



Clase 9

Modelos mixtos

Miriam Lerma

Mayo 2021

Intro

- Modelos mixtos
- AIC

Ustedes

- Conocimientos de R (saben abrirlo, cargar paquetes y datos, saben hacer operaciones y gráficos).
- Quieren crear modelos mixtos en R y conocer la sintaxis.
La clase de hoy esta en el [repositorio](#): Clonen/Descarguen los materiales.




Preguntas

Responder en el chat 

- Han visto modelos mixtos en artículos o tesis?
- Alguien ha escuchado que es el AIC?

Créditos & Recursos

Lecturas

-  Intro por Gabriela Hajduk
-  Introduccion por Athanasia Mowinckel
-  Lectura sobre mixed-models with R Michel Clark

Videos en español

-  Modelos mixtos por Alejandra Tapia

Imágenes

- Portada
Unsplash by Ilse Orsel

1. Modelos lineares mixtos

1.1. Intro

Modelos lineales:

- Errores aleatorios independientes
- Errores aleatorios siguen una distribución normal

Modelos lineales mixtos

- Incorpora efectos aleatorios para acomodar la correlación entre las observaciones
- Condicionado a los efectos aleatorios, los errores aleatorios son independientes con varianza constante
- Errores aleatorios con distribución normal
- Efectos aleatorios con distribución normal
- Efectos aleatorios y errores aleatorios son independientes

¿Porque usarlos?

- Datos pueden presentar heterogeneidad
- Por ejemplo: pueden estar agrupados por provenir de diferentes áreas, o presentar medidas repetidas
- Medidas repetidas induce a una estructura de correlación, que si no se considera puede llevar a estimaciones sesgadas
- Afectando las predicciones y por lo tanto las decisiones basadas en esos datos

1.2. Datos

Paquetes a cargar para leer los datos

```
library(tidyverse)
library(readr)
library(dplyr)
library(here)
```

Cargar datos

```
url <- "https://raw.githubusercontent.com/MiriamLL/Prueba/master/dragones.csv"
download.file(url, "dragons.csv")
dragones <- read_csv("dragons.csv")
```

Nota Estos datos son repetidamente utilizados para enseñar modelos mixtos. Entonces pueden encontrar estos ejemplos explicados de diferente manera en línea fácilmente.

1.2. Datos

Los datos contienen 480 observaciones contenidas en 4 columnas: calificación, tamaño, montaña, sitio.

```
glimpse(dragones)
```

```
## Rows: 480
## Columns: 4
## $ calificacion <dbl> 16.147309, 33.886183, 6.038333, 18.838821, 33.862328,
## $ tamaño <dbl> 165.5485, 167.5593, 165.8830, 167.6855, 169.9597, 168.
## $ montania <chr> "Bavarian", "Bavarian", "Bavarian", "Bavarian", "Bavar
## $ sitio <chr> "a", "a", "a", "a", "a", "a", "a", "a", "a", "a", "a", "a",
```

1.3. Preguntas

Imaginemos que:

Se entrenaron dragones y se recopilaron datos de diferentes montañas.



Entre mas rápido aprendieron el entrenamiento mejor calificación, también se midieron y se anotó en que sitio y en que región provienen.

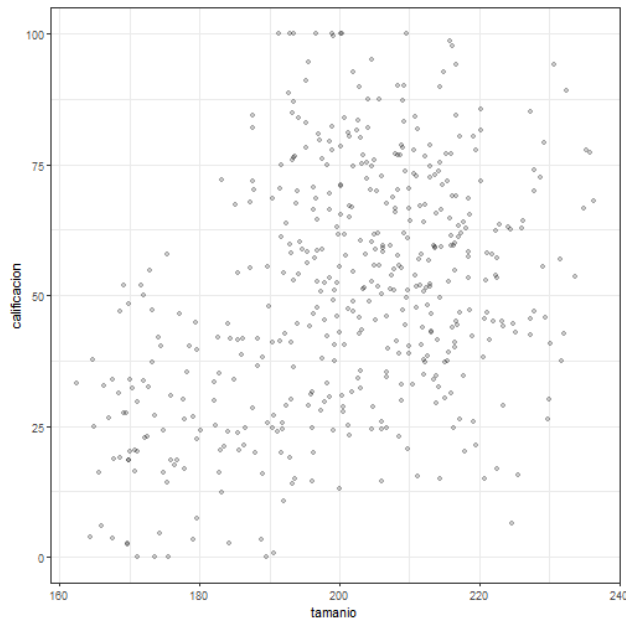


¿Afectará el tamaño de los dragones sus calificaciones?

