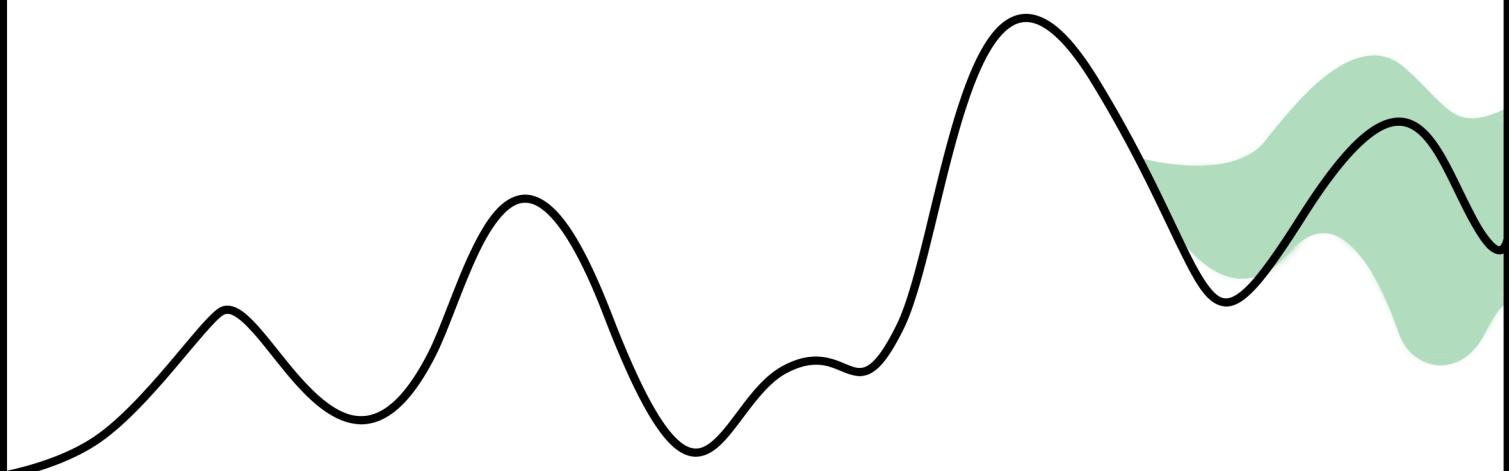


Comparación del desempeño de modelos estadísticos tradicionales, de aprendizaje automático y aprendizaje profundo para la predicción de series temporales

Tesina de grado



Alumno: Roncaglia Andrés Iván

Directora: Mag. Méndez Fernanda

Carrera: Licenciatura en Estadística



UNR Universidad
Nacional de Rosario

Agradecimientos

Este trabajo no solo es el cierre de mi paso por la carrera sino también de una de las mejores etapas de mi vida. Estuvo llena de crecimiento, esfuerzo y alegría. Decir que aprendí mucho académicamente es desmerecer todos estos años, porque también estuvieron acompañados de enseñanzas sobre la vida misma. Y como crecer no es algo que uno pueda lograr sin ayuda de nadie, está claro que debo agradecer a un gran puñado de personas.

En primer lugar a mis papás, que me dejaron venir desde lejos a pesar de que no les gustara separarse, que hicieron que no tenga que preocuparme por llegar a fin de mes, y que nunca me presionaron ni me exigieron nada.

A mis hermanos, al que tengo lejos por darme siempre mensajes de ánimo y preocuparse de que no me faltara nada, y al que tengo cerca por ser un gran compañero de piso a pesar de mis constantes quejas.

A mi novia le tengo que agradecer por estar a mi lado, en las buenas y en las malas, por levantarme el ánimo cuando me iba mal o celebrar juntos si me iba bien, por hacerme empujar siempre para adelante y obligarme a nunca bajar los brazos.

Uno de mis más grandes agradecimientos tiene que ir a todos los amigos que hice durante la carrera, tanto para con los que compartí los primeros años como para con los que compartí los últimos. Sin ellos probablemente me hubiera rendido mil y una veces, pero gracias a saber que no estaba solo y que siempre había alguien dispuesto a ayudar pude seguir adelante y llegar hasta donde estoy hoy.

También debo agradecer a todos los profesores que tuve, que sin ellos nada de esto sería posible. Fueron los que me abrieron las puertas al conocimiento y me llenaron de curiosidad, aprendí muchísimo de cada uno de ellos y muchos se convirtieron en mi rol a seguir como profesional.

Muchas gracias especialmente a Nanda, mi directora, que fue ella quien decidió apostar por mí para desarrollar este tema que me encantó investigar. Siempre estuvo disponible cuando la necesité y movió todos los hilos para que mi trabajo se desarrollara de la forma más suave y rápida posible.

Por último pero no menos importante, muchas gracias a la Universidad Nacional de Rosario, y en especial a la Facultad de Ciencias Económicas y Estadística, que brinda el lugar, el material y las oportunidades para que uno pueda crecer sin techo alguno.

Este trabajo no es solo mío, tiene una pizca de todos los que me acompañaron estos duros pero increíbles años, muchas gracias a cada uno de ustedes.

Resumen

La predicción de series temporales desempeña un rol crucial en contextos como la salud, la economía, la energía y la gestión de recursos, donde anticipar el comportamiento futuro de una variable resulta fundamental para la toma de decisiones. Esta tesis compara el desempeño de distintos enfoques para el pronóstico de series temporales, incluyendo modelos estadísticos tradicionales (ARIMA, SARIMA), algoritmos de aprendizaje automático (XGBoost, LightGBM), redes neuronales recurrentes (LSTM) y modelos fundacionales preentrenados basados en transformadores (TimeGPT y Chronos).

La evaluación se llevó a cabo sobre tres series reales representativas, con diferentes estructuras temporales: (i) número mensual de atenciones por patologías respiratorias en un hospital pediátrico, (ii) empleo privado en el sector educativo, y (iii) temperatura horaria durante el mes de marzo. Para cada caso, se implementaron y ajustaron los modelos mencionados, comparando sus resultados mediante métricas puntuales (MAPE) y probabilísticas (*Interval Score*).

Los resultados muestran que no existe un modelo universalmente superior: mientras los modelos estadísticos ofrecen interpretabilidad y precisión, por otro lado son poco automatizables y requieren de un control manual que requiere tiempo y conocimiento. Los algoritmos de *boosting* presentan buena capacidad predictiva con bajo costo computacional, pero necesitan especial cuidado al seleccionar con qué características entrenar el modelo y no ofrecen intervalos probabilísticos de forma nativa. Las redes LSTM, aunque potentes, muestran sensibilidad al sobreajuste y son exigentes computacionalmente. Por su parte, los modelos fundacionales ofrecen una alternativa rápida para personas que no tienen profundos conocimientos en el pronóstico de series de tiempo, aunque son modelos poco personalizables y su estructura interna no siempre es accesible.

Este trabajo aporta evidencia empírica y conceptual para una selección informada de modelos de pronóstico en función del contexto, destacando el potencial de las herramientas recientes como TimeGPT y Chronos, así como la importancia de incorporar métricas probabilísticas y técnicas de *conformal prediction* para mejorar la estimación de la incertidumbre.

Palabras clave: series temporales, predicción, *conformal predictions*, aprendizaje automático, redes neuronales, *transformers*, TimeGPT, *interval score*, Chronos.

Índice

1. Introducción	1
2. Objetivos	2
2.1 Objetivo general	2
2.2 Objetivos específicos	2
3. Metodología	3
3.1 Conceptos básicos de series de tiempo	3
3.2 Modelos estadísticos tradicionales para series temporales	3
3.2.1 SARIMA	4
3.3 Modelos de aprendizaje automático	6
3.3.1 Introducción a árboles de decisión y ensamblado	6
3.3.2 Diferencias entre XGBoost y LightGBM	9
3.3.3 Intervalos de confianza en algoritmos de aprendizaje automático	10
3.4 Modelos de aprendizaje profundo	11
3.4.1 Introducción a redes neuronales	11
3.4.2 <i>Long Short Term Memory</i> (LSTM)	12
3.4.3 Modelos transformadores	15
3.4.4 Diferencias entre TimeGPT y Chronos	19
3.5 Métricas de evaluación	20
3.6 Selección de parámetros y validación del modelo	21
4. Aplicación	22
4.1 Series analizadas	22
4.2 Ajuste y evaluación de modelos	24
4.2.1 Modelización con ARIMA y SARIMAX	24
4.2.2 Modelos de aprendizaje automático	35
4.2.3 Redes neuronales (LSTM)	39
4.2.4 Modelos fundacionales (TimeGPT y Chronos)	40
4.3 Comparación de resultados y análisis final	43
5. Conclusiones	45
6. Mejoras y extensiones a la investigación	46
7. Bibliografía	47
8. Anexo	49
8.1 Gráficos estacionales	49
8.2 Salidas de modelos ARIMA	50
8.3 Comprobación de supuestos de modelos arima	53

1. Introducción

La predicción de valores futuros en series de tiempo es una herramienta clave en múltiples ámbitos, tales como la economía, el comercio, la salud, la energía y el medio ambiente. En estos contextos, anticipar el comportamiento de una variable permite mejorar la planificación, asignar recursos de forma más eficiente y reducir la incertidumbre.

Actualmente, la ciencia de datos se encuentra en una etapa de constante expansión e innovación, impulsada por la gran cantidad de datos generados diariamente, por lo que en un contexto creciente de complejidad y exigencia temporal, resulta conveniente contar con herramientas que faciliten y acorten los tiempos de trabajo. Si bien los métodos más conocidos para trabajar series de tiempo son precisos, requieren de amplios conocimientos para encontrar un buen ajuste. Además, los modelos tradicionales como ARIMA son difíciles de automatizar, mientras que los algoritmos de aprendizaje automatizado que se utilizan actualmente pueden demandar largos tiempos de entrenamiento. Frente a estas limitaciones, en los últimos años se han desarrollado modelos capaces de seleccionar de forma automática el mejor ajuste para una serie temporal dada, sin requerir entrenamiento previo ni conocimientos especializados en análisis de series de tiempo. Estos son los denominados modelos fundacionales preentrenados, tales como TimeGPT o Chronos.

Sin embargo, aún persisten interrogantes sobre el desempeño de estos nuevos modelos y la falta de acceso al código fuente de algunos de estos limita la posibilidad de auditar sus resultados o replicar su implementación. Por ello, esta tesina propone realizar una comparación sistemática de modelos de pronóstico para series de tiempo, abordando tres enfoques metodológicos: modelos estadísticos tradicionales, algoritmos de *machine learning* y modelos de aprendizaje profundo. El objetivo es evaluar su desempeño en distintos contextos, utilizando métricas como el porcentaje del error absoluto medio (MAPE) y el *Interval Score*, con el fin de analizar ventajas, limitaciones y potenciales usos de cada uno.

Este análisis busca aportar una mirada crítica e informada sobre el uso de nuevas tecnologías en la predicción de series de tiempo, contribuyendo a la toma de decisiones metodológicas más sólidas desde una perspectiva estadística.

2. Objetivos

2.1 Objetivo general

El objetivo de esta tesina es, en primer lugar, comparar la precisión, eficiencia y facilidad de pronosticar series de tiempo con distintos modelos, incluyendo enfoques estadísticos clásicos, algoritmos de *machine learning* y modelos de *deep learning*, analizando al mismo tiempo sus ventajas, limitaciones y condiciones de uso más apropiadas.

2.2 Objetivos específicos

- Implementar modelos clásicos de series de tiempo, como ARIMA y SARIMA, explicando y garantizando el cumplimiento de los fundamentos teóricos y supuestos que los sostienen.
- Aplicar modelos de aprendizaje automático supervisado, como XGBoost y LightGBM, explorando distintas configuraciones para garantizar el mejor ajuste.
- Desarrollar modelos de aprendizaje profundo, en particular redes LSTM, dando introducción a las redes neuronales y modelos de pronóstico más complejos.
- Realizar pronósticos con modelos fundacionales (TimeGPT, Chronos) y comprender su funcionamiento.
- Definir y aplicar métricas de evaluación (MAPE, *Interval Score*) para comparar el rendimiento de todos los modelos bajo un mismo conjunto de datos.
- Reflexionar valorativamente sobre los criterios de selección de modelos en función del contexto de aplicación, la complejidad computacional y la interpretabilidad de los resultados.

3. Metodología

El enfoque metodológico adoptado en esta tesina consiste en comparar el desempeño de distintos modelos de pronóstico aplicados a series temporales. Para ello, se seleccionan modelos representativos de tres enfoques principales: modelos estadísticos tradicionales, algoritmos de aprendizaje automático (*machine learning*) y modelos de aprendizaje profundo (*deep learning*).

El análisis se estructura en tres componentes fundamentales: una descripción conceptual de los modelos, su implementación práctica sobre series con diferentes características, y una evaluación cuantitativa comparativa a través de métricas de error.

3.1 Conceptos básicos de series de tiempo

Se denomina serie de tiempo a un conjunto de observaciones $\{z_1, z_2, \dots, z_t, \dots, z_n\}$ cuantitativas ordenadas en el tiempo, usualmente de forma equidistante, sobre una variable de interés. El análisis de series de tiempo tiene como objetivo sintetizar y extraer información estadística relevante, tanto para interpretar el comportamiento histórico de la variable como para generar pronósticos $\{z_{n+1}, \dots, z_{n+l}, \dots, z_{n+h}\}$.

Dado que las series temporales pueden exhibir diversos patrones subyacentes, resulta útil descomponerlas en componentes separadas, cada una de las cuales representa una característica estructural específica del comportamiento de la serie.

- Estacionalidad: corresponde a las fluctuaciones periódicas que se repiten a intervalos regulares de tiempo. Un ejemplo típico es la temperatura, que tiende a disminuir en invierno y aumentar en verano, repitiendo este patrón anualmente.
- Tendencia (o tendencia-ciclo): refleja la evolución a largo plazo de la media de la serie, asociada a procesos de crecimiento o decrecimiento sostenido. Por ejemplo, la población mundial exhibe una tendencia creciente a lo largo del tiempo.
- Residuos: representa las variaciones no sistemáticas que no pueden ser explicadas por la tendencia ni la estacionalidad. Estas fluctuaciones, que suelen deberse a eventos impredecibles o factores exógenos, se asumen como aleatorias.

Otro concepto importante en series de tiempo es la estacionariedad. Se dice que una serie es débilmente estacionaria si la media y la variancia se mantienen constantes en el tiempo y la correlación entre distintas observaciones solo depende de la distancia en el tiempo entre estas. Por comodidad, cuando se mencione estacionariedad se estará haciendo referencia al cumplimiento de estas propiedades.

3.2 Modelos estadísticos tradicionales para series temporales

Son llamados modelos estadísticos tradicionales a aquellos que surgen antes del auge del *machine learning* y los modelos de aprendizaje profundo. Son caracterizados por sus fuertes fundamentos estadísticos y su capacidad en capturar dependencias temporales en los datos.

3.2.1 SARIMA

Los modelos *ARIMA* (*AutoRegressive Integrated Moving Average*) son unos de los modelos de pronóstico tradicionales mejor establecidos. Son una generalización de los modelos autoregresivos (AR), que suponen que las observaciones futuras son función de las observaciones pasadas, y los modelos promedio móvil (MA), que pronostican las observaciones como funciones de los errores de observaciones pasadas. Además, estos modelos pueden adaptarse a series no estacionarias mediante la aplicación de diferenciaciones de orden d , las cuales implican restar a cada observación el valor registrado d periodos anteriores.

Formalmente un modelo $ARIMA(p, d, q)$ se define como:

$$\psi_p(B)(1 - B)^d z_t = \theta_0 + \theta_q(B)\alpha_t \quad (1)$$

Donde z_t es la observación t -ésima, $\psi_p(B)$ y $\theta_q(B)$ son funciones de los rezagos (B), correspondientes a la parte autoregresiva y promedio móvil respectivamente, d es el grado de diferenciación y α_t es el error de la t -ésima observación.

Se debe tener en cuenta estos aspectos importantes:

- Se dice que una serie es invertible si se puede escribir cada observación como una función de las observaciones pasadas más un error aleatorio. Por definición, todo modelo AR es invertible.
- Por definición, todo modelo MA es estacionario.
- $\psi_p(B) = 1 - \psi_1 B - \psi_2 B^2 - \dots - \psi_p B^p$ es el polinomio característico de la componente AR y $\theta_q(B) = 1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q$ de la componente MA. Si las raíces de los polinomios característicos caen fuera del círculo unitario, entonces un proceso AR se puede escribir de forma MA y es estacionario, y a su vez un proceso MA se puede escribir de forma AR y es invertible.
- Un proceso *ARIMA* es estacionario e invertible si su componente AR y MA lo son respectivamente.

Sin embargo este tipo de modelos no tienen en cuenta la posible estacionalidad que puede tener una serie, es por esto que se introducen los modelos $SARIMA(p, d, q)(P, D, Q)_s$ que agregan componentes AR, MA y diferenciaciones a la parte estacional de la serie con período s .

Se denomina función de autocorrelación a la función de los rezagos, entendiendo por rezago a la distancia ordinal entre dos observaciones, que grafica la autocorrelación entre pares de observaciones. Es decir que para cada valor k se tiene la correlación entre todos los pares de observaciones a k observaciones de distancia. En su lugar, la función de autocorrelación parcial calcula la correlación condicional de los pares de observaciones, removiendo la dependencia lineal de estas observaciones con las que se encuentran entre estas. Estas funciones son necesarias para poder identificar los modelos *SARIMA* y se definen como:

$$\rho_k = Corr(z_t, z_{t+k}) = \frac{Cov(z_t, z_{t+k})}{\sqrt{Var(z_t) \cdot Var(z_{t+k})}} \quad (2)$$

y

$$\phi_{kk} = Corr(z_t, z_{t+k} | z_{t+1}, \dots, z_{t+k-1}) \quad (3)$$

Los modelos *AR* se caracterizan por tener autocorrelaciones significativas que decaen lentamente y autocorrelaciones parciales significativas únicas. Los modelos *MA* se comportan de forma

inversa, tienen autocorrelaciones significativas únicas y autocorrelaciones parciales que decaen progresivamente.

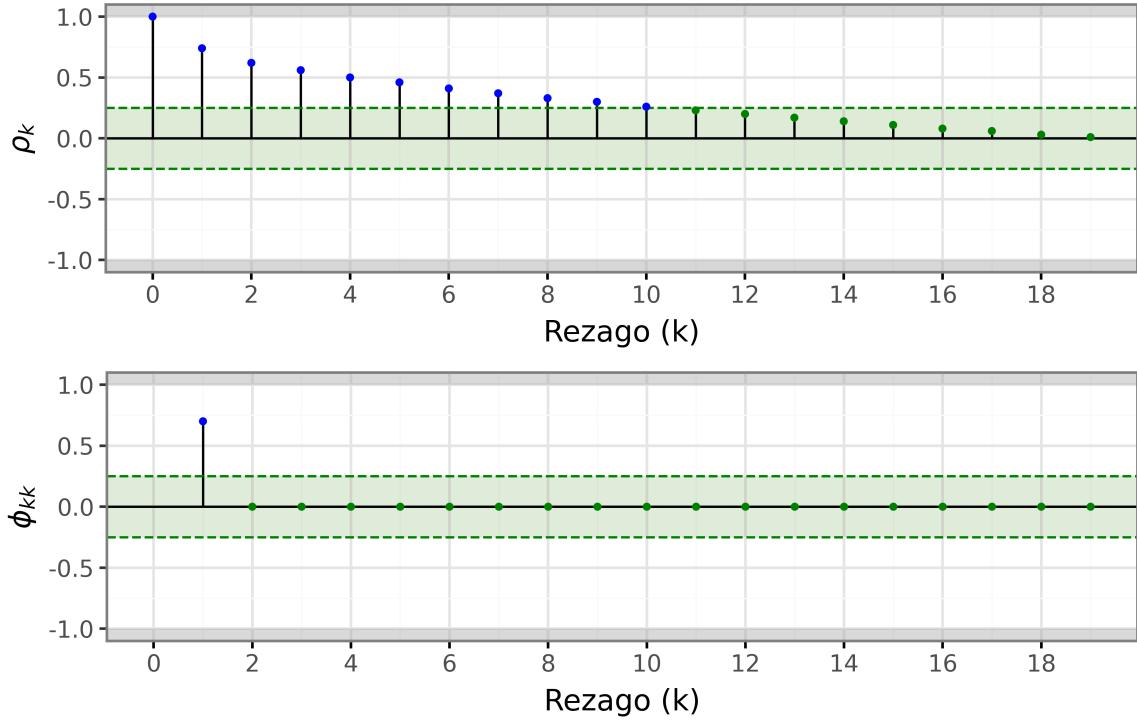


Figura 1: Ejemplo de las autocorrelaciones de un proceso $AR(1)$.

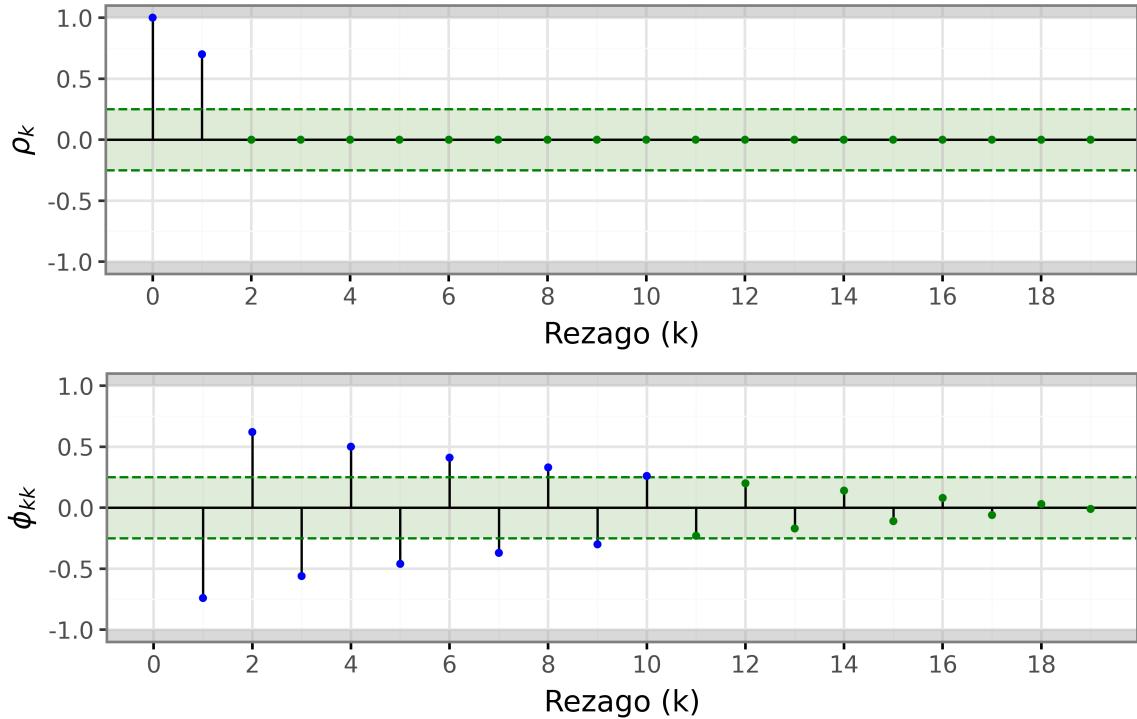


Figura 2: Ejemplo de las autocorrelaciones de un proceso $MA(1)$.

Para poder identificar un modelo $SARIMA$ a partir de las autocorrelaciones es necesario que la

serie sea estacionaria. Entre los *tests* para probar la estacionariedad de una serie se encuentran el *test* aumentado de Dickey-Fuller y el de Kwiatkowski-Phillips-Schmidt-Shin. El *test* aumentado de Dickey-Fuller plantea como hipótesis nula que existe una raíz unitaria, indicando que la serie no es estacionaria y es necesaria una diferenciación, ante la alternativa de que no existe una raíz unitaria. Por otro lado, en el *test* Kwiatkowski-Phillips-Schmidt-Shin la hipótesis nula indica que la serie es estacionaria alrededor de una constante.

Un buen modelo *SARIMA* debe cumplir las siguientes propiedades:

- Sus residuos se comportan como ruido blanco, es decir, están incorrelacionados y siguen una distribución normal, con media y variancia constantes.
- Es admisible, es decir, es invertible y estacionario.
- Es parsimonioso, en el sentido de que sus parámetros son significativos.
- Es estable en los parámetros, que se cumple cuando las correlaciones entre los parámetros no son altas.

Existen múltiples opciones para probar la normalidad de una distribución. En este trabajo se utiliza el *test* no paramétrico de Kolmogorov-Smirnov, cuya hipótesis nula sostiene que la muestra proviene de la distribución de referencia, que en este caso es la distribución normal. Para probar que los residuos no presentan correlación, se hace uso del *test* de Ljung-Box. La hipótesis nula de esta prueba indica que los datos no están correlacionados.

3.3 Modelos de aprendizaje automático

El aprendizaje automático (*machine learning*) es una rama de la inteligencia artificial que permite a las computadoras aprender de los datos y realizar tareas de forma autónoma. Aunque los métodos presentados no fueron diseñados específicamente para datos temporales, han demostrado ser útiles en múltiples contextos mediante diversas pruebas empíricas.

Los métodos de *machine learning*, a diferencia de los modelos tradicionales, se enfocan principalmente en identificar los patrones que describen el comportamiento del proceso que sean relevantes para pronosticar la variable de interés, y no se componen de reglas ni supuestos que tengan que seguir. Para la identificación de patrones, estos modelos requieren la generación de características.

3.3.1 Introducción a árboles de decisión y ensamblado

Los árboles de decisión pueden ser explicados sencillamente como un conjunto extenso de estructuras condicionales *if-else*. El modelo pronosticará un cierto valor x si una cierta condición es verdadera, u otro valor y si es falsa. Es importante ver que no hay una tendencia lineal en este tipo de lógica, por lo que los árboles de decisión pueden ajustar tendencias no lineales. El resultado que se obtiene al aplicar esta técnica puede resumirse gráficamente como un tronco con diferentes ramas, y de esta característica surge su nombre.

