Lección 27

Regresión Lineal

En esta lección, explicamos el uso de `R` para llevar a cabo el modelo de la regresión lineal, tanto simple como múltiple, explicado en el curso. Se presentan algunos ejemplos para ilustrar el uso de las funciones de `R` específicas para este modelo, así como la posterior validación y adecuación del modelo mediante el análisis de los residuos. Hay que tener en cuenta que en una de las primeras lecciones, ya se introdujo la regresión lineal sin entrar en mucha profundidad en el tema y es en esta lección, donde se desarrollará más ámpliamente el tema.

27.1. El modelo de regresión lineal en R

Uno de los problemas más recurrentes en el campo de la estadística es determinar a partir de un conjunto de observaciones de variables si existe alguna relación funcional entre una de las variables, llamada variable dependiente o de respuesta, y el resto de variables, conocidas como variables independientes o de control. Esta relación funcional puede permitir predecir de forma aproximada el valor de la variable respuesta una vez conocidos los valores de las variables independientes. En esta lección, nos centramos en el caso que la relación funcional sea lineal.

Formalmente, la situación anterior se describe como sigue. Sean $X_1, \ldots, X_k$ k variables (no necesariamente aleatorias) que representarán las variables independientes e Y la variable dependiente. Se dispone de un conjunto de datos

$$(x_{i1}, x_{i2}, \ldots, x_{ik}, y)_{i=1,\ldots,n}$$

donde se recogen los datos de las variables $X_1, \ldots, X_k$, Y para n individuos. Entonces el objetivo es encontrar el "mejor" modelo lineal que explique Y en función de $X_1, \ldots, X_k$, es decir, encontrar los "mejores" valores de $\beta_0, \beta_1, \ldots, \beta_k$ según algún criterio preestablecido en la ecuación

$$\mu_{Y|_{x_1 \ldots x_k}} = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k$$

donde $\mu_{Y|_{x_1 \ldots x_k}}$ es el valor esperado de Y cuando $X_i = x_i$ para todo $i = 1, \ldots, k$. A partir de la muestra, esta relación deriva en encontrar la recta de regresión

$$\hat{y}_i = b_0 + b_1 x_{i1} + \ldots + b_k x_{ik}$$

donde $b_0, b_1, \ldots, b_k$ son estimaciones de β_0, β_1, $\ldots$, β_k; $\hat{y}_i$ son los valores estimados de y_i según el modelo. En general, el modelo no es perfecto y por lo tanto, $\hat{y}_i \neq y_i$ habiendo de considerar un error $e_i = y_i - \hat{y}_i$ en el modelo dando lugar a

$$y_i = b_0 + b_1 x_{i1} + \ldots + b_k x_{ik} + e_i.$$

El criterio más utilizado para estimar los coeficientes del modelo es el *método de mínimos cuadradados* donde se minimiza el error cuadrático

$$SS_\epsilon = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_1 - \ldots - \beta_k x_k)^2.$$

Por esta razón, se conoce como *regresión lineal por mínimos cuadrados* a este modelo. Además, hablaremos de *regresión lineal simple* cuando $k = 1$ (una única variable independiente) y de *regresión lineal múltiple* cuando $k > 1$ (más de una variable independiente).

La función básica de `R` para realizar la regresión lineal en `R` es `lm`. Su sintaxis genérica es

```
lm(formula,data=..., subset=...)
```

con los argumentos siguientes:

- `formula`: Este argumento describe la variable respuesta y las variables independientes del modelo. La estructura es la siguiente: en primer lugar, se indica la variable respuesta que necesariamente tiene que ser una variable numérica. A continuación, el símbolo ~ indica la dependencia de la variable numérica respecto de las variables que se indiquen a su derecha. Finalmente, se incluyen las variables independientes separadas por el símbolo +.
- `data`: Opcional, sirve para especificar, si es necesario, el *data frame* al que pertenecen las variables utilizadas en la fórmula.
- `subset`: Opcional, sirve para especificar que la regresión sólo tenga en cuenta un subconjunto de las observaciones.

Así, por ejemplo, si tenemos un *data frame* llamado `DF`, con una variable numérica `Y` y tres variables independientes $Var1$, $Var2$ y $Var3$, para realizar la regresión lineal de la variable `Y` respecto del resto de variables mediante `lm` se podría ejecutar

```
lm(Y~Var1+Var2+Var3,data=DF)  o  lm(DF$Y~DF$Var1+DF$Var2+DF$Var3).
```

Un atajo muy útil en la ejecución de `lm` cuando disponemos de muchas variables independientes y se quiere obtener la recta de regresión considerándolas todas es ejecutar

`lm(Y~.,data=DF)`.

Esta instrucción realiza la regresión lineal de la variable Y en función de todas las otras variables que se encuentran en el *data frame* llamado `DF`.

El siguiente ejemplo ilustra el uso de la función `lm` y de la salida que proporciona.

Ejemplo 27.1. La tabla de datos `Davis` del paquete `car` contiene datos del peso y la altura de 200 hombres y mujeres que realizan ejercicio habitualmente obtenidos de dos formas distintas: la medición de estas magnitudes mediante los instrumentos adecuados y los valores que han notificado cada persona antes de realizarse las mediciones. [1]

```
> install.packages("car",dep=TRUE)
> library(car)
> str(Davis)
'data.frame':   200 obs. of  5 variables:
 $ sex   : Factor w/ 2 levels "F","M": 2 1 1 2 1 2 2 2 2 2 ...
 $ weight: int  77 58 53 68 59 76 76 69 71 65 ...
 $ height: int  182 161 161 177 157 170 167 186 178 171 ...
 $ repwt : int  77 51 54 70 59 76 77 73 71 64 ...
 $ repht : int  180 159 158 175 155 165 165 180 175 170 ...
```

Como se puede observar, el *data frame* contiene una variable `sex` de tipo factor con los niveles "F" y "M" indicando el sexo del sujeto; las variables cuantitativas `weight` y `height` que indican el peso (en kg.) y la altura (en cm.) medidas respectivamente; y las variables cuantitativas `repwt` y `repht` que indican el peso (en kg.) y la altura (en cm.) notificadas por el sujeto respectivamente.

