



UNIVERSIDAD
AUSTRAL | INGENIERÍA

Análisis de Series Temporales

Contaminación del Aire

Equipo #1:
Maria Cecilia Arce
Maria Virginia Ainchil
Alejandra Cecco

Trabajo Práctico #2

18 de Agosto, 2024.

Resumen Ejecutivo

El Trabajo Práctico #2 de Análisis de Series Temporales, se realizó con las mismas 3 series temporales utilizadas en el Trabajo Práctico #1; las cuales corresponden a la medición de 3 variables presentes en el aire (benceno, temperatura y humedad) con la finalidad de determinar la calidad del aire.

El objetivo del presente trabajo, es emplear diferentes modelos de aprendizaje automático y profundo para predecir el comportamiento de las tres series temporales y poder contrastar los resultados obtenidos, con el análisis realizado anteriormente con los modelos ARIMA para las mismas series de tiempo (en el TP #1).

Se ejecutaron distintos modelos y se evaluaron métricas para evaluar el rendimiento de los mismos, y poder determinar cuál de ellos predice mejor el comportamiento de las 3 series temporales en cuestión.

Contenido

Resumen Ejecutivo.....	2
Contenido.....	3
Introducción.....	4
Marco Teórico.....	7
Modelos de Machine Learning.....	8
Prophet.....	9
Modelo Prophet para Serie Temporal Benceno (C6H6).....	9
SVR (Support Vector Regression).....	10
Modelo SVR para Serie Temporal Benceno (C6H6).....	12
Modelo SVR para Serie Temporal Temperatura (T).....	13
Modelo SVR para Serie Temporal Humedad Relativa (HR).....	14
XGBoost.....	15
Modelo XGBoost para Serie Temporal Benceno (C6H6).....	15
Modelo XGBoost para Serie Temporal Temperatura (T).....	16
Modelo XGBoost para Serie Temporal Humedad Relativa (HR).....	17
LightGBM.....	18
Modelo Light GBM para Serie Temporal Benceno (C6H6).....	18
Modelo Light GBM para Serie Temporal Temperatura (T).....	19
Modelo Light GBM para Serie Temporal Humedad Relativa (HR).....	20
Modelos de Deep Learning.....	20
LSTM.....	21
Modelo LSTM para Serie Temporal Benceno (C6H6).....	22
Modelo LSTM para Serie Temporal Temperatura (T).....	23
Modelo LSTM para Serie Temporal Humedad Relativa (HR).....	24
Modelos Híbridos.....	25
Neural Prophet.....	26
Modelo Neural Prophet para Serie Temporal Benceno (C6H6).....	27
Modelo Neural Prophet para Serie Temporal Temperatura (T).....	27
Modelo Neural Prophet para Serie Temporal Humedad Relativa (HR).....	28
H2O AutoML.....	28
Modelo H2O Auto ML para Serie Temporal Benceno (C6H6).....	29
AutoTS.....	30
Análisis de Resultados.....	33
Bibliografía.....	35
Anexo.....	36
Enlace a los Scripts.....	38

Introducción

Desde hace ya algunos años, el tema del cuidado ambiental, ha tomado protagonismo como una prioridad para el correcto desarrollo de la humanidad. En este trabajo se pretende analizar particularmente la calidad del aire que respiramos.

Se considera que los contaminantes de la atmósfera en zonas urbanas, son responsables del aumento de enfermedades respiratorias en los seres humanos; y se ha descubierto, que algunos contaminantes en particular, como es el caso del benceno (C_6H_6), aumentan la probabilidad de contraer cáncer.¹ Es por eso, que una estimación precisa sobre la distribución de contaminantes en el aire que respiramos, puede ser relevante para tomar medidas de prevención en temas de salud.

Hoy en día, el seguimiento de la contaminación del aire en zonas urbanas, se lleva a cabo mediante redes de estaciones fijas distribuidas estratégicamente. Estos equipos se encargan de estimar de forma selectiva y precisa las concentraciones de muchos componentes presentes en la atmósfera.

En este contexto, el análisis de series temporales se ha establecido como una herramienta crucial para monitorear y comprender la dinámica de la calidad del aire a lo largo del tiempo. Esta metodología permite identificar patrones estacionales, tendencias a largo plazo, variaciones diarias y eventos extremos que afectan la concentración de contaminantes atmosféricos. Al analizar datos recopilados de estaciones de monitoreo distribuidas estratégicamente, es posible evaluar la eficacia de las políticas de control de la contaminación, así como anticipar y mitigar los impactos adversos en la salud pública y el medio ambiente.

El conjunto de datos con el que se realizó este trabajo práctico, contiene 9358 observaciones; que provienen del promedio por hora registrado en una serie de 5 sensores químicos integrados en un dispositivo con el objetivo de medir y controlar la calidad del aire. Dicho dispositivo, desarrollado por Pirelli Labs, se colocó en una zona significativamente contaminada de una ciudad italiana.

Los datos utilizados pertenecen a los valores registrados durante un año (de marzo del 2004 hasta febrero del 2005), todos los días, cada hora. Las variables registradas por el sensor y presentes en el dataset, son 14. Como se mencionó anteriormente, el objetivo de este trabajo es analizar las series temporales de 3 variables,

¹ S. De Vito, E. Massera, M. Piga, L. Martinotto, G. Di Francia. (2008). "On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario". ScienceDirect. <https://www.sciencedirect.com/science/article/pii/S0925400507007691?via%3Dihub>

por lo que se seleccionaron las 3 variables continuas, más significativas para el análisis:

- **C6H6:** corresponde al promedio de benceno registrado en el aire. La unidad en la que se trabaja esta variable es microrg/m³.
- **T:** se refiere a la temperatura del aire. Está medido en °C. Como se mencionó anteriormente, el estudio se realizó en Italia, por lo que las temperaturas bajan en el invierno italiano (noviembre, diciembre, enero) y suben en el verano (junio, julio, agosto).
- **RH:** se refiere a la humedad relativa. Se mide en %.

El análisis de series temporales, es una técnica utilizada para predecir valores futuros, basándose en datos históricos. Con el objetivo de identificar patrones, tendencias y comportamientos en los datos, lo que permite realizar pronósticos y ayudar en la toma de decisiones informadas. Dicha técnica, ha tenido su evolución a lo largo de los años.

En un inicio, los problemas de series temporales se resolvían con modelos de medias móviles (MA):

- **SMA (Promedio Móvil Simple):** se calcula el promedio de un número de observaciones pasadas.
- **EMA (Promedio Móvil Exponencial):** es un tipo de promedio móvil que asigna un mayor peso a los datos más recientes, mediante la utilización de un factor de suavizado que decrece exponencialmente para los valores más antiguos.

Después aparecieron los Modelos Autorregresivos (AR), tales como:

- **AR (p):** predicen valores futuros basándose en una combinación lineal de valores pasados hasta el lag “p”.
- **ARMA (p,q):** combina modelos AR y MA, capturando componentes autorregresivos y de media móvil.

Posteriormente aparecieron modelos estadísticos más avanzados como:

- **ARIMA (p,d,q):** incorpora la diferenciación al modelo ARMA para poder manejar datos no estacionarios.
- **SARIMA (p,d,q)(P,D,Q):** extiende ARIMA, tomando en cuenta la estacionalidad de los datos.

Luego, se empezó a utilizar Machine Learning y Deep Learning para resolver problemas de series temporales, ofreciendo innumerables modelos muy poderosos y

logrando mejores resultados que con los modelos tradicionales.

Actualmente, con la aparición de la Inteligencia Artificial Generativa, los avances siguen sorprendiendo y siguen surgiendo nuevos y muy poderosos modelos, para analizar series temporales.

Marco Teórico

En el presente trabajo, se pretende explicar el comportamiento de las series temporales ya mencionadas con el objetivo de poder tener una predicción confiable para dichas variables analizadas. Anteriormente se realizó este análisis empleando modelos ARIMA (TP#1). En este caso, se utilizaron distintos modelos de Machine Learning y Deep Learning, con la finalidad de obtener mejores resultados que en el TP #1.

Se sabe que los modelos ARIMA, se basan en supuestos estadísticos, por lo que es necesario que los datos sean estacionarios; es decir, que las propiedades estadísticas, tales como la media y la varianza, sean constantes a lo largo del tiempo.

En el caso de los modelos de Machine y Deep Learning, esta condición no es necesaria, por lo que no se tiene que hacer ningún tipo de análisis previo a los datos, diferenciaciones y/o transformaciones a las series para cumplir con ningún principio. Se trata de modelos más flexibles, que logran modelar relaciones no lineales y complejas en los datos. Son modelos que no dependen de supuestos estadísticos rígidos, por lo que suelen adaptarse mejor a diferentes tipos de datos.

Es importante mencionar, que al aplicar modelos de machine learning y/o deep learning en series temporales, hay cuestiones a tener en cuenta para el correcto funcionamiento de los modelos:

- División de los datos: al tratarse de modelos pre entrenados, es sumamente importante hacer la correcta separación en conjuntos de entrenamiento, validación y prueba.
- Evaluación de los modelos: el performance de los modelos se evalúa, con métricas tales como el RMSE, el MAE y/o en MAPE.

El dataset con el que se trabajó, no tenía valores nulos; pero al momento de analizar los datos, se descubrió que las 3 series temporales, tenían picos que iban hasta el valor -200; este valor de -200 se imputa automáticamente cuando el dispositivo tiene una falla de calibración. Se realizó una corrección a estos datos, para modificarlos, por valores más reales a través del método “Forward Fill”, el cual consiste en propagar el último valor válido registrado hacia adelante en el tiempo. De esta manera se logró sacar los picos que perturban tanto las series. El resto del trabajo se realizó con esta modificación sobre los datos originales.

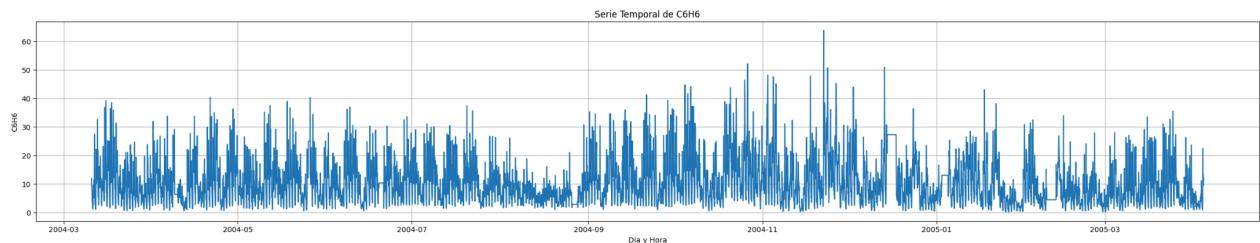


Figura 1. Serie Temporal Benceno (C6H6).

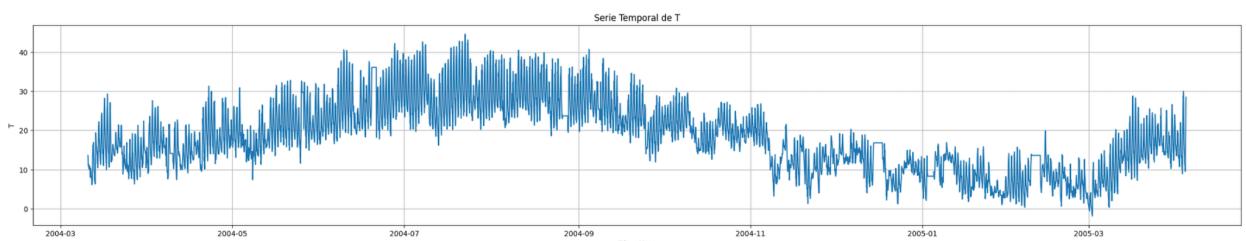


Figura 2. Serie Temporal Temperatura (T).

