Regresión Lineal Múltiple

Laura Zapata Prada

En el presente documento se realizará la solución de un ejercicio completo con sus respectivos análisis para tener claro cómo hacer un modelo de regresión lineal múltiple.

Lectura de Datos

Iniciamos con la lectura de datos, la cual será guardada con el nombre de "datos" y en este caso es extraído de una ruta en mi ordenador personal.

```
datos <- read.table("C:/Users/acer/Documents/inyectora.txt",header = TRUE)

str(datos)
```

```
## 'data.frame':    100 obs. of  11 variables:
##  $ y  : num  22.5 17.6 22.5 20.9 18.6 ...
##  $ x1 : num  -0.425 0.577 -0.182 0.766 0.881 ...
##  $ x2 : num  0.2 -0.3344 -0.0228 0.9089 -0.0342 ...
##  $ x3 : num  -0.5225 0.9247 0.2027 0.0301 -0.1949 ...
##  $ x4 : num  0.569 -0.981 0.558 0.459 0.26 ...
##  $ x5 : num  0.972 -0.726 0.811 0.153 -0.209 ...
##  $ x6 : num  -0.293 -0.267 -0.426 -0.84 -0.269 ...
##  $ x7 : num  -0.526 0.373 -0.548 -0.363 -0.652 ...
##  $ x8 : num  0.69 -0.48 -0.954 0.725 -0.331 ...
##  $ x9 : num  -0.0586 -0.2683 -0.7575 -0.906 -0.4744 ...
##  $ x10: num  0.8474 0.0852 0.7047 0.1671 0.3366 ...
```

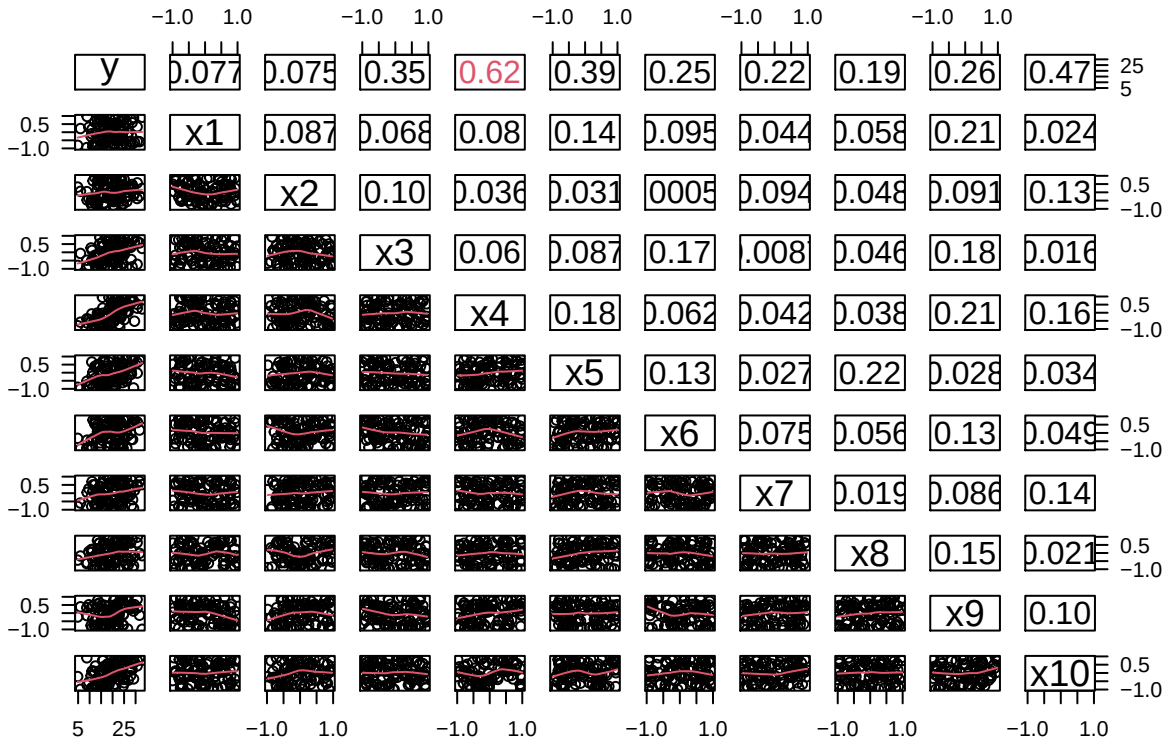
Con la función str() es posible conocer como está formada la base de datos, en este caso con 11 variables de tipo numérico.

Analisis de Correlación entre variables

Antes de iniciar con la realización del modelo de regresión múltiple, sería interesante ver la correlación existente entre las variables independientes (x1...x10) con la variable dependiente (y), además, es importante tener en cuenta que las variables independientes (x1...x10) no deben estar correlacionadas entre sí, debido a que supondría unos esfuerzos innecesarios el medir más cantidad de variables para posteriormente tener una predicción (en la que no se requieren de tantas variables)

```
##pairs plot
panel.cor <- function(x, y, digits = 2, prefix = "", cex.cor, ...)
{
    usr <- par("usr"); on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    r <- abs(cor(x, y))
    txt <- format(c(r, 0.123456789), digits = digits)[1]
    txt <- paste0(prefix, txt)
    text(0.5, 0.5, txt, cex = 1.5, col = ifelse(r > 0.5, 2, 1))
    #Donde dice col = ifelse(r > 0.5, 2, 1), solo busca escribir con color rojo
```

```
    #las correlaciones mayores a 0.5
}
pairs(datos, lower.panel = panel.smooth, upper.panel = panel.cor, las= 1)
```



Otro tipo de grafico que es llamativo visualmente y demuestra la correlación entre variables es el siguiente, es importante tener en cuenta que mientras más gruesa este la línea significa fuerte correlación, además, las que se encuentren en color verde reflejan correlación positiva, mientras que las de color rojo presentan correlación negativa entre las variables.

