Practica II

Robinson Aldair Cuayal

2023-03-11

Table of Contents

Punto 1 3
a) P(x>200) 3
b) P(x<100) 3
c) P(100<x<200) 3
d) P(200<x<250) 4
e) N = 10000 y P(x >= 200) 5
Punto 2 5
Punto 3 6
a) Regresion lineal simple 7
Pruebas 8
Validacion de suspuestos 10
Conclusión 13
b) Regresion polinomica grado 2 13
Pruebas 14
Validacion de suspuestos 15
Conclusión 19
c) Regresión con transformación logarítmica 19
Pruebas 20
Validacion de suspuestos 21
Conclusión 24
Comparacion de modelos 24
punto 4 25
Ajuste del modelo 28
Metodo 1 29
Metodo 2 31
Conclusion 34

Punto 1

Los datos de este ejercicio son:

```
media = 140
desviacion = 50
```

a) P(x>200)

Lo primero que se debe hacer es estandarizar el valor de interes $x = 200$ de la siguiente manera

```
x = 200
z = (x-media)/desviacion
z

## [1] 1.2
```

Cabe resaltar que se puede colocar los valores directos en la funcion $pnorm(\quad)$, pero para un mayor entendimiento se manejara hallando Z. Despues se calcular la probabilidad acumulada de que un valor aleatorio sea mayor o igual a z, lo cual se hace con el complemento:

```
prob = 1- pnorm(z)
prob

## [1] 0.1150697
```

La probabilidad de que un individuo elegido al azar tenga un total de surcos en los dedos de 200 o más es del 11%, aproximadamente.

b) P(x<100)

Para este caso se realiza los mismos pasos del punto anterior

```
x = 100
z = (x - media) / desviacion
prob = pnorm(z)
prob

## [1] 0.2118554
```

Esto significa que hay una probabilidad del 21.18% de que un individuo elegido al azar tenga menos de 100 surcos en los dedos.

c) P(100<x<200)

Se realiza los pasos anteriores para $x_1 = 100$ y $x_2 = 200$ y con un paso adicional de restar al ultimo las probabilidades de estos.

```
x1 = 100
x2 = 200
```

```r
z1 = (x1 - media) / desviacion
z2 = (x2 - media) / desviacion
z1

## [1] -0.8

z2

## [1] 1.2

prob1 = pnorm(z1)
prob1

## [1] 0.2118554

prob2 = pnorm(z2)
prob2

## [1] 0.8849303

prob_total = prob1-prob2
prob_total

## [1] -0.6730749
```

Lo que significa que hay un 67% de probabilidad de que un individuo elegido al azar de la población tenga un total de surcos en los dedos entre 100 y 200.

d) P(200<x<250)

La solucion es lo mismo que el ejercicio anterior

```r
x1 = 200
x2 = 250
z1 = (x1 - media) / desviacion
z2 = (x2 - media) / desviacion
z1

## [1] 1.2

z2

## [1] 2.2

prob1 = pnorm(z1)
prob1

## [1] 0.8849303

prob2 = pnorm(z2)
prob2

## [1] 0.9860966
```

```
prob_total = prob1-prob2
prob_total

## [1] -0.1011662
```

Lo que significa que hay un 10% de probabilidad de que un individuo elegido al azar de la población tenga un total de surcos en los dedos entre 200 y 250

e) N = 10000 y P(x >= 200)

Para el calculo se multiplica la poblacion n con la probabilidad $n * (1 - pnorm(z))$.

```
n = 10000
x_critico = 200
n_200surcos = (1-pnorm(x_critico, media, desviacion))*n
round(n_200surcos)

## [1] 1151
```

Por lo tanto, podemos esperar que aproximadamente 1151 personas en una población de 10000 tengan un total de 200 surcos o más en los dedos.

Punto 2

Para encontrar un intervalo de confianza para la diferencia de las medias verdaderas entre los niveles de TCDD en plasma y tejido adiposo, se sigue los siguientes pasos: En primer lugar calcular las diferencias entre $u1 - u2$ pero esta columna ya esta (di).

```
nivel_tcdd_plasma =
c(2.5,3.1,2.1,3.5,3.1,1.8,6.0,3,36,4.7,6.9,3.3,4.6,1.6,7.2,1.8,20,2,2.5,4
.1)
nivel_tcdd_tejido =
c(4.9,5.9,4.4,6.9,7,4.2,10,5.5,41,4.4,7.0,2.9,4.6,1.4,7.7,1.1,11,2.5,2.3,
2.5)
di = c(-2.4,-2.8,-2.3,-3.4,-3.9,-2.4,-4.0,-2.5,-5.0,0.3,-0.1,0.4,0,0.2,-
0.5,0.7,9,-0.5,0.2,1.6)

datos = data.frame(nivel_tcdd_plasma,nivel_tcdd_tejido,di)
datos

##    nivel_tcdd_plasma nivel_tcdd_tejido   di
## 1                2.5               4.9 -2.4
## 2                3.1               5.9 -2.8
## 3                2.1               4.4 -2.3
## 4                3.5               6.9 -3.4
## 5                3.1               7.0 -3.9
## 6                1.8               4.2 -2.4
## 7                6.0              10.0 -4.0
## 8                3.0               5.5 -2.5
## 9               36.0              41.0 -5.0
```

```
## 10                4.7                 4.4  0.3
## 11                6.9                 7.0 -0.1
## 12                3.3                 2.9  0.4
## 13                4.6                 4.6  0.0
## 14                1.6                 1.4  0.2
## 15                7.2                 7.7 -0.5
## 16                1.8                 1.1  0.7
## 17               20.0                11.0  9.0
## 18                2.0                 2.5 -0.5
## 19                2.5                 2.3  0.2
## 20                4.1                 2.5  1.6
```

Como segundo paso, se calcual el intervalo de confianza:

```
nivel_confianza = 0.95
t.test(nivel_tcdd_plasma,nivel_tcdd_tejido, alternative = 'two.sided', conf.level = nivel_confianza)
```

```
##
##  Welch Two Sample t-test
##
## data:  nivel_tcdd_plasma and nivel_tcdd_tejido
## t = -0.33153, df = 37.937, p-value = 0.7421
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -6.182749  4.442749
## sample estimates:
## mean of x mean of y
##      5.99      6.86
```

Los resultados muestran que el intervalo de confianza del 95% para la diferencia entre las medias verdaderas de los niveles de TCDD en plasma y tejido adiposo está entre -6.18 y 4.442. De esto se puede inferir que, en promedio, los niveles de TCDD en plasma son menores que los niveles de TCDD en tejido adiposo. Sin embargo, debido a que el intervalo de confianza incluye el cero, no se puede afirmar con certeza que haya una diferencia significativa entre las medias verdaderas de ambos tipos de muestras. Tambien como el valor de

$$p_{value} = 0.7421 > \alpha$$

, no se descarta la hipotesis y se la toma como verdadera.