Un árbol puede tener distinta cantidad de divisiones en un mismo nivel, llamadas hojas, y profundidad, las cuales determinan en qué medida el modelo se ajusta a los datos con los que se entrena. Lógicamente árboles más profundos y con más hojas suelen generar sobreajuste, es decir, un modelo que se adapta demasiado a los datos de entrenamiento y generaliza erróneamente.

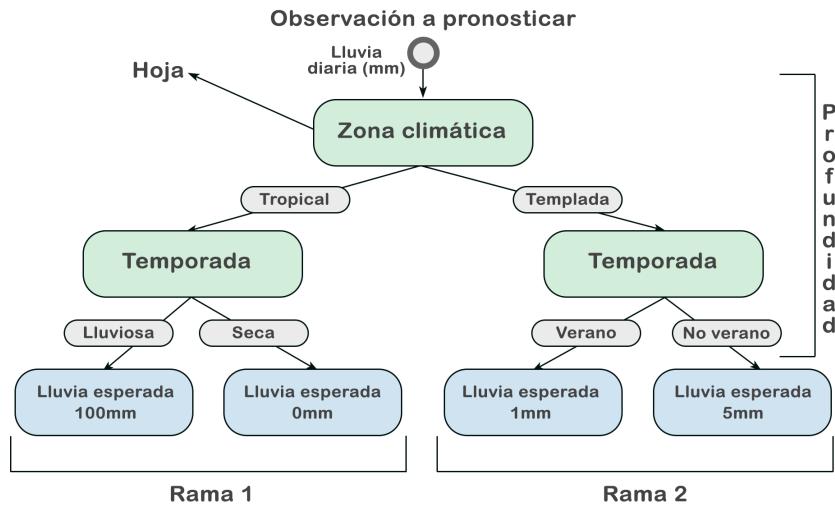


Figura 3: Ejemplo de árbol de decisión

Los métodos de ensamblaje buscan mejorar la robustez y precisión de las predicciones combinando los resultados de varios estimadores base, como los árboles de decisión. Los modelos no se construyen infinitamente, sino que se busca minimizar una función de pérdida que incluye una penalización por la complejidad del modelo, limitando así la cantidad de árboles que se producen. Existen múltiples métodos de ensamblaje (Bosques aleatorios, XGBoost, LightGBM, CatBoost, entre otros) que se diferencian en la forma en la que se construyen los árboles. En esta tesina se usarán los algoritmos *eXtreme Gradient Boosting* (XGBoost) y *Light Gradient-Boosting Machine* (LightGBM).

Se denomina *Boosting* a un proceso iterativo, que consiste en la construcción de árboles de forma secuencial donde cada nuevo árbol busca predecir los residuos de los árboles anteriores. Es así entonces que el primer árbol buscará predecir los valores futuros de la serie, mientras que el segundo intentará predecir los valores reales menos los pronosticados por el primer árbol, el tercero tratará de inferir la diferencia entre los valores reales y el valor pronosticado del primer árbol menos los errores del segundo, y así sucesivamente. En cada iteración se pesan los puntos y se corriguen aquellos que tengan un mayor error por medio del descenso del gradiente.

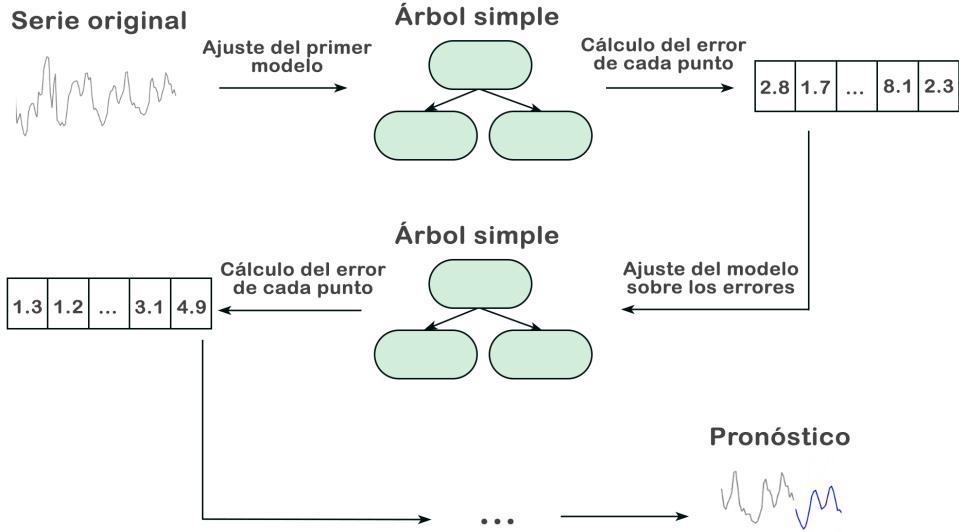


Figura 4: Proceso de ensamblado

La dirección de máximo crecimiento para una función está determinada por su gradiente, y por consiguiente, la dirección contraria es la dirección de máximo decrecimiento. El descenso del gradiente es un algoritmo iterativo en el que, con el objetivo de minimizar una función de pérdida, se calcula en cada paso la derivada de la función de costo con respecto a cada parámetro en su valor actual. Luego, se actualizan los valores de los parámetros desplazandolos una pequeña magnitud η , llamada “tasa de aprendizaje”, de forma tal que la función de pérdida decrezca.

Sea $L(x)$ una función de pérdida y θ_0 el valor actual de un parámetro del modelo, la actualización de dicho parámetro se realiza de la siguiente manera:

$$\theta_1 = \theta_0 - \eta \cdot \nabla L(\theta_0) \quad (4)$$

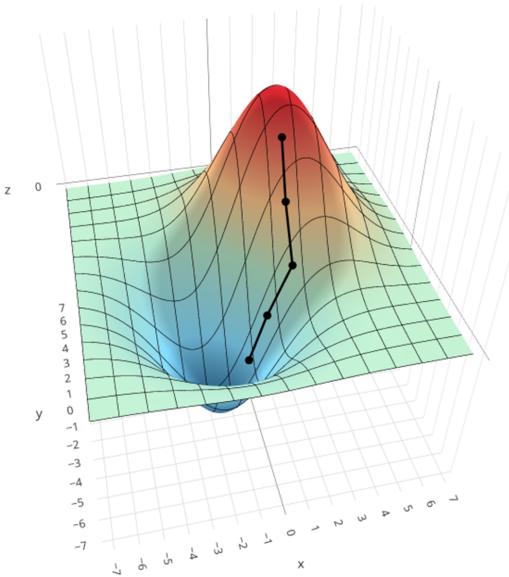


Figura 5: Ejemplo del descenso del gradiente en una función de pérdida

Este procedimiento iterativo se repite hasta lograr la convergencia o se llegue a un límite de iteraciones. Es por esto que la elección de η es sumamente importante para lograr el objetivo, ya que valores grandes podrían ocasionar divergencia, mientras que valores pequeños pueden llevar a que el algoritmo necesite demasiadas iteraciones para lograr la convergencia.

3.3.2 Diferencias entre XGBoost y LightGBM

Las diferencias entre XGBoost y LightGBM radican en la forma en que cada uno identifica las mejores divisiones dentro de los árboles y de que forma los hacen crecer.

XGBoost utiliza un método exacto en el que se calcula la ganancia de las particiones con cada característica, entendiendo por ganancia a la contribución relativa de una característica en el contexto de un árbol particular. Se elige la mejor partición y repite el proceso. Si bien este método es preciso, es lento computacionalmente. Por otro lado, LightGBM usa un método más eficiente llamado *Gradient-Based One-Side Sample* (GOSS). GOSS calcula los gradientes para cada punto y lo usa para filtrar afuera aquellos puntos que tengan un bajo gradiente. Que un punto tenga un gradiente bajo significaría que está “mejor pronosticado” que el resto, y por lo tanto no es necesario enfocarse tanto en mejorar dicho punto. Además, LightGBM utiliza un procedimiento que acelera el ajuste cuando se tienen muchas características correlacionadas de las cuales elegir, denominado *Exclusive Feature Bundle* (EFB).

A la hora de hacer crecer los árboles, XGBoost lo hace nivel a nivel, es decir que primero se crean todas las divisiones de un nivel, y luego se pasa al siguiente, priorizando que el árbol sea simétrico y tenga la misma profundidad en todas sus ramas. LightGBM, en cambio, se expande a partir de la hoja que más reduce el error, mejorando la precisión y eficiencia en series largas, pero arriesgándose a posibles sobreajustes si no se limita correctamente la profundidad de los árboles.

3.3.3 Intervalos de confianza en algoritmos de aprendizaje automático

Una de las principales ventajas de los modelos de aprendizaje automático respecto de los enfoques estadísticos tradicionales es que no requieren supuestos distribucionales estrictos sobre los errores. Sin embargo, esta flexibilidad también implica que, en general, no generan pronósticos probabilísticos de forma directa, lo que dificulta la construcción de intervalos de confianza.

Para subsanar esta limitación, se han desarrollado diversos métodos que permiten acompañar las predicciones puntuales con medidas de incertidumbre. A continuación, se describen dos de los enfoques más utilizados.

- Regresión cuantil con *boosting*: consiste en entrenar modelos para estimar directamente cuantiles de la distribución condicional de la variable objetivo, en lugar de su valor esperado. De este modo, pueden construirse intervalos de predicción utilizando, por ejemplo, los percentiles 5 y 95. Esta técnica está implementada en algunas variantes como LightGBM, pero no es directamente aplicable en XGBoost, lo que restringe su uso en ciertos entornos.
- *Conformal predictions*: se trata de una familia de métodos no paramétricos que permiten construir intervalos de predicción válidos bajo el supuesto de intercambiabilidad de las observaciones. Dado que este supuesto no se cumple en series temporales, donde existe dependencia temporal, se han desarrollado adaptaciones específicas para este tipo de datos.

Una de las más destacadas es el método *Ensemble Batch Prediction Intervals* (EnbPI), que permite aplicar *conformal predictions* en series temporales sin asumir independencia entre observaciones. Su procedimiento consiste en:

1. Seleccionar un modelo por ensamblado (como XGBoost o LightGBM).
2. Generar B muestras *bootstrap* por bloques, manteniendo así la estructura temporal de los datos.
3. Ajustar un modelo sobre cada una de las B muestras.
4. Para cada observación del conjunto de entrenamiento, calcular el residuo utilizando únicamente aquellos modelos que no la incluyeron.
5. Obtener las predicciones puntuales promediando los resultados de los B modelos.
6. Construir los intervalos de predicción sumando y restando los cuantiles empíricos de los residuos a las predicciones.

$$IC_{Z_{n+l};1-\alpha} = Z_n(l) \pm Q_{1-\alpha}(e) \quad (5)$$

Este método será el utilizado en esta tesina para estimar los intervalos de confianza en los algoritmos de aprendizaje automático y se aplica con la librería MAPIE.

3.4 Modelos de aprendizaje profundo

El *deep learning* (aprendizaje profundo) es una rama del *machine learning* que tiene como base un conjunto de algoritmos que intentan modelar niveles altos de abstracción en los datos usando múltiples capas de procesamiento, con complejas estructuras o compuestas de varias transformaciones no lineales.

Entre estos algoritmos se encuentran las redes neuronales, que imitan el funcionamiento del cerebro humano usando procesos que simulan la forma biológica en la que trabajan las neuronas para identificar fenómenos, evaluar opciones y llegar a conclusiones.

3.4.1 Introducción a redes neuronales

Una red neuronal está compuesta en grandes rasgos de 3 capas: entrada, oculta y salida. Dentro de cada capa se pueden encontrar neuronas y conexiones entre estas, donde cada neurona representa una variable y cada conexión un peso, y es por esto que a estas conexiones las llamaremos así en adelante. La suma de los pesos y las neuronas que no formen parte de la capa de entrada dan el total de parámetros que tiene que ajustar el modelo.

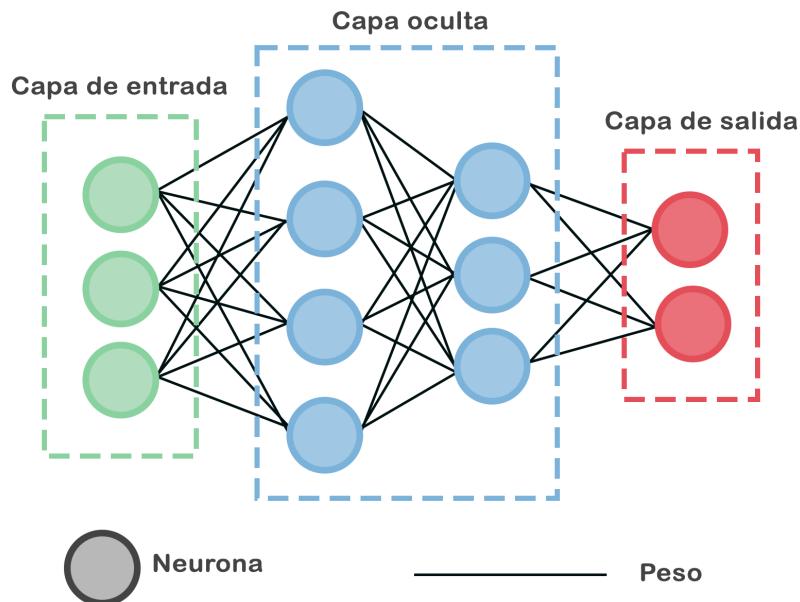


Figura 6: Ejemplo de red neuronal completamente conectada

En la capa de entrada se introducen las variables explicativas, y luego cada neurona fuera de esta capa es una función de las neuronas anteriores conectadas a la misma. Estas funciones son llamadas funciones de activación.

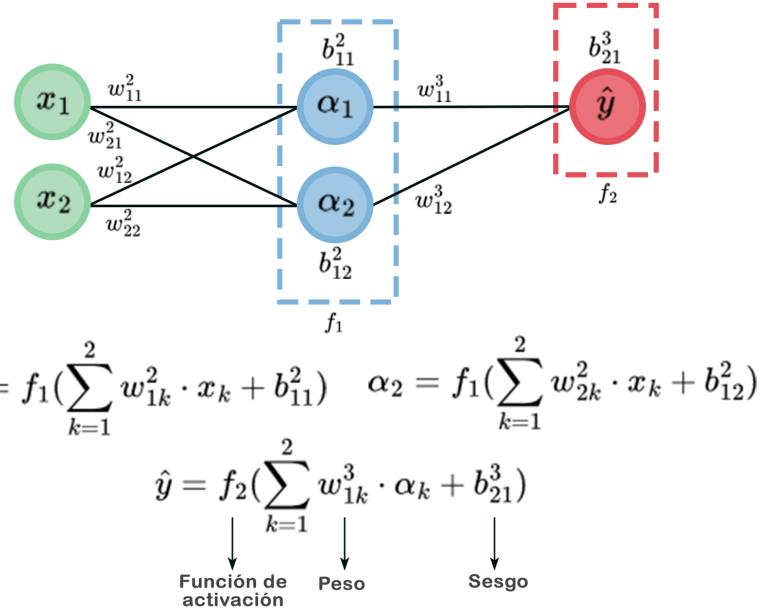


Figura 7: Obtención de una predicción en un red neuronal

Los parámetros se estiman buscando minimizar una función de costo, esto se logra con el descenso del gradiente y por medio de retropropagación. La retropropagación consiste en realizar una estimación inicial de la variable respuesta con los valores iniciales de la red neuronal, que pueden estar dados, por ejemplo, por una distribución normal, y de manera inversa a la dirección de la red neuronal calcular derivadas para encontrar la dirección de máximo decrecimiento de la función de costo para cada parámetro en la red neuronal.

Existen distintos tipos de redes neuronales según la forma en la que se conectan las neuronas. Las *Feedforward Neural Networks* (FNN) son las redes neuronales más comunes y simples. Las redes neuronales recurrentes, del inglés *Recurrent Neural Networks* (RNN) utilizan bucles de retroalimentación que las hacen especialmente buenas en la predicción de datos secuenciales. Otro tipo de red neuronal son las *Convolutional Neural Networks* (CNN), las cuales son útiles para el reconocimiento de patrones en los datos.

3.4.2 Long Short Term Memory (LSTM)

Lo que caracterizan a las redes neuronales recurrentes son los bucles de retroalimentación que se presentan en la figura 8. Mientras que cada neurona de entrada en una FNN es independiente, en las redes neuronales recurrentes se relacionan entre ellas y se retroalimentan.

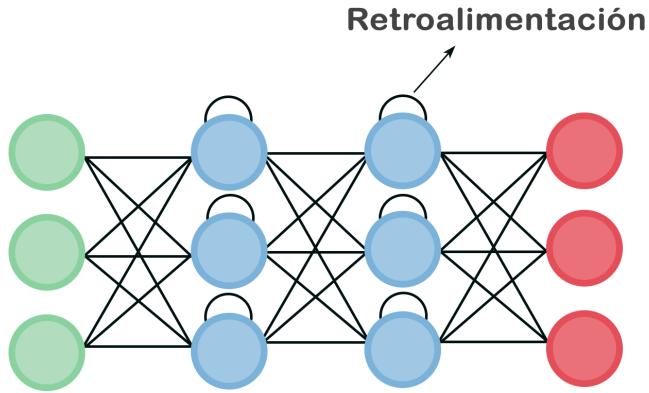


Figura 8: Ejemplo de RNN

Un problema frecuente en las RNN es su dificultad para capturar dependencias de largo plazo. Esto puede tener 2 causas, el desvanecimiento o la explosión del gradiente. El desvanecimiento del gradiente ocurre cuando, iteración tras iteración, el gradiente se aproxima a cero y se estabiliza, evitando que la red siga aprendiendo. Por el contrario, cuando el gradiente crece exponencialmente se habla de una explosión, esto lleva a inestabilidades en el aprendizaje, provocando que las actualizaciones de los parámetros sean erráticas e impredecibles.

Las redes neuronales con memoria a corto y largo plazo (LSTM) son un tipo de RNN que solucionan este problema mediante un algoritmo lógico de 3 puertas.

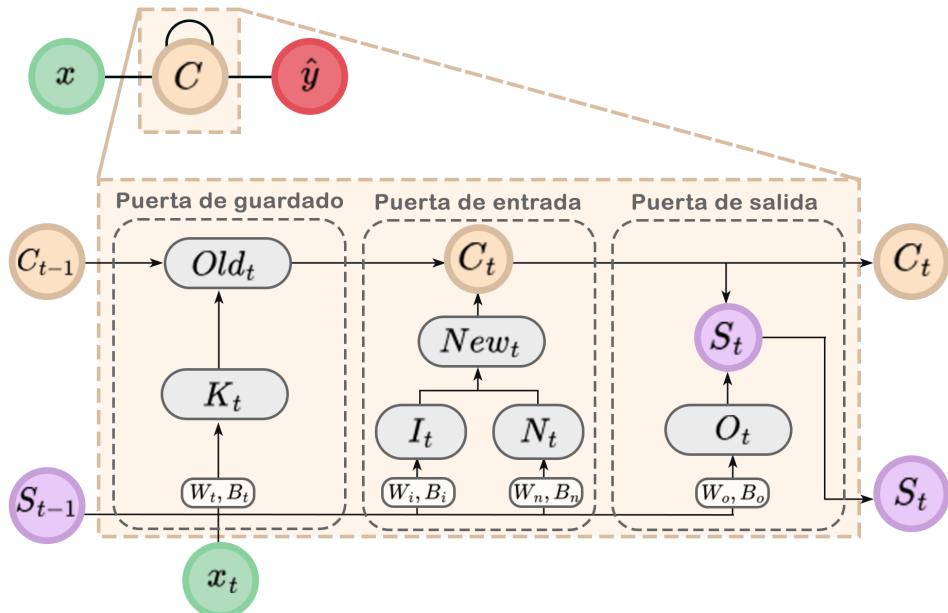


Figura 9: Estructura *Long Short Term Memory*

Puerta de guardado

La puerta de guardado se encarga de decidir que proporción de la información a largo plazo mantener en la neurona de memoria en cada iteración. Esta puerta recibe la entrada y el estado de la RNN, y las pasa como argumentos de una función sigmoide.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (6)$$

Sean S_{t-1} la información de la memoria a corto plazo actual de la red, x_t la entrada actual, y W_t y B_t los pesos y sesgos de la puerta de guardado respectivamente:

$$K_t = \sigma(W_k \times [S_{t-1}, x_t] + B_k) \quad (7)$$

$$Old_t = K_t \times C_{t-1} \quad (8)$$

Donde C_{t-1} es la información a largo plazo guardada actualmente y Old_t lo que se mantendrá para la próxima iteración de la red.

Si K_t es igual a 1, significa que la información guardada debe ser mantenida perfectamente. Si K_t fuera igual a 0, la información a largo plazo debe ser descartada completamente.

Puerta de entrada

La puerta de entrada controla que información añadir a la neurona de memoria. Propone un nuevo valor para la información a largo plazo y decide que proporción de esta sumar al valor actual. Sea $tanh(x)$ la función tangente hiperbólica y W_n y B_n pesos y sesgos respectivamente, el nuevo valor propuesto para la información a largo plazo es:

$$N_t = \tanh(W_n \times [S_{t-1}, x_t] + B_n) \quad (9)$$

Y la proporción de esta que se sumará al valor actual esta dada por I_t , cuyos pesos y sesgos son W_i y B_i .

$$I_t = \sigma(W_i \times [S_{t-1}, x_t] + B_i) \quad (10)$$

Por lo que el nuevo valor de la información a largo plazo de la red es:

$$C_t = Old_t + New_t \quad (11)$$

Donde $New_t = I_t \times N_t$

Puerta de salida

La puerta de salida se encarga de extraer la información más importante del estado actual de la neurona para usar como salida. Sean W_o y B_o los pesos y sesgos de la puerta de salida:

$$O_t = \sigma(W_o \times [S_{t-1}, x_t] + B_o) \quad (12)$$

$$S_t = O_t \times \tanh(C_t) \quad (13)$$

Donde S_t es la nueva información a corto plazo en la red neuronal.

Las 3 puertas son lógicas para que sea sencillo aplicar la retropropagación. Este sistema de puertas evita los problemas de desvanecimiento y explosión del gradiente, y evita que se acumulen muchos estados por largos períodos de tiempo, eligiendo que información es relevante guardar.

3.4.3 Modelos transformadores

Otro tipo de modelo de aprendizaje profundo son los *transformer models* (modelos transformadores), los cuales son significativamente más eficientes al entrenar y realizar inferencias que las RNNs; gracias al uso de mecanismos de atención, presentados en la publicación '*Attention is all you need*' de Google. Esos mecanismos capturan dependencias y relaciones en la secuencias de valores que se alimentan al modelo, logrando poner en contexto a cada observación.

Los modelos transformadores fueron creados originalmente con el propósito de generar texto. Sin embargo, tanto TimeGPT como Chronos explotan esta tecnología para el pronóstico de series de tiempo. Ambos modelos son preentrenados, lo cual significa que la optimización de parámetros y pesos fue realizada antes de usarse el modelo. Esto se logra entrenando y generalizando el modelo en un conjunto de datos extenso, por lo general de fuentes públicas. El preentrenamiento permite que el modelo adquiera conocimientos generales sobre la estructura y los patrones de los datos, los cuales luego pueden ser reutilizados en tareas concretas mediante técnicas como *fine-tuning* (ajuste fino). Los modelos preentrenados constituyen una gran innovación, lo que mejora la accesibilidad, precisión, eficiencia computacional y velocidad del pronóstico.

Dado que los modelos de lenguaje de texto utilizan diccionarios de *tokens*, que son segmentos de caracteres representados vectorialmente según ciertos parámetros, es necesario tokenizar los valores de la serie temporal. El diccionario de *tokens* con el que operan los modelos de lenguaje no es infinito, por lo tanto es necesario proyectar las observaciones a un set finito de *tokens*. Para cumplir esto, Chronos escala y discretiza las observaciones. TimeGTP por su parte usa las mismas observaciones como *tokens*, esto dado que, si bien su arquitectura es la de un modelo transformador, esta no está basada en ningún modelo de lenguaje existente, y en cambio trabaja con un modelo especializado en series de tiempo entrenado para minimizar el error de pronóstico.

Para el escalado se aplica a las observaciones una transformación del tipo $f(x_i) = (x_i - m)/s$. Existen variadas técnicas de escalado eligiendo apropiadamente m y s , pero se opta por elegir $m = 0$ y $s = \frac{1}{C} \sum_{i=1}^C |x_i|$ debido a que preserva los valores iguales a cero, los cuales pueden ser importantes de destacar en numerosas aplicaciones.

