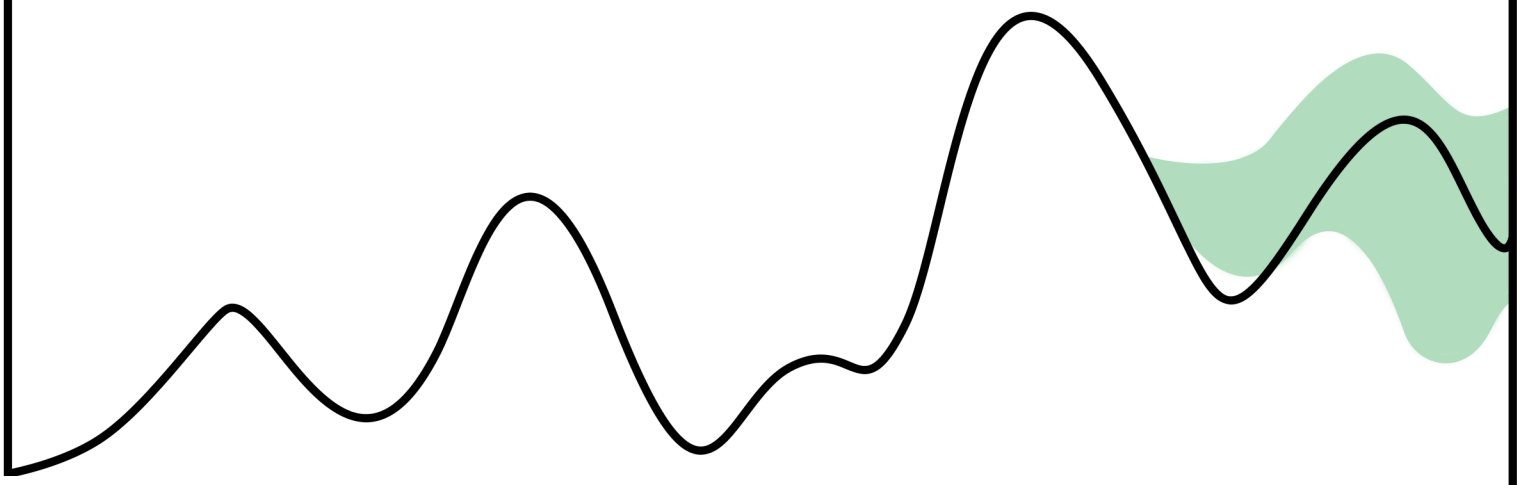
Comparación entre modelos tradicionales, de machine learning y deep learning para el pronóstico de series de tiempo

Anteproyecto de tesina



Alumno: Roncaglia Andrés

Directora: Mag. Méndez Fernanda

Carrera: Licenciatura en Estadística

UNIVERSIDAD NACIONAL DE ROSARIO
CONTINGERE HOMINEM COGITANTEM

UNR Universidad Nacional de Rosario

Tabla de contenidos

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 ambiente. En estos contextos, anticipar el comportamiento de una variable permite mejorar la planificación, asignar recursos de forma más eficiente y reducir la incertidumbre.

En la actualidad, la ciencia de datos atraviesa una etapa constante de expansión e innovación gracias a la masiva cantidad de datos que se generan y trabajan cada día, por lo que en un mundo en donde todo se vuelve más complejo y el tiempo es cada vez más valioso, es conveniente tener herramientas que faciliten y acorten los tiempos de trabajo. Si bien los métodos más conocidos para trabajar series de tiempo son precisos, los modelos tradicionales como ARIMA son difíciles de automatizar y requieren de amplios conocimientos para encontrar un buen ajuste, mientras que los algoritmos de aprendizaje automatizado que se utilizan actualmente pueden tomar un largo tiempo de entrenamiento y un gran coste computacional. Para resolver estos problemas llegan los modelos fundacionales pre-entrenados tales como *TimeGPT* o *Chronos*, los cuales seleccionan de forma automática el modelo más adecuado para la serie especificada, eliminando la necesidad de intervención manual o conocimientos especializados en el tratamiento de datos temporales.

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. Es por esto que en esta tesina se 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 tales como TimeGPT y Chronos, buscando entender como funcionan.
- Definir y aplicar métricas de evaluación (MAPE, *Interval Score*) para comparar el rendimiento de todos los modelos bajo un mismo conjunto de datos.
- Reflexionar críticamente 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 se basa en la comparación del desempeño de distintos modelos de pronóstico aplicados a series temporales. Para ello, se seleccionarán modelos representativos de tres enfoques principales: estadísticos tradicionales, algoritmos de aprendizaje automático (machine learning) y modelos de aprendizaje profundo. El análisis se estructura en tres componentes fundamentales: la caracterización de los modelos, la implementación práctica sobre series con distintas características y la evaluación comparativa mediante métricas cuantitativas.

