UNIVERSIDAD
NACIONAL DEL OESTE

Explotación de Datos

Actividad N$^{\text{o}}$ 5

Análisis de Serie Temporal

Profesores:

Dejean, Gustavo
Españadero, Juan
Mendoza, Dante

Integrantes Grupo B:

Benitez, Nicolas
Garcia Ravlic, Ignacio Agustin
Rechimon, Pablo Hernan
Rodríguez, Miguel Ángel

Fecha de entrega:

7 de Noviembre de 2020

Resumen

A partir de el leguaje R y los datos obtenidos del gobierno, acerca de el personal de servicios informaticos con medidas relativas, hemos realizado un analisis de su evolucion en el tiempo con el objetivo de desarrollar un modelo de prediccion con gran precision y bajo margen de error, para esto desarrollamos multiples modelos, los pusimos a prueba, los calibramos y los comparamos para asi quedarnos con el que mejor resultados nos dio, siendo este el desarrollado con *Prophet*.

Palabras Clave: *Analisis de Datos - Ciencia de Datos - Modelo Predictor - Lenguaje R - Time Series*

Índice

Graficos

1 Introducción

1.1 Problemática

Se realiza un estudio de la evolución relativa en el tiempo de la tasa del personal asalariado, no asalariado y ocupado del sector de servicios informáticos en la Ciudad de Buenos Aires.

1.2 Datos a utilizar

Utilizaremos el dataset *Personal ocupado, asalariado y no asalariado del sector servicios informáticos. Ciudad de Buenos Aires. Primer trimestre de 2008/segundo trimestre de 2020.*, provisto por Estadística y censos de la ciudad de Buenos Aires.

1.3 Objetivo

Mediante el análisis de series temporales queremos determinar cual es el mejor modelo predictor. Esto incluye, aquel modelo que mejor prediga la realidad, y contenga el margen de error más pequeño.

2 Desarrollo

2.1 Análisis de los datos

Nuestro dataset contiene 4 variables con 50 filas:

- Periodo. Cada año está dividido en cuartos.
- Personal ocupado. Personas remuneradas y no remuneradas contribuyentes por su trabajo de producción de bienes y servicios.
- Personal asalariado. Trabajan en relación de dependencia por un sueldo
- Personal no asalariado. Trabajan sin obtener sueldo (propietarios, pasantes, familiares)

La primer fila de las 3 variables tienen un valor de 1, siendo esta la referencia para las demás medidas realizadas.

	Período	Personal ocupado	Personal asalariado	Personal no asalariado
1	Q1/08	1.0000000	1.0000000	1.0000000
2	Q2/08	1.0143980	1.0084151	1.0785770
3	Q3/08	1.0172508	1.0173944	1.0154282
4	Q4/08	1.0159967	1.0140340	1.0369118
5	Q1/09	0.9660764	0.9806561	0.8099784
6	Q2/09	1.0220705	1.0292104	0.9455017
7	Q3/09	0.9864504	0.9910203	0.9371410
8	Q4/09	0.9350177	0.9331504	0.9551600
9	Q1/10	0.9381656	0.9354357	0.9675606
10	Q2/10	1.2175090	1.2138968	1.2561981

Fig. 1: Contenido del Dataset

2.2 Preparación de los datos

Preparamos los datos en base a nuestros requerimientos:

- Eliminamos los na con la funcion *na.omit()*
- Transformamos los datos en una serie temporal de 3 cadenas.

2.3 Análisis de las series temporales

Realizamos el plot de la serie temporal del conjunto de datos

Fig. 2: Evolución en el tiempo

Se puede visualizar que la serie del personal ocupado y pesonal asalariado se asemejan, siguen un mismo patroń a lo largo del tiempo y apartir del 2012 se puede observar una forma estacionaria, no sucede lo mismo con la serie del personal no asalariado que presenta una forma mas irregular, teniendo variados y grandes altibajos.

2.3.1 Descomposición

Descomponemos la serie de las tres cadenas para analizar sus componentes

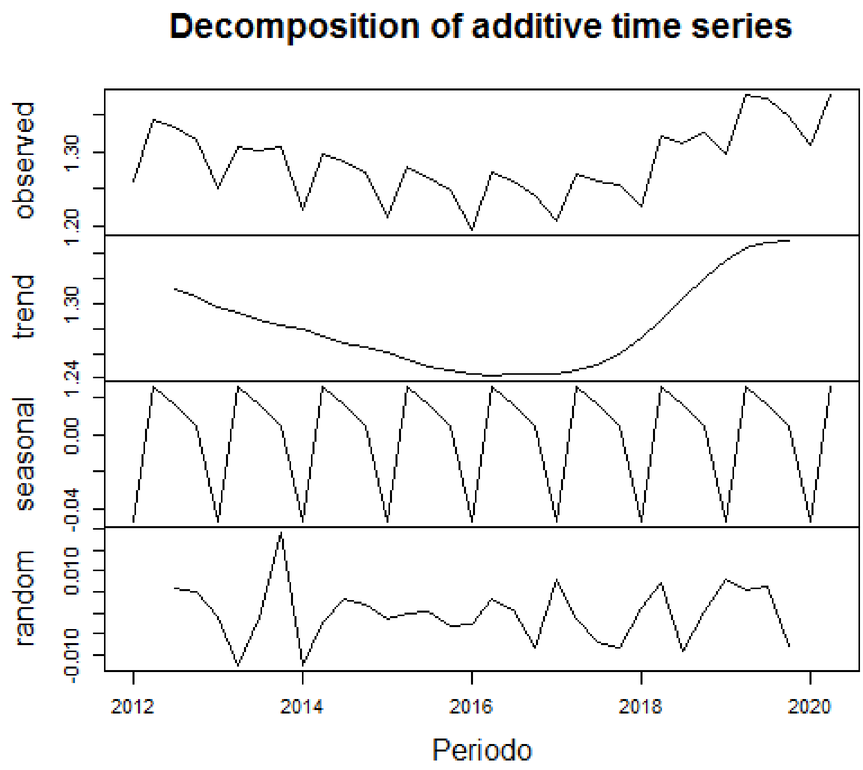

Fig. 3: Descomposición del personal ocupado

Se puede visualizar que la serie del personal ocupado presenta una tendencia no muy marcada, una estacionalidad bien definida con picos y bajadas a lo largo del tiempo y el ruido algo irregular

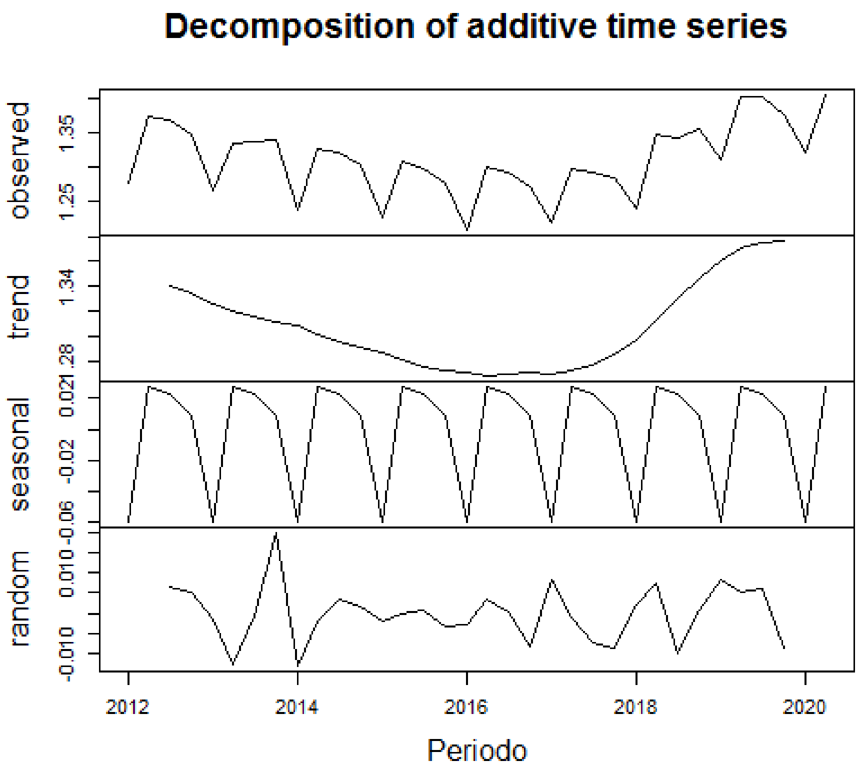

Fig. 4: Descomposición del personal asalariado

Se puede visualizar que la descomposición del personal asalariado es muy parecida a la anterior, el personal ocupado, sin una tendencia marcada, una estacionalidad bien definida y ruido mas bien irregular