1.4. Inspeccionar datos

Creemos un gráfico de puntos

```
dragones %>%  
  ggplot(aes(x=tamano,  
             y=calificacion)) +  
    geom_jitter(alpha=.2)+  
    theme_bw()
```



1.5 Modelo lineal

Cargar paquetes

```
library(broom)
```

Ajustar un modelo lineal

```
ajuste_lm <- lm(calificacion ~ tamaño, data=dragones)
```

Mirar estadísticos del modelo

```
broom::tidy(ajuste_lm) %>% tibble::as_tibble()
```

```
## # A tibble: 2 x 5
##   term          estimate std.error statistic  p.value
##   <chr>         <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)  -61.3      12.1      -5.08 5.38e- 7
## 2 tamaño        0.555     0.0597     9.29 5.59e-19
```

¿Entre más grandes, más inteligentes? 🤔

1.5 Modelo lineal

Información sobre el modelo.

```
glance(ajuste_lm) %>%  
  tibble::as_tibble()
```

Información sobre el ajuste del modelo. Esta función agrega los valores ajustados, los residuales y otros al data frame.

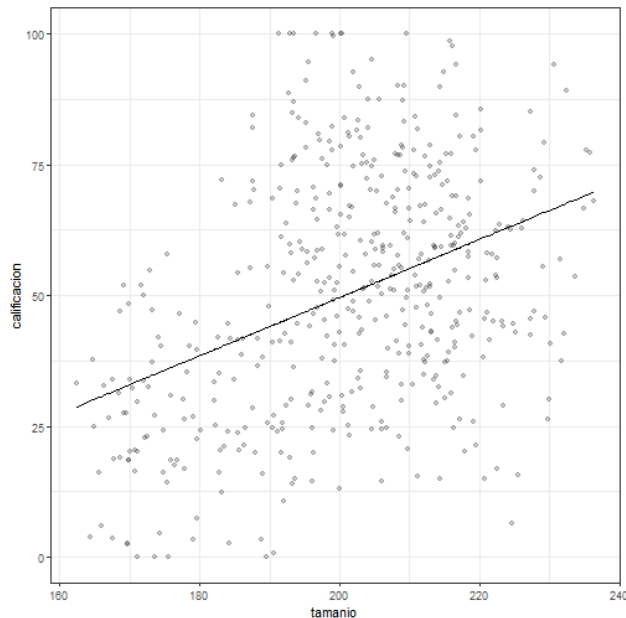
```
info_ajuste_lm<-augment_columns(ajuste_lm, dragones)  
info_ajuste_lm
```

```
## # A tibble: 480 x 11  
##   calificacion tamaño montania sitio .fitted .se.fit .resid .hat .sig  
##   <dbl> <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 16.1 166. Bavarian a 30.5 2.35 -14.4 0.0122 21  
## 2 33.9 168. Bavarian a 31.7 2.24 2.23 0.0111 21  
## 3 6.04 166. Bavarian a 30.7 2.33 -24.7 0.0121 21  
## 4 18.8 168. Bavarian a 31.7 2.23 -12.9 0.0111 21  
## 5 33.9 170. Bavarian a 33.0 2.11 0.875 0.00989 21  
## 6 47.0 169. Bavarian a 32.3 2.18 14.8 0.0105 21  
## 7 2.56 170. Bavarian a 32.8 2.13 -30.2 0.0101 21  
## 8 3.88 164. Bavarian a 29.9 2.41 -26.0 0.01292 / 5621
```

1.5 Modelo lineal

Gráficar el modelo lineal.

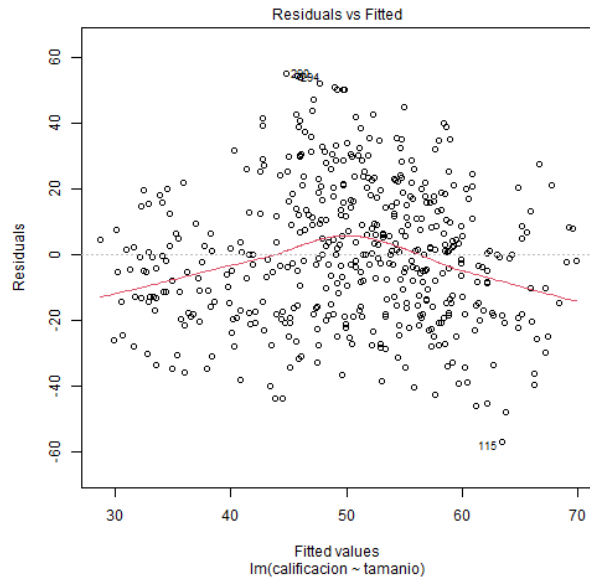
```
info_ajuste_lm %>%  
  ggplot(aes(x=tamano,y=calificacion))+  
  geom_jitter(alpha=.2)+  
  geom_line(aes(x=tamano,y=.fitted))+  
  theme_bw()
```



1.6. Supuestos

Problemas de linealidad, media no es cero, varianza no es constante.

