Universidad Complutense de Madrid

Facultad de Matemáticas

Departamento de Astronomía y Geodesia

Predicción de manchas solares mediante algoritmos de aprendizaje automático

Trabajo de Fin de Grado

Grado en Matemáticas

Leticia Andradas Jorge

Dirigido por: Gonzalo Barderas Manchado

Madrid, 2024

Predicción de manchas solares mediante algoritmos de aprendizaje automático

Trabajo de Fin de Grado
Grado en Matemáticas
Facultad de Ciencias Matemáticas

Leticia Andradas Jorge
Dirigido por: Gonzalo Barderas Manchado

Departamento de Astronomía y Geodesia
Facultad de Matemáticas
Universidad Complutense de Madrid

Abstract

In recent decades, the study and prediction of the solar cycle have gained increasing relevance due to their significant implications for space climatology and modern technology. The solar cycle, lasting approximately 11 years, is characterized by variations in solar activity that impact space weather and can potentially disrupt communications and satellite navigation systems. Traditionally, solar activity prediction has relied on statistical and physical methods that, while useful, often face limitations in terms of precision and adaptability.

This study aims to address these challenges by predicting solar cycle Sunspot Numbers using the XGBoost machine learning model. The data utilized include historical sunspot records and smoothed time series to enhance analytical precision. Statistical analysis revealed a mean Sunspot Number of 81.91 with a standard deviation of 67.67, indicating high variability. The distribution of Sunspot Numbers shows a positive skew and low kurtosis, suggesting a relatively flat distribution with some asymmetry towards higher values.

The XGBoost model demonstrated a notable improvement in performance, achieving a Mean Absolute Error (MAE) of 33.14, which represents a 42.54 % reduction compared to the baseline MAE of 57.68. Its Mean Squared Error (MSE) of 1883.80 is lower than the Moving Average model but significantly higher than the Exponential Smoothing model. The Root Mean Squared Error (RMSE) of 43.40 for XGBoost is competitive but still below the performance of models like GRU, Transformer, and Informer.

Despite these advancements, the study acknowledges that the MAE of XGBoost is significantly higher than that of more advanced models, which may impact its usefulness for applications requiring low absolute prediction errors. Additionally, the complexity of XGBoost raises concerns about overfitting and interpretability. Future work should focus on optimizing hyperparameters, exploring

additional preprocessing techniques, and integrating other data sources to further enhance the accuracy and robustness of the model.

In summary, while the XGBoost model demonstrates a significant improvement over traditional forecasting methods in terms of RMSE, its high MAE and complexity highlight areas for further refinement.

Resumen

En las últimas décadas, el estudio y la predicción del ciclo solar han adquirido una relevancia creciente debido a sus implicaciones significativas para la climatología espacial y la tecnología moderna. El ciclo solar, que dura aproximadamente 11 años, se caracteriza por variaciones en la actividad solar que impactan el clima espacial y pueden interrumpir las comunicaciones y los sistemas de navegación por satélite. Tradicionalmente, la predicción de la actividad solar se ha basado en métodos estadísticos y físicos que, aunque útiles, a menudo enfrentan limitaciones en términos de precisión y adaptabilidad.

Este estudio tiene como objetivo abordar estos desafíos mediante la predicción del número de manchas solares del ciclo solar utilizando el modelo de aprendizaje automático XGBoost. Los datos utilizados incluyen registros históricos de manchas solares y series temporales suavizadas para mejorar la precisión analítica. El análisis estadístico reveló una media del número de manchas solares de 81.91 con una desviación estándar de 67.67, lo que indica una alta variabilidad. La distribución de los números de manchas solares muestra un sesgo positivo y baja curtosis, sugiriendo una distribución relativamente plana con algo de asimetría hacia valores más altos.

El modelo XGBoost demostró una mejora notable en el rendimiento, alcanzando un Error Absoluto Medio (MAE) de 33.14, lo que representa una reducción del 42.54 % en comparación con el MAE base de 57.68. Su Error Cuadrático Medio (MSE) de 1883.80 es menor que el del modelo de Media Móvil pero significativamente mayor que el del modelo de Suavizado Exponencial. El Error Cuadrático Medio (RMSE) de 43.40 para XGBoost es competitivo pero aún inferior al rendimiento de modelos como GRU, *Transformer* e *Informer*.

A pesar de estos avances, el estudio reconoce que el MAE de XGBoost es sustancialmente más alto que el de modelos más avanzados, lo que puede afectar

a aplicaciones que requieran errores de predicción absolutos bajos. Además, la complejidad de XGBoost plantea preocupaciones sobre el sobreajuste y la interpretabilidad. Los trabajos futuros deben centrarse en la optimización de hiperparámetros, explorar técnicas adicionales de preprocesamiento e integrar otras fuentes de datos para mejorar aún más la precisión y robustez del modelo.

En resumen, aunque el modelo XGBoost demuestra una mejora significativa respecto a los métodos tradicionales de pronóstico en términos de RMSE, su alto MAE y complejidad destacan áreas que requieren una mayor refinación.

Índice general

Capítulo 1

Introducción y Objetivos

En las últimas décadas, el estudio y la predicción del ciclo solar ha adquirido una relevancia creciente debido a sus implicaciones significativas en la climatología espacial y en la tecnología moderna. El ciclo solar, que dura aproximadamente 11 años, se caracteriza por variaciones en la actividad solar que afectan al clima espacial y tienen impactos potencialmente disruptivos en las comunicaciones y en los sistemas de navegación por satélite. Tradicionalmente, la predicción de la actividad solar se ha basado en métodos estadísticos y físicos que, aunque útiles, a menudo enfrentan limitaciones en cuanto a precisión y adaptabilidad. En este contexto, el presente trabajo de fin de grado explora el uso del algoritmo XGBoost (*Extreme Gradient Boosting*), una técnica avanzada de aprendizaje automático que ha demostrado ser eficaz en una variedad de tareas de predicción debido a su capacidad para manejar grandes volúmenes de datos y para modelar relaciones complejas entre variables.

El objetivo principal de este trabajo es desarrollar un modelo predictivo del ciclo solar utilizando XGBoost, con la finalidad de mejorar la precisión de las predicciones respecto a los métodos tradicionales. Para alcanzar este objetivo, el trabajo se divide en varias etapas. Primero, se realiza una revisión exhaustiva de la literatura existente sobre la predicción del ciclo solar y de los algoritmos de *machine learning*, profundizando en el algoritmo XGBoost, con el fin de establecer un marco teórico sólido. A continuación, se lleva a cabo la recopilación y preprocesamiento de datos históricos relacionados con la actividad solar. El siguiente paso consiste en la implementación y ajuste del modelo XGBoost, evaluando su rendimiento mediante técnicas de validación cruzada y comparación con otros métodos predictivos. Finalmente, se presentan los resultados obtenidos.

Este enfoque permitirá no sólo evaluar la capacidad de XGBoost para predecir el ciclo solar, sino también contribuir al avance en la ciencia de la predicción solar mediante la integración de técnicas modernas de aprendizaje automático en un campo tradicionalmente dominado por métodos estadísticos.

1.1. Ciclo solar y *machine learning*

1.1.1. Ciclo solar

Las manchas solares son áreas más oscuras en la superficie del Sol, donde intensos campos magnéticos emergen desde su interior profundo. Estos fenómenos han sido observados durante más de 2000 años, con registros iniciales provenientes de China [1]. No obstante, se descubrió posteriormente la posibilidad de un comportamiento periódico en las manchas solares gracias al *número de Wolf*, definido en 1848. Con ello se pudo detectar rápidamente que el Sol tenía un ciclo de actividad de unos 11 años [1].

Así, la actividad solar sigue un ciclo de aproximadamente 11 años, afectando la vida moderna de diversas maneras. Este aumento de actividad conlleva un incremento en las emisiones de radiación ultravioleta extrema y rayos X del Sol, generando efectos notables sobre el contenido de los electrones en las capas altas de la ionosfera, lo que afecta a la señal de los GPS [2] .

Asimismo, el aumento en el número de fulguraciones solares y eyecciones de masa coronal aumenta la probabilidad de daños en instrumentos sensibles en el espacio, así como representa un riesgo para la salud de los astronautas [3].

Además, existe evidencia sólida que sugiere que la actividad solar también impacta en el clima terrestre, aunque el cambio en la irradiancia solar total parece ser demasiado pequeño para producir efectos climáticos significativos [4]. Sin embargo, se ha observado que el clima de la Tierra experimenta fluctuaciones correlacionadas con los ciclos de actividad solar [5].

El ciclo solar, que es de naturaleza magnética, es producido por procesos dinámicos dentro del Sol. Aunque aún hay incertidumbre en cuanto a los detalles de cómo, cuándo y dónde operan estos procesos, sabemos que los campos magnéticos y el plasma ionizado se mueven conjuntamente en el interior del Sol. En la mayoría de los modelos de dinamo, están involucrados dos procesos básicos: movimientos de cizalla que fortalecen y alinean el campo magnético con el flujo

magnético (efecto Omega), y movimientos helicoidales que elevan y retuercen el campo magnético en un plano diferente (efecto Alfa) [8].

Dada su influencia en el clima y otros aspectos de la vida terrestre, en este trabajo se va a intentar predecir el ciclo solar mediante algoritmos de aprendizaje automático.

1.1.2. Aprendizaje automático

El aprendizaje automático se define como el campo de estudio que otorga a los ordenadores la capacidad de aprender sin programación explícita [9], extrayendo patrones de los datos.

La selección de algoritmos depende de varios factores, incluyendo la naturaleza del problema a abordar, el número de variables involucradas y el modelo más adecuado para lo que se quiere predecir [10]. Dentro del *machine learning*, se realiza la siguiente clasificación:

- **Aprendizaje supervisado**

Es una técnica de aprendizaje automático donde se entrena un modelo que permite predecir o clasificar datos basándose en un conjunto de datos de entrenamiento etiquetados con la característica buscada. El modelo utiliza estos ejemplos para aprender la relación entre las características de entrada y las etiquetas de salida. Una vez entrenado, el modelo puede generalizar la relación aprendida para hacer predicciones precisas sobre nuevos datos no vistos [10].

El término "supervisado" deriva del hecho de que durante el entrenamiento, el modelo está siendo "supervisado" por un usuario que etiquete los datos sobre la base del conocimiento de la característica buscada en un conjunto de datos conocido, lo que le permite ajustarse para minimizar la discrepancia entre las predicciones y las etiquetas reales [10].

- **Aprendizaje no supervisado**

A diferencia del anterior, el aprendizaje no supervisado carece de orientación instructiva y, por tanto, de respuestas predefinidas. Los algoritmos exploran e identifican de manera autónoma estructuras inherentes dentro de los datos. Así, estos algoritmos extraen características esenciales de los datos y las utilizan para clasificar instancias de los datos entrantes [10].

- **Aprendizaje semisupervisado**

Se trata de una técnica de aprendizaje automático que combina elementos de metodologías supervisadas y no supervisadas. Se dispone de un conjunto de datos que contiene tanto datos etiquetados como no etiquetados. A diferencia del aprendizaje supervisado, donde todos los datos están etiquetados, y del aprendizaje no supervisado, donde no hay etiquetas disponibles, el aprendizaje semisupervisado aprovecha la presencia de datos no etiquetados para mejorar el rendimiento del modelo. El objetivo es utilizar tanto los datos etiquetados como los no etiquetados para aprender patrones y estructuras subyacentes en los datos y hacer predicciones más precisas [11]. Este enfoque es útil en situaciones donde obtener etiquetas para todos los datos resulta muy costoso, ya que permite aprovechar la información disponible tanto en los datos etiquetados como en los no etiquetados para mejorar la capacidad predictiva del modelo .

- **Aprendizaje por refuerzo**

El aprendizaje por refuerzo se centra en cómo los agentes deben interactuar dentro de un entorno dinámico para maximizar la recompensa acumulativa. Constituye uno de los paradigmas fundamentales del aprendizaje automático, junto con el aprendizaje supervisado y no supervisado. En este enfoque, el agente toma acciones en el entorno y recibe retroalimentación en forma de recompensas o castigos en función de las acciones tomadas [10]. El objetivo del agente es aprender una política de comportamiento que maximice la recompensa acumulativa a lo largo del tiempo. A medida que el agente explora el entorno y recibe retroalimentación, ajusta su política de comportamiento para mejorar su desempeño y maximizar la recompensa esperada. El aprendizaje por refuerzo se aplica en campos como los juegos, la robótica, el control de procesos o la toma de decisiones autónomas [12].

- **Aprendizaje Multi-tarea**

El aprendizaje multi-tarea es un subcampo del aprendizaje automático que tiene como objetivo resolver diferentes tareas al mismo tiempo, aprovechando las similitudes entre ellas. Esto puede mejorar la eficiencia del aprendizaje y también actuar como regularizador. Los enfoques convencionales de aprendizaje profundo tienen como objetivo resolver una única tarea utilizando un modelo en particular. Sin embargo, si hay n tareas, o un subconjunto de ellas relacionadas entre sí pero

no son exactamente idénticas, el aprendizaje multi-tarea (MTL) ayudará a mejorar el aprendizaje de un modelo particular utilizando el conocimiento contenido en las *n* tareas [13].

- **Aprendizaje en Conjunto (*Ensemble learning*)**

El *Ensemble learning* es una técnica de aprendizaje automático que combina múltiples modelos para mejorar el rendimiento predictivo y la generalización. En lugar de depender de un sólo modelo, el aprendizaje conjunto utiliza la fortaleza colectiva de varios modelos para tomar decisiones más precisas y robustas [14]. Esto se logra mediante la combinación de las predicciones de los diferentes modelos individuales, ya sea mediante votación, promedio o métodos más sofisticados como el *boosting* o el *bagging*. El objetivo principal del aprendizaje conjunto es reducir el sesgo y la varianza, mejorando así la capacidad de generalización del sistema [15].

Dentro de este tipo de aprendizaje, destacan los siguientes tipos de algoritmos [16]:

- Algoritmos de *bagging*

El *bagging* es una técnica que combina múltiples aprendices entrenados en subconjuntos diferentes de los datos originales [17]. Se generan múltiples conjuntos de datos y se desarrollan modelos basados en ellos, cuyas predicciones se combinan para producir un valor representativo, como la media, la mediana o el voto mayoritario para la clasificación y el promedio para la regresión, dependiendo del problema a resolver.

Dado que un aprendiz individual a menudo es sensible al ruido en los datos de entrenamiento, el *bagging*, al agregar múltiples resultados en una sola predicción, debería proporcionar resultados estables y mejorados con una varianza disminuida [17].

Uno de los algoritmos más conocidos de *bagging* es el de *Random Forest* (RF), el cuál utiliza árboles de clasificación y regresión como aprendices individuales [16]. RF excluye aproximadamente el 30 % de las muestras de entrenamiento debido al remuestreo y se utiliza para calcular el error de predicción fuera de la bolsa (OOB) . Aunque RF tiene muchas ventajas, como su capacidad para resolver problemas de clasificación y regresión y su insensibilidad al ruido en los datos de entrenamiento, ignora la correlación espacial de los datos observables cercanos [18].

- Algoritmos de *boosting*

Los algoritmos de *boosting* construyen una secuencia de modelos de forma iterativa, donde cada modelo se enfoca en corregir los errores del modelo anterior. Dentro de los algoritmos de *boosting*, hay varias implementaciones populares como AdaBoost, *Gradient Boosting Machines* (GBM), XGBoost, *Light Gradient Boosting Machine* o CatBoost, cada uno con sus propias características y ajustes específicos. Estos algoritmos son particularmente efectivos para mejorar la precisión de los modelos, especialmente en problemas donde se requiere alta precisión predictiva [16].

- AdaBoost (*Adaptive Boosting*)

AdaBoost es un método de *ensemble learning* que construye un clasificador de manera iterativa. En cada iteración, llama a un algoritmo de aprendizaje simple (llamado el aprendiz base) que devuelve un clasificador, y le asigna un coeficiente de peso. La clasificación final será decidida por una "votación" ponderada de los clasificadores base. Cuanto menor sea el error del clasificador base, mayor será su peso en la votación final. Los clasificadores base solo tienen que ser ligeramente mejores que una suposición aleatoria (de donde deriva su nombre alternativo de clasificador débil), lo que proporciona una gran flexibilidad en el diseño del conjunto de clasificadores base (o características) [16].

