Series temporales y minería de flujo de datos: Trabajo autónomo I - Series Temporales (Parte práctica)

Máster en ciencia de datos e ingeniería de computadores - UGR

10-4-2018

Mª Cristina Heredia Gómez

crstnheredia@correo.ugr.es

Índice

Predicción a escala diaria

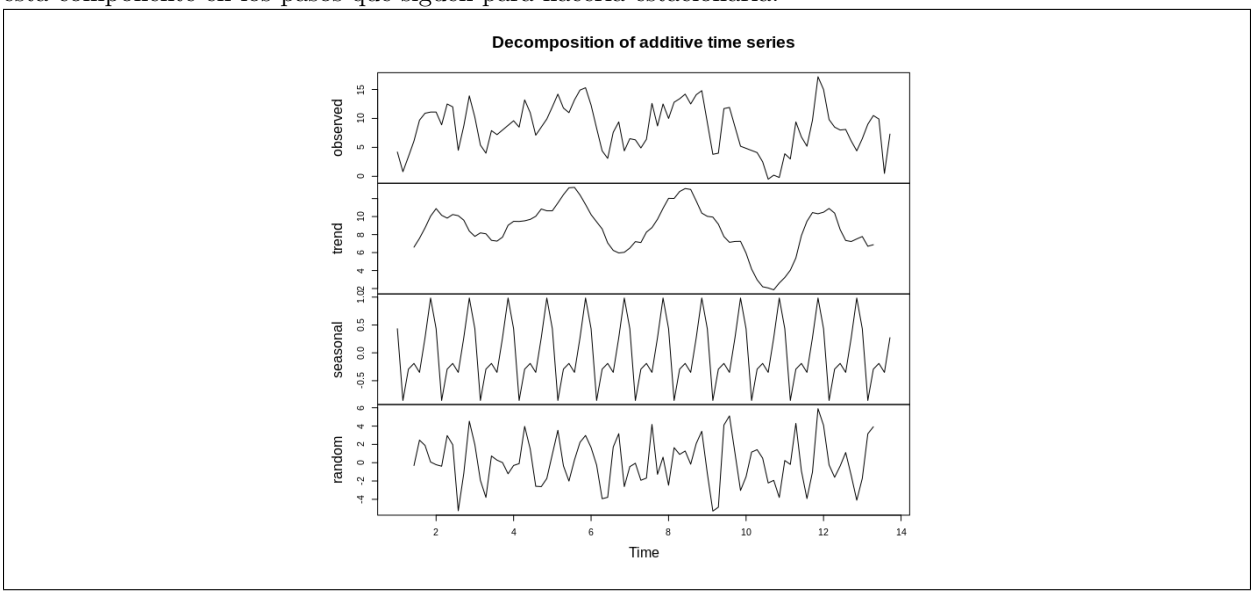
La estación meteorológica escogida es la de Morella - Castellón (9562X.csv) que contiene datos diarios desde el 2016-09-07 hasta el 2018-02-28, en total 537 filas. La razón es porque es uno de los datasets que presentan menos valores perdidos, dado que solo contiene 37 valores perdidos en total, frente a otras estaciones que contienen muchos más. En concreto, en la columna de nuestro interés (Tmax) hay solo 5 valores perdidos. En las subsecciones que siguen se detallan las etapas desarrolladas para análisis y modelado de la serie para predicción a escala diaria.

Preprocesamiento

Tras cargar los datos, descartamos la columna ID así como otras columnas necesarias para el análisis, y filtramos solamente los datos de los últimos 3 meses (desde Diciembre de 2017), pues se estima que para hacer una predicción a nivel de semana no hace falta tener en cuenta todo el histórico de datos desde 2016. Tras esto, se busca una solución a los valores perdidos. Se descarta inicialmente usar el paquete **mtsdi** para imputación en series temporales multivariantes porque los datos son anormales. Finalmente se opta por imputar los valores perdidos usando la función **na.interpolation** del paquete imputeTS, que calcula los valores perdidos interpolando a partir de los valores conocidos.

