REGRESION LINEAL MULTIPLE TOTALES MENSUALES

Miguel Angel Villegas

2025-03-11

*library(tsDyn)

*library(tidyverse)

*library(dplyr)

*library(readxl)

*library(EnvStats)

*library(corrplot)

*library(caTools)

library(GGally)

*library(forecast)

Introducción

Este modelo abarca todos las ventas de los productos, se utilizan las variables: Valor Unitario y Cantidad. Se realizan las correlaciones entre las tres variables y se determina el modelo lineal múltiple. El conjunto de entrenamiento y prueba esta dividido en una proporción de 80/20, sin embargo, la división es aleatoria,lo hace que el resultado sea mas confiable. Se siembra una semilla para permitir que los valores de la muestra sean los mismos.

```
ruta <- "/cloud/project/Ventas_Suministros_Totales.xlsx"
excel_sheets(ruta)
```

```
## [1] "Ventas Totales Original"    "Servicios Totales Original"
```

```
# "Ventas Totales Original"    "Servicios Totales Original"
```

```
Productos_Totales <- as.data.frame(read_xlsx(ruta,
                                             sheet = "Ventas Totales Original"))
Productos_Totales$Semana <- format(Productos_Totales$Fecha, format = "%Y-%U")
Productos_Totales$mes <- format(Productos_Totales$Fecha, format = "%Y-%m")

Servicios_Totales <- as.data.frame(read_xlsx(ruta,
                                             sheet = "Servicios Totales Original"))
Servicios_Totales$Semana <- format(Servicios_Totales$Fecha, format = "%Y-%U")
Servicios_Totales$mes <- format(Servicios_Totales$Fecha, format = "%Y-%m")

datatotal <- merge(x = Productos_Totales, y = Servicios_Totales, all = T)
colnames(datatotal) <- c("Indice", "Fecha", "RFC", "Empresa", "Cantidad",
                         "Pieza", "Descripcion", "ValorUnitario", "Total",
                         "Semana", "Mes" )
```

Se calcula el valor del parámetro lambda para la transformación Boxcox.

```
RLM_dftotal_mes <- datatotal %>%
  group_by(mes = as.character(Mes)) %>%
  summarize(Ventas_Totales = sum(Total),
            Ventas_Unitario = sum(ValorUnitario),
            Ventas_Cantidad = sum(Cantidad),
            .groups = "keep")
head(RLM_dftotal_mes)
```

```
## # A tibble: 6 x 4
## # Groups:   mes [6]
##   mes     Ventas_Totales Ventas_Unitario Ventas_Cantidad
##   <chr>            <dbl>           <dbl>           <dbl>
## 1 2019-07        227086.          93563.             345
## 2 2019-08        191359.          68052.             187
## 3 2019-09        468455.          66960.             908
## 4 2019-10        410463.          79738.             762
## 5 2019-11        333246.          58947.             502
## 6 2019-12        254851.         109719.             277
```

```
nrow(RLM_dftotal_mes)
```

```
## [1] 60
```

```
RLM_dftotal_mes <- as.data.frame(RLM_dftotal_mes)
head(RLM_dftotal_mes)# Colocar fechas a las semana en Excel
```

```
##       mes Ventas_Totales Ventas_Unitario Ventas_Cantidad
## 1 2019-07       227086.3        93562.60             345
## 2 2019-08       191358.9        68051.83             187
## 3 2019-09       468454.7        66959.79             908
## 4 2019-10       410463.1        79737.78             762
## 5 2019-11       333246.2        58946.77             502
## 6 2019-12       254850.6       109719.31             277
```

```
inicio_mes <- as.Date("2019-07-01")
fin_mes <- as.Date("2024-08-05")

fechas <- seq(inicio_mes, fin_mes, by = "month")
Fechas_mes <- data.frame(Fechas_mes = fechas)
head(Fechas_mes)
```

```
##   Fechas_mes
## 1 2019-07-01
## 2 2019-08-01
## 3 2019-09-01
## 4 2019-10-01
## 5 2019-11-01
## 6 2019-12-01
```

```
nrow(Fechas_mes)
```

```
## [1] 62
```

```
# Se eleiminan los dos meses que faltan en los datos, noviembre y diciembre del
# 2022, las filas 41 y 42
Fechas_mes <- Fechas_mes[-c(41, 42), ]
```

```
Fechas_mes <- as.data.frame(Fechas_mes)

# Agredar las fechas mensuales a "RLM_dftotal_mes"

RLM_dftotal_mes <- cbind(RLM_dftotal_mes, Fechas_mes)
head(RLM_dftotal_mes)
```

```
##       mes Ventas_Totales Ventas_Unitario Ventas_Cantidad Fechas_mes
## 1 2019-07       227086.3        93562.60             345 2019-07-01
## 2 2019-08       191358.9        68051.83             187 2019-08-01
## 3 2019-09       468454.7        66959.79             908 2019-09-01
## 4 2019-10       410463.1        79737.78             762 2019-10-01
## 5 2019-11       333246.2        58946.77             502 2019-11-01
## 6 2019-12       254850.6       109719.31             277 2019-12-01
```

```
nrow(RLM_dftotal_mes)
```

```
## [1] 60
```

```
VT_lambda_mes <- boxcox(RLM_dftotal_mes$Ventas_Totales,
                        objective.name = "Log-Likelihood", optimize = T)# 0.1039999
VU_lambda_mes <- boxcox(RLM_dftotal_mes$Ventas_Unitario,
                        objective.name = "Log-Likelihood", optimize = T)#-0.09042444
VC_lambda_mes <- boxcox(RLM_dftotal_mes$Ventas_Cantidad,
                        objective.name = "Log-Likelihood", optimize = T)#-0.06795844
```

