A8 -Series de Tiempo

Héctor Hibran Tapia Fernández - A01661114

2024-11-14

1. Para los datos de las ventas de televisores analiza la serie de tiempo más apropiada:

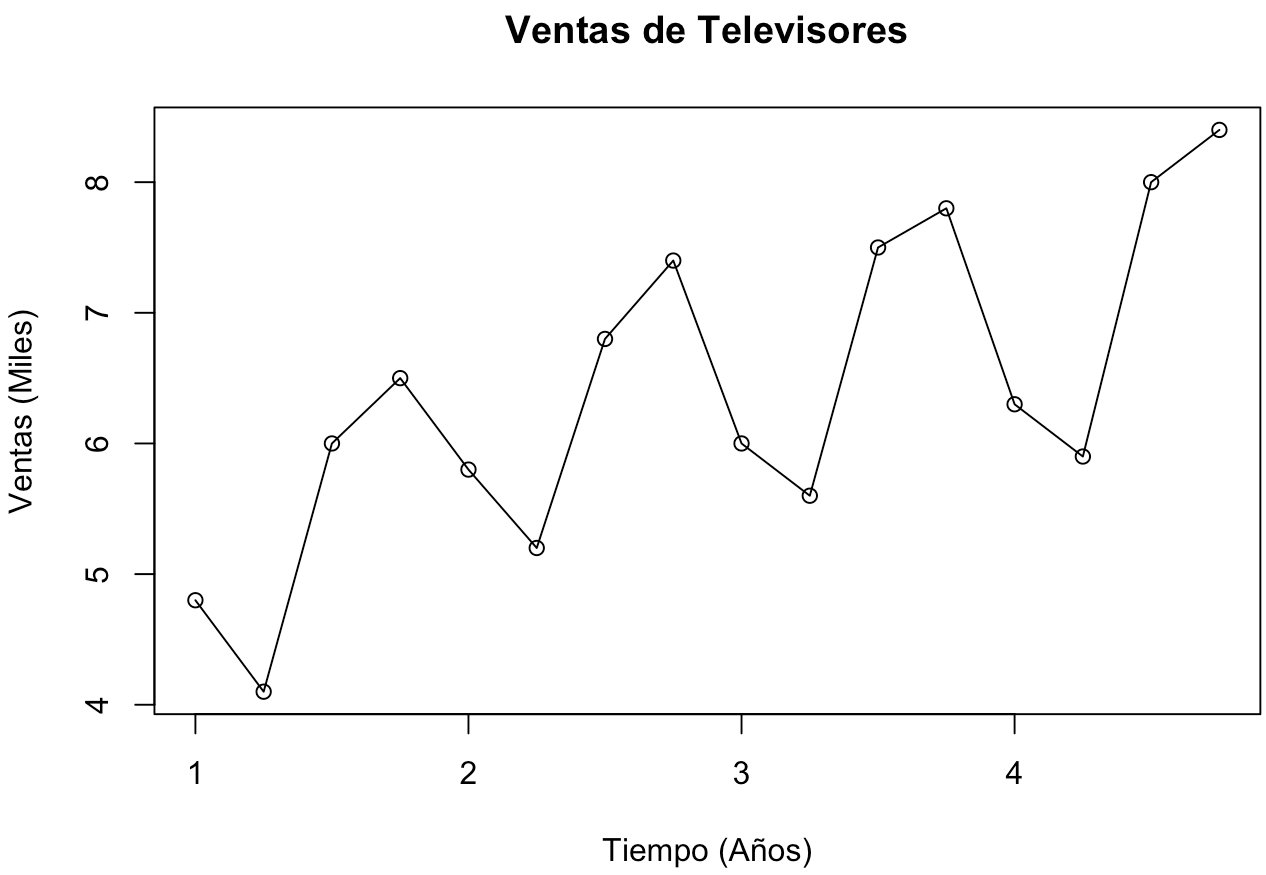
```
año = rep(1:4, each = 4)
trimestre = rep(1:4, times = 4)
ventas = c(4.8, 4.1, 6.0, 6.5, 5.8, 5.2, 6.8, 7.4, 6.0, 5.6, 7.5, 7.8, 6.3, 5.9, 8.0, 8.
4)
data = data.frame(año, trimestre, ventas)
```

Realiza el análisis de tendencia y estacionalidad:

- Identifica si es una serie estacionaria

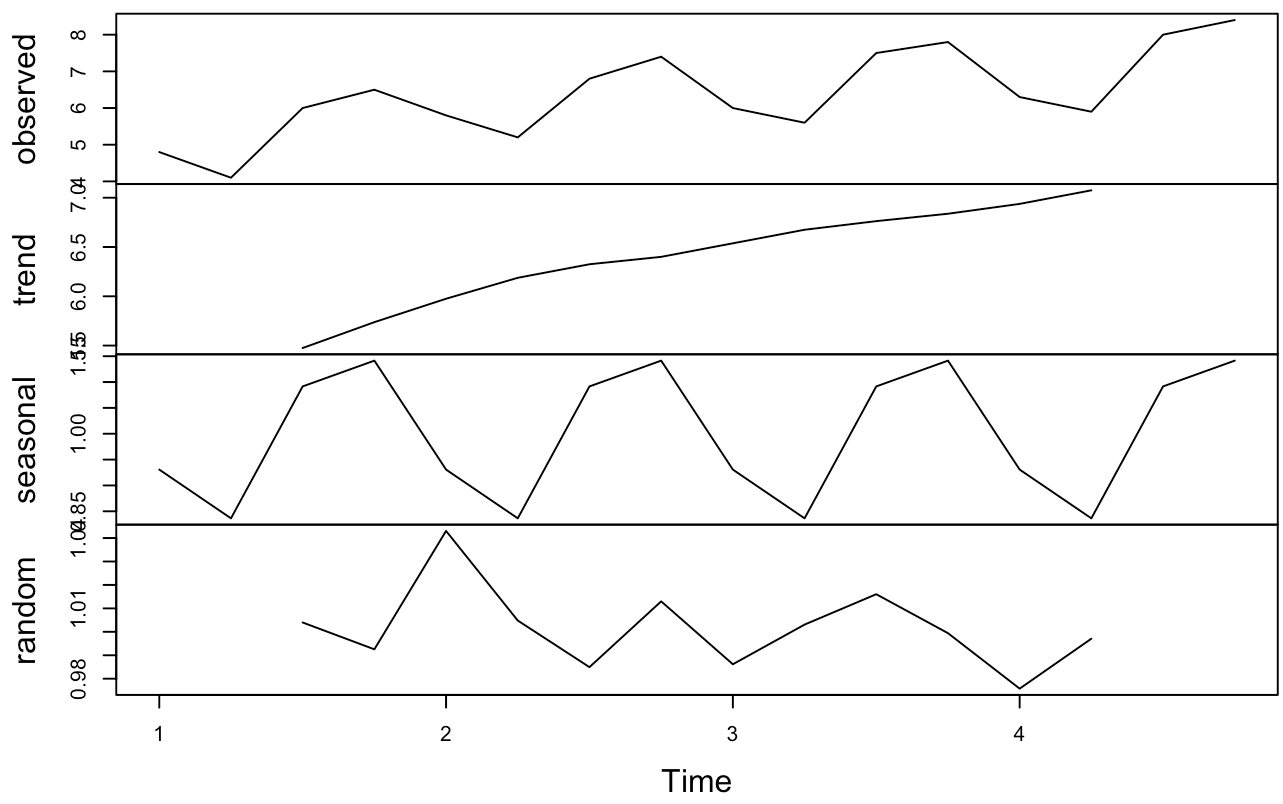
Se puede observar una tendencia positiva lo que confirma el aumento en las ventas con el tiempo, no es estacionaria ya que no se mantiene dentro del mismo rango de valores, su media es dependiente del tiempo.

```
ventas_ts = ts(data$ventas, start = c(1,1), frequency = 4)
plot(ventas_ts, main = "Ventas de Televisores", xlab = "Tiempo (Años)", ylab = "Ventas
(Miles)", type = "o")
```

