PONTIFICIA UNIVERSIDAD CATÓLICA DEL

ECUADOR FACULTAD DE INGENIERÍA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE MAGISTER EN SISTEMAS DE LA INFORMACIÓN MENCIÓN DATA SCIENCE

MODELO DE MINERÍA DE DATOS PARA EL ANÁLISIS DE VARIABLES AMBIENTALES Y SUS TENDENCIAS, TOMADAS DE SENSORES EN DISTINTAS ÉPOCAS DEL AÑO EN LA ESTACIÓN METEOROLÓGICA DE LA ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

Autor: Ing. José Fernando Esparza Parra, Mgs.

Director: Ph.D. Rafael Melgarejo Heredia.

Quito, D.M., Septiembre – 2024

TABLA DE CONTENIDOS

Contenido

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE IMÁGENES

1. CAPÍTULO 1. INTRODUCCIÓN

1.1. Antecedentes

El análisis de variables ambientales a través de la minería de datos ha ganado una creciente relevancia en la investigación científica y la toma de decisiones. Numerosos estudios han demostrado la eficacia de estas técnicas en la interpretación de datos ambientales para comprender patrones y tendencias. En este contexto, se evidencia la necesidad de aplicar estas metodologías a la problemática específica de la Estación meteorológica de la Escuela Superior Politécnica de Chimborazo.

Investigaciones recientes resaltan el papel clave de la minería de datos en ciencias ambientales, según Wang et al. (2020), la minería de datos ha demostrado ser esencial para analizar datos ambientales complejos, identificar patrones y predecir tendencias climáticas, pero con respecto a proyectos realizados en el ámbito específico de estaciones experimentales, como la de la ESPOCH, al revisar investigaciones como la de García et al. (2018), se logra comprender la importancia de utilizar técnicas de minería de datos para maximizar la utilidad de los datos recopilados, mejorando la capacidad predictiva y optimizando la gestión ambiental.

Para Modelos Predictivos usando variables ambientales, la investigación de Li et al. (2019) dan el punto de partida de casos de éxito en el uso de modelos predictivos utilizando técnicas avanzadas de minería de datos, mismo que han demostrado y aportado significativamente en el pronóstico de cambios climáticos, proporcionando una base sólida para la toma de decisiones en la agricultura y la gestión ambiental.

En el contexto latinoamericano, estudios como el de Gómez et al. (2021) subrayan la relevancia de aplicar técnicas de minería de datos para abordar desafíos ambientales específicos de la región, estas experiencias destacan la necesidad de adaptar las metodologías a entornos locales para lograr resultados efectivos.

1.2. Justificación

La investigación propuesta sobre la minería de datos para el análisis de variables ambientales se pretende realizar en la Estación meteorológica de la Escuela Superior Politécnica de Chimborazo, está ubicada en la zona centro del Ecuador, en la provincia de Chimborazo, tiene un clima que oscila entre los 5 y 20 grados centígrados, dependiendo la época del año. El proyecto se fundamenta en la necesidad de comprender y utilizar de manera efectiva la información recopilada de sensores instalados desde hace varios años y que están tomando muestras constantes, esta información en la actualidad es usada de manera independiente, por lo que se ha notado que no tiene dirección o manera de relacionar las distintas variables y muchos menos de identificar si tienen relación directa o indirecta con eventos que se suceden en la zona, donde su población es mayoritariamente agrícola y ganadera.

Bajo ese contexto y teniendo en cuenta que el uso de sensores ha aumentado significativamente en estos años y han proporcionado grandes cantidades de datos detallados, se vuelve necesario aprovechar esta tecnología y los dichos datos aplicando herramientas avanzadas de minería de datos para extraer información depurada y poder generar conocimientos valiosos en beneficio de la

comunidad agrícola y científica. Por otro lado, la identificación y comprensión de las tendencias climáticas son la piedra angular para la toma de decisiones en diversas áreas, especialmente en la agricultura, ya que los agricultores y científicos dependen de información precisa sobre variables ambientales para planificar estrategias agrícolas y anticipar posibles desafíos relacionados con el clima.

La Estación meteorológica, al ser un sitio estratégico en la región, presenta características climáticas particulares, el poder comprender las tendencias ambientales en este contexto no solo beneficiará a los agricultores locales, sino que también contribuirá al conocimiento científico sobre la variabilidad climática en la zona.

Finalmente se puede indicar que, la investigación propuesta no solo intenta encontrar resultados que posiblemente ayuden a mejorar la productividad agrícola, sino que también se alinea con objetivos de desarrollo sostenible al promover prácticas agrícolas más eficientes y resistentes a condiciones climáticas cambiantes.

1.3. Planteamiento del problema

La Estación meteorológica, ubicada en la Escuela Superior Politécnica de Chimborazo, ha sido objeto de un extenso monitoreo ambiental mediante sensores a lo largo de diversas épocas del año, estos dispositivos han recopilado un amplio conjunto de datos sobre variables ambientales, proporcionando una valiosa fuente de información, que no es muy útil si no se cuenta con enfoques analíticos avanzados que permita extraer conocimientos significativos y sacar provecho de esta rica fuente de datos.

A pesar de la disponibilidad de la información, la falta de un modelo de datos analítico, consistente y robusto, ha limitado la capacidad de la Estación Experimental, de manera que se pueda aprovechar plenamente la información recopilada y debido a la ausencia de un enfoque sistemático basado en técnicas de minería de datos ha impedido identificar patrones, tendencias y relaciones clave entre las variables ambientales registradas, por lo que, esta carencia de análisis representa un obstáculo significativo para la toma de decisiones informada en sectores agrícolas y científicos de la región.

En este contexto, se evidencia la necesidad urgente de desarrollar un modelo de minería de datos que permita analizar de manera efectiva las variables ambientales recopiladas por los sensores de dicha Estación Experimental, al contar con este modelo, no solo facilitaría la comprensión de las tendencias climáticas y fenómenos ambientales en la región, sino que también proporcionaría herramientas valiosas para la toma de decisiones estratégicas en actividades agrícolas y proyectos de investigación.

La presente investigación se propone abordar esta problemática, desarrollando un modelo de minería de datos que contribuya a desentrañar los patrones climáticos y tendencias ambientales en la Estación meteorológica, y de esta manera, se busca proporcionar a agricultores, científicos y tomadores de decisiones locales una herramienta efectiva para optimizar la gestión ambiental y fortalecer la resiliencia frente a condiciones climáticas cambiantes.

1.4. Objetivos

1.4.1. Objetivo General

Desarrollar un modelo de datos, utilizando técnicas de minería de datos para el análisis de variables ambientales y sus tendencias, con datos tomados de sensores, en la Estación Meteorológica de la Escuela Superior Politécnica de Chimborazo

1.4.2. Objetivos Específicos

- Recopilar, organizar y limpiar los datos los datos provenientes de sensores instalados en la Estación meteorológica.
- Aplicar técnicas de aprendizaje supervisado de clasificación, que identifique patrones, tendencias y relaciones entre las variables ambientales almacenadas.
- Evaluar la precisión de los resultados obtenidos en la aplicación modelos de aprendizaje supervisado a través de métricas de rendimiento.

1.5. Alcance

El estudio se centrará en el análisis de datos meteorológicos históricos, recopilados entre 1990 y 2023 en la Estación Meteorológica de la Escuela Superior Politécnica de Chimborazo (ESPOCH), con el objetivo de desarrollar un modelo de minería de datos que permita identificar patrones, tendencias y relaciones significativas entre variables ambientales como la humedad, temperatura, heliofanía, precipitación, velocidad del viento, tensión de vapor y punto de rocío.