Se obtiene la transformación boxcox para las ventas totales, valor unitario de los productos y cantidad de venta de los productos, por mes.

```
RLM_dftotal_mes <- RLM_dftotal_mes %>%
  mutate(
    Ventas_Totales = boxcoxTransform(Ventas_Totales, lambda = 0.1039999),
    Ventas_Unitario = boxcoxTransform(Ventas_Unitario, lambda = -0.09042444),
    Ventas_Cantidad = boxcoxTransform(Ventas_Cantidad, lambda = --0.06795844)
  )
head(RLM_dftotal_mes)
```

```
##       mes Ventas_Totales Ventas_Unitario Ventas_Cantidad Fechas_mes
## 1 2019-07       25.05912        7.130654        7.173968 2019-07-01
## 2 2019-08       24.44729        7.015924        6.281649 2019-08-01
## 3 2019-09       27.77120        7.010005        8.661840 2019-09-01
## 4 2019-10       27.26088        7.073447        8.385006 2019-10-01
## 5 2019-11       26.47022        6.963070        7.739046 2019-11-01
## 6 2019-12       25.47759        7.186832        6.849839 2019-12-01
```

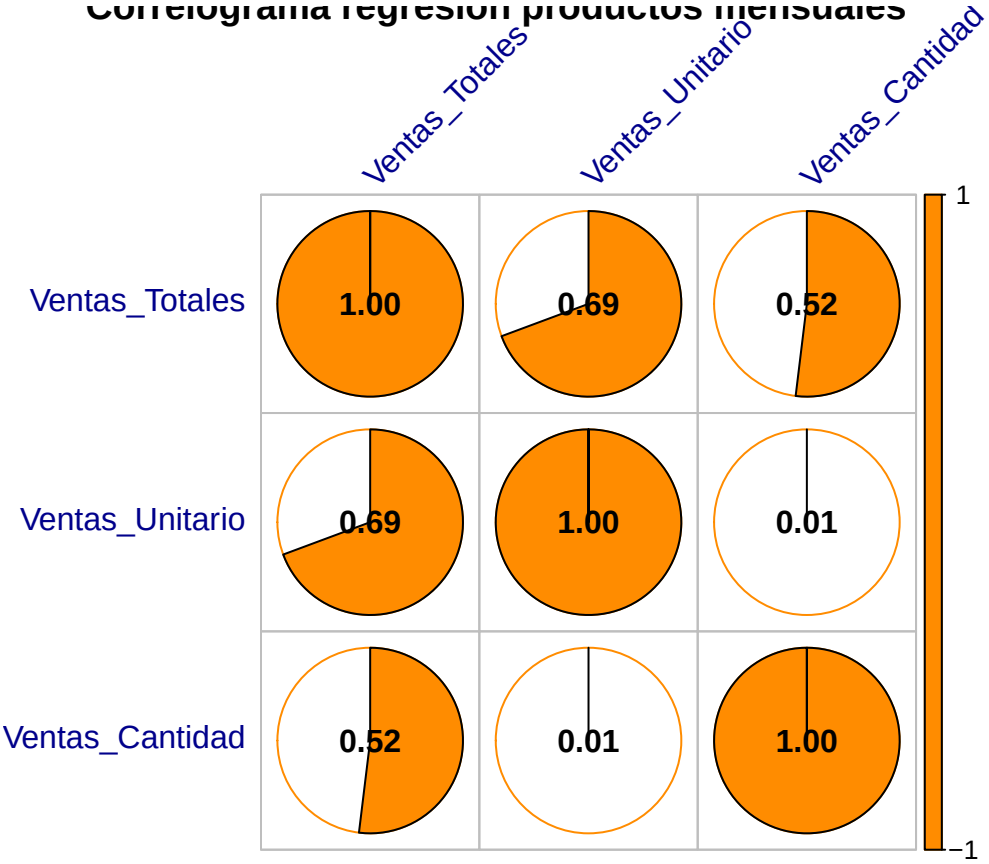
Se seleccionan las columnas que son de interés.