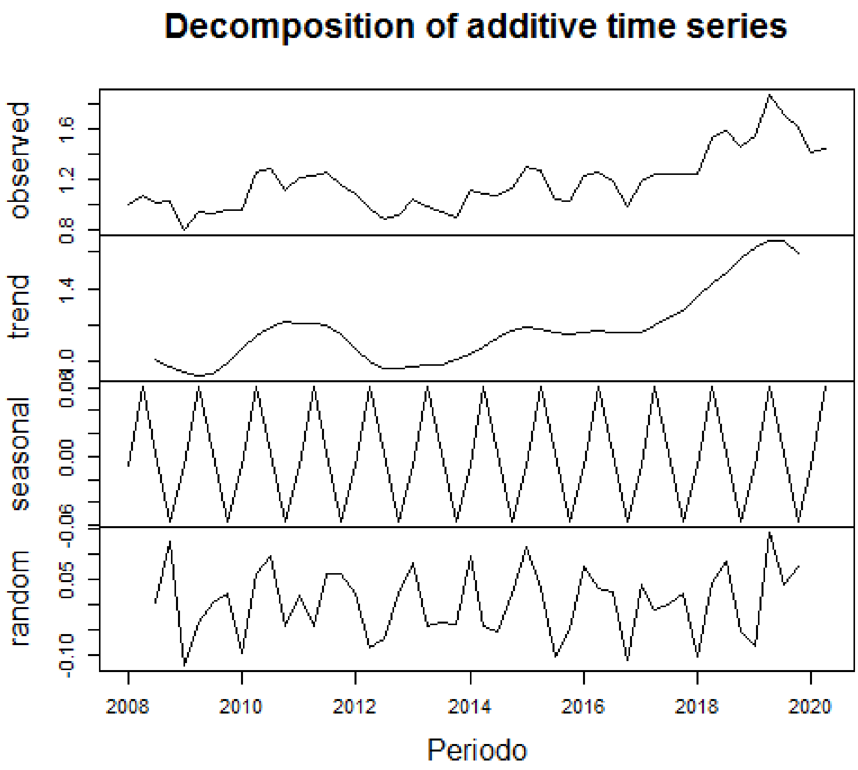

Fig. 5: Descomposición del personal no asalariado

Se puede visualizar que las componentes del personal no asalariado tiene una tendencia ascendente a lo largo del periodo, una estacionalidad bien marcada y un ruido bastante constante

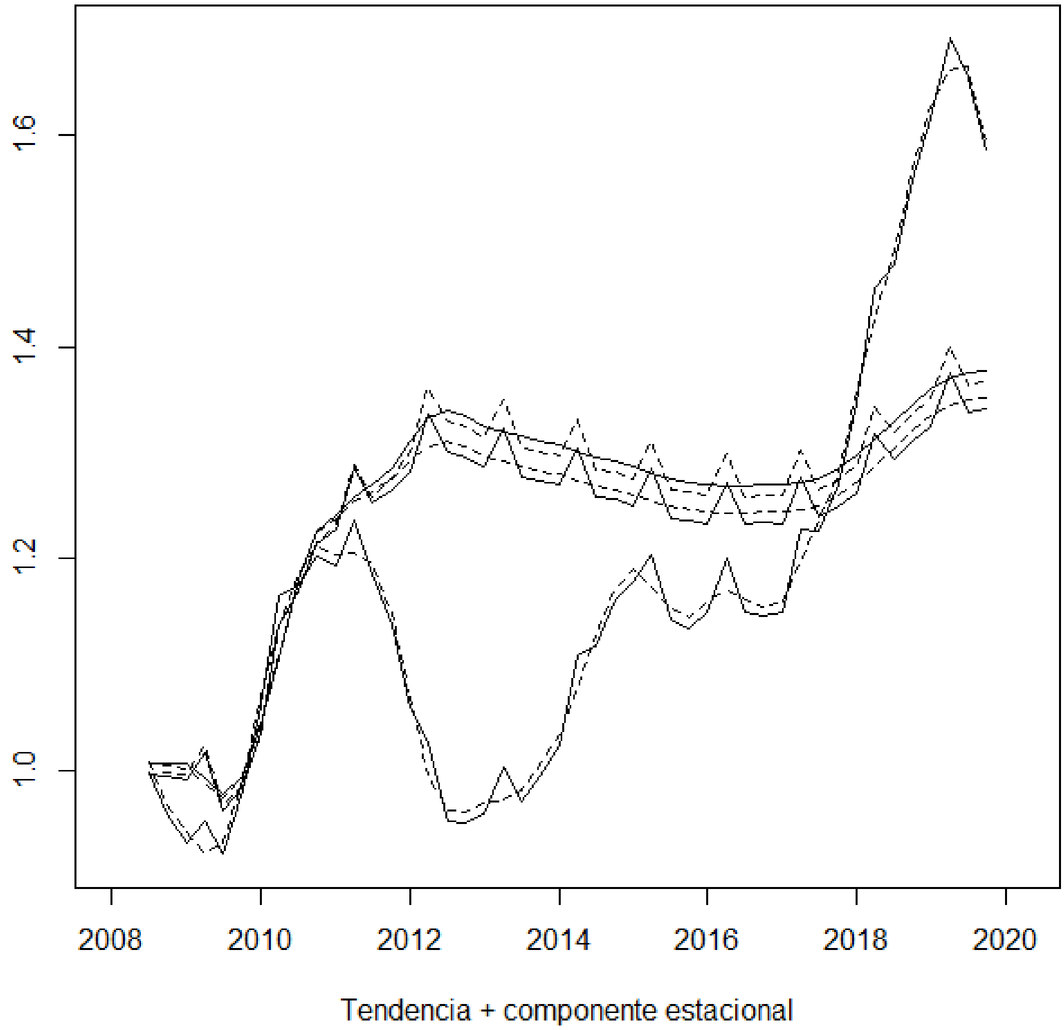

Fig. 6: Tendencia y Estacionalidad

2.3.2 Diferencias

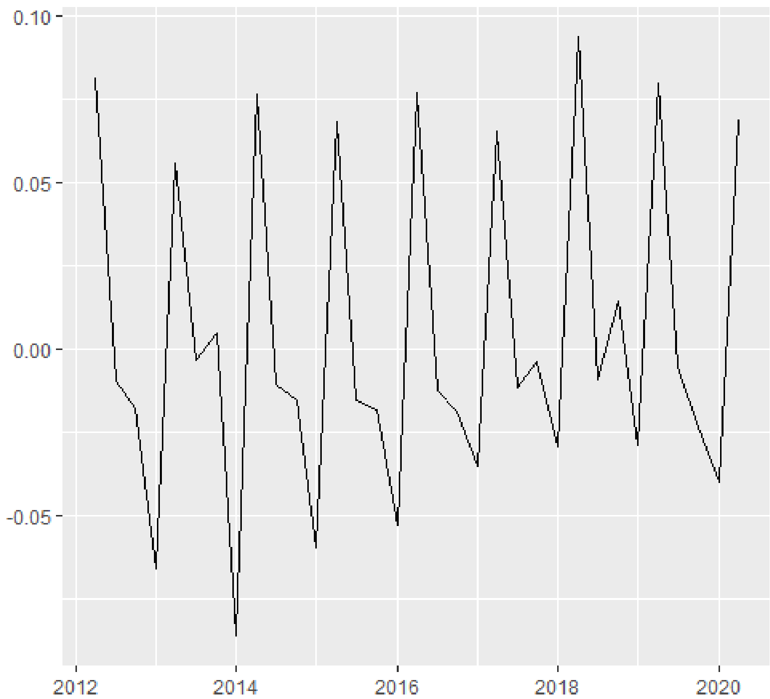

Fig. 7: Diferencias del Personal Ocupado

Podemos observar que las diferencias se encuentran entre el -0.1 y 0.1, viendo que los picos altos siempre pertenecen al primer cuarto y los mas bajos al ultimo cuarto, indicando asi una posible estacionalidad.