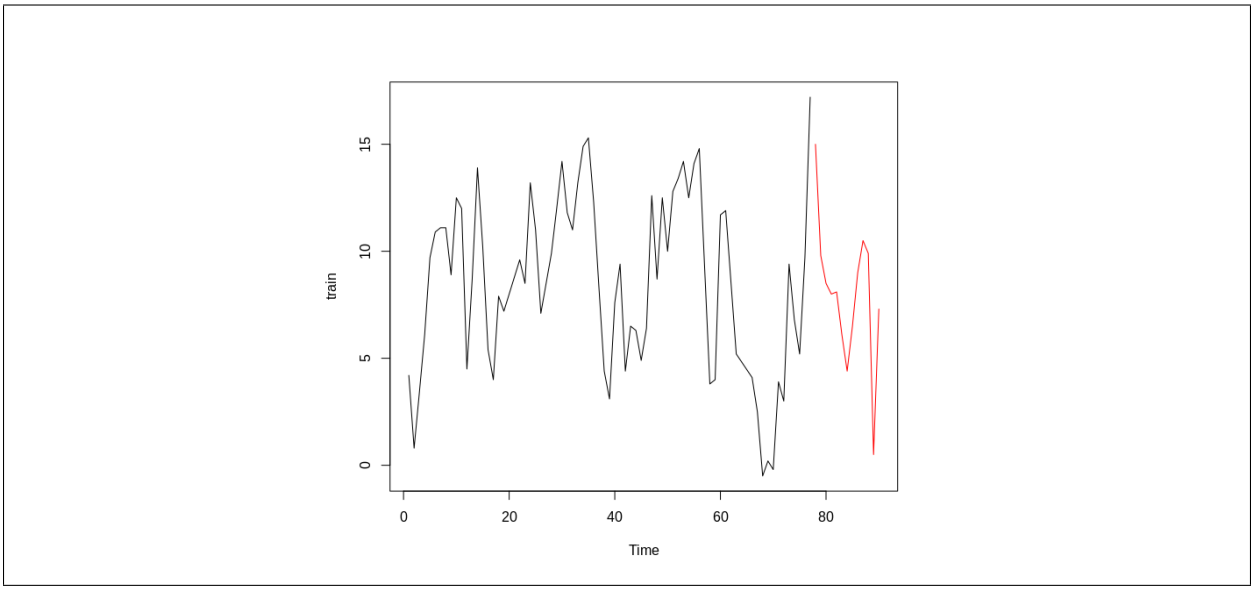
Análisis de la serie

Tras el preprocesamiento y construcción de la serie temporal mediante la función **ts** con frecuencia 7, se descompone la serie mediante la función decompose, y se observa que la serie no presenta una tendencia clara, pero que aparentemente sí que presenta una clara estacionalidad, por lo que se procederá a eliminar esta componente en los pasos que siguen para hacerla estacionaria.



División de los datos

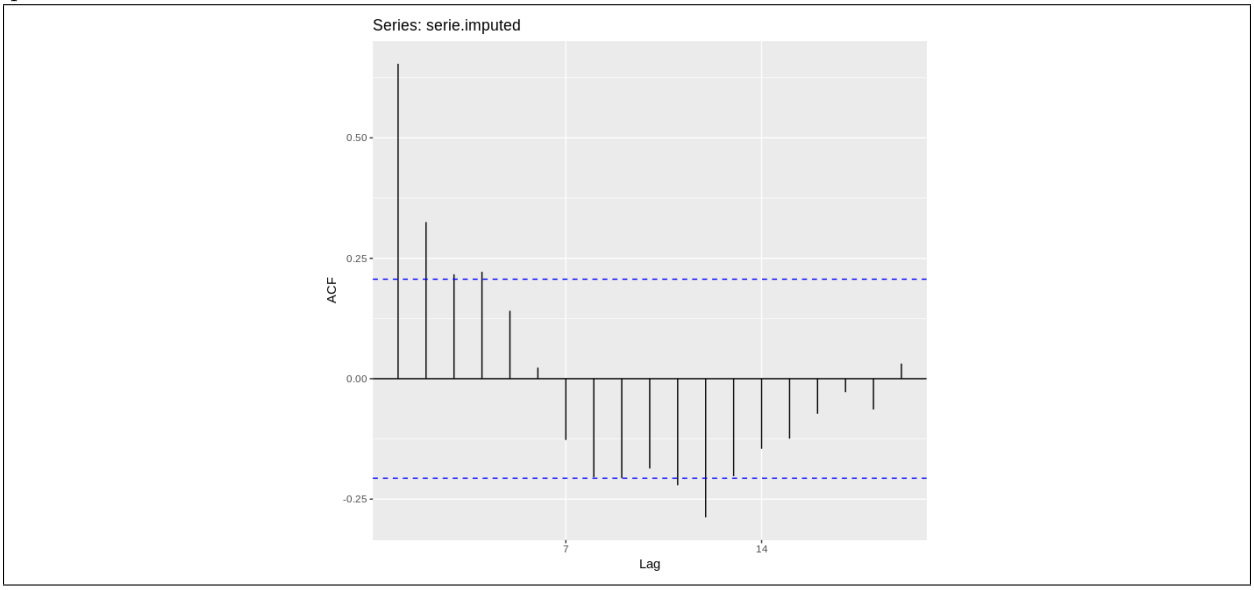
La serie total consta de 90 datos, por lo que se reservan los 13 últimos para test y el resto para train.



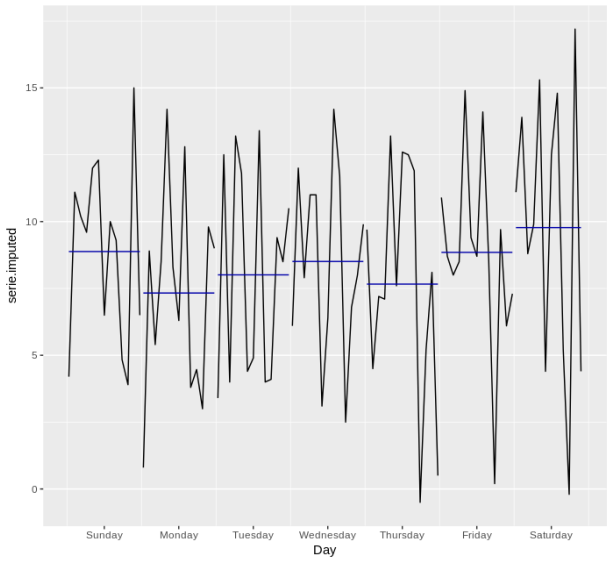
Se prueba a hacer la división de los datos a mano y usando la función **window** del paquete stats, aunque en este caso finalmente se hace de forma manual por flexibilidad.