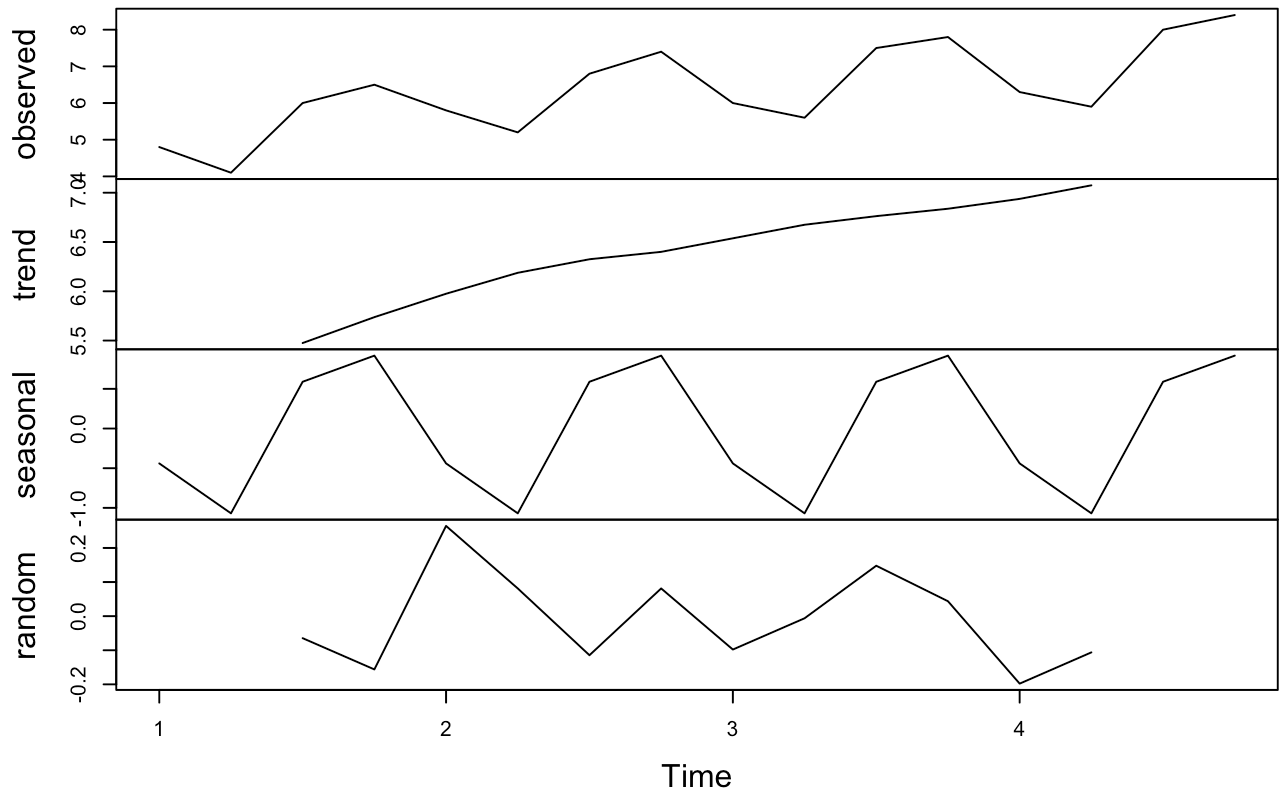


- Grafica la serie para verificar su tendencia y estacionalidad

```
decomp = decompose(ventas_ts, type = "multiplicative") # tambien se puede additive
plot(decomp)
```

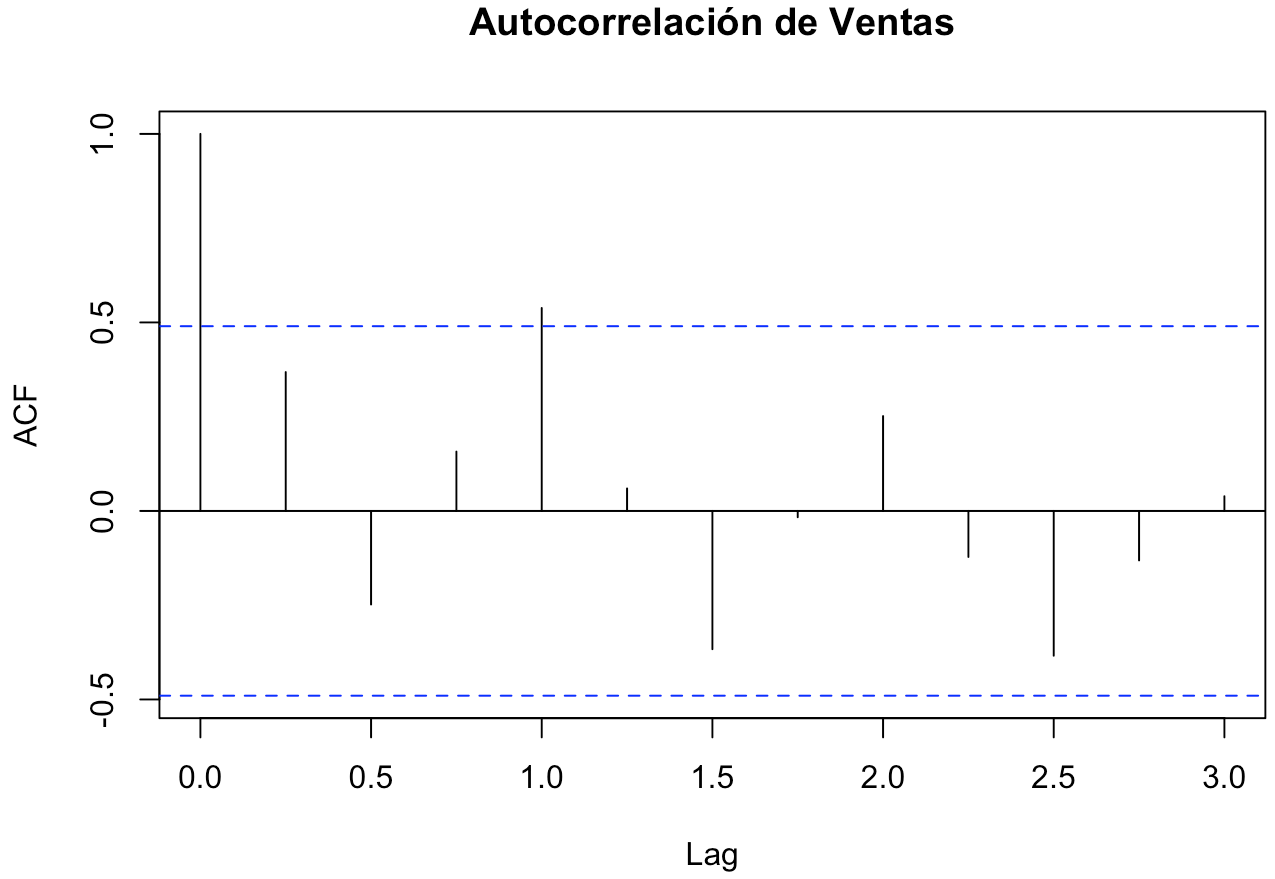



```
decomp = decompose(ventas_ts, type = "additive")
plot(decomp)
```