Se crea una matriz para el cálculo y visualización de las correlaciones, además se siembra la semilla para garantizar que los valores sean los mismos.

```
set.seed(101) # matriz
RLM_dftotal_mes_mtx <- cbind(RLM_dftotal_mes$Ventas_Totales,
                             RLM_dftotal_mes$Ventas_Unitario,
                             RLM_dftotal_mes$Ventas_Cantidad)
colnames(RLM_dftotal_mes_mtx) <- c("Ventas_Totales", "Ventas_Unitario", "Ventas_Cantidad")
head(RLM_dftotal_mes_mtx)
```

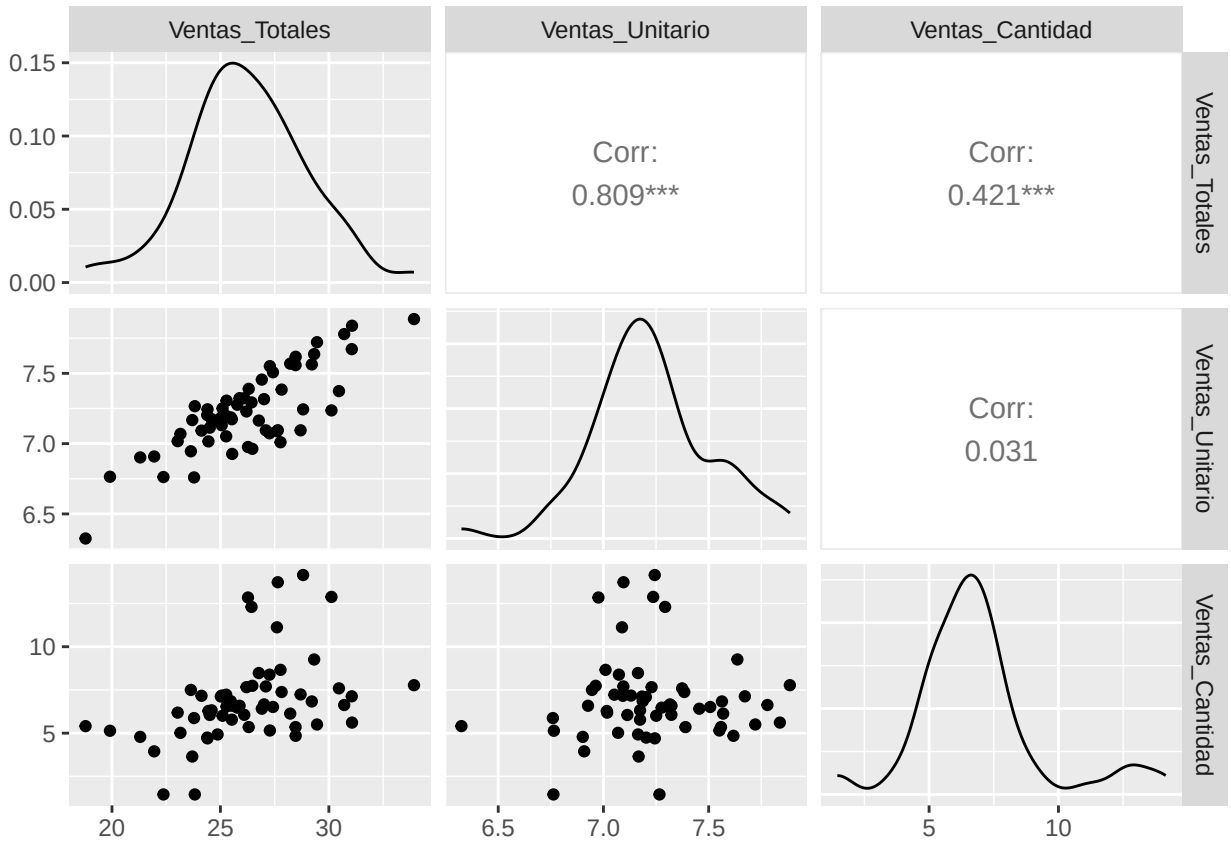
```
##      Ventas_Totales Ventas_Unitario Ventas_Cantidad
## [1,]       25.05912        7.130654        7.173968
## [2,]       24.44729        7.015924        6.281649
## [3,]       27.77120        7.010005        8.661840
## [4,]       27.26088        7.073447        8.385006
## [5,]       26.47022        6.963070        7.739046
## [6,]       25.47759        7.186832        6.849839
```