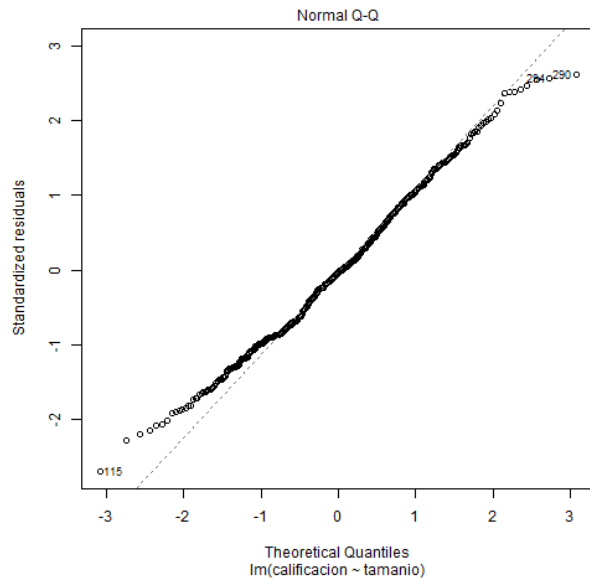
```
plot(ajuste_lm, which=1)
```



1.6. Supuestos

Normalidad

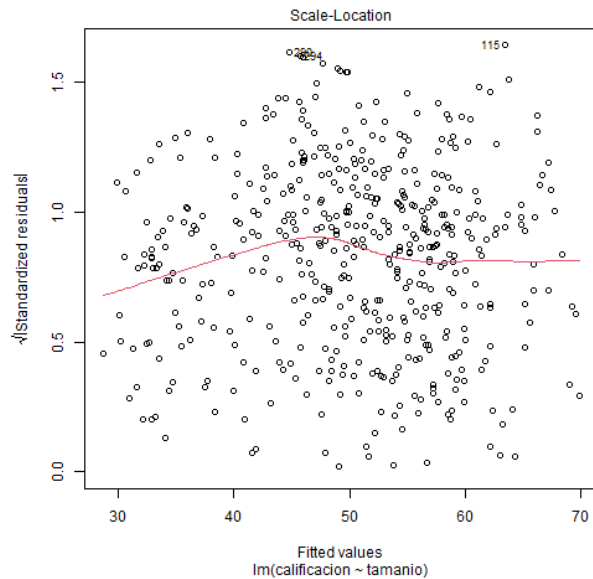
```
plot(ajuste_lm, which=2)
```



1.6. Supuestos

Homoscedasticidad

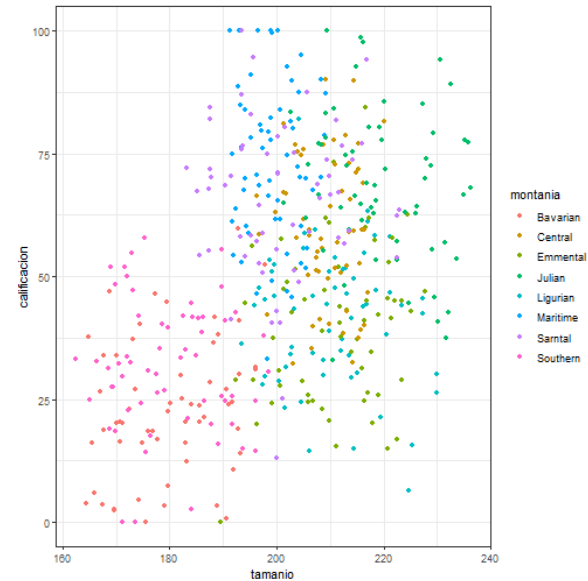
```
plot(ajuste_lm, which=3)
```



1.7. Inspeccionar

Al graficar las montañas por separado. ¿Que notan?

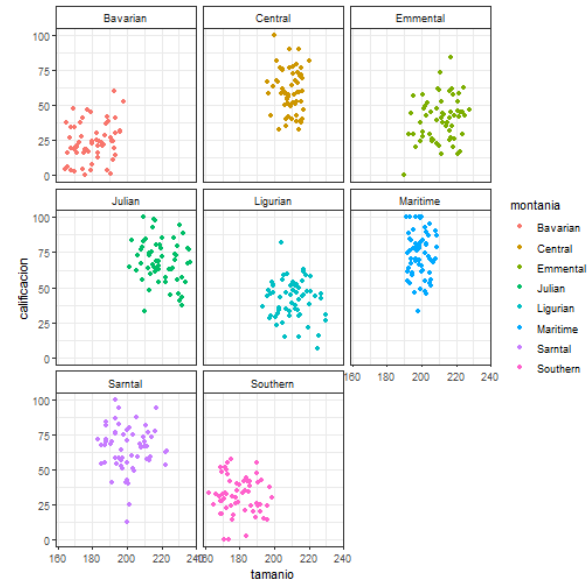
```
dragones %>%  
  ggplot(aes(x=tamaño,y=calificación,  
             colour=montania)) +  
  geom_jitter(alpha=2) +  
  theme_bw()
```



1.8. Inspeccionar

Separando por montañas. ¿Que notan?

```
dragones %>%  
  ggplot(aes(tamano, calificacion  
            colour = montania)) +  
  geom_jitter(alpha=2) +  
  facet_wrap(~ montania) +  
  theme_bw() +  
  theme(strip.background=element_
```



1.9. Efectos aleatorios

No se puede omitir la montaña , ¿pero como la incluimos?