Sin embargo, estos valores siguen siendo números reales y no pueden ser procesados directamente por un modelo de lenguaje. Es por esto que se discretizan las observaciones. Se seleccionan B centros de intervalos en la recta real, c_1, c_2, \dots, c_B , y $B - 1$ extremos b_i que los separan, $c_i < b_i < c_{i+1}$ para $i \in \{1, \dots, B - 1\}$. Las funciones de discretización $q : \mathfrak{R} \rightarrow \{1, 2, \dots, B\}$, y de descuantificación $d : \{1, 2, \dots, B\} \rightarrow \mathfrak{R}$ se definen como:

$$q(x) = \begin{cases} 1 & \text{si } -\infty \leq x < b_1 \\ 2 & \text{si } b_1 \leq x < b_2 \\ \vdots & \\ B & \text{si } b_{B-1} \leq x < \infty \end{cases} \quad \text{y} \quad d(j) = c_j \quad (14)$$

Una vez se hayan transformado las observaciones para poder ser leídas por el modelo, el funcionamiento de un transformador es el siguiente:

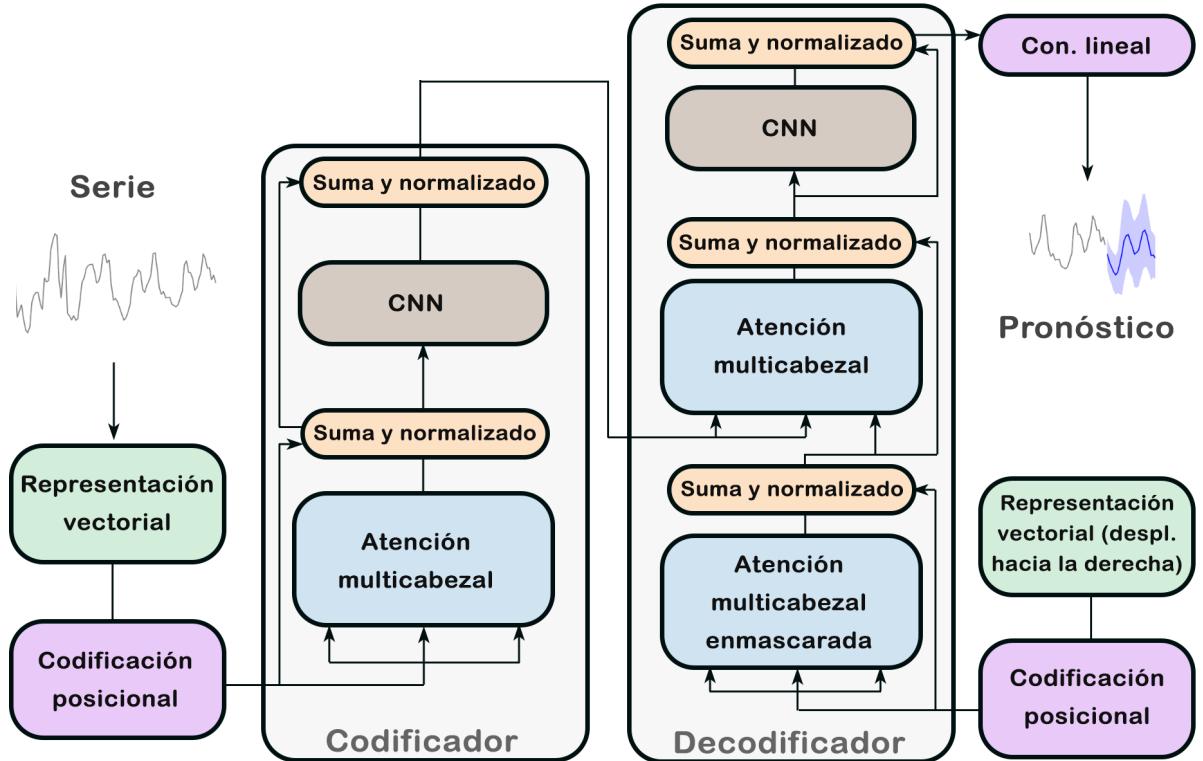


Figura 10: Diagrama de la estructura del modelo transformador de TimeGPT

Codificador

1. Representación vectorial: Cada *token* es transformado en un vector (vector de entrada) con muchas dimensiones (\vec{E}). Las dimensiones corresponden a diferentes características que el modelo definió en el preentrenado con una numerosa cantidad de parámetros.
2. Codificación posicional: Un set de valores adicionales o vectores son añadidos a los vectores de entrada antes de alimentarlos al modelo ($\vec{E} \leftarrow \vec{E} + \vec{P}$). Estas codificaciones posicionales tienen patrones específicos que agregan la información posicional del token.
3. Atención multi-cabezal (del inglés *Multi-Head Attention*): La autoatención opera en múltiples ‘cabezales de atención’ para capturar los diferentes tipos de relaciones entre *tokens*. Una cabeza de atención verifica el contexto en el que se presenta el token, y manipula los valores del vector que lo representa para añadir esta información contextual.

La verificación del contexto funciona gracias a una matriz denominada *Query* (W_Q) que examina ciertas características definidas con anterioridad en el preentrenado. El vector de entrada (\vec{E}) es multiplicado por esta matriz, resultando en un vector de consultas (\vec{Q}) para cada token. Una matriz de claves (W_K) que comprueba las relaciones con las características en la matriz de consultas, y tiene las mismas dimensiones que esta, es también post-multiplicada por \vec{E} generando así el vector de claves (\vec{K}). Luego, se forma una nueva matriz a partir de los productos cruzados entre los vectores \vec{K} y \vec{Q} de cada token, se divide por la raíz de la dimensión de los vectores¹ ($\sqrt{d_k}$) y se normaliza con softmax² por columna³ (\vec{S}). Valores altos indican que un

¹Es útil para mantener estabilidad numérica, la cual describe cómo los errores en los datos de entrada se propagan a través del algoritmo. En un método estable, los errores debidos a las aproximaciones se atenúan a medida que la computación procede.

² $\text{softmax}(x) = \frac{e^{x_i/t}}{\sum_j e^{x_j/t}}$

³Aplicar softmax hace que cada columna se comporte como una distribución de probabilidad.

token (de las columnas) esta siendo influenciado por el comportamiento de otro token (de las filas).

	UN $\downarrow \vec{E}_1$ $\downarrow W_Q$ $\downarrow \vec{Q}_1$	GRAN $\downarrow \vec{E}_2$ $\downarrow W_Q$ $\downarrow \vec{Q}_2$	BLANCO $\downarrow \vec{E}_3$ $\downarrow W_Q$ $\downarrow \vec{Q}_3$	AVIÓN $\downarrow \vec{E}_4$ $\downarrow W_Q$ $\downarrow \vec{Q}_4$	EN $\downarrow \vec{E}_5$ $\downarrow W_Q$ $\downarrow \vec{Q}_5$	EL $\downarrow \vec{E}_6$ $\downarrow W_Q$ $\downarrow \vec{Q}_6$	AMPLIO $\downarrow \vec{E}_7$ $\downarrow W_Q$ $\downarrow \vec{Q}_7$	CIELO $\downarrow \vec{E}_8$ $\downarrow W_Q$ $\downarrow \vec{Q}_8$
UN $\rightarrow \vec{E}_1 \xrightarrow{W_K} \vec{K}_1$	-4.9	-23.2	-12.0	-5.4	-84.9	-48.7	-13.1	-52.9
GRAN $\rightarrow \vec{E}_2 \xrightarrow{W_K} \vec{K}_2$	-40.2	+4.7	-63.1	+82.3	-2.8	-40.6	-18.9	-20.9
BLANCO $\rightarrow \vec{E}_3 \xrightarrow{W_K} \vec{K}_3$	-13.9	-0.2	-12.3	+83.9	-85.1	-51.7	-23.6	+4.3
AVIÓN $\rightarrow \vec{E}_4 \xrightarrow{W_K} \vec{K}_4$	-91.6	-74.7	-16.5	-13.1	-62.7	-36.3	-48.9	-24.0
EN $\rightarrow \vec{E}_5 \xrightarrow{W_K} \vec{K}_5$	-17.3	-87.4	-21.0	-40.0	-18.4	-34.3	-72.8	-13.2
EL $\rightarrow \vec{E}_6 \xrightarrow{W_K} \vec{K}_6$	-97.9	-95.1	-73.4	-60.4	-84.0	-13.1	-34.3	-46.1
AMPLIO $\rightarrow \vec{E}_7 \xrightarrow{W_K} \vec{K}_7$	-54.1	-23.0	-84.7	-33.0	-67.7	-91.8	-10.6	+87.2
CIELO $\rightarrow \vec{E}_8 \xrightarrow{W_K} \vec{K}_8$	-21.9	-46.8	-67.0	-84.2	-75.5	-53.2	-84.6	-11.4

Figura 11: Verificación del contexto

Una vez conocido que *tokens* son relevantes para otros *tokens*, es necesario saber de que forma son afectados. Una matriz de valores (W_V) es post-multiplicada por cada vector de entrada resultando en los vectores de valor (\vec{V}), los cuales son multiplicados a cada columna. La suma por filas devuelven el vector $\Delta \vec{E}$ que se suma al vector de entrada original de cada token.

	UN ↓ \vec{E}_1	GRAN ↓ \vec{E}_2	BLANCO ↓ \vec{E}_3	AVIÓN ↓ \vec{E}_4	EN ↓ \vec{E}_5	EL ↓ \vec{E}_6	AMPLIO ↓ \vec{E}_7	CIELO ↓ \vec{E}_8
UN → $\vec{E}_1 \xrightarrow{W_V} \vec{V}_1$				0.00 \vec{V}_1				
GRAN → $\vec{E}_2 \xrightarrow{W_V} \vec{V}_2$				0.17 \vec{V}_2				
BLANCO → $\vec{E}_3 \xrightarrow{W_V} \vec{V}_3$				0.83 \vec{V}_3				
AVIÓN → $\vec{E}_4 \xrightarrow{W_V} \vec{V}_4$				0.00 \vec{V}_4				
EN → $\vec{E}_5 \xrightarrow{W_V} \vec{V}_5$				0.00 \vec{V}_5				
EL → $\vec{E}_6 \xrightarrow{W_V} \vec{V}_6$				0.00 \vec{V}_6				
AMPLIO → $\vec{E}_7 \xrightarrow{W_V} \vec{V}_7$				0.00 \vec{V}_7				
CIELO → $\vec{E}_8 \xrightarrow{W_V} \vec{V}_8$				0.00 \vec{V}_8				

Figura 12: Influencia de *tokens* sobre otros luego de aplicar softmax

$$\Delta \vec{E}_j = \sum_i S_j \cdot \vec{V}_i \quad (15)$$

Con múltiples ‘cabezas’, cada una con sus propias matrices W_K , W_Q y W_V , se generan múltiples $\Delta \vec{E}$ que se suman y se añaden al vector de entrada original.

$$\vec{E} \Leftarrow \vec{E} + \sum_h \Delta \vec{E}_h \quad (16)$$

Todo el mecanismo de atención se puede resumir con la siguiente función:

$$\text{Atencion}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (17)$$

Es importante notar que todas las matrices Q , K y V son preentrenadas.

- Suma y normalizado: Este paso hace referencia a la ecuación Ecuación 16, donde, en lugar de simplemente pasar los datos por las capas y modificar directamente el vector, las conexiones residuales se añaden sobre el vector de entrada en la salida de cada capa.

Dado que las redes neuronales profundas sufren de inestabilidad en los pesos al actualizarlos, una normalización al vector estabiliza el entrenamiento y mejora la convergencia.

- Red neuronal convolucional (CNN): A diferencia de otros modelos transformadores tradicionales, TimeGPT incorpora *CNNs* para descubrir dependencias locales y patrones de corto plazo, tratando de minimizar una función de costo. *Chronos*, por otro lado, utiliza una *Feed-Forward Network*.

Decodificador

1. Representación vectorial (desplazado hacia la derecha): La entrada del decodificador son los *tokens* desplazados hacia la derecha.
2. Codificación posicional
3. *Masked multi-head attention* (Atención multicabezal enmascarada): Las predicciones deben realizarse únicamente con los valores previos a cada token. Como consecuencia, antes de aplicar la transformación softmax, se debe reemplazar todos los valores debajo de la diagonal principal de la matriz QK por $-\infty$. Esto evita que los *tokens* sean influenciados por *tokens* anteriores.
4. *Multi-head attention*: Se usan las matrices de claves y valores que da como salida el codificador, y la matriz de consultas es la salida de la capa de atención multicabezal enmascarada.
5. Conexión lineal: Es una capa completamente conectada que traduce las representaciones de atributos aprendidas en predicciones relevantes.

3.4.4 Diferencias entre TimeGPT y Chronos

TimeGPT es un modelo transformador preentrenado (de aquí las siglas GPT, *generative pretrained transformer*) para el pronóstico de series de tiempo. Puede producir predicciones en diversas áreas y aplicaciones con gran precisión, sin necesidad de entrenamiento adicional. El mismo fue desarrollado por Nixtla y tuvo su primera beta privada en Agosto de 2023, volviéndose accesible a todo público desde el 18 de julio de 2024, sin embargo es de código cerrado.

Chronos es una familia de modelos transformadores preentrenados para series de tiempo basados en arquitecturas de modelos de lenguaje. Fue publicado por Amazon en marzo de 2024 y su código es de libre acceso.

Nixtla desarrolló para TimeGPT un modelo con arquitectura *transformer* que puede trabajar directamente con series de tiempo. Por otro lado, Chronos transforma los datos de las series para poder alimentarlos a los modelos de lenguaje existentes. Dado que al transformar los valores los datos se vuelven discretos, Chronos busca minimizar la entropía cruzada entre las distribuciones de las categorías reales contra las predichas.

La función de pérdida utilizada por Chronos está dada por:

$$\ell(\theta) = \sum_{l=1}^{h+1} \sum_{i=1}^{|\nu_{ts}|} 1_{z_{n+l+1}=i} \log p_\theta(z_{n+l+1} = i | z_{1:n+l}) \quad (18)$$

Donde $|\nu_{ts}|$ es el tamaño del diccionario de *tokens*, el cual depende del número de intervalos creados. z_{n+h+1} es la serie transformada en *tokens*, cuyas primeras n observaciones se utilizan como entrenamiento para pronosticar las siguientes h , y se agrega al final un token **EOS** que se utiliza comúnmente en los modelos de lenguaje para denotar el final de la secuencia. p_θ es la probabilidad estimada por el modelo bajo la parametrización θ .

Es importante notar que no es una función que detecta distancias, por lo que se espera que el modelo asocie a los intervalos cercanos gracias a la información en el conjunto de entrenamiento. Es decir, Chronos aplica regresión por clasificación.

Otra diferencia entre TimeGPT y Chronos es el tipo de red neuronal que utilizan para detectar patrones en los datos. Mientras que el primero hace uso de las CNN, el segundo aplica *Feed-Forward Networks*. Luego de la conexión lineal, Chronos necesita volver a aplicar softmax para obtener las probabilidades del pronóstico, procedimiento que no es necesario por parte de TimeGPT.

3.5 Métricas de evaluación

Para comparar el rendimiento de los modelos se utilizan métricas cuantitativas. Para los pronósticos puntuales se usará el porcentaje del error absoluto medio (MAPE), mientras que para los pronósticos probabilísticos se aplicará el *Interval Score*, propuesto por Gneiting y Raftery (2007), que penaliza tanto la amplitud de los intervalos como la falta de cobertura. Estas comparaciones permiten evaluar la precisión, la robustez y la eficiencia de cada enfoque.

Sea $e_l = z_{n+l} - \hat{z}_n(l)$ el error de la l -ésima predicción, donde $\hat{z}_n(l)$ representa el pronóstico l pasos hacia adelante, algunas medidas del error para pronósticos h pasos hacia adelante son:

- Error Cuadrático Medio (*Mean Square Error, MSE*):

$$MSE = \frac{1}{h} \sum_{l=1}^h e_l^2 \quad (19)$$

- Error absoluto medio (*Mean Absolute Error, MAE*):

$$MAE = \frac{1}{h} \sum_{l=1}^h |e_l| \quad (20)$$

- Porcentaje del error absoluto medio (*Mean Absolute Percentage Error, MAPE*)

$$MAPE = \left(\frac{1}{h} \sum_{l=1}^h \left| \frac{e_l}{Z_{n+l}} \right| \right) \cdot 100\% \quad (21)$$

El problema de estos errores es que solo tienen en cuenta la estimación puntual, y por lo general, es buena idea trabajar con pronósticos probabilísticos para cuantificar la incertidumbre de los valores futuros de la variable. Gneiting y Raftery (2007, JASA) propusieron en *Strictly Proper Scoring Rules, Prediction, and Estimation* una nueva medida del error que tiene en cuenta los intervalos probabilísticos de la estimación, llamándola *Interval Score*:

$$S = \frac{1}{h} \sum_{l=1}^h (W_l + O_l + U_l) \quad (22)$$

Donde:

$$W_l = IS_l - II_l \quad (23)$$

$$O_l = \begin{cases} \frac{2}{\alpha} (Z_n(l) - Z_{n+l}) & \text{si } Z_n(l) > Z_{n+l} \\ 0 & \text{en otro caso} \end{cases} \quad U_l = \begin{cases} \frac{2}{\alpha} (Z_{n+l} - Z_n(l)) & \text{si } Z_n(l) < Z_{n+l} \\ 0 & \text{en otro caso} \end{cases} \quad (24)$$

Siendo IS_l e II_l los extremos superior e inferior del intervalo del l-ésimo pronóstico respectivamente. Es fácil darse cuenta que W es una penalización por el ancho del intervalo, y que O y U son penalizaciones por sobre y subestimación respectivamente.

Aquel modelo que minimice el MAPE y el *Interval Score* será considerado el mejor.

3.6 Selección de parámetros y validación del modelo

La correcta elección de los parámetros del modelo constituye una de las tareas más importantes para lograr un buen ajuste de los datos. No resulta conveniente utilizar todos los datos disponibles para este fin, ya que esto puede conducir a un sobreajuste (*overfitting*). Se considera que un modelo presenta sobreajuste cuando se ajusta en exceso a los datos de entrenamiento, comprometiendo su capacidad de generalización frente a datos no observados. Para evitar este problema se aplican técnicas específicas de validación que permiten seleccionar los parámetros sin comprometer la capacidad predictiva del modelo.

La validación *holdout* consiste en reservar una parte del conjunto de entrenamiento como validación. Se ajustan las distintas configuraciones de parámetros sobre el resto de los datos de entrenamiento, y se prueban las métricas de evaluación sobre el conjunto reservado para validar. Luego, se ajusta el modelo con la mejor combinación de parámetros utilizando los datos de entrenamiento y de validación. Debido a la ordinalidad de los datos, las observaciones del conjunto de validación deben ser posteriores a las del conjunto de entrenamiento.

$$\text{Conjunto de entrenamiento total : } \{ \underbrace{z_1, \dots, z_c}_{\text{Entrenamiento}}, \underbrace{z_{c+1}, \dots, z_n}_{\text{Validación}} \} \quad (25)$$

En series que presentan estacionalidad, se prioriza que el conjunto de validación cubra al menos un ciclo. Esto tiene el objetivo de poder evaluar el ajuste del modelo en todo el ciclo estacional.

Exclusivamente para los modelos de la familia ARIMA, se utiliza el método de Box-Jenkins. El método consiste en comparar aquellos modelos que cumplan los supuestos, y elegir como mejor combinación de parámetros aquella que minimize el AIC (*Akaike Information Criterion*), medida de ajuste que penaliza por la cantidad de parámetros.

4. Aplicación

La aplicación empírica de esta tesina tiene como objetivo implementar, ajustar y comparar los modelos presentados en la sección 3, empleando un conjunto de series temporales seleccionadas. Para este fin, se utilizó el lenguaje de programación Python, junto con librerías de código abierto ampliamente reconocidas.

Se trabajó con series temporales reales, obtenidas de fuentes públicas y confiables. Cada serie fue analizada desde tres enfoques metodológicos: modelos estadísticos tradicionales, algoritmos de aprendizaje automático y modelos de aprendizaje profundo. En cada caso se exploraron distintas configuraciones de parámetros, explicando su significado y función en el ajuste.

- Los modelos estadísticos clásicos (ARIMA y SARIMA) se ajustaron mediante la librería `pmdarima`. La selección de parámetros se efectuó tanto de manera manual como automática, empleando como criterio principal el Criterio de Información de Akaike (AIC).
- Para el enfoque de aprendizaje automático, se emplearon algoritmos de boosting, específicamente XGBoost y LightGBM, implementados con las librerías `xgboost` y `lightgbm` respectivamente.
- En el caso de los modelos de aprendizaje profundo, se entrenaron redes LSTM utilizando la librería `scalecast`.
- Finalmente, se exploraron dos modelos fundacionales preentrenados: TimeGPT, accedido a través de la API de Nixtla mediante la librería `nixtla`, y Chronos, una familia de modelos preentrenados desarrollada por *Amazon Web Services*, cuya implementación se llevó a cabo con la librería `autogluon`.

Para cada modelo se generaron pronósticos puntuales y probabilísticos, con un nivel de confianza del 80%. Su desempeño se evaluó apartir de dos métricas: el *Mean Absolute Percentage Error* (MAPE) y el *Interval Score*. La elección de la mejor combinación de hiperparámetros se realizó en función del MAPE, con excepción de los modelos ARIMA. Los resultados fueron sintetizados en tablas comparativas y visualizaciones gráficas, acompañadas de un análisis crítico.

4.1 Series analizadas

Con el propósito de evaluar el desempeño de los modelos bajo diferentes condiciones, se seleccionaron tres series temporales con características heterogéneas:

- Atenciones de guardia mensuales por patologías respiratorias en un hospital de la ciudad de Rosario.
- Número mensual de personas registradas con empleo asalariado en el sector educativo privado de Argentina.
- Temperatura horaria en la ciudad de Rosario.

La primera serie corresponde al número mensual de atenciones de guardia por patologías respiratorias (Códigos CIE10: J09–J18, J21, J22 y J44) en el Hospital de Niños Víctor J. Vilela de la ciudad de Rosario. La información fue provista por la Dirección General de Estadística de la Municipalidad de Rosario⁴. La serie presenta una marcada estacionalidad, con picos en

⁴Dirección General de Estadística. (s.f.) *Situación epidemiológica de problemas de salud priorizados*. Municipalidad de Rosario.

los meses de invierno y una disminución significativa en el año 2020, atribuida a las medidas sanitarias adoptadas durante la pandemia de COVID-19. Este patrón estacional se aprecia con mayor claridad en el gráfico 39 del anexo.

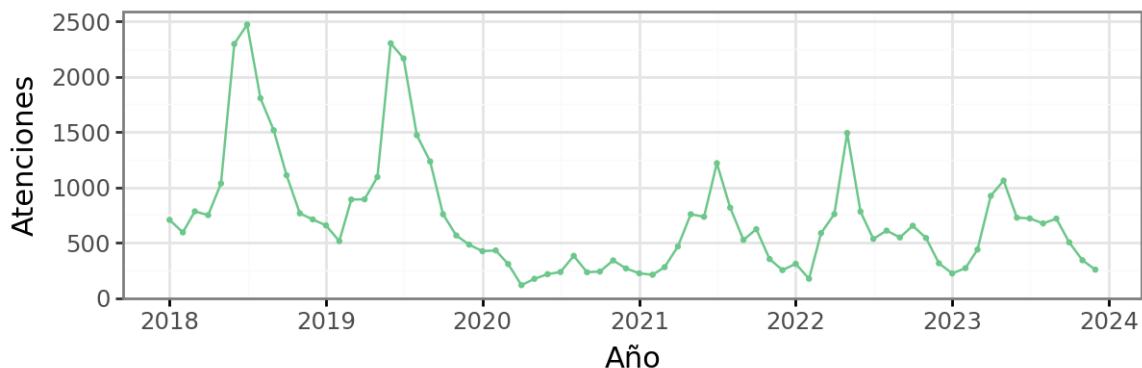


Figura 13: Atenciones en guardia por enfermedades respiratorias

La segunda serie corresponde a la cantidad mensual de personas con empleo asalariado en el área de enseñanza, registradas en el sector privado en Argentina. La serie presenta una tendencia creciente a lo largo del tiempo, con descensos marcados en diciembre y enero, que se visualizan con el gráfico 40 del anexo. Asimismo, se observan impactos atribuibles a la pandemia de COVID-19. Los datos provienen del informe Situación y evolución del Trabajo Registrado de la Secretaría de Trabajo, Empleo y Seguridad Social⁵.

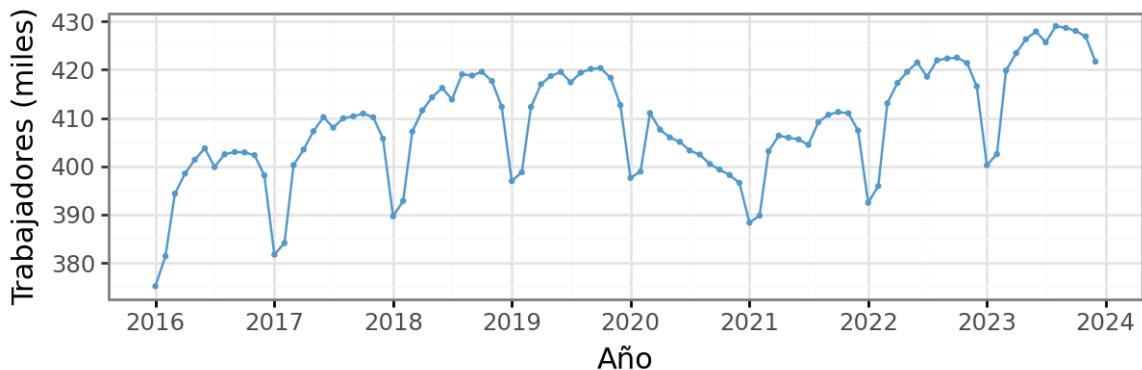


Figura 14: Personas con empleo asalariado en el área de enseñanza y registradas en el sector privado

Por último, se analizaron las temperaturas horarias en Rosario correspondientes a los primeros días de marzo de 2025, considerando su relación con la humedad relativa y la presión atmosférica estándar. El patrón estacional diario se aprecia claramente en el gráfico 41 del anexo: la temperatura se mantiene relativamente estable entre la noche y la mañana, y aumenta de forma pronunciada hacia la tarde. Los datos fueron relevados a partir de la página del Servicio Meteorológico Nacional⁶.