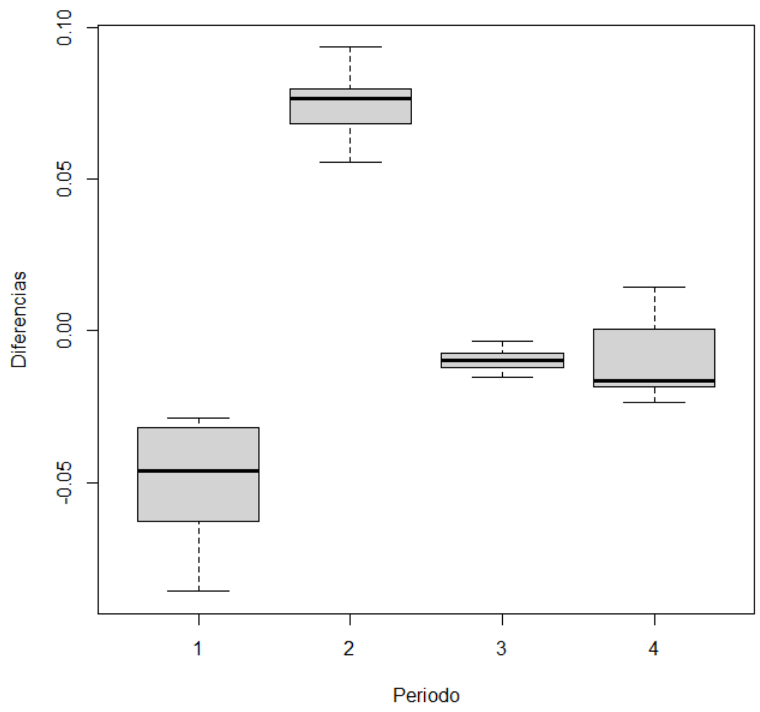

Fig. 8: Boxplot de Diferencias del Personal Ocupado

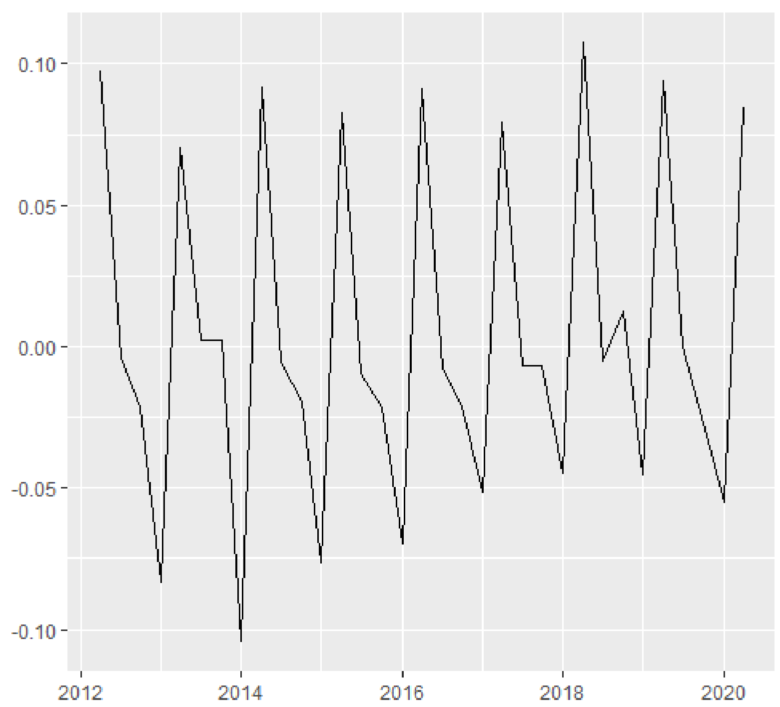

Fig. 9: Diferencias del Personal Asalariado

Podemos observar al igual que en el anterior gráfico que las diferencias se encuentran entre el -0.1 y 0.1, y que como era de esperar, el patrón de los picos es muy similar.

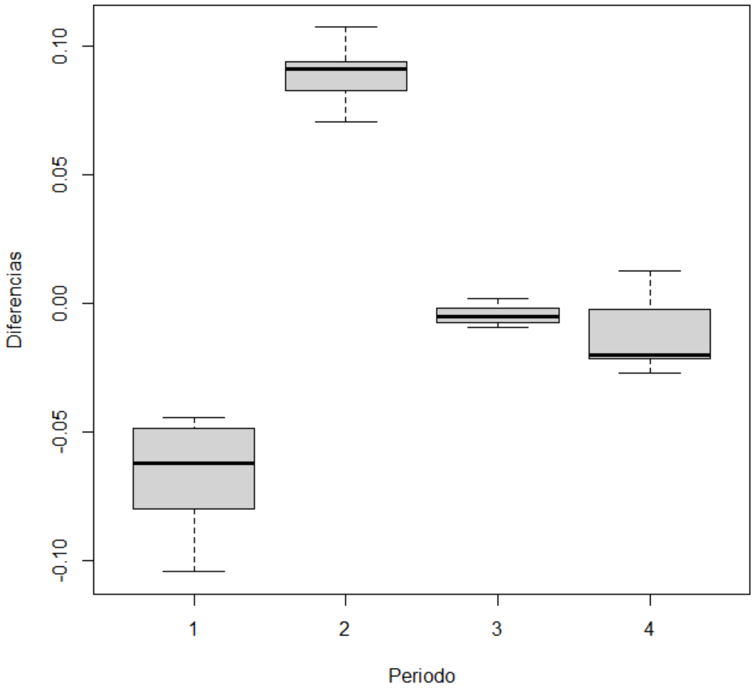

Fig. 10: Boxplot de Diferencias del Personal Asalariado

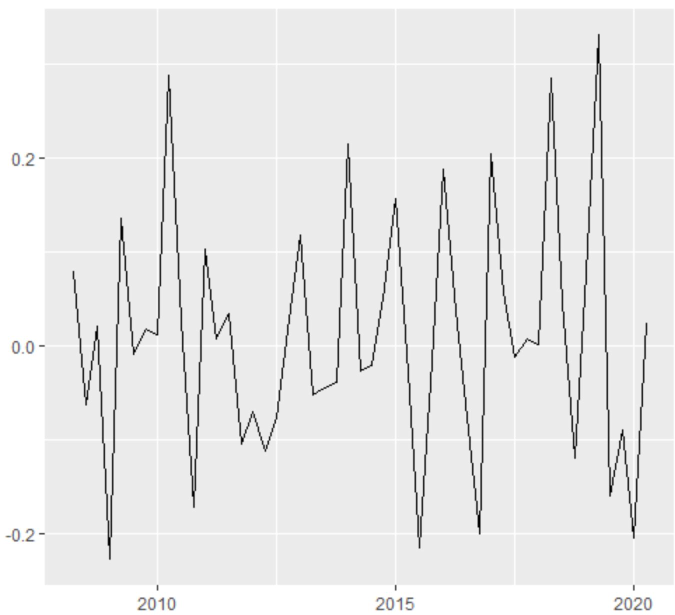

Fig. 11: Diferencias del Personal No Asalariado

Ya para este último caso vemos que las diferencias son aun mayores, y que no es posible observar algun patrón entre sus picos, dandonos un indicio en que esta serie no es estacional.