Manejo de la estacionalidad

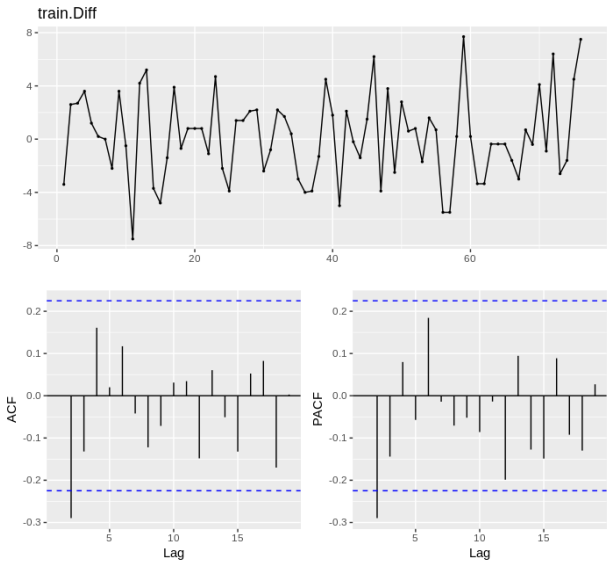
Para hacer la serie estacionaria, queremos eliminar las componentes tendencia y estacionalidad. Sin embargo, ya hemos visto que la serie no tiene tendencia, pues no se observa una tendencia clara en el gráfico. Para comprobar que existe la estacionalidad, analizamos el gráfico ACF, donde podemos ver que la serie no decrece a 0 muy rápidamente, además de que el valor de r1 es positivo y muy alto, y que presenta varios valores fuera de los límites al principio pero luego decae y se queda dentro de los límites, lo cual es indicativo de que la serie tiene estacionalidad.



Esto también se puede ver con un gráfico **ggsubseriesplot** donde vemos que las medias de cada periodo varían y por tanto indican que existe una componente estacionaria.



En lo que sigue, se supone una estacionalidad de 7 y se elimina la estacionalidad de los 7 primeros valores de la serie: del 8 al 14, del 15 al 20...etc, se comprueba si la serie es estacionaria usando la gráfica ACF y el test estadístico ADF para ello, obteniendo una gráfica ACF muy similar a la mostrada anteriormente y un p-valor de 0.3036, lo que nos indica que la serie aún tiene estacionalidad.
Como aún tiene estacionalidad, diferenciamos la serie una vez, usando la función **diff**. Ahora el ACF y PACF de la serie nos muestra que el 95 % de los r_i se mantiene dentro de los delimitadores, por lo que la serie muestra un aspecto estacionario.



Además el test de Dickey-Fuller aumentado obtiene un p-valor de 0.01, inferior a 0.05, por tanto, según el test, la serie ya es estacionaria.

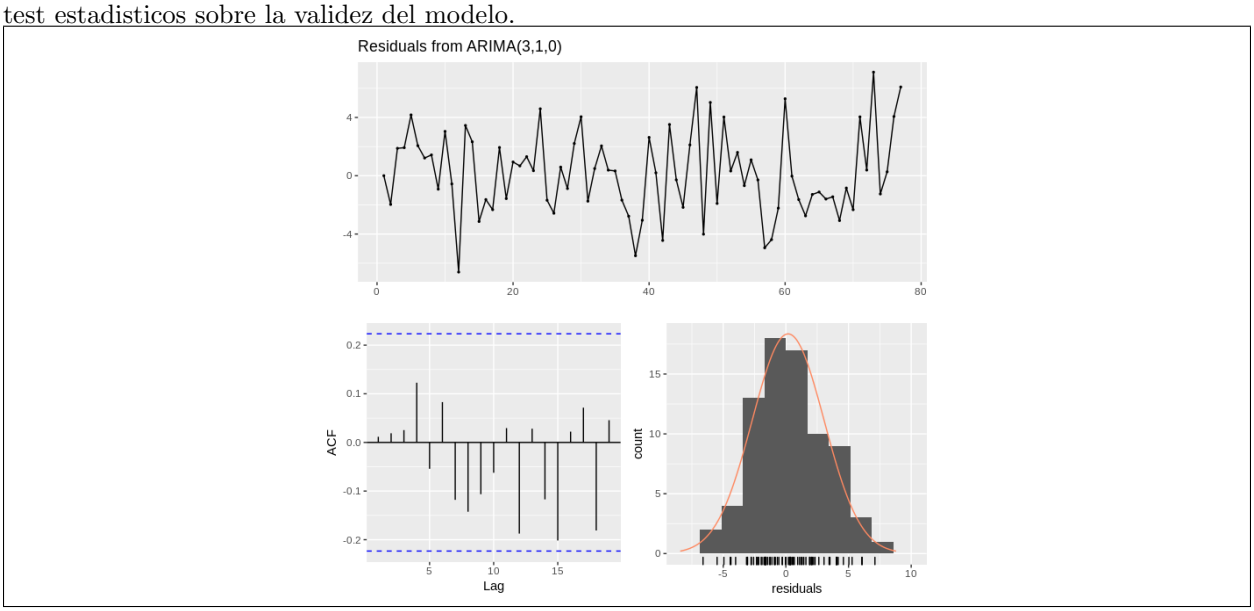
Ajuste del modelo

Se ha ajustado un modelo ARIMA(3,1,0), si bien se han probado otras variaciones sobre el mismo que han dado peores resultados: ARIMA(1,1,0), ARIMA(2,1,0) y ARIMA(4,1,0). El modelo pasa con éxito todos los test probados, en concreto el de normalidad de residuos Box-Pierce y los test de normalidad de Jarque Bera y Shapiro Test. A continuación se muestran en una tabla los valores de AIC, RMSE y MSE obtenidos por

