Correlación Lineal y Regresión Lineal Simple

Joaquín Amat Rodrigo j.amatrodrigo@gmail.com

Junio, 2016

Índice

Formato PDF: https://github.com/JoaquinAmatRodrigo/Estadistica-con-R

Introducción

La correlación lineal y la regresión lineal simple son métodos estadísticos que estudian la relación lineal existente entre dos variables. Antes de profundizar en cada uno de ellos, conviene destacar algunas diferencias:

- La correlación cuantifica como de relacionadas están dos variables, mientras que la regresión lineal consiste en generar una ecuación (modelo) que, basándose en la relación existente entre ambas variables, permita predecir el valor de una a partir de la otra.

- El cálculo de la correlación entre dos variables es independiente del orden o asignación de cada variable a X e Y, mide únicamente la relación entre ambas sin considerar dependencias. En el caso de la regresión lineal, el modelo varía según qué variable se considere dependiente de la otra (lo cual no implica causa-efecto).

- A nivel experimental, la correlación se suele emplear cuando ninguna de las variables se ha controlado, simplemente se han medido ambas y se desea saber si están relacionadas. En el caso de estudios de regresión lineal, es más común que una de las variables se controle (tiempo, concentración de reactivo, temperatura...) y se mida la otra.

- Por norma general, los estudios de correlación lineal preceden a la generación de modelos de regresión lineal. Primero se analiza si ambas variables están correlacionadas y, en caso de estarlo, se procede a generar el modelo de regresión.

Correlación lineal

Para estudiar la relación lineal existente entre dos variables continuas es necesario disponer de parámetros que permitan cuantificar dicha relación. Uno de estos parámetros es la *covarianza*, que indica el grado de variación conjunta de dos variables aleatorias.

$$Covarianza\ muestral = Cov(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})\,(y_i - \overline{y})}{N-1}$$

siendo $\overline{x}$ e $\overline{y}$ la media de cada variable y x_i e y_i el valor de las variables para la observación i.

La covarianza depende de las escalas en que se miden las variables estudiadas, por lo tanto, no es comparable entre distintos pares de variables. Para poder hacer comparaciones se estandariza la covarianza, generando lo que se conoce como *coeficientes de correlación*. Existen diferentes tipos, de entre los que destacan el *coeficiente de Pearson*, *Rho de Spearman* y *Tau de Kendall*.

- Todos ellos varían entre +1 y -1. Siendo +1 una correlación positiva perfecta y -1 una correlación negativa perfecta.
- Se emplean como medida de fuerza de asociación (tamaño del efecto):
  - 0: asociación nula.
  - 0.1: asociación pequeña.
  - 0.3: asociación mediana.
  - 0.5: asociación moderada.
  - 0.7: asociación alta.
  - 0.9: asociación muy alta.

Las principales diferencias entre estos tres coeficientes de asociación son:

- La correlación de *Pearson* funciona bien con variables cuantitativas que tienen una distribución normal. *En el libro Handbook of Biological Statatistics se menciona que sigue siendo bastante robusto a pesar de la falta de normalidad*. Es más sensible a los valores extremos que las otras dos alternativas.
- La correlación de *Spearman* se emplea cuando los datos son ordinales, de intervalo, o bien cuando no se satisface la condición de normalidad para variables continuas y los datos se pueden transformar a rangos. Es un método no paramétrico.

- La correlación de *Kendall* es otra alternativa no paramétrica para el estudio de la correlación que trabaja con rangos. Se emplea cuando se dispone de pocos datos y muchos de ellos ocupan la misma posición en el rango, es decir, cuando hay muchas ligaduras.

Además del valor obtenido para el coeficiente de correlación, es necesario calcular su significancia. Solo si el *p-value* es significativo se puede aceptar que existe correlación, y esta será de la magnitud que indique el coeficiente. Por muy cercano que sea el valor del coeficiente de correlación a $+1$ o -1, si no es significativo, se ha de interpretar que la correlación de ambas variables es 0, ya que el valor observado puede deberse a simple aleatoriedad.

El test paramétrico de significancia estadística empleado para el coeficiente de correlación es el *t-test*. Al igual que ocurre siempre que se trabaja con muestras, por un lado está el parámetro estimado (en este caso el coeficiente de correlación) y por otro su significancia a la hora de considerar la población entera. Si se calcula el coeficiente de correlación entre X e Y en diferentes muestras de una misma población, el valor va a variar dependiendo de las muestras utilizadas. Por esta razón se tiene que calcular la significancia de la correlación obtenida y su intervalo de confianza.

$$t = \frac{r\sqrt{N-2}}{\sqrt{1-r^2}}, \quad df = N - 2$$

Para este test de hipótesis, H_0 considera que las variables son independientes (coeficiente de correlación poblacional = 0) mientras que, la H_a, considera que existe relación (coeficiente de correlación poblacional ≠ 0)

La correlación lineal entre dos variables, además del valor del coeficiente de correlación y de sus significancia, también tiene un tamaño de efecto asociado. Se conoce como *coeficiente de determinación* R^2. Se interpreta como la cantidad de varianza de Y explicada por X. En el caso del coeficiente de *Pearson* y el de *Spearman*, R^2 se obtiene elevando al cuadrado el coeficiente de correlación. En el caso de Kendall no se puede calcular de este modo. *(No he encontrado como se calcula).*

Mediante *bootstrapping* también se puede calcular la significancia de un coeficiente de correlación. Es una alternativa no paramétrica al *t-test*. Resampling: Test de permutación, Simulación de Monte Carlo y Bootstrapping).

Coeficiente de Pearson

El coeficiente de correlación de Pearson es la covarianza estandarizada, y su ecuación difiere dependiendo de si se aplica a una muestra, *Coeficiente de Pearson muestral (r)*, o si se aplica la población *Coeficiente de Pearson poblacional (ρ)*.

$$\rho = \frac{Cov(X,Y)}{\sigma_x \sigma_y}$$

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})\,(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

Condiciones

- La relación que se quiere estudiar entre ambas variables es lineal (de lo contrario, el coeficiente de Pearson no la puede detectar).
- Las dos variables deben de ser cuantitativas.
- Normalidad: ambas variables se tienen que distribuir de forma normal. *Varios textos defienden su robustez cuando las variables se alejan moderadamente de la normal.*
- Homocedasticidad: La varianza de Y debe ser constante a lo largo de la variable X. Esto se puede identificar si en el *scatterplot* los puntos mantienen la misma dispersión en las distintas zonas de la variable X. *Esta condición no la he encontrado mencionada en todos los libros.*

Características

- Toma valores entre [-1, +1], siendo +1 una correlación lineal positiva perfecta y -1 una correlación lineal negativa perfecta.
- Es una medida independiente de las escalas en las que se midan las variables.
- No varía si se aplican transformaciones a las variables.
- No tiene en consideración que las variables sean dependientes o independientes.
- El coeficiente de correlación de Pearson no equivale a la pendiente de la recta de regresión.
- Es sensible a *outliers*, por lo que se recomienda en caso de poder justificarlos, excluirlos del análisis.

Interpretación

Además del valor obtenido para el coeficiente, es necesario calcular su significancia. Solo si el *p-value* es significativo se puede aceptar que existe correlación y esta será de la magnitud que indique el coeficiente. Por muy cercano que sea el valor del coeficiente de correlación a +1 o -1, si no es significativo, se ha de interpretar que la correlación de ambas variables es 0 ya que el valor observado se puede deber al azar. (Ver más adelante como calcular la significancia).

Coeficiente de Spearman (Spearman's rho)

El coeficiente de *Spearman* es el equivalente al coeficiente de *Pearson* pero con una previa transformación de los datos a rangos. Se emplea como alternativa cuando los valores son ordinales, o bien, cuando los valores son continuos pero no satisfacen la condición de normalidad requerida por el coeficiente de Pearson y se pueden ordenar transformándolos en rangos. Al trabajar con rangos, es menos sensible que *Pearson* a valores extremos. Existe una diferencia adicional con respecto a *Pearson*. El coeficiente de *Spearman* requiere que la relación entre las variables sea monótona, es decir, que cuando una variable crece la otra también lo hace o cuando una crece la otra decrece (que la tendencia sea constante). Este concepto no es exactamente el mismo que linealidad.

$$r_s = 1 - \frac{6\sum d_i^2}{n(n^2-1)},$$

siendo d_i la distancia entre los rangos de cada observación $(x_i - y_i)$ y n el número de observaciones.

Coeficiente Tau de Kendall

Trabaja con rangos, por lo que requiere que las variables cuya relación se quiere estudiar sean ordinales o que se puedan transformar en rangos. Al ser no paramétrico, es otra alternativa al *Coeficiente de correlación de Pearson* cuando no se cumple la condición de normalidad. Parece ser más aconsejable que el coeficiente de *Spearman* cuando el número de observaciones es pequeño o los valores se acumulan en una región por lo que el número de ligaduras al generar los rangos es alto.

$$\tau = \frac{C - D}{\frac{1}{2}n(n-1)},$$

siendo C el número de pares concordantes, aquellos en los que el rango de la segunda variable es mayor que el rango de la primera variable. D el número de pares discordantes, cuando el rango de la segunda es igual o menor que el rango de la primera variable.

Tau represents a probability; that is, it is the difference between the probability that the two variables are in the same order in the observed data versus the probability that the two variables are in different orders.

Jackknife correlation

El coeficiente de correlación de *Pearson* resulta efectivo en ámbitos muy diversos, sin embargo, tiene la desventaja de no ser robusto frente a *outliers* a pesar de que se cumpla la condición de normalidad. Si dos variables tienen un pico o un valle común en una única observación, por ejemplo por un error de lectura, la correlación va a estar dominada por este registro a pesar de que entre las dos variables no haya correlación real alguna. Lo mismo puede ocurrir en la dirección opuesta. Si dos variables están altamente correlacionadas excepto para una observación en la que los valores son muy dispares, entonces la correlación existente quedará enmascarada. Una forma de evitarlo es recurrir a la *Jackknife correlation*, que consiste en calcular todos los posibles coeficientes de correlación entre dos variables si se excluye cada vez una de las observaciones. El promedio de todas las *Jackknife correlations* calculadas atenuará en cierta medida el efecto del *outlier*.

$$\overline{\theta}_{(A,B)} = Promedio\ Jackknife\ correlation(A,B) = \frac{1}{n}\sum_{i=1}^{n} \hat{r}_i$$

donde n es el número de observaciones y $\hat{r}_i$ es el coeficiente de correlación de *Pearson* estimado entre las variables A y B, habiendo excluido la observación i.

Además del promedio, se puede estimar su error estándar (SE) y así obtener intervalos de confianza para la *Jackknife correlation* y su correspondiente *p-value*.

$$SE = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{r}_i - \overline{\theta})^2}$$

Intervalo de confianza del 95% ($Z = 1.96$)

$$Promedio\ Jackknife\ correlation(A,B)\ \pm\ 1.96 * SE$$

$$\overline{\theta} - 1.96\sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{r}_i - \overline{\theta})^2}\ ,\ \ \overline{\theta} + 1.96\sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{r}_i - \overline{\theta})^2}$$

P-value para la hipótesis nula de que $\overline{\theta} = 0$:

$$Z_{calculada} = \frac{\overline{\theta} - H_0}{SE} = \frac{\overline{\theta} - 0}{\sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{r}_i - \overline{\theta})^2}}$$

$$p_{value} = P(Z > Z_{calculada})$$

Cuando se emplea este método es conveniente calcular la diferencia entre el valor de correlación obtenido por *Jackknife correlation* ($\overline{\theta}$) y el que se obtiene si se emplean todas las observaciones ($\overline{r}$). A esta diferencia se le conoce como *Bias*. Su magnitud es un indicativo de cuanto está influenciada la estimación de la correlación entre dos variables debido a un valor atípico u *outlier*.

$$Bias = (n-1) * (\overline{\theta} - \hat{r})$$

Si se calcula la diferencia entre cada correlación ($\hat{r}_i$) estimada en el proceso de *Jackknife* y el valor de correlación ($\hat{r}$) obtenido si se emplean todas las observaciones, se puede identificar que observaciones son más influyentes.

Cuando el estudio requiere minimizar al máximo la presencia de falsos positivos, a pesar de que se incremente la de falsos negativos se puede seleccionar como valor de correlación el menor de entre todos los calculados en el proceso de *Jackknife*.

$$Correlacion = min\{\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_n\}$$

A pesar de que el método de *Jackknife* permite aumentar la robustez de la correlación de *Pearson*, si los *outliers* son muy extremos su influencia seguirá siendo notable. Siempre es conveniente una representación gráfica de los datos para poder identificar si hay valores atípicos y eliminarlos. Otras alternativas robustas son la correlación de *Spearman* o el método de *Bootstrapping*.

Ejemplo correlación lineal

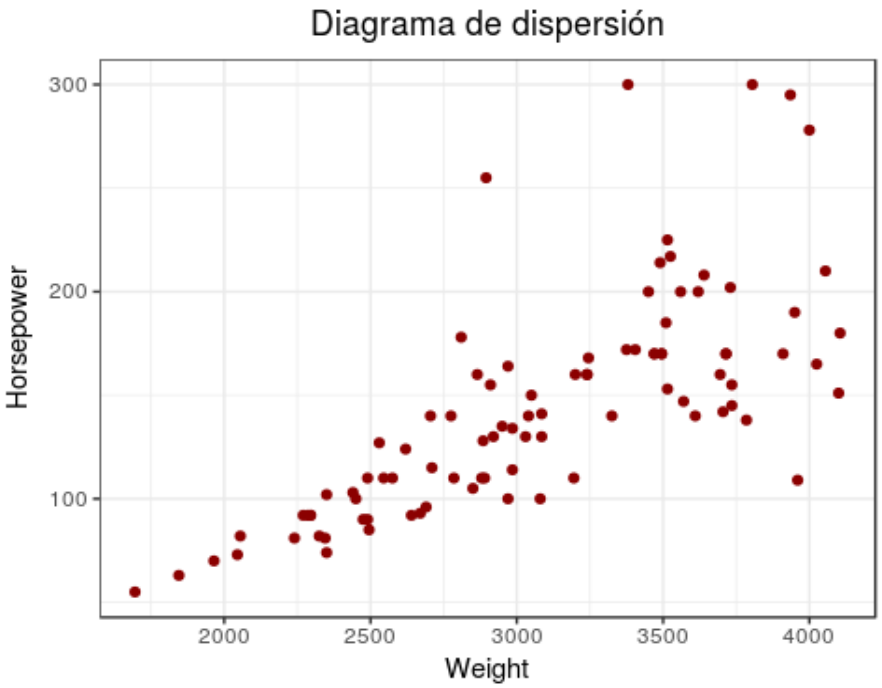
Se dispone de un data set con información sobre diferentes coches. Se quiere estudiar si existe una correlación entre el peso de un vehículo (Weight) y la potencia de su motor (Horsepower).

R contiene funciones que permiten calcular los diferentes tipos de correlaciones y sus niveles de significancia: `cor()` y `cor.test()`. La segunda función es más completa ya que además de calcular el coeficiente de correlación indica su significancia (*p-value*) e intervalo de confianza.

```
require(MASS)
require(ggplot2)
data("Cars93")
```

En primer lugar se representan las dos variables mediante un diagrama de dispersión (+) para intuir si existe relación lineal o monotónica. Si no la hay, no tiene sentido calcular este tipo de correlaciones.

```
ggplot(data = Cars93, aes(x = Weight, y = Horsepower)) + geom_point(colou
r = "red4") +
ggtitle("Diagrama de dispersión") +
theme_bw() +
theme(plot.title = element_text(hjust = 0.5))
```

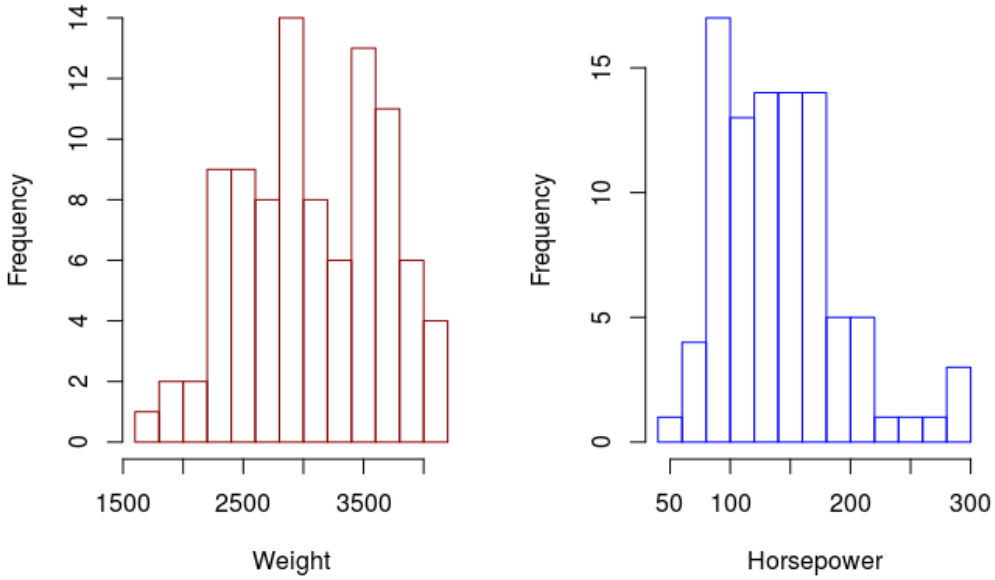


El diagrama de dispersión parece indicar una posible relación lineal positiva entre ambas variables.

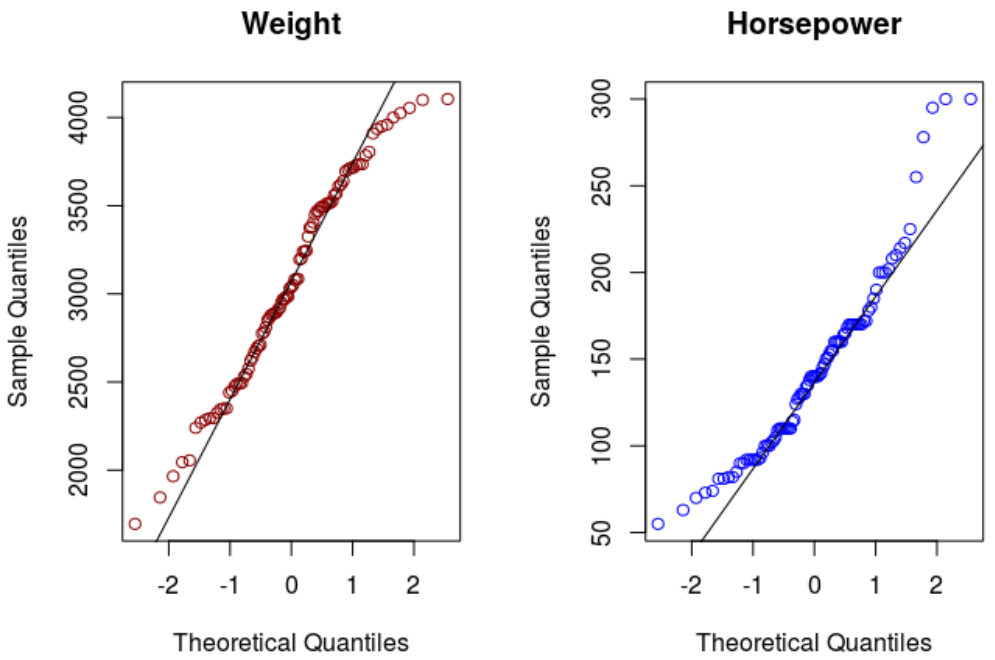
Para poder elegir el coeficiente de correlación adecuado, se tiene que analizar el tipo de variables y la distribución que presentan. En este caso, ambas variables son cuantitativas continuas y pueden transformarse en rangos para ordenarlas, por lo que *a priori* los tres coeficientes podrían aplicarse. La elección se hará en función de la distribución que presenten las observaciones.

1.Análisis de normalidad

```
# Representación gráfica
par(mfrow = c(1, 2))
hist(Cars93$Weight, breaks = 10, main = "", xlab = "Weight",
      border = "darkred")
hist(Cars93$Horsepower, breaks = 10, main = "", xlab = "Horsepower",
     border = "blue")
```




```
qqnorm(Cars93$Weight, main = "Weight", col = "darkred")
qqline(Cars93$Weight)
qqnorm(Cars93$Horsepower, main = "Horsepower", col = "blue")
qqline(Cars93$Horsepower)
```



```
par(mfrow = c(1, 1))

# Test de hipótesis para el análisis de normalidad
shapiro.test(Cars93$Weight)
```

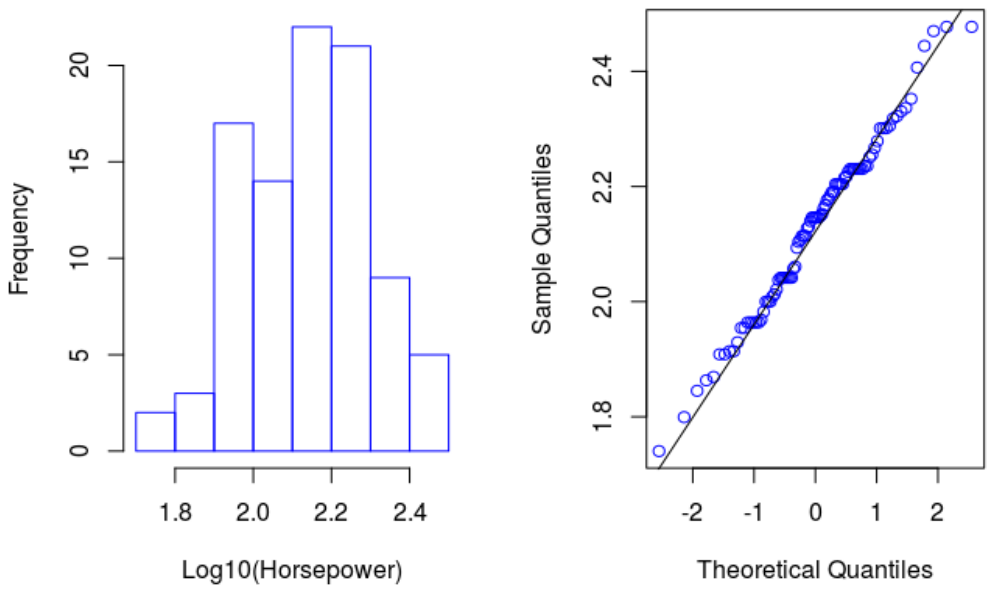
```
##
##  Shapiro-Wilk normality test
##
## data:  Cars93$Weight
## W = 0.97432, p-value = 0.06337
```

```
shapiro.test(Cars93$Horsepower)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Cars93$Horsepower
## W = 0.93581, p-value = 0.0001916
```

El análisis gráfico y el contraste de normalidad muestran que para la variable *Horsepower* no se puede asumir normalidad y que la variable *Weight* está en el límite. Siendo estrictos, este hecho excluye la posibilidad de utilizar el *coeficiente de Pearson*, dejando como alternativas el de *Spearman* o *Kendall*. Sin embargo, dado que la distribución no se aleja mucho de la normalidad y de que el *coeficiente de Pearson* tiene cierta robustez, a fines prácticos sí que se podría utilizar siempre y cuando se tenga en cuenta este hecho en los resultados. Otra posibilidad es tratar de transformar las variables para mejorar su distribución.

```
# Representación gráfica
par(mfrow = c(1, 2))
hist(log10(Cars93$Horsepower), breaks = 10, main = "",
     xlab = "Log10(Horsepower)", border = "blue")
qqnorm(log10(Cars93$Horsepower), main = "", col = "blue")
qqline(log10(Cars93$Horsepower))
```




```
par(mfrow = c(1, 1))
shapiro.test(log10(Cars93$Horsepower))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  log10(Cars93$Horsepower)
## W = 0.98761, p-value = 0.5333
```

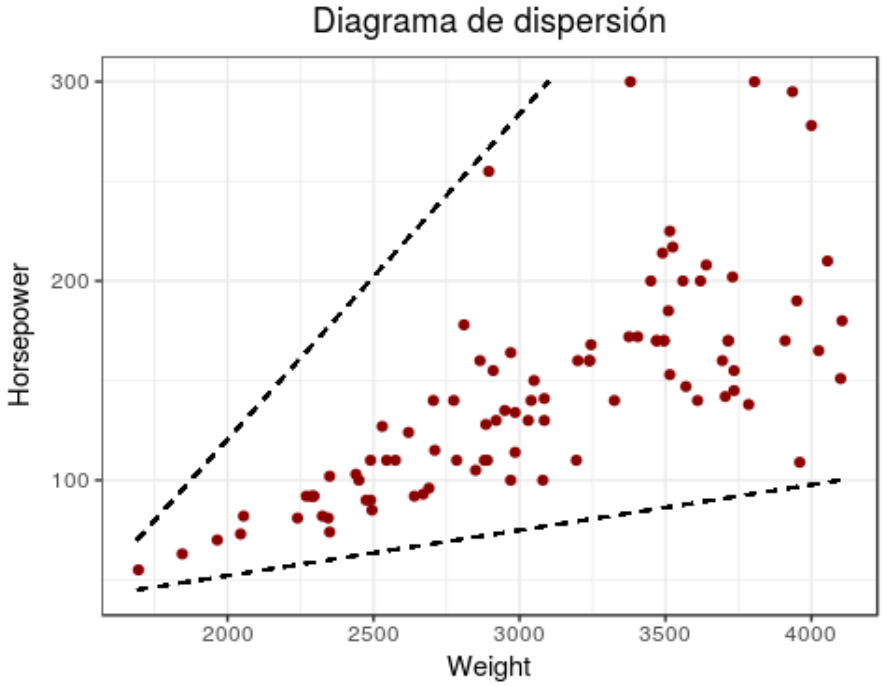
La trasformación logarítmica de la variable *Horsepower* consigue una distribución de tipo normal.