⁵Secretaría de Trabajo, Empleo y Seguridad Social. (2025). *Situación y evolución del Trabajo Registrado*. <https://www.argentina.gob.ar/trabajo/estadisticas/situacion-y-evolucion-del-trabajo-registrado>

⁶Servicio Meteorológico Nacional. (s.f.). *Datos horarios*. <https://www.smn.gob.ar/descarga-de-datos>

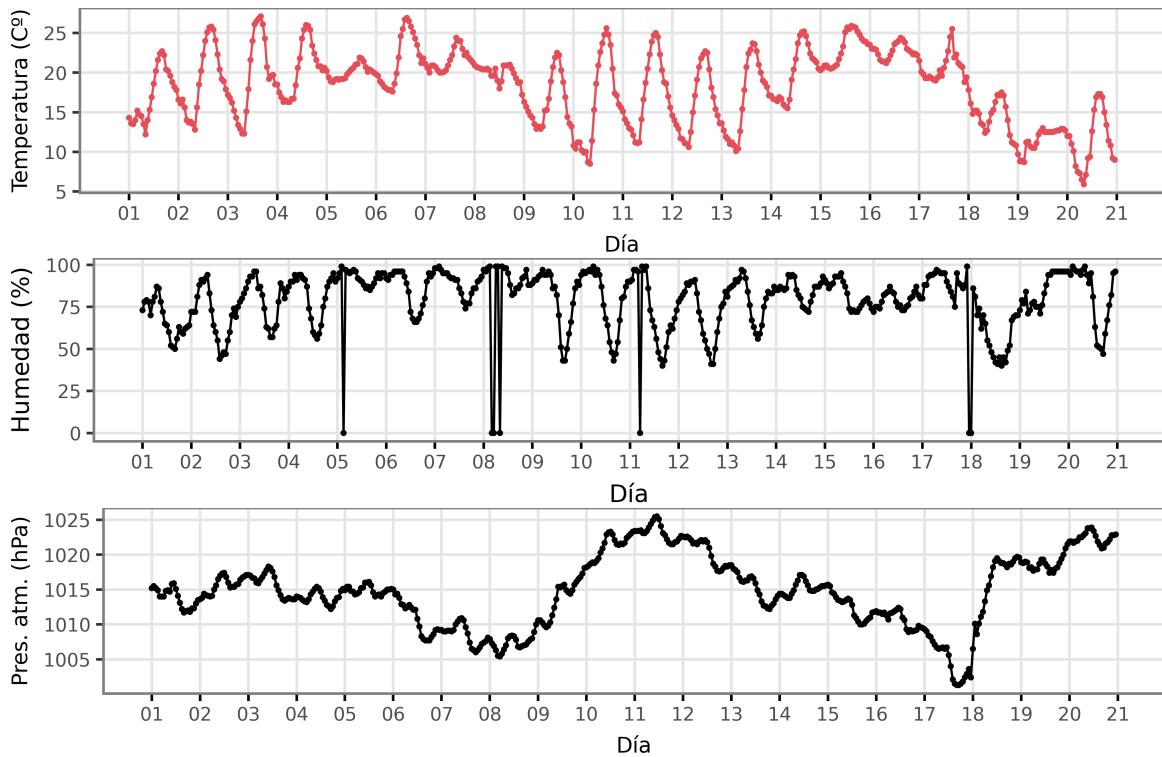


Figura 15: Temperatura (C°), humedad (%) y presión atmosférica (hPa) en Rosario por hora.

Es importante señalar que, en este análisis, tanto la humedad como la presión atmosférica se suponen conocidas en las horas a pronosticar. Si bien esta condición no refleja una situación realista, en algunos contextos ciertas variables exógenas pueden disponerse con antelación. La inclusión de esta serie tuvo como finalidad evaluar la capacidad de los modelos para explotar información adicional. Cabe destacar que algunos de los modelos analizados permiten incorporar variables exógenas sin requerir el conocimiento de sus valores futuros. No obstante, para asegurar la comparabilidad de los resultados, en este trabajo se asumió que tanto la humedad como la presión atmosférica eran conocidas a futuro en todos los casos.

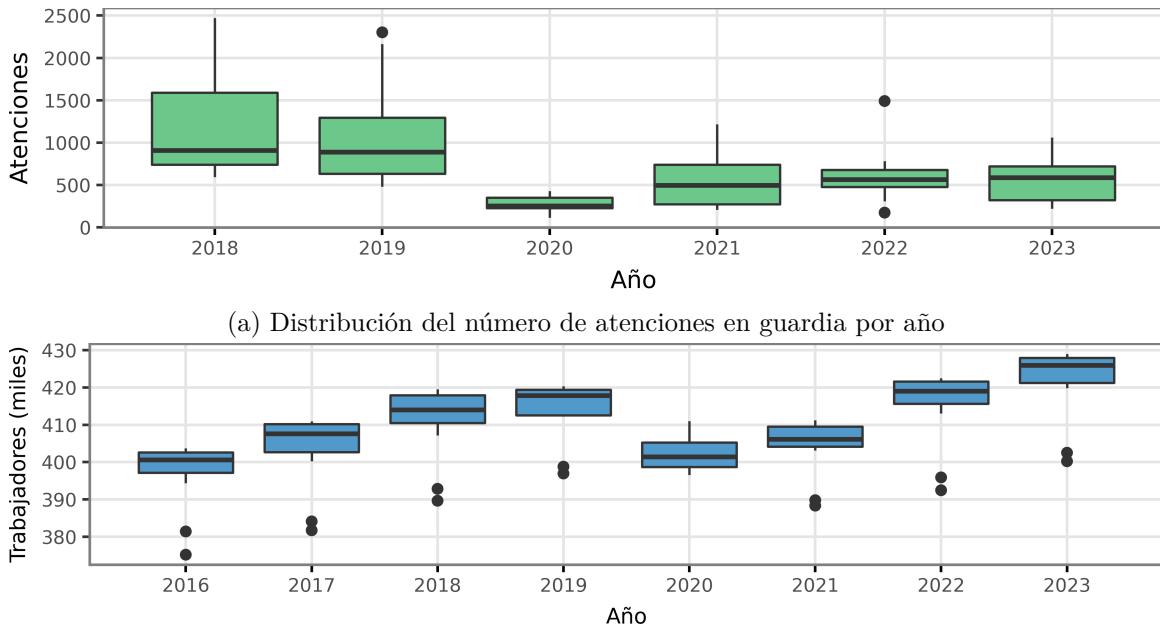
4.2 Ajuste y evaluación de modelos

4.2.1 Modelización con ARIMA y SARIMAX

Debido a la complejidad que implica modelar series de alta frecuencia con variables exógenas mediante ARIMA, los únicos modelos que se probaron para la serie de temperatura son el obtenido a partir del método automático de selección de modelos, descrito más adelante, y correcciones del mismos.

Previo a proponer modelos, es importante realizar un breve análisis exploratorio sobre las series. El objetivo de este análisis es verificar que estas sean estacionarias, lo cual implica identificar y eliminar posibles tendencias o patrones sobre la variancia y/o media de las series.

Como primer instancia, se realizan gráficos de cajas con el propósito de verificar que la variancia en cada período se mantenga constante. De lo contrario, se debe aplicar una transformación a la variable de interés.



(b) Distribución del número de trabajadores asalariados en el rubro de la enseñanza privada por año

Figura 16: Gráficos de caja por período.

Se observa en el gráfico 16a ciertas diferencias en la variabilidad anual de las atenciones. Como consecuencia, se calcula λ para la transformación de Box y Cox, resultando igual a -0.04. Con este resultado, se concluye que aplicar la transformación logaritmo a las atenciones en guardia por patologías respiratorias es necesario.

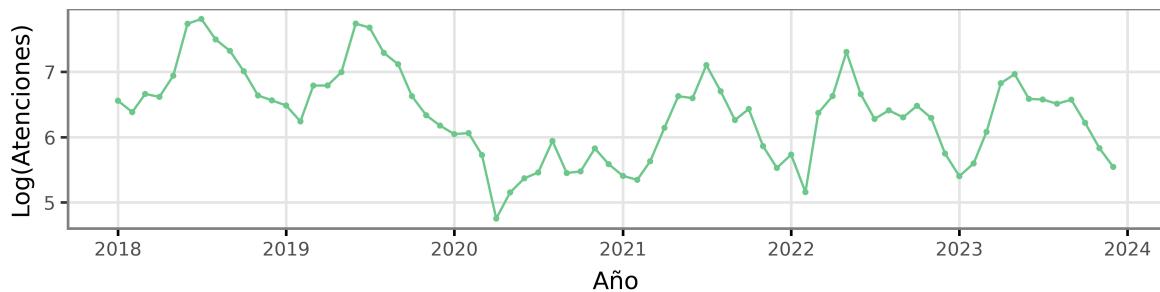


Figura 17: Logaritmo de las atenciones mensuales por guardia

El siguiente paso es verificar que no se presenten patrones estacionales ni tendencias en la media de los datos. De presentarse ambas, el orden en el que se hagan las diferenciaciones no afecta el resultado. Sin embargo, si se aplica primero la diferenciación regular, el patrón estacional permanecerá presente; en cambio, al comenzar con la diferenciación estacional, es posible que la tendencia desaparezca simultáneamente.

La función de autocorrelación muestral (FAM) y la función de autocorrelación parcial muestral (FAPM) para la serie transformada de atenciones son las siguientes:

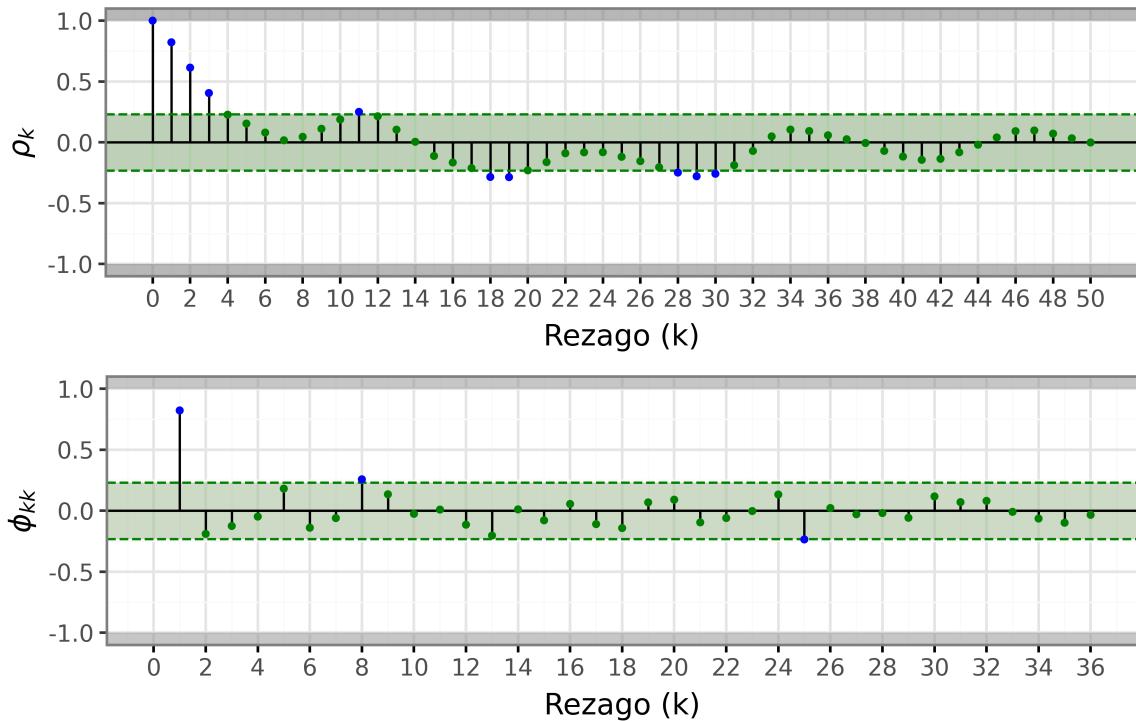


Figura 18: Autocorrelaciones del logaritmo de atenciones mensuales por guardia.

En el primer gráfico se observa como las autocorrelaciones parciales descienden lentamente de forma sinusoidal, mostrando valores significativos cerca de los puntos estacionales. El segundo gráfico muestra una función de autocorrelación parcial con un valor significativo en el primer rezago, y luego en los rezagos 8 y 25. Con un *p-value* del 0.3837 no se rechaza la hipótesis nula del test aumentado de Dickey-Fuller, y por lo tanto no se puede rechazar la existencia de una raíz unitaria, lo que a su vez indica que la serie no es estacionaria.

Para la serie de trabajadores las funciones de autocorrelación muestral son las siguientes:

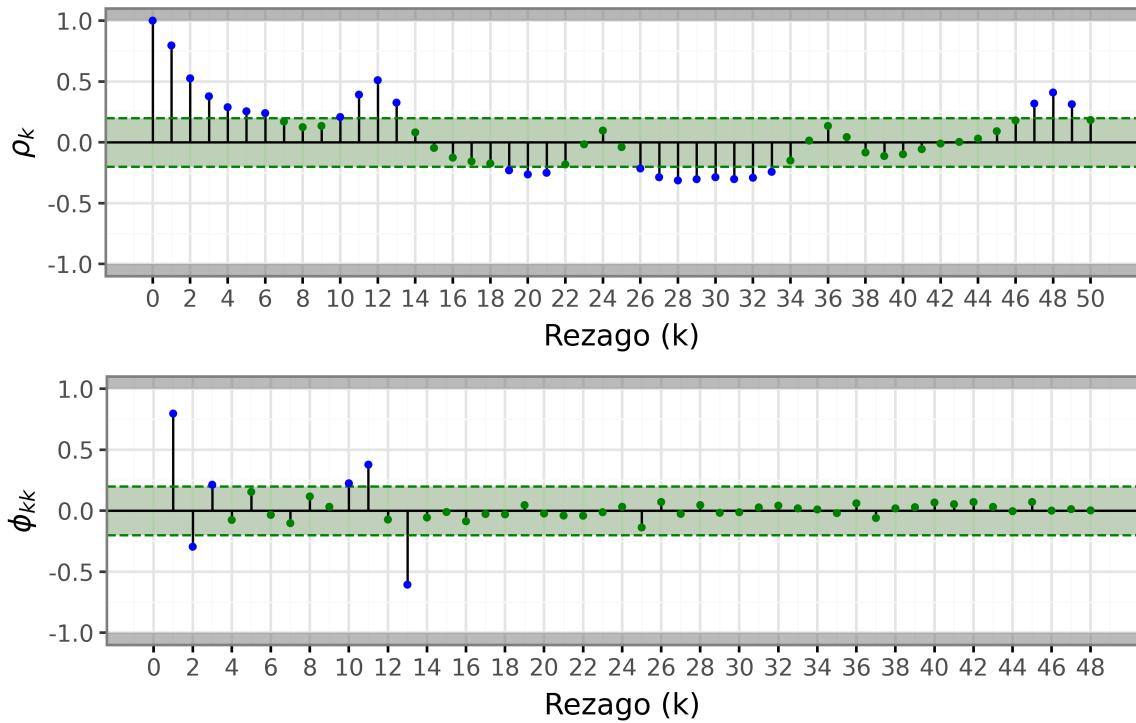
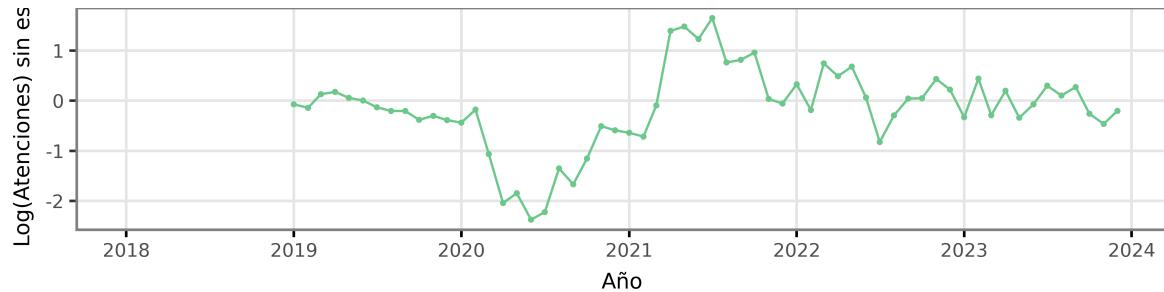


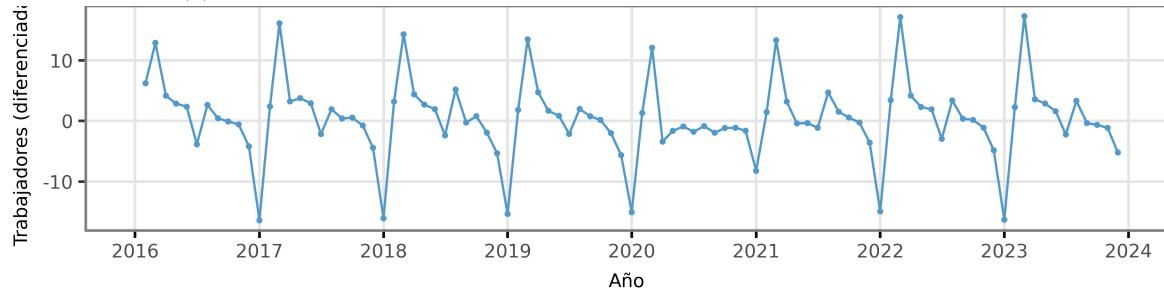
Figura 19: Autocorrelaciones del número mensual de trabajadores asalariados en el área de enseñanza privada.

Se observa nuevamente un descenso gradual de la FAM, exhibiendo valores significativos en los rezagos estacionales de la serie. El test de Dickey-Fuller rechaza la hipótesis nula con un *p-value* del 0.036. Sin embargo, se observa en el gráfico 14 que existe una tendencia positiva en la media de la serie. Esto se confirma con el test de Kwiatkowski-Phillips-Schmidt-Shin, cuya hipótesis nula sostiene que la serie es estacionaria alrededor de una constante, y esta se rechaza con un *p-value* igual a 0.0147. Estos resultados indican que una diferenciación en la parte regular es necesaria.

Dado que las series no presentan estacionariedad, se aplica una diferenciación estacional a la serie de atenciones y una regular a la de trabajadores.



(a) Logaritmo del número de atenciones en guardias sin estacionalidad



(b) Número de trabajadores asalariados en el rubro de la educación privada diferenciada

Figura 20: Series diferenciadas

No se observa en la figura 20 ningún patrón en la media a través del tiempo. Esto se confirma con el test de Kwiatkowski-Phillips-Schmidt-Shin, que devuelve *p-values* mayores a 0.1 para las 2 series estacionarias.

Las FAM y FAPM para la serie del logaritmo de atenciones diferenciada estacionalmente son:

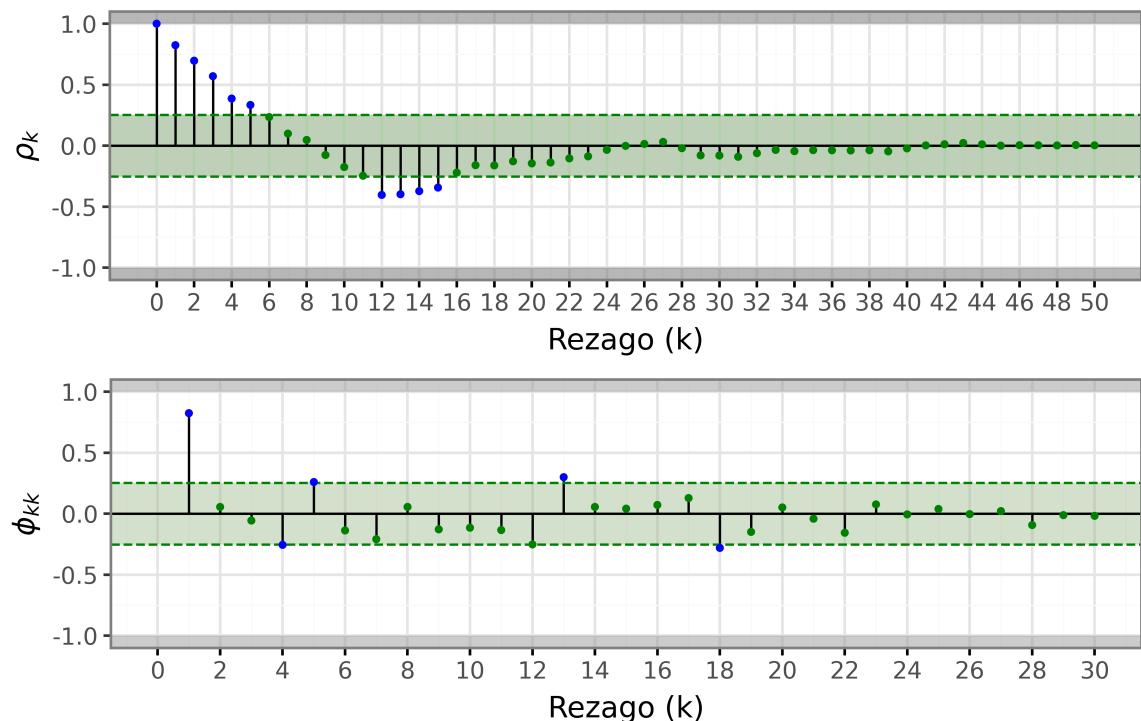


Figura 21: Autocorrelaciones del logaritmo de atenciones por guardia mensuales sin estacionalidad.

La FAM desciende lentamente en los primeros rezagos y los rezagos estacionales son significativos. A su vez, en la FAPM se destacan el primer y decimotercer rezago. Estos resultados conllevan a proponer un modelo con una componente AR en la parte estacionaria y una componente MA en la estacional, formando el modelo $SARIMA(1, 0, 0)(0, 1, 1)_{12}$, el cual se lo identifica como AT-1. En la FAPM se ignoran los rezagos significativos 4, 5, y 18 porque no tienen sentido en el contexto del problema.

Para la serie del número de trabajadores diferenciada las funciones de autocorrelación son:

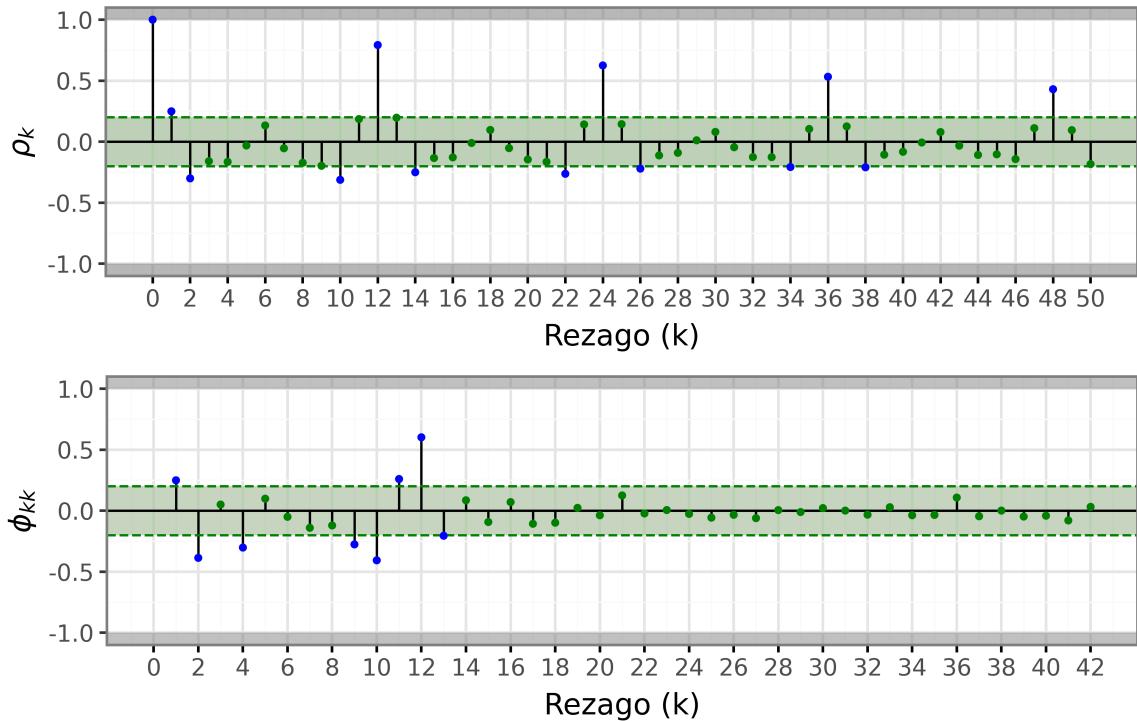


Figura 22: Autocorrelaciones del número mensual de trabajadores asalariados en el área de enseñanza privada sin estacionalidad.

La FAM presenta valores significativos en todos los rezagos estacionales, los cuales descienden paulatinamente. Este patrón es característico de una componente AR en la parte estacional. También son significativos el primer y segundo rezago, lo que puede indicar componentes AR(1) o MA(2) en la parte regular de la serie. En la FAPM se destacan los rezagos 1, 2, 11, 12 y 13. Con estas observaciones se proponen los modelos $SARIMA(1, 1, 0)(1, 0, 0)_{12}$ (TR-1) y $SARIMA(1, 1, 0)(1, 0, 1)_{12}$ (TR-2).

La función `auto_arima` de la librería `pmdarima` elige modelos de la familia ARIMA de forma automática. Utilizando esta herramienta se propone el modelo $SARIMA(0, 1, 0)(1, 0, 0)_{12}$ (AT-2) para la serie de atenciónes en guardia, mientras que para la serie de trabajadores se selecciona el modelo $SARIMA(2, 0, 0)(2, 1, 0)_{12}$ (TR-3).

Los modelos ARIMA suponen que los residuos se comportan como ruido blanco, por lo que se deberá verificar que no presenten correlación y que se distribuyan de manera aproximadamente normal, con media y variancia constantes.

