Modelos Mixtos

Paula Vargas Pellicer

05/04/2022

Explora los datos

Nos vamos a centrar en un sistema de estudio ficticio, los dragones, para que no tengamos que distraernos demasiado con los detalles de este ejemplo. Imagina que decidimos entrenar dragones y, por lo tanto, salimos a las montañas y recopilamos datos sobre la inteligencia del dragón (testScore) como requisito previo. Tomamos muestras de individuos con un rango de longitudes corporales en tres sitios en ocho cadenas montañosas diferentes.

```
load("~/Downloads/dragons.RData")
head(dragons)
```

Digamos que queremos saber cómo la longitud del cuerpo de los dragones afecta sus puntajes en las pruebas

```
hist(dragons$testScore)
```

Un truquito

Puedes estandarizar tus variables explicativas antes de continuar para que tengan una media de cero y una desviación estándar de uno. Garantiza que los coeficientes estimados estén todos en la misma escala, lo que facilita la comparación de los tamaños del efecto. Puedes usar scale() para hacer eso:

```
dragons$bodyLength2 <- scale(dragons$bodyLength, center = TRUE, scale = TRUE)
hist(dragons$bodyLength)
```

```
hist(dragons$bodyLength2)
```

Ajusta los datos en un análisis

Una forma de analizar estos datos sería ajustar un modelo lineal a todos nuestros datos, ignorando los sitios y las cadenas montañosas.

```
basic.lm <- lm(testScore ~ bodyLength2, data = dragons)
summary(basic.lm)

library(tidyverse)
```

```
(prelim_plot <- ggplot(dragons, aes(x = bodyLength, y = testScore)) +
  geom_point() +
  geom_smooth(method = "lm"))
```

Checamos los supuestos

```
plot(basic.lm, which = 1)

plot(basic.lm, which = 2)
```

Recolectamos múltiples muestras de ocho cadenas montañosas. Es perfectamente plausible que los datos de cada cadena montañosa sean más similares entre sí que los datos de diferentes cadenas montañosas.

```
boxplot(testScore ~ mountainRange, data = dragons)

(colour_plot <- ggplot(dragons, aes(x = bodyLength, y = testScore, colour = m
ountainRange)) +
  geom_point(size = 2) +
  theme_classic() +
  theme(legend.position = "none"))
```

Ejecutamos múltiples análisis

Como solución podríamos ejecutar muchos análisis separados y ajustar una regresión para cada una de las cadenas montañosas.

```
(split_plot <- ggplot(aes(bodyLength, testScore), data = dragons) +
  geom_point() +
  facet_wrap(~ mountainRange) +   xlab("length") +
  ylab("test score"))
```

Son ocho análisis. Ah, pero también tenemos diferentes sitios en cada cadena montañosa, y tampoco son independientes... Así que podríamos realizar un análisis para cada sitio en cada cadena por separado.

Para hacer lo anterior, tendríamos que estimar un parámetro de pendiente e intercepto para cada regresión. ¡Eso son dos parámetros, tres sitios y ocho cadenas montañosas, lo que significa 48 estimaciones de parámetros (2 x 3 x 8 = 48)! Además, el tamaño de la muestra para cada análisis sería de solo 20 (dragones por sitio).

Esto presenta problemas: no solo estamos disminuyendo enormemente el tamaño de nuestra muestra, sino que también estamos aumentando las posibilidades de un error de tipo I (donde se rechaza falsamente la hipótesis nula) al realizar comparaciones múltiples.

Modifiquemos el modelo actual

Queremos usar todos los datos, pero ten en cuenta los datos que provienen de diferentes cadenas montañosas (pongamos los sitios en espera por un segundo para simplificar las cosas).

Agrega la cordillera como un efecto fijo a nuestro basic.lm

```
mountain.lm <- lm(testScore ~ bodyLength2 + mountainRange, data = dragons)
summary(mountain.lm)
```

Ahora la longitud del cuerpo no es significativa. Pero pensemos en lo que estamos haciendo aquí por un segundo: El modelo anterior está estimando la diferencia en los puntajes de las pruebas entre las cadenas montañosas. Pero no estamos interesados en cuantificar los puntajes de las pruebas para cada cadena montañosa específica: solo queremos saber si la longitud del cuerpo afecta los puntajes de las pruebas y simplemente queremos controlar la variación que proviene de las cadenas montañosas. Estos son los "factores aleatorios".

