Métodos Lineales Generalizados: Regresión Logística Probit Bayesiana

David Montaño Castro

Para el conjunto de datos *bank*, del paquete *bayess*. Resolver los incisos:

1. **Estima un modelo probit** . Justifica las distribuciones iniciales utilizadas. Obtén las estimaciones de los parámetros. Interpreta los resultados.
2. **Incluye las estimaciones usando JAGS**. Simula 2 o 3 cadenas MCMC, explica los resultados para garantizar la convergencia. Compara los resultados con el esquema de datos aumentados (incluyendo variables latentes)

Introducción: Análisis Frecuentista

De primera mano, iniciaré analizando rápidamente la información que se va a estudiar.

Descripción de variables.

200 tuplas con 5 columnas:

- x1 length of the bill (in mm)
- x2 width of the left edge (in mm)
- x3 width of the right edge (in mm)
- x4 bottom margin width (in mm)
- y response variable

Resumen de los datos.

```
       x1              x2              x3              x4              y
 Min.   :213.8   Min.   :129.0   Min.   :129.0   Min.   : 7.200   Min.   :0.0
 1st Qu.:214.6   1st Qu.:129.9   1st Qu.:129.7   1st Qu.: 8.200   1st Qu.:0.0
 Median :214.9   Median :130.2   Median :130.0   Median : 9.100   Median :0.5
 Mean   :214.9   Mean   :130.1   Mean   :130.0   Mean   : 9.418   Mean   :0.5
 3rd Qu.:215.1   3rd Qu.:130.4   3rd Qu.:130.2   3rd Qu.:10.600   3rd Qu.:1.0
 Max.   :216.3   Max.   :131.0   Max.   :131.1   Max.   :12.700   Max.   :1.0
```

Lo más importante a destacar son los valores que toma la variable "y": 1 y 0. Esto inmediatamente da luz del porqué se pide una regresión logística (con liga probit).

Sin entrar en más detalles, las variables independientes son continuas y, si se compara el valor máximo y mínimo de cada una de ellas, noto que no existe una varianza tan grande.

Un hecho muy importante a observar es que la varaible *x2* y *x3* parecieran tener una distribución muy similar, pues su similitud llega al nivel de cuartiles.

Valores NaN

```
FALSE
 1000
```

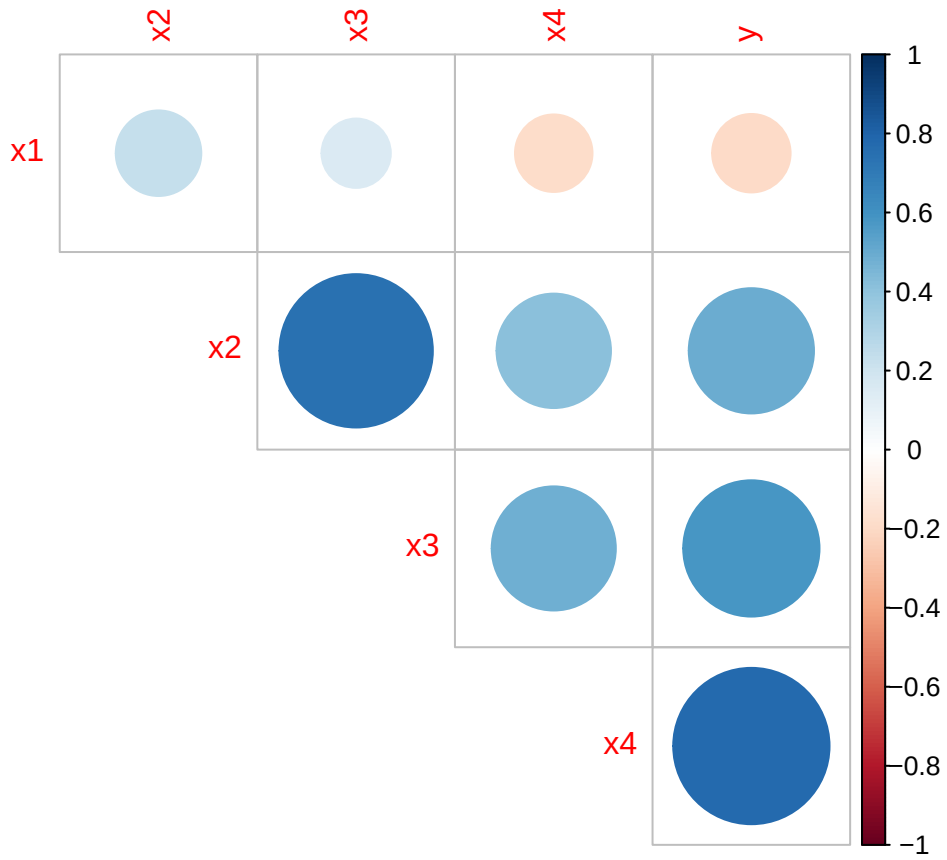
No hay valores faltantes presentes en el conjunto de datos.

