Análisis temporal de temperaturas: predicciones y geovisualización de observatorios de la NASA

Héctor González Milà
Septiembre 2023

ABSTRACT

En este artículo se presenta un código python para analizar online, como serie temporal, un perfil de temperaturas de un observatorio cualquiera de la NASA. De aquí se realiza una previsión de temperaturas a 2 años mediante un ajuste a un modelo SARIMA. También se visualiza en un mapa tanto el histórico de temperatura como la previsión de la misma.

1. INTRODUCCIÓN

En los tiempos actuales en los que el calentamiento global es cada vez más evidente, es necesario disponer de herramientas para el seguimiento de los datos históricos de temperatura, así como utilizar herramientas de predicción para saber hacia donde nos dirigimos, y poder así tomar decisiones acordes con conocimiento pasado y predicción futura.

Las series temporales son observaciones registradas a intervalos regulares. Así es como interpretaremos los datos en origen. Es un paso preparatorio para desarrollar una predicción. Una serie temporal se puede descomponer en varios componentes: nivel base + tendencia + estacionalidad + error [1][2].

El modelo ARIMA (Autoregressive Integrated Moving Average) es un algoritmo de predicción basado en la idea de que la información en los valores pasados de la serie temporal pueden ser usados para predecir los valores futuros. Además si la series temporal tiene patrones estacionales, entonces hay que añadir términos estacionales, lo que convierte el modelo en SARIMA (Seasonal Autoregressive Integrated Moving Average). Es por ello que SARIMA parece ser el candidato ideal para ayudarnos a obtener dicha predicción [3][4][5][6]

El modelo ARIMA se compone de tres componentes principales: AR (Autoregressive), I (Integrated), y MA (Moving Average), representados por las letras p, d y q respectivamente. La parte no estacional de ARIMA se compone de estos términos, y se denota como ARIMA(p, d, q).

En el caso de SARIMA, se agrega una dimensión estacional, lo que da lugar a la notación SARIMA(p, d, q)x(P, D, Q)m. Esto es: dos modelos ARIMA simultáneamente, uno para la parte estacional y otro para la no estacional. Esto resulta en dos conjuntos de órdenes: uno para los términos AR, I y MA no estacionales, y otro conjunto que añade una nueva orden (S) que representa la longitud del ciclo estacional. Esta inclusión aporta robustez al modelo. Los términos AR, I y MA siguen desempeñando su papel en SARIMA, y se suman los términos adicionales para la estacionalidad, representados por P, D y Q.

2. ESTADO DEL ARTE

Actualmente hay un pequeño vacío respecto al hecho de que no se han combinado la recolección y visualización de datos históricos de temperatura (mediante herramientas que lo permitan accediendo directamente a bases de datos, como webscraping), el uso de Machine Learning (ML) para la obtención de predicciones futuras y la geovisualización de datos. Lo hemos podido encontrar por separado, pero en este proyecto pretendemos combinarlo todo para hacer nuestra propia aplicación.

Se han dado casos sin geovisualización [7] [8] [9]. Se han dado casos sin ML [10] [11]. Ha habido intentos con pluviometría y humedad del suelo [12] [13], con fuegos forestales [14], e incluso con temperatura trabajados en local, sin una aplicación que conecte a una base de datos [15], sin embargo en ningún caso se ha combinado todo lo mencionado anteriormente.

3. METODOLOGÍA

En primer lugar se utiliza web scraping con las librerias requests y BeautifulSoup para acceder al repositorio de observatorios de la NASA [16] y, de uno de ellos, obtener el histórico de temperatura como dataframe (df_T) y la latitud y longitud del observatorio para la posterior geolocalización en el mapa. Pandas y re han sido también importados para este propósito.

El preprocesado incluye eliminar las columnas correspondientes a las medias de temperatura trimestrales y la media anual. Los datos de temperatura son mensuales. A continuación se transforma el dataframe en una serie temporal con las fechas como columna índice y una sola columna con las temperaturas (new_df_T). Allá donde no existen datos se imputa un valor basado en el mismo valor de 12 meses antes (ya que existe un patrón anual).