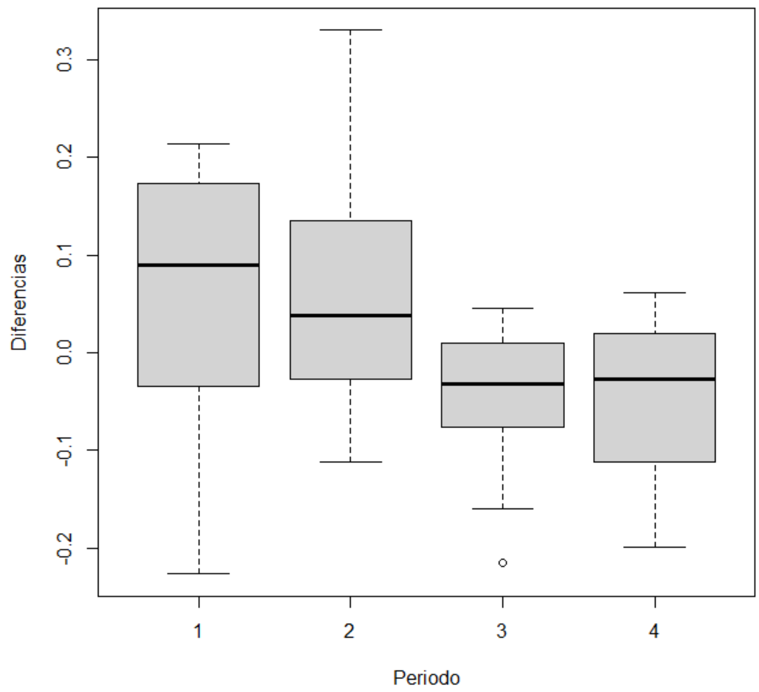

Fig. 12: Boxplot de Diferencias del Personal No Asalariado

2.3.3 Autocorrelaciones

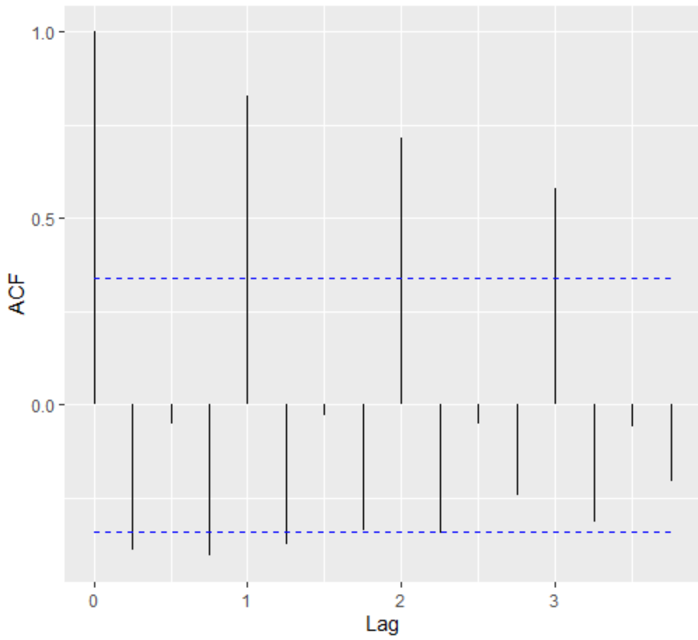

Fig. 13: Autocorrelaciones del Personal Ocupado

Se observa como los primeros cualros tienen una alta autocorrelación, confirmandonos de que esta serie es estacional.

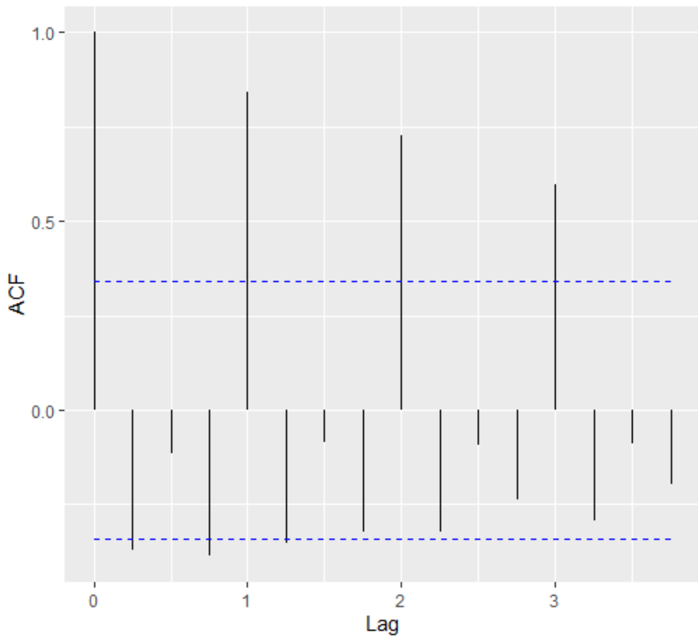

Fig. 14: Autocorrelaciones del Personal Asalariado

Como era de esperarse, podemos observar como al igual que la anterior serie los primeros picos presentan una alta autocorrelación, indicando que la serie es estacional.

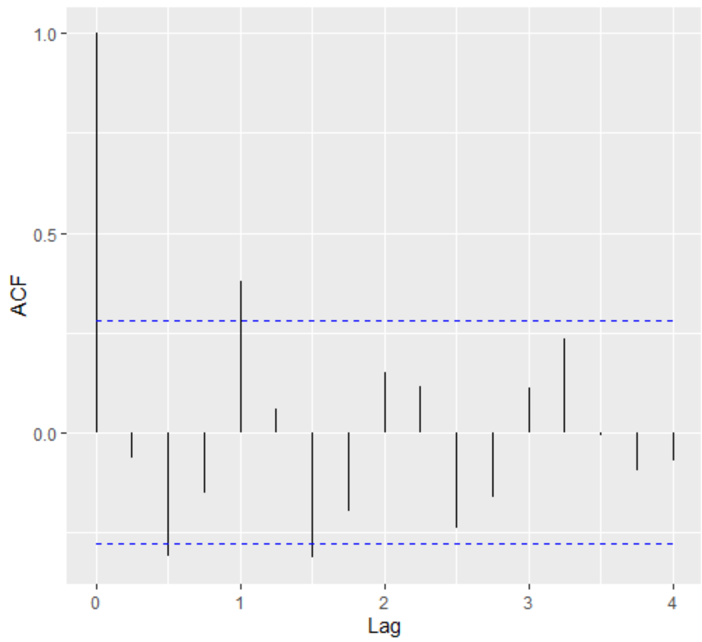

Fig. 15: Autocorrelaciones del Personal No Asalariado

En esta serie vemos como no hay suficiente autorrelación para determinar que la serie es estacional, tal como lo veniamos deduciendo anteriormente, esta serie no es estacional.

2.3.4 Diferencias con Retraso

Aplicamos un retraso de 4 para calcular las diferencias de acuerdo a la estacionalidad

Tambien realizamos el Test ADF, el cual su Hipótesis Nula es de que la serie temporal NO es estacionaria. Y el Test de KPSS nos indica lo contrario, la serie temporal es estacionaria ya que posee raiz unitaria.