2.Homocedasticidad

La homocedasticidad implica que la varianza se mantenga constante. Puede analizarse de forma gráfica representando las observaciones en un diagrama de dispersión y viendo si mantiene una homogeneidad en su dispersión a lo largo del eje X. Una forma cónica es un claro indicativo de falta de homocedasticidad. *En algunos libros se menciona el test de Goldfeld-Quandt o el de Breusch-Pagan como test de hipótesis para la homocedasticidad en correlación y regresión.*

Tal como muestra el diagrama de dispersión generado al inicio del ejercicio, sí hay un patrón cónico. Esto debe de tenerse en cuenta si se utiliza *Pearson* puesto que viola una de sus condiciones.

```
ggplot(data = Cars93, aes(x = Weight, y = Horsepower)) + geom_point(colou
r = "red4") +
geom_segment(aes(x = 1690, y = 70, xend = 3100, yend = 300),
             linetype = "dashed") +
geom_segment(aes(x = 1690, y = 45, xend = 4100, yend = 100),
              linetype = "dashed") +
ggtitle("Diagrama de dispersión") + theme_bw() +
theme(plot.title = element_text(hjust = 0.5))
```



3.Cálculo de correlación

Debido a la falta de homocedasticidad, los resultados generados por *Pearson* no son precisos, desde el punto de vista teórico *Spearman* o *Kendall* son más adecuados. Sin embargo, en la bibliografía emplean *Pearson*, así que se van a calcular tanto *Pearson* como *Spearman*.

```
cor(x = Cars93$Weight, y = log10(Cars93$Horsepower), method = "pearson")
```

```
## [1] 0.809672
```

```
cor(x = Cars93$Weight, y = log10(Cars93$Horsepower), method = "spearman")
# La función cor() también acepta matrices o data frames y calcula todas
las correlaciones dos a dos.
```

```
## [1] 0.8042527
```

Ambos test muestran una correlación alta (>0.8). Sin embargo para poder considerar que existe realmente correlación entre las dos variables es necesario calcular su significancia, de lo contrario podría deberse al azar.

4.Significancia de la correlación

Por muy alto que sea un coeficiente de correlación, si no es significativa se ha de considerar inexistente.

```
cor.test(x = Cars93$Weight, y = log10(Cars93$Horsepower),
         alternative = "two.sided",conf.level = 0.95, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
## data:  Cars93$Weight and log10(Cars93$Horsepower)
## t = 13.161, df = 91, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7256502 0.8699014
## sample estimates:
##      cor
## 0.809672
```

```
cor.test(x = Cars93$Weight, y = log10(Cars93$Horsepower),
         alternative = "two.sided",conf.level = 0.95,method = "spearman")
```

```
## Warning in cor.test.default(x = Cars93$Weight, y =
## log10(Cars93$Horsepower), : Cannot compute exact p-value with ties

##
##  Spearman's rank correlation rho
##
## data:  Cars93$Weight and log10(Cars93$Horsepower)
## S = 26239, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.8042527
```

Ambos coeficientes de correlación son significativos (p_value ≈ 0).

5.Coeficiente de determinación R^2 (tamaño del efecto)

```
R2_pearson <- cor(x = Cars93$Weight, y = log10(Cars93$Horsepower),
                  method = "pearson")^2
R2_pearson
```

```
## [1] 0.6555688
```

```
R2_spearman <- cor(x = Cars93$Weight, y = log10(Cars93$Horsepower),
                   method = "spearman")^2
R2_spearman
```

```
## [1] 0.6468225
```

6.Conclusión

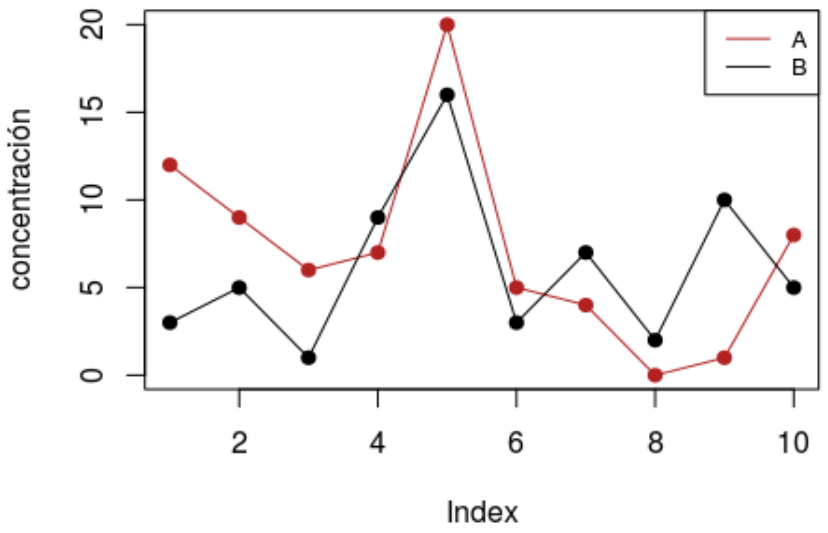
Existe una correlación significativa entre el peso del vehículo y la potencia de su motor (r=0.8, *p-value* < 2.2e-16), con un tamaño de efecto medio-alto (R^2 = 0.66).

Ejemplo Jackknife correlation

Un equipo de investigadores quiere estudiar si existe correlación en la presencia de dos sustancias (A y B) en el agua de los ríos. Para ello han realizado una serie de mediciones en las que se cuantifica la concentración de las dos sustancias en 10 muestras independientes de agua. Se sospecha que el instrumento de lectura sufre alguna avería que provoca que algunas lecturas se disparen, por esta razón se quiere emplear un método de correlación robusto. El objetivo de este ejemplo es ilustrar el método de Jackknife, por lo que se asume que se cumplen las condiciones para la correlación de Pearson.

```
# Datos simulados de dos variables A y B
a <- c(12, 9, 6, 7, 2, 5, 4, 0, 1, 8)
b <- c(3, 5, 1, 9, 5, 3, 7, 2, 10, 5)

# Se introduce un outlier
a[5] <- 20
b[5] <- 16
datos <- data.frame(a, b)
```

```
plot(datos$a, type = "o", lty = 1, pch = 19, col = "firebrick",
     ylab = "concentración", main = "Con outlier")
lines(datos$b, type = "o", pch = 19, lty = 1)
legend("topright", legend = c("A", "B"), col = c("firebrick", "black"),
       lty = c(1, 1), cex = 0.8)
```

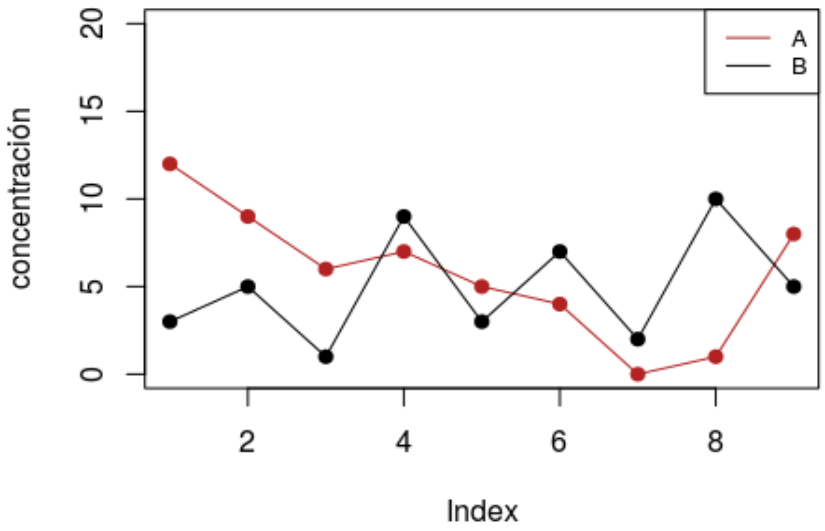



```
cor(datos$a, datos$b, method = "pearson")
```

```
## [1] 0.5249277
```

```
# Se elimina el outlier
a <- a[-5]
b <- b[-5]
datos_sin_outlier <- data.frame(a, b)
plot(datos_sin_outlier$a, type = "o", pch = 19, col = "firebrick",
     ylim = c(0, 20), ylab = "concentración", main = "Sin outlier")
lines(datos_sin_outlier$b, type = "o", pch = 19, lty = 1)
legend("topright", legend = c("A", "B"), col = c("firebrick", "black"),
       lty = c(1, 1), cex = 0.8)
```

Sin outlier




```
cor(datos_sin_outlier$a, datos_sin_outlier$b, method = "pearson")
```

```
## [1] -0.1790631
```

La observación numero 5 tiene una gran influencia en el resultado de la correlación, siendo de 0.52 en su presencia y de -0.18 si se excluye.

```
# FUNCIÓN PARA APLICAR JACKKNIFE A LA CORRELACIÓN DE PEARSON
correlacion_jackknife <- function(matriz, method = "pearson") {
    n <- nrow(matriz)  #   número de observaciones
    valores_jackknife <- rep(NA, n)
    for (i in 1:n) {
        # Loop para excluir cada observación y calcular la correlación
        valores_jackknife[i] <- cor(matriz[-i, 1], matriz[-i, 2],
                                    method = method)
    }

    promedio_jackknife <- mean(valores_jackknife)
    standar_error <- sqrt(((n - 1)/n) * sum((valores_jackknife -
                                             promedio_jackknife)^2))
    bias <- (n - 1) * (promedio_jackknife - cor(matriz[, 1], matriz[, 2],
                                                method = method))
    return(list(valores_jackknife = valores_jackknife,
                promedio=promedio_jackknife,se=standar_error,bias=bias))
}
```

```
correlacion <- correlacion_jackknife(datos)
correlacion$promedio
```

```
## [1] 0.4854695
```

```
correlacion$valores_jackknife
```

```
##  [1]  0.6409823  0.5394608  0.5410177  0.5414076 -0.1790631  0.5121559
##  [7]  0.5504217  0.4528914  0.7237978  0.5316224
```

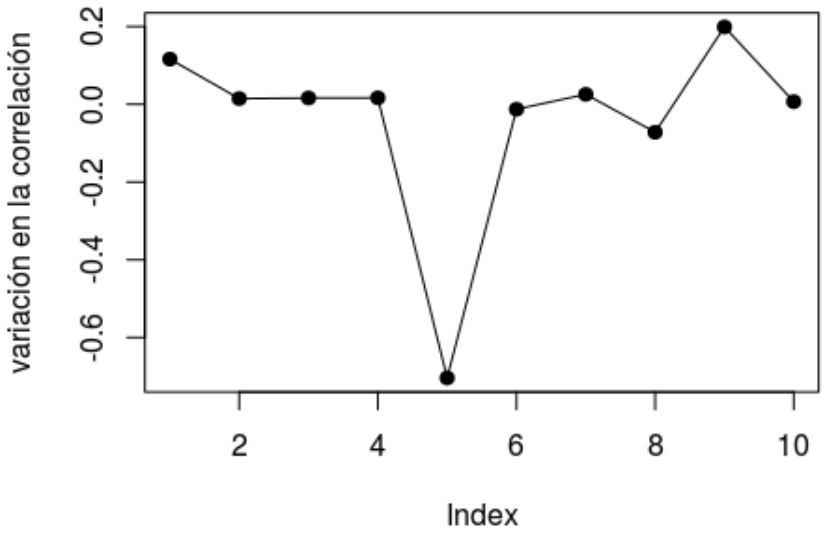
```
correlacion$se
```

```
## [1] 0.697034
```

```
correlacion$bias
```

```
## [1] -0.3551246
```

```
plot((correlacion$valores_jackknife - cor(datos$a, datos$b,
                                          method = "pearson")),
    type = "o", pch = 19, ylab = "variación en la correlación")
```



El método *Jackknife correlation* solo ha sido capaz de amortiguar una pequeña parte de la influencia del *outlier*, sin embargo, si ha permitido identificar qué observación está afectando en mayor medida.

Matriz de correlaciones

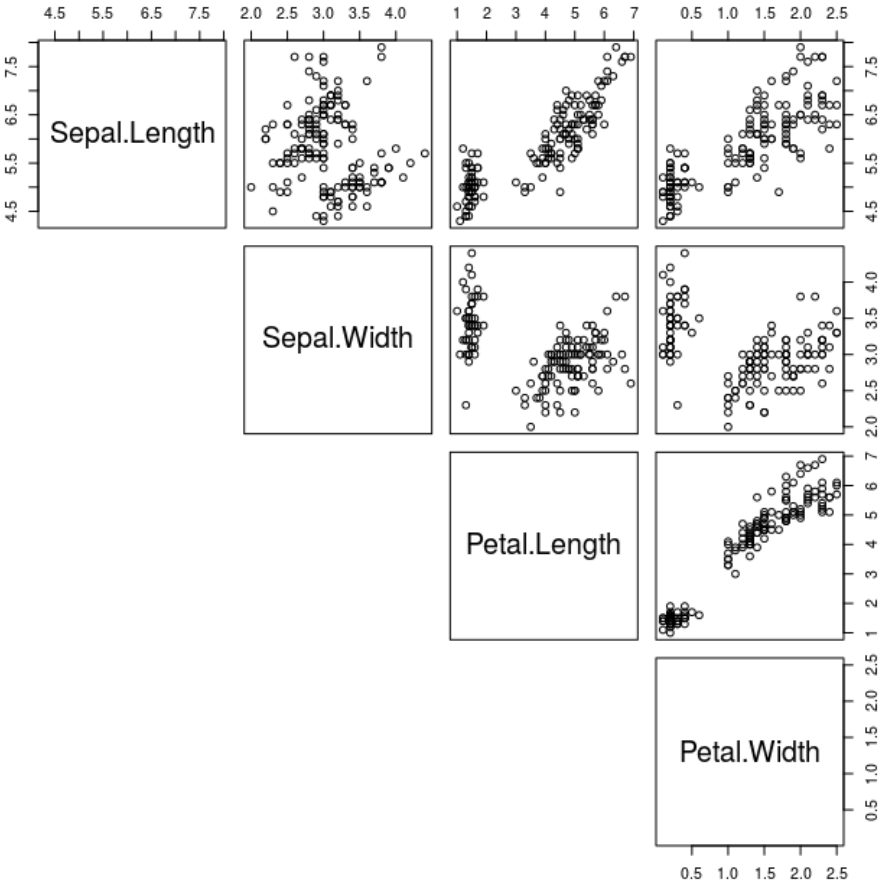
Cuando se dispone de múltiples variables y se quiere estudiar la relación entre todas ellas se recurre al cálculo de matrices con el coeficiente de correlación de cada par de variables (*pairwise correlation*). También se generan gráficos de dispersión dos a dos. En R existen diferentes funciones que permiten realizar este estudio, las diferencias entre ellas son el modo en que se representan gráficamente los resultados.

Se quiere estudiar la relación entre el tamaño de diferentes partes de las flores. Para ello, se dispone del set de datos iris que consta de cuatro variables numéricas.

```
data(iris)
# Se seleccionan únicamente las variables numéricas
datos <- iris[, c(1, 2, 3, 4)]
head(datos)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
## 4          4.6         3.1          1.5         0.2
## 5          5.0         3.6          1.4         0.2
## 6          5.4         3.9          1.7         0.4
```

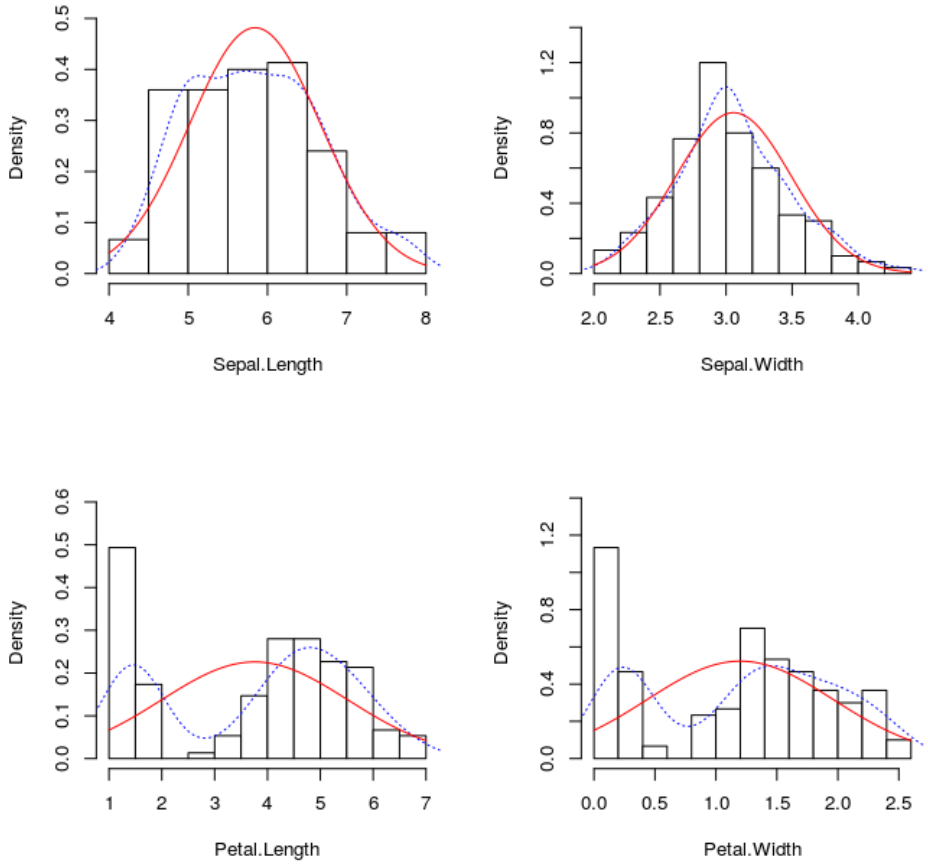
```
pairs(x = datos, lower.panel = NULL)
# No se muestra la diagonal inferior ya que es lo mismo que la superior
```




```
cor(x = datos, method = "pearson")
```

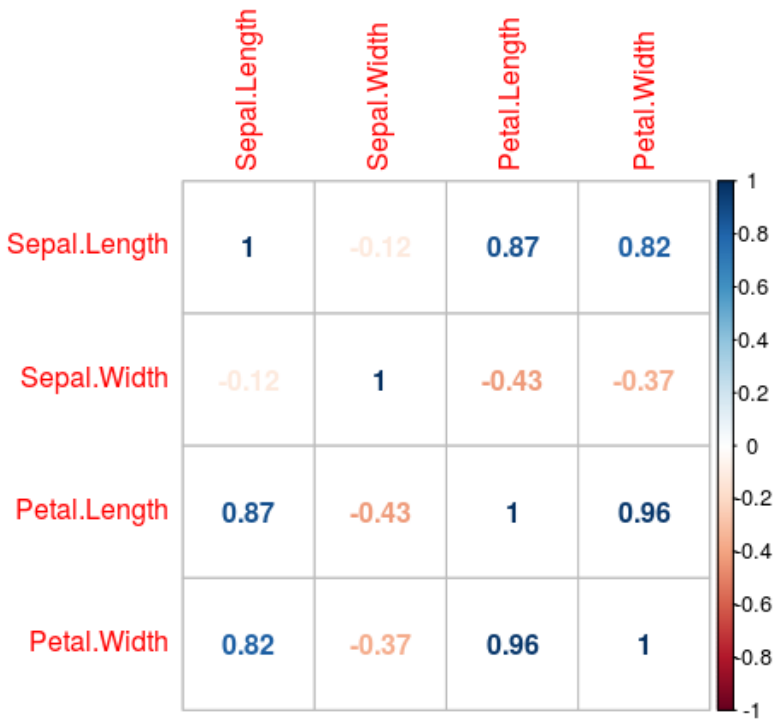
```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
## Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
## Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
## Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

```
require(psych)
multi.hist(x = datos, dcol = c("blue", "red"),
           dlty = c("dotted", "solid"), main = "")
```



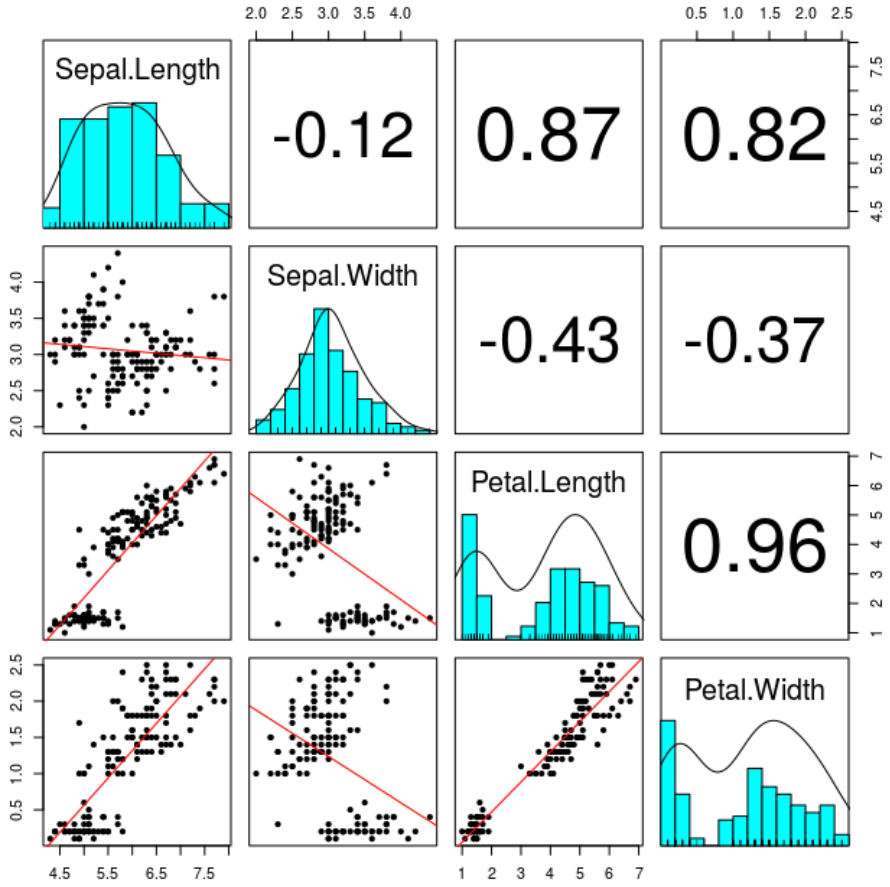
La función `corrplot()` del paquete *`corrplot`* recibe como argumento la matriz de correlaciones generada por la función `cor()` y genera diferentes tipos de *heat maps* mucho más visuales que la matriz numérica.