Punto 3

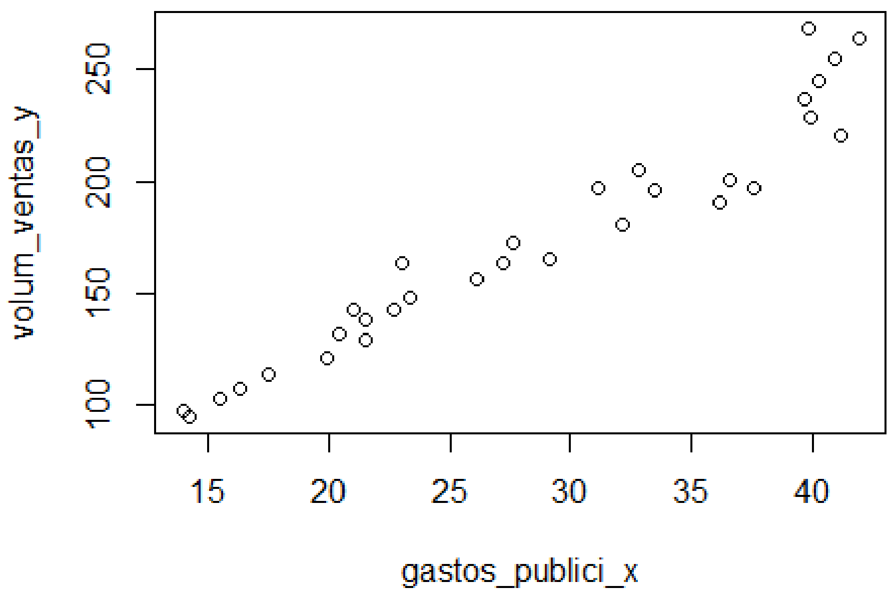
En primer lugar se almacena los datos en un data frame para hacer el tratamiento y graficar los datos.

```
gastos_publici_x = c(14.2226,
13.9336,15.5040,16.3105,17.4936,19.8906,21.4803,20.4046,21.4776,22.6821,20.9722,23.3538,26.1040,29.1101,27.2418,23.0096,27.6116,32.1111,36.1788,37
```

```
.5671,33.5069,36.6088,31.1554,32.7752,41.1886,39.9715,39.6866,40.2991,40.
9538,41.9323,39.8393)
volum_ventas_y =
c(95.065,97.281,103.159,107.607,113.860,121.153,129.102,132.340,138.663,1
42.856,143.120,147.928,155.955,164.946,163.921,163.426,172.485,180.519,19
0.509,196.497,196.024,200.832,196.769,205.341,220.230,228.703,236.500,244
.560,254.771,263.683,268.304)
data = data.frame(gastos_publici_x, volum_ventas_y)
```

Con la funcion plot graficamos los datos para ver su comportamiento de una forma visual, dando a entender que tienen una tendencia lineal, pero aun falta hacer mas pruebas.

```
plot(data)
```



a) Regresion lineal simple

Se implementa el primer modelo con una regresion lineal simple:

```
# Modelo de regresión lineal simple
model1 = lm(volum_ventas_y ~ gastos_publici_x, data)
# Mostrar coeficientes y estadísticas del modelo
summary(model1)

##
## Call:
## lm(formula = volum_ventas_y ~ gastos_publici_x, data = data)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.121  -5.945  -0.590   6.176  34.562
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       21.1667     7.6873   2.753   0.0101 *
## gastos_publici_x   5.3358     0.2568  20.779   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.94 on 29 degrees of freedom
## Multiple R-squared:  0.9371, Adjusted R-squared:  0.9349
## F-statistic: 431.8 on 1 and 29 DF,  p-value: < 2.2e-16
```

Como se observa en la tabla anova, la ecuacion del modelo es

$$volum_{ventas} = 5.3358 * gastos_{public} + 21.1667$$

. Tambien se puede observar que

$$p_{valor} = 2 * 10^{-16} < \alpha$$

, dando a entender que hay una asociacion de los datos y el modelo. El valor de $R^2 = 0.9371$ tendiendo a 1, lo que dice que los valores pronosticados por el modelo se ajustan a los valores reales observados.

Pruebas

Con el modelo desarrollado empieza la fase de pruebas.Se muestra en la tabla los valores de y, donde se observa los valores pronosticados por el modelo y los reales con el error en cada dato.

```
datos = data.frame(gastos_publici_x,
volum_ventas_y,model1$fitted.values,model1$residuals)
datos

##    gastos_publici_x volum_ventas_y model1.fitted.values
model1.residuals
## 1           14.2226         95.065             97.05589       -
1.9908939
## 2           13.9336         97.281             95.51384
1.7671570
## 3           15.5040        103.159            103.89321       -
0.7342088
## 4           16.3105        107.607            108.19654       -
0.5895447
## 5           17.4936        113.860            114.50935       -
0.6493489
## 6           19.8906        121.153            127.29930       -
```

```
6.1463005
## 7              21.4803        129.102            135.78165       -
6.6796477
## 8              20.4046        132.340            130.04191
2.2980899
## 9              21.4776        138.663            135.76724
2.8957590
## 10             22.6821        142.856            142.19423
0.6617683
## 11             20.9722        143.120            133.07052
10.0494806
## 12             23.3538        147.928            145.77830
2.1497006
## 13             26.1040        155.955            160.45286       -
4.4978614
## 14             29.1101        164.946            176.49286      -
11.5468587
## 15             27.2418        163.921            166.52395       -
2.6029531
## 16             23.0096        163.426            143.94171
19.4842885
## 17             27.6116        172.485            168.49714
3.9878620
## 18             32.1111        180.519            192.50564      -
11.9866433
## 19             36.1788        190.509            214.21014      -
23.7011432
## 20             37.5671        196.497            221.61786      -
25.1208569
## 21             33.5069        196.024            199.95338       -
3.9293757
## 22             36.6088        200.832            216.50454      -
15.6725442
## 23             31.1554        196.769            187.40620
9.3627963
## 24             32.7752        205.341            196.04916
9.2918411
## 25             41.1886        220.230            240.94152      -
20.7115156
## 26             39.9715        228.703            234.44729       -
5.7442936
## 27             39.6866        236.500            232.92712
3.5728805
## 28             40.2991        244.560            236.19531
8.3646930
## 29             40.9538        254.771            239.68867
15.0823341
## 30             41.9323        263.683            244.90976
18.7732379
```

```
## 31          39.8393         268.304             233.74190
34.5621013
```

Hay valores con un margen de error elevado, como los son los datos de la fila 6,7,19,20, entre otros.

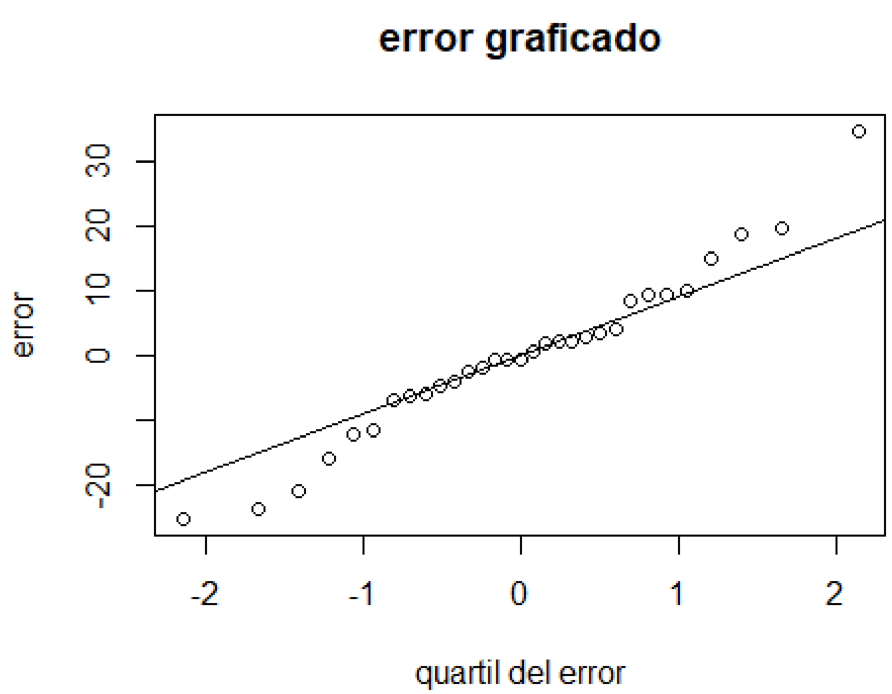
Validacion de suspuestos

Los errores aleatorios suelen distribuirse normalmente, son independientes o tienen igual varianza (homoscedasticidad). Por lo cual se va a validar con 3 analisis:

Analisis residuales

Para este test se trabaja con los errores dados en cada punto con respecto los $value_{reales} - value_{modelo}$

