Primas Mapfre

José María Álvarez Silva

Predicción

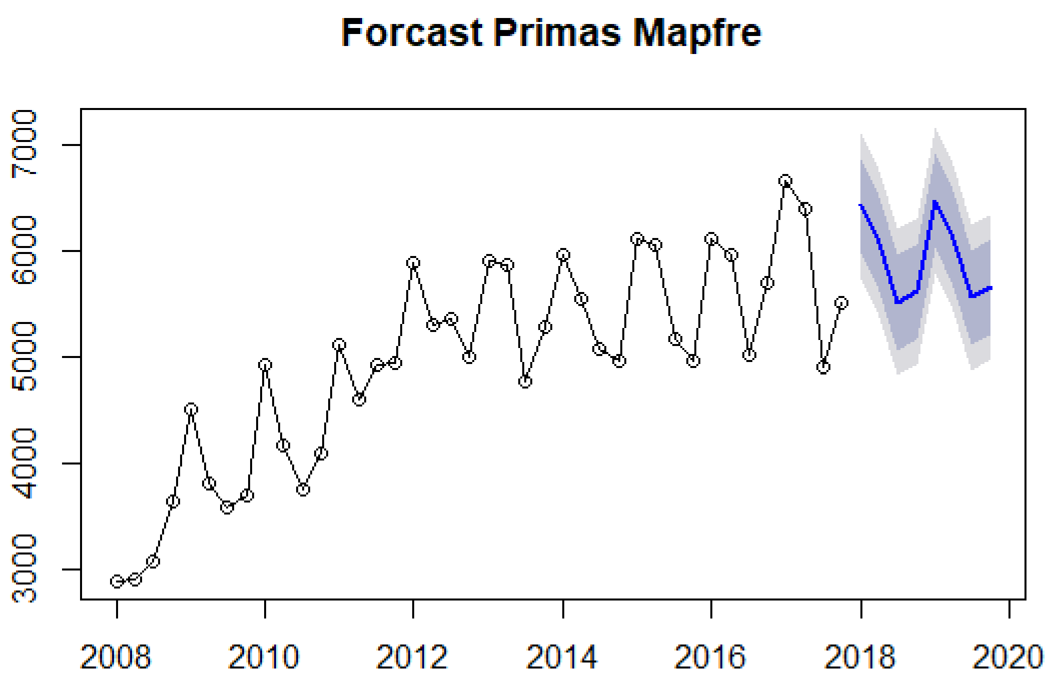
Propósito

Predicción del número de primas total, vida y no vida por separado para después sumar y comparar. Realizar un análisis para seleccionar el mejor modelo (capacidad predictiva) con el cual se realizará la predicción.

Predicción

Se llevo a cabo la predicción de las primas de Mapfre para los trimestres de los próximos dos años. La predicción:

	Q1	Q2	Q3	Q4
2018	6431.081	6070.836	5516.262	5619.637
2019	6484.425	6120.941	5563.325	5663.844



El modelo utilizado fue un modelo ETS sobre el total de primas de Mapfre.

Resumen Ejecutivo

Proceso General

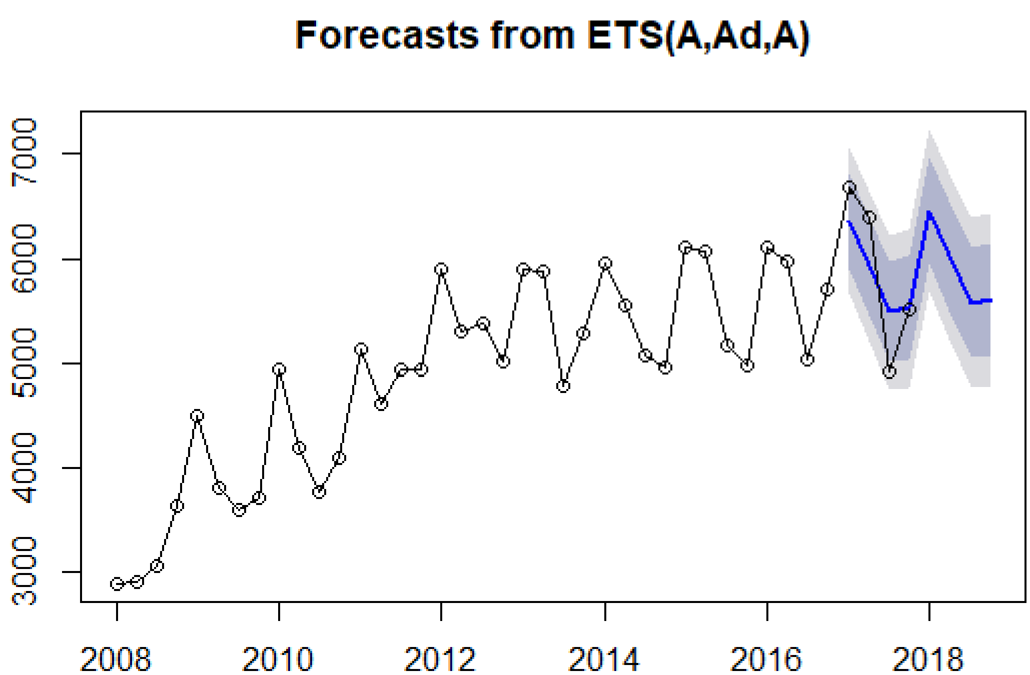
En este análisis utilizamos dos enfoques para la predicción de las primas de los seguros vendidos por Mapfre. Se trabajó primero con el total de primas (sin distinguir por tipo de producto) y, por otro lado, distinguiendo entre primas de seguros de vida y seguros de no vida. Los enfoques utilizados para predicir:

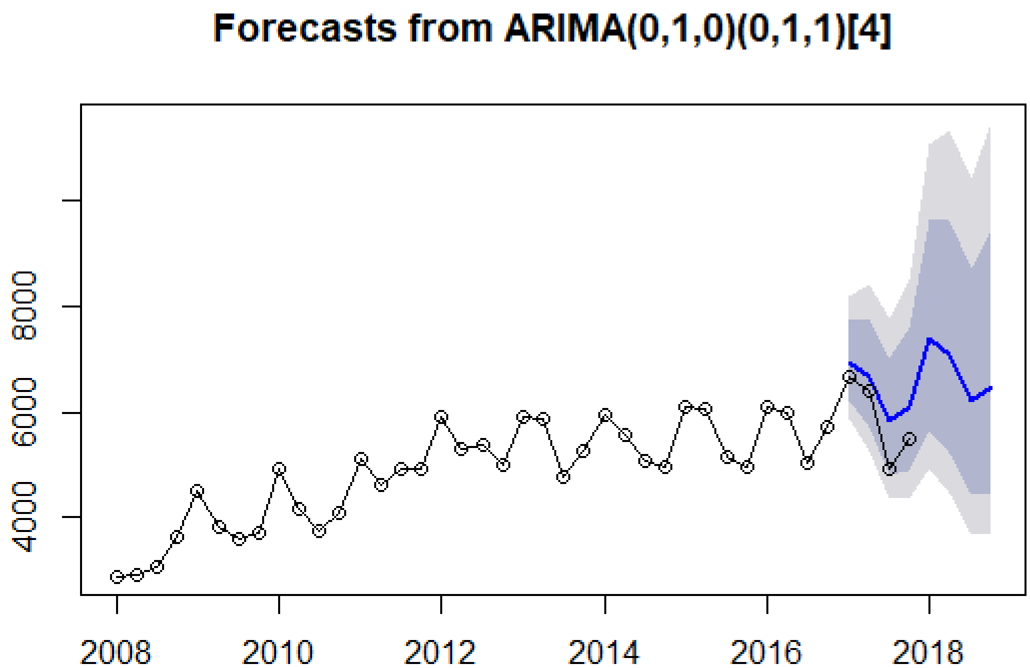
- Exponential Smoothing (Ets - Error, Trend and Season).
- ARIMA.

Resultados

Comparando el poder predictivo de cada uno de los modelos (generados con el set de entrenamiento) a través de la métricas de error de predicción (MSE, MAE y Bias) en el set de Test (primas vendidas por trimestre en el año 2017), el modelo que tuvo mejor desempeño fue el ETS para el total de primas vendidas (i.e. sin distinción de productos). Como se muestra en la tabla a continuación:

	MSE	MAE	Bias
Total ETS	163016.7	345.3866	45.9941
Suma ETS	251082.2	464.3306	17.35465
Total ARIMA	316798	492.7053	492.7053
Suma ARIMA	353436.3	522.6284	522.6284

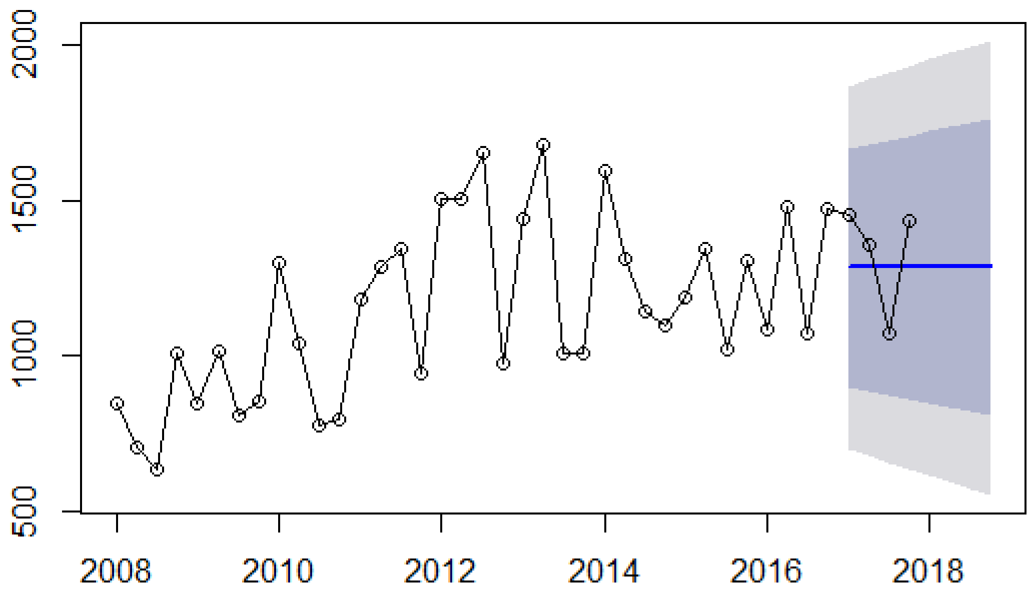




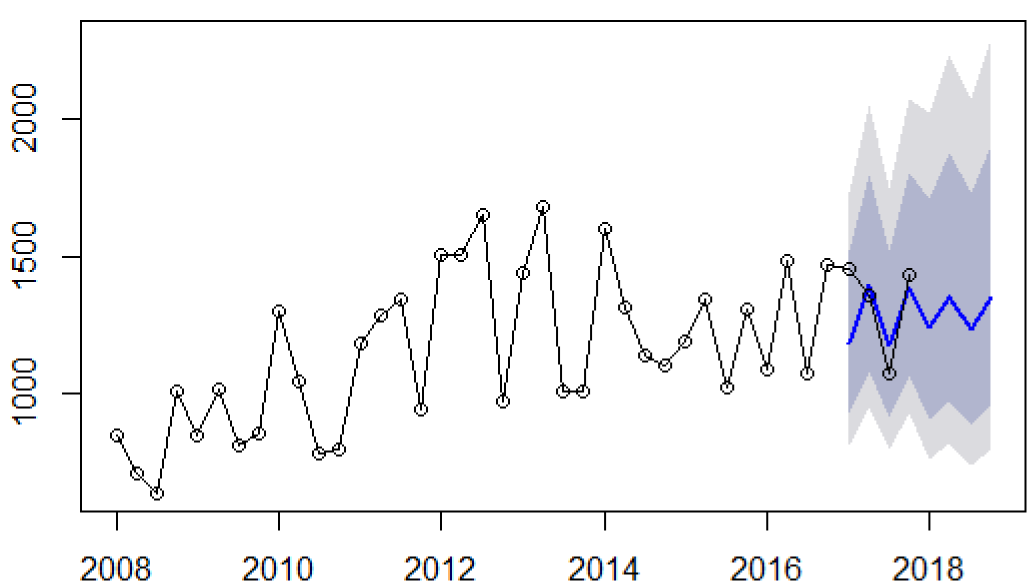
Comparando el poder predictivo para el producto de seguros de Vida. El modelo ARIMA es el que moejor poder predictivo tiene.

	MSE	MAE	Bias
Vida ETS	25336.91	151.6686	-47.48728
Vida ARIMA	21651.86	112.5497	-44.29429

Forecasts from ETS(M,N,N)



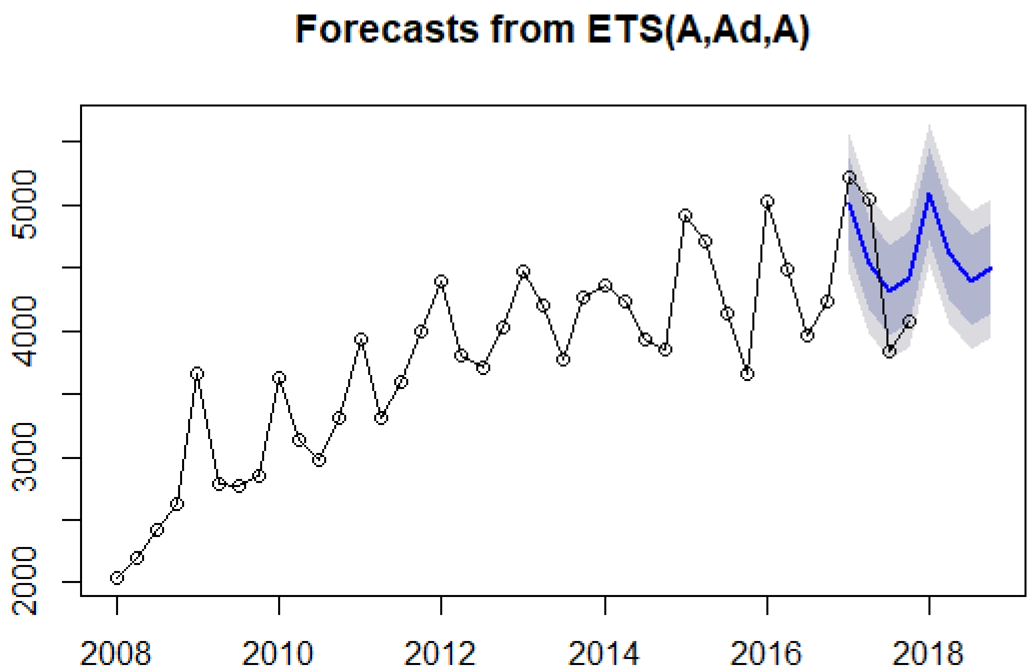
Forecasts from ARIMA(0,1,1)(1,0,0)[4]

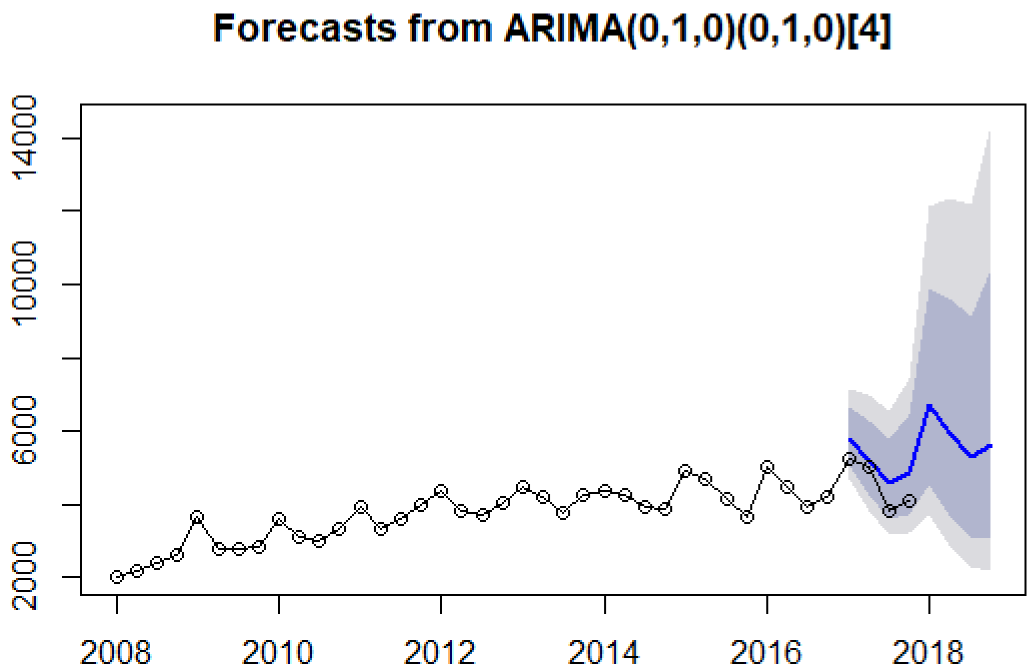


El modelo ARIMA parece explicar mejor la componente estacional de los seguros de vida. El gráfico del modelo ETS devuelve una prediccion plana para este tipo de seguro.

Comparando el poder predictivo para el producto de seguros de Vida. El modelo ARIMA es el que mejor poder predictivo tiene.

	MSE	MAE	Bias
Vida ETS	164442.5	387.3306	30.13263
Vida ARIMA	388540.4	566.9227	566.9227

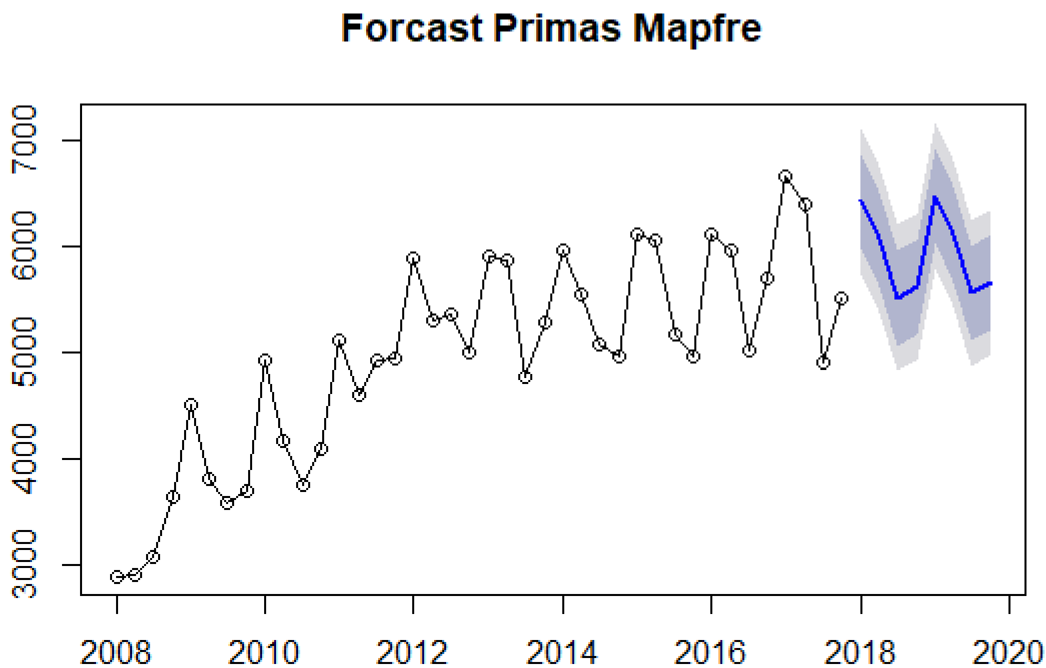




Predicción 2018 y 2019

Al ser el mejor modelo en Test el ETS del total, se llevo a cabo la predicción para los trimestres de los próximos dos años de las primas totales de Mapfre. La predicción:

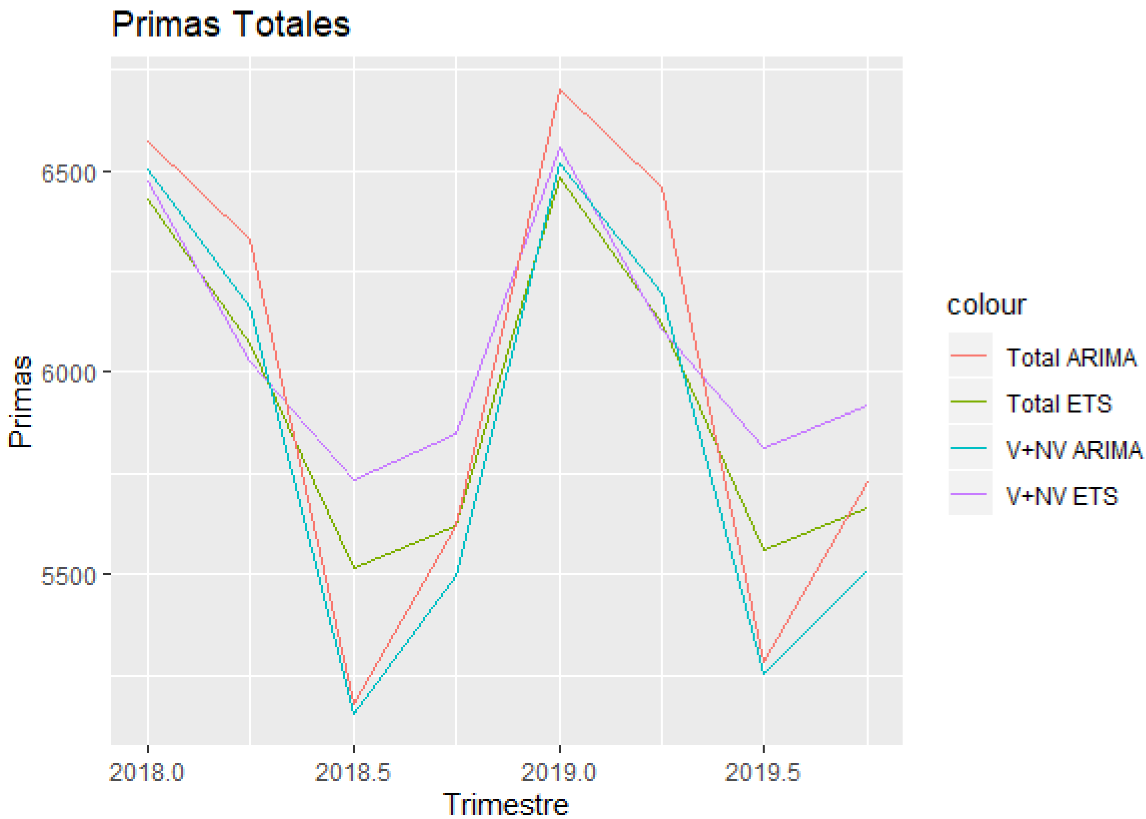
	Q1	Q2	Q3	Q4
2018	6431.081	6070.836	5516.262	5619.637
2019	6484.425	6120.941	5563.325	5663.844

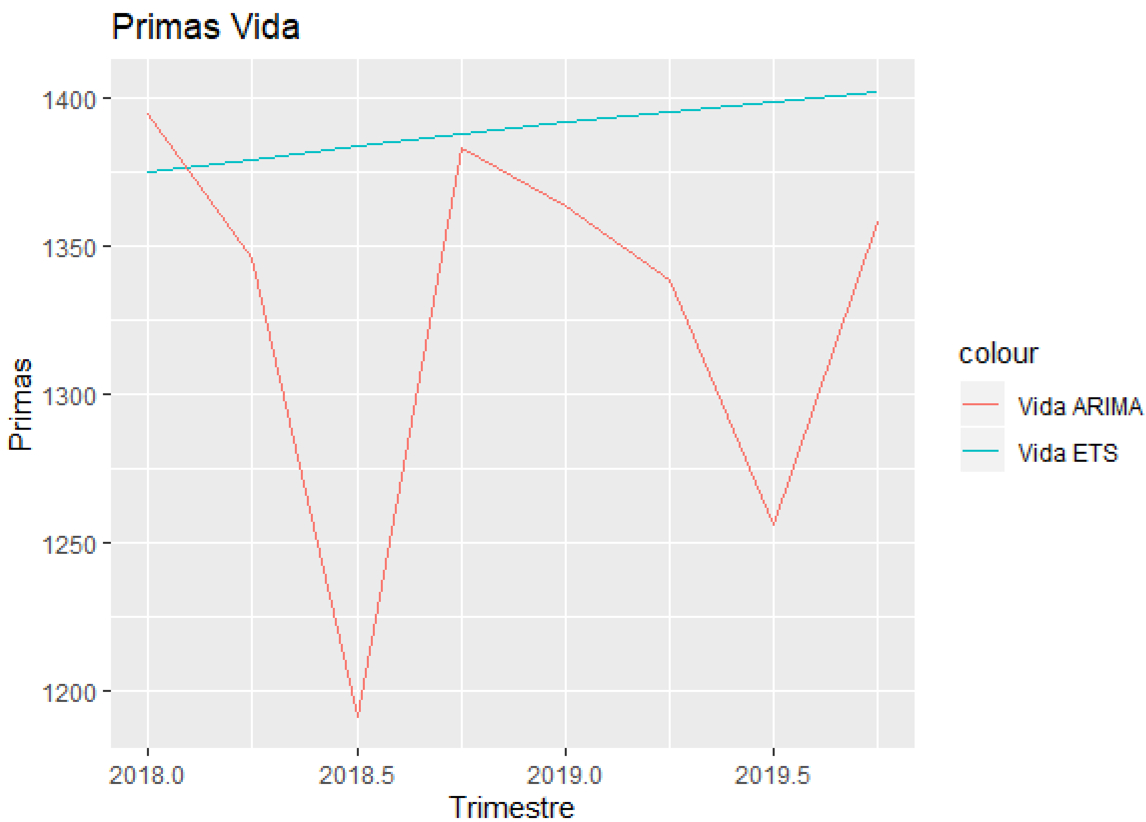


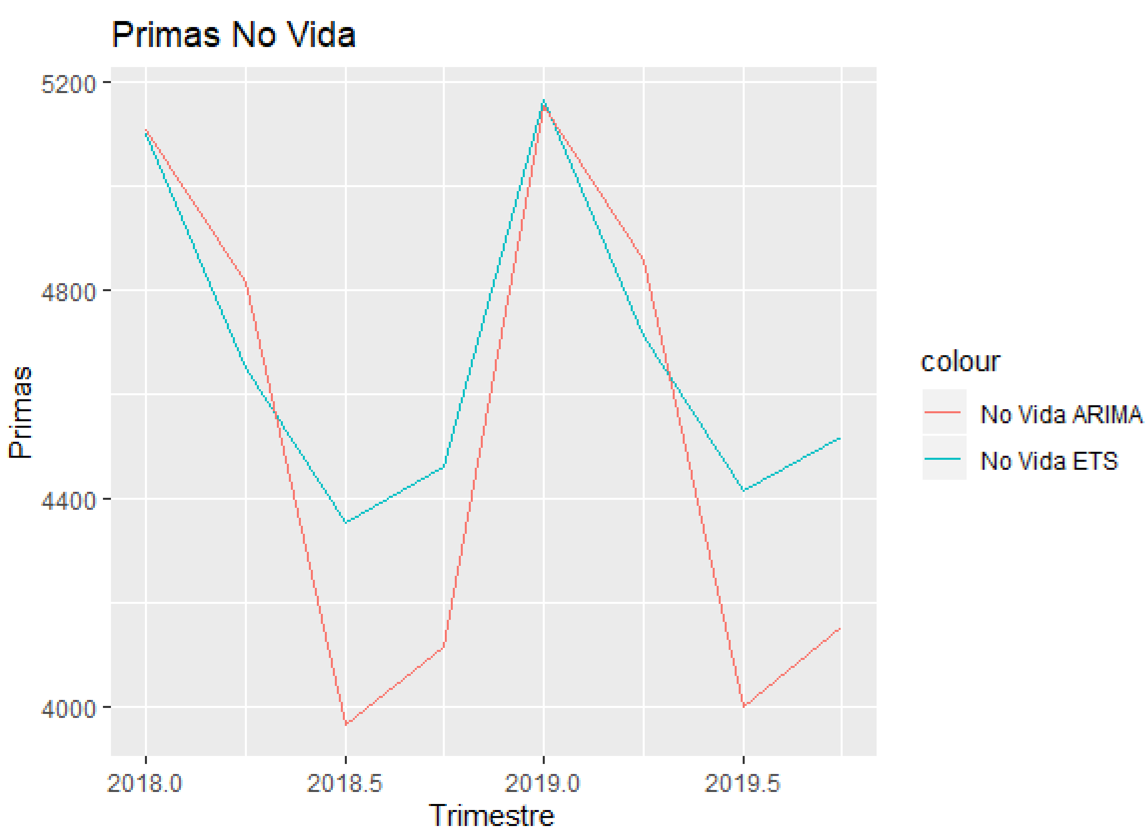
Observaciones

Basados en las comparativas de métricas de errores de prediccón se ha mostrado cual sería la predicción para los próximos dos años según el mejor modelos. Algo que resulta interesante es que para total y no vida, el mejor modelo era ETS en ambos casos. Sin embargo, el mejor modelo para predecir las primas de vida es un modelo ARIMA (se ha dado una explicación del porque creemos que sucede esto).

