

```
library(rethinking)
```

11.2.2. Negative binomial (gamma-Poisson) models

Por lo general, hay una gran cantidad de inexplicables variación en los modelos de Poisson, que surge de influencias que varían de un caso a otro, generando variación en las λ verdaderas. Ignorar esta variación, puede causar confusiones al igual que en los modelos binomiales. Entonces, una extensión muy común de los GLM de Poisson es intercambiar la distribución de Poisson por algo llamado distribución binomial negativa. En realidad, se trata de una distribución de Poisson disfrazada y, a veces, también se la denomina distribución gamma-Poisson por esta razón. Es un Poisson disfrazado, porque es una mezcla de diferentes distribuciones de Poisson. Trabajaremos con mezclas en el próximo capítulo.

11.2.3 Ejemplo de *exposure* y *offset*

El parámetro λ se lo suele considerar una tasa y darnos cuenta de esto nos permite hacer modelos de Poisson para los cuales la exposición varía. Supongamos, por ejemplo, que un monasterio vecino realiza totales semanales de manuscritos completos mientras que su monasterio realiza totales diarios, ¿cómo podría analizar ambos en el mismo modelo, dado que los recuentos se agregan en diferentes períodos de tiempo, diferentes exposiciones?

Implícitamente, λ es igual a un número esperado de eventos, μ , por unidad de tiempo o distancia, τ . Entonces:

$$y_i = \text{Poisson}(\lambda_i)$$
$$\log \lambda_i = \log \frac{\mu_i}{\tau_i} = \alpha + \beta_i$$

Y lo podemos reescribir como:

$$\log \mu_i = \log \tau_i + \alpha + \beta_i$$

Donde los valores de τ son llamados **exposures**. Cuando la exposición varía entre los casos, entonces τ_i hace el importante trabajo de escalar correctamente el número esperado de eventos para cada caso i .

Dado que una distribución de Poisson supone que la tasa de eventos es constante en el tiempo, todo lo que necesitamos hacer es agregar el logaritmo de la exposición al modelo lineal. El término que agregamos es generalmente llamado offset.

Código Siguiendo con el ejemplo de monasterio, supon que la tasa verdadera es $\tau = 1.5$ manuscritos por día.

```
num_days <- 30
y <- rpois( num_days , 1.5 )
```

Suponga que la tarifa diaria en el nuevo monasterio es en realidad $\lambda = 0,5$ manuscritos por día. Simulamos datos semanalmente, simplemente multiplicamos este promedio por 7:

```
num_weeks <- 4
y_new <- rpois( num_weeks , 0.5*7 )
```

Ahora agridamos el logaritmo al *exposure* y formamos un dataframe:

```
y_all <- c( y , y_new )
exposure <- c( rep(1,30) , rep(7,4) )
monastery <- c( rep(0,30) , rep(1,4) )
d <- data.frame( y=y_all , days=exposure , monastery=monastery )
```

Para ajustar el modelo y estimar la tasa de producción de manuscritos en cada monasterio, simplemente calculamos el logaritmo de cada exposición y luego incluía esa variable en el modelo lineal:

```

# compute the offset
d$log_days <- log( d$days )
# fit the model
m11.12 <- quap(
  alist(
    y ~ dpois( lambda ),
    log(lambda) <- log_days + a + b*monastery,
    a ~ dnorm( 0 , 1 ),
    b ~ dnorm( 0 , 1 )
  ), data=d )

```

Para calcular las distribuciones posteriores de λ en cada monasterio, tomamos muestras de la parte posterior y luego solo usamos el modelo lineal, pero ahora sin el *offset*, el cual no se vuelve a utilizar en la predicciones, porque los parámetros ya están en la escala diaria, para ambos monasterios.

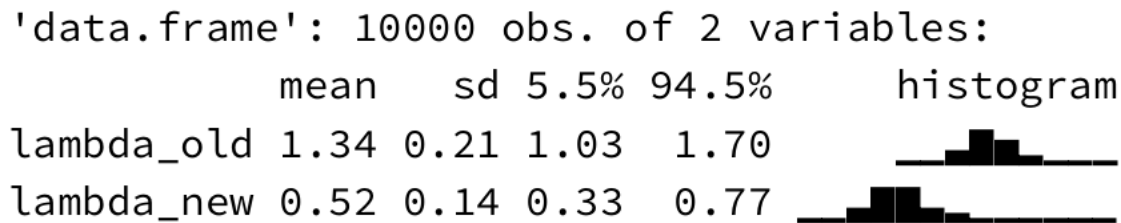


Figure 1: Figura 1. Diferencia entre modelos

El segundo monasterio sólo produce cerca de medio manuscrito por día.

11.3 Multinomial y categorical models

Cuando más de dos tipos de eventos sin orden son posibles, y la probabilidad de cada tipo de evento es constante en cada una de las pruebas, entonces la distribución de entropía máxima es la *multinomial distribution*. La binomial es un caso especial de esta distribución. Si hay K tipos de eventos con probabilidades p_1, \dots, p_K , entonces la probabilidad de los y_1, \dots, y_K eventos observados de cada tipo, de los n pruebas totales es:

$$Pr(y_1, \dots, y_K | n, p_1, \dots, p_K) = \frac{n!}{\prod_i y_i!} \prod_{i=1}^K P_i^{y_i}$$

Estos modelos también pueden ser llamados *categorical regression*. En ML también son conocidos como *maximun entropy clasifier*.

