

Tarea 3

Parte teórica. Esta parte del proyecto será sobre regresión lineal. Supongamos que quieren explicar una variable estadística Y (por ejemplo altura) utilizando la información de p variables X^1, \dots, X^p (peso, ancho de huesos, etc.). Si se toma una muestra de N individuos, cada variable está representada por un vector de tamaño N . La información de las variables explicativas se pueden juntar en una matriz

$$X = [X^1 \mid \dots \mid X^p]$$

del tamaño $n \times p$ donde cada columna es una variable y cada fila uno de los individuos de la muestra. Tienen que contestar lo siguiente:

- Plantear el problema de regresión como un problema de mínimos cuadrados, encontrar el vector $\hat{\beta} = [\hat{\beta}^1, \dots, \hat{\beta}^p]^T$ que resuelva

$$\hat{\beta} = \operatorname{argmin}_{\beta \in \mathbb{R}^p} \|Y - X\beta\|^2$$

y encontrar la solución teórica. ¿Por qué este planteamiento nos da un ajuste lineal a nuestros datos?
¿Podríamos usarlo para ajustar polinomios (ej $y = x^2$)?

Se requiere estimar β del problema

$$Y = \beta^T x$$

La estimación se obtiene:

$$\begin{aligned}\hat{\beta} &= \min_{\beta} \sum_{i=1}^n (y_i - \sum_{j=1}^p \beta_j x_{ij})^2 \\ &= \min_{\beta} \sum_{i=1}^n (y_i - \beta^T x_i)^2\end{aligned}$$

que se puede escribir en términos matriciales como:

$$\hat{\beta} = \min_{\beta} \|Y - X\beta\|^2$$

Se resuelve al igualar el gradiente con respecto a β a cero.

$$\begin{aligned}\nabla \|Y - X\beta\|^2 \\ &= \nabla (Y^T Y + \beta^T X^T X \beta - 2\beta^T X^T Y) \\ &= 2X^T X \beta - 2X^T Y\end{aligned}$$

. La solución β es el vector que resuelve

$$X^T X \beta = X^T Y$$

“sistema de ecuaciones normales”.

Si $X^T X$ es no singular, entonces

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

es solución y es el estimador por el método de mínimos cuadrados para los coeficientes de la regresión.

Se puede utilizar para ajustar polinomios, pues aunque este modelo puede entenderse como una relación no lineal entre Y y X , las regresiones polinomiales todavía se consideran regresiones lineales al serlo en los coeficientes de regresión $\hat{\beta}$.

Es decir, se ajustan los valores de las variables X^1, \dots, X^n y la relación lineal se mantiene para los coeficientes $\hat{\beta}^1, \dots, \hat{\beta}^p$.

- Argumentar la relación entre la solución encontrada y un problema de proyección en subespacios vectoriales de álgebra lineal. ¿Cuál es la relación particular con el teorema de Pitágoras?

La relación radica en que solucionar el problema de mínimos cuadrados es hallar la proyección óptima, la que mejor aproxima la serie de datos dependiente (Y), dadas las variables independientes (X).

La serie de datos Y será ortogonal a la solución (proyección) encontrada \hat{Y} pues las distancias que se tienen son mínimas al resolver el problema de mínimos cuadrados (se minimiza el error de proyección entre Y y \hat{Y}).

Entonces el error de proyección es ortogonal a cualquier punto de \hat{Y} .

La relación particular con el teorema de Pitágoras es que al ser los datos ortogonales con la solución, el ángulo que se forma con el error de proyección es $\pi/2$. Además la norma del error de proyección $\varepsilon = Y - \hat{Y}$ es

$$\|\varepsilon\|^2 = \|Y\|^2 - \|\hat{Y}\|^2$$

pues $Y \cdot \hat{Y} = 0$.

- ¿Qué logramos al agregar una columna de unos en la matriz X ? Es decir, definir mejor

$$X = [1_n \mid X^1 \mid \dots \mid X^p]$$

con $1_n = [1, 1, \dots, 1]^T$

Si la variable Y parte del origen, se empeora el ajuste obtenido antes.

Si Y no pasa por el origen, se logra explicar mejor la variable estadística Y al eliminar desviación hacia el origen.

Si no se agrega, la línea de regresión se ve obligada a pasar por el origen. Esto significa que todos los valores X^i y Y deben ser iguales a cero en ese punto. Si la curva ajustada no pasa naturalmente por el origen, los coeficientes de regresión y predicciones presentarán un sesgo. Pues la curva no estará centrada en la media de los valores, estará recorrida hacia el origen.

Además, garantiza que la media de los residuales del modelo sean cero.

- Plantear el problema de regresión ahora como un problema de estadística $Y_i = \beta_0 + \beta_1 X_i^1 + \dots + \beta_p X_i^p + \varepsilon_i$ donde los errores son no correlacionados con distribución

$$\varepsilon_i \sim N(0, \sigma^2)$$

Tenemos entonces que

$$f(\varepsilon_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{\varepsilon_i^2}{2\sigma^2}\right\}, \quad i = 1, \dots, n$$

Maximizar la probabilidad de obtener la muestra ya disponible equivale maximizar la función de densidad conjunta del vector aleatorio, ε . Para ello, hemos de suponer homoscedasticidad y ausencia de autocorrelación. Por tanto, la expresión de la función de densidad conjunta es la siguiente:

$$f(\varepsilon) = \prod_{i=1}^n f(\varepsilon_i) = \frac{1}{(2\pi\sigma)^{n/2}} \exp\left\{-\frac{\sum \varepsilon_i^2}{2\sigma^2}\right\}$$

Como ε sigue una distribución Normal Multivariada de orden p , la variable Y , al ser una combinación lineal de las perturbaciones aleatorias, también se distribuirá con una distribución Normal Multivariada.