Comparación de predicciones de los modelos por tipo de producto




Primas Vida
1400
1350
1300
1250
1200
Primas
2018.0
2018.5
2019.0
2019.5
Trimestre
colour
Vida ARIMA
Vida ETS


Primas No Vida
5200
4800
4400
4000
Primas
2018.0
2018.5
2019.0
2019.5
Trimestre
colour
No Vida ARIMA
No Vida ETS

Anexos y Código

Paquetes y Datos

```
## Paquetes ##########################################################################################
##
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------ tidyverse 

## v ggplot2 3.2.1     v readr   1.3.1
## v tibble  2.1.3     v purrr   0.3.2
## v tidyr   1.0.0     v stringr 1.4.0
## v ggplot2 3.2.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------ tidyverse_confli
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
require(forecast)
```

```
## Loading required package: forecast

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff
```

```r
require(xts)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```r
require(ggplot2)
library(zoo)
library(hts)
library(ggfortify)
```

```
## Registered S3 methods overwritten by 'ggfortify':
##   method                 from
##   autoplot.Arima         forecast
##   autoplot.acf           forecast
##   autoplot.ar            forecast
##   autoplot.bats          forecast
##   autoplot.decomposed.ts forecast
##   autoplot.ets           forecast
##   autoplot.forecast      forecast
##   autoplot.stl           forecast
##   autoplot.ts            forecast
##   fitted.ar              forecast
##   fortify.ts             forecast
##   residuals.ar           forecast
```

```r
## Datos ##############################################################################################
##
datos <- read.csv("Primas_mapfre.csv", sep = ";", dec = ",")
## Total
datos <- datos %>% mutate(total = Primas_vida + Primas_no_vida)
## Fecha
datos$Fecha <- as.Date(datos$Fecha, "%m/%d/%Y")
```

Análisi Exploratorio de Datos

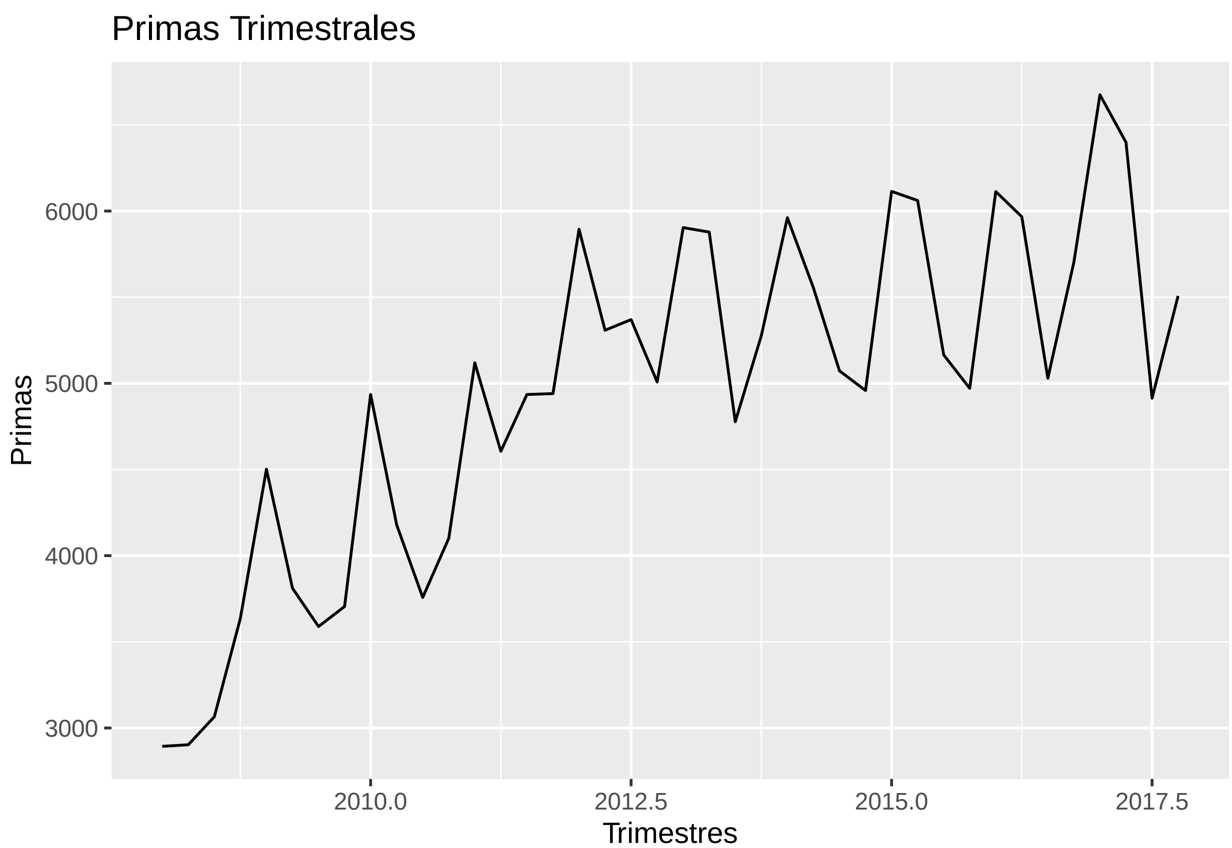
```r
## Análisis exploratorio de datos ##########################################################################
##
ggplot(data = datos, aes(x = Fecha)) +
  geom_line(aes(y = Primas_vida, colour = "Vida")) +
  geom_line(aes(y = Primas_no_vida, colour = "No Vida")) +
  geom_line(aes(y = total, colour = "Total"))
```

Modelo ETS para primas totales

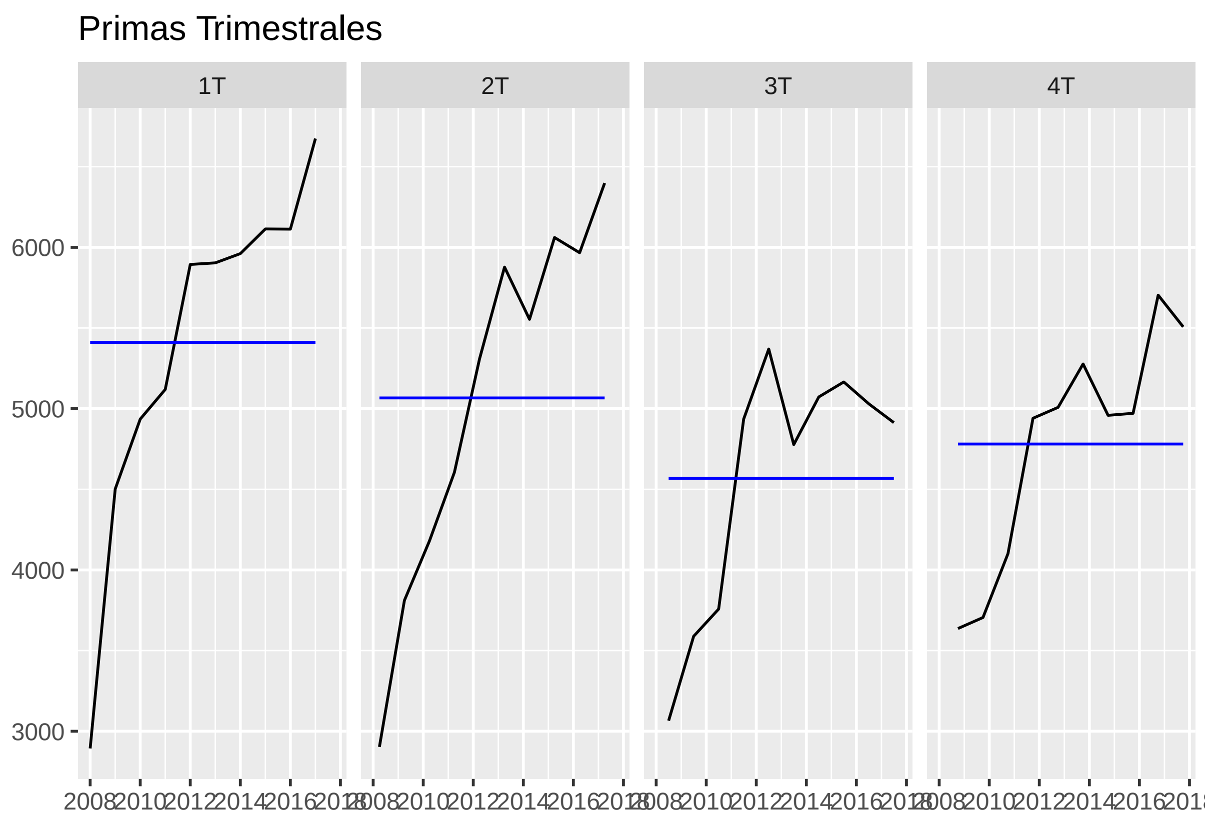
Preparando datos

```r
## Trabajando un modelo ETS para Total ####################################################################
##

## Plot Serie Total
total = xts((datos$total), order.by = datos$Fecha, frequency = 4)
## Generate quarterly data
total = to.quarterly(total)  ## OJO cambia a que sea trimestral
## paqueteria zoo para mejor funcionamiento
total = as.zoo(total$total.Close)
autoplot(total) + ggtitle("Primas Trimestrales") + xlab("Trimestres") + ylab("Primas")
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```




```r
## Seasonal Plot
ggfreqplot(as.ts(total), freq = 4, nrow = 1, facet.labeller = c("1T","2T","3T","4T")) +
  ggtitle("Primas Trimestrales")
```



Trainig set

```
#- Training set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## Select number of observation to compare forecast
cOmit = 4

## Data Size
nObs = length(total)

## sub_sample
oTotal <- window(total,start = index(total[1]),end = index(total[nObs - cOmit]))
```

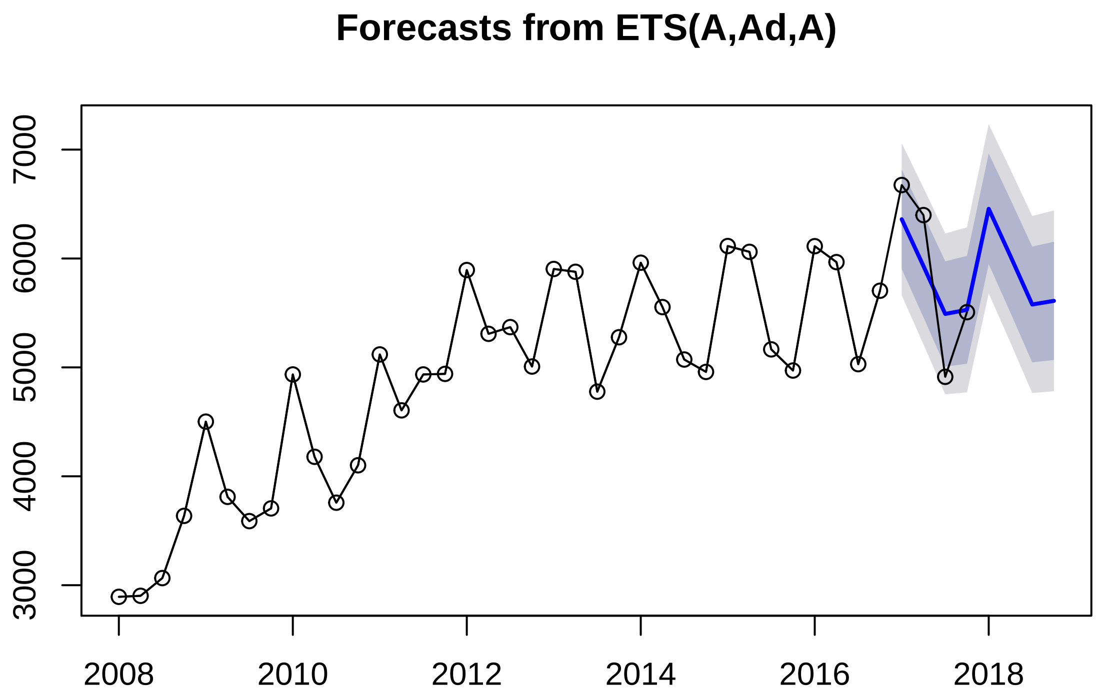
Test

```
Totaletsfit <- ets(oTotal)
#f orecast model
fTotal.ets = forecast(Totaletsfit)
# Results
summary(fTotal.ets)
```

```
##
```

```
## Forecast method: ETS(A,Ad,A)
##
## Model Information:
## ETS(A,Ad,A)
##
## Call:
##  ets(y = oTotal)
##
##   Smoothing parameters:
##     alpha = 0.2424
##     beta  = 1e-04
##     gamma = 1e-04
##     phi   = 0.9505
##
##   Initial states:
##     l = 2931.1472
##     b = 178.7185
##     s = -335.3013 -349.5291 114.6065 570.2239
##
##   sigma:  356.3793
##
##      AIC     AICc      BIC
## 561.7218 570.5218 577.5570
##
## Error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -5.403611 308.6335 249.7589 -0.4948873 5.595178 0.6033052
##                   ACF1
## Training set 0.1276507
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2017 Q1       6360.067 5903.349 6816.786 5661.577 7058.558
## 2017 Q2       5930.371 5460.419 6400.324 5211.641 6649.102
## 2017 Q3       5490.845 5008.011 5973.679 4752.414 6229.275
## 2017 Q4       5528.540 5033.151 6023.929 4770.908 6286.172
## 2018 Q1       6456.184 5948.533 6963.836 5679.798 7232.570
## 2018 Q2       6021.727 5502.104 6541.349 5227.033 6816.421
## 2018 Q3       5577.675 5046.343 6109.006 4765.074 6390.276
## 2018 Q4       5611.068 5068.274 6153.862 4780.937 6441.200
```

```r
## Eligio un MAM
## - M - multiplicativa la tendencia
## - A - aditiva en la pendiente
## - M - multiplicativa en estacionalidad

## ojo -> simpre que se hagan predicciones se debe dar un intervalo de confianza

# Plot
plot(fTotal.ets)
lines(window(total),type = "o")
```



Metricas de predicción

```
#Actual and Forecast
totalFitmat <- matrix(c(fTotal.ets$mean[1:cOmit],total[(nObs - cOmit + 1):nObs]),ncol = 2)
totalFitmat
```

```
##          [,1]   [,2]
## [1,] 6360.067 6674.6
## [2,] 5930.371 6398.6
## [3,] 5490.845 4913.4
## [4,] 5528.540 5507.2
```

```
## MSE
mean((totalFitmat[,1] - totalFitmat[,2])^2)
```

```
## [1] 163016.7
```

```
## MAE
mean(abs(totalFitmat[,1] - totalFitmat[,2]))
```

```
## [1] 345.3866
```

```r
## Bias
mean(totalFitmat[,1] - totalFitmat[,2])
```

```
## [1] -45.9941
```

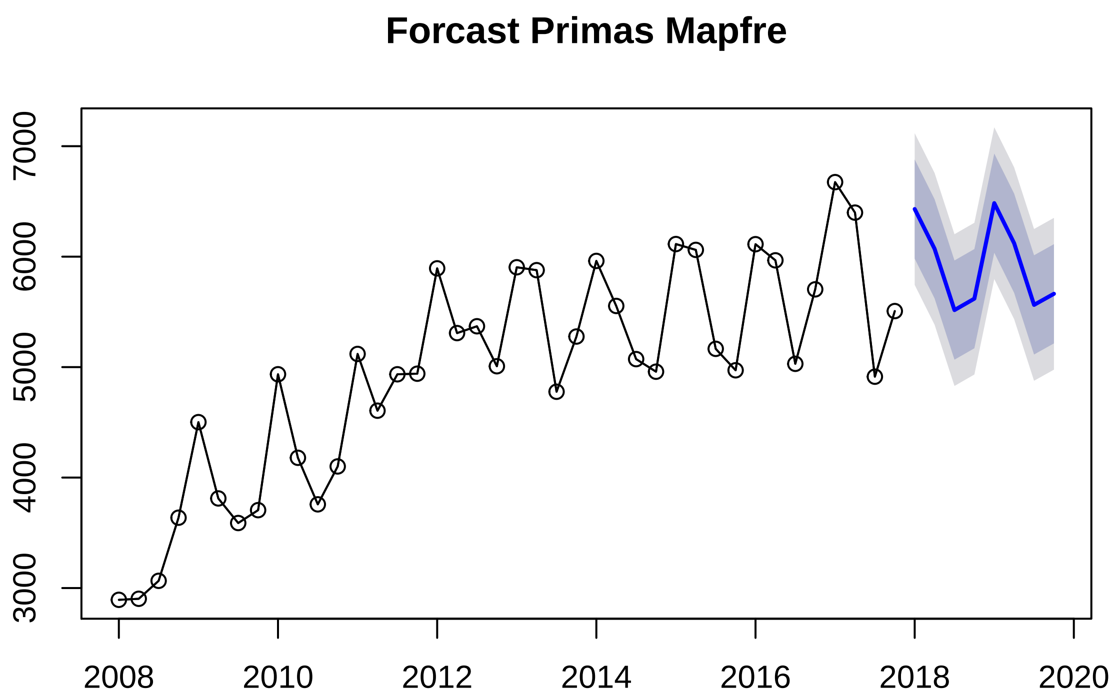
Predicción

```r
#- Complete set        -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## Select automatic ETS
etsfit <- ets(total)
## forecast model
total.ets <- forecast(etsfit)
## Results
summary(total.ets)
```

```
##
## Forecast method: ETS(A,Ad,A)
##
## Model Information:
## ETS(A,Ad,A)
##
## Call:
##  ets(y = total)
##
##   Smoothing parameters:
##     alpha = 0.0023
##     beta  = 0.0023
##     gamma = 1e-04
##     phi   = 0.9393
##
##   Initial states:
##     l = 2909.4154
##     b = 210.9936
##     s = -310.0402 -400.4518 167.846 542.646
##
##   sigma:  350.3869
##
##      AIC     AICc      BIC
## 626.0825 633.6687 642.9713
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -30.1137 308.4599 252.1897 -1.409148 5.580647 0.6237988
##                   ACF1
## Training set 0.2312009
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2018 Q1       6431.081 5982.042 6880.120 5744.335 7117.827
## 2018 Q2       6070.836 5621.793 6519.879 5384.084 6757.588
```

```
## 2018 Q3       5516.262 5067.210 5965.314 4829.496 6203.028
## 2018 Q4       5619.637 5170.570 6068.705 4932.848 6306.427
## 2019 Q1       6484.425 6035.334 6933.515 5797.600 7171.249
## 2019 Q2       6120.941 5671.820 6570.062 5434.070 6807.813
## 2019 Q3       5563.325 5114.165 6012.485 4876.394 6250.256
## 2019 Q4       5663.844 5214.635 6113.052 4976.839 6350.848
```

```
## Eligio un MAM
## - M - multiplicativa la tendencia
## - A - aditiva en la pendiente
## - M - multiplicativa en estacionalidad

## ojo -> simpre que se hagan predicciones se debe dar un intervalo de confianza

# Plot
plot(total.ets, main = "Forcast Primas Mapfre")
lines(window(total),type = "o")
```

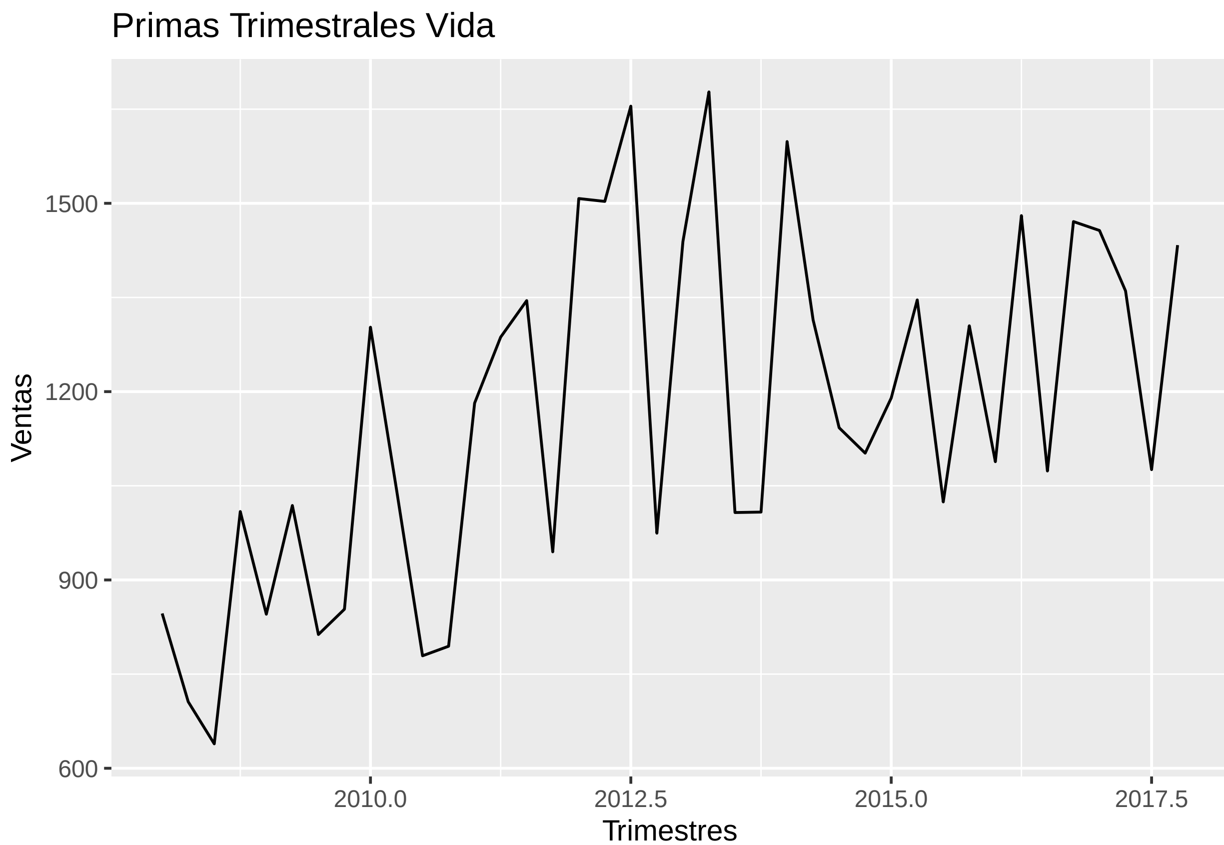


Modelo ETS para primas vida

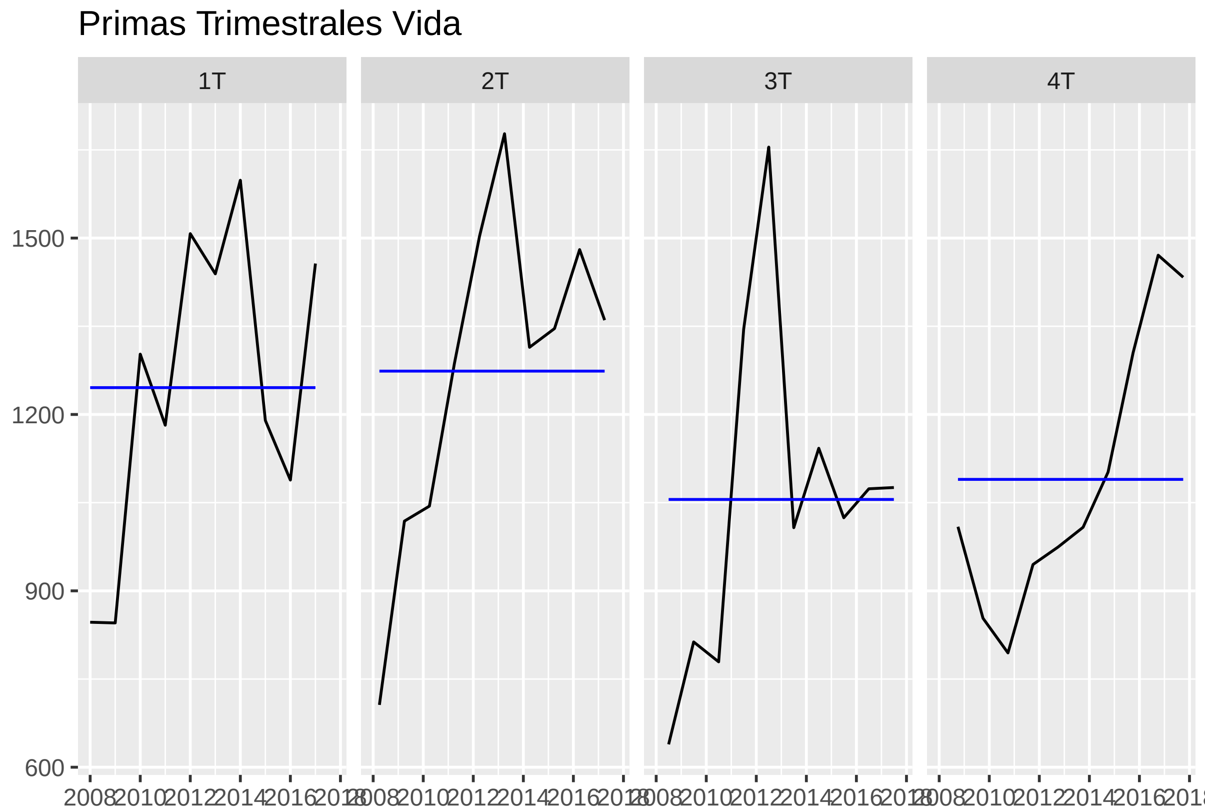
Preparando datos

```r
## Trabajando un modelo ETS por separado ##############################################################
##

## Plot Serie Primas Vida
vida = xts((datos$Primas_vida), order.by = datos$Fecha, frequency = 4)
## Generate quarterly data
vida = to.quarterly(vida)  ## OJO cambia a que sea trimestral
## paqueteria zoo para mejor funcionamiento
vida = as.zoo(vida$vida.Close)
autoplot(vida) + ggtitle("Primas Trimestrales Vida") + xlab("Trimestres") + ylab("Ventas")
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```




```r
## Seasonal Plot
ggfreqplot(as.ts(vida), freq = 4, nrow = 1, facet.labeller = c("1T","2T","3T","4T")) +
  ggtitle("Primas Trimestrales Vida")
```