- Analiza su gráfico de autocorrelación

```
acf(ventas_ts, main = "Autocorrelación de Ventas")
```



- Identifica si el modelo puede ser sumativo o multiplicativo (puedes probar con ambos para ver con cuál es mejor el modelo)

Según Wikipedia…

“Un modelo aditivo es una forma de representar una serie de tiempo en la que los distintos componentes (tendencia, estacionalidad y ruido) se suman para dar el valor observado en cada punto temporal.”

Teniendo eso en cuenta diría que el modelo aditivo es el más adecuado para los datos de ventas de los televisores porque la tendencia es lineal, la estacionalidad es constante y el ruido es pequeño. Esto permite representar la serie de tiempo sumando sin necesidad de un modelo más complejo como lo sería el multiplicativo.

En resumen, esta serie de tiempo es no estacionaria, muestra tendencia creciente y estacionalidad trimestral, y un modelo aditivo es probable que capture bien su estructura.

2. Calcula los índices estacionales y grafica la serie desestacionalizada.

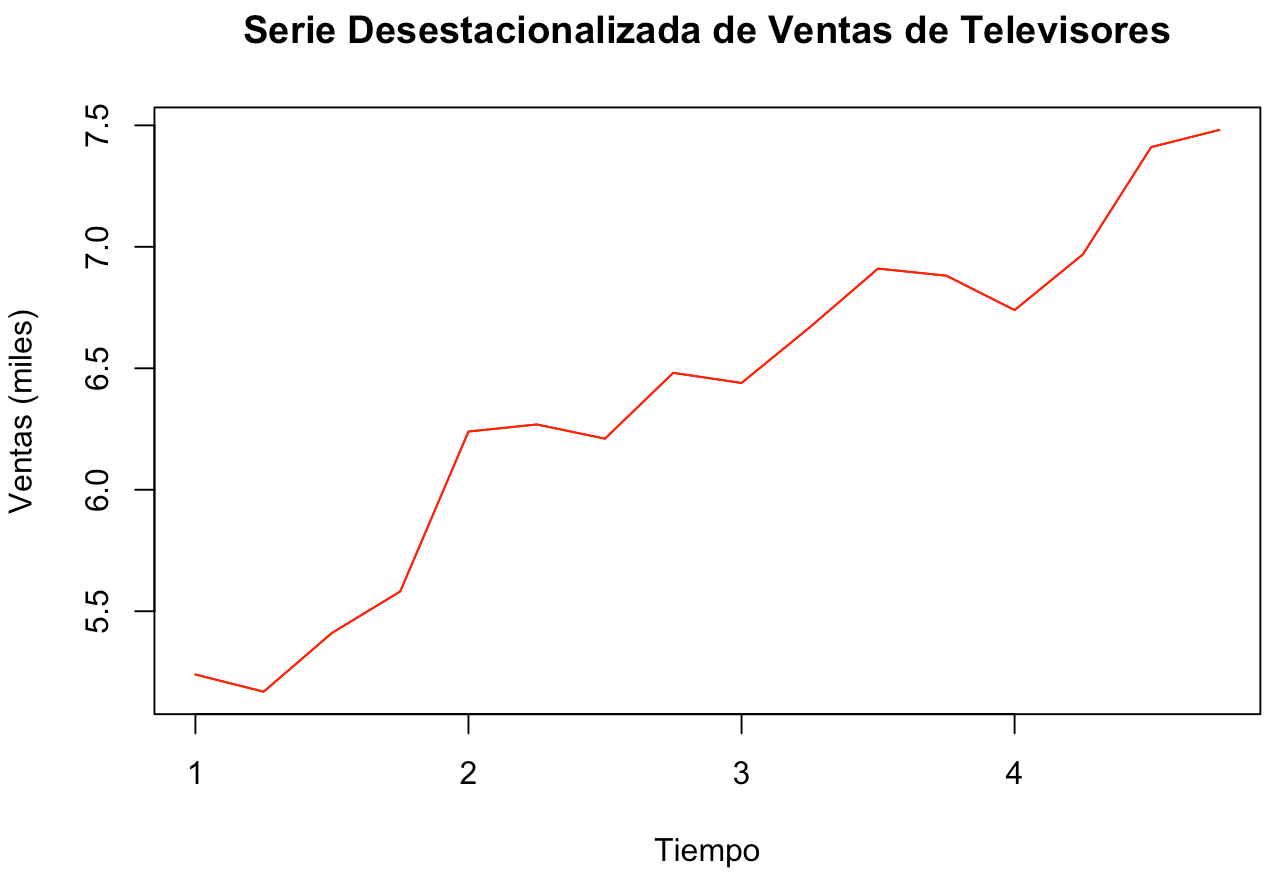
```
decompose_add = decompose(ventas_ts, type = "additive")
indices_estacionales = decompose_add$seasonal
print("Índices Estacionales:")
```

```
## [1] "Índices Estacionales:"
```