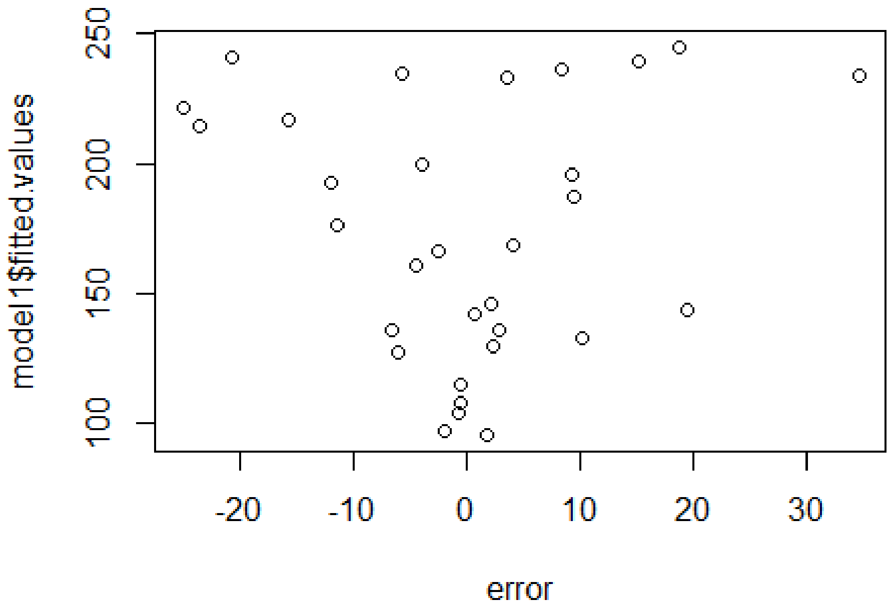
```
error = datos$model1.residuals
qqnorm(error, main = 'error graficado', xlab = 'quartil del error',ylab =
'error')
qqline(error)
```




```
hist(error)
```

La tendencia de los errores deben alinearse a la linea, pero se observa que estos empiezan a dispersarse por lo cual si pasa esta validacion. El error se acumula entre los valores de -10 a 10.

```
plot(error, model1$fitted.values)
```



La grafica no muestra una tendencia lineal o patron, por lo cual pasa este test.

Prueba de normalidad

Para este test se aplica la prueba de shapiro:

```
shapiro.test(error)

##
##  Shapiro-Wilk normality test
##
## data:  error
## W = 0.96954, p-value = 0.5064

library(lmtest)

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

bptest(model1, studentize = FALSE)

##
##  Breusch-Pagan test
```

```
##
## data:  model1
## BP = 11.121, df = 1, p-value = 0.0008536
```

Para pasar la prueba de Shapiro el valor de $p_{valor} > \alpha$ y para este caso $0.504 > 0.05$, por lo cual pasa este test. Para pasar la prueba de Breush Pagan el valor de $p_{valor} > \alpha$ y para este caso $0.0008536 < 0.05$, por lo cual NO pasa este test.

Prueba de Homocedasticidad

Para este test, se aplica la prueba de Durvin Wattson.

```
dwtest(model1, alternative='two.sided')

##
##  Durbin-Watson test
##
## data:  model1
## DW = 0.89343, p-value = 0.0003002
## alternative hypothesis: true autocorrelation is not 0
```

Para pasar esta prueba $p_{valor} > \alpha$ y para este caso $0.0003002 < 0.05$, por lo cual No pasa este test.

Conclusión

Este modelo no pasa al no cumplir con los requerimientos de varios test planteados.

b) Regresion polinomica grado 2

Se implementa el segundo modelo con una regresion polinomica de grado dos:

```
model2 = lm(volum_ventas_y ~ gastos_publici_x + I(gastos_publici_x^2),
data)
summary(model2)

##
## Call:
## lm(formula = volum_ventas_y ~ gastos_publici_x +
I(gastos_publici_x^2),
##     data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.111  -5.878  -1.735   6.511  33.317
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            41.05144   27.44164   1.496   0.1459
## gastos_publici_x        3.78618    2.06787   1.831   0.0778 .
## I(gastos_publici_x^2)   0.02715    0.03595   0.755   0.4564
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.04 on 28 degrees of freedom
## Multiple R-squared:  0.9383, Adjusted R-squared:  0.9339
## F-statistic:   213 on 2 and 28 DF,  p-value: < 2.2e-16
```

Como se observa en la tabla anova, la ecuacion del modelo es

$$volum_{ventas} = 0.02715 * gastos_{public}^2 + 3.78618 * gastos_{public} + 21.1667$$

. Tambien se puede observar que $P_{valor} = 2.2*10^{-16} < $, dando a entender que hay una asociacion de los datos y el modelo. El valor de $R^2 = 0.9383$ tendiendo a 1, lo que dice que los valores pronosticados por el modelo se ajustan a los valores reales observados.

Pruebas

Con el modelo desarrollado empieza la fase de pruebas.Se muestra en la tabla los valores de y, donde se observa los valores pronosticados y los reales con el error en cada dato.

```
datos = data.frame(gastos_publici_x,
volum_ventas_y,model2$fitted.values,model2$residuals)
datos
```

```
##    gastos_publici_x volum_ventas_y model2.fitted.values
model2.residuals
## 1           14.2226         95.065            100.39336       -
5.3283637
## 2           13.9336         97.281             99.07821       -
1.7972080
## 3           15.5040        103.159            106.27929       -
3.1202862
## 4           16.3105        107.607            110.02955       -
2.4225486
## 5           17.4936        113.860            115.59493       -
1.7349346
## 6           19.8906        121.153            127.10361       -
5.9506078
## 7           21.4803        129.102            134.90829       -
5.8062943
## 8           20.4046        132.340            129.61210
2.7279049
## 9           21.4776        138.663            134.89492
3.7680778
## 10          22.6821        142.856            140.89967
1.9563330
## 11          20.9722        143.120            132.39884
10.7211624
## 12          23.3538        147.928            144.28249
3.6455146
```

```
## 13          26.1040        155.955             158.38859       -
2.4335917
## 14          29.1101        164.946             174.27710       -
9.3311009
## 15          27.2418        163.921             164.34462       -
0.4236245
## 16          23.0096        163.426             142.54596
20.8800370
## 17          27.6116        172.485             166.29555
6.1894481
## 18          32.1111        180.519             190.62816      -
10.1091552
## 19          36.1788        190.509             213.57191      -
23.0629104
## 20          37.5671        196.497             221.60825      -
25.1112546
## 21          33.5069        196.024             198.39986       -
2.3758620
## 22          36.6088        200.832             216.04983      -
15.2178289
## 23          31.1554        196.769             185.36791
11.4010863
## 24          32.7752        205.341             194.31262
11.0283816
## 25          41.1886        220.230             243.06439      -
22.8343918
## 26          39.9715        228.703             235.77403       -
7.0710332
## 27          39.6866        236.500             234.07912
2.4208819
## 28          40.2991        244.560             237.72843
6.8315742
## 29          40.9538        254.771             241.65169
13.1193093
## 30          41.9323        263.683             247.55871
16.1242920
## 31          39.8393        268.304             234.98701
33.3169936
```

Hay valores con un margen de error elevado, como los son los datos de la fila 6,7,16,19,30, entre otros.

