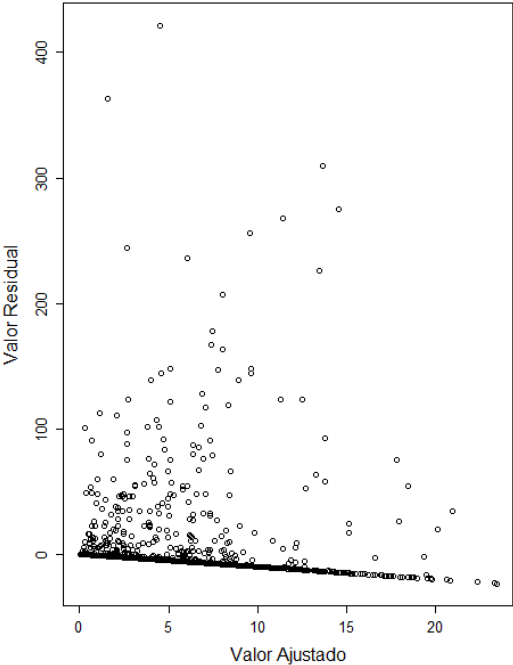
Modelando los ceros

- El volumen de grasa parda depende de
 - Season
 - BMI
 - Cancer
 - Season
- La variabilidad del volumen de grasa parda depende de
 - Sexo



Validar

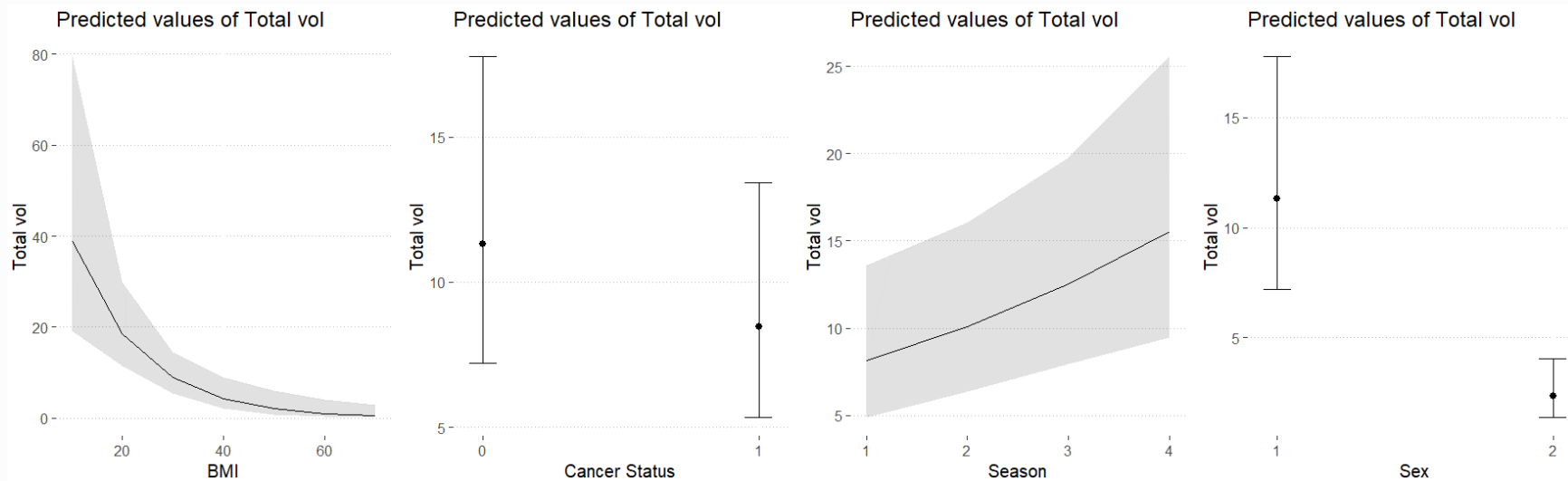
Siempre verificar



Como presentar los resultados

¿Qué hacer cuando el modelo es muy complejo?

```
p <- plot_model(m7.3, type = "pred", vcov.fun = "vcovCL", grid = T)
do.call("grid.arrange", c(p, ncol = 4))
```