```
require(corrplot)
corrplot(corr = cor(x = datos, method = "pearson"), method = "number")
```



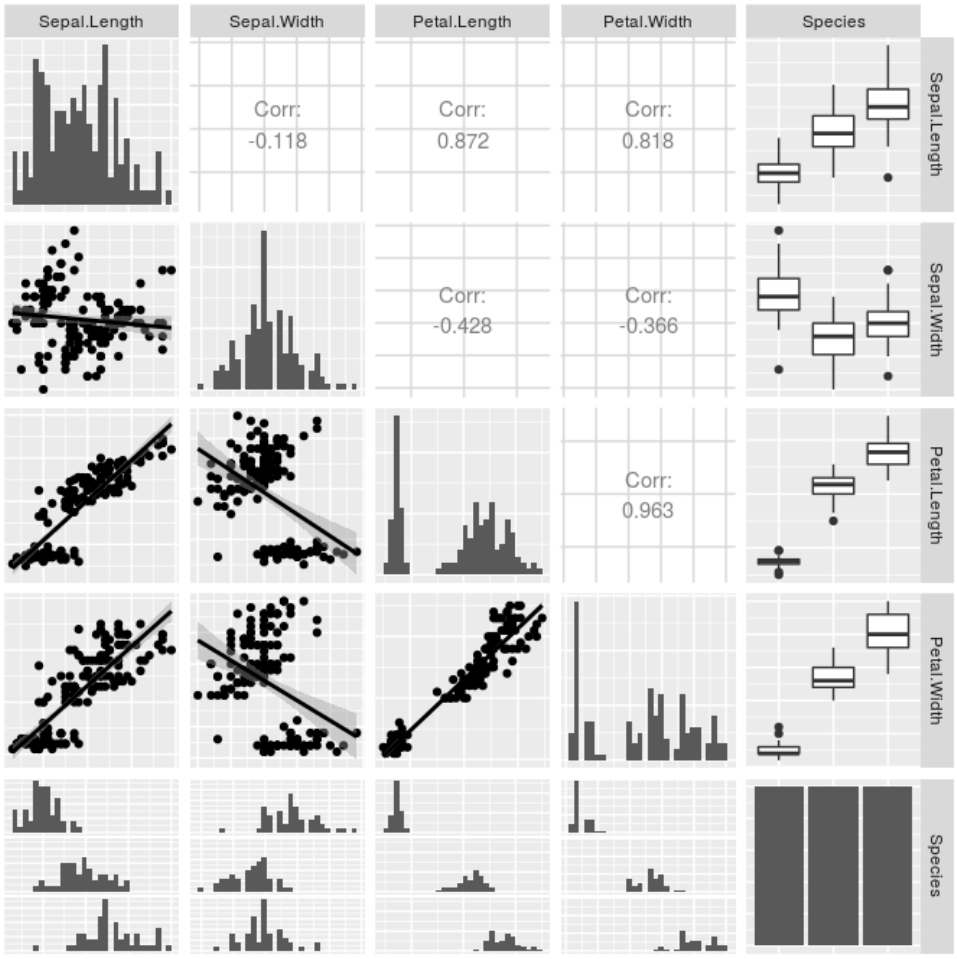
Otros paquetes permiten representar a la vez los diagramas de dispersión y los valores de correlación para cada par de variables. Además de la distribución de cada una de las variables.

```
require(psych)
pairs.panels(x = datos, ellipses = FALSE, lm = TRUE, method = "pearson")
```



La función `ggpairs()` del paquete *`GGally`* basada en *`ggplot2`* representa los diagramas de dispersión, el valor de la correlación e incluso interpreta el tipo de variable para que, en caso de ser categórica, representarla en forma de *boxplot*.

```
require(GGally)
ggpairs(iris, lower = list(continuous = "smooth"),
        diag = list(continuous = "bar"), axisLabels = "none")
```



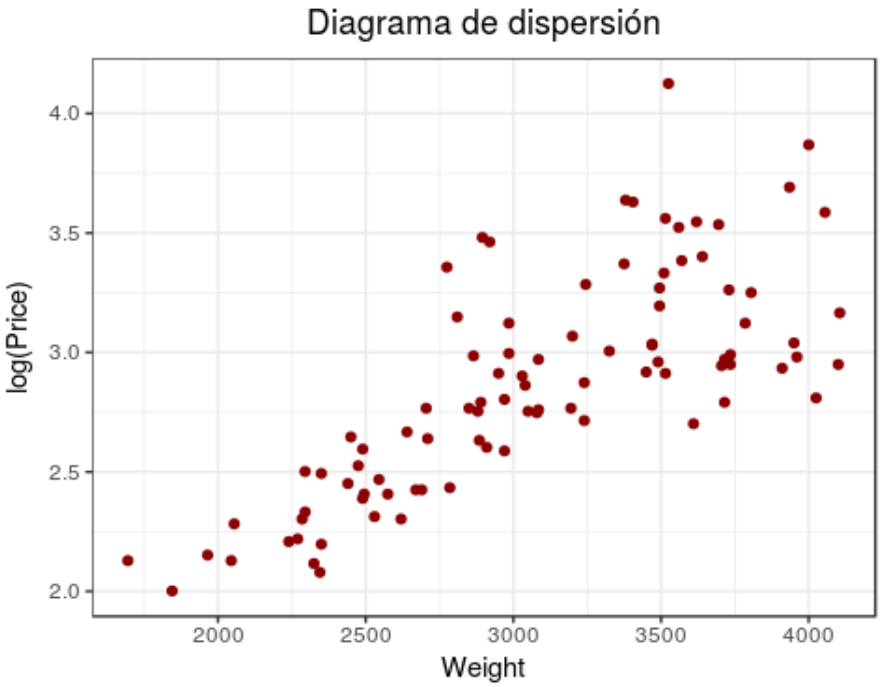
Correlación parcial

Como se ha explicado, la correlación estudia la relación (lineal o monotónica) existente entre dos variables. Puede ocurrir que la relación que muestran dos variables se deba a una tercera variable que influye sobre las otras dos, a este fenómeno se le conoce como *confounding*. Por ejemplo, si se correlaciona el tamaño del pie de una persona con su inteligencia, se encuentra una correlación positiva alta. Sin embargo, dicha relación se debe a una tercera variable que está relacionada con las otras dos, la edad. La correlación parcial permite estudiar la relación lineal entre dos variables bloqueando el efecto de una tercera (o más) variables. Si el valor de correlación de dos variables es distinto al valor de correlación parcial de esas mismas dos variables cuando se controla una tercera, significa que la tercera variable influye en las otras dos. La función en `pcor.test()` del paquete `ppcor` permite estudiar correlaciones parciales.

Ejemplo

Se quiere estudiar la relación entre las variables precio y peso de los automóviles. Se sospecha que esta relación podría estar influenciada por la variable potencia del motor, ya que a mayor peso del vehículo se requiere mayor potencia y, a su vez, motores más potentes son más caros.

```
require(MASS)
require(ggplot2)
data("Cars93")
# Se emplea el log del precio porque mejora la linealidad
ggplot(data = Cars93, aes(x = Weight, y = log(Price))) + geom_point(colou
r = "red4") +
ggtitle("Diagrama de dispersión") +
theme_bw() +
theme(plot.title = element_text(hjust = 0.5))
```



El gráfico permite intuir que existe una relación lineal entre el peso de un coche y el logaritmo de su precio.

```
require(ppcor)
cor.test(x = Cars93$Weight, y = log(Cars93$Price), method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  Cars93$Weight and log(Cars93$Price)
## t = 11.279, df = 91, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6629513 0.8370563
## sample estimates:
##      cor
## 0.763544
```

```
pcor.test(x = Cars93$Weight, y = log(Cars93$Price),
          z = Cars93$Horsepower, method = "pearson")
```

```
##    estimate      p.value statistic  n gp  Method
## 1 0.4047414 6.288649e-05  4.199019 93  1 pearson
```

La correlación entre el peso y el logaritmo del precio es alta (r=0.764) y significativa (*p-value* < 2.2e-16). Sin embargo, cuando se estudia su relación bloqueando la variable potencia de motor, a pesar de que la relación sigue siendo significativa (*p-value* = 6.288649e-05) pasa a ser baja (r=0.4047).

Conclusión

La relación lineal existente entre el peso y el logaritmo del precio está influenciada por el efecto de la variable potencia de motor. Si se controla el efecto de la potencia, la relación lineal existente es baja (r=0.4047).

Regresión lineal simple

La información aquí presente recoge los principales conceptos de la regresión lineal. Se puede encontrar una descripción mucho más detallada en los libros Introduction to Statistical Learning y en Linear Models with R.

La regresión lineal simple consiste en generar un modelo de regresión (ecuación de una recta) que permita explicar la relación lineal que existe entre dos variables. A la variable dependiente o respuesta se le identifica como Y y a la variable predictora o independiente como X.

El modelo de regresión lineal simple se describe de acuerdo a la ecuación:

$$Y = \beta_0 + \beta_1 X_1 + \epsilon$$

siendo β_0 la ordenada en el origen, β_1 la pendiente y ϵ el error aleatorio. Este último representa la diferencia entre el valor ajustado por la recta y el valor real. Recoge el efecto de todas aquellas variables que influyen en Y pero que no se incluyen en el modelo como predictores. Al error aleatorio también se le conoce como residuo.

En la gran mayoría de casos, los valores β_0 y β_1 poblacionales son desconocidos, por lo que, a partir de una muestra, se obtienen sus estimaciones $\hat{\beta}_0$ y $\hat{\beta}_1$. Estas estimaciones se conocen como coeficientes de regresión o *least square coefficient estimates*, ya que toman aquellos valores que minimizan la suma de cuadrados residuales, dando lugar a la recta que pasa más cerca de todos los puntos. (Existen alternativas al método de mínimos cuadrados para obtener las estimaciones de los coeficientes).

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2} = \frac{Sy}{Sx} R$$

$$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x}$$

donde Sy y Sx son las desviaciones típicas de cada variable y R el coeficiente de correlación.

$\hat{\beta}_0$ es el valor esperado la variable Y cuando $X = 0$, es decir, la intersección de la recta con el eje y. Es un dato necesario para generar la recta pero, en ocasiones, no tiene interpretación práctica (situaciones en las que X no puede adquirir el valor 0).

Una recta de regresión puede emplearse para diferentes propósitos y dependiendo de ellos es necesario satisfacer distintas condiciones. En caso de querer medir la relación lineal entre dos variables, la recta de regresión lo va a indicar de forma directa (ya que calcula la correlación). Sin embargo, en caso de querer predecir el valor de una variable en función de la otra, no solo se necesita calcular la recta, sino que además hay que asegurar que el modelo sea bueno.

Inferencia mediante regresión lineal. Significancia e intervalo de confianza para β_0 y β_1

En la mayoría de casos, aunque el estudio de regresión se aplica a una muestra, el objetivo último es obtener un modelo lineal que explique la relación entre las dos variables en toda la población. Esto significa que el modelo generado es una estimación de la relación poblacional a partir de la relación que se observa en la muestra y, por lo tanto, está sujeta a variaciones. Para cada uno de los parámetros de la ecuación de regresión lineal simple (β_0 y β_1) se puede calcular su significancia (*p-value*) y su intervalo de confianza. El test estadístico más empleado es el *t-test* (existen alternativas no paramétricas).

El test de significancia para la pendiente (β_1) del modelo lineal considera como hipótesis:

- H_0: No hay relación lineal entre ambas variables por lo que la pendiente del modelo lineal es cero. $\beta 1 = 0$
- H_a: Sí hay relación lineal entre ambas variables por lo que la pendiente del modelo lineal es distinta de cero. $\beta 1 \neq 0$

De esta misma forma también se aplica a (β_0)

Cálculo del estadístico T y del *p-value*:

$$t = \frac{\hat{\beta}_1 - 0}{SE(\hat{\beta}_1)} \quad ; \quad t = \frac{\hat{\beta}_0 - 0}{SE(\hat{\beta}_0)}$$

El error estándar de $\hat{\beta}_0$ y $\hat{\beta}_1$ se calcula con las siguientes ecuaciones:

$$SE(\hat{\beta}_0)^2 = \sigma^2 \left[\frac{1}{n} + \frac{\overline{x}^2}{\sum_{i=1}^{n}(x_i - \overline{x})^2}\right]$$

$$SE(\hat{\beta}_1)^2 = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

La varianza del error σ^2 se estima a partir del *Residual Standar Error (RSE)*, que puede entenderse como la diferencia promedio que se desvía la variable respuesta de la verdadera línea de regresión. En el caso de regresión lineal simple, *RSE* equivale a:

$$RSE = \sqrt{\frac{1}{n-2}RSS} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)}$$

- Grados de libertad (*df*) = número observaciones - 2 = número observaciones - número predictores - 1
- *p-value* = P(|t| > valor calculado de t)

Intervalos de confianza

$$\hat{\beta}_0 \pm t_{df}^{\alpha/2} SE(\hat{\beta}_0)$$

$$\hat{\beta}_1 \pm t_{df}^{\alpha/2} SE(\hat{\beta}_1)$$

Cuanto menor es el número de observaciones *n*, menor la capacidad para calcular el error estándar del modelo. Como consecuencia, la exactitud de los coeficientes de regresión estimados se reduce. Esto tiene importancia sobretodo en la regresión múltiple.

En R, cuando se genera el modelo de regresión lineal, se devuelve junto con el valor de la pendiente y la ordenada en el origen el valor del estadístico *t* obtenido para cada uno y los *p-value* correspondientes. Esto permite saber, además de la estimación de β_0 y β_1 , si son significativamente distintos de 0. Si se desea conocer los intervalos de confianza para cada uno de los parámetros se pueden calcular con la función `conint()`.

Residuos del modelo

El residuo de una estimación se define como la diferencia entre el valor observado y el valor esperado acorde al modelo. A la hora de sumarizar el conjunto de residuos hay dos posibilidades:

- El sumatorio del valor absoluto de cada residuo.
- El sumatorio del cuadrado de cada residuo (*RSS*). Esta es la aproximación más empleada (mínimos cuadrados) ya que magnifica las desviaciones más extremas. En R, cuando se genera un modelo los residuos también se calculan automáticamente y se almacenados dentro del modelo.

Cuanto mayor es el sumatorio del cuadrado de los residuos menor la precisión con la que el modelo puede predecir el valor de la variable dependiente a partir de la variable predictora. Los residuos son muy importantes puesto que en ellos se basan las diferentes medidas de la bondad de ajuste del modelo.

Bondad de ajuste del modelo

Una vez que se ha ajustado un modelo es necesario verificar su eficiencia, ya que aun siendo la línea que mejor se ajusta a las observaciones de entre todas las posibles, el modelo puede ser malo. Las medidas más utilizadas para medir la calidad del ajuste son: *error estándar de los residuos*, *el test F* y *el coeficiente de determinación* R^2.

Error estándar de los residuos (Residual Standar Error, RSE): Mide la desviación promedio de cualquier punto estimado por el modelo respecto de la verdadera recta de regresión poblacional. Tiene las mismas unidades que la variable dependiente *Y*. Una forma de saber si el valor del *RSE* es grande consiste en dividirlo entre el valor medio de la variable respuesta, obteniendo así un % de la desviación.

$$RSE = \sqrt{\frac{1}{n-p-1}RSS}$$

En modelos lineales simples, dado que hay un único predictor

$$(n-p-1) = (n-2)$$

Coeficiente de determinación R^2: Describe la proporción de variabilidad observada en la variable dependiente Y explicada por el modelo y relativa a la variabilidad total. Su valor está acotado entre 0 y 1. Al ser adimensional presenta la ventaja frente al *RSE* de ser más fácil de interpretar.

$$R^2 = \frac{Suma\ de\ cuadrados\ totales - Suma\ de\ cuadrados\ residuales}{Suma\ de\ cuadrados\ totales} =$$

$$1 - \frac{Suma\ de\ cuadrados\ residuales}{Suma\ de\ cuadrados\ totales} =$$

$$1 - \frac{\sum(\hat{y}_i - y_i)^2}{\sum(y_i - \overline{y})^2}$$

En los modelos de regresión lineal simple el valor de R^2 se corresponde con el cuadrado del *coeficiente de correlación de Pearson (r)* entre X e Y, no siendo así en regresión múltiple. Existe una modificación de R^2 conocida como $R^2 - ajustado$ que se emplea principalmente en los modelos de regresión múltiple. Introduce una penalización cuantos más predictores se incorporan al modelo. En los modelos lineales simples no se emplea.

Test F: El test F es un test de hipótesis que considera como hipótesis nula que todos los coeficientes de correlación estimados son cero, frente a la hipótesis alternativa de que al menos uno de ellos no lo es. Se emplea en modelos de regresión múltiple para saber si al menos alguno de los predictores introducidos en el modelo contribuye de forma significativa. En modelos lineales simples, dado que solo hay un predictor, el *p-value* del test F es igual al *p-value* del *t-test* del predictor.

Condiciones para la regresión lineal

1. **Linealidad:** La relación entre ambas variables debe ser lineal. Para comprobarlo se puede recurrir a:
   - Graficar ambas variables a la vez (*scatterplot* o diagrama de dispersión), superponiendo la recta del modelo generado por regresión lineal.
   - Calcular los residuos para cada observación acorde al modelo generado y graficarlos (*scatterplot*). Deben distribuirse de forma aleatoria en torno al valor 0.
2. **Distribución Normal de los residuos:** Los residuos se tiene que distribuir de forma normal, con media igual a 0. Esto se puede comprobar con un histograma, con la distribución de cuantiles (*qqnorm() + qqline()*) o con un test de hipótesis de normalidad. Los valores extremos suelen ser una causa frecuente por la que se viola la condición de normalidad.

3. **Varianza de residuos constante (homocedasticidad):** La varianza de los residuos ha de ser aproximadamente constante a lo largo del eje X. Se puede comprobar mediante gráficos (*scatterplot*) de los residuos de cada observación (formas cónicas son un claro indicio de falta de homocedasticidad) o mediante contraste de hipótesis mediante el test de *Breusch-Pagan*.

4. **Valores atípicos y de alta influencia:** Hay que estudiar con detenimiento los valores atípicos o extremos ya que pueden generar una falsa correlación que realmente no existe, u ocultar una existente. (Ver descripción detallada en la sección de apuntes varios).

5. **Independencia, Autocorrelación**: Las observaciones deben ser independientes unas de otras. Esto es importante tenerlo en cuenta cuando se trata de mediciones temporales. Puede detectarse estudiando si los residuos siguen un patrón o tendencia. Otro caso frecuente es el de tener varias mediciones para un mismo sujeto. En estos casos, primero se obtiene la media de cada uno y después se ajusta el modelo empleando las medias.

Dado que las condiciones se verifican a partir de los residuos, primero se suele generar el modelo y después se valida. De hecho, el ajuste de un modelo debe verse como un proceso iterativo en el que se ajusta un modelo inicial, se evalúa mediante sus residuos y se mejora. Así hasta llegar a un modelo óptimo.

Predicción de valores

Una vez generado un modelo que se pueda considerar válido, es posible predecir el valor de la variable dependiente Y para nuevos valores de la variable predictora X. Es importante tener en cuenta que las predicciones deben, *a priori*, limitarse al rango de valores dentro del que se encuentran las observaciones con las que se ha generado el modelo. Esto es importante puesto que solo en esta región se tiene certeza de que se cumplen las condiciones para que el modelo sea válido. Para calcular las predicciones se emplea la ecuación generada por regresión.

Dado que el modelo generado se ha obtenido a partir de una muestra y por lo tanto las estimaciones de los coeficientes de regresión tienen un error asociado, también lo tienen los valores de las predicciones. Existen dos formas de medir la incertidumbre asociada con una predicción:

- **Intervalos de confianza:** Responden a la pregunta ¿Cuál es el intervalo de confianza del valor promedio de la variable respuesta Y para un determinado valor del predictor X?

$$\hat{y} \pm t_{\alpha/2,n-2} \sqrt{MSE\left(\frac{1}{n} + \frac{(x_k - \overline{x})^2}{\sum(x_i - \overline{x}^2)}\right)}$$

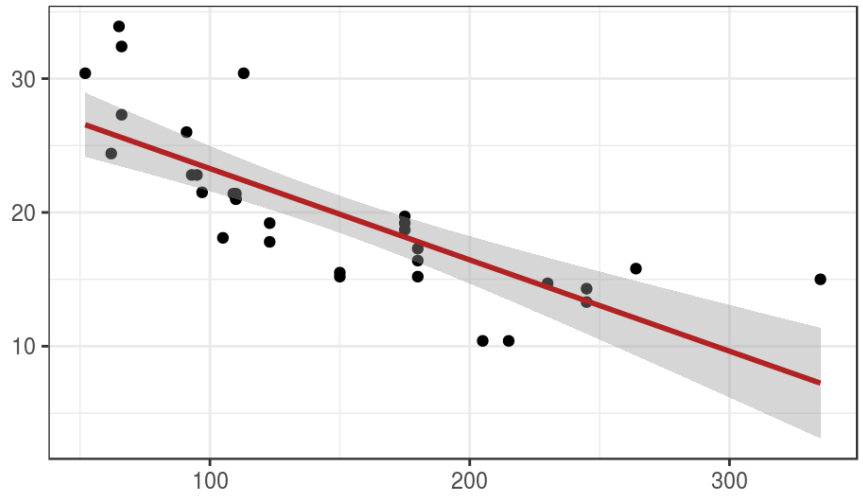
- **Intervalos de predicción:** Responden a la pregunta ¿Dentro de que intervalo se espera que esté el valor de la variable respuesta Y para un determinado valor del predictor X?

$$\hat{y} \pm t_{\alpha/2,n-2} \sqrt{MSE\left(1 + \frac{1}{n} + \frac{(x_k - \overline{x})^2}{\sum(x_i - \overline{x})^2}\right)}$$

Si bien ambas preguntas parecen similares, la diferencia se encuentra en que los intervalos de confianza se aplican al valor promedio que se espera de Y para un determinado valor de X, mientras que los intervalos de predicción no se aplican al promedio. Por esta razón los segundos siempre son más amplios que los primeros.

En `R` se puede emplear la función `predict()` que recibe como argumento el modelo calculado, un *dataframe* con los nuevos valores del predictor X y el tipo de intervalo (*confidence* o *prediction*).

Una característica que deriva de la forma en que se calcula el margen de error en los intervalos de confianza y predicción, es que el intervalo se ensancha a medida que los valores de X se aproximan a los extremos el rango observado.



¿Por qué ocurre esto? Prestando atención a la ecuación del error estándar del intervalo de confianza, el numerador contiene el término $(x_k - \overline{x})^2$ (lo mismo ocurre para el intervalo de predicción).

$$\sqrt{MSE\left(\frac{1}{n} + \frac{(x_k - \overline{x})^2}{\sum(x_i - \overline{x}^2)}\right)}$$

Este término se corresponde con la diferencia al cuadrado entre el valor x_k para el que se hace la predicción y la media $\overline{x}$ de los valores observados del predictor X. Cuanto más se aleje x_k de $\overline{x}$ mayor es el numerador y por lo tanto el error estándar.

Ejemplo

Un analista de deportes quiere saber si existe una relación entre el número de bateos que realiza un equipo de béisbol y el número de runs que consigue. En caso de existir y de establecer un modelo, podría predecir el resultado del partido.

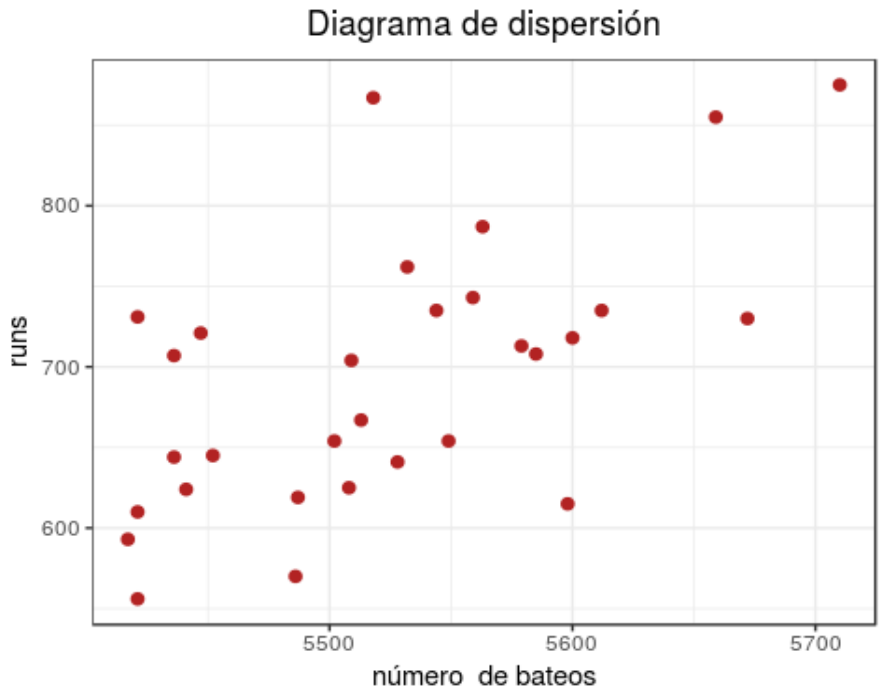
```
equipos <- c("Texas", "Boston", "Detroit", "Kansas", "St.", "New_S.", "Ne
w_Y.", "Milwaukee", "Colorado", "Houston", "Baltimore", "Los_An.", "Chica
go", "Cincinnati", "Los_P.", "Philadelphia", "Chicago", "Cleveland", "Ari
zona", "Toronto", "Minnesota", "Florida", "Pittsburgh", "Oakland", "Tampa
", "Atlanta", "Washington", "San.F", "San.I", "Seattle")
numero_bateos <- c(5659, 5710, 5563, 5672, 5532, 5600, 5518, 5447, 5544,
5598, 5585, 5436, 5549, 5612, 5513, 5579, 5502, 5509, 5421, 5559, 5487, 5
508, 5421, 5452, 5436, 5528, 5441, 5486, 5417, 5421)
runs <- c(855, 875, 787, 730, 762, 718, 867, 721, 735, 615, 708, 644, 654
, 735, 667, 713, 654, 704, 731, 743, 619, 625, 610, 645, 707, 641, 624, 5
70, 593, 556)
datos <- data.frame(equipos, numero_bateos, runs)
head(datos)
```