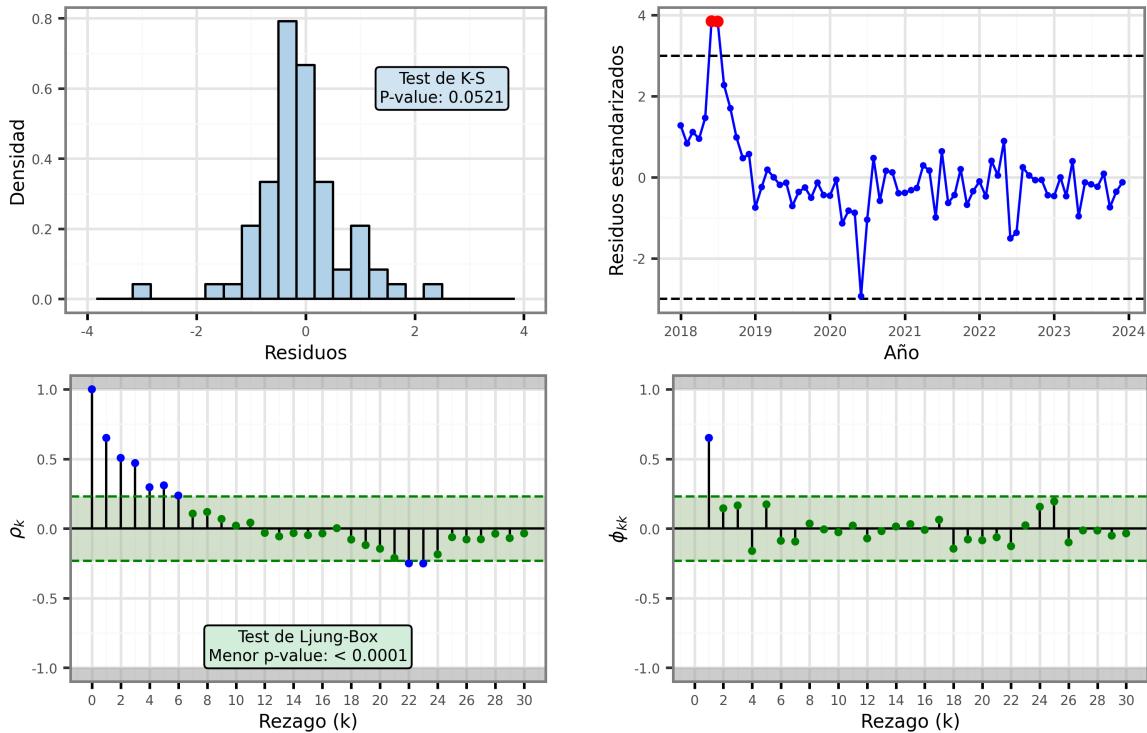


Figura 23: Comprobación de supuestos del modelo AT-1 para la serie de atenciones.

La figura 23 muestra la comprobación de supuestos del modelo AT-1. En el gráfico superior izquierdo se presenta la distribución de los residuos estandarizados, acompañada con el *test* de Kolmogorov-Smirnov, que con un *p-value* de 0.0521 no rechaza la hipótesis nula que sostiene que la distribución es normal. El gráfico superior derecho muestra los residuos estandarizados a través del tiempo, con el objetivo de comprobar la ausencia de patrones sistemáticos. Para el modelo AT-1 esto se cumple y se destacan únicamente 2 *outlayers*. En los gráficos inferiores, de izquierda a derecha, se visualizan las funciones de autocorrelación y autocorrelación parcial de los residuos estandarizados, junto al *test* de Ljung-Box. Este *test* contrasta la hipótesis nula de que las correlaciones entre los valores no difieren significativamente de cero. El valor que se muestra en el gráfico es el menor *p-value* registrado, producto de haber realizado el *test* para los 30 primeros rezagos. Con un *p-value* menor a 0.0001 no se puede concluir que los residuos están incorrelacionados, por lo tanto no se comportan como ruido blanco y se debe descartar el modelo.

Se probaron varias correcciones al modelo sin lograr resultados satisfactorios. Debido a esto, se decide agregar una diferenciación en la parte regular.

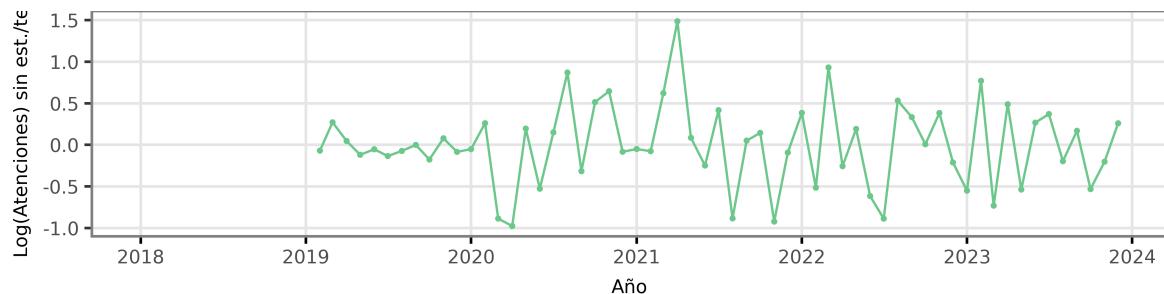


Figura 24: Logaritmo del número de atenciones en guardias sin estacionalidad ni tendencia.

La FAM y la FAPM de esta serie son:

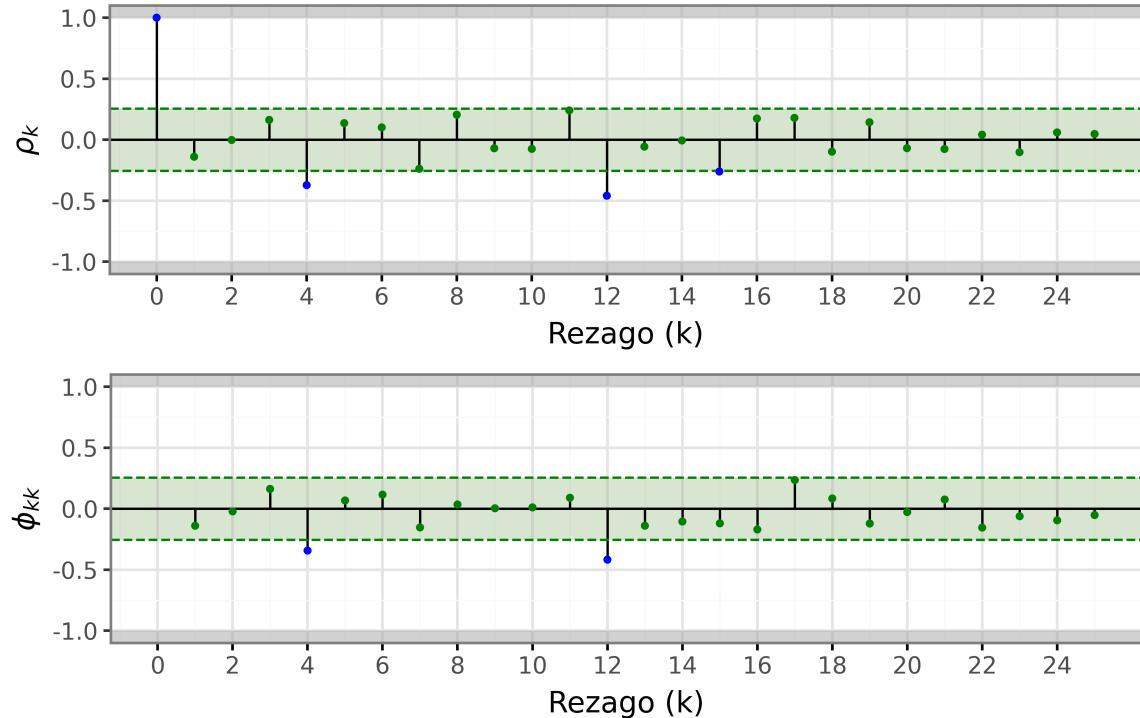


Figura 25: Autocorrelaciones del logaritmo del número de atenciones sin estacionalidad ni tendencia.

Se observan valores significativos en el cuarto y doceavo rezago, lo que podría significar que existen componentes MA en la parte estacional o regular de la serie. Se proponen entonces los modelos $SARIMA(0, 1, 1)(0, 1, 0)_{12}$ (AT-3) y $SARIMA(0, 1, 0)(0, 1, 1)_{12}$ (AT-4).

Para la serie de temperaturas en Rosario la función `auto_arima` propone el modelo $SARIMAX(1, 1, 1)(2, 0, 1)_{24}$ (TE-1). Sin embargo, habiendo analizado los coeficientes, presentes en la tabla 18 del anexo, se concluye que el modelo no goza de estacionariedad en parte estacional. Como correcciones, se postulan los modelos $SARIMAX(1, 1, 1)(1, 0, 1)_{24}$ (TE-2), $SARIMAX(1, 1, 0)(2, 0, 1)_{24}$ (TE-3) y $SARIMAX(1, 1, 0)(1, 0, 1)_{24}$ (TE-4), de los cuales únicamente el último cumple con todas las propiedades requeridas.

Tabla 1: Cumplimiento de las condiciones de estacionariedad e invertibilidad de los modelos ajustados.

ID	Modelo	AIC	Parámetros significativos*	Parte regular		Parte estacional	
				Est.	Inv.	Est.	Inv.
Atenciones en guardia							
AT-1	$SARIMA(1, 0, 0)(0, 1, 1)_{12}$	861.2	Si	Si	Si	Si	Si
AT-2	$SARIMA(0, 1, 0)(1, 0, 0)_{12}$	1013.5	Si	Si	Si	Si	Si
AT-3	$SARIMA(0, 1, 1)(0, 1, 0)_{12}$	857.6	No	Si	Si	Si	Si
AT-4	$SARIMA(0, 1, 0)(0, 1, 1)_{12}$	852.1	Si	Si	Si	Si	Si
Trabajadores							
TR-1	$SARIMA(1, 1, 0)(1, 0, 0)_{12}$	447.7	Si	Si	Si	Si	Si
TR-2	$SARIMA(1, 1, 0)(1, 0, 1)_{12}$	423.8	Si	Si	Si	Si	Si
TR-3	$SARIMA(2, 0, 0)(2, 1, 0)_{12}$	356.7	Si	Si	Si	Si	Si
Temperatura							
TE-1	$SARIMAX(1, 1, 1)(2, 0, 1)_{24}$	1067.1	No	Si	Si	No	Si
TE-2	$SARIMAX(1, 1, 1)(1, 0, 1)_{24}$	1046.1	No	Si	Si	Si	Si
TE-3	$SARIMAX(1, 1, 0)(2, 0, 1)_{24}$	1051.9	No	Si	Si	Si	Si
TE-4	$SARIMAX(1, 1, 0)(1, 0, 1)_{24}$	1047.2	Si	Si	Si	Si	Si
(*)							
Todos los parámetros del modelo deben ser significativos							

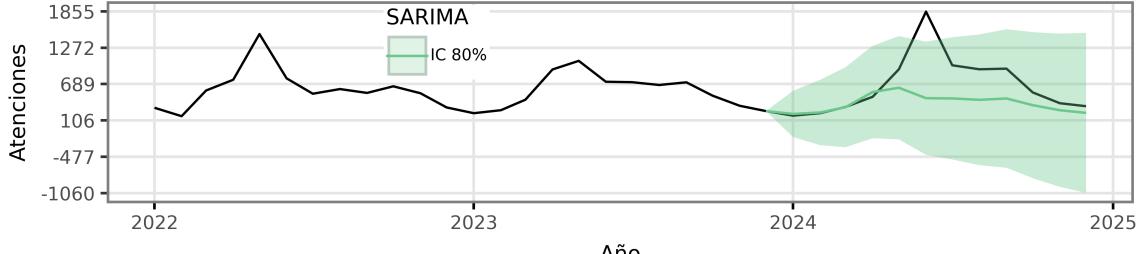
Con el propósito de no poblar de gráficos el documento, tanto las salidas como la comprobación de supuestos de todos los modelos se encuentran en los anexos 8.2 y 8.3 respectivamente. La tabla 2 resume la información de la comprobación de supuestos.

Tabla 2: Cumplimiento de los supuestos de los modelos ajustados.

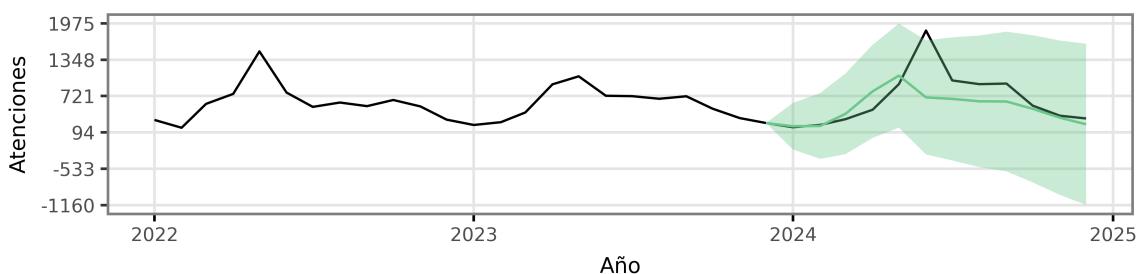
ID	Modelo	Normalidad	Homocedasticidad	Incorrelación
Atenciones en guardia				
AT-1	$SARIMA(1, 0, 0)(0, 1, 1)_{12}$	Si	Si	No
AT-2	$SARIMA(0, 1, 0)(1, 0, 0)_{12}$	Si	Si	Si
AT-3	$SARIMA(0, 1, 1)(0, 1, 0)_{12}$	Si	Si	Si
AT-4	$SARIMA(0, 1, 0)(0, 1, 1)_{12}$	Si	Si	Si
Trabajadores				
TR-1	$SARIMA(1, 1, 0)(1, 0, 0)_{12}$	No*	Si	Si
TR-2	$SARIMA(1, 1, 0)(1, 0, 1)_{12}$	No*	Si	Si
TR-3	$SARIMA(2, 0, 0)(2, 1, 0)_{12}$	No	No	No
Temperatura				
TE-1	$SARIMAX(1, 1, 1)(2, 0, 1)_{24}$	No*	Si	Si
TE-2	$SARIMAX(1, 1, 1)(1, 0, 1)_{24}$	No*	Si	Si
TE-3	$SARIMAX(1, 1, 0)(2, 0, 1)_{24}$	No*	Si	Si
TE-4	$SARIMAX(1, 1, 0)(1, 0, 1)_{24}$	No*	Si	Si
(*)				
Si bien se rechaza la normalidad, esto se puede deber a ciertos <i>outlayers</i> .				

A partir de los resultados presentados en las tablas 1 y 2 los modelos candidatos seleccionados y con los que se pronostican las series son:

- Modelos AT-2 y AT-4 para la serie de atenciones
- Modelos TR-1 y TR-2 para la serie de trabajadores
- Modelo TE-4 para la serie de temperaturas.



(a) Pronósticos con el modelo AT-2



(a) Pronósticos con el modelo AT-4

Figura 27: Pronósticos del número de atenciones en guardia por patologías respiratorias con SARIMA.

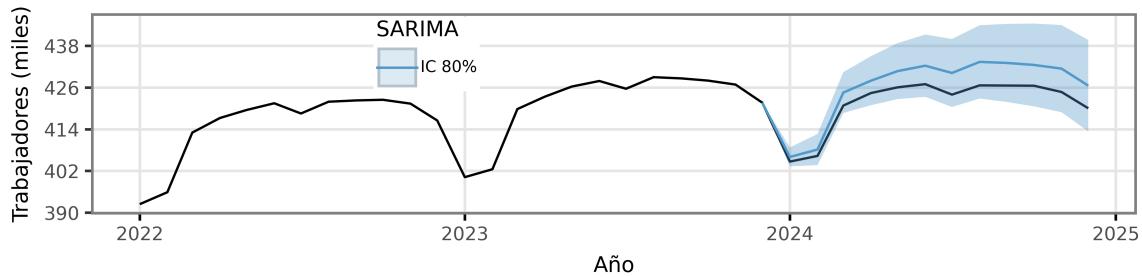
Tabla 3: Métricas de evaluación para la serie del número de atenciones en guardia por patologías respiratorias con SARIMA.

ID	Modelo	Horizonte	MAPE	<i>Interval Score</i>
AT-2	$SARIMA(0, 1, 0)(1, 0, 0)_{12}$	3	0.0694	1023.0534
		6	0.2391	2138.0755
		12	0.3328	2205.1062
AT-4	$SARIMA(0, 1, 0)(0, 1, 1)_{12}$	3	0.1529	1108.9433
		6	0.3157	1730.9364
		12	0.2794	2096.9139

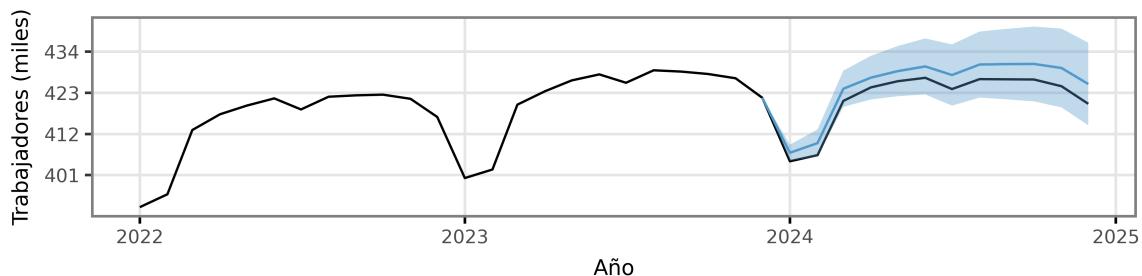
Se observa en la tabla 3 que para la serie de atenciones en guardia por patologías respiratorias el modelo seleccionado de forma automática, identificado como AT-2, parece pronosticar mejor en los primeros 6 meses. Sin embargo, este no logra captar el aumento de atenciones en invierno, y es por esto que en el pronóstico a 12 meses el modelo AT-4 obtiene mejores resultados. Se puede destacar la alta incertidumbre del modelo gracias a la amplitud de los intervalos de confianza y los altos valores en los *Interval Scores*.

Un problema a destacar es que el intervalo de confianza para las atenciones por guardia toma valores negativos, algo ilógico para variables de conteo. Este problema no es único de ARIMA y

se verá reflejado en los pronósticos de los demás modelos. Una solución es ajustar los modelos sobre el logaritmo de la variable, y luego transformando los valores pronosticados a la escala original. Esto queda propuesto para una futura extensión.



(a) Pronósticos con el modelo TR-1



(a) Pronósticos con el modelo TR-2

Figura 29: Pronóstico del número de trabajadores asalariados en el rubro de la enseñanza privada con SARIMA.

Tabla 4: Métricas de evaluación para la serie del número de trabajadores asalariados en el rubro de la enseñanza privada con SARIMA.

ID	Modelo	Horizonte	MAPE	<i>Interval Score</i>
TR-1	<i>SARIMA</i> (1, 1, 0)(1, 0, 0) ₁₂	3	0.0056	23.0939
		6	0.0081	12.3815
		12	0.0116	17.7216
TR-2	<i>SARIMA</i> (1, 1, 0)(1, 0, 1) ₁₂	3	0.0071	29.1066
		6	0.0068	10.3768
		12	0.0085	14.8479

Los pronósticos de los modelos AT-1 y AT-2 sobre el número de trabajadores asalariados en el rubro de la enseñanza privada son cercanos a los valores reales de la serie. Ambos logran aprender correctamente el patrón de los datos, logrando a su vez medidas satisfactorias. El modelo AT-1 se destaca en el pronóstico a corto plazo mientras que el AT-2, cuyo AIC es menor, lo supera luego de los 3 primeros meses, logrando un MAPE del 0.0085 pronosticando a largo plazo. Esto significa que la diferencia media absoluta entre los valores reales y los pronosticados es únicamente del 0.8513%.

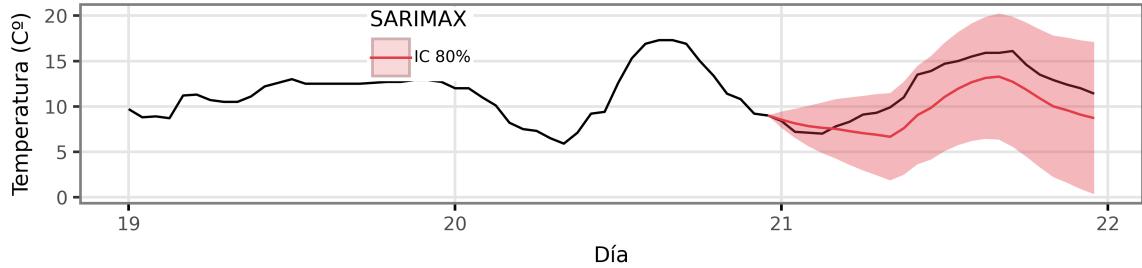


Figura 30: Pronóstico de la temperatura por hora en la ciudad de Rosario con el modelo SARIMAX TE-4.

Tabla 5: Métricas de evaluación para la serie de temperaturas por hora en la ciudad de Rosario con SARIMAX.

ID	Modelo	Horizonte	MAPE	<i>Interval Score</i>
TE-4	<i>SARIMA</i> (1, 1, 0)(1, 1, 0) ₂₄	6	0.0829	4.8344
		12	0.1869	7.3556
		24	0.1974	10.915

La diferencia media absoluta entre los valores reales y los pronosticados es de apenas del 483.4397% en el pronóstico del modelo TE-4 sobre las temperaturas en las primeras 6 horas. Sin embargo, esta diferencia aumenta en gran medida en los demás horizontes, al igual que la amplitud del intervalo de confianza, generando *Interval Scores* altos a medida que se aumenta el horizonte.

4.2.2 Modelos de aprendizaje automático

Los modelos de pronóstico basados en el aprendizaje automático no tienen supuestos que cumplir, por lo que el modelaje se vuelve sencillo y automatizable. Algunos aspectos importantes en lo que se debe tener especial cuidado y atención es en la selección de características y parámetros del modelo.

Tanto para XGBoost como para LightGBM las características elegidas para todas las series son las siguientes:

- Identificación temporal
- El promedio de las 3 observaciones anteriores
- El desvío estándar de las 3 observaciones anteriores
- El valor del primer rezago
- El valor del segundo rezago
- El valor del rezago estacional

Con el objetivo de seleccionar los mejores hiperparámetros, se elaboró una grilla con diferentes combinaciones de valores para cada parámetro. Se ajustaron modelos con cada combinación en la grilla y se evaluaron sobre un conjunto de validación.

XGBoost y LightGBM permiten parametrizar los modelos de varias formas, los parámetros que se decidieron probar en este trabajo son los siguientes:

- Número de árboles que se contruyen paralelamente en cada iteración (A). Opciones: 20, 50, 100, 150.
- Profundidad máxima del árbol (P). Opciones: 2, 3, 4, 5.
- Número máximo de hojas del árbol (H). Opciones: 2, 4, 8, 16.
- Tasa de aprendizaje en el método del gradiente (η). Opciones: 0.001, 0.1, 0.2.
- Proporción de características que se usa en cada árbol (C). Opciones: 0.7, 1.

En las tablas 6 y 7 se muestran para cada serie que combinación de parámetros, de las 384 posibles, fue la que menor MAPE produjo sobre el conjunto de validación, aplicando XGBoost o LightGBM respectivamente. Además, se presenta el MAPE e *Interval Score* que devuelve el modelo entrenado con todos los datos de entrenamiento y evaluado sobre el conjunto de prueba.

Tabla 6: Modelos XGBoost seleccionados y métricas de evaluación.

Serie	Hor.	A	P	H	η	C	MAPE	Interval Score
Atenciones	3	50	2	2	0.2	0.7	0.8615	1454.4965
	6	150	2	2	0.001	0.7	1.2669	1849.6949
	12	100	2	2	0.2	1.0	0.3684	1515.9289
Trabajadores	3	150	2	2	0.2	1.0	0.0083	20.514
	6	150	2	2	0.2	1.0	0.0091	20.514
	12	150	5	2	0.2	1.0	0.0069	21.9308
Temperatura	6	50	3	2	0.1	1.0	0.2775	35.6635
	12	50	2	2	0.2	1.0	0.2995	36.3026
	24	50	2	2	0.2	1.0	0.0583	2.7636

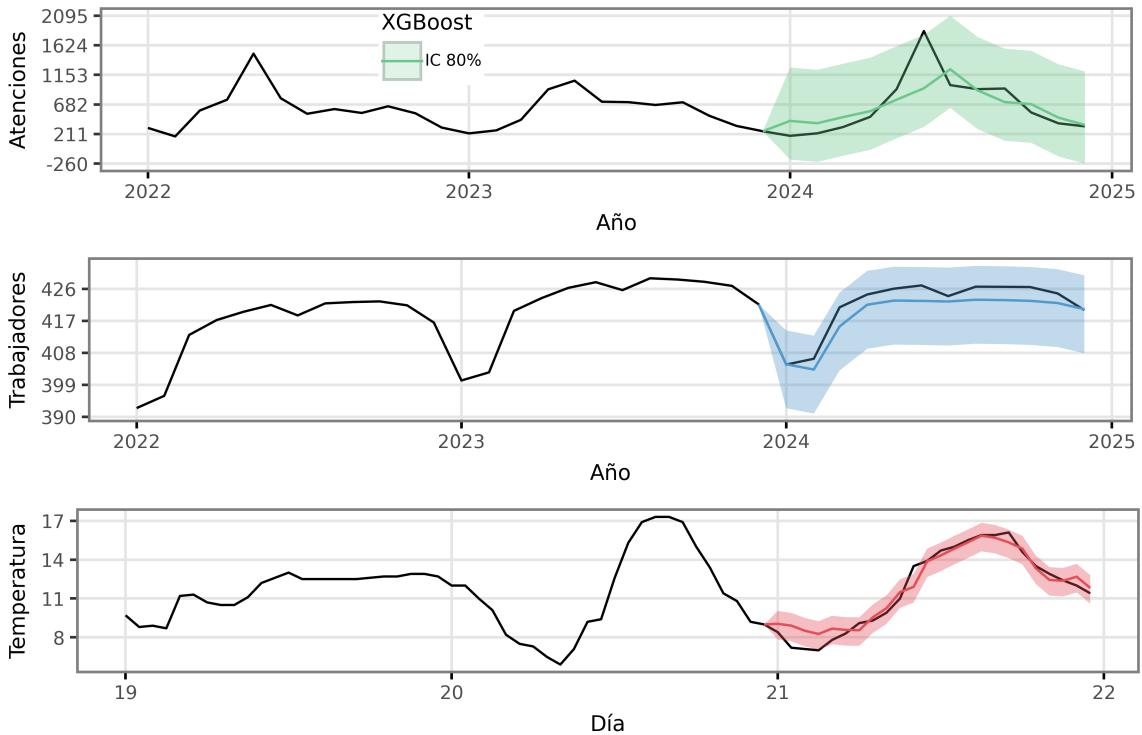


Figura 31: Pronósticos con XGBoost.