A programar

Por ejemplo

```
library(mgcv)

m <- lm(BMI ~ Weigth, # Formula Y ~ X
        data = d, )

summary(m)

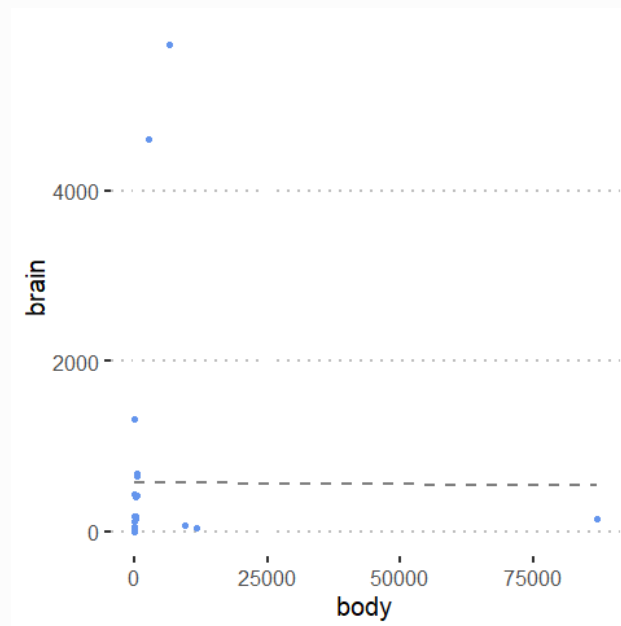
plot(m)
```


Rompiendo las reglas

Cuando los datos no son normales

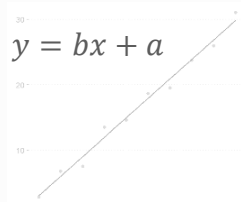
Transformaciones

- Cuando los datos son:
 - No lineales
- Transformaciones monotónicas
 - Normalizar = $\frac{x_i - \mu}{\sigma^2}$
 - Logaritmo = $\log(x)$
 - Ratios = $\frac{x_1}{x_2}$
- Cuidado, cuantas más transformaciones hagamos, más difícil es **interpretar** el modelo



El modelo

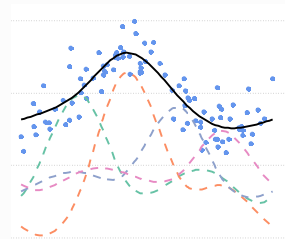
Modelos Aditivos


$$y = bx + a$$

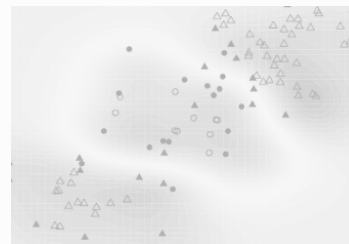
LM



GLM



GAM



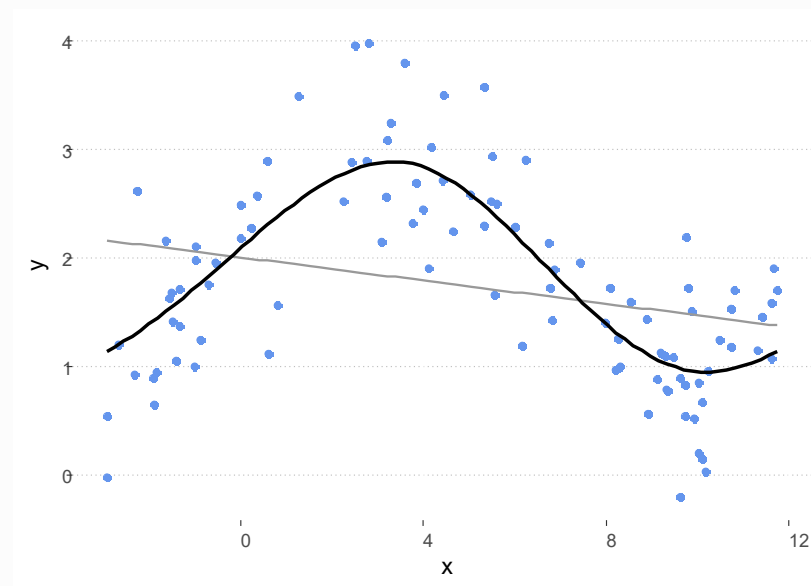
ML

Rompiendo las reglas

Cuando los dantos no son normales

Modelos aditivos

- Cuando los datos son:
 - No lineales
 - No **monotónicas**
- Polinomios
 - $\mathbb{E}(y) = a + \beta_1 x_1 + \beta_2 x_1^2 + \dots + \beta_n x_n^n + \varepsilon$
- Modelos aditivos
 - $\mathbb{E}(y) = a + \mathbf{f}_1(x_1) + \mathbf{f}_2(x_2) + \dots + \mathbf{f}_n(x_n) + \varepsilon$