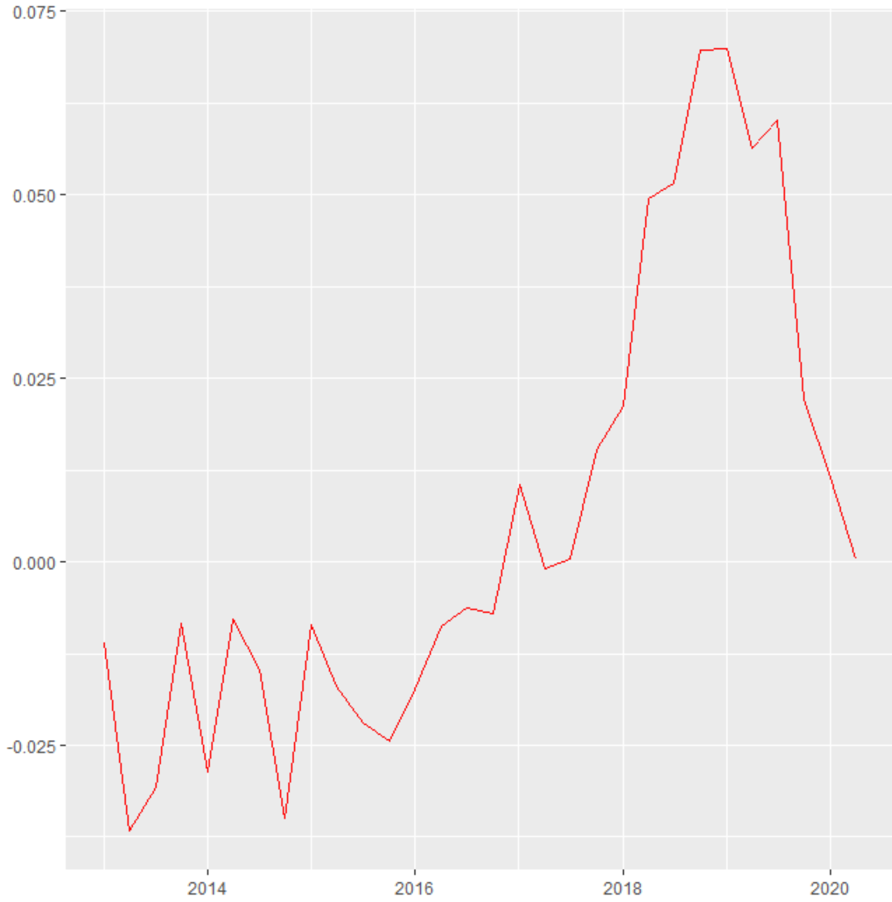

Fig. 16: Diferencias con lag 4 del Personal Ocupado

```
        Augmented Dickey-Fuller Test

data:  diff.cadena_1.ts.4
Dickey-Fuller = -3.3717, Lag order = 3, p-value = 0.08002
alternative hypothesis: stationary
```

Fig. 17: ADF Test Personal Ocupado

Este resultado nos indica que la serie posee raiz unitaria, aunque el p-value es muy cercano al limite para rechazar la hipotesis nula.

```
        KPSS Test for Level Stationarity

data:  diff.cadena_1.ts.4
KPSS Level = 0.76416, Truncation lag parameter = 2, p-value = 0.01
```

Fig. 18: KPSS Test Personal Ocupado

Se rechaza la hipotesis nula, la serie no es estacionaria.

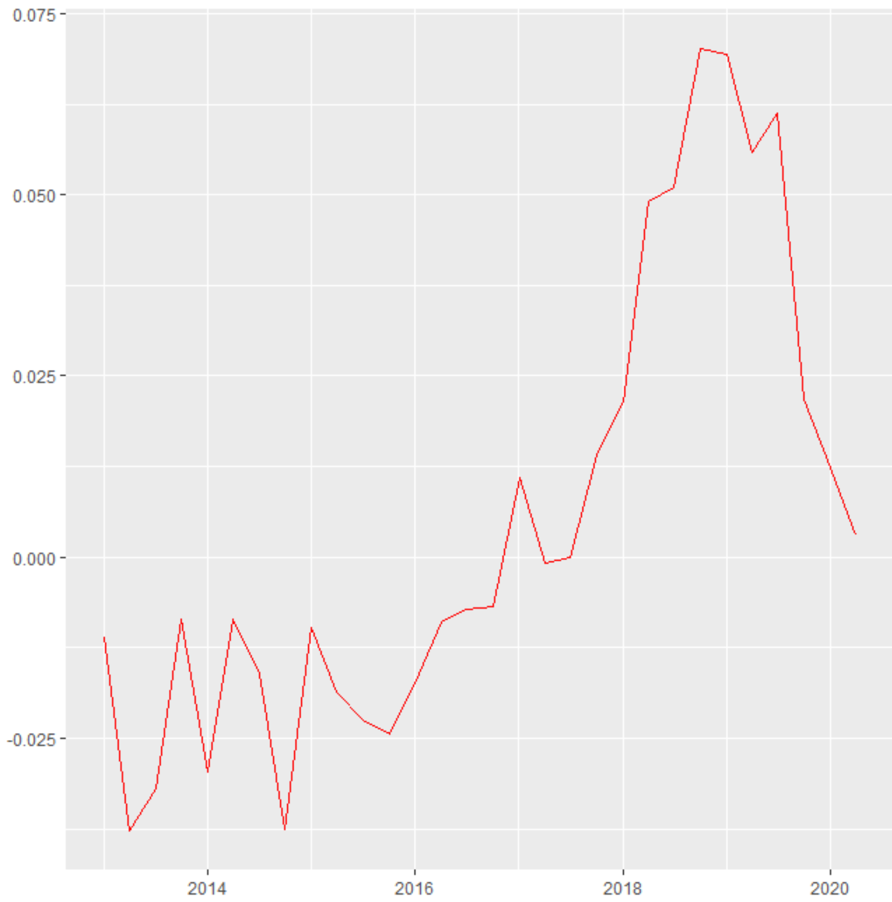

Fig. 19: Diferencias con lag 4 del Personal Asalariado

```
        Augmented Dickey-Fuller Test

data:  diff.cadena_2.ts.4
Dickey-Fuller = -3.3055, Lag order = 3, p-value = 0.08938
alternative hypothesis: stationary
```

Fig. 20: ADF Test Personal Asalariado

No se puede rechazar la hipotesis nula, al igual que en la anterior serie, nos indica que no es estacionaria.

```
        KPSS Test for Level Stationarity

data:  diff.cadena_2.ts.4
KPSS Level = 0.77889, Truncation lag parameter = 2, p-value = 0.01
```

Fig. 21: KPSS Test Personal Asalariado

Se rechaza la hipotesis nula, la serie no es estacionaria.

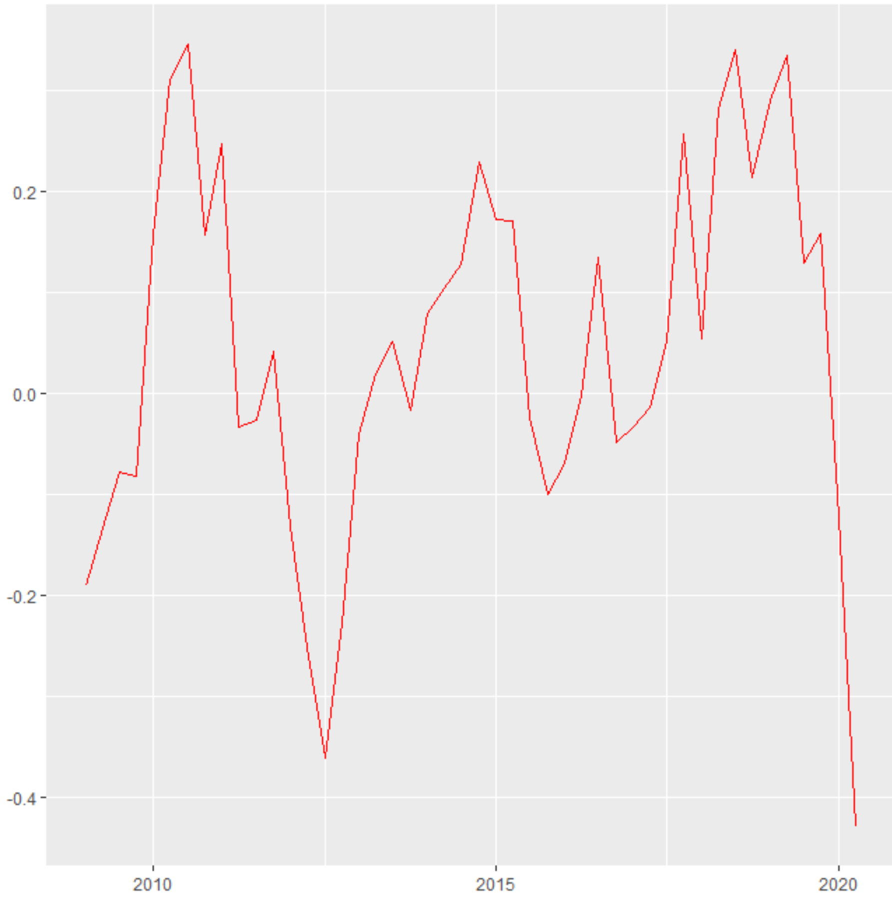

Fig. 22: Diferencias con lag 4 del Personal No Asalariado

```
        Augmented Dickey-Fuller Test

data:  diff.cadena_3.ts.4
Dickey-Fuller = -3.0296, Lag order = 3, p-value = 0.1646
alternative hypothesis: stationary
```

Fig. 23: ADF Test Personal No Asalariado

No se puede rechazar la hipotesis nula, nos indicandonos que no es estacionaria.

```
        KPSS Test for Level Stationarity

data:  diff.cadena_3.ts.4
KPSS Level = 0.13333, Truncation lag parameter = 3, p-value = 0.1
```

Fig. 24: KPSS Test Personal No Asalariado

No se puede rechazar la hipotesis nula, indicando que la serie es estacionaria alrededor de una tendencia determinista.