Este tipo de grafico sirve como un complemento del anterior. Para este se requieren tener 2 paquetes instalados como se muestra en el siguiente código.

```
## Verifica disponibilidad del paquete IsingSampler
if(!require(IsingSampler)) install.packages("IsingSampler")
```

```
## Loading required package: IsingSampler
```

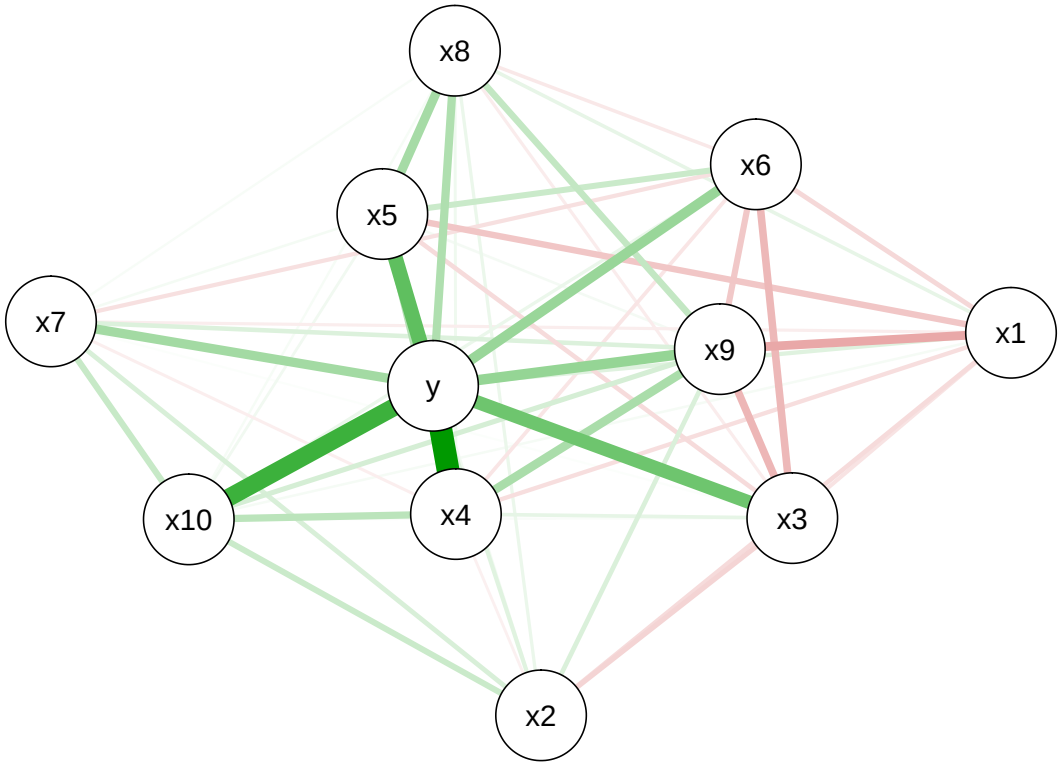
```
## Loading required package: Rcpp
```

```
require("IsingSampler")

## Verifica disponibilidad del paquete qgraph
if(!require(qgraph)) install.packages("qgraph")
```

```
## Loading required package: qgraph
```

```
require("qgraph")

## matriz de correlación
```

```r
par(mfrow = c(1,1), mar = c(.1, .1, .1, .1))
corMat <- cor(datos)
qgraph(corMat, graph = "cor", layout = "spring",
       sampleSize = nrow(datos),
       legend.cex = 1, alpha = 0.05)
```



Modelo de Regresión Lineal Múltiple

Después de realizar este análisis previo es posible pasar a crear el modelo de regresión lineal múltiple, este se llamará "modelo" y se ajustará mediante la función lm, en donde y ~. significa que y es la variable dependiente y (.) que el resto de las variables dentro de la base de datos (sin contar la y) serían las independientes.

```r
## modelo de RLM
modelo <- lm(y ~ ., data = datos)
summary(modelo)
```

```
##
## Call:
## lm(formula = y ~ ., data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.09900 -0.61503 -0.04698  0.49843  2.35473
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 19.97501    0.09496 210.353  < 2e-16 ***
## x1           3.04897    0.17692  17.234  < 2e-16 ***
## x2           0.85418    0.18412   4.639 1.20e-05 ***
## x3           4.98623    0.17088  29.179  < 2e-16 ***
## x4           4.97033    0.17161  28.963  < 2e-16 ***
## x5           2.99030    0.17537  17.052  < 2e-16 ***
## x6           4.03875    0.16979  23.786  < 2e-16 ***
## x7           2.08153    0.16505  12.612  < 2e-16 ***
## x8           0.89152    0.16997   5.245 1.05e-06 ***
## x9           2.90108    0.18030  16.090  < 2e-16 ***
## x10          2.89888    0.17135  16.918  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9418 on 89 degrees of freedom
## Multiple R-squared:  0.9778, Adjusted R-squared:  0.9753
## F-statistic: 391.2 on 10 and 89 DF,  p-value: < 2.2e-16
```

Al realizar el summary del modelo es posible obtener varias conclusiones.

0. El "p-value" al final de la tabla se refiere a la prueba de significancia global, en donde quiero rechazar H0, y esto se logra si el valor p es menor a Alpha=0.05; es importante que sea así, para poder validar el modelo, en caso contrario el modelo no sería útil.

Recordando la prueba de hipótesis para la significancia global seria la siguiente:

H0: b0=b1=bk=0 H1: Al menos un b!=0

1. El "Estimate" se refiere al coeficiente (los betas) de cada variable al hacer el modelo y = b0 + b1*x1 + b2 °x2+ bk °xk

Donde b0 sería el "intercept" y en general no es tan importante al momento de hacer análisis comparado con las variables independientes.

Interpretación de los "bi": por cada cambio unitario en "xk", aumentara el "y" en "bk".

La variable con un valor mayor de "Estimate" significa que tiene un gran efecto o influencia sobre el modelo.

2. "Std. Error" puede presentar que tan estable es la variable, es decir, a menor error mayor estabilidad tendrá la variable.

3. "t value" presenta la importancia (equilibrio entre el efecto y la estabilidad) de la variable respectiva, siendo el cociente entre "Estimate" y "Std. Error", a mayor valor de "t value" mayor importancia para mi modelo.

