Carga de librerías para el apartado en R

```r
# Directorio de trabajo
setwd("~/Actuaría/Maestría/3er. Semestre/GLM/Tareas/Tarea 2")
# Para la carga de datos
library(readxl)
# Para usar %>%
library(dplyr)
# Una alternatica para las pruebas en tablas de contingencia
library(coin)
# Para la función logit
library(LaplacesDemon)
# Para gráficos monitos
library(ggplot2)
# Para meter varios gráficos en uno
library(egg)
```

Ejercicio 5

Los datos en la tabla anexa corresponden al problema de byssinosis. Las variables involucradas son: años de empleado, fuma, sexo, raza, lugar y byssinosis. Los datos corresponden a tres lugares de trabajo, dos razas, tres periodos de tiempos trabajados y sexo, la variable respuesta es byssinosis, la idea es ajustar un modelo para datos binomiales.

Respuesta

Comenzamos haciendo una descripción la tabla que tenemos.

```r
byssinosis <- read_xls(path = "byssinosis.xls",range = "B4:K27",col_names = F)
names(byssinosis) <- c("Employment","Smoking","Sex","Race",
                       "W1y","W1n","W2y","W2n","W3y","W3n")
```

Tabla 1: Fragmento de la tabla de datos.

Employment	Smoking	Sex	Race	W1y	W1n	W2y	W2n	W3y	W3n
<10	yes	M	W	3	37	0	74	2	258
<10	yes	M	OR	25	139	0	88	3	242
<10	yes	F	W	0	5	1	93	3	180
<10	yes	F	OR	2	22	2	145	3	260
<10	no	M	W	0	16	0	35	0	134
<10	no	M	OR	6	75	1	47	1	122

De donde nos percatamos que, por la naturaleza de la estructura de los datos, será necesario hacer un procesamiento de la información para poder aplicar la funciones que nosotros conocemos para hacer el modelo solicitado. Este procesamiento se muestra a continuación, la idea es básicamente transformar la información de como viene presentada, de forma agrupada dentro una tabla de contingencia, a como si fueran individuos representados por tuplas.

```r
# Procesando la información a individuos ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Primero pasamos las columnas a una sola.
datos <- reshape2::melt(byssinosis)
# Separamos estas columnas en las dos características deseadas.
datos <- datos %>%
  mutate(Workplace = ifelse(variable %in% c("W1y", "W1n"),1,
                            ifelse(variable %in% c("W2y", "W2n"),2,3)),
```

```
        Byssinosis = ifelse(variable %in% c("W1y", "W2y", "W3y"),"yes","no"))
# Repetimos con base en value.
individuos=rep(seq_len(nrow(datos)),datos$value)
datos <- datos[individuos,]
# Nos quedamos solo las columnas deseadas
datos <- datos %>% dplyr::select(-c(variable,value))
```

De tal manera que ahora nuestros datos tienen la siguiente forma

Tabla 2: Fragmento de la tabla de datos procesados.

	Employment	Smoking	Sex	Race	Workplace	Byssinosis
1	<10	yes	M	W	1	yes
1.1	<10	yes	M	W	1	yes
1.2	<10	yes	M	W	1	yes
2	<10	yes	M	OR	1	yes
2.1	<10	yes	M	OR	1	yes
2.2	<10	yes	M	OR	1	yes
2.3	<10	yes	M	OR	1	yes

Ya teniendo la información de esta manera, basta con identificar a las variables que son categóricas de acuerdo a la descripción de la información y luego ajustar los modelos deseados. En nuestro caso haremos 3 modelos los cuales mostramos a continuación.

- `fit1`: Probit

```
fit1 = glm(
  Byssinosis %>% as.factor() ~ Employment + Smoking + Sex + Race + as.factor(Workplace),
  binomial(link = "probit"),
  datos
)
summary(fit1)
```

```
##
## Call:
## glm(formula = Byssinosis %>% as.factor() ~ Employment + Smoking +
##     Sex + Race + as.factor(Workplace), family = binomial(link = "probit"),
##     data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7690  -0.1994  -0.1585  -0.1303   3.2648
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -1.24732    0.11487 -10.859  < 2e-16 ***
## Employment>20          0.32818    0.10040   3.269  0.00108 **
## Employment10 to 20     0.23509    0.12364   1.901  0.05725 .
## Smokingyes             0.26325    0.08571   3.071  0.00213 **
## SexM                  -0.08175    0.09486  -0.862  0.38875
## RaceW                 -0.06536    0.09874  -0.662  0.50800
## as.factor(Workplace)2 -1.19981    0.12334  -9.728  < 2e-16 ***
## as.factor(Workplace)3 -1.25749    0.09569 -13.141  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1477.2  on 5418  degrees of freedom
## Residual deviance: 1202.6  on 5411  degrees of freedom
## AIC: 1218.6
##
## Number of Fisher Scoring iterations: 7
```

