Diego Regresión 2

2023-03-07

```
#PREGUNTA 1

library(faraway)
teengamb

##    sex status income verbal gamble
## 1    1     51   2.00      8   0.00
## 2    1     28   2.50      8   0.00
## 3    1     37   2.00      6   0.00
## 4    1     28   7.00      4   7.30
## 5    1     65   2.00      8  19.60
## 6    1     61   3.47      6   0.10
## 7    1     28   5.50      7   1.45
## 8    1     27   6.42      5   6.60
## 9    1     43   2.00      6   1.70
## 10   1     18   6.00      7   0.10
## 11   1     18   3.00      6   0.10
## 12   1     43   4.75      6   5.40
## 13   1     30   2.20      4   1.20
## 14   1     28   2.00      6   3.60
## 15   1     38   3.00      6   2.40
## 16   1     38   1.50      8   3.40
## 17   1     28   9.50      8   0.10
## 18   1     18  10.00      5   8.40
## 19   1     43   4.00      8  12.00
## 20   0     51   3.50      9   0.00
## 21   0     62   3.00      8   1.00
## 22   0     47   2.50      9   1.20
## 23   0     43   3.50      5   0.10
## 24   0     27  10.00      4 156.00
## 25   0     71   6.50      7  38.50
## 26   0     38   1.50      7   2.10
## 27   0     51   5.44      4  14.50
## 28   0     38   1.00      6   3.00
## 29   0     51   0.60      7   0.60
## 30   0     62   5.50      8   9.60
## 31   0     18  12.00      2  88.00
## 32   0     30   7.00      7  53.20
## 33   0     38  15.00      7  90.00
## 34   0     71   2.00     10   3.00
## 35   0     28   1.50      1  14.10
## 36   0     61   4.50      8  70.00
## 37   0     71   2.50      7  38.50
## 38   0     28   8.00      6  57.20
```

```
## 39   0     51  10.00      6   6.00
## 40   0     65   1.60      6  25.00
## 41   0     48   2.00      9   6.90
## 42   0     61  15.00      9  69.70
## 43   0     75   3.00      8  13.30
## 44   0     66   3.25      9   0.60
## 45   0     62   4.94      6  38.00
## 46   0     71   1.50      7  14.40
## 47   0     71   2.50      9  19.20
```

```
model <- lm(gamble ~ sex + status + income + verbal, data = teengamb)
summary(model)
```

```
## 
## Call:
## lm(formula = gamble ~ sex + status + income + verbal, data = teengamb)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -51.082 -11.320  -1.451   9.452  94.252 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  22.55565   17.19680   1.312   0.1968    
## sex         -22.11833    8.21111  -2.694   0.0101 *  
## status        0.05223    0.28111   0.186   0.8535    
## income        4.96198    1.02539   4.839 1.79e-05 ***
## verbal       -2.95949    2.17215  -1.362   0.1803    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 22.69 on 42 degrees of freedom
## Multiple R-squared:  0.5267, Adjusted R-squared:  0.4816 
## F-statistic: 11.69 on 4 and 42 DF,  p-value: 1.815e-06
```

```
#A partir de los resultados, podemos ver que el modelo es significativo
(p < 0.001) y explica una proporción significativa de la variación en la
variable de respuesta (apuesta). El valor de R-cuadrado múltiple es de
0.5267, lo que significa que aproximadamente el 52,67% de la variación en
la respuesta es explicada por los predictores en el modelo. El valor de
R-cuadrado ajustado es de 0.4816, lo que tiene en cuenta el número de
predictores en el modelo.