Proporción en la variable respuesta "y".

```
y
  0   1
100 100
```

La variable respuesta está bien balanceada, pues contiene 50% de valores positivos y 50% de negativos.

Correlograma



Analizando la variable respuesta "y":

- Y tiene una fuerte relación con casi todas las variables independientes, en menor medida con **x1**, pues su correlación puede llegar a ser despreciable. En contraparte, *x4* es la variable con quien mayor correlación tiene.

Analizando las correlaciones entre las variables independientes:

- Como venía presumiendo desde le inicio, *x2* y *x3* presentaban una distribución muy similar y por lo tanto, su correlación es demasiado alta. Esto, sin duda alguna, generará un conflicto en la regresión por multicolinealidad. No se puede presumír que variable será mejor conservar hasta ver como se comportan en conjunto. Fuera de este par, todas las demás no presentan este suceso.

Modelos Frecuentistas.

Para poder sugerir los hiperparámetros de las distribuciones *a priori*, un buena buena aproximación es obtener los posibles valores a partir de modelos frecuentistas.

Se ajustarán 4 modelos. A medida que se van interpretando los resultados de cada uno, se precederá a realizar modificaciones en el modelo en aras de encontrar un mejor ajuste. Se utilizará un valor de riesgo del .05.

Modelo Saturado

```
regP1 = glm(y ~., data = bank, family = binomial(link = "probit"))
summary(regP1)
```

```
Call:
glm(formula = y ~ ., family = binomial(link = "probit"), data = bank)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6839  -0.3204  -0.0137   0.2114   3.5791

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -113.1117    83.4072  -1.356   0.1751
x1            -0.8075     0.3833  -2.107   0.0351 *
x2             1.0632     0.5993   1.774   0.0760 .
x3             1.0621     0.5415   1.961   0.0499 *
x4             1.1065     0.1691   6.543 6.02e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 277.259  on 199  degrees of freedom
Residual deviance:  90.292  on 195  degrees of freedom
AIC: 100.29

Number of Fisher Scoring iterations: 7
```

Aparentemente, los resultados concuerdan con lo descrito en la sección anterior.

- **Coeficientes**:
  - Tanto *(intercept)* como *x2* son variables estadísticamente no significativas.

  - *x3* es significativa pero, francamente, el p-valor está muy cercano al nivel de riesgo como para considerarlo signiicativo. Para sacar una mejor conclusión, se prestará más atención a él en el siguiente modelo.
  - *x1* y *x4* resultan ser las varaibles con mayor poder predictivo. No era de esperarse que la segunda lo fuera, pues su correlación es muy fuerte con respecto a "y".

- **Devianza**:

  - La devianza nula es mucho mayor respecto a la residual.

Conclusión:

Se decide calcular otro modelo sin la variable *x2*.

Modelo (*x1*, *x3*, *x4*)

```
regP2 = glm(y ~ x1 + x3 + x4, data = bank, family = binomial(link = "probit"))
summary(regP2)
```

```
Call:
glm(formula = y ~ x1 + x3 + x4, family = binomial(link = "probit"),
    data = bank)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5583  -0.3388  -0.0146   0.2264   3.5565

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -100.9258    84.0083  -1.201   0.2296
x1            -0.6332     0.3658  -1.731   0.0834 .
x3             1.7463     0.4275   4.085 4.41e-05 ***
x4             1.0854     0.1631   6.657 2.80e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 277.259  on 199  degrees of freedom
Residual deviance:  93.213  on 196  degrees of freedom
AIC: 101.21

Number of Fisher Scoring iterations: 7
```

- **Coeficientes**:

  - Es ahora la variable *x1* la que no es significativa para el modelo. Esto también concuerdacon lo comentado en la sección anterior, pues su correlación con "y" es bastante decadente.
  - En este modelo *x3* ya muestra significancia muy fuerte para el modelo. Esto puedo atribuirselo a la correlación que antes presentaba con *x2* y de ahí su mal desempeño. *x4* sigue siendo candidata para variable independiente.