3.1 Modelos tradicionales

Son llamados modelos tradicionales a aquellos que surgen antes del 'boom' 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.

3.1.1 ARIMA y SARIMA

Los modelos $ARIMA$ (*AutoRegresive Integrated Moving Average*) son unos de los modelos de pronostico tradicionales mejor establecidos. Son una generalización de los modelos autoregresivos (AR), que suponen que las observaciones futuras son combinaciones lineales de las p observaciones pasadas, y los modelos promedio móvil (MA), que pronostican las observaciones como funciones de los errores de las q observaciones pasadas. Además, generaliza en el sentido de los modelos diferenciados (I), en los que se resta a cada observacion los d-ésimo valores anteriores para estacionarizar en media, eliminando así las tendencias determinísticas.

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.

3.2 Modelos de *Machine Learning*

El aprendizaje automático (del inglés *machine learning*) se define como una rama de la inteligencia artificial enfocada a permitir que las computadoras y máquinas imiten la forma en que los humanos aprenden, para realizar tareas de forma autónoma y mejorar la eficiencia y eficacia a través de la experiencia y la exposición a mas información. Si bien los métodos que se presentan no fueron diseñados especificamente para el análisis de datos temporales, como los modelos tradicionales o aquellos que utilizan aprendizaje profundo que se mencionarán más adelante, si probaron ser útiles a lo largo del tiempo y a través de distintas pruebas.

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.

Es importante remarcar que lo que se presenta a continuación como modelos, no son más que técnicas de *boosting* aplicadas a modelos de bosques aleatorios. El concepto de *boosting* es crear modelos de forma secuencial con el objetivo de que los últimos modelos corrijan los errores de los previos.

3.2.1 XGBoost

XGBoost construye 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.

Sin embargo, los modelos no se construyen infinitamente, sino que XGBoost busca minimizar una función de pérdida que incluye una penalización por la complejidad del modelo, limitando así la cantidad de árboles que se producen.

3.2.2 LightGBM

LightGBM funciona de una forma similar a XGBoost, las diferencias radican en la forma en que cada uno identifica las mejores divisiones dentro de los árboles y de que forma los hacen crecer. Mientras que XGBoost usa un método en el que se construyen histogramas para cada una de las características generadas para elegir la mejor división por característica, 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, ya que esto significaría que esps puntos están mejor pronosticados que el resto y no hace falta enfocarse tanto en ellos. Además, LightGBM utiliza un procedimiento que acelera el ajuste cuando se tienen muchas características correlacionadas de las cuáles elegir.

A la hora de hacer crecer los árboles, XGBoost los 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. LigthGBM, en cambio, se expande a partir de la hoja que más reduce el error, mejorando la precisión y eficiencia en series largas, pero arriesgandose a posibles sobreajustes si no se limita correctamente la profundidad de los árboles.

3.3 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 el conjunto de algoritmos que se menciona están las redes neuronales, las cúales son un tipo de modelo que toma decisiones de la misma forma que las personas, usando procesos que simulan la forma biologica en la que trabajan las neuronas para indentificar fenómenos, evaluar opciones y llegar a conclusiones. Una red neuronal funciona con varias neuronas de entrada y salida, y distintos pesos. 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.