Gráficas de correlación




```
RLM_dftotal_mes <- as.data.frame(RLM_dftotal_mes_mtx)

RLM_dftotal_mes %>% GGally::ggpairs(cardinality_threshold = 10)
```



Se hace la división del conjunto de datos en una proporción de 80-20.

```
mod_lm_tot_mes <- sample.split(RLM_dftotal_mes$Ventas_Totales, SplitRatio = 0.80)
train_lm_tot_mes <- subset(RLM_dftotal_mes, mod_lm_tot_mes == T)
test_lm_tot_mes <- subset(RLM_dftotal_mes, mod_lm_tot_mes == F)
```

Modelo

Se determina el modelo lineal.

```
Mod_lm_TOTAL_mes <- lm(Ventas_Totales ~ ., data = train_lm_tot_mes)
```

Resumen del modelo obtenido.

```
print(summary(Mod_lm_TOTAL_mes))
```

```
##
## Call:
## lm(formula = Ventas_Totales ~ ., data = train_lm_tot_mes)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5313 -0.7917 -0.4075  0.4232  3.3352
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -32.65660    4.86419  -6.714 2.71e-08 ***
## Ventas_Unitario   7.76168    0.66927  11.597 4.11e-15 ***
## Ventas_Cantidad   0.40775    0.08409   4.849 1.52e-05 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.304 on 45 degrees of freedom
## Multiple R-squared:  0.7828, Adjusted R-squared:  0.7732
## F-statistic:  81.1 on 2 and 45 DF,  p-value: 1.198e-15
```

```
# Multiple R-squared:  0.7828,  Adjusted R-squared:  0.7732
```

Pronostico

```
pronostico_lm_TOTAL_mes <- predict(Mod_lm_TOTAL_mes, test_lm_tot_mes)
```

Se crea una data frame con los resultados y los valores actuales

```
resp_tot_mes <- cbind(pronostico_lm_TOTAL_mes, test_lm_tot_mes$Ventas_Totales)
resp_tot_mes <- as.data.frame(resp_tot_mes)
colnames(resp_tot_mes) <- c("prediccion", "actual")
head(resp_tot_mes)
```

```
##    prediccion   actual
## 11   18.64275 18.78257
## 14   27.82426 26.90848
## 17   25.55031 27.09356
## 22   26.86993 25.88342
## 25   28.83140 29.21300
## 26   26.06967 25.09830
```

Si es hay valores menores que cero se substituyen por cero.

```
any(resp_tot_mes < 0)
```

```
## [1] FALSE
```