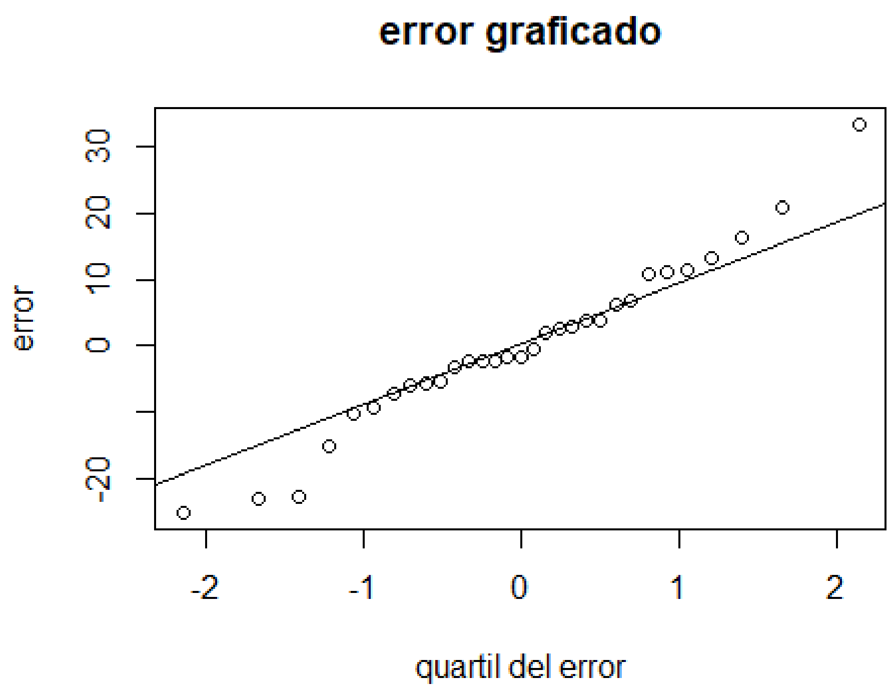
Validacion de suspuestos

Los errores aleatorios suelen distribuirse normalmente, son independientes o tienen igual varianza (homoscedasticidad). Por lo cual se va a validar con 3 analisis:

Analisis residuales

Para este test se trabaja con los errores dados en cada punto con respecto los $value_{reales} - value_{modelo}$

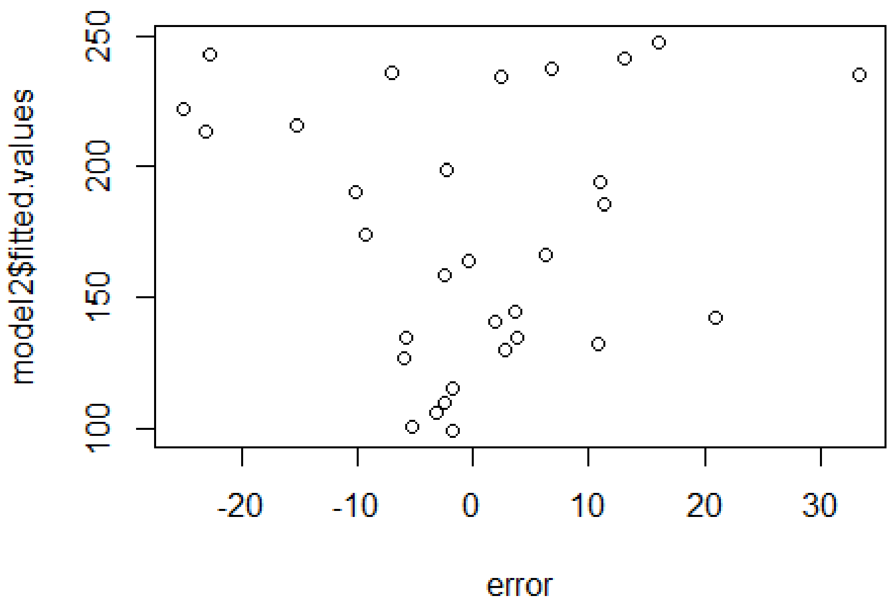
```
error = datos$model2.residuals
qqnorm(error, main = 'error graficado', xlab = 'quartil del error',ylab =
'error')
qqline(error)
```




```
hist(error)
```

La tendencia de los errores deben alinearse a la linea, pero se observa que estos empiezan a dispersarse por lo cual si pasa esta validacion. El error se acumula entre los valores de -10 a 10.

```
plot(error, model2$fitted.values)
```



La grafica no muestra una tendencia lineal o patron, por lo cual pasa este test.

Prueba de normalidad

Para este test se aplica la prueba de shapiro:

```
shapiro.test(error)

##
##  Shapiro-Wilk normality test
##
## data:  error
## W = 0.97044, p-value = 0.5312

library(lmtest)
bptest(model2, studentize = FALSE)

##
##  Breusch-Pagan test
##
## data:  model2
## BP = 10.271, df = 2, p-value = 0.005885
```

Para pasar la prueba de Shapiro el valor de $p_{valor} > \alpha$ y para este caso $0.5064 > 0.05$, por lo cual pasa este test, los errores se distriuyen de manera normal. Para pasar la prueba de Breush Pagan el valor de $p_{valor} > \alpha$ y para este caso $0.005885 > 0.05$, por lo cual pasa este test, los errores en varianza son constantes.

Prueba de Homocedasticidad

Para este test, se aplica la prueba de Durvin Wattson.

```
dwtest(model2, alternative='two.sided')

##
##  Durbin-Watson test
##
## data:  model2
## DW = 0.99738, p-value = 0.0005566
## alternative hypothesis: true autocorrelation is not 0
```

Para pasar esta prueba $p_{valor} > \alpha$ y para este caso $0.0005566 < 0.05$, por lo cual NO pasa este test, el error es dependiente.

Conclusión

Este modelo no pasa al no cumplir la prueba de Homocedasticidad, en donde el error es dependiente.

c) Regresión con transformación logarítmica

Se implementa el ultimo modelo con una transformacion logaritmica:

```
model3 = lm(log(volum_ventas_y) ~ gastos_publici_x, data)
summary(model3)

##
## Call:
## lm(formula = log(volum_ventas_y) ~ gastos_publici_x, data = data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.122298 -0.044668  0.006683  0.040461  0.160176
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.201071   0.042528   98.78   <2e-16 ***
## gastos_publici_x 0.031948   0.001421   22.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0716 on 29 degrees of freedom
## Multiple R-squared:  0.9458, Adjusted R-squared:  0.9439
## F-statistic: 505.7 on 1 and 29 DF,  p-value: < 2.2e-16
```

Como se observa en la tabla anova, la ecuacion del modelo es

$$log(volum_{ventas}) = 0.031948 * gastos_{public} + 4.201071$$

. Tambien se puede observar que $P_{valor} = 2.2*10^{-16} < $, dando a entender que hay una asociacion de los datos y el modelo. El valor de $R^2 = 0.9458$ tendiendo a 1, lo que dice que los valores pronosticados por el modelo se ajustan a los valores reales observados.

Pruebas

Con el modelo desarrollado empieza la fase de pruebas.Se muestra en la tabla los valores de y, donde se observa los valores pronosticados y los reales con el error en cada dato.

```
datos = data.frame(gastos_publici_x,
log(volum_ventas_y),model3$fitted.values,model3$residuals)
datos
```

```
##    gastos_publici_x log.volum_ventas_y. model3.fitted.values
model3.residuals
## 1           14.2226            4.554561             4.655456     -
0.100895426
## 2           13.9336            4.577604             4.646223     -
0.068619583
## 3           15.5040            4.636271             4.696395     -
0.060123160
## 4           16.3105            4.678486             4.722161     -
0.043675125
## 5           17.4936            4.734970             4.759959     -
0.024989053
## 6           19.8906            4.797054             4.836538     -
0.039484170
## 7           21.4803            4.860603             4.887326     -
0.026723556
## 8           20.4046            4.885374             4.852960
0.032414643
## 9           21.4776            4.932047             4.887240
0.044806444
## 10          22.6821            4.961837             4.925722
0.036115504
## 11          20.9722            4.963683             4.871093
0.092589946
## 12          23.3538            4.996726             4.947181
0.049544474
## 13          26.1040            5.049568             5.035045
0.014522521
## 14          29.1101            5.105618             5.131084     -
0.025466154
## 15          27.2418            5.099385             5.071396
0.027989022
## 16          23.0096            5.096360             4.936185
0.160175645
## 17          27.6116            5.150310             5.083210
```

```
0.067100270
## 18          32.1111            5.195836            5.226961     -
0.031124649
## 19          36.1788            5.249699            5.356916     -
0.107216715
## 20          37.5671            5.280647            5.401270     -
0.120622598
## 21          33.5069            5.278237            5.271554
0.006683196
## 22          36.6088            5.302469            5.370654     -
0.068185118
## 23          31.1554            5.282030            5.196428
0.085602610
## 24          32.7752            5.324672            5.248177
0.076494565
## 25          41.1886            5.394672            5.516970     -
0.122297514
## 26          39.9715            5.432424            5.478086     -
0.045661663
## 27          39.6866            5.465948            5.468984     -
0.003035648
## 28          40.2991            5.499461            5.488552
0.010908583
## 29          40.9538            5.540365            5.509469
0.030896558
## 30          41.9323            5.574748            5.540730
0.034017820
## 31          39.8393            5.592121            5.473862
0.118258329
```

Hay valores con un margen de error relativamente pequeño.

