REGRESION LINEAL MULTIPLE PRODUCTOS DIARIOS

Miguel Angel Villegas

2025-03-06

*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")
```

```
Productos_Totales <- Productos_Totales %>%
  group_by(Fecha = as.Date(Fecha)) %>%
  summarize(Ventas_Totales = sum(Total),
            Ventas_Unitario = sum(ValorUnitario),
            Ventas_Cantidad = sum(Cantidad),
            .groups = "keep")
head(Productos_Totales)
```

```
## # A tibble: 6 x 4
## # Groups:   Fecha [6]
```

```
##   Fecha      Ventas_Totales Ventas_Unitario Ventas_Cantidad
##   <date>              <dbl>           <dbl>           <dbl>
## 1 2019-07-01         25826.          15751.              61
## 2 2019-07-03          3138.            545               42
## 3 2019-07-04          5330.           1727.               7
## 4 2019-07-05         10146.           8746.               3
## 5 2019-07-06         10962             630               15
## 6 2019-07-08         16194.           3740               16
```

Selección

```
Productos_Totales <- as.data.frame(Productos_Totales)
head(Productos_Totales)
```

```
##        Fecha Ventas_Totales Ventas_Unitario Ventas_Cantidad
## 1 2019-07-01      25826.333        15751.08              61
## 2 2019-07-03       3137.800          545.00              42
## 3 2019-07-04       5329.713         1727.29               7
## 4 2019-07-05      10145.534         8746.15               3
## 5 2019-07-06      10962.000          630.00              15
## 6 2019-07-08      16193.600         3740.00              16
```

Se calcula el valor del parámetro lambda para la transformación Boxcox

```
VP_lambda_dia <- boxcox(Productos_Totales$Ventas_Totales,
                        objective.name = "Log-Likelihood", optimize = T)
# 0.03343714
VUp_lambda_dia <- boxcox(Productos_Totales$Ventas_Unitario,
                         objective.name = "Log-Likelihood", optimize = T)
# 0.1156236
VCp_lambda_dia <- boxcox(Productos_Totales$Ventas_Cantidad,
                         objective.name = "Log-Likelihood", optimize = T)
# -0.2173778
```

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

```
RLM_prod_dia <- Productos_Totales %>%
  mutate(
    Ventas_Totales = boxcoxTransform(Productos_Totales$Ventas_Totales, lambda = 0.03343714),
    Ventas_Unitario = boxcoxTransform(Productos_Totales$Ventas_Unitario, lambda = 0.1156236),
    Ventas_Cantidad = boxcoxTransform(Productos_Totales$Ventas_Cantidad, lambda = -0.2173778)
  )
head(RLM_prod_dia)
```

```
##        Fecha Ventas_Totales Ventas_Unitario Ventas_Cantidad
## 1 2019-07-01      12.097812       17.791258       2.7179645
## 2 2019-07-03       9.239193        9.271557       2.5588936
## 3 2019-07-04       9.938811       11.828388       1.5867298
## 4 2019-07-05      10.805775       16.052590       0.9772782
## 5 2019-07-06      10.911278        9.574392       2.0468280
## 6 2019-07-08      11.447303       13.741643       2.0824010
```

Se seleccionan las columnas que son de interes