- Como factor aleatorio.
-  Que es un factor aleatorio y como saber cuando es aleatorio

Paquetes

```
library(broom.mixed)
library(lme4)
```

Sintaxis de los efectos aleatorios (random effects):

```
ajuste_lmer <- lmer(calificacion ~ tamaño +
                    (1|montania), data = dragones)
```

Nota Que se llamen efectos aleatorios poco tiene que ver con que aleatoriedad matemática. Es confuso pero por ahora lo más sencillo es pensar en ellos como variables de **agrupamiento**.

1.9. Efectos aleatorios

Mirar nuestro nuevo modelo.

```
tidy(ajuste_lmer) %>%  
  tibble::as_tibble()
```

```
## # A tibble: 4 x 6  
##   effect    group    term          estimate std.error statistic  
##   <chr>    <chr>    <chr>          <dbl>    <dbl>    <dbl>  
## 1 fixed    <NA>    (Intercept)    43.7     17.1     2.55  
## 2 fixed    <NA>    tamaño         0.0332   0.0786   0.422  
## 3 ran_pars montania sd__(Intercept) 18.4     NA       NA  
## 4 ran_pars Residual sd__Observation 15.0     NA       NA
```

1.9. Efectos aleatorios

Mirar los ajustes del modelo.

```
glance(ajuste_lmer) %>%  
  tibble::as_tibble()
```

```
## # A tibble: 1 x 6  
##   sigma logLik   AIC   BIC REMLcrit df.residual  
##   <dbl> <dbl> <dbl> <dbl>   <dbl>     <int>  
## 1  15.0 -1996. 3999. 4016.   3991.     476
```

¿No hay valor p ?  

Lecturas sobre valor p :

- Explicación por parte del desarrollador del paquete lmer: [Douglas Bates](#)
- Compilación de recursos sobre el valor p por [Rocio Joo](#)

1.9. Valor p

A pesar de que no se recomienda usar valor p, la librería lmerTest te puede dar un valor p.

Considera que: *all the F ratios use the same denominator* - Douglas Bates

```
library(lmerTest)
ajuste_lmer <- lmer(calificacion ~ tamano +
                   (1|montania), data = dragones)
tidy(ajuste_lmer) %>%
  tibble::as_tibble()
```

```
## # A tibble: 4 x 8
##   effect    group    term          estimate std.error statistic    df p.va
##   <chr>    <chr>    <chr>          <dbl>    <dbl>    <dbl> <dbl> <d
## 1 fixed    <NA>    (Intercept)   43.7     17.1     2.55   177.  0.0
## 2 fixed    <NA>    tamano        0.0332   0.0786   0.422  473.  0.6
## 3 ran_pars montania sd__(Intercept) 18.4     NA       NA     NA   NA
## 4 ran_pars Residual sd__Observation 15.0     NA       NA     NA   NA
```

1.10. Efecto de la montaña

Ver información del ajuste: valores ajustados (.fitted), residuales (.resid)
`augment_columns` agrega los valores ajustados, los residuales y otros al data frame.

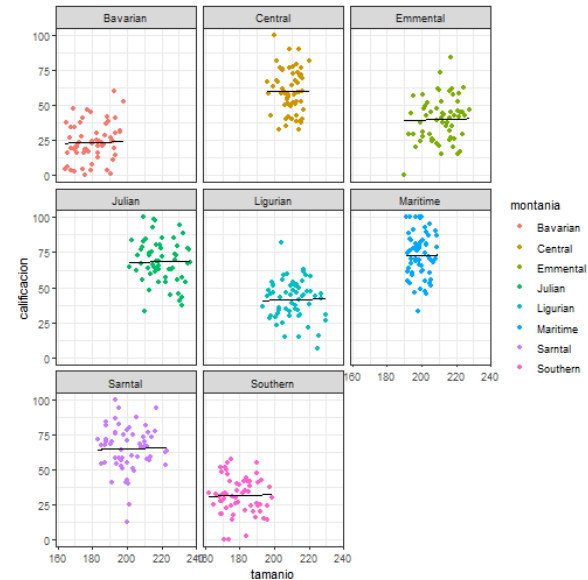
```
info_ajuste_lmer<- augment_columns(ajuste_lmer, dragones)  
info_ajuste_lmer
```

```
## # A tibble: 480 x 8  
##   calificacion tamaño montania sitio .fitted .resid .hat .cooksd  
##   <dbl> <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 16.1 166. Bavarian a 22.9 -6.78 0.0224 0.00240  
## 2 33.9 168. Bavarian a 23.0 10.9 0.0209 0.00577  
## 3 6.04 166. Bavarian a 22.9 -16.9 0.0221 0.0147  
## 4 18.8 168. Bavarian a 23.0 -4.16 0.0208 0.000836  
## 5 33.9 170. Bavarian a 23.1 10.8 0.0194 0.00524  
## 6 47.0 169. Bavarian a 23.0 24.0 0.0201 0.0270  
## 7 2.56 170. Bavarian a 23.1 -20.5 0.0195 0.0191  
## 8 3.88 164. Bavarian a 22.9 -19.0 0.0233 0.0197  
## 9 3.60 168. Bavarian a 23.0 -19.4 0.0209 0.0183  
## 10 7.36 180. Bavarian a 23.4 -16.0 0.0165 0.00980  
## # ... with 470 more rows
```

1.10. Efecto de la montaña

Veamos el efecto de las montañas en la calificación.