Trainig set

```
#- Training set      -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## Select number of observation to compare forecast
cOmit = 4

## Data Size
nObs = length(vida)

## sub_sample
ovida <- window(vida,start = index(vida[1]),end = index(vida[nObs - cOmit]))
```

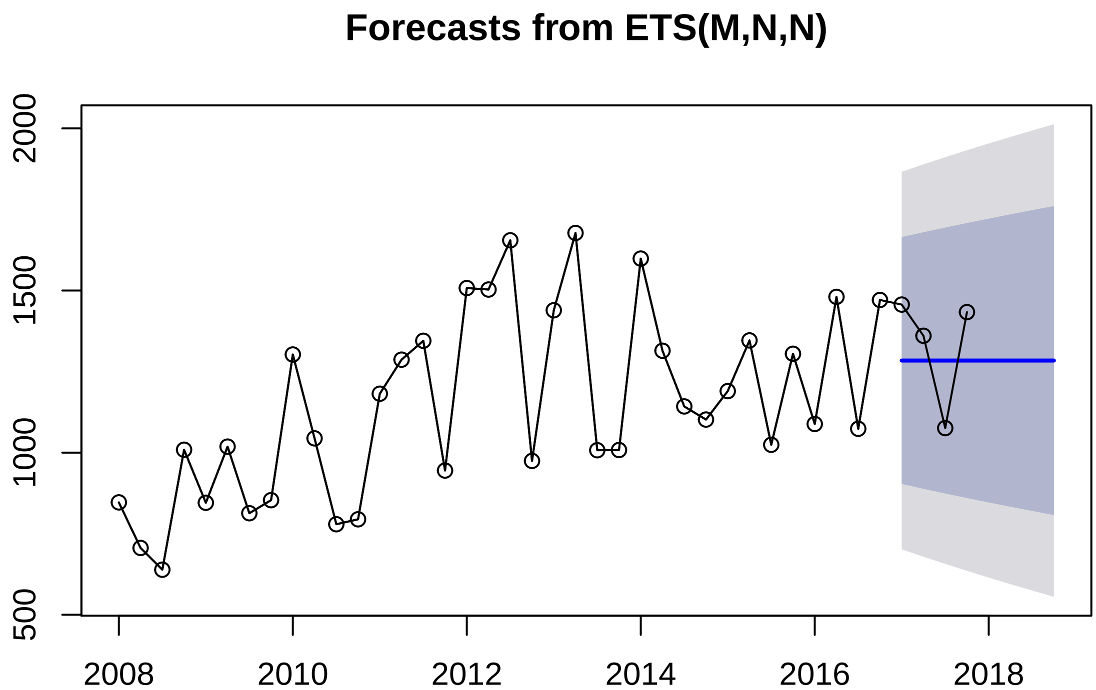
Test

```
vidaetsfit <- ets(ovida)
#f orecast model
fvida.ets = forecast(vidaetsfit)
# Results
summary(fvida.ets)
```

```
##
```

```
## Forecast method: ETS(M,N,N)
##
## Model Information:
## ETS(M,N,N)
##
## Call:
##  ets(y = ovida)
##
##   Smoothing parameters:
##     alpha = 0.2755
##
##   Initial states:
##     l = 815.9
##
##   sigma:  0.2314
##
##      AIC     AICc      BIC
## 530.5641 531.3141 535.3147
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 47.20866 246.8192 212.6659 0.7133133 18.32793 1.121323
##                    ACF1
## Training set -0.1436506
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2017 Q1       1284.063 903.3433 1664.782 701.8026 1866.323
## 2017 Q2       1284.063 888.4176 1679.708 678.9757 1889.150
## 2017 Q3       1284.063 873.9775 1694.148 656.8915 1911.234
## 2017 Q4       1284.063 859.9731 1708.152 635.4736 1932.652
## 2018 Q1       1284.063 846.3624 1721.763 614.6578 1953.468
## 2018 Q2       1284.063 833.1095 1735.016 594.3893 1973.736
## 2018 Q3       1284.063 820.1836 1747.942 574.6208 1993.505
## 2018 Q4       1284.063 807.5578 1760.568 555.3114 2012.814
```

```
## Eligio un MAM
## - M - multiplicativa la tendencia
## - A - aditiva en la pendiente
## - M - multiplicativa en estacionalidad

## ojo -> simpre que se hagan predicciones se debe dar un intervalo de confianza

# Plot
plot(fvida.ets)
lines(window(vida),type = "o")
```



Métrica de predicción

```
#Actual and Forecast
vidaFitmat <- matrix(c(fvida.ets$mean[1:cOmit],vida[(nObs - cOmit + 1):nObs]),ncol = 2)
vidaFitmat
```

```
##          [,1]   [,2]
## [1,] 1284.063 1456.7
## [2,] 1284.063 1360.4
## [3,] 1284.063 1075.7
## [4,] 1284.063 1433.4
```

```
## MSE
mean((vidaFitmat[,1] - vidaFitmat[,2])^2)
```

```
## [1] 25336.91
```

```
## MAE
mean(abs(vidaFitmat[,1] - vidaFitmat[,2]))
```

```
## [1] 151.6686
```

```r
## BIAS
mean((vidaFitmat[,1] - vidaFitmat[,2]))
```

```
## [1] -47.48728
```

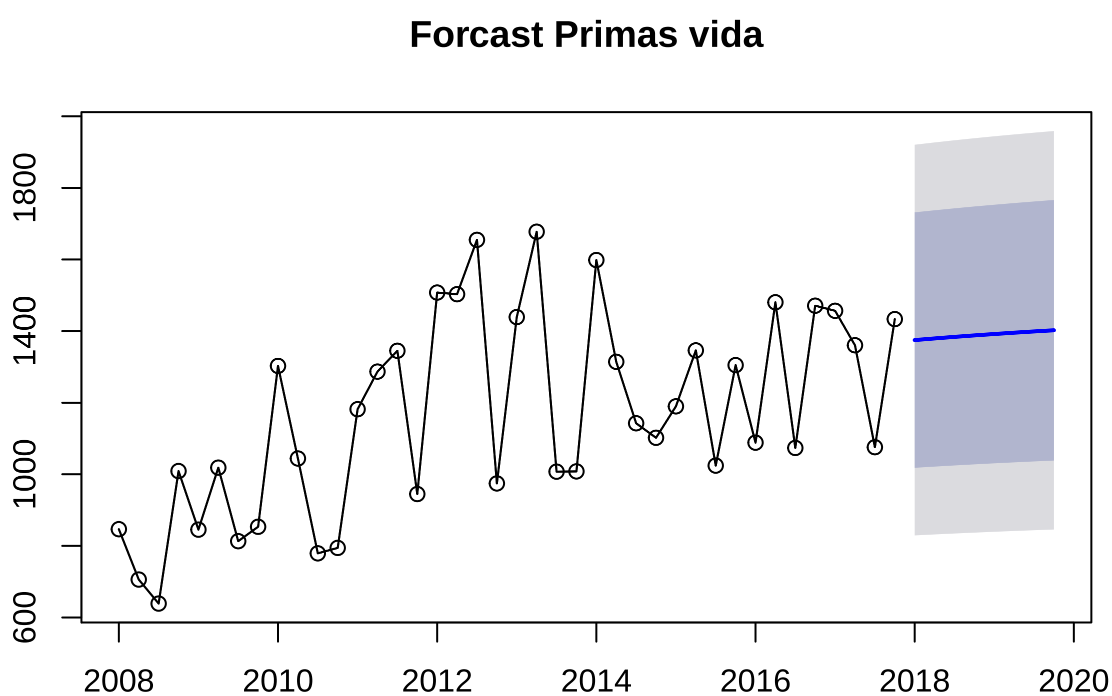
Predicción

```r
#- Complete set        -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## Select automatic ETS
etsfit <- ets(vida)
## forecast model
vida.ets <- forecast(etsfit)
## Results
summary(vida.ets)
```

```
##
## Forecast method: ETS(M,Ad,N)
##
## Model Information:
## ETS(M,Ad,N)
##
## Call:
##  ets(y = vida)
##
##   Smoothing parameters:
##     alpha = 1e-04
##     beta  = 1e-04
##     phi   = 0.9488
##
##   Initial states:
##     l = 691.9139
##     b = 41.6224
##
##   sigma:  0.2025
##
##      AIC     AICc      BIC
## 589.1138 591.6593 599.2471
##
## Error measures:
##                    ME    RMSE      MAE       MPE     MAPE     MASE
## Training set 16.04757 218.207 185.0561 -1.754454 16.14654 1.009871
##                     ACF1
## Training set -0.02219114
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2018 Q1       1374.896 1018.090 1731.702 829.2089 1920.583
## 2018 Q2       1379.456 1021.467 1737.445 831.9590 1926.953
## 2018 Q3       1383.782 1024.670 1742.894 834.5681 1932.996
## 2018 Q4       1387.887 1027.710 1748.064 837.0436 1938.730
```

```
## 2019 Q1       1391.781 1030.593 1752.969 839.3922 1944.170
## 2019 Q2       1395.476 1033.329 1757.623 841.6204 1949.331
## 2019 Q3       1398.981 1035.925 1762.038 843.7345 1954.228
## 2019 Q4       1402.307 1038.388 1766.227 845.7402 1958.874
```

```
## Eligio un MAM
## - M - multiplicativa la tendencia
## - A - aditiva en la pendiente
## - M - multiplicativa en estacionalidad

## ojo -> simpre que se hagan predicciones se debe dar un intervalo de confianza

# Plot
plot(vida.ets, main = "Forcast Primas vida")
lines(window(vida),type = "o")
```



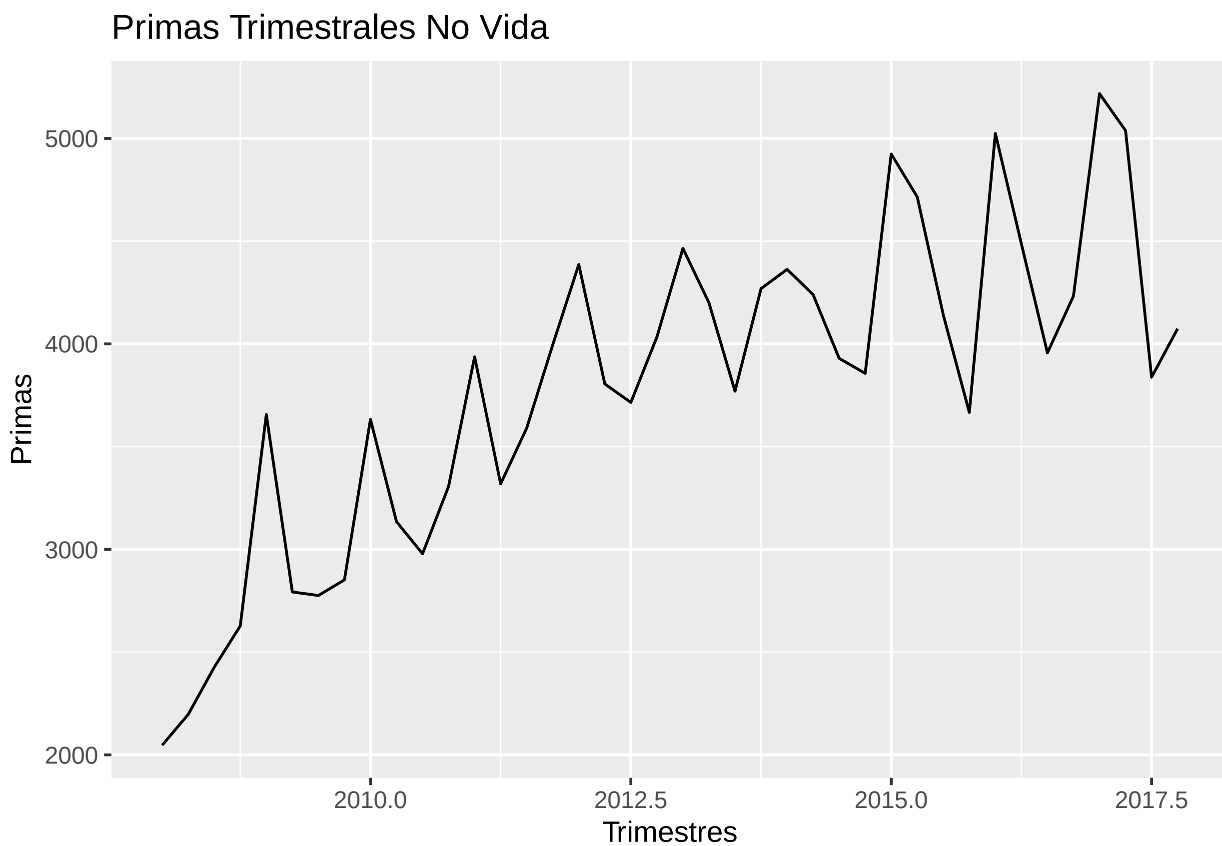
Modelo ETS para primas no vida

Preparando datos

```
## Plot Serie Primas No Vida
no_vida <- xts((datos$Primas_no_vida), order.by = datos$Fecha, frequency = 4)
## Generate quarterly data
```

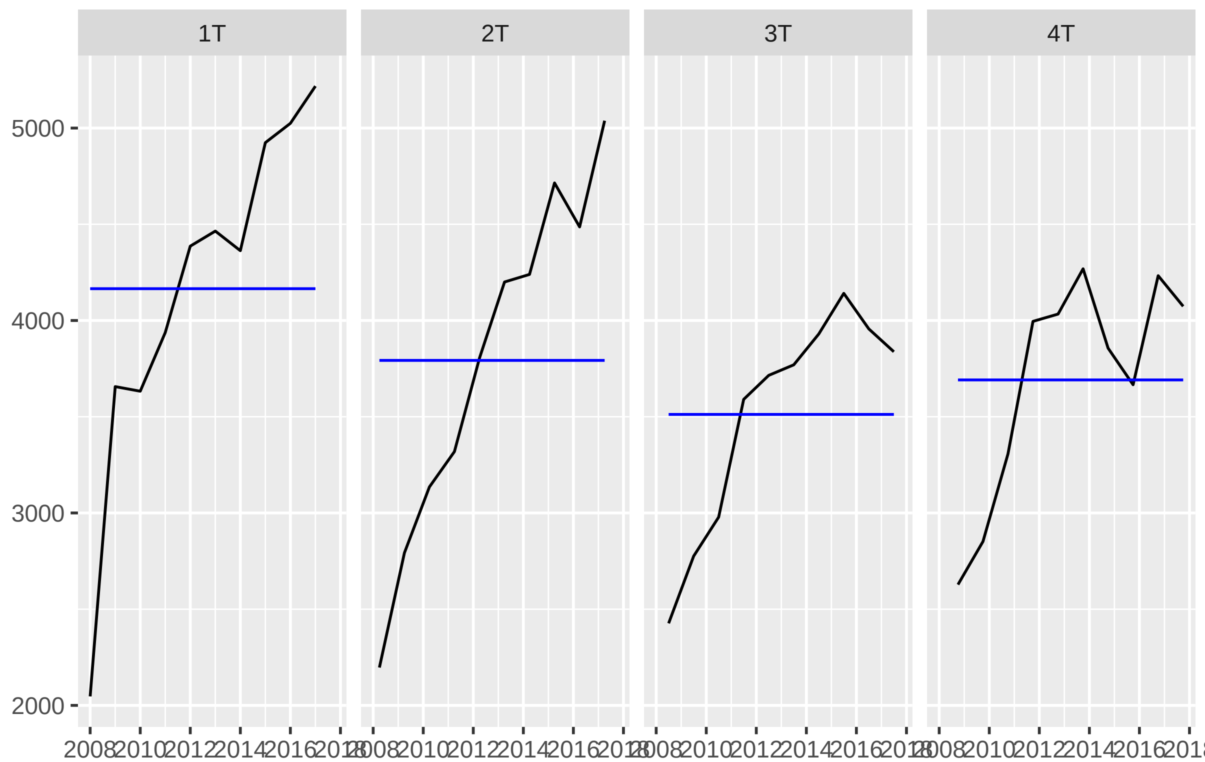
```r
no_vida <- to.quarterly(no_vida)  ## OJO cambia a que sea trimestral
## paqueteria zoo para mejor funcionamiento
no_vida <- as.zoo(no_vida$no_vida.Close)
autoplot(no_vida) + ggtitle("Primas Trimestrales No Vida") + xlab("Trimestres") + ylab("Primas")
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```




```r
## Seasonal Plot
ggfreqplot(as.ts(no_vida), freq = 4, nrow = 1, facet.labeller = c("1T","2T","3T","4T")) +
  ggtitle("Primas Trimestrales No Vida")
```

Primas Trimestrales No Vida



Trainig set

```r
#- Training set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## Select number of observation to compare forecast
cOmit = 4

## Data Size
nObs = length(no_vida)

## sub_sample
ono_vida <- window(no_vida,start = index(no_vida[1]),end = index(no_vida[nObs - cOmit]))
```

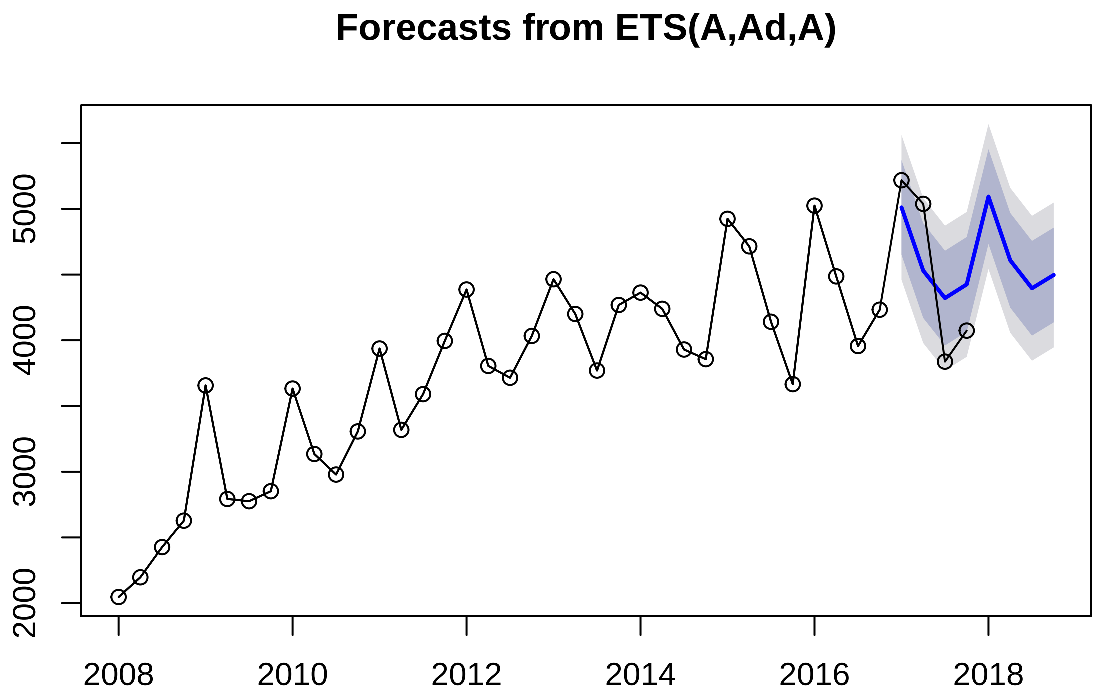
Test

```r
no_vidaetsfit <- ets(ono_vida)
#f orecast model
fno_vida.ets = forecast(no_vidaetsfit)
# Results
summary(fno_vida.ets)
```

```
##
```

```
## Forecast method: ETS(A,Ad,A)
##
## Model Information:
## ETS(A,Ad,A)
##
## Call:
##  ets(y = ono_vida)
##
##   Smoothing parameters:
##     alpha = 1e-04
##     beta  = 1e-04
##     gamma = 1e-04
##     phi   = 0.9528
##
##   Initial states:
##     l = 2198.3617
##     b = 139.3579
##     s = -177.7518 -262.1253 -31.6318 471.5088
##
##   sigma:  281.2049
##
##      AIC     AICc      BIC
## 544.6642 553.4642 560.4993
##
## Error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -2.874209 243.5306 177.8518 -0.8179414 5.275064 0.5325852
##                    ACF1
## Training set 0.08366937
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2017 Q1       5011.332 4650.954 5371.711 4460.181 5562.484
## 2017 Q2       4530.372 4169.993 4890.751 3979.220 5081.524
## 2017 Q3       4320.964 3960.585 4681.343 3769.813 4872.116
## 2017 Q4       4425.462 4065.084 4785.841 3874.311 4976.614
## 2018 Q1       5093.817 4733.438 5454.196 4542.665 5644.969
## 2018 Q2       4608.960 4248.581 4969.339 4057.808 5160.112
## 2018 Q3       4395.839 4035.460 4756.218 3844.687 4946.991
## 2018 Q4       4496.800 4136.421 4857.179 3945.648 5047.952
```

```r
## Eligio un MAM
## - M - multiplicativa la tendencia
## - A - aditiva en la pendiente
## - M - multiplicativa en estacionalidad

## ojo -> simpre que se hagan predicciones se debe dar un intervalo de confianza

# Plot
plot(fno_vida.ets)
lines(window(no_vida),type = "o")
```



Métrica de predicción

```
#Actual and Forecast
no_vidaFitmat <- matrix(c(fno_vida.ets$mean[1:cOmit],no_vida[(nObs - cOmit + 1):nObs]),ncol = 2)
no_vidaFitmat
```

```
##          [,1]   [,2]
## [1,] 5011.332 5217.9
## [2,] 4530.372 5038.2
## [3,] 4320.964 3837.7
## [4,] 4425.462 4073.8
```

```
## MSE
mean((no_vidaFitmat[,1] - no_vidaFitmat[,2])^2)
```

```
## [1] 164442.5
```

```
## MAE
mean(abs(no_vidaFitmat[,1] - no_vidaFitmat[,2]))
```

```
## [1] 387.3306
```

```r
## Bias
mean((no_vidaFitmat[,1] - no_vidaFitmat[,2]))
```

```
## [1] 30.13263
```

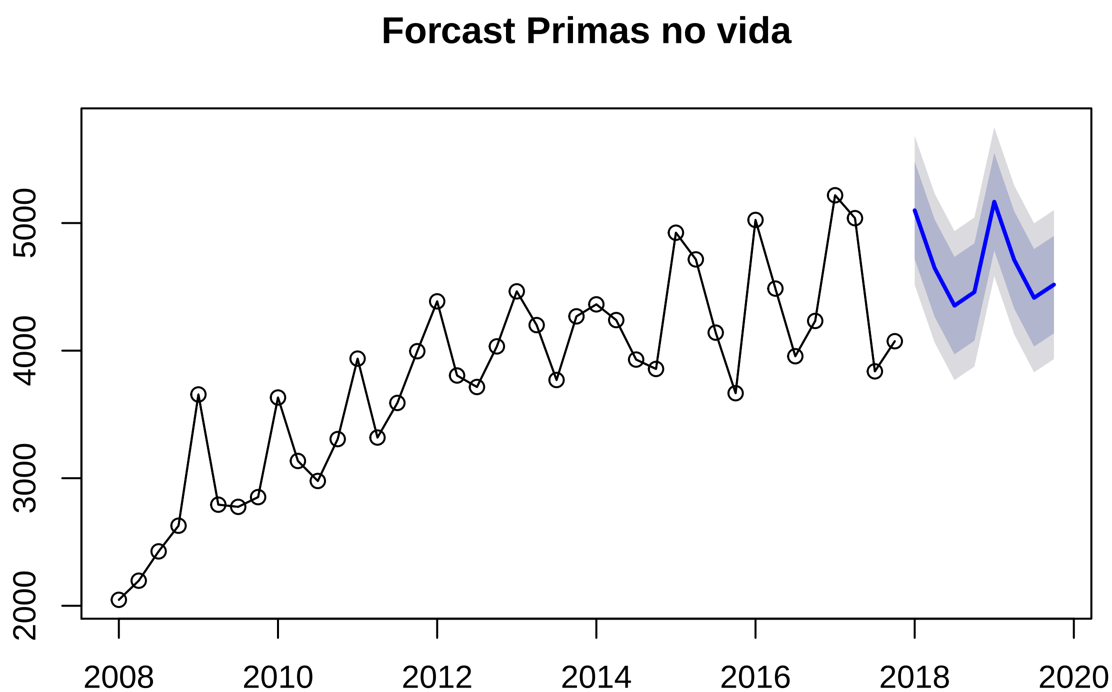
Predicción

```r
#- Complete set        -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## Select automatic ETS
etsfit <- ets(no_vida)
## forecast model
no_vida.ets <- forecast(etsfit)
## Results
summary(no_vida.ets)
```

```
##
## Forecast method: ETS(A,Ad,A)
##
## Model Information:
## ETS(A,Ad,A)
##
## Call:
##  ets(y = no_vida)
##
##   Smoothing parameters:
##     alpha = 4e-04
##     beta  = 1e-04
##     gamma = 2e-04
##     phi   = 0.9527
##
##   Initial states:
##     l = 2203.7506
##     b = 138.7075
##     s = -206.3655 -295.8864 16.1935 486.0584
##
##   sigma:  297.9073
##
##      AIC     AICc      BIC
## 613.1021 620.6883 629.9909
##
## Error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -2.103476 262.2599 198.7776 -0.8237639 5.729095 0.6111534
##                   ACF1
## Training set 0.08167387
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2018 Q1       5099.550 4717.767 5481.334 4515.663 5683.438
## 2018 Q2       4647.861 4266.078 5029.645 4063.974 5231.749
```

```
## 2018 Q3       4352.951 3971.167 4734.735 3769.063 4936.839
## 2018 Q4       4458.972 4077.188 4840.756 3875.084 5042.860
## 2019 Q1       5166.934 4785.150 5548.718 4583.046 5750.822
## 2019 Q2       4712.056 4330.272 5093.840 4128.168 5295.945
## 2019 Q3       4414.108 4032.324 4795.892 3830.219 4997.996
## 2019 Q4       4517.235 4135.450 4899.019 3933.346 5101.123
```

```
## Eligio un MAM
## - M - multiplicativa la tendencia
## - A - aditiva en la pendiente
## - M - multiplicativa en estacionalidad

## ojo -> simpre que se hagan predicciones se debe dar un intervalo de confianza

# Plot
plot(no_vida.ets, main = "Forcast Primas no vida")
lines(window(no_vida),type = "o")
```

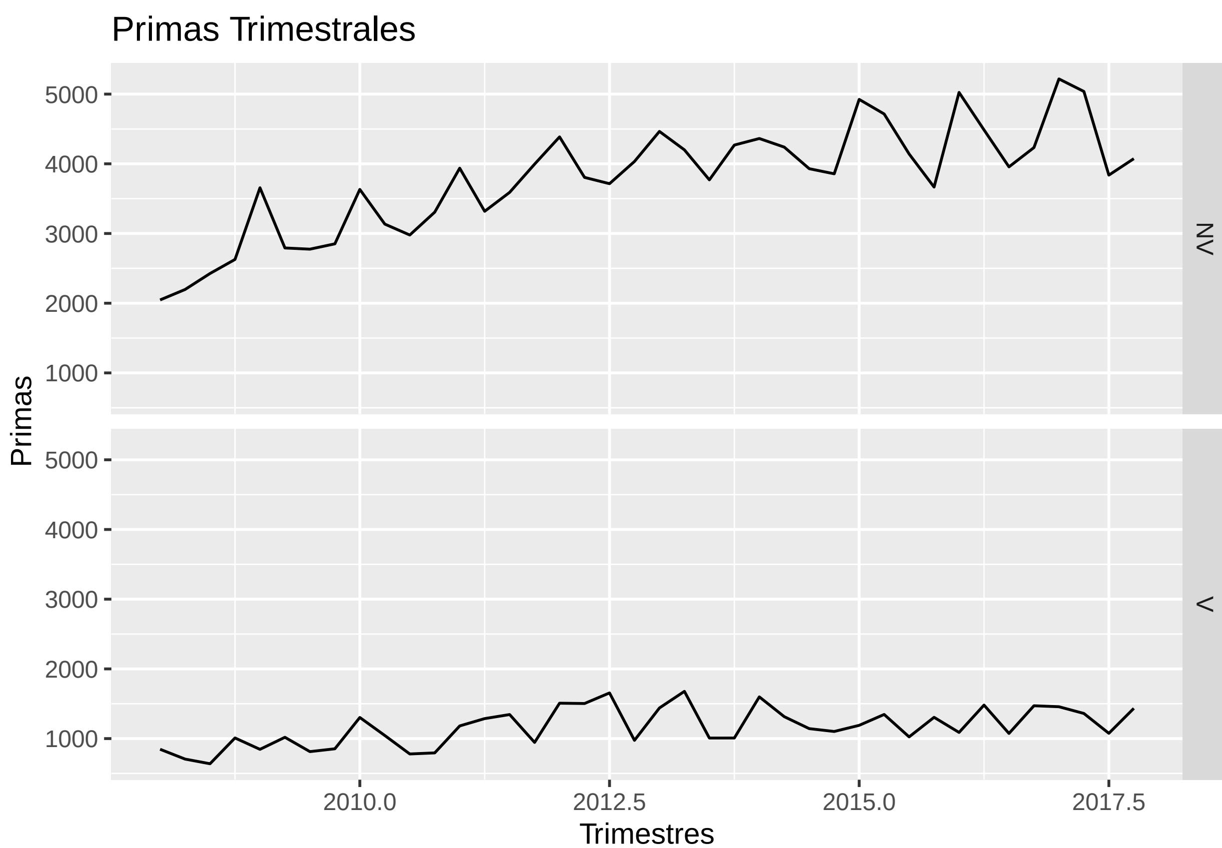


Modelo ETS para primas HTS ((???))