Construir MLG's de una verosimilitud multinomial es complejo, dado que mientras el tipo de eventos se multiplica, también los modelos.

Hay dos diferentes formas de construir este tipo de verosimilitudes. La primera está basada directamente en la verosimilitud multinomial, y la segunda transforma la verosimilitud multinomial en una serie de verosimilitudes de Poisson.

La relación convencional y natural en este contexto es el *multinomial logit*, también conocido como función *softmax*

$$Pr(k | s_1, s_2, \dots, s_K) = \frac{\exp(s_k)}{\sum_{i=1}^K \exp(s_i)}$$

En un MLG multinomial, se necesitan $K - 1$ modelos lineales para K tipos de eventos. Uno de las salidas es elegido como "pivote" y los demás son modelados en relación a este.

Existen dos tipos básicos para este tipo de modelos: (1) los predicadores tienen diferentes valores para los diferentes valores de la salida y (2) los parámetros son distintos para cada valor de la salida. El primer caso es útil cuando cada tipo de evento tiene sus propios rasgos cuantitativos, y se quiere estimar la asociación entre esos rasgos y la probabilidad de que cada evento aparezca en los datos. El segundo es útil cuando se está interesado en los *features* de cierta entidad que produce cada evento.

11.1.3 Predictors matched to outcomes

Por ejemplo supongamos que estás modelando la elección de carrera para un conjunto de adultos jóvenes. Una de los predictores relevantes es el *expected income*. En este caso el parámetro $\beta_i \text{income}$ aparece en cada modelo lineal, esto para conocer el impacto que esta variable tiene en la selección de carrera.

Pero el diferente ingreso multiplica $\beta_i \text{income}$ en cada modelo lineal

Estos diferentes ingresos se usan para asignar un *score*

```
# simulate career choices among 500 individuals
N <- 500                # number of individuals
income <- c(1,2,5)      # expected income of each career
score <- 0.5*income      # scores for each career, based on income
# next line converts scores to probabilities
p <- softmax(score[1],score[2],score[3])

# now simulate choice
# outcome career holds event type values, not counts
career <- rep(NA,N)     # empty vector of choices for each individual
# sample chosen career for each individual
set.seed(34302)
for ( i in 1:N ) career[i] <- sample( 1:3 , size=1 , prob=p )
```

Ajustamos el modelo usando la verosimilitud `dcategorical`, la cual es una regresión logística multinomial, la cual funciona cuando la variable de salida del modelo contiene un evento individual en cada renglón. Para convertir en probabilidades los `scores` utilizamos `softmax` la cual se presentó previamente.

```

code_m11.13 <- "
data{
  int N; // number of individuals
  int K; // number of possible careers
  int career[N]; // outcome
  vector[K] career_income;
}
parameters{
  vector[K-1] a; // intercepts
  real<lower=0> b; // association of income with choice
}
model{
  vector[K] p;
  vector[K] s;
  a ~ normal( 0 , 1 );
  b ~ normal( 0 , 0.5 );
  s[1] = a[1] + b*career_income[1];
  s[2] = a[2] + b*career_income[2];
  s[3] = 0; // pivot
  p = softmax( s );
  career ~ categorical( p );
}
"

```

Configuramos el modelo con `stan`

```

dat_list <- list( N=N , K=3 , career=career , career_income=income )
m11.13 <- stan( model_code=code_m11.13 , data=dat_list , chains=4 )
precis( m11.13 , 2 )

```

	mean	sd	5.5%	94.5%	n_eff	Rhat
a[1]	0.06	0.21	-0.31	0.37	423	1
a[2]	-0.49	0.38	-1.19	0.04	435	1
b	0.27	0.19	0.02	0.61	460	1

Ten en cuenta que interpretar los coeficientes de este modelo es extraordinariamente difícil de interpretar. Dato que los parámetros son calculados en función de a un particular valor de salida, este caso, uno de los tres ingresos propuestos, estos pueden ser negativos o positivos, dependiendo del contexto. Si embargo en este ejemplo es claro que el ingreso `b` es positivo, pero no es claro que tan grande es el efecto que tiene este en el modelo.

Para poder visualizar mejor el efecto que tiene esta variable en la decisión de elección de carrera, multiplicamos por dos el ingreso `b`.

```

post <- extract.samples( m11.13 )


# set up logit scores
s1 <- with( post , a[,1] + b*income[1] )
s2_orig <- with( post , a[,2] + b*income[2] )
s2_new <- with( post , a[,2] + b*income[2]*2 )

# compute probabilities for original and counterfactual
p_orig <- sapply( 1:length(post$b) , function(i)
  softmax( c(s1[i],s2_orig[i],0) ) )
p_new <- sapply( 1:length(post$b) , function(i)
  softmax( c(s1[i],s2_new[i],0) ) )

# summarize
p_diff <- p_new[2,] - p_orig[2,]
precis( p_diff )

```

```

'data.frame': 4000 obs. of 1 variables:
  mean    sd 5.5% 94.5% histogram
p_diff 0.13 0.09 0.01 0.29 

```

Entonces en promedio aumenta, 13 de probabilidad de elegir cierta carrera, cuando los ingresos se duplican.