En este primer ejemplo, nos centramos en las variables `weight` y `repwt`. Aunque se dispongan de datos de 200 personas, sólo 183 de ellas son observaciones completas.

```
> datospeso=data.frame(Davis$weight,Davis$repwt)
> colnames(datospeso)=c("peso","pesorep")
> head(datospeso)
  peso pesorep
1   77      77
2   58      51
3   53      54
4   68      70
5   59      59
```

[1] Véase Davis, C. (1990) *Body image and weight preoccuppation: A comparison between exercising and non-exercising women.* Appetite, 15, p. 13-21.

```
6   76      76
> datospeso=na.omit(datospeso)
> dim(datospeso)
[1] 183   2
```

Para determinar la recta de regresión por mínimos cuadrados de la variable `peso` respecto a la variable `pesorep`, ejecutaremos la función `lm`.

```
> lm(peso~pesorep,data=datospeso)

Call:
lm(formula = peso ~ pesorep, data = datospeso)

Coefficients:
(Intercept)      pesorep
     5.3363       0.9278
```

El resultado obtenido significa que la recta de regresión buscada tiene término independiente 5.3363 y coeficiente de la variable `pesorep` 0.9278, es decir,

$$peso = 5.3363 + 0.9278 \cdot pesorep.$$

Podemos representar gráficamente los puntos de la muestra conjuntamente con la recta de regresión encontrada:

```
> plot(datospeso$pesorep,datospeso$peso)
> abline(lm(peso~pesorep,data=datospeso))
```

(ver Figura 27.1-(a)). Como se puede observar, la recta de regresión se ajusta notablemente a los puntos de la muestra indicando un buen comportamiento del modelo a nivel visual. Sin embargo, destaca una de las observaciones que se encuentra muy alejada tanto del patrón seguido por el resto de puntos como de la recta de regresión. Para obtener a qué observación corresponde ese punto, podemos utilizar la función `identify` de `R` con la que será suficiente pulsar cerca del punto anómalo en el gráfico para obtener el número de esta observación.

```
> identify(datospeso$pesorep,datospeso$peso)
> #Podemos observar que se trata del individuo 12
> datospeso[12,]
   peso pesorep
12  166      56
```

(ver Figura 27.1-(b)). El punto anómalo, correspondiente al individuo 12 de la muestra, ¡tiene un peso de 166 kg aunque dijo que su peso era de 56 kg! Más adelante, en la sección

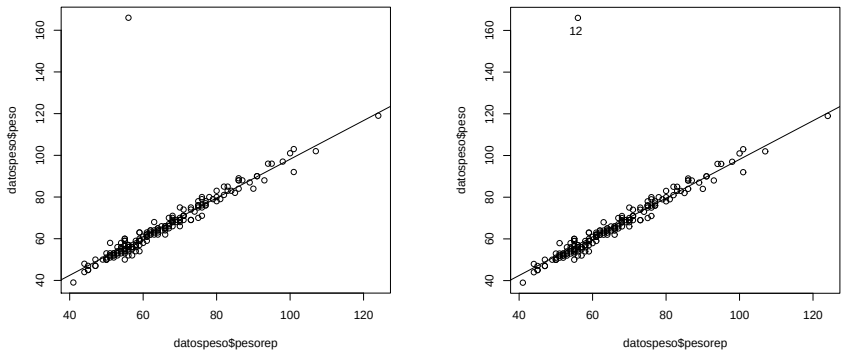

Figura 27.1. Gráficos de las observaciones $(pesorep, peso)$ con la recta de regresión. En el de la derecha, se ha identificado un punto anómalo.

del estudio de los residuos, trataremos en profundidad el procedimiento a seguir en estas situaciones.

Ya hemos visto que los coeficientes de la recta de regresión pueden ser obtenidos simplemente ejecutando `lm`. Sin embargo, esta función nos puede proporcionar mucha información adicional, usando `summary(lm())`.

```
> summary(lm(peso~pesorep,data=datospeso))

Call:
lm(formula = peso ~ pesorep, data = datospeso)

Residuals:
    Min      1Q  Median      3Q     Max
 -7.048  -1.868  -0.728   0.601 108.705

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.3363     3.0369   1.757   0.0806 .
pesorep       0.9278     0.0453  20.484   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.419 on 181 degrees of freedom
Multiple R-squared:  0.6986,    Adjusted R-squared:  0.697
F-statistic: 419.6 on 1 and 181 DF,  p-value: < 2.2e-16
```

En esta salida encontramos la siguiente información:

- En `Residuals`, se proporciona un resumen de los residuos o errores e_i del modelo, concretamente, el valor mínimo de los residuos, el valor máximo y los cuartiles.
- En la tabla de `Coefficients`, en la columna `Estimate` se dan los coeficientes de cada variable de la recta de regresión, junto a sus respectivas desviaciones estándar en la columna `Std. Error`. A continuación, las columnas `t value` y `Pr(>|t|)` proporcionan el valor del estadístico y el p-valor del contraste

$$\begin{cases} H_0 : \beta_i & = 0, \\ H_1 : \beta_i & \neq 0. \end{cases}$$

  Según lo significativo que sea el p-valor, `R` lo indica al final de la tabla mediante su código característico de estrellas.
- Después de la tabla, encontramos `Residual standard error` que corresponde a la raíz del valor estimado de la varianza común de los residuos $\sqrt{S^2} = \sqrt{\frac{SS_E}{n-k-1}}$, junto con los grados de libertad $n-k-1$.
- En la siguiente fila, tenemos `Multiple R-squared` y `Adjusted R-squared`, es decir, los valores de los coeficientes de determinación R^2 y de determinación ajustado R^2_{adj}, respectivamente. Recordemos que $R^2 = \frac{SS_R}{SS_T}$, $R^2_{adj} = 1 - (1-R^2)\frac{n-1}{n-k-1}$ y que como mejor aproxime la ecuación de regresión el conjunto de puntos, más cercano a 1 serán sus valores.
- En la última fila, se indican el valor del estadístico F, los grados de libertad $n-k-1$ y el p-valor en este orden del siguiente contraste ANOVA

$$\begin{cases} H_0 : & \beta_1 = \ldots = \beta_k = 0, \\ H_1 : & \text{existe al menos un } \beta_i \neq 0. \end{cases}$$

  En la regresión lineal simple, este contraste es equivalente al contraste para β_1 dado en la tabla de coeficientes.