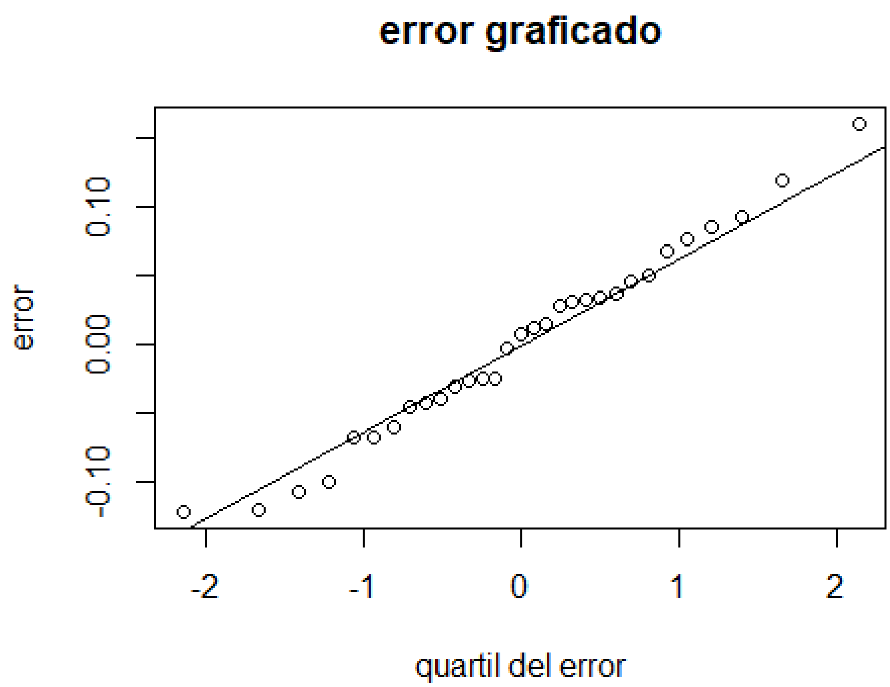
Validacion de suspuestos

Los errores aleatorios suelen distribuirse normalmente, son independientes o tienen igual varianza (homoscedasticidad). Por lo cual se va a validar con 3 analisis:

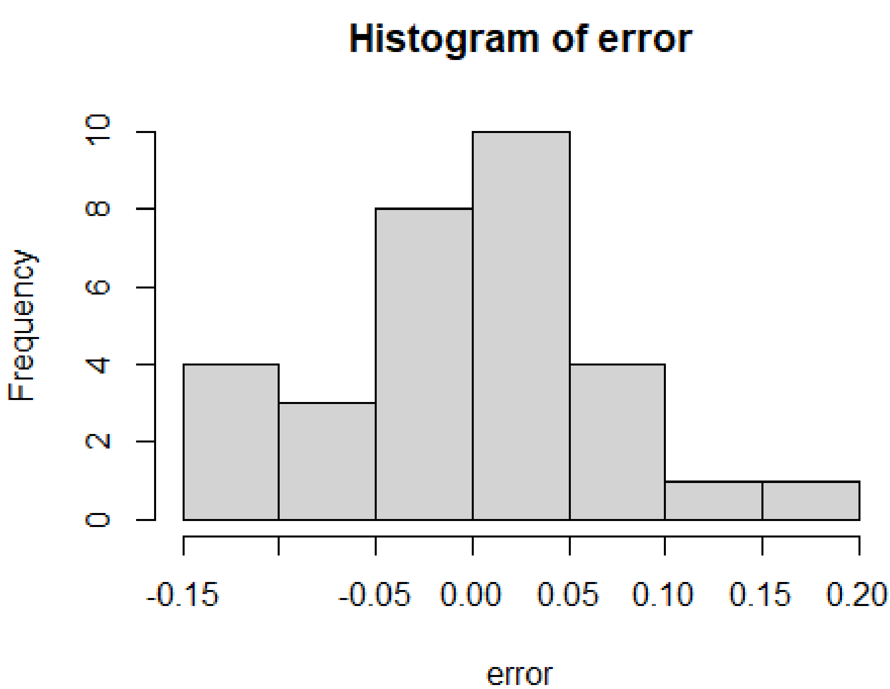
Analisis residuales

Para este test se trabaja con los errores dados en cada punto con respecto los $value_{reales} - value_{modelo}$

```
error = datos$model3.residuals
qqnorm(error, main = 'error graficado', xlab = 'quartil del error',ylab =
'error')
qqline(error)
```

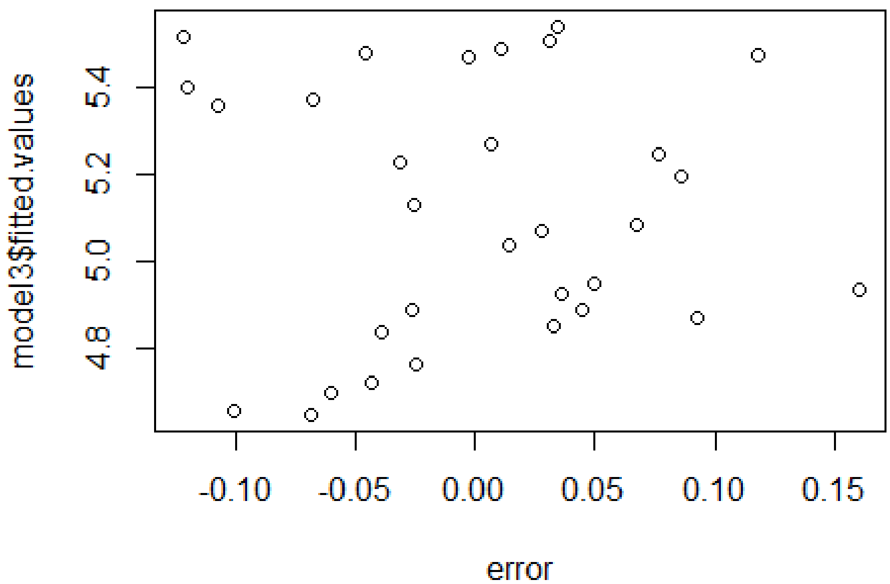



```
hist(error)
```



La tendencia de los errores deben alinearse a la linea y es el patron que comienza a seguir, por lo cual si pasa esta validacion. El error se acumula entre los valores de +-0.05.

```
plot(error, model3$fitted.values)
```



La grafica no muestra una tendencia lineal o patron, por lo cual pasa este test.

Prueba de normalidad

Para este test se aplica la prueba de shapiro:

```
shapiro.test(error)

##
##  Shapiro-Wilk normality test
##
## data:  error
## W = 0.98262, p-value = 0.8808

library(lmtest)
bptest(model3, studentize = FALSE)

##
##  Breusch-Pagan test
##
## data:  model3
## BP = 0.16626, df = 1, p-value = 0.6835
```

Para pasar la prueba de Shapiro el valor de $p_{valor} > \alpha$ y para este caso $0.8808 > 0.05$, por lo cual pasa este test, los errores se distriuyen de manera normal. Para pasar la prueba de Breush Pagan el valor de $p_{valor} > \alpha$ y para este caso $0.6835 > 0.05$, por lo cual pasa este test, los errores en varianza son constantes.

Prueba de Homocedasticidad

Para este test, se aplica la prueba de Durvin Wattson:

```
dwtest(model3, alternative='two.sided')
```

```
##
##  Durbin-Watson test
##
## data:  model3
## DW = 1.068, p-value = 0.002796
## alternative hypothesis: true autocorrelation is not 0
```

Para pasar esta prueba $p_{valor} > \alpha$ y para este caso $0.002796 < 0.05$, por lo cual NO pasa este test, el error es dependiente.