Las temperaturas vienen dadas en grados centígrados. Se incluye otra columna con la temperatura en grados Kelvin para el posterior modelado. Los grados centígrados serán para las visualizaciones. También se incluyen columnas para el año y el mes (como número) para la posterior visualización en el mapa. Se han importado numpy y datetime.

Se han realizado visualizaciones de toda la serie temporal como lineplot, y boxplots para identificar outliers si los hubiera. Para las diversas visualizaciones se han utilizado matplotlib y seaborn.

El análisis de la serie temporal es con el paquete seasonal_decompose (STL) de statsmodels para descomponerla en: tendencia, estacionalidad y residuales. Para comprobar la estacionalidad desde un punto de vista estadístico se usa el Augmented Dickey_Fuller test, importando adfuller de statsmodels.

De statmodels se importa lowess para poder hacer el suavizado de la serie temporal. Nos inclinamos por un suavizado tipo “moving average” de 3 meses, con el que acabamos el preprocesado. El dataframe resultante (df_ma_3) es el utilizado para el modelado.

Para el modelado primero utilizamos la librería TimeSeriesSplit de sklearn para dividir la serie temporal en 5 splits (también llamados folds). Se asume que el modelo SARIMA es un buen modelo para este tipo de dataset.

Realizamos el modelado con auto arima, que encontrará los parámetros (p, d, q, y P, D, Q) que mejor se ajusten. Hay que importar pmdarima y ARIMA de statsmodels.

Para validar el modelo, se comprueban los residuales, el histograma de densidad, los quantiles y el correlograma con plot_diagnostics. También hacemos backtesting del modelo obtenido con auto arima con los folds obtenidos con TimeSeriesSplit. Se registran los MSE (mean squared error) en cada fold y se comprueba que no sean muy grandes para poder ser validados.

Los resultados de temperatura a 2 años vista se guardan en un dataframe (future_data) que tendrá el mismo formato y las mismas columnas que el histórico de temperatura (new_df_T). Esto es para poder concatenarlos y usarlos para el siguiente paso, la geovisualización.

Para mostrar los datos en un mapa, importamos folium para el mapa, y widgets (de ipywidgets) y display (de Ipython) para los sliders que se usarán para escoger mes y año del dato que queremos ver representado en el mapa.

4. RESULTADOS

En nuestro caso hemos utilizado el perfil de temperatura del observatorio del Ebro en Tortosa [17]. El código, eso sí, funcionará con el link de cualquier otro observatorio obtenido de este reservorio de la NASA. Antes del preprocesado la serie tiene este aspecto.