```
##   equipos numero_bateos runs
## 1   Texas          5659  855
## 2  Boston          5710  875
## 3 Detroit          5563  787
## 4  Kansas          5672  730
## 5     St.          5532  762
## 6  New_S.          5600  718
```

1.Representación gráfica de las observaciones

El primer paso antes de generar un modelo de regresión es representar los datos para poder intuir si existe una relación y cuantificar dicha relación mediante un coeficiente de correlación. Si en este paso no se detecta la posible relación lineal, no tiene sentido seguir adelante generando un modelo lineal (*se tendrían que probar otros modelos*).

```
require(ggplot2)
ggplot(data = datos, mapping = aes(x = numero_bateos, y = runs)) + geom_p
oint(color = "firebrick", size = 2) +
labs(title = "Diagrama de dispersión", x = "número  de bateos") +
theme_bw() +
theme(plot.title = element_text(hjust = 0.5))
```




```
cor.test(x = datos$numero_bateos, y = datos$runs, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  datos$numero_bateos and datos$runs
## t = 4.0801, df = 28, p-value = 0.0003388
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3209675 0.7958231
## sample estimates:
##      cor
## 0.610627
```

El gráfico y el test de correlación muestran una relación lineal, de intensidad considerable (r = 0.61) y significativa (*p-value* = 0.0003388). Tiene sentido intentar generar un modelo de regresión lineal que permita predecir el número de *runs* en función del número de bateos del equipo.

2.Cálculo del modelo de regresión lineal simple

```
modelo_lineal <- lm(runs ~ numero_bateos, datos)
# lm() devuelve el valor de la variable y para x=0 (intersección) junto c
on la pendiente de la recta.  Para ver la información del modelo se requi
ere summary().
summary(modelo_lineal)
```

```
##
## Call:
## lm(formula = runs ~ numero_bateos, data = datos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -125.58  -47.05  -16.59   54.40  176.87
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2789.2429   853.6957  -3.267 0.002871 **
## numero_bateos     0.6305     0.1545   4.080 0.000339 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66.47 on 28 degrees of freedom
## Multiple R-squared:  0.3729, Adjusted R-squared:  0.3505
## F-statistic: 16.65 on 1 and 28 DF,  p-value: 0.0003388
```

La primera columna (*Estimate*) devuelve el valor estimado para los dos parámetros de la ecuación del modelo lineal ($\hat{\beta}_0$ y $\hat{\beta}_1$) que equivalen a la ordenada en el origen y la pendiente.

Se muestran los errores estándar, el valor del estadístico *t* y el *p-value* (dos colas) de cada uno de los dos parámetros. Esto permite determinar si los parámetros son significativamente distintos de 0, es decir, que tienen importancia en el modelo. En los modelos de regresión lineal simple, el parámetro más informativo suele ser la pendiente. Para el modelo generado, tanto la ordenada en el origen como la pendiente son significativas (*p-values* < 0.05).

El valor de R^2 indica que el modelo calculado explica el 37.29% de la variabilidad presente en la variable respuesta (*runs*) mediante la variable independiente (*número de bateos*).

El *p-value* obtenido en el test F (0.0003388) determina que sí es significativamente superior la varianza explicada por el modelo en comparación a la

varianza total. Es el parámetro que determina si el modelo es significativo y por lo tanto se puede aceptar.

El modelo lineal generado sigue la ecuación *runs = -2789.2429 + 0.6305 bateos*. Por cada unidad que se incrementa el número de bateos, el número de runs aumenta **en promedio** 0.6305 unidades.

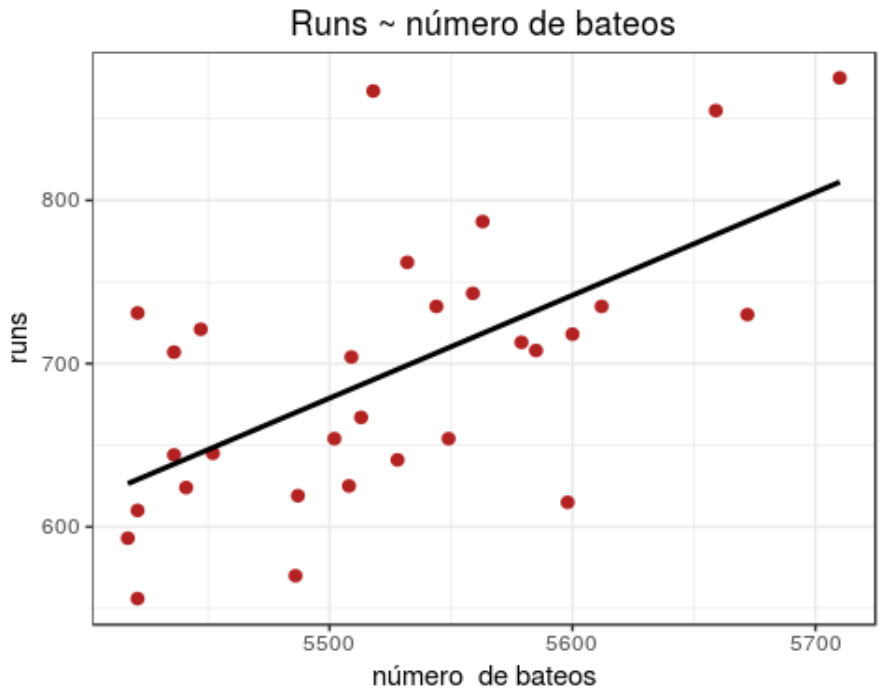
3.Intervalos de confianza para los parámetros del modelo

```
confint(modelo_lineal)
```

```
##                       2.5 %        97.5 %
## (Intercept)   -4537.9592982 -1040.5264727
## numero_bateos     0.3139863     0.9471137
```

4.Representación gráfica del modelo

```
ggplot(data = datos, mapping = aes(x = numero_bateos, y = runs)) + geom_p
oint(color = "firebrick", size = 2) +
labs(title = "Runs ~ número de bateos", x = "número  de bateos") +
geom_smooth(method = "lm", se = FALSE, color = "black") +
theme_bw() + theme(plot.title = element_text(hjust = 0.5))
```



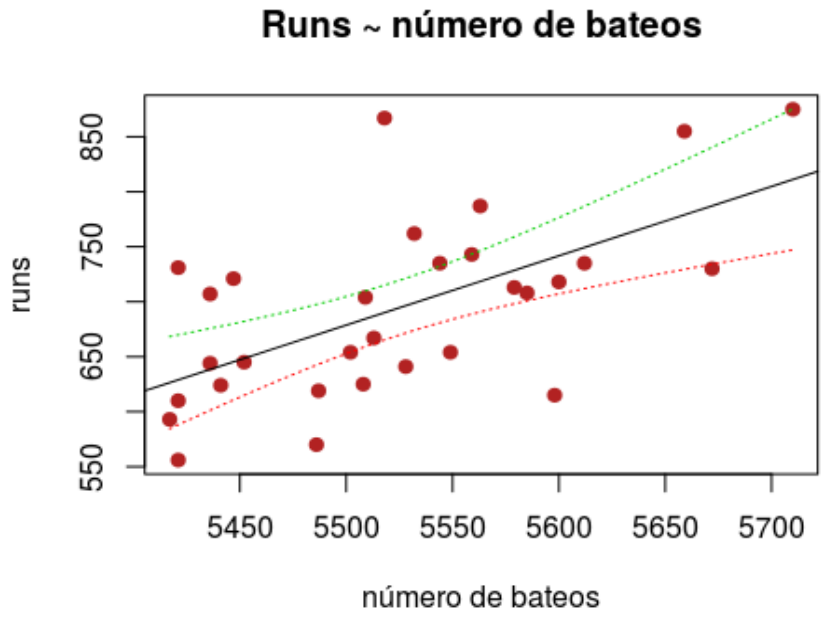
Además de la línea de mínimos cuadrados es recomendable incluir los límites superior e inferior del intervalo de confianza. Esto permite identificar la región en la que, según el modelo generado y para un determinado nivel de confianza, se encuentra el valor promedio de la variable dependiente.

Para poder representar el intervalo de confianza a lo largo de todo el modelo se recurre a la función `predict()` para predecir valores que abarquen todo el eje *X*. Se añaden al gráfico líneas formadas por los límites superiores e inferiores calculados para cada predicción.

```
# Se genera una secuencia de valores x_i que abarquen todo el rango de la
s observaciones de la variable X
puntos <- seq(from = min(datos$numero_bateos),
              to = max(datos$numero_bateos), length.out = 100)
# Se predice el valor de la variable Y junto con su intervalo de confianz
a para cada uno de los puntos generados. En la función predict() hay que
nombrar a los nuevos puntos con el mismo nombre que la variable X del mod
elo.  Devuelve una matriz.
limites_intervalo <- predict(object = modelo_lineal,
                             newdata = data.frame(numero_bateos = puntos),
                             interval = "confidence", level = 0.95)
head(limites_intervalo, 3)
```

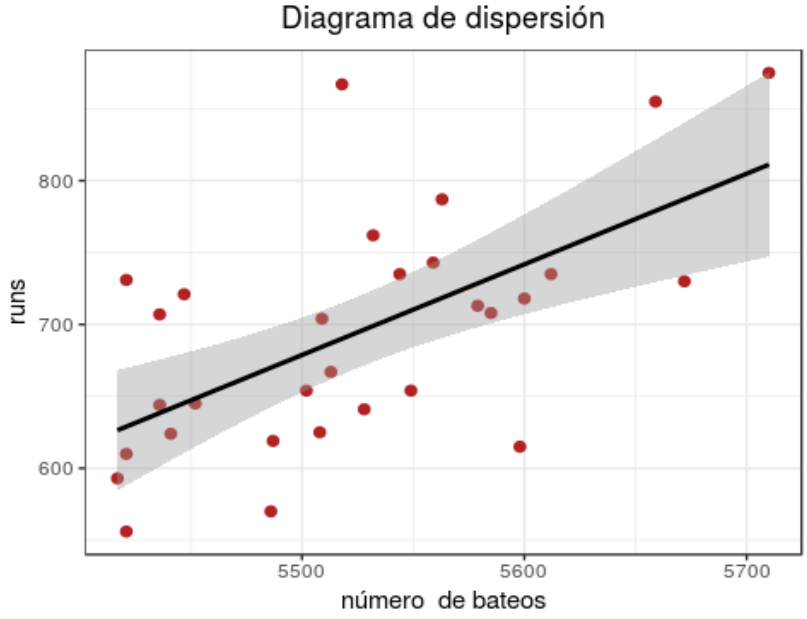
```
##        fit      lwr      upr
## 1 626.4464 584.5579 668.3350
## 2 628.3126 587.1743 669.4509
## 3 630.1788 589.7830 670.5745
```

```
# Finalmente se añaden al gráfico las líneas formadas por los límites
# superior e inferior.
plot(datos$numero_bateos, datos$runs, col = "firebrick", pch = 19,
     ylab = "runs", xlab = "número de bateos",
     main = "Runs ~ número de bateos")
abline(modelo_lineal, col = 1)
lines(x = puntos, y = limites_intervalo[, 2], type = "l", col = 2,
      lty = 3)
lines(x = puntos, y = limites_intervalo[, 3], type = "l", col = 3,
      lty = 3)
```



La función `geom_smooth()` del paquete *`ggplot2`* genera la regresión y su intervalo de forma directa.

```
ggplot(data = datos, mapping = aes(x = numero_bateos, y = runs)) + geom_p
oint(color = "firebrick", size = 2) +
labs(title = "Diagrama de dispersión", x = "número  de bateos") +
geom_smooth(method = "lm", se = TRUE, color = "black") +
theme_bw() + theme(plot.title = element_text(hjust = 0.5))
# Por defecto incluye la región de 95% de confianza.
```



5.Verificar condiciones para poder aceptar un modelo lineal

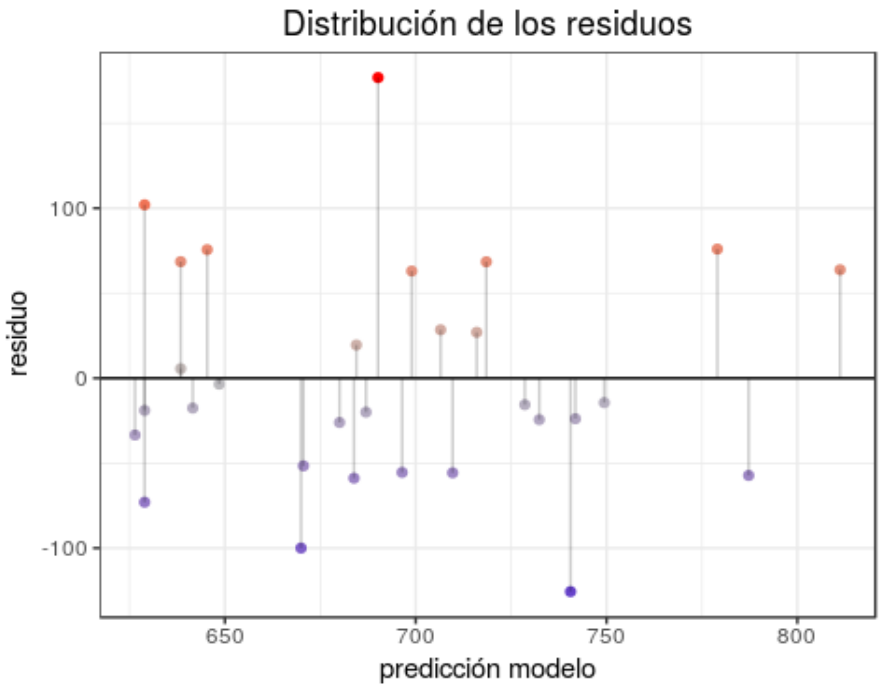
Relación lineal entre variable dependiente e independiente:

Se calculan los residuos para cada observación y se representan (scatterplot). Si las observaciones siguen la línea del modelo, los residuos se deben distribuir aleatoriamente entorno al valor 0.

```
# La función lm() calcula y almacena los valores predichos por el modelo
y los residuos.
datos$prediccion <- modelo_lineal$fitted.values
datos$residuos <- modelo_lineal$residuals
head(datos)
```

```
##   equipos numero_bateos runs prediccion  residuos
## 1   Texas          5659  855   779.0395  75.96048
## 2  Boston          5710  875   811.1976  63.80243
## 3 Detroit          5563  787   718.5067  68.49328
## 4  Kansas          5672  730   787.2367 -57.23667
## 5     St.          5532  762   698.9597  63.04033
## 6  New_S.          5600  718   741.8371 -23.83707
```

```
ggplot(data = datos, aes(x = prediccion, y = residuos)) + geom_point(aes(
color = residuos)) +
scale_color_gradient2(low = "blue3", mid = "grey", high = "red") + geom_h
line(yintercept = 0) +
geom_segment(aes(xend = prediccion, yend = 0), alpha = 0.2) +
labs(title = "Distribución de los residuos", x = "predicción modelo",
     y = "residuo") +
theme_bw() +
theme(plot.title = element_text(hjust = 0.5), legend.position = "none")
```

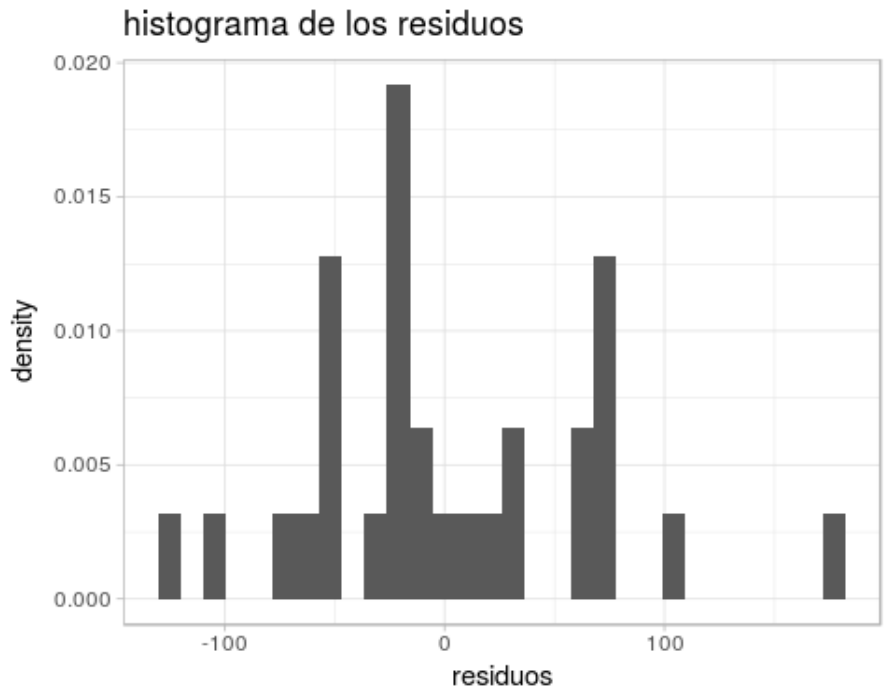


Los residuos se distribuyen de forma aleatoria entorno al 0 por lo que se acepta la linealidad.

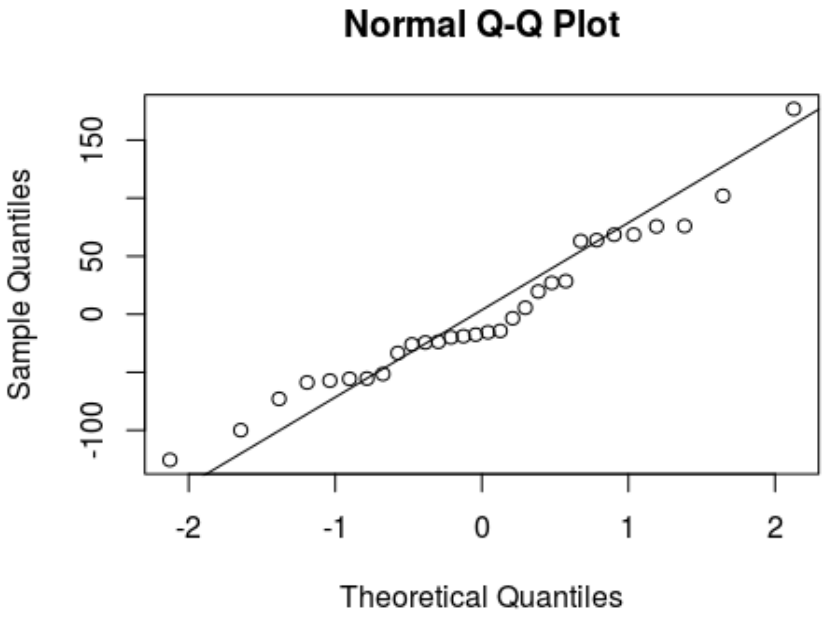
Distribución normal de los residuos:

Los residuos se deben distribuir de forma normal con media 0. Para comprobarlo se recurre a histogramas, a los cuantiles normales o a un test de contraste de normalidad.