- **Devianza**:

- La devianza nula es mucho mayor respecto a la residual, aunque 3 unidades mayor que el modelo anterior.

Conclusión:

Se decide calcular otro modelo sin la variable *x1*.

Modelo (*x3*, *x4*)

```
regP3 = glm(y ~ x3 + x4, data = bank, family = binomial(link = "probit"))
summary(regP3)
```

```
Call:
glm(formula = y ~ x3 + x4, family = binomial(link = "probit"),
    data = bank)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3724  -0.3088  -0.0076   0.2329   3.2559

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -225.9755    54.1299  -4.175 2.98e-05 ***
x3             1.6579     0.4169   3.977 6.99e-05 ***
x4             1.1317     0.1615   7.008 2.43e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 277.259  on 199  degrees of freedom
Residual deviance:  96.221  on 197  degrees of freedom
AIC: 102.22

Number of Fisher Scoring iterations: 7
```

- **Coeficientes**:
  - Todos los coeficientes muestran una significancia fuerte, incluso *(intercept)*.
- **Devianza**:
  - La devianza nula es mucho mayor respecto a la residual. Se aumentó en 3 unidades la residual respecto al segundo modelo. Si se realiza la prueba de chí cuadrado se obtiene un p-valor de $5.2638326 \times 10^{-39}$, cuyo valor definitivamente retrata un modelo con bastante poder predictivo.

Conclusión:

Se decide calcular otro modelo con solamente *x4* solo para corroborar que el presente es el mejor.

Modelo (*x4*)

```
regP4 = glm(y ~ x4, data = bank, family = binomial(link = "probit"))
summary(regP4)
```

```
Call:
glm(formula = y ~ x4, family = binomial(link = "probit"), data = bank)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.31936  -0.46699  -0.04943   0.28753   3.01953

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -11.6835     1.3972  -8.362   <2e-16 ***
x4            1.2668     0.1532   8.268   <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: 277.26  on 199  degrees of freedom
Residual deviance: 116.19  on 198  degrees of freedom
AIC: 120.19

Number of Fisher Scoring iterations: 7
```

Pese que todos los coeficientes son significativos, la devianza aumenta 10 unidades. No obstante, el AIC también lo hace respecto a los otros modelos. Parsimoniosamente hablando, este modelo trabaja con una sola variable independiente, pero su poder predictivo no es tan fuerte como en el anterior.

Conclusión: El mejor modelo es el tercero. Con el se trabajará para proponer coeficientes a priori.

Modelo ganador (*x3*, *x4*)

Interpretación de los parámetros.

Las variables que explican la variable de respuesta son :

- ancho del extremo derecho *x3*: Mientras más ancho sea el extremo derecho, mayor es la probabilidad de y = 1.
- ancho del margen inferior *x4*: Mientras más ancho sea el extremo inferior, mayor es la probabilidad de y = 1.

Intervalos de confianza.

```
                  2.5 %      97.5 %
(Intercept) -332.7328422 -127.719884
x3             0.9000580    2.477863
x4             0.8439107    1.468846
```

Adicionalmente en los intervalos de confianza se confirma la significancia estadística al no estar contenido el 0 en alguno de los intervalos creados para los coeficientes calculados.

Matriz de confusión.

```
   predicted_frequentist
y    0  1
  0 91  9
  1  8 92
```

Accuracy (Cuánto clasifiqué bien de todo) = 91.5%.

Modelo Bayesiano

Simularé un modelo bayesiano para compararlo con el modelo frecuentista analizado anteriormente. Voy a generar 3 cadenas.

Como dije en la sección anterior, puedo ocupar las estimaciones que el modelo frecuentista dio para hacer que la cadena pueda converger más rápido.

Las distribuciones prior que utilizaré para los parámetros (Beta1, Beta2 y Constante) serán normales. Todas son distribuciones normales porque las betas del modelo logístico están calculadas como si fuera un modelo lineal, solo que se mete la liga para hacer la respectiva transformación a probabilidades. La normalidad se sigue del mismo hecho en el que se asume normal el término aleatorio.

Como primer intento, voy a ajustar un modelo en el que ocupe ditribuciones priori normales poco informativas, es decir, Normal(0,1).