- fit2: Logit

```
fit2 = glm(
  Byssinosis %>% as.factor() ~ Employment + Smoking + Sex + Race + as.factor(Workplace),
  binomial(link = "logit"),
  datos
)
summary(fit2)
```

```
##
## Call:
## glm(formula = Byssinosis %>% as.factor() ~ Employment + Smoking +
##     Sex + Race + as.factor(Workplace), family = binomial(link = "logit"),
##     data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8080  -0.1980  -0.1535  -0.1362   3.2269
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -2.3463     0.2639  -8.892  < 2e-16 ***
## Employment>20           0.7531     0.2161   3.484 0.000493 ***
## Employment10 to 20      0.5641     0.2617   2.156 0.031091 *
## Smokingyes              0.6413     0.1944   3.299 0.000971 ***
## SexM                   -0.1239     0.2288  -0.542 0.587982
## RaceW                  -0.1163     0.2072  -0.562 0.574425
## as.factor(Workplace)2  -2.5799     0.2921  -8.834  < 2e-16 ***
## as.factor(Workplace)3  -2.7306     0.2153 -12.681  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1477.2  on 5418  degrees of freedom
## Residual deviance: 1197.9  on 5411  degrees of freedom
## AIC: 1213.9
##
## Number of Fisher Scoring iterations: 7
```

- fit3: CLogLog

```
fit3 = glm(
  Byssinosis %>% as.factor() ~ Employment + Smoking + Sex + Race + as.factor(Workplace),
  binomial(link = "cloglog"),
  datos
```

```
)
summary(fit3)
```

```
##
## Call:
## glm(formula = Byssinosis %>% as.factor() ~ Employment + Smoking +
##     Sex + Race + as.factor(Workplace), family = binomial(link = "cloglog"),
##     data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8158  -0.1972  -0.1528  -0.1382   3.2169
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -2.42916    0.25610  -9.485  < 2e-16 ***
## Employment>20          0.71114    0.20225   3.516 0.000438 ***
## Employment10 to 20     0.53760    0.24430   2.201 0.027763 *
## Smokingyes             0.61774    0.18648   3.313 0.000924 ***
## SexM                  -0.10296    0.22292  -0.462 0.644175
## RaceW                 -0.09754    0.19179  -0.509 0.611060
## as.factor(Workplace)2 -2.48838    0.28474  -8.739  < 2e-16 ***
## as.factor(Workplace)3 -2.63911    0.20740 -12.725  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1477.2  on 5418  degrees of freedom
## Residual deviance: 1197.1  on 5411  degrees of freedom
## AIC: 1213.1
##
## Number of Fisher Scoring iterations: 7
```

Donde finalmente y tras haber aplicado los modelos podemos elegir uno con un ajuste más adecuado con base en los AIC y BIC.

Tabla 3: AIC's y BIC's de los modelos para Byssinosis.

	Probit	Logit	CLogLog
AIC	1218.633	1213.936	1213.134
BIC	1271.415	1266.718	1265.915

Donde podemos concluir con la elección del modelo CLogLog, en particular este modelo por ejemplo nos está indicando que las variables `Sex` y `Race` parecen no ser significativas.

Ejercicio 7

Los datos de la tabla corresponden a compras de café instantáneo. Ajusta los modelos de simetría, homogeneidad marginal y cuasi independencia e interpreta.

```r
# Creación de los datos
nombres <- c("High Point","Tasters Choice","Sanka","Nescafe","Brim")
tabla <- matrix(c(93, 17, 44, 7, 10,
                  9, 46, 11, 0, 9,
                  17, 11, 155, 9, 12,
                  6, 4, 9, 15, 2,
                  10, 4, 12, 2, 27),
                5,5,TRUE,
                list(nombres,nombres))
```

Tabla 4: Tabla de Contingencia de Cafés

2da. compra	High Point	Tasters Choice	Sanka	Nescafe	Brim
1era. compra					
High Point	93	17	44	7	10
Tasters Choice	9	46	11	0	9
Sanka	17	11	155	9	12
Nescafe	6	4	9	15	2
Brim	10	4	12	2	27

Respuesta

Para poder aplicar este tipo de modelos debemos hacer uso de variables dummys, la cuales, debido al desconocimiento de una función pre-programada que haga la tarea, tendremos que hacer esto de forma "artesanal". Primero pasemos nuestros datos a una vista estilo tabla (`data.frame`).

```r
datos = tabla %>% as.table %>%  as.data.frame()
```

Tabla 5: Fragmento de los datos en data.frame

Primera.compra	Segunda.compra	Freq
High Point	High Point	93
Tasters Choice	High Point	9
Sanka	High Point	17
Nescafe	High Point	6
Brim	High Point	10
High Point	Tasters Choice	17