- ¿Cuál es la función de verosimilitud del problema anterior? **Hint:** empiecen por escribir el problema como

$$Y = X\beta + \varepsilon$$

con

$$\varepsilon \sim N(0, \sigma^2 I_n)$$

con I_n la matriz identidad. Y concluyen entonces que

$$Y \sim N(X\beta, \sigma^2 I_n)$$

Escriban entonces la verosimilitud como $L(\beta, \sigma^2) = f(Y | \beta, \sigma^2, X)$.

Para que la función de densidad conjunta sea una función de verosimilitud, el vector aleatorio ε ha de expresarse en función del vector Y , es decir:

$$L(Y; \beta, \sigma^2) = \frac{1}{(2\pi\sigma)^{n/2}} \exp\left\{-\frac{(Y - X\beta)^T(Y - X\beta)}{2\sigma^2}\right\}$$

- Mostrar que la solución de máxima verosimilitud es la misma que la del problema de mínimos cuadrados.

El problema es maximizar la función de verosimilitud. Como la expresión anterior resulta complicada, aplicaremos una transformación monótona; en concreto, una función logarítmica:

$$\ln(L(Y; \beta, \sigma^2)) = -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln(2\sigma^2) - \frac{(Y - X\beta)^T(Y - X\beta)}{2\sigma^2}$$

Derivando la función de verosimilitud con respecto de β y σ^2 , e igualando las derivadas a cero, obtenemos los resultados:

$$\hat{\beta}_{MV} = (X^T X)^{-1} X^T Y$$

cuya varianza es

$$Var[\hat{\beta}_{MV}] = \sigma^2 (X^T X)^{-1}$$

Observamos que el estimador de Máxima Verosimilitud de β coincide con el estimador de Mínimos Cuadrados, con lo que tendrá las mismas propiedades: será lineal, insesgado, óptimo y consistente.

- Investiga el contenido del Teorema de Gauss-Markov sobre mínimos cuadrados.

El Teorema establece en primer lugar, que cuando el rango de la matriz no es máximo y, por tanto, la estimación de los parámetros no es única, la estimación de cualquier función paramétrica estimable utilizando cualquiera de los estimadores sí es única.

Además establece que los estimadores obtenidos bajo el método de Mínimos Cuadrados Ordinarios tienen la mínima varianza entre los estimadores lineales e insesgados. Es decir, son los más eficientes entre los estimadores lineales.

Para que esto suceda, los estimadores mínimo-cuadráticos deben cumplir:

1. $E(\varepsilon_i) = 0, \quad i = 1, \dots, n$

** El modelo realmente ajusta.

2. $var(\varepsilon_i) = \sigma^2, \quad i = 1, \dots, n$

** Propiedad de homocedasticidad, es decir la varianza no crece con x .

3. $E(\varepsilon_i \cdot \varepsilon_j) = 0, \quad \forall i \neq j$

** Las observaciones son incorrelacionadas.

Parte aplicada. Para esta parte pueden usar la base de datos *diamonds* que sugiero, aunque hay puntos adicionales si usan alguna base original interesante. Cargar la base *diamonds* que se encuentra en el paquete *ggplot2*. Los comandos que pueden usar para cargar la base *diamonds* a su ambiente de trabajo en *R* son:

```

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.2.5
library(broom)

## Warning: package 'broom' was built under R version 3.2.5
library(MASS)
library(plyr)

## Warning: package 'plyr' was built under R version 3.2.5
data(diamonds)
head(diamonds)

## # A tibble: 6 × 10
##   carat      cut color clarity depth table price     x     y     z
##   <dbl>    <ord> <ord>   <ord> <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1 0.23     Ideal    E     SI2  61.5    55  326  3.95  3.98  2.43
## 2 0.21     Premium  E     SI1  59.8    61  326  3.89  3.84  2.31
## 3 0.23     Good     E     VS1  56.9    65  327  4.05  4.07  2.31
## 4 0.29     Premium  I     VS2  62.4    58  334  4.20  4.23  2.63
## 5 0.31     Good     J     SI2  63.3    58  335  4.34  4.35  2.75
## 6 0.24 Very Good  J     VVS2 62.8    57  336  3.94  3.96  2.48

```

Posteriormente deben hacer una regresión lineal. Su objetivo es explicar la variable *price* usando las demás variables. Noten que algunas variables no son numéricas, por lo que no pueden incluirse en un análisis crudo de regresión lineal. Para este proyecto *NO* es necesario saber transformar las variables no numéricas para poder usarlas en la regresión; hacerlo es optativo, de hecho, las paqueteterías lo hacen por ustedes pero deben ser cuidadosos. Pueden usar la función *lm* de *R* para su análisis de regresión.

Modelo de regresión lineal múltiple:

Utilizando función *lm*.

```

reg = lm(formula = price ~ carat + cut + color + clarity + depth + table + x + y + z, data = diamonds)
summary(reg)

```

```

##
## Call:
## lm(formula = price ~ carat + cut + color + clarity + depth +
##     table + x + y + z, data = diamonds)
##
## Residuals:
##      Min        1Q        Median       3Q        Max
## -21376.0   -592.4   -183.5    376.4   10694.2
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5753.762    396.630 14.507 < 2e-16 ***
## carat       11256.978    48.628 231.494 < 2e-16 ***
## cut.L        584.457    22.478  26.001 < 2e-16 ***
## cut.Q       -301.908    17.994 -16.778 < 2e-16 ***
## cut.C        148.035    15.483   9.561 < 2e-16 ***
## cut^4       -20.794    12.377  -1.680  0.09294 .
## color.L     -1952.160   17.342 -112.570 < 2e-16 ***
## color.Q      -672.054   15.777  -42.597 < 2e-16 ***