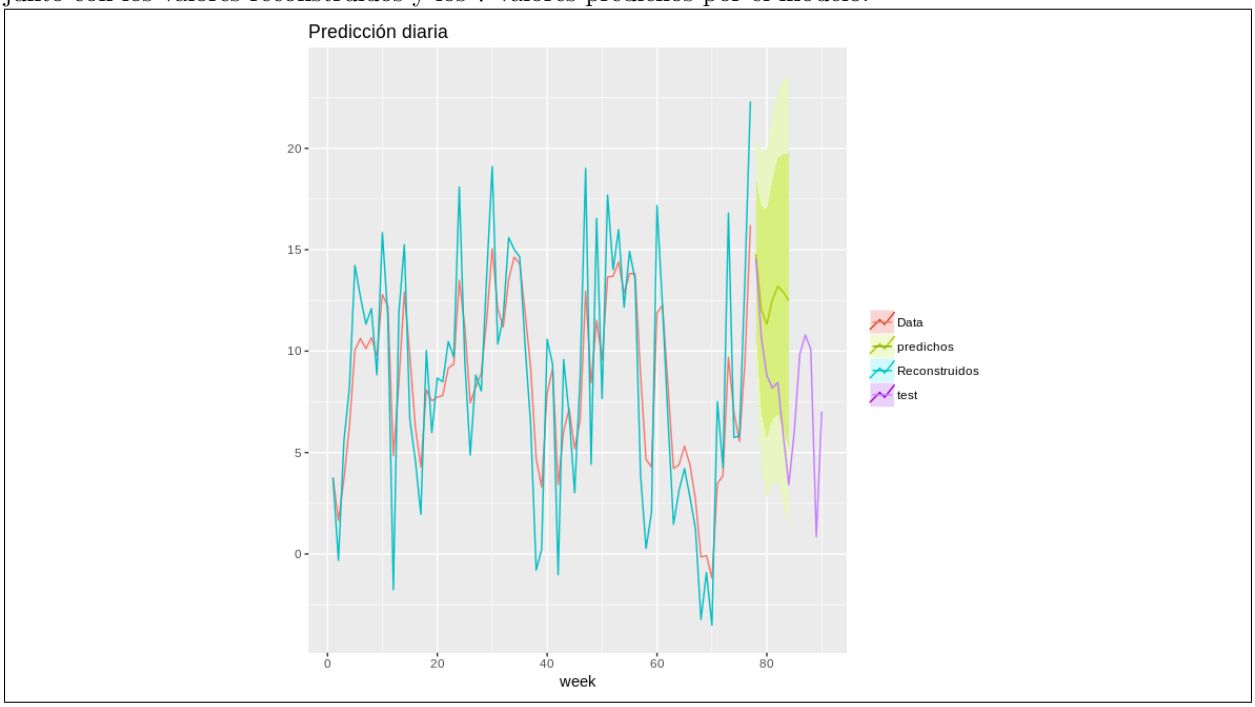
cada modelo.

	AIC	RMSE	MSE (Tr)	MSE (Ts)
ARIMA(3,1,0)	384	2.856799	628.42	411.35
ARIMA(1,1,0)	390.14	3.049332	715.97	1010.3
ARIMA(2,1,0)	385.31	2.911616	652.76	554.98
ARIMA(4,1,0)	385.93	2.844559	623.04	462.91

El ACF y la gŕafica de residuos junto con la función de densidad, sirven para contrastar el resultado de los test estadisticos sobre la validez del modelo.



Parece que, a pesar del MSE tan elevado, el modelo es válido. Por último representamos la serie sin tendencia junto con los valores reconstruidos y los 7 valores predichos por el modelo:



Donde en rojo se representan los datos de train sin tendencia, en azul los datos reconstruidos, en morado el

test sin tendencia y en verde los datos predichos. La franja con distintas tonalidades de verde indica lo que podrían ser los valores predichos, y vemos que en las 7 semanas que hemos predicho, se solapa con los datos reales de test para esa franja de tiempo, lo cual es un buen indicativo.

Predicciones reales

Una vez validado el modelo, se repiten los pasos anteriores sin dividir en conjuntos de train y test, considerando la serie completa con la imputación de valores realizada, tras lo que se deshacen los cambios y se realizan las predicciones reales para la primera semana de Marzo, obteniendo que las temperaturas de la primera semana de Marzo de 2018 en Morella (Castellón) serán: 9.940735, 8.668575, 6.909452, 6.647888, 7.270760, 8.497789, 8.980770.

Predicción a escala mensual

En este caso, la estación meteorológica escogida es la de Zumaia (Gipuzkoa), dado que es una de las base de datos mas grandes (incluye datos desde 2013 a 2018) y con menos valores perdidos. Tiene 1754 filas y 37 valores perdidos en la columna de nuestro interés Tmax.

Preprocesamiento

En este caso, el preprocesamiento consta de dos etapas: por un lado, la imputación de valores perdidos es fundamental, y por otro, la agregación de los datos diarios en datos mensuales. También se descartan todas las columnas que no son de interés para alguna de estas dos tareas, quedándonos solo con las columnas *Fecha*, ya que la vamos a necesitar para agregar por fechas y *Tmax*.