Para el proceso de investigación, inicia con la recopilación, verificación y limpieza de datos para asegurar su integridad y precisión, posteriormente, se realizará un análisis exploratorio de los datos para identificar características clave como estacionalidad y variabilidad, seguido por la aplicación de técnicas de minería de datos para modelar las interacciones entre las variables y descubrir patrones significativos, con estos resultados, se desarrollará un modelo supervisado predictivo utilizando la metodología CRIPS-DM.

Con los resultados se pretende proporcionar un marco robusto para el análisis de tendencias climáticas basadas en datos históricos de sensores de la ESPOCH.

2. CAPITULO 2. REVISIÓN DE LA LITERATURA

2.1. Variables ambientales

Humedad

La humedad es un factor crítico en la agricultura, ya que afecta directamente el crecimiento y la productividad de los cultivos, tanto en el aire y en el suelo desempeña un papel esencial en la evapotranspiración de las plantas, el proceso mediante el cual el agua se evapora del suelo y se transpira a través de las plantas, según Peixoto y Oort (1992), "la humedad relativa en la atmósfera es un factor determinante en la regulación del ciclo del agua y en la distribución de la precipitación" (p. 137), el poder medir y controlar la humedad son fundamentales para ayudar a mejorar las condiciones de crecimiento de los cultivos y maximizar los rendimientos agrícolas, pues en condiciones de alta humedad, las plantas pueden absorber agua fácilmente, lo que favorece su crecimiento, pero es importante tener en cuenta que niveles en niveles excesivos pueden propiciar la aparición de enfermedades fúngicas y bacterianas que pueden afectar los cultivos.

Por otro lado, niveles bajos de humedad pueden causar estrés hídrico en las plantas, reduciendo su capacidad para llevar a cabo la fotosíntesis y otros procesos vitales, como o señala Wallace y Hobbs (2006), quien indica que "la humedad relativa es determinante en la agricultura, ya que afecta la evapotranspiración, mismo que un proceso vital para el transporte de nutrientes y el enfriamiento de las plantas", lo que confirma su gran importancia.

Podemos realizar la medición precisa de la humedad con higrómetros y sensores de humedad del suelo.

- Higrómetros: Utilizados para medir la humedad relativa del aire, pueden ser digitales, mecánicos, de cabello o capacitivos, los higrómetros digitales son los más utilizados en agricultura, debido a su precisión y capacidad para proporcionar lecturas en tiempo real.
- Sensores de Humedad del Suelo: Miden el contenido de agua en el suelo, proporcionando datos importantes para la gestión del riego, los tensiómetros y sensores capacitivos de humedad del suelo son comunes en aplicaciones agrícolas.

Humedad Relativa Media

Peixoto y Oort (1992), menciona que la humedad relativa es decisiva para regular la transferencia de energía y vapor de agua entre la superficie de la Tierra y la atmósfera, es el promedio de la cantidad de humedad atmosférica durante un período específico, se mide en porcentajes y es un parámetro esencial en meteorología, la climatología y la agricultura, en fundamental en el crecimiento de plantas y la estabilidad de materiales, en términos de salud humana, una humedad relativa adecuada puede prevenir problemas respiratorios y de la piel, mientras que en la agricultura, la humedad relativa afecta la evapotranspiración de las plantas y, por lo tanto, su desarrollo y productividad.

Temperatura

La temperatura es una variable climática fundamental que afecta todos los aspectos del crecimiento y desarrollo de las plantas, es una medida que determina cuán caliente o frío está algo y su unidad de medida puede estar dada en en grados Celsius (°C) o Fahrenheit (°F). Peixoto y Oort (1992), menciona que "la temperatura es un factor clave que regula los procesos fisiológicos en las plantas, incluyendo la fotosíntesis y la respiración" (p. 175), lo que indica que las plantas deben tener rangos óptimos de temperatura para su crecimiento y desarrollo, las temperaturas que se encuentran fuera de estos rangos, pueden causar estrés térmico, reduciendo el rendimiento y la calidad de los cultivos.

Las temperaturas extremas, tanto altas como bajas pueden ser perjudiciales, por un lado que las temperaturas altas pueden acelerar la evapotranspiración, aumentando la demanda de agua de las plantas y provocando estrés hídrico, mientras que las temperaturas bajas pueden inhibir la germinación de semillas, retrasar el crecimiento y, en casos extremos, causar daños por heladas. Wallace y Hobbs (2006) afirman que "las temperaturas extremas pueden tener efectos devastadores en los cultivos, desde el daño directo a las plantas hasta la reducción de la eficiencia en el uso del agua" (p. 93).

Al igual que la humedad, existen varios instrumentos que pueden medir la temperatura, entre los que tenemos:

- Termómetros de Mercurio y Alcohol: Utilizados tradicionalmente en la medición de la temperatura, aunque su uso ha disminuido debido restricciones del uso del mercurio por afectaciones ambientales.
- Termómetros Digitales: Utilizan sensores electrónicos para medir la temperatura con alta precisión, son comunes en estaciones meteorológicas, invernaderos y otras instalaciones agrícolas.
- Sensores de Temperatura de Resistencia (RTD): Basan su funcionamiento en el cambio de resistencia eléctrica de un material, generalmente platino, con la temperatura; estos suelen ser bastante precisos y se usan en aplicaciones industriales y científicas.

Heliofanía

La heliofanía se refiere a la cantidad de horas de sol directo que recibe una superficie en un período específico, este parámetro es de suma importancia en estudios de energía solar, agricultura y climatología, considerado que la cantidad de luz solar influye en la fotosíntesis de las plantas, el diseño de edificios y la generación de energías limpias, según el WMO (2010), "la heliofanía es un factor clave en la evaluación del potencial solar de una región y en la planificación agrícola" (p. 250).

Precipitación

La precipitación es la cantidad de agua, en forma de lluvia, nieve o granizo que cae en la superficie, en un período específico de tiempo, se considera a la precipitación como fundamental para el ciclo del agua, la agricultura y la gestión de recursos hídricos, la cantidad y distribución de la precipitación determinan la disponibilidad de agua para el riego, la recarga de acuíferos y el suministro de agua potable.

Velocidad del Viento

La velocidad del viento es la medida de la rapidez con la que el aire se desplaza en una dirección específica, se mide metros por segundo, con este parámetro se puede evaluar la erosión del suelo, la dispersión de contaminantes y la generación de energía eólica, la velocidad del viento también influye en la sensación térmica, por otro lado como menciona Jones y Moberg (2003), "la velocidad del viento es un factor determinante en la evaluación del potencial eólico de una región" (p. 230), lo que indica que con esta medición se puede determinará zonas potencialmente útiles para sacar provecho de energías limpias, a través de viento.

2.2. Sensores de Medición de Variables Ambientales

Higrómetros

Los higrómetros son dispositivos que miden la humedad relativa del aire, esta medida es útil en diversas aplicaciones, incluyendo la meteorología, la climatología, la agricultura y la gestión del ordenamiento territorial.

Tipos de Higrómetros

- Higrómetros Digitales, que se utilizan sensores electrónicos que miden la capacitancia o la resistencia eléctrica para determinar la humedad relativa, son reconocidos con de alta precisión y facilidad de uso.
- Higrómetros Mecánicos, funcionan mediante materiales que cambian de forma o tamaño, dependiendo de la cantidad de humedad.
- Higrómetros de Cabello, estos utilizan un cabello humano o animal que cambia de longitud con la humedad, son bastante sensibles y proporcionan una respuesta rápida.
- Higrómetros Capacitivos, miden la humedad relativa basándose en los cambios en la capacitancia de un material dieléctrico cuando absorbe vapor de agua.

Termómetros

Los termómetros miden la temperatura del aire, del suelo, de persona y/o animales, en el caso de la temperatura del aire y el suelo, son un parámetro fundamental para evaluar las condiciones climáticas, esto permite recolectar información para el estudio de fenómenos meteorológicos.