En este ejemplo concreto, hay que destacar que el valor de R^2 es 0.6986, un valor bastante bajo. También podemos concluir que $\beta_1 \neq 0$ ya que el p-valor correspondiente al contraste para la variable *pesorep* es pequeño, así como el p-valor correspondiente al contraste ANOVA. Este hecho permite determinar que el modelo lineal es un modelo válido ya que en caso contrario, si $\beta_1 = 0$, entonces los valores de la variable *peso* no dependerían de los de la variable *pesorep* y el modelo carecería de sentido.

La información proporcionada por `summary(lm())` se puede extraer individualmente al tratarse de una lista de 12 elementos.

```
> str(summary(lm(peso~pesorep,data=datospeso)))
```

```
List of 12
 $ call         : language lm(formula = peso ~ pesorep, data = datospeso)
 $ terms        :Classes 'terms', 'formula' length 3 peso ~ pesorep
  .. ..- attr(*, "variables")= language list(peso, pesorep)
  .. ..- attr(*, "factors")= int [1:2, 1] 0 1
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:2] "peso" "pesorep"
  .. .. .. ..$ : chr "pesorep"
  .. ..- attr(*, "term.labels")= chr "pesorep"
  .. ..- attr(*, "order")= int 1
  .. ..- attr(*, "intercept")= int 1
  .. ..- attr(*, "response")= int 1
  .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
  .. ..- attr(*, "predvars")= language list(peso, pesorep)
  .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
  .. .. ..- attr(*, "names")= chr [1:2] "peso" "pesorep"
 $ residuals    : Named num [1:183] 0.22 5.34 -2.44 -2.29 -1.08 ...
  ..- attr(*, "names")= chr [1:183] "1" "2" "3" "4" ...
 $ coefficients : num [1:2, 1:4] 5.3363 0.9278 3.0369 0.0453 1.7571 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "(Intercept)" "pesorep"
  .. ..$ : chr [1:4] "Estimate" "Std. Error" "t value" "Pr(>|t|)"
 $ aliased      : Named logi [1:2] FALSE FALSE
  ..- attr(*, "names")= chr [1:2] "(Intercept)" "pesorep"
 $ sigma        : num 8.42
 $ df           : int [1:3] 2 181 2
 $ r.squared    : num 0.699
 $ adj.r.squared: num 0.697
 $ fstatistic   : Named num [1:3] 420 1 181
  ..- attr(*, "names")= chr [1:3] "value" "numdf" "dendf"
 $ cov.unscaled : num [1:2, 1:2] 1.30e-01 -1.90e-03 -1.90e-03 2.89e-05
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "(Intercept)" "pesorep"
  .. ..$ : chr [1:2] "(Intercept)" "pesorep"
 $ na.action    :Class 'omit'  Named int [1:17] 47 48 55 76 100 125 127 138 154 158 ...
  .. ..- attr(*, "names")= chr [1:17] "47" "48" "55" "76" ...
 - attr(*, "class")= chr "summary.lm"
>
> summary(lm(peso~pesorep,data=datospeso))$r.squared #El coeficiente de determinación
[1] 0.6986308
> summary(lm(peso~pesorep,data=datospeso))$adj.r.squared #El coeficiente ajustado
[1] 0.6969658
> summary(lm(peso~pesorep,data=datospeso))$coefficients #La tabla de coeficientes
             Estimate Std. Error   t value     Pr(>|t|)
```

```
(Intercept) 5.3362605 3.03690979  1.757135 8.058558e-02
pesorep     0.9278428 0.04529609 20.483950 5.102807e-49
> #Las estimaciones de los coeficientes
> summary(lm(peso~pesorep,data=datospeso))$coefficients[,1]
(Intercept)     pesorep
  5.3362605   0.9278428
> head(summary(lm(peso~pesorep,data=datospeso))$residuals) #Los residuos
         1          2          3          4          5          6
 0.2198430  5.3437561 -2.4397724 -2.2852573 -1.0789864  0.1476858
```

Como podemos observar, se proporcionan todos los residuos del modelo. Éstos también se pueden obtener directamente del `lm` sin la necesidad de aplicar el `summary`. Además, también se pueden recuperar los valores predichos por el modelo $\hat{y}_i$.

```
> recta_regresion=lm(peso~pesorep,data=datospeso)
> residuos=recta_regresion$residuals #Residuos o errores del modelo
> head(residuos)
         1          2          3          4          5          6
 0.2198430  5.3437561 -2.4397724 -2.2852573 -1.0789864  0.1476858
> estimados=recta_regresion$fitted.values #Valores predichos por el modelo
> head(estimados)
       1        2        3        4        5        6
76.78016 52.65624 55.43977 70.28526 60.07899 75.85231
```

Recordemos que en este caso, el valor del coeficiente de determinación R^2 ha resultado bajo. Sin embargo, hemos observado que visualmente la gran mayoría de observaciones se ajustan a la recta de regresión salvo la observación anómala 12. A continuación, calcularemos la recta de regresión sin tener en cuenta esta observación mediante el uso del parámetro `subset` de la función `lm`.

```
> summary(lm(peso~pesorep,data=datospeso,subset=-12))$r.squared
[1] 0.9719157
```

En la instrucción, hemos excluído la observación 12 del cálculo de la recta de regresión obteniendo un coeficiente de determinación mucho mayor que el inicial. Hay que notar que se ha utilizado nuevamente la función `lm` para calcular la nueva recta de regresión. Sin embargo, la función `update` permite recalcular la recta de regresión a partir de una recta de regresión anterior. La sintaxis es la siguiente:

```
update(x,formula.,subset=...)
```

con los argumentos siguientes:

- `x` es el modelo generado por una función como `lm`.

- `formula.` indica un cambio en la `formula` especificada para obtener el nuevo modelo. Es muy útil en regresión lineal múltiple para eliminar una de las variables consideradas. Su estructura es `.~.-`X_i, donde se indica que se utilice la misma `formula` considerada en el modelo x pero ahora sin tener en cuenta la variable independiente X_i.
- `subset` es opcional y tiene el mismo significado que en la función `lm`.