residuals <- resid(model)
which.max(residuals)
```

```
## 24 
## 24
```

```
mean_residuals <- mean(residuals)
mean_residuals
```

```
## [1] -3.065293e-17

median_residuals <- median(residuals)
median_residuals

## [1] -1.451392

fitted_values <- fitted(model)
correlation <- cor(residuals, fitted_values)
correlation

## [1] -1.070659e-16

cor(residuals, teengamb$income)

## [1] -7.242382e-17

# Establecer los valores predictores a sus medias
mean_status <- mean(teengamb$status)
mean_income <- mean(teengamb$income)
mean_verbal <- mean(teengamb$verbal)
mean_sex <- mean(teengamb$sex)
mean_gamble <- mean(teengamb$gamble)

# Crea un marco de datos con los valores predictores para una mujer
female <- data.frame(status = mean_status,
                      income = mean_income,
                      verbal = mean_verbal,
                     gamble = mean_gamble,
                      sex = 0)

# Crea un marco de datos con los valores predictores para un hombre
male <- data.frame(status = mean_status,
                    income = mean_income,
                    verbal = mean_verbal,
                   gamble = mean_gamble,
                    sex = 1)

# Predice el gasto en juegos de azar para una mujer y un hombre
pred_female <- predict(model, newdata = female)
pred_male <- predict(model, newdata = male)
pred_female

##        1
## 28.24252

pred_male

##        1
## 6.124186
```

```r
#pred_female > pred male
# Calcula la diferencia en el gasto en juegos de azar predicho entre un hombre y una mujer.

diff_pred <- abs(pred_male - pred_female)

# Imprimir la diferencia en el gasto predicho en juegos de azar.

diff_pred

##        1
## 22.11833

#22.11 es la diferencia en el gasto predicho en juegos de azar entre un hombre y una mujer, manteniendo constantes todos los demás predictores.

model2 <- lm(wage ~ educ + exper, data = uswages)
summary(model2)

##
## Call:
## lm(formula = wage ~ educ + exper, data = uswages)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1018.2  -237.9   -50.9   149.9  7228.6
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -242.7994    50.6816  -4.791 1.78e-06 ***
## educ          51.1753     3.3419  15.313  < 2e-16 ***
## exper          9.7748     0.7506  13.023  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 427.9 on 1997 degrees of freedom
## Multiple R-squared:  0.1351, Adjusted R-squared:  0.1343
## F-statistic:   156 on 2 and 1997 DF,  p-value: < 2.2e-16

model_log2 <- lm(log(wage) ~ educ + exper, data = uswages)
summary(model_log2)

##
## Call:
## lm(formula = log(wage) ~ educ + exper, data = uswages)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7533 -0.3495  0.1068  0.4381  3.5699
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.650319   0.078354   59.35   <2e-16 ***
## educ        0.090506   0.005167   17.52   <2e-16 ***
## exper       0.018079   0.001160   15.58   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6615 on 1997 degrees of freedom
## Multiple R-squared:  0.1749, Adjusted R-squared:  0.174
## F-statistic: 211.6 on 2 and 1997 DF,  p-value: < 2.2e-16
```

```
#La interpretación de este coeficiente es que la educación está asociada
positivamente con los salarios. Esto significa que, en promedio, las
personas con más años de educación ganan salarios más altos que aquellas
con menos años de educación. Observa que el coeficiente para "exper"
tiene una interpretación similar, lo que indica que, en promedio, las
personas con más experiencia laboral ganan salarios más altos que
aquellas con menos experiencia.

# Cargamos el dataset
data(prostate)

# Ajustar un modelo de regresión lineal con lpsa como respuesta y lcavol
como predictor.

model3 <- lm(lpsa ~ lcavol, data = prostate)

# Registrar el error estándar residual y R2.
summary(model3)$sigma
```

```
## [1] 0.7874994
```

```
summary(model3)$r.squared
```

```
## [1] 0.5394319
```