Función

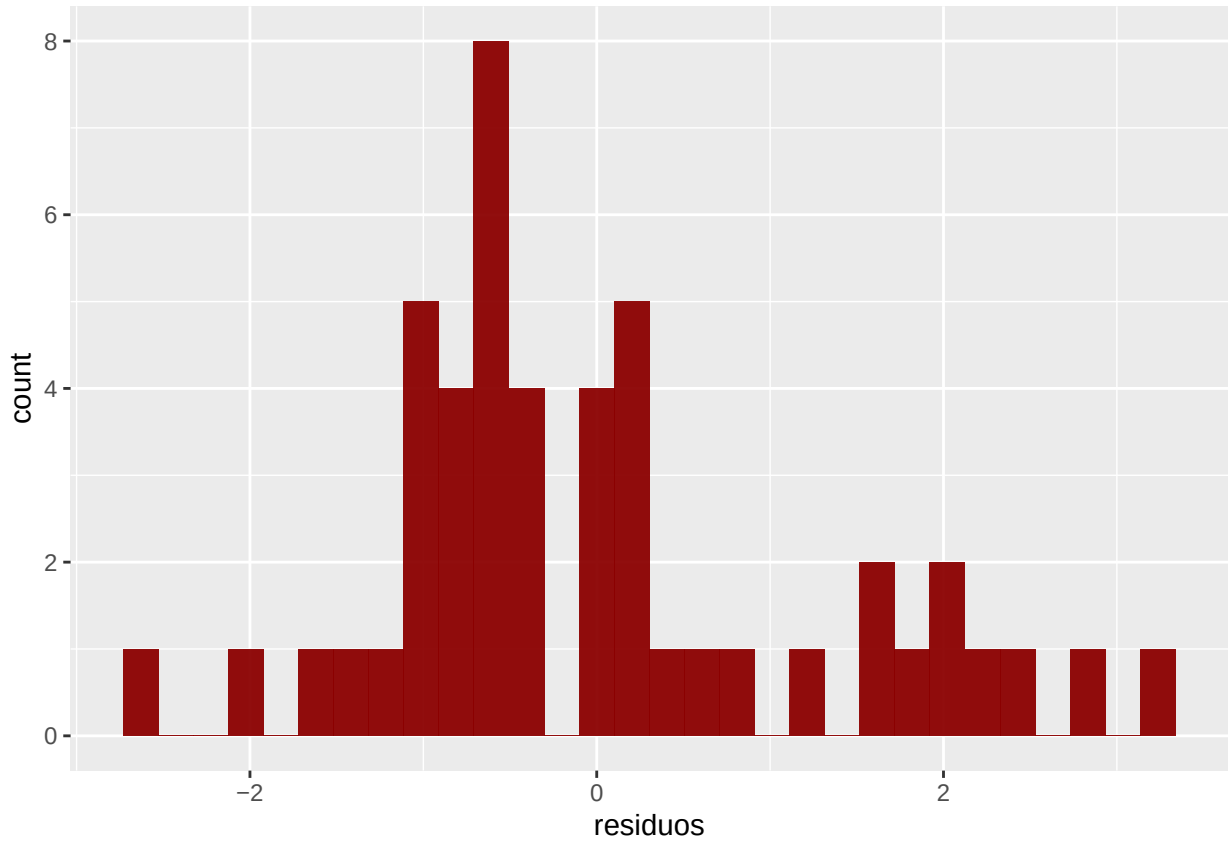
Exactitud del modelo

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

```
## [1] 0.7828177
```

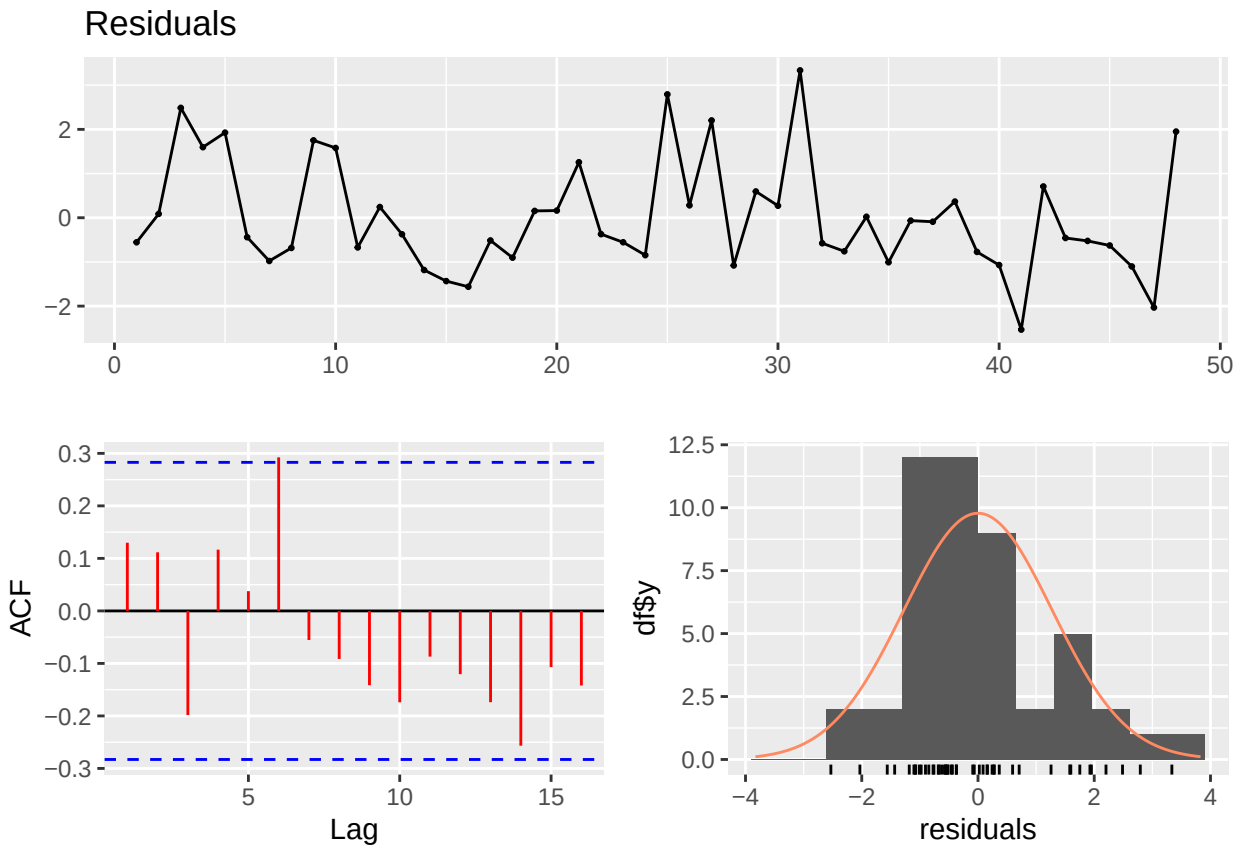