En este caso existe un paquete que organiza de mayor a menor la importancia de las variables y se puede realizar un gráfico con esto:

```
## variable importance
if(!require('vip')) install.packages('vip')
```

```
## Loading required package: vip
```

```
##
## Attaching package: 'vip'
```

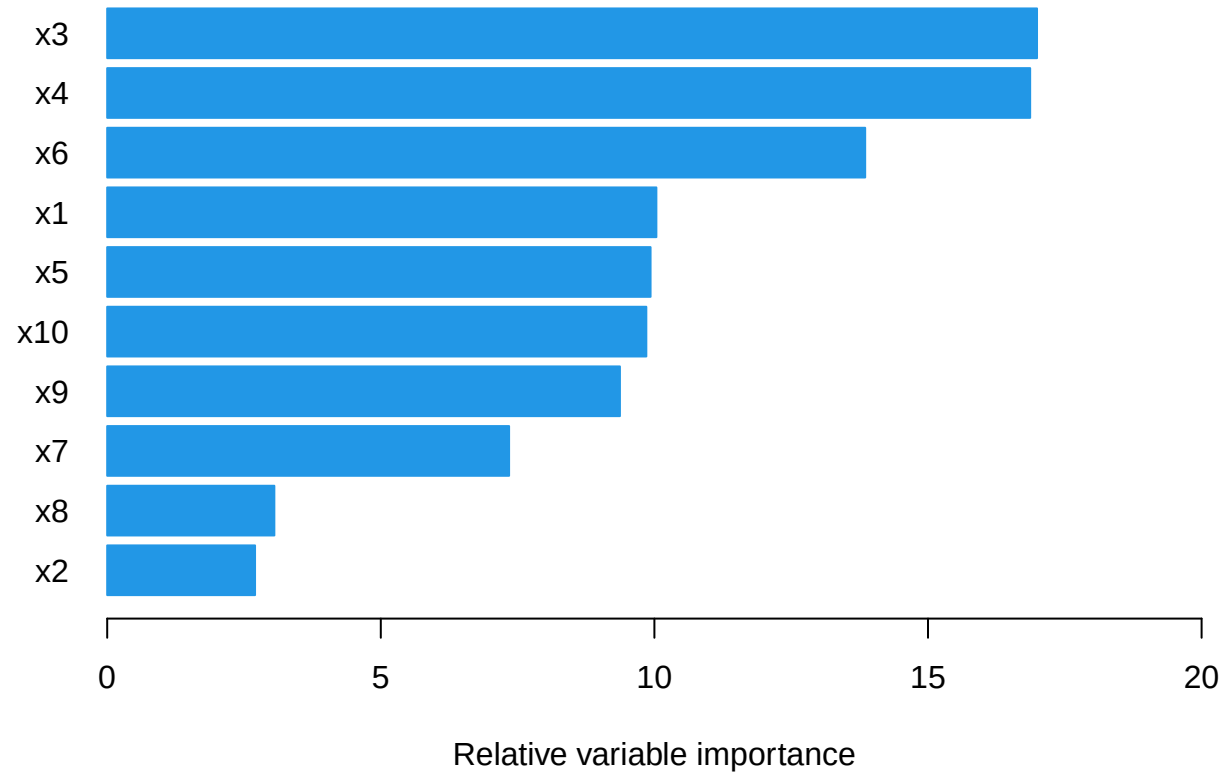
```
## The following object is masked from 'package:utils':
##
##     vi
```

```
require('vip')
```

```r
## calculations
(vi_result <- vi(modelo))
```

```
## # A tibble: 10 x 3
##    Variable Importance Sign
##    <chr>         <dbl> <chr>
##  1 x3            29.2  POS
##  2 x4            29.0  POS
##  3 x6            23.8  POS
##  4 x1            17.2  POS
##  5 x5            17.1  POS
##  6 x10           16.9  POS
##  7 x9            16.1  POS
##  8 x7            12.6  POS
##  9 x8             5.24 POS
## 10 x2             4.64 POS
```

```r
vi_mlr <-  100*vi_result$Importance / sum(vi_result$Importance)
names(vi_mlr) <- vi_result$Variable


## plot
par(mfrow = c(1, 1), mar = c(4, 3, 2, 1))
barplot(sort(vi_mlr), horiz = TRUE, col = 4, border = 4, las = 1,
        xlab = 'Relative variable importance', xlim = c(0, 20))
```



4. El "Pr (>|t|)" es el valor p para la prueba de significancia marginal para cada uno de los betas de las

variables. Recordando la prueba de hipótesis:

H0: b1=0 H1: b1!=0

Importante que ese valor p sea mejor a un Alpha de 0.05 para poder rechazar la hipótesis nula y poder afirmar que la variable es importante para mí modelo.

5. El "F-statistic" significa cuantas veces se explica el modelo más que los errores, por esto quiero que sea »1 y en caso de comparar 2 modelos de regresión, el que tenga un mayo "F-statistic" sería un mejor modelo, además, es importante tener claro que al eliminar error del modelo (mediante la reducción de variables insignificantes), el "F-statistic" va a aumentar.

6. Después de retirar las variables que no son significativas, se debe reajustar el modelo de regresión lineal múltiple, para así reducir el error de pronóstico.

Multicolinealidad

Si se rechazara la prueba de significancia global y no se rechazan todas las marginales, existiría la sospecha de multicolinealidad, para verificar esto se puede iniciar con el uso de:

Kappa(x), sí es mayor a 30, se confirma multicolinealidad general en el modelo, sin embargo, aún no conocería cual es la variable o variables que me generan multicolinealidad.

VIF, es útil para conocer que variables generan multicolinealidad, si el VIF de una variable o varias es **mayor a 5** se confirma multicolinealidad, y se debería proceder a eliminar del modelo la que presente un mayor VIF. Tener en cuenta que solo se puede eliminar UNA variable y recalcular el modelo, si siguen existiendo variables con multicolinealidad, se debe repetir el mismo proceso.

```
## Verifica disponibilidad del paquete car
if(!require("car")) install.packages("car")
```

```
## Loading required package: car
```