```
## modelo ETS desde el enfoque de series de tiempo jerarquicas (hts)
##
```

```r
## Plot Series
sepxts <- xts(datos[,c(3,4)], order.by = datos$Fecha, frequency = 4)
## Generate quarterly data
sepxtsVida   <- to.quarterly(sepxts$Primas_vida)      ## OJO cambia a que sea trimestral
sepxtsNoVida <- to.quarterly(sepxts$Primas_no_vida)  ## OJO cambia a que sea trimestral
## paqueteria zoo para mejor funcionamiento
sepxts <- cbind(sepxtsNoVida$`sepxts$Primas_no_vida.Close`, sepxtsVida$`sepxts$Primas_vida.Close`)
sepxts <- as.zoo(sepxts)
names(sepxts) <- c("NV", "V")
autoplot(sepxts) + ggtitle("Primas Trimestrales") + xlab("Trimestres") + ylab("Primas")
```

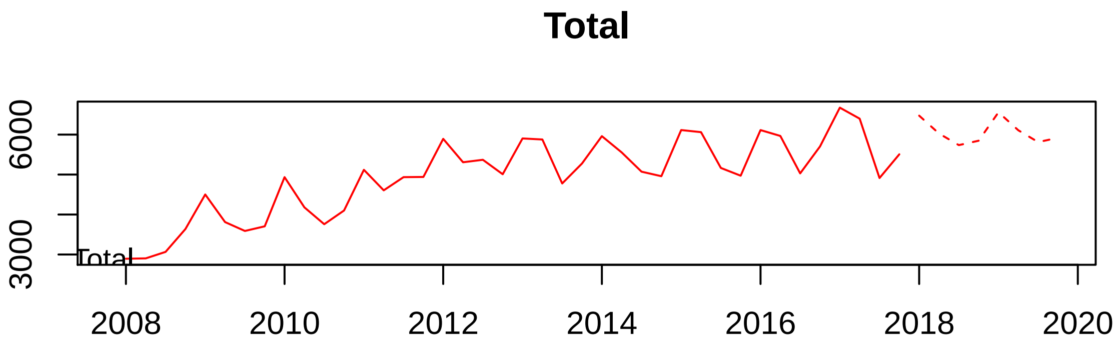
```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```

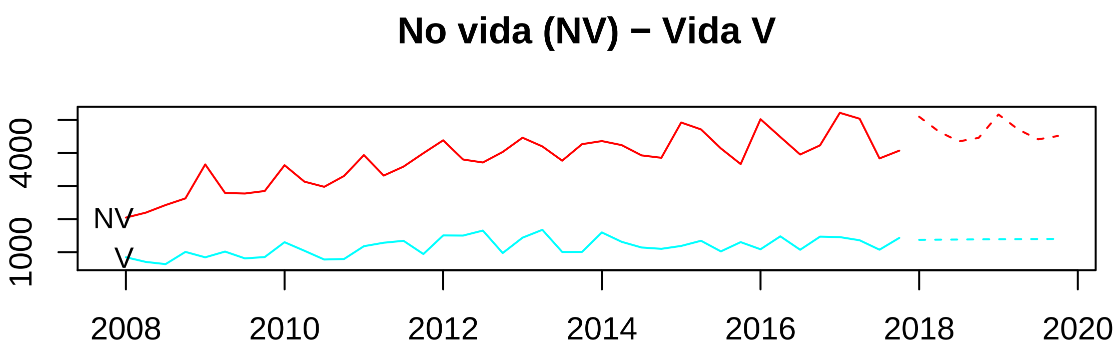



```r
## Select automatic HTS
sepmod <- hts(sepxts, nodes = list(2))
```

```
## Since argument characters are not specified, the default labelling system is used.
```

```r
## Forcast
sep.fit <- forecast(sepmod, method = 'bu', fmethod = 'ets') # buttom up
names(sep.fit$labels) = c("Total", "No vida (NV) - Vida V")
plot(sep.fit)
```





Suma de vida y no vida ETS

```r
## Despues de trabajar por separado vida y no vida sumamos para ver la prediccion total

#Actual and Forecast
sumaFitmat <- vidaFitmat + no_vidaFitmat
sumaFitmat
```

```
##           [,1]   [,2]
## [1,] 6295.395 6674.6
## [2,] 5814.435 6398.6
## [3,] 5605.027 4913.4
## [4,] 5709.525 5507.2
```

```r
## MSE
mean((sumaFitmat[,1] - sumaFitmat[,2])^2)
```

```
## [1] 251082.2
```

```r
## MAE
mean(abs(sumaFitmat[,1] - sumaFitmat[,2]))
```

```
## [1] 464.3306
```

```r
## Bias
mean(sumaFitmat[,1] - sumaFitmat[,2])
```

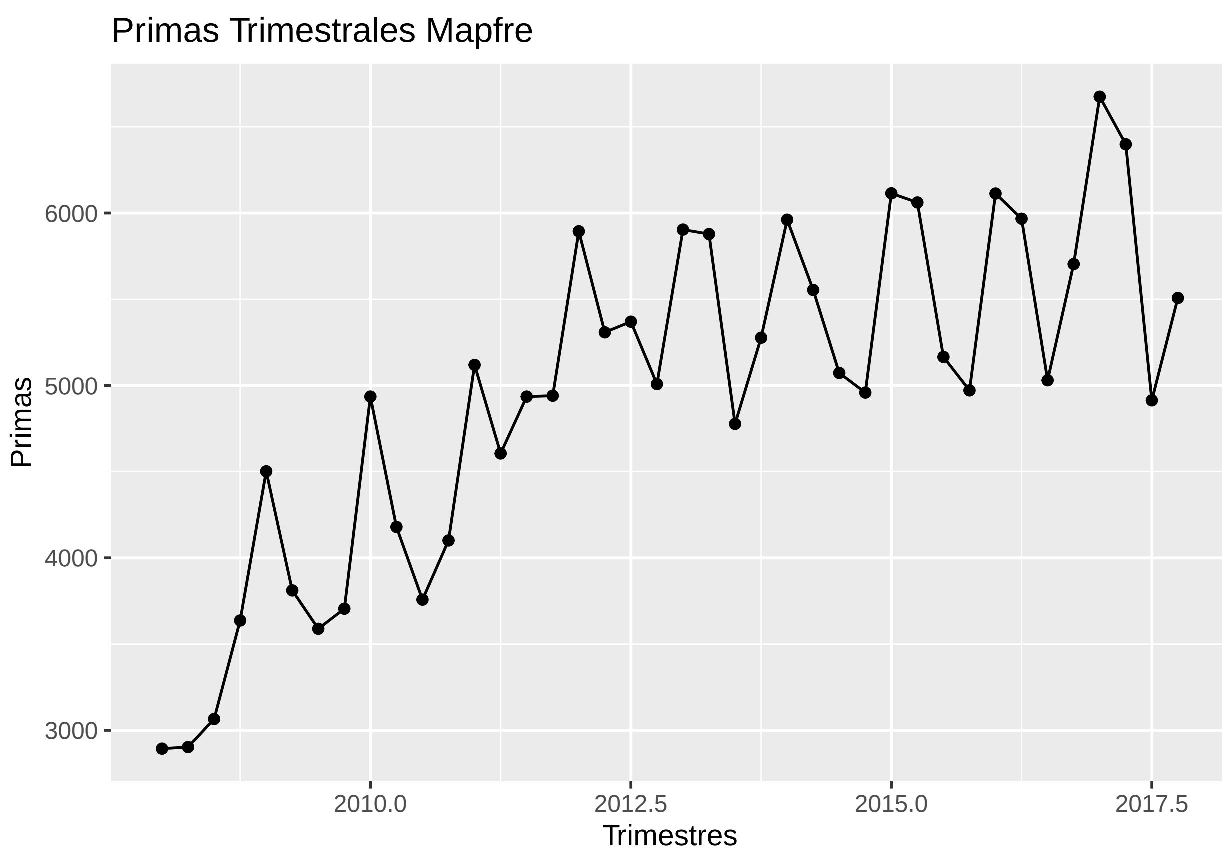
```
## [1] -17.35465
```

Modelo ARIMA para primas totales

Preparando datos y análisis por diferencias

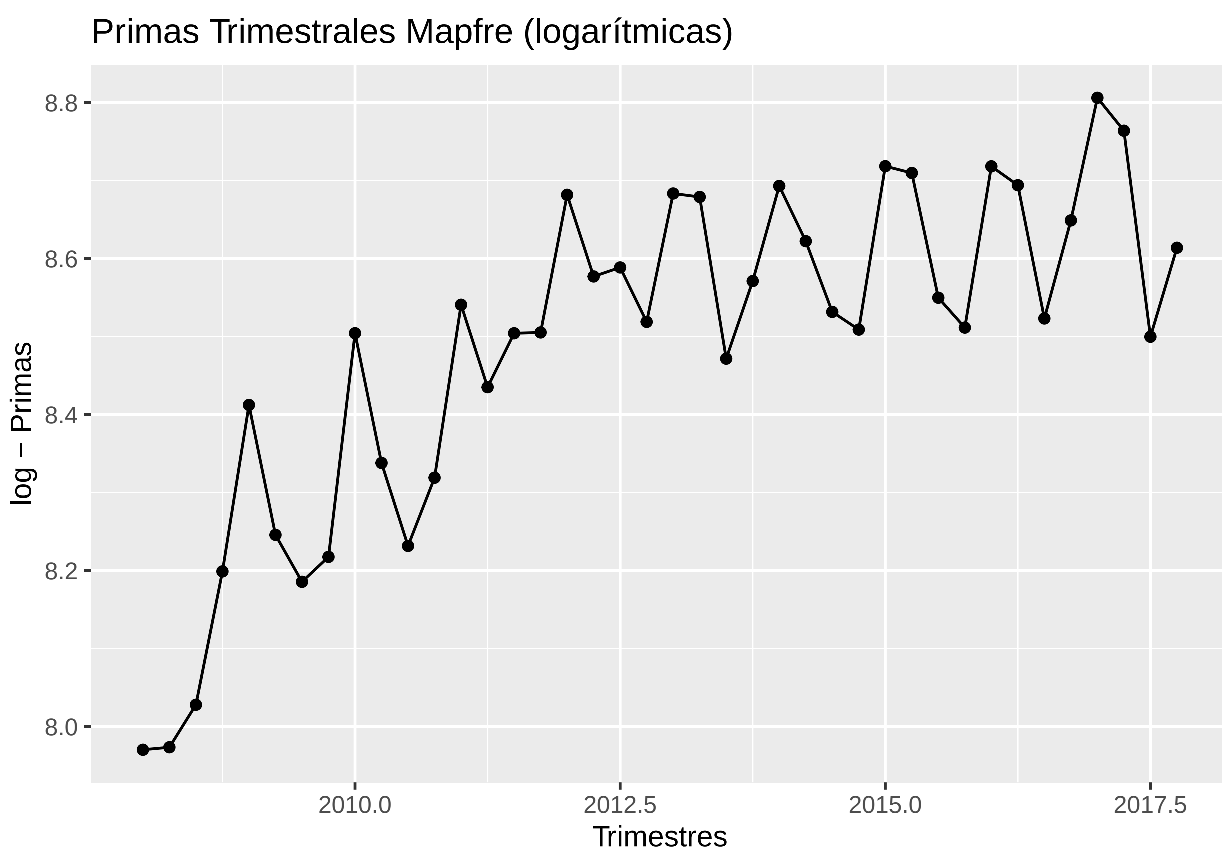
```r
## Nuestra ts de primas totales se llama "total"

df_total <- data.frame(value = as.vector(total),
                       time = time(total))
ggplot(df_total) + geom_point(aes(x = time, y = value)) +
  geom_line(aes(x = time, y = value)) +
  ylab("Primas") +
  ggtitle("Primas Trimestrales Mapfre") +
  xlab("Trimestres")
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```

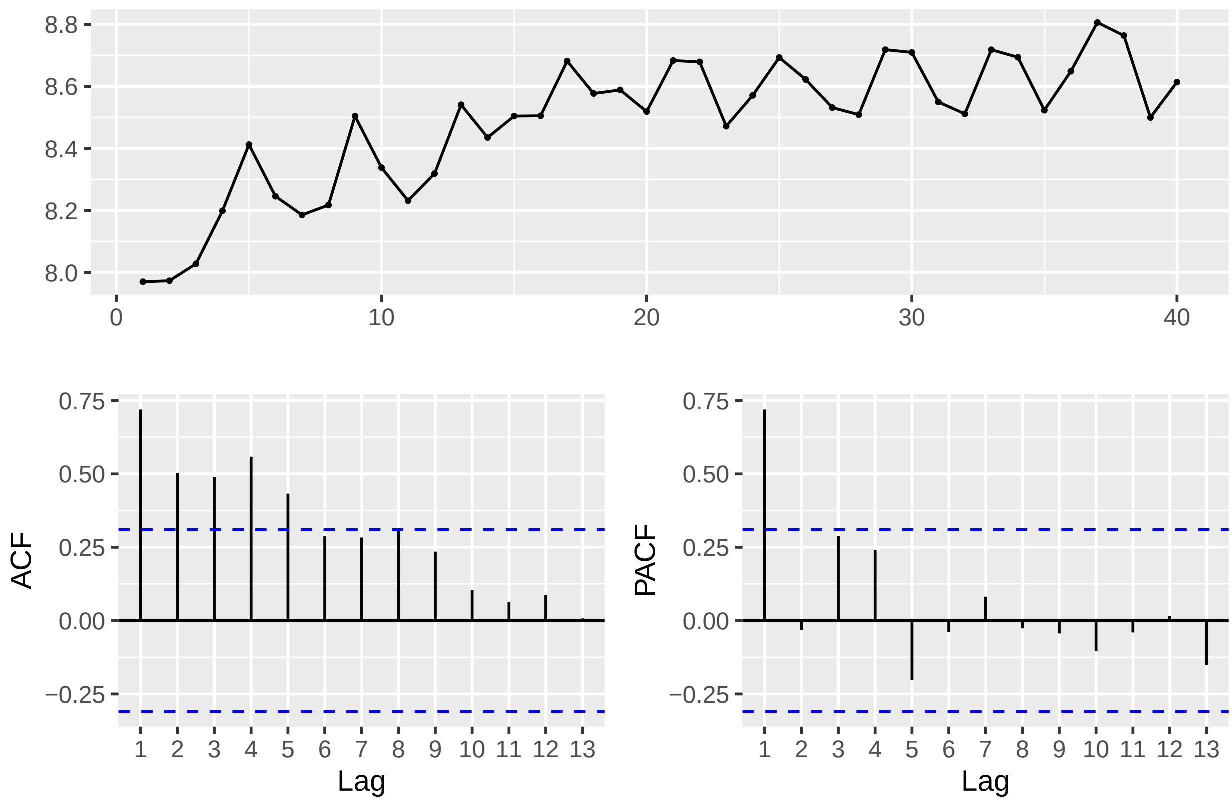


```r
## trabajamos con transformacion logaritmica
logTotal <- log(total)
df_logTotal <- data.frame(value = as.vector(logTotal),
                      time = time(logTotal))
ggplot(df_logTotal) + geom_point(aes(x = time, y = value)) +
  geom_line(aes(x = time, y = value)) +
  ylab("log - Primas") +
  ggtitle("Primas Trimestrales Mapfre (logarítmicas)") +
  xlab("Trimestres")
```

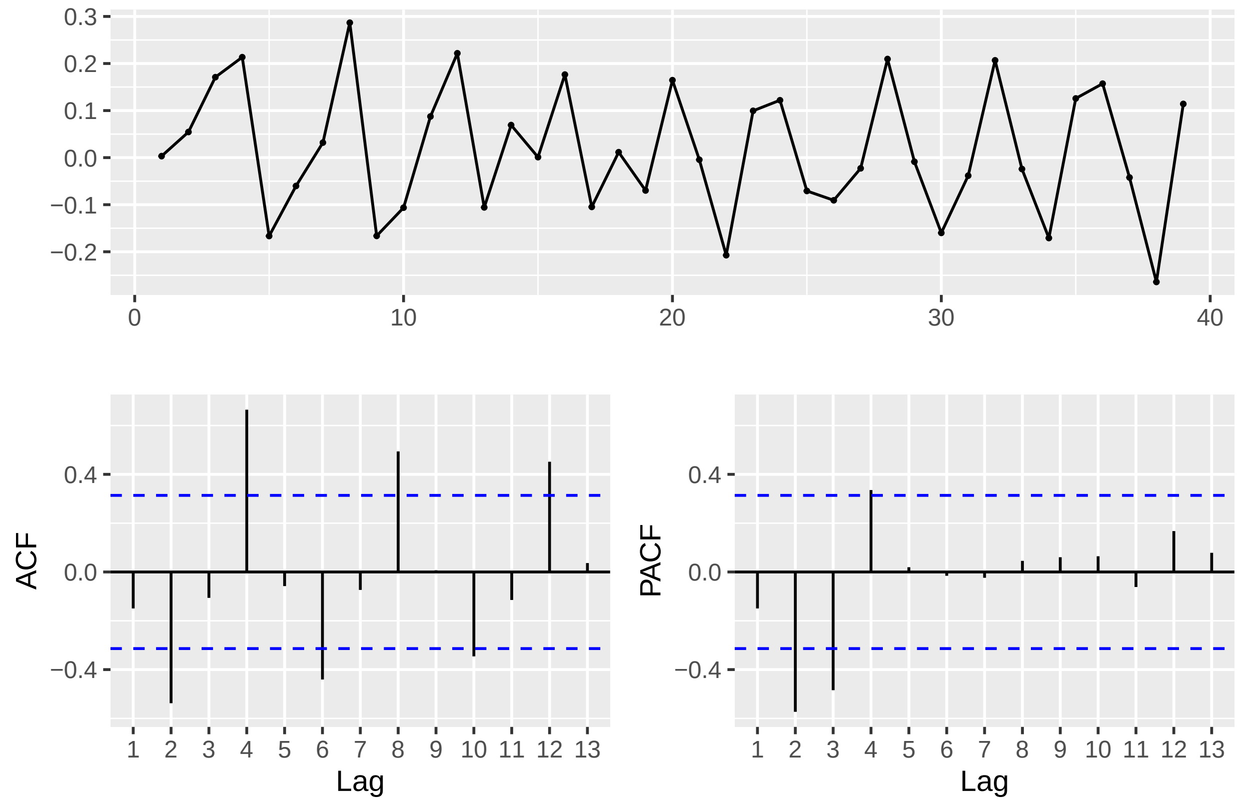
```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```



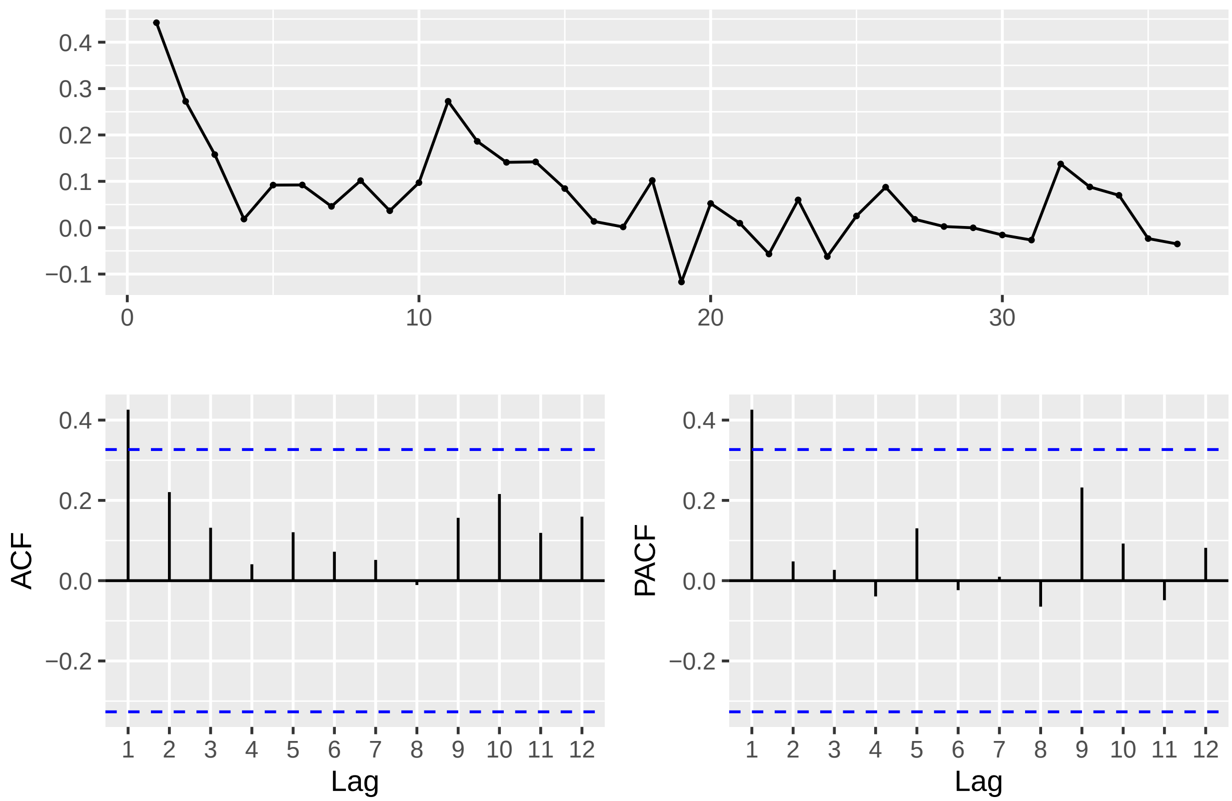

```r
## Difference
ggtsdisplay(logTotal)
```



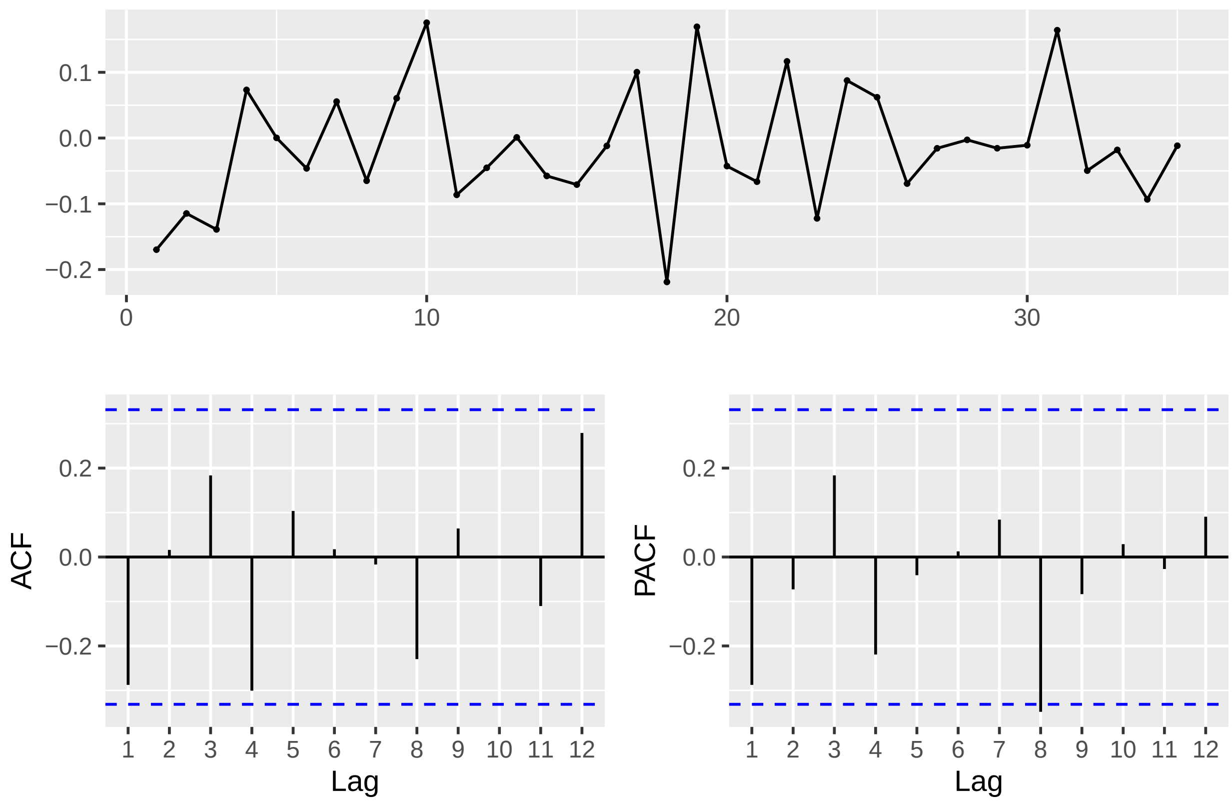

```r
ggtsdisplay(diff(logTotal))
```




```r
ggtsdisplay(diff(logTotal,4))
```




```
ggtsdisplay(diff(diff(logTotal,4),1))
```



Trainig set

```r
#- Training set      -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

#Select number of observation to compare forecast
cOmit = 4

#Data Size
nObs = length(total)

#sub_sample TRAINING
oatotal <- window(total, start = index(total[1]), end = index(total[nObs - cOmit]))
```