```

```

## color.C      -165.283   14.725  -11.225 < 2e-16 ***
## color^4       38.195   13.527    2.824  0.00475 **
## color^5      -95.793   12.776   -7.498 6.59e-14 ***
## color^6      -48.466   11.614   -4.173 3.01e-05 ***
## clarity.L     4097.431  30.259  135.414 < 2e-16 ***
## clarity.Q    -1925.004  28.227   -68.197 < 2e-16 ***
## clarity.C      982.205  24.152    40.668 < 2e-16 ***
## clarity^4     -364.918  19.285   -18.922 < 2e-16 ***
## clarity^5      233.563  15.752    14.828 < 2e-16 ***
## clarity^6       6.883   13.715    0.502  0.61575
## clarity^7      90.640   12.103    7.489  7.06e-14 ***
## depth          -63.806   4.535   -14.071 < 2e-16 ***
## table          -26.474   2.912   -9.092 < 2e-16 ***
## x              -1008.261  32.898   -30.648 < 2e-16 ***
## y                 9.609   19.333    0.497  0.61918
## z              -50.119   33.486   -1.497  0.13448
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1130 on 53916 degrees of freedom
## Multiple R-squared:  0.9198, Adjusted R-squared:  0.9198
## F-statistic: 2.688e+04 on 23 and 53916 DF,  p-value: < 2.2e-16

```

Asignando valores numéricos a las variables tipo carácter:

```

X <- cbind(diamonds[1:10])

X$cut <- revalue(X$cut, c("Fair" = 1, "Good" = 2, "Very Good" = 3, "Ideal" = 4, "Premium" = 5))
X$cut <- as.numeric(X$cut)

X$color <- revalue(X$color, c("D" = 1, "E" = 2, "F" = 3, "G" = 4, "H" = 5, "I" = 6, "J" = 7))
X$color <- as.numeric(X$color)

X$clarity <- revalue(X$clarity, c("I1" = 1, "SI1" = 2, "SI2" = 3, "VS1" = 4, "VS2" = 5, "VVS1" = 6, "VV"
X$clarity <- as.numeric(X$clarity)

reg1 = lm(formula = price ~ carat + cut + color + clarity + depth + table + x + y + z, data = X)
summary(reg1)

##
## Call:
## lm(formula = price ~ carat + cut + color + clarity + depth +
##     table + x + y + z, data = X)
##
## Residuals:
##      Min      1Q      Median      3Q      Max 
## -23560.7  -629.7   -127.9    494.9   9903.1 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 5362.718   427.816 12.535 <2e-16 ***
## carat        10743.908   51.837 207.263 <2e-16 ***
## cut           120.750    5.715 21.130 <2e-16 ***
## color        -322.696    3.259 -99.003 <2e-16 ***
## clarity       501.856   3.523 142.450 <2e-16 ***

```

```

## depth          -79.793    4.794 -16.644 <2e-16 ***
## table         -26.760    2.948 -9.078 <2e-16 ***
## x             -877.631   35.226 -24.914 <2e-16 ***
## y              43.735    20.751  2.108  0.0351 *
## z             -29.335    36.017 -0.814  0.4154
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1217 on 53930 degrees of freedom
## Multiple R-squared:  0.907, Adjusted R-squared:  0.907
## F-statistic: 5.845e+04 on 9 and 53930 DF, p-value: < 2.2e-16

```

Comprobamos con la solución teórica matricial:

```

X <- cbind(rep(1,dim(diamonds)[1]),X[1:6],X[8:10])
Y <- diamonds[7]

X <- as.matrix(X)
Y <- as.matrix(Y)

TX <- t(X)

Beta = ginv(TX %*% X) %*% TX %*% Y

```

Beta

```

##               price
## [1,]  5362.71764
## [2,] 10743.90786
## [3,]  120.75047
## [4,] -322.69639
## [5,]  501.85580
## [6,] -79.79270
## [7,] -26.75953
## [8,] -877.63064
## [9,]  43.73546
## [10,] -29.33474

```

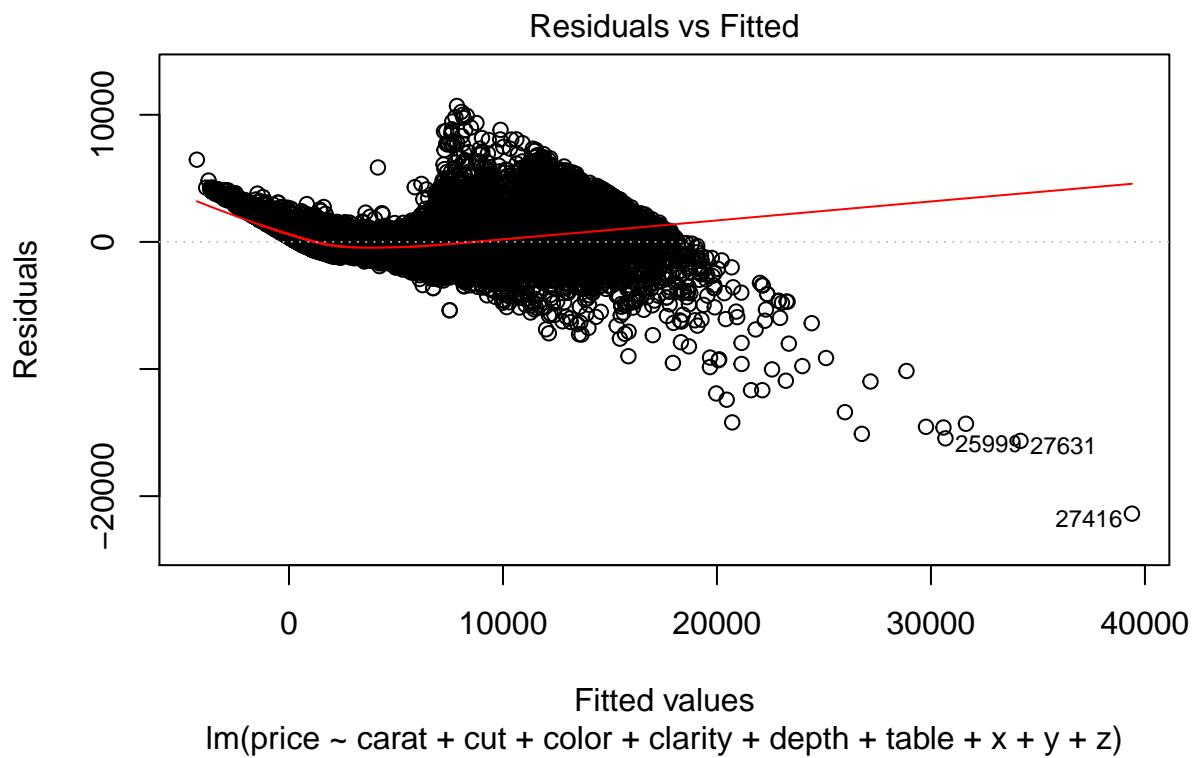
** Son los mismos coeficientes.