Conclusión

Este modelo no pasa al no cumplir la prueba de Homocedasticidad, en donde el error es dependiente.

Comparacion de modelos

La AIC (Akaike Information Criterion) es una medida utilizada en la selección de modelos estadísticos que tiene en cuenta tanto la bondad de ajuste del modelo como la complejidad del mismo. Es una herramienta que permite comparar distintos modelos y elegir el que mejor se ajusta a los datos, siendo la opción con menor valor de AIC la preferida. Para el caso segun esta metrica seria el modelo 3, que se basa en una transformacion logaritmica.

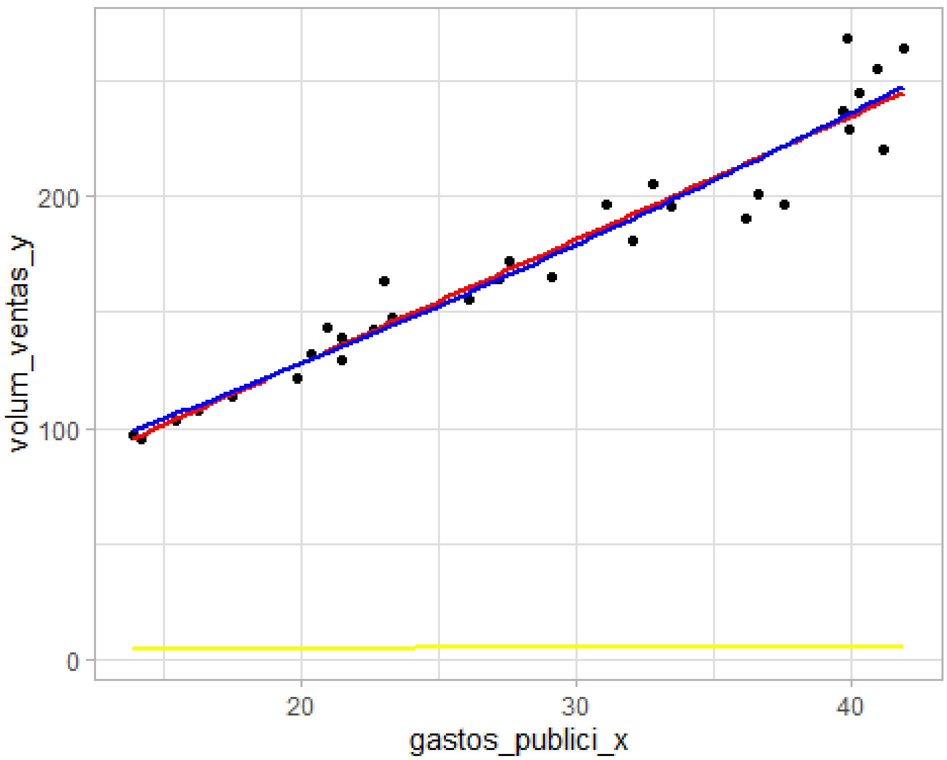
```
# Comparación de modelos
AIC(model1, model2, model3)
```

```
##        df       AIC
## model1  3 250.65457
## model2  4 252.02928
## model3  3 -71.56957
```

En las pruebas realizadas los modelos mas completos y que pasaron la mayoria de pruebas son el modelo 3 (logaritmico) y el modelo 2 (polinomico), y dando unos mejore valores de correlacion. Asi se escoger el menos peor de estos modelos seria el modelo 3.

```
library(ggplot2)
ggplot(datos, aes(x=gastos_publici_x, y=volum_ventas_y))+
  geom_point()+
  geom_smooth(method = 'lm', formula = y~x, se = FALSE, col= 'red' )+
  geom_smooth(method = 'lm', formula = y~x+I(x^2), se = FALSE, col=
```

```
'blue' )+
  geom_smooth(method = 'lm', formula = log(y)~x, se = FALSE, col=
'yellow' )+
  theme_light()
```



Por ultimo se muestra los modelos creados y analizados para este ejercicio. Algo que falto decir es que falto hacer una conversion a la inversa para tener los datos del modelo 3 en escala normal y no logaritmica.

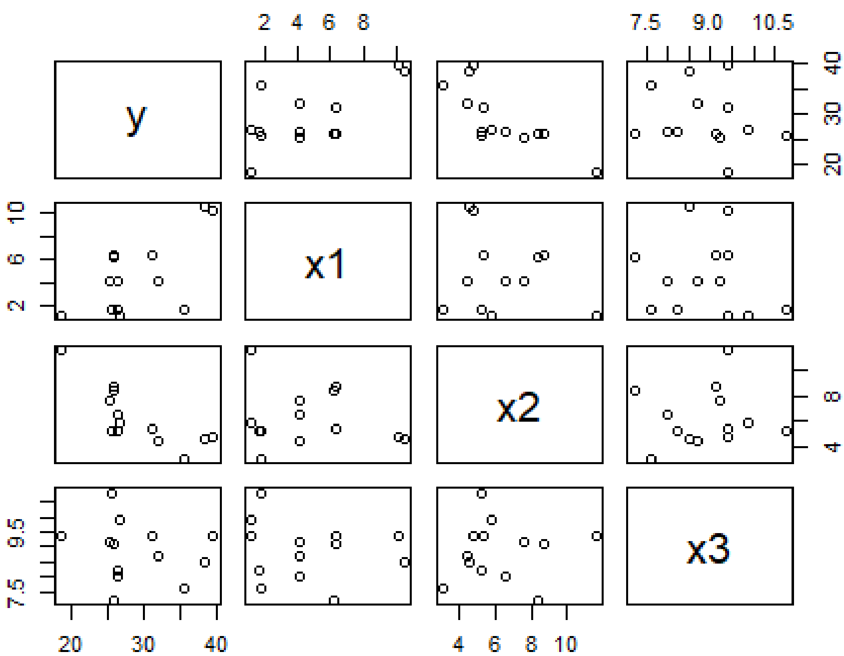
punto 4

Como primer paso, se hace un almacenamiento de la informacion y se grafica la informacion para dar una aproximacion de las variables y ver cual puede tener una tendencia lineal.

```
y = c(25.5,31.2,25.9,38.4,18.4,26.7,26.4,25.9,32.0,25.2,39.7,35.7,26.5)
x1 =
c(1.74,6.32,6.22,10.52,1.19,1.22,4.10,6.32,4.08,4.15,10.15,1.72,1.70)
x2 = c(5.30,5.42,8.41,4.63,11.60,5.85,6.62,8.72,4.42,7.60,4.83,3.12,5.30)
x3 = c(10.80,9.40,7.20,8.50,9.40,9.90,8,9.10,8.70,9.20,9.40,7.60,8.20)
datos = data.frame(y,x1,x2,x3)
datos

##       y    x1    x2   x3
## 1  25.5  1.74  5.30 10.8
## 2  31.2  6.32  5.42  9.4
```

```
## 3   25.9  6.22  8.41  7.2
## 4   38.4 10.52  4.63  8.5
## 5   18.4  1.19 11.60  9.4
## 6   26.7  1.22  5.85  9.9
## 7   26.4  4.10  6.62  8.0
## 8   25.9  6.32  8.72  9.1
## 9   32.0  4.08  4.42  8.7
## 10 25.2  4.15  7.60  9.2
## 11 39.7 10.15  4.83  9.4
## 12 35.7  1.72  3.12  7.6
## 13 26.5  1.70  5.30  8.2
```

```
plot(datos)
```



Segun lo visto en la grafica, no se ve alguna tendencia lineal en los datos.