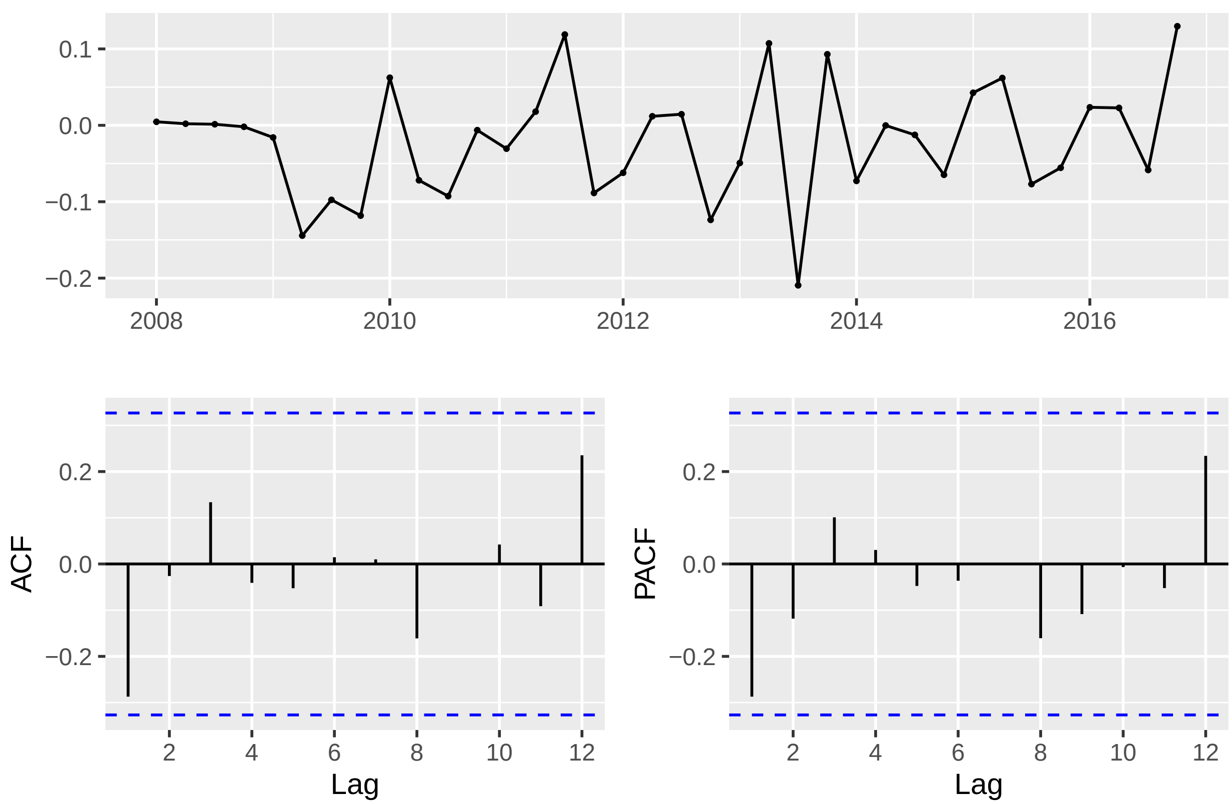
Test

```r
## ARIMA MODEL Automatic selection####
total.train.arima = auto.arima(oatotal,lambda = 0) ## lamnda cero is log transformation
summary(total.train.arima)
```

```
## Series: oatotal
## ARIMA(0,1,0)(0,1,1)[4]
## Box Cox transformation: lambda= 0
```

```
##
## Coefficients:
##          sma1
##       -0.6185
## s.e.   0.1830
##
## sigma^2 estimated as 0.007238:  log likelihood=31.97
## AIC=-59.94   AICc=-59.51   BIC=-57.07
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -93.54831 388.8768 301.9985 -2.36378 6.164447 0.7294924
##                    ACF1
## Training set -0.3575378
```

```
#residual analysis
ggtsdisplay(total.train.arima$residuals)
```




```
#box-Ljung Test
Box.test(total.train.arima$residuals,lag = 4, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  total.train.arima$residuals
## X-squared = 4.0615, df = 1, p-value = 0.04387
```

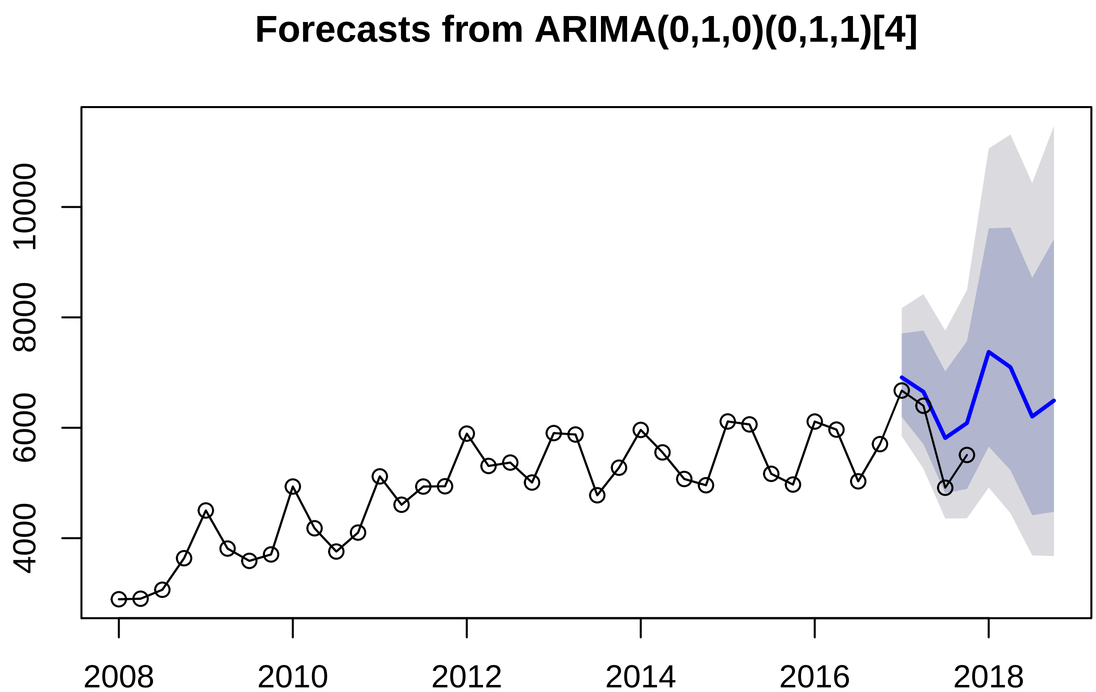
```r
Box.test(total.train.arima$residuals,lag = 8, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  total.train.arima$residuals
## X-squared = 5.4651, df = 5, p-value = 0.3618
```

```r
Box.test(total.train.arima$residuals,lag = 12, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  total.train.arima$residuals
## X-squared = 9.1663, df = 9, p-value = 0.4221
```

```r
plot(forecast(total.train.arima))
lines(window(total),type = "o")
```



Métrica de predicción

```r
ftotal_arima <- forecast(total.train.arima)

totalArimaMatrix <- matrix(c(ftotal_arima$mean[1:4], as.double(tail(total,4))), ncol = 2)
totalArimaMatrix
```

```
##          [,1]   [,2]
## [1,] 6912.247 6674.6
## [2,] 6652.463 6398.6
## [3,] 5814.438 4913.4
## [4,] 6085.474 5507.2
```

```r
## MSE
mean((totalArimaMatrix[,1] - totalArimaMatrix[,2])^2)
```

```
## [1] 316798
```

```r
## MAE
mean(abs(totalArimaMatrix[,1] - totalArimaMatrix[,2]))
```

```
## [1] 492.7053
```

```r
## Bias
mean(totalArimaMatrix[,1] - totalArimaMatrix[,2])
```

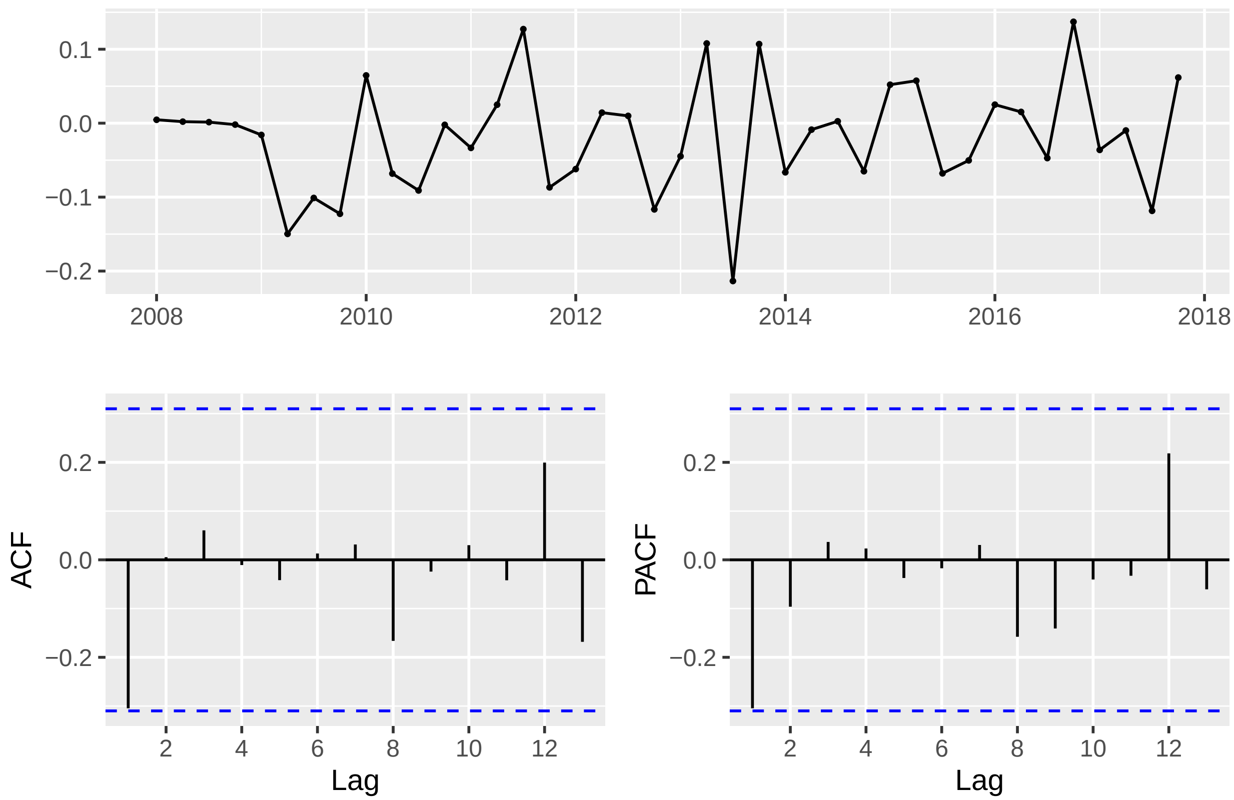
```
## [1] 492.7053
```

Predicción

```r
#- Complete set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## ARIMA MODEL Automatic selection
total.fit.arima <- auto.arima(total, lambda = 0) ## lamnda cero para transformacion log
summary(total.fit.arima)
```

```
## Series: total
## ARIMA(0,1,0)(0,1,1)[4]
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          sma1
##       -0.5349
## s.e.   0.1644
##
## sigma^2 estimated as 0.007115:  log likelihood=36.74
## AIC=-69.47   AICc=-69.1   BIC=-66.36
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -89.20838 389.9248 302.2919 -2.22244 6.118013 0.7477281
##                    ACF1
## Training set -0.3814568
```

```r
#residual analysis
ggtsdisplay(total.fit.arima$residuals)
```




```r
#box-Ljung Test
Box.test(total.fit.arima$residuals, lag = 4, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  total.fit.arima$residuals
## X-squared = 4.168, df = 1, p-value = 0.04119
```

```r
Box.test(total.fit.arima$residuals, lag = 8, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  total.fit.arima$residuals
## X-squared = 5.7618, df = 5, p-value = 0.3301
```

```r
Box.test(total.fit.arima$residuals, lag = 12, fitdf = 3, type = "Lj")
```

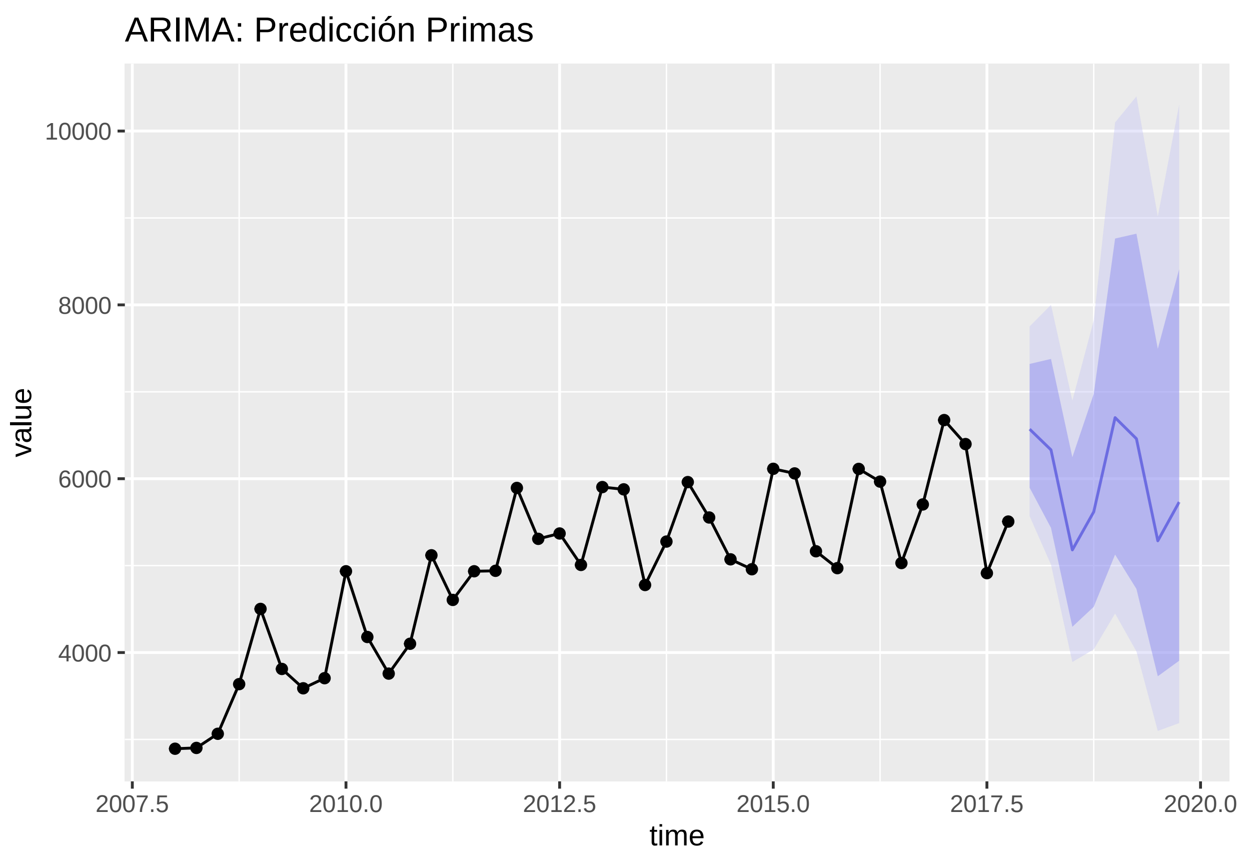
```
##
```

```
##  Box-Ljung test
##
## data:  total.fit.arima$residuals
## X-squared = 8.3337, df = 9, p-value = 0.5009
```

```
tota.arima <- forecast(total.fit.arima)

ggplot(df_total) + geom_point(aes(x = time,y = value)) +
  geom_line(aes(x = time, y = value)) +
  geom_forecast(tota.arima, alpha = 0.4) +
  ggtitle("ARIMA: Predicción Primas")
```

```
## Warning in geom_forecast(tota.arima, alpha = 0.4): Use autolayer instead of
## geom_forecast to add a forecast layer to your ggplot object.
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```

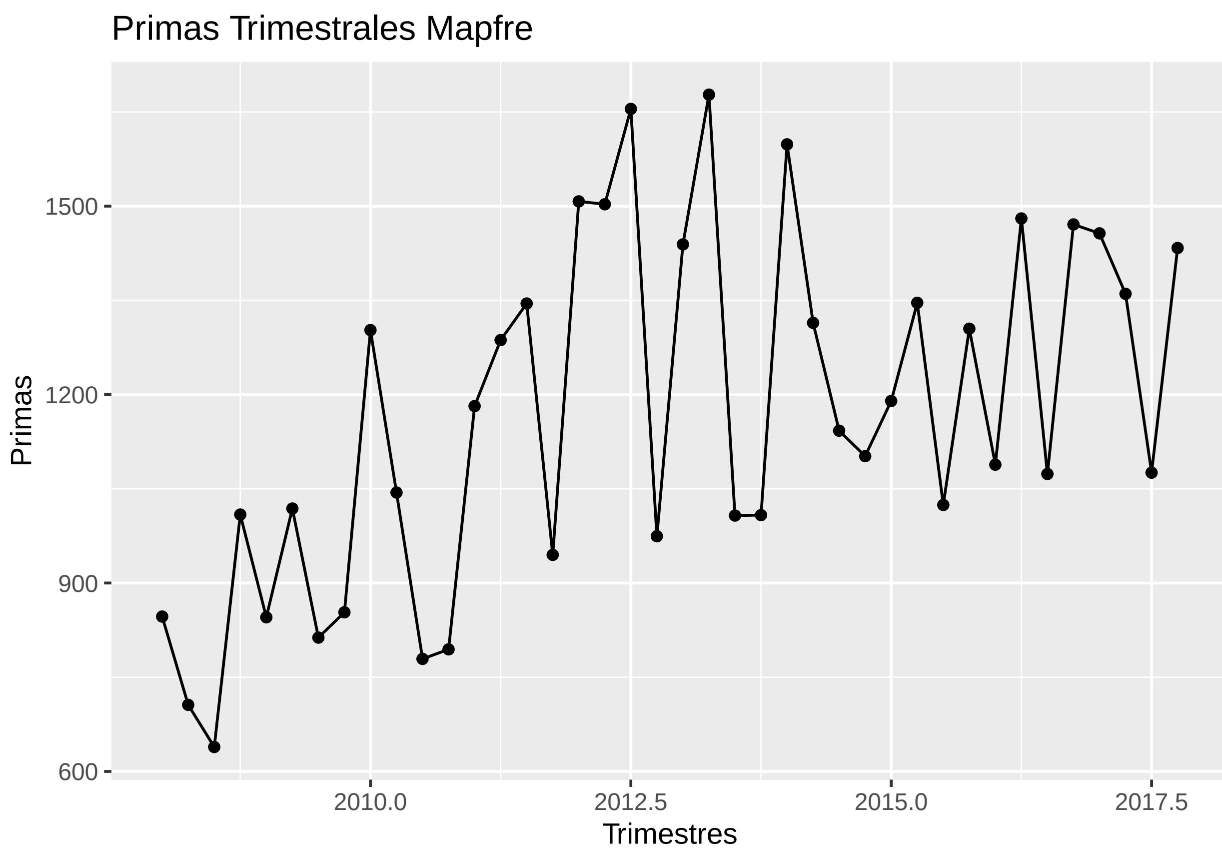


Modelo ARIMA para primas vida

Preparando datos y análisis por diferencias

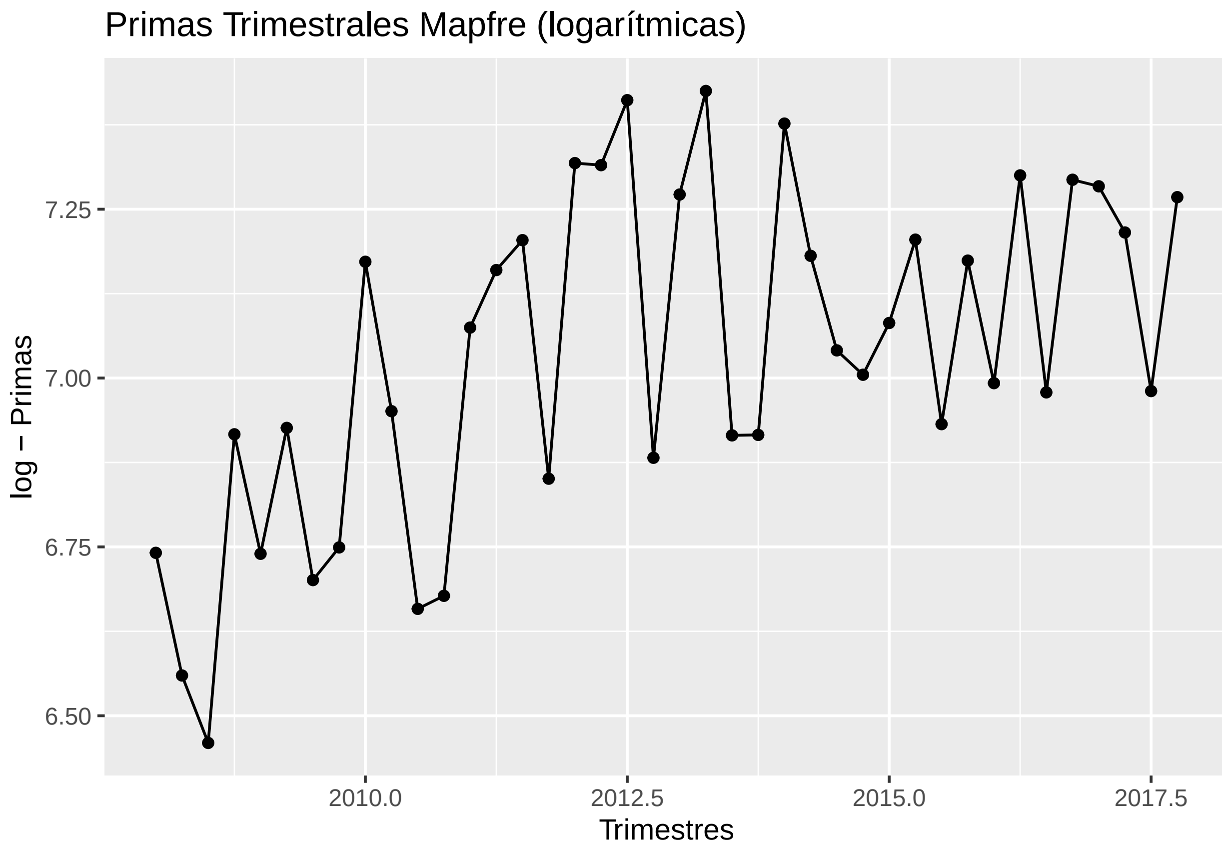
```r
## Trabajando un modelo ARIMA para Primas por separado ##################################################
##

## Nuestra ts de primas vida se llama "vida"

df_vida <- data.frame(value = as.vector(vida),
                      time = time(vida))
ggplot(df_vida) + geom_point(aes(x = time, y = value)) +
  geom_line(aes(x = time, y = value)) +
  ylab("Primas") +
  ggtitle("Primas Trimestrales Mapfre") +
  xlab("Trimestres")
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```

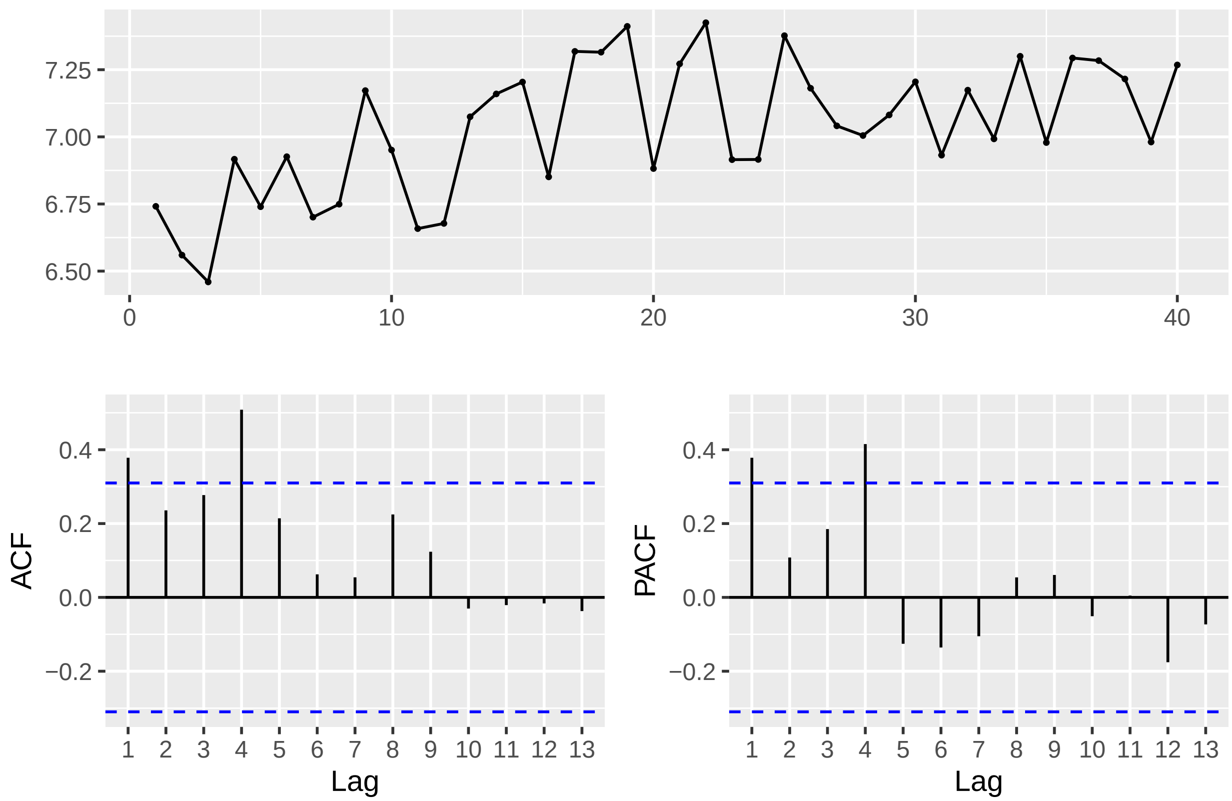



```r
## trabajamos con transformacion logaritmica
logvida <- log(vida)
df_logvida <- data.frame(value = as.vector(logvida),
                         time = time(logvida))
ggplot(df_logvida) + geom_point(aes(x = time, y = value)) +
  geom_line(aes(x = time, y = value)) +
  ylab("log - Primas") +
  ggtitle("Primas Trimestrales Mapfre (logarítmicas)") +
  xlab("Trimestres")
```