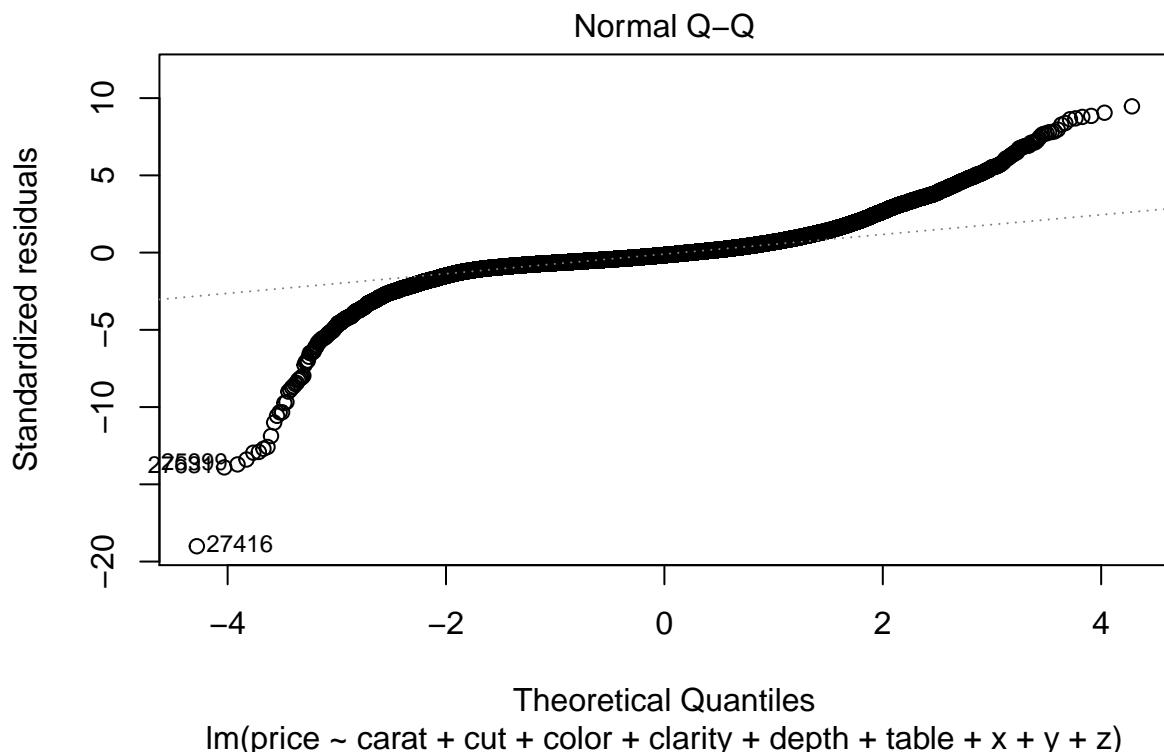
- ¿Qué tan bueno fue el ajuste? Una buena respuesta incluye argumentaciones teóricas y visualizaciones. Puntos adicionales si investigan como usar alguna de las librerías *ggplot2* o *plotly* para sus gráficas.

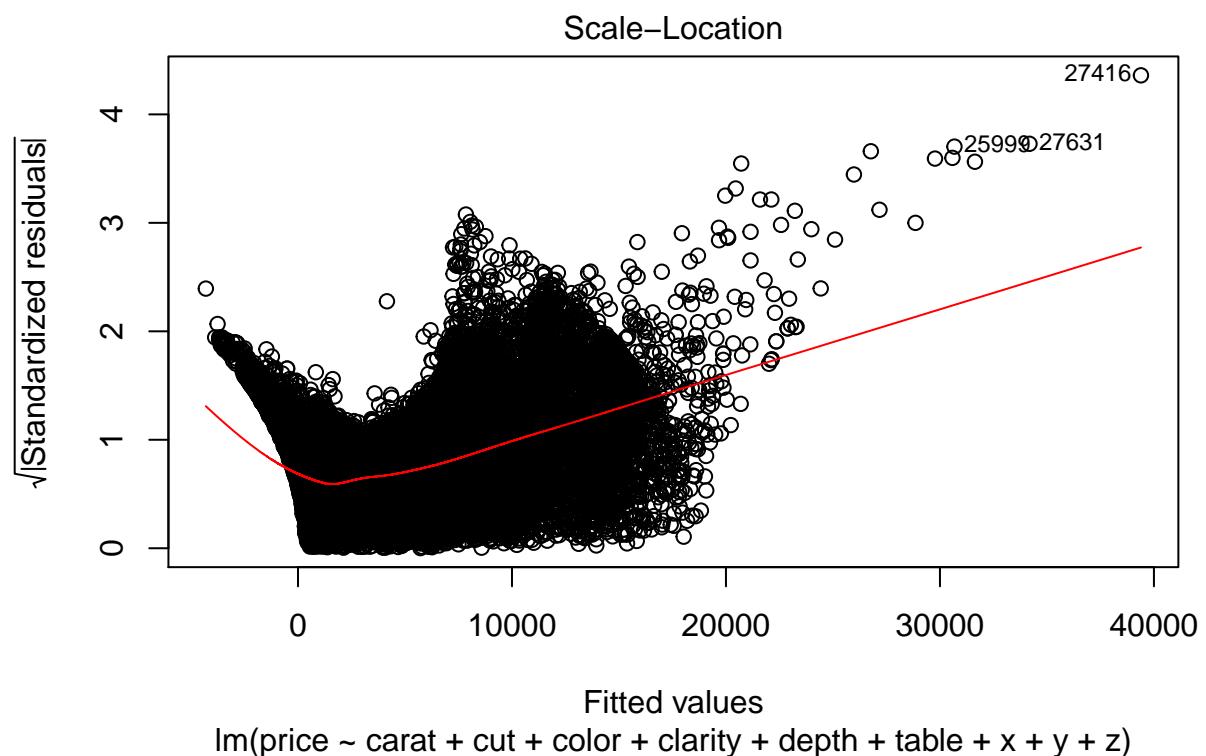
Checamos los dos ajustes:

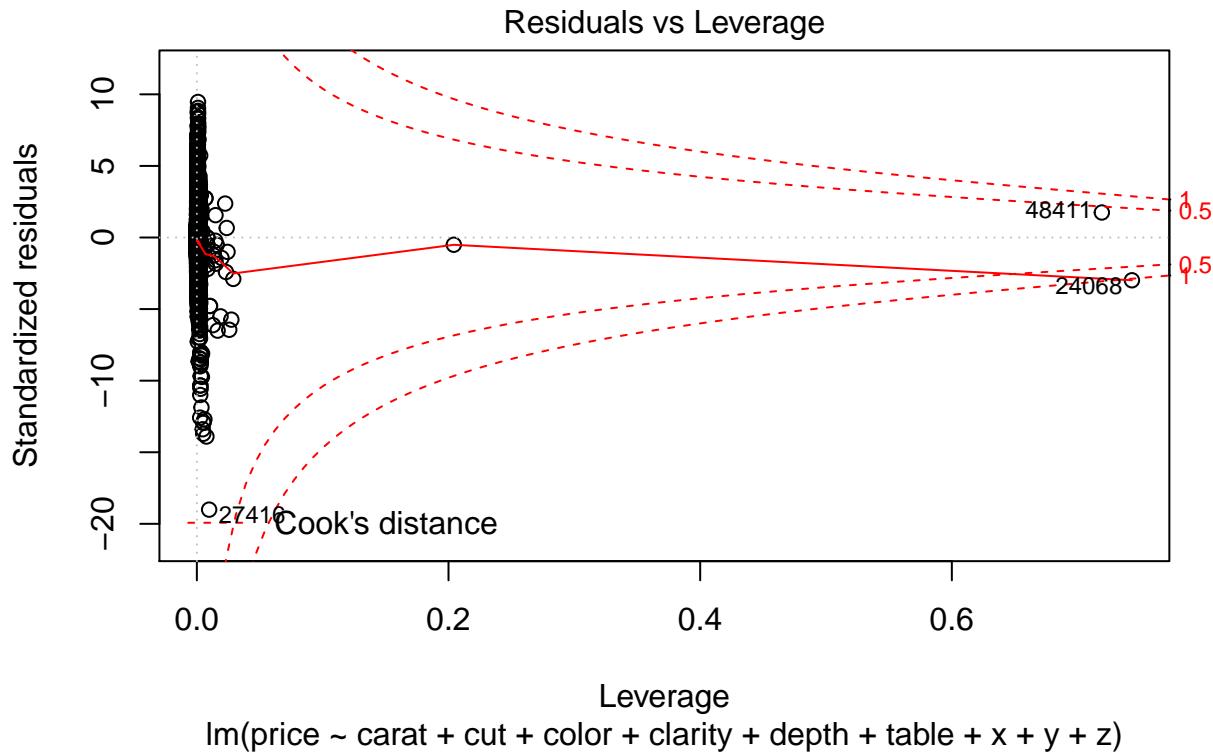
Tomando datos originales:

```
plot(reg)
```





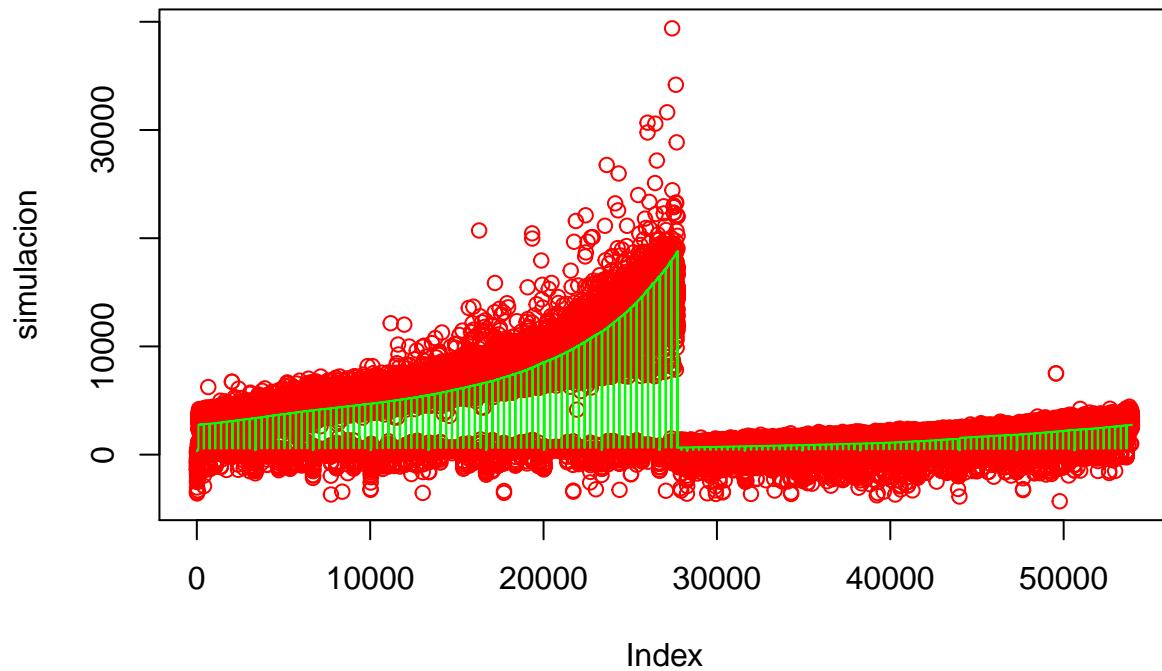




Simulamos datos con el modelo ajustado y comparamos gráficamente con los datos originales:

- Simulación (rojo)
- Datos originales(verde)

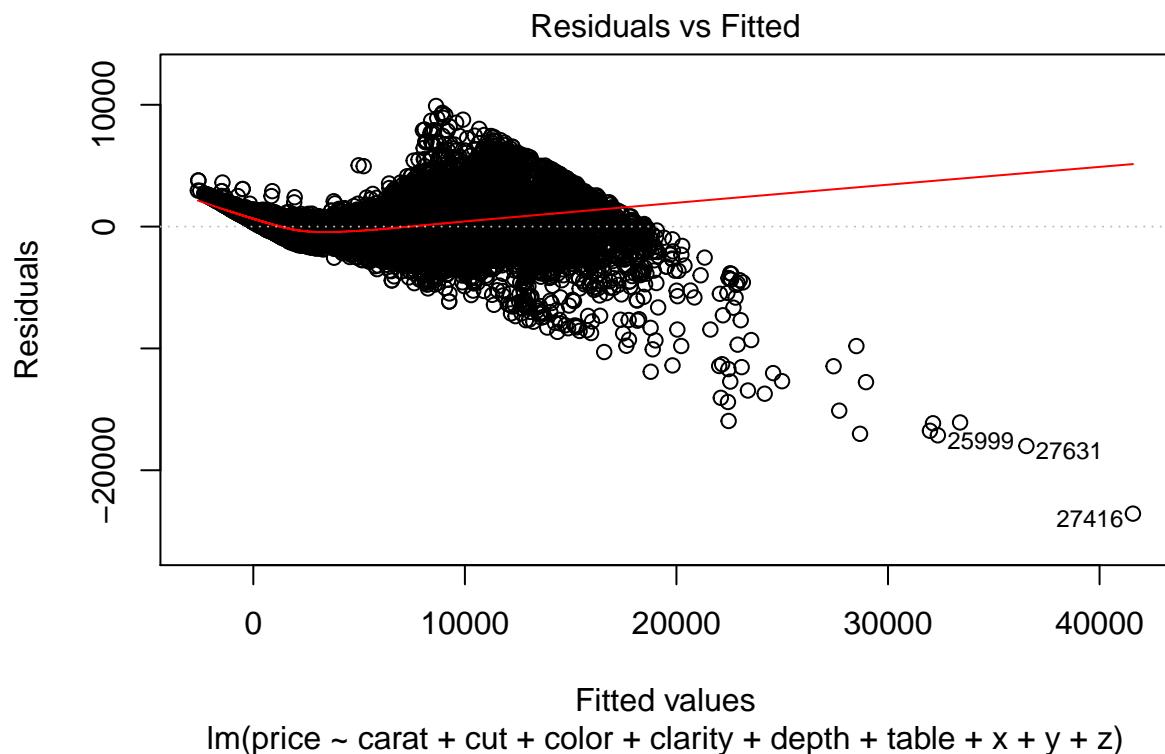
```
simulacion <- fitted(reg)
price <- diamonds[7]
y = c(1:dim(price)[1])
plot(simulacion, col="red")
lines(price,col="green")
```