El código define los datos para el análisis de regresión lineal con las 3 variables independientes. Luego, crea un modelo de regresion multiple utilizando la función lm. La función summary se utiliza para ver el resumen estadístico del modelo.

```
model = lm(y ~ x1 + x2 + x3, data = datos)
summary(model)

##
## Call:
## lm(formula = y ~ x1 + x2 + x3, data = datos)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -1.8532 -1.4495 -0.3219  0.5919  3.2121
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  39.1573     5.8871   6.651 9.36e-05 ***
## x1            1.0161     0.1909   5.323 0.000479 ***
## x2           -1.8616     0.2673  -6.964 6.58e-05 ***
## x3           -0.3433     0.6171  -0.556 0.591572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.073 on 9 degrees of freedom
## Multiple R-squared:  0.9117, Adjusted R-squared:  0.8823
## F-statistic: 30.98 on 3 and 9 DF,  p-value: 4.496e-05
```

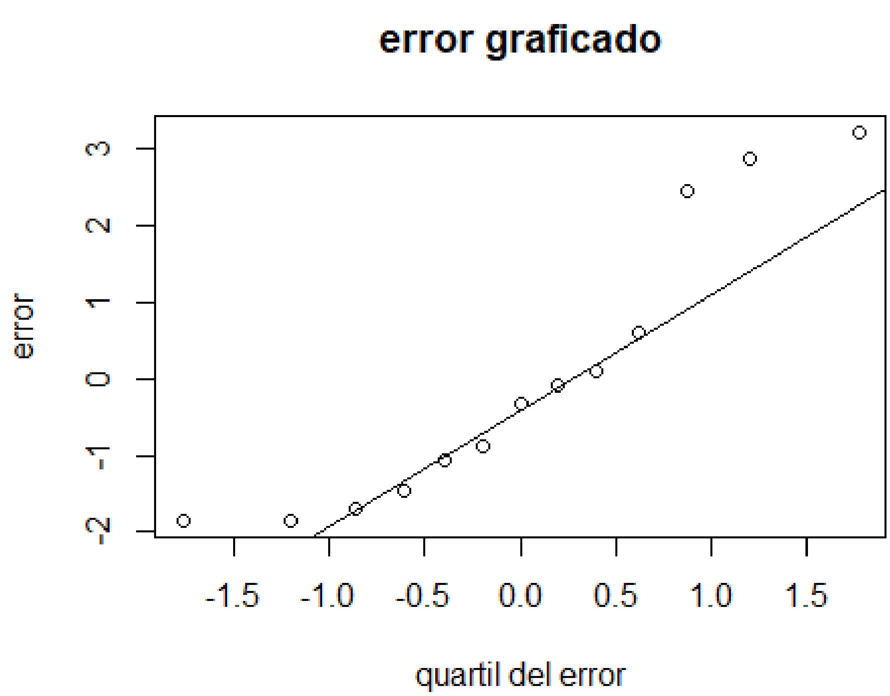
La ecuaion seria la siguiente

$$y = 1.0161 * x_1 - 1.8616 * x_2 - 0.3433 * x_3 + 39.1573$$

Se tiene un correlacion de datos con el modelo de $R^2 = 0.9117$.

Se realiza las pruebas y validacion de supuestos para ver el estado del modelo.

```
error = model$residuals
qqnorm(error, main = 'error graficado', xlab = 'quartil del error',ylab =
'error')
qqline(error)
```



```
hist(error)
```


```
shapiro.test(error)

##
##  Shapiro-Wilk normality test
##
## data:  error
## W = 0.86898, p-value = 0.05068

dwtest(model, alternative='two.sided')

##
##  Durbin-Watson test
##
## data:  model
## DW = 1.5677, p-value = 0.4112
## alternative hypothesis: true autocorrelation is not 0
```

El modelo pasa las pruebas al haber una tendencia lineal del error de los datos, y Shapiro y Durvin son mayo a alpha.

Ajuste del modelo

Como se requiere saber que variables se puede retirar, se va a utilizar dos metodos. Para hallar el modelo reducido vamos a utilizar dos formas diferentes; (i) el metodo 1 uno con una libreria de multicolinealidad y (ii) el metodo 2 con los una libreria de correlacion.

Metodo 1

La libreria car se carga para usar la función vif, que calcula el factor de inflación de varianza para cada variable en el modelo. Un VIF alto indica multicolinealidad con las otras variables, lo que puede afectar la interpretación del modelo. En este caso, $x1$ tiene un alto VIF y se elimina del modelo.

```
library(car)

## Warning: package 'car' was built under R version 4.2.3

## Loading required package: carData

## Warning: package 'carData' was built under R version 4.2.3

vif(model)

##       x1       x2       x3 
## 1.043348 1.027098 1.024503
```

Se crea un nuevo modelo lm denominado model2, sin la variable x1. La función summary se utiliza nuevamente para ver el resumen estadístico del nuevo modelo.

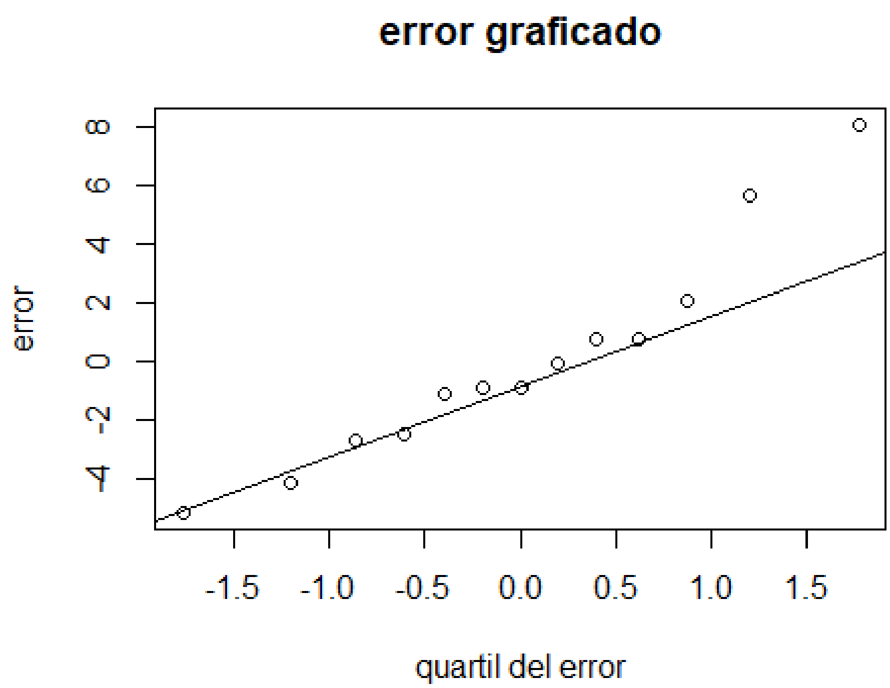
```
model2 = lm(y ~ x2 + x3, data = datos)
summary(model2)

## 
## Call:
## lm(formula = y ~ x2 + x3, data = datos)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -5.1272 -2.4488 -0.8833  0.7677  8.0480 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)   
## (Intercept)  49.0541    10.7925   4.545  0.00107 **
## x2           -2.0674     0.5111  -4.045  0.00234 **
## x3           -0.7890     1.1812  -0.668  0.51927   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.005 on 10 degrees of freedom
## Multiple R-squared:  0.6338, Adjusted R-squared:  0.5606 
## F-statistic: 8.655 on 2 and 10 DF,  p-value: 0.006583
```