Tipos de Termómetros

- Termómetros de Mercurio, utilizan la expansión y contracción del mercurio en un tubo de vidrio para medir la temperatura. Aunque son muy precisos, su uso ha disminuido debido a preocupaciones ambientales y de salud asociadas con el mercurio. Se utilizaban ampliamente en aplicaciones médicas y meteorológicas.
- Termómetros de Alcohol, funcionan de manera similar a los de mercurio, pero utilizan alcohol coloreado. Son más seguros y se emplean en

aplicaciones de campo y laboratorios donde la toxicidad del mercurio es una preocupación.

- Termómetros Digitales, usan sensores electrónicos como termistores o RTD (Detectores de Temperatura de Resistencia) para medir la temperatura con alta precisión. Son comunes en hogares, estaciones meteorológicas y laboratorios.
- Termómetros de Resistencia (RTD), basan su funcionamiento en la variación de la resistencia eléctrica de un metal con la temperatura, este metal usualmente es el platino.

Heliógrafos

Los heliógrafos miden la duración del sol directo, lo cual es esencial para estudios relacionados con la energía solar, la agricultura y la climatología. La cantidad de luz solar directa influye en la fotosíntesis de las plantas, el diseño de edificios solares y la generación de energía.

Tipos de Heliógrafos

- Heliógrafos Clásicos, estos utilizan papel fotográfico para registrar la duración de la luz solar. Estos dispositivos son útiles para registros históricos y análisis de tendencias a largo plazo.
- Heliógrafos Digitales, emplean sensores de radiación solar para medir la duración del sol directo de manera continua y precisa. Son ideales para estudios de energía solar y aplicaciones agrícolas, proporcionando datos en tiempo real.

Pluviómetros

Los pluviómetros miden la cantidad de precipitación en forma de lluvia, nieve, granizo, sin esta medición, no se podría tener estudios del ciclo hidrológico.

Tipos de Pluviómetros

- Pluviómetros de Balancín, utilizan un balancín que se llena y vacía alternativamente para medir la cantidad de precipitación.
- Pluviómetros de Pesaje, miden el peso del agua acumulada en un recipiente, de esta manera normalmente tiene una alta precisión,

- Pluviómetros de Cubo Basculante, utilizan un cubo que se llena hasta un cierto nivel y luego bascula para vaciarse, midiendo la precipitación acumulada.

Anemómetros

Los anemómetros miden la velocidad del viento, un parámetro importante para evaluar la erosión del suelo, la dispersión de contaminantes y la generación de energía eólica.

Tipos de Anemómetros

- Anemómetros de Cazoletas, se componen de cazoletas montadas en un eje vertical que giran con el viento.
- Anemómetros de Hélice, este tipo de anemómetros utilizan una hélice que gira con el viento.
- Anemómetros Sónicos, son bastante precisos y hacen uso de ondas ultrasónicas para medir la velocidad del viento sin partes móviles.
- Anemómetros Láser (LIDAR), usan tecnología láser para medir la velocidad del viento a diferentes alturas.

2.3. Ciencia de Datos

En la actualidad la ciencia de datos, es un eje transversal y multidisciplinario, que utiliza métodos, procesos, algoritmos y sistemas científicos para extraer conocimiento y discernimientos de datos estructurados y no estructurados, combina técnicas de estadística, informática y gestión de datos para analizar y comprender fenómenos complejos.

Multidisciplinariedad de la Ciencia de Datos

La ciencia de datos en estos días es multidisciplinaria, puede cubrir varios campos tanto en el sector público y privado, en diferentes aristas como salud, educación, industria, etc.

Salud: en el campo de la salud, es usado para analizar gran cantidad de información de pacientes, ayudando a identificar patrones y tendencias que pueden mejorar la

atención médica, por ejemplo, en la predicción de enfermedades, optimización de tratamientos personalizados y análisis de imágenes médicas.

Según Dhar (2013), "la ciencia de datos tiene el potencial de transformar el sector de la salud al proporcionar herramientas avanzadas para el análisis de datos, lo que permite una atención médica más precisa y eficiente" (p. 80).

Finanzas: Para Provost y Fawcett (2013), "la aplicación de técnicas de ciencia de datos en las finanzas permite una toma de decisiones más informada y una gestión de riesgos más efectiva", sustentando que la ciencia de datos se emplea para modelar riesgos, detectar fraudes y optimizar carteras de inversión, un caso se puede evidenciar en un análisis de riesgos crediticios, detección de transacciones fraudulentas y modelado predictivo de precios de activos.

Marketing y Ventas: ayuda a las empresas a comprender mejor a sus clientes y a optimizar las estrategias de marketing, esto se puede notar cuando haces una segmentación de mercado, análisis de sentimiento en redes sociales y optimización de campañas publicitarias, Davenport y Harris (2007) destacan que "las empresas que utilizan ciencia de datos para analizar el comportamiento del consumidor pueden diseñar campañas de marketing más efectivas y personalizadas" (p. 50).

Industria: la ciencia de datos se aplica en la mejora de procesos de manufactura, mantenimiento predictivo y optimización de cadenas de suministro, según Waller y Fawcett (2013), la integración de la ciencia de datos es este campo, ayuda a aumentar significativamente la eficiencia operativa y reducir costos, al mismo tiempo de reducir costos, un ejemplo clásico se puede evidenciar en la monitorización de maquinaria, predicción de fallos y gestión eficiente de inventarios.

Gobierno y Políticas Públicas: se utiliza para analizar grandes conjuntos de datos gubernamentales y mejorar la toma de decisiones políticas, como lo menciona Gandomi y Haider (2015) que indica que la ciencia de datos dará un impulso a la eficiencia de las políticas públicas, ya que proporciona análisis detallados y fundamentados en datos con respecto a las necesidades y comportamientos de la población.

Utilidad de la Ciencia de Datos

La ciencia de datos proporciona numerosas utilidades en diferentes sectores:

Toma de Decisiones Basada en Datos, ayuda a las organizaciones a tomar decisiones informadas y basadas en datos, lo que puede mejorar la eficiencia operativa y la efectividad estratégica, según McAfee y Brynjolfsson (2012), "las organizaciones que basan sus decisiones en datos tienen un rendimiento significativamente mejor que aquellas que no lo hacen" (p. 62).

Innovación y Desarrollo de Productos, Davenport y Patil (2012) textualmente dice "la ciencia de datos facilita la innovación al permitir a las empresas identificar y capitalizar oportunidades emergentes en el mercado" (p. 73), lo que se puede interpretar que la ciencia de datos permite la creación de nuevos productos y servicios basados en el análisis de necesidades y tendencias del mercado.

Mejora de la Experiencia del Cliente, Al hablar de experiencia de Kumar et al. (2013), indica que cuando se analiza los datos del cliente, estos proporcionan insights valiosos para mejorar la experiencia del cliente, en otras palabras, ayuda a personalizar y mejorar la experiencia del cliente mediante el análisis de datos de comportamiento y preferencias

2.4. Machine Learning

El aprendizaje automático, es parte de la inteligencia artificial que comprende de algoritmos y modelos de datos, estos que permiten a las máquinas aprender y mejorar automáticamente a partir de la experiencia sin que hayan sido creadas específicamente para realizar dichas tareas, usando el análisis de datos, estos modelos pueden identificar patrones y hacer predicciones o tomar decisiones basadas en nuevos datos.

Uno de los aspectos clave del machine learning es su capacidad para manejar grandes volúmenes de datos y extraer información útil de ellos, para satisfacer una variedad de campos como la medicina, finanzas, marketing, manufactura, educación, entre otros, según Mitchell (1997), "un programa de computadora se dice que aprende de la experiencia E en relación con una clase de tareas T y un valor de rendimiento P, si su rendimiento en tareas de T, medida por P, mejora con la experiencia E" (p. 2), lo que indica que este enfoque basado en la experiencia, es lo que destaca a machine learning de los métodos tradicionales de programación.