La salida de la función `update` es similar a la de `lm`. Así, con la instrucción siguiente, se obtiene el mismo resultado para el coeficiente de determinación:

```
> summary(update(recta_regresion,subset=-12))$r.squared
[1] 0.9719157
```

El siguiente objetivo será el cálculo de los intervalos de confianza para los coeficientes β_i del modelo lineal así como para el valor esperado $\mu|_{Y_{x_1 \dots x_k}}$ y para $y_0 = y(x_1, \dots, x_k)$. La función de `R` donde se implementa el cálculo de los intervalos de confianza para los coeficientes β_i es `confint`, cuya sintaxis es la siguiente

```
confint(object, parm, level=0.95).
```

Los parámetros de entrada son:

- `object` es un modelo de regresión lineal, es decir, la salida de la función `lm`.
- `parm` indica para qué parámetros se tienen que calcular los intervalos de confianza. Tanto se puede introducir un vector de números como un vector de nombres de parámetros. Por defecto se calculan los intervalos para todos los parámetros.
- `level` es el nivel de confianza del intervalo. Por defecto, se calculan intervalos de confianza al 95 %.

Una vez introducida la función `confint`, vamos a usarla para encontrar los intervalos de confianza para los parámetros β_0 y β_1 del Ejemplo 27.1.

```
> confint(recta_regresion)
                 2.5 %    97.5 %
(Intercept) -0.6560394 11.328560
pesorep      0.8384665  1.017219
> confint(recta_regresion,"pesorep")
            2.5 %   97.5 %
pesorep 0.8384665 1.017219
```

En la salida de la función, se indican el extremo inferior y superior del intervalo de confianza mediante el nivel del cuantil utilizado para su cálculo para el parámetro correspondiente a

cada variable independiente, indicada en la primera columna, más el término independiente. Así, el intervalo de confianza para β_0 es $(-0.656, 11.330)$ y para β_1, $(0.838, 1.017)$. En la segunda instrucción, se ha indicado que sólo se calcule el correspondiente a β_1. Una de las utilidades básicas de los intervalos de confianza es determinar si la variable correspondiente aporta información al modelo o no. Este hecho, además de venir determinado por el valor del p-valor obtenido en la tabla de coeficientes de salida de `lm`, también se puede determinar comprobando si 0 pertenece al intervalo de confianza respectivo. En ese caso, no se puede descartar que $\beta_i = 0$ y en consecuencia, podría ocurrir que la variable X_i no influyera en el modelo. En este ejemplo concreto, 0 pertenece al IC de β_0 pero no al de β_1. En resumen, el modelo no tiene problemas en este sentido ya que β_0 (el término independiente) puede valer 0 sin afectar a la validez del modelo.

Llegados a este punto, vamos a introducir la función `predict` que resultará útil para el cálculo de intervalos de confianza para $\mu|_{Y_{x_1 \dots x_k}}$ y para $y_0 = y(x_1, \dots, x_k)$. La sintaxis de esta función es la siguiente:

```
predict(object,newdata,interval=c(‘‘none’’,‘‘confidence’’,‘‘prediction’’),level=0.95)
```

donde los parámetros de entrada son

- `object` es un modelo de regresión lineal, es decir, la salida de la función `lm`.
- `newdata` corresponde a un *data frame* donde la función recaba los valores de las variables con las que se predice. Podemos indicar un único valor para una única variable x como `data.frame(x=x0)`, varios valores para una única variable x como `data.frame(x=seq(x0,x1,p))` o valores para diversas variables $x_1, \dots, x_k$ como `data.frame(x1=x10, x2=x20, ..., xk=xk0)`.
- `interval` es un parámetro con tres posibles valores. Si se indica `"none"`, simplemente se calcula el valor predicho por la recta de regresión para los valores indicados en `newdata`. Si se indica `çonfidence"`, se determina el intervalo de confianza para el valor esperado $\mu|_{Y_{x10 \dots xk0}}$ donde $x_{10}, \dots, x_{k0}$ son los valores asignados a las variables independientes en `newdata`. Finalmente, si se indica `"prediction"`, se calcula el intervalo de confianza para el valor $y_0 = y(x_1, \dots, x_k)$.
- `level` vuelve a indicar el nivel de confianza del intervalo.

Volviendo a nuestro ejemplo anterior, encontremos algunos intervalos de confianza para valores esperados y valores de y_0.

```
> #Intervalo de confianza para el valor esperado del peso y el peso en si
> #cuando el individuo dice que su peso es de 70 kg.
> newdata=data.frame(pesorep=70)
> predict(recta_regresion,newdata,interval="confidence",level=0.95)
       fit      lwr      upr
```

```
1 70.28526 68.99651 71.57401
> predict(recta_regresion,newdata,interval="prediction",level=0.95)
       fit      lwr      upr
1 70.28526 53.62409 86.94642
> #Intervalo de confianza para el valor esperado del peso y el peso en si
> #cuando el individuo dice que su peso es de 100 kg.
> newdata=data.frame(pesorep=100)
> predict(recta_regresion,newdata,interval="confidence",level=0.95)
       fit      lwr      upr
1 98.12054 94.81176 101.4293
> predict(recta_regresion,newdata,interval="prediction",level=0.95)
       fit      lwr      upr
1 98.12054 81.18296 115.0581
```

La salida de la función `predict` es un vector con tres componentes. En `fit`, encontramos el valor predicho por la recta de regresión para los valores establecidos en `newdata` y en `lwr` y `upr`, se indican el extremo inferior y superior respectivamente del intervalo de confianza que se ha pedido. Por ejemplo, tenemos que si una persona dice que su peso es de 100 kg, entonces la recta de regresión predice un peso de 98.12 kg, un intervalo de confianza para el valor esperado de $(94.812, 101.429)$ y de $(81.183, 115.058)$ para el peso en sí.

Finalmente, para acabar con este primer ejemplo de regresión lineal simple, vamos a mostrar algunos criterios visuales preliminares para determinar la validez y buen comportamiento del método además del ya conocido valor de R^2. En primer lugar, es habitual comprobar si los residuos tienen una varianza común y se sitúan de forma simétrica con respecto a la recta $y = 0$. Recordemos que éste es uno de los requisitos del modelo de regresión lineal. Para comprobarlo, se comprueba el gráfico `valores predichos` vs `residuos`. El gráfico puede mostrar tres fenómenos:

1. Sin problemas, cuando el gráfico se asemeja a un cielo estrellado donde no hay zonas grandes sin estrellas.

2. Heteroscedasticidad, cuando los residuos pasan de ser muy pequeños en valor absoluto a ir aumentando paulatinamente. En este caso, la varianza no es constante.