```
ggplot(data = datos, aes(x = residuos)) +
geom_histogram(aes(y = ..density..)) +
labs(title = "Histograma de los residuos") +
theme_bw() + theme(plot.title = element_text(hjust = 0.5)
```




```
qqnorm(modelo_lineal$residuals)
qqline(modelo_lineal$residuals)
```



```
shapiro.test(modelo_lineal$residuals)
```

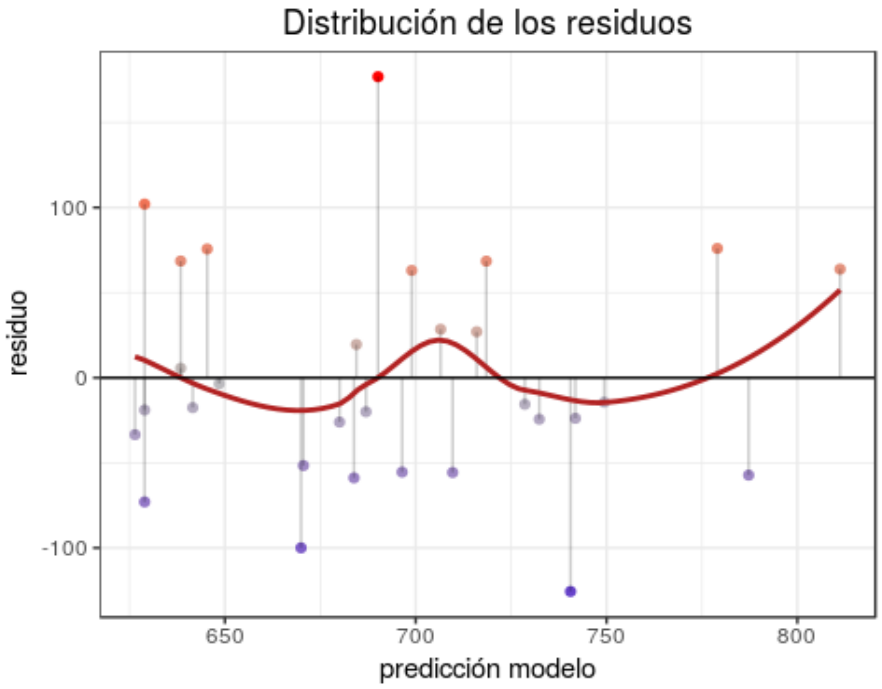
```
##  Shapiro-Wilk normality test
##
## data:  modelo_lineal$residuals
## W = 0.96144, p-value = 0.337
```

Tanto la representación gráfica como el contraste de hipótesis confirman la distribución normal de los residuos.

Varianza constante de los residuos (Homocedasticidad):

La variabilidad de los residuos debe de ser constante a lo largo del eje X. Un patrón cónico es indicativo de falta de homogeneidad en la varianza.

```
ggplot(data = datos, aes(x = prediccion, y = residuos)) + geom_point(aes(
color = residuos)) +
scale_color_gradient2(low = "blue3", mid = "grey", high = "red") + geom_s
egment(aes(xend = prediccion, yend = 0), alpha = 0.2) +
geom_smooth(se = FALSE, color = "firebrick") +
labs(title = "Distribución de los residuos", x = "predicción modelo",
     y = "residuo") +
geom_hline(yintercept = 0) +
theme_bw() +
theme(plot.title = element_text(hjust = 0.5), legend.position = "none")
```



```
# Test de Breush-Pagan
library(lmtest)
bptest(modelo_lineal)
```

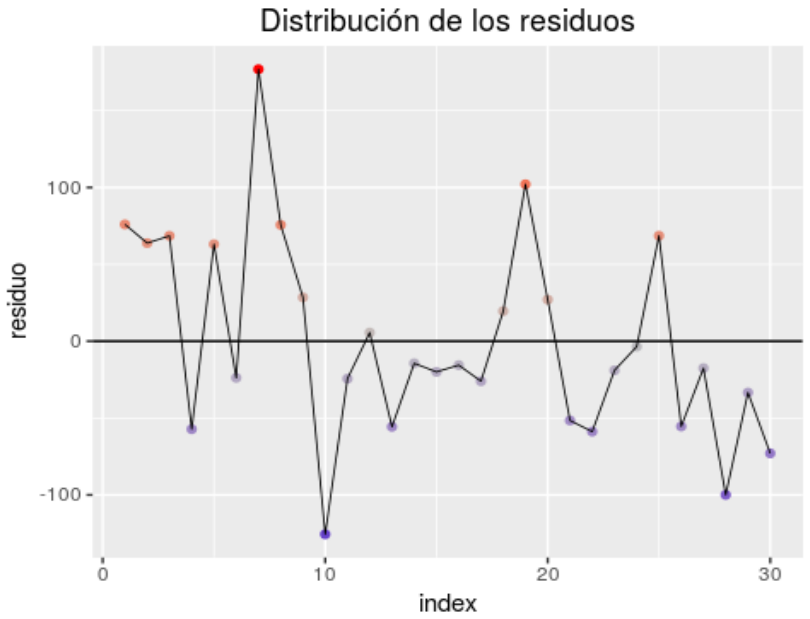
```
##  studentized Breusch-Pagan test
##
## data:  modelo_lineal
## BP = 0.01269, df = 1, p-value = 0.9103
```

Ni la representación gráfica ni el contraste de hipótesis muestran evidencias que haga sospechar falta de homocedasticidad.

Autocorrelación de residuos:

Cuando se trabaja con intervalos de tiempo, es muy importante comprobar que no existe aoutocorrelación de los residuos, es decir que son independientes. Esto puede hacerse detectando visualmente patrones en la distribución de los residuos cuando se ordenan según se han registrado o con el test de Durbin-Watson `dwt()` del paquete `Car`.

```
ggplot(data = datos, aes(x = seq_along(residuos), y = residuos)) + geom_p
oint(aes(color = residuos)) +
scale_color_gradient2(low = "blue3", mid = "grey", high = "red") + geom_l
ine(size = 0.3) +
labs(title = "Distribución de los residuos", x = "index", y = "residuo")+
geom_hline(yintercept = 0) +
 theme(plot.title = element_text(hjust = 0.5), legend.position = "none")
```



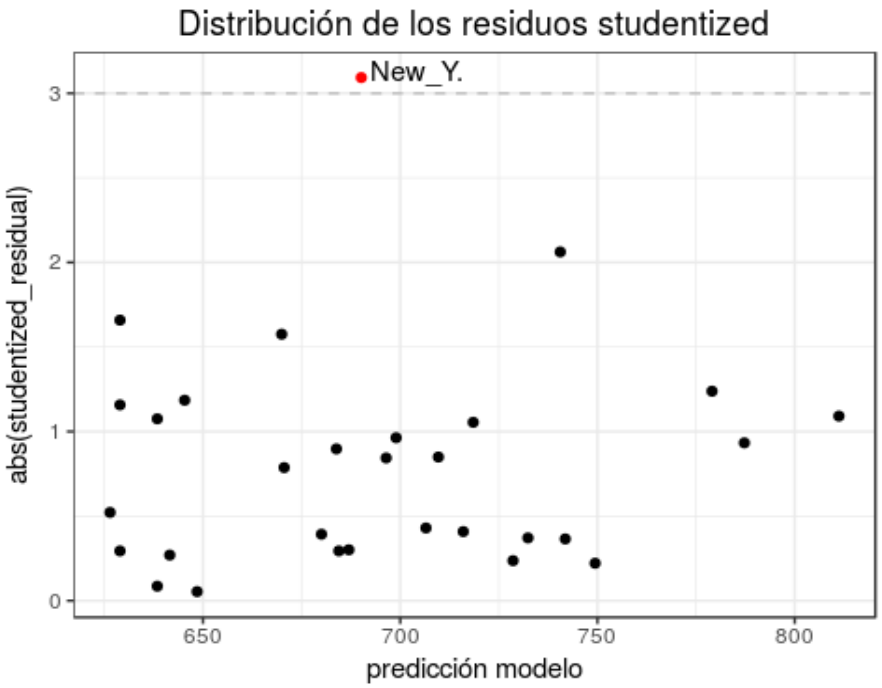
En este caso, la representación de los residuos no muestra ninguna tendencia.

6.Identificación de valores atípicos: *outliers*, *leverage* y observaciones influyentes

- **Outlier u observación atípica:** Observaciones que no se ajustan bien al modelo. El valor real se aleja mucho del valor predicho, por lo que su residuo es excesivamente grande. En una representación bidimensional se corresponde con desviaciones en el eje Y.
- **Observación influyente:** Observación que influye sustancialmente en el modelo, su exclusión afecta al ajuste. No todos los *outliers* tienen por qué ser influyentes.
- **Observación con alto leverage:** Observación con un valor extremo para alguno de los predictores. En una representación bidimensional se corresponde con desviaciones en el eje X. Son potencialmente puntos influyentes.

Independientemente de que el modelo se haya podido aceptar, siempre es conveniente identificar si hay algún posible *outlier*, *observación con alto leverage* u observación altamente influyente, puesto que podría estar condicionando en gran medida el modelo. La eliminación de este tipo de observaciones debe de analizarse con detalle y dependiendo de la finalidad del modelo. Si el fin es predictivo, un modelo sin estas observaciones puede lograr mayor precisión la mayoría de casos. Sin embargo, es muy importante prestar atención a estos valores ya que, de no ser errores de medida, pueden ser los casos más interesantes. El modo adecuado a proceder cuando se sospecha de algún posible valor atípico o influyente es calcular el modelo de regresión incluyendo y excluyendo dicho valor.

```
library(ggrepel)
library(dplyr)
datos$studentized_residual <- rstudent(modelo_lineal)
ggplot(data = datos,aes(x = prediccion, y = abs(studentized_residual))) +
geom_hline(yintercept = 3, color = "grey", linetype = "dashed") +
# Se identifican en rojo residuos studentized absolutos > 3
geom_point(aes(color = ifelse(abs(studentized_residual) > 3,
                              'red', 'black'))) +
scale_color_identity() +
#se muestra el equipo al que pertenece la observación atípica
geom_text_repel(data = filter(datos, abs(studentized_residual) > 3), aes(
label = equipos)) +
labs(title = "Distribución de los residuos studentized",
     x = "predicción modelo") +
theme_bw() +
theme(plot.title = element_text(hjust = 0.5), legend.position = "none")
```




```
datos %>% filter(abs(studentized_residual) > 3)
```

```
##   equipos numero_bateos runs prediccion residuos studentized_residual
## 1  New_Y.          5518  867    690.132  176.868             3.092876
```

```
which(abs(datos$studentized_residual) > 3)
```

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

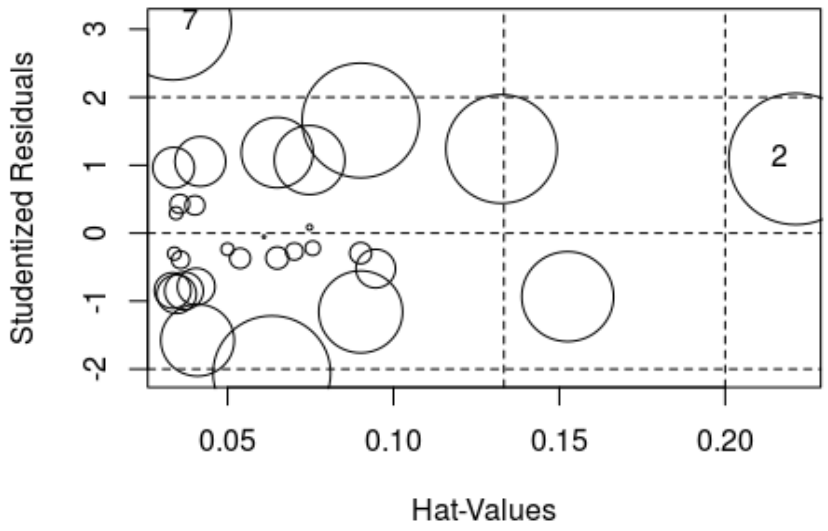
El estudio de los residuos *studentized* identifica al equipo de New_Y. como una posible observación atípica. Esta observación ocupa la posición 7 en la tabla de datos.

El hecho de que un valor sea atípico o con alto grado de *leverage* no implica que sea influyente en el conjunto del modelo. Sin embargo, si un valor es influyente, suele ser o atípico o de alto *leverage*. En R se dispone de la función `outlierTest()` del paquete `car` y de las funciones `influence.measures()`, `influencePlot()` y `hatvalues()` para identificar las observaciones más influyentes en el modelo.

```
library(car)
summary(influence.measures(model = modelo_lineal))
```

```
## Potentially influential observations of
##   lm(formula = runs ~ numero_bateos, data = datos) :
##
##   dfb.1_ dfb.nmr_ dffit cov.r   cook.d hat
## 2 -0.53   0.54     0.58  1.27_*  0.17   0.22_*
## 7  0.05  -0.04     0.58  0.61_*  0.13   0.03
```

```
influencePlot(model = modelo_lineal)
```

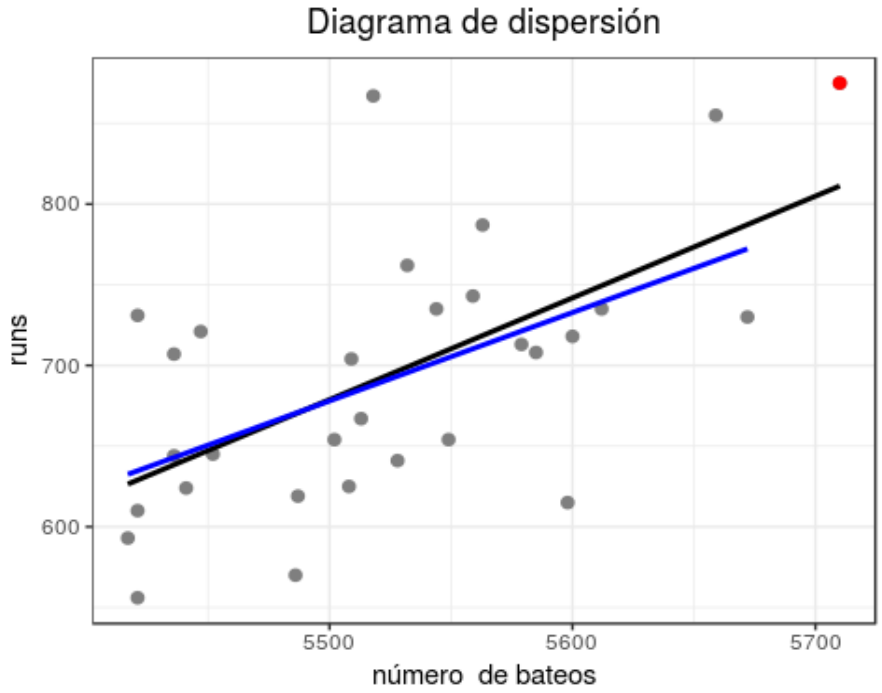



```
##    StudRes        Hat     CookD
## 2 1.091428 0.22133381 0.1681516
## 7 3.092876 0.03349684 0.1269339
```

Las funciones `influence.measures()` e `influencePlot()` detectan la observación 7 como atípica pero no significativamente influyente. Sí detectan como influyente la observación que ocupa la segunda posición. Para evaluar hasta qué punto condiciona el modelo, se recalcula la recta de mínimos cuadrados excluyendo esta observación.

```
ggplot(data = datos, mapping = aes(x = numero_bateos, y = runs)) + geom_p
oint(color = "grey50", size = 2) +
geom_smooth(method = "lm", se = FALSE, color = "black") +
# se resalta el valor excluido
geom_point(data = datos[2, ], color = "red", size = 2) +
# se añade la nueva recta de mínimos cuadrados
geom_smooth(data = datos[-2, ], method="lm", se =FALSE,color =  "blue") +
labs(title = "Diagrama de dispersión", x = "número  de bateos") + theme_b
```

```
w() +
theme(plot.title = element_text(hjust = 0.5), legend.position = "none")
```



La eliminación del valor identificado como influyente apenas cambia la recta de mínimos cuadrados. Para conocer con exactitud el resultado de excluir la observación se comparan las pendientes de ambos modelos.

```
lm(formula = runs ~ numero_bateos, data = datos)$coefficients
```

```
##   (Intercept) numero_bateos
##   -2789.24289       0.63055
```

```
lm(formula = runs ~ numero_bateos, data = datos[-2, ])$coefficients
```

```
##   (Intercept) numero_bateos
## -2335.7478247     0.5479527
```

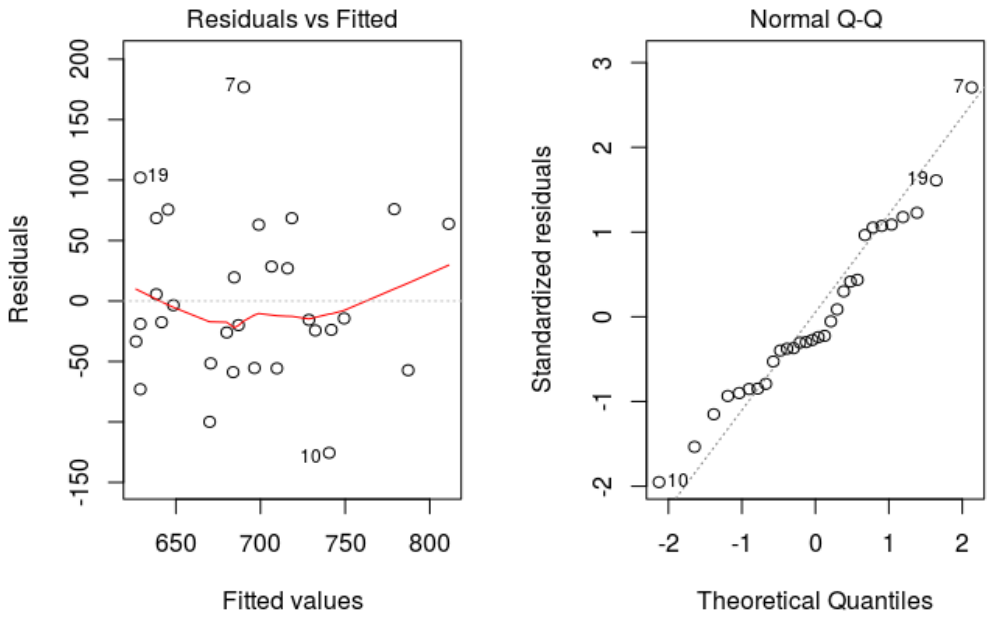
Conclusión

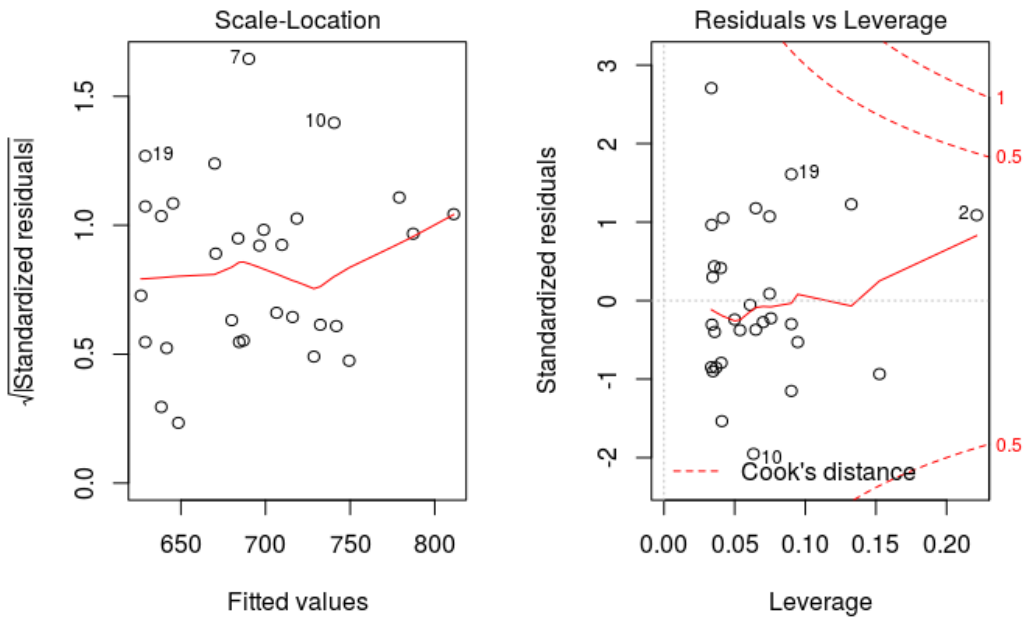
Dado que se satisfacen todas las condiciones para considerar válido un modelo de regresión lineal por mínimos cuadrados y que el *p-value* indica que el ajuste es significativo, se puede aceptar el modelo lineal. A pesar de ello, el valor de R^2 no es muy alto por lo que el número de bateos no es muy buen predictor del número de runs.

Evaluación de los residuos de un modelo lineal simple mediante gráficos R

Como se ha descrito en el apartado anterior, la evaluación de las condiciones que hacen válido un modelo de regresión lineal simple se hace en gran medida mediante representaciones gráficas de los residuos. El objeto devuelto por la función `lm()` puede pasarse como argumento a la función `plot()` obteniendo así 4 gráficos que permiten evaluar los residuos.