Machine learning se divide generalmente en tres categorías principales:

Aprendizaje supervisado, donde el modelo es entrenado con un conjunto de datos etiquetados, es decir, datos en los que las respuestas correctas son conocidas, y el objetivo es aprender a predecir la etiqueta de nuevos datos

- Modelos de Clasificación, son a aquellos que tienen por objetivo predecir una categoría o clase a la que pertenece una instancia, entre los mas importantes están:
    - Regresión Logística: Utilizado para problemas de clasificación binaria (Hosmer, Lemeshow & Sturdivant, 2013).
    - Máquinas de Soporte Vectorial (SVM): Usado para clasificación tanto binaria como multiclase (Cortes & Vapnik, 1995).
    - Árboles de Decisión: Modelos basados en la partición del espacio de características en subespacios más simples (Quinlan, 1986).
    - K-Nearest Neighbors (K-NN): Clasifica nuevas instancias según la mayoría de sus vecinos más cercanos en el espacio de características (Cover & Hart, 1967).
- Modelos de regresión, esta enfocados a predecir valores continuos, aquí podemos encontrar
    - Regresión Lineal: Modelo simple para predecir un valor continuo en función de una o más variables independientes (Montgomery, Peck & Vining, 2012).
    - Regresión Ridge y Lasso: Variantes de la regresión lineal que incluyen regularización para prevenir el sobreajuste (Tibshirani, 1996).
    - Redes Neuronales: Modelos flexibles capaces de captar relaciones no lineales (Goodfellow, Bengio & Courville, 2016).
- Modelos de Series Temporales, estos modelos son un subcampo del aprendizaje supervisado que se enfocan en predecir valores futuros basados en datos secuenciales, considerando la dependencia temporal, los más comunes son:
    - ARIMA (AutoRegressive Integrated Moving Average) es un modelo estadístico utilizado para predecir series temporales mono variantes.
    - Modelos SARIMA (Seasonal ARIMA), según Jenkins y Reinsel (2008) son una extensión de los modelos ARIMA que incorporan componentes estacionales para analizar y predecir series temporales con patrones recurrentes, es decir, se usan en datos que muestran

variaciones estacionales y permiten capturar tanto las dependencias temporales regulares como las estacionales.

- Las Redes Neuronales Recurrentes son una red neuronal, esta diseñada para procesar secuencias de datos y son usadas para tareas de procesamiento de lenguaje natural, la predicción de series temporales y el reconocimiento de patrones en datos secuenciales, según Goodfellow, Bengio y Courville (2016), "las RNN son fundamentales para modelar datos secuenciales porque permiten que la red mantenga un estado interno que capta la dependencia temporal en las secuencias de entrada" (p. 367), en este tipo de modelos, se incluye a LSTM (Long Short-Term Memory), que se usan en modelos con datos de largo plazo.
- Prophet es un modelo de predicción de series temporales desarrollado por Facebook, según Taylor y Letham (2018), su ventaja radica en manejar datos con fuertes patrones estacionales y con la capacidad de adaptarse a cambios repentinos en las tendencias.

Aprendizaje no supervisado, estos modelos intentan identificar patrones o estructuras en datos no etiquetados, como la agrupación de datos en clústeres.

- Clustering, este tipo de modelos agrupan datos en clústeres o grupos, que compartan características similares entre los mas comunes tenemos
  - K-Means: Agrupa los datos en K clústeres minimizando la variación dentro de cada clúster (MacQueen, 1967).
  - Clustering Jerárquico: Agrupa los datos en una jerarquía de clústeres (Johnson, 1967).
  - DBSCAN: Agrupa datos en función de la densidad, permitiendo la identificación de clústeres de forma arbitraria (Ester et al., 1996).

- Reducción de Dimensionalidad, al usar estos modelos, se pretende reducir el número de variables bajo consideración, mientras se preserva la mayor cantidad posible de información relevante, entre estos podemos encontrar:
  - Análisis de Componentes Principales (PCA): Transforma los datos a un nuevo espacio de menor dimensión, maximizando la variabilidad capturada (Pearson, 1901).
  - Análisis de Correspondencias: Similar a PCA, pero para datos categóricos (Greenacre, 1984).

- Autoencoders: Redes neuronales que aprenden una representación más compacta de los datos (Hinton & Salakhutdinov, 2006).

Aprendizaje por refuerzo a diferencia de los anteriores, el modelo aplicado aprende a tomar decisiones secuenciales a través de la interacción con un entorno, recibiendo recompensas o castigos según la calidad de sus decisiones (Russell & Norvig, 2021), a continuación, se muestra una clasificación

- Q-Learning: Algoritmo básico de aprendizaje por refuerzo que busca aprender una política óptima a través de la actualización de una función de valor de acción (Watkins & Dayan, 1992).
- Deep Q-Networks (DQN): Combina Q-learning con redes neuronales profundas para manejar entornos de alta dimensionalidad (Mnih et al., 2015).
- Política Gradiente: se enfocan en mejorar directamente la estrategia o "política" que el robot o sistema de machine learning sigue para tomar decisiones en un entorno (Sutton et al., 2000).
- Actor-Critico: Combina las aproximaciones de política gradiente y de función de valor para mejorar la estabilidad y eficiencia del aprendizaje (Konda & Tsitsiklis, 2000).

2.5. Análisis Exploratorio de Datos (EDA)

Análisis Exploratorio de Datos es un enfoque fundamental en la ciencia de datos que se utiliza para analizar y resumir las principales características de un conjunto de datos, generalmente antes de aplicar cualquier modelo de machine learning o análisis estadístico más complejo, fue popularizado por el estadístico John W. Tukey en su libro *Exploratory Data Analysis* publicado en 1977, donde se enfatiza la importancia de comprender los datos, antes de proceder con modelado que se desea aplicar.

(Tukey, 1977) en su libro denominado *Exploratory Data Analysis*, afirmó que "el objetivo del análisis exploratorio de datos es maximizar nuestra comprensión de los datos antes de realizar cualquier tipo de modelado formal", lo que indica la importancia del EDA para los científicos de datos en el proceso de análisis inicial, pues permite obtener una visión clara y precisa de los datos antes de realizar inferencias o construir modelos predictivos, en consecuencia, es un paso que no puede faltar en cualquier análisis de datos, porque proporciona las bases sobre las

que se pueden construir modelos predictivos más precisos y útiles, los pasos que se deben seguir son:

1. Identificación de Patrones y Relaciones: este paso permite detectar la correlación entre las variables del dataframe o base de datos que posiblemente al inicio de la investigación o estudio no hayan sido evidentes
2. Detección de Valores Atípicos y Anomalías: como se sabe en el mundo real es muy común que al ingresar datos pudieran existir errores de digitación que eventualmente perjudiquen los resultados de los modelos a evaluarse, por ello se vuelve importante poder detectarlo a tiempo y de ser el caso eliminarlos, estos casos se logran identificar con gráficos de diagramas de caja o gráficos de dispersión, mismos que nos permiten observar visualmente ver donde se encuentran.
3. Evaluación de Supuestos Estadísticos: es el proceso de verificar si los datos cumplen con los supuestos necesarios para aplicar correctamente un modelo estadístico, como la normalidad, homocedasticidad o independencia de errores, según Wilcox (2012), "la evaluación de los supuestos estadísticos es fundamental para garantizar que las inferencias basadas en los modelos sean confiables y no se vean comprometidas por violaciones de los supuestos" (p. 5), es por ello que este paso es necesario para garantizar la validez de los resultados y evitar interpretaciones erróneas en los modelos que se vayan a aplicar.
4. Reducción de Dimensionalidad y Transformación de Datos: son técnicas utilizadas para simplificar un conjunto de datos, logran este efecto al disminuir el número de variables o dimensiones, conservando la mayor cantidad posible de información relevante, lo que facilita el análisis y mejora el rendimiento de los modelos, especialmente cuando se trabaja con datos de alta dimensionalidad, Jolliffe (2002) indica que "la reducción de dimensionalidad es una herramienta esencial en la estadística multivariante, ya que permite simplificar los datos mientras se retiene la estructura subyacente más importante" (p. 1), este paso puede ser aplicado dependiendo de la data, puedo no ser necesario en algunos casos.