- XGBoost (*Extreme Gradient Boosting*)

XGBoost es una técnica de *ensemble learning* basada en árboles de decisión y *Gradient Boosting* que ha ganado reconocimiento por su escalabilidad y eficiencia en el ámbito del aprendizaje automático.

Se destaca que XGBoost adopta una estrategia de expansión aditiva de la función objetivo, minimizando una función de pérdida, lo cual es coherente con los principios fundamentales de *Gradient Boosting*. Es crucial mencionar que esta metodología se especializa en el uso de árboles de decisión como clasificadores base y emplea una variante de la función de pérdida para regular la complejidad de los árboles. Además, se introduce un hiperparámetro de regularización conocido como *shrinkage*, que permite ajustar el tamaño del paso en la expansión aditiva, otorgando flexibilidad al modelo y mejorando su capacidad de generalización [16].

Asimismo, se resalta la aplicación de técnicas de aleatorización, como el submuestreo aleatorio y el submuestreo de columnas, con el fin de mitigar el sobreajuste y acelerar el proceso de entrenamiento del modelo. Un aspecto relevante es el algoritmo propuesto por XGBoost para la selección de la mejor división, el cual considera la dispersión de atributos y elimina automáticamente las entradas con valores cero o faltantes. Esto demuestra la robustez y adaptabilidad de XGBoost frente a datos incompletos o dispersos [16].

Además, se destacan características específicas como las restricciones de monotonicidad e interacción de características, que pueden resultar valiosas en casos donde se cuenta con información previa sobre el dominio del problema. Por último, se mencionan los métodos implementados por XGBoost para mejorar la velocidad de entrenamiento, tales como la optimización de la estructura de almacenamiento y el uso de técnicas basadas en percentiles para la selección de divisiones [19].

- LightGBM (*Light Gradient Boosting Machine*)

LightGBM es un modelo preciso centrado en proporcionar un rendimiento de entrenamiento extremadamente rápido utilizando un muestreo selectivo de instancias con alto gradiente.

Se destaca por su enfoque en la eficiencia computacional, basándose en la precomputación del histograma de características, similar a XGBoost. La biblioteca ofrece numerosos hiperparámetros de aprendizaje que la hacen adaptable a una amplia variedad de escenarios y es compatible con GPU y CPU. Entre sus características adicionales se encuentran el muestreo basado en gradientes unilaterales (GOSS) y la agrupación exclusiva de características (EFB), diseñadas para mejorar la velocidad de entrenamiento y la importancia de las instancias con mayor incertidumbre en las clasificaciones. GOSS y EFB ofrecen beneficios significativos en el proceso de entrenamiento, con EFB considerada como una técnica de preprocesamiento de características. Este estudio se centra en el análisis de LightGBM con GOSS, ya que la implementación estándar del aumento de gradiente está cubierta por *Gradient Boosting* [16].

- CatBoost

CatBoost es una biblioteca de *Gradient Boosting* que tiene como objetivo reducir el cambio en las predicciones que ocurre durante el entrenamiento, con el fin de mejorar el modelo.

Este cambio de distribución se refiere a la diferencia entre:

$$F(x_i \mid x_i) - F(x \mid x) \tag{1.1}$$

donde x_i es una instancia de entrenamiento, respecto a $F(x \mid x)$ para una instancia de prueba x. Aborda este problema estimando los gradientes utilizando para ello una secuencia de modelos base que no incluyen esa instancia en su conjunto de entrenamiento. Para lograr esto, introduce una permutación aleatoria en las instancias de entrenamiento y construye modelos base simétricos a nivel de árbol o tabla de decisión. También maneja características categóricas sustituyéndolas por una característica numérica que mide el valor objetivo esperado para cada categoría.

Además, incluye características como el entrenamiento en GPU y una amplia gama de hiperparámetros para adaptarse a diversas situaciones de aprendizaje [20].

1.2. Objetivos

En este trabajo, se empleará el algoritmo XGBoost para tratar de realizar predicciones sobre el ciclo solar por su capacidad para manejar grandes volúmenes de datos y mejorar la precisión predictiva. Se selecciona este enfoque debido a su eficacia en la reducción de errores y su capacidad para captar relaciones complejas en los datos, lo cual es crucial para la modelización del ciclo solar.

Para ello, la memoria se ha estructurado de la siguiente forma. En el Capítulo 2 se detallarán los fundamentos teóricos del algoritmo XGBoost. En el Capítulo 3 se describe en detalle la fuente de datos utilizada, así como la metodología aplicada para el preprocesamiento y entrenamiento del modelo, concluyendo con una evaluación exhaustiva del rendimiento del modelo, que incluye estadísticas descriptivas y una comparación con otros enfoques, como la Media Móvil, el Promedio Exponencial y otros modelos desarollados para la predicción del número de manchas solares [21]. Finalmente, en el Capítulo 4 se resumen los principales resultados y conclusiones de este trabajo.

Capítulo 2

Desarrollo teórico del algoritmo XGBoost

En este capítulo se profundiza en el desarrollo teórico del algoritmo XGBoost, desde su estructura general hasta las técnicas avanzadas para optimizar su rendimiento y evitar el sobreajuste.

Se exploran varios aspectos críticos: primero, el concepto de modelo de conjunto de árboles, donde se describe cómo el XGBoost utiliza una serie de árboles de decisión para hacer predicciones agregadas. Luego, se aborda la función objetivo regularizada, que se minimiza durante el entrenamiento, combinando un término de pérdida que mide la precisión de las predicciones y un término de regularización que controla la complejidad del modelo para evitar el sobreajuste.

A continuación, se explica el proceso de *Gradient Tree Boosting*, que optimiza el modelo de forma aditiva a lo largo de varias iteraciones, utilizando una aproximación de segundo orden para mejorar la eficiencia del entrenamiento. Se discuten también las técnicas para prevenir el sobreajuste, como la contracción y el submuestreo de columnas, que ayudan a mejorar la generalización del modelo.

Además, el capítulo cubre los algoritmos de búsqueda dividida, incluyendo tanto el enfoque exacto como el aproximado, que permiten la construcción eficiente de los árboles de decisión. También se introduce el algoritmo *Weighted Quantile Sketch*, diseñado para manejar características ponderadas en grandes conjuntos de datos, asegurando una selección de divisiones precisa y eficiente.

Finalmente, se describe la técnica *Sparsity-aware Split Finding*, que permite al XGBoost manejar eficazmente los datos dispersos, asegurando que las instancias con valores faltantes se procesen correctamente sin necesidad de preprocesamien-

to adicional.

Así, se pretende proporcionar una visión completa de los fundamentos y técnicas clave que conforman el algoritmo XGBoost.

2.1. Desarrollo algoritmo XGBoost

Se considera un conjunto de datos:

$$D = \{(y_i, z_i)\}$$

donde:

- $|D| = n$ es el tamaño del conjunto de datos.
- $y_i \in \mathbb{R}^m$ son los valores reales que se quieren predecir.
- $z_i \in \mathbb{R}$ representan las predicciones actuales del modelo en la iteración t.

Como se ha comentado en el capítulo anterior, XGBoost es una técnica de *ensemble learning* basada en árboles de decisión. Debido a que se aplicará este algoritmo para un modelo de regresión, se detallará la construcción de árboles de regresión. Un modelo de conjunto de árboles, usa K funciones aditivas para predecir el *output*, definido como:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{2.1}$$

$$\mathcal{F} = \{f(x) = w_q(x)\} \tag{2.2}$$

$$q : \mathbb{R}^m \to T, \quad w \in \mathbb{R}^T \tag{2.3}$$

Donde:

- $\hat{y}_i$ es la predicción final para el dato de entrada x_i.
- K es el número total de árboles en el modelo.
- f_k es la función de predicción del árbol en la iteración k.
- f_k pertenece a $\mathcal{F}$ que representa el conjunto de árboles de regresión (CART).

- q representa la estructura de cada árbol, que mapea un ejemplo al correspondiente índice de la hoja.
- T representa el número de hojas en el árbol.

Cada f_k corresponde a una estructura q de árbol independiente y la hoja pesa w_q. Cada árbol de regresión contiene una puntuación en la hoja i-ésima. Para un ejemplo dado, usaremos las reglas de decisión en los árboles (dados por q) para clasificarlo dentro de las hojas y calcular la predicción final resumiendo la puntuación en las hojas correspondientes (dados por w_q). El modelo aprende ajustando las funciones f_k para minimizar la siguiente función objetivo regularizada:

$$L = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{2.4}$$

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{2.5}$$

donde:

- γ es el parámetro de regularización que penaliza el número de hojas T o complejidad del modelo.
- T es el número de hojas en el árbol.
- λ es el parámetro de regularización L2.
- w_j es el peso de la hoja j.
- L es la función de pérdida.
- y_i es el valor real.
- $\hat{y}_i$ es la predicción inicial.
- l es la función de pérdida específica (por ejemplo, el Error Cuadrático Medio), que mide la diferencia entre la predicción $\hat{y}_i$ y el dato objetivo y_i .
- $\Omega(f_k)$ es el término de regularización.
- n es el número de ejemplos en el conjunto de datos.

El término de regularización adicional ayuda a suavizar los pesos finales aprendidos para evitar el sobreajuste. De manera intuitiva, la función objetivo tenderá a seleccionar un modelo empleando funciones simples y predictivas. Cuando el parámetro de regularización se pone a cero, el objetivo cae de nuevo al tradicional *Gradient Tree Boosting*.

2.1.1. Gradient Tree Boosting

El modelo de conjunto de árboles en la ecuación (2.4) incluye funciones como parámetros y no pueden ser optimizados usando métodos tradicionales de optimización en el espacio euclídeo. En cambio, el modelo es entrenado de manera específica debido a la naturaleza aditiva de los árboles. En términos simples, en cada iteración t, se calcula la predicción actual $\hat{y}_i^{(t)}$para cada instancia i. Para mejorar la precisión del modelo, es necesario añadir una nueva función f_t que minimice el siguiente objetivo:

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \tag{2.6}$$

Esto significa que añadimos la función f_t que mejora más el modelo, siguiendo la ecuación (2.6). Para optimizar de manera rápida, utilizamos la aproximación de Taylor segundo orden en los ajustes generales:

$$L^{(t)} \approx \sum_{i=1}^{n} \left[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{2.7}$$

donde:

- y_i es el valor real del i-ésimo ejemplo.
- $\hat{y}_i^{(t-1)}$ es la predicción acumulada hasta la iteración $t-1$.
- $f_t(x_i)$ es la predicción del modelo en la iteración t para el i-ésimo ejemplo.
- g_i es el gradiente de la función de pérdida con respecto a $\hat{y}_i^{(t-1)}$, es decir, $g_i = \frac{\partial l(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}}$.
- h_i es el hessiano de la función de pérdida con respecto a $\hat{y}_i^{(t-1)}$, es decir, $h_i = \frac{\partial^2 l(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)^2}}$.

- $\Omega(f_t)$ es el término de regularización para el modelo en la iteración t, definido como:
$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \|w\|^2$$
- γ es el parámetro de regularización que penaliza el número de hojas T.
- λ es el parámetro de regularización $L2$.
- $\|w\|^2$ representa la norma $L2$ del vector de pesos de las hojas del árbol.

Se pueden quitar los valores constantes para obtener la siguiente función objetivo simplificada en el paso t:

$$L^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{2.8}$$

Se define:

$$I_j = \{ i \mid q(x_i) = j \} \tag{2.9}$$

donde:

- I_j es el conjunto de instancias que pertenecen a la hoja j.
- $q(x_i)$ es la función que asigna la instancia x_i a una hoja específica en el árbol.
- j es el índice de la hoja.
- i es el índice de la instancia en el conjunto de datos.

Esta I_j permite agrupar las instancias que terminan en la misma hoja del árbol. Este agrupamiento es esencial para calcular las métricas de evaluación del modelo en cada hoja.

Se puede reescribir la ecuación (2.8) expandiendo Ω de la siguiente forma:

$$\tilde{L}^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{2.10}$$

Si se reorganiza el sumatorio considerando las instancias la hoja resulta:

$$
\begin{aligned}
\tilde{L}^{(t)} &\approx \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t) \\
&= \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2 \\
&= \sum_{j=1}^{T}\left[\sum_{i\in I_j}\left(g_i w_j + \frac{1}{2}h_i w_j^2\right)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2 \\
&= \sum_{j=1}^{T}\left[w_j\sum_{i\in I_j} g_i + \frac{1}{2}w_j^2\left(\sum_{i\in I_j} h_i + \lambda\right)\right] + \gamma T
\end{aligned} \tag{2.11}
$$

Para una estructura fija $q(x)$, se puede calcular el peso óptimo w_j^* de la hoja j con:

$$
w_j^* = -\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i + \lambda} \tag{2.12}
$$

Posteriormente se calcula el correspondiente valor óptimo usando:

$$
\tilde{L}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T \tag{2.13}
$$

donde:

- T es el número total de hojas en el árbol.
- $\sum_{i\in I_j} g_i$ es la suma de los gradientes de la función de pérdida para todas las instancias en la hoja j.
- $\sum_{i\in I_j} h_i$ es la suma de los hessianos (segundas derivadas) de la función de pérdida para todas las instancias en la hoja j.
- λ es el parámetro de regularización $L2$.
- γ es el parámetro de regularización que penaliza el número de hojas T.

La ecuación (2.13) puede ser usada como una función de puntuación para medir la calidad de la estructura q del árbol. Esta puntuación puede entenderse como la puntuación de impureza para evaluar árboles de decisión, a excepción de que se deriva para una gama más amplia de funciones objetivo.

Normalmente, es imposible enumerar todas las posibles estructuras q del árbol. En su lugar, se usa un algoritmo que comienza desde una sola hoja y agrega ramas al árbol de forma iterativa. Se asume que I_L y I_R son los conjuntos de instancias de los nodos de derecha e izquierda después de la división, siendo:

$$I = I_L \cup I_R \tag{2.14}$$

Así, la reducción de pérdidas después de la división viene dada por:

$$\text{Gain} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \tag{2.15}$$

donde:

- I_L es el conjunto de instancias que caen en el nodo izquierdo después de la división.
- I_R es el conjunto de instancias que caen en el nodo derecho después de la división.
- $I = I_L \cup I_R$ es el conjunto de instancias antes de la división.
- g_i es el gradiente de la función de pérdida con respecto a la predicción del modelo para la instancia i.
- h_i es el hessiano (segunda derivada) de la función de pérdida con respecto a la predicción del modelo para la instancia i.
- λ es el parámetro de regularización $L2$.
- γ es el parámetro de regularización que penaliza la complejidad del modelo (número de hojas).

Esta fórmula se usa durante la construcción del árbol para evaluar posibles divisiones y seleccionar la mejor.

2.1.2. Contracción y submuestreo de columnas

Además de lo mencionado en la subsección anterior, se utilizan dos técnicas adicionales para prevenir el sobreajuste. La primera es la técnica de contracción

introducida por Friedman. La contracción escala los pesos recién agregados por un factor η después de cada paso de mejora de los árboles. De una forma similar a una tasa de aprendizaje en la optimización estocástica, la contracción reduce la influencia que tiene cada árbol individual y deja espacio para que los árboles futuros mejoren el modelo.
La segunda técnica es la del submuestreo de columnas. Esta técnica consiste en seleccionar aleatoriamente un subconjunto de características (columnas) del conjunto de datos para entrenar cada modelo individual en el *ensemble*.

2.1.3. Algoritmos de búsqueda dividida

Dentro de los algoritmos de búsqueda dividida se encuentran el algoritmo *Basic Exact Greedy* y el algoritmo aproximado. Uno de los principales problemas en el aprendizaje de árboles es encontrar la mejor división. Para ello, el algoritmo *Basic Exact Greedy* construye árboles de decisión evaluando exhaustivamente todas las posibles divisiones para minimizar la función objetivo, que incluye el error y la regularización. Iterativamente, actualiza el modelo con nuevas predicciones ajustadas por un factor de aprendizaje. Así asegura que en cada paso se toma la mejor decisión posible para reducir el error, garantizando una construcción óptima del árbol en cada iteración. Este algoritmo es computacionalmente demandante [19].