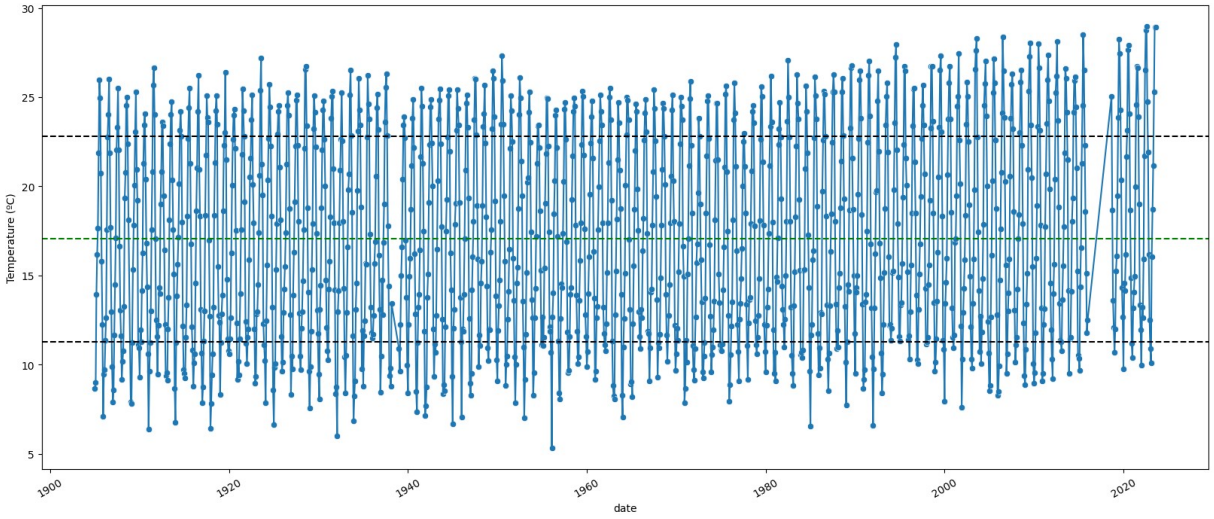

Fig.1. Datos experimentales de temperatura (ºC) del observatorio del Ebro (Tortosa).

A continuación se muestra la descomposición STL (multiplicativa), en tendencia, estacionalidad y ruido. La gráfica TEMP_K es la serie temporal ya preprocesada.

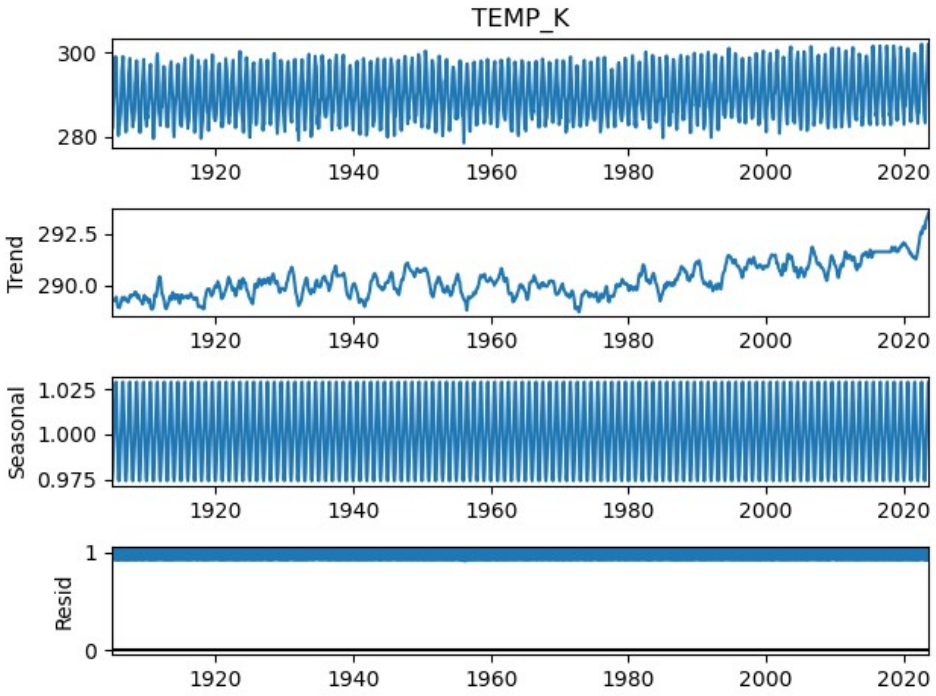

Fig. 2. Descomposición STL multiplicativa de la serie temporal. Nótese la T en ºK.

Podemos observar que la estacionalidad es obvia, y que la tendencia va en aumento desde mediados de los 70 en adelante. Y además, más aumenta según pasa el tiempo.

El Augmented Dickey Fuller test (ADF Test) nos confirma estadísticamente que la serie es no estacionaria. En lo que se refiere al suavizado de la serie, al disponer datos que son medias mensuales, la serie no presenta mucho ruido en sí. En la siguiente figura observamos la serie original y el suavizado de tipo “moving average” de 3 meses.