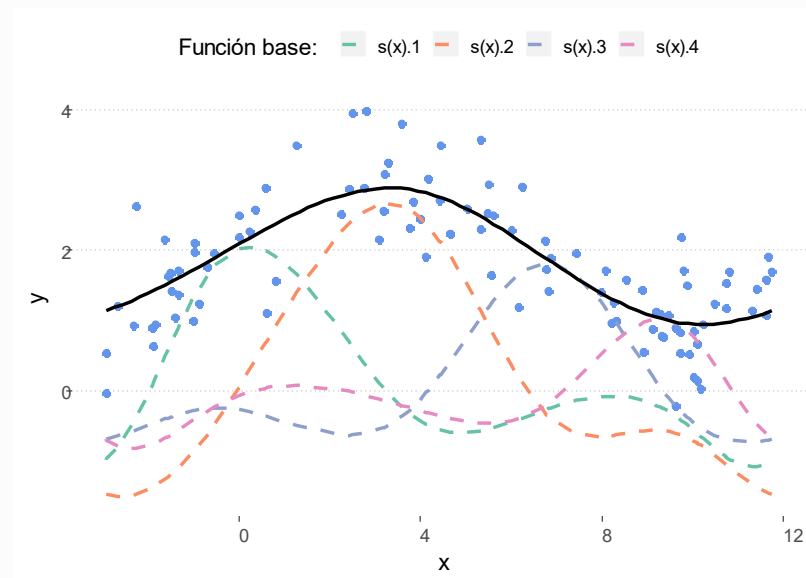


Rompiendo las reglas

Cuando los datos no son normales

Función base

- Cuando los datos son:
 - No lineales
 - No **monotónicas**
- Polinomios
 - $\mathbb{E}(y) = a + \beta_1 x_1 + \beta_2 x_1^2 + \dots + \beta_n x_n^n + \varepsilon$
- Modelos aditivos
 - $\mathbb{E}(y) = a + f_1(x_1) + f_2(x_2) + \dots + f_n(x_n) + \varepsilon$

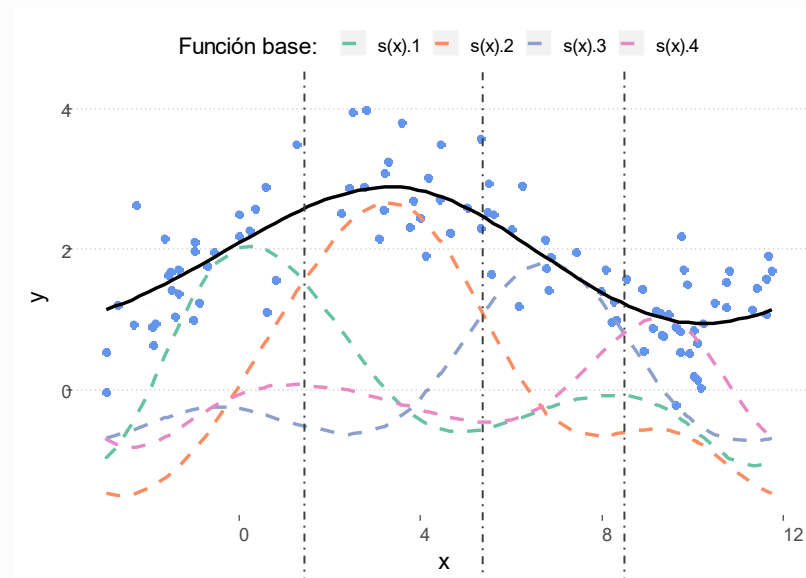


Rompiendo las reglas

Cuando los datos no son normales

Función base

- Cuando los datos son:
 - No lineales
 - No **monotónicas**
- Polinomios
 - $\mathbb{E}(y) = a + \beta_1 x_1 + \beta_2 x_1^2 + \dots + \beta_n x_n^n + \varepsilon$
- Modelos aditivos
 - $\mathbb{E}(y) = a + f_1(x_1) + f_2(x_2) + \dots + f_n(x_n) + \varepsilon$



Modelos aditivos

mgcv

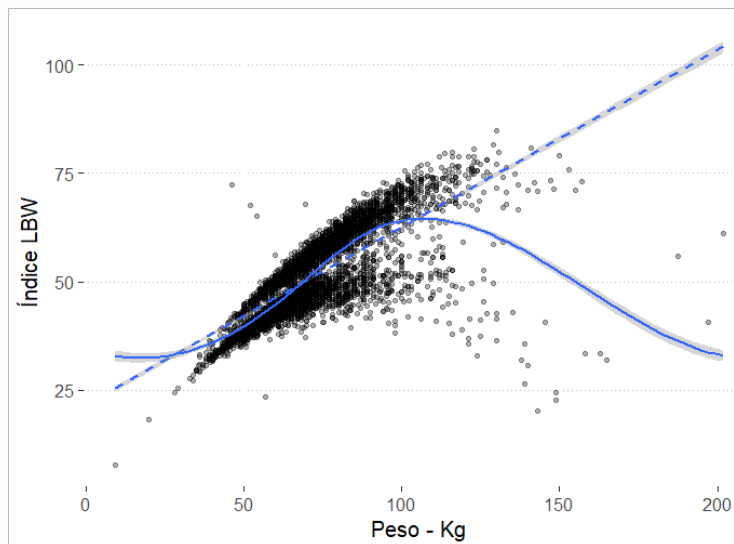
mgcv::gam

```
gam(formula = y ~ x1 +          ← Variables lineales
     s(x2,                       ← Variable aditiva s()
     bs = "tp",                  ← Tipo de función
     k = 10, ...) ,            ← Número de nodos

     data = d,                  ← Datos
     family = ...               ← Distribución de los datos
)
```