Modelos de efectos mixtos

Un modelo mixto es una buena opción aquí: nos permitirá usar todos los datos que tenemos (mayor tamaño de muestra) y dar cuenta de las correlaciones entre los datos provenientes de los sitios y las cadenas montañosas. También estimaremos menos parámetros y evitaremos problemas con comparaciones múltiples que encontraríamos al usar regresiones separadas.

Vamos a trabajar en lme4, así que carga el paquete (o use install.packages si no tiene lme4 en tu computadora).

```
library(lme4)
```

Tenemos una variable de respuesta, el puntaje de la prueba y estamos tratando de explicar parte de la variación en el puntaje de la prueba ajustando la longitud del cuerpo como un efecto fijo. Pero la variable de respuesta tiene alguna variación residual (es decir, una variación no explicada) asociada con las cadenas montañosas. Mediante el uso de efectos aleatorios, estamos modelando esa variación no explicada a través de la varianza.

Ten en cuenta que nuestra pregunta cambia ligeramente aquí: si bien todavía queremos saber si existe una asociación entre la longitud del cuerpo del dragón y el puntaje de la prueba, queremos saber si esa asociación existe después de controlar la variación en las cadenas montañosas.

Ajustaremos el efecto aleatorio usando la sintaxis (1|variable):

```
mixed.lmer <- lmer(testScore ~ bodyLength2 + (1|mountainRange), data = dragon
s)
summary(mixed.lmer)
```

Una vez que tomamos en cuenta las cadenas montañosas, la longitud del cuerpo del dragón en realidad no explica las diferencias en los puntajes de las pruebas. Observa cómo la estimación del modelo es más pequeña que su error asociado. Eso significa que el efecto, o pendiente, no se puede distinguir de cero.

Ten en cuenta que el efecto aleatorio de la cadena montañosa está destinado a capturar todas las influencias de las cadenas montañosas en los puntajes de las pruebas del dragón, ya sea que observemos esas influencias explícitamente o no, ya sean grandes o pequeñas, etc. Podrían ser muchas, muchas influencias minúsculas que, cuando se combinan, afectan los puntajes de las pruebas y eso es lo que esperamos controlar.

Podemos ver la varianza para mountainRange = 339.7. Las cadenas montañosas son claramente importantes: explican muchas variaciones. ¿Cómo lo sabemos? Podemos tomar la varianza de mountainRange y dividirla por la varianza total:

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

Entonces, las diferencias entre las cadenas montañosas explican ~60% de la varianza que "sobró" después de la varianza explicada por nuestros efectos fijos.

Checar los supuestos

```
plot(mixed.lmer)

qqnorm(resid(mixed.lmer))
qqline(resid(mixed.lmer))
```

Variables anidadas explícitas

Nuestra variable de sitio es un factor de tres niveles, con sitios llamados a, b y c. La anidación del sitio dentro de la cordillera está implícita: nuestros sitios no tienen sentido sin estar asignados a cadenas montañosas específicas, es decir, no hay nada que vincule el sitio b de la cordillera bávara con el sitio b de la cordillera central. Para evitar futuras confusiones, debemos crear una nueva variable que esté explícitamente anidada. Llamémoslo muestra:

```
dragons <- within(dragons, sample <- factor(mountainRange:site))
```

De esta manera tenemos 24 muestras (8 cadenas montañosas x 3 sitios) y no solo 3: nuestra muestra es un factor de 24 niveles y deberíamos usar eso en lugar de usar sitio en nuestros modelos: cada sitio pertenece a una cadena montañosa específica.

En resumen: para efectos aleatorios anidados, el factor aparece SOLO dentro de un nivel particular de otro factor (cada sitio pertenece a una cadena montañosa específica y solo a esa cadena); para efectos cruzados, un factor dado aparece en más de un nivel de otro factor (dragones que aparecen dentro de más de una cadena montañosa). Ósea si no están anidados, están cruzados.