```
# Agregar lweight y ajustar el modelo.
model4 <- lm(lpsa ~ lcavol + lweight, data = prostate)
summary(model4)$sigma
```

```
## [1] 0.7506469
```

```
summary(model4)$r.squared
```

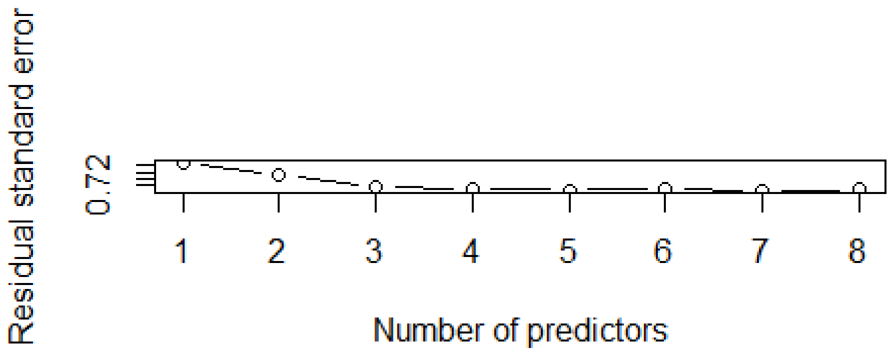
```
## [1] 0.5859345
```

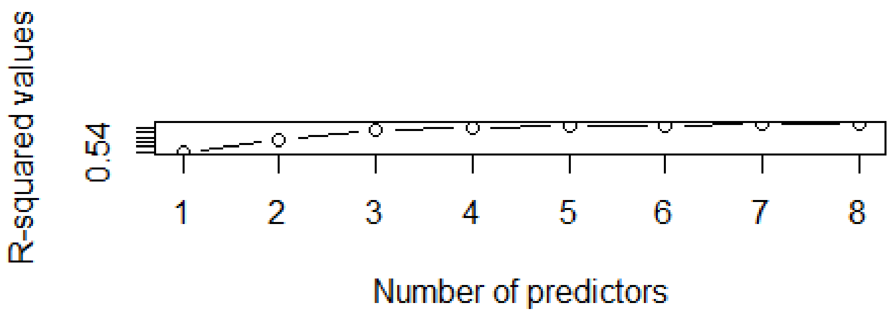
```
# Agregar svi y ajustar el modelo
model5 <- lm(lpsa ~ lcavol + lweight + svi, data = prostate)
summary(model5)$sigma
```

```
## [1] 0.7168094
```

```r
summary(model5)$r.squared

## [1] 0.6264403

# Agregar lbph y ajustar el modelo
model6 <- lm(lpsa ~ lcavol + lweight + svi + lbph, data = prostate)
summary(model6)$sigma

## [1] 0.7108232

summary(model6)$r.squared

## [1] 0.6366035

# Agregar age y ajustar el modelo
model7 <- lm(lpsa ~ lcavol + lweight + svi + lbph + age, data = prostate)
summary(model7)$sigma

## [1] 0.7073054

summary(model7)$r.squared

## [1] 0.6441024

# Agregar lcp y ajustar el modelo
model8 <- lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp, data =
prostate)
summary(model8)$sigma

## [1] 0.7102135

summary(model8)$r.squared

## [1] 0.645113

# Agregar pgg45 y ajustar el modelo
model9 <- lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45,
data = prostate)
summary(model9)$sigma

## [1] 0.7047533

summary(model9)$r.squared

## [1] 0.6544317

# Agregar gleason y ajustar el modelo
model10 <- lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45 +
gleason, data = prostate)
summary(model10)$sigma

## [1] 0.7084155

summary(model10)$r.squared
```

```
## [1] 0.6547541

# Almacena los errores estándar residuales en un vector.
sds <- c(summary(model3)$sigma,
         summary(model4)$sigma,
         summary(model5)$sigma,
         summary(model6)$sigma,
         summary(model7)$sigma,
         summary(model8)$sigma,
         summary(model9)$sigma,
         summary(model10)$sigma)

# Almacena los valores de R-cuadrado en un vector.
rsq <- c(summary(model3)$r.squared,
         summary(model4)$r.squared,
         summary(model5)$r.squared,
         summary(model6)$r.squared,
         summary(model7)$r.squared,
         summary(model8)$r.squared,
         summary(model9)$r.squared,
         summary(model10)$r.squared)

# Crear un gráfico de los errores estándar residuales y los valores de R-
cuadrado.