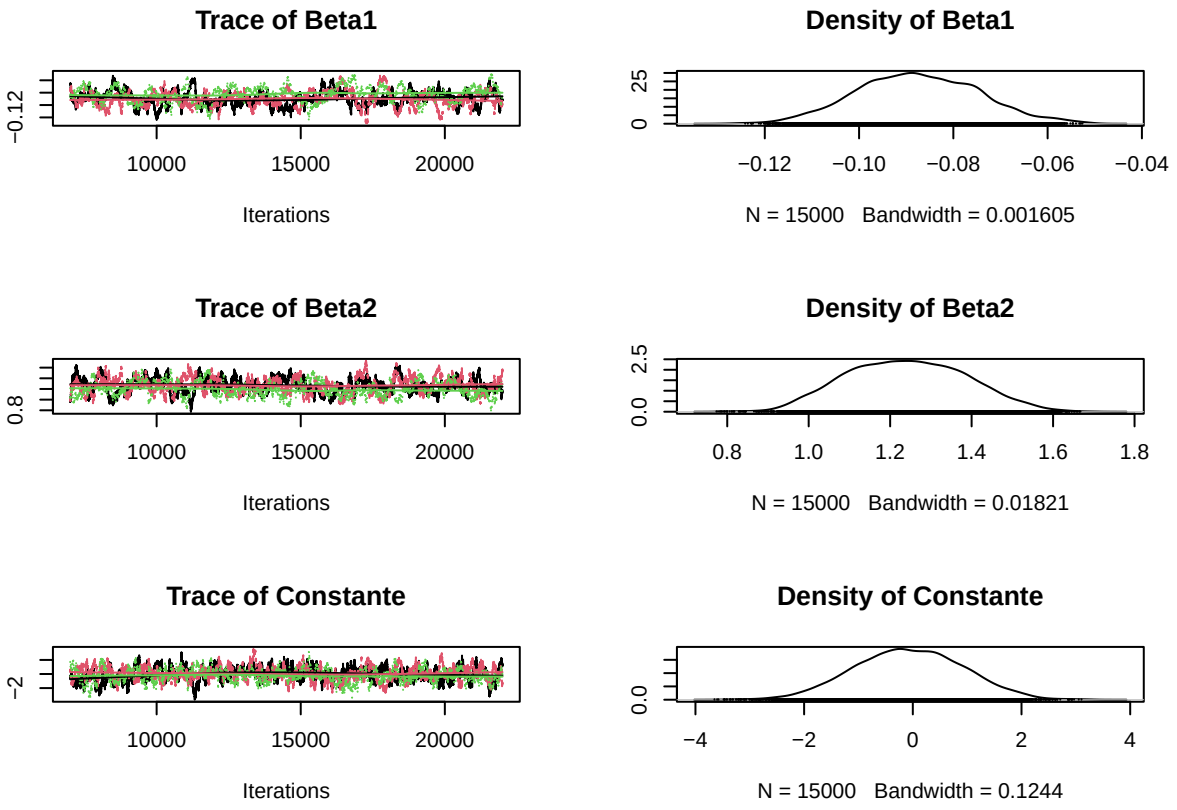
Modelo Bayesiano con prioris Normal(0,1)

```
n = length(y)


data = list(
  y = y,
  x3 = x3,
  x4 = x4,
  n = n
)

params = c("Constante", "Beta1", "Beta2")

inits = function(){list(
  "Constante" = rnorm(1),
  "Beta1" = rnorm(1),
  "Beta2" = rnorm(1)
)
}

modelo = "model{

#### LIKELIHOOD

for(i in 1:n){
```

```
eta[i] = Constante + Beta1 * x3[i] + Beta2 * x4[i]
probit(p[i]) = eta[i]
y[i] ~dbern(p[i])

}

#### PRIORS

Beta1 ~ dnorm(0,1) #  1.6579
Beta2 ~ dnorm(0,1) # 1.1317
Constante ~ dnorm(0,1) # -225.9755


}
"
set.seed(8)
fit = jags.model(file = textConnection(modelo), data = data, inits = inits, n.chains = 3 )
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 200
   Unobserved stochastic nodes: 3
   Total graph size: 1007

Initializing model
```

```
set.seed(8)
update(fit, 6000)

set.seed(8)
sample = coda.samples(fit, params, n.iter = 15000, thin = 1)

plot(sample)
```