1. Comenzamos agregando a `datos` algunas variables dummys para ver si están en la diagonal y una bandera por variable sobre las diversas categorías que hay.

```r
for(d in nombres){

  #  Para la diagonal
  expr <- paste0("datos$d",gsub(" ","_",d),
                 "= ifelse(datos[,1]=='",d,"' & datos[,2]== '",d,"',1,0)")
  eval(parse(text=expr))

  # Bandera sobre la primera variable
```

```
  expr <- paste0("datos$Primera.compra_",gsub(" ","_",d),
                 "= ifelse(datos[,1]=='",d,"',1,0)")
  eval(parse(text=expr))

  # Bandera sobre la segunda variable
  expr <- paste0("datos$Segunda.compra_",gsub(" ","_",d),
                 "= ifelse(datos[,2]=='",d,"',1,0)")
  eval(parse(text=expr))

}
```

De tal manera que nuestros datos tienen la siguiente forma

Tabla 6: Fragmento de los datos con banderas por variables

Primera.compra	Segunda.compra	Freq	dHigh_Point	Primera.compra_High_Point	Segunda.compra_High_Point
High Point	High Point	93	1	1	1
Tasters Choice	High Point	9	0	0	1
Sanka	High Point	17	0	0	1
Nescafe	High Point	6	0	0	1
Brim	High Point	10	0	0	1
High Point	Tasters Choice	17	0	1	0

dTasters_Choice	Primera.compra_Tasters_Choice	Segunda.compra_Tasters_Choice	dSanka
0	0	0	0
0	1	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	1	0

Primera.compra_Sanka	Segunda.compra_Sanka	dNescafe	Primera.compra_Nescafe
0	0	0	0
0	0	0	0
1	0	0	0
0	0	0	1
0	0	0	0
0	0	0	0

Segunda.compra_Nescafe	dBrim	Primera.compra_Brim	Segunda.compra_Brim
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	1	0
0	0	0	0

Basándonos en la forma de la matriz en la **Tabla 4**, lo que hicimos fue crear columnas auxiliares que denotan, por ejemplo `dBrim` vale 1 si ese renglón se encuentra en la diagonal para la categoría `Brim`, y cero en otro caso. Otro ejemplo es \texttt{Primera.compra_Sanka} que vale 1 si ese renglón simboliza que en la variable `Primera.compra` tomó el valor `Sanka` y análogo para `Segunda.compra`. Como podemos ver, tenemos ya algunas variables dummys.

2. Procedemos agregando a `datos` algunas variables dummys para etiquetar las banderas de la simetría.

```
# Este es difícil, pero es la cantidad de simetrías que hay en la tabla de
# contingencia.
total_simetrías = sum(1:ncol(tabla))

# Todo esto se hizo pensando en la tabla que queremos como resultado.
# OJO la tabla, no el data.frame. Hay que pensar un poco.

# Comenzamos en esta parte de la matriz
i = 1
j = 1
datos$symm <- 0

# La idea es ir recorriendo la diagonal superior de arriba a abajo
# de izquierda a derecha.
for(s in 1:(total_simetrías)){

  # Aquí estamos creando las variables symmi
  expr <- paste0("datos$symm",s,
                 " <- ifelse((datos[,1]=='",nombres[i],
                 "' & datos[,2]== '",nombres[j],
                 "') | (datos[,1]=='",nombres[j],
                 "' & datos[,2]== '",nombres[i],
                 "'),",s,",0)")
  eval(parse(text=expr))
  #print(paste("i=",i,", j=",j))

  # Avanzamos a la siguiente columna
  j=j+1

  # Si nos salimos de la tabla entonces más bien recorremos al
  # siguiente renglón y reiniciamos.
  if(j>ncol(tabla)) {
    # Avanzamos al siguiente renglón
    i = i + 1
    # Y la columna la reiniciamos pero a partir del renglón en cuestión
    j = i
  }

  # Aquí finalmente vamos sumando en la columna symm las que vamos crando.
  expr <- paste0("datos$symm<-datos$symm + datos$symm",s)
  eval(parse(text=expr))

}
```

De donde vemos que nuestra matriz de "banderas" de simetría tiene la siguiente forma.

```
# Vamos a comprobar que funciona la simetría
# Esto se ve si usamos el siguiente vector y lo ponemos como lo siguiente
Maux<-matrix(data = datos$symm,nrow = 5,ncol = 5,byrow = TRUE,
             dimnames = list(nombres,nombres))
```

Tabla 7: Matriz con banderas de simetrías

	High Point	Tasters Choice	Sanka	Nescafe	Brim
High Point	1	2	3	4	5
Tasters Choice	2	6	7	8	9
Sanka	3	7	10	11	12
Nescafe	4	8	11	13	14
Brim	5	9	12	14	15

Esto, se le agregó al objeto `datos` anterior, de tal manera que, además de lo que antes ya vimos que había en este objeto, se le agregó adicionalmente una variable de este estilo donde se pueda mostrar la simetría.