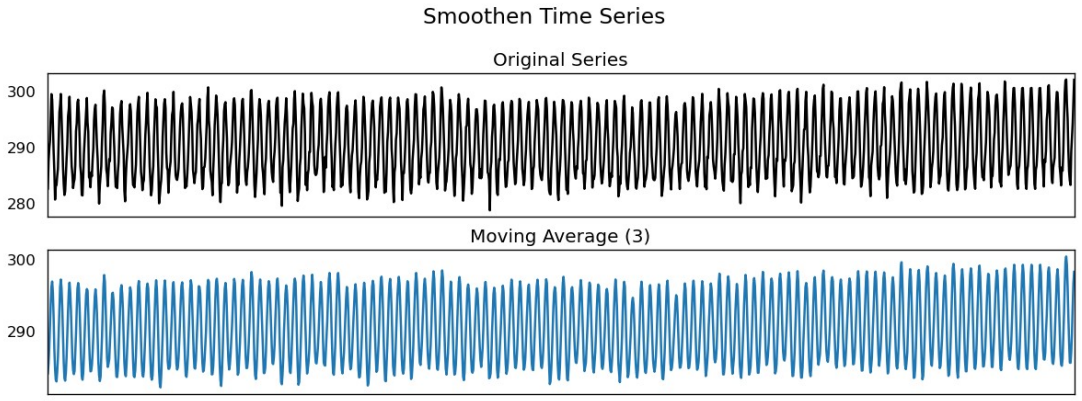

Fig. 3. Suavizado de la serie temporal (3 months moving average)

Con los datos preprocesados, se ejecuta el algoritmo de auto arima, que nos ofrece como resultado que el mejor modelo es : SARIMA (4, 1, 0) x (1, 0, 1) 12. Además se observa en la columna std err unos valores de errores bastante bajos.

ARIMA order (p, d, q): (4, 1, 0)
Seasonal order (P, D, Q, S): (1, 0, 1, 12)

SARIMAX Results

Dep. Variable:	y	No. Observations:	1423
Model:	SARIMAX(4, 1, 0)x(1, 0, [1], 12)	Log Likelihood	-660.998
Date:	Sun, 17 Sep 2023	AIC	1337.996
Time:	12:31:24	BIC	1380.074
Sample:	02-01-1905 - 08-01-2023	HQIC	1353.713
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	6.479e-05	0.001	0.085	0.933	-0.001	0.002
ar.L1	0.1858	0.024	7.854	0.000	0.139	0.232
ar.L2	0.0930	0.024	3.886	0.000	0.046	0.140
ar.L3	0.0057	0.025	0.229	0.819	-0.043	0.054
ar.L4	-0.4985	0.024	-20.745	0.000	-0.546	-0.451
ar.S.L12	0.9932	0.002	639.786	0.000	0.990	0.996
ma.S.L12	-0.7855	0.021	-37.946	0.000	-0.826	-0.745
sigma2	0.1460	0.006	25.809	0.000	0.135	0.157

Ljung-Box (L1) (Q):	3.92	Jarque-Bera (JB):	2.40
Prob(Q):	0.05	Prob(JB):	0.30
Heteroskedasticity (H):	0.79	Skew:	0.00
Prob(H) (two-sided):	0.01	Kurtosis:	3.20

Tabla 1. Tabla resumen del modelo SARIMA obtenido con auto arima.

Los gráficos de diagnósticos nos ofrecen unos residuales hacia ambos lados por igual, un histograma de densidad bastante simétrico, unos quantiles bastante cercanos a la diagonal principal y eso sí, un correlograma donde unos pocos valores se salen de la región azul (que es la idónea).