Los pronósticos con XGBoost fueron satisfactorios sobre las 3 series. Es importante notar que en la serie de atenciones el pronóstico puntual no capta del todo el pico de atenciones en invierno, sin embargo sí está contemplado por el intervalo de confianza del 80%. El intervalo se obtuvo usando *Ensemble Batch Prediction Intervals*. Otro aspecto interesante es que los intervalos para la serie de temperaturas son estrechos, lo que indica una mejora en la incertidumbre del modelo con respecto a los resultados vistos con el modelo SARIMAX.

Tabla 7: Modelos LightGBM seleccionados y métricas de evaluación.

Serie	Hor.	A	P	H	η	C	MAPE	Interval Score
Atenciones	3	150.0	2.0	4.0	0.2	0.7	0.8639	1559.4147
	6	150.0	2.0	4.0	0.001	0.7	1.2655	1846.2058
	12	100.0	2.0	4.0	0.2	1.0	0.3781	1537.1091
Trabajadores	3	20.0	2.0	2.0	0.001	1.0	0.0144	32.417
	6	150.0	2.0	4.0	0.2	1.0	0.0092	23.8802
	12	150.0	3.0	4.0	0.2	0.7	0.0089	25.2876
Temperatura	6	150.0	5.0	16.0	0.1	0.7	0.265	33.1287
	12	100.0	5.0	16.0	0.1	1.0	0.2507	29.8732
	24	100.0	2.0	2.0	0.2	1.0	0.0931	4.6673

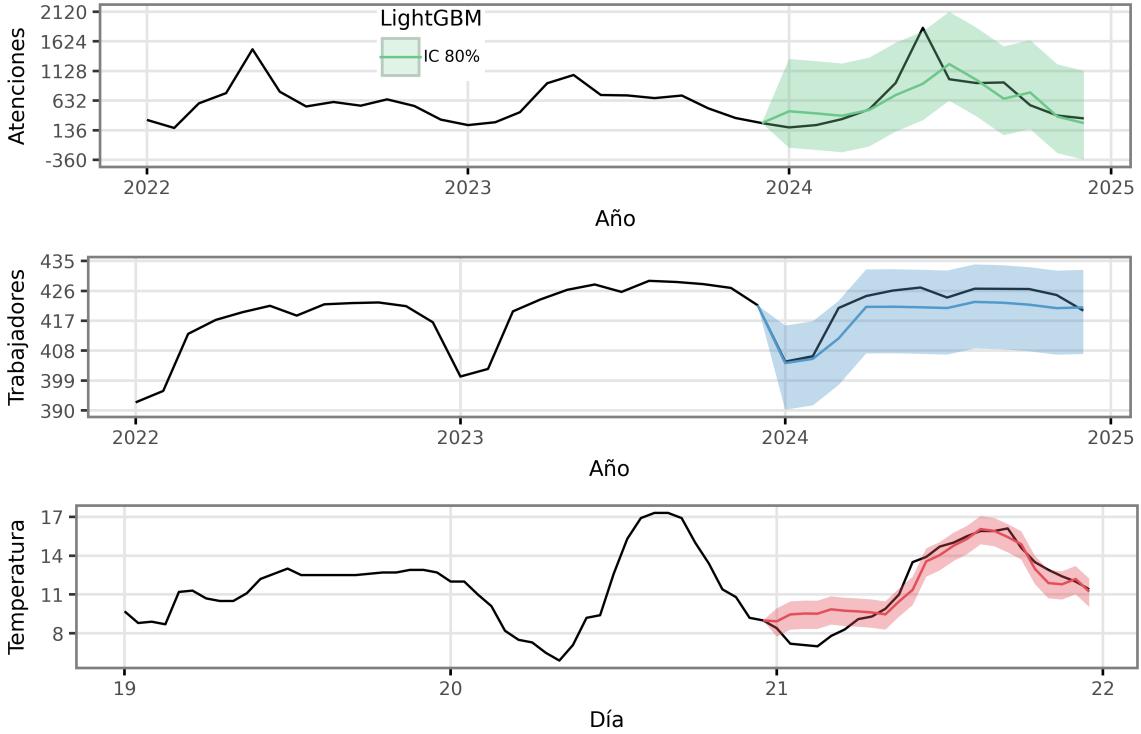


Figura 32: Pronósticos con LightGBM.

Los pronósticos con LightGBM son ligeramente peores a los conseguidos con XGBoost en todos los casos. Esta diferencia es especialmente importante en la serie de temperaturas, ya que en el pronóstico a 24 horas los intervalos de confianza en las primeras 6 horas no suelen cubrir a las observaciones, lo que podría indicar una subestimación en la variabilidad de los pronósticos. Este modelo tampoco pudo identificar el pico de atenciones en guardia en los meses invernales.

Resulta interesante destacar en las tablas 6 y 7 como XGBoost suele elegir bosques más sencillos, con menos árboles, que a su vez son menos profundos y tienen menos hojas. Por otro lado, se ve también la tendencia de LightGBM a expandir los árboles por las hojas. También se observa que los modelos optaron, en general, por elegir una tasa de aprendizaje y una proporción de características utilizadas altas, elecciones que podrían llevar a un sobreajuste. Aún así, esto no se vio evidenciado en los resultados.

4.2.3 Redes neuronales (LSTM)

Los modelos basados en aprendizaje profundo se encargan de definir las características más relevantes y los parámetros más adecuados de forma automática. En las redes neuronales, como es el caso de LSTM, solo se tendrá que elegir la estructura del modelo.

Las redes neuronales tienen tendencia a sobreajustar, para evitar esto existen numerosas alternativas. En primer lugar se puede optar por detener el entrenamiento antes de completar todas las iteraciones, si es que no se ve mejora en la función de pérdida, esto se conoce como *early stopping* o detención temprana en español. Otra opción es “ignorar” una cierta proporción de neuronas en cada iteración, que es equivalente a entrenar distintas redes neuronales. Esta última técnica es denominada *dropout*.

Para cada serie se entrenaron modelos LSTM con 300 iteraciones de entrenamiento y una tasa de aprendizaje de 0.001. Se adoptó una paciencia para el *early stopping* de 10, lo cual quiere decir que si la función de pérdida no muestra mejoras significativas en 10 iteraciones seguidas se detiene el entrenamiento. Además, se probaron las siguientes características para la estructura del modelo:

- Capas y neuronas en el modelo (N). Opciones: [32], [12, 24], [24, 42].
- Rezagos con los que se entrena el modelo (R). Opciones: 1, 12, 24.
- Proporción de *dropout* en cada capa (D). Opciones: 0.1, 0.3.
- Función de activación (A). Opciones: ReLu, tangente hiperbólica (tanh).

Tabla 8: Modelos LSTM seleccionados y métricas de evaluación.

Serie	Hor.	N	R	D	A	MAPE	Interval Score
Atenciones	3	[12, 24]	24	0.3	relu	0.7485	759.3464
	6	[24, 42]	1	0.3	tanh	0.244	1318.8364
	12	[32]	1	0.3	tanh	0.2996	913.0587
Trabajadores	3	[32]	24	0.1	tanh	0.0075	20.8112
	6	[24, 42]	24	0.1	tanh	0.0058	5.6342
	12	[32]	24	0.1	relu	0.007	14.3802
Temperatura	6	[32]	24	0.1	tanh	0.0804	1.9593
	12	[24, 42]	12	0.1	relu	0.1896	6.6552
	24	[32]	12	0.3	tanh	0.169	7.2241

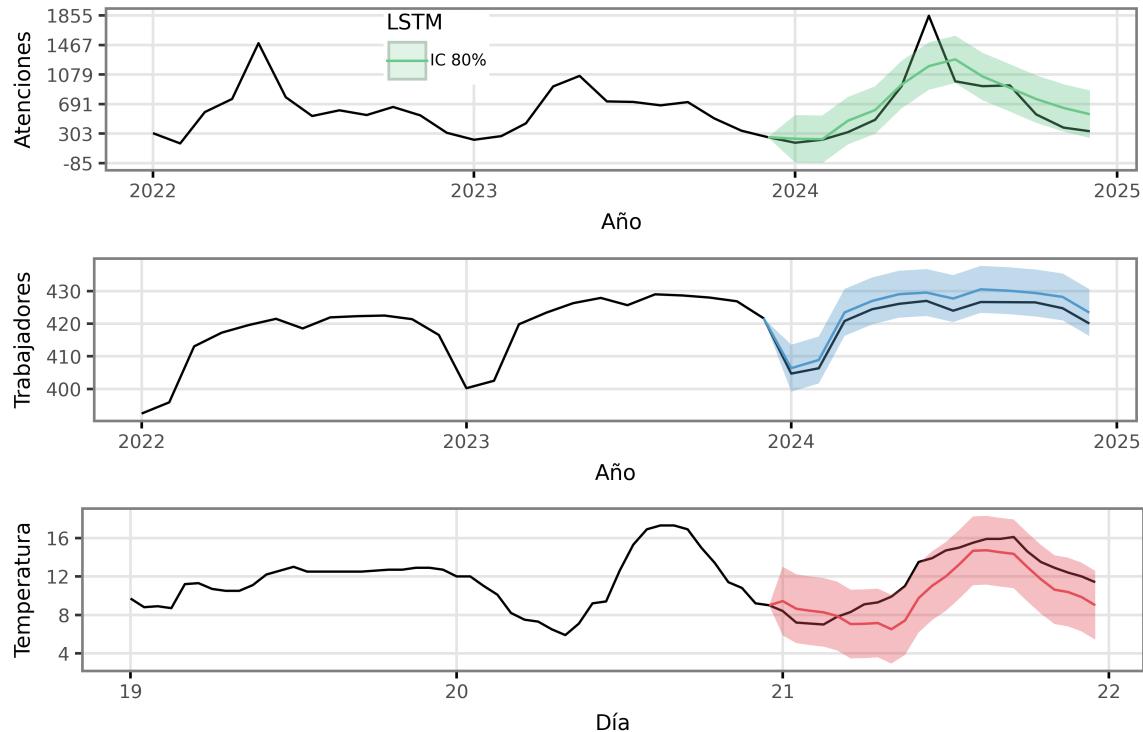


Figura 33: Pronósticos con LSTM.

Al igual que en los métodos de aprendizaje automatizado, los intervalos de confianza se obtuvieron usando *conformal predictions*, pero se realizaron con funciones ya integradas en la librería `scalecast`.

Los pronósticos sobre todas las series fueron satisfactorios, sin embargo, se destaca la falta de ajuste en el pico de atenciones por guardia. Se puede apreciar como en la serie de atenciones, que no tenía un patrón estacional muy marcado, ajustaron mejor aquellos modelos con un solo rezago de entrenamiento, mientras que en la serie de trabajadores, la cual contaba con una estacionalidad marcada, ajustaron mejor aquellos con 24 rezagos. Además, estas dos series se ven diferenciadas por la proporción de *dropout* con la que cuentan los modelos seleccionados. Estos aspectos tal vez se deban a las diferencias en variabilidad estacional que tienen las series, observable a partir del gráfico 16. Otra curiosidad es la presencia de redes de una sola capa de 32 neuronas para los pronósticos a largo plazo, mientras que aquellas redes con dos capas, de 24 y 42 neuronas, son especialmente prominentes en los pronósticos a medio plazo.

4.2.4 Modelos fundacionales (TimeGPT y Chronos)

La ventaja de los modelos fundacionales sobre las redes neuronales convencionales radica en que el ajuste de parámetros se realiza con un preentrenamiento en grandes conjuntos de datos. Esto significa que no se requiere ninguna intervención manual. No es necesario definir características, ajustar parámetros, ni especificar la forma del modelo. Este modo de pronóstico se conoce como *zero-shot*.

Si se desea un ajuste más controlado sobre la serie se podría hacer uso del “ajuste fino” (*fine tuning*). El ajuste fino consiste en evaluar la función de pérdida con los parámetros preestablecidos y realizar iteraciones extra de entrenamiento para ajustar el modelo específicamente al conjunto de datos, con el objetivo de minimizar aún más el error del pronóstico. Sin embargo, luego de numerosas pruebas, esta herramienta no logró

cambios significativos en el pronóstico de las series estudiadas por sobre las configuraciones base, logrando en ciertos casos resultados peores y aumentos excesivos en los tiempos de procesamiento. Es por esto que no se utiliza esta característica en los resultados finales, pero se la propone para futuras aplicaciones.

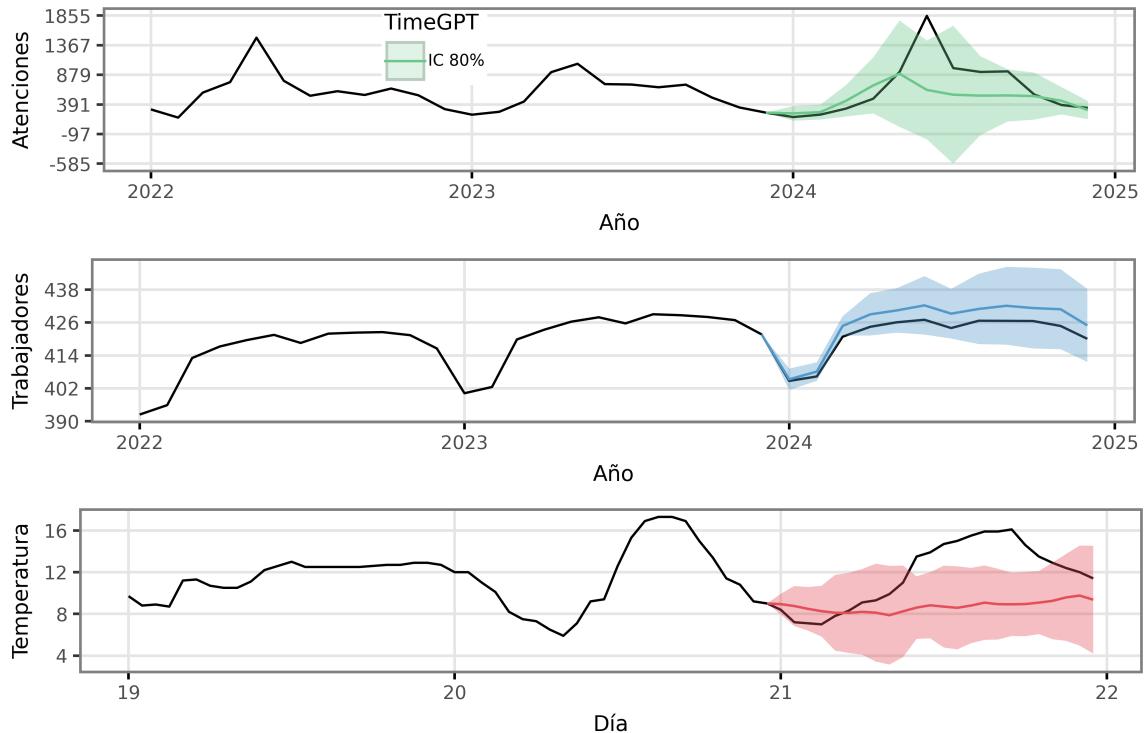


Figura 34: Pronósticos con TimeGPT.

Tabla 9: Métricas de evaluación para los ajustes con TimeGPT.

Serie	Hor.	MAPE	<i>Interval Score</i>
Atenciones	3	0.6542	1168.9787
	6	0.3651	793.6251
	12	0.3063	1276.4925
Trabajadores	3	0.0015	2.6232
	6	0.0025	11.1076
	12	0.0101	19.6513
Temperatura	3	0.238	9.212
	6	0.1219	5.1091
	12	0.2625	18.0171

Los pronósticos con TimeGPT sobre las series de temperatura y atenciones fueron regulares al largo plazo, donde no pudo detectar correctamente los patrones estacionales de las series. Por otro lado, si logra pronosticar correctamente las observaciones futuras en la serie de trabajadores. Los problemas de pronóstico de TimeGPT parecen estar en el corto y largo plazo, ya que en la tabla 9 se puede ver como para los pronósticos a medio plazo las métricas de evaluación son menores que para el resto de horizontes. Se observa en el intervalo de confianza para la serie de

atenciones como la incertidumbre en el período de invierno es mucho mayor que en otros puntos del ciclo estacional, período que fue especialmente complicado de pronosticar para varios de los modelos.

Chronos, al ser una familia de modelos, cuenta con múltiples opciones para realizar pronósticos *zero-shot*. En esta tesina se utiliza el modelo **bolt-small**, el cual cuenta con 48 millones de parámetros.

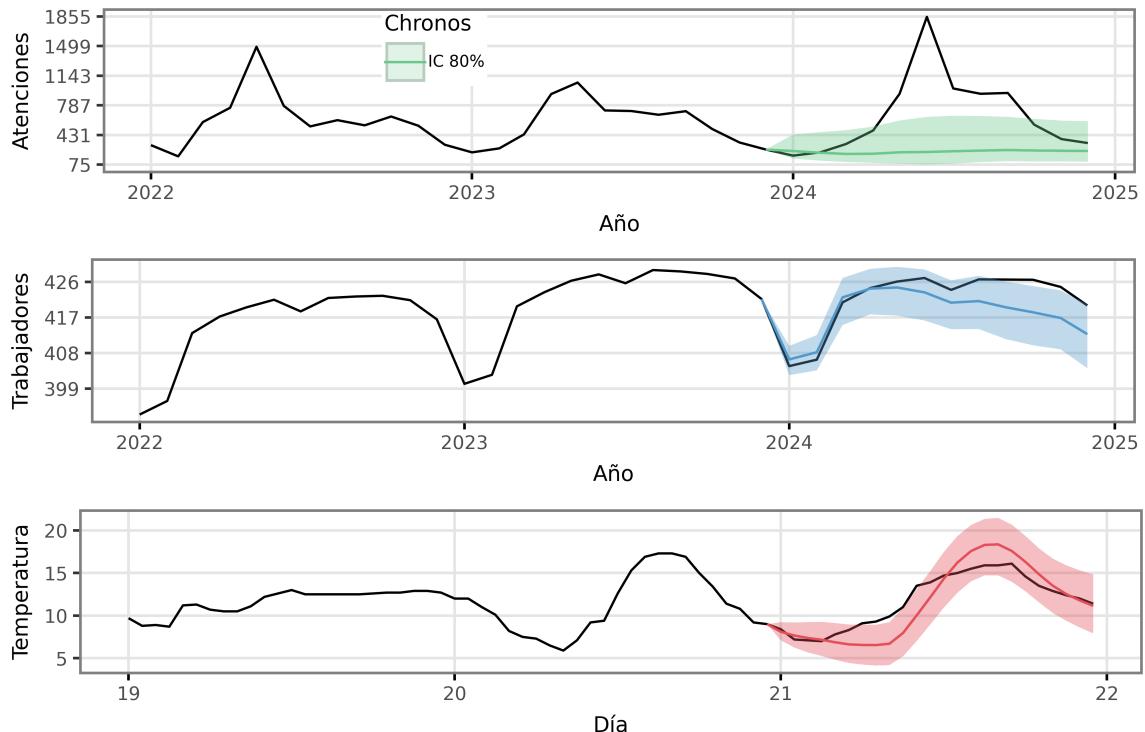


Figura 35: Pronósticos con Chronos.

Tabla 10: Métricas de evaluación para los ajustes con Chronos.

Serie	Hor.	MAPE	<i>Interval Score</i>
Atenciones	3	0.2312	338.2622
	6	0.4844	2961.5729
	12	0.5292	2479.0359
Trabajadores	3	0.0318	117.8766
	6	0.0212	67.4687
	12	0.0097	15.1275
Temperatura	3	0.3489	26.5617
	6	0.4502	49.5046
	12	0.125	6.2749

Chronos detecta correctamente los cambios de temperatura en las 24 horas, pero por otro lado, el pronóstico sobre las demás series es deficiente. Mientras que en la serie de atenciones, a medida que se incrementa el horizonte las métricas empeoran, para la serie de trabajadores mejoran.

4.3 Comparación de resultados y análisis final

Para las comparativas en la primera serie se utiliza el modelo SARIMA AT-4, dado que obtuvo mejores métricas y un AIC menor que el modelo AT-2, mientras que para la serie de trabajadores se utiliza el modelo TR-2.

Atenciones en guardia por patologías respiratorias en el hospital HNVV

Modelo	Horizonte 3		Horizonte 6		Horizonte 12	
	MAPE	Interval Score	MAPE	Interval Score	MAPE	Interval Score
ARIMA	0.1529	1108.9433	0.3157	1730.9364	0.2794	2096.9139
XGBoost	0.8615	1454.4965	1.2669	1849.6949	0.3684	1515.9289
LightGBM	0.8639	1559.4147	1.2655	1846.2058	0.3781	1537.1091
LSTM	0.7485	759.3464	0.2440	1318.8364	0.2996	913.0587
TimeGPT	0.6542	1168.9787	0.3651	793.6251	0.3063	1276.4925
Chronos	0.2312	338.2622	0.4844	2961.5729	0.5292	2479.0359

Figura 36: Comparación de métricas de evaluación entre distintos modelos y horizontes para la serie de atenciones.

El modelo ARIMA superó a los modelos más modernos en términos de MAPE a corto y largo plazo, pero sus *Interval Scores* son grandes a comparación de otros modelos con MAPEs similares, provocado por la gran amplitud en los intervalos de confianza. Por otro lado, los pronósticos de Chronos fueron considerablemente peores en los horizontes largos. Sin embargo, ningún modelo se destacó claramente en ninguno de los horizontes.

En comparación con el resto de series, el número de atenciones mensuales en guardia fue especialmente difícil de pronosticar, evidenciado por los altos MAPEs e *Interval Scores*.

Trabajadores asalariados en el rubro de educación privada

Modelo	Horizonte 3		Horizonte 6		Horizonte 12	
	MAPE	Interval Score	MAPE	Interval Score	MAPE	Interval Score
ARIMA	0.0071	29.1066	0.0068	10.3768	0.0085	14.8479
XGBoost	0.0083	20.5140	0.0091	20.5140	0.0069	21.9308
LightGBM	0.0144	32.4170	0.0092	23.8802	0.0089	25.2876
LSTM	0.0075	20.8112	0.0058	5.6342	0.0070	14.3802
TimeGPT	0.0015	2.6232	0.0025	11.1076	0.0101	19.6513

Figura 37: Comparación de métricas de evaluación entre distintos modelos y horizontes para la serie de trabajadores.

La clara tendencia y estacionalidad del número de trabajadores asalariados en el rubro de la enseñanza privada provocó que sea excepcionalmente fácil de pronosticar, logrando métricas de evaluación bajas en todos los horizontes. Se destacan sobre todo TimeGPT en el corto plazo y LSTM en los horizontes más largos.

Temperaturas horarias en Rosario

Modelo	Horizonte 6		Horizonte 12		Horizonte 24	
	MAPE	Interval Score	MAPE	Interval Score	MAPE	Interval Score
ARIMA	0.0829	4.8344	0.1869	7.3556	0.1974	10.9150
XGBoost	0.2775	35.6635	0.2995	36.3026	0.0583	2.7636
LightGBM	0.2650	33.1287	0.2507	29.8732	0.0931	4.6673
LSTM	0.0804	1.9593	0.1896	6.6552	0.1690	7.2241
TimeGPT	0.2380	9.2120	0.1219	5.1091	0.2625	18.0171
Chronos	0.3489	26.5617	0.4502	49.5046	0.1250	6.2749

Figura 38: Comparación de métricas de evaluación entre distintos modelos y horizontes para la serie de temperaturas.

Para la serie de temperaturas, la cual cuenta con la humedad y la presión atmosférica como variables exógenas, tanto ARIMA como LSTM tuvieron resultados superiores al resto de modelos en el corto plazo. Pronosticando medio día hacia adelante, TimeGPT y LSTM fueron los modelos con mejor rendimiento, mientras que en el pronóstico de un día entero, XGBoost superó con creces al resto de modelos.

5. Conclusiones

Se evidenció que ningún modelo es universalmente superior al resto ni existe combinación única de hiperparámetros que logre el mejor ajuste en cualquier circunstancia. A lo largo del trabajo se observaron las diferencias, ventajas y desventajas de cada método.

Los modelos ARIMA requieren un ajuste manual, que además de ser demandante temporalmente, su aplicación a diferentes series implicaría comenzar desde el inicio. En términos de resultados, logró MAPEs bajos en comparación al resto de modelos en casi todos los escenarios, sin embargo, evidenciaron mucha incertidumbre en sus resultados, provocando que la amplitud de sus pronósticos probabilísticos desencadenaran *Interval Scores* altos.

Los algoritmos de aprendizaje automático precisan la definición de una serie de características que facilitan a los modelos a ajustarse a los datos. Ante nuevas series, sería prudente cambiar las características a otras más acordes a los datos. En relación a otros modelos, no se destacaron especialmente en ninguna instancia, con excepción de XGBoost pronosticando a largo plazo la serie de temperaturas. No obstante, los algoritmos de aprendizaje automatizado obtuvieron resultados acertados consistentemente en todas las series analizadas y con baja incertidumbre.

El único requerimiento para ajustar redes neuronales LSTM es la definición de la estructura del modelo, ya que tanto el ajuste de hiperparámetros como la elección de características se realizan de forma automática. A pesar de haber sido el modelo de pronóstico más exigente computacionalmente, obtuvo métricas favorables a través de todas las series y horizontes.

Finalmente, los modelos basados en arquitecturas *transformer*, como TimeGPT y Chronos, no precisan de ningún ajuste manual, únicamente los datos y su periodicidad son necesarios como entrada para los modelos. TimeGPT tuvo problemas para detectar los patrones estacionales de las series, sin embargo, logró buenas métricas en el corto y medio plazo. Chronos, por otro lado, presentó resultados menos satisfactorios. Esto podría solucionarse eligiendo otro modelo de la familia de Chronos diferente a **bolt-small**, o haciendo uso del ajuste fino. Cabe destacar que, si bien TimeGPT y Chronos destacan por su facilidad de uso, son poco personalizables, dejando pocas alternativas ante ajustes deficientes. TimeGPT tiene además la problemática de no ser de código abierto, lo que evita conocer el proceso con el que se ajustan los modelos.