Por otro lado, el algoritmo aproximado construye construye árboles de decisión utilizando técnicas heurísticas para evaluar una selección limitada de posibles divisiones, en lugar de todas, lo que reduce significativamente el tiempo de cálculo. Iterativamente, el modelo se actualiza con nuevas predicciones ajustadas por un factor de aprendizaje. Este método equilibra la precisión y la eficiencia computacional, permitiendo manejar conjuntos de datos grandes. El algoritmo primero propone puntos de división candidatos según los percentiles de la característica. Posteriormente, mapea las características continuas en grupos divididos por esos puntos candidatos, agrega las estadísticas y encuentra la mejor solución entre las propuestas basadas en las estadísticas agregadas. Este algoritmo tiene dos variantes principales: el algoritmo de *Binning* (histograma) y el algoritmo de cuantiles (*Scketching*). El algoritmo de *Binning* agrupa los valores de las características en *bins* (intervalos), y luego calcula las mejores divisiones usando estos *bins* en lugar de los valores originales. Su principal ventaja es que reduce el número de posibles divisiones a considerar, acelerando el proceso de construcción del árbol. Es útil para conjuntos de datos grandes, ya que disminuye el tiempo e cálculo y la memo-

ria necesaria. Por otro lado, el algoritmo de cuantiles utiliza un método de bocetos (*Scketching*) para aproximar la distribución de los datos y encontrar buenos puntos de división de manera eficiente. Permite manejar características con muchas categorías o valores continuos de manera más eficiente. Es, por tanto, adecuado para características de alta cardinalidad, mejorando la escalabilidad del modelo [19].

2.1.4. Algoritmo *Weighted Quantile Sketch*

Un paso importante en el algoritmo aproximado es proponer puntos de división candidatos. Generalmente, se utilizan percentiles de una característica para distribuir uniformemente los candidatos en los datos. Formalmente, dado el conjunto de valores de la k-ésima característica y estadísticas del segundo orden de cada instancia de entrenamiento $D_k = \{(x_{1k}, h_1), (x_{2k}, h_2), \ldots, (x_{nk}, h_n)\}$ se define una función de rango $r_k : \mathbb{R} \to [0, +\infty)$ como:

$$r_k(z) = \frac{1}{P} \sum_{(x,h)\in D_k} h \sum_{(x,h)\in D_k, x<z} h \tag{2.16}$$

Esta función representa la proporción de instancias cuyo valor de la característica k es menor que z. El objetivo es encontrar puntos de división candidatos $\{s_{k1}, s_{k2}, \ldots, s_{kl}\}$, tales que:

$$|r_k(s_{k,j}) - r_k(s_{k,j+1})| < \epsilon, \quad s_{k1} = \min_i x_{ik}, \quad s_{kl} = \max_i x_{ik}. \tag{2.17}$$

Aquí, ϵ es un factor de aproximación. Intuitivamente, esto significa que hay aproximadamente $1/\epsilon$ puntos candidatos. Aquí, cada punto de datos está ponderado por h_i. Para ver por qué h_i representa el peso, podemos reescribir la ecuación (2.8) como:

$$\sum_{i=1}^{n} \frac{1}{2} h_i (f_t(x_i) - g_i/h_i)^2 + \Omega(f_t) + \text{constante} \tag{2.18}$$

Se trata exactamente de una pérdida cuadrada ponderada con etiquetas g_i/h_i y pesos h_i. Para conjuntos de datos grandes, no es trivial encontrar divisiones candidatas que satisfagan los criterios. Cuando cada instancia tiene pesos iguales, existe un algoritmo llamado *Quantile Skech* [6] que resuelve el problema. Sin embargo, no existe un *Quantile Skech* para conjuntos de datos ponderados. Por lo tanto, la mayoría de los algoritmos aproximados existentes dependen de la ordenación

en un subconjunto aleatorio de los datos, lo que conlleva a un riesgo de error o emplea heurísticas que carecen de garantía teórica.

2.1.5. *Sparsity-aware Split Finding*

En muchos problemas del mundo real, es bastante común que la entrada x sea dispersa. Esto puede deberse a múltiples causas: presencia de valores faltantes en los datos, entradas cero frecuentes en las estadísticas o artefactos de ingeniería de características como la codificación one-hot.

Es importante que el algoritmo sea consciente del patrón de escasez en los datos. Para hacerlo, se propone agregar una dirección por defecto en cada nodo del árbol. Cuando un valor está ausente en la matriz dispersa x, la instancia se clasifica en la dirección por defecto. Existen dos opciones por defecto en cada rama del árbol, eligiendose una de ellas por defecto. Las direcciones por defecto óptimas se aprenden a partir de los datos. La mejora clave es visitar únicamente las entradas no faltantes I_k.
El algoritmo trata la ausencia como un valor faltante y aprende la mejor dirección para manejar los valores faltantes. El mismo algoritmo también puede aplicarse cuando la ausencia corresponde a un valor especificado por el usuario, limitando la enumeración sólo a soluciones consistentes.

La mayoría de los algoritmos de aprendizaje de árboles existentes están optimizados solo para datos densos o requieren procedimientos específicos para manejar casos limitados como la codificación categórica. XGBoost maneja todos los patrones de escasez de manera unificada [19].

2.2. Conclusión

En este capítulo, se han explorado en profundidad los fundamentos teóricos del algoritmo XGBoost, una herramienta poderosa y versátil en el campo del aprendizaje automático. XGBoost no sólo se destaca por su capacidad para manejar grandes volúmenes de datos y mejorar de manera significativa la precisión de los modelos a través del *boosting*, sino también por su eficiencia computacional y su capacidad para evitar el sobreajuste mediante técnicas avanzadas de regularización. La inclusión de métodos como el *Weighted Quantile Sketch* y el

Sparsity-aware Split Finding demuestra su adaptabilidad a diferentes tipos de datos y escenarios.

Con estos fundamentos teóricos, se sienta una base sólida para comprender cómo y por qué XGBoost se ha convertido en una herramienta de elección para muchos en la comunidad de ciencia de datos. En los próximos capítulos, se explorará la implementación práctica del algoritmo para predecir el ciclo solar.

Capítulo 3

Metodología y análisis de resultados

En este apartado se hablará sobre los datos utilizados, el preprocesamiento de los mismos, la implementación y posterior evaluación del modelo. El código completo utilizado se encuentra en el Apéndice B.

3.1. Datos

En esta sección se describirá la base de datos utilizada, el preprocesamiento que se les ha realizado y se realizará una exploración de los mismos.

3.1.1. Descripción de las bases de datos

Los datos se han obtenido de la National Oceanic and Atmospheric Administration (NOAA), que es responsable de describir y predecir cambios en el medio ambiente mediante la investigación de los océanos, la atmósfera, el espacio y el Sol. Se trata de datos públicos, recogidos mensualmente desde enero de 1749 hasta julio de 2024.

Para el caso particular de las manchas solares recogen las siguientes variables:

- ***Time tag***: Muestra la fecha de la observación en formato año-mes. El tipo de dato es una cadena de texto o *string*.
- ***Sunspot Number (SNN)***: Representa el número de manchas solares visibles en el Sol. Se trata de un contador de manchas solares individuales y un conteo de grupos de manchas solares. Además, se trata de una métrica fundamental para el estudio de los ciclos solares. Varía en ciclos de aproximadamente 11 años, conocidos como ciclos solares. Cada ciclo tiene un

periodo de máxima actividad solar (máximo solar), que está asociado con un mayor número de eventos solares como erupciones solares o eyecciones de masa coronal, y un periodo de mínima actividad solar (mínimo solar).

- ***Smoothed Sunspot Number (Smoothed SNN)***: Número de manchas solares suavizado. Se trata de un promedio móvil que se usa para reducir la variabilidad diaria y resaltar las tendencias a largo plazo en la actividad solar.

- ***Observed SWPC Sunspot Number (Observed SWPC SNN)***: Número de manchas solares observado y registrado por el SWPC, que es una parte de la Administración Nacional Oceánica y Atmosférica (NOAA) de los Estados Unidos. Se trata de una medida diaria de la cantidad de manchas solares visibles en el Sol. Es fundamental para monitorizar la actividad solar y entender el ciclo solar, ya que permite a los científicos y meteorólogos espaciales seguir los ciclos solares, identificando los periodos de máxima y mínima actividad solar. También ayuda a predecir eventos solares como erupciones o eyecciones de masa coronal, que pueden impactar a la Tierra, así como facilitar el estudio de la dinámica solar en el clima espacial y terrestre.

- ***Smoothed SWPC Sunspot Number (Smoothed SWPC SNN)***: Número de manchas solares suavizado registrado por el SWPC. Representa un promedio móvil para reducir la variabilidad diaria y resaltar las tendencias a largo plazo.

- ***Índice de flujo solar F10.7 cm (F10.7)***: Es una medida de la radiación solar en una longitud de onda de 10.7 centímetros (alrededor de 2800 MHz). Es medido diariamente por radiotelescopios y las medidas se toman a una misma hora cada día para asegurar la consistencia y comparabilidad de los datos a lo largo del tiempo. Se trata de un indicador clave de la actividad solar. Un mayor flujo F10.7 indica una mayor actividad solar. Además, el ciclo solar afecta a la ionosfera terrestre, influyendo en la propagación de ondas de radio y en las condiciones del clima espacial. Es crucial para la predicción de condiciones en las comunicaciones por radio y satélite.

- ***Smoothed F10.7***: Índice del flujo solar F10.7 suavizado, calculado mediante un promedio móvil de 12 meses. Ayuda a predecir eventos solares y sus efectos en la ionosfera terrestre de manera más precisa que los datos no suavizados. Además, en la investigación científica facilita el estudio de la

dinámica solar y su influencia en el entorno espacial y terrestre, al eliminar la variabilidad a corto plazo.

3.1.2. Preprocesamiento de datos

El preprocesamiento de datos es una etapa fundamental en cualquier proyecto de modelado predictivo, especialmente cuando se trabaja con series temporales. En este trabajo, se han seguido varios pasos para transformar y preparar los datos de manera que sean adecuados para el entrenamiento del modelo de *machine learning* basado en XGBoost. A continuación, se detallan las principales etapas del preprocesamiento.

3.1.2.1. Carga y formateo de los datos

Se cargan los datos en formato JSON. A partir de este archivo, se extraen las columnas relevantes para el análisis, que incluyen:

- **Fecha (*time-tag*)**: Convertida al formato *datetime* para facilitar su manipulación.
- ***Sunspot Number (ssn)***: Número de manchas solares observadas.
- ***Smoothed Sunspot Number (smoothed_ssn)***: Número de manchas solares suavizado.

3.1.2.2. Generación del Índice Estacional

Dado que el ciclo solar tiene un componente estacional muy marcado, se introdujo un índice estacional que repite un ciclo cada 11 años, correspondiente a la duración típica del ciclo solar. Este índice se incorpora al conjunto de datos como una característica adicional, permitiendo al modelo capturar patrones estacionales que pueden ser críticos para la predicción.

Primero, se crea un vector que se repite cíclicamente, representando la estacionalidad del ciclo solar. Este vector se genera utilizando una función personalizada, que toma como entrada la longitud de la serie temporal (en meses) y el número de meses que abarca un ciclo completo (132 meses). La función genera un vector cíclico que contiene valores de 1 a 132, repitiéndose a lo largo de toda la longitud

de la serie de datos.

Para su implementación en el código, se define la función *generate_vector*, la cual crea un iterador cíclico que recorre los valores del 1 al 132. Estos valores se asignan a un nuevo campo en el conjunto de datos denominado *Index_for_seasonality*. Esta columna actúa como una característica adicional para el modelo, indicando en qué parte del ciclo solar (del 1 al 132) se encuentra cada observación.

3.1.2.3. Transformación de la Serie Temporal en un Problema de Aprendizaje Supervisado

El principal desafío en la modelización de series temporales es transformar la secuencia temporal en un problema de aprendizaje supervisado, donde el objetivo es predecir valores futuros basados en observaciones pasadas.

Para ello, en este trabajo la serie temporal se ha reconvertido en un conjunto de datos observacionales en función del tiempo. Se han programado tres funciones para este fin (Código del programa). Primero, se crea la función *Extended_titles*, que genera una lista de nombres de columnas extendidos para un *DataFrame*. Estos nombres indican la posición temporal de cada valor en relación con el tiempo actual. Seguidamente, se crea la función *series_to_supervised*, que organiza los datos que cada fila incluya los datos de días anteriores como entrada y días futuros como salida. Por último se crea la función *to_supervided2*, que une todo el proceso anterior, generando un *DataFrame* completo con los nombres de las columnas adecuadas para que pueda ser utilizado en un modelo de aprendizaje automático.

3.1.2.4. División de los datos en conjuntos de entrenamiento y prueba

El objetivo principal de dividir los datos en conjuntos de entrenamiento y prueba es asegurarse de que el modelo se pueda generalizar bien a nuevos datos, es decir, que no sólo funcione bien en los datos en los que fue entrenado, sino que también pueda hacer predicciones precisas en datos que nunca ha visto antes.

Par ello, realizamos una serie de transformaciones claves en el *DataFrame*. Primero, se renombran las columnas para facilitar su comprensión. Posteriormente, se convierte la columna *time-tag* a formato fecha y se reorganiza el *DataFrame* para situarla en la primera posición. Luego, se reordenan las columnas para

que sigan en el formato estándar y se reemplazan los valores *-1* por *NaN* (*Not a number*) para representar los valores faltantes. Finalmente, se crea una copia del *DataFrame* para poder realizar análisis adicionales sin modificar el conjunto de datos original.

Seguidamente, se configuran los datos para preparar el modelo. Primero, se definen dos parámetros clave: *n_in*, que especifica el número de observaciones anteriores usadas para hacer predicciones, y *n_out*, que determina el número de observaciones que se quieren predecir. Luego, se prepara el DataFrame *Df* utilizando la función *to_supervided2* previamente definida, para prepararlo para el aprendizaje supervisado, se eliminan las filas con valores faltantes y se establece la columna *Date (t)* como índice. A continuación, se genera un vector de estacionalidad que se repite a lo largo del *DataFrame*, y se inserta en una nueva columna llamada *Index_for_seasonality*. Se dividen las columnas en características (*features*) y objetivos (*targets*), seleccionando sólo las relevantes. Posteriormente, se convierten los datos a formato numérico. Esto es importante porque los modelos de *machine learning* generalmente sólo trabajan con datos numéricos. Finalmente, se divide el conjunto de datos en entrenamiento *(X_train, Y_train)* y de prueba *(X_test, Y_test)* basándose en una fecha de corte, con los datos anteriores a 1971 asignados al conjunto de entrenamiento y los datos posteriores al conjunto de prueba. El conjunto de entrenamiento se usa para ajustar los parámetros internos del modelo de manera que pueda hacer predicciones basadas en las características de entrada. Por otro lado, el conjunto de prueba se mantiene separado del proceso de entrenamiento y estos datos se utilizan únicamente para evaluar la capacidad del modelo de hacer predicciones en datos no vistos.
Este paso es importante para evitar el sobreajuste. Al mantener un conjunto de prueba separado, podemos evaluar si el modelo realmente ha aprendido algo útil.

3.1.2.5. Validación cruzada y ajuste del modelo

Para optimizar el rendimiento del modelo XGBoost, se implementó un esquema de validación cruzada basado en la técnica *TimeSeriesSplit*, la cual preserva el orden temporal de los datos. Se definió una cuadrícula de hiperparámetros (*param_grid*) que incluye la tasa de aprendizaje, profundidad del árbol, número de estimadores, y otros factores clave. Posteriormente, se utilizó *RandomizedSearchCV* para evaluar todas las combinaciones posibles de estos hiperparámetros, seleccionando aquellos que minimizan el error cuadrático medio (MSE). Los mejores

parámetros obtenidos se emplearon para entrenar el modelo final.

3.1.3. Exploración de datos

En esta sección, se muestran gráficos que ayudan a explorar y entender los datos del modelo.