```
RLM_prod_dia <- RLM_prod_dia %>%
  select(Ventas_Totales, Ventas_Unitario, Ventas_Cantidad)
```

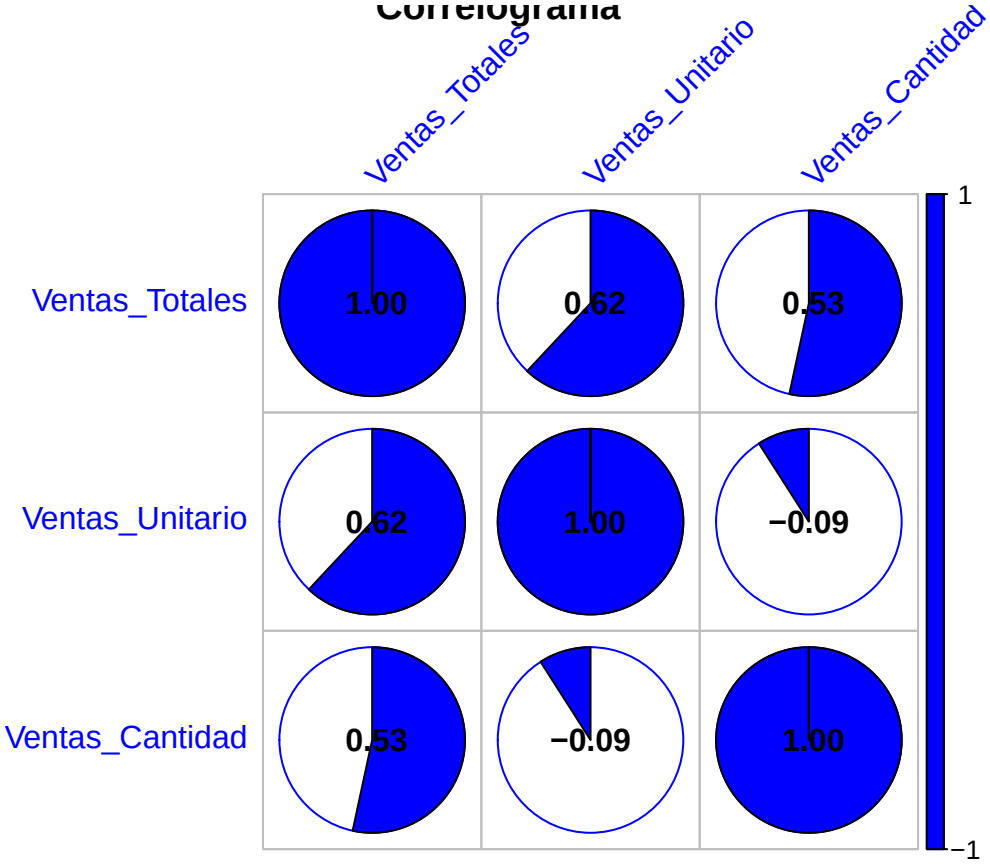
Se crea una matriz para el cálculo y visualización de las correlaciones, además se siembra la semilla para

garantizar que los valores sean los mismos.

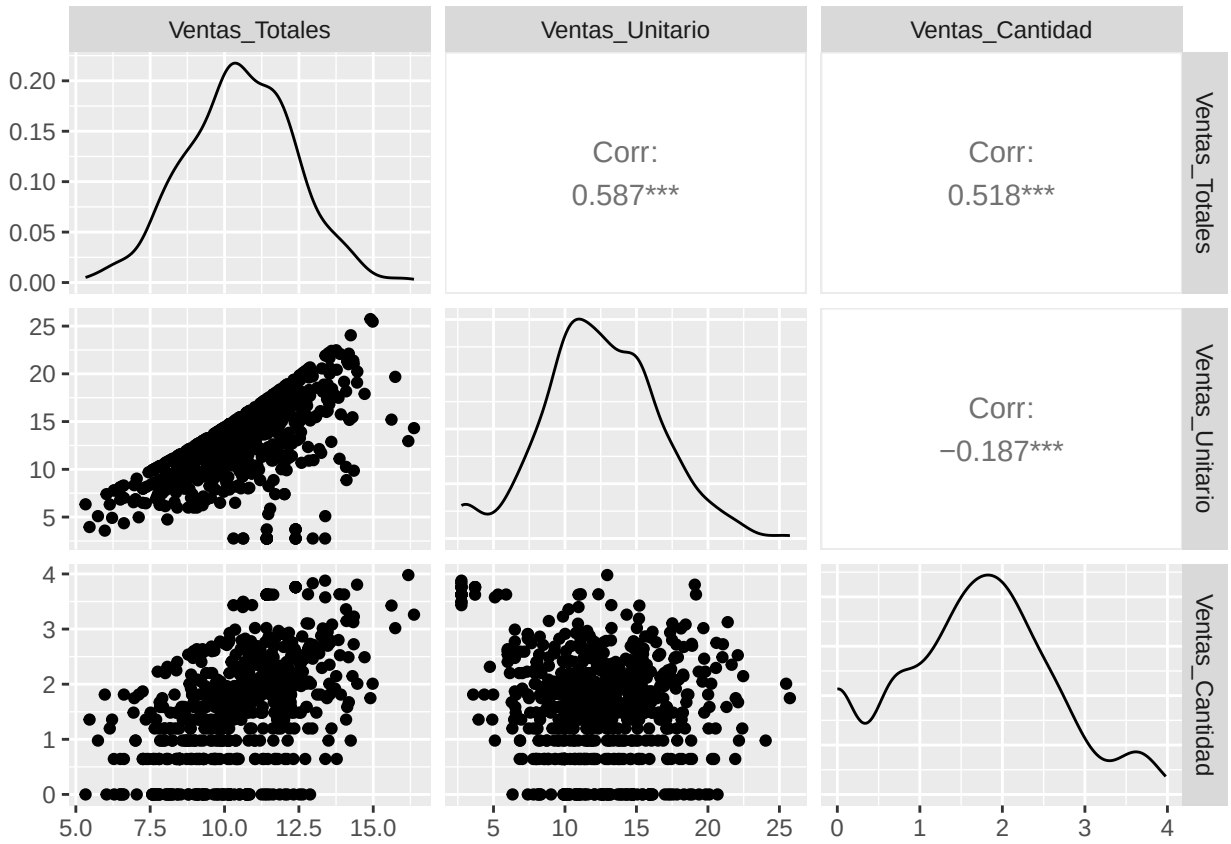
```
set.seed(101)
RLM_prod_dia_mtx <- cbind(RLM_prod_dia$Ventas_Totales,
                          RLM_prod_dia$Ventas_Unitario,
                          RLM_prod_dia$Ventas_Cantidad)
colnames(RLM_prod_dia_mtx) <- c("Ventas_Totales", "Ventas_Unitario", "Ventas_Cantidad")
head(RLM_prod_dia_mtx)
```