Tabla 8: Fragmento de los datos con banderas por simetría

Primera.compra	Segunda.compra	symm	symm1	symm2	symm3	symm4	symm5	symm6	symm7	symm8	symm9	symm10	symm11	symm12	symm13	symm14	symm15
High Point	High Point	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tasters Choice	High Point	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
Sanka	High Point	3	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0
Nescafe	High Point	4	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0
Brim	High Point	5	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0
High Point	Tasters Choice	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0

Basados en la **Tabla 7**, creamos columnas que indican la simetría, la columna `symm` indica el tipo de simetría del renglón en cuestión. Mientras que las demás columnas `symmX` son una indicadora sobre los renglones sobre si cumplen con la simetría 'X'. Ahora haremos un pequeño código que nos ayudará a extraer información reelevante de cada modelo:

```r
# Cálculo de G^2 y p-value
G_p <- function(fit){
  sfit<-summary(fit)
  df = sfit$df.residual
  dev=sfit$deviance
  p = 1-pchisq(q = dev,df = df)
```

```
  return(c(df=df,`G-Cuadrada`=dev,`p-value`=p))
}
```

Al fin, estamos listos para poder aplicar los modelos apoyándonos de estas variables.

```
# En esta lista guardaremos todos los modelos
Lista = list()
```

- **Modelo de Independencia**: Nos indica si las variables `Primera.compra` y `Segunda.compra` tienen un comportamiento independiente.

```
Lista$Independence=glm(Freq~
                         # Variables
                         Primera.compra+Segunda.compra,
                       family=poisson(link=log),
                       data = datos)
summary(Lista$Independence)
```

```
##
## Call:
## glm(formula = Freq ~ Primera.compra + Segunda.compra, family = poisson(link = log),
##     data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.5246  -2.5005  -1.1303  -0.6549   7.7031
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    3.7535     0.1068  35.144  < 2e-16 ***
## Primera.compraTasters Choice  -0.8242     0.1385  -5.951 2.66e-09 ***
## Primera.compraSanka            0.1765     0.1037   1.702  0.08877 .
## Primera.compraNescafe         -1.5581     0.1834  -8.497  < 2e-16 ***
## Primera.compraBrim            -1.1343     0.1550  -7.318 2.52e-13 ***
## Segunda.compraTasters Choice  -0.4986     0.1400  -3.561  0.00037 ***
## Segunda.compraSanka            0.5371     0.1083   4.958 7.11e-07 ***
## Segunda.compraNescafe         -1.4088     0.1942  -7.255 4.02e-13 ***
## Segunda.compraBrim            -0.8109     0.1552  -5.226 1.73e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 768.54  on 24  degrees of freedom
## Residual deviance: 346.38  on 16  degrees of freedom
## AIC: 468.92
##
## Number of Fisher Scoring iterations: 5
```

- **Modelo de Cuasi-Independencia**: Nos indica si las variables `Primera.compra` y `Segunda.compra` tienen un comportamiento "casi" independiente.

```
Lista$Quasi_Independence=glm(Freq~
                               # Variables
                               Primera.compra+Segunda.compra+
                               # Diagonal
```

```r
                                dHigh_Point+dTasters_Choice+dSanka+dNescafe+dBrim,
                              family=poisson(link=log),
                              data = datos)
summary(Lista$Quasi_Independence)
```

```
##
## Call:
## glm(formula = Freq ~ Primera.compra + Segunda.compra + dHigh_Point +
##     dTasters_Choice + dSanka + dNescafe + dBrim, family = poisson(link = log),
##     data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2123  -0.3375   0.0000   0.2768   1.7751
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    3.2417     0.2231  14.530  < 2e-16 ***
## Primera.compraTasters Choice  -1.1102     0.2257  -4.918 8.73e-07 ***
## Primera.compraSanka           -0.2472     0.2080  -1.188   0.2348
## Primera.compraNescafe         -1.5214     0.2515  -6.049 1.46e-09 ***
## Primera.compraBrim            -1.1610     0.2281  -5.090 3.57e-07 ***
## Segunda.compraTasters Choice  -0.4965     0.2382  -2.085   0.0371 *
## Segunda.compraSanka            0.4679     0.2187   2.139   0.0324 *
## Segunda.compraNescafe         -1.2366     0.2896  -4.270 1.96e-05 ***
## Segunda.compraBrim            -0.5906     0.2432  -2.429   0.0152 *
## dHigh_Point                    1.2909     0.2460   5.247 1.55e-07 ***
## dTasters_Choice                2.1936     0.3026   7.250 4.17e-13 ***
## dSanka                         1.5810     0.2315   6.830 8.48e-12 ***
## dNescafe                       2.2244     0.4197   5.300 1.16e-07 ***
## dBrim                          1.8057     0.3322   5.435 5.48e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 768.544  on 24  degrees of freedom
## Residual deviance:  13.786  on 11  degrees of freedom
## AIC: 146.32
##
## Number of Fisher Scoring iterations: 5
```