```
par(mfrow = c(1, 2))
plot(modelo_lineal)
```





Apuntes varios (miscellaneous)

En este apartado recojo comentarios, definiciones y puntualizaciones que he ido encontrando en diferentes fuentes y que, o bien no he tenido tiempo de introducir en el cuerpo principal del documento, o que he considerado mejor mantenerlos al margen como información complementaria.

Origen del método de mínimos cuadrados y regresión

Linear Models with R, by Julian J. Faraway

El método de mínimos cuadrados fue publicado en 1805 por Adrien Marie Legendre. El término *regresión* proviene de la publicación que hizo Francis Galton en 1985 llamada *Regression to mediocrity*. En ella, Galton emplea el método de mínimos cuadrados para demostrar que los hijos de parejas altas tienden a ser también altos pero no tanto como sus padres y que los hijos de parejas de poca estatura tienden a ser bajos pero no tanto como sus padres.

Significado de modelo lineal

Linear Models with R, by Julian J. Faraway

En los modelos lineales los parámetros se incorporan en la ecuación de forma lineal, sin embargo, los predictores no tienen por qué ser lineales. La siguiente ecuación muestra un modelo lineal en el que el predictor X_2 no es lineal:

$$y = \beta_0 + \beta_1 X_1 + \beta_2 log(X_2) + \epsilon$$

En contraposición, el siguiente no es un modelo lineal:

$$y = \beta_0 + \beta_1 X_1^{\beta_2} + \epsilon$$

En ocasiones, algunas relaciones no lineales pueden transformarse de forma que se pueden expresar de manera lineal:

$$y = \beta_0 X_1^{\beta_1} \epsilon$$

$$log(y) = log(\beta_0) + \beta_1 log(X_1) + log(\epsilon)$$

Estimación de la varianza de un modelo lineal por mínimos cuadrados

Linear Models with R, by Julian J. Faraway

La estimación de la varianza (σ^2) de un modelo lineal obtenido por mínimos cuadrados es:

$$\sigma^2 = \frac{RSS}{n-p}$$

El termino *Residual Standar Error (RSE)*, es la raíz cuadrada de σ^2:

$$RSE = \sqrt{\frac{RSS}{n-p}}$$

Ventajas del método de mínimos cuadrados para estimar los coeficientes de un modelo lineal

Linear Models with R, by Julian J. Faraway

Si bien existen alternativas al método de mínimos cuadrados para obtener la estimación de los coeficientes de correlación $\hat{\beta}_i$ de un modelo lineal, este presenta una serie de ventajas:

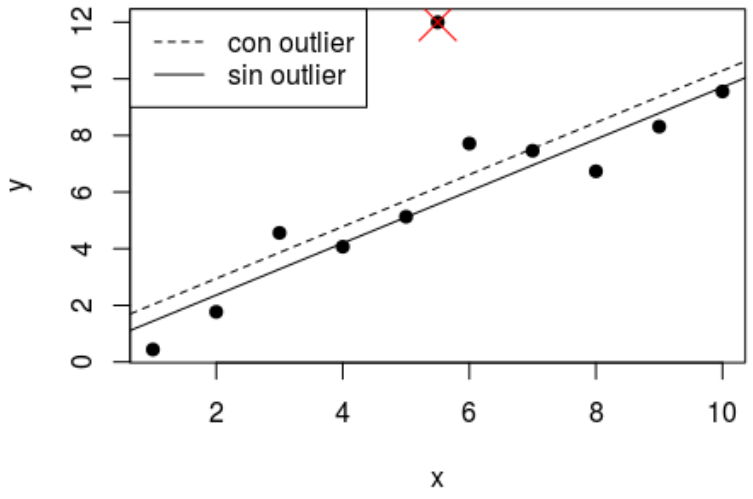
- Tiene una interpretación geométrica, lo que facilita su comprensión,
- Si los errores son independientes y se distribuyen de forma normal, su solución equivale a la estimación de máxima verosimilitud (*likelihood*).
- Los $\hat{\beta}_i$ son estimaciones insesgadas.

Identificación de outliers, observaciones con alto leverage y observaciones influyentes

Linear Models with R, by Julian J. Faraway

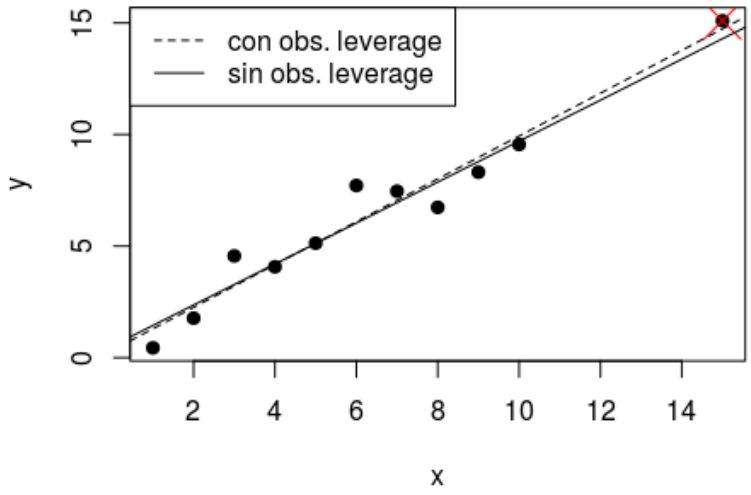
Outlier u observación atípica: Observaciones que no se ajustan bien al modelo. El valor real de la variable respuesta se aleja mucho del valor predicho, por lo que su residuo es excesivamente grande. En una representación bidimensional se corresponde con desviaciones en el eje Y.

```
set.seed(123)
datos <- data.frame(x = 1:10, y = 1:10 + rnorm(10))
modelo_1 <- lm(y ~ x, data = datos)

observacion <- c(5.5, 12)
modelo_2 <- lm(y ~ x, data = rbind(datos, observacion))
plot(y ~ x, data = rbind(datos, observacion), pch = 19)
points(5.5, 12, pch = 4, cex = 3, col = "red")
abline(modelo_1)
abline(modelo_2, lty = 2)
```



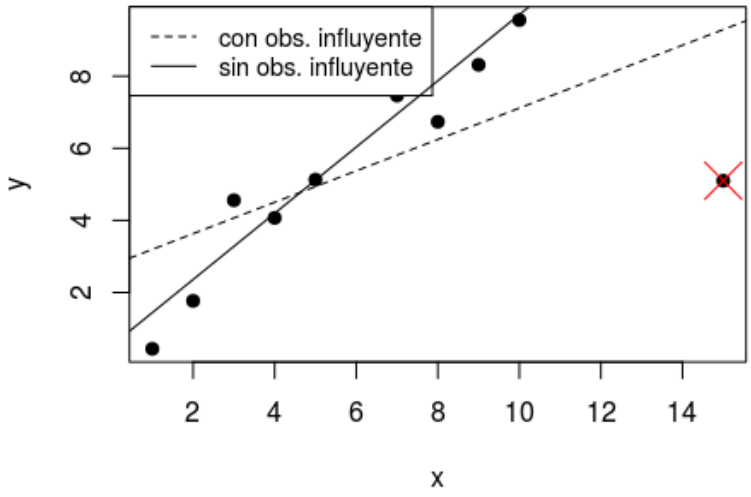
Observación con alto leverage: Observación con un valor extremo para alguno de los predictores. En una representación bidimensional se corresponde con desviaciones en el eje X. Son potencialmente puntos influyentes.

```
observacion <- c(15, 15.1)
modelo_2 <- lm(y ~ x, data = rbind(datos, observacion))
plot(y ~ x, data = rbind(datos, observacion), pch = 19)
points(15, 15.1, pch = 4, cex = 3, col = "red")
abline(modelo_1)
abline(modelo_2, lty = 2)
```



Observación influyente: Observación que influye sustancialmente en el modelo, su exclusión afecta al ajuste. No todos los *outliers* u observaciones con alto *leverage* tienen por qué ser influyentes.

```
observacion <- c(15, 5.1)
modelo_2 <- lm(y ~ x, data = rbind(datos, observacion))
plot(y ~ x, data = rbind(datos, observacion), pch = 19)
points(15, 5.1, pch = 4, cex = 3, col = "red")
abline(modelo_1)
abline(modelo_2, lty = 2)
```



Independientemente de que el modelo se haya podido aceptar, siempre es conveniente identificar si hay algún posible *outlier*, *observación con alto leverage* u observación altamente influyente, puesto que podría estar condicionando en gran medida el modelo. La eliminación de este tipo de observaciones debe de analizarse con detalle y dependiendo de la finalidad del modelo. Si el fin es predictivo, un modelo sin estas observaciones puede ser más útil para predecir con precisión la mayoría de casos. Sin embargo, es muy importante prestar atención a estos valores ya que, de no ser errores de medida, pueden ser los casos más interesantes. El modo adecuado a proceder cuando se sospecha de algún posible valor atípico o influyente es calcular el modelo de regresión incluyendo y excluyendo dicho valor.

Para el estudio de los grados de *leverage* (hi) de cada observación se tiene en cuenta únicamente el valor de los predictores X, no el de la variable respuesta Y. hi equivale al cuadrado de la distancia de *Mahalanobis*, por lo que valores altos se corresponden con valores extremos del o de los predictores. Observaciones cuyos valores (hi) superen $2.5x((p+1)/n)$, siendo p el número de predictores y n el número de observaciones, deben tenerse en consideración por su posible influencia. La función `hatvalues()` calcula el *leverage* de las observaciones de un modelo.

El estudio de *ouliers* puede hacerse utilizando los residuos. Si la variable respuesta real de una observación está muy alejada del valor esperado acorde al modelo, su residuo será grande. Asumiendo que los residuos de un modelo se distribuyen de forma normal, se pueden estandarizar/normalizar, $(\epsilon_i - \hat{\epsilon})/sd(\epsilon)$, e identificar aquellos cuyo valor exceda ± 3 como atípicos. Esta aproximación, aunque útil, tiene aun limitación importante. Si la observación es un *outlier* tal que influye sobre el modelo lo suficiente para aproximarlo hacia ella, el residuo será pequeño y pasará desapercibido en la estandarización. Una forma de evitar pasar por alto este tipo de *outliers* es emplear los residuos *studentized* (también conocidos como *jackknife residuals*) en lugar de los residuos estandarizados. Se trata de un proceso iterativo en el que se va excluyendo cada vez una observación i distinta y se reajusta el modelo con las $n-1$ restantes. En cada proceso de exclusión y reajuste se calcula la diferencia (d_i) entre el valor predicho para i habiendo y sin haber excluido esa observación. Finalmente, se normalizan las diferencias d_i y se detectan aquellas cuyo valor absoluto es mayor que 3. El estudio de *outliers* mediante *studentized residuals* es el más adecuado. En R se pueden calcular con la función `rstudent()`.

Tanto el valor estandarizado como el *studentized* de los residuos sigue una distribución *t-student*, por lo que es posible obtener el *p-value* asociado. Sí se emplean los *p-values* para extraer conclusiones sobre las observaciones, es importante tener en cuenta que se trata de comparaciones múltiples, por lo que es necesario corregir o controlar la inflación del error de tipo I.

El hecho de que un valor sea atípico o con alto grado de *leverage* no implica que sea influyente en el conjunto del modelo. Sin embargo, si un valor es influyente, suele ser o atípico o de alto *leverage*. Existen diferentes formas de evaluar la influencia de las observaciones:

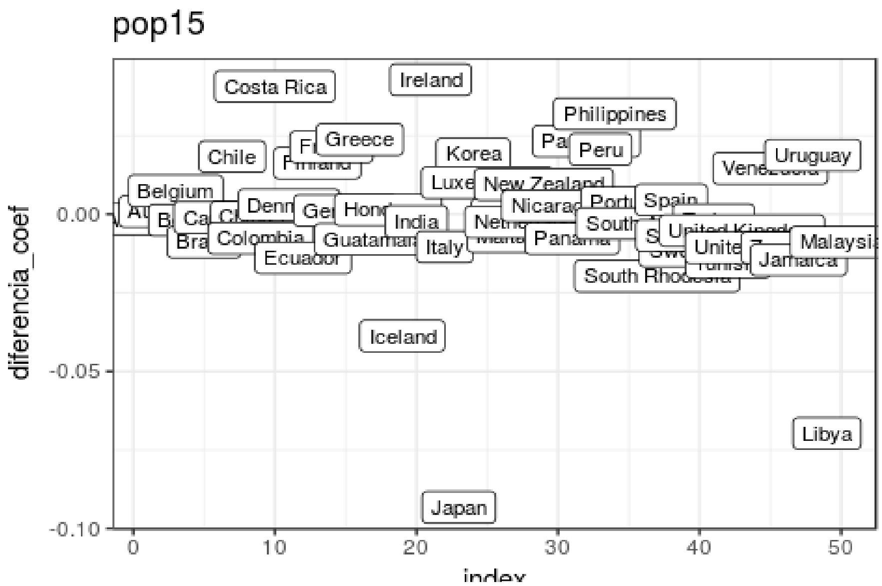
- La distancia de *Cook* es una medida muy utilizada que combina, en un único valor, la magnitud del residuo y el grado de *leverage*. Valores de *Cook* mayores a 1 suelen considerarse como influyentes.

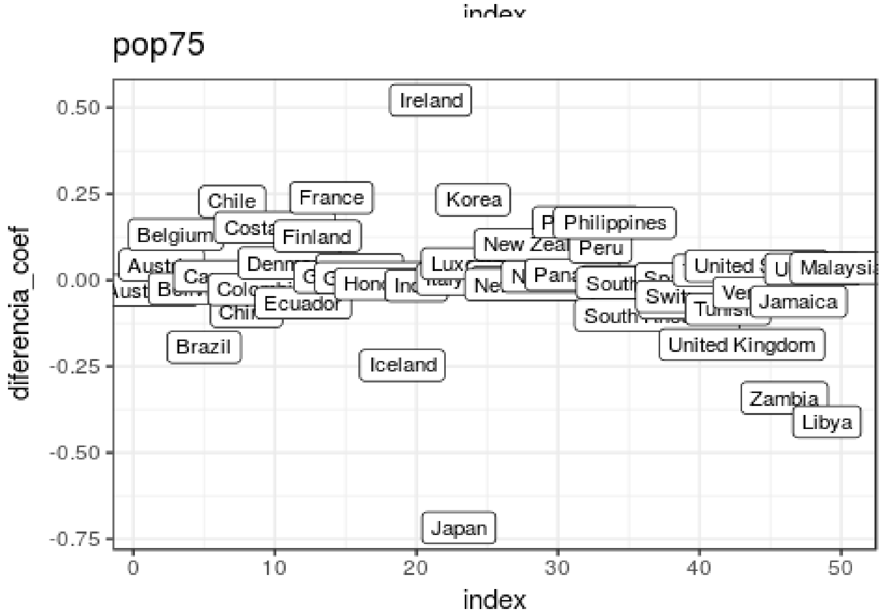
- Evaluar el cambio en los coeficientes de regresión tras excluir la observación: Se trata de un proceso iterativo en el que cada vez se excluye una observación distinta y se reajusta el modelo. En cada iteración se registra la diferencia en los coeficientes de regresión con y sin la observación, dividida entre el SE del predictor en el modelo sin la observación.

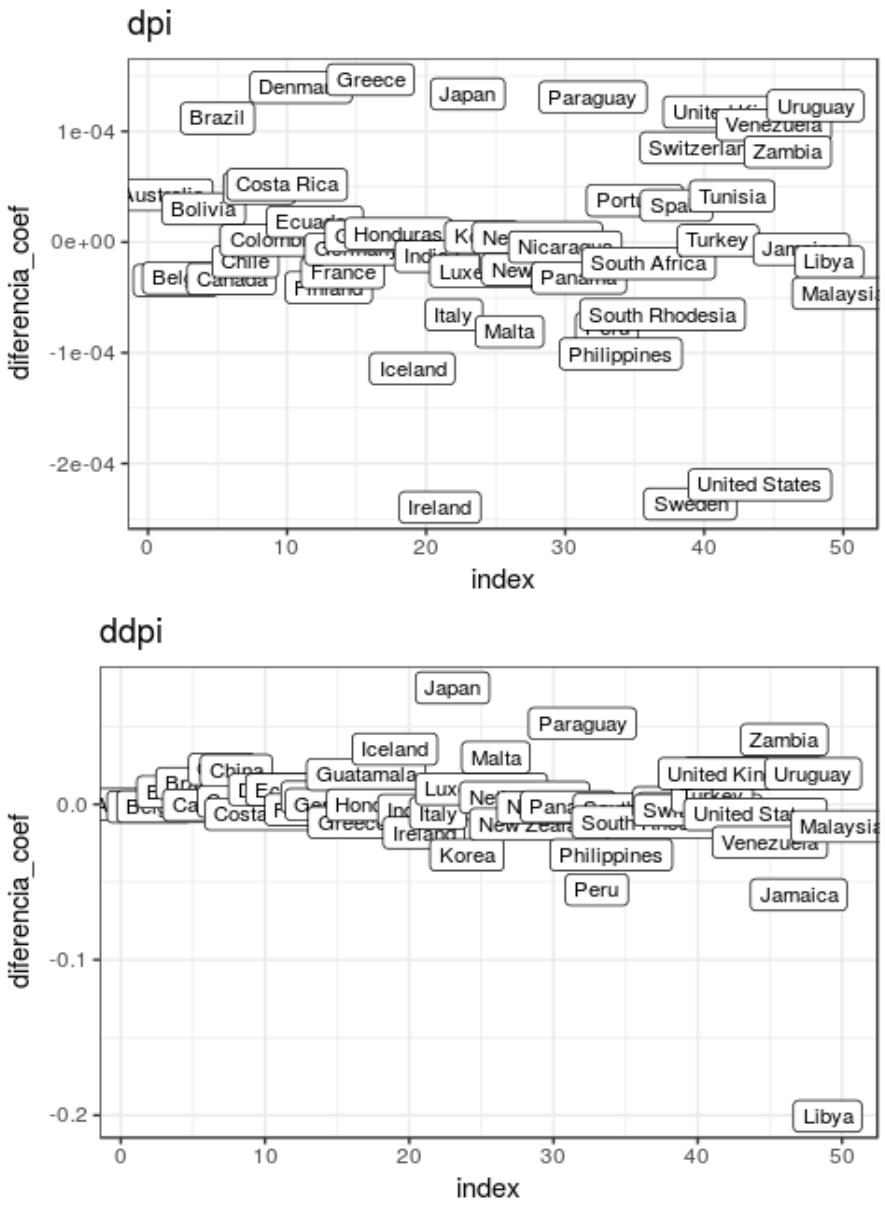
$$Dfbetas_i = \frac{\hat{\beta} - \hat{\beta}_i}{SE_{\hat{\beta}_i}}$$

  Al tratarse de un valor estandarizado, es sencillo identificar que observaciones influyen más y en que magnitud. En la práctica, se considera una observación influyente cuando $|Dfbetas| > 1$ para un pequeño conjunto de datos y $|Dfbetas| > \sqrt{2/n}$ en general. La función `dfbeta()` realiza esta comparación.

```
library(faraway)
library(ggplot2)
data("savings")
savings$pais <- rownames(savings)
modelo_lineal <- lm(sr ~ pop15 + pop75 + dpi + ddpi, savings)
# El siguiente método puede aplicarse a modelos simples y múltiples
# Se evalúa la influencia de cada observación sobre cada predictor
predictores <- names(coef(modelo_lineal))[-1]
for (i in seq_along(predictores)) {
    diferencia_coef <- dfbeta(modelo_lineal)[, predictores[i]]
    diferencia_coef <- data.frame(diferencia_coef = diferencia_coef,
                                  pais = names(diferencia_coef),
                                  index = 1:length(diferencia_coef))
    p<-ggplot(data=diferencia_coef,aes(x=index,y= diferencia_coef)) +
       geom_point() + geom_label(aes(label = pais), size = 3) +
       labs(title = predictores[i]) +
       theme_bw()
    print(p)
}
```


pop15
diferencia_coef
index
0.00
-0.05
-0.10
0
10
20
30
40
50
Costa Rica
Ireland
Philippines
Greece
Chile
Korea
Peru
Uruguay
Belgium
New Zealand
Spain
India
Colombia
Italy
Ecuador
Malaysia
Jamaica
Iceland
Libya
Japan


pop75
diferencia_coef
index
0.50
0.25
0.00
-0.25
-0.50
-0.75
0
10
20
30
40
50
Ireland
Chile
France
Korea
Belgium
Finland
Philippines
Peru
Ecuador
Malaysia
Jamaica
Brazil
Iceland
United Kingdom
Zambia
Libya
Japan



Para este ejemplo, las observaciones Japan y Libya son las más influyentes sobre el predictor *pop15*. El mismo análisis debería hacerse para cada predictor.

Regresión lineal con un predictor categórico de dos niveles y su relación con el t-test

Linear Models with R, by Julian J. Faraway

El set de datos `sexab` del paquete `faraway` contiene los resultados de un estudio en el que se investigó las secuelas que padecen mujeres adultas debido a abusos sufridos durante la infancia. En una clínica médica se midió el nivel de estrés post-traumático (*ptsd*) y nivel de abuso físico sufrido (*cpa*), ambos en escalas estandarizadas, en 45 mujeres que fueron víctimas en su infancia (*csa*). Las mismas mediciones se registraron para 31 mujeres que no sufrieron ningún tipo de abuso.

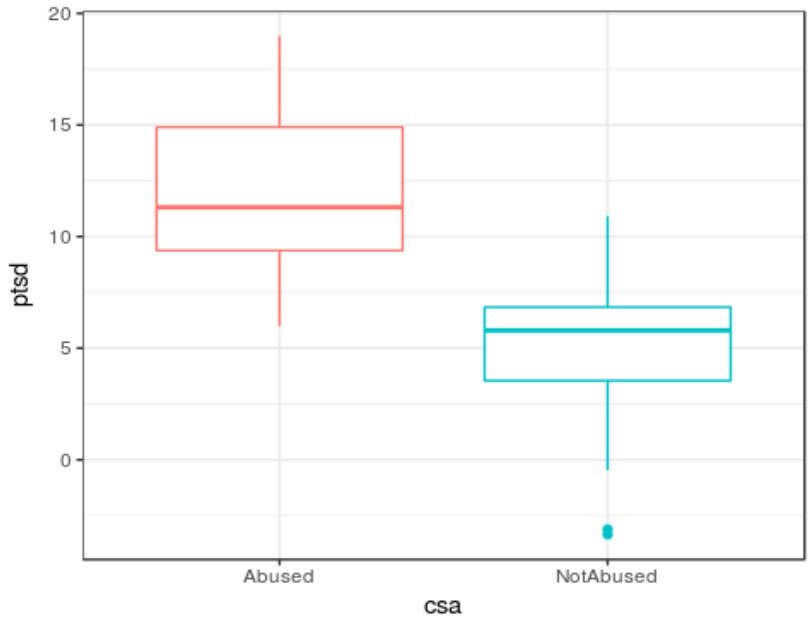
```
library(faraway)
library(ggplot2)
data(sexab)
head(sexab)
```

```
##        cpa     ptsd    csa
## 1  2.04786  9.71365 Abused
## 2  0.83895  6.16933 Abused
## 3 -0.24139 15.15926 Abused
## 4 -1.11461 11.31277 Abused
## 5  2.01468  9.95384 Abused
## 6  6.71131  9.83884 Abused
```

```
by(data = sexab, INDICES = sexab$csa, summary)
```

```
## sexab$csa: Abused
##       cpa              ptsd              csa
##  Min.   :-1.115   Min.   : 5.985   Abused   :45
##  1st Qu.: 1.415   1st Qu.: 9.374   NotAbused: 0
##  Median : 2.627   Median :11.313
##  Mean   : 3.075   Mean   :11.941
##  3rd Qu.: 4.317   3rd Qu.:14.901
##  Max.   : 8.647   Max.   :18.993
## ------------------------------------------------------------
## sexab$csa: NotAbused
##       cpa               ptsd              csa
##  Min.   :-3.1204   Min.   :-3.349   Abused   : 0
##  1st Qu.:-0.2299   1st Qu.: 3.544   NotAbused:31
##  Median : 1.3216   Median : 5.794
##  Mean   : 1.3088   Mean   : 4.696
##  3rd Qu.: 2.8309   3rd Qu.: 6.838
##  Max.   : 5.0497   Max.   :10.914
```

```
ggplot(data = sexab, aes(x = csa, y = ptsd, colour = csa)) + geom_boxplot
() +
theme_bw() + theme(legend.position = "none")
```



Se observa que las víctimas de abusos tienen niveles más altos de estrés post-traumático en comparación a las mujeres que no han sufrido abusos.

Una forma de comparar si está diferencia es significativa, es mediante un *t-test*.