```
##      Ventas_Totales Ventas_Unitario Ventas_Cantidad
## [1,]      12.097812       17.791258       2.7179645
## [2,]       9.239193        9.271557       2.5588936
## [3,]       9.938811       11.828388       1.5867298
## [4,]      10.805775       16.052590       0.9772782
## [5,]      10.911278        9.574392       2.0468280
## [6,]      11.447303       13.741643       2.0824010
```

Gráficas de correlación




```
RLM_prod_dia <- as.data.frame(RLM_prod_dia_mtx)
RLM_prod_dia %>% GGally::ggpairs(cardinality_threshold = 10)
```



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

```
mp_lm <- sample.split(RLM_prod_dia$Ventas_Totales, SplitRatio = 0.80)
ep_lm <- subset(RLM_prod_dia, mp_lm == T)
pp_lm <- subset(RLM_prod_dia, mp_lm == F)
```

Modelo

Se determina el modelo lineal.

```
Moplm <- lm(Ventas_Totales ~ ., data = ep_lm)
```

Resumen del modelo obtenido.

```
print(summary(Moplm))
```

```
##
## Call:
## lm(formula = Ventas_Totales ~ ., data = ep_lm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4186 -0.6269 -0.0910  0.6183  3.2096
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     4.560227   0.152484   29.91   <2e-16 ***
## Ventas_Unitario 0.319916   0.009719   32.92   <2e-16 ***
## Ventas_Cantidad 1.232842   0.040563   30.39   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.904 on 553 degrees of freedom
## Multiple R-squared:  0.7528, Adjusted R-squared:  0.7519
## F-statistic: 841.9 on 2 and 553 DF,  p-value: < 2.2e-16
# Multiple R-squared:  0.7528,  Adjusted R-squared:  0.7519
```

Pronostico

```
pronostico_lmp <- predict(Moplm, pp_lm)
```

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

```
resultados_p <- cbind(pronostico_lmp, pp_lm$Ventas_Totales)
resultados_p <- as.data.frame(resultados_p)
colnames(resultados_p) <- c("prediccion", "actual")
head(resultados_p)
```