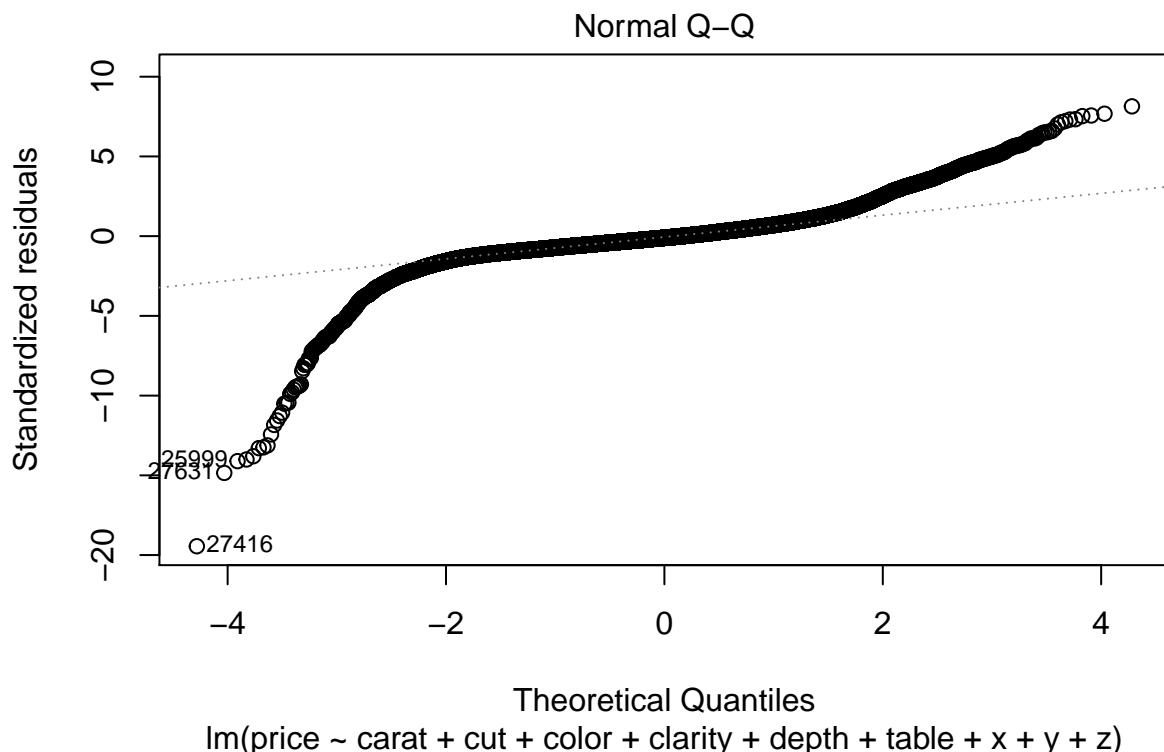


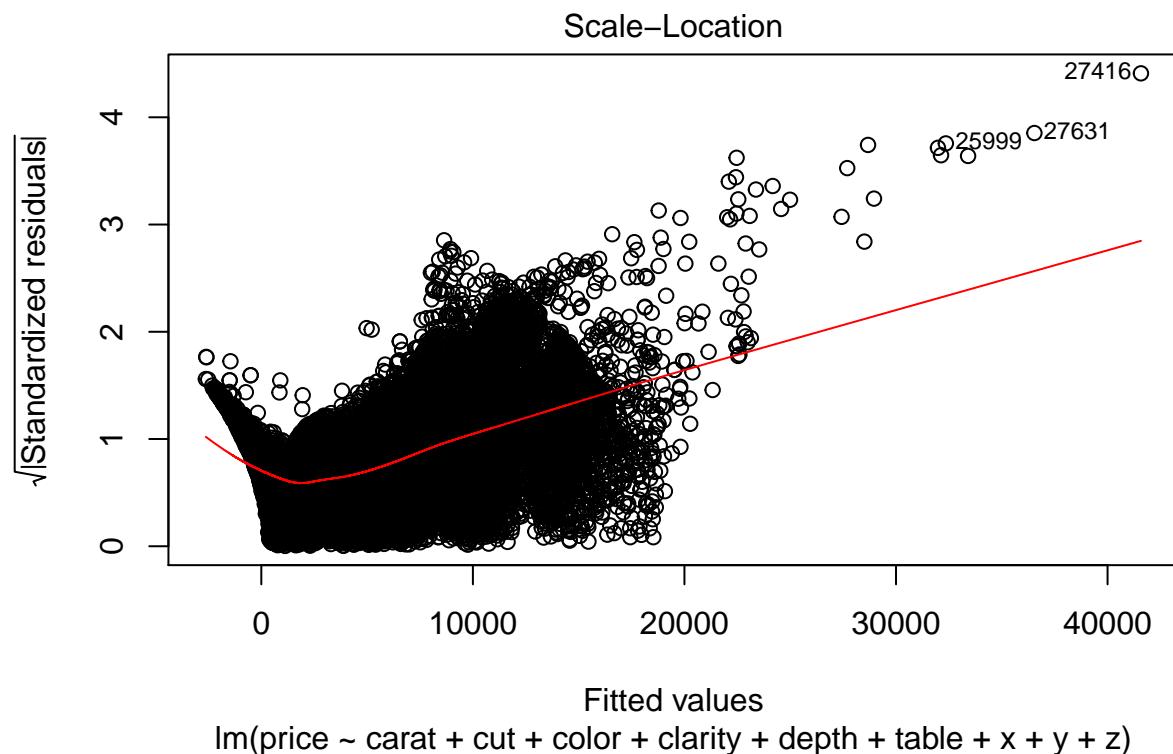
** El ajuste es bueno

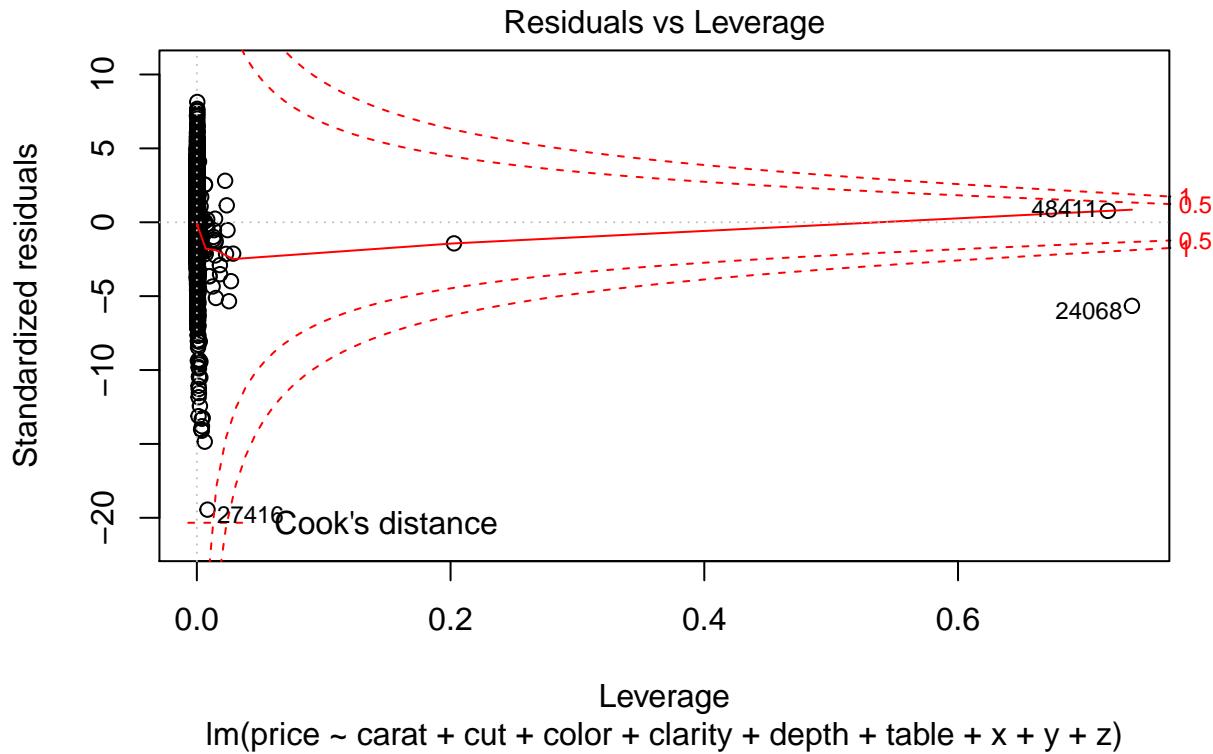
Asignando valores a las variables tipo caracter:

```
plot(reg1)
```