Modelos aditivos

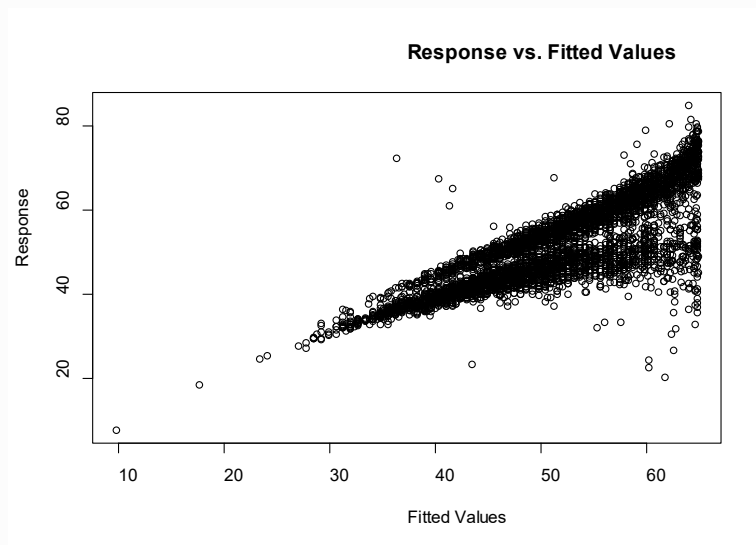
Regresión no lineal simple



```
ggplot(d,  
  aes(x = Weight , y = LBW)) +  
  geom_point(alpha = 0.3) +  
  
  stat_smooth(method = "lm", linetype = 2) +  
  stat_smooth(method = "gam",  
  
  labs(x = "Peso - Kg", y = "Índice LBW")  
  
m0 <- lm(LBW ~ Weight, data = d)  
  
m1 <- gam(LBW ~ s(Weight , k = 3),  
  data = d, method = "REML")
```

Modelos aditivos

Verificar el modelo



```
gam.check(m)
```

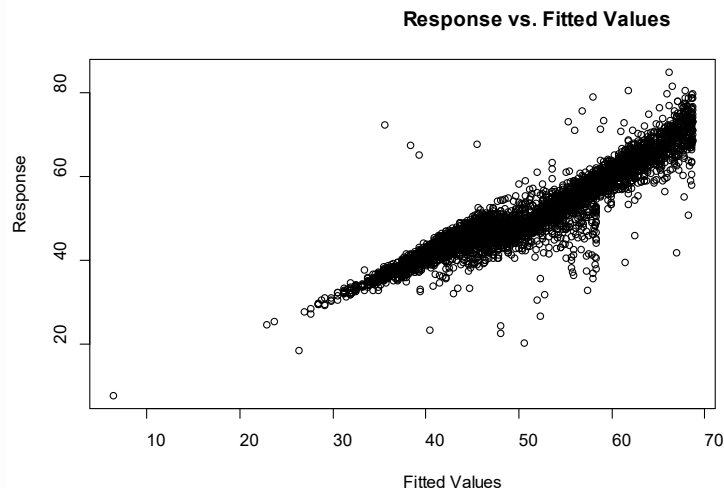
```
>Method: REML   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-0.0008537355,0.0008522113]
(score 13091.98 & scale 13.02218).
Hessian positive definite, eigenvalue range
[0.5001828,2419.501].
Model rank = 4 / 4
```

Basis dimension (k) checking results. Low p-value (k-index<1) may indicate that k is too low, especially if edf is close to k'.

| | k' | edf | k-index | p-value |
|-----------|----|-----|---------|---------|
| s(weight) | 2 | 2 | 0.97 | 0.03 * |
| --- | | | | |

Modelos aditivos

Verificar el modelo



```
m <- gam(LBW ~ s(weight, k = 9), data = d,
method = "REML")
gam.check(m)
```