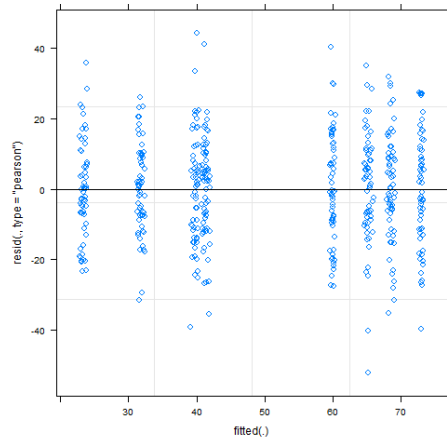
```
info_ajuste_lmer %>%  
  ggplot(aes(x=tamano, y=calificac  
             colour=montania))+  
  geom_jitter(alpha=2)+  
  facet_wrap(~ montania)+  
  geom_line(aes(x=tamano,  
                y=.fitted),  
            colour="black")+  
  theme_bw()
```



1.11. Supuestos

Al incluir montaña en el modelo, no se observan problema se linealidad, la media y varianza son constantes.

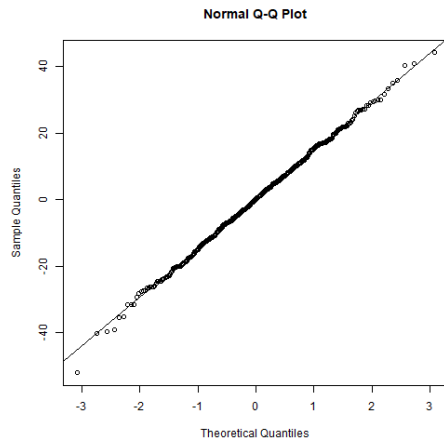
```
plot(ajuste_lmer)
```



1.11. Supuestos

Al incluir montaña en el modelo, no se ven problemas en los residuales del modelo.

```
qqnorm(resid(ajuste_lmer))  
qqline(resid(ajuste_lmer))
```



1.11. Modelo mixto

Ver ajustes del modelo mixto que incluye **montañas**

Se ve que la varianza de el sistema montañoso es 339.7. Entonces las montañas son claramente importantes porque explican mucha de la variación. Como saber cuanto explican de la variación? Se puede tomar la varianza de la montaña y dividirla por el total de la varianza.

```
summary(ajuste_lmer)
```

```
339.7/(339.7 + 223.8)
```

```
## [1] 0.6028394
```

Otro valor interesante es el AIC. Hablaremos más adelante de esto.

```
AIC(ajuste_lmer)
```

```
## [1] 3999.203
```

2.1. Modelo completo

Al incluir montaña nos hemos dado cuenta que es importante incluirla, ¿sera importante incluir sitio también?

Lo que nos lleva a preguntarnos, con los datos que tenemos como sabemos:
¿Que modelo se ajusta mejor a nuestros datos?



2.1. Efectos aleatorios anidados

En nuestros datos de dragones: la variable de **sitio** es un factor con tres niveles: un sitio a, b y c, pero existe un anidamiento (nesting) del sitio con la montaña. Los sitios no tienen significado si no le asignamos las montañas.

Para no tener que estar recordando que esto ocurre, lo mejor es crear una nueva variable anidada (nested).

```
dragones$muestra <- factor(paste0(dragones$montania,  
                                dragones$sitio))
```

2.2. Efectos aleatorios cruzados

De acuerdo a lo que vimos anteriormente, entonces no podemos poner los efectos aleatorios separados.

```
modelo_incorrecto <- lmer(calificacion ~ tamaño +  
                          (1|montania) + (1|sitio), data = dragones)
```

Este modelo trata los efectos aleatorios como **cruzados**.

Para efectos cruzados es cuando el factor aparece en más de un nivel en otros factores, un ejemplo sería si el mismo dragón aparece en más de un sistema montañoso.

2.3. Efectos aleatorios anidados

Con los datos de dragones, una manera de crear un modelo completo sería la siguiente.

```
lmer_completo <- lmer(calificacion ~ tamaño +  
                      (1|montania) + (1|muestra),  
                      data = dragones)
```

```
summary(lmer_completo)
```

Si buscan ejemplos otras maneras de escribir esto son:

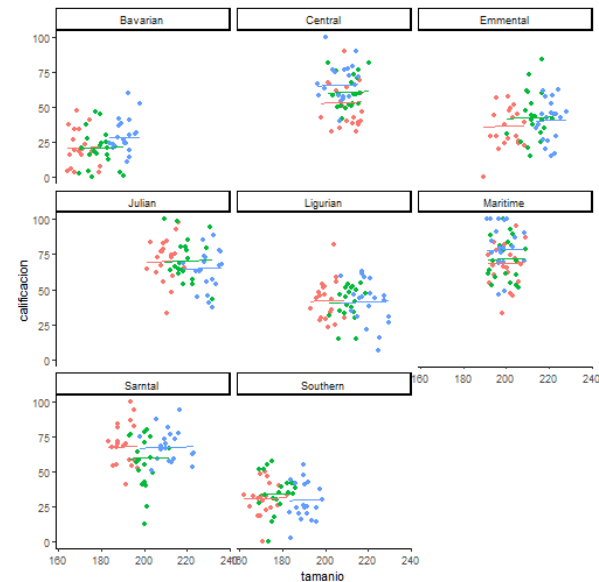
```
(1|montania/sitio)  
(1|montania) + (1|montania:sitio)
```

2.4. Modelo completo

Ahora que creamos un modelo considerando los efectos aleatorios, nos queda claro que el análisis inicial (el modelo lineal) nos daba resultados que **no eran correctos**, los dragones no son más inteligentes entre más grandes.

En el futuro podemos elegir dragones más pequeños para entrenarlos. 😊

```
ggplot(dragones, aes(x = tamaño,
  facet_wrap(~montania, nrow=3) +
  geom_point() +
  theme_classic() +
  geom_line(data = cbind(dragones
  theme(legend.position = "none")
```



3. AIC

3.1. AIC

El AIC (Akaike Information Criterion) es un modelo matemático para evaluar que tan bien se ajusta el modelo a los datos generados.

En estadística se usa mucho para comparar modelos posibles y determinar cual es el mejor modelo.

AIC se calcula a partir de:

- El numero de variables independientes que se usó para construir el modelo
- El estimado de máxima verosimilitud (maximum likelihood), (que tan bien el modelo reproduce los datos)

$$AIC = 2K - 2\ln(L)$$

Fuente: [Scribbr](#)

K es el número de variables independientes y L es el estimado de máxima verosimilitud.

El mejor modelo se identifica a partir de un AIC, que explica la mayor cantidad de variación usando el menor numero de variables independientes.

3.1. AIC

Los AIC son muy usados para evaluar la combinación completa de los predictores y el mejor modelo se considera el que tiene el menor valor de AIC

Model selection based on AICc:

	K	AICc	Delta_AICc	AICcWt	Cum.Wt	LL
combination.mod	5	1743.02	0.00	0.96	0.96	-866.45
interaction.mod	9	1749.35	6.33	0.04	1.00	-865.49
age.sex.mod	4	1760.59	17.57	0.00	1.00	-876.26
age.mod	3	1764.91	21.89	0.00	1.00	-879.43
sex.mod	3	2815.68	1072.66	0.00	1.00	-1404.82
consumption.mod	3	2820.86	1077.84	0.00	1.00	-1407.41

Fuente: [Scribbr](#)

- **K** es el numero de parametros del modelo. El default es 2, asi que cualquier modelo con un parametro dara una K de $2+1=3$.
- **AICc** es el calculo del AIC, la c viene de corregido para muestras pequeñas. Entre más pequeño el AIC mejor es el ajuste del modelo.
- **Delta AIC** es la diferencia entre los valores de AIC. Diferencias pequeñas (<2 unidades) no son consideradas significativas. Si varios modelos tienen menos de estas unidades, el mejor modelo es el que es más parsimonioso, es decir el que tiene el menor numero de parámetros
- **AICcWT**: es la proporción del total de poder predictivo dado por el set de modelos probados en el modelo.

3.2. Selección del modelo

De acuerdo a lo que vimos, es fácil omitir algunas variables pero también podríamos incluir algunas que no son importantes (lo que se conoce como **overfit**).

Para resolver este problema lo que se hace es una **selección del modelo**.

Pero, hay que ser muy cuidadosos en lo que se refiere a la selección del modelo. Para hacer una buena selección hay que tener claro cual es la **pregunta**.

En lo que respecta a biología, se puede usar la selección de modelos para revisar parámetros que no corresponden al núcleo de la pregunta.

También se **recomienda** tener 10 veces más datos que parámetros.

- **Da click aquí para mirar algunas formulas y explicacion a detalle**

3.3. Selección del modelo

De mejor a peor lo que podemos usar para comparar modelos son:

- Z-test de Wald
- t-test de Wald (en este modelo los valores tendrían que estar balanceados)
- **Prueba de razón de verosimilitud** (en inglés: Likelihood ratio tests).
Usando `anova()` o `drop1()`
- Markov chain Monte Carlo (MCMC) o un bootstramp paramétrico con intervalos de confianza

3.4. Compara modelos

En nuestro ejemplo de dragones podemos comparar

- Modelo completo.

```
lmer_completo <- lmer(calificacion ~ tamaño +  
                      (1|montania) + (1|muestra),  
                      data = dragones, REML = FALSE)
```

- Modelo reducido. **Nota** que este modelo no incluye tamaño.

```
lmer_reducido <- lmer(calificacion ~ 1 +  
                      (1|montania) + (1|muestra),  
                      data = dragones, REML = FALSE)
```