Herramientas y Técnicas para EDA

Para este proceso, se pueden combinar el uso de varias herramientas estadísticas y de visualización que permite proporcionar una visión completa de los datos

Estadísticas Descriptivas, entre las más comunes y efectivas son la media, mediana, moda, desviación estándar, percentiles, y rango intercuartílico para resumir la distribución de los datos.

Visualizaciones Gráficas, Según Cleveland (1993), "las visualizaciones gráficas son herramientas esenciales en el análisis de datos, permitiendo a los investigadores captar rápidamente información clave y comunicar resultados de manera efectiva" (p. 4), lo que nos indica la relevancia de estos al momento de encontrar o determinar el comportamiento de los datos, aquí podemos encontrar una variedad de gráficos, entre los más eficientes a la hora de análisis exploratorio tenemos:

- Histogramas: Utilizados para mostrar la distribución de una variable numérica.
- Gráficos de Caja (Box Plots): Para identificar valores atípicos y visualizar la dispersión de los datos.
- Gráficos de Dispersión (Scatter Plots): Para explorar relaciones entre dos variables numéricas.
- Matriz de Correlación: nos permite ver las relaciones entre múltiples variables numéricas, aquí se evidencia en qué medida las variables están correlacionadas, esta correlación puede ser positiva o negativa.

Análisis Multivariante, si se analiza lo que mencionan Hair, Black, Babin, y Anderson (2010), donde indican que "el análisis multivariante permite a los investigadores explorar la estructura de relaciones complejas entre variables, proporcionando una comprensión más profunda y completa de los datos" (p. 2), se puede añadir que el análisis multivariante es un conjunto de técnicas estadísticas utilizadas para analizar datos que involucran múltiples variables al mismo tiempo, que ayudarán a comprender las relaciones y patrones entre ellas, como herramientas mas comunes tenemos a los gráficos de Pares y el análisis de Componentes Principales (PCA).

Tratamiento de Valores Faltantes, es el proceso mediante el cual se manejan los datos incompletos en un conjunto de datos para evitar sesgos y mejorar la precisión de los análisis posteriores, según Little y Rubin (2002), "el manejo adecuado de los valores faltantes es fundamental para obtener resultados válidos, ya que la presencia de datos incompletos puede distorsionar los resultados " (p. 1), de ahí

radica la importancia de un buen tratamiento y manejo de la información faltante para que los modelos tengan la efectividad deseada.

Es importante considera que al rellenar valores faltantes debe emplear una técnica correcta, pues no todos los datos pueden o deben ser rellenados, considerando que, si se lo hace de manera incorrecta, se puede desviar los resultados de los modelos a emplearse

Tabla 1: Manejo de valores nulos

Imputación	**Porcentaje**	**Acción**	**Técnica**
Simple	Hasta 5l 5%	Rellenar valores	media, mediana o moda
Avanzada	Entre el 5% y el 30%	Rellenar valores	KNN (K-Nearest Neighbors) o regresión
Eliminar columnas	Mas del 30%	Eliminar valores	Eliminar columnas o variables completas

Elaboración: Propia

2.6. CRISP-DM

Cross-Industry Standard Process for Data Mining, diseñado la IBM, es un modelo de proceso ampliamente utilizado para la minería de datos, proporcionando un marco estructurado para abordar proyectos de ciencia de datos, según Wirth y Hipp (2000), CRISP-DM es "un modelo de proceso independiente de la industria y de la herramienta que se utiliza para guiar el desarrollo de proyectos de minería de datos, desde la comprensión del negocio hasta la implementación de los resultados" (p. 1), este enfoque muestra a científicos de datos un camino bien definido, que puede lograr aumentar la eficiencia y la efectividad en la gestión de proyectos complejos.

Complementando esta visión, Chapman et al. (2000) describen CRISP-DM como "un proceso iterativo que consta de seis fases, la primera es la comprensión del negocio, después viene en análisis de los datos, luego la preparación de los datos, el modelado, la evaluación y finalmente el despliegue, diseñado para ser flexible y adaptable a diferentes tipos de proyectos y dominios" (p. 14), esta naturaleza iterativa y flexible del modelo permite ajustes y ajustes continuos a medida que se avanza en el proyecto, asegurando los resultados finales.

FASES DE LA METODOLOGÍA CRISP-DM

El modelo CRIPS-DM tiene seis fases iterativas que permiten que un proyecto tenga un enfoque sistemático y flexible que ayude a resolver problemas complejos.

Primera fase, conocida como Comprensión del Negocio, donde se identifican los objetivos y los requisitos del proyecto desde una perspectiva comercial o empresarial, según Wirth y Hipp (2000), esta fase "establece el fundamento del proyecto al asegurar que los objetivos del negocio estén claramente definidos y comprendidos antes de proceder con el análisis de datos" (p. 3), de modo que en esta etapa es fundamental alinear el proyecto con las necesidades y expectativas del negocio, para garantizar que los resultados finales sean útiles y relevantes.

La **segunda fase** es la Comprensión de los Datos, implica recolectar, describir y explorar los datos disponibles para familiarizarse con ellos, de acuerdo con lo que indican Chapman et al. (2000) donde señalan que "esta fase es esencial para identificar problemas potenciales en los datos, como valores faltantes o inconsistencias, que deben ser abordados antes del modelado" (p. 15), afirmando que la exploración detallada de los datos ayuda a establecer una buena base para el modelado posterior.

Tercera fase o Preparación de los Datos, esta fase incluye la limpieza y transformación de los datos para hacerlos adecuados para el análisis, según Shearer (2000) describe esta etapa como "un proceso iterativo en el que los datos se transforman y seleccionan para preparar conjuntos de datos que alimentarán el modelado" (p. 9), es decir, aquí es donde se determina qué variables y datos serán utilizados en los modelos predictivos, esta fase se considera la base del modelado, porque la calidad de los datos alimentados en un modelo afecta directamente la calidad de las predicciones que el modelo puede generar, según Shearer (2000), "aproximadamente el 70% del tiempo en un proyecto de minería de datos se dedica a la preparación de los datos, lo que determina su importancia en el éxito general del proyecto" (p. 9).

La preparación de datos incluye varias actividades relevantes que deben cumplirse, inicialmente tenemos la selección de los datos, aquí se decide qué subconjunto de datos se utilizará en el modelado, se puede incluso llegar a eliminar aquellos que no son necesarios o rellenar algunos de ser el caso, además, se corrigen posibles errores y se pueden eliminar los valores atípicos, en esta fase también se lleva a cabo la transformación de los datos, esto implica, modificar los datos para que sean compatibles con los algoritmos de modelado que se van a utilizar, aquí generalmente se utiliza la normalización o estandarización de las variables numéricas, la codificación de variables categóricas, o la creación de nuevas variables derivadas que puedan mejorar el rendimiento del modelo.