Las gráficas de las simulaciones se miran bien, pues dan pinta de que estas convergen.

```
Iterations = 7001:22000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 15000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

              Mean     SD  Naive SE Time-series SE
Beta1     -0.08753 0.0129 6.083e-05       0.001204
Beta2      1.24079 0.1464 6.902e-04       0.010770
Constante -0.07034 1.0003 4.716e-03       0.059372

2. Quantiles for each variable:

              2.5%      25%      50%      75%    97.5%
Beta1     -0.1121 -0.09666 -0.08784 -0.07842 -0.06127
Beta2      0.9732  1.13146  1.23883  1.34711  1.52567
Constante -2.0074 -0.76413 -0.07412  0.62260  1.86046
```

Bajo este modelo, se observa que al menos los valores de *Beta1* y *Constante* son diferentes a los calculador por la regresión frecuentista. Incluso el valor *Constante* sale no significativo pues con probabilidad 0.95 el HDI contiene al 0.

Matriz de confusión.

```
   jags.aux.j1
y    0  1
  0 92  8
  1 10 90
```

Los resultados son similares, pero estríctamente no mejores que el modelo frecuentista. Esto debido al posible ruido que causa bajo este modelo la variable *Constante*.

Modelo Bayesiano con a priori normales con media parámetros frecuentistas.

Ahora, se calculará otra versión en el que se utilicen como medias de las normales a priori los valores proporcionados por la regresión frecuentista, donde:

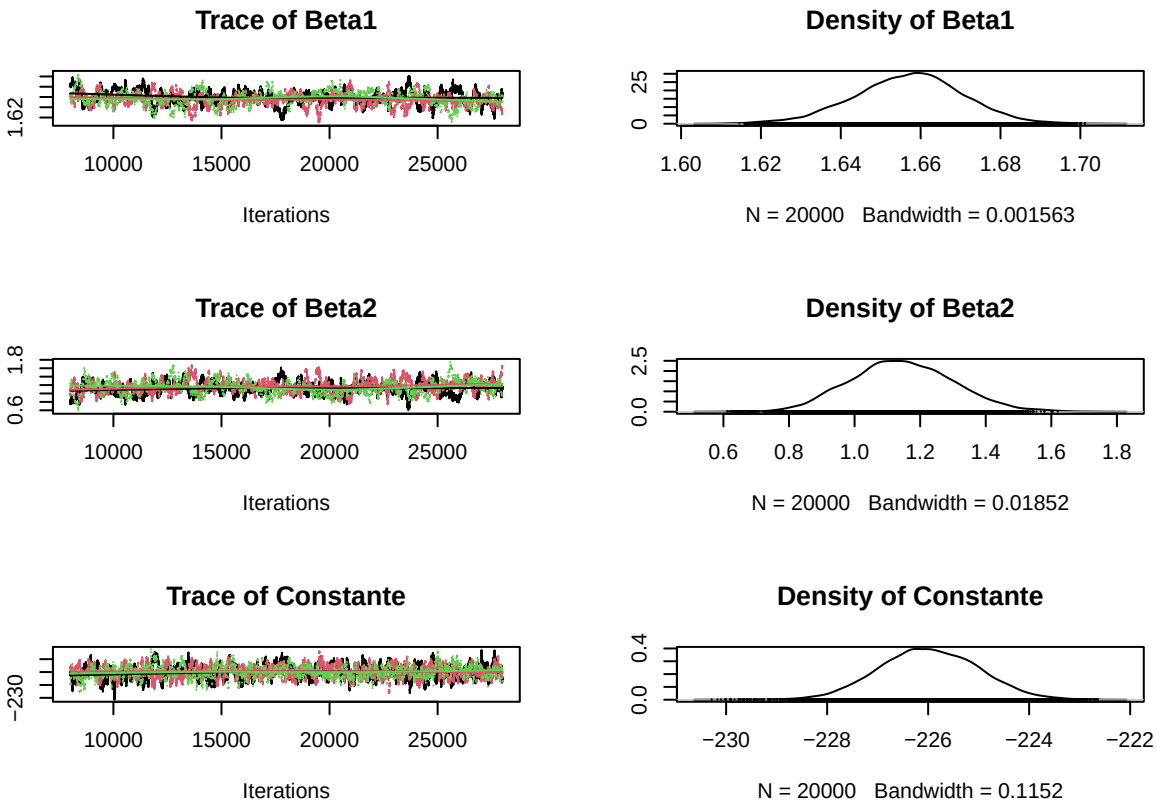
- Constate ~ Normal(-225.9755,1)
- Beta1 ~ Normal(1.6579,1)
- Beta2 ~ Normal(1.1317,1)

```
n = length(y)


data = list(
  y = y,
  x3 = x3,
  x4 = x4,
  n = n
)

params = c("Constante", "Beta1", "Beta2")

inits = function(){list(
  "Constante" = rnorm(1),
  "Beta1" = rnorm(1),
  "Beta2" = rnorm(1)
)
}

modelo = "model{

#### LIKELIHOOD

for(i in 1:n){

eta[i] = Constante + Beta1 * x3[i] + Beta2 * x4[i]
probit(p[i]) = eta[i]
y[i] ~dbern(p[i])

}

#### PRIORS
```

```
Beta1 ~ dnorm(1.6579,1) #
Beta2 ~ dnorm(1.1317,1) #
Constante ~ dnorm(-225.9755,1) #


}
"
set.seed(8)
fit = jags.model(file = textConnection(modelo), data = data, inits = inits, n.chains = 3 )
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 200
   Unobserved stochastic nodes: 3
   Total graph size: 1010

Initializing model
```