- **Modelo de Simetría**: Nos indica si es indistinto el comportamiento simétrico entre las variables `Primera.compra` y `Segunda.compra`.

```r
Lista$Symmetry=glm(Freq~
                     # Simetrías
                     symm1+symm2+symm3+symm4+symm5+symm6+symm7+symm8+symm9+
                     symm10+symm11+symm12+symm13+symm14+symm15,
                   family=poisson(link=log),
                   data = datos)
summary(Lista$Symmetry)
```

```
##
## Call:
```

```
## glm(formula = Freq ~ symm1 + symm2 + symm3 + symm4 + symm5 +
##     symm6 + symm7 + symm8 + symm9 + symm10 + symm11 + symm12 +
##     symm13 + symm14 + symm15, family = poisson(link = log), data = datos)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.670   0.000   0.000   0.000   2.291
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.29584    0.19245  17.126  < 2e-16 ***
## symm1        1.23676    0.21861   5.657 1.54e-08 ***
## symm2       -0.36544    0.13739  -2.660 0.007814 **
## symm3        0.04063    0.07705   0.527 0.597973
## symm4       -0.35601    0.08439  -4.218 2.46e-05 ***
## symm5       -0.19865    0.05900  -3.367 0.000761 ***
## symm6        0.08880    0.04041   2.198 0.027971 *
## symm7       -0.12828    0.04103  -3.126 0.001770 **
## symm8       -0.32534    0.06697  -4.858 1.19e-06 ***
## symm9       -0.15823    0.03751  -4.218 2.46e-05 ***
## symm10       0.17476    0.02085   8.380  < 2e-16 ***
## symm11      -0.09987    0.02766  -3.610 0.000306 ***
## symm12      -0.06758    0.02338  -2.891 0.003845 **
## symm13      -0.04521    0.02477  -1.825 0.067963 .
## symm14      -0.18591    0.03827  -4.858 1.19e-06 ***
## symm15            NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 768.544  on 24  degrees of freedom
## Residual deviance:  22.473  on 10  degrees of freedom
## AIC: 157.01
##
## Number of Fisher Scoring iterations: 5
```

- **Modelo de Cuasi-Simetría**: Nos indica si es "casi" indistinto el comportamiento simétrico entre las variables `Primera.compra` y `Segunda.compra`.

```r
Lista$Quasi_Symmetry=glm(Freq~
                           # Primera compra bandera por categoría
                           Primera.compra_High_Point+
                           Primera.compra_Tasters_Choice+
                           Primera.compra_Sanka+
                           Primera.compra_Nescafe+
                           Primera.compra_Brim+
                           # Segunda compra bandera por categoría
                           Segunda.compra_High_Point+
                           Segunda.compra_Tasters_Choice+
                           Segunda.compra_Sanka+
                           Segunda.compra_Nescafe+
                           Segunda.compra_Brim+
                           # Simetrías
                           symm1+symm2+symm3+symm4+symm5+
```

```
                           symm6+symm7+symm8+symm9+symm10+
                           symm11+symm12+symm13+symm14+symm15,
                         family=poisson(link=log),
                         data = datos)
summary(Lista$Quasi_Symmetry)
```

```
##
## Call:
## glm(formula = Freq ~ Primera.compra_High_Point + Primera.compra_Tasters_Choice +
##     Primera.compra_Sanka + Primera.compra_Nescafe + Primera.compra_Brim +
##     Segunda.compra_High_Point + Segunda.compra_Tasters_Choice +
##     Segunda.compra_Sanka + Segunda.compra_Nescafe + Segunda.compra_Brim +
##     symm1 + symm2 + symm3 + symm4 + symm5 + symm6 + symm7 + symm8 +
##     symm9 + symm10 + symm11 + symm12 + symm13 + symm14 + symm15,
##     family = poisson(link = log), data = datos)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8419  -0.1925   0.0000   0.1999   1.0215
##
## Coefficients: (7 not defined because of singularities)
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    3.29584    0.19245  17.126  < 2e-16 ***
## Primera.compra_High_Point     -0.73921    0.31294  -2.362 0.018168 *
## Primera.compra_Tasters_Choice -1.42604    0.37575  -3.795 0.000148 ***
## Primera.compra_Sanka          -0.86918    0.31841  -2.730 0.006338 **
## Primera.compra_Nescafe        -2.46300    0.56229  -4.380 1.19e-05 ***
## Primera.compra_Brim                 NA         NA      NA       NA
## Segunda.compra_High_Point     -1.33465    0.35053  -3.808 0.000140 ***
## Segunda.compra_Tasters_Choice -1.42203    0.37546  -3.787 0.000152 ***
## Segunda.compra_Sanka          -0.75588    0.31111  -2.430 0.015113 *
## Segunda.compra_Nescafe        -2.76511    0.58340  -4.740 2.14e-06 ***
## Segunda.compra_Brim                 NA         NA      NA       NA
## symm1                          3.31062    0.50498   6.556 5.53e-11 ***
## symm2                          0.84291    0.22620   3.726 0.000194 ***
## symm3                          0.63661    0.12752   4.992 5.97e-07 ***
## symm4                          0.55406    0.16187   3.423 0.000620 ***
## symm5                               NA         NA      NA       NA
## symm6                          0.56348    0.10089   5.585 2.34e-08 ***
## symm7                          0.19102    0.06407   2.981 0.002871 **
## symm8                          0.17797    0.09799   1.816 0.069349 .
## symm9                               NA         NA      NA       NA
## symm10                         0.33727    0.04587   7.352 1.95e-13 ***
## symm11                         0.20969    0.05638   3.719 0.000200 ***
## symm12                              NA         NA      NA       NA
## symm13                         0.35695    0.08095   4.410 1.04e-05 ***
## symm14                              NA         NA      NA       NA
## symm15                              NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 768.544  on 24  degrees of freedom
```

```
## Residual deviance:   9.974  on  6  degrees of freedom
## AIC: 152.51
##
## Number of Fisher Scoring iterations: 5
```