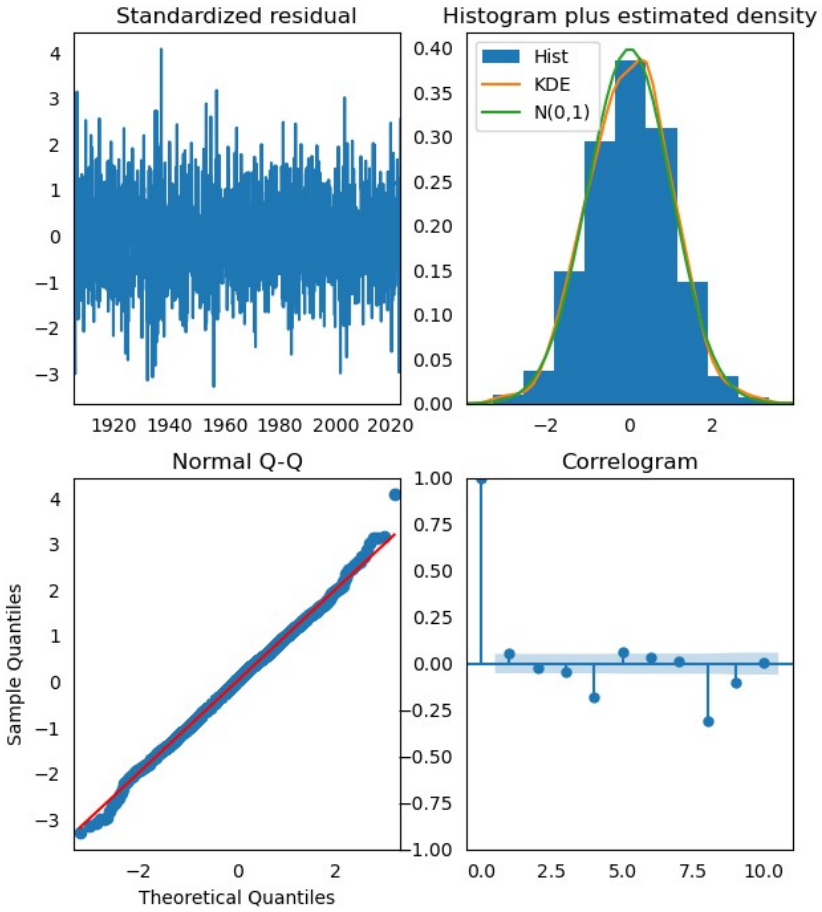

Fig. 4. Gráficos de diagnósticos para verificar el modelo

El último paso para la validación del modelo es el backtesting. En la siguiente gráfico se observan los splits de los sets en train (azul), test (amarillo), las predicciones de SARIMA sobre el test (en rojo, que es donde se calculan los MSE) y en verde la previsión a 2 años.

Con el backtesting hemos obtenido valores de errores cuadráticos medios (MSE) bastante aceptables. La media es 0.73 $^{o}K^{2}$, debajo los errores en cada fold respectivamente:

Mean Squared Error (all folds): 0.73
Mean Squared Error (across folds): [0.64 0.56 0.58 1.0 0.86]