3. No linealidad, cuando se observa una tendencia o patrón en los residuos. Este caso es indicativo de que los datos no siguen un modelo lineal.

En el primer gráfico de la Figura 27.2, se puede observar que la observación anómala 12 distorsiona el gráfico y el modelo en sí. En cambio si se elimina la observación 12, recobramos un gráfico de `valores predichos` vs `residuos` mucho más interpretable. Observemos pero que encontramos zonas sin puntos y existe una cierta tendencia de los puntos a alinearse en rectas paralelas. Por lo tanto, el modelo es problemático en este sentido.

```
> plot(recta_regresion$fitted.values,recta_regresion$residuals,
```

```
+ xlab="Valores predichos",ylab="Residuos")
> recta_regresion2=update(recta_regresion,subset=-12)
> plot(recta_regresion2$fitted.values,recta_regresion2$residuals,
+ xlab="Valores predichos",ylab="Residuos")
```

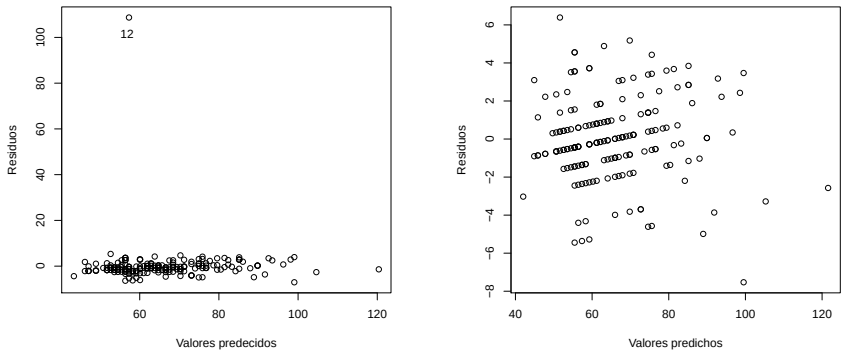

Figura 27.2. Gráficos `valores predichos` vs `residuos`. En el de la derecha, se ha eliminado la observación anómala 12.

Por otro lado, se puede comprobar la normalidad de los residuos mediante las funciones `qqnorm` y `qqline`, ya introducidas en lecciones anteriores.

```
> qqnorm(recta_regresion$residuals)
> qqline(recta_regresion$residuals)
> recta_regresion2=lm(peso~pesorep,data=datospeso,subset=-12)
> qqnorm(recta_regresion2$residuals)
> qqline(recta_regresion2$residuals)
```

Dando lugar a los gráficos de la Figura 27.3, donde se observa nuevamente el mal comportamiento del modelo cuando se incluye la observación 12. Sin embargo, aunque se elimine esta observación, aún no podemos asegurar la normalidad de los residuos ya que para valores pequeños y grandes del cuantil teórico, los cuantiles de la muestra se desvían de la recta dada por el `qqline`.

Como recurso adicional, vamos a realizar un `ks.test` para contrastar si los residuos de la regresión lineal cuando se elimina la observación 12 siguen o no una normal. También, por razones de completitud, realizaremos el contraste para los datos completos.

```
> #Calculamos la estimación de la desviación típica de los errores
> S=summary(lm(peso~pesorep,data=datospeso))$sigma
> ks.test(recta_regresion$residuals,"pnorm",0,S)
```

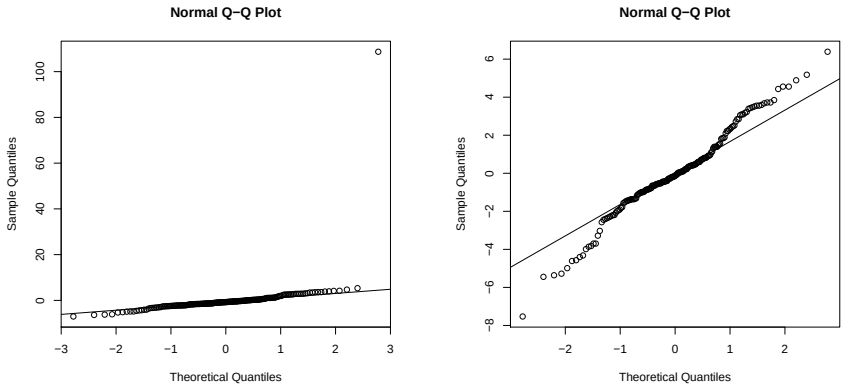

Figura 27.3. Gráficos `qqnorm` con las rectas dadas con el `qqline`. En el de la derecha, se ha eliminado la observación anómala 12.

```
        One-sample Kolmogorov-Smirnov test

data:  recta_regresion$residuals
D = 0.2948, p-value = 3.064e-14
alternative hypothesis: two-sided

Warning message:
In ks.test(recta_regresion$residuals, "pnorm", 0, S) :
  ties should not be present for the Kolmogorov-Smirnov test
> #Ahora para el modelo sin la observación 12
> Ssin12=summary(lm(peso~pesorep,data=datospeso,subset=-12))$sigma
> ks.test(recta_regresion2$residuals,"pnorm",0,Ssin12)

        One-sample Kolmogorov-Smirnov test

data:  recta_regresion2$residuals
D = 0.0762, p-value = 0.2417
alternative hypothesis: two-sided

Warning message:
In ks.test(recta_regresion2$residuals, "pnorm", 0, Ssin12) :
  ties should not be present for the Kolmogorov-Smirnov test
```

A partir de los resultados obtenidos, sí podemos asegurar la normalidad de los residuos cuando se elimina el dato anómalo.

El siguiente ejemplo ilustrará el uso de las funciones explicadas en esta lección para modelos de regresión lineal múltiple.