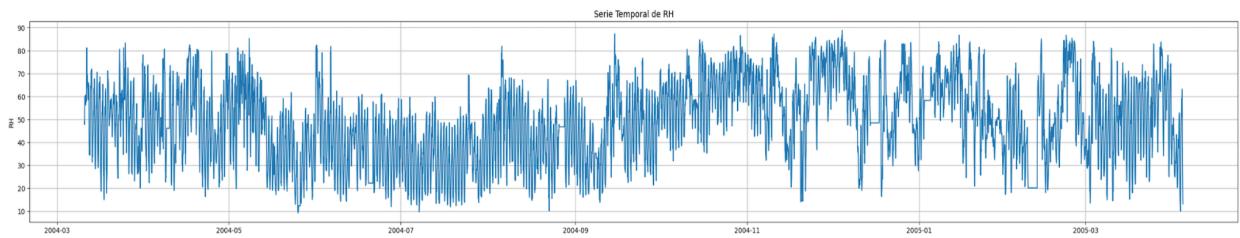


Figura 3. Serie Temporal Humedad Relativa (RH).

Modelos de Machine Learning

El Machine Learning (ML) es un subcampo de la inteligencia artificial que se centra en el desarrollo de algoritmos y técnicas que permiten encontrar patrones recurrentes en conjuntos de datos; los cuales pueden ser números, palabras, imágenes, estadísticas, etc.

El Machine Learning se utiliza para enseñar a las computadoras a aprender de los datos y ajustarse automáticamente para mejorar su rendimiento. El ML, se divide en dos categorías principales:

- *Aprendizaje supervisado:* se utilizan conjuntos de datos etiquetados para enseñar a la computadora a hacer predicciones.

- *Aprendizaje no supervisado:* la computadora aprende patrones y relaciones por sí misma.

Los modelos de Machine Learning son muy útiles para el correcto análisis de series temporales; son capaces de predecir el futuro basándose en patrones y tendencias en los datos históricos. De igual manera, los modelos ML, son capaces de identificar patrones ocultos y relaciones en los datos que no son fáciles de identificar fácilmente por el ojo humano.

A continuación se detallan los Modelos ML que se utilizaron para analizar las series temporales. Es importante mencionar que primero se probaron con la variable Benceno (C6H6) y aquellos modelos que arrojaron mejores resultados (métricas de desempeño), se aplicaron al resto de las series temporales (Temperatura y Humedad Relativa).

Prophet

Prophet es un modelo desarrollado por Facebook, con el objetivo de ofrecer a los usuarios una herramienta intuitiva, fácil de usar y potente para predecir datos en series temporales. El algoritmo que utiliza Prophet tiene en cuenta la estacionalidad y la tendencia de los datos; dos componentes de las series temporales que son muy difíciles de cuantificar. Además, tiene una alta relación de precisión y velocidad; es decir, se logran buenos resultados en poco tiempo.

Modelo Prophet para Serie Temporal Benceno (C6H6)

A pesar de seleccionar los hiperparámetros mediante un grid search, el modelo no ha mejorado sustancialmente respecto al modelo realizado con los parámetros por defecto. Esto probablemente se deba a que los parámetros predeterminados son cuidadosamente previstos para una gran cantidad de casos simples como es el caso de la serie C6H6.

De los modelos sin las variables exógenas, los que presentan mejores métricas son los Modelos 1, 5 y 6. Se considera el Modelo 1 por ser más simple. La diferencia entre estos modelos es el valor de **fourier_order**, que representa la cantidad de términos de Fourier utilizados para modelizar la estacionalidad. Cuanto más simple es el patrón de estacionalidad, más bajo es este valor. La similitud en las métricas obtenidas con diferentes valores de **fourier_order** sugiere que la estacionalidad no es muy compleja

y que con pocos términos se captura adecuadamente, por lo que agregar más términos de Fourier no aporta información adicional.

El hecho de que el modelo no haya mejorado al incluir las variables exógenas podría indicar que el modelo Prophet ya ha capturado adecuadamente el patrón subyacente y no necesita de las variables exógenas.

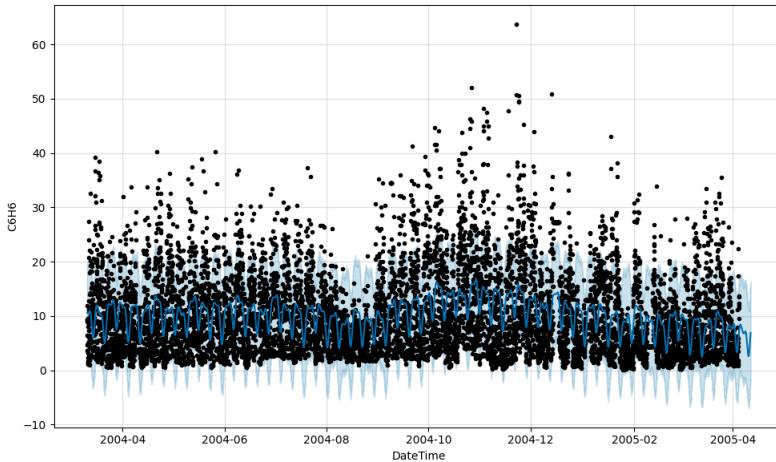


Figura 4. Resultado del Modelo Prophet aplicado a la serie C6H6.

En este caso la predicción fue de 168 horas (una semana).

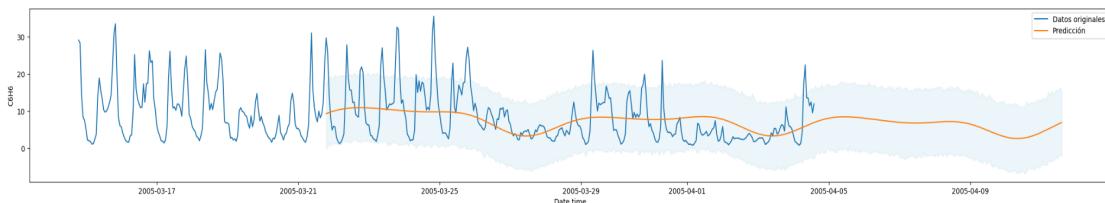


Figura 5. Predicción de 1 semana lograda con el Modelo Prophet para la variable C6H6.

SVR (Support Vector Regression)

Las SVM son un tipo de algoritmo de aprendizaje supervisado utilizado tanto para clasificación como para regresión. Esta herramienta es más comúnmente utilizada para problemas de clasificación; pero de igual manera, al contar con las adecuaciones correctas de preprocesamiento resulta ser una herramienta poderosa para el análisis de series temporales. La extensión del SVM para el forecasting de series temporales es el SVR.

La estimación de una SVR busca encontrar una función de regresión que ajuste mejor a los datos maximizando el margen entre puntos y minimizando los puntos fuera del margen.

La estimación de los parámetros de una SVM es equivalente a la solución de un modelo de programación cuadrática con restricciones lineales; lo que implica que la solución óptima es global y única. Esto representa una clara ventaja sobre otro tipo de modelos que se caracterizan por poseer múltiples puntos de mínima local.

El problema de programación cuadrática de cuya solución se obtienen los parámetros de las SVM , depende de los datos (serie de tiempo) y de varias constantes que representan los parámetros de la función de núcleo (o kernel) utilizada, y de los parámetros de la función de riesgo. Dichas constantes, dependen de cada serie temporal particular, y no existen métodos que permitan su estimación; es decir, tienen que ser definidos de manera heurística.²

La función kernel mapea los datos originales en un espacio de alta dimensión para encontrar el hiperparámetro óptimo para la regresión. La elección del kernel adecuado para analizar una serie temporal depende de las características y la complejidad de los mismos datos. No existe un kernel universalmente mejor; ya que la efectividad del mismo varía según el problema a analizar. Los distintos tipos de kernel son:

- Kernel Lineal.
- Kernel Polinomico.
- Kernel Radial (RBF).
- Kernel Sigmoide.

Sin embargo, el Kernel Radial (RBF) suele ser el más utilizado porque permite capturar patrones tanto lineales como no lineales.³

Otro de los parámetros a considerar en el modelo SVM es el parámetro C (de regulación). Controla el trade-off entre lograr un margen grande y clasificar correctamente los puntos de entrenamiento. Corresponde a la distancia del hiperplano.

En cuanto al parámetro gamma, es un parámetro específico del kernel RBF, polinómico y sigmoide. Define cómo influye un solo ejemplo de entrenamiento. Un valor

² J. D. Velasquez, Y. Olaya, C. J. Franco. (2010). “Predicción de Series Temporales usando Máquinas de Vectores de Soporte”. Ingeniare. Revista chilena de ingeniería, vol. 18 Nº 1, 2010, pp. 64-7. https://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-33052010000100008&lng=es&nrm=iso&tlang=es

³A. Sethi (2024). “ Support Vector Regression Tutorial for Machine Learning”. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/>

pequeño de gamma (por ejemplo, 0.1) significa que la influencia de un solo punto de entrenamiento es grande, y por lo tanto, el modelo es más suave. Un valor grande de gamma significa que la influencia de un solo punto de entrenamiento es pequeña, lo que resulta en un modelo más complejo y menos suave.

Modelo SVR para Serie Temporal Benceno (C6H6)

Se probaron seis modelos distintos, con diferentes valores de C: 1, 10, 100 y 1000, y de gamma: 0.1, 0.01, y "auto" (esta opción permite al propio algoritmo elegir el valor de gamma más adecuado). Todos los modelos se probaron con kernel radial, por ser el más utilizado. Se seleccionó el modelo 5 con C=1 y gamma="auto".

Además, para que el modelo pueda capturar las dependencias temporales y las fluctuaciones de los datos se crearon 24 lags y 24 periodos estacionales. Se tomó el valor 24 por el patrón de 24 horas que se reflejaba en la gráfica de toda la serie.

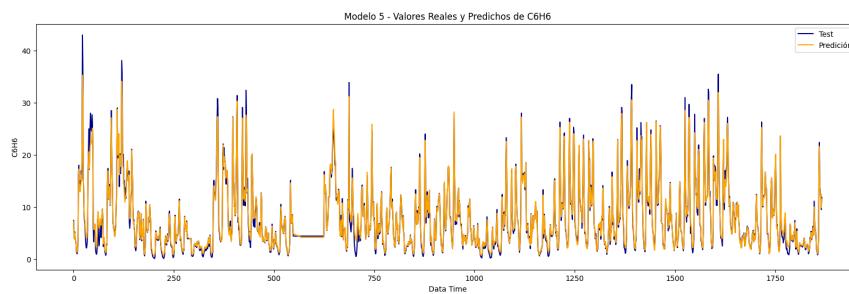


Figura 6. Predicción realizada con Modelo SVM vs. Serie Original C6H6.