2.4 Creación del Modelo

2.4.1 Dividir datos

Como primer paso para la creación del modelo utilizamos la función *initial time split* para dividir nuestros datos en un subset de entrenamiento y otro para probar la calidad del modelo

2.4.2 Crear modelos

Empezamos a crear diferentes modelos, como el *Auto Arima, Arima XGBoost, ETS, Prophet, Mars y Linear Reg*

2.4.3 Crear tabla de modelos

Agrupamos todos los modelos en una tabla con la función *modeltime table* para aplicar las calibraciones y reajustes en simultáneos

2.4.4 Calibrar modelos

A partir de la función *modeltime calibrate* y la tabla de modelos realizamos la calibración de los modelos

2.4.5 Probamos los modelos

Realizamos la predicción y contrastamos con la realidad, esto gracias a la función *modeltime forecast*. Y luego graficamos los resultados de cada modelo, con su predicción.

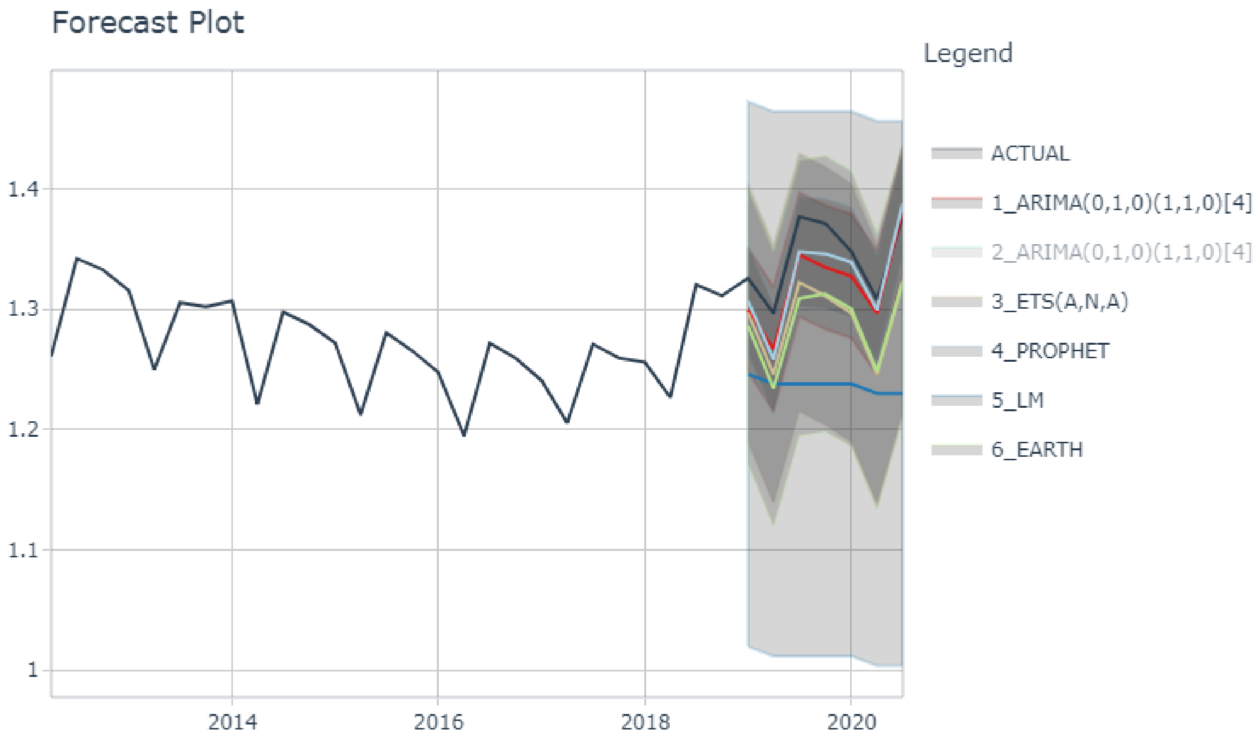

Fig. 25: Modelos de predicción

Y con la funcion *modeltime accuracy* evaluamos el porcentaje de acierto y demas medidas que nos hablan de la confiabilidad del modelos. Mostramos en forma de tabla tales medidas

Search

.model_id	.model_desc	.type	mae	mape	mase	smape	rmse	rsq
1	ARIMA(0,1,0)(1,1,0)[4]	Test	0.02	1.72	0.56	1.74	0.03	0.85
2	ARIMA(0,1,0)(1,1,0)[4]	Test	0.02	1.72	0.56	1.74	0.03	0.85
3	ETS(A,N,A)	Test	0.05	3.85	1.26	3.93	0.05	0.89
4	PROPHET	Test	0.02	1.49	0.49	1.5	0.02	0.87
5	LM	Test	0.11	7.87	2.59	8.22	0.11	0.01
6	EARTH	Test	0.06	4.13	1.35	4.22	0.06	0.92

Fig. 26: Presición de los modelos de predicción

2.4.6 Reajuste

Por ultimo hacemos un reajuste de los datos con la función *modeltime refit* y a partir de esto volvemos a realizar la prediccion del modelo

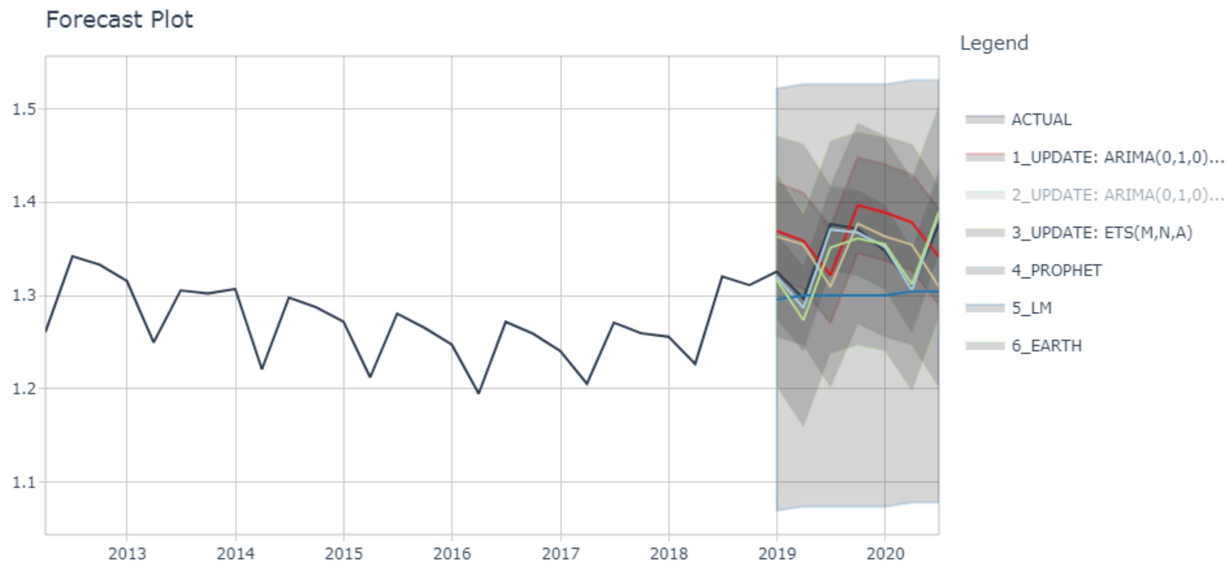

Fig. 27: Modelos de predicción reajustados

3 Conclusión

Luego de lo desarrollado podemos concluir que el mejor modelo para predecir la tasa de personal ocupado en el proximo cuardo de año es el obtenido con *Prophet*, ya que tiene una alta precision y una muy pequeña incertidumbre