Ejemplo 27.2. En el paquete `Faraway`, se dispone de la tabla de datos `gala` que recoge la diversidad de especies de tortugas en las Islas Galápagos. Este archipiélago del Oceáno Pacífico está compuesto por 30 islas. En la tabla de datos, se recogen para cada una de las islas 7 variables. Aunque la tabla de datos original[2] contenía algunos valores perdidos, posteriormente ha sido completada en su totalidad. Las variables son las siguientes:

- `Species`, el número de especies de tortuga encontradas en la isla.
- `Endemics`, el número de especies endémicas.
- `Area`, el área de la isla en km^2.
- `Elevation`, la máxima elevación de la isla en m.
- `Nearest`, la distancia a la isla más cercana en km.
- `Scruz`, la distancia a la Isla de Santa Cruz en km.
- `Adjacent`, el área de la isla adyacente en km^2.

Después de cargar el paquete `Faraway`, chequeamos la tabla de datos:

```
> install.packages("faraway",dep=TRUE)
> library(faraway)
> data(gala)
> str(gala)
'data.frame':   30 obs. of  7 variables:
 $ Species  : num  58 31 3 25 2 18 24 10 8 2 ...
 $ Endemics : num  23 21 3 9 1 11 0 7 4 2 ...
 $ Area     : num  25.09 1.24 0.21 0.1 0.05 ...
 $ Elevation: num  346 109 114 46 77 119 93 168 71 112 ...
 $ Nearest  : num  0.6 0.6 2.8 1.9 1.9 8 6 34.1 0.4 2.6 ...
 $ Scruz    : num  0.6 26.3 58.7 47.4 1.9 ...
 $ Adjacent : num  1.84 572.33 0.78 0.18 903.82 ...
> head(gala)
            Species Endemics  Area Elevation Nearest Scruz Adjacent
Baltra           58       23 25.09       346     0.6   0.6     1.84
Bartolome        31       21  1.24       109     0.6  26.3   572.33
Caldwell          3        3  0.21       114     2.8  58.7     0.78
Champion         25        9  0.10        46     1.9  47.4     0.18
Coamano           2        1  0.05        77     1.9   1.9   903.82
Daphne.Major     18       11  0.34       119     8.0   8.0     1.84
```

[2]Véase M.P. Johnson y P.H. Raven (1973) *Species number and endemism: The Galapagos Archipelago revisited.* Science, 179, p. 893-895.

Como podemos observar, todas las variables son númericas y tenemos 30 observaciones de las 7 variables correspondientes a las 30 islas que conforman el archipiélago.

El objetivo del estudio es estudiar la relación entre el número de especies de tortugas en una isla con respecto a las variables geográficas anteriormente expuestas. Aquí nos centramos en encontrar una dependencia lineal entre la variable `Species` y el resto de variables. Al disponer de 6 variables independientes, nos encontramos ante un problema de regresión lineal múltiple.

En primer lugar, calcularemos la ecuación de regresión lineal múltiple mediante la función `lm`.

```
> ec_regresion=lm(Species~Endemics+Area+Elevation+Nearest+
+ Scruz+Adjacent, data=gala) #Encontramos la ecuación de regresión
> summary(ec_regresion)

Call:
lm(formula = Species ~ Endemics + Area + Elevation + Nearest +
    Scruz + Adjacent, data = gala)

Residuals:
    Min      1Q  Median      3Q     Max
-68.219 -10.225   1.830   9.557  71.090

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -15.337942   9.423550  -1.628    0.117
Endemics      4.393654   0.481203   9.131 4.13e-09 ***
Area          0.013258   0.011403   1.163    0.257
Elevation    -0.047537   0.047596  -0.999    0.328
Nearest      -0.101460   0.500871  -0.203    0.841
Scruz         0.008256   0.105884   0.078    0.939
Adjacent      0.001811   0.011879   0.152    0.880
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 28.96 on 23 degrees of freedom
Multiple R-squared:  0.9494,    Adjusted R-squared:  0.9362
F-statistic: 71.88 on 6 and 23 DF,  p-value: 9.674e-14
```

A simple vista, vemos que la instrucción necesaria para realizar la regresión es compleja debido al número de variables independientes. Sin embargo, recordemos que existe un atajo para realizar la regresión en función del resto de variables del *data frame*

```
> summary(lm(Species~.,data=gala))$coefficients[,1]
```

```
  (Intercept)      Endemics          Area     Elevation       Nearest
-15.337942100   4.393653649   0.013258111  -0.047537328  -0.101459511
        Scruz      Adjacent
  0.008255847   0.001810681
```

Se obtienen los mismos coeficientes y la instrucción es mucho más simple. Volviendo a la tabla de coeficientes obtenida con `lm` podemos destacar el notable valor del coeficiente de determinación (0.9494) y el p-valor muy significativo ($\approx 10^{-14}$) obtenido en el contraste de ANOVA que nos asegura que existe al menos un $\beta_i \neq 0$. Por otra parte, a partir de los p-valores de la columna `Pr(>|t|)` sólo podemos asegurar que β_1 (el coeficiente de la variable `Endemics`) es distinto de 0. El resto de coeficientes no podemos descartar que valgan 0 y que por lo tanto, no podemos descartar que el resto de variables no influyan en el modelo. Para confirmar este hecho, calculemos los intervalos de confianza para los coeficientes.

```
> confint(ec_regresion)
                    2.5 %     97.5 %
(Intercept) -34.83203994 4.15615574
Endemics      3.39820859 5.38909871
Area         -0.01033151 0.03684773
Elevation    -0.14599740 0.05092274
Nearest      -1.13758991 0.93467089
Scruz        -0.21078156 0.22729325
Adjacent     -0.02276207 0.02638344
```

Lógicamente sólo el intervalo de confianza para β_1 no contiene el 0. Por otro lado, podríamos determinar una predicción para el número de especies de tortugas y para su valor esperado en una isla imaginaria del archipiélago con, por ejemplo, 15 especies endémicas, un área de 7 km^2, una elevación de 200 m, situada a 5 km de la isla más cercana y a 30 km de Santa Cruz y adyacente a una isla con 3 km^2 de área.

```
> newdata=data.frame(Endemics=15,Area=7,Elevation=200,Nearest=5,Scruz=30,
+ Adjacent=3)
> predict(ec_regresion,newdata,interval="prediction",level=0.95)
       fit       lwr     upr
1 40.89801 -20.59093 102.387
> predict(ec_regresion,newdata,interval="confidence",level=0.95)
       fit      lwr     upr
1 40.89801 27.07972 54.7163
```

El modelo predice un valor estimado próximo a 41 especies de tortugas con un intervalo de confianza para el valor esperado de $(27.08, 54.72)$ y un intervalo de confianza para el valor de especies de tortugas de $(-20.59, 102.39)$. Óbviamente no se puede tener un número negativo de especies de tortugas y se debe interpretar el intervalo como $[0, 102.39)$. Aún así el intervalo es muy ancho y de escasa utilidad.