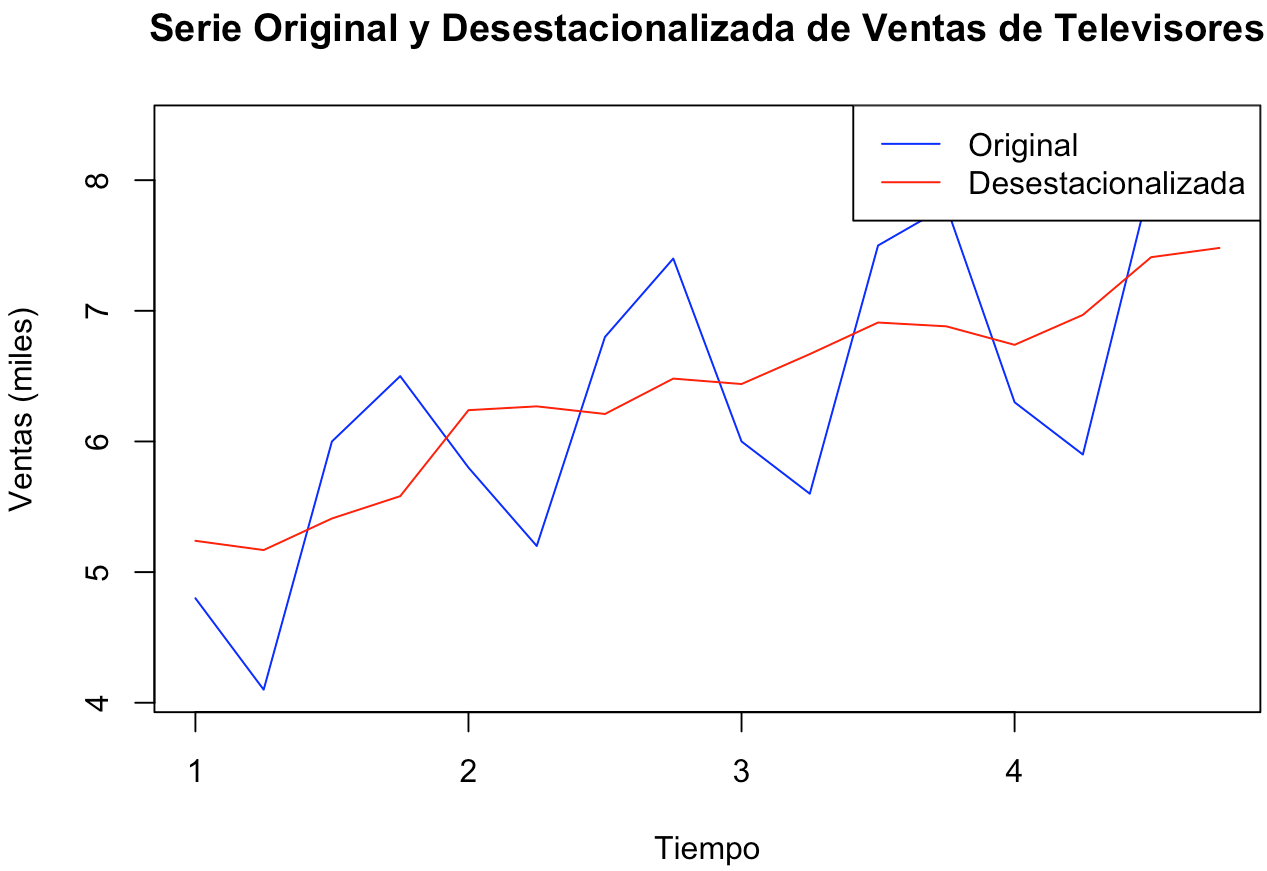
```
print(indices_estacionales)
```

```
##         Qtr1       Qtr2       Qtr3       Qtr4
## 1 -0.4395833 -1.0687500  0.5895833  0.9187500
## 2 -0.4395833 -1.0687500  0.5895833  0.9187500
## 3 -0.4395833 -1.0687500  0.5895833  0.9187500
## 4 -0.4395833 -1.0687500  0.5895833  0.9187500
```

```
serie_desestacionalizada = ventas_ts - indices_estacionales
plot(serie_desestacionalizada, main = "Serie Desestacionalizada de Ventas de Televisore
s",
     xlab = "Tiempo", ylab = "Ventas (miles)", col = "red")
lines(serie_desestacionalizada, col = "red")
```




```
plot(ventas_ts, main = "Serie Original y Desestacionalizada de Ventas de Televisores",
     xlab = "Tiempo", ylab = "Ventas (miles)", col = "blue")
lines(serie_desestacionalizada, col = "red")
legend("topright", legend = c("Original", "Desestacionalizada"), col = c("blue", "red"),
lty = 1)
```