Entonces, fíjate como NO debemos hacer el modelo cruzado, en este caso:

```
mixed.WRONG <- lmer(testScore ~ bodyLength2 + (1|mountainRange) + (1|site), d
ata = dragons)
summary(mixed.WRONG)
```

Pero podemos ajustar un nuevo modelo, uno que tenga en cuenta tanto las diferencias entre las cadenas montañosas como las diferencias entre los sitios dentro de esas cadenas montañosas utilizando nuestra variable de muestra.

Nuestra pregunta se ajusta ligeramente nuevamente: ¿Existe una asociación entre la longitud del cuerpo y la inteligencia en los dragones después de controlar la variación en las cadenas montañosas y los sitios dentro de las cadenas montañosas?

```
mixed.lmer2 <- lmer(testScore ~ bodyLength2 + (1|mountainRange) + (1|sample),
data = dragons)
summary(mixed.lmer2)
```

Aquí, estamos tratando de tener en cuenta todas las influencias del nivel de la cordillera y del sitio y esperamos que nuestros efectos aleatorios hayan absorbido todas estas influencias para que podamos controlarlas en el modelo.

También podrías usar la siguiente sintaxis, es cuestión de gusto.

(1|cordillera/sitio) o (1|cordillera) + (1|cordillera:sitio)

Grafiquemos esto de nuevo: siempre es útil visualizar lo que está pasando. Deberías poder ver ocho cadenas montañosas con tres sitios (puntos de diferentes colores) dentro de ellas, con una línea que pasa por cada sitio.

```
(mm_plot <- ggplot(dragons, aes(x = bodyLength, y = testScore, colour = site)
) +
      facet_wrap(~mountainRange, nrow=2) +         geom_point(alpha = 0.5) +
      theme_classic() +
      geom_line(data = cbind(dragons, pred = predict(mixed.lmer2)), aes(y = p
red), size = 1) +        theme(legend.position = "none",
            panel.spacing = unit(2, "lines"))
)
```

Pendientes aleatorias

Es posible que hayas notado que todas las líneas en la figura anterior son paralelas: eso se debe a que, hasta ahora, solo hemos ajustado modelos de intersección aleatoria. Un modelo de intersección aleatoria permite que la intersección varíe para cada nivel de los efectos aleatorios, pero mantiene constante la pendiente entre ellos. Entonces, en nuestro caso, usar este modelo significa que esperamos que los dragones en todas las cadenas montañosas muestren la misma relación entre la longitud del cuerpo y la inteligencia (pendiente fija), aunque reconocemos que, para empezar, algunas poblaciones pueden ser más inteligentes o más tontas (intersección aleatoria).

Ahora, en ecología asumimos más a menudo que no todas las poblaciones mostrarían exactamente la misma relación, por ejemplo, si tus sitios/poblaciones de estudio están muy separados y tienen algunas diferencias ambientales, genéticas, etc. relativamente importantes. Por lo tanto, a menudo queremos ajustar un modelo de pendiente e intersecciones aleatorias.

Tal vez los dragones en una cadena montañosa muy fría versus muy cálida hayan desarrollado diferentes formas corporales para la conservación del calor y, por lo tanto, pueden ser inteligentes incluso si son más pequeños que el promedio.

Solo necesitamos hacer un cambio en nuestro modelo para permitir pendientes aleatorias, así como la intercepción, y eso es agregar la variable fija en los efectos aleatorios:

```
mixed.ranslope <- lmer(testScore ~ bodyLength2 + (1 + bodyLength2|mountainRan
ge/site), data = dragons)

summary(mixed.ranslope)
```

Aquí, decimos, modelemos la inteligencia de los dragones en función de la longitud del cuerpo, sabiendo que las poblaciones tienen diferentes líneas de base de inteligencia y que la relación puede variar entre las poblaciones.