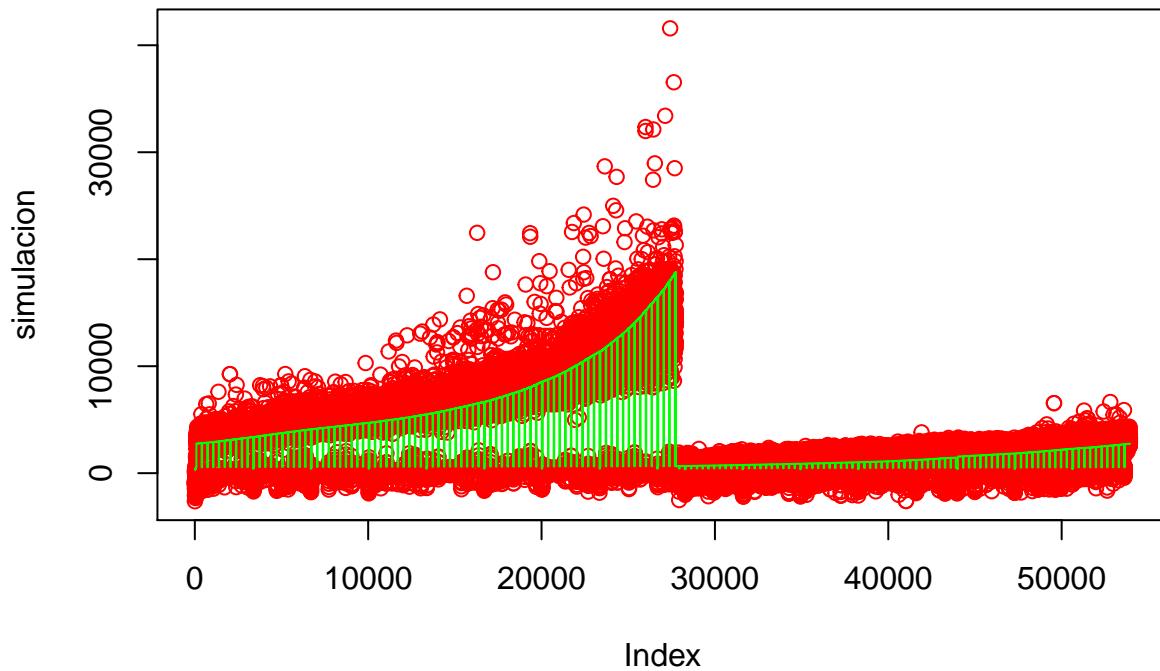




```

simulacion <- fitted(reg1)
price <- diamonds[7]
y = c(1:dim(price)[1])
plot(simulacion, col="red")
lines(price,col="green")

```



** El ajuste es bueno

- ¿Qué medida puede ayudarnos a saber la calidad del ajuste? ¿Cuál fue el valor de σ^2 que ajuste su modelo y qué relación tiene con la calidad del ajuste?

Los valores que nos pueden ayudar a saber la calidad del ajuste son:

– p-value

Se rechaza la hipótesis nula, los coeficientes ajustan bien al modelo

En el summary se puede apreciar que el p-value para ambos ajustes es 2.2e-16, prácticamente 0.

```
glance(reg)$p.value
```

```
## [1] 0
```

```
glance(reg1)$p.value
```

```
## [1] 0
```

- Coeficiente de Correlación

Entre más se acerque el coeficiente de correlación a 1, es mejor el ajuste.

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

```
## [1] 0.9197915
```

```
summary(reg1)$r.squared
```

```
## [1] 0.9070086
```

El primer ajuste es mejor.

- Desviación Estánar

Directamente del modelo:

```
summary(reg)$sigma
```

```
## [1] 1130.094
```

```
summary(reg1)$sigma
```

```
## [1] 1216.661
```

Se puede calcular:

```
sqrt(sum(residuals(reg)^2)/dim(price)[1])
```

```
## [1] 1129.843
```

```
sqrt(sum(residuals(reg1)^2)/dim(price)[1])
```

```
## [1] 1216.548
```

Se tiene una desviación estándar baja en relación al rango en el que están los precios:

```
range(diamonds[7])
```

```
## [1] 326 18823
```

- ¿Cuál es el ángulo entre Y y \hat{Y} ? Hint: usen la R^2 y el arcocoseno.
- Ángulo para el primer ajuste:

```
R <- sqrt(summary(reg)$r.squared)
```

```
anguloY <- acos(R)
```

```
anguloY
```

```
## [1] 0.2871406
```

- Ángulo para el segundo ajuste:

```
R1 <- sqrt(summary(reg1)$r.squared)
```

```
anguloY1 <- acos(R1)
```

```
anguloY1
```

```
## [1] 0.3098806
```

- Defininan una funcion que calcule la logverosimilitud de unos parámetros β y σ^2 .

```
logLik(reg)
```

```
## 'log Lik.' -455726.8 (df=25)
```

```
logLik(reg1)
```

```
## 'log Lik.' -459715 (df=11)
```

- Utilicen la función *optim* de R para numéricamente el máximo de la función de verosimilitud. Si lo hacen correctamente, su solución debe coincidir con la del método *lm*.