3. Analiza el modelo lineal de la tendencia

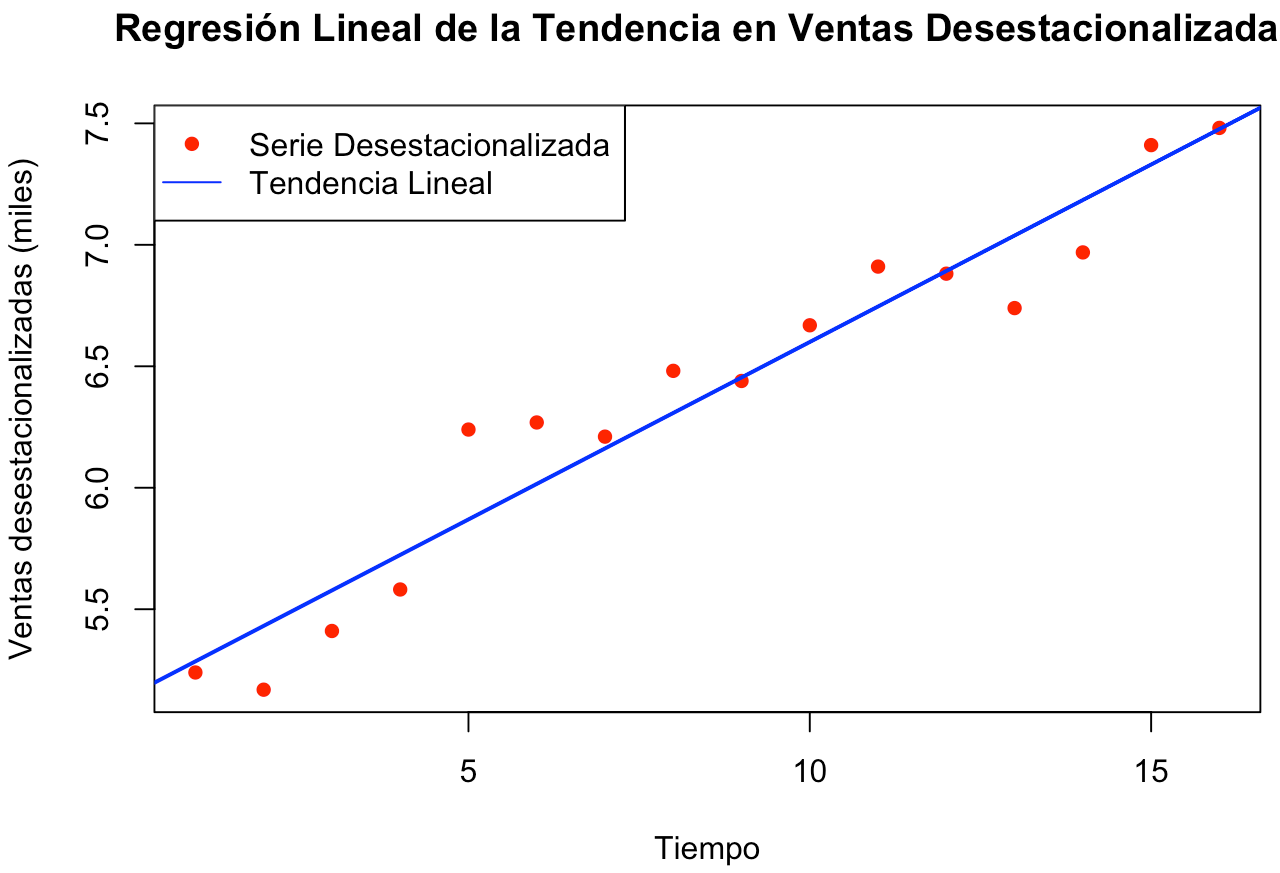
- Realiza la regresión lineal de la tendencia (ventas desestacionalizadas vs tiempo)

```
tiempo = 1:length(serie_desestacionalizada)
modelo_lineal = lm(serie_desestacionalizada ~ tiempo)
summary_modelo = summary(modelo_lineal)
summary_modelo
```

```
##
## Call:
## lm(formula = serie_desestacionalizada ~ tiempo)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2992 -0.1486 -0.0037  0.1005  0.3698
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.13917    0.10172   50.52  < 2e-16 ***
## tiempo       0.14613    0.01052   13.89  1.4e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.194 on 14 degrees of freedom
## Multiple R-squared:  0.9324, Adjusted R-squared:  0.9275
## F-statistic:   193 on 1 and 14 DF,  p-value: 1.399e-09
```

```
plot(tiempo, serie_desestacionalizada, main = "Regresión Lineal de la Tendencia en Venta
s Desestacionalizadas",
     xlab = "Tiempo", ylab = "Ventas desestacionalizadas (miles)", col = "red", pch = 1
6)

abline(modelo_lineal, col = "blue", lwd = 2)
legend("topleft", legend = c("Serie Desestacionalizada", "Tendencia Lineal"), col = c("r
ed", "blue"), pch = c(16, NA), lty = c(NA, 1))
```



- Analiza la significancia del modelo lineal, global e individual

```
p_valor_global = summary_modelo$fstatistic[1]
p_valor_global
```

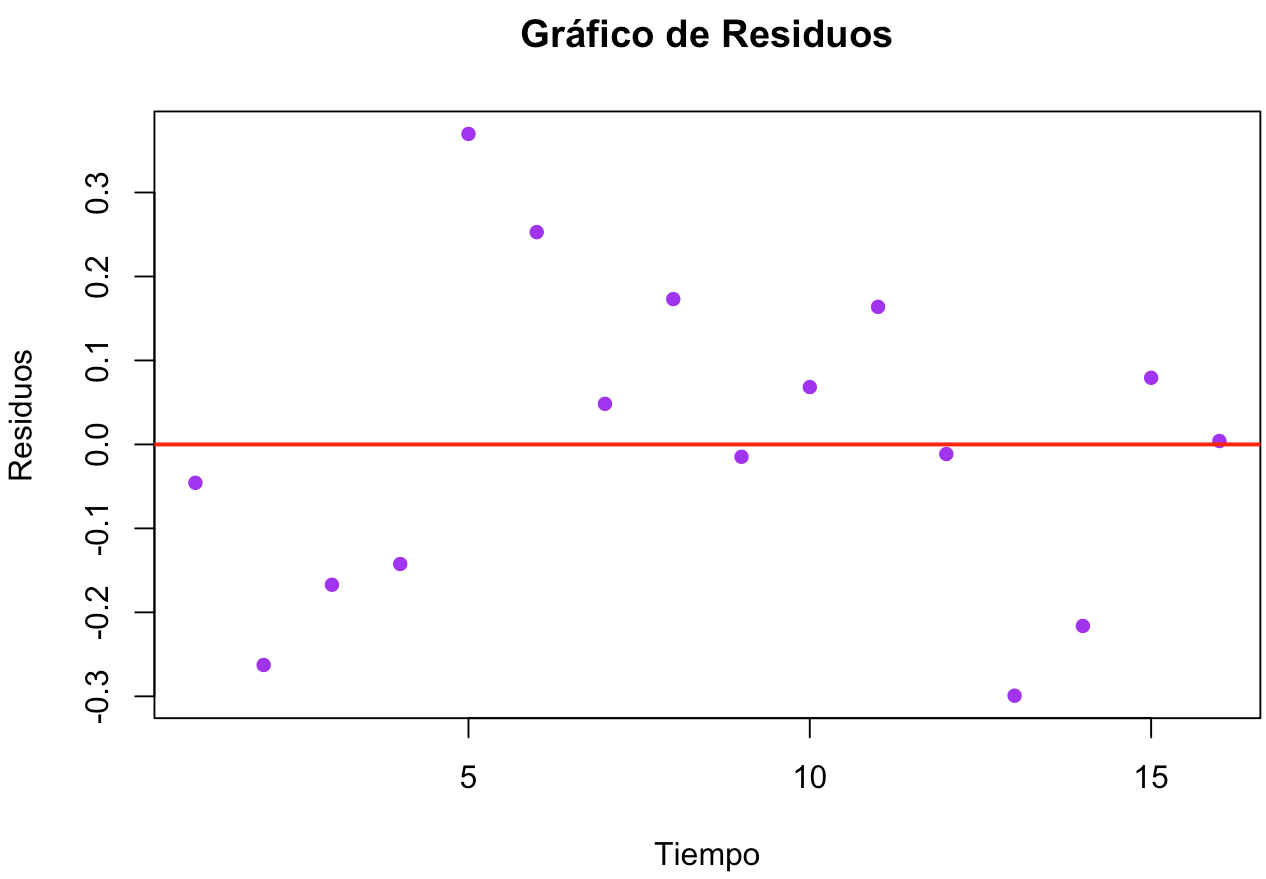
```
##    value
## 192.9753
```

```
p_valores_individuales = summary_modelo$coefficients[, 4]
p_valores_individuales
```