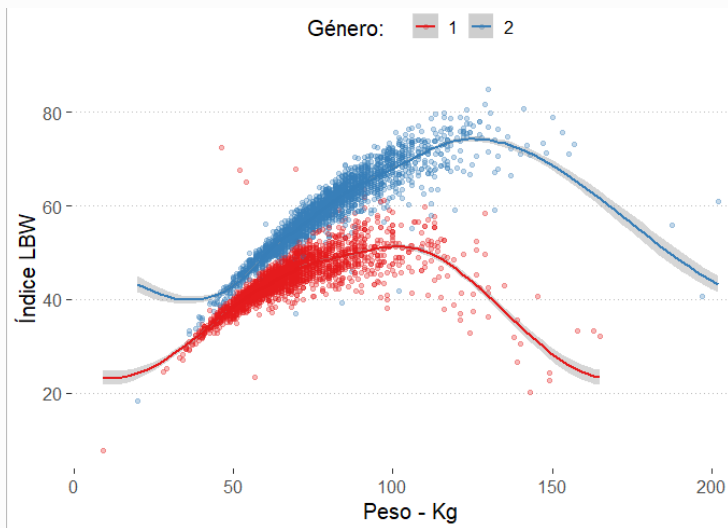
```
>Method: REML   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-0.0008537355,0.0008522113]
(score 13091.98 & scale 13.02218).
Hessian positive definite, eigenvalue range
[0.5001828,2419.501].
Model rank = 4 / 4
```

Basis dimension (k) checking results. Low p-value (k-index<1) may indicate that k is too low, especially if edf is close to k'.

| | k' | edf | k-index | p-value |
|-----------|------|------|---------|---------|
| s(weight) | 8.00 | 7.67 | 1.01 | 0.79 |
| --- | | | | |

Modelos aditivos

Modelo mixto



```
ggplot(d,  
  aes(x = Weight , y = LBW, y = Sex)) +  
  geom_point(alpha = 0.3) +  
  
  stat_smooth(method = "lm", linetype = 2) +  
  stat_smooth(method = "gam",  
    formula = y ~ s(x, k = 3, bs = "cc")) +  
  labs(x = "Peso - Kg", y = "Índice LBW")  
  
m1.3 <- gam(LBW ~ s(Weight , k = 9) + Sex,  
  data = d,  
  method = "REML")
```

Modelos aditivos

Modelo mixto

```

Family: gaussian
Link function: identity

Parametric coefficients:
      Estimate Std. Error t value Pr(>|t|)
(Intercept) 46.28519   0.07576  610.94 <2e-16 ***
Sex2       10.30548   0.10998  93.71  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '.'
Approximate significance of smooth terms:
      edf Ref.df   F p-value
s(Weight) 7.669  7.963 1609 <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '.'

R-sq.(adj) = 0.875   Deviance explained = 87.5%
-REML = 13009   Scale est. = 12.528   n = 4842

```

```

ggplot(d,
  aes(x = Weight , y = LBW, y = Sex)) +
  geom_point(alpha = 0.3) +

  stat_smooth(method = "lm", linetype = 2) +
  stat_smooth(method = "gam",
    formula = y ~ s(x, k = 3, bs = "cc")) +
  labs(x = "Peso - Kg", y = "Índice LBW")

m1.3 <- gam(LBW ~ s(Weight , k = 9) + Sex,
  data = d,
  method = "REML")

summary(m1.3)

```


A programar

Por ejemplo

```
library(mgcv)

m <- lm(BMI ~ Weigth, # Formula Y ~ X
        data = d, )

summary(m)

plot(m)
```

Las interacciones

Como expresarlas en mgcv

- $y \sim s(x_1) + s(x_2)$ ← • 2 variables aditivas
- $y \sim s(x_1, x_2)$ ← • 2D-Interacción aditiva, misma curvatura
- $y \sim te(x_1, x_2)$ ← • 2D-Interacción aditiva, diferente curvatura
- $y \sim te(x_1) + s(x_2) + ti(x_1, x_2)$ ← • 2D curva, interacción independiente

Las interacciones

Comparar entre modelos aditivos

```
m2 <- gam(LBW ~ s(weight, bs = "fs", by = sex) ,  
          data = d, method = "REML")          # Peso modulado por sexo  
  
m3 <- gam(LBW ~ s(weight) + height ,          # Peso combinado con regression lineal altura  
          data = d, method = "REML")  
  
m4 <- gam(LBW ~ s(weight) + s(height) ,      # Peso combinado con curva altura  
          data = d, method = "REML")  
  
m5 <- gam(LBW ~ s(weight, height) + s(weight) + s(height),  
          data = d, method = "REML")        # Reg. curva de peso, curva altura e inteacción  
  
AIC(m, m2, m3, m4, m5) %>% data.table(., keep.rownames = T) %>% .[order(AIC)]
```

```
> rn      df      AIC  
1: m5  36.59825 22645.69  
2: m2  11.42847 24285.65  
3: m4  17.63645 25448.30  
4: m3  11.83294 25675.65  
5: m1.1 10.78283 25993.12
```