```
##    prediccion    actual
## 11  11.568720 10.521060
## 14  11.749650 10.928227
## 17   9.476425  9.679650
## 22   7.938893  5.974259
## 25   9.996085 10.607856
## 26   9.688164  8.554654
```

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

```
any(resultados_p < 0)
```

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

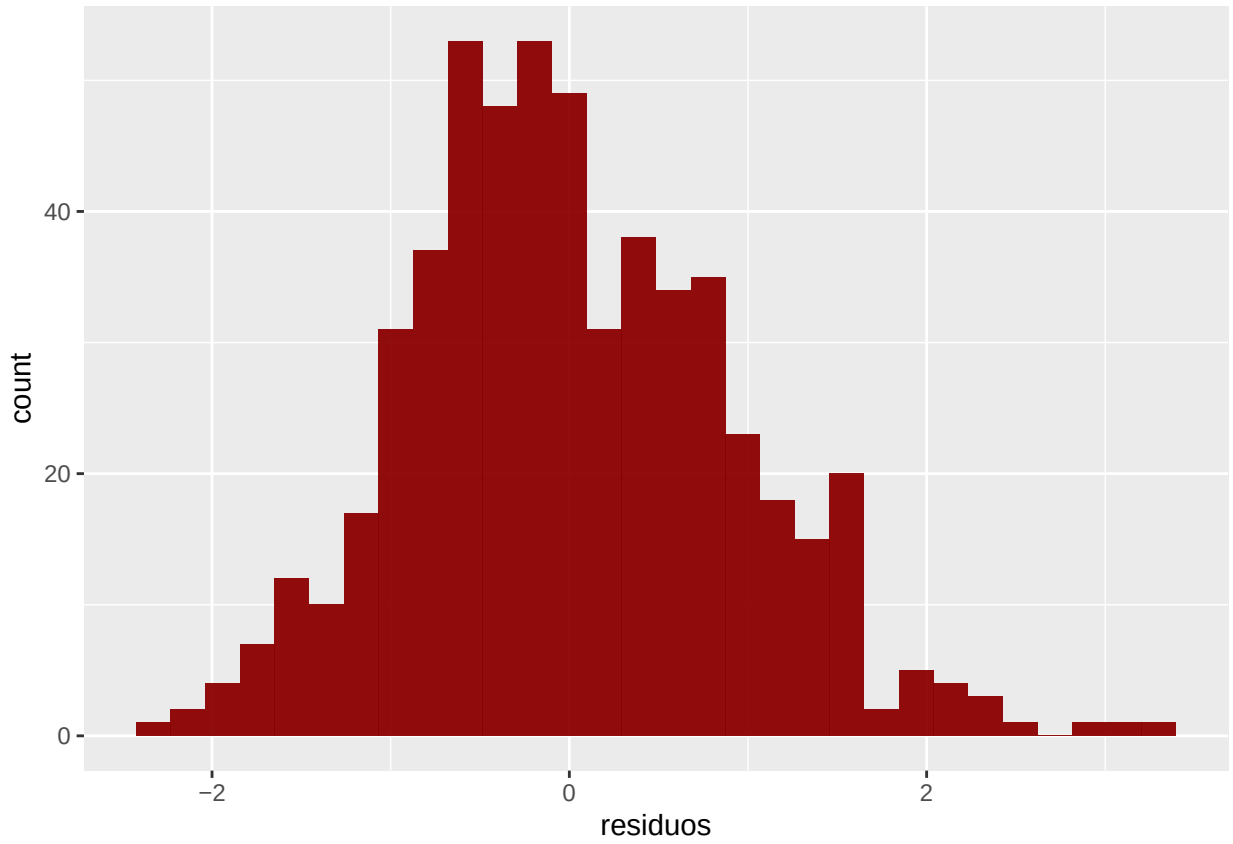
Función