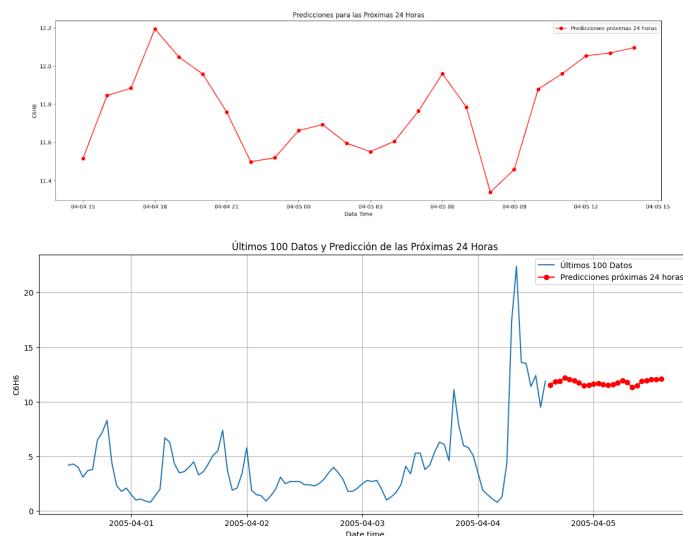


Figura 7. Últimos 100 días y Forecast de 24 horas para C6H6 con Modelo SVM.

Modelo SVR para Serie Temporal Temperatura (T)

Para la serie de Temperatura, se probaron 5 distintos modelos SVM, resultando el Modelo#4 el del mejor desempeño; con un kernel radial, un valor de $C= 1000$ y $\gamma=0,01$.

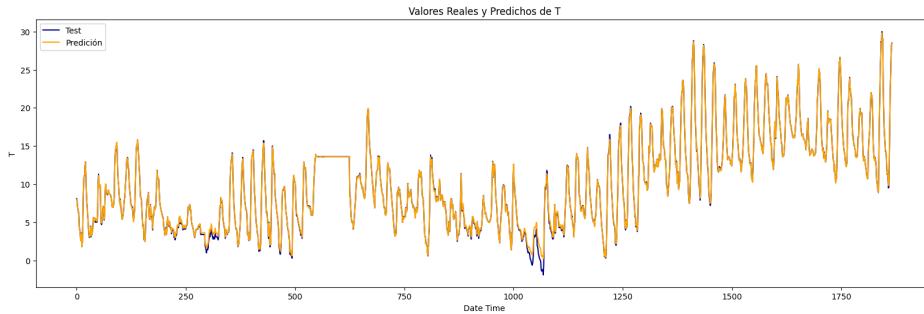


Figura 8. Predicción realizada con Modelo SVM vs. Serie Original variable 'T'.



Figura 10. Forecast de 24 horas para T con Modelo SVM.

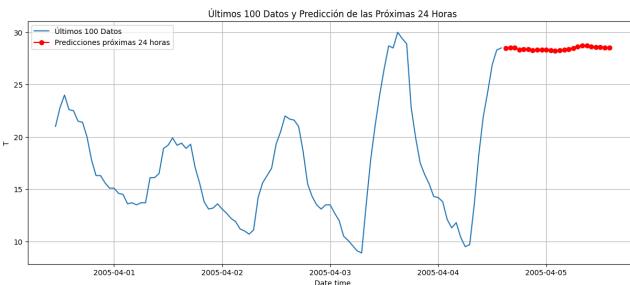


Figura 11. Últimos 100 días y Forecast de 24 horas para T con Modelo SVM.

Modelo SVR para Serie Temporal Humedad Relativa (RH)

Se corrieron 5 modelos SVM, resultando el Modelo#3 el del mejor desempeño; con un kernel radial, un valor de $C= 10$ y $\gamma=0,01$.

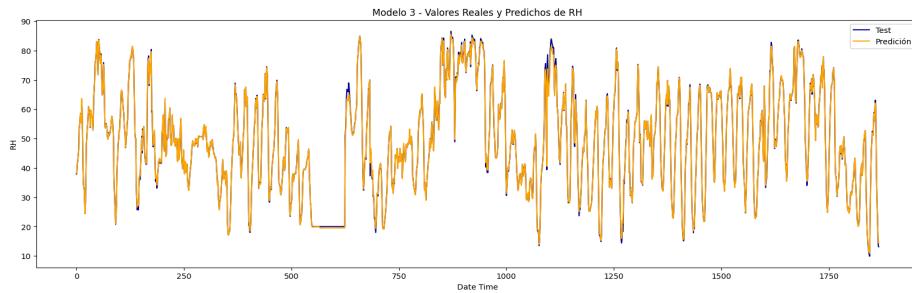


Figura 12. Predicción realizada con Modelo SVM vs. Serie Original variable 'RH'.

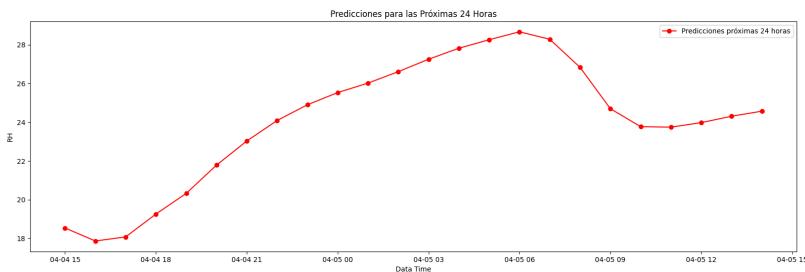


Figura 13. Forecast de 24 horas para T con Modelo SVM.

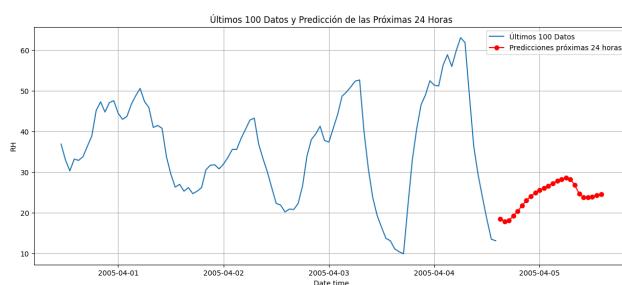


Figura 14. Últimos 100 días y Forecast de 24 horas para RH con Modelo SVM.

XGBoost

XGBoost (Extreme Gradient Boosting) es una implementación del algoritmo Gradient Boosting (GBM) para tareas de aprendizaje supervisado. Este algoritmo utiliza ensambles de árboles de decisión, combinando múltiples modelos “débiles” para generar un modelo predictivo con mejor desempeño. XGBoost ofrece gran flexibilidad en cuanto al ajuste de hiperparámetros para controlar la complejidad y evitar el sobreentrenamiento (overfitting). Además, proporciona soporte para trabajar con valores faltantes y variables categóricas.⁴

Modelo XGBoost para Serie Temporal Benceno (C6H6)

Previo al entrenamiento de los modelos, se utilizaron el comando `TimeSeriesSplit` de scikit-learn para separar los datos en 5 carpetas, cada una con un grupo de entrenamiento y prueba, para realizar la validación cruzada. Este comando respeta el orden temporal de los datos, lo cual es esencial en series temporales.

Se crearon 8 nuevas features que se usaron en los distintos modelos. En algunos modelos, no se agregaron aquellas que resultaban de poca incidencia, lo cual se verifica con el gráfico de feature importance. También se evaluaron los gráficos de entrenamiento y prueba para evitar el sobreajuste. Se consideraron 3 rezagos de 24, 48 y 72 horas. Se analizaron 8 modelos, y el que presentó la mejor métrica fue el Modelo 8, que incorpora las variables exógenas T y RH.

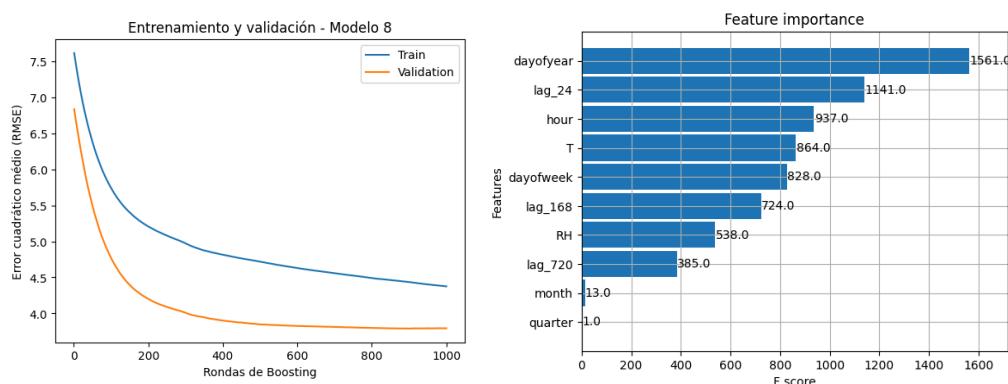


Figura 16. Gráfico de Curva de Aprendizaje e Importancia de Features para el Modelo 8 XGBoost.

⁴ I. Lee. (Abril 10, 2022). “Series de Tiempo: Forecasting con XGBoost”. Medium. <https://ivan-lee.medium.com/series-de-tiempo-con-xgboost-f732f1da3056>

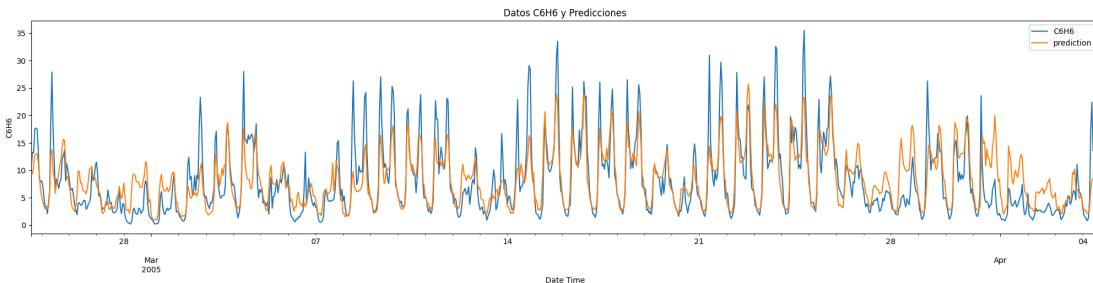


Figura 17. Serie Temporal C6H6 vs. Predicciones logradas con el Modelo 8 XGBoost.

Al predecir las horas futuras (en este caso se consideraron 24), fue necesario simular los valores futuros de las variables agregadas; para las variables exógenas T y RH se tomó su valor promedio.

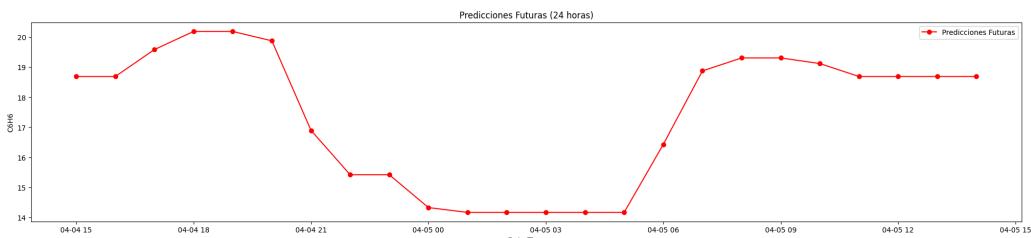


Figura 18. Pronóstico para las próximas 24 horas para C6H6 con el modelo XGBoost.