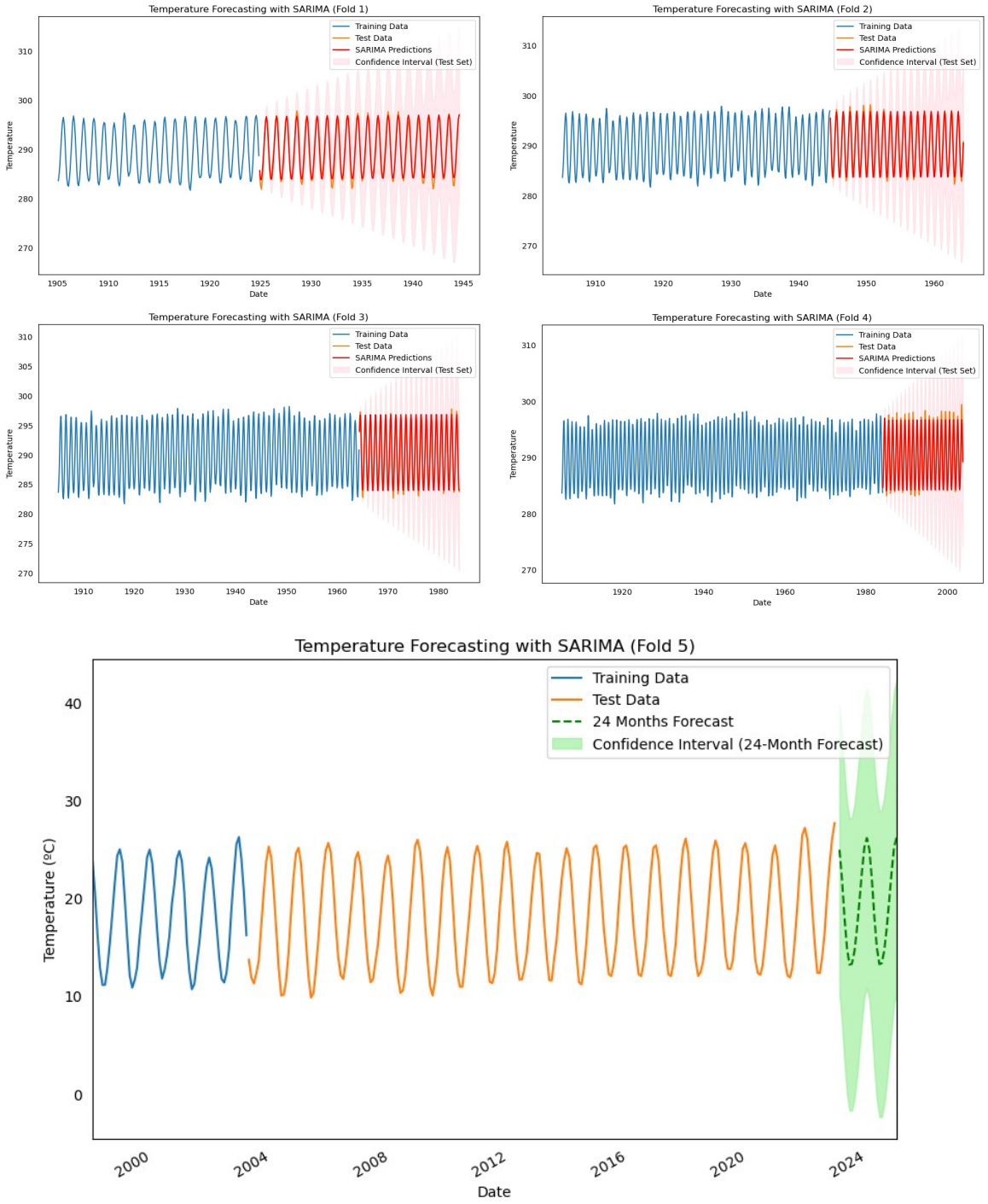

Fig. 5. Backtesting. el quinto y último fold contiene el detalle de la previsión de temperatura a 2 años con su intervalo de confianza (en verde).

Por último ofrecemos una vista del mapa con folium, donde se localiza la posición del observatorio, así como la temperatura y su fecha correspondiente