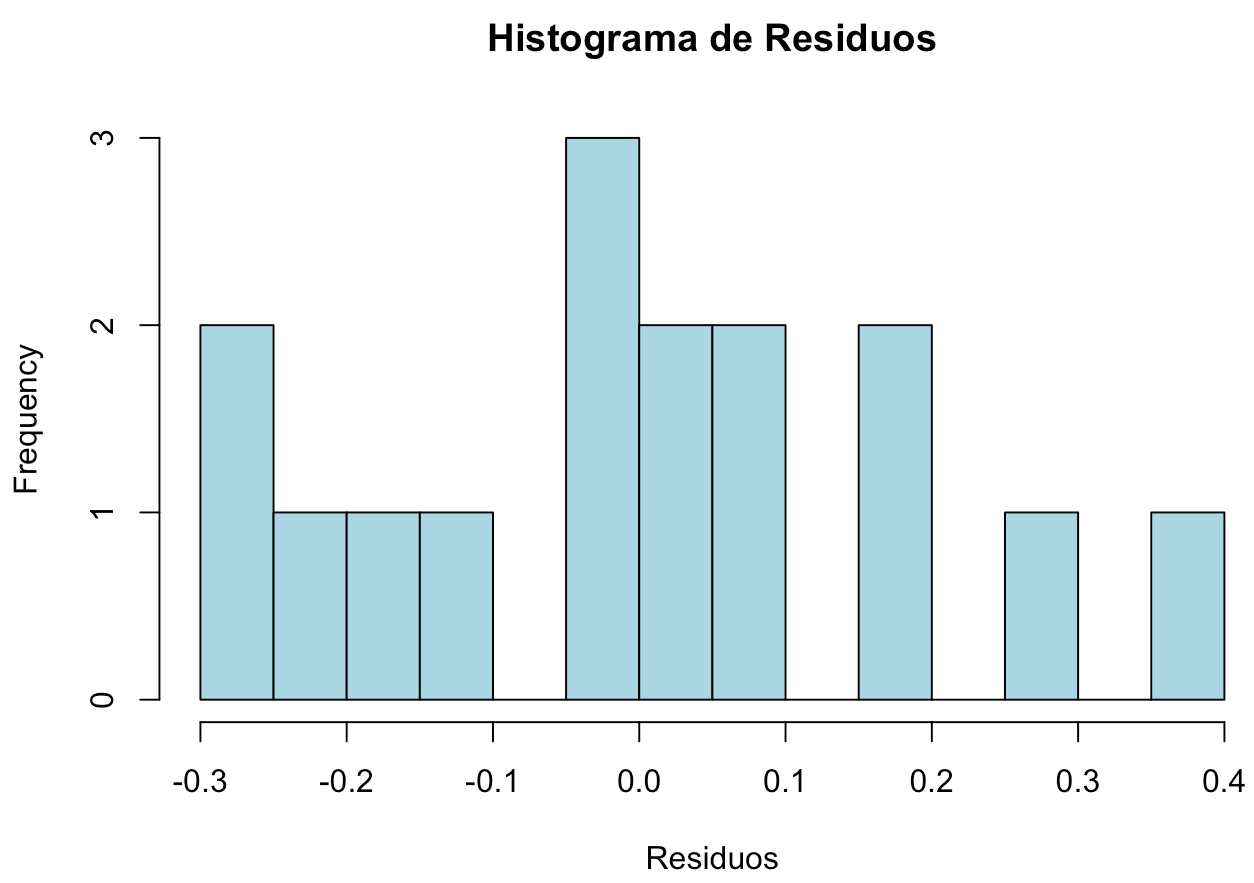
```
##  (Intercept)       tiempo
## 3.015451e-17 1.399111e-09
```

- Haz el análisis de residuos

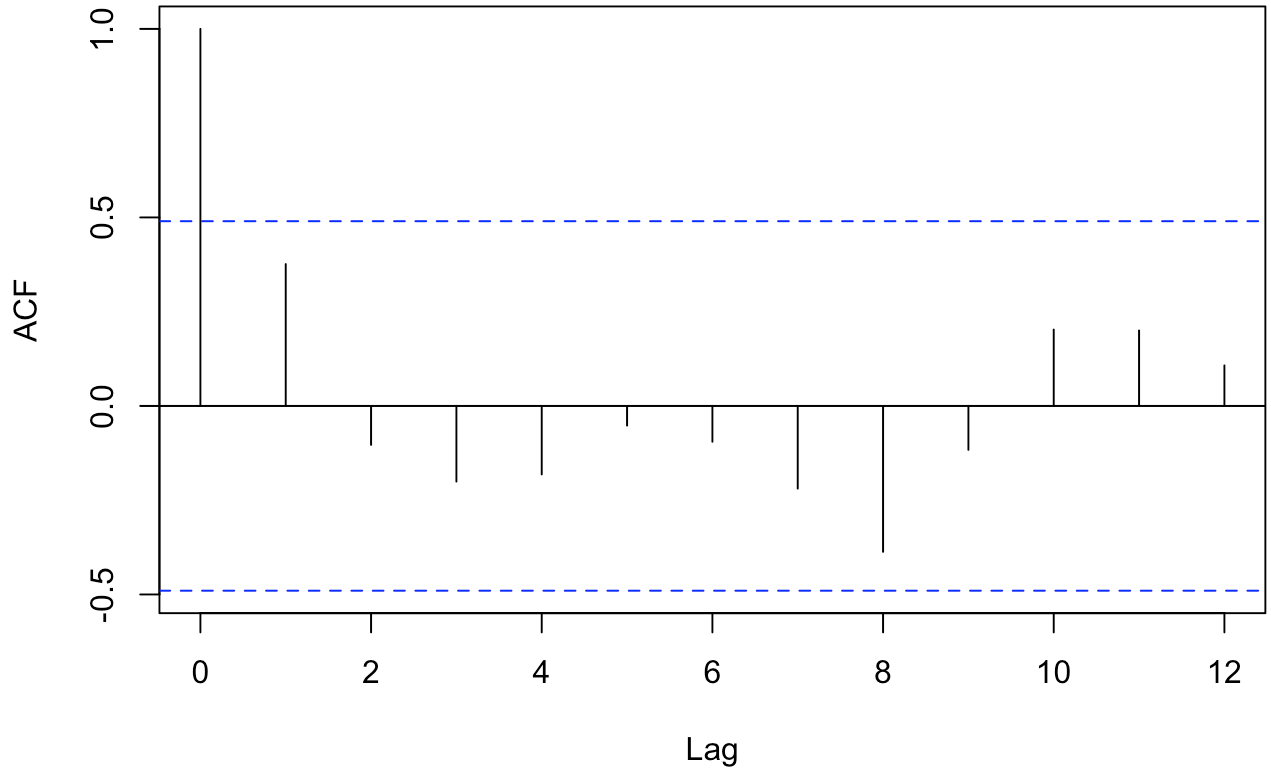
```
residuos = residuals(modelo_lineal)
plot(tiempo, residuos, main = "Gráfico de Residuos", xlab = "Tiempo", ylab = "Residuos",
col = "purple", pch = 16)
abline(h = 0, col = "red", lwd = 2)
```




```
hist(residuos, main = "Histograma de Residuos", xlab = "Residuos", col = "lightblue", br
eaks = 10)
```




```
acf(residuos, main = "Autocorrelación de los Residuos")
```




```
shapiro.test(residuos)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuos
## W = 0.97816, p-value = 0.9473
```