```
# Cálculo de la varianza de cada grupo para determinar si son similares
aggregate(ptsd ~ csa, data = sexab, FUN = var)
```

```
##         csa     ptsd
## 1    Abused 11.83464
## 2 NotAbused 12.38859
```

```
t.test(formula = ptsd ~ csa, data = sexab, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  ptsd by csa
## t = 8.9387, df = 74, p-value = 2.172e-13
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  5.630165 8.860273
## sample estimates:
##    mean in group Abused mean in group NotAbused
##               11.941093                4.695874
```

El contraste de hipótesis muestra una clara significancia en la diferencia de medias (*p-value = 2.172e-13*).

Este mismo contraste puede realizarse desde la perspectiva de un modelo lineal incluyendo la variable cualitativa como predictor. Para hacerlo, cada uno de los niveles del predictor se tiene que convertir en una variable *dummy* cuyo valor puede ser 0 o 1.

$$ptsd = \beta_0 + \beta_1 dummy_{Abused} + \beta_2 dummy_{NotAbused}$$

$$dummy_i = \begin{cases} 0 & \text{la observacion no pertenece al nivel i} \\ 1 & \text{la observacion pertenece al nivel i} \end{cases}$$

Para cada observación, únicamente una de las variables *dummy* toma el valor 1, por ejemplo, si una mujer sí ha sufrido abusos infantiles, el modelo queda:

$$ptsd = \beta_0 + \beta_1 dummy_{Abused}$$

```
sexab$abused <- ifelse(test = sexab$csa == "Abused", yes = 1, no = 0)
sexab$not_abused <- ifelse(test = sexab$csa == "NotAbused", yes = 1, no =
0)
rbind(head(sexab, 3), tail(sexab, 3))
```

```
##         cpa     ptsd       csa abused not_abused
## 1   2.04786  9.71365    Abused      1          0
## 2   0.83895  6.16933    Abused      1          0
## 3  -0.24139 15.15926    Abused      1          0
## 74 -1.85753 -0.46996 NotAbused      0          1
## 75  2.85253  6.84304 NotAbused      0          1
## 76  0.81138  7.12918 NotAbused      0          1
```

Se puede observar que la información de las dos variables *dummy* es redundante, al haber solo dos niveles, y dado que solo uno de ellos puede tomar el valor 1, conociendo uno se conoce el otro. Para evitar que aparezca el problema de la singularidad al ajustar el modelo, una de las dos variables se excluye del modelo y se considera como el nivel de referencia.

```
modelo <- lm(ptsd ~ abused, data = sexab)
summary(modelo)
```

```
##
## Call:
## lm(formula = ptsd ~ abused, data = sexab)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0451 -2.3123  0.0951  2.1645  7.0514
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.6959     0.6237   7.529 1.00e-10 ***
## abused        7.2452     0.8105   8.939 2.17e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.473 on 74 degrees of freedom
## Multiple R-squared:  0.5192, Adjusted R-squared:  0.5127
## F-statistic:  79.9 on 1 and 74 DF,  p-value: 2.172e-13
```

El *p-value* obtenido para el predictor *abused* es el mismo que el obtenido en el *t-test* empleado anteriormente para comparar las medias. El valor estimado de la intersección (4.6959), se corresponde con valor promedio de la variable respuesta en el nivel de referencia. La pendiente estimada (7.2452) se interpreta como el valor promedio de la influencia que tiene el predictor sobre la variable respuesta en comparación al nivel de referencia. En este caso, las mujeres que han sufrido abusos durante su infancia tienen en promedio 7.2452 unidades más de *ptsd* que las que no los han sufrido. Esta cantidad se corresponde con la diferencia de medias de ambos niveles.

```
# media de ptsd en mujeres víctimas de abusos
mean(sexab[sexab$csa == "Abused", "ptsd"])
```

```
## [1] 11.94109
```

```
# media de ptsd en mujeres no víctimas de abusos
mean(sexab[sexab$csa == "NotAbused", "ptsd"])
```

```
## [1] 4.695874
```

```
mean(sexab[sexab$csa == "Abused", "ptsd"]) - mean(sexab[sexab$csa == "Not
Abused",
                                                    "ptsd"])
```

```
## [1] 7.245219
```

Independientemente del nivel que se escoja como referencia, el resultado es el mismo. Lo único que cambia es el valor de la intersección, ya que cambia el nivel de referencia, y el signo de la pendiente.

```
modelo <- lm(ptsd ~ not_abused, data = sexab)
summary(modelo)
```

```
##
## Call:
## lm(formula = ptsd ~ not_abused, data = sexab)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0451 -2.3123  0.0951  2.1645  7.0514
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.9411     0.5177  23.067  < 2e-16 ***
## not_abused   -7.2452     0.8105  -8.939 2.17e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.473 on 74 degrees of freedom
## Multiple R-squared:  0.5192, Adjusted R-squared:  0.5127
## F-statistic:  79.9 on 1 and 74 DF,  p-value: 2.172e-13
```

En R la función `lm()` identifica automáticamente si un predictor es de tipo cualitativo y escoge como nivel de referencia el primero en base al orden alfabético.

```
modelo <- lm(ptsd ~ csa, data = sexab)
summary(modelo)
```

```
##
## Call:
## lm(formula = ptsd ~ csa, data = sexab)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0451 -2.3123  0.0951  2.1645  7.0514
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   11.9411     0.5177  23.067  < 2e-16 ***
## csaNotAbused  -7.2452     0.8105  -8.939 2.17e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 3.473 on 74 degrees of freedom
## Multiple R-squared:  0.5192, Adjusted R-squared:  0.5127
## F-statistic:  79.9 on 1 and 74 DF,  p-value: 2.172e-13
```

Si se desea especificar cuál debe ser el nivel de referencia empleado por `lm()`, se puede recurrir a la función `relevel()`

```
sexab$csa <- relevel(sexab$csa, ref = "NotAbused")
summary(lm(ptsd ~ csa, data = sexab))
```

```
##
## Call:
## lm(formula = ptsd ~ csa, data = sexab)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0451 -2.3123  0.0951  2.1645  7.0514
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.6959     0.6237   7.529 1.00e-10 ***
## csaAbused     7.2452     0.8105   8.939 2.17e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.473 on 74 degrees of freedom
## Multiple R-squared:  0.5192, Adjusted R-squared:  0.5127
## F-statistic:  79.9 on 1 and 74 DF,  p-value: 2.172e-13
```

Es importante tener en cuenta que los *p-values* obtenidos por un *t-test* y por un modelo lineal que contenga un predictor cualitativo con dos niveles serán los mismos siempre y cuando el *t-test* no incluya una corrección de varianzas desiguales.

Representación gráfica de un modelo y su diagnóstico con ggplot2

Si bien la función `plot(lm)` es una forma muy rápida de obtener los gráficos, no son estéticamente muy "bonitos". A continuación se describe como obtener las mismas representaciones mediante el sistema gráfico *ggplot2*. *Información obtenida de https://rpubs.com/therimalaya/43190 y de https://drsimonj.svbtle.com/visualising-residuals.*

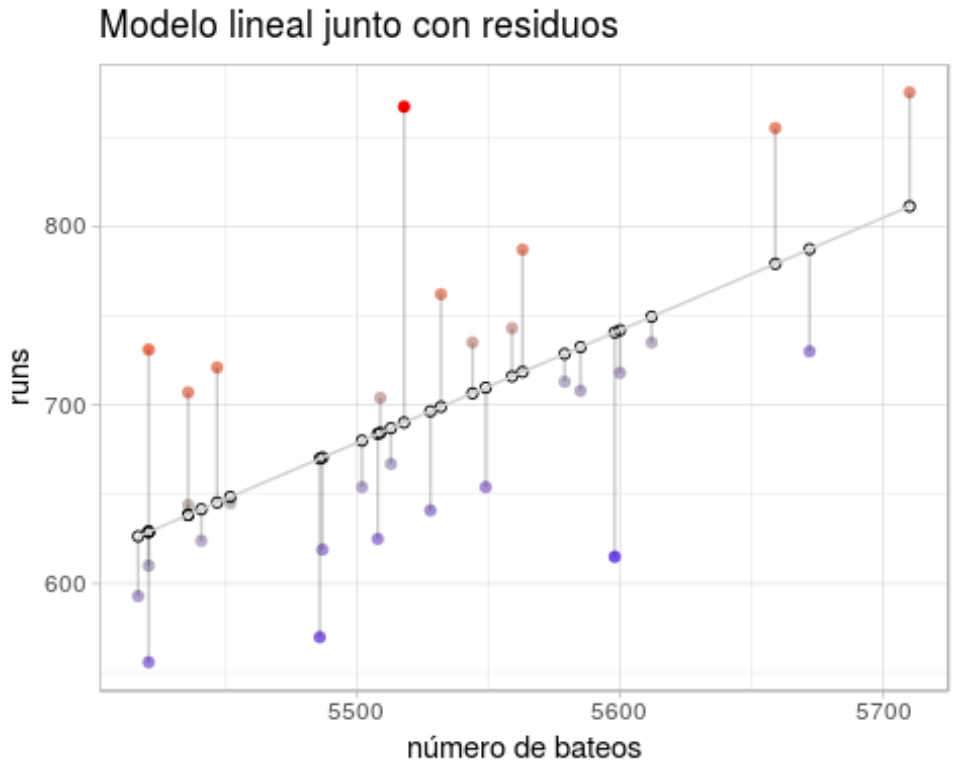
Modelo + residuos

```
equipos <- c("Texas", "Boston", "Detroit", "Kansas", "St.", "New_S.", "New_Y.", "Milwaukee", "Colorado", "Houston", "Baltimore", "Los_An.", "Chicago", "Cincinnati", "Los_P.", "Philadelphia", "Chicago", "Cleveland", "Arizona", "Toronto", "Minnesota", "Florida", "Pittsburgh", "Oakland", "Tampa", "Atlanta", "Washington", "San.F", "San.I", "Seattle")
numero_bateos <- c(5659, 5710, 5563, 5672, 5532, 5600, 5518, 5447, 5544, 5598, 5585, 5436, 5549, 5612, 5513, 5579, 5502, 5509, 5421, 5559, 5487, 5508, 5421, 5452, 5436, 5528, 5441, 5486, 5417, 5421)
runs <- c(855, 875, 787, 730, 762, 718, 867, 721, 735, 615, 708, 644, 654, 735, 667, 713, 654, 704, 731, 743, 619, 625, 610, 645, 707, 641, 624, 570, 593, 556)
datos <- data.frame(equipos, numero_bateos, runs)

# ajuste del modelo lineal simple
modelo <- lm(formula = runs ~ numero_bateos, data = datos)

# cálculo de los valores predichos y de los residuos
datos$prediccion <- predict(modelo)
datos$residuos <- residuals(modelo)
head(datos, 4)
```

```
##   equipos numero_bateos runs prediccion  residuos
## 1   Texas          5659  855   779.0395  75.96048
## 2  Boston          5710  875   811.1976  63.80243
## 3 Detroit          5563  787   718.5067  68.49328
## 4  Kansas          5672  730   787.2367 -57.23667
```

```
# representar las observaciones el color de los puntos se asocia al tamaño  del residuo
p <- ggplot(data = datos, aes(x = numero_bateos, y = runs))
p <- p + geom_point(aes(color = residuos))
p <- p + scale_color_gradient2(low = "blue", mid = "grey", high = "red")
# añadir los valores acorde al modelo generado representandolos con otro
# tipo de puntos
p <- p + geom_point(aes(x = numero_bateos, y = prediccion), shape = 1)
```

```
# añadir segmentos que unan cada observación con su residuo
p <- p + geom_segment(aes(xend = numero_bateos, yend = prediccion),
                          alpha = 0.2)

# añadir recta de mínimos cuadrados
p <- p + geom_smooth(method = "lm", se = FALSE, colour = "lightgrey",
                     size = 0.5)

p <- p + labs(title = "Modelo lineal junto con residuos",
              x = "número de bateos")
p <- p + theme_light() + guides(color = FALSE)
p
```



Análisis gráfico de residuos

La combinación de los paquetes `broom` y `ggplot2` (ambos de Hadley Wickham) permiten obtener representaciones gráficas de modelos estadísticos. El paquete `broom` permite convertir objetos de análisis estadísticos tales como `lm`, `t.test`, `anova`... en tablas ordenadas *Tidy Data Frames*.

Una vez se ha obtenido el *data frame* a partir de un objeto estadístico, es muy sencillo generar representaciones gráficas.

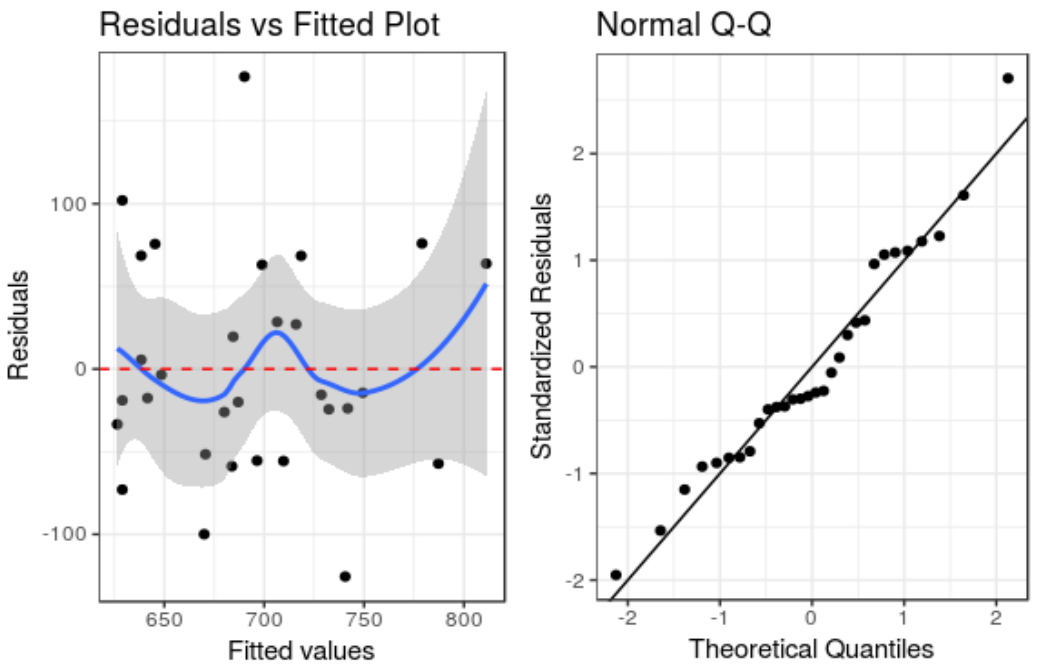
```
library(broom)
broom_modelo <- augment(modelo)
head(broom_modelo, n = 3)
```

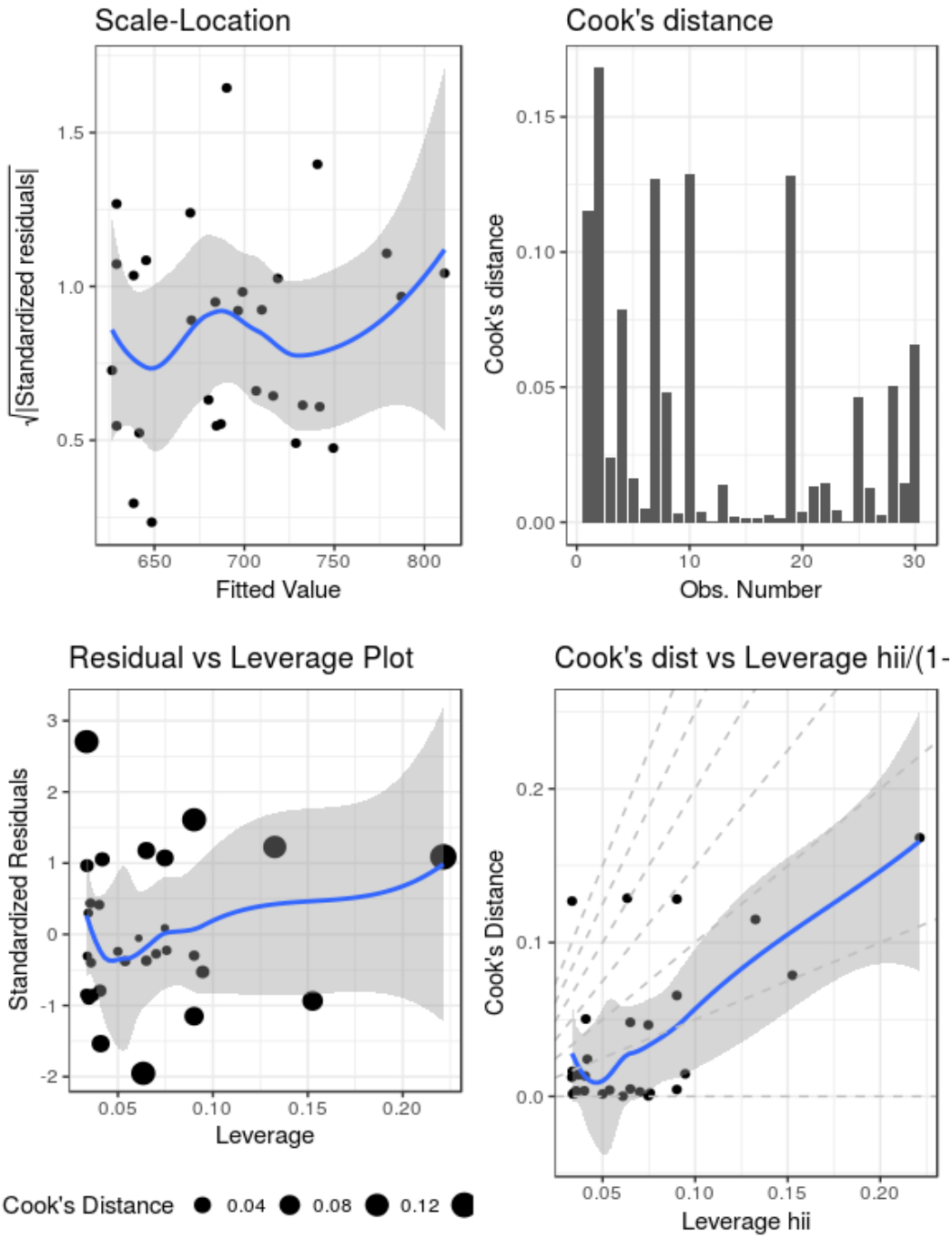
```
##   runs numero_bateos  .fitted  .se.fit   .resid       .hat   .sigma
## 1  855          5659 779.0395 24.20303 75.96048 0.13257176 65.84776
## 2  875          5710 811.1976 31.27290 63.80243 0.22133381 66.24702
## 3  787          5563 718.5067 13.58497 68.49328 0.04176659 66.33977
##      .cooksd .std.resid
## 1 0.11503787   1.226949
## 2 0.16815163   1.087721
## 3 0.02414704   1.052611
```

Se genera un *data frame* que contiene los datos originales más los datos que suelen emplearse de un modelo. Cuando se pasa como argumento a la función `ggplot()` un objeto estadístico tal como *lm* este proceso tiene lugar automáticamente, permitiendo acceder a las nuevas variables (.fitted, .se.fit, .resid, .hat, .sigma, .cooksd .std.resid).

```
diagnostico_residuos <- function(modelo) {
    library(gridExtra)
    p1 <- ggplot(data = modelo, aes(.fitted, .resid)) + geom_point()
    p1 <- p1 + stat_smooth(method = "loess") +
               geom_hline(yintercept = 0, col="red",linetype = "dashed")
    p1 <- p1 + xlab("Fitted values") + ylab("Residuals")
    p1 <- p1 + ggtitle("Residuals vs Fitted Plot") + theme_bw()

    p2 <- ggplot(modelo, aes(qqnorm(.stdresid, plot.it = FALSE)[[1]],
                                    .stdresid)) +
        geom_point(na.rm = TRUE)
    p2 <- p2 + geom_abline() + xlab("Theoretical Quantiles") +
             ylab("Standardized Residuals")
    p2 <- p2 + ggtitle("Normal Q-Q") + theme_bw()

    p3 <- ggplot(modelo, aes(.fitted, sqrt(abs(.stdresid)))) +
          geom_point(na.rm = TRUE)
    p3 <- p3 + stat_smooth(method = "loess", na.rm = TRUE) +
               xlab("Fitted Value")
    p3 <- p3 + ylab(expression(sqrt("|Standardized residuals|")))
    p3 <- p3 + ggtitle("Scale-Location") + theme_bw()

    p4 <- ggplot(modelo, aes(seq_along(.cooksd), .cooksd)) +
          geom_bar(stat = "identity", position = "identity")
    p4 <- p4 + xlab("Obs. Number") + ylab("Cook's distance")
    p4 <- p4 + ggtitle("Cook's distance") + theme_bw()
```

```
    p5 <- ggplot(modelo, aes(.hat, .stdresid)) +
          geom_point(aes(size = .cooksd), na.rm = TRUE)
    p5 <- p5 + stat_smooth(method = "loess", na.rm = TRUE)
    p5 <- p5 + xlab("Leverage") + ylab("Standardized Residuals")
    p5 <- p5 + ggtitle("Residual vs Leverage Plot")
    p5 <- p5 + scale_size_continuous("Cook's Distance", range = c(1, 5))
    p5 <- p5 + theme_bw() + theme(legend.position = "bottom")

    p6 <- ggplot(modelo, aes(.hat, .cooksd)) + geom_point(na.rm = TRUE) +
          stat_smooth(method = "loess", na.rm = TRUE)
    p6 <- p6 + xlab("Leverage hii") + ylab("Cook's Distance")
    p6 <- p6 + ggtitle("Cook's dist vs Leverage hii/(1-hii)")
    p6 <- p6 + geom_abline(slope = seq(0, 3, 0.5), color = "gray",
                           linetype = "dashed")
    p6 <- p6 + theme_bw()

    grid.arrange(p1, p2, ncol = 2)
    grid.arrange(p3, p4, ncol = 2)
    grid.arrange(p5, p6, ncol = 2)
}

diagnostico_residuos(modelo = modelo)
```




Scale-Location
√|Standardized residuals|
Fitted Value
Cook's distance
Cook's distance
Obs. Number
Residual vs Leverage Plot
Standardized Residuals
Leverage
Cook's Distance
0.04
0.08
0.12
Cook's dist vs Leverage hii/(1-
Cook's Distance
Leverage hii

Estimación parámetros de un modelo de regresión lineal mediante métodos de optimización convexa

Considérese el modelo de regresión lineal simple:

$$Y = \beta_0 + \beta_1 X_1 + \epsilon$$

Dada una muestra de entrenamiento, los parámetros estimados óptimos $(\hat{\beta}_0, \hat{\beta}_1)$ se obtienen minimizando la suma de cuadrados residuales:

$$RSS(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

Si bien, para este problema en cuestión, existe una solución explicita con la que obtener los valores que minimizan la función *RSS*:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n} (x_i - \overline{x})^2}$$

$$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x}$$

en muchos otros no existe tal posibilidad, haciendo necesaria la utilización de algoritmos de optimización. Dada la simplicidad del problema de regresión lineal simple, es un buen ejemplo para ilustrar cómo se solucionan problemas de optimización. Además, la función a minimizar es una *norma*, por lo que se trata de una función convexa, tiene un único mínimo que es el mínimo global.

Los algoritmos de optimización siguen un proceso iterativo en el que, partiendo de una posición inicial dentro del dominio de la función, realizan desplazamientos en la dirección correcta hasta alcanzar el mínimo. Para poder llevar a cabo el proceso se necesita conocer:

- La función objetivo a minimizar.
- La posición desde la que iniciar la búsqueda.
- El step: la distancia que se va a desplazar el algoritmo en cada iteración de búsqueda.
- La dirección de búsqueda: dirección en la que se produce el desplazamiento de búsqueda.
- La tolerancia o *learning rate*: Al tratarse de un algoritmo iterativo, hay que indicar una regla de parada. Lo idóneo es que el algoritmo se detenga al encontrar el mínimo pero, como normalmente se desconoce cuál es, hay que

conseguir que se pare lo más cerca posible. Una forma de cuantificar la proximidad al mínimo es controlar cuanto desciende el valor de la función entre iteraciones consecutivas. Si el descenso es muy pequeño, significa que se encuentra muy cerca del valor. La tolerancia se define como un valor tal que, si la diferencia en el descenso es menor o igual, se considera que se ha alcanzado el mínimo (el algoritmo converge). Una alternativa a calcular la distancia entre las dos iteraciones es medir la longitud del vector gradiente en cada iteración. Si su longitud es muy pequeña, también lo es la distancia al mínimo.

- Iteraciones máximas: Si la posición de inicio en la búsqueda está muy alejada del mínimo de la función y el *step* es muy pequeño, se pueden necesitar muchas iteraciones para alcanzarlo. Para evitar un proceso iterativo excesivamente largo, se suele establecer un número máximo de iteraciones permitidas.

Optimización por el método de descenso de gradiente

El algoritmo de descenso de gradiente se caracteriza por que emplea como dirección de búsqueda aquella en la que la función desciende en mayor medida. Esta dirección se corresponde con *-1 x gradiente de la función*. El gradiente de la función es el vector formado por las derivadas parciales de la función respecto a cada variable.

La función objetivo a optimizar es la suma de los residuos cuadrados:

$$J(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

Para facilitar posteriores simplificaciones, se multiplica la función por el factor $\frac{1}{2}$.

$$J(\beta_0, \beta_1) = \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

Tal y como se ha mencionado previamente, el método de gradiente se caracteriza porque la dirección de búsqueda viene dada por el gradiente de la función, que es el vector formado por las derivadas parciales de la función respecto a cada una de las variables, en este caso.

$$\frac{dJ}{d(\beta_0)} = 2 * \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) * -1 = -\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)$$

$$\frac{dJ}{d(\beta_1)} = 2 * \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) * -x_i = -\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)x_i$$

El vector gradiente para un modelo lineal simple es por lo tanto:

$$\begin{bmatrix} -\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) \\ -\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)x_i \end{bmatrix}$$

Algoritmo: Método de descenso de gradiente para $J(\beta_0, \beta_1)$

1. Seleccionar valores iniciales $\hat{\beta}_0, \hat{\beta}_1, t > 0$
2. Iniciar un proceso iterativo en el que
   - $\hat{\beta}_0 = \hat{\beta}_0 - t * -\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)$
   - $\hat{\beta}_1 = \hat{\beta}_1 - t * -\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)x_i$

hasta alcanzar una condición de parada.

En primer lugar se simula una muestra a partir de una función conocida en la que el usuario determina los parámetros poblacionales. Como los parámetros poblacionales se conocen, se puede evaluar cómo de buenas son las estimaciones obtenidas.