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

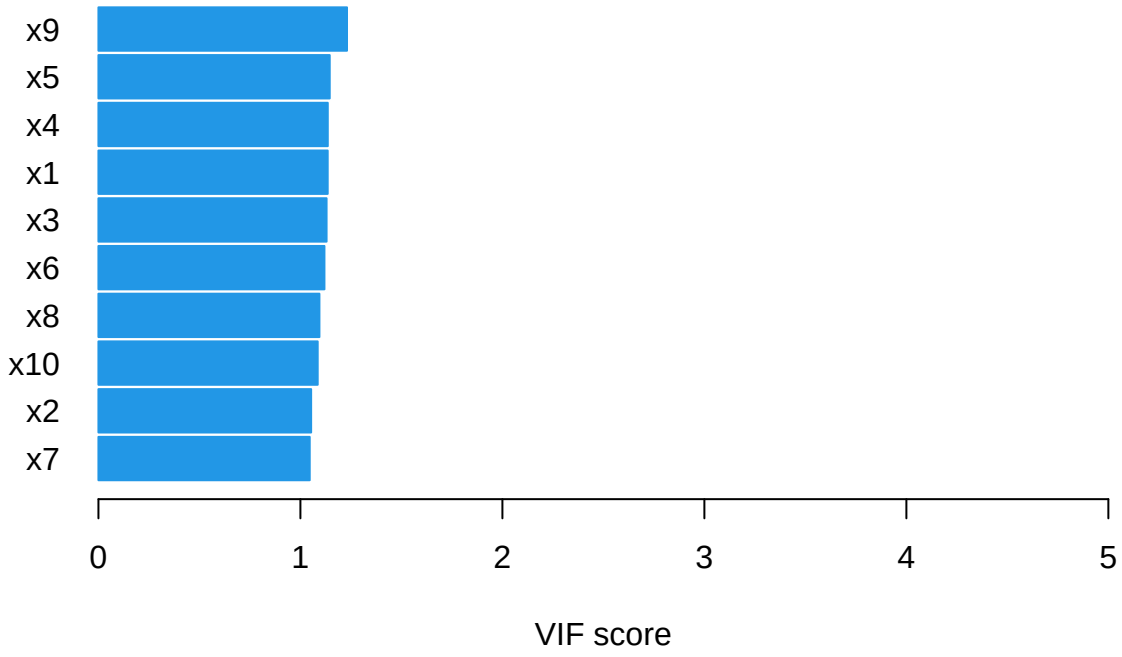
```
require("car")

## ICN
kappa(modelo)    # esta función es de base
```

```
## [1] 2.798045
```

```
## VIF
vif_result <- car::vif(modelo)
vif_result
```

```
##       x1       x2       x3       x4       x5       x6       x7       x8
## 1.135031 1.053053 1.129447 1.135348 1.145059 1.118952 1.046071 1.094115
##       x9      x10
## 1.230562 1.085487
```

```
## nice plot
barplot(sort(vif_result), horiz = TRUE, las = 1, space = 0.1, xlab = 'VIF score', col = 4,
        border = 4, xlim = c(0, 5))
```



En este ejemplo se obtiene un **Kappa menor a 30** y los **VIF de cada variable son menor a 5**, por lo que no existe multicolinealidad.

```
## Otro test EXTRA para comprobar multicolinealidad
if(!require("mctest")) install.packages("mctest")
```

```
## Loading required package: mctest
```

```
require("mctest")
omcdiag(modelo)
```

```
##
## Call:
## omcdiag(mod = modelo)
##
##
## Overall Multicollinearity Diagnostics
##
##                        MC Results detection
## Determinant |X'X|:         0.5906         0
## Farrar Chi-Square:        49.9457         0
## Red Indicator:             0.1036         0
## Sum of Lambda Inverse:    11.1731         0
## Theil's Method:           -7.7675         0
## Condition Number:          1.7981         0
##
## 1 --> COLLINEARITY is detected by the test
## 0 --> COLLINEARITY is not detected by the test
```

Como se observa en los resultados si en "detection" tuviese algún 1 indicaría multicolinealidad, sin embargo, debería verificar con el VIF cada variable para determinar en donde está el conflicto.

```
## Intervalos de Confianza para los Betas

result <- summary(modelo)
all <- coefficients(result)
ll <- all[,1] - 1.96*all[,2]
ul <- all[,1] + 1.96*all[,2]
data.frame(ll, betahat = all[,1], ul)
```

```
##                       ll    betahat        ul
## (Intercept) 19.7888867 19.9750069 20.161127
## x1           2.7022141  3.0489749  3.395736
## x2           0.4933062  0.8541805  1.215055
## x3           4.6512914  4.9862251  5.321159
## x4           4.6339778  4.9703275  5.306677
## x5           2.6465763  2.9902953  3.334014
## x6           3.7059575  4.0387494  4.371541
## x7           1.7580297  2.0815255  2.405021
## x8           0.5583660  0.8915160  1.224666
## x9           2.5476847  2.9010779  3.254471
## x10          2.5630307  2.8988761  3.234721
```

Debo verificar si alguno de los intervalos de confianza para los betas tiene el cero incluido.

Los **intervalos** que **NO incluyan al cero**, implican que a nivel poblacional, esas variables que controlo en el proceso realmente si afectan el valor esperado de la variable respuesta.

```
#Otra función EXTRA para el calculo automatico de intervalos de confianza de betas
confint(modelo)
```

```
##                   2.5 %    97.5 %
## (Intercept) 19.7863248 20.163689
## x1           2.6974411  3.400509
## x2           0.4883389  1.220022
## x3           4.6466812  5.325769
## x4           4.6293480  5.311307
## x5           2.6418452  3.338745
## x6           3.7013768  4.376122
## x7           1.7535769  2.409474
## x8           0.5537803  1.229252
## x9           2.5428204  3.259335
## x10          2.5584079  3.239344
```

El análisis que se puede hacer con los intervalos de confianza para los betas, es que el efecto de cambiar xk en una unidad, manteniendo todas las demás variables constantes a nivel poblacional estaría entre el límite inferior del intervalo de confianza y el límite superior.

En caso de que el cero estuviese dentro del intervalo significaría que no tiene efecto sobre el modelo cambiar xk en una unidad manteniendo las demás variables constantes.