Cuarta fase o fase de Modelado en CRISP-DM es donde se construyen, ajustan, depuran y optimizan los modelos predictivos que permitirán obtener insights valiosos a partir de los datos, en esta fase, se seleccionan las técnicas de modelado más adecuadas, se entrenan los modelos con los datos preparados, y se realizan ajustes iterativos para mejorar el rendimiento del modelo, según Wirth y Hipp (2000), "el modelado implica la aplicación de algoritmos matemáticos y estadísticos a los datos con el fin de identificar patrones o relaciones que pueden ser utilizados para hacer predicciones o clasificaciones" (p. 4), de manera que esta fase no solo consiste en elegir un modelo, sino también en probar diferentes algoritmos para determinar cuál se adapta mejor a las características del conjunto de datos y al problema específico.

Durante el modelado, los datos preparados se dividen en dos conjuntos, uno de entrenamiento y otro de validación, donde el conjunto de entrenamiento como su nombre lo indica, es utilizado para entrenar el modelo, es decir, para ajustar los parámetros del modelo con el fin de que aprenda a realizar predicciones basadas en los datos y con estos resultados, el modelo se evalúa utilizando el conjunto de validación para comprobar su capacidad de generalización a nuevos datos, este proceso de validación es esencial para evitar el sobreajuste, donde un modelo podría funcionar bien con los datos de entrenamiento pero no generalizar correctamente a datos no vistos.

En esta fase también es importante considerar cambios de modelos de ser necesario, hearer (2000) explica que "es común probar múltiples algoritmos de modelado, como árboles de decisión, redes neuronales, máquinas de soporte vectorial, entre otros, para comparar sus rendimientos y seleccionar el que ofrezca la mejor precisión, interpretabilidad y eficiencia" (p. 9), además, en esta fase se puede realizar la sintonización de hiperparámetros, este es un proceso mediante el cual se ajustan los parámetros clave del modelo para mejorar su rendimiento.

Una vez que se ha seleccionado un modelo y se ha optimizado, se evalúa su desempeño utilizando métricas específicas que varían según el tipo de problema, por ejemplo, precisión, recall, F1-score para clasificación, o error cuadrático medio para regresión, lo que convierte a esta fase como dinámica, debido a que puede requerir múltiples iteraciones y ajustes, tanto en el modelo como en los datos, hasta que se alcance un rendimiento satisfactorio.

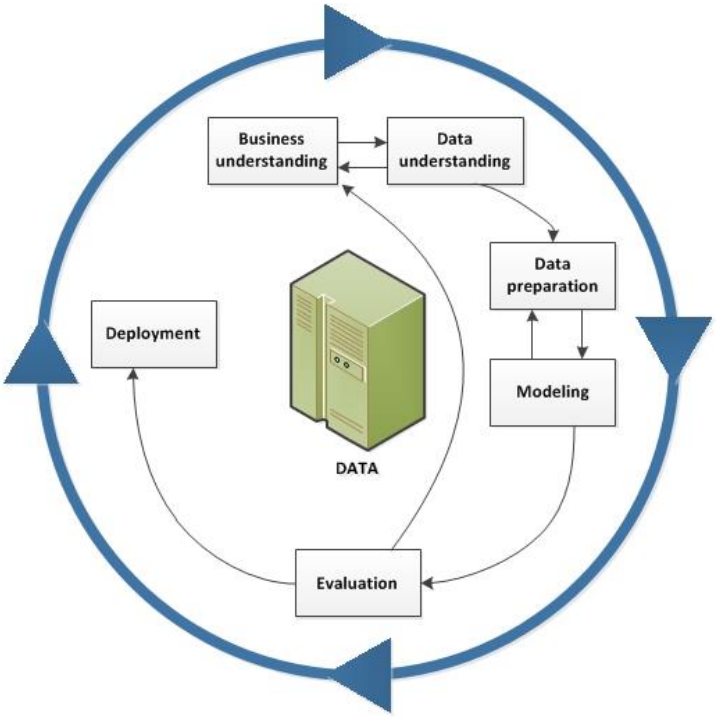
Quinta fase, en esta ser realiza la evaluación del modelado para asegurar que los resultados cumplen con los objetivos establecidos al principio del proyecto, tomando

en cuenta lo que indica Chapman et al. (2000), "en esta fase, se evalúan los modelos para determinar su eficacia y se revisa si realmente se alinean con los objetivos comerciales identificados" (p. 17), durante esta fase los modelos se someten a una serie de pruebas que verifican su desempeño en términos de precisión, generalización y capacidad para manejar nuevos datos, esto generalmente implica comparar el modelo con datos de prueba que no fueron utilizados durante la fase de modelado, según Wirth y Hipp (2000), "una evaluación adecuada implica no solo medir la precisión del modelo, sino también entender cómo y por qué el modelo está tomando las decisiones que toma" (p. 5), para ello, en esta etapa también se puede incluir la generación de métricas como la precisión, recall, F1-score para tareas de clasificación, o el error cuadrático medio (MSE) para tareas de regresión, pero a diferencia de la fase de modelado, en esta fase, las métricas se usan para confirmar que el modelo seleccionado se ajusta bien a los datos no vistos, además, en la evaluación se realiza un análisis más profundo de la adecuación del modelo al contexto comercial, considerando no solo las métricas técnicas sino también la interpretabilidad y el valor comercial.

Finalmente, se lleva a cabo una revisión global del proceso y resultados obtenidos, aquí se decide si es necesario realizar ajustes adicionales en los datos o en los modelos, de encontrarse problemas de gran magnitud, se pueden realizar ajuste adicionales al odelo antes de pasar a la fase de despliegue.

Quinta fase o fase de Despliegue, esta comprende la implementación práctica de los modelos desarrollados, Shearer (2000) menciona que "esta fase asegura que los resultados del modelo se integren en los procesos comerciales del cliente, ya sea a través de informes, sistemas automáticos o aplicaciones interactivas" (p. 10), lo que indica que el despliegue es donde el valor del proyecto de minería de datos se materializa y muestra resultados tangibles del proceso completo.

Imagen 1: Ciclo de Vida de de minería de datos



Fuente: https://www.ibm.com/docs/es/spss-modeler/saas?topic=dm-crisp-help-overview

3. CAPITULO 3. METODOLOGÍA

Los datos con los que se cuenta para este estudio datan desde 1990, esto implica que se debe manejar una gran cantidad de información, por lo que se toma como referencia la metodología CRIPS-DM, debido a que es un estándar probado que se ajusta tanto a modelos supervisados como no supervisado, la metodología además contine las fase necesarias para el correcto manejo de los datos, iniciando de su fase de comprensión del negocio, pasando por el modelado y culminando con la presentación de resultados, dentro de cada fase se tiene en cuenta la comunicación y retroalimentación que cada fase mantiene.

3.1. Comprensión o Entendimiento del Negocio

En la primera fase de la metodología se alineará los objetivos técnicos del proyecto con las necesidades y prioridades estratégicas de Escuela Superior Politécnica de Chimborazo (ESPOCH), que, para este caso, se centra en el análisis de datos de variables ambientales recopilados en la estación meteorológica desde 1990 hasta 2023, estos datos incluyen variables clave como temperatura, humedad, velocidad del viento, heliofanía y precipitación, todas ella medidas y recolectadas a través de sensores instalados en la estación meteorológica de la institución, los sensores guardan información diariamente, generando un extenso conjunto de datos numéricos con registros fechados día a día.

El proyecto busca encontrar las relaciones que pudieran existir entre las variables ambientales y comprender a fondo cuál de ellas es la más relevante en el sentido de influencia sobre las demás, se busca además, definir claramente las preguntas que el análisis de datos debe responder, como por ejemplo, la identificación de tendencias climáticas a lo largo de los años, la relación entre distintas variables meteorológicas y cómo esta información puede utilizarse para mejorar la gestión ambiental o académica en la institución y los aportes al sector agrícola de la zona, que a través de los distintos proyectos de vinculación la ESPOCH transmite.