```
n <- 100      # Tamaño de la muestra
set.seed(123) # Para permitir reproducibilidad
x <- runif(n, min = 0, max = 5) # n puntos aleatorios en el intervalo [mi
n,max]
beta0 <- 2    # Parámetro beta0 del modelo
beta1 <- 5    # Parámetro beta1 del modelo
set.seed(123) # Para permitir reproducibilidad
epsilon <- rnorm(n, sd = 1) # error (con desviación típica sd=1)
y <- beta0 + beta1 * x + epsilon # calculo de cada y_i
datos <- data.frame(x, y)
head(datos)
```

```
##           x         y
## 1 1.4378876  8.628962
## 2 3.9415257 21.477451
## 3 2.0448846 13.783131
## 4 4.4150870 24.145943
## 5 4.7023364 25.640970
## 6 0.2277825  4.853977
```

Véase los valores estimados por la función `lm()` que aplica la solución explicita.

```
modelo_lm <- lm(y ~ x , data = datos)
coefficients(modelo_lm)
```

```
## (Intercept)           x
##    2.117657    4.989068
```

Como era de esperar, las estimaciones de los coeficientes de regresión obtenidos mediante `lm()` se aproximan mucho a los parámetros poblacionales.

Ajuste del modelo por optimización de la suma de residuos cuadrados

```
# FUNCIÓN OBJETIVO A MINIMIZAR
# ================================================================================

sum_residuos <- function(x, y, beta_0, beta_1){
  return(sum((y - (beta_0 + beta_1 * x))^2))
}

# CÁLCULO DE GRADIENTE PARA MODELO LINEAL SIMPLE
# ================================================================================
calc_gradiente <- function(beta_0, beta_1, x, y){
  # beta_0: valor del intersección
  # beta_1: coeficiente de regresión del predictor
  # x: valores del predictor observados en la muestra
  # y: valores de la variable respuesta observados en la muestra
  grad_1 <- sum(y - beta_0 - beta_1 * x)
  grad_2 <- sum((y - beta_0 - beta_1 * x) * x)
  return(c(grad_1, grad_2))
}
```

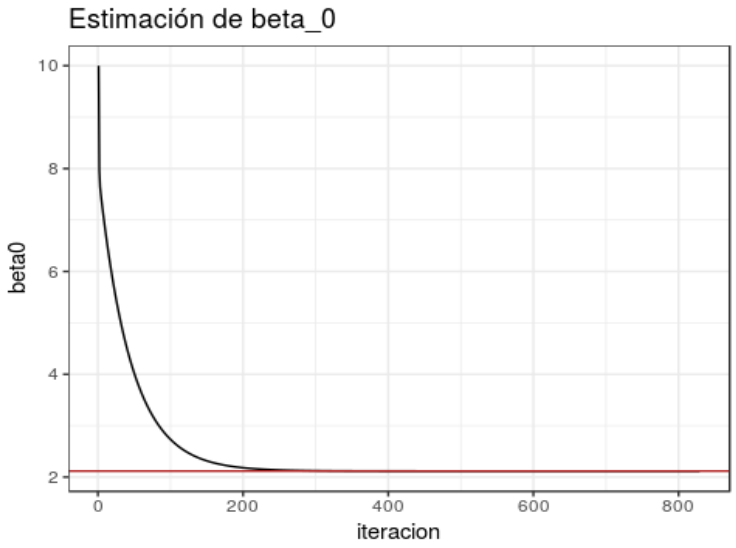
```
# OPTIMIZACIÓN
# ================================================================================
======
optimizacion_grad <- function(beta_0 = 0, beta_1 = 0, x, y, t = 0.001,
                              max_iter = 10000, tolerancia = 1e-10){

  # Matriz para almacenar el valor de las estimaciones en cada iteración
  estimaciones <- matrix(NA, nrow = max_iter, ncol = 4)
  colnames(estimaciones) <- c("iteracion", "beta0", "beta1", "residuos")

  for(i in 1:max_iter){

    gradiente <- calc_gradiente(beta_0 = beta_0, beta_1 = beta_1, x = x,
y = y)

    if(sqrt(sum(gradiente^2)) < tolerancia){
      # Si el tamaño del vector es menor que la tolerancia,
      # se considera que el proceso a llegado a convergencia.
      message("El algoritmo ha alcanzado convergencia")
      break
    }else{
    estimaciones[i, 1] <- i
    estimaciones[i, 2] <- beta_0
    estimaciones[i, 3] <- beta_1
    estimaciones[i, 4] <- sum_residuos(x, y, beta_0, beta_1)
    beta_0 <- beta_0 + t * gradiente[1]
    beta_1 <- beta_1 + t * gradiente[2]
    }
  }
  print(paste("Estimación de beta_0:", beta_0))
  print(paste("Estimación de beta_1;", beta_1))
  print(paste("Número de iteraciones:", i))
  print(paste("Suma de residuos cuadrados:", estimaciones[i-1, 4]))

  return(list(beta_0 = beta_0,
              beta_1 = beta_1,
              iteraciones = i,
              estimaciones = as.data.frame(na.omit(estimaciones))
              )
  )
}

resultados <- optimizacion_grad(beta_0 = 10, beta_1 = 10, x = datos$x,
                                y = datos$y, t = 0.001, max_iter = 10000,
                                tolerancia = 1e-6)
```

```
## El algoritmo ha alcanzado convergencia

## [1] "Estimación de beta_0: 2.11765688165228"
## [1] "Estimación de beta_1; 4.98906810875534"
## [1] "Número de iteraciones: 830"
## [1] "Suma de residuos cuadrados: 82.4660264805963"
```

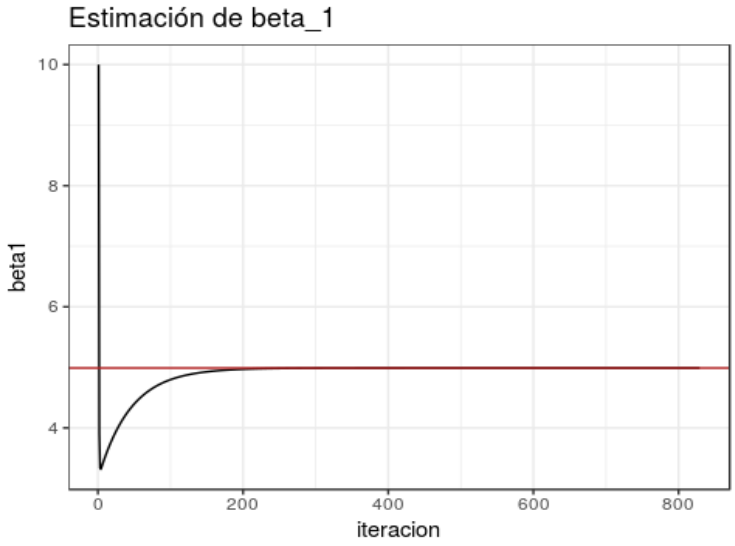
Empleando un valor $t = 0.001$, los valores estimados mediante optimización por descenso de gradiente son casi idénticos a los obtenidos mediante la función `lm()`.

Los siguientes gráficos muestran la evolución de los parámetros estimados y el error tras cada iteración.

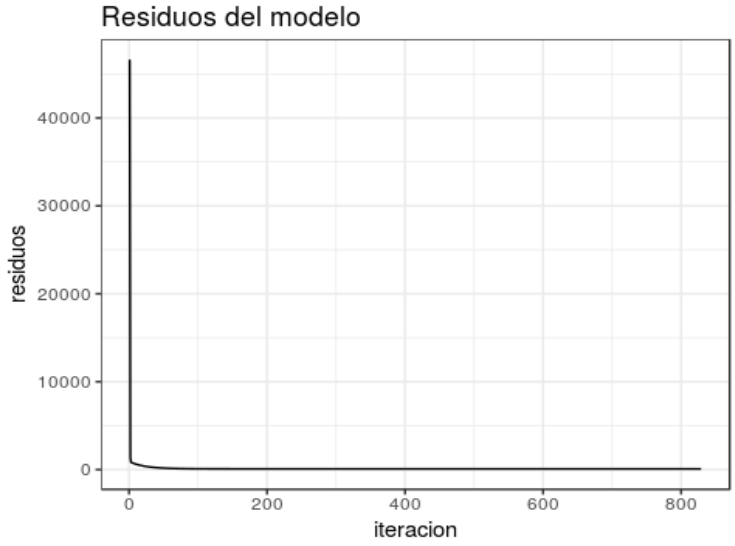
```
library(ggplot2)

ggplot(data = resultados$estimaciones,
       aes(x = iteracion, y = beta0)) +
geom_path() +
geom_hline(yintercept = modelo_lm$coefficients[1], color = "firebrick") +
theme_bw() +
ggtitle("Estimación de beta_0")
```




```
ggplot(data = resultados$estimaciones,
       aes(x = iteracion, y = beta1)) +
geom_path() +
geom_hline(yintercept = modelo_lm$coefficients[2], color = "firebrick") +
theme_bw() +
ggtitle("Estimación de beta_1")
```

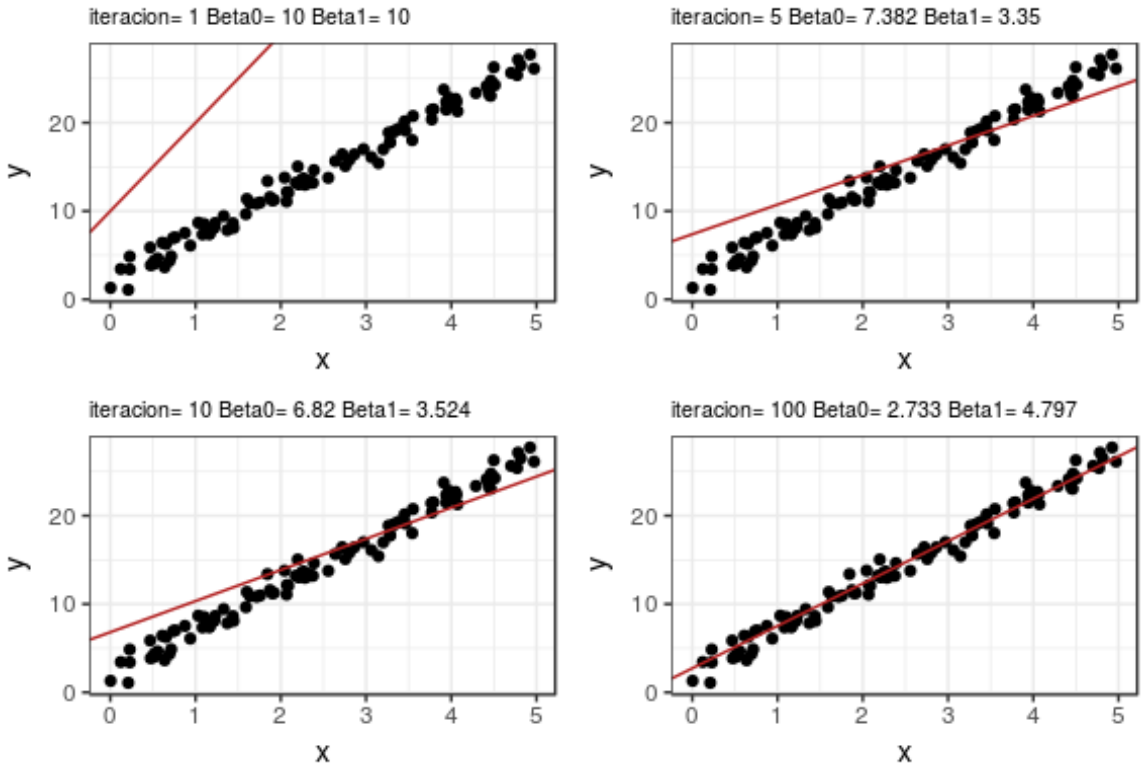



```
ggplot(data = resultados$estimaciones,
       aes(x = iteracion, y = residuos)) +
geom_path() +
theme_bw() +
ggtitle("Residuos del modelo")
```




```
for(i in c(1,5,10,100)) {
  beta_0 <- round(resultados$estimaciones$beta0[i],3)
  beta_1 <- round(resultados$estimaciones$beta1[i],3)
  p  <- ggplot(data = datos, aes(x = x, y = y)) +
        geom_point() +
        geom_abline(intercept = beta_0, slope = beta_1,
        color = "firebrick") +
        ggtitle(paste("iteracion=",i,"Beta0=",beta_0, "Beta1=",beta_1)) +
        theme_bw() +
```

```
        theme(plot.title = element_text(size = 8))
    print(p)
}
```



Este método está muy influenciado por el tamaño de *t*. Para valores grandes, la suma de cuadrados residuales puede ser un valor demasiado grande para el software empleado. Por ejemplo, este mismo código ejecutado para una muestra de tamaño *n=1000* da error. Las simulaciones que he realizado parecen indicar que empleando un valor de *t* aproximadamente de *1/(10n)* se adapta bastante bien.

```
n <- 10000
set.seed(123)
x <- runif(n, min = 0, max = 5)
beta0 <- 2
beta1 <- 5
set.seed(123)
epsilon <- rnorm(n, sd = 1)
y <- beta0 + beta1 * x + epsilon
datos2 <- data.frame(x, y)

resultados <- optimizacion_grad(beta_0 = 10, beta_1 = 10, x = datos2$x,
                                y = datos2$y, t = 1/(10*length(x)),
                                max_iter = 10000, tolerancia = 1e-6)
```

```
## El algoritmo ha alcanzado convergencia

## [1] "Estimación de beta_0: 1.99327105964167"
## [1] "Estimación de beta_1; 5.0017514798928"
## [1] "Número de iteraciones: 1015"
## [1] "Suma de residuos cuadrados: 9971.6909756411"
```

Ajuste del modelo por optimización de la media de residuos cuadrados

En lugar de minimizar la suma de cuadrados residuales, se puede minimizar la media de los residuos cuadrados.

```
# FUNCIÓN OBJETIVO A MINIMIZAR
# ================================================================================
======

mean_residuos <- function(x, y, beta_0, beta_1){
  return((1/length(x)) * sum((y - (beta_0 + beta_1 * x))^2))
}

# CÁLCULO DE GRADIENTE PARA MODELO LINEAL SIMPLE
# ================================================================================
======
calc_gradiente <- function(beta_0, beta_1, x, y){
  # beta_0: valor del intersección
  # beta_1: coeficiente de regresión del predictor
  # x: valores del predictor observados en la muestra
  # y: valores de la variable respuesta observados en la muestra
  grad_1 <- (1/length(x)) * sum(y - beta_0 - beta_1 * x)
  grad_2 <- (1/length(x)) * sum((y - beta_0 - beta_1 * x) * x)
  return(c(grad_1, grad_2))
}




# OPTIMIZACIÓN
# ================================================================================
======
optimizacion_grad <- function(beta_0 = 0, beta_1 = 0, x, y, t = 0.001,
                              max_iter = 10000, tolerancia = 1e-10){

  # Matriz para almacenar el valor de las estimaciones en cada iteración
  estimaciones <- matrix(NA, nrow = max_iter, ncol = 4)
  colnames(estimaciones) <- c("iteracion", "beta0", "beta1", "residuos")
```

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

    gradiente <- calc_gradiente(beta_0 = beta_0, beta_1 = beta_1, x = x,
                                y = y)

    if(sqrt(sum(gradiente^2)) < tolerancia){
      # Si el tamaño del vector es menor que la tolerancia,
      # se considera que el proceso a llegado a convergencia.
      message("El algoritmo ha alcanzado convergencia")
      break
    }else{
    estimaciones[i, 1] <- i
    estimaciones[i, 2] <- beta_0
    estimaciones[i, 3] <- beta_1
    estimaciones[i, 4] <- mean_residuos(x, y, beta_0, beta_1)
    beta_0 <- beta_0 + t * gradiente[1]
    beta_1 <- beta_1 + t * gradiente[2]
    }
  }
  print(paste("Estimación de beta_0:", beta_0))
  print(paste("Estimación de beta_1;", beta_1))
  print(paste("Número de iteraciones:", i))
  print(paste("Media residuos cuadrados:", estimaciones[i-1, 4]))

  return(list(beta_0 = beta_0,
              beta_1 = beta_1,
              iteraciones = i,
              estimaciones = as.data.frame(na.omit(estimaciones))
              )
  )
}

resultados <- optimizacion_grad(beta_0 = 10, beta_1 = 10, x = datos$x,
                                y = datos$y, t = 0.1, max_iter = 10000,
                                tolerancia = 1e-6)
```

```
## El algoritmo ha alcanzado convergencia
```

```
## [1] "Estimación de beta_0: 2.11766110184018"
## [1] "Estimación de beta_1; 4.98906679390207"
## [1] "Número de iteraciones: 626"
## [1] "Media residuos cuadrados: 0.824660264810621"
```

Empleando como función objetivo la media de los residuos cuadrados, el valor de *t* puede ser mayor que cuando se utiliza la suma de los residuos cuadrados (partiendo del mismo punto inicial).

Método de descenso de gradiente estocástico (Stochastic Gradient Descent)

En el método de descenso de gradiente, para cada actualización del valor de los parámetros estimados, se necesita la muestra de entrenamiento al completo. Esto supone un alto coste computacional cuando la muestra contiene muchas observaciones. Existe una alternativa al método de descenso de gradiente que también proporciona buenos resultados. El algoritmo para el caso particular del modelo de regresión lineal simple es el siguiente:

Algoritmo: Método de descenso de gradiente estocástico para $J(\beta_0, \beta_1)$

1. Seleccionar valores iniciales $\hat{\beta}_0, \hat{\beta}_1, t > 0$
2. Iniciar un proceso iterativo en el que para $i = 1, \ldots, n$
   - $\hat{\beta}_0 = \hat{\beta}_0 - t * -(y_i - \beta_0 - \beta_1 x_i)$
   - $\hat{\beta}_1 = \hat{\beta}_1 - t * -x_i(y_i - \beta_0 - \beta_1 x_i)$

hasta alcanzar una condición de parada.

Con este método se actualiza el valor de los parámetros con cada observación de la muestra de entrenamiento, mientras que el método de descenso de gradiente (batch) necesita leer todos los datos para hacer una única iteración.

```
opt_grad_estoc <- function(beta_0 = 0, beta_1 =0, x, y, t = 0.01,
                           tolerancia = 1e-10){

  # Matriz para almacenar el valor de las estimaciones en cada iteración
  n <- length(x)
  estimaciones <- matrix(NA, nrow = n, ncol = 3)
  colnames(estimaciones) <- c("iteracion", "beta0", "beta1")

  for(i in 1:n){
    estimaciones[i, 1] <- i
    estimaciones[i, 2] <- beta_0
    estimaciones[i, 3] <- beta_1

    beta_0 <- beta_0 + t * (y[i] - beta_0 - beta_1 * x[i])
    beta_1 <- beta_1 + t *  x[i] * (y[i] - beta_0 - beta_1 * x[i])
```

```
    if( i > 1){
      diferencia_beta0 <- estimaciones[i, 2] - estimaciones[i-1 ,2]
      diferencia_beta1 <- estimaciones[i, 3] - estimaciones[i-1 ,3]
      distancia <- sqrt(diferencia_beta0^2 + diferencia_beta1^2)
      if(distancia < tolerancia){
         print(distancia)
         # Si la distancia es menor que la tolerancia,
         # se considera que el proceso a llegado a convergencia.
         message("El algoritmo ha alcanzado convergencia")
         break
      }
    }
  }
  print(paste("Estimación de beta_0:", beta_0))
  print(paste("Estimación de beta_1;", beta_1))
  print(paste("Número de iteraciones:", i))

  return(list(beta_0 = beta_0,
              beta_1 = beta_1,
              iteraciones = i,
              estimaciones = na.omit(estimaciones)
              )
  )
}

resultados <- opt_grad_estoc(beta_0 = 10, beta_1 = 10, x = datos$x,
                             y = datos$y, t = 0.1,
                             tolerancia = 1e-10)
```

```
## [1] "Estimación de beta_0: 2.59596356190074"
## [1] "Estimación de beta_1; 4.60336260310393"
## [1] "Número de iteraciones: 100"
```

Bibliografía

OpenIntro Statistics: Third Edition, David M Diez, Christopher D Barr, Mine Çetinkaya-Rundel

An Introduction to Statistical Learning: with Applications in R (Springer Texts in Statistics)

Linear Models with R, Julian J.Faraway

An introduction to Logistic Regresion Analysis and Reporting. Chao-Ying Joanne Peng

http://www.ats.ucla.edu/stat/r/dae/logit.htm

http://ww2.coastal.edu/kingw/statistics/R-tutorials/index.html

R Tutorials by William B. King, Ph.D http://ww2.coastal.edu/kingw/statistics/R-tutorials/

Points of Significance: Association, correlation and causation. Naomi Altman & Martin Krzywinski Nature Methods

Points of Significance: Simple linear regression Naomi Altman & Martin Krzywinski. Nature Methods

Resampling Data: Using a Statistical Jackknife S. Sawyer | Washington University | March 11, 2005

http://www.biostat.jhsph.edu/~bcaffo/651/files/lecture12.pdf

https://en.wikipedia.org/wiki/Resampling_(statistics)#Jackknife

The Trusty Jackknife Method identifies outliers and bias in statistical estimates by I. Elaine Allen and Christopher A. Seaman