Validación de Supuestos

Media Cero

El supuesto de Media Cero no se prueba, este se garantiza por la forma en que se hace la estimación del modelo, utilizando mínimos cuadrados ordinarios.

```r
## calculo de residuales estudentizados
r <- rstudent(modelo)

## media cero de los residuales?
t.test(r, mu = 0)
```

```
##
##  One Sample t-test
##
## data:  r
## t = 0.035961, df = 99, p-value = 0.9714
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.1989555  0.2063002
## sample estimates:
##  mean of x
## 0.00367236
```

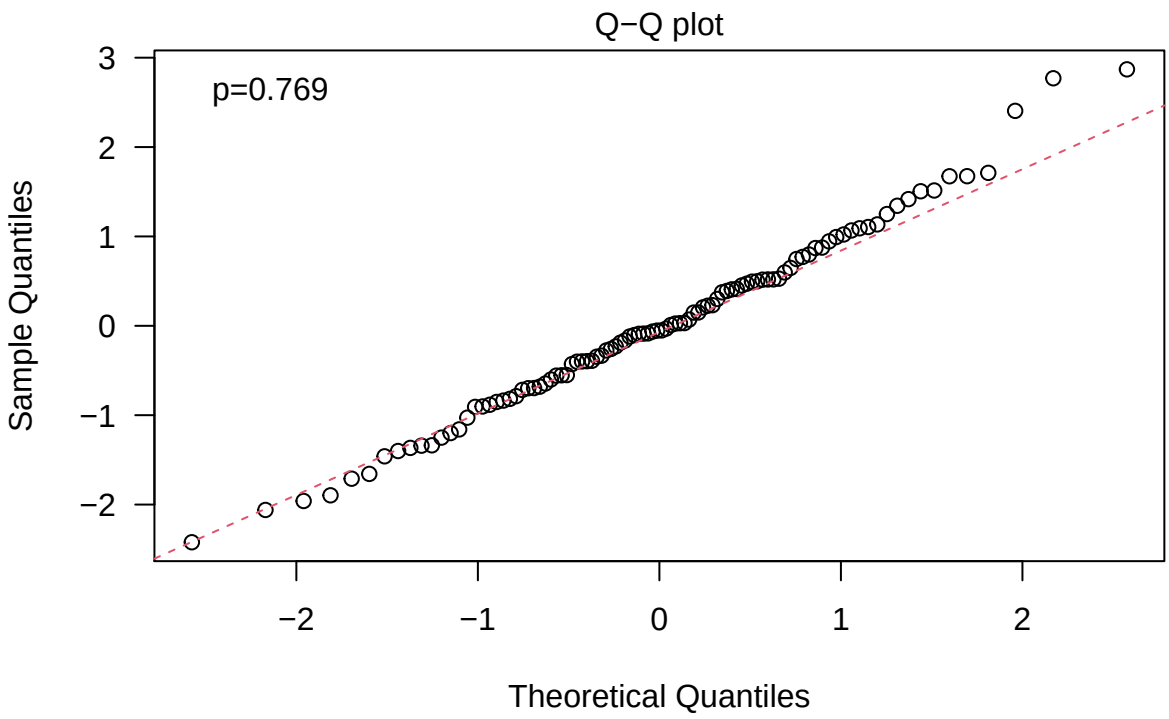
Al obtener un valor p mucho mayor a 0.05, no se rechaza que se cumpla el supuesto de media cero.

Recordar:

H0: Cumple Supuesto Media Cero H1: No Cumple Supuesto Media Cero

Normalidad

```r
## gráfico QQ-plot y prueba de Shapiro-Wilks
qqnorm(r, las = 1, main = "")
mtext("Q-Q plot", side = 3, line = .2)
qqline(r, lty = 2, col = 2)
legend('topleft', paste0("p=",round(shapiro.test(r)$p.value, 3)), bty = 'n')
```



Al obtener un valor p (0.769) mayor al Alpha de 0.05, no se rechaza que se cumpla el supuesto de normalidad.

Recordar:

H0: Cumple Supuesto Normalidad H1: No Cumple Supuesto Normalidad

Nota:

En casos en donde el tamaño de muestra sea mayor a 3000, se recomienda aplicar la prueba de "Kolmogorov-Smirnoff":

```
ks.test(r,"pnorm",mean(r),sd(r))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  r
## D = 0.054066, p-value = 0.9319
## alternative hypothesis: two-sided
```

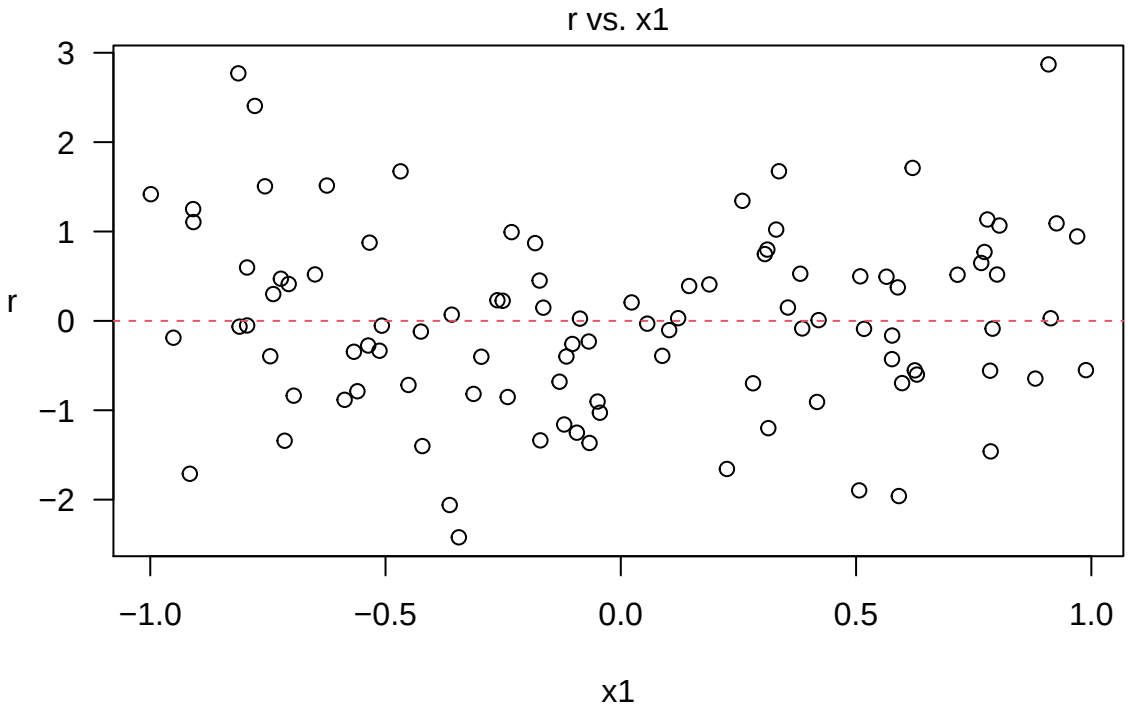
Al obtener un valor p (0.9319) mayor al Alpha de 0.05, no se rechaza que se cumpla el supuesto de normalidad.

