Modelos Lineales Generalizados (2o día)

Fernando Anorve

3/12/2020

Devianza

Lo primero que hay que definir para entender la devianza es el “modelo saturado”, es decir, un modelo completamente sobreajustado en el que a cada punto i se le asigne una media estimada μ_i. Básicamente, ¡un modelo donde no queden grados de libertad!

```
y <- c(0,0,0,1,1,1)
x <- factor(1:6)

saturated.model <- glm(y ~ x , family = binomial(link = "logit"))

summary(saturated.model)
```

```
##
## Call:
## glm(formula = y ~ x, family = binomial(link = "logit"))
##
## Deviance Residuals:
## [1]  0  0  0  0  0  0
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.457e+01  1.310e+05       0        1
## x2          -2.628e-14  1.853e+05       0        1
## x3          -5.360e-14  1.853e+05       0        1
## x4           4.913e+01  1.853e+05       0        1
## x5           4.913e+01  1.853e+05       0        1
## x6           4.913e+01  1.853e+05       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8.3178e+00  on 5  degrees of freedom
## Residual deviance: 2.5720e-10  on 0  degrees of freedom
## AIC: 12
##
## Number of Fisher Scoring iterations: 23
```

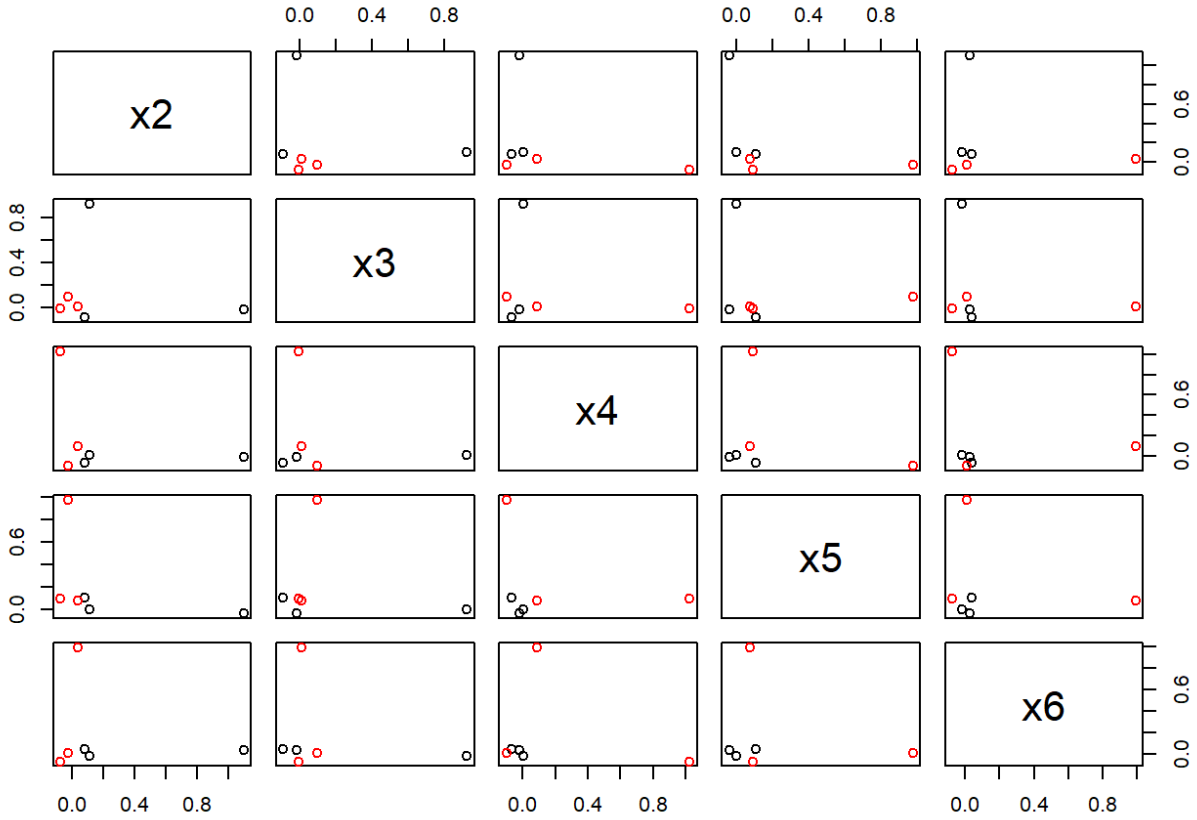
Notamos que no quedan grados de libertad de residuo. Por eso está saturado.

A diferencia del ejemplo de ayer, este modelo no tiene una separación lineal perfecta.

```
mod.mat = model.matrix(y ~ x)

x2 <- mod.mat[,2] + rnorm(6,sd = 0.07)
x3 <- mod.mat[,3] + rnorm(6,sd = 0.07)
x4 <- mod.mat[,4] + rnorm(6,sd = 0.07)
x5 <- mod.mat[,5] + rnorm(6,sd = 0.07)
x6 <- mod.mat[,6] + rnorm(6,sd = 0.07)

X <- data.frame(x2,x3,x4,x5,x6)
pairs(X,col = y+1)
```



La devianza de un modelo se calcula como:

$$G^2 = -2\log\left(\frac{L_M}{L_S}\right) = -2(loglik(modelo) - loglik(saturado)) = 2(loglik(saturado) - loglik(modelo))$$

Como el modelo saturado tiene sobreajuste, $loglik(modelo) \leq loglik(saturado)$, i.e. $G^2 \geq 0$

Devianza nula: Diferencia entre un modelo sin covariables y el modelo saturado

Devianza del modelo (a.k.a. devianza residual): Diferencia entre nuestro modelo (usualmente con al menos una covariable) y el modelo saturado

```
x_1 = c(1,2,4,3,5,6)

random_model <- glm(y ~ x_1 , family = binomial(link = "logit"))

summary(random_model)
```

```
##
## Call:
## glm(formula = y ~ x_1, family = binomial(link = "logit"))
##
## Deviance Residuals:
##       1        2        3        4        5        6
## -0.3064  -0.5478  -1.4436   1.4436   0.5478   0.3064
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.2491     3.3878  -1.254    0.210
## x_1           1.2140     0.9126   1.330    0.183
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8.3178  on 5  degrees of freedom
## Residual deviance: 4.9560  on 4  degrees of freedom
## AIC: 8.956
##
## Number of Fisher Scoring iterations: 5
```