La estación meteorológica está bajo la dependencia directa del Decanato de la Facultad de Recursos Naturales, y a través de esta la consecuente vinculación con la comunidad y sociedad en los distintos proyectos de vinculación, a continuación, se muestra la estructura orgánica de las facultades de la ESPOCH.

Ilustración 1: Estructura orgánica de Facultades de la ESPOCH

Fuente: http://lotaip.espoch.edu.ec/pdf/ORGANICO_FUNCIONAL_DE_LA_ESPOCH.pdf

Como se evidencia en la figura anterior, en la parte correspondiente a servicios se ubica la estación meteorológica, que para este caso de estudio es estrictamente competencia de la FRN, sin embargo, es importante mencionar que la ESPOCH cuentan con varias extensiones y Estaciones experimentales en el país, donde también cuenta con otras estaciones meteorológicas.

3.2. Comprensión o Entendimiento de los Datos

3.2.1. Recolección Inicial de Datos

Los datos de este proyecto fueron tomados de archivos Excel y archivos planos con lo que se contaba en la estación meteorológica de la ESPOCH, dado que cada sensor toma la información de diferente manera, los administradores y custodios de esta información, tomaban los datos y almacenaban en matrices, que a lo largo del tiempo acumulaban estos datos, este proceso lo realizan todos los días, durante los 365 días del año, por lo que esta información tubo que ser procesada previamente para poder generar un archivo de tipo csv, en el que se crearon columnas con los diferentes variables ambientales y así poder unificarlos por fecha y variable en un solo documento, las variables que encontramos en las matrices son de tipo numéricas y de texto, así tenemos mes, día, año, humedad relativa media, humedad relativa mínima, humedad relativa máxima, temperatura

media, temperatura máxima, temperatura mínima, heliofanía, precipitación, velocidad del viento, tensión de vapor, punto de rocío.

3.2.2. Descripción de los Datos

La información que se encontró en las matrices que almacenan los datos meteorológicos, estaban repartidas por sus fuentes, es decir, los de velocidad de viento en una matriz propia, los de húmedas lo propio y así en cada caso con las diferentes fuentes u orígenes de cada sensor, a continuación, se muestra las variables y su descripción:

Tabla 2: Descripción de campos de matrices de datos meteorológicos

MES	Mes en el cual fue tomada la medida
DIA	Día en el cual fue tomada la medida
ANIO	Año en el cual fue tomada la medida
HUMEDAD RELATIVA MED	La humedad realtiva media de las muestras tomadas en un día
HUMEDAD RELATIVA MIN	La humedad realtiva máxima de las muestras tomadas en un día
HUMEDAD RELATIVA MAX	La humedad realtiva mínima de las muestras tomadas en un día
TEMPERATURA MEDIA	La temperatura realtiva media de las muestras tomadas en un día
TEMPERATURA MAXIMA	La temperatura realtiva máxima de las muestras tomadas en un día
TEMPERATURA MINIMA	La temperatura realtiva mínima de las muestras tomadas en un día
HELIOFANIA	Cantidad de luz solar directa que llega a la superficie
PRECIPITACION	Cantidad de agua en sus diferentes maneras que caen en la superficie
VELOCIDAD DEL VIENTO	Velocidad del viento en el punto de la toma de la muestra
TENSION DE VAPOR	Fuerza que ejerce el vapor cuando se equilibra con el agua líquida
PUNTO DE ROCIO	Temperatura a la cual el aire se enfría y el vapor de agua se condensa en líquido

Elaboración: Propia

La mayoría de los datos son de tipo numérico con decimales, excepto los campos de día, me y año que son de tipo numérico entero, en la siguiente tabla se muestra el tipo de dato por cada variable

Tabla 3: Tipo de datos por variable

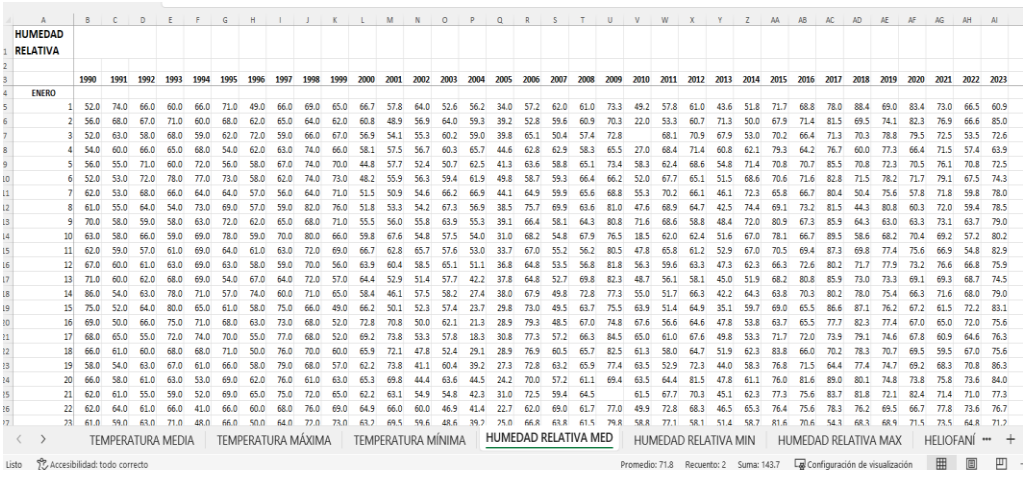
Variable	Tipo de dato
MES	Numérico entero
DIA	Numérico entero
ANIO	Numérico entero
HUMEDAD RELATIVA MED	Numérico decimal
HUMEDAD RELATIVA MIN	Numérico decimal
HUMEDAD RELATIVA MAX	Numérico decimal
TEMPERATURA MEDIA	Numérico decimal
TEMPERATURA MAXIMA	Numérico decimal
TEMPERATURA MINIMA	Numérico decimal
HELIOFANIA	Numérico decimal
PRECIPITACION	Numérico decimal
VELOCIDAD DEL VIENTO	Numérico decimal
TENSION DE VAPOR	Numérico decimal
PUNTO DE ROCIO	Numérico decimal

Elaboración: Propia

3.2.3. Exploración de los Datos

La primera acción que se realizó luego de tener las matrices y analizar la información contenida en ellas, fue consolidar la información en un solo archivo, debido a que la información estaba contenida en diferentes pestañas y con un formato que no se podía manejar apropiadamente para los procesos de limpieza, depuración y demás procesos del modelado que se pretende conseguir, como se evidencia en la figura que continúa

Imagen 2: matrices de datos recolectadas

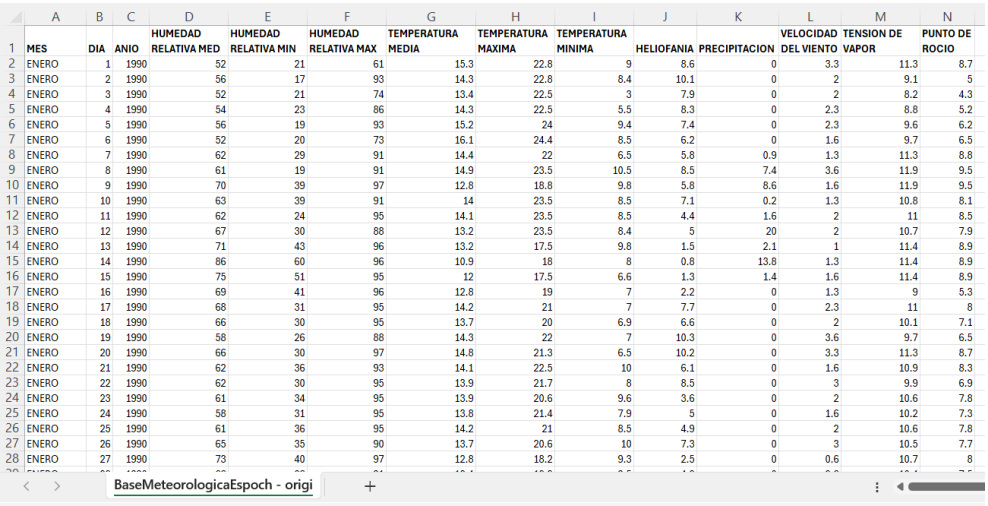


Elaboración: Técnicos de la estación meteorológica ESPOCH.