A la fecha en la que se comenzó este trabajo, la API de TimeGPT ofrecía un plan gratuito con un número de consultas mensuales limitadas, sin embargo, a lo largo del desarrollo los planes de suscripción cambiaron. Se eliminó el plan gratuito y se reemplazó por un mes de prueba con consultas limitadas para usuarios nuevos únicamente. Este cambio afectó significativamente el desarrollo del trabajo.

Este documento presenta tres contribuciones claves al campo de la estadística. En primer lugar, la introducción de los modelos transformadores para el pronóstico de series temporales. En segundo lugar, se exploró el *Interval Score* como medida del error para evaluar el desempeño de los modelos ante pronósticos probabilísticos. Por último, se presenta la construcción de intervalos de pronóstico por medio de *conformal predictions*, que no depende de conocer la distribución de los residuos.

6. Mejoras y extensiones a la investigación

En esta tesina se buscó abordar el tema de la forma más amplia posible sin sacrificar profundidad. Sin embargo, por la amplitud del mismo se dejaron fuera numerosos temas interesantes que se podrían tratar en otros trabajos.

En primer lugar, se podrían estudiar series con características distintas y aplicar las respectivas correcciones a aquellas series acotadas, como el caso de la cantidad de atenciones en guardia y el número de trabajadores, que al ser variables de conteo no pueden ser negativas.

En la selección de los modelos, se eligió como mejor aquel que minimizara el MAPE, pero se podría haber hecho la elección en base al *Interval Score* o alguna otra medida de error. A su vez, podría explorarse el uso de distintas medidas de error probabilísticas diferentes al *Interval Score*, tales como *Scaled quantile loss*, *Weighted quantile loss* o *Implicit quantile loss*.

Boosting y el ensamblaje de modelos no está limitado únicamente a los árboles de decisión, por lo que se prone indagar como funcionan estas técnicas en otros modelos, como en redes neuronales.

Para obtener pronósticos probabilísticos en los algoritmos de aprendizaje automático se optó por EnbPI, pero queda propuesto probar otros métodos o alternativas, como *Natural Gradient Boosting* (NGBoost).

Al comparar los modelos también se pudo haber estudiado el tiempo de cómputo en los ajustes. Una prueba de esto fue realizada, pero los métodos utilizados para obtener estos resultados no fueron del todo adecuados, y ante la falta de alternativas se decidió recortar estos resultados y no mostrarlos empíricamente. Sin embargo, se hicieron menciones a los resultados a lo largo del trabajo. A raíz de esto, se propone estudiar distintas metodologías para medir los tiempos de cómputo de los modelos.

Otra expansión a la investigación se puede dar en el ajuste de hiperparámetros, y características en el caso de los algoritmos de aprendizaje automatizado. Si bien con la búsqueda de parámetros se intentó abordar múltiples opciones, por cuestiones de tiempo y exigencia computacional es imposible explorarlas todas. Es por esto que aún queda un amplio campo de investigación en este aspecto y en la aplicación del ajuste fino en modelos fundacionales preentrenados.

7. Bibliografía

- Alammar, J.** (27 de junio de 2018). *The Illustrated Transformer*. Jay Alammar. <https://jalammar.github.io/illustrated-transformer/>
- Ansari et al.** (2024). *Chronos: Learning the Language of Time Series*. Transactions on Machine Learning Research. <https://arxiv.org/abs/2403.07815>
- Awan, A.** (2 de septiembre de 2024). *Time Series Forecasting With TimeGPT*. Datacamp. <https://www.datacamp.com/tutorial/time-series-forecasting-with-time-gpt>
- Bermejo, J.** (21 de mayo de 2024). *Redes neuronales*. Facultad de Ciencias Económicas y Estadística de la Universidad Nacional de Rosario.
- Chen, Y., Yao, X.** (2023). *Conformal prediction for time series*. Proceedings of Machine Learning Research. <https://arxiv.org/abs/2010.09107>
- Elhariri, K.** (1 de marzo de 2022). *The Transformer Model*. Medium. <https://medium.com/data-science/attention-is-all-you-need-e498378552f9>
- GeeksforGeeks.** (s.f.). *What is LSTM – Long short term memory?*. Recuperado el 15 de julio de 2025 de <https://www.geeksforgeeks.org/deep-learning/deep-learning-introduction-to-long-short-term-memory/>
- Gilliland, M., Sglavo, U., & Tashman, L.** (2016). *Forecast Error Measures: Critical Review and Practical Recommendations*. John Wiley & Sons Inc.
- Gneiting, T., & Raftery A. E.** (2007). *Strictly Proper Scoring Rules, Prediction, and Estimation*. Journal of the American Statistical Association, 102(477), 359–378. <https://doi.org/10.1198/016214506000001437>
- Hyndman, R. J., & Athanasopoulos, G.** (2021). *Forecasting: principles and practice (3rd ed.)*. OTexts. <https://otexts.com/fpp3/>
- Hyndman, R.J., Athanasopoulos, G., Garza, A., Challu, C., Mergenthaler, M., & Olivares, K.G.** (2024). *Forecasting: Principles and Practice, the Pythonic Way*. OTexts. [OTexts.com/fpppy](https://otexts.com/fpppy).
- IBM.** (s.f.). *Explainers*. Recuperado el 14 de marzo de 2025 de <https://www.ibm.com/think/topics>
- Kamtziris, G.** (27 de febrero de 2023). *Time Series Forecasting with XGBoost and LightGBM: Predicting Energy Consumption*. Medium. <https://medium.com/@geokam/time-series-forecasting-with-xgboost-and-lightgbm-predicting-energy-consumption-460b675a9cee>
- Keith, M.** (22 de septiembre de 2023). *Five Practical Applications of the LSTM Model for Time Series, with Code*. Towards data science. <https://towardsdatascience.com/five-practical-applications-of-the-lstm-model-for-time-series-with-code-a7aac0aa85c0/>
- Korstanje, J.** (2021). *Advanced Forecasting with Python*. Apress.
- Nielsen, A.** (2019). *Practical Time Series Analysis: Prediction with Statistics and Machine Learning*. O'Reilly Media.
- Nixtla.** (s.f.). *About TimeGPT*. Recuperado en diciembre de 2024 de https://docs.nixtla.io/docs/getting-started-about_timegpt
- Parmezan, A., Souza, V., & Batista, G.** (1 de mayo de 2019). *Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the*

best conditions for the use of each model. Information Sciences. <https://www.sciencedirect.com/science/article/abs/pii/S0020025519300945>

Prunello, M., & Marfetán, D. (12 de mayo de 2024). *Árboles de decisión.* Facultad de Ciencias Económicas y Estadística de la Universidad Nacional de Rosario.

Sabino Parmezan, A. R., Souza, V. M. A., & Batista, G. E. A. P. A. (2019). *Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model.* Information Sciences, 484, 302–337. <https://doi.org/10.1016/j.ins.2019.01.076>

Sanderson, G. [3Blue1Brown]. (2024). *Attention in transformers, step-by-step / DL6 [Video].* Youtube. <https://www.youtube.com/watch?v=eMlx5fFNoYc&t=1204s>

Sanderson, G. [3Blue1Brown]. (2024). *Transformers (how LLMs work) explained visually / DL5 [Video].* Youtube. <https://www.youtube.com/watch?v=wjZofJX0v4M>

Shastri, Y. (26 de abril de 2024). *Attention Mechanism in LLMs: An Intuitive Explanation.* Datacamp. <https://www.datacamp.com/blog/attention-mechanism-in-langs-intuition>

Silberstein, E. (7 de noviembre de 2024). *Tracing the Transformer in Diagrams.* Medium. <https://medium.com/data-science/tracing-the-transformer-in-diagrams-95dbeb68160c>

Valeriy, M. (11 de agosto de 2023). *Demystifying EnbPI: Mastering Conformal Prediction Forecasting.* Medium. <https://valeman.medium.com/demystifying-enbpi-mastering-conformal-prediction-forecasting-d49e65532416>

Vaswani et al. (2017). *Attention is all you need.* Google. <https://arxiv.org/pdf/1706.03762>

8. Anexo

8.1 Gráficos estacionales

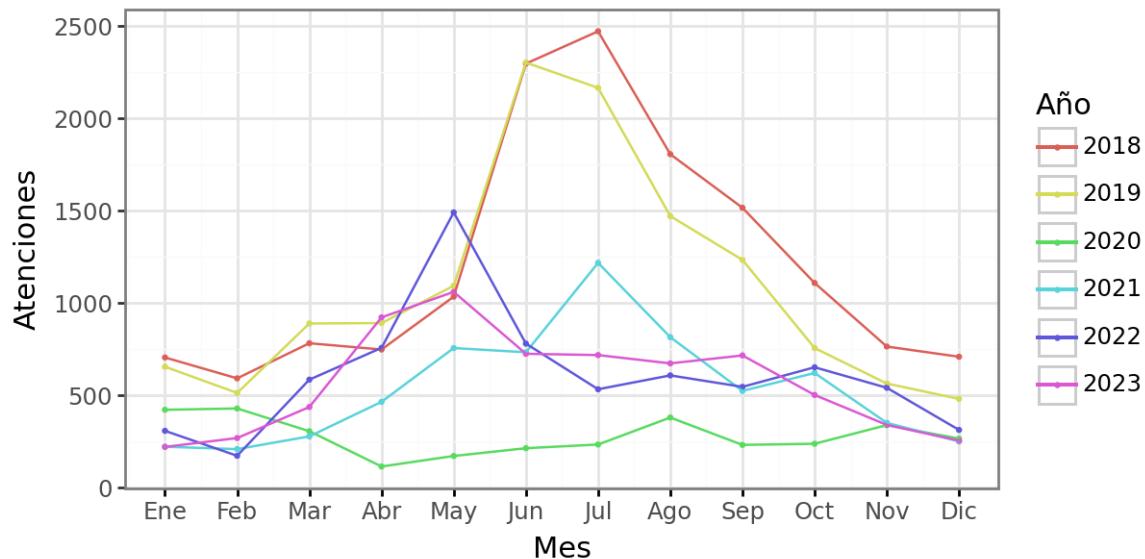


Figura 39: Atenciones por guardia mensuales por patologías respiratorias por año

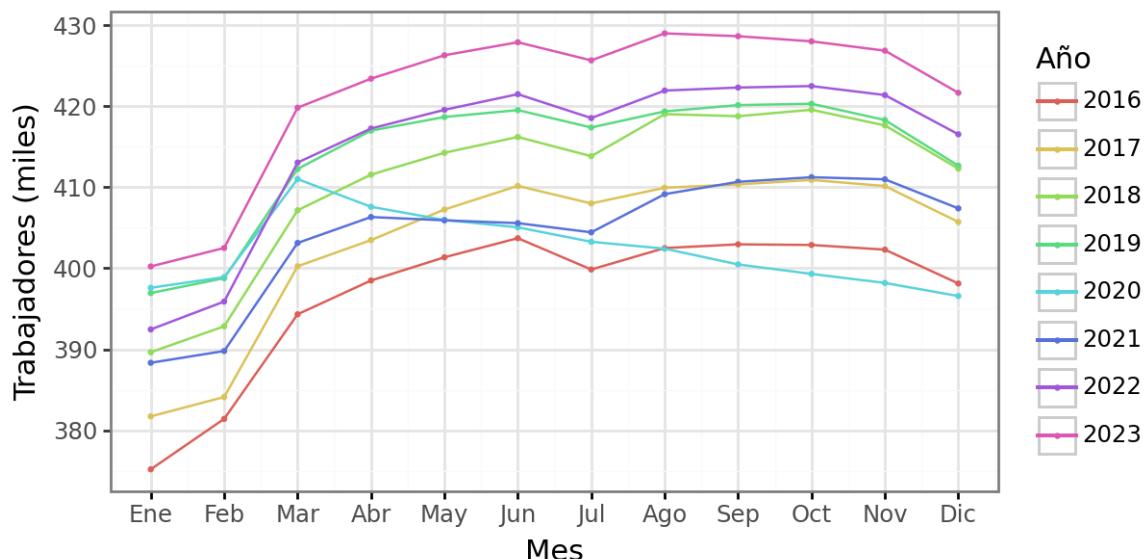


Figura 40: Número mensual de trabajadores asalariados en el área de enseñanza privada por año

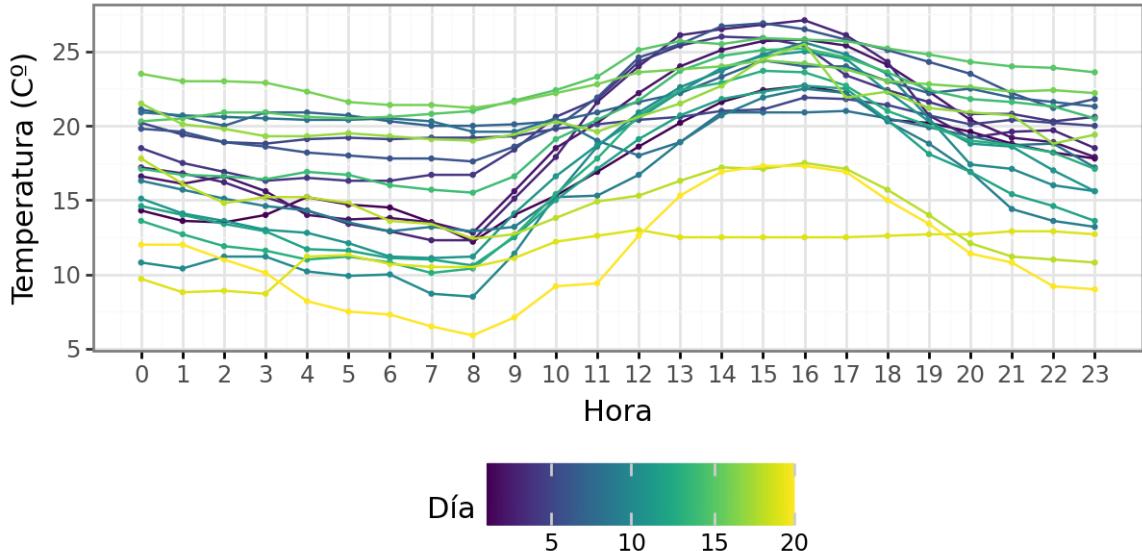


Figura 41: Atenciones por guardia mensuales por patologías respiratorias por año

8.2 Salidas de modelos ARIMA

Salidas de modelos ARIMA para la serie de atenciones

Tabla 11: Salida modelo ARIMA AT-1 para la serie de atenciones.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	-25.448525	-88.924446	38.027396	0.432000
ar.L1	0.765447	0.640224	0.890670	0.000000
ma.S.L12	-0.332922	-0.557123	-0.108721	0.003600
sigma2	85582.596206	64685.589734	106479.602679	0.000000
Modelo	SARIMA(1, 0, 0)(0, 1, 1, 12)		AIC	861.233900

Tabla 12: Salida modelo ARIMA AT-2 para la serie de atenciones.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
ar.S.L12	0.500890	0.369767	0.632014	0.000000
sigma2	83405.165734	61730.526000	105079.805467	0.000000
Modelo	SARIMA(0, 1, 0)(1, 0, 0, 12)		AIC	1013.497500

Tabla 13: Salida modelo ARIMA AT-3 para la serie de atenciones.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	-0.325716	-86.788755	86.137324	0.994100
ma.L1	-0.057733	-0.310244	0.194778	0.654100
sigma2	108637.574754	78101.077256	139174.072252	0.000000
Modelo	SARIMA(0, 1, 1)(0, 1, 0, 12)		AIC	857.589100

Tabla 14: Salida modelo ARIMA AT-4 para la serie de atenciones.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	0.694117	-64.910940	66.299173	0.983500
ma.S.L12	-0.291923	-0.502291	-0.081556	0.006500
sigma2	97997.311628	73338.891113	122655.732142	0.000000
Modelo	SARIMA(0, 1, 0)(0, 1, 1, 12)		AIC	852.062000

Salidas de modelos ARIMA para la serie de trabajadores

Tabla 15: Salida modelo ARIMA TR-1 para la serie de trabajadores.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	0.019800	-0.201100	0.240800	0.860200
ar.L1	0.307400	0.126000	0.488800	0.000900
ar.S.L12	0.946600	0.914000	0.979300	0.000000
sigma2	4.499100	3.592700	5.405600	0.000000
Modelo	SARIMA(1, 1, 0)(1, 0, 0, 12)		AIC	447.747400

Tabla 16: Salida modelo ARIMA TR-2 para la serie de trabajadores.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	0.000100	-0.001600	0.001700	0.941500
ar.L1	0.348400	0.203200	0.493700	0.000000
ar.S.L12	0.999700	0.994700	1.004700	0.000000
ma.S.L12	-0.906100	-1.620800	-0.191300	0.013000
sigma2	2.717600	1.205200	4.230000	0.000400
Modelo	SARIMA(1, 1, 0)(1, 0, 1, 12)		AIC	423.847900

Tabla 17: Salida modelo ARIMA TR-3 para la serie de trabajadores.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
ar.L1	1.372355	1.195117	1.549592	0.000000
ar.L2	-0.393721	-0.551523	-0.235918	0.000000
ar.S.L12	-0.519896	-0.709546	-0.330247	0.000000
ar.S.L24	-0.336315	-0.569505	-0.103126	0.004700
sigma2	3.307211	2.472612	4.141810	0.000000
Modelo	SARIMA(2, 0, 0)(2, 1, 0, 12)		AIC	356.697300

Salidas de modelos ARIMA para la serie de temperaturas

Tabla 18: Salida modelo ARIMA TE-1 para la serie de temperaturas.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	-0.000859	-0.013286	0.011568	0.892200
HUM	-0.007626	-0.010491	-0.004761	0.000000
PNM	-0.133841	-0.252529	-0.015154	0.027100
ar.L1	0.225100	0.035248	0.414951	0.020100
[H]	ma.L1	0.180523	-0.010026	0.371073
	ar.S.L24	0.979171	0.807724	1.150618
	ar.S.L48	-0.033935	-0.169718	0.101849
	ma.S.L24	-0.749386	-0.894063	-0.604709
	sigma2	0.496032	0.447138	0.544926
	Modelo	SARIMA(1, 1, 1)(2, 0, 1, 24)	AIC	1067.051400

Tabla 19: Salida modelo ARIMA TE-2 para la serie de temperaturas.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	-0.000044	-0.002397	0.002309	0.970600
HUM	-0.007228	-0.010041	-0.004414	0.000000
PNM	-0.231847	-0.352854	-0.110841	0.000200
ar.L1	0.522353	0.358452	0.686254	0.000000
ma.L1	-0.106620	-0.281739	0.068498	0.232700
ar.S.L24	0.986917	0.965591	1.008242	0.000000
ma.S.L24	-0.886257	-0.983218	-0.789295	0.000000
sigma2	0.468208	0.420775	0.515641	0.000000
Modelo	SARIMA(1, 1, 1)(1, 0, 1, 24)	AIC	1046.135800	

Tabla 20: Salida modelo ARIMA TE-3 para la serie de temperaturas.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	0.000314	-0.003770	0.004397	0.880400
HUM	-0.007706	-0.010548	-0.004865	0.000000
PNM	-0.126373	-0.251482	-0.001265	0.047700
ar.L1	0.456544	0.388087	0.525001	0.000000
ar.S.L24	1.063896	0.881826	1.245965	0.000000
ar.S.L48	-0.089158	-0.227106	0.048791	0.205200
ma.S.L24	-0.882658	-1.045724	-0.719592	0.000000
sigma2	0.509617	0.454175	0.565059	0.000000
Modelo	SARIMA(1, 1, 0)(2, 0, 1, 24)	AIC	1051.948400	

Tabla 21: Salida modelo ARIMA TE-4 para la serie de temperaturas.

Componente	Coeficiente	IC(0.025)	IC(0.975)	p-value
intercept	0.000200	-0.003100	0.003400	0.918700
HUM	-0.007300	-0.010100	-0.004600	0.000000
PNM	-0.128700	-0.246700	-0.010800	0.032400
ar.L1	0.496900	0.431000	0.562800	0.000000
ar.S.L24	0.981200	0.952400	1.010100	0.000000
ma.S.L24	-0.882500	-0.981300	-0.783700	0.000000
sigma2	0.483800	0.434900	0.532600	0.000000
Modelo	SARIMA(1, 1, 0)(1, 0, 1, 24)		AIC	1047.239600

8.3 Comprobación de supuestos de modelos arima

Comprobación de supuestos de modelos ARIMA para la serie de atenciones sobre los residuos estandarizados

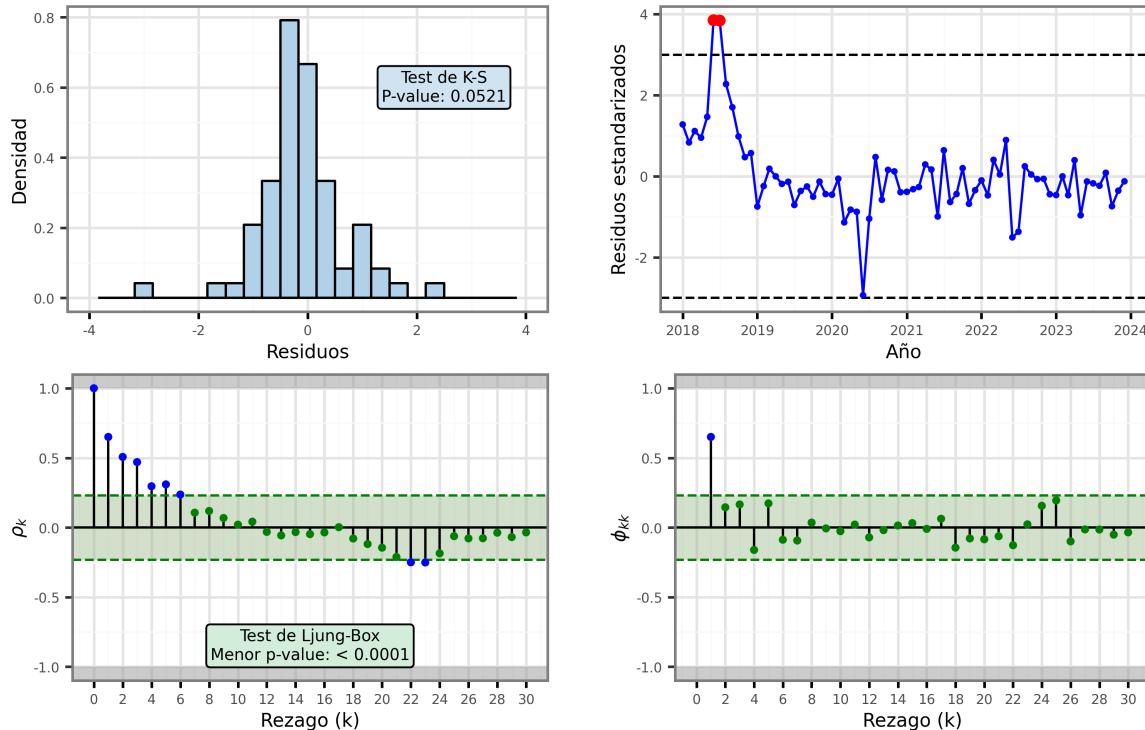


Figura 42: Comprobación de supuestos del modelo ARIMA AT-1 para la serie de atenciones.

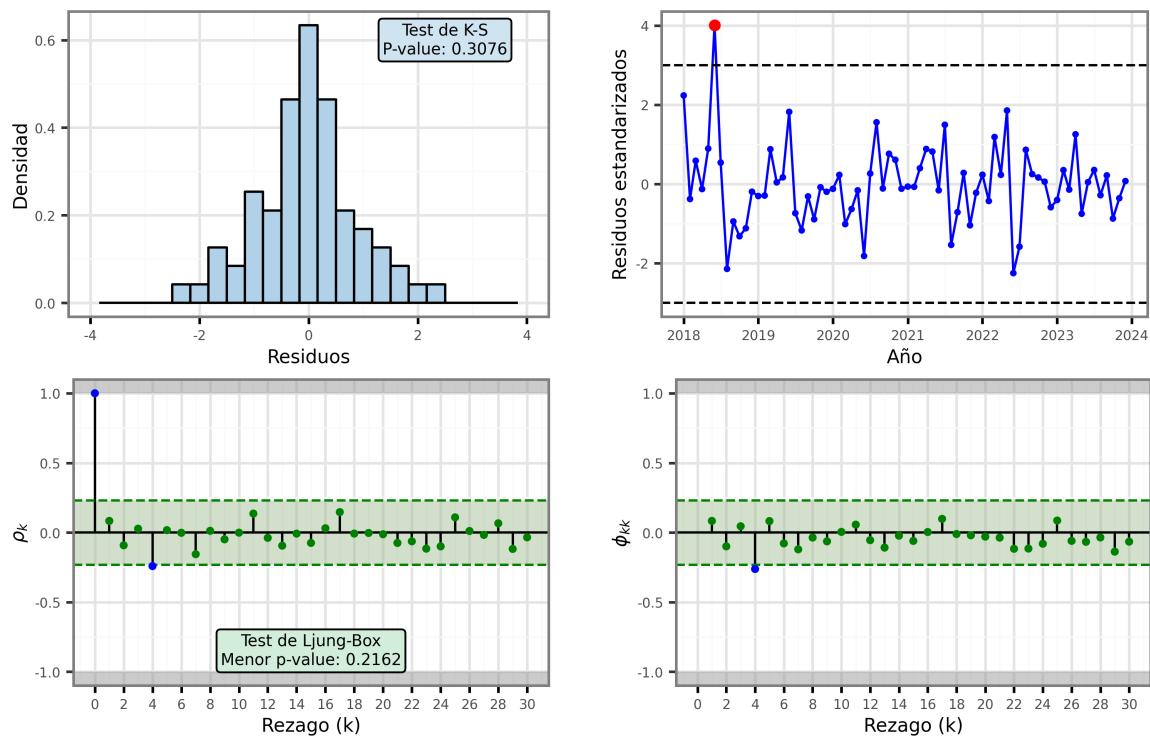


Figura 43: Comprobación de supuestos del modelo ARIMA AT-2 para la serie de atenciones.