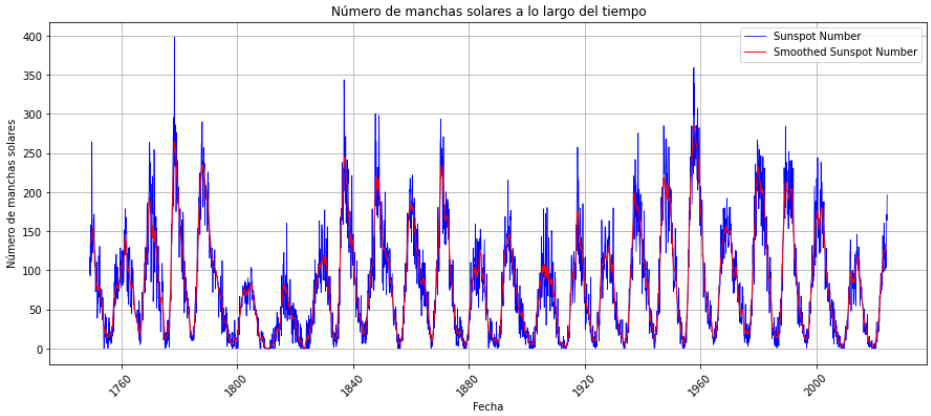

Figura 3.1: Número de manchas solares a lo largo del tiempo

En la figura 3.1 se presenta el gráfico de series temporales del número de manchas solares observadas y el número suavizado de manchas solares a lo largo del tiempo. Este gráfico revela varias características clave del ciclo solar. En primer lugar, se puede observar una tendencia cíclica que refleja los ciclos solares aproximadamente cada 11 años. Estos ciclos están marcados por fluctuaciones en el número de manchas solares, con períodos de incremento seguidos por disminuciones notables. Además, se observan períodos de alta y baja variabilidad en el número de manchas solares. Los picos representan eventos de alta actividad solar, mientras que los valles indican períodos de baja actividad. Estos cambios en la variabilidad pueden estar asociados con eventos solares específicos o con cambios en las condiciones solares globales.

Seguidamente, en la figura 3.2 se muestra un histograma para entender la distribución del número de manchas solares.

El gráfico muestra un sesgo a la derecha, donde la mayoría de los valores corresponden a un número bajo de manchas solares (entre 0 y 25), mientras que

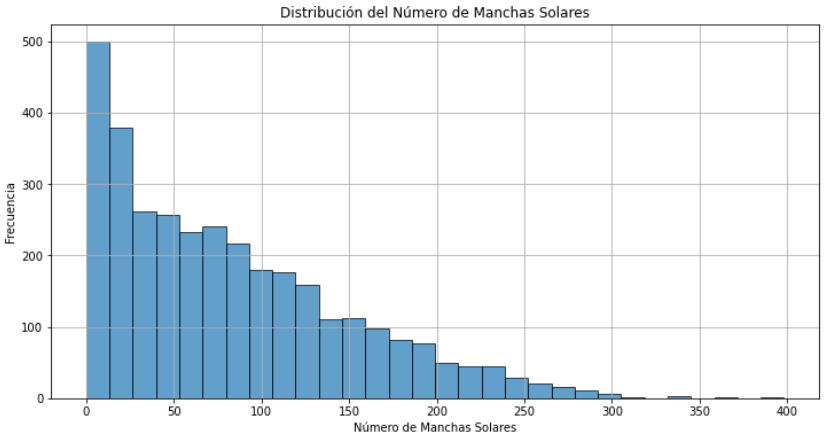

Figura 3.2: Histograma del número de manchas solares

los valores altos, aunque menos frecuentes, son significativos. Este sesgo indica que los picos de alta actividad solar son eventos raros pero importantes. XGBoost, siendo un modelo basado en árboles de decisión, es una buena elección para este caso, ya que maneja eficientemente distribuciones sesgadas y valores atípicos (*outliers*), permitiendo capturar tanto los períodos de baja actividad como los picos, sin necesidad de transformar los datos previamente.

Posteriormente, se ha elaborado un gráfico de dispersión para examinar detalladamente la relación entre variables. El gráfico de dispersión 3.3 se ha diseñado para ilustrar cómo el número de manchas solares evoluciona a lo largo del tiempo, permitiendo así confirmar la existencia de patrones específicos y tendencias significativas en los datos.

En el gráfico se incluye una línea de media móvil de 12 meses (en rojo) y una representación del estadístico Z (*Z-Score*) para detectar anomalías. La forma ondulante en el gráfico de dispersión y la alineación de los picos y valles con la línea de media móvil confirman la existencia del ciclo solar de aproximadamente 11 años en los datos de manchas solares. Este patrón cíclico es una característica clave en el análisis de la actividad solar. Por otro lado, el análisis del estadístico Z ayuda a identificar eventos que se desvían significativamente del patrón esperado. Estas anomalías pueden indicar eventos solares inusuales o cambios en la actividad solar que no siguen el ciclo regular.

Seguidamente, se analizan las matrices de correlación entre las distintas variables para tratar de detectar posibles redundancias, patrones y la selección de

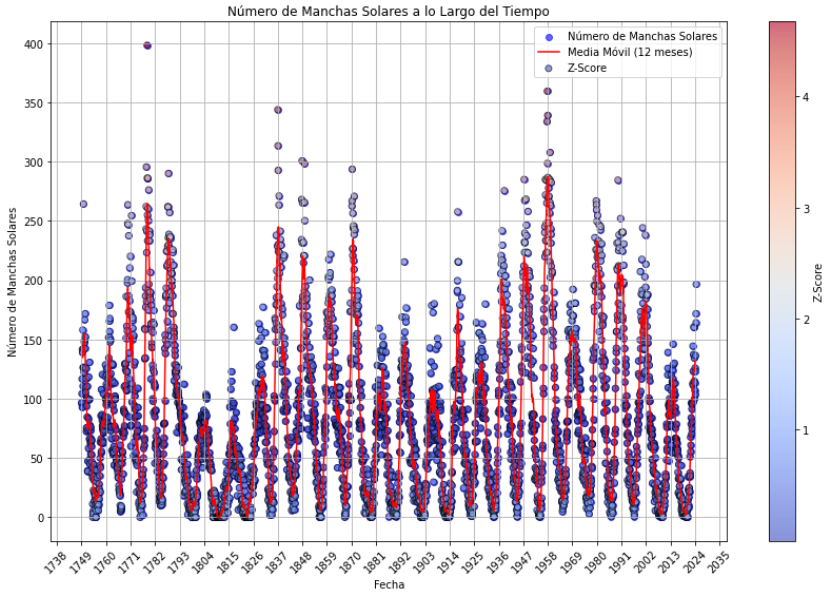

Figura 3.3: Número de Manchas Solares a lo Largo del Tiempo con Media Móvil y estadístico Z

características relevantes que mejoren la precisión y eficiencia del modelo.

Se realiza una primera matriz de correlación utilizando todos los datos disponibles. Los resultados revelan una fuerte correlación positiva entre variables, como el número de manchas solares y su versión suavizada, así como entre el número de manchas solares observado por SWPC y su versión suavizada, o entre el índice de radio solar F10.7 y su correspondiente valor suavizado. Estas correlaciones eran esperadas, dado que las versiones suavizadas de las variables deben reflejar tendencias similares a las de sus contrapartes originales, pero con menos ruido.

Además, se identificó una fuerte correlación positiva entre la media móvil y el estadístico Z, así como entre el número de manchas solares (SSN) y la media móvil, el SSN y el estadístico Z, y entre el SSN suavizado con la media móvil y el estadístico Z. Estas correlaciones indican que las transformaciones aplicadas a las series temporales logran reflejar patrones consistentes en los datos.

Por otro lado, no se encontró ninguna correlación negativa fuerte entre las variables, y la mayoría de las otras relaciones presentan correlaciones débiles o nulas, lo que indica que no hay una relación lineal significativa entre estas variables. Esto podría sugerir que algunas de las variables en el conjunto de datos son

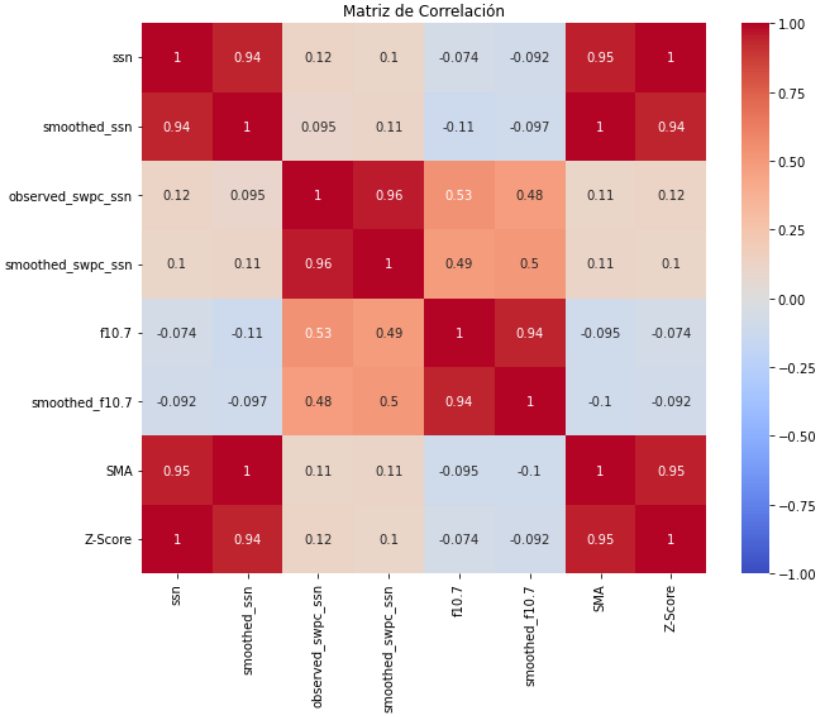

Figura 3.4: Matriz de correlación

independientes o que sus relaciones podrían no ser lineales. Los resultados de este análisis de la matriz de correlación se muestran en la figura 3.4

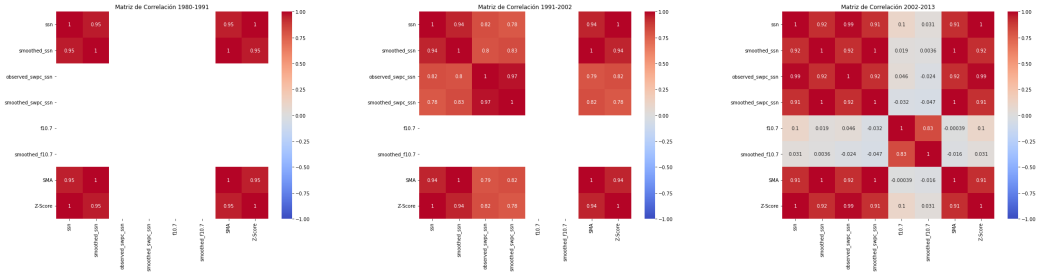

Figura 3.5: Matrices de correlación temporales

A continuación, se calcularon matrices de correlación para periodos de 11 años, correspondientes a la duración de un ciclo solar. En la figura 3.5 se presentan tres ejemplos representativos: el primero abarca de 1980 a 1991, el segundo de 1991 a 2002 y el tercero de 2002 a 2013.

En el primer periodo, los datos disponibles se limitan a las variables *SSN* y *SSN suavizado*, mostrando una correlación positiva fuerte entre ellas, lo cual es esperable. Asimismo, se observa una fuerte correlación positiva de ambas variables con la media móvil y el estadístico Z, indicando la coherencia entre las series temporales y las medidas derivadas de ellas.

En la segunda matriz de correlación, se observa una fuerte correlación positiva entre *SSN* y *SSN suavizado*, así como con el número de manchas solares observado y suavizado por SWPC. La correlación positiva también se extiende entre *SSN suavizado* y las medidas observadas y suavizadas por SWPC. Además, la media móvil y el estadístico Z muestran fuertes correlaciones positivas con todas estas variables (*SSN*, *SSN suavizado*, número de manchas solares observado y suavizado por SWPC). Es importante mencionar que no se cuentan con datos para *f10.7* y *f10.7 suavizado*.

Por último, en el tercer periodo se observa una fuerte correlación positiva entre *SSN*, *SWPC SSN* y *F10.7* con sus correspondientes versiones suavizadas. También se observa una fuerte correlación positiva del *SSN* con el *SWPC SSN* y el *SSN* con la versión suavizada del *SWPC*. Sin embargo, la correlación del *F10.7* y su versión suavizada con el resto de las variables es nula o débil. No se observa ninguna correlación negativa fuerte.

3.1.4. Estadísticas descriptivas

En la tabla 3.1 se presentan los valores obtenidos para ciertos parámetros estadísticos habituales para las variables *SSN* y *Smoothed SSN*.

El análisis descriptivo de las variables *SSN* y *Smoothed SSN* revela varias características clave de sus distribuciones. La proximidad entre la media y la mediana de ambas variables sugiere que las distribuciones están bien centradas, sin mostrar asimetrías extremas. Sin embargo, *SSN* presenta una mayor desviación estándar y varianza en comparación con *Smoothed SSN*, indicando una mayor dispersión en sus valores.

Ambas series presentan un sesgo positivo, lo que indica una ligera inclinación hacia valores más altos. La curtosis de *SSN* es positiva, sugiriendo colas más pesadas en la distribución, mientras que la curtosis de *Smoothed SSN* es negativa, indicando colas más ligeras. Además, el rango intercuartílico (IQR) de *SSN* es mayor que el de *Smoothed SSN*, reflejando una mayor dispersión en el centro de la distribución. El rango total también es más amplio para *SSN* en comparación con *Smoothed SSN*, evidenciando una mayor variabilidad en los valores de *SSN*. Estos

Estadística	SSN	Smoothed SSN
Media	81.91	81.43
Desviación Estándar	67.67	63.13
Varianza	4579.67	3985.03
Sesgo (Skewness)	0.92	0.77
Curtosis (Kurtosis)	0.34	-0.17
Cuartil Q1	24.15	25.20
Mediana (Q2)	68.00	71.70
Cuartil Q3	122.60	119.20
Rango Intercuartílico (IQR)	98.45	94.00
Rango	398.20	286.00

Tabla 3.1: Estadísticas descriptivas para las variables SSN y Smoothed SSN.

resultados destacan las diferencias en la variabilidad y la forma de las distribuciones de ambas series temporales, proporcionando una comprensión más profunda de sus características y comportamientos.

3.2. Metodología

3.2.1. Selección de variables

En cuanto a la selección de variables, las características (*features*) fueron seleccionadas identificando aquellas columnas que contienen información sobre las manchas solares y su versión suavizada, excluyendo explícitamente las columnas relacionadas con las fechas. Este enfoque asegura que el modelo utilice únicamente datos relevantes para capturar la dinámica de las manchas solares sin introducir variables que no aporten valor predictivo.

En cuanto a las variables objetivo (*targets*), se seleccionaron los valores futuros de las manchas solares, específicamente aquellos que no han sido suavizados. Esta selección permite que el modelo enfoque su predicción en los datos originales, preservando la variabilidad natural de los ciclos solares.

Estas variables fueron utilizadas en el modelo XGBoost para optimizar la precisión en la predicción del ciclo solar.

3.2.2. División del conjunto de datos

Para la división de los datos en el modelo XGBoost, se realizó una partición en la que el 80 % de los datos se destinó al entrenamiento y el 20 % restante se utilizó para la prueba. Esta partición como se ha indicado anteriormente, se estableció fijando enero de 1971 como el punto de separación entre ambos conjuntos. De esta manera, el conjunto de entrenamiento comprende los datos desde enero de 1749 hasta diciembre de 1970, mientras que el conjunto de prueba abarca desde enero de 1971 hasta julio de 2024. Esta división asegura que el modelo se entrene con datos históricos extensos y se evalúe con datos recientes.
Como se comentó en la subsección 3.1.2.5, se implementó la validación cruzada temporal para evaluar la capacidad de generalización del modelo al dividir los datos, reduciendo el riesgo de sobreajuste y optimizando su rendimiento.

3.2.3. Implementación del algoritmo XGBoost

El modelo XGBoost final fue entrenado utilizando los mejores hiperparámetros obtenidos a través de la validación cruzada. Estos hiperparámetros incluyeron la tasa de aprendizaje, la profundidad máxima de los árboles, el peso mínimo de los nodos hijos, el número de árboles, y los parámetros de regularización L2 y de reducción mínima de pérdida. Esta configuración permitió optimizar la precisión del modelo al predecir el ciclo solar.

3.3. Resultados

3.3.1. Entrenamiento del modelo

En el proceso de entrenamiento del modelo, se utilizó una búsqueda de hiperparámetros para optimizar el rendimiento del modelo XGBoost en la predicción de la serie temporal de manchas solares. La búsqueda se llevó a cabo utilizando *RandomizedSearchCV*, que permite explorar una amplia gama de combinaciones de hiperparámetros de manera más eficiente en comparación con la búsqueda exhaustiva (*GridSearchCV*).