par(mfrow = c(2, 1))
plot(sds, type = "b", xlab = "Number of predictors", ylab = "Residual
standard error")
plot(rsq, type = "b", xlab = "Number of predictors", ylab = 'R-squared
values')
```






```
cheddar

##    taste Acetic    H2S Lactic
## 1   12.3  4.543  3.135   0.86
## 2   20.9  5.159  5.043   1.53
## 3   39.0  5.366  5.438   1.57
## 4   47.9  5.759  7.496   1.81
## 5    5.6  4.663  3.807   0.99
## 6   25.9  5.697  7.601   1.09
## 7   37.3  5.892  8.726   1.29
## 8   21.9  6.078  7.966   1.78
## 9   18.1  4.898  3.850   1.29
## 10  21.0  5.242  4.174   1.58
## 11  34.9  5.740  6.142   1.68
## 12  57.2  6.446  7.908   1.90
## 13   0.7  4.477  2.996   1.06
## 14  25.9  5.236  4.942   1.30
## 15  54.9  6.151  6.752   1.52
## 16  40.9  6.365  9.588   1.74
## 17  15.9  4.787  3.912   1.16
## 18   6.4  5.412  4.700   1.49
## 19  18.0  5.247  6.174   1.63
## 20  38.9  5.438  9.064   1.99
## 21  14.0  4.564  4.949   1.15
## 22  15.2  5.298  5.220   1.33
## 23  32.0  5.455  9.242   1.44
## 24  56.7  5.855 10.199   2.01
```

```
## 25  16.8  5.366  3.664   1.31
## 26  11.6  6.043  3.219   1.46
## 27  26.5  6.458  6.962   1.72
## 28   0.7  5.328  3.912   1.25
## 29  13.4  5.802  6.685   1.08
## 30   5.5  6.176  4.787   1.25
```

```
model_ch <- lm(taste ~ Acetic + H2S + Lactic, data = cheddar)
summary(model_ch)$coefficients
```

```
##                Estimate Std. Error    t value    Pr(>|t|)
## (Intercept) -28.8767696  19.735418 -1.4631952 0.155399149
## Acetic        0.3277413   4.459757  0.0734886 0.941979774
## H2S           3.9118411   1.248430  3.1334077 0.004247081
## Lactic       19.6705434   8.629055  2.2795710 0.031079481
```

```
cor(fitted(model_ch), cheddar$taste)^2
```

```
## [1] 0.6517747
```

```
#Este valor aparece en la salida de resumen bajo "Multiple R-squared",
que es el coeficiente de determinación (R-cuadrado) para el modelo.

summary(model_ch)
```

```
##
## Call:
## lm(formula = taste ~ Acetic + H2S + Lactic, data = cheddar)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.390  -6.612  -1.009   4.908  25.449
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -28.8768    19.7354  -1.463  0.15540
## Acetic        0.3277     4.4598   0.073  0.94198
## H2S           3.9118     1.2484   3.133  0.00425 **
## Lactic       19.6705     8.6291   2.280  0.03108 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.13 on 26 degrees of freedom
## Multiple R-squared:  0.6518, Adjusted R-squared:  0.6116
## F-statistic: 16.22 on 3 and 26 DF,  p-value: 3.81e-06
```

```
model_ch2 <- lm(taste ~ Acetic + H2S + Lactic, data = cheddar)
summary(model_ch2)
```

```
##
## Call:
## lm(formula = taste ~ Acetic + H2S + Lactic, data = cheddar)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.390  -6.612  -1.009   4.908  25.449
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -28.8768    19.7354  -1.463  0.15540
## Acetic        0.3277     4.4598   0.073  0.94198
## H2S           3.9118     1.2484   3.133  0.00425 **
## Lactic       19.6705     8.6291   2.280  0.03108 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.13 on 26 degrees of freedom
## Multiple R-squared:  0.6518, Adjusted R-squared:  0.6116
## F-statistic: 16.22 on 3 and 26 DF,  p-value: 3.81e-06
```

```r
# Ajustar el modelo de regresión sin intercept
model_no_intercept <- lm(taste ~ 0 + Acetic + H2S + Lactic, data =
cheddar)

# Imprimir sumario
summary(model_no_intercept)
```

```
##
## Call:
## lm(formula = taste ~ 0 + Acetic + H2S + Lactic, data = cheddar)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.4521  -6.5262  -0.6388   4.6811  28.4744
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