```
# [1] 0.7828177
```

Inspección de los residuales.



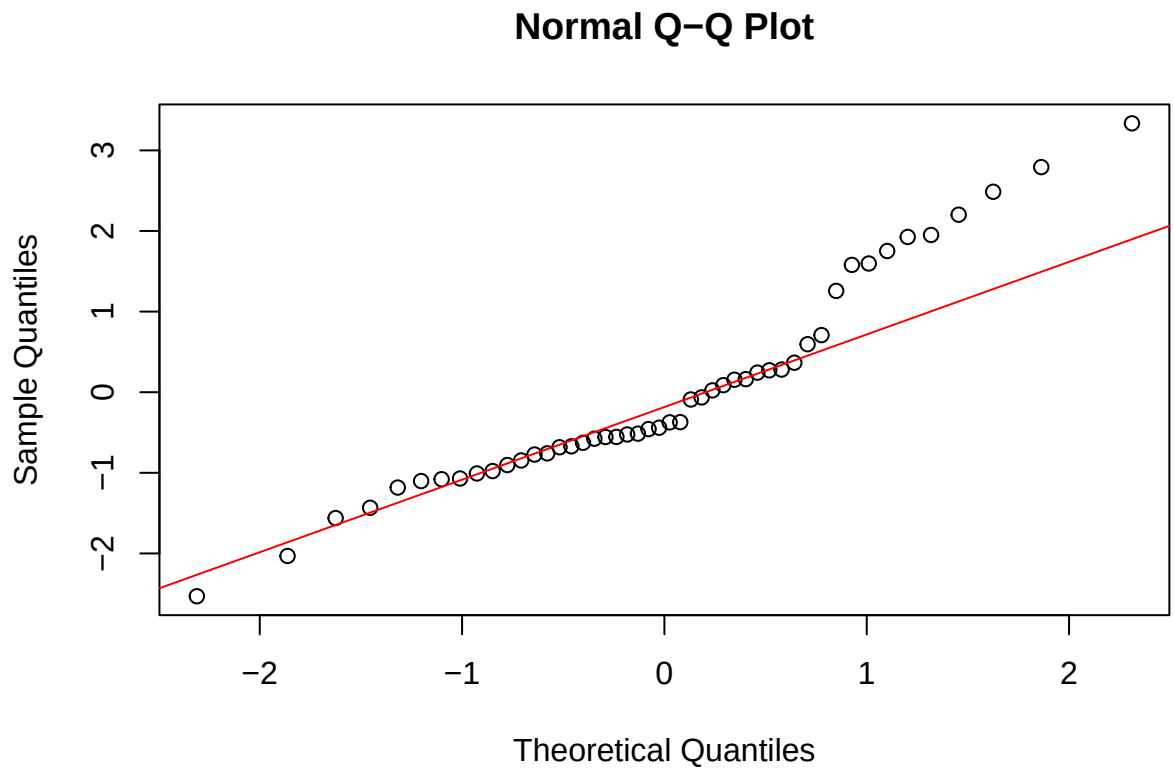
Residuales

```
checkresiduals(Mod_lm_TOTAL_mes, col = "red") # p-value = 0.214
```