En el primer ejemplo no hay devianza residual porque la devianza del modelo saturado y la devianza del primer modelo son iguales, ¡justo porque es el saturado! :)

Ejemplo 1 - Regresión binomial

Este ejemplo es bastante ilustrativo (afortunadamente, los datos se comportan bien. No siempre es así en la vida real. De hecho, en mi experiencia, casi nunca). La librería MASS incluye el conjunto de datos **menarche** (Milicer, H. and Szczotka, F., 1966, Age at Menarche in Warsaw girls in 1965, Human Biology, 38, 199-203), en el que hay tres variables:

- “Age”: la edad promedio de grupos homogéneos de niñas/adolescentes
- “Total”: número de niñas/adolescentes en cada grupo
- “Menarche”: número de niñas/adolescentes en el grupo que han alcanzado la menarquía

Para ajustar un modelo de regresión binomial se hacen los siguientes ajustes:

- Las respuestas y_i son proporciones de 0 a 1
- $y_i \sim Binom(n_i, p_i)$
- En el parámetro $weights$ se indica el valor de n_i para ese valor de y_i

```
library(MASS)
data("menarche")

fit <- glm(formula = Menarche/Total ~ Age , family = binomial(link = logit),
           data = menarche, weights = Total)

summary(fit)
```

```
##
## Call:
## glm(formula = Menarche/Total ~ Age, family = binomial(link = logit),
##     data = menarche, weights = Total)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0363  -0.9953  -0.4900   0.7780   1.3675
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -21.22639    0.77068  -27.54   <2e-16 ***
## Age           1.63197    0.05895   27.68   <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: 3693.884  on 24  degrees of freedom
## Residual deviance:   26.703  on 23  degrees of freedom
## AIC: 114.76
##
## Number of Fisher Scoring iterations: 4
```

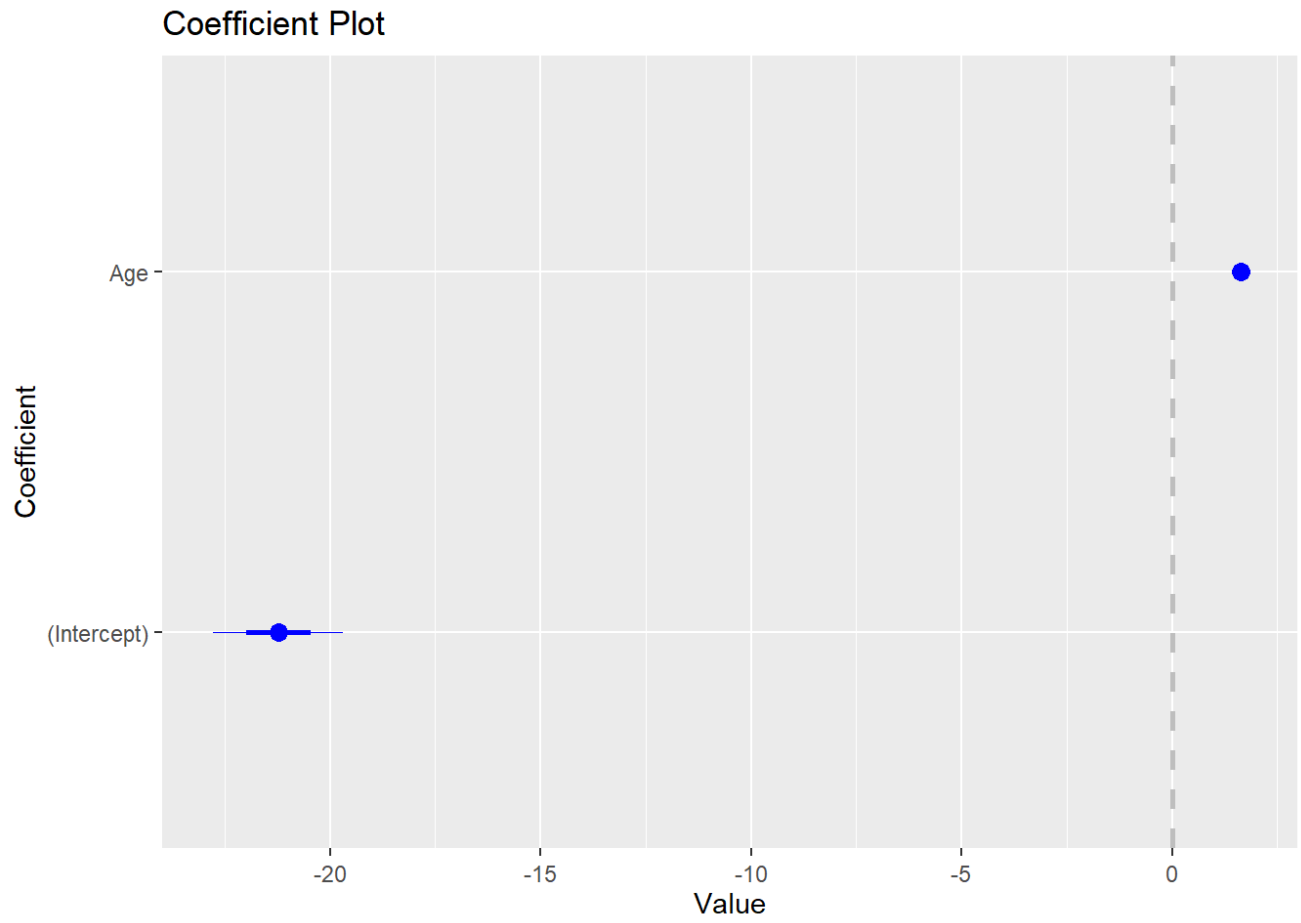
A mayor edad promedio del grupo x_i, mayor probabilidad p_i. Tiene sentido!