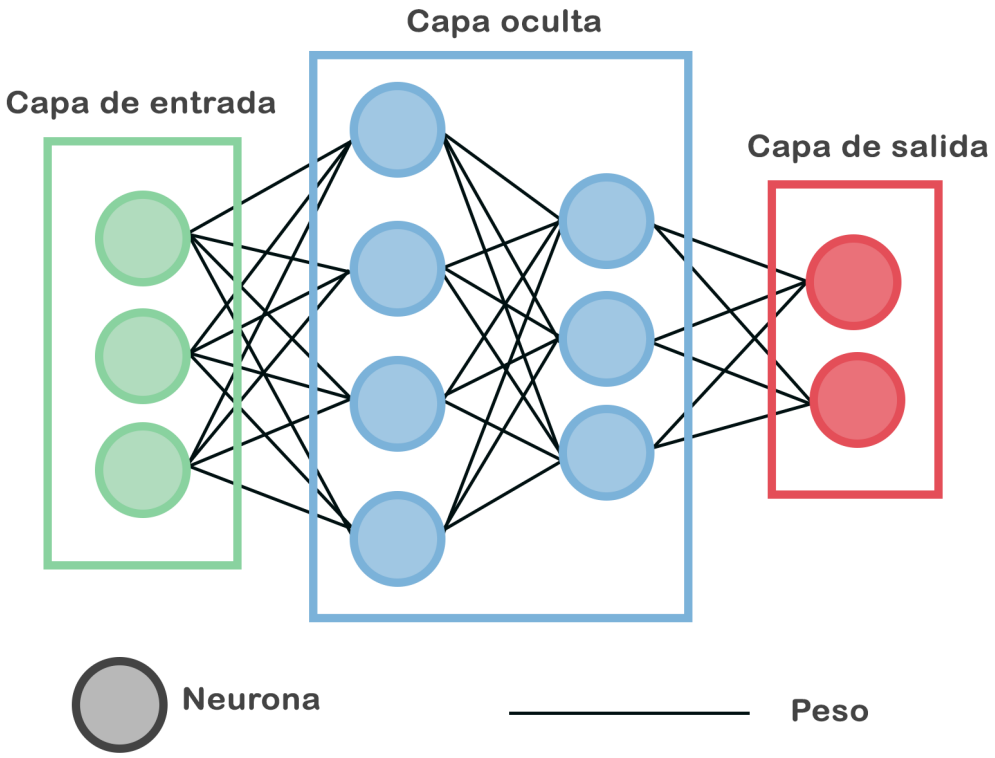


Figura 1: Red neuronal completamente conectada

Hay distintos tipos de redes neuronales según la forma en la que se conectan las neuronas. En esta tesina son de interés especialmente las *Convolutional Neural Networks* (CNN) y las *Recurrent Neural Networks* (RNN), redes neuronales convolucionales y recurrentes respectivamente. Las primeras son útiles para el reconocimiento de patrones en los datos, mientras que las últimas son especialmente buenas en la predicción de datos secuenciales.

Otro tipo de modelo de aprendizaje profundo son los *transformer models* (modelos transformadores), los cúales son significativamente más eficientes al entrenar y realizar inferencias que las CNNs y las RNNs gracias al uso de mecanismos de atención, presentados en la publicación '*Attention is all you need*' de Google.

3.3.1 *Long Short Term Memory (LSTM)*

Lo que caracterizan a las redes neuronales recurrentes son los bucles de retroalimentación que se presentan en la Figura 2. Mientras que cada neurona de entrada en una red neuronal completamente conectada es independiente, en las redes neuronales recurrentes se relacionan entre ellas y se retroalimentan.

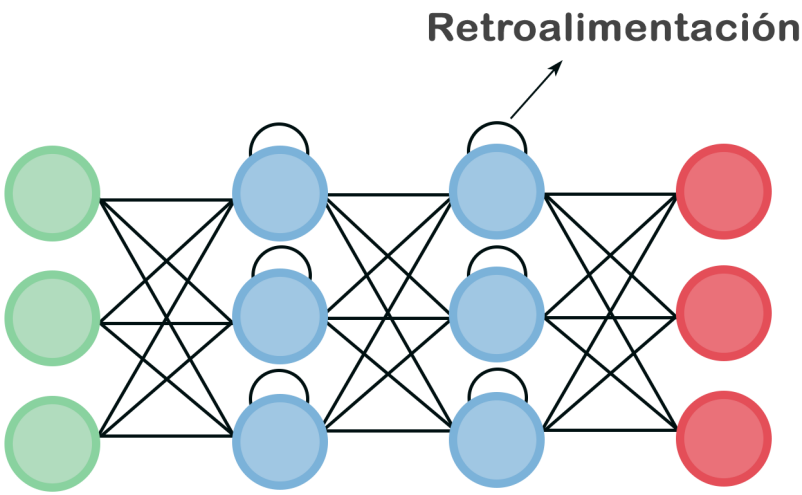


Figura 2: Ejemplo de RNN

Las redes neuronales con memoria a corto y largo plazo (LSTM) son un tipo de RNN que, para evitar que los gradientes se vayan a cero o al infinito muy rápidamente al actualizar los parámetros, usa un algoritmo logístico de 3 puertas:

- Puerta de guardado: Guarda la información relevante del estado actual de la red neuronal.
- Puerta de entrada: Guarda la información que debe ser actualizada en la red neuronal.
- Puerta de salida: Combina la información de las puertas de guardado y entrada, y decide que tanto de esa información usar para seguir utilizando en la red neuronal.