Modelo XGBoost para Serie Temporal Temperatura (T)

Después de probar con 7 modelos distintos, el Modelo que demostró el mejor performance fue el Modelo 7; en el cual se incluyen 4 features nuevas, las variables C6H6 y HR como variables exógenas, y 3 distintos lags. A continuación los resultados obtenidos:

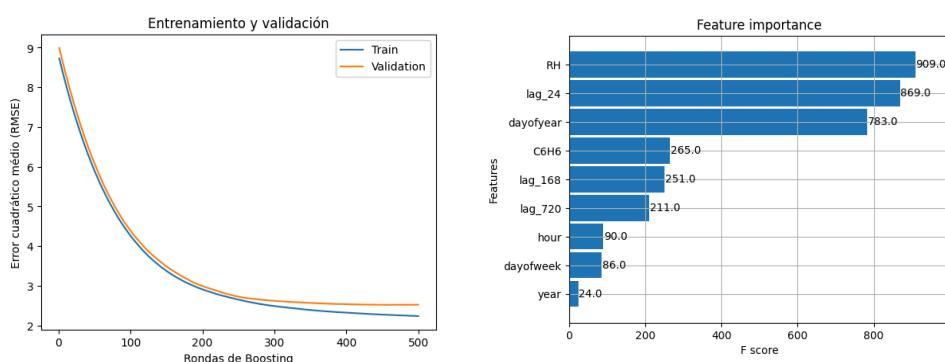


Figura 19. Gráfico de Curva de Aprendizaje e Importancia de Features para el Modelo 7 XGBoost.

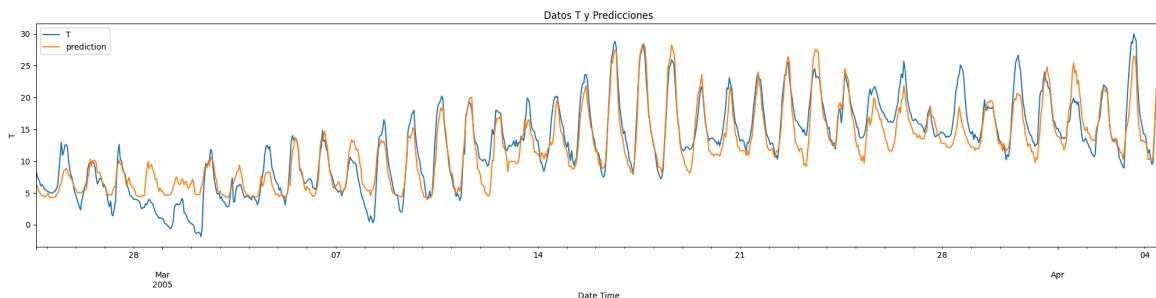


Figura 20. Serie Temporal T vs. Predicciones logradas con el Modelo 7 XGBoost.

Modelo XGBoost para Serie Temporal Humedad Relativa (HR)

Tras evaluar 8 modelos diferentes, el que mostró el mejor rendimiento fue el Modelo 2, que incorpora 6 nuevas features y 3 lags diferentes. A continuación, se presentan los resultados obtenidos:

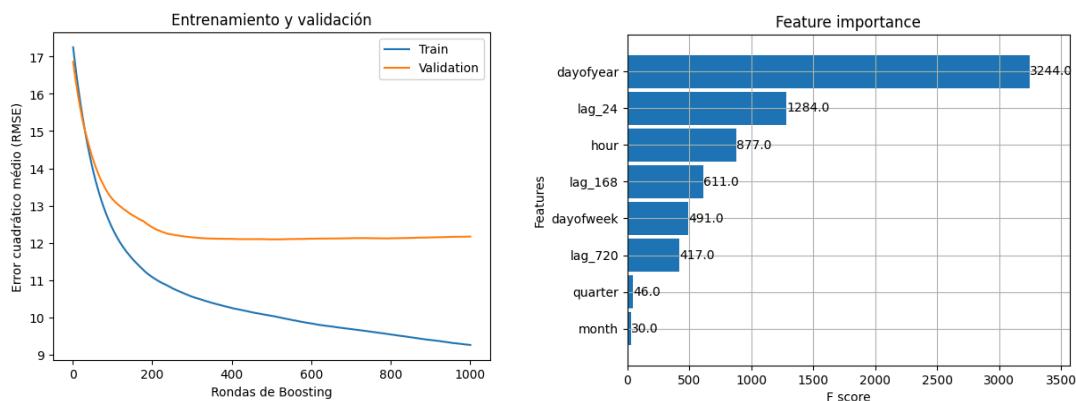


Figura 22. Gráfico de Curva de Aprendizaje e Importancia de Features para el Modelo 2 XGBoost.

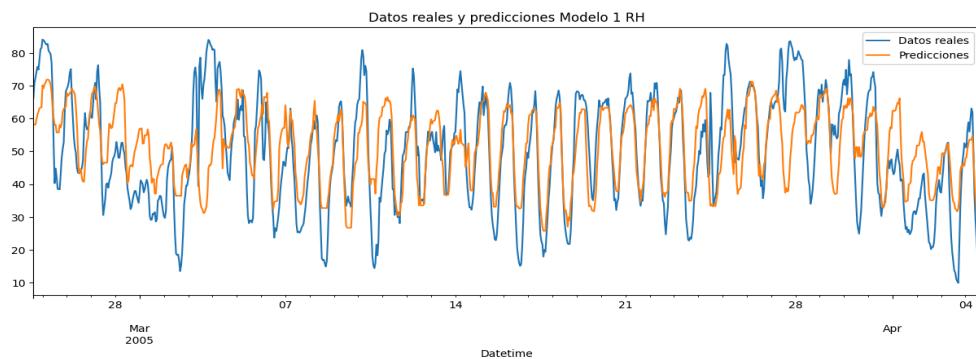


Figura 23. Serie Temporal RH vs. Predicciones logradas con el Modelo 2 XGBoost.

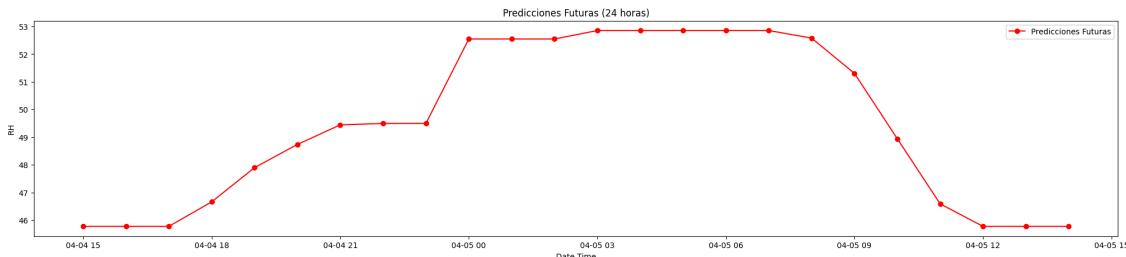


Figura 24. Pronóstico para las próximas 24 horas para HR con el modelo 2 XGBoost.

LightGBM

LightGBM es una implementación altamente eficiente del algoritmo gradient boosting por lo que se ha convertido en un referente en el campo del Machine Learning.

En primer lugar, se entrena un modelo ForecasterAutoreg utilizando valores pasados de la variable de respuesta (lags) como predictores. Posteriormente, se añaden variables exógenas al modelo y se evalúa la mejora de su rendimiento. Dado que los modelos de Gradient Boosting tienen un gran número de hiperparámetros, se realiza una búsqueda bayesiana utilizando la función `bayesian_search_forecaster()` para encontrar la mejor combinación de hiperparámetros y lags. Por último se evalúa la capacidad predictiva del modelo mediante un proceso de backtesting; el cual consiste en generar una predicción para cada observación del conjunto de test, siguiendo el mismo procedimiento que se seguiría si el modelo estuviese en producción, para finalmente comparar el valor predicho con el valor real.⁵

Modelo Light GBM para Serie Temporal Benceno (C6H6)

Se analizaron 10 modelos con distintos parámetros, algunos de los cuales se seleccionaron a través de un pequeño grid search con diferentes valores de `learning_rate` y `bagging_freq` distintos. El mejor modelo de los probados fue el Modelo 1, que, de manera similar al modelo XGBoost, incluye las variables exógenas T y RH, además de 4 nuevas características y 3 lags. Esto era previsible, ya que el análisis del modelo VAR

⁵ J. Amat Rodrigo, J. Escobar Ortiz (2021). “Forecasting series temporales con gradient boosting: Skforecast, XGBoost, LightGBM y CayBoost”. Skforecast.

<https://cienciadedatos.net/documentos/py39-forecasting-series-temporales-con-skforecast-xgboost-lightgbm-catboost>

sugiere una relación de causalidad grangeriana entre las series T, RH y C6H6. En la predicción para las 24 horas siguientes, también se tuvieron que estimar las características, rezagos y variables exógenas.

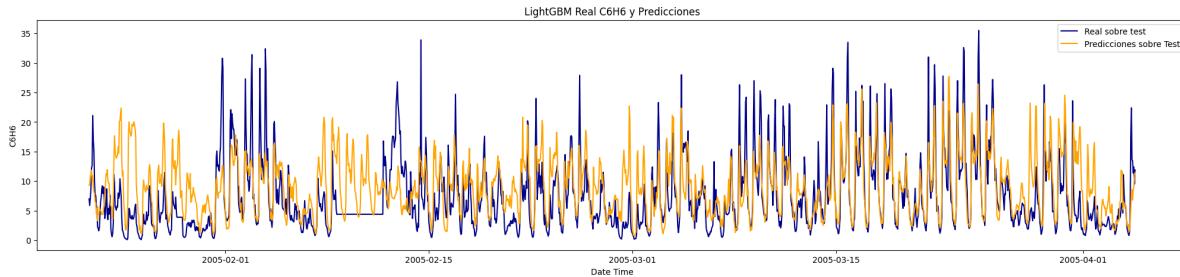


Figura 25. Valores reales de C6H6 (test) vs Valores predichos con Modelo 1 LighGBM.

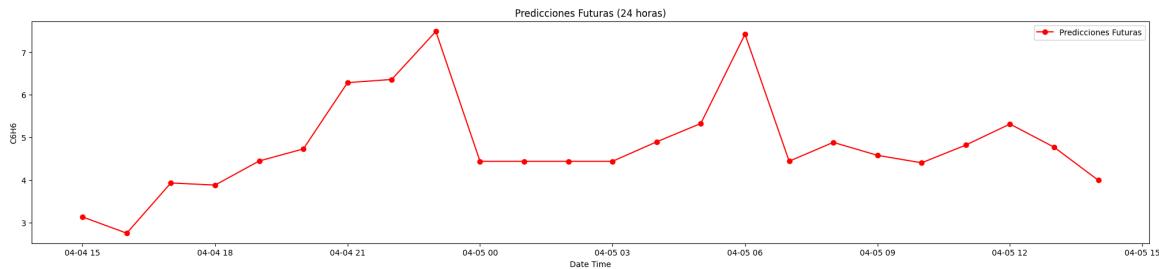


Figura 26. Pronóstico de 24 horas futuras para C6H6 con Modelo 1 LighGBM.

Modelo Light GBM para Serie Temporal Temperatura (T)

Se probaron 4 modelos distintos para la serie Temperatura; no encontrando diferencias demasiado significativas entre ellos al modificar los mismos hiperparametros que con la serie C6H6. El modelo “ganador” fue el Modelo 1.

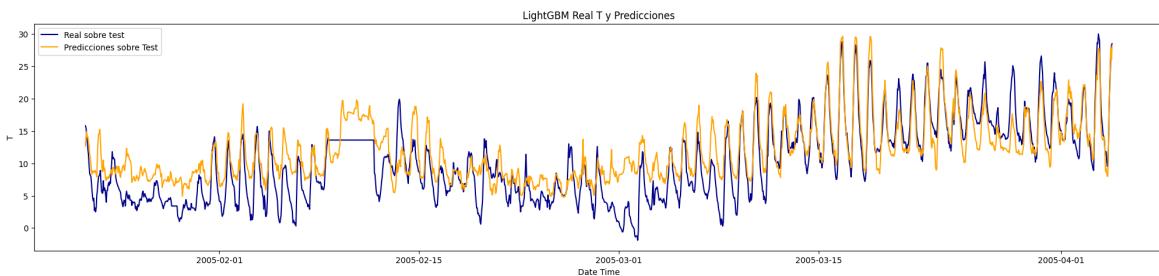


Figura 27. Valores reales de T (test) vs Valores predichos con Modelo 1 LighGBM.