El estimador del efecto de la edad se puede interpretar como que por cada incremento de un año en Age, las posibilidades (odds) de haber alcanzado la menarquía incrementan por un factor de $\exp(1.63197) = 5.11$.

```
library(coefplot)
```

```
## Loading required package: ggplot2
```

```
coefplot(fit)
```



Observamos que la devianza residual es mucho menor que la devianza nula. Eso nos sugiere que el modelo efectivamente aporta información.

```
fit0 <- glm(formula = Menarche/Total ~ 1 , family = binomial(link = logit),
           data = menarche, weights = Total)

anova(fit0,fit, test = "Chisq")
```

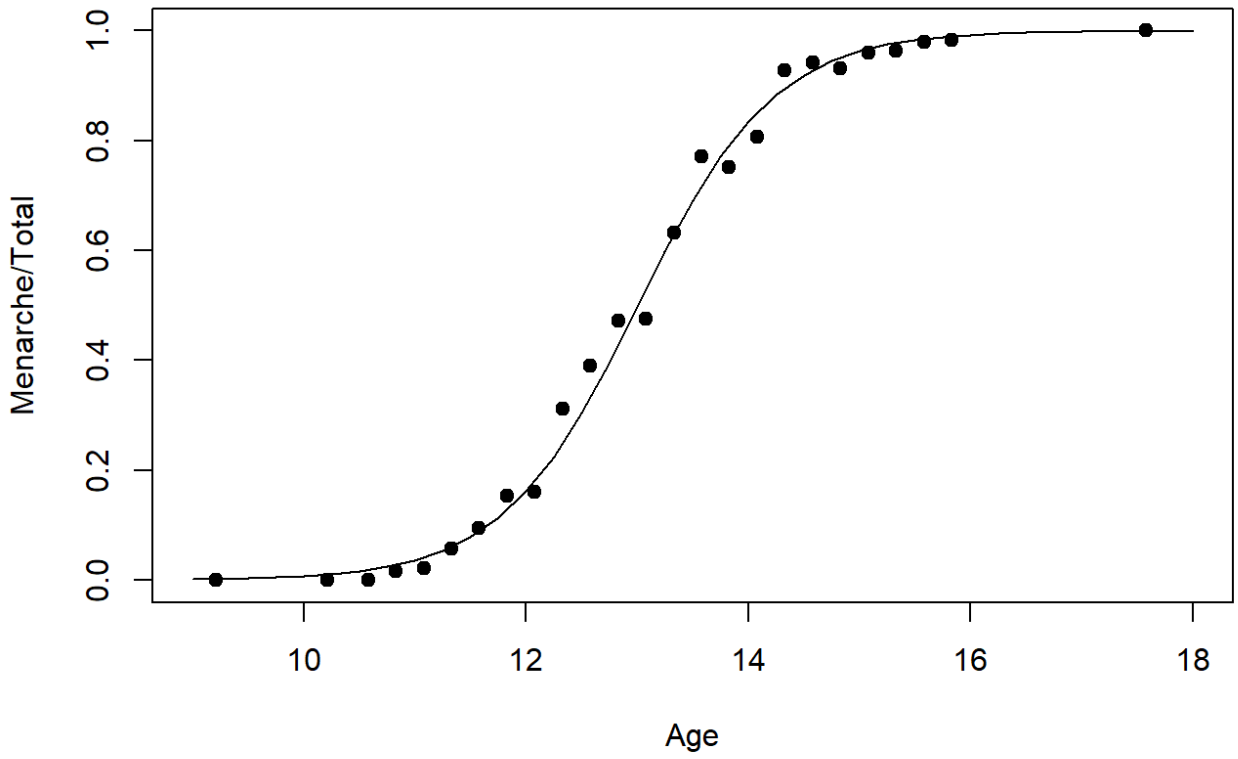
```
## Analysis of Deviance Table
##
## Model 1: Menarche/Total ~ 1
## Model 2: Menarche/Total ~ Age
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1        24     3693.9
## 2        23       26.7  1   3667.2 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

La diferencia de 3667.2 es bastante significativa, sugiriendo que el modelo que utiliza la edad promedio para predecir es de hecho útil.

Podemos graficar la curva de probabilidades estimadas como:

```
new_data = data.frame("Age" = seq(from = 9 , to = 18 , by = 0.25))
logit_hat = predict.glm(fit,newdata = new_data,se.fit = T)
prob_hat=exp(logit_hat$fit)/(1+exp(logit_hat$fit))