## Acetic   -5.454      2.111  -2.583  0.01553 *
## H2S       4.576      1.187   3.854  0.00065 ***
## Lactic   19.127      8.801   2.173  0.03871 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.34 on 27 degrees of freedom
## Multiple R-squared:  0.8877, Adjusted R-squared:  0.8752
## F-statistic: 71.15 on 3 and 27 DF,  p-value: 6.099e-13
```

```r
n <- nrow(cheddar)
p <- 3  # número de predictores
adj_r2 <- 1 - (1 - summary(model)$r.squared) * (n - 1) / (n - p - 1)

adj_r2
```

```
## [1] 0.4721146
```

```r
x <- model.matrix(model_ch)
qrx <- qr(x)
qrx
```

```
## $qr
##    (Intercept)        Acetic          H2S       Lactic
## 1   -5.4772256 -30.113968786 -32.54439635 -7.898159279
## 2    0.1825742   3.074274055   7.07781670  0.986788408
## 3    0.1825742  -0.005013048   9.00496231  0.564800424
## 4    0.1825742  -0.132848105  -0.11596299 -1.174022623
## 5    0.1825742   0.223658823   0.01258146 -0.159684554
## 6    0.1825742  -0.112680742  -0.14352592 -0.338757278
## 7    0.1825742  -0.176110351  -0.21844050 -0.231665577
## 8    0.1825742  -0.236612439  -0.08633452  0.185470921
## 9    0.1825742   0.147218013   0.06808265  0.062656262
## 10   0.1825742   0.035321678   0.12033678  0.252523800
## 11   0.1825742  -0.126667784   0.02952514  0.205817307
## 12   0.1825742  -0.356315495   0.01449655  0.239818159
## 13   0.1825742   0.284160911   0.05493483 -0.048311675
## 14   0.1825742   0.037273358   0.03351150 -0.010345596
## 15   0.1825742  -0.260357882   0.06720418 -0.006052249
## 16   0.1825742  -0.329967812  -0.19284335  0.059397585
## 17   0.1825742   0.183324098   0.03272661 -0.035095909
## 18   0.1825742  -0.019975930   0.10552870  0.135625109
## 19   0.1825742   0.033695277  -0.10048051  0.228851424
## 20   0.1825742  -0.028433211  -0.37242411  0.414889059
## 21   0.1825742   0.255861548  -0.13963052 -0.047461390
## 22   0.1825742   0.017105995   0.01854233 -0.002293439
## 23   0.1825742  -0.033962972  -0.38783058 -0.061721815
## 24   0.1825742  -0.164074989  -0.39150730  0.338311088
## 25   0.1825742  -0.005013048   0.20877761  0.022499942
## 26   0.1825742  -0.225227638   0.43184195  0.073302621
## 27   0.1825742  -0.360218856   0.12262768  0.115897900
## 28   0.1825742   0.007347594   0.17149042 -0.031599889
## 29   0.1825742  -0.146835147  -0.01487225 -0.331437099
## 30   0.1825742  -0.268489883   0.29182957 -0.174974648
## attr(,"assign")
## [1] 0 1 2 3
##
## $rank
## [1] 4
##
## $qraux
## [1] 1.182574 1.062320 1.011775 1.269577
##
## $pivot
## [1] 1 2 3 4
##
```

```
## attr(,"class")
## [1] "qr"
```

```
y <- cheddar$taste
coefficients_qr <- qr.coef(qrx, y)
coefficients_qr
```

```
## (Intercept)      Acetic         H2S      Lactic
## -28.8767696   0.3277413   3.9118411  19.6705434
```

```
#Para extraer la matriz X usando la función model.matrix, podemos usar el
siguiente código:
wafer
```

```
##    x1 x2 x3 x4 resist
## 1   -  -  -  -  193.4
## 2   +  -  -  -  247.6
## 3   -  +  -  -  168.2
## 4   +  +  -  -  205.0
## 5   -  -  +  -  303.4
## 6   +  -  +  -  339.9
## 7   -  +  +  -  226.3
## 8   +  +  +  -  208.3
## 9   -  -  -  +  220.0
## 10  +  -  -  +  256.4
## 11  -  +  -  +  165.7
## 12  +  +  -  +  203.5
## 13  -  -  +  +  285.0
## 14  +  -  +  +  268.0
## 15  -  +  +  +  169.1
## 16  +  +  +  +  208.5
```

```
X <- model.matrix(resist ~ x1 + x2 + x3 + x4, data = wafer)
#Bajos y altos niveles están codificados como - y +

cor(X)
```

```
## Warning in cor(X): the standard deviation is zero
```

```
##             (Intercept) x1+ x2+ x3+ x4+
## (Intercept)           1  NA  NA  NA  NA
## x1+                  NA   1   0   0   0
## x2+                  NA   0   1   0   0
## x3+                  NA   0   0   1   0
## x4+                  NA   0   0   0   1
```

```
#Existen valores faltantes en la matriz de correlación porque una de las
columnas en la matriz X es una combinación lineal de las otras columnas.