4. Calcula el CME y el EPAM de la predicción de la serie de tiempo.

```
predicciones = predict(modelo_lineal)
CME = mean((serie_desestacionalizada - predicciones)^2)
CME
```

```
## [1] 0.03291917
```

```
EPAM = mean(abs((serie_desestacionalizada - predicciones) / serie_desestacionalizada) *
100)
EPAM
```

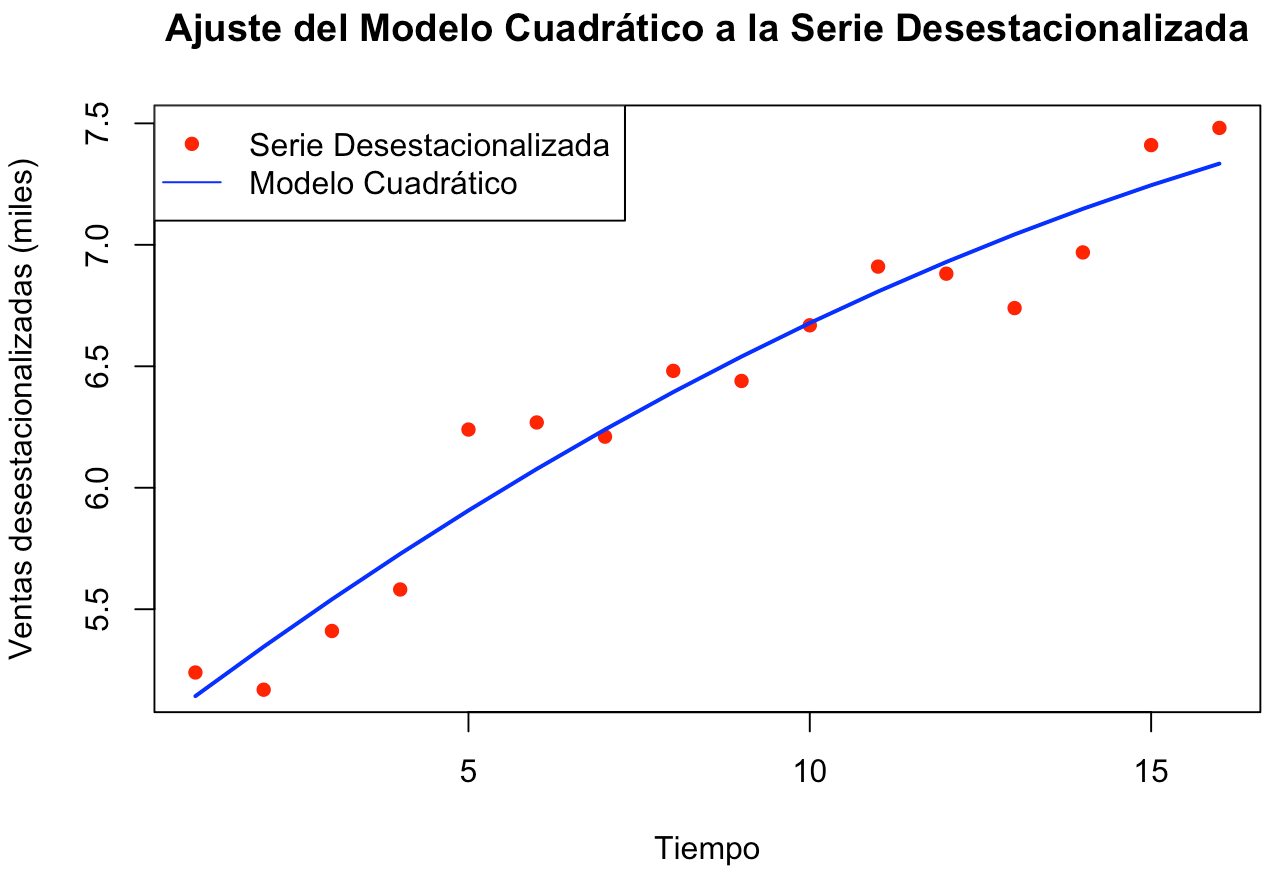
```
## [1] 2.341319
```

5. Explora un mejor modelo, por ejemplo un modelo cuadrático: $y = \beta_0 + \beta_1 x + \beta_2 x^2$. Para ello transforma la variable ventas (recuerda que la regresión no lineal es una regresión lineal con una tranformación).

```
tiempo = 1:length(serie_desestacionalizada)
tiempo_cuadrado = tiempo^2
modelo_cuadratico = lm(serie_desestacionalizada ~ tiempo + tiempo_cuadrado)
summary(modelo_cuadratico)
```

```
##
## Call:
## lm(formula = serie_desestacionalizada ~ tiempo + tiempo_cuadrado)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.30333 -0.13440 -0.01928  0.11368  0.33301
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.930833   0.155679  31.673 1.08e-13 ***
## tiempo           0.215572   0.042149   5.115 0.000199 ***
## tiempo_cuadrado -0.004085   0.002410  -1.695 0.113918
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1822 on 13 degrees of freedom
## Multiple R-squared:  0.9446, Adjusted R-squared:  0.9361
## F-statistic: 110.8 on 2 and 13 DF,  p-value: 6.805e-09
```

```
predicciones_cuadraticas = predict(modelo_cuadratico)
plot(tiempo, serie_desestacionalizada, main = "Ajuste del Modelo Cuadrático a la Serie D
esestacionalizada",
     xlab = "Tiempo", ylab = "Ventas desestacionalizadas (miles)", col = "red", pch = 1
6)
lines(tiempo, predicciones_cuadraticas, col = "blue", lwd = 2)
legend("topleft", legend = c("Serie Desestacionalizada", "Modelo Cuadrático"), col = c
("red", "blue"), pch = c(16, NA), lty = c(NA, 1))
```




```
CME_cuadratico = mean((serie_desestacionalizada - predicciones_cuadraticas)^2)
CME_cuadratico
```

```
## [1] 0.02696192
```

```
EPAM_cuadratico = mean(abs((serie_desestacionalizada - predicciones_cuadraticas) / serie
_desestacionalizada) * 100)
EPAM_cuadratico
```

```
## [1] 2.229868
```