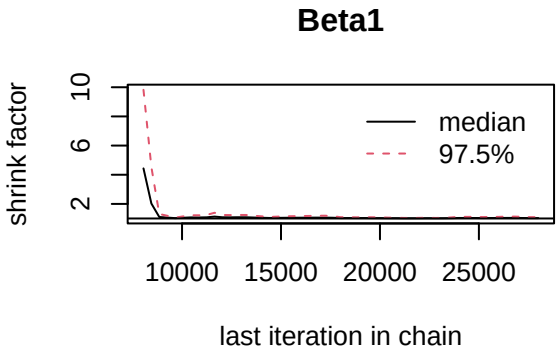
```
set.seed(8)
update(fit, 7000)

set.seed(8)
sample = coda.samples(fit, params, n.iter = 20000, thin = 1)

plot(sample)
```

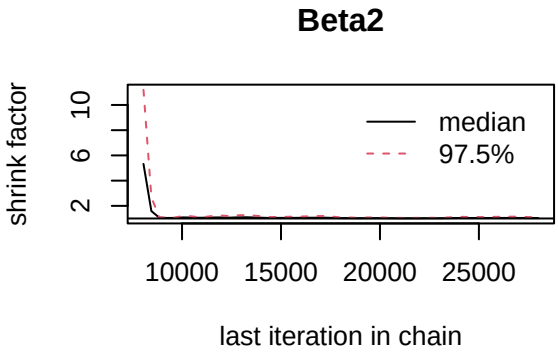


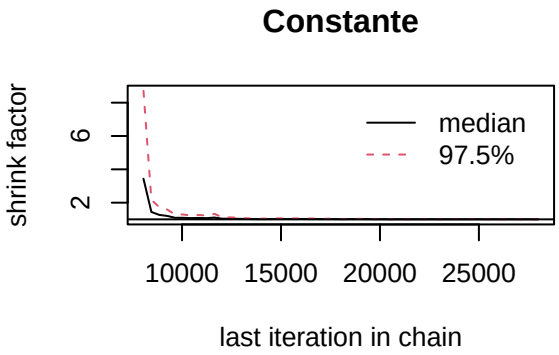
Las gráficas no son del todo convincentes en cuanto a la convergencia de las distribuciones.

```
Iterations = 8001:28000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 20000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

              Mean      SD  Naive SE Time-series SE
Beta1        1.657 0.01362 5.562e-05       0.001071
Beta2        1.146 0.15777 6.441e-04       0.010219
Constante -225.983 0.98122 4.006e-03       0.044135

2. Quantiles for each variable:

               2.5%      25%      50%      75%    97.5%
Beta1        1.6290    1.648    1.657    1.666    1.683
Beta2        0.8533    1.038    1.141    1.251    1.463
Constante -227.8658 -226.653 -225.995 -225.308 -224.069
```

Observar que ahora las estimaciones son más parecidas a las del ajuste frecuentista. En este caso, *Constante* ya es estadísticamente significativo para el modelo.







La gráfica de Gelman confirma que las simulaciones si convergen pese a la confusa distribución que se muestra en sus trazas.

Matriz de confusión.

```
   jags.aux.j1
y    0  1
  0 91  9
  1  8 92
```

Se obtienen los mismos resultados que con el modelo frecuentista.

Modelo Bayesiano con a priori normales con media parámetros frecuentistas y variable latente.