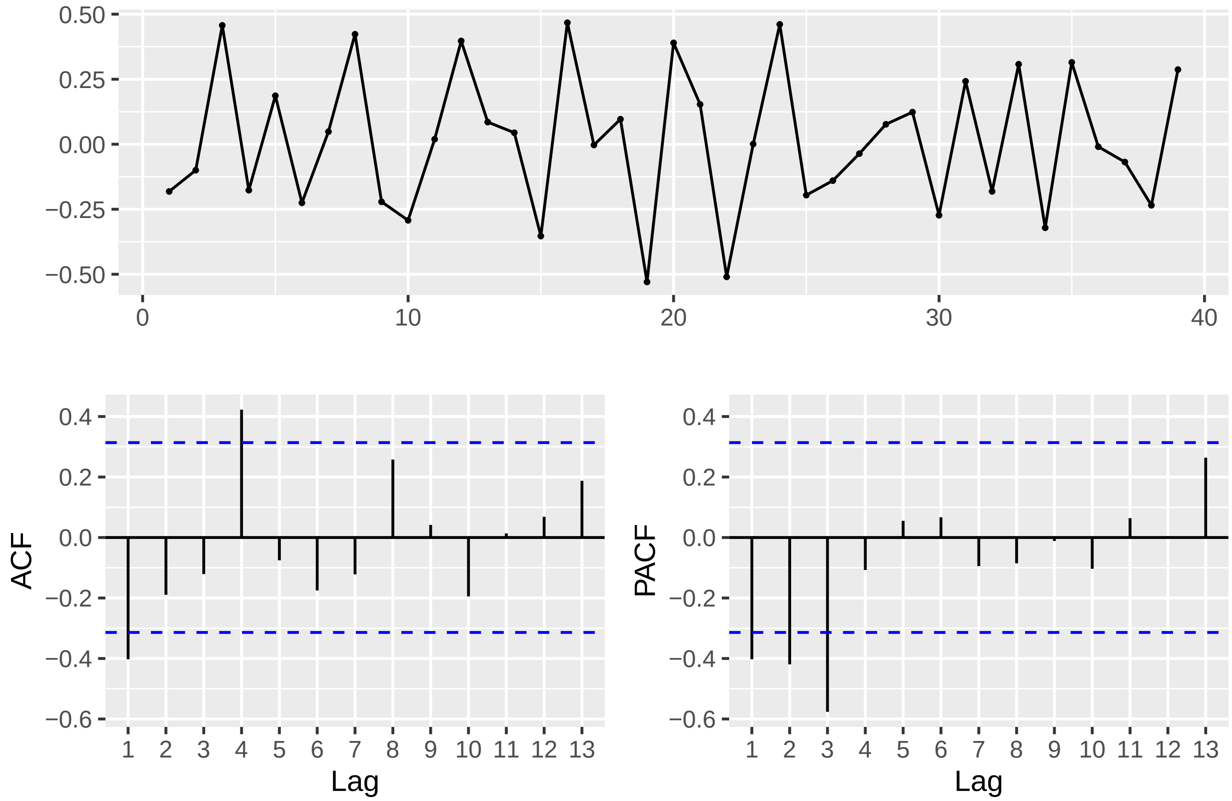
```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```



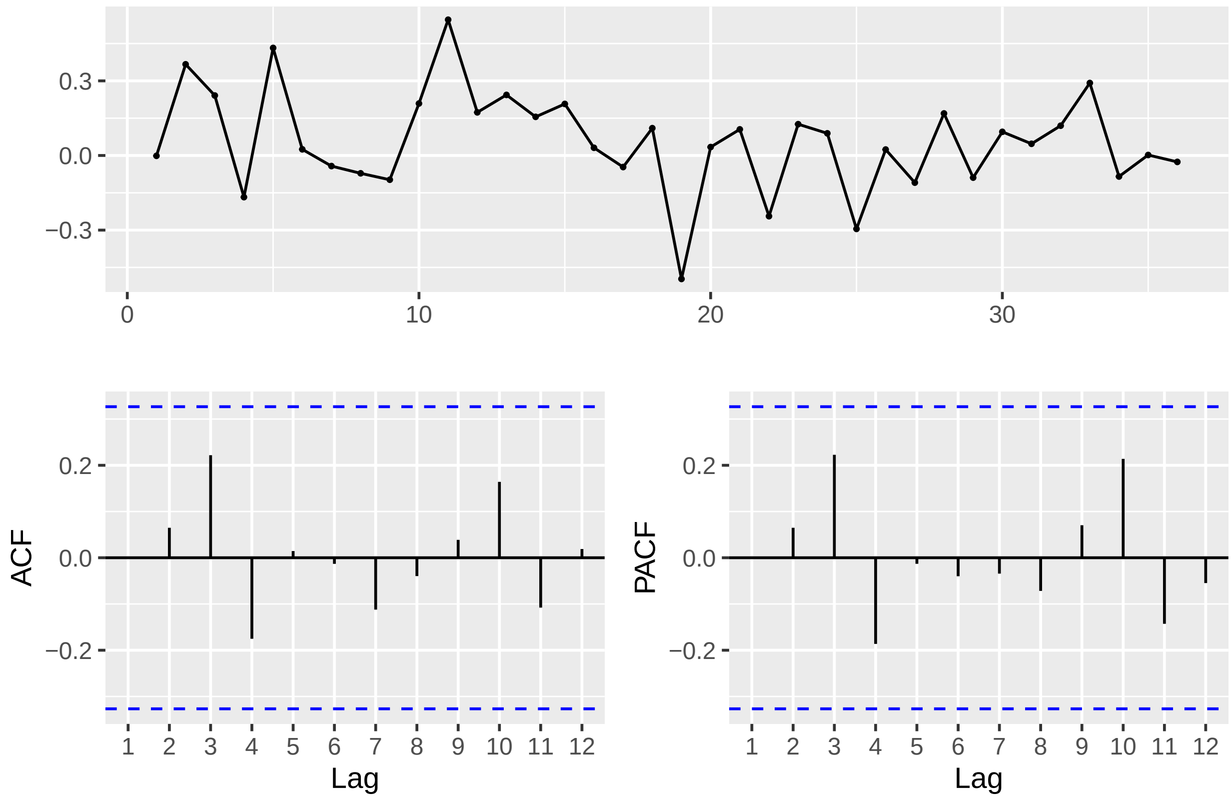

```r
## Difference
ggtsdisplay(logvida)
```



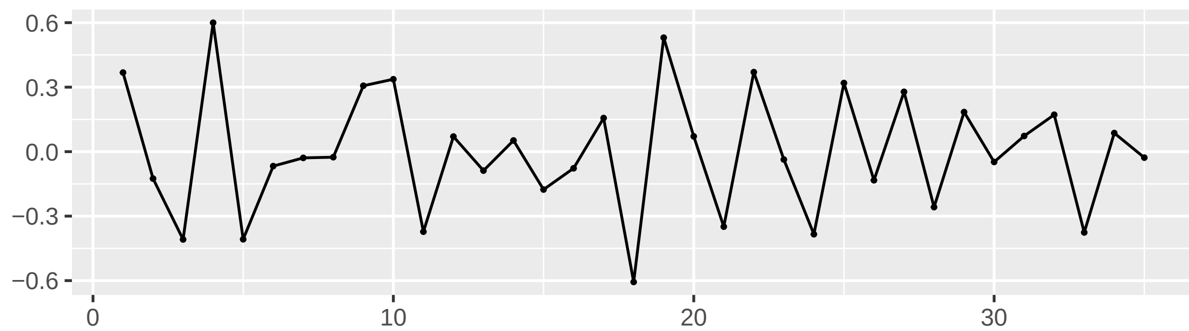

```r
ggtsdisplay(diff(logvida))
```

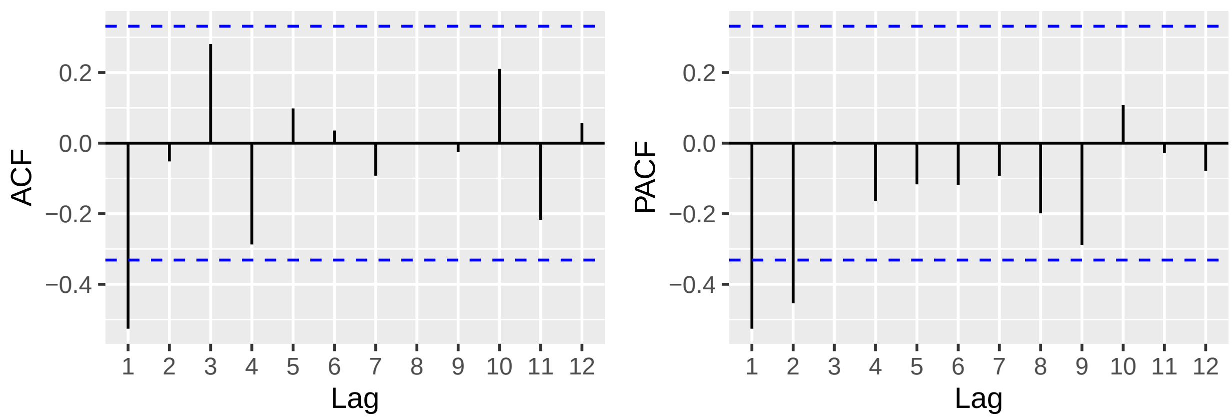



```r
ggtsdisplay(diff(logvida,4))
```




```r
ggtsdisplay(diff(diff(logvida,4),1))
```





Trainig set

```
#- Training set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

#Select number of observation to compare forecast
cOmit = 4

#Data Size
nObs = length(vida)

#sub_sample TRAINING
oavida <- window(vida, start = index(vida[1]), end = index(vida[nObs - cOmit]))
```

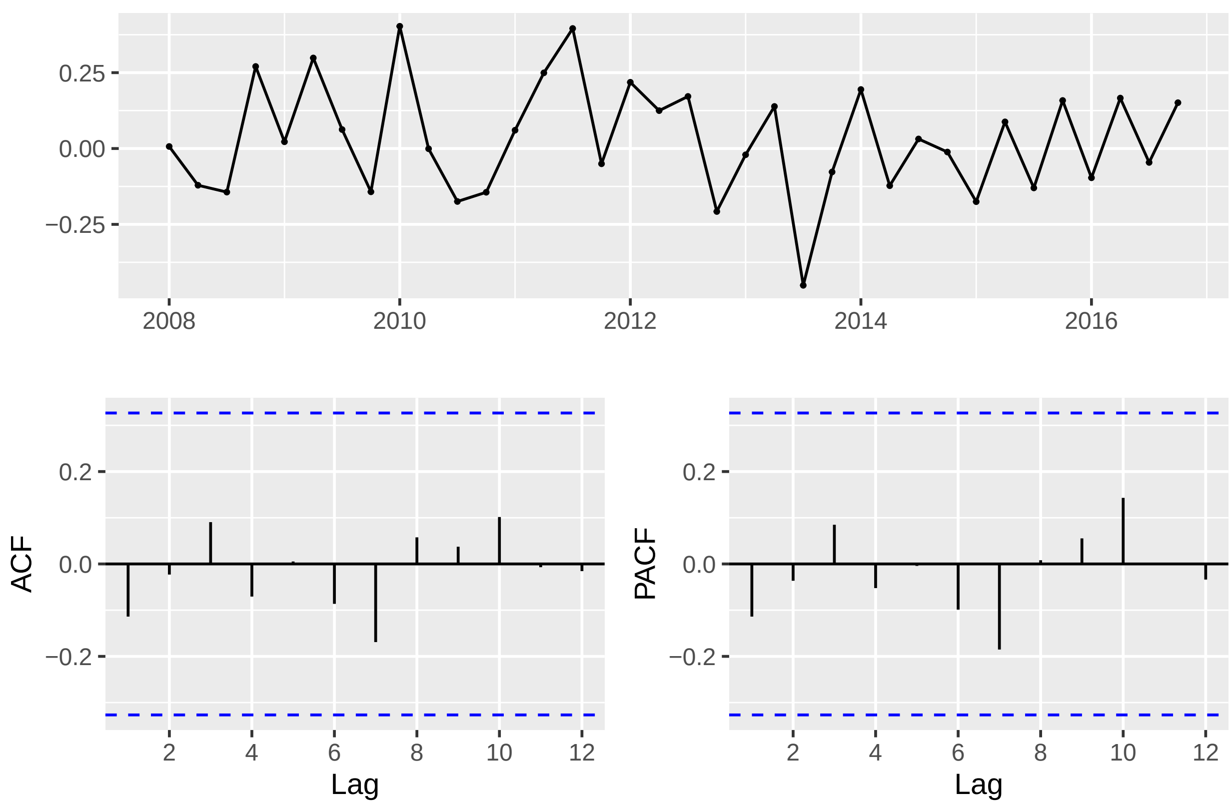
Test

```
## ARIMA MODEL Automatic selection####
vida.train.arima = auto.arima(oavida,lambda = 0) ## lamnda cero is log transformation
summary(vida.train.arima)
```

```
## Series: oavida
## ARIMA(0,1,1)(1,0,0)[4]
## Box Cox transformation: lambda= 0
```

```
##
## Coefficients:
##           ma1    sar1
##       -0.7913  0.5266
## s.e.   0.1044  0.1529
##
## sigma^2 estimated as 0.03703:  log likelihood=8.14
## AIC=-10.28   AICc=-9.5   BIC=-5.61
##
## Training set error measures:
##                    ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 42.77565 212.8668 168.6744 1.383046 14.47594 0.8893692
##                    ACF1
## Training set -0.1575173
```

```
#residual analysis
ggtsdisplay(vida.train.arima$residuals)
```




```
#box-Ljung Test
Box.test(vida.train.arima$residuals,lag = 4, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  vida.train.arima$residuals
## X-squared = 1.0806, df = 1, p-value = 0.2986
```

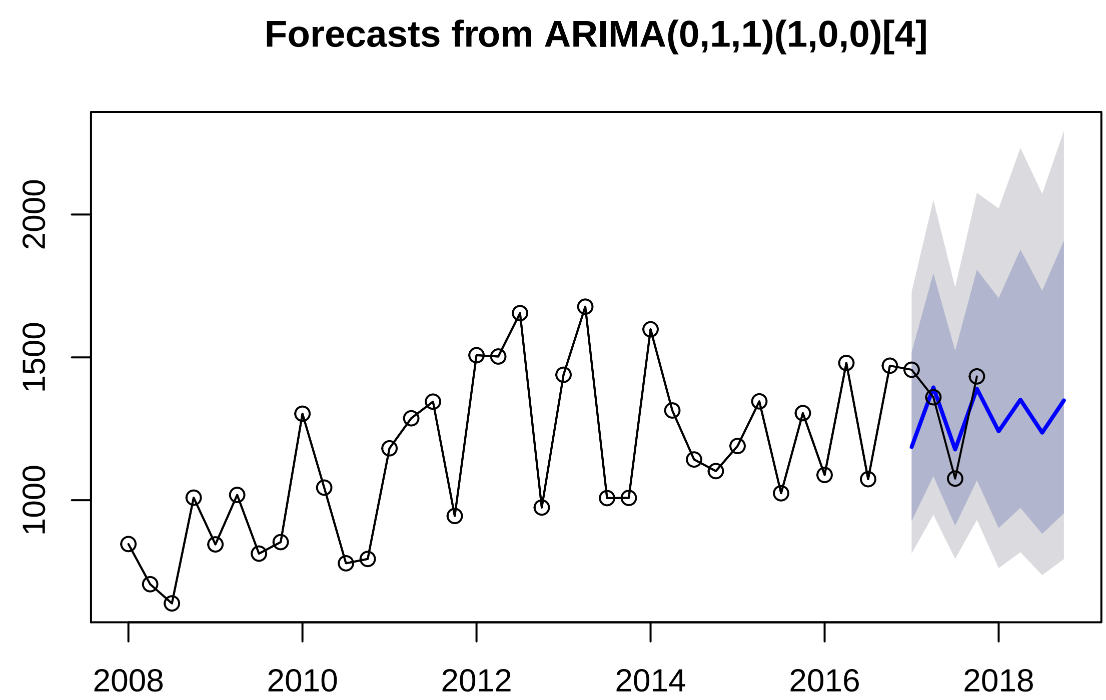
```r
Box.test(vida.train.arima$residuals,lag = 8, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  vida.train.arima$residuals
## X-squared = 2.9317, df = 5, p-value = 0.7105
```

```r
Box.test(vida.train.arima$residuals,lag = 12, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  vida.train.arima$residuals
## X-squared = 3.5615, df = 9, p-value = 0.9378
```

```r
plot(forecast(vida.train.arima))
lines(window(vida),type = "o")
```



Métrica de predicción

```r
fvida_arima <- forecast(vida.train.arima)

vidaArimaMatrix <- matrix(c(fvida_arima$mean[1:4], as.double(tail(vida,4))), ncol = 2)
vidaArimaMatrix
```

```
##          [,1]   [,2]
## [1,] 1186.262 1456.7
## [2,] 1394.871 1360.4
## [3,] 1177.740 1075.7
## [4,] 1390.150 1433.4
```

```r
## MSE
mean((vidaArimaMatrix[,1] - vidaArimaMatrix[,2])^2)
```

```
## [1] 21651.86
```

```r
## MAE
mean(abs(vidaArimaMatrix[,1] - vidaArimaMatrix[,2]))
```

```
## [1] 112.5497
```

```r
## Bias
mean(vidaArimaMatrix[,1] - vidaArimaMatrix[,2])
```

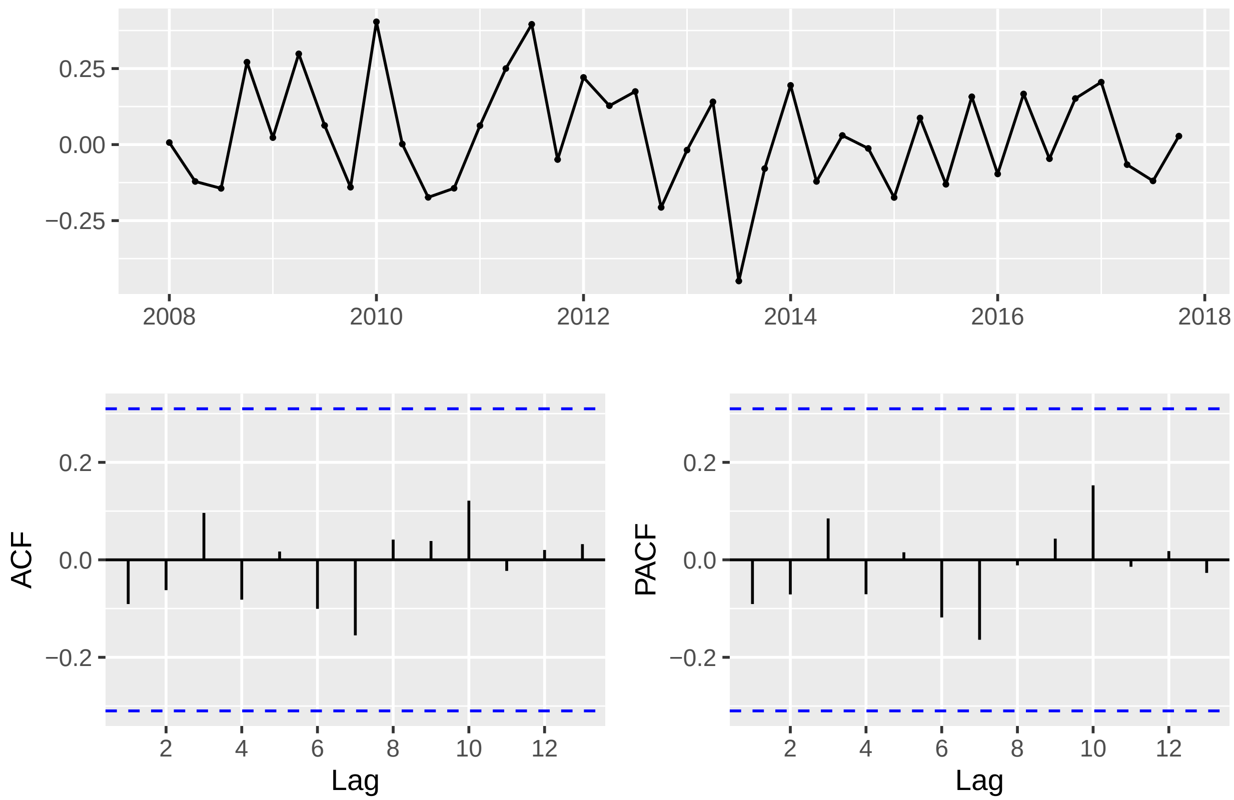
```
## [1] -44.29429
```

Predicción

```r
#- Complete set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

## ARIMA MODEL Automatic selection
vida.fit.arima <- auto.arima(vida, lambda = 0) ## lamnda cero para transformacion log
summary(vida.fit.arima)
```

```
## Series: vida
## ARIMA(0,1,1)(1,0,0)[4]
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ma1    sar1
##       -0.7938  0.5221
## s.e.   0.0988  0.1441
##
## sigma^2 estimated as 0.03471:  log likelihood=10.32
## AIC=-14.64   AICc=-13.96   BIC=-9.65
##
## Training set error measures:
##                    ME     RMSE      MAE     MPE     MAPE      MASE
## Training set 41.25398 208.2169 165.4141 1.34855 14.05127 0.9026825
##                    ACF1
## Training set -0.1268236
```

```r
#residual analysis
ggtsdisplay(vida.fit.arima$residuals)
```




```r
#box-Ljung Test
Box.test(vida.fit.arima$residuals, lag = 4, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  vida.fit.arima$residuals
## X-squared = 1.2578, df = 1, p-value = 0.2621
```

```r
Box.test(vida.fit.arima$residuals, lag = 8, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  vida.fit.arima$residuals
## X-squared = 3.0841, df = 5, p-value = 0.687
```

```r
Box.test(vida.fit.arima$residuals, lag = 12, fitdf = 3, type = "Lj")
```

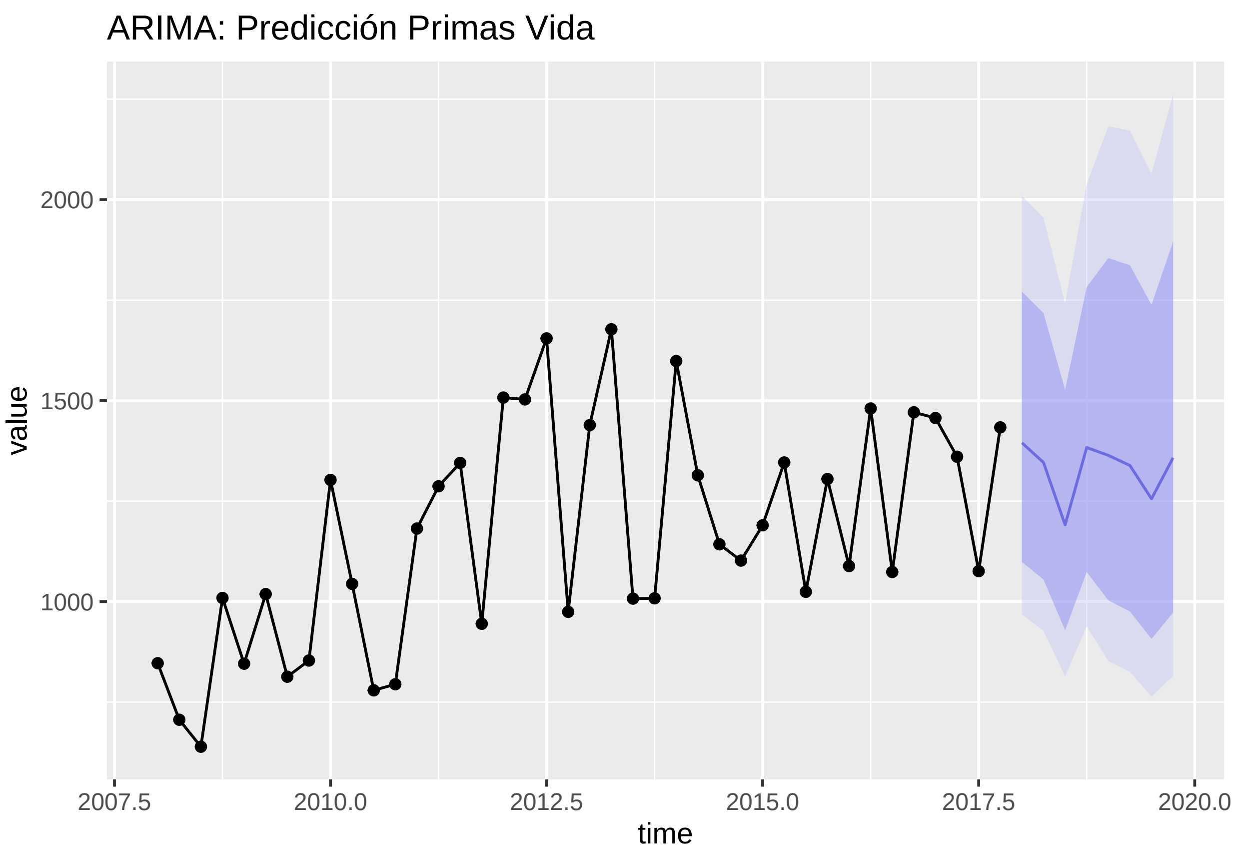
```
##
```

```
##  Box-Ljung test
##
## data:  vida.fit.arima$residuals
## X-squared = 4.0443, df = 9, p-value = 0.9085
```

```r
vida.arima <- forecast(vida.fit.arima)

ggplot(df_vida) + geom_point(aes(x = time,y = value)) +
  geom_line(aes(x = time, y = value)) +
  geom_forecast(vida.arima, alpha = 0.4) +
  ggtitle("ARIMA: Predicción Primas Vida")
```

```
## Warning in geom_forecast(vida.arima, alpha = 0.4): Use autolayer instead of
## geom_forecast to add a forecast layer to your ggplot object.
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```