Imputación de valores perdidos

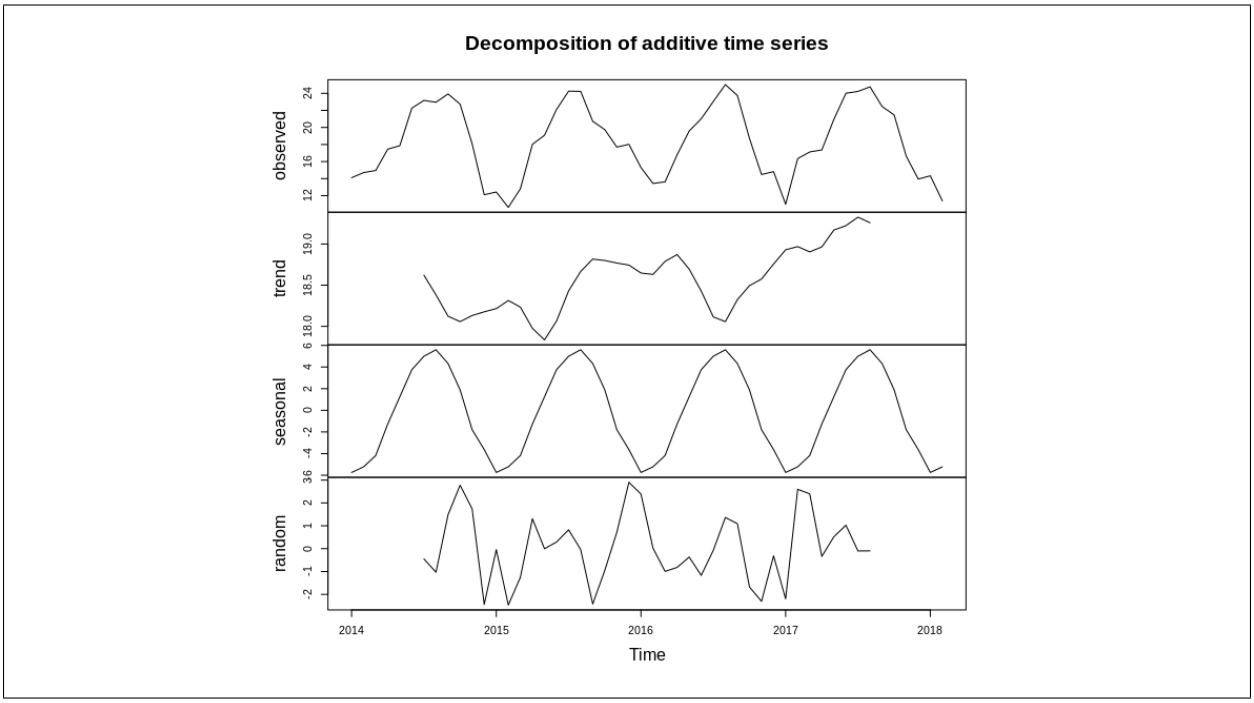
Al igual que en la predicción diaria, se ha usado la función **na.interpolation** del paquete **imputeTS** sobre la columna Tmax, que hace la imputación por interpolación de valores perdidos en series temporales univariantes.

Agregación de datos

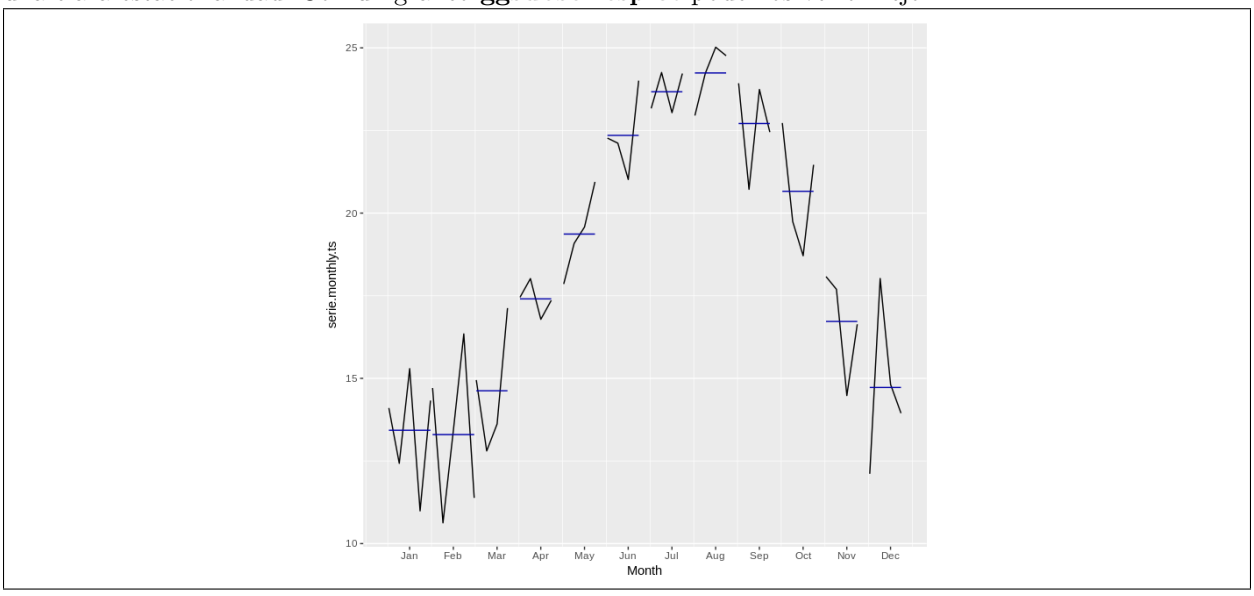
Dado que tenemos un dato diario desde el 2013-05-07 hasta el 2018-02-28 y queremos predicciones de temperatura media por mes, necesitamos agregar los datos de cada mes de cada año, tal que solo tengamos un dato por mes por año. Agregamos esos datos por la media del mes, para lo que, primero, pasamos la serie ya imputada a formato extensible con la función **xts** del paquete **xts**, y luego agregamos los datos por mes usando la función **apply.monthly** del mismo paquete, especificando que queremos agregar por la media.

Análisis de la serie

Tras el preprocesamiento, creamos una serie temporal con frecuencia 12 (frecuencia mensual). Para evitar problemas luego al eliminar la estacionalidad restando los 12 primeros valores repetidos, se descartan los datos de 2013, ya que no contienen un ciclo completo (comprenden desde mayo 2013 a diciembre 2013).