La ecuaion seria la siguiente

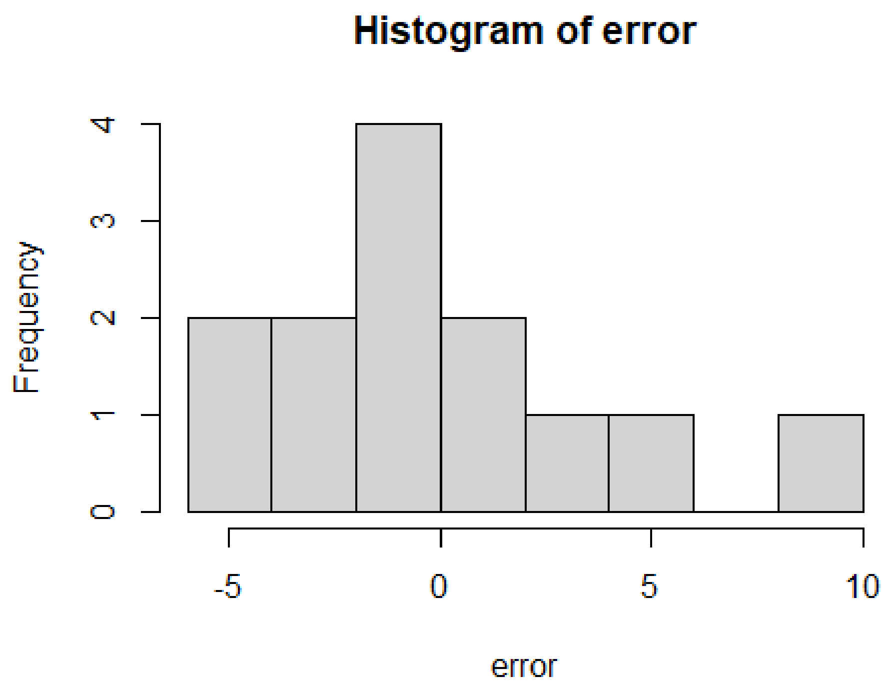
$$y = -2.0674 * x_2 - 0.7890 * x_3 + 49.0541$$

Se realiza las pruebas y validacion de supuestos para ver el estado del modelo.

```
error = model2$residuals
qqnorm(error, main = 'error graficado', xlab = 'quartil del error',ylab =
'error')
qqline(error)
```




```
hist(error)
```




```
shapiro.test(error)

##
##  Shapiro-Wilk normality test
##
## data:  error
## W = 0.92959, p-value = 0.3366

dwtest(model2, alternative='two.sided')

##
##  Durbin-Watson test
##
## data:  model2
## DW = 1.873, p-value = 0.8253
## alternative hypothesis: true autocorrelation is not 0
```

Pasa las pruebas de Shapiro y Durvin pero se tiene un R^2 muy bajo.

Metodo 2

Se va a realizar un ajuste del modelo por lo cual se va a calcular la correlacion de y respecto a x_i. Se eliminara la variable que tenga menor correlacion.

```
r_x1 = cor(x1,y)
r_x1

## [1] 0.6538538
```

```
r_x2 = cor(x2,y)
r_x2

## [1] -0.785805

r_x3 = cor(x3,y)
r_x3

## [1] -0.186275
```

Como se puede observar el que maneja una menor correlacion es la variable x_2, por lo cual la eliminamos $x_2$del modelo y generamos un nuevo modelo.

```
model3 = lm(y ~ x1 + x3, data = datos)
summary(model3)

##
## Call:
## lm(formula = y ~ x1 + x3, data = datos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.2522 -3.1436  0.5424  2.2986  9.3737
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  28.6123    13.6414   2.097   0.0623 .
## x1            1.2083     0.4529   2.668   0.0236 *
## x3           -0.5742     1.4775  -0.389   0.7057
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.971 on 10 degrees of freedom
## Multiple R-squared:  0.436,  Adjusted R-squared:  0.3233
## F-statistic: 3.866 on 2 and 10 DF,  p-value: 0.05705
```

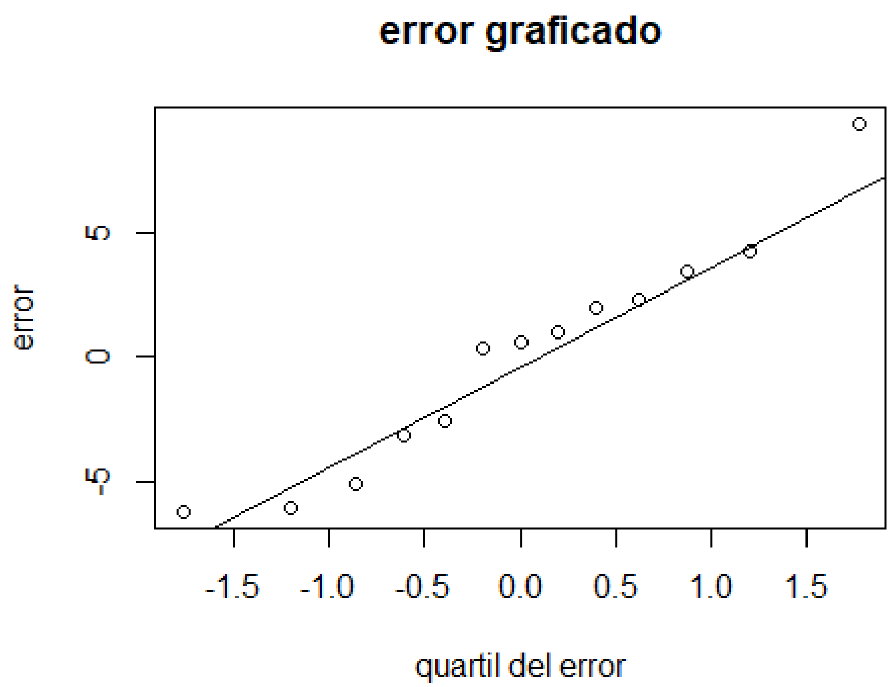
La ecuaion seria la siguiente

$$y = 1.2083 * x_1 - 0.5742 * x_3 + 28.6123$$

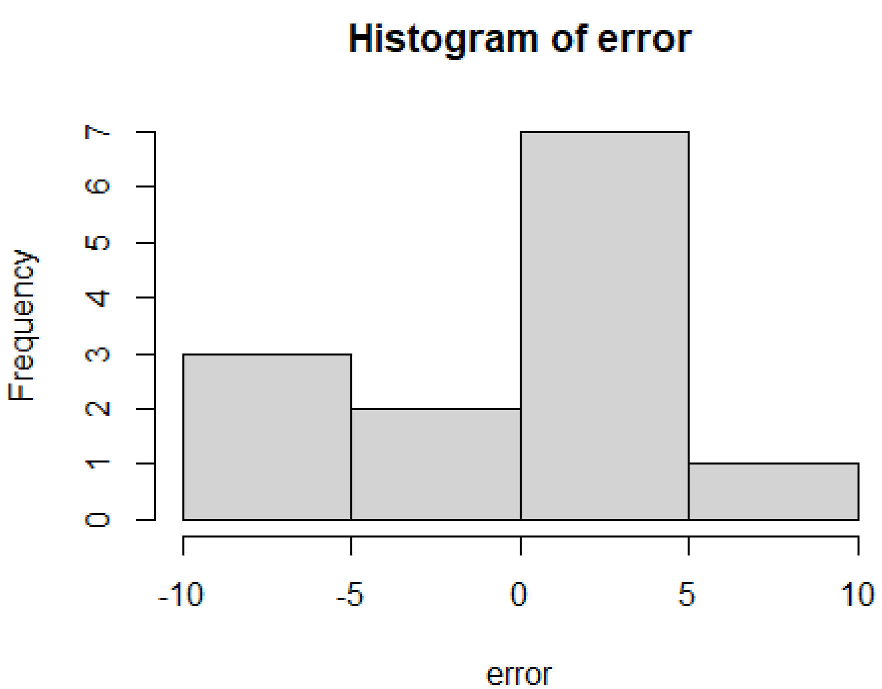
Se tiene un correlacion de datos con el modelo de $R^2 = 0.436$, siendo muy bajo.

Se realiza las pruebas y validacion de supuestos para ver el estado del modelo.

```
error = model3$residuals
qqnorm(error, main = 'error graficado', xlab = 'quartil del error',ylab =
'error')
qqline(error)
```




```
hist(error)
```




```
shapiro.test(error)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  error
## W = 0.95146, p-value = 0.6205
```

```
dwtest(model3, alternative='two.sided')
```

```
##
##  Durbin-Watson test
##
## data:  model3
## DW = 2.2395, p-value = 0.7046
## alternative hypothesis: true autocorrelation is not 0
```

El modelo pasa las pruebas al haber una tendencia lineal del error de los datos, y Shapiro y Durvin son mayo a alpha.

Conclusion

Se aplicaron dos metodologias distintas y la que da un mejor modelo eliminando una variable independiente es la metodologia 1, en el que genera un mejor R_2 y pasa los test.