plot(menarche$Age,menarche$Menarche/menarche$Total,xlab="Age",ylab="Menarche/Total", pch=20,cex=1.
5,xlim=c(9,18))
lines(new_data$Age,prob_hat)
```

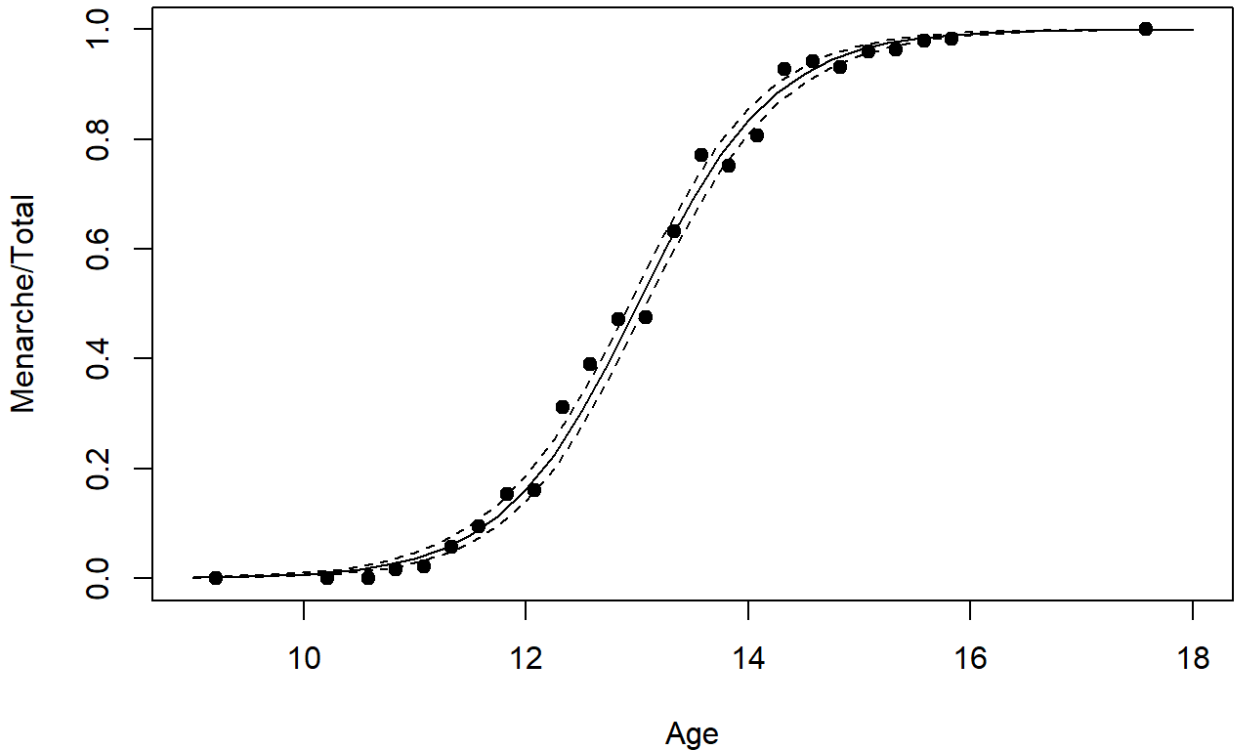


Cuando agregamos los intervalos de confianza, notamos que son bastante angostos.

(Nota: hay que recordar que los intervalos de confianza son estimados y además son puntuales, por lo que sólo nos sirven como referencia en este caso. No confundir con regiones de confianza)

```
prob_lwr=exp(logit_hat$fit-1.96*logit_hat$se.fit)/(1+exp(logit_hat$fit-1.96*logit_hat$se.fit))
prob_upr=exp(logit_hat$fit+1.96*logit_hat$se.fit)/(1+exp(logit_hat$fit+1.96*logit_hat$se.fit))
plot(menarche$Age,menarche$Menarche/menarche$Total,xlab="Age",ylab="Menarche/Total", pch=20,cex=1.
5,xlim=c(9,18))
lines(new_data$Age,prob_hat)

lines(new_data$Age,prob_hat)
lines(new_data$Age,prob_lwr,lty=2)
lines(new_data$Age,prob_upr,lty=2)
```



Ejemplo 2 - Regresión Poisson

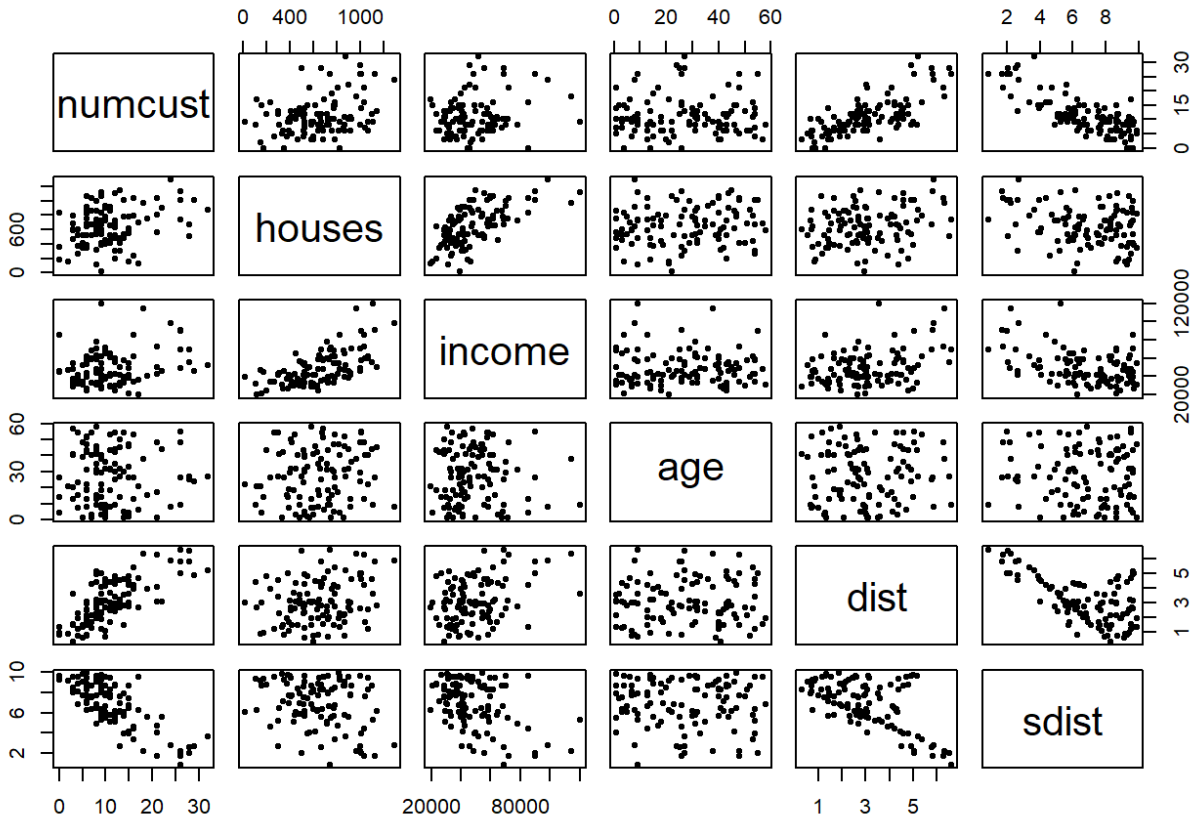
Veamos en el ejemplo de una tienda de mejoramiento del hogar y ferretería hace una serie de encuestas durante un periodo de tiempo representativo en el que le pregunta a sus clientes por su dirección. Dichas direcciones son utilizadas para identificar en qué zona del área metropolitana reside el cliente, y para obtener información demográfica de dicha zona de acuerdo con datos censales (por ejemplo en bases de datos del Inegi).

Por cada zona censal, se obtienen las siguientes variables:

- **houses**: número de unidades habitacionales
- **income** ingreso promedio de la zona (en dólares)
- **age**: antigüedad promedio de las unidades habitacionales
- **dist**: distancia en millas del competidos más cercano
- **sdist**: distancia en millas a la tienda
- **numcust**: número de clientes de esa zona censal que visitaron la tienda