Varianza Constante

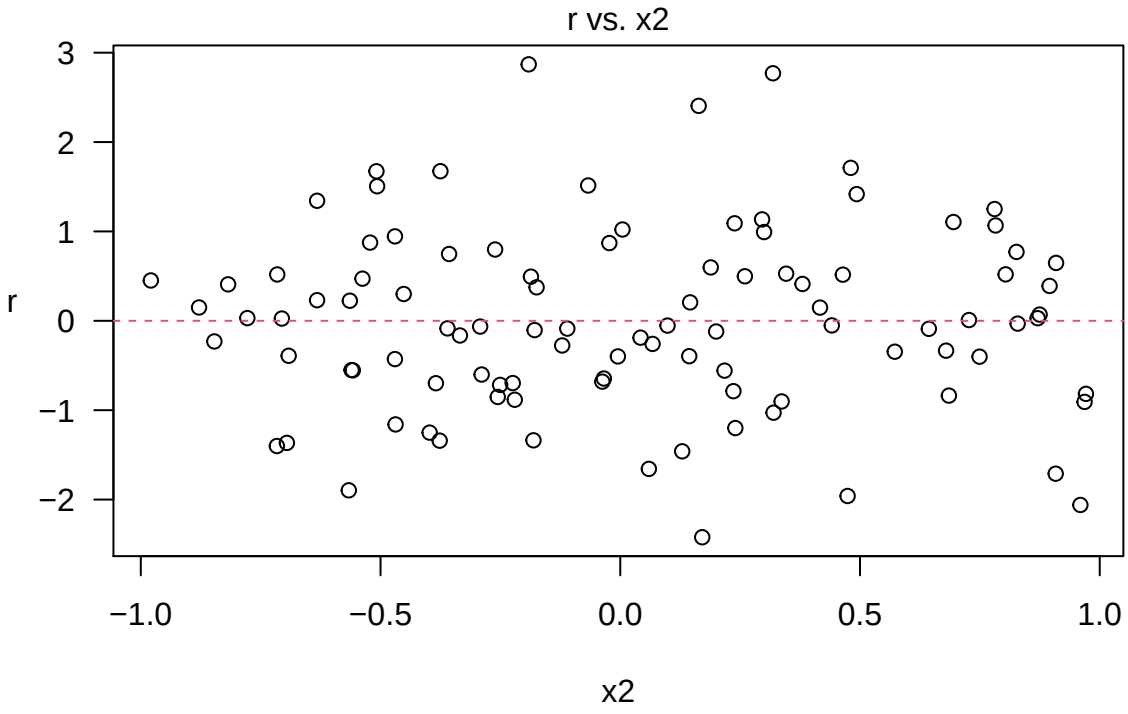
La varianza constante se puede verificar gráficamente al no observar patrones en los datos, la idea es que si los datos se encuentran dispersos sin ningún patrón como u, n entre otros; se cumpliría el supuesto de varianza constante.

```
## validación varianza constante:  gráfico r vs x1
plot(datos$x1, r, las = 1, ylab = "", xlab = 'x1')
```

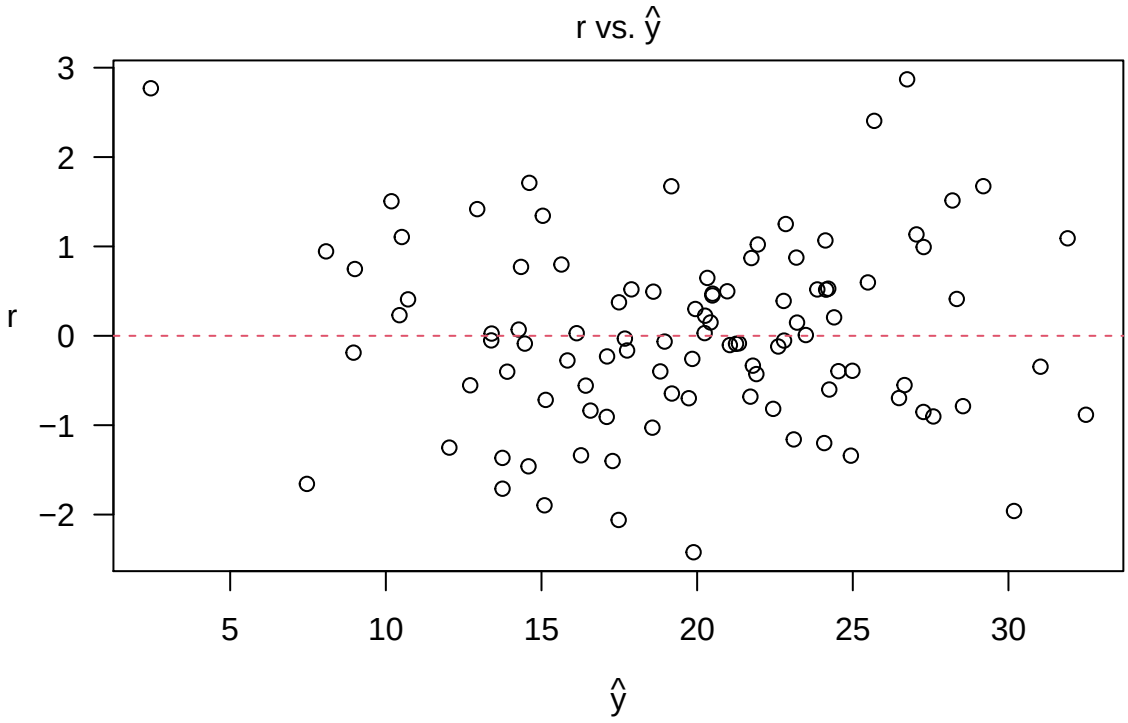
```
mtext("r", side = 2, line = 2.5, las = 1)
abline(h = 0, lty = 2, col = 2)
mtext("r vs. x1", side = 3, line = .2)
```




```
## validación varianza constante:   gráfico r vs x2
plot(datos$x2, r, las = 1, ylab = "", xlab = 'x2')
mtext("r", side = 2, line = 2.5, las = 1)
abline(h = 0, lty = 2, col = 2)
mtext("r vs. x2", side = 3, line = .2)
```




```
## validación varianza constante:   gráfico r vs yhat
plot(fitted(modelo), r, las = 1, xlab = expression(hat(y)), ylab = '')
mtext("r", side = 2, line = 2.5, las = 1)
abline(h = 0, lty = 2, col = 2)
mtext(expression("r vs. "*hat(y)), side = 3, line = .2)
```



Para realizar la comprobación con una prueba se haría de la siguiente forma (SOLO PARA REGRESIÓN LINEAL MULTIPLE):

```
## prueba de varianza constante
if(!require(car)) install.packages('car')
require(car)
car:::ncvTest(modelo)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.6473779, Df = 1, p = 0.42105
```

Al obtener un valor P mayor al Alpha de 0.05 no se rechaza que se cumpla el supuesto de varianza constante.