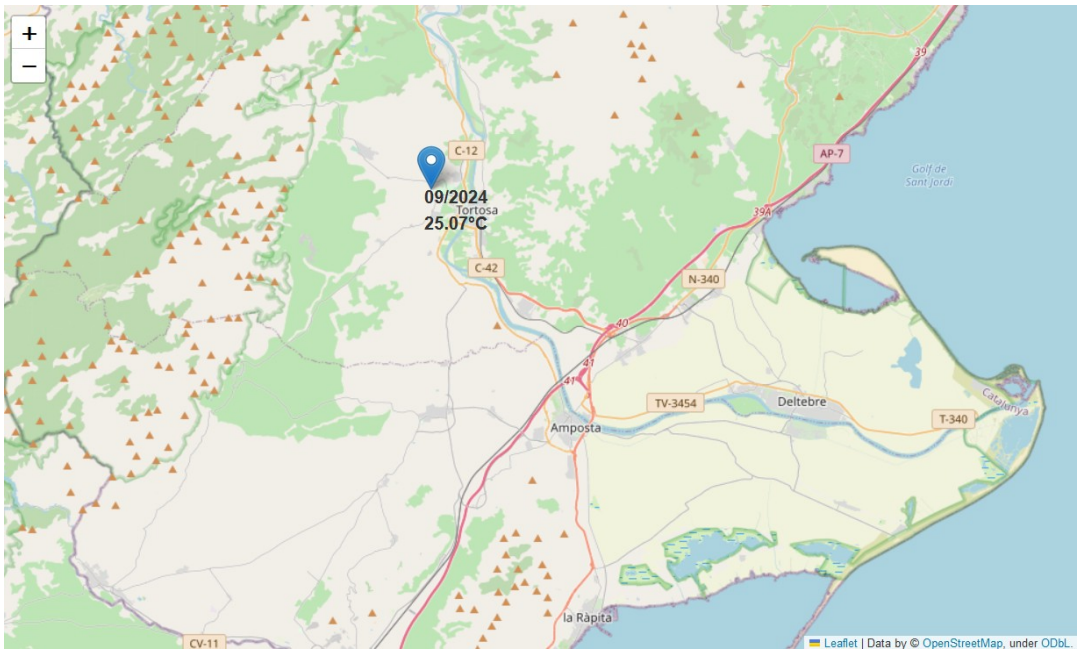

Fig. 6. Geolocalización del observatorio del Ebro y fecha y temperatura correspondientes.

5. CONCLUSIÓN

El proyecto ha sido exitoso en lo que se refiere a extracción de datos, preprocesado, ajuste al modelo SARIMA, comprobación del mismo y muestra de datos en mapa con geolocalización.

Como siguientes pasos en este proyecto se plantea mostrar el mapa con los datos en un dashboard, mostrar datos de más de un observatorio en el mismo mapa y, por ser un gas muy relevante en lo que a calentamiento global se refiere, añadir datos de concentración dióxido de carbono (CO_2) al modelo. Para ello habrá que cambiar el modelo a SARIMAX, ya que esta concentración será considerado una variable exógena.

6. REFERENCIAS

[1] https://www.machinelearningplus.com/time-series/time-series-analysis-python/#google_vignette

[2] https://neptune.ai/blog/time-series-prediction-vs-machine-learning

[3] https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/#google_vignette

[4] https://towardsdatascience.com/time-series-in-python-part-2-dealing-with-seasonal-data-397a65b74051

[5] https://towardsai.net/p/l/time-series-forecasting-with-arima-models-in-python-part-2

[6] https://neptune.ai/blog/arima-sarima-real-world-time-series-forecasting-guide

[7] https://github.com/alabarga/taller-cambio-climatico/blob/main/Climate%20Change.ipynb

[8] https://github.com/alabarga/taller-cambio-climatico/blob/main/Local%20Climate%20Data.ipynb

[9] https://github.com/Onurcn93/Global_Temperature_Change/blob/main/Global%20Temperature%20Change%20Analysis.ipynb.

[10] https://github.com/SameedAhmedKhan/GeospatialAnalysisUsingFolium/blob/main/Temperature_(degrees_Celsius)_Pins.ipynb

[11] https://github.com/SiddheshBangar/Global-Temperature-Analysis/blob/main/GlobalTemperature.ipynb

[12] https://github.com/xsuryanshx/Data-Science-on-Geospatial-Data/blob/main/Geospatial_Data_Science.ipynb

[13] https://github.com/white-beard/Geospatial-data-analysis/blob/main/Wikilimo_Ml_Task.ipynb

[14] https://github.com/amitsingh4183/Predicting-Forest-fire-Based-On-Geospatial-Image-Analysis/blob/master/Insights%20into%20Forest%20Fires.ipynb)

[15] https://github.com/DjangoJain/Geospatial-Analysis-and-Temperature-Prediction-Using-Time-Series-Model/blob/main/Geospatial_Analysis__Temperature_Prediction.ipynb)

[16] [https://data.giss.nasa.gov/gistemp/station_data_v4_globe/]

[17] https://data.giss.nasa.gov/cgi-bin/gistemp/stdata_show_v4.cgi?id=SP000009981&dt=1&ds=14