Modelos no lineales

Criterios de información

```
summary(m3)
```

Parametric coefficients:

| | Estimate | Std. Error | t value | Pr(> t) |
|-------------|------------|------------|---------|------------|
| (Intercept) | -46.905749 | 1.000730 | -46.87 | <2e-16 *** |
| height | 0.594772 | 0.006038 | 98.50 | <2e-16 *** |

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

Approximate significance of smooth terms:

| | edf | Ref.df | F | p-value |
|-----------|-------|--------|-------|------------|
| s(weight) | 8.695 | 8.97 | 958.4 | <2e-16 *** |

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

```
R-sq.(adj) = 0.883   Deviance explained = 88.3%
-REML = 12856   Scale est. = 11.73       n = 4842
```

```
summary(m5)
```

Parametric coefficients:

| | Estimate | Std. Error | t value | Pr(> t) |
|-------------|----------|------------|---------|------------|
| (Intercept) | 51.5465 | 0.0359 | 1436 | <2e-16 *** |

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

Approximate significance of smooth terms:

| | edf | Ref.df | F | p-value |
|------------------|--------|--------|---------|------------|
| s(weight,height) | 25.312 | 27.00 | 256.475 | <2e-16 *** |
| s(weight) | 1.000 | 1.00 | 1.149 | 0.284 |
| s(height) | 8.005 | 8.71 | 115.501 | <2e-16 *** |

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

```
R-sq.(adj) = 0.938   Deviance explained = 93.8%
-REML = 11380   Scale est. = 6.2417       n = 4842
```

Modelos no lineales

Crterios de información

Funciones curvas vs Tensores

- Using the usual `s()` function for the smooth for interactions uses thin plate splines.
- In this option isotropy is assumed, i.e. the same amount of smoothing is used in both directions (time and month).
- This could be reasonable for spatial fitting, or for interactions where both variables are in the same unit, but not in our case.
- For interactions between variables that should not be smoothed with the same amount, we can use tensor products `te()`

```
m6 <- gam(LBW ~ te(weight, height) + s(weight) +
          s(height), data = d, method = "REML")

m7 <- gam(LBW ~ te(weight, height, by = sex) +
          s(weight) + s(height), data = d,
          method = "REML")

AIC(m5, m6, m7) %>%
  data.table(., keep.rownames = T) %>%
  .[order(AIC)]

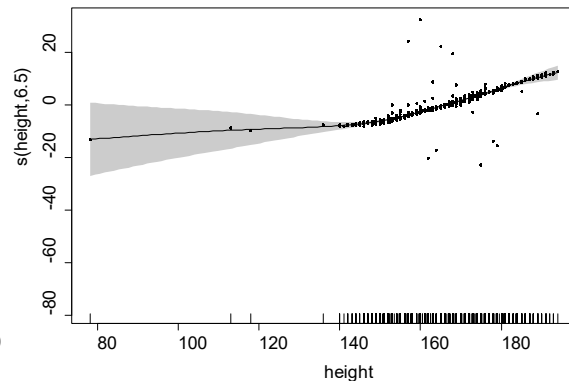
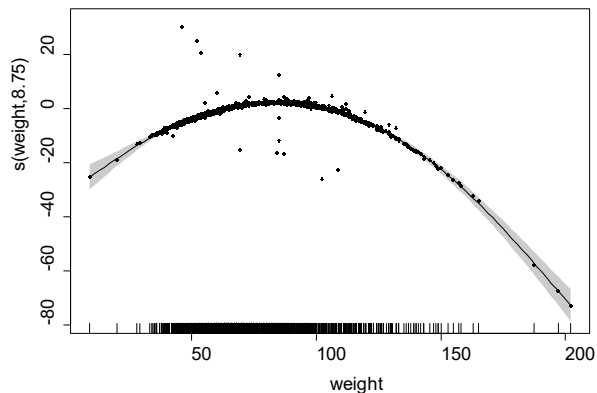
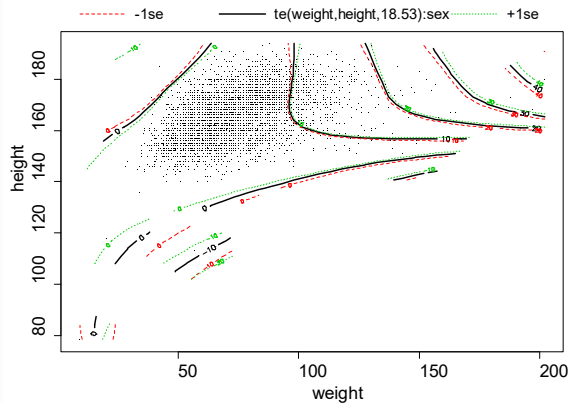
>   rn      df      AIC
1: m7 36.83306 14660.95
2: m6 28.92468 22634.36
3: m5 36.59825 22645.69
```