Específicamente, la última columna (correspondiente a x4) es igual al
producto de las primeras tres columnas (x1, x2 y x3). Esto significa que
la cuarta columna es perfectamente predecible a partir de las primeras
tres columnas, y por lo tanto, la correlación entre la cuarta columna y
```

```
las otras columnas no está definida.

coefficients_wafer <- coef(lm(resist ~ x1 + x2 + x3 + x4, data = wafer))
coefficients_wafer

## (Intercept)          x1+          x2+          x3+          x4+
##    236.7813      25.7625     -69.8875      43.5875     -14.4875

#Reajustamos:
lm1 <- lm(resist ~ x1 + x2 + x3, data = wafer)
summary(lm1)

##
## Call:
## lm(formula = resist ~ x1 + x2 + x3, data = wafer)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.137 -20.550   3.575  18.462  41.013
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   229.54      13.32  17.231 7.88e-10 ***
## x1+            25.76      13.32   1.934 0.077047 .
## x2+           -69.89      13.32  -5.246 0.000206 ***
## x3+            43.59      13.32   3.272 0.006677 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.64 on 12 degrees of freedom
## Multiple R-squared:  0.7777, Adjusted R-squared:  0.7221
## F-statistic: 13.99 on 3 and 12 DF,  p-value: 0.0003187

#Los coeficientes de regresión y los errores estándar han cambiado en
comparación con el ajuste original. Los coeficientes para x1, x2 y x3 se
interpretan ahora como el cambio esperado en la resistencia asociado con
un aumento de una unidad en el valor codificado para cada factor,
manteniendo constantes los otros factores. El coeficiente para x4 no se
incluye en el modelo y se ha absorbido en la intercepción.

#El cambio en los coeficientes de regresión está relacionado con la
matriz de correlación de X porque los coeficientes de regresión se
determinan mediante la solución de mínimos cuadrados para los
coeficientes, que implica la inversión de la matriz X'X. Cuando hay una
alta correlación entre dos o más variables predictoras, la matriz X'X se
vuelve mal condicionada, lo que significa que está cerca de ser singular
o no invertible. Esto puede conducir a estimaciones inestables para los
coeficientes de regresión y errores estándar inflados. En algunos casos,
eliminar uno o más predictores correlacionados puede mejorar la
estabilidad y la interpretabilidad del modelo.
```

```r
#PREGUNTA 2

# (1)
Xa  <- matrix(c(1, 1, 0, 1, 2, 3, 1, 3, 8), nrow = 3, ncol = 3, byrow =
TRUE)
Xb <- matrix(c(1, 1, exp(1), 1, 2, exp(2), 1, 3, exp(3)), nrow = 3, ncol
= 3, byrow = TRUE)
#Xc no es lineal por tanto no hay matriz de diseño. No es lineal por la
función tangente

Xa

##      [,1] [,2] [,3]
## [1,]    1    1    0
## [2,]    1    2    3
## [3,]    1    3    8

Xb

##      [,1] [,2]      [,3]
## [1,]    1    1  2.718282
## [2,]    1    2  7.389056
## [3,]    1    3 20.085537

#(2)
X_z <- matrix(c(1, 1, 1, 1, 1, 1,
-1, 1, 1, 1, 0, 0,
1, 0, 0, -1, 1, 0,
1, 1, 1, 1, -1, 1),
ncol = 4, byrow = TRUE)
Y_z <- c(9.2, 8.3, 5.4, -1.6, 8.7, 3.5)
X_z

##      [,1] [,2] [,3] [,4]
## [1,]    1    1    1    1
## [2,]    1    1   -1    1
## [3,]    1    1    0    0
## [4,]    1    0    0   -1
## [5,]    1    0    1    1
## [6,]    1    1   -1    1

beta <- solve(t(X_z) %*% X_z) %*% t(X_z) %*% Y_z
beta

##          [,1]
## [1,] 2.562687
## [2,] 1.798507
## [3,] 1.974627
## [4,] 3.643284

#Por lo tanto, las estimaciones de β1, β2, β3 y β4 son 2.56, 1.79, 1.97 y
3.64, respectivamente.
```

```r
n_Z <- length(Y_z)
p_Z <- ncol(X_z)
df_Z <- n_Z - p_Z
epsilon_Z <- Y_z - X_z %*% beta
mse_Z <- sum(epsilon_Z^2) / df_Z
mse_Z
```

```
## [1] 6.889701
```