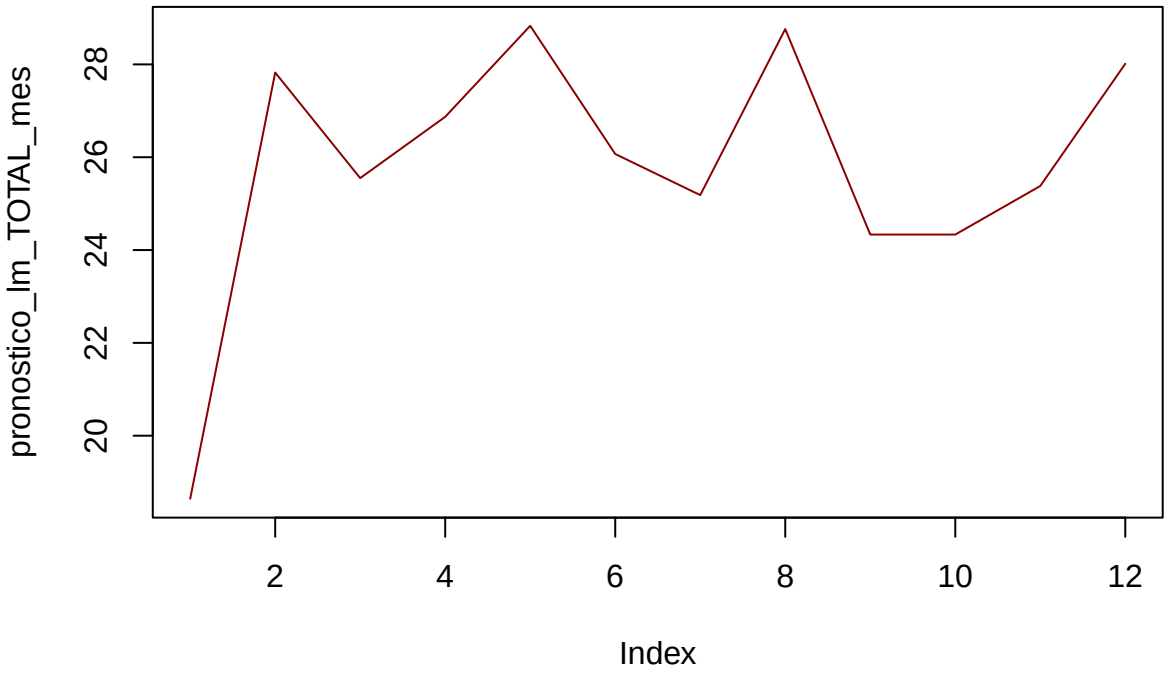

```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 13.176, df = 10, p-value = 0.214
```

Inspeccionando si existe normalidad en los residuales.