Exactitud del modelo.

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

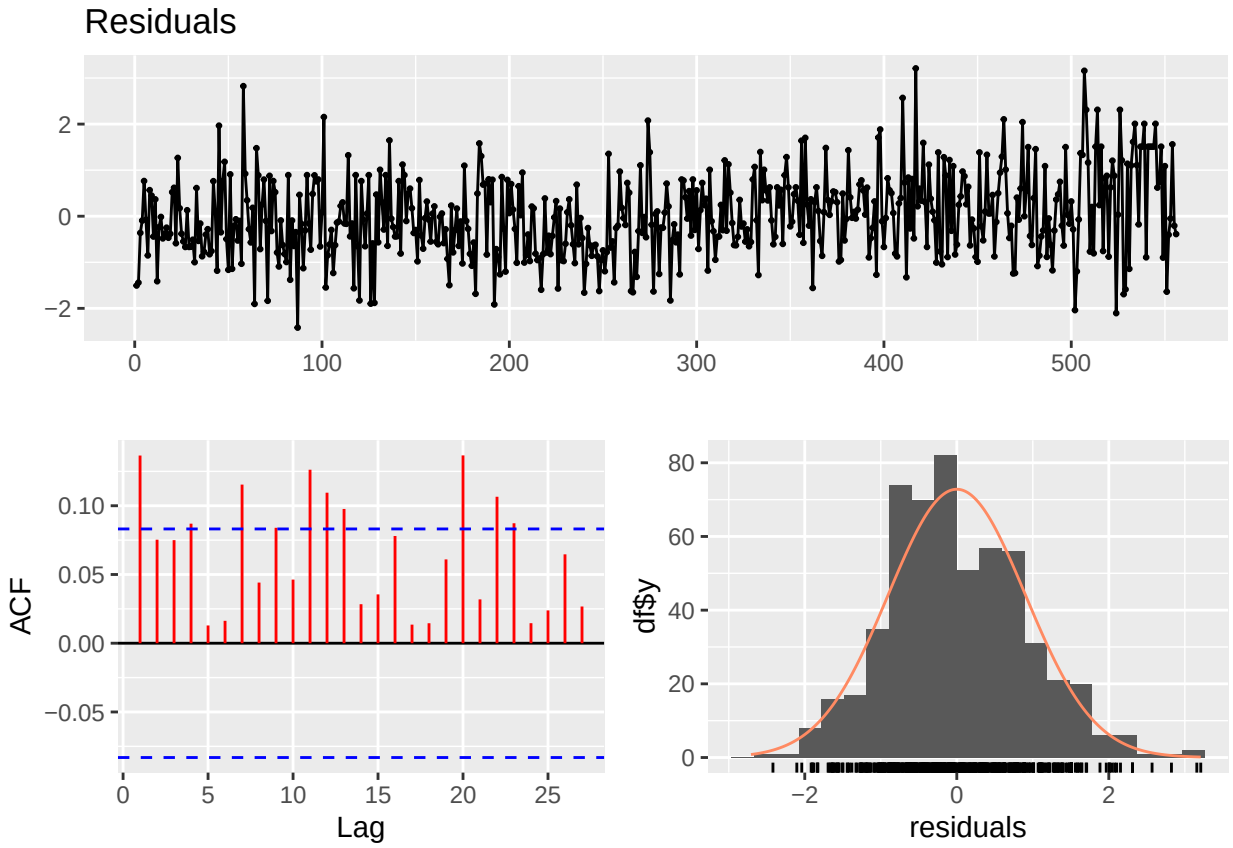
```
## [1] 0.7527771
# [1] 0.7527771
```

Inspección de los residuales.



Residuales

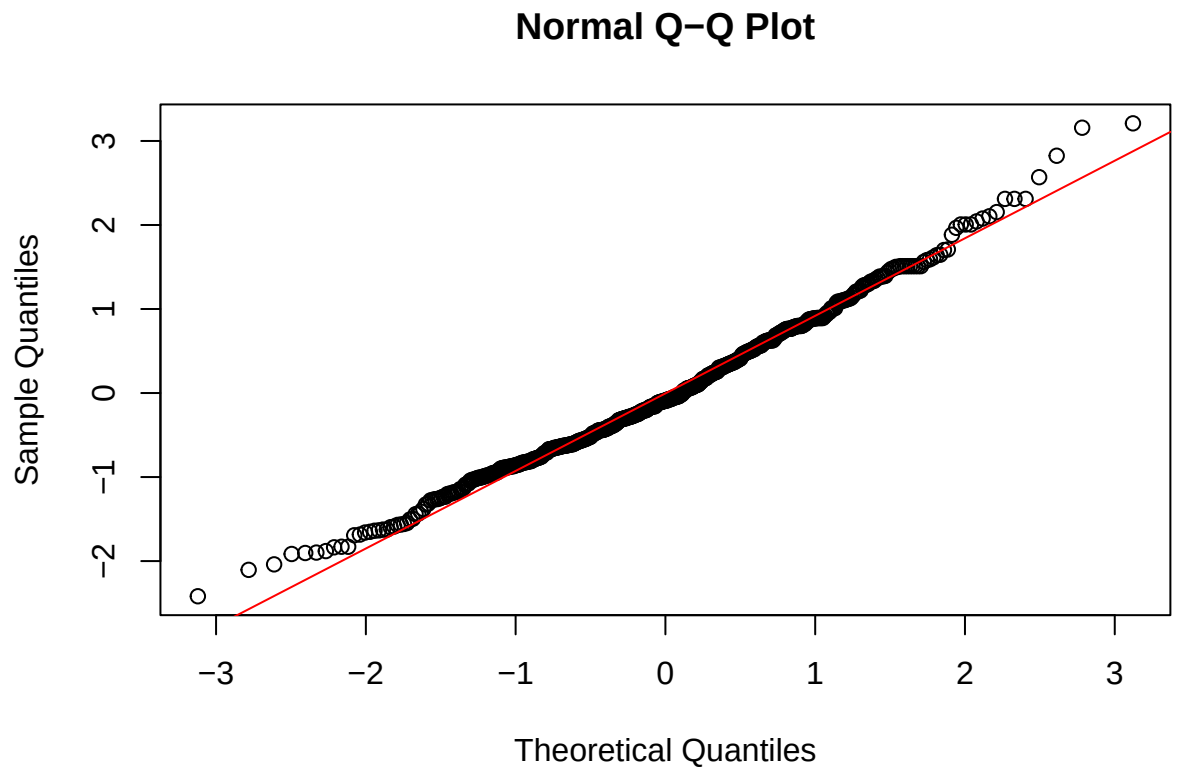
```
checkresiduals(Moplm, col = "red")
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 27.098, df = 10, p-value = 0.002513
```