```
dat=read.table("miller.txt", header = FALSE)
colnames(dat)=c("numcust","houses","income","age","dist","sdist")

# Plot counts versus predictors
plot(dat,pch=20)
```



¿Qué patrones se pueden observar?

- **houses** parece tener una relación más bien débil con el número de clientes
- Entre el ingreso y el número de clientes podría haber correlación… ¿tal vez?
- **dist** y **sdist** tienen una relación más fuerte, aunque en direcciones opuestas (tiene mucho sentido)

Se ajusta el modelo como sigue:

```
fit = glm(numcust ~ ., family = poisson(link="log"), data = dat)
summary(fit)
```

```
##
## Call:
## glm(formula = numcust ~ ., family = poisson(link = "log"), data = dat)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.93195  -0.58868  -0.00009   0.59269   2.23441
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.942e+00  2.072e-01  14.198  < 2e-16 ***
## houses       6.058e-04  1.421e-04   4.262 2.02e-05 ***
## income      -1.169e-05  2.112e-06  -5.534 3.13e-08 ***
## age         -3.726e-03  1.782e-03  -2.091   0.0365 *
## dist         1.684e-01  2.577e-02   6.534 6.39e-11 ***
## sdist       -1.288e-01  1.620e-02  -7.948 1.89e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 422.22  on 109  degrees of freedom
## Residual deviance: 114.99  on 104  degrees of freedom
## AIC: 571.02
##
## Number of Fisher Scoring iterations: 4
```

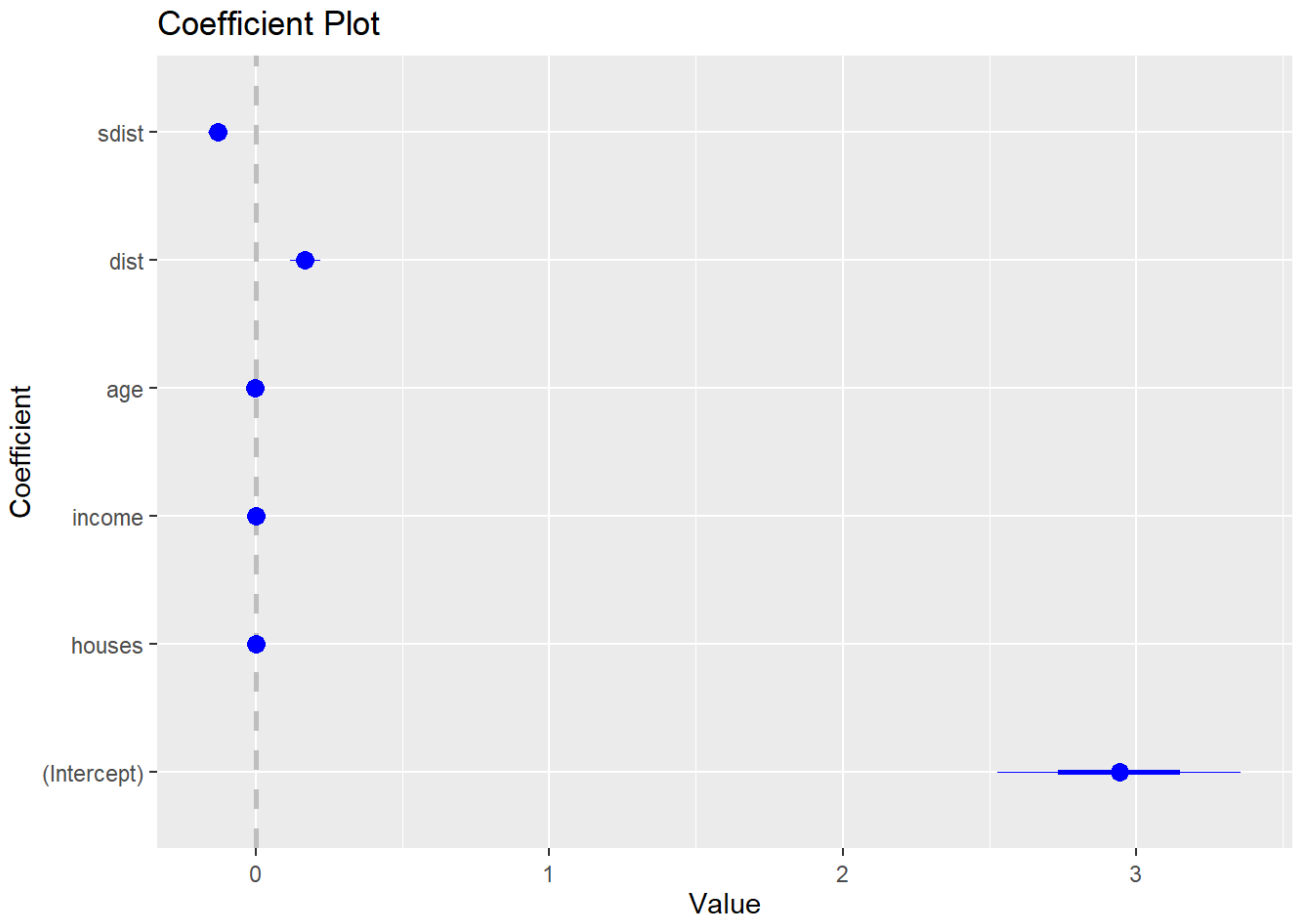
Podemos calcular los intervalos de confianza:

```
# we can also get confidence intervals
confint(fit)
```

```
## Waiting for profiling to be done...
```

```
##                    2.5 %        97.5 %
## (Intercept)  2.536768e+00  3.349269e+00
## houses       3.273675e-04  8.845217e-04
## income      -1.585282e-05 -7.574589e-06
## age         -7.222279e-03 -2.361904e-04
## dist         1.176987e-01  2.187234e-01
## sdist       -1.608136e-01 -9.728963e-02
```

```
coefplot(fit)
```



Recordemos que en este modelo:

$$g(E(Y_i)) = \log(\lambda_i) = \beta^T \mathbf{x}_i = \beta_0 + x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4 + x_5\beta_5$$

Es decir, el modelo de la i-ésima ocurrencia media sería

$$\lambda_i = \exp(\beta_0 + x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4 + x_5\beta_5)$$

Por lo que tiene sentido estudiar los coeficientes estimado en escala exponencial

```
exp(coef(fit))
```

```
## (Intercept)      houses      income         age        dist       sdist
##  18.9620188   1.0006060   0.9999883   0.9962805   1.1833897   0.8791728
```

```
exp(confint(fit))
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %     97.5 %
## (Intercept) 12.6387529 28.4819069
## houses       1.0003274  1.0008849
## income       0.9999841  0.9999924
## age          0.9928037  0.9997638
## dist         1.1249051  1.2444870
## sdist        0.8514508  0.9072932
```

- Un incremento de una unidad en **dist** se relaciona aproximadamente con un incremento (multiplicativo) con factor 1.1833897 de la ocurrencia media de número de clientes.
- En cambio, Un incremento de una unidad en **houses** se relaciona aproximadamente con un incremento (multiplicativo) con factor 1 de la ocurrencia media de número de clientes. Es decir, hay relativamente poco efecto.

```
set.seed(2020)

new_data = dat[sample(110,10),c(2,3,4,5,6,1)]

fitted <- predict.glm(fit,newdata = new_data,type = "response", se.fit = T)

new_data$fitted <- fitted$fit

new_data
```

```
##     houses income age dist sdist numcust    fitted
## 28     925  70030  36 4.58  8.66      10  9.081660
## 108    817  54429  47 1.90  9.90       6  5.318607
## 87     758  40305  15 3.95  5.58      19 16.798005
## 22     302  42191  54 3.41  5.21      12 10.323034
## 88    1141  50026  45 2.79  6.18      13 12.874979
## 65     669  34595  38 4.06  8.78       9 10.536143
## 17     377  36828  15 1.92  8.91       4  6.426701
## 36     392  36998   7 1.03  7.74      10  6.673720
## 42     643  58315   8 0.78  6.26       6  6.999761
## 70     551  41045   2 3.62  7.45      15 11.464892
```

Se puede también analizar con tabla anova

```
anova(fit,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: numcust
##
## Terms added sequentially (first to last)
##
##
##         Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                      109     422.22
## houses   1   42.662       108     379.56 6.507e-11 ***
## income   1    0.807       107     378.75    0.3691
## age      1    0.316       106     378.43    0.5741
## dist     1  195.949       105     182.49 < 2.2e-16 ***
## sdist    1   67.500       104     114.99 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Y comparar con el modelo naive

```
fit0=glm(numcust ~ 1, family = poisson(link="log"), data = dat)
anova(fit0,fit,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: numcust ~ 1
## Model 2: numcust ~ houses + income + age + dist + sdist
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       109     422.22
## 2       104     114.99  5   307.23 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

De nuevo, nuestrio modelo parece ser informativo.

Veamos un modelo que use menos variables y comparémoslo:

```
fit2=glm(numcust~houses+income+dist+sdist,family=poisson(link="log"),data=dat)
anova(fit2,fit,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: numcust ~ houses + income + dist + sdist
## Model 2: numcust ~ houses + income + age + dist + sdist
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       105     119.36
## 2       104     114.98  1   4.3792  0.03638 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Hace falta la edad al parecer!

En la librería car hay otro tipo de tabla de Anova… Es importante notar que hay cierta diferencia, aunque no nos centraremos en ello por lo pronto

```
library(car)
```

```
## Loading required package: carData
```