3.3.2 TimeGPT

TimeGPT es un modelo fundacional pre-entrenado para el pronóstico de series de tiempo que puede producir predicciones en diversas áreas y aplicaciones con gran precisión y sin entrenamiento adicional. Los modelos pre-entrenados constituyen una gran innovación haciendo que el pronóstico de series de tiempo sea más accesible, preciso, tenga menor complejidad computacional y consuma menos tiempo.

TimeGPT es un modelo de tipo transformer con mecanismos de autoatención que no es de código abierto. La autoatención captura dependencias y relaciónes en la secuencias de valores que se alimentan al modelo, logrando poner en contexto a cada observación.

La arquitectura del modelo consiste en una estructura con codificador y decodificador de múltiples capas, cada una con conexiones residuales y normalización de capas. Por último, contiene una capa lineal que mapea la salida del decodificador a la dimensión del pronóstico.

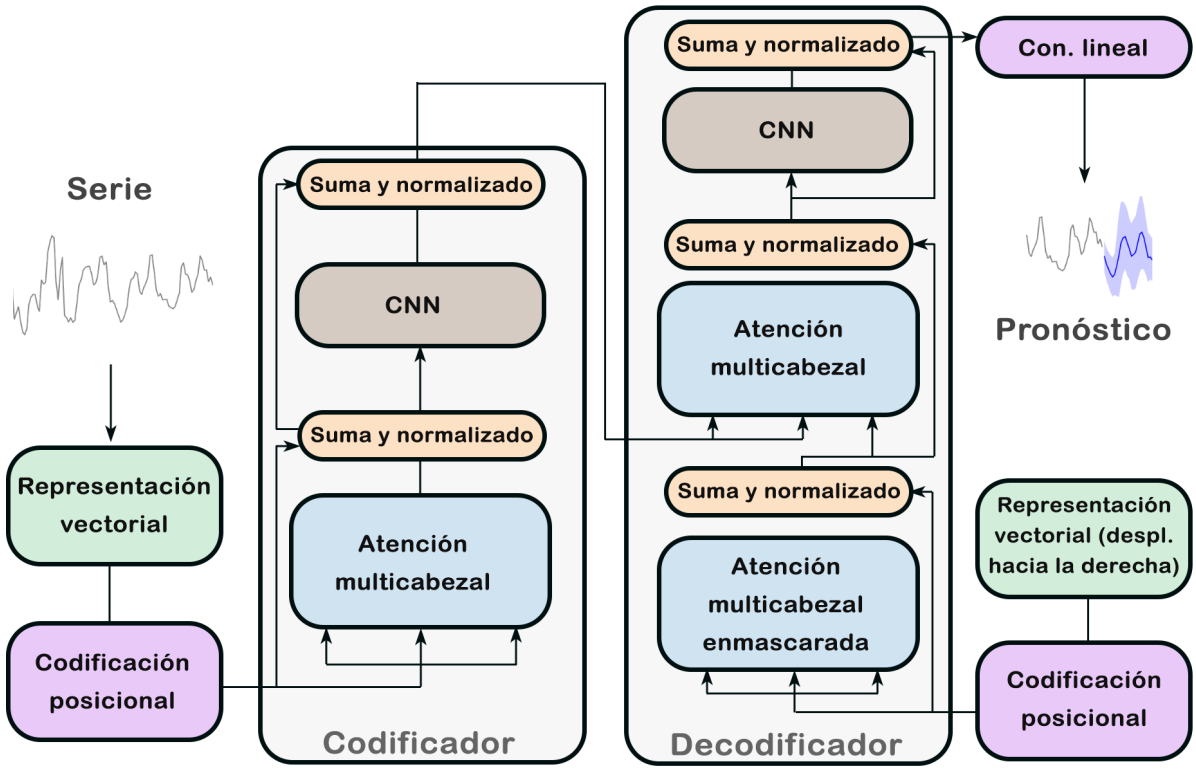


Figura 3: Diagrama de la estructura del modelo

3.3.3 Chronos

Chronos es una familia de modelos fundacionales pre-entrenados para series de tiempo basados en arquitecturas de modelos de lenguaje. Chronos trabaja transformando la serie de tiempo en una secuencia de *tokens* por medio de escalado y cuantificación, para luego entrenar un modelo de lenguaje con estos, usando entropía cruzada. Una vez entrenado el modelo, se generan múltiples pronósticos probabilísticos a partir del contexto historico.

3.4 Métricas de evaluación

Para comparar el rendimiento de los modelos, se utilizarán 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. Las comparaciones permitirán evaluar precisión, robustez y eficiencia de cada enfoque.