6. Concluye sobre el mejor modelo

```
CME_lineal = mean((serie_desestacionalizada - predicciones)^2)
EPAM_lineal = mean(abs((serie_desestacionalizada - predicciones) / serie_desestacionaliz
ada) * 100)

CME_cuadratico = mean((serie_desestacionalizada - predicciones_cuadraticas)^2)
EPAM_cuadratico <- mean(abs((serie_desestacionalizada - predicciones_cuadraticas) / seri
e_desestacionalizada) * 100)

AIC_lineal = AIC(modelo_lineal)
AIC_cuadratico = AIC(modelo_cuadratico)
BIC_lineal = BIC(modelo_lineal)
BIC_cuadratico = BIC(modelo_cuadratico)

cat("Modelo Lineal:\n")
```

```
## Modelo Lineal:
```

```
cat("CME:", CME_lineal, "\nEPAM:", EPAM_lineal, "\nAIC:", AIC_lineal, "\nBIC:", BIC_line
al, "\n\n")
```

```
## CME: 0.03291917
## EPAM: 2.341319
## AIC: -3.213169
## BIC: -0.8954033
```

```
cat("Modelo Cuadrático:\n")
```

```
## Modelo Cuadrático:
```

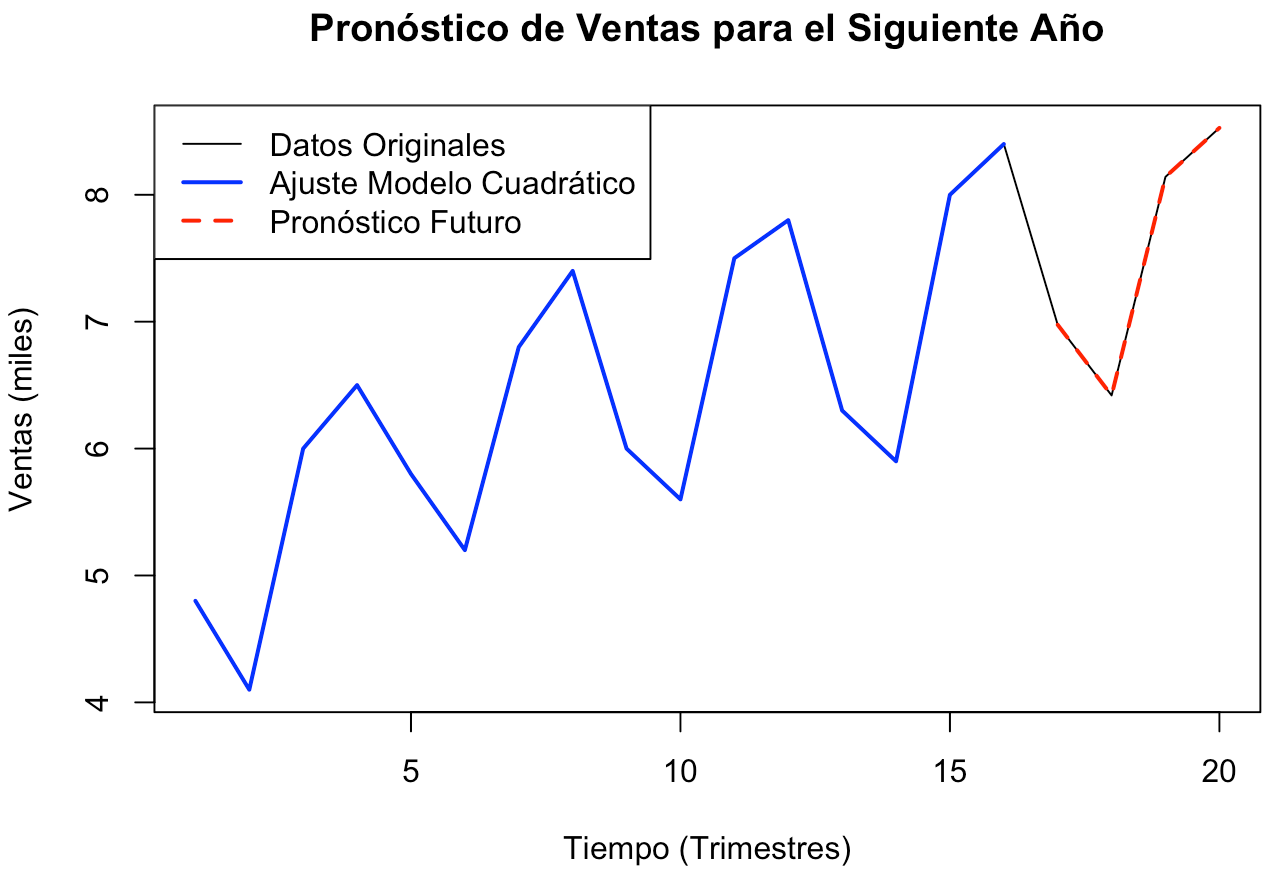
```
cat("CME:", CME_cuadratico, "\nEPAM:", EPAM_cuadratico, "\nAIC:", AIC_cuadratico, "\nBI
C:", BIC_cuadratico, "\n")
```

```
## CME: 0.02696192
## EPAM: 2.229868
## AIC: -4.407241
## BIC: -1.316886
```

Con base a los indicadores, diría que el modelo cuadrático es el mejor para nuestro conjunto de datos, o al menos sería la elección recomendada para realizar pronósticos en esta serie de tiempo. Ya que tiene un mejor ajuste (menor CME y EPAM, es mejor) y lo prefiero según los criterios AIC y BIC, lo que nos dice que captura mejor la relación en los datos sin un exceso de complejidad.

7. Realiza el pronóstico para el siguiente año y grafícalo junto con los pronósticos previos y los datos originales.

```
tiempo_futuro = (length(serie_desestacionalizada) + 1):(length(serie_desestacionalizada)
+ 4)
tiempo_cuadrado_futuro = tiempo_futuro^2
datos_futuros = data.frame(tiempo = tiempo_futuro, tiempo_cuadrado = tiempo_cuadrado_fut
uro)
pronostico_futuro = predict(modelo_cuadratico, newdata = datos_futuros)
indices_estacionales_futuros = tail(indices_estacionales, 4)
pronostico_final = pronostico_futuro + indices_estacionales_futuros
ventas_totales = c(serie_desestacionalizada + indices_estacionales, pronostico_final)
tiempo_total = 1:length(ventas_totales)
plot(tiempo_total, ventas_totales, type = "l", col = "black", ylim = range(ventas_totale
s),
     main = "Pronóstico de Ventas para el Siguiente Año", xlab = "Tiempo (Trimestres)",
ylab = "Ventas (miles)")
lines(1:length(serie_desestacionalizada), serie_desestacionalizada + indices_estacionale
s, col = "blue", lwd = 2)
lines(tiempo_futuro, pronostico_final, col = "red", lwd = 2, lty = 2)
legend("topleft", legend = c("Datos Originales", "Ajuste Modelo Cuadrático", "Pronóstico
Futuro"),
       col = c("black", "blue", "red"), lty = c(1, 1, 2), lwd = c(1, 2, 2))
```


Pronóstico de Ventas para el Siguiente Año
Datos Originales
Ajuste Modelo Cuadrático
Pronóstico Futuro
Ventas (miles)
4
5
6
7
8
5
10
15
20
Tiempo (Trimestres)