Ante la gráfica de la serie resultante, podemos ver que no se observa una tendencia clara, si bien, si se observa una clara estacionalidad. Con un gráfico **ggsubseriesplot** podemos verlo mejor:



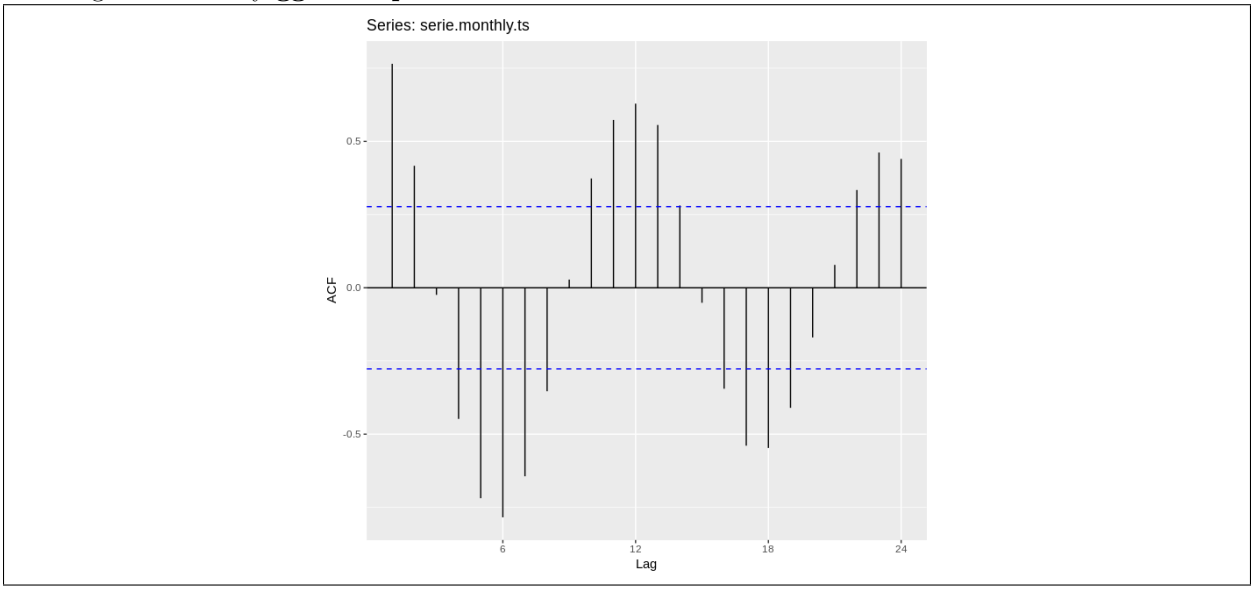
Donde observamos una gran diferencia entre las medias (barras azules) de cada una de las componentes, que además se repite en el tiempo de forma similar, lo que indica una clara estacionalidad.

División de los datos

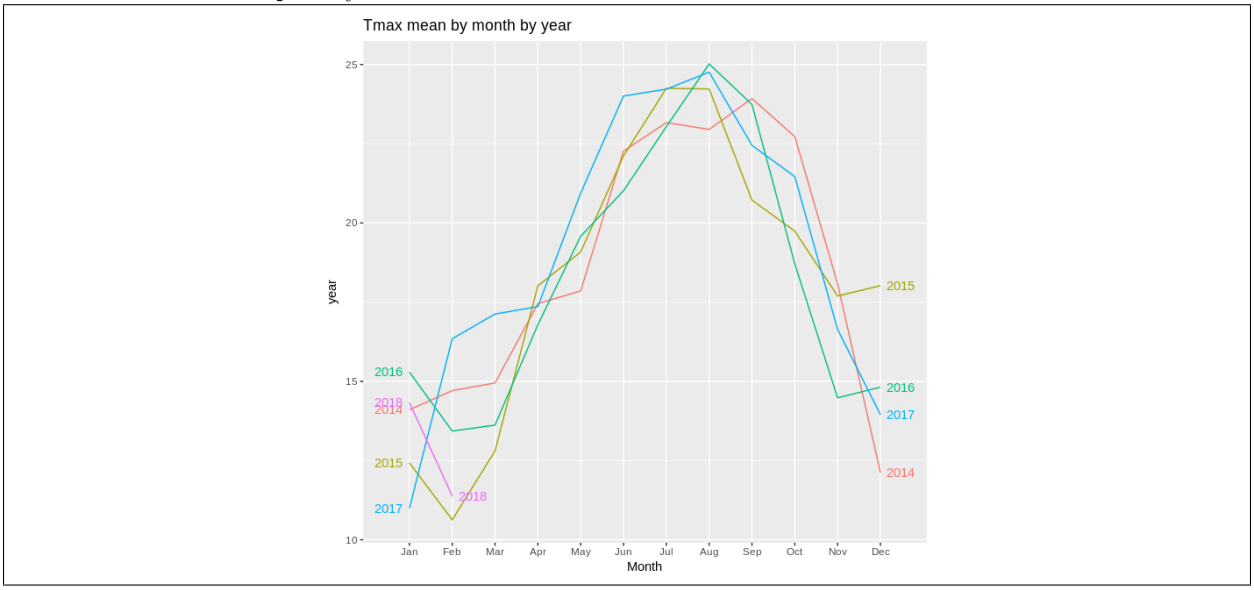
Tras agrupar, la serie total tiene 58 datos. Usando la función **window** del paquete **stats** seleccionamos para entrenamiento los datos completos desde 2014 a 2017, ambos inclusive, reservando para test los dos últimos meses de 2018 (Enero y Febrero).

Manejo de la estacionalidad

Como ya hemos mencionado, la serie no tiene tendencia, pero sí tiene estacionalidad. Lo podemos comprobar con los gráficos ACF y **ggseasonplot**.