4. Aplicación

La aplicación empírica del trabajo se desarrollará en varias etapas, con el objetivo de implementar, ajustar y comparar los modelos previamente detallados sobre un conjunto de series temporales seleccionadas, usando el software de programación de código abierto Python.

4.1 Etapa 1: Selección y preparación de los datos

Se trabajará con series temporales reales obtenidas de bases de datos públicas y confiables. Las series seleccionadas presentarán diferentes características, incluyendo, por ejemplo, estacionalidad, tendencia o diferentes periodicidades, con el fin de evaluar el comportamiento de los modelos bajo condiciones variadas.

Una vez seleccionadas, las series serán sometidas a un proceso de limpieza y transformación, que incluirá la conversión de fechas a formatos estándar, el tratamiento de valores faltantes (si los hubiera) y la división de los datos en subconjuntos de entrenamiento y validación. Se explorará también la opción de aplicar una normalización o estandarización de las series para facilitar el entrenamiento de los modelos de aprendizaje profundo.

4.2 Etapa 2: Implementación de los modelos

En esta etapa se implementarán los distintos modelos de pronóstico contemplados en el trabajo, agrupados en función de su enfoque metodológico: estadístico tradicional, aprendizaje automático y modelos de aprendizaje profundo.

Los modelos estadísticos clásicos, como ARIMA y SARIMA, serán ajustados utilizando la librería `pmdarima`, basando la selección de parámetros en el criterio de información de Akaike (AIC) y usando validación cruzada temporal.

En el caso de los modelos de aprendizaje automático, se emplearán bosques aleatorios con *boosting* haciendo uso de los algoritmos XGBoost y LightGBM. Estos modelos serán implementados con las librerías `xgboost` y `lightgbm`.

Para los modelos de aprendizaje profundo, se desarrollarán redes neuronales del tipo LSTM, con diferentes configuraciones de capas y funciones de activación, usando en este caso la librería `neuralforecast`.

Por último, se explorarán dos modelos fundacionales pre-entrenados. El primero es *TimeGPT*, que será accedido a través de la API provista por Nixtla y haciendo uso de la librería `nixtla`. El segundo es *Chronos*, una familia de modelos pre-entrenados desarrollada por *Amazon Web Services* cuya implementación será llevada a cabo con la librería `autogluon`.

4.3 Etapa 3: Evaluación y comparación

Cada modelo será evaluado en función de su desempeño predictivo, utilizando métricas como MAPE e *Interval Score*, junto con observaciones sobre el tiempo de cómputo, la facilidad de implementación e interpretabilidad de cada uno. Los resultados se sistematizarán en tablas comparativas y visualizaciones gráficas, acompañadas de un análisis crítico.

5. Cronograma de actividades

Mes	Actividad
Mayo	Revisión bibliográfica, definición del enfoque metodológico y selección preliminar de series temporales.
Junio	Implementación de modelos clásicos (ARIMA/SARIMA) y redacción de sus fundamentos teóricos.
Julio	Implementación de modelos de machine learning (XGBoost y LightGBM), desarrollo de su funcionamiento, generación de características y ajuste de hiperparámetros.
Agosto	Implementación de modelos LSTM, investigación de sus parámetros y funcionamiento
Septiembre	Implementación de modelos fundacionales (TimeGPT y Chronos), explicación de la forma en la que trabajan
Octubre	Comparación de resultados entre todos los modelos. Análisis final de métricas y elaboración de gráficos y tablas.
Noviembre	Redacción final del trabajo, revisión integral, correcciones, y preparación para la entrega.

6. Bibliografía

Ansari et al. (4 de noviembre de 2024). Chronos: Learning the Language of Time Series. Transactions on Machine Learning Research. https://arxiv.org/abs/2403.07815

Awan, A. A. (2 de septiembre de 2024). Time Series Forecasting With TimeGPT. Datacamp. https://www.datacamp.com/tutorial/time-series-forecasting-with-time-gpt

Elhariri, K. (1 de marzo de 2022). The Transformer Model. Medium. https://medium.com/data-science/attention-is-all-you-need-e498378552f9

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.

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: principles and practice (3rd ed.). OTexts. https://otexts.com/fpp3/

IBM. (s.f.). Explainers. Recuperado el 14 de marzo de 2025 en 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

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.-a). About TimeGPT. Recuperado en diciembre de 2024 de https://docs.nixtla.io/docs/getting-started-about_timegpt

Nixtla. (s.f.-b). LSTM. Recuperado el 9 de abril de 2025 en https://nixtlaverse.nixtla.io/neuralforecast/models.lstm.html#lstm

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-llms-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

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