Recordar:

H0: Cumple Supuesto Varianza Constante H1: No Cumple Supuesto Varianza Constante

Nota:

Si al realizar los graficos correspondientes para validar Varianza Constante se ven tendencias, se recomienda ajustar el modelo con alguna transformación, por ejemplo con logaritmo.

En ese caso tu nuevo modelo seria por ejemplo:

modelo <- lm(log(y) ~ . , data)

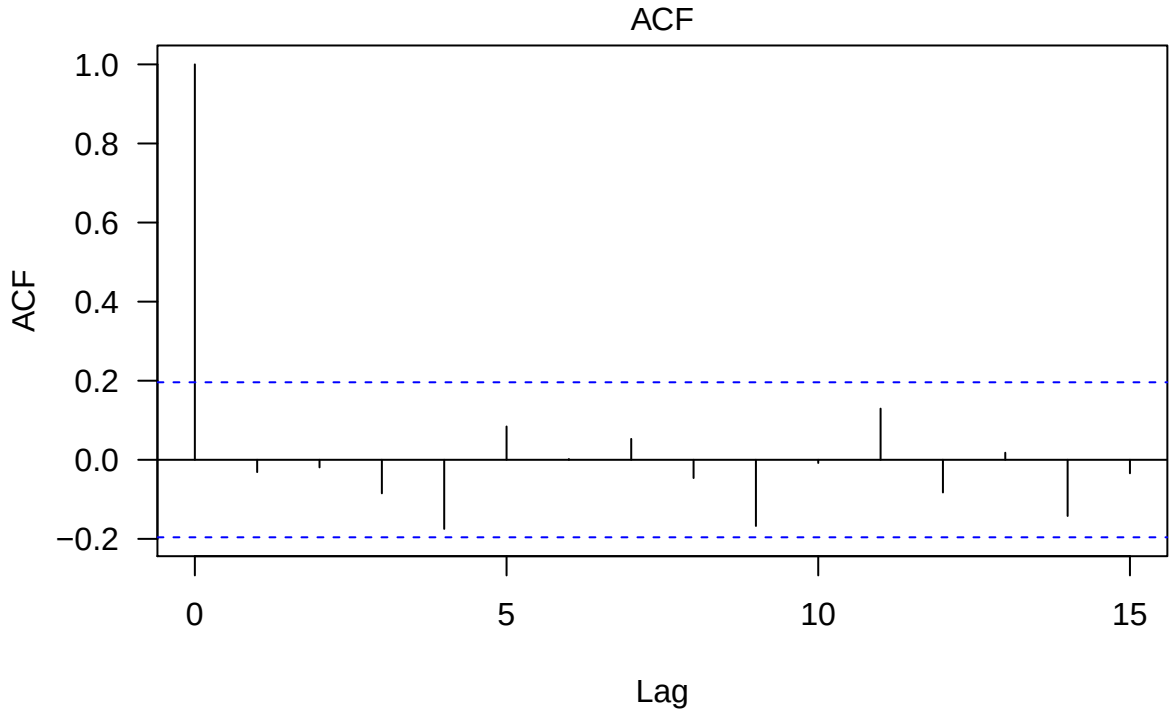
Y luego que hagas predicciones les debes quitar la transformación, por ejemplo: exp(predicciones)

Independencia

Para hacer la prueba de independencia se realiza el grafico de ACF, en el cual siempre la línea del valor 0 va a llegar a uno, por ser la relación consigo mismo.

La idea es que ninguna línea se salga de los limites hasta que llegue al valor del eje x en raíz de n, donde n es la cantidad de datos

```
## prueba de independencia usando la ACF
acf(r, las = 1, lag.max = 15, main = "")
mtext("ACF", side = 3, line = .2)
```



Por ejemplo, en este caso, ninguna línea se salió de los limites azules (además de la primera, que siempre se sale) antes del Lag igual a 10 (debido a que se tienen 100 datos y la raíz de 100 seria 10, por esto ese sería el limite a estudiar) Y como ninguna línea se salió, se asegura el supuesto de independencia.

De manera analítica se puede aplicar el test de Durbin-Watson

```
## prueba de independencia usando Durbin-Watson
car:::durbinWatsonTest(modelo)$p
```

```
## [1] 0.754
```

Al obtener un valor P mayor al Alpha de 0.05 no se rechaza que se cumpla el supuesto de Independencia.

Recordar:

H0: Cumple Supuesto Independencia H1: No Cumple Supuesto Independencia

Prueba Global para los Supuestos

En general, si se quiere verificar rápidamente el cumplimiento de los supuestos del modelo se podría verificar de la siguiente forma:

```
## prueba global para los supuestos del modelo usando el paquete gvlma
if(!require(gvlma)) install.packages('gvlma')
```

```
## Loading required package: gvlma
```

```
require(gvlma)
gvmodel <- gvlma(modelo)
summary(gvmodel)
```