La búsqueda de hiperparámetros incluyó las siguientes configuraciones:

- **Tasa de aprendizaje (learning_rate):** [0.001, 0.01]
- **Profundidad máxima del árbol (max_depth):** [3, 5]

- **Peso mínimo de un nodo hijo (min_child_weight):** [1, 3]
- **Número de árboles (n_estimators):** [100, 300]
- **Regularización L2 (lambda):** [0.1, 1]
- **Reducción mínima de pérdida (gamma):** [0, 0.1]

Para asegurar la robustez del modelo, se utilizó un esquema de validación cruzada específico para series temporales (*TimeSeriesSplit*) con 3 divisiones, lo que permitió evaluar el rendimiento del modelo en diferentes particiones del conjunto de datos sin romper la secuencia temporal.

Tras ejecutar el proceso de búsqueda, los mejores hiperparámetros encontrados fueron:

- **Número de árboles (n_estimators):** 300
- **Peso mínimo de un nodo hijo (min_child_weight):** 1
- **Profundidad máxima del árbol (max_depth):** 3
- **Tasa de aprendizaje (learning_rate):** 0.01
- **Regularización L2 (lambda):** 0.1
- **Reducción mínima de pérdida (gamma):** 0

Estos parámetros fueron utilizados en la versión final del modelo para realizar las predicciones, optimizando así la precisión en la estimación de la serie temporal de manchas solares.

3.3.2. Interpretación de resultados

En la figura 3.6 se comparan los datos reales y las predicciones del modelo XGBoost. Como puede observarse, la gráfica muestra que el modelo ha logrado un ajuste notablemente preciso a lo largo del período analizado. Las predicciones de XGBoost siguen de cerca las variaciones y tendencias de los datos reales, capturando tanto las fluctuaciones estacionales como los cambios abruptos en el ciclo solar. El modelo XGBoost parece manejar eficazmente tanto los picos como los valles en los datos, lo que resalta su capacidad para modelar las complejas dinámicas de las manchas solares. La correspondencia visual entre las predicciones

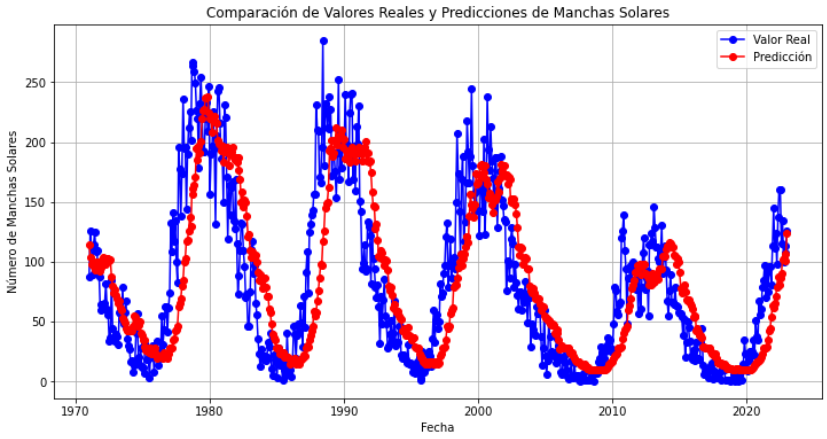

Figura 3.6: Comparativa predicciones XGBoost

y los datos reales confirma la eficacia de XGBoost en comparación con métodos más simples, como se evidenció previamente con el calculo de métricas de error. Los datos reales y las predicciones generadas por el modelo pueden consultarse en detalle en la tabla del Apéndice A.

3.3.3. Evaluación del modelo

Para evaluar el rendimiento del modelo, se utilizaron varias métricas estadísticas, incluyendo el Error Absoluto Medio (MAE), el Error Cuadrático Medio (MSE) y la Raíz del Error Cuadrático Medio (RMSE). Los valores obtenidos se muestran en la figura 3.7. Estas métricas permiten cuantificar la precisión del modelo en la predicción del ciclo solar, proporcionando una medida del error entre las predicciones y los valores reales.

La validación del modelo se llevó a cabo utilizando un conjunto de datos de prueba, separado del conjunto de datos de entrenamiento, para garantizar que los resultados obtenidos sean generalizables y no estén sobreajustados al conjunto de datos de entrenamiento. El rendimiento del modelo se comparó con un modelo baseline que utiliza la media histórica de los datos como predicción.

- **MAE:** El MAE mide el error promedio en unidades absolutas entre las predicciones y los valores reales. Para el modelo propuesto, el MAE es de 33.14 manchas, significativamente menor que el MAE del modelo baseline (media histórica), lo que indica que el modelo captura mejor los patrones en los datos.

- **MSE:** El MSE toma en cuenta el cuadrado de los errores, penalizando de manera más severa los errores grandes. El MSE del modelo es 1883.79 manchas, mientras que el MSE del baseline es 4785.86 manchas, lo que representa una mejora del 60.6 %. Esta reducción en el MSE demuestra la superioridad del modelo avanzado en la predicción.

- **RMSE:** El RMSE, que es la raíz cuadrada del MSE, proporciona una métrica en las mismas unidades que las predicciones. El RMSE del modelo XGBoost es 43.40 manchas, frente a 69.18 manchas del baseline, reflejando una mejora considerable en la precisión. Dado un rango de 284.5 en los datos, el modelo XGBoost muestra una notable capacidad para capturar la variabilidad de los datos, confirmando su eficacia en comparación con el modelo simple.

Finalmente, se analizó la distribución de los errores para verificar si el modelo comete errores sistemáticos o si estos se distribuyen aleatoriamente, lo cual es un buen indicio de la robustez del modelo. Esta evaluación confirma que el modelo XGBoost no solo mejora las métricas de error respecto al baseline, sino que también presenta una distribución de errores más controlada, lo que indica su fiabilidad en la predicción del ciclo solar.

```
MAE del modelo: 33.14219651580811
MAE del baseline: 57.68259309297913
Mejoría del modelo sobre el baseline: 42.54%
Model improvement over baseline: 42.54%
MSE del modelo: 1883.7990068169936
MSE del baseline: 4785.86738816
Mejoría del modelo sobre el baseline: 60.64%
RMSE del modelo: 43.402753447413836
RMSE del baseline: 69.17996377680463
Rango de los valores: 284.5
```

Figura 3.7: Valores MAE, MSE Y RMSE

3.3.4. Comparativas

Se compararon los resultados obtenidos de diferentes modelos predictivos aplicados a los datos del ciclos solar con los obtenido por el modelo XGBoost realizado. Se comparó con los modelos de la Media Móvil y el Promedio Exponencial. Para ello, se han utilizado dos métricas principales para la evaluación: el Error Cuadrático Medio (MSE) y la Raíz del Error Cuadrático Medio (RMSE),

```
MSE del modelo de Media Móvil: 724.6574041666668
RMSE del modelo de Media Móvil: 26.91946143901595
MSE del modelo de Promedio Exponencial: 268592.9166649728
RMSE del modelo de Promedio Exponencial: 518.2595070666555
MSE del modelo XGBoost: 1883.7990068169936
RMSE del modelo XGBoost: 43.402753447413836
```

Figura 3.8: Comparativas entre modelos

las cuales proporcionan una visión detallada de la precisión y el ajuste de cada modelo.

En la figura 3.8 se observa un MSE de la Media Móvil relativamente bajo en comparación con el Promedio Exponencial, pero significativamente más alto que el de XGBoost. Su RMSE también es menor en comparación con el Promedio Exponencial, indicando que el modelo tiene una capacidad moderada para capturar las variaciones en los datos, aunque no es el mejor en términos de precisión absoluta.

Por otro lado, el modelo de Promedio Exponencial presenta un MSE considerablemente alto y un RMSE elevado, sugiriendo que tiene un desempeño deficiente en comparación con los otros modelos. Estos resultados indican que el modelo de Promedio Exponencial no se ajusta adecuadamente a los datos, lo que puede ser debido a su incapacidad para capturar patrones complejos en las series temporales de manchas solares.

El modelo XGBoost destaca por su bajo MSE y RMSE en comparación con los otros modelos. Con un MSE significativamente menor que el de la Media Móvil y el Promedio Exponencial, y un RMSE más competitivo, XGBoost demuestra una capacidad superior para ajustarse a los datos y hacer predicciones precisas. Estos resultados confirman que XGBoost es el modelo más efectivo entre los evaluados, ofreciendo un equilibrio robusto entre precisión y capacidad de modelado en la predicción de ciclos solares.

En resumen, mientras que la Media Móvil ofrece una solución básica con un rendimiento aceptable, y el Promedio Exponencial muestra limitaciones en su ajuste, XGBoost se posiciona como el modelo de referencia para la predicción precisa de datos complejos en este contexto. Este análisis confirma que XGBoost no solo mejora la precisión de las predicciones, sino que también maneja mejor las variaciones en los datos, alineándose con el objetivo de obtener un modelo predictivo más avanzado y efectivo.

Tabla 3.2: Análisis del rendimiento de modelos de predicción de manchas solares: Comparativa de MAE y RMSE

Modelo	RMSE	MAE
SARIMA	54.11	45.51
Exponential Smoothing	61.41	49.76
Prophet	60.15	56.09
LSTM	46.14	39.44
GRU	37.14	26.77
Transformer	33.99	25.56
Informer	29.90	22.35
NASA	48.38	38.45
Modelo XGBoost realizado	**43.40**	**1883.80**

Por otro lado, se ha llevado a cabo una comparación entre las predicciones de nuestro modelo y las del modelo presentado en el artículo [21] utilizando las métricas de error RMSE y MAE. Aunque los conjuntos de datos utilizados para entrenar y evaluar los modelos son diferentes en términos de período temporal (el modelo realizado utiliza datos desde enero de 1749 hasta julio de 2024, mientras que los modelos del artículo están basados en datos hasta enero de 2022), la comparación directa de estas métricas sigue siendo válida.

En la tabla 3.2 se observan diferencias notables en términos de las métricas de error. El modelo XGBoost muestra un RMSE de **43.40**, que es significativamente menor que el RMSE de modelos como SARIMA (54.11), *Exponential Smoothing* (61.41) y *Prophet* (60.15), pero mayor que los de GRU (37.14), *Transformer* (33.99) y *Informer* (29.90). Esto sugiere que, aunque XGBoost presenta una mejora respecto a ciertos modelos, no alcanza el rendimiento superior de los modelos GRU, *Transformer* o *Informer* en términos de RMSE. Sin embargo, el MAE del modelo XGBoost es notablemente alto con **1883.80**, en comparación con los valores mucho menores de los modelos base como GRU (26.77), *Transformer* (25.56) y *Informer* (22.35). Estos resultados indican que, a pesar de su desempeño aceptable en términos de RMSE, XGBoost no iguala la precisión absoluta de los modelos más precisos.

En resumen, mientras que el modelo XGBoost puede ofrecer mejoras en comparación con algunos modelos en términos de RMSE, su MAE elevado sugiere que podría no ser la mejor opción para obtener predicciones con un margen de

error absoluto bajo y consistente. Por tanto, se puede concluir que el modelo XGBoost, con un RMSE de **43.40**, supera a varios modelos base como SARIMA y Exponential Smoothing en precisión. Sin embargo, su MAE de **1883.80** es significativamente más alto que el de los modelos más precisos como GRU, Transformer e Informer. Esto sugiere que, aunque XGBoost ofrece una mejora en términos de RMSE, no iguala la precisión absoluta de los modelos más avanzados.

Capítulo 4

Resumen y conclusiones

En este capítulo final, se realiza un breve resumen del estudio realizado y las conclusiones obtenidas a partir de este.

4.1. Resumen y conclusiones de los resultados

El objetivo principal de este estudio fue predecir el número de manchas solares utilizando un modelo basado en datos históricos y versiones suavizadas de las series temporales. Los datos utilizados incluyen registros históricos de manchas solares y valores suavizados para mejorar la precisión del análisis.

Las estadísticas descriptivas revelaron que la media del número de manchas solares es 81.91, con una desviación estándar de 67.67, lo que indica una alta dispersión en los datos. La distribución presenta un sesgo positivo (0.92), sugiriendo que la distribución de las manchas solares está ligeramente asimétrica hacia valores más altos. La curtosis es baja (0.34), lo que sugiere que la distribución no tiene picos pronunciados y es relativamente plana en comparación con una distribución normal. Los cuartiles y el rango intercuartílico muestran que el número de manchas solares tiene una dispersión considerable, con un rango intercuartílico de 98.45, reflejando una variabilidad significativa en los datos. Finalmente, el rango total de 398.2 entre los valores máximos y mínimos observados destaca la amplia gama de valores presentes en el conjunto de datos.

El modelo desarrollado ha demostrado ser eficaz para predecir el número de manchas solares, con una capacidad de ajuste buena a los datos históricos. Los

resultados de la evaluación del modelo XGBoost muestran un desempeño notablemente mejorado en comparación con los modelos de referencia. El Error Absoluto Medio del modelo XGBoost es de 33.14, lo que representa una mejora del 42.54 % en comparación con el baseline (MAE de 57.68). Además, el Error Cuadrático Medio (MSE) del modelo XGBoost es de 1883.80, superior al MSE del modelo de Media Móvil (724.66) y significativamente mejor que el del modelo de Promedio Exponencial (268592.92). El Modelo XGBoost también supera al modelo de Media Móvil en términos de Error Cuadrático Medio, con 43.40 frente a 26.92, aunque queda por debajo del modelo de Promedio Exponencial (518.26).

A pesar de estas mejoras, es importante considerar las limitaciones del estudio. Aunque XGBoost ha mostrado un rendimiento competitivo en comparación con algunos modelos, su MAE es significativamente más alto que el de modelos avanzados como GRU, *Transformer* o *Informer*, lo que indica que XGBoost podría no ser la mejor opción para obtener predicciones con un margen de error absoluto bajo y consistente. El modelo XGBoost ha demostrado ser superior a los modelos tradicionales de Media Móvil y Promedio Exponencial en términos de RMSE, pero su rendimiento en MAE es menos favorable. Además, la complejidad del modelo XGBoost puede llevar a problemas de sobreajuste si no se ajustan adecuadamente los hiperparámetros, y su menor interpretabilidad en comparación con modelos más simples es una limitación adicional.

En el futuro, se recomienda explorar técnicas adicionales de preprocesamiento y ajuste de hiperparámetros, así como considerar la integración de otras fuentes de datos que puedan enriquecer el modelo. También es aconsejable realizar una búsqueda más exhaustiva de hiperparámetros y validaciones cruzadas adicionales para optimizar el modelo y mejorar su robustez. En resumen, a pesar de sus limitaciones, el modelo XGBoost proporciona una base sólida para la predicción de manchas solares y ofrece una mejora en comparación con algunos modelos base, aunque no iguala la precisión de los modelos más avanzados.

Referencias

[1] Marcote, B. (2009). La actividad solar y su efecto en el clima terrestre. *Agrupación Astronómica Astrosantander*.

[2] Bekker, S., Milligan, R. O., & Ryakhovsky, I. A. (2024). The influence of different phases of a solar flare on changes in the total electron content in the Earth?s ionosphere. *The Astrophysical Journal*, 971(2), 188. IOP Publishing.

[3] Jadav, R. M., Iyer, K. N., Joshi, H. P., & Vats, H. O. (2005). Coronal mass ejection of 4 April 2000 and associated space weather effects. *Planetary and Space Science*, 53(6), 671–679. Elsevier.

[4] Lean, J., & Rind, D. (1999). Evaluating sunclimate relationships since the Little Ice Age. *Journal of Atmospheric and Solar-Terrestrial Physics*, 61(1-2), 25–36. Elsevier.

[5] Gray, L. J., Beer, J., Geller, M., Haigh, J. D., Lockwood, M., Matthes, K., Cubasch, U., Fleitmann, D., Harrison, G., Hood, L., & others. (2010). Solar influences on climate. *Reviews of Geophysics*, 48(4). Wiley Online Library.

[6] Greenwald, M., & Khanna, S. (2001). Space-efficient online computation of quantile summaries. *ACM SIGMOD Record*, 30(2), 58–66. ACM New York, NY, USA.

[7] Zhang, Q., & Wang, W. (2007). A fast algorithm for approximate quantiles in high speed data streams. In *Proceedings of the 19th International Conference on Scientific and Statistical Database Management (SSDBM 2007)* (pp. 29–29). IEEE.

[8] Hathaway, D. H. (2015). The solar cycle. *Living Reviews in Solar Physics*, 12(1), 4. Springer.

[9] Samuel, A. (1959).

[10] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[11] Semi-Supervised Learning. (2006). Semi-supervised learning. *CSZ2006. html*, 5, 2.