Comparación de los modelos.

```
anova(lmer_reducido, lmer_completo)
```

3.5. Compara modelos

Al comparar los modelos encontramos que no son significativamente diferentes. Es decir, **tamaño** no ayuda a explicar la calificación de los dragones.

```
anova(lmer_reducido, lmer_completo)
```

```
## Data: dragones
## Models:
## lmer_reducido: calificacion ~ 1 + (1 | montania) + (1 | muestra)
## lmer_completo: calificacion ~ tamaño + (1 | montania) + (1 | muestra)
##           npar      AIC      BIC  logLik deviance  Chisq Df Pr(>Chisq)
## lmer_reducido     4 3987.1 4003.7 -1989.5   3979.1
## lmer_completo     5 3988.8 4009.6 -1989.4   3978.8 0.2868  1     0.5923
```

3.6. Otro ejemplo

Veamos otro ejemplo.



```
library(readr)
```

Que pasa si tenemos tres modelos en mente a comparar.

```
url <- "https://raw.githubusercontent.com/MiriamLL/Prueba/master/bmi_data.csv"
download.file(url, "bmi_data.csv")
bmi_datos <- read_csv("bmi_data.csv")
```

Fuente: [Scribbr](#)

Estos datos de bebidas azucaradas, y contiene información de:

- Sexo (Female/Male)
- Edad (numérico) 
- Consumo (numérico) 
- BMI (índice de masa corporal)

3.7. Construir modelos

Primer modelo

```
mod_edad <- lm(bmi ~ age, data = bmi_datos)
```

Segundo modelo

```
mod_sexo <- lm(bmi ~ sex, data = bmi_datos)
```

Tercer modelo

```
mod_consumo <- lm(bmi ~ consumption, data = bmi_datos)
```


3.8. Usar AICmodavg

Comparar modelos

```
library(AICcmodavg)
```

```
modelos <-list(mod_edad, mod_sexo, mod_consumo)  
modelos_nombres<-c('mod_edad', 'mod_sexo', 'mod_consumo')
```

La edad parece ser de las variables más importantes para explicar índice de masa corporal.

```
aictab(cand.set = modelos, modnames = modelos_nombres)
```

```
##  
## Model selection based on AICc:  
##  
##           K      AICc Delta_AICc AICcWt Cum.Wt      LL  
## mod_edad   3 1764.91      0.00      1      1 -879.43  
## mod_sexo   3 2815.68    1050.77      0      1 -1404.82  
## mod_consumo 3 2820.86    1055.96      0      1 -1407.41
```

3.9. Comparar muchos modelos

Pero resulta evidente que habrá que incluir la combinación de sexo y edad.

```
mod_sexo_edad <- lm(bmi ~ age + sex, data = bmi_datos)
```

Incluir la combinación de sexo, edad y consumo de bebidas azucaradas.

```
mod_combinacion <- lm(bmi ~ age + sex + consumption, data = bmi_datos)
```

Finalmente probar la interacción entre estas variables.

```
mod_interaccion <- lm(bmi ~ age*sex*consumption, data = bmi_datos)
```

Con todos los modelos se puede crear una lista y un vector con los nombres de los modelos para poder identificarlos.

```
modelos <- list(mod_edad, mod_sexo, mod_consumo, mod_sexo_edad, mod_comb:  
modelos_nombres <- c('mod_edad', 'mod_sexo', 'mod_consumo', 'mod_sexo_edad
```

3.9. Comparar muchos modelos

Responder en el chat  En base al AIC ¿Cual es el mejor modelo?

```
aictab(cand.set = modelos, modnames = modelos_nombres)
```

```
##  
## Model selection based on AICc:  
##  
##           K      AICc Delta_AICc AICcWt Cum.Wt      LL  
## mod_combinacion 5 1743.02      0.00  0.96  0.96 -866.45  
## mod_interaccion 9 1749.35      6.33  0.04  1.00 -865.49  
## mod_sexo_edad   4 1760.59     17.57  0.00  1.00 -876.26  
## mod_edad        3 1764.91     21.89  0.00  1.00 -879.43  
## mod_sexo        3 2815.68    1072.66  0.00  1.00 -1404.82  
## mod_consumo     3 2820.86    1077.84  0.00  1.00 -1407.41
```

```
mod_combinacion <- lm(bmi ~ age + sex + consumption, data = bmi_datos)
```

3.10. MASS

Otra cosa que podemos hacer es pedirle a R que construya los modelos con base a la información que le damos.

```
bmi_mod_comp <- lm(bmi ~., data = bmi_datos)
```

Nota tu dataframe solo debe incluir las columnas de las variables que te interesa incluir.