Validación visual

Efectos individuales

```
m6 <- gam(LBW ~ te(weight, height, by = sex) + s(weight) + s(height), data = d, method = "REML")
```

```
plot(m6, residuals = TRUE, all.terms = TRUE, shade = TRUE, shade.col = "gray80",
      cex = 0.5, pch = 16, cex.main = 1.25, cex.lab = 1.5, cex.axis = 1.5)
```

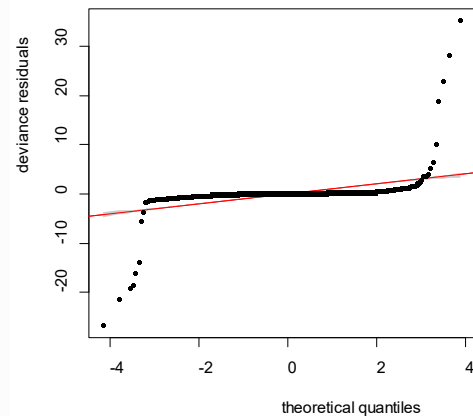
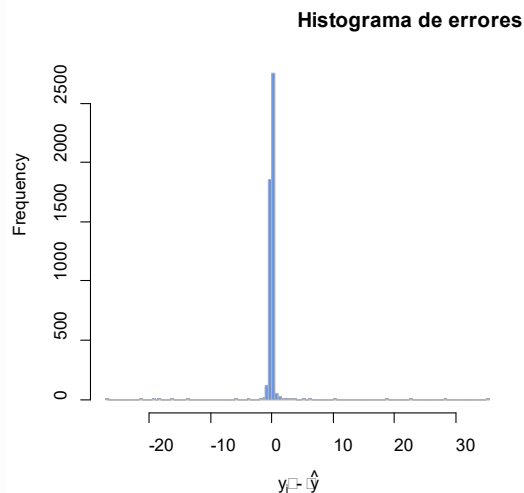
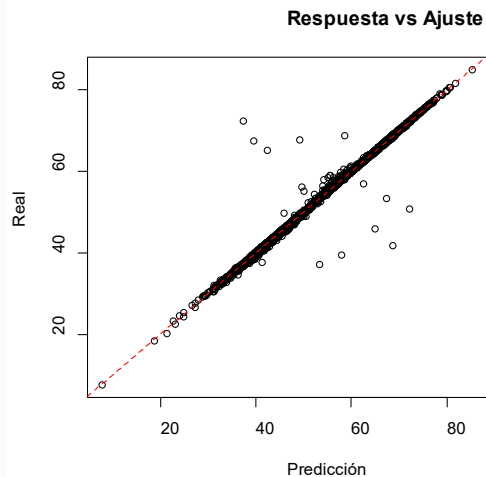


Validación visual

Efectos individuales

```
m6 <- gam(LBW ~ te(weight, height, by = sex) + s(weight) + s(height), data = d, method = "REML")
```

```
plot(m6, residuals = TRUE, all.terms = TRUE, shade = TRUE, shade.col = "gray80",
      cex = 0.5, pch = 16, cex.main = 1.25, cex.lab = 1.5, cex.axis = 1.5)
```



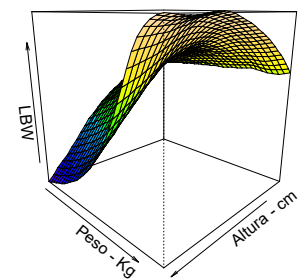
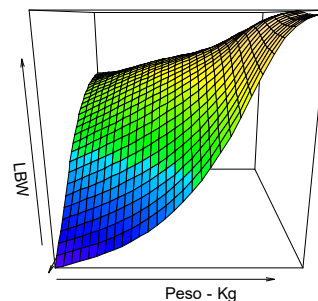
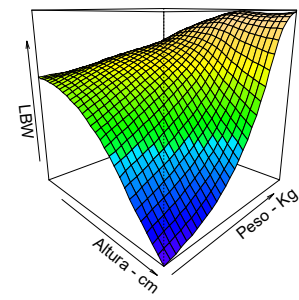
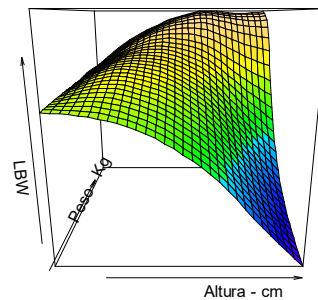
Presentar resultados