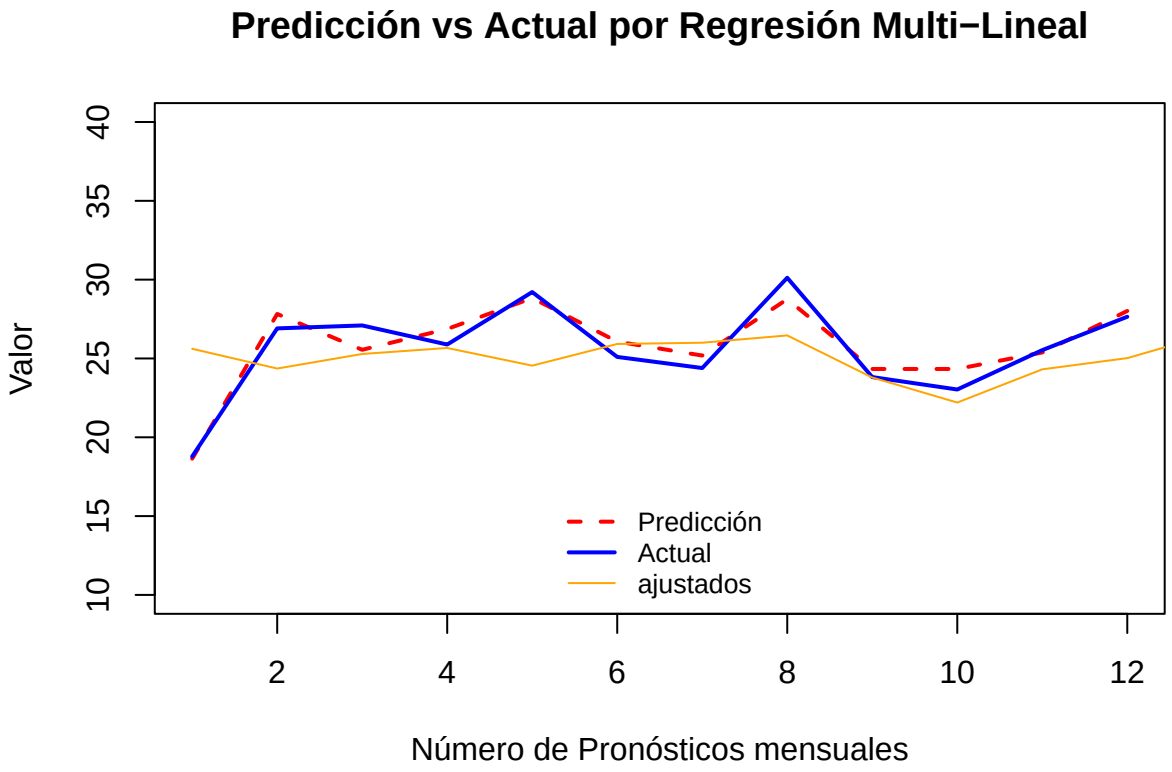


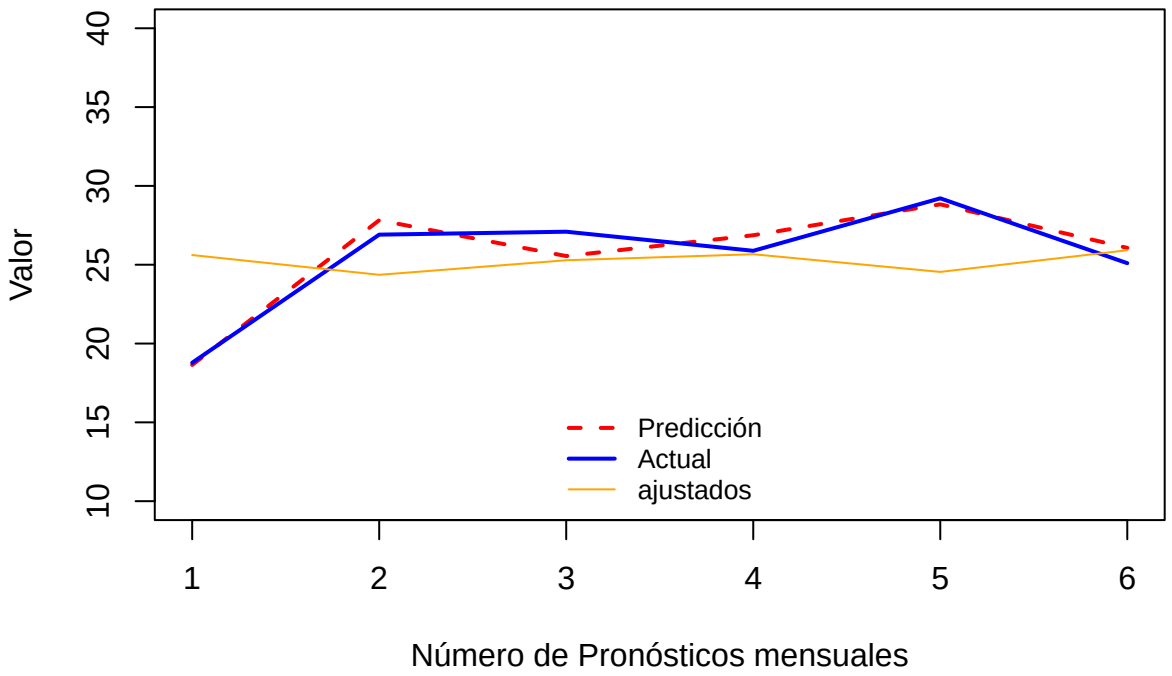
Se grafican los pronósticos.

```
## Warning in plot.xy(xy, type, ...): plot type 'lines' will be truncated to first
## character
```



Gráfica de los pronósticos junto con los valores reales.





Se determina la exactitud del modelo.

```
accuracy(resp_tot_mes$prediccion, test_lm_tot_mes$Ventas_Totales)
```

```
##                    ME      RMSE       MAE        MPE     MAPE
## Test set -0.1900724 0.9079058 0.7849562 -0.8872305 3.025572
```

```
#                    ME      RMSE       MAE        MPE     MAPE
# Test set -0.1900724 0.9079058 0.7849562 -0.8872305 3.025572
```

```
accuracy(resp_tot_mes$prediccion[1:6], test_lm_tot_mes$Ventas_Totales[1:6])
```

```
##                    ME      RMSE       MAE        MPE     MAPE
## Test set -0.1348306 0.9400555 0.8230585 -0.5563705 3.138618
```

```
#                    ME      RMSE       MAE        MPE     MAPE
# Test set -0.1348306 0.9400555 0.8230585 -0.5563705 3.138618
```

Conclusiones

El modelo captura 78.28% la dinámica de la serie, los residuales están muy por encima del valor ideal de $p > 0.05$, lo que indica que no existe una correlación entre los residuos. Esto significa que el modelo captura la dinámica de la serie temporal.