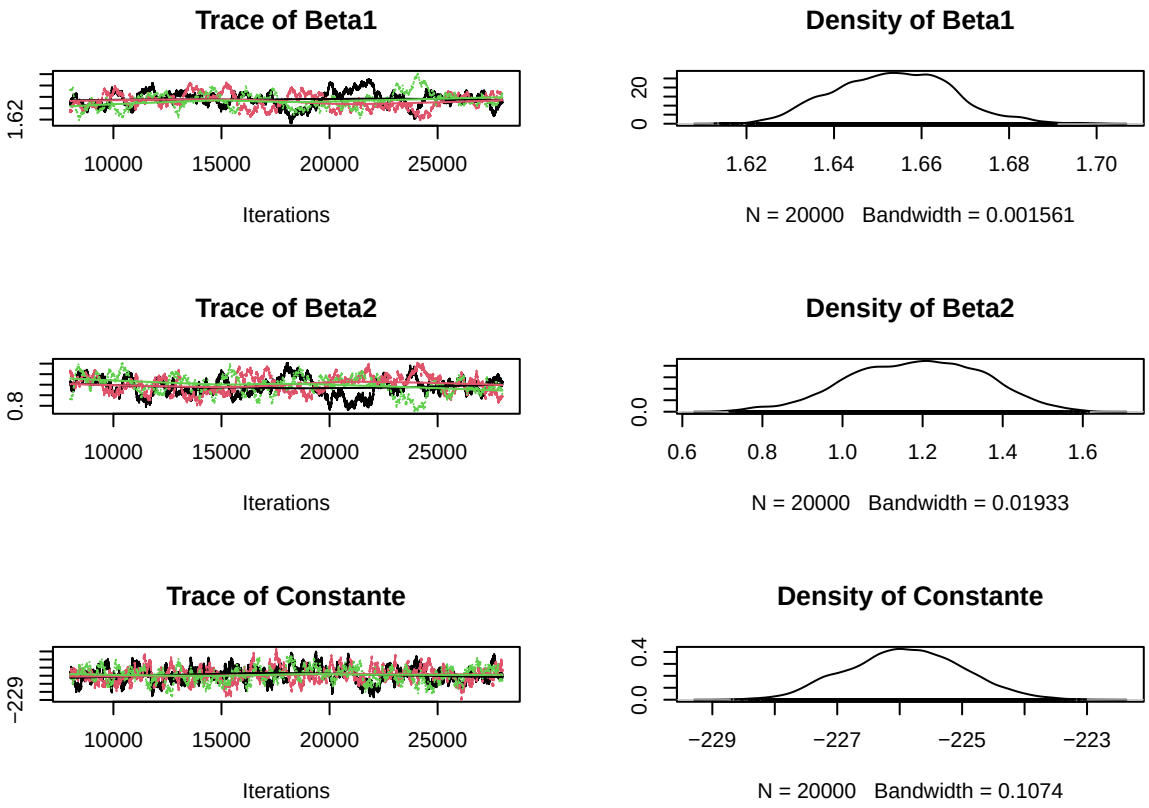
En este modelo se suponen las mismas distribuciones pero se agregará una variable latente.

```
inits2 = function(){list(
  "Constante" = rnorm(1),
  "Beta1" = rnorm(1),
  "Beta2" = rnorm(1),
  "z" = rep(0,n)
)}
```

```
modelo2 = "model{

#### LIKELIHOOD

for(i in 1:n){

eta[i] = Constante + Beta1 * x3[i] + Beta2 * x4[i]
z[i] ~ dnorm(eta[i], 1)
p[i] = step(z[i]) * 0.99999999
y[i] ~ dbern(p[i])

}

#### PRIORS

Beta1 ~ dnorm(1.6579,1) # Con uno funciona bien en todas
Beta2 ~ dnorm(1.1317,1)
Constante ~ dnorm(-225.9755,1)


}
"
set.seed(8)
fit2 = jags.model(file = textConnection(modelo2), data = data, inits = inits2, n.chains = 3 )
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 200
   Unobserved stochastic nodes: 203
   Total graph size: 1445

Initializing model
```

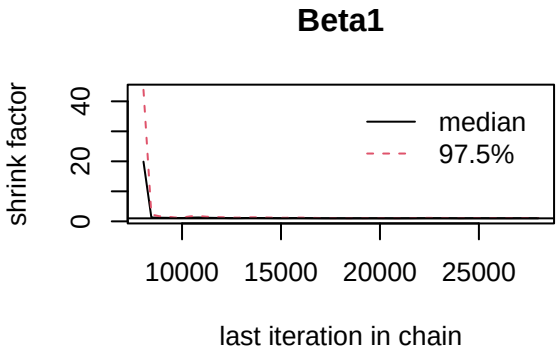
```
set.seed(8)
update(fit2, 7000)

set.seed(8)
sample2 = coda.samples(fit2, params, n.iter = 20000, thin = 1)


plot(sample2)
```