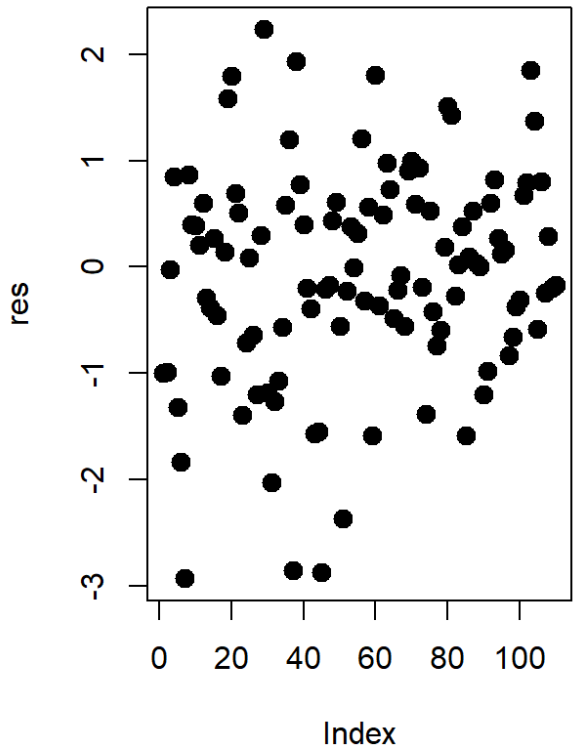
```
Anova(fit)
```

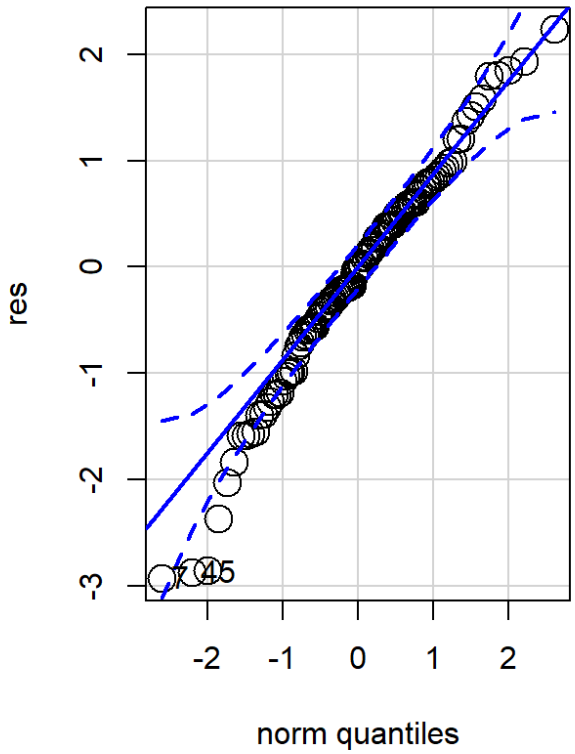
```
## Analysis of Deviance Table (Type II tests)
##
## Response: numcust
##        LR Chisq Df Pr(>Chisq)
## houses   18.203  1  1.986e-05 ***
## income   31.794  1  1.714e-08 ***
## age       4.379  1    0.03638 *
## dist     41.660  1  1.086e-10 ***
## sdist    67.500  1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Parece que todas las variables podrían ser significativas

En este modelo los residuales parecen más amistosos que en el caso logístico

```
# Residual plots
res=residuals(fit)
par(mfrow=c(1,2))
plot(res,pch=20,cex=2)
qqPlot(res,cex=2)
```

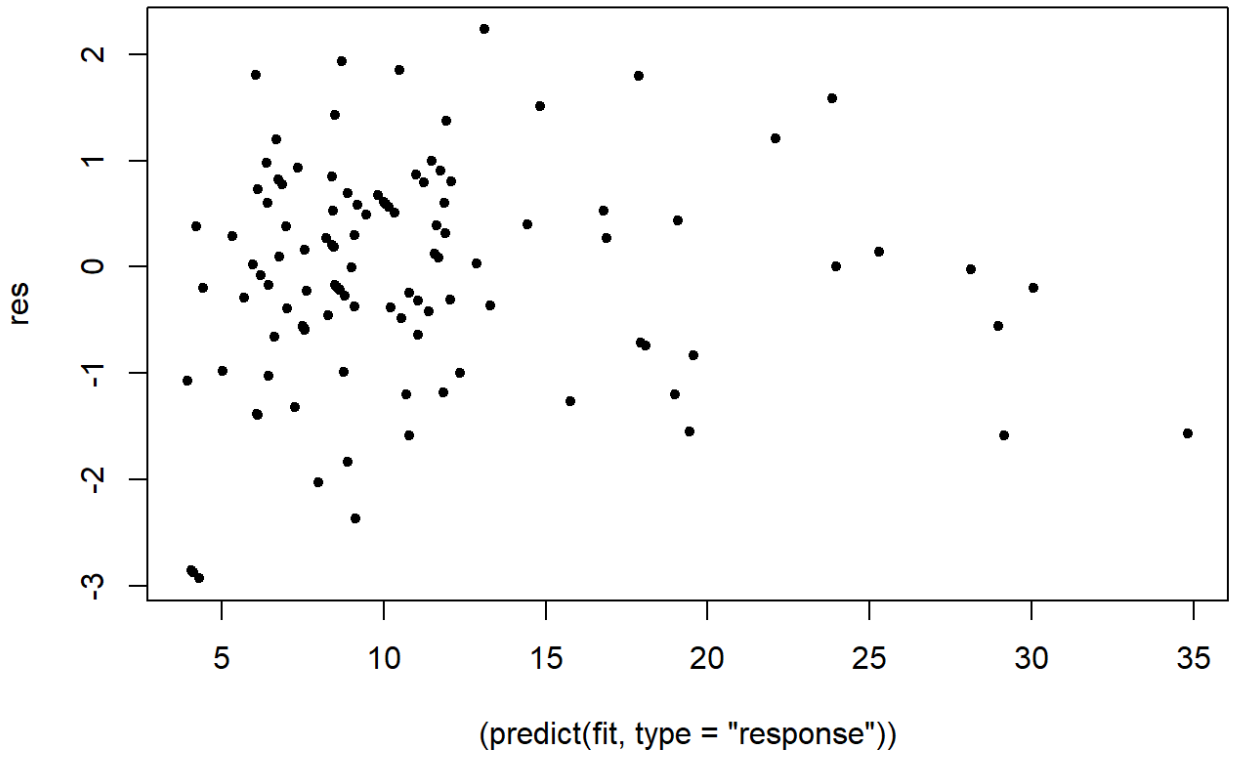