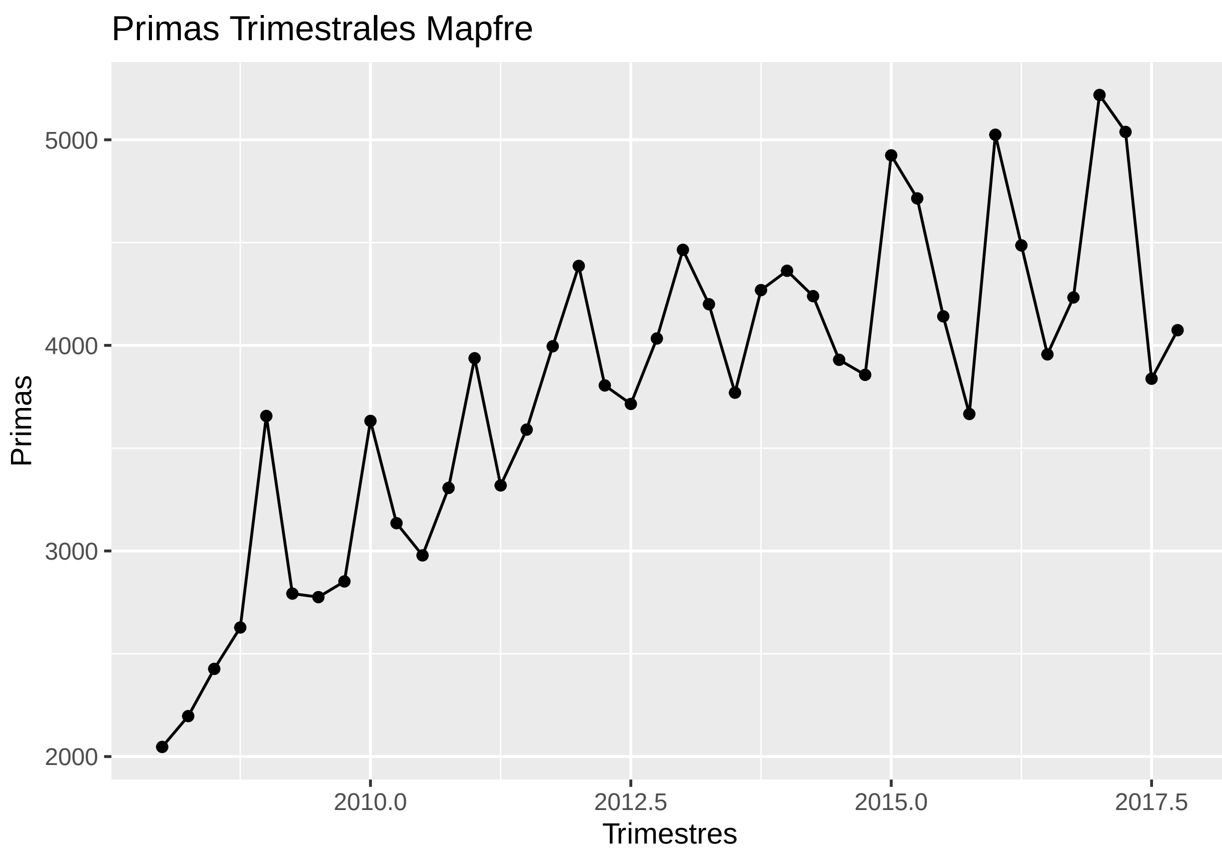


Modelo ARIMA para primas no vida

Preparando datos y análisis por diferencias

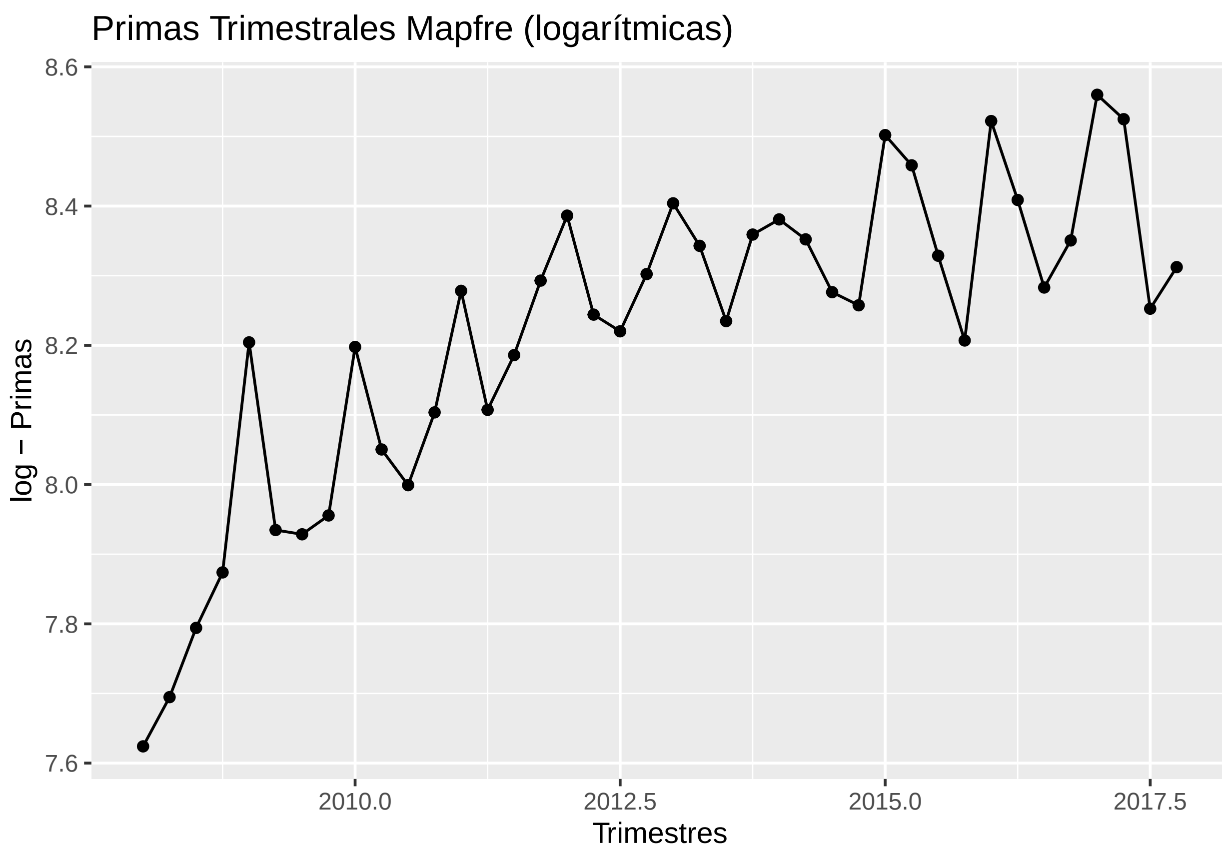
```r
## Nuestra ts de primas no vida se llama "no_vida"

df_no_vida <- data.frame(value = as.vector(no_vida),
                         time = time(no_vida))
ggplot(df_no_vida) + geom_point(aes(x = time, y = value)) +
  geom_line(aes(x = time, y = value)) +
  ylab("Primas") +
  ggtitle("Primas Trimestrales Mapfre") +
  xlab("Trimestres")
```

```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```

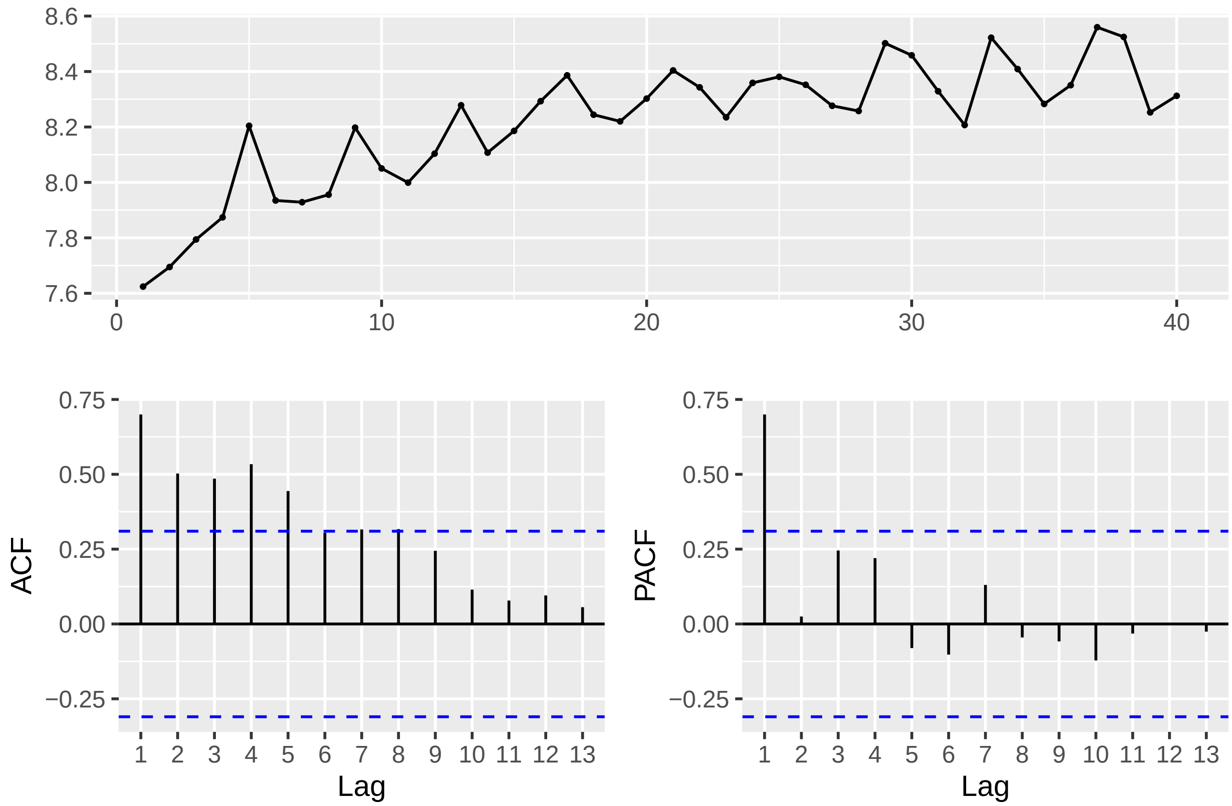



```r
## trabajamos con transformacion logaritmica
logno_vida <- log(no_vida)
df_logno_vida <- data.frame(value = as.vector(logno_vida),
                            time = time(logno_vida))
ggplot(df_logno_vida) + geom_point(aes(x = time, y = value)) +
  geom_line(aes(x = time, y = value)) +
  ylab("log - Primas") +
  ggtitle("Primas Trimestrales Mapfre (logarítmicas)") +
  xlab("Trimestres")
```

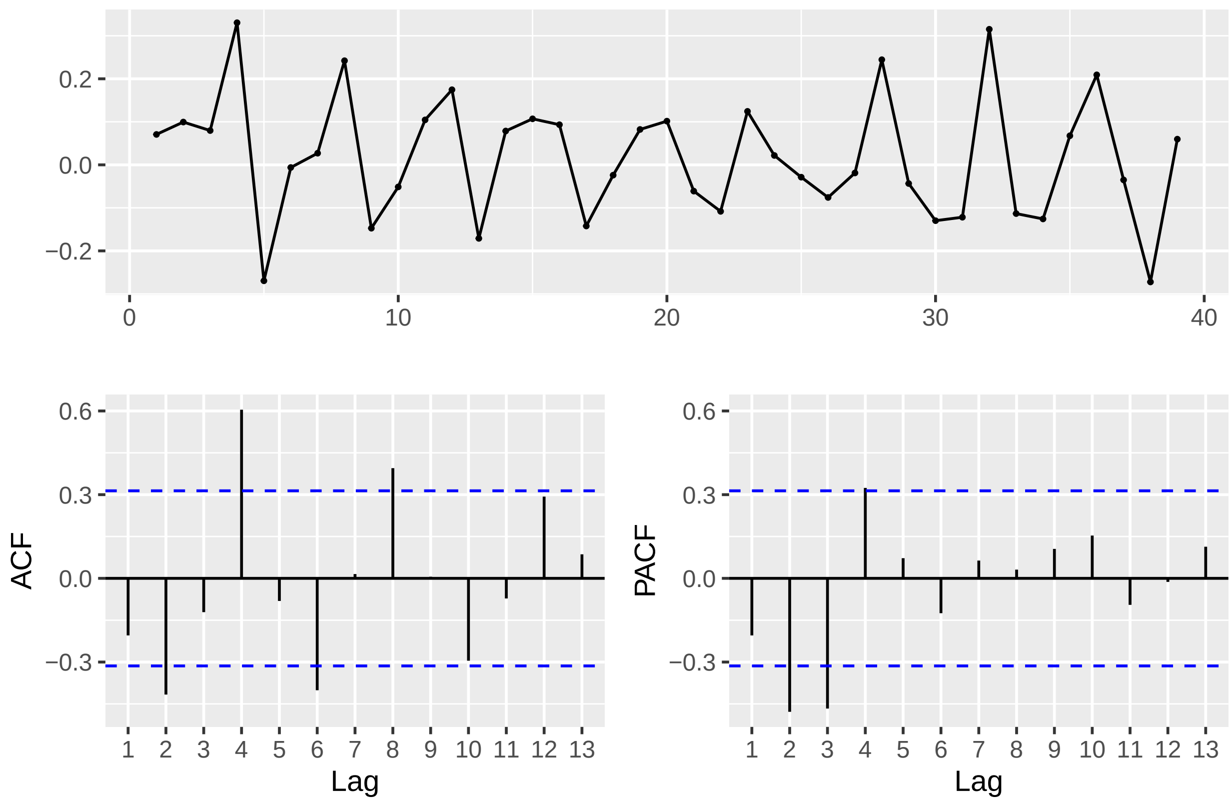
```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```



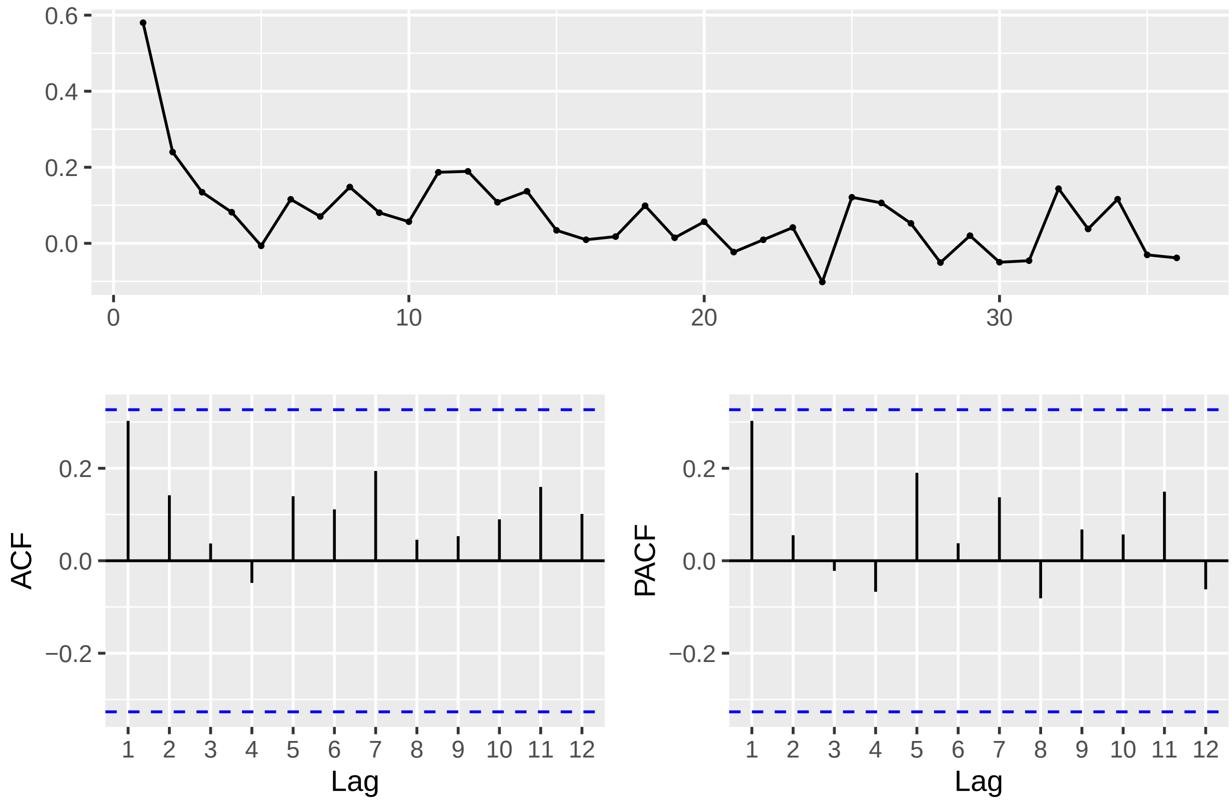

```r
## Difference
ggtsdisplay(logno_vida)
```



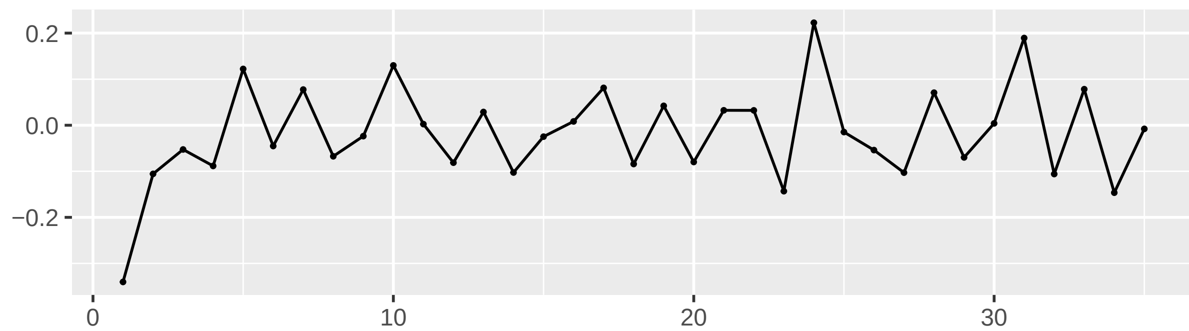

```
ggtsdisplay(diff(logno_vida))
```

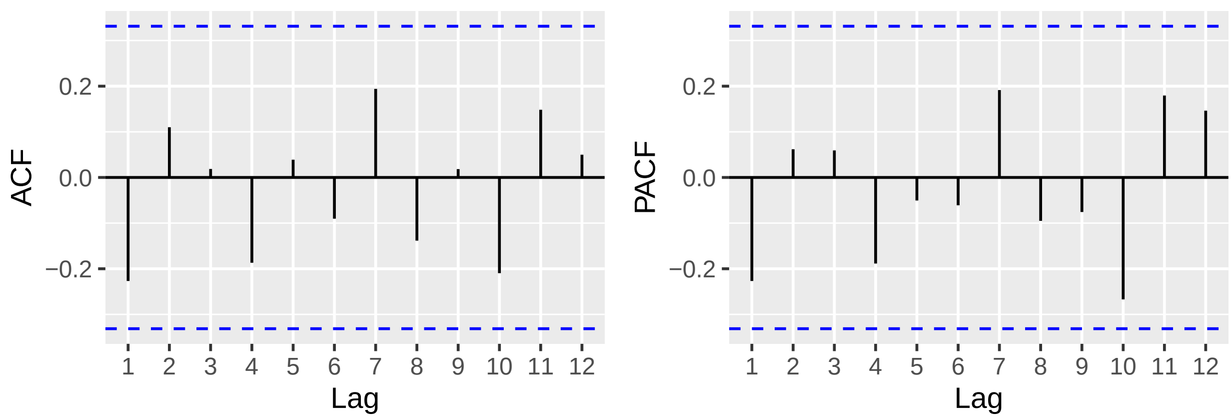



```
ggtsdisplay(diff(logno_vida,4))
```




```r
ggtsdisplay(diff(diff(logno_vida,4),1))
```





Trainig set

```r
#- Training set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

#Select number of observation to compare forecast
cOmit = 4

#Data Size
nObs = length(no_vida)

#sub_sample TRAINING
oano_vida <- window(no_vida, start = index(no_vida[1]), end = index(no_vida[nObs - cOmit]))
```

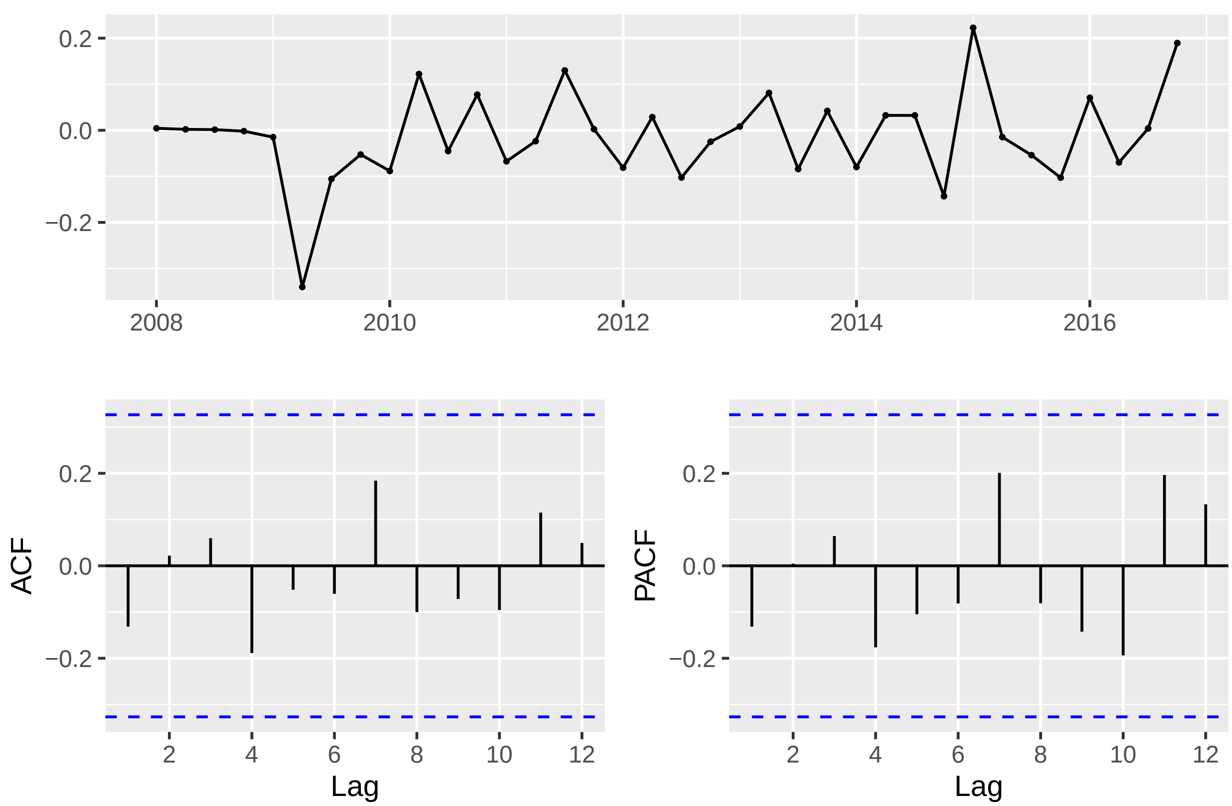
Test

```r
## ARIMA MODEL Automatic selection
no_vida.train.arima = auto.arima(oano_vida,lambda = 0) ## lamnda cero is log transformation
summary(no_vida.train.arima)
```

```
## Series: oano_vida
## ARIMA(0,1,0)(0,1,0)[4]
## Box Cox transformation: lambda= 0
```

```
##
## sigma^2 estimated as 0.01144:  log likelihood=25.31
## AIC=-48.63   AICc=-48.49   BIC=-47.19
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -44.81741 373.5875 269.8852 -1.744338 7.219293 0.8081832
##                    ACF1
## Training set -0.2095555
```

```r
#residual analysis
ggtsdisplay(no_vida.train.arima$residuals)
```




```r
#box-Ljung Test
Box.test(no_vida.train.arima$residuals,lag = 4, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  no_vida.train.arima$residuals
## X-squared = 2.3656, df = 1, p-value = 0.124
```

```r
Box.test(no_vida.train.arima$residuals,lag = 8, fitdf = 3, type = "Lj")
```

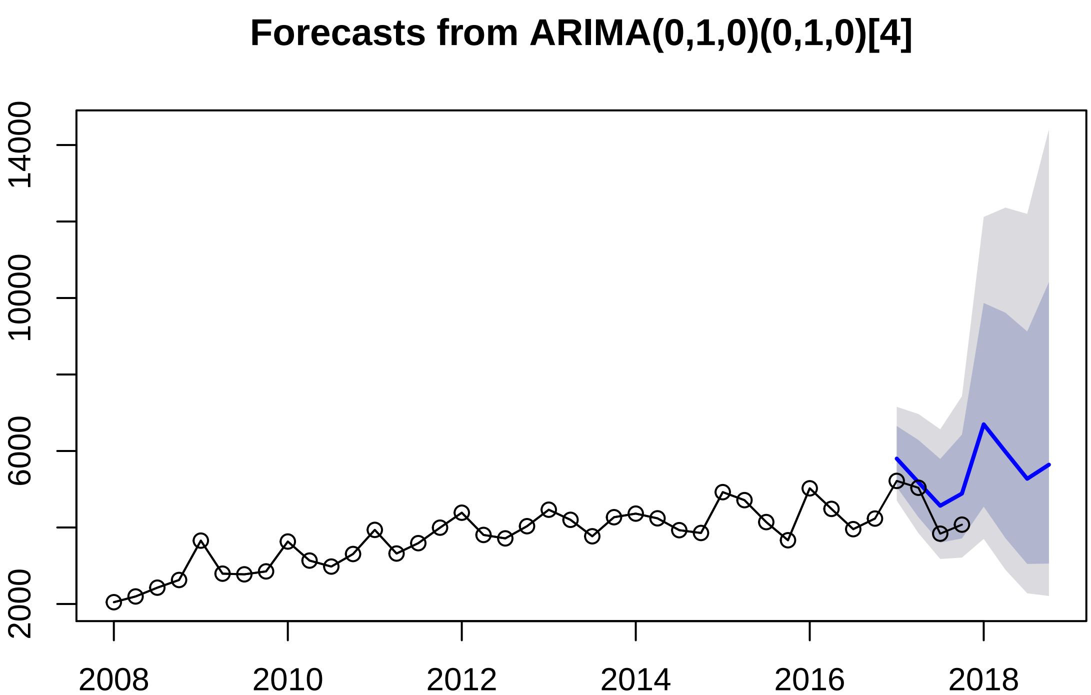
```
##
```

```
##  Box-Ljung test
##
## data:  no_vida.train.arima$residuals
## X-squared = 4.7391, df = 5, p-value = 0.4485
```

```
Box.test(no_vida.train.arima$residuals,lag = 12, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  no_vida.train.arima$residuals
## X-squared = 6.3414, df = 9, p-value = 0.7053
```

```
plot(forecast(no_vida.train.arima))
lines(window(no_vida),type = "o")
```



Métrica de predicción

```
fno_vida_arima <- forecast(no_vida.train.arima)

no_vidaArimaMatrix <- matrix(c(fno_vida_arima$mean[1:4], as.double(tail(no_vida,4))), ncol = 2)
no_vidaArimaMatrix
```

```
##          [,1]   [,2]
## [1,] 5801.000 5217.9
## [2,] 5179.625 5038.2
## [3,] 4567.602 3837.7
## [4,] 4887.064 4073.8
```

```r
## MSE
mean((no_vidaArimaMatrix[,1] - no_vidaArimaMatrix[,2])^2)
```

```
## [1] 388540.4
```

```r
## MAE
mean(abs(no_vidaArimaMatrix[,1] - no_vidaArimaMatrix[,2]))
```

```
## [1] 566.9227
```

```r
## Bias
mean(no_vidaArimaMatrix[,1] - no_vidaArimaMatrix[,2])
```

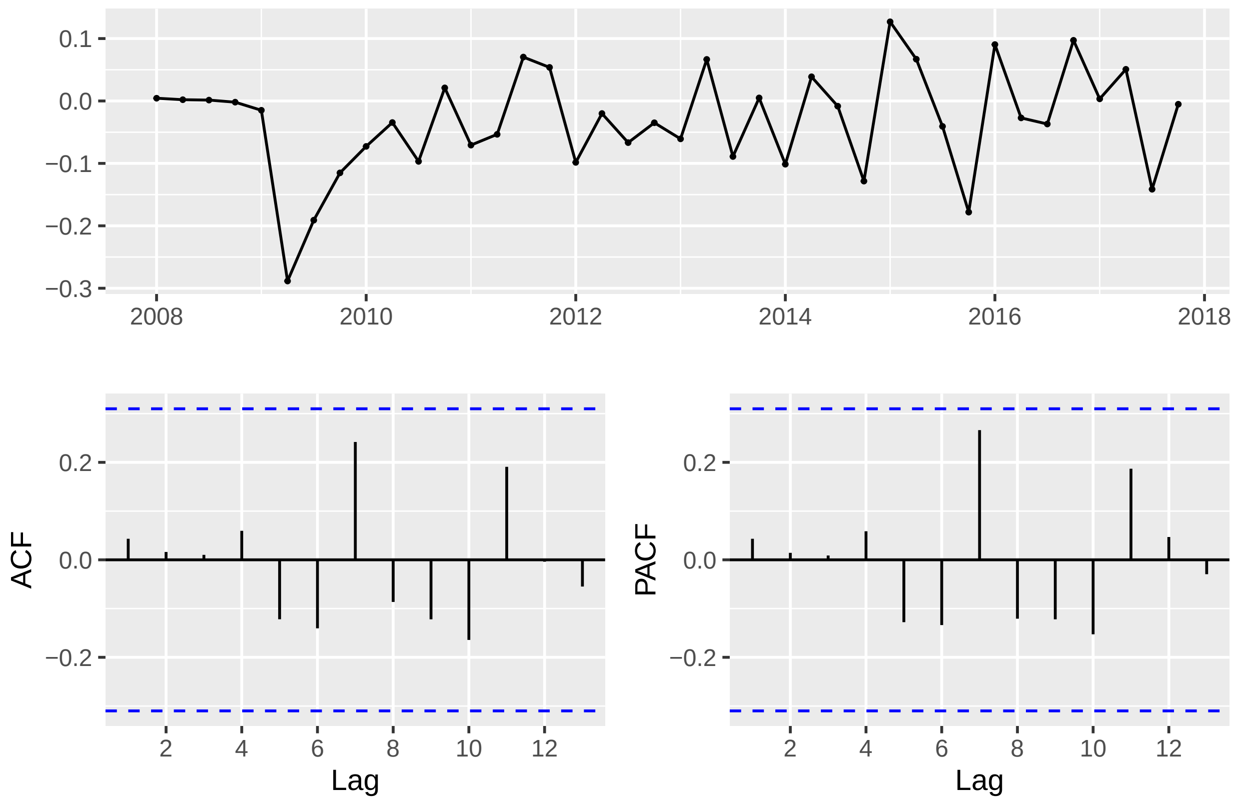
```
## [1] 566.9227
```

Predicción

```r
#- Complete set     -#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#


## ARIMA MODEL Automatic selection
no_vida.fit.arima <- auto.arima(no_vida, lambda = 0) ## lamnda cero para transformacion log
summary(no_vida.fit.arima)
```

```
## Series: no_vida
## ARIMA(0,1,1)(0,1,1)[4]
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ma1     sma1
##       -0.3582  -0.4830
## s.e.   0.2172   0.2045
##
## sigma^2 estimated as 0.009745:  log likelihood=31.82
## AIC=-57.63   AICc=-56.86   BIC=-52.97
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -111.42 338.9629 260.2416 -3.616537 6.976557 0.8001279
##                     ACF1
## Training set -0.08049807
```

```r
#residual analysis
ggtsdisplay(no_vida.fit.arima$residuals)
```




```r
#box-Ljung Test
Box.test(no_vida.fit.arima$residuals, lag = 4, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  no_vida.fit.arima$residuals
## X-squared = 0.26198, df = 1, p-value = 0.6088
```

```r
Box.test(no_vida.fit.arima$residuals, lag = 8, fitdf = 3, type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  no_vida.fit.arima$residuals
## X-squared = 5.3205, df = 5, p-value = 0.378
```