Veamos eso con un gráfico rápido (trazaremos las predicciones con más detalle en la siguiente sección). ¿Te das cuenta de que las pendientes de los diferentes sitios y cadenas montañosas ya no son paralelas?

```
(mm_plot <- ggplot(dragons, aes(x = bodyLength, y = testScore, colour = site)
) +
      facet_wrap(~mountainRange, nrow=2) +

      geom_point(alpha = 0.5) +
      theme_classic() +
      geom_line(data = cbind(dragons, pred = predict(mixed.ranslope)), aes(y
= pred), size = 1) +        theme(legend.position = "none",
            panel.spacing = unit(2, "lines"))
)
```

Listo, hemos ajustado modelos mixtos de intersección aleatoria y pendientes aleatorias, y hemos aprendido cómo tener en cuenta los efectos aleatorios jerárquicos y cruzados. No tener en cuenta la correlación en los datos podría llevar a resultados engañosos: parecía que la longitud del cuerpo afectaba el puntaje de la prueba hasta que tomamos en cuenta la variación proveniente de las cadenas montañosas. Ahora podemos ver que la longitud del cuerpo no influye en los puntajes de las pruebas, por lo que podemos elegir dragones más pequeños para cualquier entrenamiento futuro.

Trazado de predicciones del modelo

A menudo vas a querer visualizar tu modelo como una línea de regresión con algún error alrededor, tal como lo harías con un modelo lineal simple. Sin embargo, las opciones de estadísticas de ggplot2 no están diseñadas para estimar correctamente los objetos del modelo de efectos mixtos, por lo que usaremos el paquete ggeffects para ayudarnos a dibujar los gráficos.

```
library(ggeffects)

# Extrae la predicción del modelo
pred.mm <- ggpredict(mixed.lmer2, terms = c("bodyLength2"))

(ggplot(pred.mm) +
   geom_line(aes(x = x, y = predicted)) +          # pendiente
   geom_ribbon(aes(x = x, ymin = predicted - std.error, ymax = predicted + st
d.error),
               fill = "lightgrey", alpha = 0.5) +  # error
   geom_point(data = dragons,                      # datos (escalados)
              aes(x = bodyLength2, y = testScore, colour = mountainRange)) +
   labs(x = "Largo cuerpo (indexado)", y = "Prueba") +
   theme_minimal()
)
```

¿Qué sucede si quieres visualizar cómo varían las relaciones según los diferentes niveles de efectos aleatorios? Puedes especificar type = "re" (para "efectos aleatorios") en la función ggpredict() y agregar el nombre del efecto aleatorio al argumento de los términos.

```
ggpredict(mixed.lmer2, terms = c("bodyLength2", "mountainRange"), type = "re"
) %>%
   plot() +
   labs(x = "Largo cuerpo", y = "Prueba") +
   theme_minimal()
```

Otra forma de visualizar los resultados del modelo mixto, si estás interesado en mostrar la variación entre los niveles de tus efectos aleatorios, es trazar la desviación de la estimación general del modelo para intersecciones y pendientes, si tienes un modelo de pendiente aleatoria:

```
library(sjPlot)

(re.effects <- plot_model(mixed.ranslope, type = "re", show.values = TRUE))

summary(mixed.ranslope)
```

¡Cuidado aquí! Los valores que ves NO son valores reales, sino la diferencia entre el valor general de intersección o pendiente que se encuentra en el resumen del modelo y la estimación para este nivel específico de efecto aleatorio. Por ejemplo, la relación para los

dragones en la cordillera Marítima tendría una pendiente de (-2.91 + 0.67) = -2.24 y una intersección de (20.77 + 51.43) = 72.20.

Tablas

Para modelos de lme4, usa lo siguiente

```
library(stargazer)
```

stargazer tiene muchos recursos que puedes encontrar en línea, por lo que no entraré en demasiados detalles.

Aquí hay un ejemplo rápido: simplemente ingresa el nombre de tu modelo, en este caso mixed.lmer2 en la función. Configura a "texto" para que puedas ver la tabla en tu consola.

```
stargazer(mixed.lmer2, type = "text",
          digits = 3,
          star.cutoffs = c(0.05, 0.01, 0.001),
          digit.separator = "")
```

Valores -p

Si insistes en tener un valor p para tu modelo, prueba lo siguiente

```
full.lmer <- lmer(testScore ~ bodyLength2 + (1|mountainRange) + (1|sample),
                  data = dragons, REML = FALSE)
reduced.lmer <- lmer(testScore ~ 1 + (1|mountainRange) + (1|sample),
                         data = dragons, REML = FALSE)
anova(reduced.lmer, full.lmer)
```