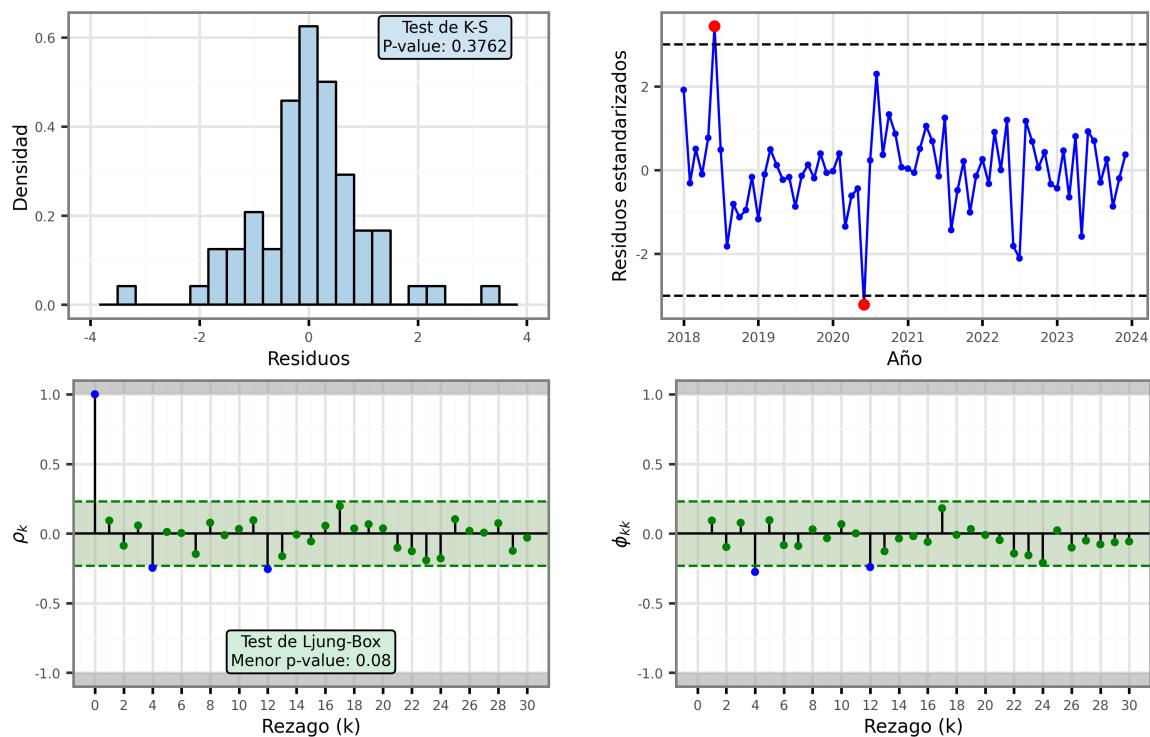


Figura 44: Comprobación de supuestos del modelo ARIMA AT-3 para la serie de atenciones.

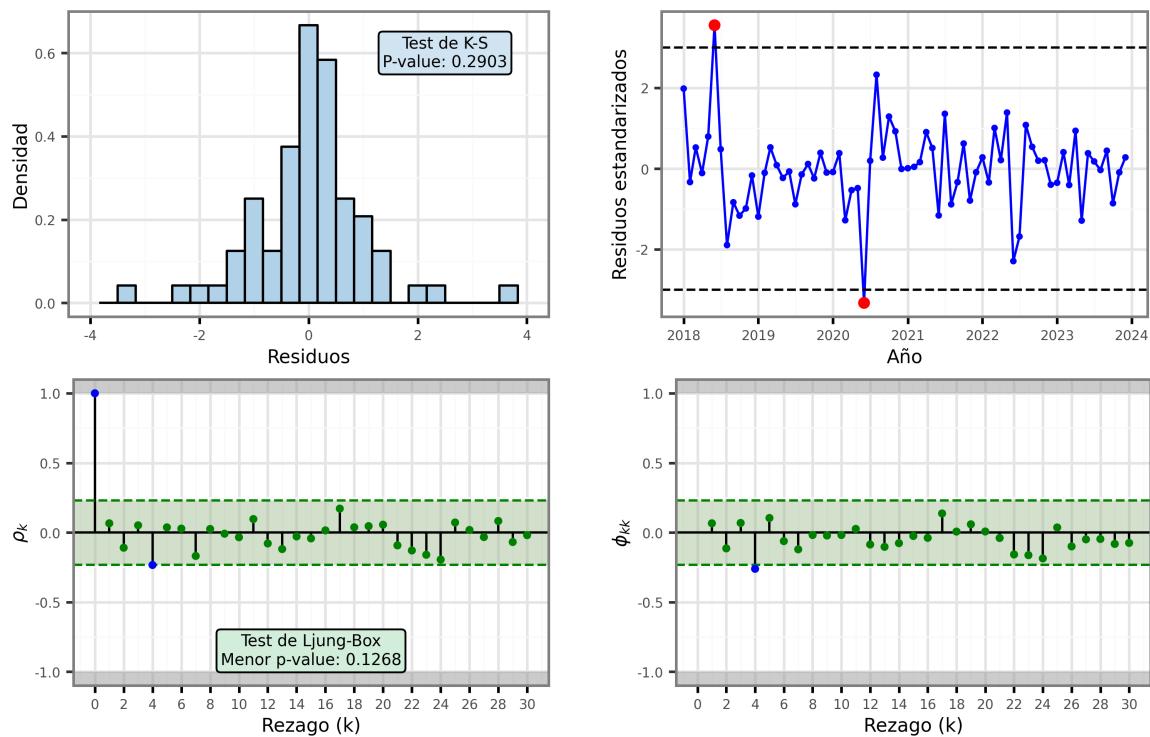


Figura 45: Comprobación de supuestos del modelo ARIMA AT-4 para la serie de atenciones.

Comprobación de supuestos de modelos ARIMA para la serie de trabajadores sobre los residuos estandarizados

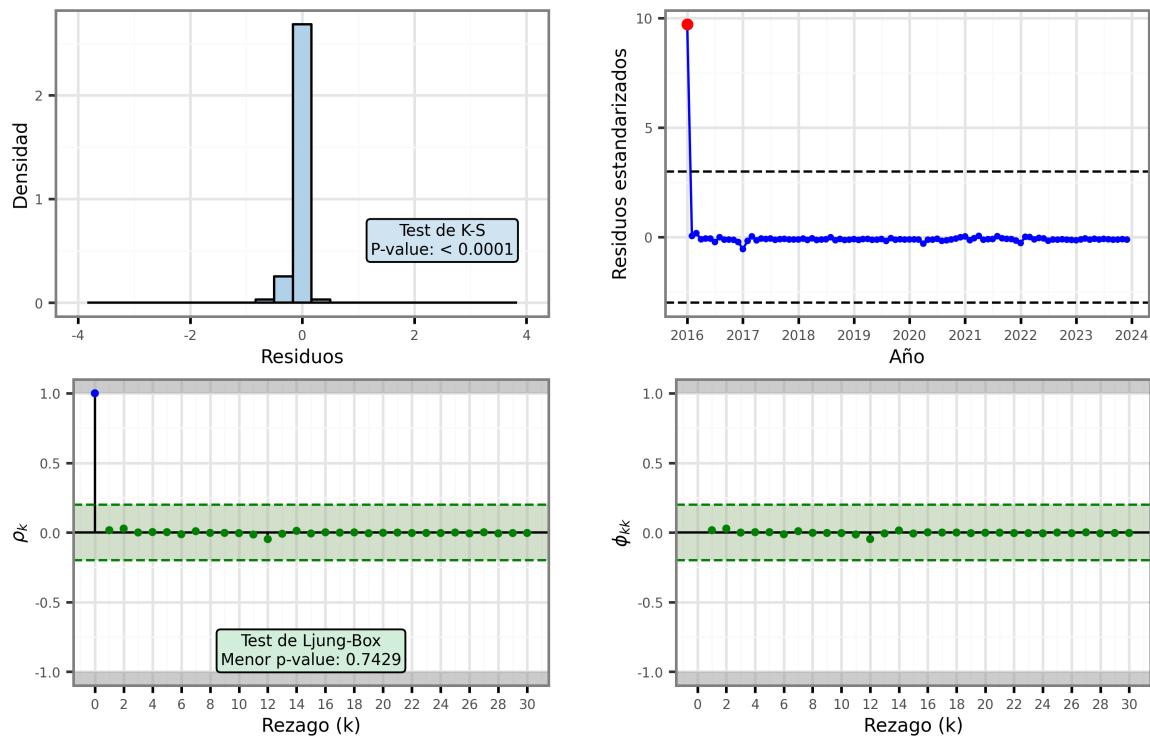


Figura 46: Comprobación de supuestos del modelo ARIMA TR-1 para la serie de trabajadores.

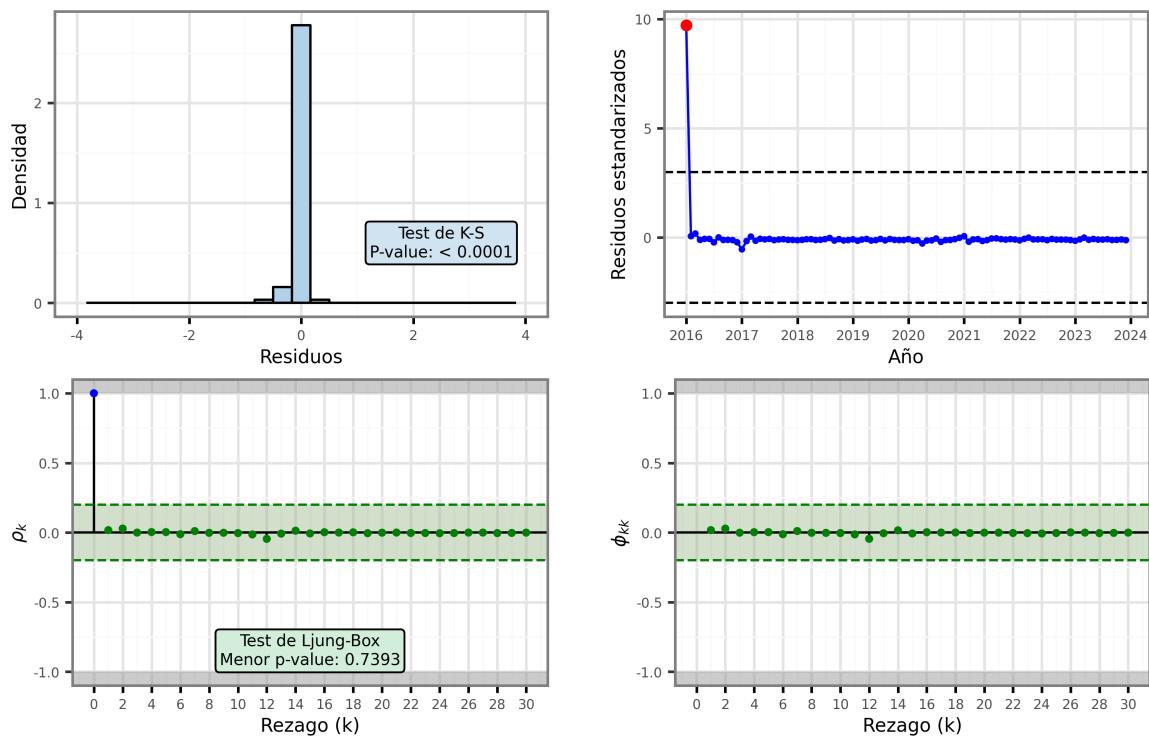


Figura 47: Comprobación de supuestos del modelo ARIMA TR-2 para la serie de trabajadores.

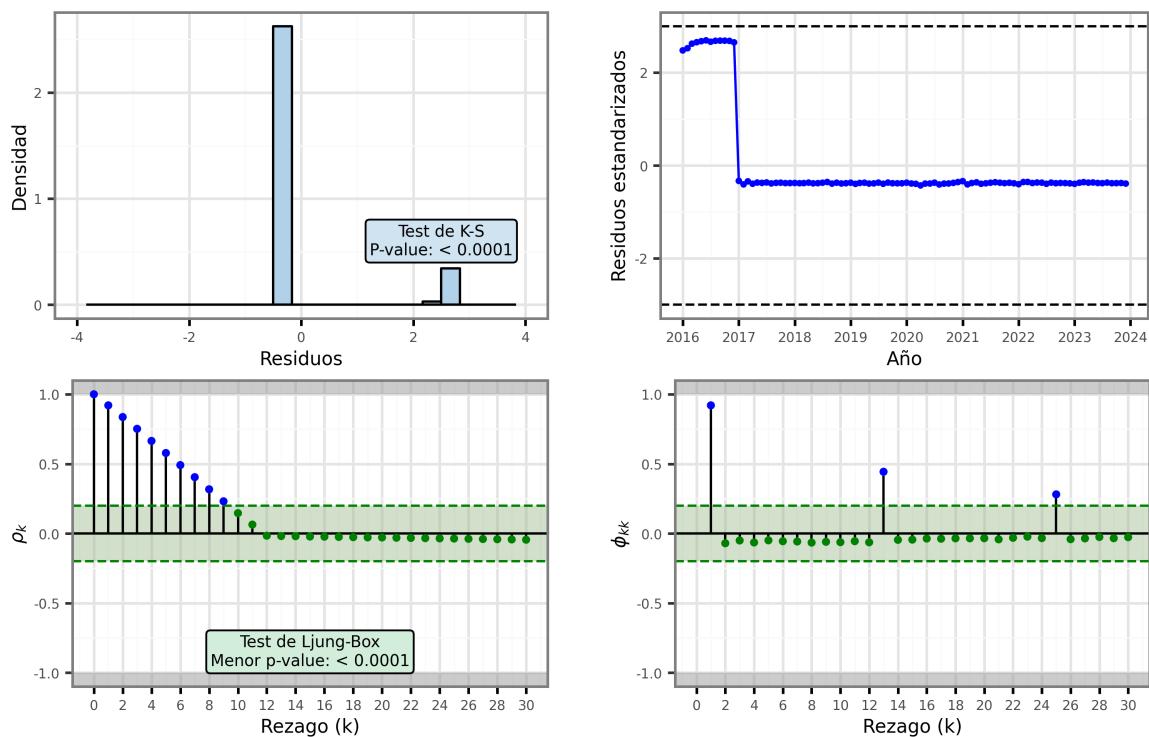


Figura 48: Comprobación de supuestos del modelo ARIMA TR-3 para la serie de trabajadores.

Comprobación de supuestos de modelos ARIMA para la serie de temperaturas sobre los residuos estandarizados

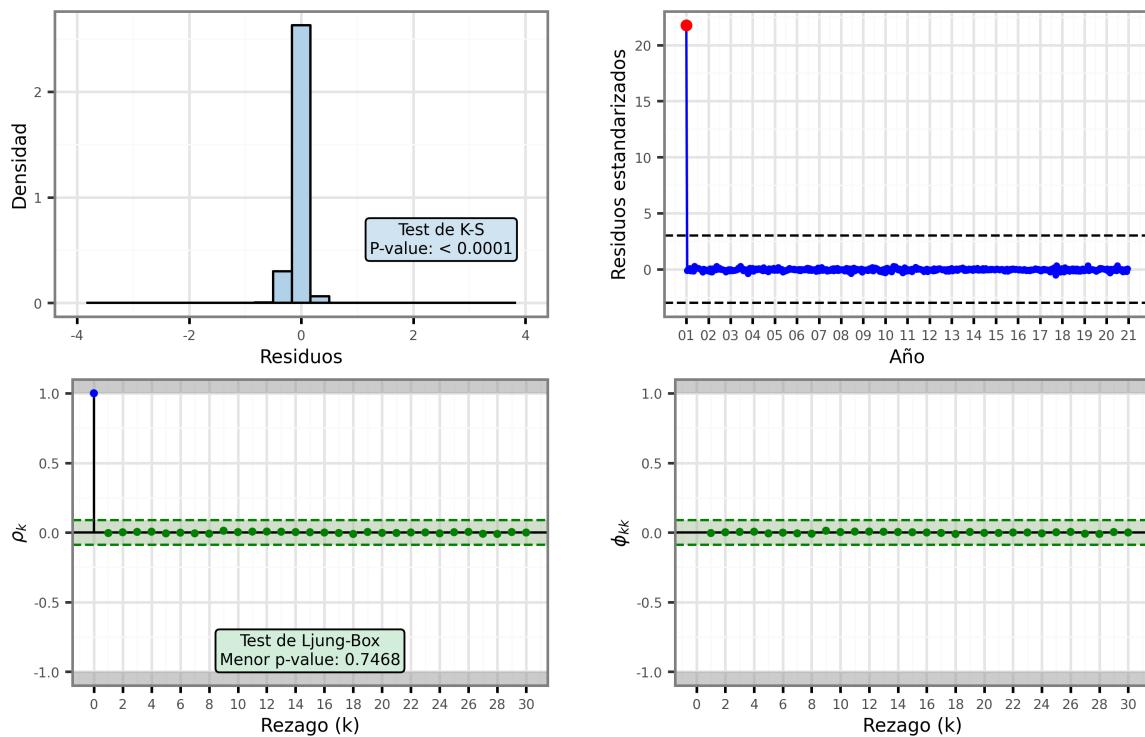


Figura 49: Comprobación de supuestos del modelo ARIMA TE-1 para la serie de temperaturas.

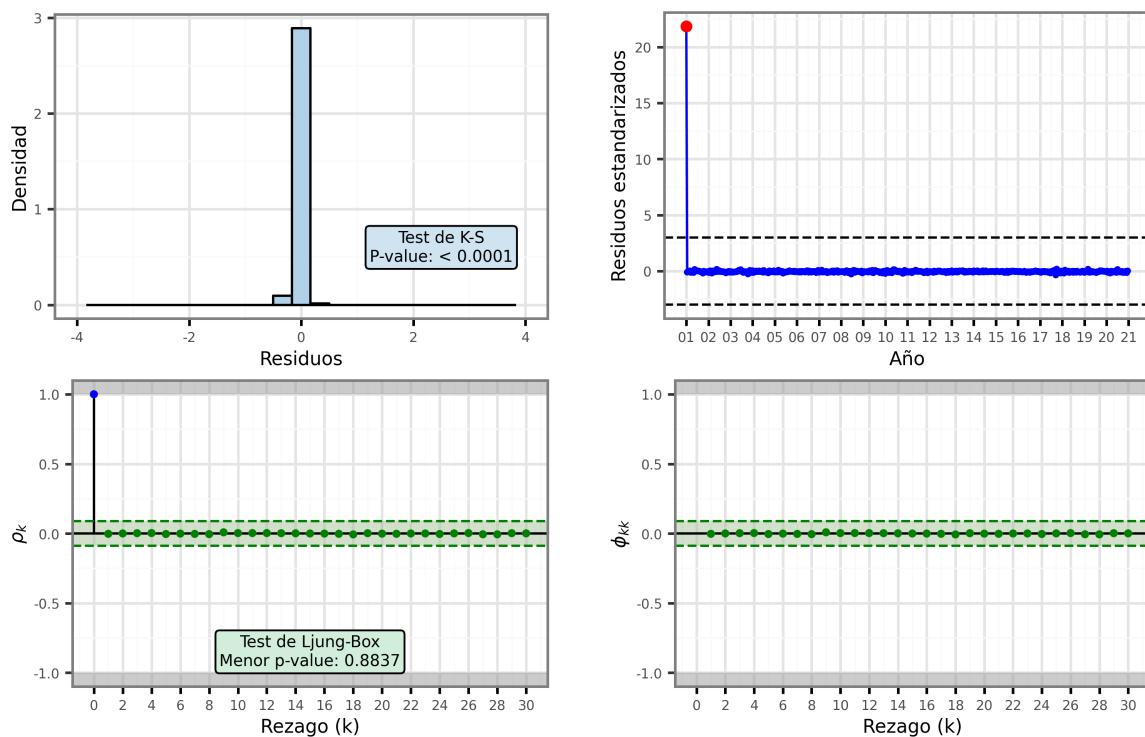


Figura 50: Comprobación de supuestos del modelo ARIMA TE-2 para la serie de temperaturas.

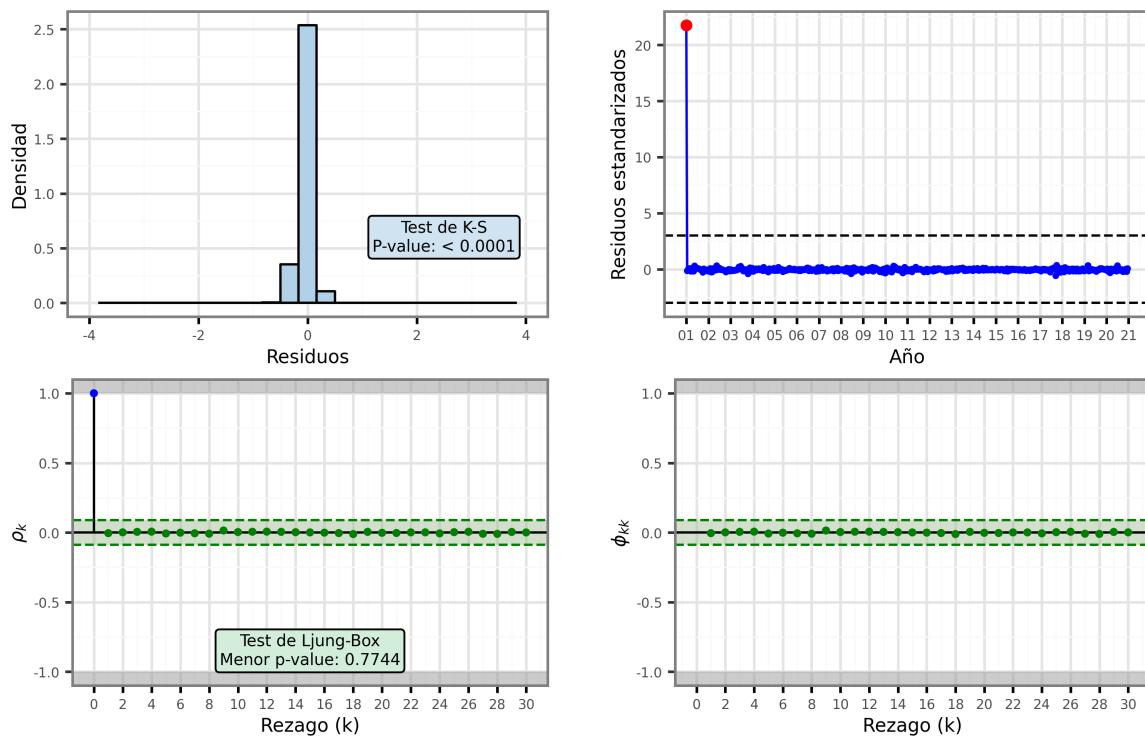


Figura 51: Comprobación de supuestos del modelo ARIMA TE-3 para la serie de temperaturas.

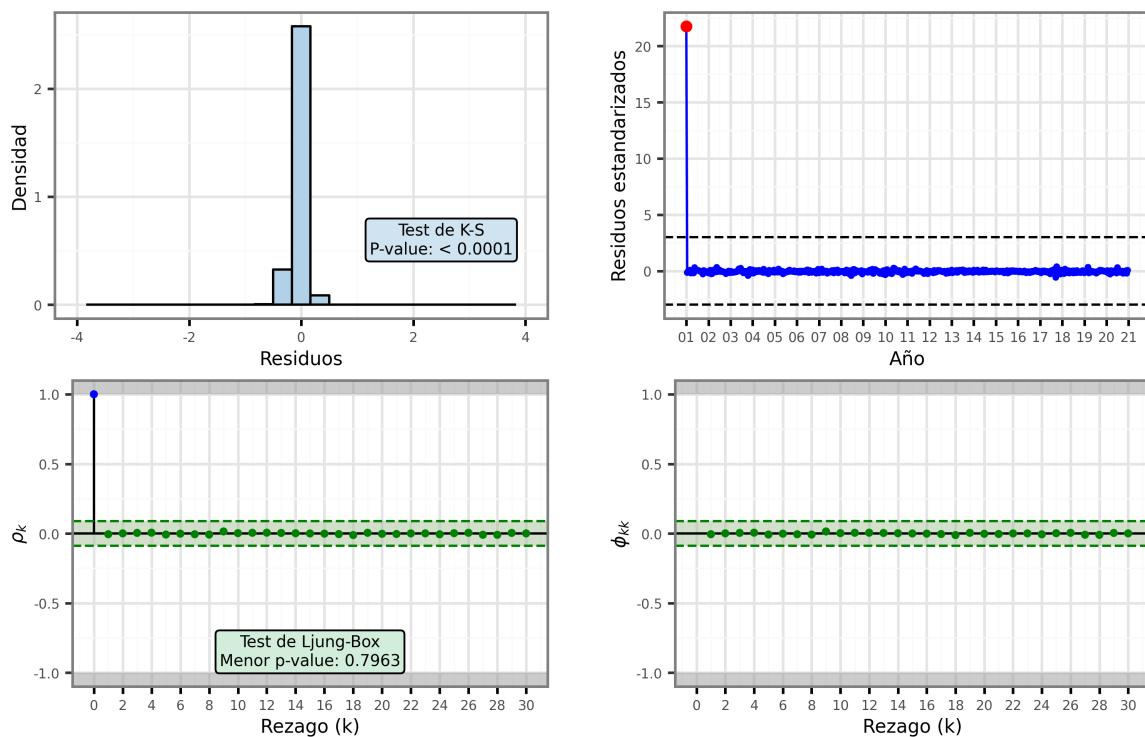


Figura 52: Comprobación de supuestos del modelo ARIMA TE-4 para la serie de temperaturas.