```
## [1]  7 45
```

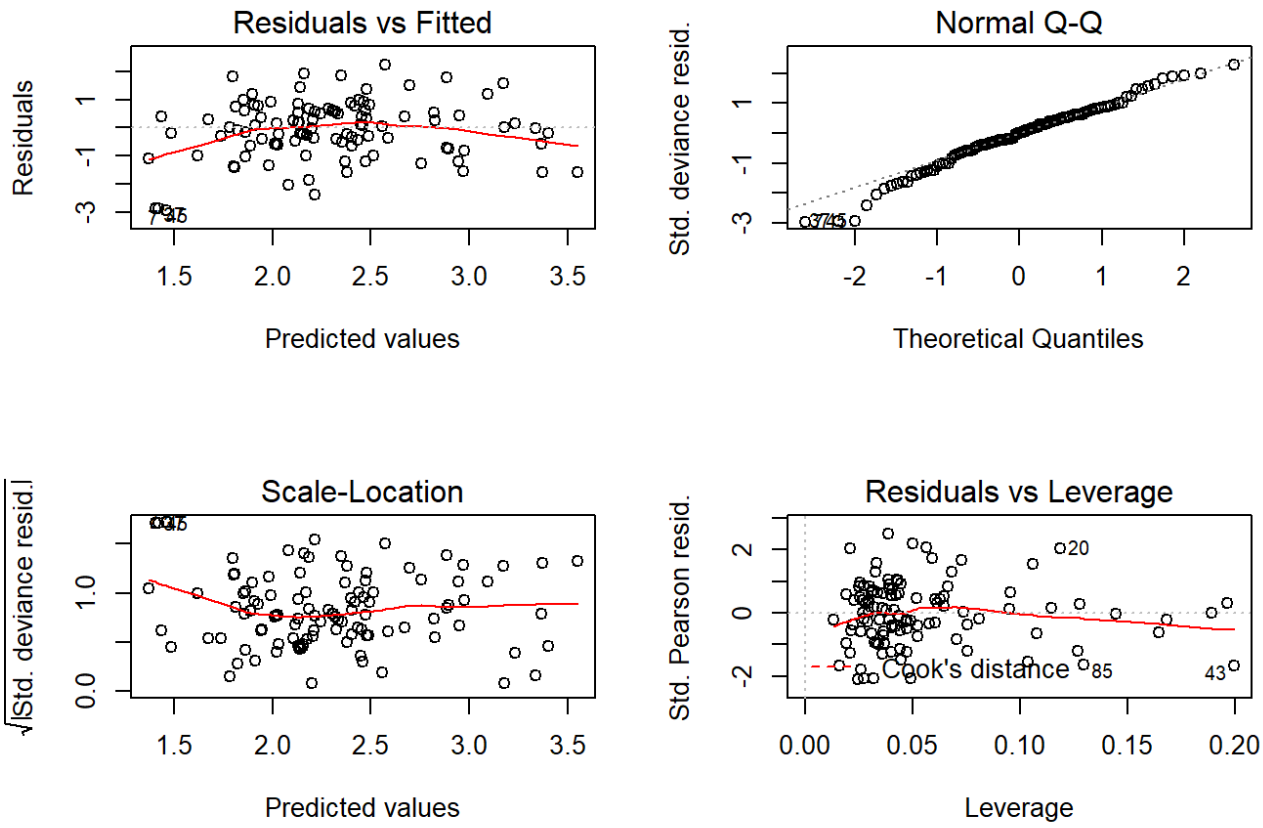
Al graficar los residuales vs los valores de ajuste podríamos sospechar que la varianza cambia según el valor ajustado, aunque con tan pocas observaciones a la derecha de la gráfica es difícil hacer inferencias

```
par(mfrow=c(1,1))
plot((predict(fit,type="response")),res,pch=20)
```



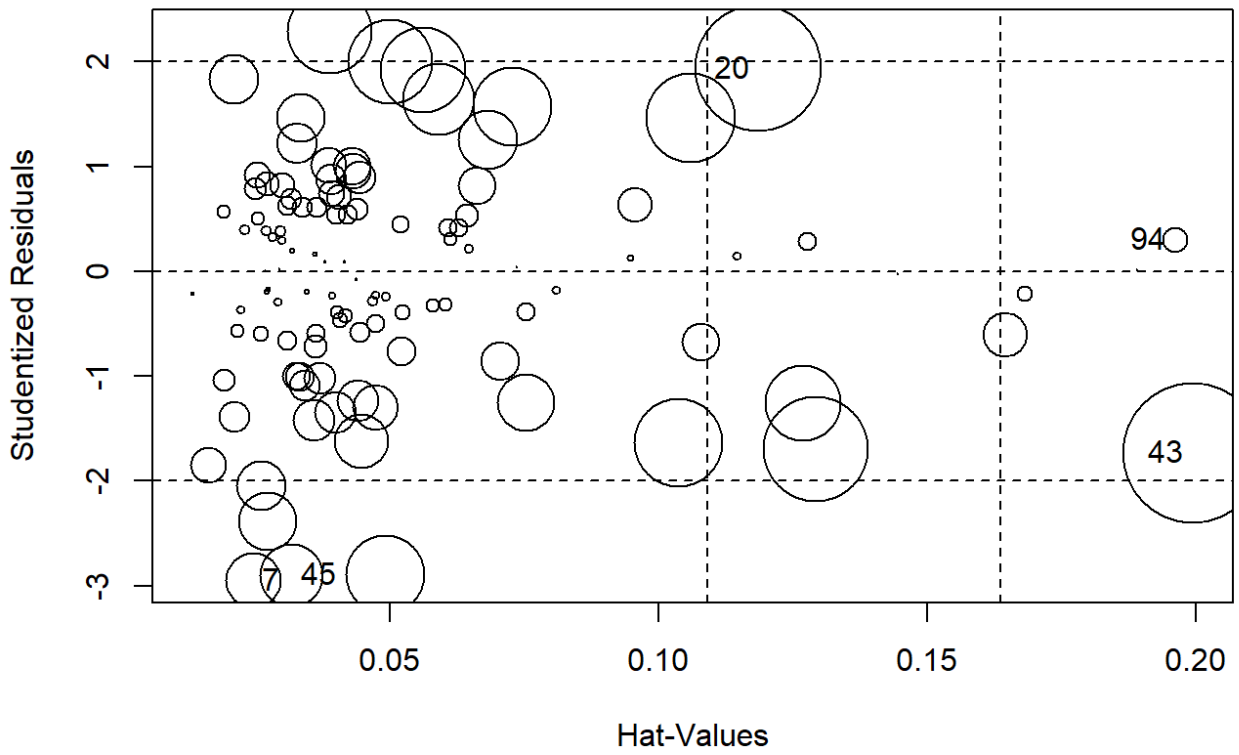
Podemos hacer gráficos como lo hicimos para el modelo lineal

```
# R's plot.glm also gives some additional nice plots
par(mfrow=c(2,2))
plot(fit)
```



Por último, la gráfica que muestra potenciales valores influyentes

```
influencePlot(fit)
```




```
##       StudRes        Hat       CookD
## 7  -2.9503470 0.02455516 0.018487156
## 20  1.9339055 0.11862291 0.094230654
## 43 -1.7352076 0.19945629 0.116008681
## 45 -2.8978368 0.03173734 0.023307243
## 94  0.3002519 0.19613487 0.003729944
```