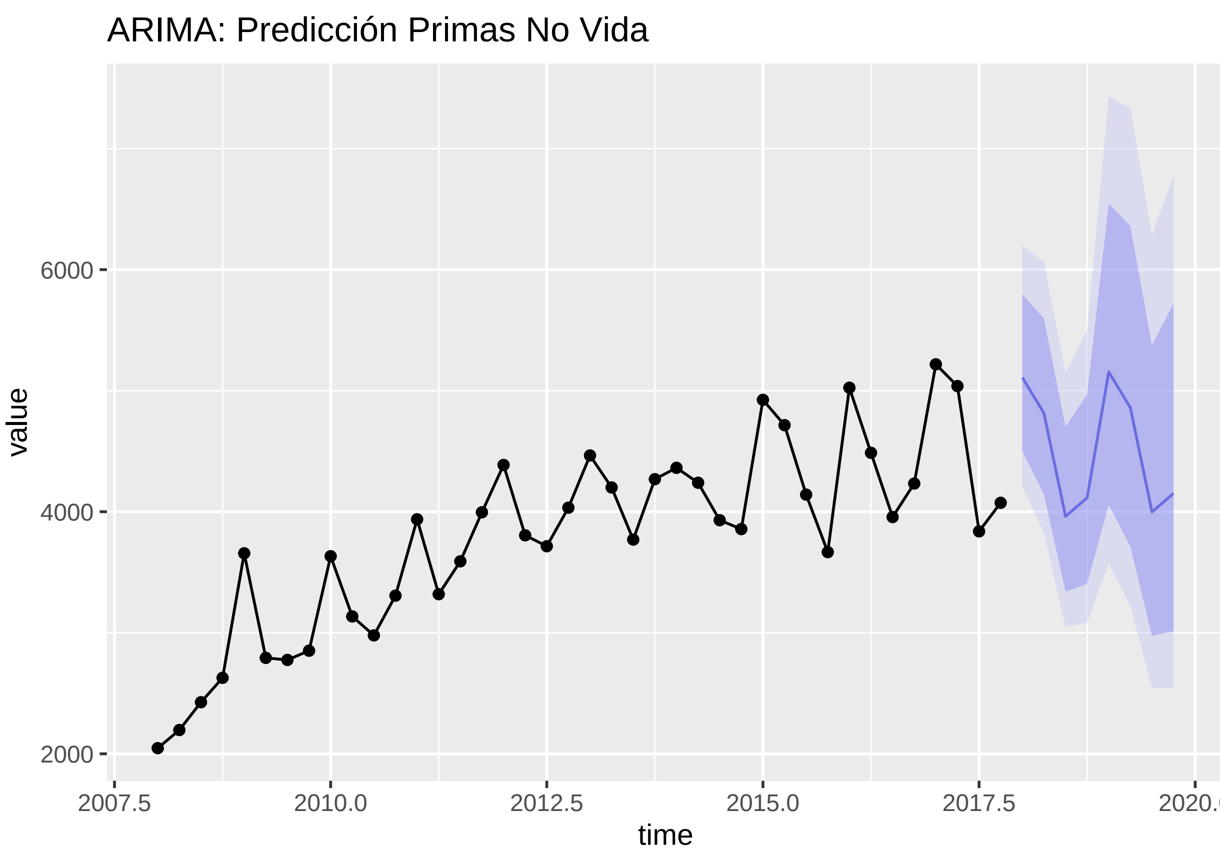
```r
Box.test(no_vida.fit.arima$residuals, lag = 12, fitdf = 3, type = "Lj")
```

```
##
```

```
##  Box-Ljung test
##
## data:  no_vida.fit.arima$residuals
## X-squared = 9.7503, df = 9, p-value = 0.3711
```

```
no_vida.arima <- forecast(no_vida.fit.arima)

ggplot(df_no_vida) + geom_point(aes(x = time,y = value)) +
  geom_line(aes(x = time, y = value)) +
  geom_forecast(no_vida.arima, alpha = 0.4) +
  ggtitle("ARIMA: Predicción Primas No Vida")
```

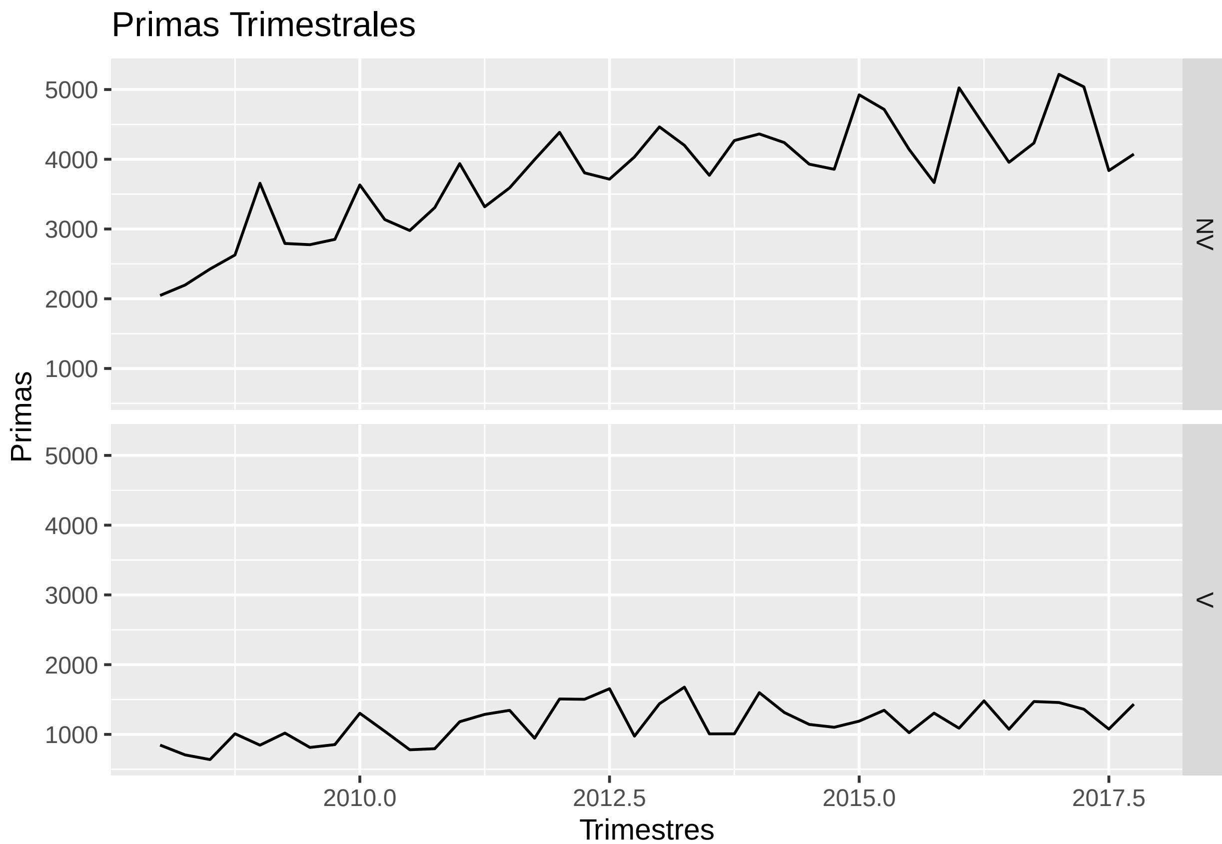
```
## Warning in geom_forecast(no_vida.arima, alpha = 0.4): Use autolayer instead
## of geom_forecast to add a forecast layer to your ggplot object.

## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```



Modelo ARIMA para primas HTS ((???))

```
## modelo ARIMA desde el enfoque de series de tiempo jerarquicas (hts)
##

## Plot Series
autoplot(sepxts) + ggtitle("Primas Trimestrales") + xlab("Trimestres") + ylab("Primas")
```

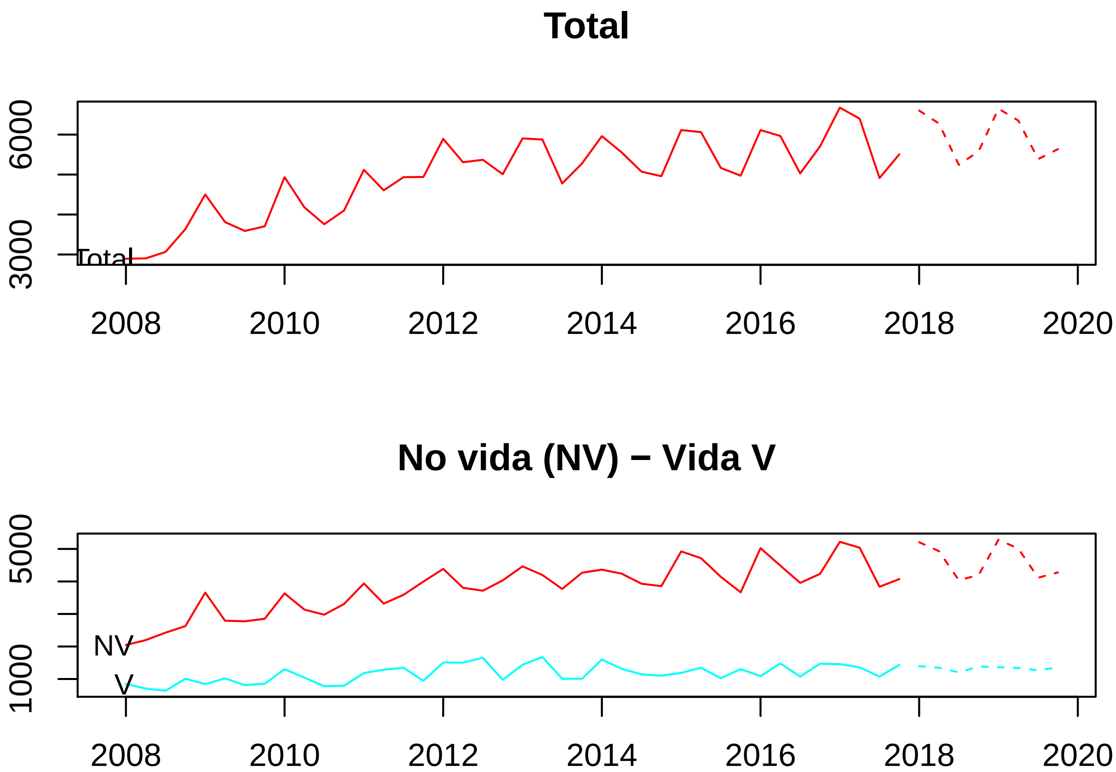
```
## Don't know how to automatically pick scale for object of type yearqtr. Defaulting to continuous.
```




```r
## Select automatic HTS
sepmod <- hts(sepxts, nodes = list(2))
```

```
## Since argument characters are not specified, the default labelling system is used.
```

```r
## Forcast
sep.fit.arima <- forecast(sepmod, method = 'bu', fmethod = 'arima') ## buttom up arima
names(sep.fit.arima$labels) = c("Total", "No vida (NV) - Vida V")
plot(sep.fit.arima)
```



Suma vida y no vida ARIMA

```r
## Despues de trabajar por separado vida y no vida sumamos para ver la prediccion total

sumaFitArimaMat <- vidaArimaMatrix + no_vidaArimaMatrix
sumaFitArimaMat
```

```
##          [,1]   [,2]
## [1,] 6987.262 6674.6
## [2,] 6574.496 6398.6
## [3,] 5745.342 4913.4
## [4,] 6277.214 5507.2
```

```r
## MSE
mean((sumaFitArimaMat[,1] - sumaFitArimaMat[,2])^2)
```

```
## [1] 353436.3
```

```r
## MAE
mean(abs(sumaFitArimaMat[,1] - sumaFitArimaMat[,2]))
```

```
## [1] 522.6284
```

```r
## Bias
mean(sumaFitArimaMat[,1] - sumaFitArimaMat[,2])
```

```
## [1] 522.6284
```

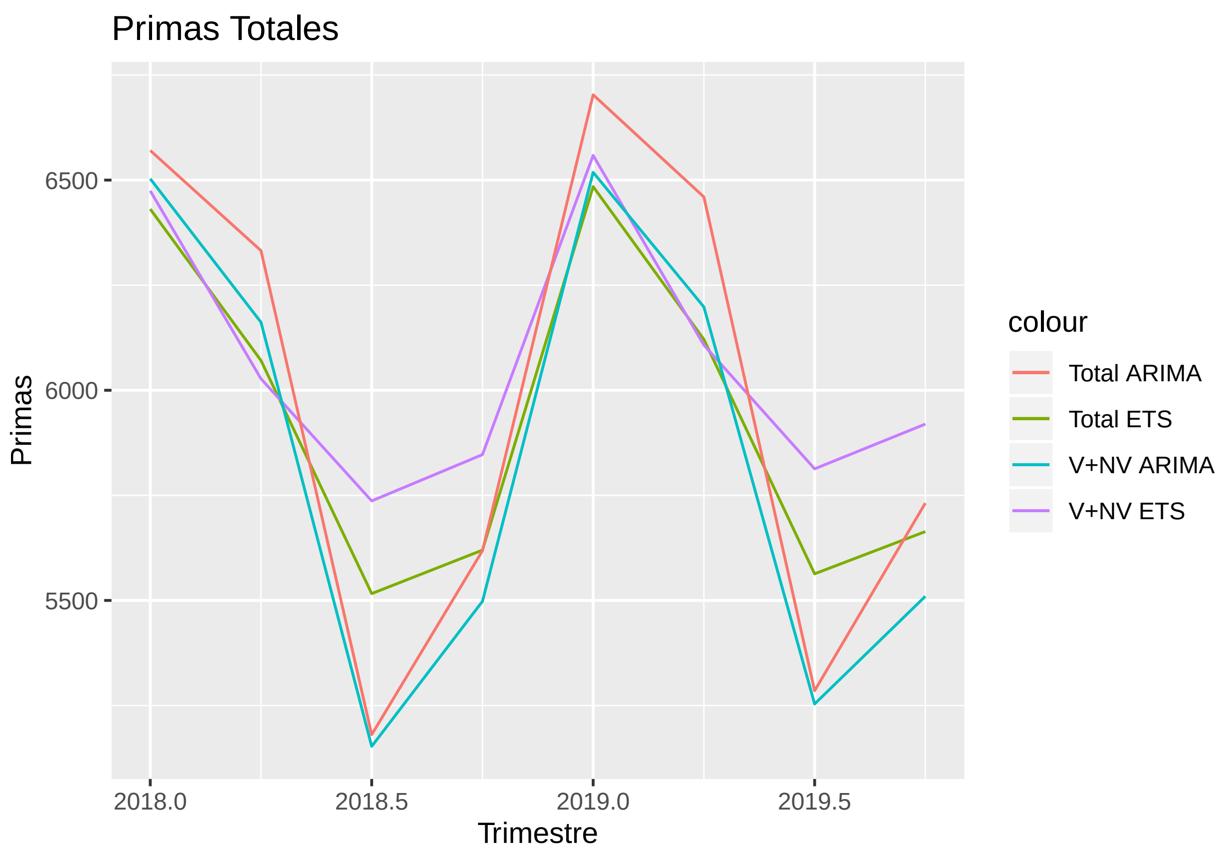
Conclusiones y comparación de resultados

```r
## Conclusiones ##########################################################################################
##

## Comparamos los forcast de los modelos
compmat <-  matrix(c(as.vector(total.ets$mean),
                     as.vector(vida.ets$mean),
                     as.vector(no_vida.ets$mean),
                     as.vector(vida.ets$mean) + as.vector(no_vida.ets$mean),
                     as.vector(sep.fit$bts[,2]),
                     as.vector(sep.fit$bts[,1]),
                     as.vector(sep.fit$bts[,1]) + as.vector(sep.fit$bts[,2]),
                     as.vector(tota.arima$mean),
                     as.vector(vida.arima$mean),
                     as.vector(no_vida.arima$mean),
                     as.vector(vida.arima$mean) + as.vector(no_vida.arima$mean),
                     as.vector(sep.fit.arima$bts[,2]),
                     as.vector(sep.fit.arima$bts[,1]),
                     as.vector(sep.fit.arima$bts[,1]) + as.vector(sep.fit.arima$bts[,2]),
                     as.vector(time(total.ets$mean))),ncol = 15)

colnames(compmat) <- c("Total ETS",
                       "Vida ETS",
                       "No Vida ETS",
                       "V+NV ETS",
                       "Vida ETS HTS",
                       "No Vida ETS HTS",
                       "V+NV ETS HTS",
                       "Total ARIMA",
                       "Vida ARIMA",
                       "No Vida ARIMA",
                       "V+NV ARIMA",
                       "Vida ARIMA HTS",
                       "No Vida ARIMA HTS",
                       "V+NV ARIMA HTS",
                       "Trimestre")

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

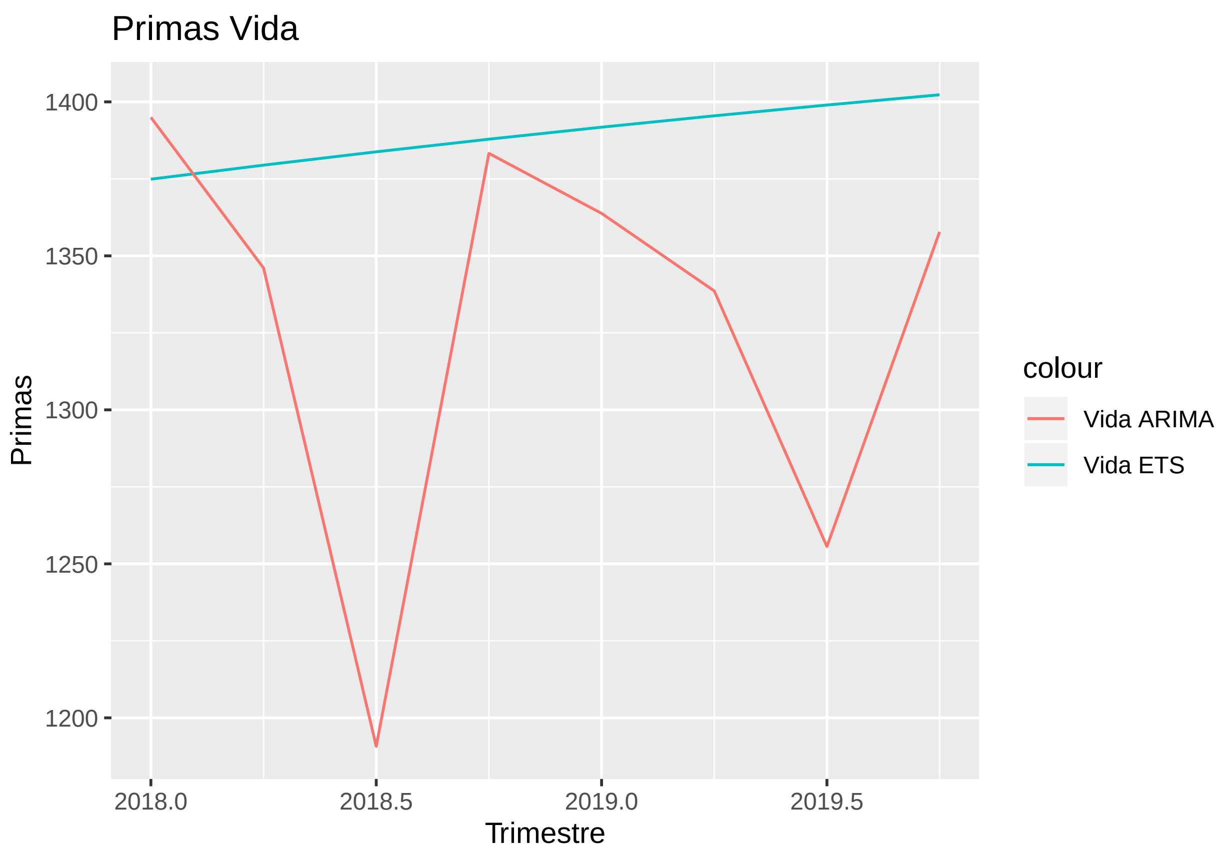
```
##   Total ETS Vida ETS No Vida ETS V+NV ETS Vida ETS HTS No Vida ETS HTS
## 1  6431.081 1374.896    5099.550 6474.446     1374.896        5099.550
## 2  6070.836 1379.456    4647.861 6027.317     1379.456        4647.861
## 3  5516.262 1383.782    4352.951 5736.733     1383.782        4352.951
## 4  5619.637 1387.887    4458.972 5846.859     1387.887        4458.972
```

```
## 5  6484.425 1391.781    5166.934 6558.715     1391.781        5166.934
## 6  6120.941 1395.476    4712.056 6107.532     1395.476        4712.056
## 7  5563.325 1398.981    4414.108 5813.089     1398.981        4414.108
## 8  5663.844 1402.307    4517.235 5919.542     1402.307        4517.235
##   V+NV ETS HTS Total ARIMA Vida ARIMA No Vida ARIMA V+NV ARIMA
## 1     6474.446    6570.393   1394.975      5108.165   6503.140
## 2     6027.317    6332.097   1346.039      4815.670   6161.709
## 3     5736.733    5180.705   1190.733      3962.121   5152.854
## 4     5846.859    5618.303   1383.281      4114.581   5497.862
## 5     6558.715    6702.945   1363.794      5154.649   6518.443
## 6     6107.532    6459.842   1338.602      4859.493   6198.094
## 7     5813.089    5285.221   1255.602      3998.176   5253.778
## 8     5919.542    5731.648   1357.813      4152.024   5509.836
##   Vida ARIMA HTS No Vida ARIMA HTS V+NV ARIMA HTS Trimestre
## 1       1394.124          5206.736       6600.859   2018.00
## 2       1346.643          4930.610       6277.253   2018.25
## 3       1206.271          4037.064       5243.335   2018.50
## 4       1382.635          4199.617       5582.252   2018.75
## 5       1363.270          5286.058       6649.328   2019.00
## 6       1339.860          5009.933       6349.792   2019.25
## 7       1270.649          4116.386       5387.035   2019.50
## 8       1357.606          4278.939       5636.545   2019.75
```

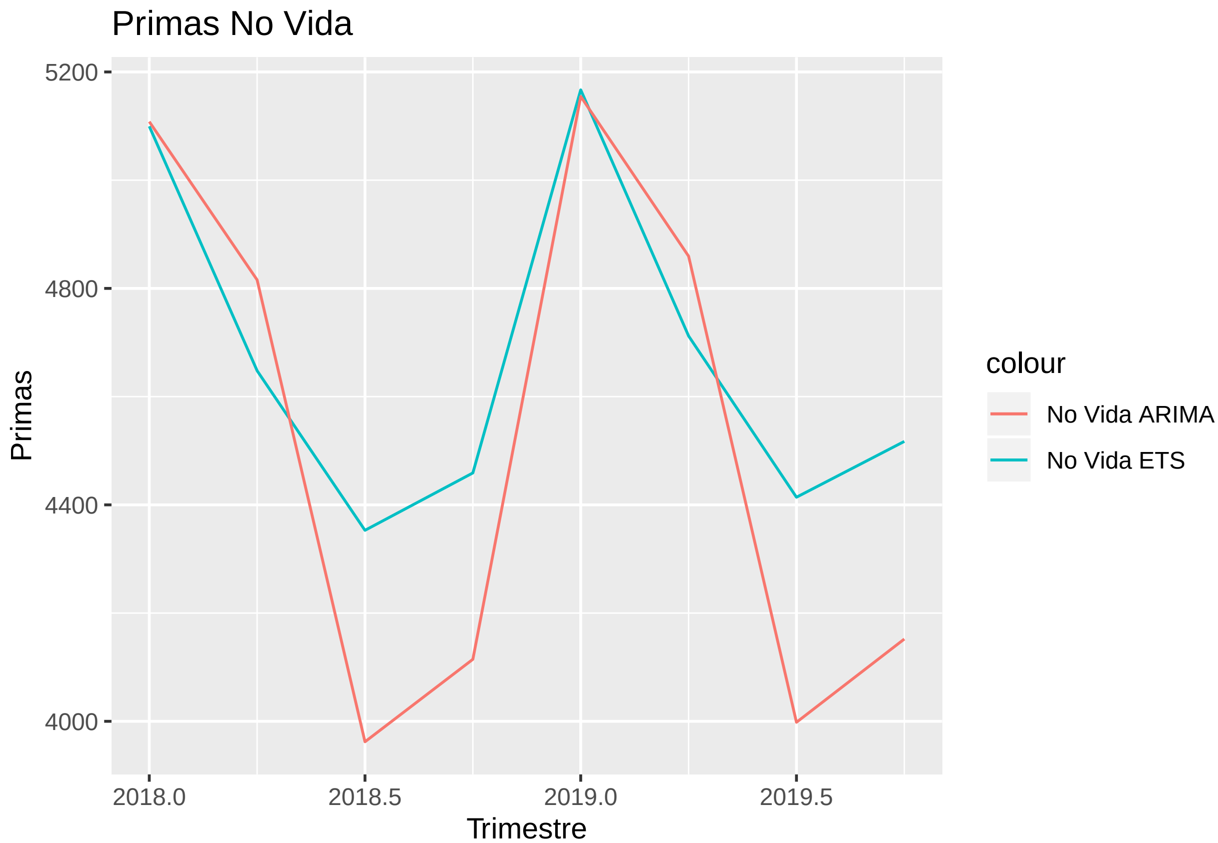
```r
## comparamos graficamente
ggplot(data = compmat, aes(x = Trimestre)) +
  geom_line(aes(y = `Total ETS`, colour = "Total ETS")) +
  geom_line(aes(y = `V+NV ETS`, colour = "V+NV ETS")) +
  geom_line(aes(y = `Total ARIMA`, colour = "Total ARIMA")) +
  geom_line(aes(y = `V+NV ARIMA`, colour = "V+NV ARIMA")) +
  ggtitle("Primas Totales") + ylab("Primas")
```




```
ggplot(data = compmat, aes(x = Trimestre)) +
  geom_line(aes(y = `Vida ETS`, colour = "Vida ETS")) +
  geom_line(aes(y = `Vida ARIMA`, colour = "Vida ARIMA")) +
  ggtitle("Primas Vida") + ylab("Primas")
```




```r
ggplot(data = compmat, aes(x = Trimestre)) +
  geom_line(aes(y = `No Vida ETS`, colour = "No Vida ETS")) +
  geom_line(aes(y = `No Vida ARIMA`, colour = "No Vida ARIMA")) +
  ggtitle("Primas No Vida") + ylab("Primas")
```


Primas No Vida
5200
4800
4400
4000
Primas
2018.0
2018.5
2019.0
2019.5
Trimestre
colour
No Vida ARIMA
No Vida ETS