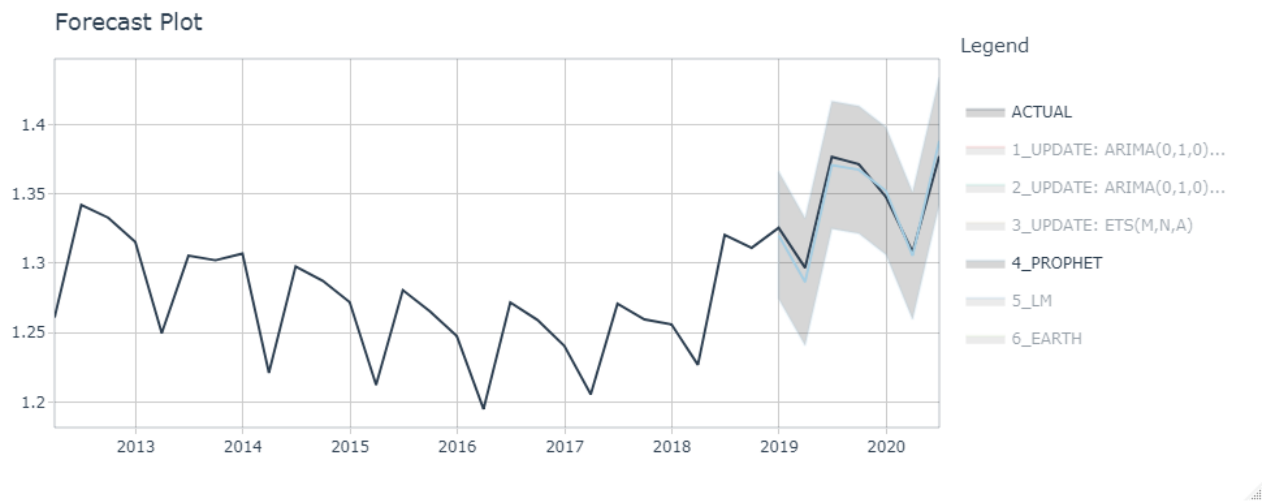

Fig. 28: Modelo predictor Prophet

4 Anexo

4.1 Código en R

```
###############
# Time Series
# Actividad 5
#
# Creator : Group B - Benitez, Garcia, Rodriguez, Rechimon
#
# Create date: 2020/10/29
# Update date: 2020/11/6
#        comment: comentarios
###############

##----------------START LIBRARIES
library(readxl)
library(ggfortify)
library(scales)
library(forecast)
library(tseries)

library(tidymodels)
library(modeltime)
library(tidyverse)
library(lubridate)
library(timetk)
library(earth)
library(zoo) # Para trabajar cuartos como date

library(rsample)
##----------------END LIBRARIES


##----------------START READ DATA
datos <- read_excel("SINF_POCUP_0810.xlsx", skip = 1)
##----------------END READ DATA


##----------------START PROCESS DATA
datos <- na.omit(datos)
head(datos, 5)

# Transformamos los datos en una serie temporal
cadenas.ts  <- ts(datos[, 2:4], start = c(2008,1), frequency = 4)

cadena_1.ts <- ts(datos[17:50, 2],   start = c(2012,1), frequency =
    4)
cadena_2.ts <- ts(datos[17:50, 3],   start = c(2012,1), frequency =
    4)
cadena_3.ts <- ts(datos[, 4],   start = c(2008,1), frequency = 4)


# Plots
```

```
# Serie Temporal de las 3 cadenas juntas y por separado
autoplot(cadenas.ts, xlab = "Periodo", ylab = "Tasa")

# Tomamos desde 2012 hasta la fecha
autoplot(cadena_1.ts, colour = "blue", linetype = "dashed")
autoplot(cadena_2.ts, colour = "blue", linetype = "dashed")

autoplot(cadena_3.ts, colour = "blue", linetype = "dashed")


# Descomponemos las series
desc.cadena_1 <- decompose(cadena_1.ts)
plot(desc.cadena_1, xlab ='Periodo')

desc.cadena_2  = decompose(cadena_2.ts)
plot(desc.cadena_2, xlab ='Periodo')

desc.cadena_3  = decompose(cadena_3.ts)
plot(desc.cadena_3, xlab ='Periodo')


# obtener la tendencia y la componente estacional
desc.cadena = decompose(cadenas.ts)

sTrend   = desc.cadena$trend
sSeasonal= desc.cadena$seasonal
ts.plot(cbind(sTrend,sTrend + sSeasonal),xlab='Tendencia + componente estacional', lty=2:1)

# Numero estimado de diferencias requeridas para realizar una ST estacionaria
# Prueba de raiz unitaria para identificar si la serie es estacionaria
ndiffs(cadena_1.ts)  ## [1] 1 desde 2008 a la fecha | 0 desde 2012 a la fecha
nsdiffs(cadena_1.ts) ## [1] 1 desde 2008 a la fecha | 1 desde 2012 a la fecha

ndiffs(cadena_2.ts)  ## [1] 1 desde 2008 a la fecha | 0 desde 2012 a la fecha
nsdiffs(cadena_2.ts) ## [1] 1 desde 2008 a la fecha | 1 desde 2012 a la fecha

ndiffs(cadena_3.ts)  ## [1] 1
nsdiffs(cadena_3.ts) ## [1] 0

# Remueve la tendencia y compara el ruido con la estacionalidad
diff.cadena_1.ts <- autoplot(diff(cadena_1.ts))
diff.cadena_1.ts
boxplot(diff(cadena_1.ts) ~ cycle(diff(cadena_1.ts)), xlab = "Periodo", ylab = "Diferencias")

diff.cadena_2.ts<-autoplot(diff(cadena_2.ts))
diff.cadena_2.ts
boxplot(diff(cadena_2.ts) ~ cycle(diff(cadena_2.ts)), xlab = "Periodo", ylab = "Diferencias")
```

```

diff.cadena_3.ts<-autoplot(diff(cadena_3.ts))
diff.cadena_3.ts
boxplot(diff(cadena_3.ts) ~ cycle(diff(cadena_3.ts)), xlab = "
    Periodo", ylab = "Diferencias")

# Autocorrelation Function:
# Visualizamos que tan correlacionado se encuentra el  ltimo  valor
     con el pasado (LAG)
# Si la serie es estacional, la autocorrelaci n sera muy alta para
     retrasos multiplos
# de la frecuencia
autoplot(acf(cadena_1.ts, plot = FALSE))
autoplot(acf(diff(cadena_1.ts), plot = FALSE))

autoplot(acf(cadena_2.ts, plot = FALSE))
autoplot(acf(diff(cadena_2.ts), plot = FALSE))

autoplot(acf(cadena_3.ts, plot = FALSE))
autoplot(acf(diff(cadena_3.ts), plot = FALSE))

# Partial ACF
# .Permite visualizar cuando determinados retrasos (LAG) son buenos
     para el modelo,
#  tiles  para data estacional
# .Remueve la dependencia de lags en otros lags utilizando la
    correlaci n de los
# residuos

autoplot(acf(cadena_1.ts, plot = FALSE))
autoplot(pacf(diff(cadena_1.ts), plot = FALSE))
autoplot(acf(cadena_2.ts, plot = FALSE))
autoplot(pacf(diff(cadena_2.ts), plot = FALSE))
autoplot(acf(cadena_3.ts, plot = FALSE))
autoplot(pacf(diff(cadena_3.ts), plot = FALSE))

# Se agrega Lag determinado por el ACF para ver nuevamente la serie
     sin tendencia
diff.cadena_1.ts.4<-diff(cadena_1.ts, lag = 4)
autoplot(diff.cadena_1.ts.4, ts.colour = "red")
##
sd(diff(cadena_1.ts, lag = 4))
summary(diff(cadena_1.ts, lag = 4))
##

diff.cadena_2.ts.4<-diff(cadena_2.ts, lag = 4)
autoplot(diff.cadena_2.ts.4, ts.colour = "red")

diff.cadena_3.ts.4<-diff(cadena_3.ts, lag = 4)
autoplot(diff.cadena_3.ts.4, ts.colour = "red")


##----------------START TESTS TS
# Test ADF
# Ho: Existe raiz unitaria, no es estacionaria
# Ha: No existe raiz unitaria, es estacionaria
```

```
adf.test(diff.cadena_1.ts.4)
## 0.08002   No se puede rechazar la hipotesis nula

adf.test(diff.cadena_2.ts.4)
## 0.08938   No se puede rechazar la hipotesis nula

adf.test(diff.cadena_3.ts.4)
## 0.1646    No se puede rechazar la hipotesis nula.

# Test KPSS
# Ho: Es estacionaria alrededor de una tendencia determinista
# Ha: Existe raiz unitaria, no es estacionaria
kpss1<-kpss.test(diff.cadena_1.ts.4)
kpss1$p.value
# 0.01    Se rechaza la hipotesis nula

kpss2<-kpss.test(diff.cadena_2.ts.4)
kpss2$p.value