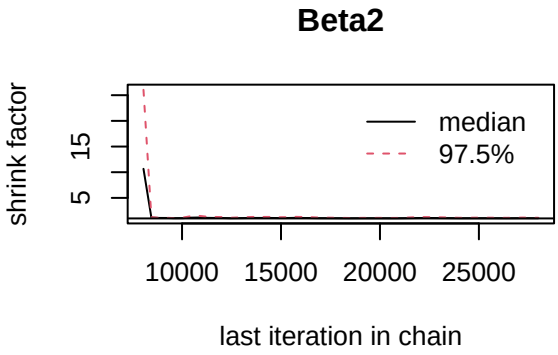


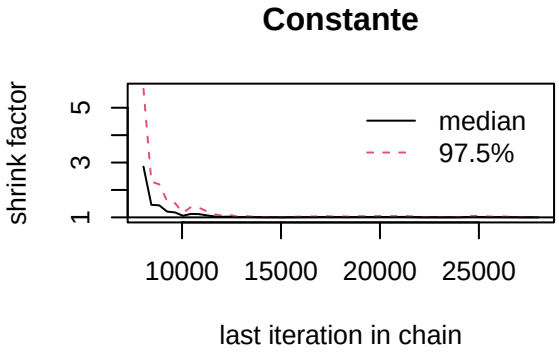
Las trazas no se ven, como en el caso anterior, del todo convincentes como para asegurar que convergen. Todas las distribuciones tienen en la punta de la densidad un "hoyo". Esto podría sugerir que se necesitan más de 20,000 iteraciones para poder obtener una mejor estimación.

```
Iterations = 8001:28000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 20000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

              Mean      SD  Naive SE Time-series SE
Beta1        1.654 0.01329 5.427e-05       0.001855
Beta2        1.185 0.16466 6.722e-04       0.020343
Constante -225.908 0.91498 3.735e-03       0.069104

2. Quantiles for each variable:

               2.5%      25%      50%      75%    97.5%
Beta1        1.6291    1.644    1.654    1.663    1.681
Beta2        0.8565    1.066    1.190    1.307    1.487
Constante -227.6401 -226.534 -225.904 -225.275 -224.125
```

Los coeficientes lucen igualmente similares a los de la regresión anterior y a los de la frecuentista. Además, los 3 coeficientes resultan estadísticamente significativos.







La gráfica de Gelman nos asegura una convergencia. Hay que destacar que en cuanto al coeficiente *Beta1* su convergencia se da más tardiamente respecto a los otros dos coeficientes. Esto podría sugerir quemás más datos.

```
   jags.aux.j2
y     0  1
  0 91  9
  1  8 92
```

Se da exactamente el mismo resultado que en la frecuentista y en la anterior.

CONCLUSIÓN

Como punto de partida se puede calcular un modelo frecuentista para posteriormente tomarlos como información a priori. Así fue pues como se concluyó que las variables *x3* y *x4* fueron las que más describian la varianza de la variable *y*.

Se calcularon 3 modelos bayesianos:

1. El primer modelo tuvo distribuciones a priori poco informativas (Normal(0,1)). Los parámetros, exceptuando *Beta2*, diferieron en valor a los frecuentistas; lo que es más, la constante resultó estadísticamente cero. No obstante, los resultados en la tabla de confusión con un punto de corte 0.5 fueron casi identicos al bayesiano.

2. El segundo modelo tuvo distribuciones más representativas (Normal(*Parámetro_frecuentista* , 1)). En esta simulación los parámetros fueron muy similares a los frecuentistas. Mejor aún, los resultados en la tabla de confusión fueron los mismos que en el modelo frecuentista. Esto definitivamente representa la utilidad del enfoque bayesiano, aprovechando información previa se pudo llegar a un modelo similar con el mismo poder predictivo pero aprovechando la bondad de que los intervalos son probabilisticos.

3. El tercer modelo fue igual al anterior pero con una variable latente. En este caso se comportó exactamente igual que al anterior y por lo tanto, es muy parecido al frecuentista.