[12] Sutton, R. S. (2018). Reinforcement learning: An introduction. *A Bradford Book*.

[13] Crawshaw, M. (2020). Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

[14] Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4), e1249. Wiley Online Library.

[15] Morales, L. N. Z. (2023). Ensemble learning. *Journal of Machine Learning*, 15(3), 123–145.

[16] Mienye, I. D., & Sun, Y. (2022). A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *IEEE Access*, 10, 99129–99149. IEEE.

[17] Ngo, G., Beard, R., & Chandra, R. (2022). Evolutionary bagging for ensemble learning. *Neurocomputing*, 510, 1–14. Elsevier.

[18] Matthew, W. (2011). Bias of the random forest out-of-bag (OOB) error for certain input parameters. *Open Journal of Statistics*, 2011. Scientific Research Publishing.

[19] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).

[20] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31.

[21] Dang, Y., Chen, Z., Li, H., & Shu, H. (2022). A comparative study of non-deep learning, deep learning, and ensemble learning methods for sunspot number prediction. *Applied Artificial Intelligence*, *36*(1), 2074129. Taylor & Francis.

Índice de figuras

Índice de tablas

Apéndice A

Comparativas datos históricos y predicciones del Sunspot Number

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1971-01-01	87	114.4608536
1971-02-01	125.3	103.6386871
1971-03-01	113.5	97.62378693
1971-04-01	89.6	99.90734863
1971-05-01	113.9	95.6233139
1971-06-01	124.7	93.29229736
1971-07-01	108.3	99.78559875
1971-08-01	108.9	95.00521088
1971-09-01	90.7	93.81839752
1971-10-01	86.9	91.89444733
1971-11-01	59.2	94.92037964
1971-12-01	64.3	100.2737656
1972-01-01	61.8	98.23640442

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1972-02-01	60.9	103.3682632
1972-03-01	65.4	101.7184753
1972-04-01	81.8	97.86019135
1972-05-01	60.3	102.4682999
1972-06-01	56.1	102.3063507
1972-07-01	33.2	100.2802429
1972-08-01	36.6	101.3730774
1972-09-01	84.1	87.17215729
1972-10-01	43.7	79.45545959
1972-11-01	34.3	73.61941528
1972-12-01	33.3	77.10705566
1973-01-01	39.4	72.29405975
1973-02-01	37.3	68.6317749
1973-03-01	30.9	65.26319885
1973-04-01	57.5	66.8995285
1973-05-01	56.3	60.92765808
1973-06-01	51.5	59.18487549
1973-07-01	79.1	52.28544998
1973-08-01	47.9	50.83883286
1973-09-01	57.2	48.77486038
1973-10-01	67.2	43.67482376
1973-11-01	35.9	41.91215897
1973-12-01	29.6	42.42544174
1974-01-01	27.3	41.74395752
1974-02-01	16.7	41.74591064
1974-03-01	16.9	41.91215897

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1974-04-01	7.7	46.11794281
1974-05-01	13.1	54.76681137
1974-06-01	16.7	53.24208832
1974-07-01	40.4	48.95010376
1974-08-01	56.7	45.71871567
1974-09-01	20.3	47.24343872
1974-10-01	13.6	49.80984879
1974-11-01	27.9	40.32962036
1974-12-01	11.6	38.03665924
1975-01-01	11.9	34.51799393
1975-02-01	6.4	34.21605301
1975-03-01	31.5	28.72057152
1975-04-01	27.3	26.16585922
1975-05-01	18.2	27.81078148
1975-06-01	17.9	24.72979736
1975-07-01	2.9	23.82211685
1975-08-01	24.1	27.37162209
1975-09-01	20.0	23.38511276
1975-10-01	29.7	23.38511276
1975-11-01	7.9	25.10532188
1975-12-01	22.3	27.81078148
1976-01-01	23.8	23.38511276
1976-02-01	33.3	19.48223114
1976-03-01	13.0	21.56414413
1976-04-01	19.0	21.271698
1976-05-01	27.0	20.88133812

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1976-06-01	54.9	20.88133812
1976-07-01	30.6	19.23642921
1976-08-01	43.0	23.57631493
1976-09-01	62.4	23.13931084
1976-10-01	62.1	21.56414413
1976-11-01	41.6	19.48223114
1976-12-01	61.4	23.13931084
1977-01-01	73.7	28.00197983
1977-02-01	132.6	28.24777985
1977-03-01	108.4	27.81078148
1977-04-01	141.2	34.34186935
1977-05-01	117.1	35.29716873
1977-06-01	134.6	41.78499985
1977-07-01	99.7	43.01089859
1977-08-01	82.4	45.99074173
1977-09-01	195.7	62.02480698
1977-10-01	177.1	65.07823944
1977-11-01	138.5	69.4672699
1977-12-01	173.9	79.68844604
1978-01-01	235.9	84.30425262
1978-02-01	214.7	95.45304871
1978-03-01	216.2	95.8817215
1978-04-01	234.0	102.3071671
1978-05-01	197.7	104.2321929
1978-06-01	185.5	104.4305649
1978-07-01	207.5	110.449234

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1978-08-01	267.8	113.1128922
1978-09-01	248.4	109.542778
1978-10-01	254.2	116.0068741
1978-11-01	210.0	113.8701477
1978-12-01	225.2	116.5955429
1979-01-01	235.3	122.6115189
1979-02-01	256.6	121.4241867
1979-03-01	245.3	121.620697
1979-04-01	220.7	127.6446152
1979-05-01	265.5	132.027832
1979-06-01	258.7	132.6135406
1979-07-01	240.0	138.4425507
1979-08-01	294.6	140.9016266
1979-09-01	279.2	145.0802002
1979-10-01	315.2	152.9927673
1979-11-01	286.4	153.9747314
1979-12-01	266.0	161.0655975
1980-01-01	340.0	163.738327
1980-02-01	302.7	163.5550995
1980-03-01	329.1	166.2260132
1980-04-01	267.3	169.3693695
1980-05-01	280.5	169.7499542
1980-06-01	298.3	176.2453766
1980-07-01	319.3	177.7839203
1980-08-01	372.2	178.9328766
1980-09-01	311.7	183.115036

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1980-10-01	365.7	193.128479
1980-11-01	380.8	194.0765839
1980-12-01	365.0	202.3379364
1981-01-01	149,8	183,2520294
1981-02-01	230,9	191,5648041
1981-03-01	221,1	191,5648041
1981-04-01	170,3	195,6667023
1981-05-01	119,3	190,4636841
1981-06-01	163,7	180,0606689
1981-07-01	139,4	191,117569
1981-08-01	161,9	191,7164764
1981-09-01	167,4	196,1349182
1981-10-01	134,3	187,4827881
1981-11-01	127,5	185,200882
1981-12-01	169,0	187,4827881
1982-01-01	115,5	182,8047943
1982-02-01	73,1	187,0394592
1982-03-01	88,7	176,1906128
1982-04-01	109,6	168,800827
1982-05-01	132,5	150,0567932
1982-06-01	131,5	157,9655151
1982-07-01	108,9	146,1442719
1982-08-01	96,0	151,9049835
1982-09-01	69,9	149,7450256
1982-10-01	72,5	137,6654053
1982-11-01	45,7	121,0082474

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (*t*+12)	Valor real: SSN (*t*+12)	Predicción: SSN (*t*+12)
1982-12-01	45,6	123,0092545
1983-01-01	74,8	109,590889
1983-02-01	110,2	103,3196335
1983-03-01	116,7	108,5752487
1983-04-01	90,4	103,1747513
1983-05-01	96,9	104,4601059
1983-06-01	65,1	105,7536697
1983-07-01	55,7	99,31706238
1983-08-01	35,0	90,90724182
1983-09-01	22,6	84,31034088
1983-10-01	12,6	88,9763031
1983-11-01	26,5	83,66938019
1983-12-01	21,4	77,46892548
1984-01-01	17,8	78,58462524
1984-02-01	20,7	85,88296509
1984-03-01	16,9	78,35535431
1984-04-01	20,4	70,63663483
1984-05-01	32,4	71,41855621
1984-06-01	28,3	60,90331268
1984-07-01	39,9	57,66605377
1984-08-01	10,1	47,84779358
1984-09-01	4,3	39,11177826
1984-10-01	22,0	36,7279892
1984-11-01	17,9	34,60643005
1984-12-01	15,8	27,69387817
1985-01-01	2,8	27,42290115

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1985-02-01	27,9	27,42290115
1985-03-01	13,8	22,99723244
1985-04-01	22,4	23,65608978
1985-05-01	16,1	24,09309387
1985-06-01	0,6	23,82211685
1985-07-01	18,1	23,82211685
1985-08-01	9,9	20,28308296
1985-09-01	5,1	21,49440193
1985-10-01	40,1	20,49345779
1985-11-01	15,4	20,49345779
1985-12-01	5,8	19,24261856
1986-01-01	9,8	14,81440353
1986-02-01	3,4	17,09315109
1986-03-01	17,4	16,38516998
1986-04-01	46,0	16,13984489
1986-05-01	39,1	18,87650681
1986-06-01	18,8	14,81440353
1986-07-01	38,2	16,38516998
1986-08-01	47,9	15,54110909
1986-09-01	42,2	14,74026108
1986-10-01	63,4	16,82217407
1986-11-01	48,8	18,87650681
1986-12-01	29,1	19,189785
1987-01-01	70,5	22,29525185
1987-02-01	45,4	21,49440193
1987-03-01	91,2	27,56497955

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1987-04-01	108,8	30,01044273
1987-05-01	74,2	29,15757179
1987-06-01	124,3	34,34186935
1987-07-01	131,4	38,55937195
1987-08-01	139,4	42,28546143
1987-09-01	142,7	45,99074173
1987-10-01	156,5	58,46646118
1987-11-01	156,8	55,05852127
1987-12-01	231,2	57,67385101
1988-01-01	210,1	67,44411469
1988-02-01	208,7	76,03752136
1988-03-01	170,4	86,10389709
1988-04-01	166,3	98,31207275
1988-05-01	195,4	96,72156525
1988-06-01	284,5	117,2534485
1988-07-01	180,5	125,4142914
1988-08-01	232,0	144,8538361
1988-09-01	225,1	148,3963623
1988-10-01	212,2	149,6985321
1988-11-01	238,2	162,4421387
1988-12-01	211,4	192,8692474
1989-01-01	227,4	194,5080261
1989-02-01	171,8	201,8547974
1989-03-01	191,7	187,8479462
1989-04-01	189,7	191,3581238
1989-05-01	175,2	201,7937927

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1989-06-01	153,3	211,5529938
1989-07-01	191,1	197,3876648
1989-08-01	252,1	202,846756
1989-09-01	169,1	205,504837
1989-10-01	199,4	206,2195587
1989-11-01	178,8	210,1115875
1989-12-01	197,1	195,980545
1990-01-01	195,3	202,7449036
1990-02-01	240,3	186,154068
1990-03-01	197,0	192,6665344
1990-04-01	197,6	192,2192993
1990-05-01	166,9	185,2137604
1990-06-01	224,7	182,8047943
1990-07-01	240,2	190,6742401
1990-08-01	240,8	196,1432495
1990-09-01	168,9	189,5015106
1990-10-01	197,1	192,2192993
1990-11-01	159,5	183,9813385
1990-12-01	212,6	191,117569
1991-01-01	198,3	190,4636841
1991-02-01	230,7	195,3512573
1991-03-01	151,0	191,117569
1991-04-01	142,2	191,117569
1991-05-01	94,3	183,9813385
1991-06-01	98,5	191,117569
1991-07-01	114,2	195,4819489

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (*t*+12)	Valor real: SSN (*t*+12)	Predicción: SSN (*t*+12)
1991-08-01	91,9	200,8714294
1991-09-01	94,0	183,9813385
1991-10-01	133,4	191,117569
1991-11-01	129,6	183,4631958
1991-12-01	122,0	184,4320221
1992-08-01	62,5	104,4266205
1992-09-01	31,2	103,2598495
1992-10-01	71,1	109,2664948
1992-11-01	48,2	107,1101227
1992-12-01	68,4	101,6544647
1993-01-01	84,9	96,96011353
1993-02-01	54,9	101,9290161
1993-03-01	47,5	100,2846832
1993-04-01	27,4	92,92399597
1993-05-01	29,8	79,7436142
1993-06-01	39,7	75,33463287
1993-07-01	50,6	78,43854523
1993-08-01	34,3	72,26751709
1993-09-01	40,5	60,98984909
1993-10-01	67,1	64,29901886
1993-11-01	29,5	59,67682266
1993-12-01	32,2	57,25451279
1994-01-01	32,6	58,29918671
1994-02-01	45,8	54,76681137
1994-03-01	46,3	52,52790833
1994-04-01	21,6	51,35211563

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1994-05-01	19,4	42,21829987
1994-06-01	22,5	41,74591064
1994-07-01	20,4	43,67482376
1994-08-01	18,2	38,60282898
1994-09-01	15,7	39,24341965
1994-10-01	30,6	44,91809845
1994-11-01	14	38,60282898
1994-12-01	14,9	37,4620285
1995-01-01	13,3	34,70106888
1995-02-01	7,7	36,41572571
1995-03-01	12,6	36,36909485
1995-04-01	6,8	28,72057152
1995-05-01	7,6	27,74930191
1995-06-01	16,5	27,74930191
1995-07-01	11,8	27,81078148
1995-08-01	19,7	23,38511276
1995-09-01	3	21,12714005
1995-10-01	0,7	19,92542458
1995-11-01	24,9	19,19597435
1995-12-01	14	16,45931244
1996-01-01	7,4	16,38516998
1996-02-01	11	14,74026108
1996-03-01	12,1	16,38516998
1996-04-01	23	14,49493504
1996-05-01	25,4	14,49493504
1996-06-01	20,8	16,13984489

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1996-07-01	12,9	16,13984489
1996-08-01	35,7	16,13984489
1996-09-01	59,7	14,49493504
1996-10-01	32,8	14,49493504
1996-11-01	50,4	16,57684898
1996-12-01	55,5	16,13984489
1997-01-01	44,5	14,49493504
1997-02-01	50,2	15,2957859
1997-03-01	82	20,76055145
1997-04-01	70,6	23,13931084
1997-05-01	74	28,00197983
1997-06-01	90,5	27,81078148
1997-07-01	96,7	28,72057152
1997-08-01	121,1	34,77887726
1997-09-01	132	45,59299088
1997-10-01	78,5	44,22807693
1997-11-01	97,3	45,99074173
1997-12-01	119,2	56,58324432
1998-01-01	86	59,43651581
1998-02-01	95,7	59,87237968
1998-03-01	104,8	61,70906939
1998-04-01	116,7	62,7455469
1998-05-01	122,8	69,83545886
1998-06-01	103,8	72,24849039
1998-07-01	114,7	79,65186077
1998-08-01	105,4	81,47907202

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
1998-09-01	113,2	84,31463646
1998-10-01	91,5	92,70878957
1998-11-01	108,8	95,6938751
1998-12-01	123,4	105,6866107
1999-01-01	110,4	113,8531942
1999-02-01	126,8	117,563497
1999-03-01	119,3	123,2815188
1999-04-01	132,6	134,3934681
1999-05-01	118,5	137,509747
1999-06-01	138,3	142,0927818
1999-07-01	147,8	152,3778668
1999-08-01	140,6	155,0930985
1999-09-01	139,5	155,4608274
1999-10-01	160,4	155,0930985
1999-11-01	162,2	168,4695045
1999-12-01	175,1	167,4396096
2000-01-01	161,5	176,2949654
2000-02-01	169,5	172,8144136
2000-03-01	175,8	174,2430663
2000-04-01	174,4	177,6551516
2000-05-01	191,5	181,2959891
2000-06-01	182,3	189,5334903
2000-07-01	183,1	198,7289684
2000-08-01	200,3	192,1915588
2000-09-01	207,7	203,4722743
2000-10-01	223,7	209,8390271