Un paquete que también te permite comparar modelos es **MASS**.

```
library(MASS)
```

La ventaja de este paquete es que la función `stepAIC` te permite elegir: `direction = c("both", "backward", "forward")`.

```
step.model <- stepAIC(bmi_mod_comp, direction = "both",  
                    trace = FALSE)
```

- [Da click aquí la explicación de Stepwise regression a detalle](#)

3.10. MASS

De entrada ya nos dice cual fue el mejor modelo.

```
summary(step.model)
```

```
##
## Call:
## lm(formula = bmi ~ sex + age + consumption, data = bmi_datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.67918 -1.18845  0.00031  1.22444  2.48676
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.78422    0.26730   66.534 < 2e-16 ***
## sexMale      -0.28402    0.12392   -2.292  0.0223 *
## age          0.16703    0.00272   61.398 < 2e-16 ***
## consumption  1.35082    0.30323    4.455 1.04e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.374 on 496 degrees of freedom
```

3.11. drop1

Otra manera de comparar modelos es usando `drop1`

Igualmente se puede crear el modelo completo y la función `drop1` te da los resultados similares como reducir tu modelo.

Los resultados se pueden interpretar en que modelo sin el termino es significativamente diferente del modelo con tal termino. Y de nuevo ofrece un AIC que entre menor es el mejor modelo.

```
drop1(bmi_mod_comp, test="F") # Tambien llamado 'type II' anova
```

```
## Single term deletions
##
## Model:
## bmi ~ sex + age + consumption
##           Df Sum of Sq    RSS    AIC   F value    Pr(>F)
## <none>                936.9  321.96
## sex           1         9.9  946.8  325.23    5.2529  0.02233 *
## age           1    7120.2 8057.1 1395.85 3769.6720 < 2.2e-16 ***
## consumption  1         37.5  974.3  339.57   19.8446 1.039e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.12. Usando dredge

La función `dredge` del paquete MuMIN genera una tabla de selección de modelos usando las combinaciones de los efectos fijos en el modelo global.

```
library(MuMIn)
options(na.action = "na.fail")
```

- **AICc**: Valores AIC con corrección
- **delta**: diferencia entre modelos. Por convención $AIC < 2$ se consideran modelos aceptables.
- **weight**: Peso del modelo.

Todos estos valores pueden reportarse.

```
dredge(bmi_mod_comp)
```

3.13. Usando dredge

Veamos con los datos de dragones.

```
library(MuMIn)
options(na.action = "na.fail")
dragones$muestra <- factor(paste0(dragones$montania,
                                dragones$sitio))
lmer_completo <- lmer(calificacion ~ tamano +
                    (1|montania) + (1|muestra),
                    data = dragones, REML=FALSE)
dredge(lmer_completo)
```

Nota REML es el restricted maximum likelihood.

Cuando se prueba el modelo regularmente se pone REML=FALSE.

Esto es para que pueda crear modelos más sencillos a la hora de probar las combinaciones.

Pero una vez sepamos cual es el mejor modelo, podemos poner de nuevo REML=TRUE para obtener los resultados.

3.14. Usando dredge

El mejor modelo incluye todos los parámetros.

```
lmer_completo <- lmer(calificacion ~ tamaño +  
                      (1|montaña) + (1|muestra),  
                      data = dragones, REML=TRUE)
```

Ver estimados del modelo.

```
summary(lmer_completo)
```

Cuando más de un modelo se encuentra dentro de las 2 unidades AIC, depende de tu criterio de selección.

Una opción es primero reportar que varios modelos estuvieron dentro de las dos unidades AIC, y que se usaron todos los parámetros de los modelos con menos de 2 unidades AIC.

3.16. Usando dredge

Para reportar en la selección del modelo.

```
(Intrc)  taman df    logLik  AICc delta weight
1   50.39                4 -1989.527 3987.1  0.00  0.706
2   39.09 0.05611      5 -1989.384 3988.9  1.76  0.294
```

Para reportar en los estimados del modelo seleccionado.

Fixed effects:

```
          Estimate Std. Error      df t value Pr(>|t|)
(Intercept) 40.06668   21.86373  90.35454   1.833   0.0702 .
tamaño      0.05126    0.10368  94.18716   0.494   0.6222
```

4. Seguir aprendiendo

En la página de [CRAN Task Viewer](#) pueden encontrar referencias de paquetes de acuerdo al modelo que quieran crear.

- Base R provee: `lm()` y `glm()`. Ya usamos `lm` en la [clase modelos lineales simples](#)
- [lme4](#) provee funciones para realizar modelos generalizados mixtos lineares o no lineares (GLMM). Usamos este paquete en esta clase. Si quieren ver tutoriales de como funciona les recomiendo [Video explicando Teoría: Kyle Tomlinson](#).
- [nlme](#) provee funciones para realizar modelos lineares y no lineares con efectos mixtos (GLM)
- [mgvc](#) provee funciones para realizar modelos aditivos (Generalized Additive Mixed Models: GAMMS) [gms introduction to generalized additive models with R and mgvc](#)

Otros recursos

- [Usar Machine learning](#)
- [Usar estadística bayesiana](#)

Contacto

Resumiendo

- Modelos mixtos
- AIC

Para dudas, comentarios y sugerencias:

- Escríbeme a miriamjlerma@gmail.com

Este material está accesible y se encuentra en
mi [github](#) y mi [página](#)

 **Volver**