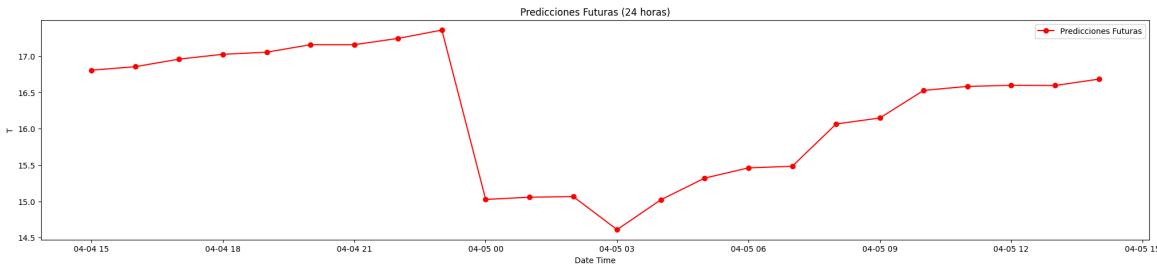


Figura 28. Pronóstico de 24 horas futuras para T con Modelo 1 LighGBM.

Modelo Light GBM para Serie Temporal Humedad Relativa (HR)

Los 4 modelos corridos no mostraron resultados significativamente diferentes entre sí. El modelo que se seleccionó como "mejor" fue el Modelo 1.

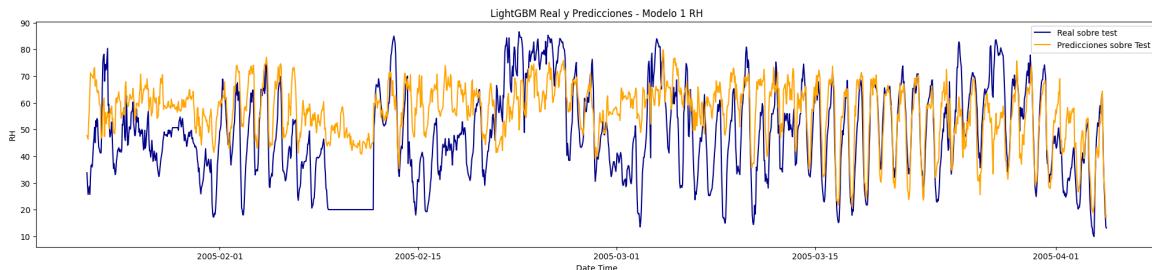


Figura 29. Valores reales de HR (test) vs Valores predichos con Modelo 1 LighGBM.

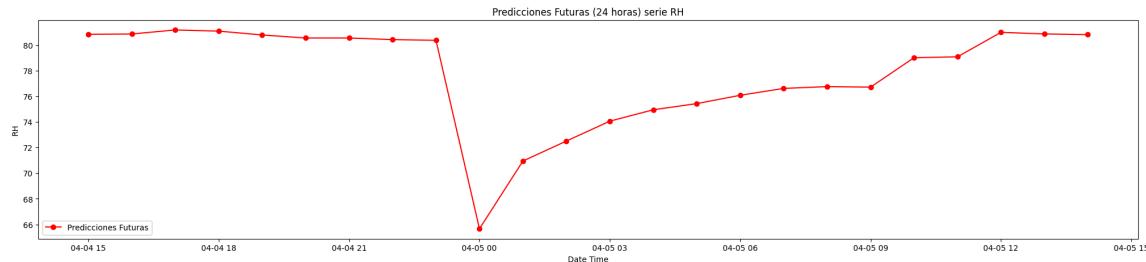


Figura 30. Pronóstico de 24 horas futuras para HR con Modelo 1 LighGBM.

Modelos de Deep Learning

El Deep Learning es una rama del Machine Learning enfocada en crear modelos basados en redes neuronales que permiten aprender representaciones no lineales de manera jerárquica. Se trata de un aprendizaje profundo, precisamente, porque sus modelos aprenden muchas capas de transformaciones. Si bien esto puede parecer limitado, el Deep Learning ha dado lugar a una vertiginosa variedad de modelos,

técnicas, formulaciones de problemas y aplicaciones.⁶

Deep Learning ha sido ampliamente aplicado en el análisis de series temporales debido a su capacidad para capturar patrones complejos y no lineales en los datos. Algunas de las arquitecturas y técnicas más comunes utilizadas en el análisis de series temporales son:

- *Redes Neuronales Recurrentes (RNN)*: se diseñan redes neuronales para manejar datos secuenciales donde las conexiones entre las unidades forman un ciclo. Esto les permite mantener una “memoria” de los estados anteriores.
- *Long Short-Term Memory (LSTM)*: puede aprender dependencias a largo plazo mediante el uso de celdas de memoria que pueden retener información durante largos períodos.
- *Gated Recurrent Units (GRU)*: similar a las LSTM pero con una estructura más simple y menos parámetros, lo que las hace más eficientes en términos computacionales.
- *Redes Neuronales Convolucionales (CNN)*: redes que utilizan convoluciones para procesar datos, originalmente diseñadas para imágenes pero adaptadas para series temporales.
- *Transformers*: arquitecturas basadas en mecanismos de atención que pueden procesar secuencias de datos en paralelo y aprender dependencias a largo plazo sin la necesidad de procesar los datos en orden secuencial.

LSTM

El LSTM (Long Short-Term Memory) es un tipo de red neuronal recurrente (RNN) que está diseñada para manejar secuencias de datos y abordar el problema del desvanecimiento del gradiente que afecta a las redes neuronales tradicionales. Fue introducido por Hochreiter y Schmidhuber en 1997. Dentro de las aplicaciones más efectivas de este algoritmo se pueden encontrar tareas que involucran dependencias a largo plazo, como es el caso de las series temporales, procesamiento de lenguaje natural, la traducción automática y la generación de texto.

Dentro de las ventajas que tiene este algoritmo, se puede destacar la capacidad para recordar patrones de datos a largo plazo, lo cual es crucial en series temporales donde las dependencias entre eventos pueden abarcar grandes períodos; esto lo logra

⁶ A. ZHang. Z. C. Lipton, M. Li, A. J. Smola. (2023). “Dive into Deep Learning”. Cambridge University Press. <https://d2l.ai/index.html>

mediante las celdas de memoria. Las LSTM están diseñadas para superar el problema del desvanecimiento del gradiente; lo que les permite aprender de secuencias más largas sin perder información importante.

Las LSTM utilizan 3 puertas principales para controlar el flujo de información:

- *Puerta de Entrada (Input)*: controla cuánta información nueva se almacena en la celda de memoria.
- *Puerta de Olvido (Forget)*: decide cuánta información de la celda de memoria se olvida.
- *Puerta de Salida (Output)*: controla cuanta información de la celda de memoria se usa para calcular la salida de la LSTM.

Modelo LSTM para Serie Temporal Benceno (C6H6)

Para la serie C6H6 se analizaron 10 modelos univariados con valores de output_length de 1 (unistep) y de 3, 6, 12, 24 (multistep). Los input_length fueron de 12, 24 y 48. El mejor modelo corresponde al número 8. En este modelo se utilizó un input_length de 12 y un output_length de 1, lo que resultó en las mejores métricas. Se podría pensar que un input_length de 24 era el más adecuado debido a la estacionalidad marcada de esta serie cada 24 horas; sin embargo, el input_length de 12 ha dado mejores resultados porque estaría capturando mejor la estacionalidad, ya que los patrones a corto plazo son más predecibles en este caso. Los patrones largos podrían incluir datos menos relevantes que pueden llegar a confundir al modelo. La predicción de 1 hora es 12.103, lo cual es razonable ya que el valor anterior es 11.9.

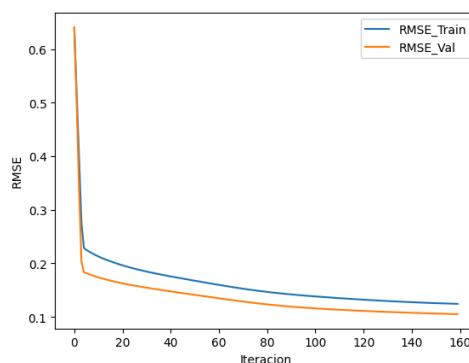


Figura 31. Gráfico de Curva de Aprendizaje para modelo 8 LSTM para serie C6H6.

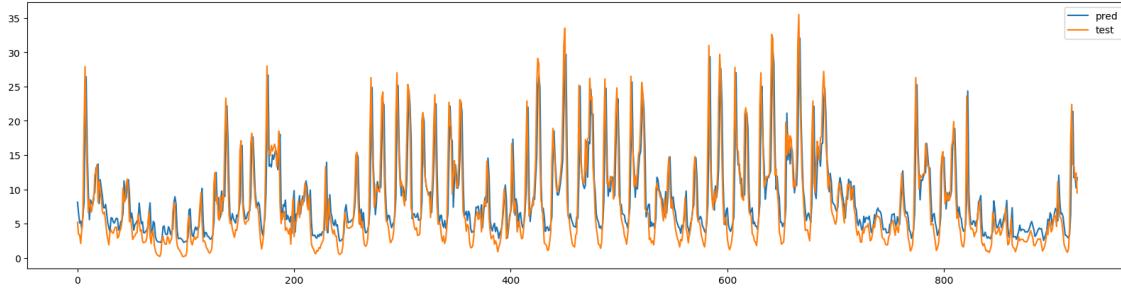


Figura 32. Valores reales de C6H6 (test) vs Valores predichos con Modelo 8 LSTM.

Para la predicción de las 24 horas siguientes se tomó el modelo 10 con un output_length de 24:

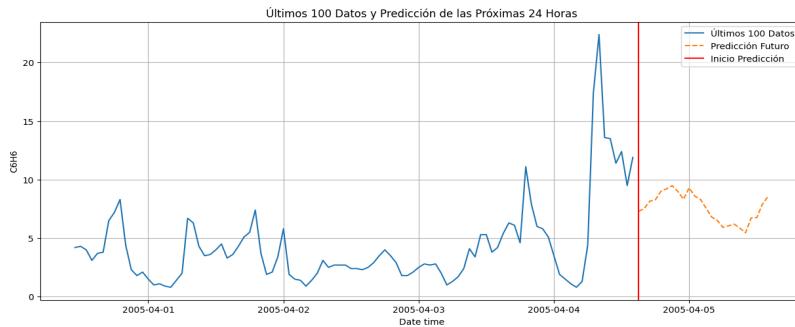


Figura 33. Últimos 100 días y Forecast de 24 horas para C6H6 con Modelo 8 LSTM.

Modelo LSTM para Serie Temporal Temperatura (T)

Para la serie Temperatura se analizaron 8 modelos univariados, modificando los hiperparametros mencionados anteriormente. Todos los modelos probados, dieron resultados bastante buenos, pero el Modelo 6 fue el que resultó tener la mejor performance.