Aprovechamos para extraer ciertas estadísticas de los modelos hasta el momento

```
# (Casi) todos los modelos
Resumen = sapply(Lista,G_p) %>% as.data.frame()
```

- **Homogeneidad Marginal**: Nos indica si las distribuciones marginales de cada variable (pensado como suma de los renglones y columnas) son equivalentes. Las estadísticas (que denotaremos en general como ϕ) de interés en este modelo podemos calcularlas a partir de tomar la diferencia

$$\phi(\text{Homogeneidad Marginal}) = \phi(\text{Simetría}) - \phi(\text{Cuasi Simetría})$$

Esto lo vemos en el siguiente código

```
# Se toma como la diferencia entre el de simetría y quasi simetría.
Resumen$Marginal_Homogeneity = Resumen$Symmetry-Resumen$Quasi_Symmetry
Resumen$Marginal_Homogeneity[3] = 1-pchisq(q = Resumen$Marginal_Homogeneity[2],
                                            df = Resumen$Marginal_Homogeneity[1])
```

Nota: Recordemos que Simetría implica Homogeneidad marginal pero no al revés.

Finalmente, concluimos mostrando la siguiente **tabla de resultados de los modelos**.

Tabla 9: Tabla de resultados de los modelos

	Independence	Quasi_Independence	Symmetry	Quasi_Symmetry	Marginal_Homogeneity
df	16.000	11.0000	10.0000	6.0000	4.0000
G-Cuadrada	346.381	13.7856	22.4729	9.9740	12.4989
p-value	0.000	0.2451	0.0129	0.1257	0.0140

De donde concluimos:

1. Hay cierto grado de dependencia entre las variables `Primera.compra` y `Segunda.compra`.
2. El comportamiento de la tabla tiene cierto grado de simetría.
3. No hay evidencia para decir que existe la homogeneidad marginal.
4. **El mejor modelo para estos datos es el de Cuasi-Independencia, basándonos en las estadísticas anteriores**.

Ejercicio 8

Los datos representan el numero de insectos muertos(`kill`) de un total (`number`) por tres venenos diferentes (poison) en dosis en escala logaritmica (`Logdose`) Determina la dosis letal 50 para cada veneno y di cual es el mas efectivo en este caso. ¿Es lo mismo si lo que nos interés es que se muera el 80% de los insectos?

Tabla 10: 'datos' - Venenos para insectos.

Kill	Number	Poison	LogDose
44	50	R	1.01
42	49	R	0.89
24	46	R	0.71
16	48	R	0.58
6	50	R	0.41
48	48	D	1.70
47	50	D	1.61
47	49	D	1.48
34	48	D	1.31
18	48	D	1.00
16	49	D	0.71
48	50	M	1.40
43	46	M	1.31
38	48	M	1.18
27	46	M	1.00
22	46	M	0.71
7	47	M	0.40

Respuesta

Notemos que el modelo que podemos aplicar es usando una **familia binomial** pues tenemos observaciones del estilo "éxitos" (`Kill`) y "ensayos" (`Number`)

Nota: Por la naturaleza de los datos, aquí estamos modelando p como la probabilidad de muerte. De tal manera que cuando buscamos una "Dosis Letal X" $(LD(x))$ con $x \geq 50$ tendremos que, usando la liga `logit`, el momio de muerte por veneno resultará en favor de la probabilidad de muerte.