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (*t*+12)	Valor real: SSN (*t*+12)	Predicción: SSN (*t*+12)
2000-11-01	210,3	217,6890168
2000-12-01	233,2	220,4178251
2001-01-01	247,4	226,1164858
2001-02-01	239,5	229,4776791
2001-03-01	248,8	237,759396
2001-04-01	249,1	248,357681
2001-05-01	235,6	253,3760429
2001-06-01	240,6	265,515914
2001-07-01	242,8	278,623739
2001-08-01	232,7	285,2672616
2001-09-01	221,9	294,4816631
2001-10-01	223,7	303,0326368
2001-11-01	238,3	319,2776802
2001-12-01	241,2	330,8938452
2002-01-01	226,6	336,9644637
2002-02-01	229,2	340,2194735
2002-03-01	235,7	347,0146227
2002-04-01	228,1	356,7734419
2002-05-01	226,6	358,4575604
2002-06-01	212,8	357,7129897
2002-07-01	209,2	355,6792636
2002-08-01	227,1	348,4350234
2002-09-01	240,6	353,9059144
2002-10-01	247,3	348,8492451
2002-11-01	249,6	348,8281214
2002-12-01	237,5	344,6414841

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2003-01-01	236,8	336,0557771
2003-02-01	237,8	337,5824794
2003-03-01	244,4	344,9309261
2003-04-01	251,7	347,5580524
2003-05-01	256,5	346,2806828
2003-06-01	270,2	345,1552421
2003-07-01	259,6	348,7359278
2003-08-01	272,4	345,5596495
2003-09-01	287,6	351,3763127
2003-10-01	290,2	362,6732301
2003-11-01	307,1	377,0740208
2003-12-01	304,5	377,7016061
2004-01-01	285,5	368,0913706
2004-02-01	272,5	375,6037968
2004-03-01	284,8	372,0753796
2004-04-01	281,2	362,5218178
2004-05-01	272,5	359,7526923
2004-06-01	265,5	348,4911663
2004-07-01	281,7	341,4639405
2004-08-01	288,2	335,6016337
2004-09-01	293,8	334,2147857
2004-10-01	275,5	326,8053865
2004-11-01	268,3	326,1340525
2004-12-01	257,1	329,9695973
2005-01-01	261,1	340,5336933
2005-02-01	261,5	337,0299702

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (*t*+12)	Valor real: SSN (*t*+12)	Predicción: SSN (*t*+12)
2005-03-01	272,3	341,9190993
2005-04-01	286,3	343,7610522
2005-05-01	271,2	351,1044104
2005-06-01	272,3	357,2108257
2005-07-01	274,4	359,4426792
2005-08-01	283,1	356,3312286
2005-09-01	295,7	359,0088456
2005-10-01	293,1	355,1428305
2005-11-01	291,4	354,5933834
2005-12-01	302,6	358,167028
2006-01-01	298,5	355,3747732
2006-02-01	303,7	353,2448732
2006-03-01	311,1	358,5834875
2006-04-01	306,4	361,3063847
2006-05-01	317,4	362,5312825
2006-06-01	319,7	368,6390557
2006-07-01	314,9	374,0752368
2006-08-01	312,9	368,8128322
2006-09-01	316,7	367,4738147
2006-10-01	307,5	368,5809176
2006-11-01	302,2	371,7542178
2006-12-01	292,6	377,7736115
2007-01-01	294,3	382,293161
2007-02-01	290,4	385,8491465
2007-03-01	289,1	379,7947276
2007-04-01	282,7	386,2539931

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2007-05-01	270,1	379,8960497
2007-06-01	265,6	368,2802137
2007-07-01	257,8	369,5850058
2007-08-01	254,7	367,4315415
2007-09-01	249,6	356,1133508
2007-10-01	240,8	359,1319644
2007-11-01	252,6	362,7373912
2007-12-01	258,6	359,1963548
2008-01-01	257,2	348,1557907
2008-02-01	263,7	343,0249645
2008-03-01	269,4	339,8313255
2008-04-01	257,2	339,2186789
2008-05-01	248,4	336,6974705
2008-06-01	256,3	335,790493
2008-07-01	263,2	331,2190421
2008-08-01	256,2	329,9128523
2008-09-01	254,6	333,3049238
2008-10-01	261,1	327,0657433
2008-11-01	254,2	334,9798341
2008-12-01	250,6	335,1141241
2009-01-01	259,1	332,1520642
2009-02-01	268,1	327,3433191
2009-03-01	276,2	328,0767334
2009-04-01	270,2	336,7311581
2009-05-01	259,6	330,3764749
2009-06-01	260,4	332,7314536

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2009-07-01	266,3	334,5260857
2009-08-01	272,8	329,2266845
2009-09-01	270,4	334,9443556
2009-10-01	269,2	340,0227327
2009-11-01	274,3	348,7276278
2009-12-01	275,8	355,1965675
2010-01-01	280,7	358,7965816
2010-02-01	291,6	369,3324104
2010-03-01	290,5	374,1274798
2010-04-01	279,6	380,2059534
2010-05-01	282,4	377,564379
2010-06-01	276,5	381,5197397
2010-07-01	280,6	382,0657219
2010-08-01	278,5	373,2086504
2010-09-01	269,8	372,3744514
2010-10-01	278,9	364,5873094
2010-11-01	278,2	362,7888891
2010-12-01	283,7	358,6446011
2011-01-01	274,6	362,239663
2011-02-01	275,9	363,6466902
2011-03-01	283,3	368,7370907
2011-04-01	291,7	367,510598
2011-05-01	293,1	358,0473374
2011-06-01	294,1	359,8474791
2011-07-01	295,7	367,5463411
2011-08-01	274,7	335,7392157

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2011-09-01	270,5	334,2617115
2011-10-01	274,3	334,6917125
2011-11-01	275,5	340,1604818
2011-12-01	275,1	342,1391371
2012-01-01	276,6	343,8269644
2012-02-01	274,1	340,3792431
2012-03-01	270,4	340,6378401
2012-04-01	272,4	337,9134319
2012-05-01	268,4	334,2773866
2012-06-01	274,1	336,7026346
2012-07-01	275,2	338,0145599
2012-08-01	277,3	339,0938048
2012-09-01	281,8	339,2821917
2012-10-01	279,2	339,0700186
2012-11-01	279,8	339,1494577
2012-12-01	279,4	342,9086902
2013-01-01	280,8	340,9359141
2013-02-01	277,9	339,6120794
2013-03-01	270,7	340,3195482
2013-04-01	273,2	339,1918226
2013-05-01	271,6	339,2787926
2013-06-01	272,1	336,7169382
2013-07-01	276,2	337,3582427
2013-08-01	270,4	334,1390925
2013-09-01	266,4	336,2916914
2013-10-01	272,0	336,5201712

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (*t*+12)	Valor real: SSN (*t*+12)	Predicción: SSN (*t*+12)
2013-11-01	274,5	334,0867080
2013-12-01	274,2	336,2177401
2014-01-01	278,4	338,7781128
2014-02-01	277,0	339,1870953
2014-03-01	278,0	339,6609893
2014-04-01	279,3	338,9792268
2014-05-01	282,0	336,6844536
2014-06-01	281,1	339,0388007
2014-07-01	277,4	337,5530460
2014-08-01	277,5	335,7700788
2014-09-01	270,4	332,4540480
2014-10-01	264,0	330,2345908
2014-11-01	268,5	328,6833377
2014-12-01	265,3	329,6286717
2015-01-01	275,2	331,0138065
2015-02-01	277,0	331,3500358
2015-03-01	283,4	334,3002546
2015-04-01	278,2	334,9272610
2015-05-01	281,0	335,3309443
2015-06-01	278,2	335,0448447
2015-07-01	275,1	334,1591320
2015-08-01	279,3	335,0740928
2015-09-01	280,1	334,8920772
2015-10-01	283,3	332,4980668
2015-11-01	285,7	333,5845110
2015-12-01	277,1	333,2566163

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2016-01-01	275,4	333,5857987
2016-02-01	276,2	333,6797642
2016-03-01	277,0	335,7957366
2016-04-01	272,0	338,0317084
2016-05-01	276,0	335,8767577
2016-06-01	272,0	335,9384623
2016-07-01	270,0	334,1747245
2016-08-01	271,7	332,0709807
2016-09-01	269,7	331,4528130
2016-10-01	274,5	331,3528396
2016-11-01	275,7	331,2906590
2016-12-01	270,5	329,9452281
2017-01-01	266,8	328,3067412
2017-02-01	266,0	328,6338002
2017-03-01	265,4	328,2072263
2017-04-01	271,7	327,8768380
2017-05-01	270,1	327,4905682
2017-06-01	268,4	326,9636364
2017-07-01	265,4	324,6306041
2017-08-01	266,0	324,6493740
2017-09-01	261,6	322,8738808
2017-10-01	265,0	321,4760925
2017-11-01	264,7	323,3088230
2017-12-01	266,5	326,7321386
2018-01-01	264,4	326,6714156
2018-02-01	267,5	327,1718056

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (*t*+12)	Valor real: SSN (*t*+12)	Predicción: SSN (*t*+12)
2018-03-01	272,7	326,9168445
2018-04-01	266,5	328,8321486
2018-05-01	267,7	331,2165394
2018-06-01	265,4	334,0369021
2018-07-01	260,9	331,9343438
2018-08-01	266,7	331,1482476
2018-09-01	266,3	334,6577317
2018-10-01	265,1	336,9496994
2018-11-01	274,0	337,1817011
2018-12-01	272,4	340,5275207
2019-01-01	276,0	340,1338741
2019-02-01	277,2	337,8554910
2019-03-01	278,3	339,6098471
2019-04-01	282,3	340,3258261
2019-05-01	278,1	338,9396712
2019-06-01	275,1	339,0887174
2019-07-01	277,6	339,7579528
2019-08-01	279,8	340,4761314
2019-09-01	275,7	339,2384510
2019-10-01	274,1	337,3477446
2019-11-01	275,2	337,3618481
2019-12-01	275,1	335,6540892
2020-01-01	278,3	334,8118720
2020-02-01	278,7	332,9075610
2020-03-01	274,6	331,7784516
2020-04-01	277,8	332,4696280

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2020-05-01	275,5	334,3366787
2020-06-01	274,6	334,7651433
2020-07-01	272,0	331,3227070
2020-08-01	270,2	329,9028206
2020-09-01	273,2	331,7280370
2020-10-01	275,1	332,9563021
2020-11-01	274,2	331,4383804
2020-12-01	274,7	331,5608485
2021-01-01	273,2	334,3707941
2021-02-01	274,1	336,2788702
2021-03-01	274,7	334,3191447
2021-04-01	272,7	336,5420815
2021-05-01	271,3	336,6641182
2021-06-01	269,6	339,4849020
2021-07-01	274,6	338,0479051
2021-08-01	275,4	338,9152062
2021-09-01	276,1	337,8068342
2021-10-01	274,7	337,4947261
2021-11-01	275,6	339,3297215
2021-12-01	274,6	340,3588322
2022-01-01	274,4	340,4765336
2022-02-01	274,6	337,0163140
2022-03-01	272,2	336,5711465
2022-04-01	271,2	334,2748687
2022-05-01	270,5	331,5875665
2022-06-01	270,8	331,5833882

Tabla A.1: Comparativa valores reales y predicciones del número de manchas solares obtenido con el modelo XGBoost realizado

Date (t+12)	Valor real: SSN (t+12)	Predicción: SSN (t+12)
2022-07-01	267,4	328,6402185
2022-08-01	270,6	329,8499314
2022-09-01	271,4	331,0791883
2022-10-01	269,5	328,2931190
2022-11-01	270,5	326,9203326
2022-12-01	269,1	327,1917244
2023-01-01	270,2	328,0788277
2023-02-01	269,0	327,7407967
2023-03-01	270,2	328,2212810
2023-04-01	270,2	328,1060652
2023-05-01	270,3	328,7649606
2023-06-01	269,8	329,5503298
2023-07-01	269,4	329,0761845
2023-08-01	269,7	329,4371534
2023-09-01	270,5	329,9557897
2023-10-01	270,5	329,6153228
2023-11-01	274,3	318,0891558
2023-12-01	277,1	319,4534417