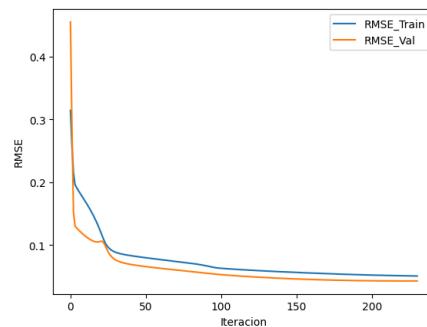


Figura 34. Gráfico de Curva de Aprendizaje para modelo 6 LSTM para serie T.

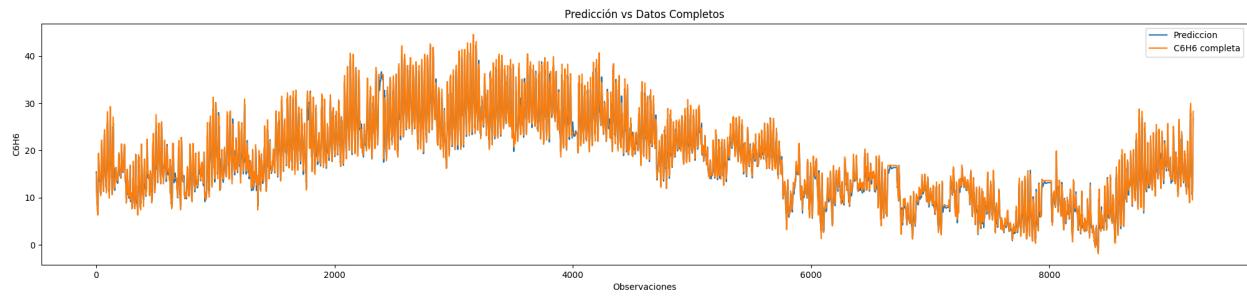


Figura 35. Valores reales de T (serie completa) vs Valores predichos con Modelo 6 LSTM.

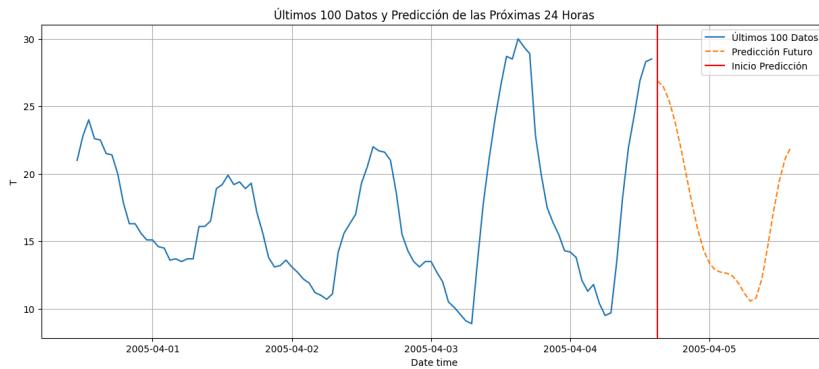


Figura 36. Últimos 100 días y Forecast de 24 horas para T con Modelo 6 LSTM.

Modelo LSTM para Serie Temporal Humedad Relativa (HR)

Se evaluaron 8 modelos univariados, ajustando los hiperparámetros previamente indicados. Se destacó el Modelo 6 ya que obtuvo el mejor desempeño.

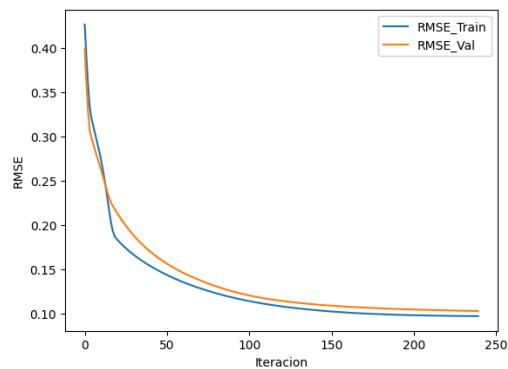


Figura 37. Gráfico de Curva de Aprendizaje para modelo 6 LSTM para serie HR.

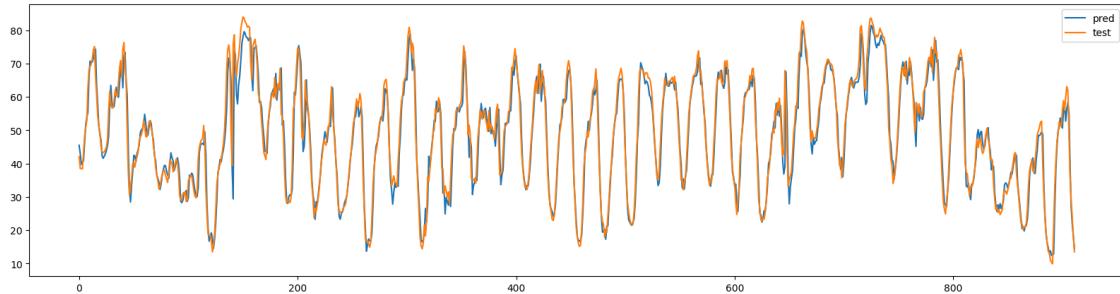


Figura 38. Valores reales de HR (test) vs Valores predichos con Modelo 6 LSTM.

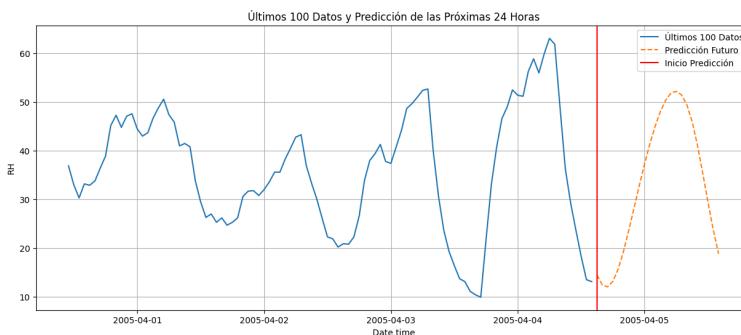


Figura 39. Últimos 100 días y Forecast de 24 horas para HR con Modelo 6 LSTM.

Modelos Híbridos

Los modelos híbridos combinan diferentes técnicas y enfoques para aprovechar las fortalezas de cada algoritmo y mejorar el rendimiento de los mismos en el análisis de series de tiempo. Estos modelos son especialmente útiles cuando se trata de analizar datos complejos que presentan tanto patrones lineales como no lineales, o cuando se desea capturar diferentes tipos de dependencias y características en los datos.

Actualmente, son muchos los modelos que existen para el análisis de series temporales; que buscan la combinación de herramientas tanto estadísticas como de Machine Learning y Deep Learning. Y se sabe, que siguen apareciendo algoritmos cada vez más poderosos. El desafío es encontrar el adecuado para los datos con lo que se trabaja, procesarlos de la manera correcta, encontrar los parámetros para el algoritmo adecuado y así poder lograr el mejor desempeño para la predicción de datos futuros.

Neural Prophet

El modelo Neural Prophet combina técnicas de Machine Learning con métodos tradicionales de modelado de series temporales. Está basado en el modelo Prophet propuesto por Facebook y la arquitectura de redes neuronales (LSTM) para mejorar la capacidad de capturar patrones complejos en los datos. Está diseñado para facilitar el modelado de series temporales de manera automatizada; es decir, sin necesidad de ajustar demasiados parámetros manualmente. Tiene la capacidad de capturar estacionalidades diarias, semanales y anuales, así como tendencias a largo plazo. Otra de sus características, es que permite la incorporación de eventos (como vacaciones, o eventos especiales) que pueden afectar los datos de la serie temporal. Proporciona flexibilidad para ajustar los componentes del modelo según las necesidades específicas de los datos a analizar.

El modelo Neural Prophet está compuesto de distintos módulos; donde cada uno aporta un componente aditivo al pronóstico. La mayoría de los componentes también se pueden configurar para que se escalen según la tendencia para lograr tener un efecto multiplicativo. Cada módulo tiene sus entradas individuales y proceso de modelado. Sin embargo, todos los módulos deben producir h resultados (h define el número de pasos que se pronosticaron en el futuro). Estos se suman como los valores predichos $\hat{y}_t, \dots, \hat{y}_{t+h-1}$ para los valores futuros de la serie temporal y_t, \dots, y_{t+h-1} . Si el modelo solo depende del tiempo, se puede producir un número arbitrario de pronósticos donde $h=1$. Matemáticamente el modelo Neural Prophet se define así:

$$\hat{y}_t = T(t) + S(t) + E(t) + F(t) + A(t) + L(t)$$

Donde:

- $T(t)$ = tendencia en el tiempo t .
- $S(t)$ = efectos estacionales en el tiempo t .
- $E(t)$ = efectos de eventos y días festivos en el momento t .
- $F(t)$ = efectos de regresión en el momento t para variables exógenas conocidas en el futuro.
- $A(t)$ = efectos de autoregresión en el momento t basados en observaciones del pasado.
- $L(t)$ = efectos de regresión en el momento t para observaciones rezagadas de variables exógenas.

Todos los módulos de componentes del modelo se pueden configurar y combinar individualmente para componer el modelo. Si todos los módulos están desconectados,

solo se instala un parámetro de compensación estático como componente de tendencia. Por defecto, sólo están activados los módulos de tendencia y estacionalidad.⁷

Modelo Neural Prophet para Serie Temporal Benceno (C6H6)

En total se evaluaron 22 modelos. El mejor modelo fue aquel al que se incorporó el rezago 12. Se realizó un grid search con distintas opciones de changepoints_range y n_changepoints, pero los resultados fueron similares y no mejoraron respecto a los obtenidos con los parámetros por default. Al igual que en Prophet, los parámetros por default funcionan bien, al menos para series con pocas complicaciones.

Al agregar las dos variables exógenas T y RH, el modelo mejoró. Además, las variables exógenas T y RH aportan más información juntas que individualmente, lo que puede deberse a que, de alguna manera, estas variables están relacionadas.

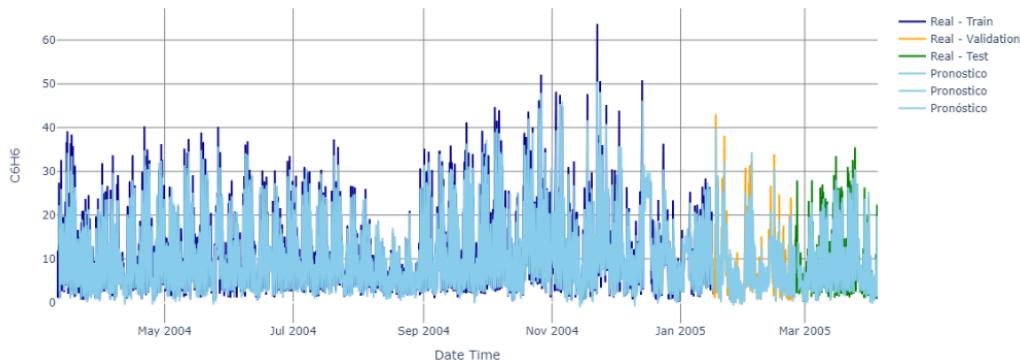


Figura 40. Pronóstico de C6H6 con Modelo 6 de Neural Prophet.

Modelo Neural Prophet para Serie Temporal Temperatura (T)

Para esta serie temporal, se evaluaron un total de 7 modelos, con modificaciones en los hiperparametros previamente mencionados, añadiendo variables exógenas C6H6, HR y ambas. El modelo con mejor performance, resultó ser el Modelo 2. Con un rezago de 24 y sin variables exógenas.