Del análisis preliminar realizado, nos encontramos ante una situación compleja. El modelo parece ser válido y efectivo en base al valor del coeficiente de determinación pero hay muchas variables que pueden no ser relevantes en el modelo. Por otro lado, es conocido que el valor de R^2 aumenta en relación al número de variables aunque éstas sean redundantes. Las variables redundantes dificultan la interpretación del modelo y es conveniente eliminarlas. Sin embargo, surge la duda de determinar de forma objetiva cuál es el mejor modelo, en el sentido de cuál es el mínimo número de variables y cuáles son las que me explican la mayor cantidad de varianza posible. Así, llegamos a la cuestión de cómo comparar dos modelos de regresión múltiple con un número distinto de variables. Tenemos disponibles tres métodos para este fin:

- Comparación de los R^2_{adj}: El coeficiente de determinación ajustado tiene en cuenta el número de variables utilizadas en el modelo. El modelo con un mayor R^2_{adj} es el óptimo.

- Las medidas AIC (*Akaike's Information Criterion*) y BIC (*Bayesian Information Criterion*): La medida AIC cuantifica cuánta información de la variable dependiente se pierde con el modelo y el número de variables utilizado. En cambio, la medida BIC también tiene en cuenta el número de datos empleado. Se calculan con la función de `R`

  `extractAIC(x,k)`

  donde `x` es un modelo lineal obtenido mediante la función `lm` y `k` es el peso que afecta al número de variables del modelo. El valor de k es 2 por defecto correspondiendo al criterio AIC. Si se indica $k = log(n)$ donde n es el número de observaciones, `extractAIC` calcula la medida BIC. Como menor sea el valor de AIC o BIC, mejor es el modelo. Los valores de estas medidas se obtienen a partir de `extractAIC(x,k)[2]`.

Los tres criterios no son equivalentes y por lo tanto, pueden darse resultados contradictorios entre ellos. En ese caso, se debe indicar la preferencia de cada criterio y realizar más análisis para determinar qué modelo es mejor. A continuación, eliminaremos la variable `Scruz` del modelo de regresión y compararemos los valores de las tres medidas para determinar qué modelo es mejor.

```
> n=dim(gala)[1]
> c(summary(ec_regresion)$adj.r.squared,extractAIC(ec_regresion)[2],
+extractAIC(ec_regresion,k=log(n))[2])
[1]   0.9361603 207.9916752 217.8000568
> ec_regresion2=update(ec_regresion,.~.-Scruz)
> c(summary(ec_regresion2)$adj.r.squared,extractAIC(ec_regresion2)[2],
+extractAIC(ec_regresion2,k=log(n))[2])
[1]   0.9388041 205.9996038 214.4067881
```

Las tres medidas coinciden en su veredicto y por lo tanto, el segundo modelo es mejor. La dificultad estriba en encontrar cuál es la mejor combinación de variables. El proceso manual es tedioso y el número de casos es muy elevado. Para solucionar este problema, en `R` disponemos de la función

```
step(x,k)
```

donde `x` es un modelo lineal obtenido mediante la función `lm` y `k` es el peso que afecta al número de variables del modelo. El valor de k es 2 por defecto correspondiendo al criterio AIC. Si se indica $k = log(n)$ donde n es el número de observaciones, `step` se ejecuta en base a BIC. Esta función realiza un proceso secuencial de eliminación o adición de variables consiguiendo en cada paso del algoritmo un modelo con un valor de AIC (o BIC) menor al modelo obtenido en el paso anterior.

```
> step(ec_regresion)
Start:  AIC=207.99
Species ~ Endemics + Area + Elevation + Nearest + Scruz + Adjacent

            Df Sum of Sq   RSS    AIC
- Scruz      1         5 19300 206.00
- Adjacent   1        19 19314 206.02
- Nearest    1        34 19329 206.04
- Elevation  1       837 20132 207.26
- Area       1      1134 20429 207.71
<none>                  19295 207.99
- Endemics   1     69937 89231 251.93

Step:  AIC=206
Species ~ Endemics + Area + Elevation + Nearest + Adjacent

            Df Sum of Sq   RSS    AIC
- Adjacent   1        19 19318 204.03
- Nearest    1        32 19332 204.05
- Elevation  1       838 20138 205.28
- Area       1      1129 20429 205.71
<none>                  19300 206.00
- Endemics   1     74567 93867 251.45

Step:  AIC=204.03
Species ~ Endemics + Area + Elevation + Nearest

            Df Sum of Sq    RSS    AIC
- Nearest    1        50  19369 202.11
<none>                   19318 204.03
```

```
- Area       1      1376  20694 204.09
- Elevation  1      2232  21550 205.31
- Endemics   1    150599 169918 267.26

Step:  AIC=202.11
Species ~ Endemics + Area + Elevation

            Df Sum of Sq    RSS    AIC
<none>                    19369 202.11
- Area       1      1498  20866 202.34
- Elevation  1      2287  21655 203.45
- Endemics   1    150578 169947 265.26

Call:
lm(formula = Species ~ Endemics + Area + Elevation, data = gala)

Coefficients:
(Intercept)     Endemics         Area    Elevation
  -15.89124      4.33179      0.01267     -0.04144
```

En la primera iteración del algoritmo, tenemos en `Start` el valor de AIC para el modelo dado por `x`. A continuación, en cada paso se disponen en una tabla las variables y el valor de AIC que obtendría el modelo si se eliminara la variable en cuestión. Así, en la primera iteración se elimina `Scruz` ya que su eliminación proporciona un modelo con un valor AIC=206. Al principio de cada iteración se indican la `formula` correspondiente al modelo en ese momento. El algoritmo finaliza cuando si se elimina cualquiera de las variables restantes, aumenta el AIC empeorando el modelo. En este ejemplo, se eliminan `Scruz`, `Adjacent` y `Nearest`, resultando un modelo de regresión con sólo tres variables y con un valor de AIC=202.11. Finalmente, se indican los coeficientes de la ecuación de regresión del modelo simplificado.