Apéndice B

Código del programa

```
Importación de librerías
import pandas as pd  # para cargar el Excel
import numpy as np  # para operaciones con vectores
import matplotlib.pyplot as plt  # para gráficos
import matplotlib.dates as mdates  # para formatear y manejar las fechas en los
gráficos
from scipy.stats import zscore, skew, kurtosis
import seaborn as sns  # para gráficos
import xgboost as xgb
from xgboost import XGBRegressor
from sklearn.model_selection import TimeSeriesSplit, RandomizedSearchCV
from itertools import cycle  # Para crear iteradores cíclicos
from sklearn.metrics import mean_squared_error, mean_absolute_error
from statsmodels.tsa.api import ExponentialSmoothing
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
Definición de funciones
# Se genera un vector cíclico, para la estacionalidad del ciclo solar:
def generate_vector(length, n_to_repeat):
values = list(range(1, n_to_repeat+1))
vector = []
cycle_iterator = cycle(values)
for _ in range(length):
vector.append(next(cycle_iterator))
return vector

# Se crean los títulos del DataFrame, mostrando la relación temporal (`t-n`, `t`,
`t+n`):
def Extended_titles(Cn,n_in,n_out):
Total_titles=[]
Tl=n_in+n_out+1
for i in range(Tl-1):
if i<(n_in-1):
letter_to_add = " (t-" + str(n_in-i-1) +")"
elif i==(n_in-1):
letter_to_add = " (t)"
else:
letter_to_add = " (t+" + str(i-n_in+1) +")"
Cn_aux= [word + letter_to_add for word in Cn]
Total_titles=Total_titles+Cn_aux
return(Total_titles)
```

```
# Se convierte la serie temporal en el formato adecuado para el aprendizaje automático:
def series_to_supervised(data, n_in, n_out, dropnan=True):
n_vars = 1 if type(data) is list else data.shape[1]
df = pd.DataFrame(data)
cols = list()
for i in range(n_in, 0, -1):
cols.append(df.shift(i))
for i in range(0, n_out):
cols.append(df.shift(-i))
agg = pd.concat(cols, axis=1)
if dropnan:
agg.dropna(inplace=True)
return agg.values

# Se transforman los datos de la serie temporal en un formato adecuado para el aprendizaje supervisado
def to_supervided2(yiel, n_in, n_out):
values = yiel.values
data = series_to_supervised(values, n_in, n_out)
Cn=list(yiel.columns.values)
Cn_total=Extended_titles(Cn, n_in, n_out)
Data = pd.DataFrame(data, columns=Cn_total)
return Data
Carga de los datos
Df = pd.read_json("C:/Users/Letia/Desktop/TFG_2024/PYTHON/observed-solar-cycle-indices.json")
Cálculo de las estadísticas descriptivas
#Media
mean_values = Df[['ssn','smoothed_ssn']].mean()
print("Media:\n", mean_values)

#Desviación estandar
std_dev = Df[['ssn','smoothed_ssn']].std()
print("Desviación Estándar:\n", std_dev)

#Varianza
variance = Df[['ssn','smoothed_ssn']].var()
print("Varianza:\n", variance)

#Sesgo
skewness = Df[['ssn','smoothed_ssn']].apply(lambda x: skew(x.dropna()))
print("Sesgo (Skewness):\n", skewness)
```

```
#Curtosis: Mide la forma de la distribución, especialmente de las colas
kurtosis_vals = Df[['ssn','smoothed_ssn']].apply(lambda x: kurtosis(x.dropna()))
print("Curtosis (Kurtosis):\n", kurtosis_vals)

#Cuartiles
quartiles = Df[['ssn','smoothed_ssn']].quantile([0.25, 0.5, 0.75])
print("Cuartiles:\n", quartiles)

#Rango intercuartílico
iqr = quartiles.loc[0.75] - quartiles.loc[0.25]
print("Rango Intercuartílico (IQR):\n", iqr)

#Rango
range_values = Df[['ssn','smoothed_ssn']].apply(lambda x: x.max() - x.min())
print("Rango:\n", range_values)
Visualizaciones iniciales
# Histograma del número de manchas solares
plt.figure(figsize=(12, 6))
plt.hist(Df['ssn'], bins=30, edgecolor='k', alpha=0.7)
plt.xlabel('Número de Manchas Solares')
plt.ylabel('Frecuencia')
plt.title('Distribución del Número de Manchas Solares')
plt.grid(True)
plt.show()

# Gráfico de dispersión
Df['Date'] = pd.to_datetime(Df['time-tag'])
Df = Df.sort_values(by='Date')

# Cálculo de la Media Móvil y el Z-Score
Df['SMA'] = Df['ssn'].rolling(window=12, center=True).mean()
Df['Z-Score'] = zscore(Df['ssn'])

# Visualización
plt.figure(figsize=(12, 8))
plt.scatter(Df['Date'], Df['ssn'], color='blue', alpha=0.6, label='Número de
Manchas Solares')
plt.plot(Df['Date'], Df['SMA'], color='red', label='Media Móvil (12 meses)')
plt.scatter(Df['Date'], Df['ssn'], c=abs(Df['Z-Score']), cmap='coolwarm',
label='Z-Score', alpha=0.6, edgecolors='k')
plt.gca().xaxis.set_major_locator(mdates.YearLocator(11))
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y'))
plt.xticks(rotation=45)
```

```
                plt.xlabel('Fecha')
                plt.ylabel('Número de Manchas Solares')
                plt.title('Número de Manchas Solares a lo Largo del Tiempo')
                plt.colorbar(label='Z-Score')
                plt.legend()
                plt.grid(True)
                plt.tight_layout()
                plt.show()

                # Matriz de correlación
                Df_numeric = Df.select_dtypes(include=[np.number])
                Df_numeric = Df_numeric.fillna(Df_numeric.mean())
                correlation_matrix = Df_numeric.corr()
                print(correlation_matrix)

                plt.figure(figsize=(10, 8))
                sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1,
vmax=1)
                plt.title('Matriz de Correlación')
                plt.show()
                Comparación por periodos mediante matrices de correlación
                # Se ajustan los formatos de las fechas y nos quedamos con las columnas que
queremos
                Df['Date'] = pd.to_datetime(Df['Date'], errors='coerce')
                Df.set_index('Date', inplace=True)

                # Se asegura de que sólo las columnas numéricas se utilicen para la correlación
                Df_numeric = Df.select_dtypes(include=[np.number])
                Df_numeric = Df_numeric.fillna(Df_numeric.mean())

                # Se definen los periodos para dividir la serie temporal
                periodos = pd.date_range(start=Df_numeric.index.min(),
end=Df_numeric.index.max(), freq='11Y')
                correlation_matrices = {}

                for i in range(len(periodos)-1):
                start_date = periodos[i]
                end_date = periodos[i+1]
                Df_periodo = Df_numeric[start_date:end_date]
                if not Df_periodo.empty:
                corr_matrix = Df_periodo.corr()
                correlation_matrices[f"{start_date.year}-{end_date.year}"] = corr_matrix
                plt.figure(figsize=(10, 8))
                sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
```

```
            plt.title(f"Matriz de Correlación {start_date.year}-{end_date.year}")
            plt.show()

            Df = Df.rename(columns={'ssn': 'Sunspot_Number', 'smoothed_ssn':
'Smoothed_Sunspot_Number'})
            Df['Date'] = pd.to_datetime(Df['time-tag'])
            Df.insert(0, 'Date', Df.pop('Date'))
            Df = Df[["Date", "Sunspot_Number", "Smoothed_Sunspot_Number"]]
            Df.replace(-1, np.nan, inplace=True)
            Visualización inicial de los datos: Sunspot Number vs Smoothed Sunspot
Number
            fig, ax = plt.subplots(figsize=(15, 6))
            ax.plot(Df['Date'], Df['Sunspot_Number'], color='blue', linestyle='-', linewidth=0.75,
label='Sunspot Number')
            ax.plot(Df['Date'], Df['Smoothed_Sunspot_Number'], color='red', linestyle='-',
linewidth=1, label='Smoothed Sunspot Number')
            ax.set_title('Número de manchas solares a lo largo del tiempo')
            ax.set_xlabel('Fecha')
            ax.set_ylabel('Número de manchas solares')
            plt.xticks(rotation=45)
            ax.grid(True)
            ax.legend()
            plt.show()
            Preparación de datos para el modelado
            # Transformación de los datos para el aprendizaje supervisado
            n_in=24 # número de observaciones anteriores utilizadas para predecir
            n_out=12 # número de observaciones anteriores utilizadas a predecir

            Df=to_supervided2(Df,n_in,n_out) #Df listo para el aprendizaje supervisado
            Df = Df.dropna(subset=[f'Date (t-{n_in-1})']) #Se eliminan las filas que no nos
valen
            Df.set_index('Date (t)', inplace=True)


            # Se revisa el rango de fechas después de la transformación
            print("Rango de fechas en el DataFrame supervisado:")
            print(Df.index.min(), "a", Df.index.max())


            n_to_repeat=11*12 # 11 años y 12 meses tiene cada ciclo aproximadamente
            v_sta = generate_vector(len(Df), n_to_repeat) #se genera el vector a repetir

            Df.insert(0, 'Index_for_seasonality', v_sta)
            Separación entre "target" y "features"
```

```
Div_index=Df.columns.get_loc("Smoothed_Sunspot_Number (t)")

features=Df.iloc[:,:Div_index+1].columns.to_list()
features=[element for element in features if "Date" not in element]

targets=Df.iloc[:,Div_index:].columns.to_list()
targets=[element for element in targets if "Sunspot_Number" in element]
targets=[element for element in targets if "Smoothed" not in element]


X=Df[features]
y=Df[targets]

X = X.apply(pd.to_numeric, errors='coerce')
y = y.apply(pd.to_numeric, errors='coerce')


Cut_Date=pd.Timestamp(1971,1,1)

X_train=X[X.index<Cut_Date]
y_train=y[y.index<Cut_Date]


X_test=X[X.index>=Cut_Date]
y_test=y[y.index>=Cut_Date]
Validación cruzada y entrenamiento del modelo XGBoost
#Se inicializa el modelo XGBoost
XGB_model = XGBRegressor()

# Se definen los hiperparámetros a buscar en RandomizedSearchCV
param_grid = {
        'learning_rate': [0.001, 0.01],    # Tasa de aprendizaje
        'max_depth': [3, 5],               # Profundidad máxima del árbol
        'min_child_weight': [1, 3],        # Peso mínimo de un nodo hijo
        'n_estimators': [100, 300],        # Número de árboles (estimadores)
        'lambda': [0.1, 1],               # Regularización L2
        'gamma': [0, 0.1]                  # Reducción mínima de pérdida
}

# Se define el esquema de validación cruzada
tscv = TimeSeriesSplit(n_splits=3)

# Configuración de RandomizedSearchCV con la validación cruzada
random_search = RandomizedSearchCV(
```

```
        estimator=XGB_model,
        param_distributions=param_grid,
        n_iter=50,                    # Número de combinaciones a probar
        cv=tscv,                      # Validación cruzada con TimeSeriesSplit
        scoring="neg_mean_squared_error",     # Métrica de evaluación
        random_state=42,
        n_jobs=-1,
        verbose=2
        )

        # Se buscan de los mejores hiperparámetros
        random_result = random_search.fit(X_train, y_train)

        # Se obtienen los mejores hiperparámetros
        best_params = random_result.best_params_

        # Se entrena el modelo con los mejores parámetros
        XGB_model = XGBRegressor(**best_params)
        XGB_model.fit(X_train, y_train)
        Predicción del Sunspot Number y posterior comparación datos históricos
        # Se realizan las predicciones
        y_test_pred = XGB_model.predict(X_test)

        # Se convierte y_test_pred en un DataFrame con la fecha y la predicción

        # La primera columna de y_test_pred es la predicción para 'Sunspot Number'
        sunspot_number_pred = y_test_pred[:, 0]  # Se selecciona la primera columna

        # X_test tiene el índice de fechas que se quiere usar
        prediction_dates = X_test.index + pd.DateOffset(months=12) # Se asume que el
índice de X_test es la fecha

        #El índice de X_test es la fecha real (t), no (t+12)
        prediction_dates = X_test.index  # Las fechas no se desplazan, ya que ya
corresponden a t+12 en X_test

        # Se crea un DataFrame con las fechas y las predicciones
        y_test_pred_Df = pd.DataFrame({
                'Date (t+12)': prediction_dates,
                'Sunspot_Number (t+12)_pred': sunspot_number_pred
        })

        # Se crea la figura y los ejes
        plt.figure(figsize=(12, 6))
```

```
                # Se grafican los valores reales usando la fecha
                plt.plot(y_test.index, y_test['Sunspot_Number (t+12)'], label='Valor Real',
color='blue', marker='o')

                # Se grafican las predicciones usando la fecha
                plt.plot(y_test_pred_Df['Date (t+12)'], y_test_pred_Df['Sunspot_Number
(t+12)_pred'], label='Predicción', color='red', marker='o')

                # Se agrega el título y las etiquetas a los ejes
                plt.title('Comparación de Valores Reales y Predicciones de Manchas Solares')
                plt.xlabel('Fecha')
                plt.ylabel('Número de Manchas Solares')

                #Se muestra la leyenda
                plt.legend()

                #Se muestra la gráfica
                plt.grid(True)
                plt.show()

                # Esto supone que la primera predicción corresponde a la primera fecha en
y_test_pred_Df
                y_test_pred_Df.index = y_test.index

                #Se crea un DataFrame combinando los valores reales y las predicciones
                comparison_df = pd.DataFrame({
                        'Date (t+12)': y_test.index,
                        'Sunspot_Number (t+12)_Real': y_test['Sunspot_Number (t+12)'],
                        'Sunspot_Number (t+12)_Pred': y_test_pred_Df['Sunspot_Number
(t+12)_pred']
                })

                #Se define la ruta donde se quiere guardar el archivo
                file_path = 'comparacion_predicciones.xlsx'

                #Se exporta el DataFrame a un archivo Excel
                comparison_df.to_excel(file_path, index=False)
                Evaluación del rendimiento del modelo
                #TÉCNICAS DE EVALUACION DEL RENDIMIENTO DEL MODELO

                #Error Absoluto Medio

                # Paso 1: Se calcula el MAE del modelo
```

```
        mae = mean_absolute_error(y_test['Sunspot_Number (t+12)'],
y_test_pred_Df['Sunspot_Number (t+12)_pred'])

        # Paso 2: Se calcula la media de Sunspot_Number (t+12) en el conjunto de
entrenamiento
        mean_sunspot_number = y_train['Sunspot_Number (t+12)'].mean()

        # Paso 3: Se crean predicciones del baseline utilizando la media calculada
        baseline_predictions = [mean_sunspot_number] * len(y_test)

        # Paso 4: Se calcula el MAE del baseline
        baseline_mae = mean_absolute_error(y_test['Sunspot_Number (t+12)'],
baseline_predictions)

        # Se compara con el MAE del modelo
        print(f'MAE del modelo: {mae}')
        print(f'MAE del baseline: {baseline_mae}')

        # Se calcula la mejora porcentual del modelo sobre el baseline
        improvement = (baseline_mae - mae) / baseline_mae
        print(f'Mejoría del modelo sobre el baseline: {improvement:.2%}')

        #Error Cuadrático Medio

        # Paso 1: Se calcula el MSE del modelo
        mse_model = mean_squared_error(y_test['Sunspot_Number (t+12)'],
y_test_pred_Df['Sunspot_Number (t+12)_pred'])

        # Paso 2: Se calcula la media de la variable objetivo en el conjunto de prueba
        mean_sunspot_number = y_test['Sunspot_Number (t+12)'].mean()

        # Paso 3: Se crea predicciones del baseline utilizando la media calculada
        baseline_predictions = [mean_sunspot_number] * len(y_test)

        # Paso 4: Se calcula el MSE del baseline
        mse_baseline = mean_squared_error(y_test['Sunspot_Number (t+12)'],
baseline_predictions)

        # Se compara con el MSE del modelo
        print(f'MSE del modelo: {mse_model}')
        print(f'MSE del baseline: {mse_baseline}')

        # Se calcula la mejora porcentual del modelo sobre el baseline
        improvement = (mse_baseline - mse_model) / mse_baseline
```

```
        print(f'Mejoría del modelo sobre el baseline: {improvement:.2%}')

        #Raiz del Error Cuadrático Medio

        # Paso 1: Se calcula el RMSE del modelo
        rmse = mean_squared_error(y_test['Sunspot_Number (t+12)'],
y_test_pred_Df['Sunspot_Number (t+12)_pred'], squared=False)

        # Paso 2: Se calcula el RMSE del baseline
        baseline_predictions = [y_test['Sunspot_Number (t+12)'].mean()] * len(y_test)
        rmse_baseline = mean_squared_error(y_test['Sunspot_Number (t+12)'],
baseline_predictions, squared=False)

        # Se compara con el MSE del modelo
        print(f'RMSE del modelo: {rmse}')
        print(f'RMSE del baseline: {rmse_baseline}')

        #Se calcula el rango de valores
        range_y_test = y_test['Sunspot_Number (t+12)'].max() - y_test['Sunspot_Number
(t+12)'].min()
        print(f'Rango de los valores: {range_y_test}')


        #CREACIÓN TABLA ESTADÍSTICAS DESCRIPTIVAS

        #Se crea un DataFrame con los resultados
        stats_Df = pd.DataFrame({
                'Estadística': [
                'Media',
                'Desviación Estándar',
                'Varianza',
                'Sesgo (Skewness)',
                'Curtosis (Kurtosis)',
                'Cuartil Q1',
                'Mediana (Q2)',
                'Cuartil Q3',
                'Rango Intercuartílico (IQR)',
                'Rango'
                ],
                'Valor': [
                mean_values,
                std_dev,
                variance,
                skewness,
```

```
            kurtosis_vals,
            quartiles.loc[0.25],  # Cuartil Q1
            quartiles.loc[0.5],   # Mediana (Q2)
            quartiles.loc[0.75],  # Cuartil Q3
            iqr,
            range_values
            ]
})

# Se convierte el diccionario a un DataFrame
stats_Df = pd.DataFrame(stats_Df)

# Se convierte el diccionario a un DataFrame
stats_Df = pd.DataFrame(stats_Df)

# Se guarda el DataFrame en un archivo Excel
file_path = 'estadisticas_descriptivas.xlsx'
stats_Df.to_excel(file_path, index=False)
Comparativas modelo XGBoost con otros modelos
# Se vuelven a cargar los datos
Df_comparativa = pd.read_json('observed-solar-cycle-indices.json')
Df_comparativa = Df_comparativa.sort_values('time-tag')

# Se define el tamaño del conjunto de prueba
test_size = int(len(Df_comparativa) * 0.2)  # 20% para prueba

# Se calculan los índices
train_indices = list(range(len(Df_comparativa) - test_size))
test_indices = list(range(len(Df_comparativa) - test_size, len(Df_comparativa)))

# Se dividen los datos en conjuntos de entrenamiento y prueba
train_data = Df_comparativa.iloc[train_indices]
test_data = Df_comparativa.iloc[test_indices]

# Se define el tamaño de la ventana para modelos comparativos
window_size = 12

# Se define y_test (valores reales de prueba)
y_test = test_data['ssn']

# MODELO DE LA MEDIA MÓVIL
y_train_ma = Df_comparativa['ssn'].iloc[train_indices]
y_test_ma = Df_comparativa['ssn'].iloc[test_indices]
```

```
                # MEDIA MÓVIL SIMPLE
                y_pred_ma =
y_test_ma.rolling(window=window_size).mean().shift(-window_size+1)
                y_pred_ma = y_pred_ma.dropna()

                # Se ajusta y_test_ma para que coincida con la longitud de y_pred_ma
                y_test_ma = y_test_ma.loc[y_pred_ma.index]
                mse_ma = mean_squared_error(y_test_ma, y_pred_ma)
                rmse_ma = mean_squared_error(y_test_ma, y_pred_ma, squared=False)
                print(f'MSE del modelo de Media Móvil: {mse_ma}')
                print(f'RMSE del modelo de Media Móvil: {rmse_ma}')

                # MODELO DE PROMEDIO EXPONENCIAL
                y_train_es = Df_comparativa['ssn'].iloc[train_indices]
                y_test_es = Df_comparativa['ssn'].iloc[test_indices]

                # Ajuste del modelo Exponential Smoothing
                es_model = ExponentialSmoothing(y_train_es, trend='add', seasonal='add',
seasonal_periods=window_size)
                es_fit = es_model.fit()
                y_pred_es = es_fit.forecast(len(y_test_es))

                # Se asegura que y_test_es coincida con la longitud de y_pred_es
                y_test_es = y_test_es.loc[y_pred_es.index]
                mse_es = mean_squared_error(y_test_es, y_pred_es)
                rmse_es = mean_squared_error(y_test_es, y_pred_es, squared=False)
                print(f'MSE del modelo de Promedio Exponencial: {mse_es}')
                print(f'RMSE del modelo de Promedio Exponencial: {rmse_es}')

                print(f'MSE del modelo XGBoost: {mse}')
                print(f'RMSE del modelo XGBoost: {rmse}')
```