⁷ O. Triebel, H. Hewamalage, P. Pilyugina, N. Laptev, C. Bergmeir, Rm. Rajagopal. (2021). “NeuralProphet: Explainable Forecasting at Scale” Satnfor University. <https://arxiv.org/pdf/2111.15397>

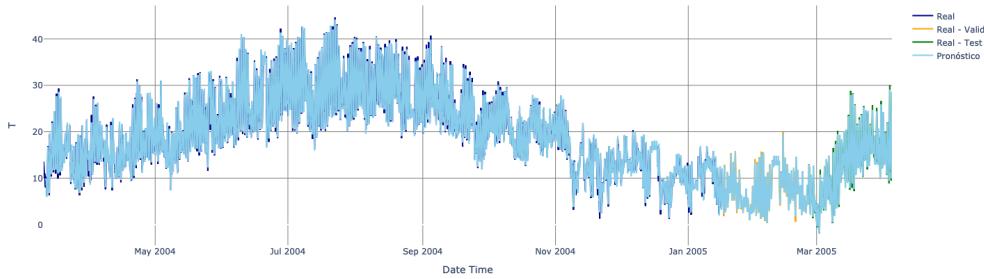


Figura 41. Pronóstico de T con Modelo 2 de Neural Prophet.

Modelo Neural Prophet para Serie Temporal Humedad Relativa (HR)

El modelo más destacado fue aquel al que se le incorporaron las dos variables exógenas T y C6H6.

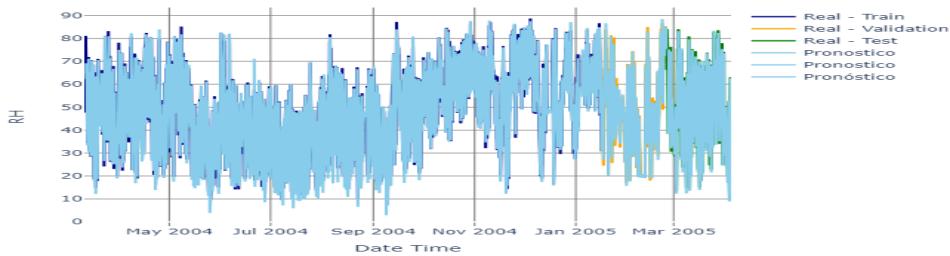


Figura 41. Pronóstico de HR con Modelo 6 de Neural Prophet.

H2O AutoML

H2O es una plataforma que al simplificar y automatizar la creación de modelos y aplicaciones de inteligencia artificial, logra ponerla al alcance de todos.⁸ Esta plataforma de aprendizaje automático automatiza el proceso de entrenamiento y ajuste de modelos de Machine Learning. H2O AutoML, puede ser utilizado para una variedad enorme de tareas, incluido el análisis de series temporales. Dentro de las principales características de este modelo, se pueden mencionar:

- Automatiza la selección de características, la construcción de modelos, la optimización de hiperparámetros y la validación cruzada.
- Entrena una variedad de modelos, incluyendo: árboles de decisión,

⁸ J. Lerdo de Tejada. (2021). “Inteligencia Artificial fácil y rápida con H2O.ai”. <https://www.youtube.com/watch?v=AEfribPXdIE&t=201s>

bosques aleatorios, modelos de gradient boosting, modelos lineales y redes neuronales.

- Construye ensambles de modelos para mejorar el rendimiento.
- proporciona una interfaz simple y directa a través de su API para Python y R.
- Además de retrasos, se pueden incluir personalizaciones en los datos para incluir características adicionales como días de la semana, vacaciones u otros eventos relevantes.

H2O AutoML es una herramienta poderosa para el análisis de series temporales, especialmente cuando se combinan sus capacidades automáticas con técnicas adecuadas de preprocesamiento y validación. La clave está en transformar la serie temporal en un formato adecuado para el aprendizaje supervisado, lo que permite a H2O AutoML explorar una variedad de modelos y encontrar la mejor combinación para los datos específicos.

Modelo H2O Auto ML para Serie Temporal Benceno (C6H6)

Se analizaron dos entrenamientos de la serie C6H6. En el segundo, se amplió el tiempo de ejecución, la cantidad de modelos y se incluyeron las variables exógenas T y RH.

En el gráfico de residuos, se observa un patrón: los residuos deberían estar distribuidos aleatoriamente alrededor de cero; en este caso, se observa un patrón, lo que podría indicar que el modelo no está capturando adecuadamente la variabilidad.

Los resultados de las métricas de ambos modelos fueron prácticamente similares, al igual que el algoritmo seleccionado: Gradient Boosting. El gráfico de predicción sobre el conjunto de prueba se ve bien; sin embargo, en ambos casos se observa sobreajuste: la curva de entrenamiento sigue bajando mientras que la curva de prueba se mantiene horizontal. También en el gráfico de residuos se observa un patrón. Para poder utilizar alguno de estos dos modelos, se deberían entrenar con distintos valores de los hiperparámetros de Gradient Boosting para evitar el sobreajuste.

Los gráficos que se adjuntan son del segundo entrenamiento.

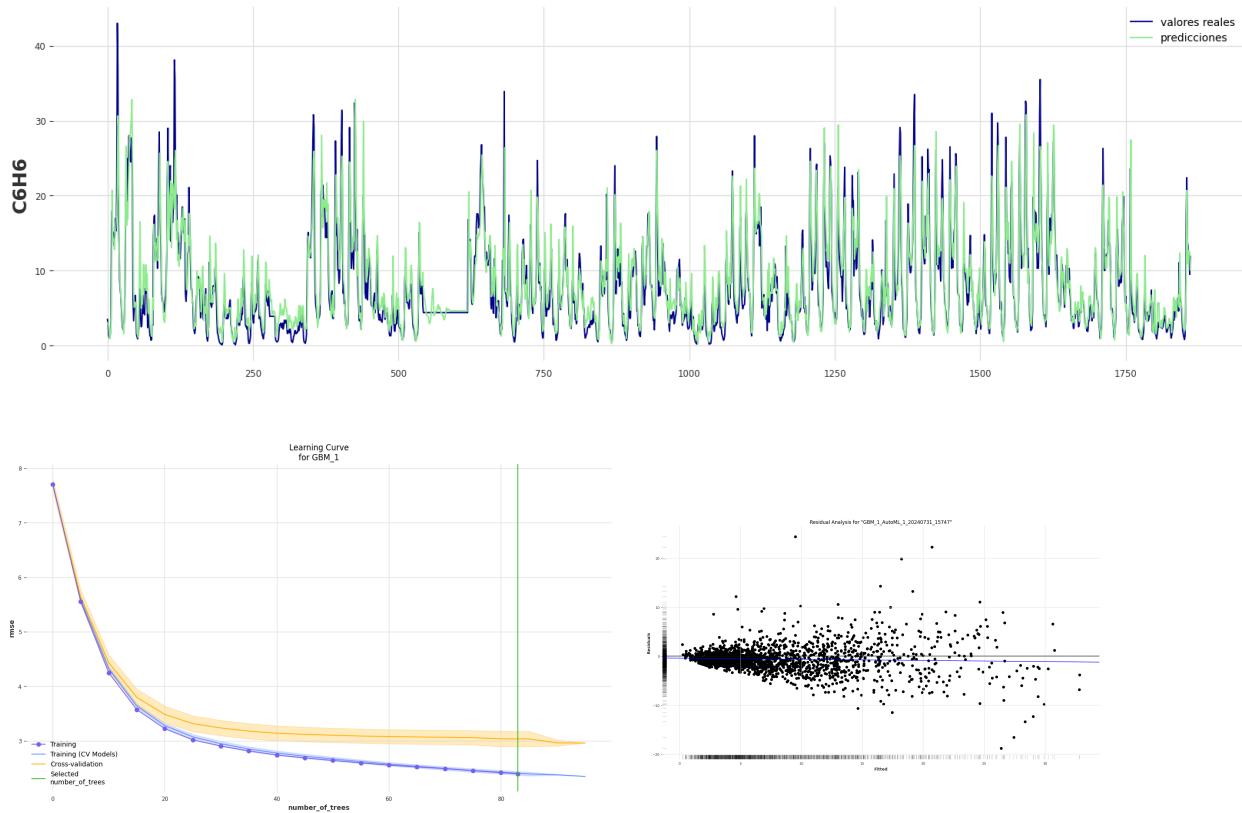


Figura 42. Segundo Entrenamiento del Modelo H2O AutoML para la serie C6H6.

AutoTS

AutoTS es un paquete de Python diseñado para implementar rápidamente pronósticos de alta precisión para series temporales. En 2023, AutoTS ganó el concurso de M6 forecasting, consiguiendo obtener el mejor desempeño como modelo predictivo para 12 meses en el mercado de valores.⁹

Su objetivo es automatizar muchas de las tareas complejas asociadas con el análisis de series temporales, como la selección de modelos, el ajuste de hiperparámetros y la validación del modelo. Dentro de las principales características que lo diferencian de otros modelos híbridos, se pueden mencionar:

- Despues de probar con distintos modelos de series temporales, selecciona la mejor opción dependiendo de la métrica de rendimiento especificada.
- automatiza la búsqueda de los mejores hiperparametros para cada modelo.
- Incluye herramientas para manejar datos faltantes, normalización y

⁹ C. Caitlin. (2024). “AutoTs”. <https://pypi.org/project/autots/#description>

creación de características.

- Permite la creación de modelos híbridos y ensambles, buscando maximizar el rendimiento del mismo.
- Soporta una amplia variedad de modelos, incluyendo ARIMA, Prophet, redes neuronales y modelos de machine learning como Random Forest y XGBoost.
- Proporciona una interfaz sencilla para entrenar, evaluar y predecir con modelos de series temporales.

Modelo Auto TS para Serie Temporal Benceno (C6H6)

Se analizaron dos entrenamientos de la serie C6H6. En el segundo se incluyeron las variables exógenas T y RH. No se pudo mejorar el tiempo de ejecución por falta de capacidad computacional.

Se puede observar gráficamente que los resultados no son satisfactorios, por lo que se recomienda probar con otros parámetros. Las líneas verdes que se ven en la gráfica representan las bandas de confianza. Al tener casi la misma amplitud que la serie, esto indica que no se trata de un buen modelo. También se observa que el límite inferior es igual a cero, lo cual se debe a que se especificó que no debería haber valores negativos de C6H6, ya que no tendrían sentido.

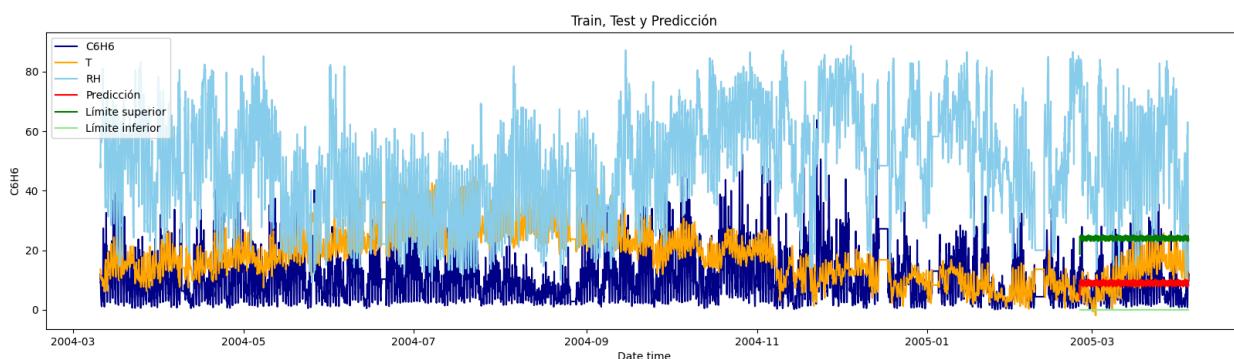


Figura 43. Modelo Auto TS para C6H6.