Veamos el resultado obtenido para la optimización según los valores de BIC.

```
> step(ec_regresion,k=log(n))
Start:  AIC=217.8
Species ~ Endemics + Area + Elevation + Nearest + Scruz + Adjacent

            Df Sum of Sq   RSS    AIC
- Scruz      1         5 19300 214.41
- Adjacent   1        19 19314 214.43
- Nearest    1        34 19329 214.45
- Elevation  1       837 20132 215.67
- Area       1      1134 20429 216.11
<none>                  19295 217.80
- Endemics   1     69937 89231 260.34
```

```
Step:  AIC=214.41
Species ~ Endemics + Area + Elevation + Nearest + Adjacent

            Df Sum of Sq   RSS    AIC
- Adjacent   1        19 19318 211.03
- Nearest    1        32 19332 211.06
- Elevation  1       838 20138 212.28
- Area       1      1129 20429 212.71
<none>                  19300 214.41
- Endemics   1     74567 93867 258.46

Step:  AIC=211.03
Species ~ Endemics + Area + Elevation + Nearest

            Df Sum of Sq    RSS    AIC
- Nearest    1        50  19369 207.71
- Area       1      1376  20694 209.70
- Elevation  1      2232  21550 210.91
<none>                   19318 211.03
- Endemics   1    150599 169918 272.86

Step:  AIC=207.71
Species ~ Endemics + Area + Elevation

            Df Sum of Sq    RSS    AIC
- Area       1      1498  20866 206.54
- Elevation  1      2287  21655 207.66
<none>                   19369 207.71
- Endemics   1    150578 169947 269.46

Step:  AIC=206.54
Species ~ Endemics + Elevation

            Df Sum of Sq    RSS    AIC
- Elevation  1      1007  21874 204.56
<none>                   20866 206.54
- Endemics   1    152387 173254 266.64

Step:  AIC=204.56
Species ~ Endemics

           Df Sum of Sq    RSS    AIC
<none>                  21874 204.56
```

```
- Endemics  1    359208 381081 286.89

Call:
lm(formula = Species ~ Endemics, data = gala)

Coefficients:
(Intercept)     Endemics
    -21.048        4.072
```

Como se puede observar, los resultados son diferentes. Según los valores de BIC, se han eliminado en este orden las variables `Scruz`, `Adjacent`, `Nearest`, `Area` y `Elevation` quedando solamente la variable `Endemics`. Por lo tanto, según BIC, ¡el mejor modelo es una regresión lineal simple! Veamos los valores de los respectivos R^2 y R^2_{adj} en cada caso.

```
> ec_regresionAIC=summary(lm(Species ~ Endemics + Area + Elevation, data = gala))
> c(ec_regresionAIC$r.squared,ec_regresionAIC$adj.r.squared)
[1] 0.9491740 0.9433095
> ec_regresionBIC=summary(lm(Species ~ Endemics, data = gala))
> c(ec_regresionBIC$r.squared,ec_regresionBIC$adj.r.squared)
[1] 0.9426012 0.9405513
```

Ambos indicadores concluyen que el modelo dado por AIC es mejor en este problema concreto y por lo tanto, podemos concluir que la ecuación de regresión óptima según los indicadores es

```
Species = -15.89 + 4.33Endemics + 0.01267Area -0.04144Elevation
```

Estudiemos finalmente, los gráficos de los residuos de este modelo para determinar si existe alguna deficiencia en esta regresión. Ambos gráficos se muestran en la Figura 27.4.

```
> #Gráfico fitted vs residuals
> predichosAIC=lm(Species ~ Endemics + Area + Elevation, data = gala)
+ $fitted.values
> plot(predichosAIC,ec_regresionAIC$residuals, xlab="Valores predichos",
+ ylab="Residuos")
> abline(h=0)
> #Gráfico qqnorm
> qqnorm(ec_regresionAIC$residuals)
> qqline(ec_regresionAIC$residuals)
```

Se puede observar que la varianza de los residuos no es constante y que tenemos un claro ejemplo de heteroscedasticidad. En cambio, el gráfico de la normalidad de los residuos es dudoso y por lo tanto, realizaremos un test de Kolmogorov-Smirnov para contrastar la normalidad.

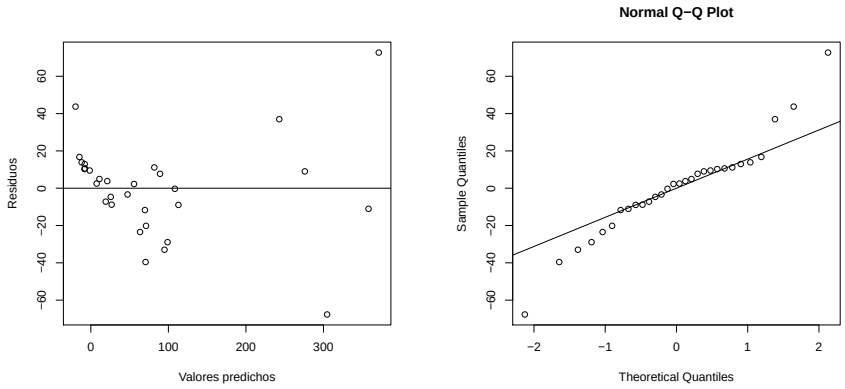

Figura 27.4. Gráficos `valores predichos` vs `residuos` y gráfico `qqnorm` para contrastar la normalidad de los residuos gráficamente.

```
> #Calculamos la estimación de la desviación típica de los errores
> S=summary(lm(Species ~ Endemics + Area + Elevation, data = gala))$sigma
> ks.test(ec_regresionAIC$residuals,"pnorm",0,S)

        One-sample Kolmogorov-Smirnov test

data:  ec_regresionAIC$residuals
D = 0.1724, p-value = 0.2992
alternative hypothesis: two-sided
```

A partir de los resultados obtenidos, sí podemos asegurar la normalidad de los residuos.