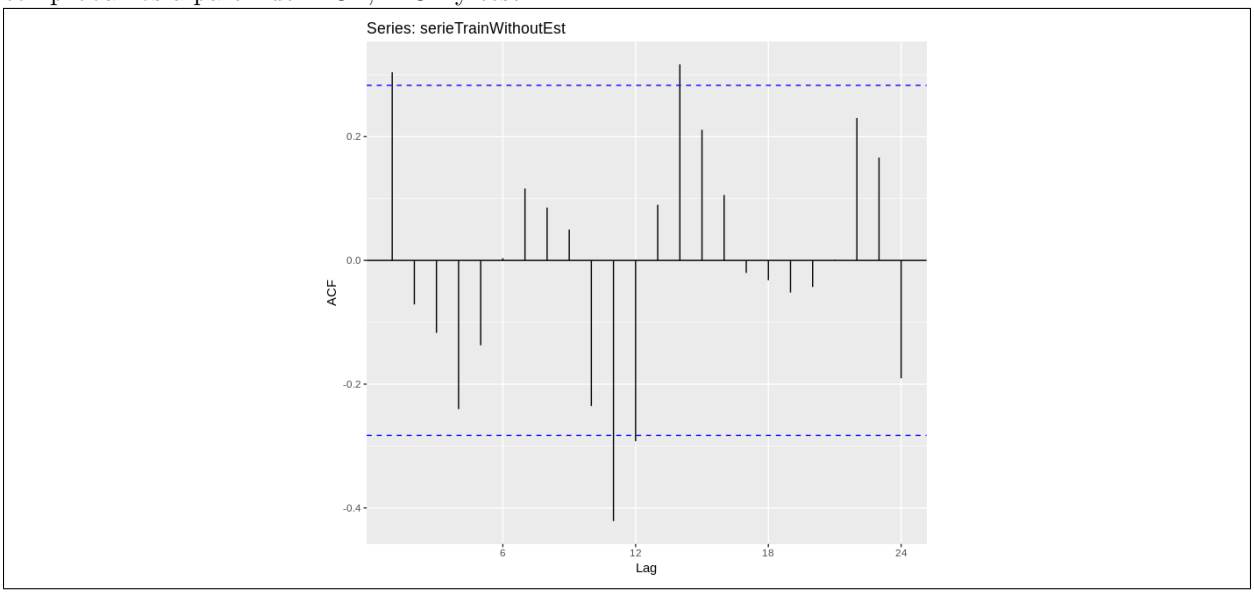
En el gráfico ACF el patrón ondulado que se repite arriba y abajo cada siete lags y las barras que sobrepasan los límites nos indican que hay estacionalidad en los datos.

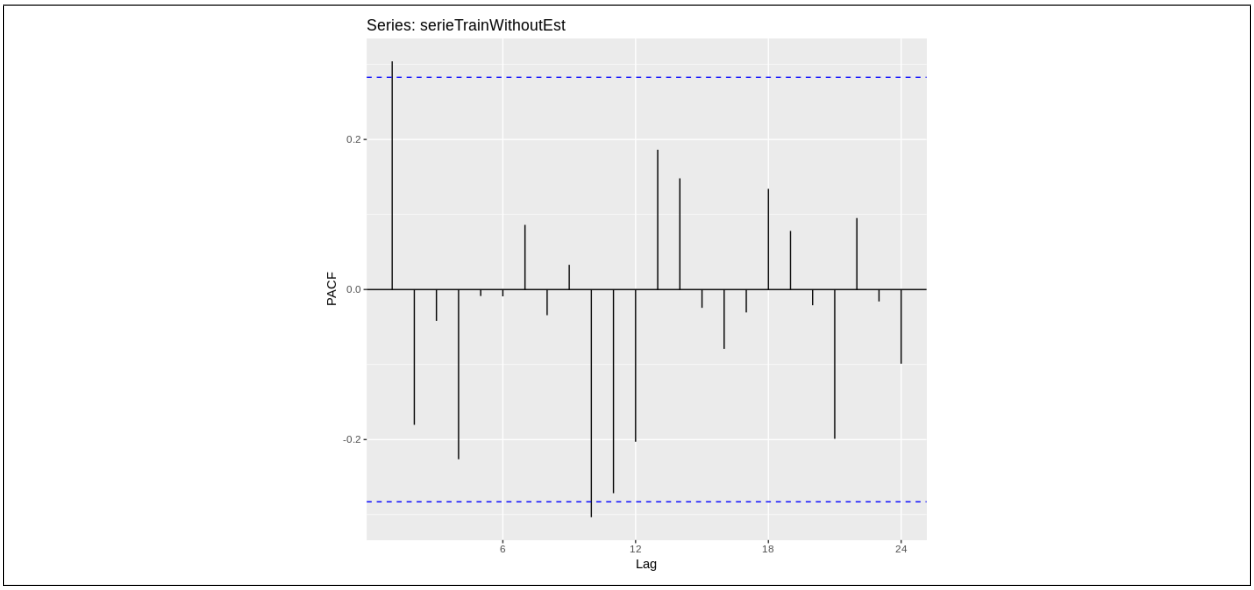


El **ggseasonplot** nos sirve pra contrastar lo que nos dice el ACF, ya que sirve para observar si se dan o no patrones estacionarios. Por ejemplo, observamos que en enero-febrero se han alcanzado las temperaturas más bajas del año en media, mientras que por lo general, en agosto se alcalzan las temperaturas más altas. Este gráfico también sirve para identificar excepciones, por ejemplo, febrero de 2017 que fue excepcionalmente cálido.