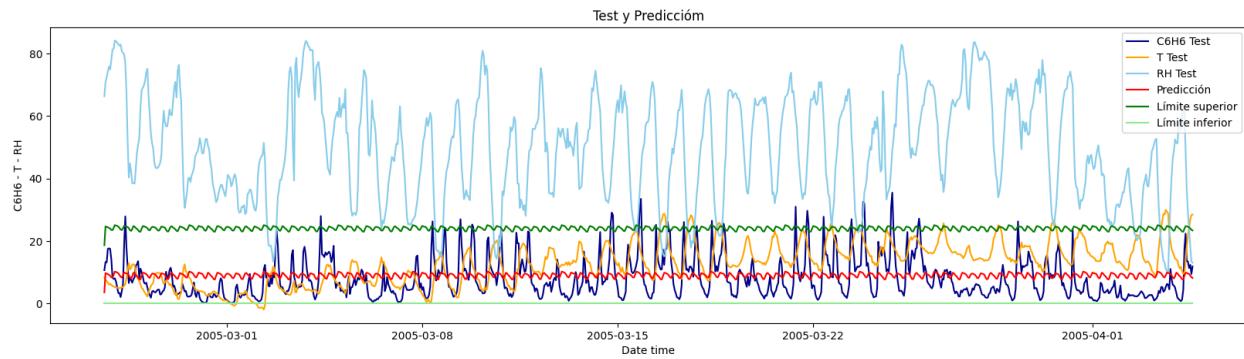


Figura 44. Conjunto de datos Test y Predicción para Modelo Auto TS de C6H6.

Análisis de Resultados

Fue necesario generar la función de RMSE, para medir el performance de los Modelos “ganadores” con la misma métrica y de esta manera, poder determinar cuál sería el mejor modelo para pronosticar cada una de las series analizadas. Para obtener la métrica “estándar” se utilizó la librería sklearn y se calculó el desempeño de cada modelo en base a las métricas: MAE, MSE y RMSE. A continuación los resultados.

Resultados Serie C6H6

	LSTM	Prophet	Neural Prophet	Neural Prophet	SVM	XGBoost	Light GBM
	Modelo 8	Modelo 1	Modelo 3	Modelo 6 (variables exógenas)	Modelo 5	Modelo 8	Modelo 1
MAE	2,53	6,4283	1,903	3,4363	0,4529	2,7307	3,6934
MSE	12,7508	55,5237	6,1362	23,1189	0,6036	14,3815	25,3799
RMSE	3,5708	7,4514	2,4771	4,8082	0,7769	3,7922	5,0378

Resultados Serie T

	LSTM	Neural Prophet	SVM	XGBoost	Light GBM
	Modelo 6	Modelo 2	Modelo 4	Modelo 7	Modelo 1
MAE	0,792017	0,8625	0,11231709	1,95548	2,8731
MSE	1,152880	1,1407	0,05533799	6,36247	12,2283
RMSE	1,073723	1,068	0,2352	2,52239	3,4969

Resultados Serie HR

	LSTM	Neural Prophet	SVM	XGBoost	Light GBM
	Modelo 6	Modelo 6	Modelo 3	Modelo 2	Modelo 1
MAE	2,8197	10,5843	0,3152	9,2906	13,3561
MSE	16,7806	200,02	0,3119	146,205	263,6228
RMSE	4,0964	14,1428	0,5584	12,0915	16,2364

Conclusiones

Los modelos automáticos pueden ser útiles para orientar sobre qué algoritmos utilizar, pero no siempre garantizan los mejores resultados. Es fundamental realizar un análisis detallado de los resultados obtenidos y considerar la posibilidad de reentrenar los modelos ajustando los hiperparámetros para mejorar el rendimiento.

En general, los mejores resultados tienden a obtenerse con algoritmos que emplean redes neuronales profundas (Deep Learning). Sin embargo, en ciertos casos, como cuando se trabaja con series temporales con un comportamiento predecible, los modelos de Machine Learning también pueden ofrecer resultados satisfactorios. Un ejemplo de esto es el modelo SVR (Support Vector Regression), que en este trabajo ha demostrado ser eficaz para las variables analizadas (C6H6, T, RH). Esto podría explicarse por la capacidad del SVR para aprender patrones locales, mientras que el LSTM está diseñado para capturar dependencias a más largo plazo. En el caso de estas tres series temporales, se observó un patrón local cada 24 horas, lo que hizo que el SVR fuera especialmente adecuado.

Los algoritmos de Machine Learning, como LightGBM y XGBoost, se destacan por ser modelos robustos, capaces de ofrecer predicciones más seguras. Aunque las métricas de desempeño no fueron las mejores entre los modelos probados, sus predicciones son confiables, gracias en parte a la validación cruzada utilizada para verificar la precisión de los datos.

Finalmente, es importante destacar la notable mejora de los modelos experimentados en este segundo trabajo práctico en comparación con los modelos de predicción estadística utilizados en el primer trabajo práctico. Los modelos de Machine Learning y Deep Learning demostraron predecir con un mayor nivel de confianza los datos analizados, superando a los modelos estadísticos como ARIMA.

Bibliografía

- Fuente del dataset utilizado: <https://archive.ics.uci.edu/dataset/360/air+quality>
- S. De Vito, E. Massera, M. Piga, L. Martinotto, G. Di Francia. (2008). "On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario". ScienceDirect. <https://www.sciencedirect.com/science/article/abs/pii/S0925400507007691?via%3Dihub>
- Peña, D. (2010). Análisis de Series Temporales. Madrid: Alianza.
- G.E.P. Box and G.M. Jenkins. "Time Series Analysis: Forecasting and Control". Holden-Day Inc. 1970.
- C.J.C. Burges. "A tutorial on support vector machines for pattern recognition". Data Mining and Knowledge Discovery. Vol. 2 Nº 2, pp. 121-167. 1998.
- Amat Rodrigo, J., & Escobar Ortiz, J. (2023). skforecast (Version 0.13.0) [Computer software]. <https://doi.org/10.5281/zenodo.8382787>
- F. Ballesteros López. (2023). "Estructuras Híbridas para el modelado y pronóstico de series temporales: metodologías y aplicaciones". Universidad de Córdoba. <https://repositorio.unicordoba.edu.co/server/api/core/bitstreams/90793c08-236c-4d92-927f-c282328d9324/content>
- O. Triebel, H. Hewamalage, P. Pilyugina, N. Laptev, C. Bergmeir, Rm. Rajagopal. (2021). " NeuralProphet: Explainable Forecasting at Scale" Stanford University. <https://arxiv.org/pdf/2111.15397>
- Chat GPT.
- Material de Clase (2024). Del Rosso, Rodrigo. Análisis de Series Temporales. Universidad Austral.

Anexo

Modelo Chronos para la Serie Temporal C6H6 incluidas las variables exógenas T y RH

Chronos es una familia de modelos de predicción de series temporales pre entrenados basados en arquitecturas de modelos de lenguaje. Una serie temporal se transforma en una secuencia de tokens mediante escalamiento y cuantificación, y un modelo de lenguaje se entrena en estos tokens. Una vez entrenados, se obtienen pronósticos probabilísticos mediante el muestreo de múltiples trayectorias futuras dado el contexto histórico. Los modelos de Chronos se han entrenado en un gran corpus de datos de series temporales disponibles públicamente.

Se aplicó el modelo Chronos sobre la serie C6H6, con la opción de un pre entrenamiento -t5-large con 710 Millones de parámetros.

La longitud de ventana de pronóstico recomendada para Chronos es de 64 puntos de datos. Por lo tanto, se utilizó un método de ventana deslizante para evaluar el rendimiento del modelo.

Debido a que el modelo no tiene un buen rendimiento para valores mayores a 64, no fue posible separarlo en conjuntos de entrenamiento y prueba de modo habitual. Por lo tanto, como una forma de conocer la confianza de las predicciones, se evaluó el modelo en las últimas 24 horas de las observaciones.

Como se puede observar en el segundo gráfico, los valores generados por el modelo logran seguir la tendencia de la serie real. Sin embargo, en gran parte de la ventana de 24 horas, el modelo no logra predecir correctamente y muestra un desfase con respecto a los valores observados.

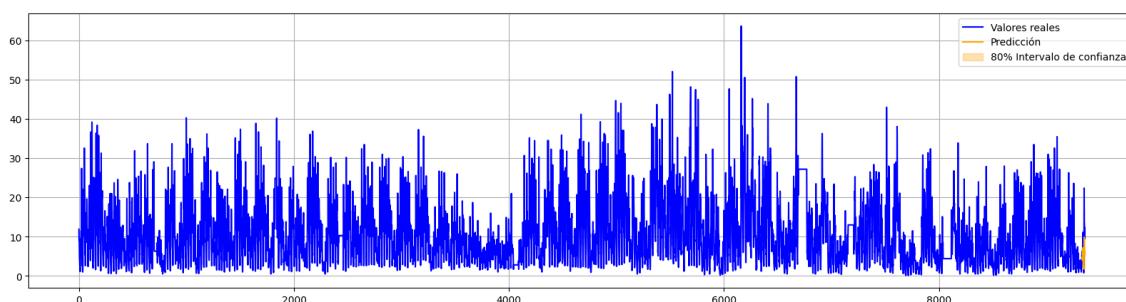


Figura 1. Modelo Chronos para C6H6.

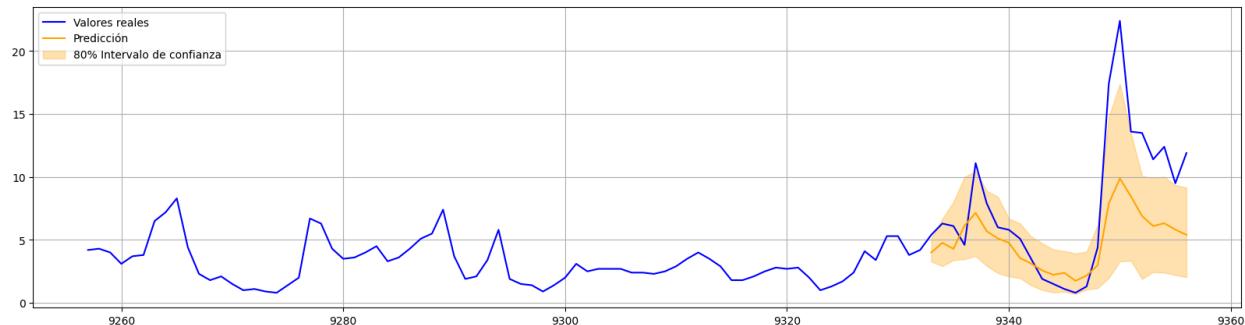


Figura 2. Modelo Chronos para los últimos 100 datos y predicción para C6H6.

Bibliografía del Anexo:

P. Joshi (2024). “*Tokenized Language Models for Time-Series Forecasting*”. Medium.
<https://medium.com/@joshi.pranjal5/tokenized-language-models-for-time-series-forecasting-efe5f7aa44e2>

Abdul Fatir Ansari. (2024). “*Chronos: Learning the Language of Time Series*”.
<https://arxiv.org/abs/2403.07815>

Enlace a los Scripts

En este enlace, se accede a todos los scripts realizados para probar los Modelos de este trabajo:

https://drive.google.com/drive/folders/1ylgP9Rd6Wn_Cbi_IPNq4v9TiGXR2_dyk?usp=share_linkx