Ahora, podemos proponer aquí dos modelos

- `fit1`: Le decimos `R` que **no** conocemos el tamaño de las binomiales (esto significaría que no conocemos los pesos `Number`).

```
fit1 <- glm((Kill/Number)~Poison+LogDose,
            family=quasibinomial(link = "logit"),
            data = datos)
```

- `fit2`: Le decimos `R` que **sí** conocemos el tamaño de las binomiales (esto significaría que conocemos los pesos `Number`).

```
fit2 <- glm((Kill/Number)~Poison+LogDose,
            family=binomial(link = "logit"),
            weights = Number,
            data = datos)
```

Para ambos modelos, vamos a conocer información estadística importante que mostraremos en la siguiente tabla

```
# Vamos a extraer la información importante
info <- function(fit){
  sfit=summary(fit)
  Coef = sfit$coefficients[,1]
  p = sfit$coefficients[,4]
  return(data.frame(Coef=Coef,`p-value`=p))
}
```

- `fit1`

Tabla 11: Información estadística de 'fit1'

	Coef	p.value
(Intercept)	-4.8549695	0.0000041
PoisonM	0.9142227	0.0288279
PoisonR	1.6000195	0.0016959
LogDose	4.8112116	0.0000004

- `fit2`

Tabla 12: Información estadística de 'fit2'

	Coef	p.value
(Intercept)	-4.8687634	0.0000000
PoisonM	0.9123013	0.0001956
PoisonR	1.6033907	0.0000000
LogDose	4.8277172	0.0000000

De donde vemos que ambos modelos presentan todos coeficientes como significativos y de hecho, según los datos que tenemos hasta este punto, todo indicaría que el modelo más apropiado es `fit2`. Ahora vamos a calcular las dosis letales para cada uno de los venenos. Notemos que R tomó como veneno base el tipo D. De tal manera que tendremos las siguientes ecuaciones para las dosis letales.

- Para el veneno D.

$$logit(0.5) = \beta_0 + \beta_{LogDose} * LD_{50}(D)$$

- Para el veneno M.

$$logit(0.5) = \beta_0 + \beta_{PoissonM} + \beta_{LogDose} * LD_{50}(D)$$

- Para el veneno R.

$$logit(0.5) = \beta_0 + \beta_{PoissonR} + \beta_{LogDose} * LD_{50}(D)$$

Creamos una función para calcular varias LD de los modelos que tenemos

```
Dosis_Letales_Veneno <- function(x,fit){

  # Esta función es muy particular para la información que estamos manejando.
  # x   := El nivel deseado de la dosis letal.
  # fit := El modelo que utilizará para obtener las dosis letales
```

```r
  LD <- c()
  ## Para el veneno D.
  LD["Veneno D"] = (logit(x/100)-coef(fit)[1])/coef(fit)["LogDose"]
  ## Para el veneno M.
  LD["Veneno M"] = (logit(x/100)-coef(fit)[1]-coef(fit)["PoisonM"])/coef(fit1)["LogDose"]
  ## Para el veneno R.
  LD["Veneno R"] = (logit(x/100)-coef(fit)[1]-coef(fit)["PoisonR"])/coef(fit1)["LogDose"]

  # Regresamos todos
  return(LD)

}
```

- Para `fit1`

```r
LD1 <- data.frame(`LD50`=Dosis_Letales_Veneno(50,fit1),
                  `LD80`=Dosis_Letales_Veneno(80,fit1))
```

Tabla 13: Dosis letales para 'fit1'

	LD50	LD80
Veneno D	1.0090950	1.2972333
Veneno M	0.8190758	1.1072141
Veneno R	0.6765343	0.9646727

En todo caso, el mejor veneno es el del tipo R, pues es el que necesita una dosis menor para lograr una p suficiente para matar al 50% de los insectos.

- Para `fit2`

```r
LD2 <- data.frame(`LD50`=Dosis_Letales_Veneno(50,fit2),
                  `LD80`=Dosis_Letales_Veneno(80,fit2))
```

Tabla 14: Dosis letales para 'fit2'

	LD50	LD80
Veneno D	1.0085022	1.295655
Veneno M	0.8223422	1.110480
Veneno R	0.6787007	0.966839

De igual manera, el mejor veneno es el del tipo R, pues es el que necesita una dosis menor para lograr una p suficiente para matar al 50% de los insectos.