```
# p-value = 0.002513
```

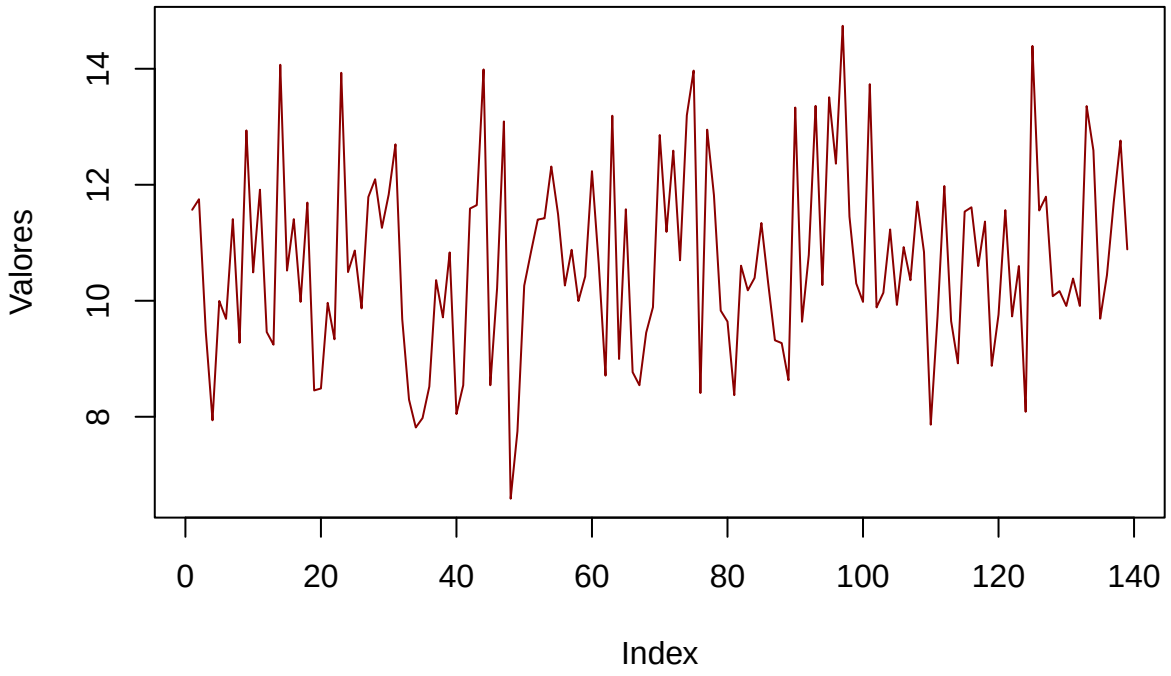
Inspeccionando si existe normalidad en los residuales



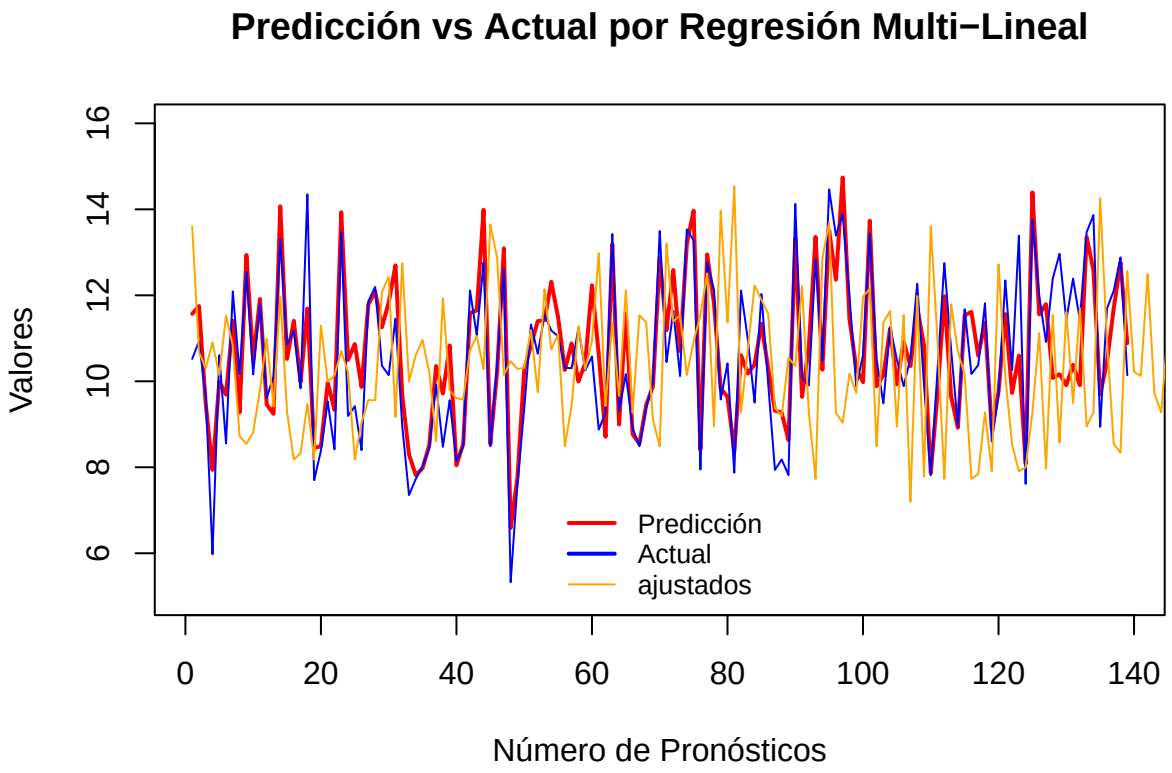
Se grafican los pronósticos.

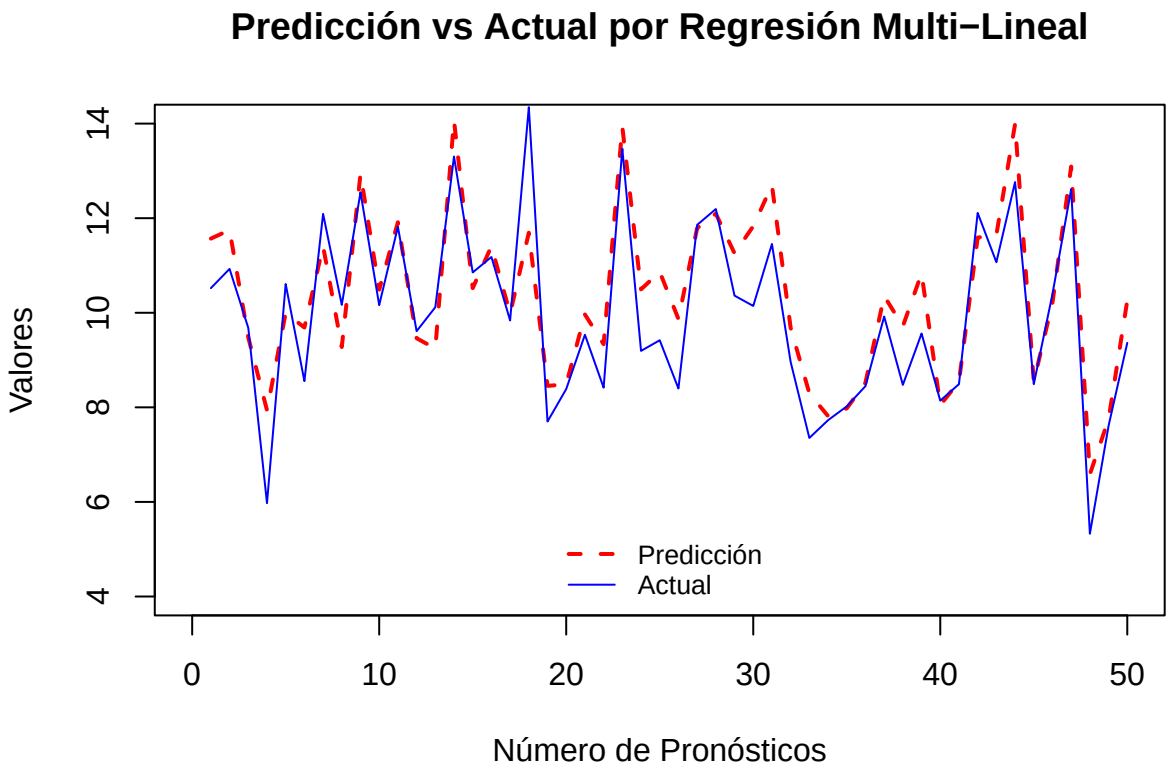
```
pronostico_prod_lm <- predict(Moplm, pp_lm)

plot(pronostico_prod_lm, type = "lines", col = "darkred", ylab = "Valores")
```

```
## 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(resultados_p$prediccion, pp_lm$Ventas_Totales)
```

```
##                    ME      RMSE       MAE       MPE     MAPE
## Test set -0.08633042 0.9082222 0.7066877 -1.617599 6.819981
```

```
#                    ME      RMSE       MAE       MPE     MAPE
# Test set -0.08633042 0.9082222 0.7066877 -1.617599 6.819981
```

```
accuracy(resultados_p$prediccion[1:50], pp_lm$Ventas_Totales[1:50])
```

```
##                  ME      RMSE       MAE       MPE     MAPE
## Test set -0.393436 0.8945261 0.6897526 -4.798475 7.325404
```

```
# Test set -0.393436 0.8945261 0.6897526 -4.798475 7.325404
```

Conclusiones

El modelo captura 75.28% la dinámica de la serie, sin embargo, los residuales están muy por debajo del valor ideal de $p > 0.05$, lo que indica que existe una fuerte correlación entre los residuos. Si bien captura la dinámica de los valores, la correlación residual puede presentar un problema para ser considerado un modelo ideal de pronostico.