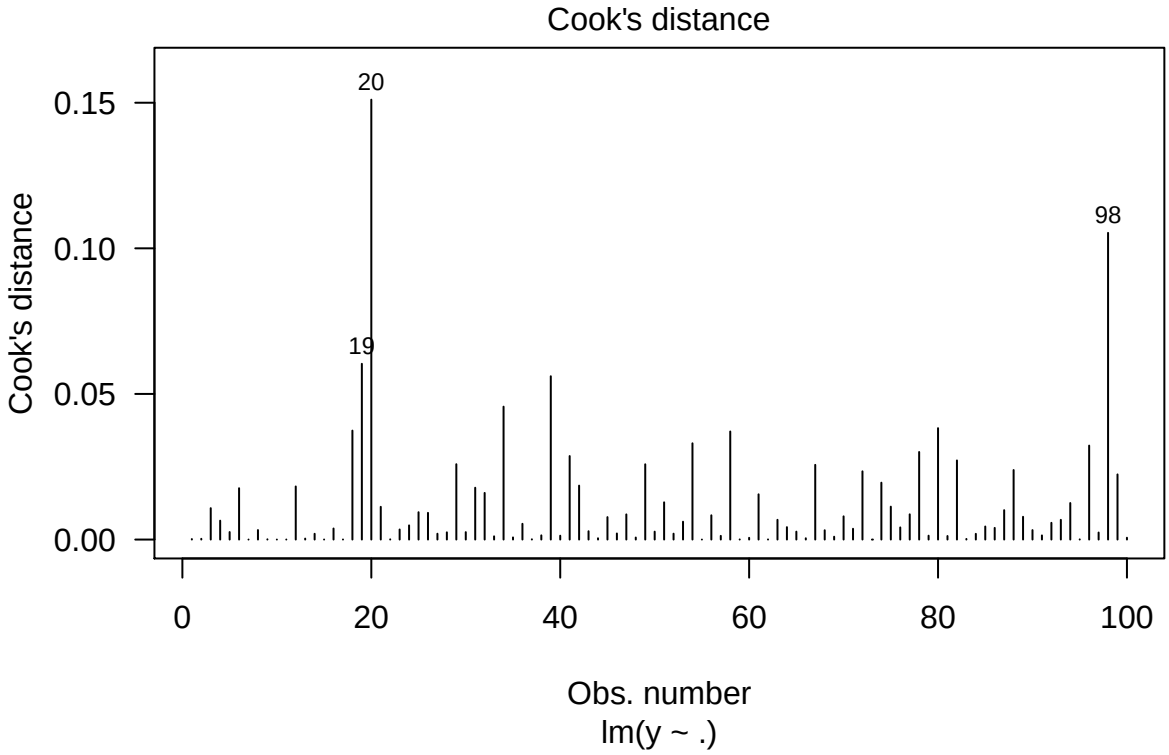
```
##
## Call:
## lm(formula = y ~ ., data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.09900 -0.61503 -0.04698  0.49843  2.35473
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19.97501    0.09496 210.353  < 2e-16 ***
## x1           3.04897    0.17692  17.234  < 2e-16 ***
## x2           0.85418    0.18412   4.639 1.20e-05 ***
## x3           4.98623    0.17088  29.179  < 2e-16 ***
## x4           4.97033    0.17161  28.963  < 2e-16 ***
## x5           2.99030    0.17537  17.052  < 2e-16 ***
## x6           4.03875    0.16979  23.786  < 2e-16 ***
## x7           2.08153    0.16505  12.612  < 2e-16 ***
## x8           0.89152    0.16997   5.245 1.05e-06 ***
## x9           2.90108    0.18030  16.090  < 2e-16 ***
## x10          2.89888    0.17135  16.918  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9418 on 89 degrees of freedom
## Multiple R-squared:  0.9778, Adjusted R-squared:  0.9753
## F-statistic: 391.2 on 10 and 89 DF,  p-value: < 2.2e-16
##
##
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
## Level of Significance =  0.05
##
## Call:
##  gvlma(x = modelo)
##
##                       Value p-value                   Decision
## Global Stat        6.192539 0.18522    Assumptions acceptable.
## Skewness           0.683052 0.40854    Assumptions acceptable.
## Kurtosis           0.006367 0.93640    Assumptions acceptable.
## Link Function      4.565942 0.03261 Assumptions NOT satisfied!
## Heteroscedasticity 0.937178 0.33300    Assumptions acceptable.
```

En este caso la única decisión a revisar debería ser el Global Stat, en este caso dice que es acceptable, por lo que se cumplirían todos los supuestos, en caso que dijera que no está satisfecho, significaría que existe algún supuesto que se incumple.

Datos Influenciales

Con datos influenciales nos referimos a los datos que realmente afectan a la variable respuesta y que sin estos datos, probablemente cambiaria el impacto sobre el modelo.

```
## Cook's distance
plot(modelo, which = 4, las = 1)
```



Se podría ir a los 3 datos que se observan en el gráfico, y estudiar cuales fueron sus causas y en caso de encontrar una causa asignable, lo correcto sería remover esas observaciones y calcular todo el modelo nuevamente.

Intervalos de Confianza y Predicción

El intervalo de predicción es más amplio comparado con el de confianza. Al cumplir con todos los supuestos se puede proceder a realizar predicciones.

Confianza

```
x0 <- data.frame(x1 = 0.8, x2 = 0.8, x3 = -0.4,
                 x4 = 0.7, x5 = -0.2, x6 = 0.32,
                 x7 = -0.9, x8 = 0.15, x9 = 0.25,
                 x10 = -0.98)
```

```
predict(modelo, newdata = x0, interval = 'confidence')
```

```
##        fit      lwr      upr
## 1 21.42134 20.67805 22.16462
```

Esperamos que con las condiciones de operación de (x1...x10), observar un valor de la respuesta en promedio lo que dice el "fit" y que el intervalo de confianza alrededor del cual se va a mover ese valor, a nivel poblacional con una confianza de 95% es desde "lwr" hasta "upr"

Cuando yo hago el intervalo de confianza, lo que busco es saber en que rango (con una confianza especifica) se van a mover el promedio de esa población cuando trabajo en esas condiciones de operación. (Por lo que requiero fabricar muchas piezas)

Predicción

```
x0 <- data.frame(x1 = 0.8, x2 = 0.8, x3 = -0.4,
                 x4 = 0.7, x5 = -0.2, x6 = 0.32,
                 x7 = -0.9, x8 = 0.15, x9 = 0.25,
                 x10 = -0.98)

predict(modelo, newdata = x0, interval = 'prediction')
```

```
##        fit     lwr      upr
## 1 21.42134 19.4078 23.43487
```

Si solo quiero hacer un ensayo para saber que valor obtendria y el rango en cuanto estaria. (Diferente al de confianza, porque no estoy trabajando para el promedio sino para el peso de una unidad)

Recordar que la predicción es cual valor esperaría obtener en la próxima unidad que voy a producir.