# 0.01    Se rechaza la hipotesis nula

kpss3<-kpss.test(diff.cadena_3.ts.4)
kpss3$p.value
## 0.1    Se acepta la hipotesis nula
##------------------END TESTS TS
##------------------END PROCESS DATA


##------------------START ARIMA MODEL
##-- CREATE MODEL
# (X,X,X) Parte no Estacional (X,X,X) Parte estacional del modelo [
    X] Frecuencia
modelArima <- auto.arima(cadena_1.ts, stepwise = FALSE,
    approximation = FALSE)

modelArima
# ARIMA(1,0,0)(2,1,0)[4] con tendencia, y que arroja un AIC=-160.39

##-- TEST MODEL
# Ho: Los datos se distribuyen de forma independiente, las
    correlaciones
# en la poblaci n de la que se toma la muestra son 0.
# Ha: Los datos no se distribuyen de forma independiente.
# Test de Box-Pierce
Box.test(modelArima$residuals) # 0.966   Ho

# Test de Ljung-Box
Box.test(modelArima$residuals, type="Ljung-Box") # 0.9645  Ho

# Ho: Hay distribucion normal en los residuos,  la asimetr a y el
    exceso de curtosis
# son nulos (asimetr a = 0 y curtosis = 3)
# Test de Jarque-Bera
jarque.bera.test(modelArima$residuals) # 0.8863 Ho

# Ho: La distribuci n de residuos no es normal.
# Test de Shapiro-Wilk
```

```
shapiro.test(modelArima$residuals) # 0.5031 Ho

##-- MAKE PREDICTION
forecast <- forecast(modelArima, level = c(95), h = 4)
autoplot(forecast)
ggtsdiag(modelArima)
autoplot(acf(forecast$residuals, plot = FALSE))
autoplot(pacf(forecast$residuals, plot = FALSE))

forecast$method
forecast$fitted
forecast$residuals

y_cadena_1 <- as.data.frame(forecast$mean)
y_cadena_1$x
##-----------------END ARIMA MODEL


##----------------START MODELS
# Modifico los nombres para trabajar con el script facilmente
colnames(datos) <- c("date", "value", "value2", "value3")

data <- datos[1:2]
data <- data[17:50,]
# Para utilizar graficos dinamicos de plotly en ggplot
interactive <- TRUE

# Trabajar los periodos como date
# data$date <- as.yearqtr(data$date, format = "Q%q/%y")
data$date <- as.Date(as.yearqtr(data$date, format = "Q%q/%y"), frac
    = 1)
class(data$date)

# Visualizamos la serie temporal de forma dinamica
data %>%
  plot_time_series(date, value, .interactive = interactive)

data %>%
  tk_tbl(rename_index = 2008) %>%
  plot_acf_diagnostics(
    .date_var = date,
    .value    = value,
    .lags     = 20,
    .show_white_noise_bars = TRUE,
    .interactive = FALSE
  )

##-- STEP 1: SPLIT DATA
splits <- initial_time_split(data, prop = 0.8)

##-- STEP 2: BUILD MODELS
# 1- ARIMA REG
model.arimaReg <- arima_reg() %>%
  set_engine(engine = "auto_arima") %>%
  fit(value ~ date, data = training(splits))
model.arimaReg
```

```
# ARIMA(0,1,0)(1,1,0)[4]  AIC=-123.45   AICc=-122.82   BIC=-121.27

# 2- ARIMA BOOST
model.arimaBoost <- arima_boost(
  min_n = 2,
  learn_rate = 0.015
) %>%
  set_engine(engine = "auto_arima_xgboost") %>%
  fit(value ~ date , data = training(splits))
model.arimaBoost
# ARIMA(0,1,0)(1,1,0)[4] AIC=-123.45   AICc=-122.82   BIC=-121.27

# 3- ETS
model.ets <- exp_smoothing() %>%
  set_engine(engine = "ets") %>%
  fit(value ~ date, data = training(splits))
model.ets
# AIC -133.7518  AICc -127.8571   BIC -124.6810

# 4- PROPHET
model.prophet <- prophet_reg() %>%
  set_engine(engine = "prophet") %>%
  fit(value ~ date, data = training(splits))

# 5- LINEAL REGRESSION
model.lm <- linear_reg() %>%
  set_engine("lm") %>%
  fit(value ~ year(date), data = training(splits))

# 6- MARS
model_spec_mars <- mars(mode = "regression") %>%
  set_engine("earth")
recipe_spec <- recipe(value ~ date, data = training(splits)) %>%
  step_date(date, features = "month", ordinal = FALSE) %>%
  step_mutate(date_num = as.numeric(date)) %>%
  step_normalize(date_num) %>%
  step_rm(date)

model.mars <- workflow() %>%
  add_recipe(recipe_spec) %>%
  add_model(model_spec_mars) %>%
  fit(training(splits))

##-- STEP 3: CREATE MODEL TABLE
models_tbl <- modeltime_table(
  model.arimaReg,
  model.arimaBoost,
  model.ets,
  model.prophet,
  model.lm,
  model.mars
)
models_tbl

##-- STEP 4: CALIBRATION
```

```
# Calibramos para determinar los intervalos de confianza y metricas de
# precision. Estos son predicciones y residuales que se calculan a partir de los
# datos de testing
calibration_tbl <- models_tbl %>%
  modeltime_calibrate(new_data = testing(splits))

calibration_tbl

##-- STEP 5: TEST MODELS
# Visualizamos: Forecast vs Test Dataset
# Evaluamos la prueba. Nivel de precisi n
calibration_tbl %>%
  modeltime_forecast(
    new_data = testing(splits),
    actual_data = data
  ) %>%
  plot_modeltime_forecast(
    .interactive = interactive
  )

# Probar modeltime_accuracy para recolectar metricas comunes
calibration_tbl %>%
  modeltime_accuracy() %>%
  table_modeltime_accuracy(
    .interactive = interactive
  )

# MAE Media de error absouto
# MAPE Porcentaje de Media de error absoluto
# MASE Media de error absoluto a escala
# SMAPE Media de error absoluto simetrica
# RMSE Raiz media de error cuadratico
# RSQ error cuadratico


# STEP 6: REFIT WITH FULL DATA
# Paso 6: Reajustar al dataset completo y preveer el alcance
refit_tbl <- calibration_tbl %>%
  modeltime_refit(data = data)
refit_tbl %>%
  modeltime_forecast(actual_data = data) %>%
  plot_modeltime_forecast(
    .legend_max_width = 25,
    .interactive = interactive
  )
##----------------END MODELS
```