Efectos combinados - gráficos isobáricos

- El paquete mgcv contiene herramientas pre-instaladas muy potentes para representación visual

```
par(mfrow=c(2, 2), mar = c(1.8, 1.8, 1.8, 1.8))

for(An in c(0, 45, 90, 135)) {
  vis.gam(m6 , theta = An, color = "topo",
         plot.type = "persp", zlab = "LBW",
         xlab = "Altura - cm",
         ylab = "Peso - Kg")
  rm(An)
}
```

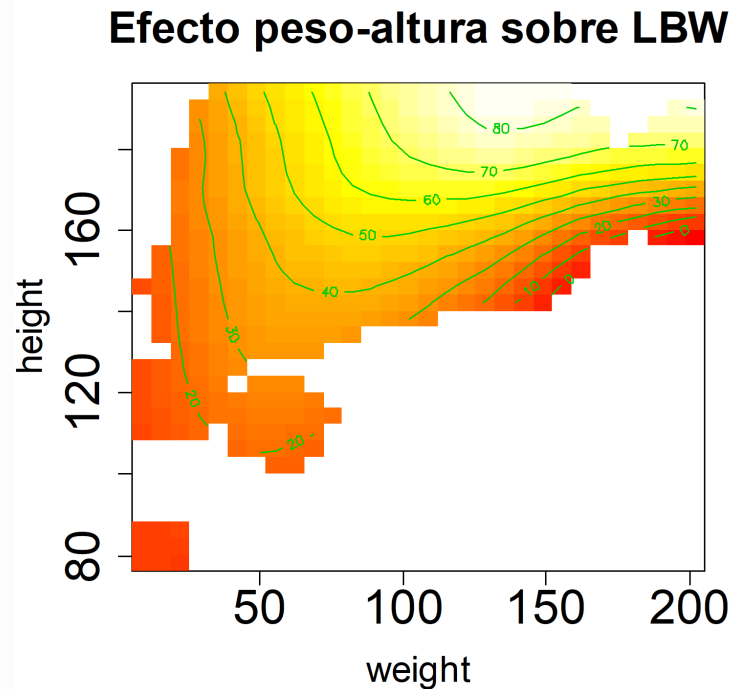


Presentar resultados

Efectos combinados - gráficos isobáricos

- El paquete mgcv contiene herramientas pre-instaladas muy potentes para representación visual

```
vis.gam(m6, view=c("weight", "height"),  
        plot.type = "contour",  
        too.far = 0.1, # 0.1 // 0.3 // 1  
        # Elementos estéticos  
        main = "Efecto peso-altura sobre LBW",  
        cex.main = 1.8, cex.lab = 1.5, cex.axis = 2)
```



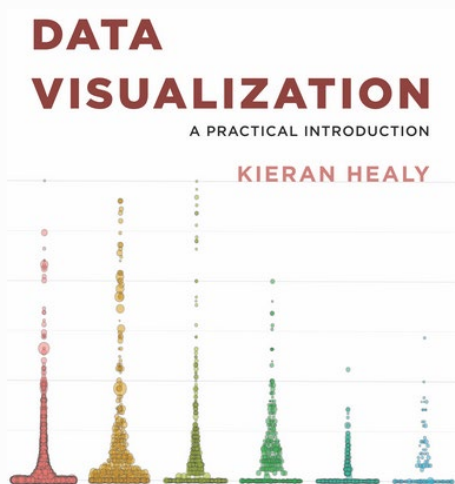
Recordar

¡Cuidado con el sobre-ajuste!

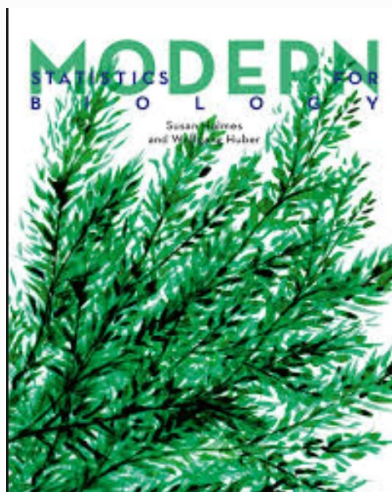


Para saber más

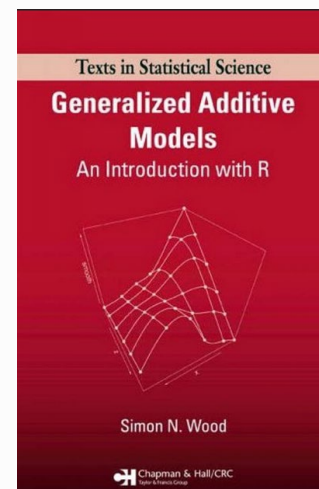
Libros de referencia



<http://socviz.co/>



<https://web.stanford.edu/class/bios221/book/introduction.html>



Generalized Additive Models: An Introduction with R, S Wood



¡Gracias por
¿Preguntas?
vuestro tiempo!