Ante una evidente estacionalidad que hemos supuesto de 12, obtenemos los 12 elementos que se repiten y se eliminan de los 12 primeros elementos se la serie de train. Como para test solo hemos dejado enero y febrero de 2018, eliminamos solo los dos primeros valores estacionales. Tras esto, comprobamos si la serie ya es estacionaria o si por el contrario necesita de más tratamiento, por ejemplo, una diferenciación. Lo

comprobamos a partir del ACF,PACF y test ADF.





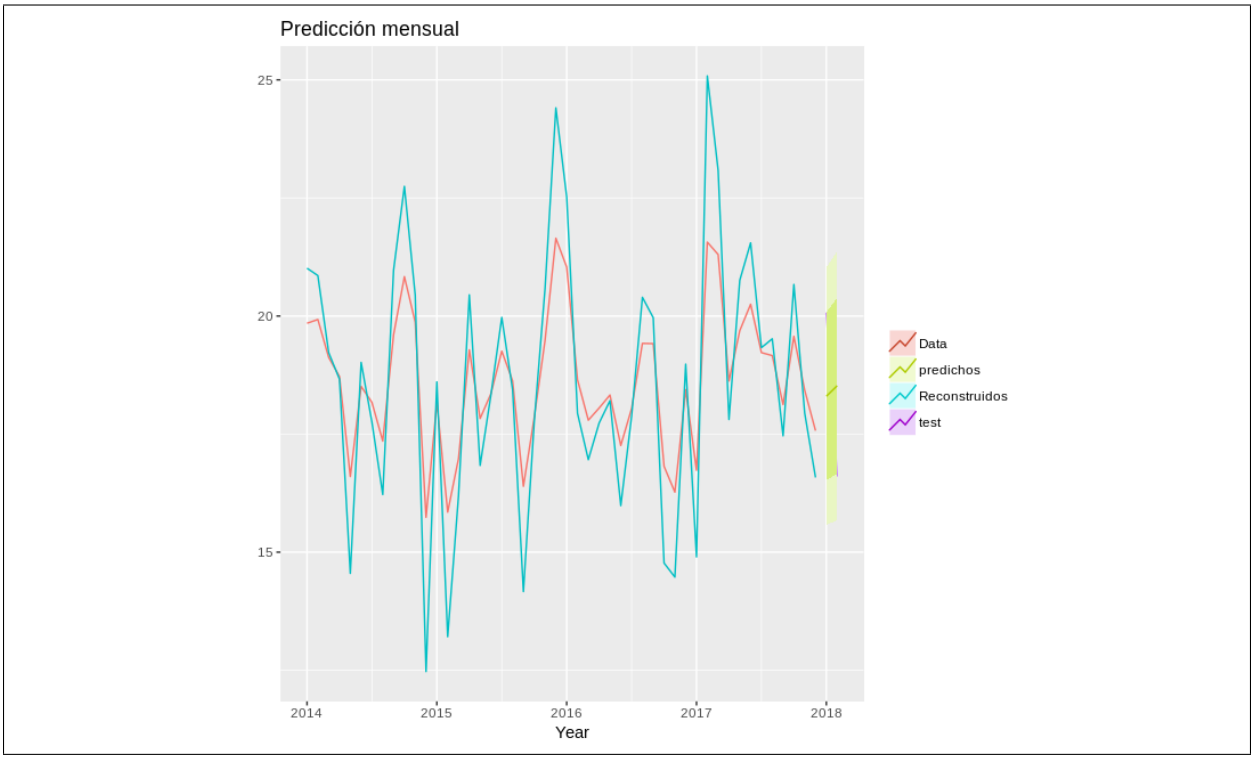
Aunque el ACF muestra que los lags 1,11 y 14 sobrepasan los límites, el resto de los valores son cercanos a 0 y el PACF es típico de una serie estacionaria, por lo que, como pasa el test de Dickey-Fuller con un p-valor de $0.01 < 0.05$, concluimos que la serie puede considerarse estacionaria y no es necesario diferenciar esta vez.

Ajuste del modelo

Ante el PACF y como no hemos diferenciado ninguna vez, podemos modelar la serie resultante como un ARIMA(1,0,0) o un ARIMA(10,0,0). Probamos los dos, obteniendo las siguientes medidas de error:

	AIC	RMSE	MSE (Tr)	MSE (Ts)
ARIMA(1,0,0)	171.79	1.359297	88.68	6.82
ARIMA(10,0,0)	180.18	1.210454	70.32	9.18

Decidiéndonos por el ARIMA(1,0,0), que parece dar mejores resultados en AIC y MSE que el ARIMA(10,0,0).



Al graficarlo, vemos que los valores predichos (franja verde) caen donde están los reales de test (franja morada), que al caer en el mismo sitio quedan debajo. Para validar el modelo, aplicamos el test de Box-Pierce o Ljung–Box que evalúa si los residuos son ruido blanco, es decir, si los errores son aleatorios. El modelo pasa el test con un p-valor de 0.6957, claramente superior a 0.05. Para comprobar si los errores son normales, validamos con los test de Jarque Bera y Shapiro, obteniendo un p-valor de p-value = 0.6957 en el primero y 0.9861 en el segundo, ambos > 0.05, por tanto el modelo queda validado.

Predicciones reales

Una vez que hemos validado todo el modelo, volvemos a seguir los pasos iniciales, sin dividir la serie en conjuntos de train y test, usando la serie con los valores perdidos imputados y la agrupación de datos por mes por año. Con esto, ya podemos hacer la predicción de las temperaturas medias de marzo y abril de 2018, obteniendo que la temperatura media en estos meses para 2018 será de 12,35° para Marzo y 13,26° para Abril.