En conclusión, creemos que el mejor modelo es `fit2`, que es donde asumimos conocidas las m's, pues de hecho es información que se nos proporciona en la tabla de datos. Sin embargo, vemos que los coeficientes entre ambos modelos no cambian mucho y resultan significativos, de tal manera que la interpretación final para este caso resulta ser exactamente igual, tenemos que **el mejor veneno es el tipo R**.

Ejercicio 9

El archivo `tarealog1.xls` contiene una variable explicativa uni y cuatro variables respuesta, haz los cuatro ajustes y di que observas, trata de explicarlo en términos de la relación de las variables (explicativa y respuesta).

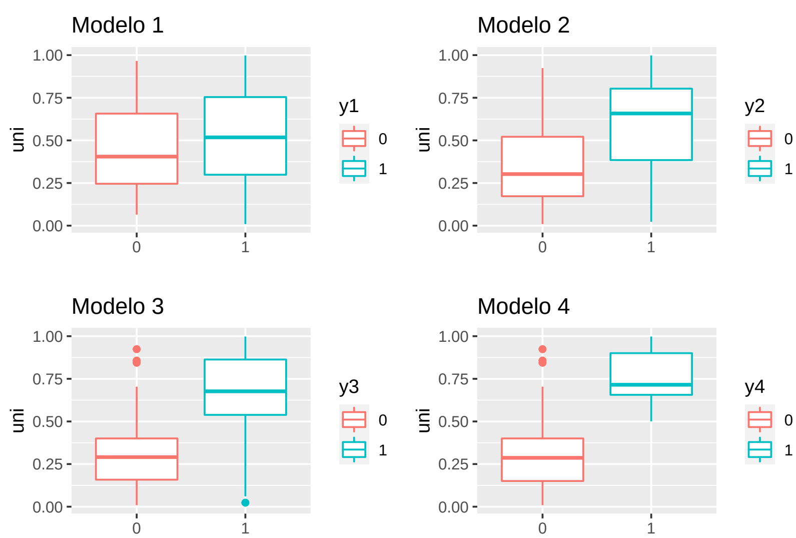
Respuesta

Veamos rápidamente cómo se compone la información que estamos trabajando

Tabla 15: Fragmento de los datos en 'tarealog1.xls'

uni	y1	y2	y3	y4
0.8442258	0	0	0	0
0.0316049	1	0	0	0
0.9597832	1	1	1	1
0.9979368	1	1	1	1
0.6842272	1	1	1	1
0.1606886	0	0	0	0

Comenzamos haciendo un pequeño análisis descriptivo de la relación que hay entre la variable explicativa con las respuestas. En este caso, al estar analizando 4 posibles variables respuesta tendremos y una única explicativa, haremos este breve análisis exploratorio en forma de gráficos de caja.



Algo que podemos notar gracias a esto, es que dependiendo de la variable respuesta, se logra percibir una separación entre los valores que son 0 y 1. De hecho esta propiedad va mejorando del modelo 1 al modelo 4, siendo este último el que a primera vista parece separar mejor las respuestas. Procedemos a crear los modelos y una función que nos pueda hacer un resumen de los mismos con información estadística reelevante

```
Lista <- list()
for(i in 1:4){
  expr <- paste0(
    "Lista$fit",i," <- glm(y",i," ~ uni,
                       family = binomial(link = logit),
                       data = tarealog1)"
  )
  eval(parse(text=expr))
}
```

```r
info <- function(fit){

  # Aplicamos un resumen al modelo
  aux <- summary(fit)
  # Vamos a extraer ciertas estadísticas
  betas = aux$coefficients[, 1]
  names(betas) = paste("Coeficiente", names(betas))
  P = aux$coefficients[, 4]
  names(P) = paste("p-value", names(P))
  aic = aux$aic
  dev = aux$deviance

  # Regresamos toda la información
  return(c(betas,P,AIC=aic,Devianza=dev) %>% round(3))

}
```

Finalmente, vamos a ver los resultados de este modelo

Tabla 16: Estadísticas de los 4 modelos

	fit1	fit2	fit3	fit4
Coeficiente (Intercept)	0.482	-0.932	-2.859	-6.083
Coeficiente uni	0.749	3.131	6.000	10.933
p-value (Intercept)	0.259	0.034	0.000	0.000
p-value uni	0.330	0.000	0.000	0.000
AIC	125.213	120.017	98.160	63.416
Devianza	121.213	116.017	94.160	59.416

De donde podemos apreciar lo dicho anteriormente, en efecto, el modelo 4 con la variable respuesta y4 es el que resulta tener las estadísticas más favorables, haciendo significativos a los coeficientes y teniendo tanto un AIC como una Devianza baja. Efecto que se visualizaba desde los gráficos anteriores.