Fuente: Estación Meteorológica de la ESPOCH

Para ello, se tomó los datos de todas las pestañas y se unifico en una sola pestaña, dando formato único, donde las variables se convierten en columnas y se añaden dos columnas adicionales en la que se asigna el año y mes respectivamente, como se muestra en la siguiente figura

Imagen 3: Matriz unificada de datos meteorológicos



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	MES	DIA	ANIO	HUMEDAD RELATIVA MED	HUMEDAD RELATIVA MIN	HUMEDAD RELATIVA MAX	TEMPERATURA MEDIA	TEMPERATURA MAXIMA	TEMPERATURA MINIMA	HELIOFANIA	PRECIPITACION	VELOCIDAD DEL VIENTO	TENSION DE VAPOR	PUNTO DE ROCIO
2	ENERO	1	1990	52	21	61	15.3	22.8	9	8.6	0	3.3	11.3	8.7
3	ENERO	2	1990	56	17	93	14.3	22.8	8.4	10.1	0	2	9.1	5
4	ENERO	3	1990	52	21	74	13.4	22.5	3	7.9	0	2	8.2	4.3
5	ENERO	4	1990	54	23	86	14.3	22.5	5.5	8.3	0	2.3	8.8	5.2
6	ENERO	5	1990	56	19	93	15.2	24	9.4	7.4	0	2.3	9.6	6.2
7	ENERO	6	1990	52	20	73	16.1	24.4	8.5	6.2	0	1.6	9.7	6.5
8	ENERO	7	1990	62	29	91	14.4	22	6.5	5.8	0.9	1.3	11.3	8.8
9	ENERO	8	1990	61	19	91	14.9	23.5	10.5	8.5	7.4	3.6	11.9	9.5
10	ENERO	9	1990	70	39	97	12.8	18.8	9.8	5.8	8.6	1.6	11.9	9.5
11	ENERO	10	1990	63	39	91	14	23.5	8.5	7.1	0.2	1.3	10.8	8.1
12	ENERO	11	1990	62	24	95	14.1	23.5	8.5	4.4	1.6	2	11	8.5
13	ENERO	12	1990	67	30	88	13.2	23.5	8.4	5	20	2	10.7	7.9
14	ENERO	13	1990	71	43	96	13.2	17.5	9.8	1.5	2.1	1	11.4	8.9
15	ENERO	14	1990	86	60	96	10.9	18	8	0.8	13.8	1.3	11.4	8.9
16	ENERO	15	1990	75	51	95	12	17.5	6.6	1.3	1.4	1.6	11.4	8.9
17	ENERO	16	1990	69	41	96	12.8	19	7	2.2	0	1.3	9	5.3
18	ENERO	17	1990	68	31	95	14.2	21	7	7.7	0	2.3	11	8
19	ENERO	18	1990	66	30	95	13.7	20	6.9	6.6	0	2	10.1	7.1
20	ENERO	19	1990	58	26	88	14.3	22	7	10.3	0	3.6	9.7	6.5
21	ENERO	20	1990	66	30	97	14.8	21.3	6.5	10.2	0	3.3	11.3	8.7
22	ENERO	21	1990	62	36	93	14.1	22.5	10	6.1	0	1.6	10.9	8.3
23	ENERO	22	1990	62	30	95	13.9	21.7	8	8.5	0	3	9.9	6.9
24	ENERO	23	1990	61	34	95	13.9	20.6	9.6	3.6	0	2	10.6	7.8
25	ENERO	24	1990	58	31	95	13.8	21.4	7.9	5	0	1.6	10.2	7.3
26	ENERO	25	1990	61	36	95	14.2	21	8.5	4.9	0	2	10.6	7.8
27	ENERO	26	1990	65	35	90	13.7	20.6	10	7.3	0	3	10.5	7.7
28	ENERO	27	1990	73	40	97	12.8	18.2	9.3	2.5	0	0.6	10.7	8

BaseMeteorologicaEspoch - origi

Elaboración: Propia

Con la información consolidada en un solo archivo, la cantidad de registros o filas es de 12444, con lo que se procede a realizar la descripción de datos, los análisis de distribución y los estadísticos que determinen como se encuentra la data.

Todo el proceso se lo realiza en Phyton, utilizando la herramienta de Visual code con interfaz de programación, se procede a cargar la data y generar la descripción de los datos obteniendo el siguiente resultado

Imagen 4: Descripción de datos cargado en phyton

```
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   FECHA                 12444 non-null  object
 1   HUMEDAD RELATIVA MED  12143 non-null  float64
 2   HUMEDAD RELATIVA MIN  11636 non-null  float64
 3   HUMEDAD RELATIVA MAX  11654 non-null  float64
 4   TEMPERATURA MEDIA     12166 non-null  float64
 5   TEMPERATURA MAXIMA    11272 non-null  float64
 6   TEMPERATURA MINIMA    11529 non-null  float64
 7   HELIOFANIA            11606 non-null  float64
 8   PRECIPITACION         11631 non-null  float64
 9   VELOCIDAD DEL VIENTO  11111 non-null  object
 10  TENSION DE VAPOR      8397 non-null   object
 11  PUNTO DE ROCIO        8396 non-null   object
```

Elaboración: propia

De la misma manera, se muestra los valores estadísticos básicos como la media, mediana, desviación estándar, mínimos y máximos

Imagen 5: estadísticos iniciales

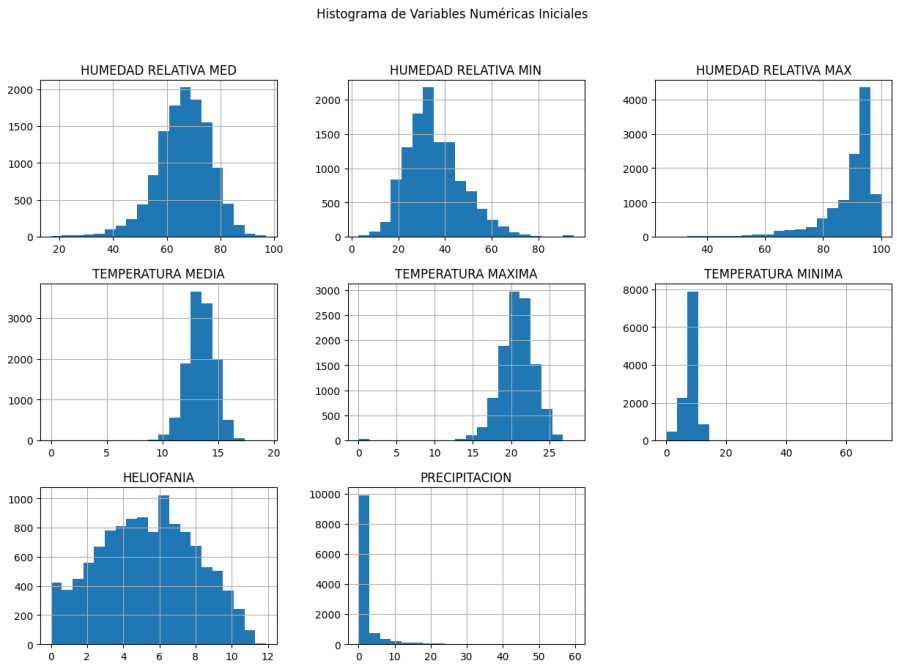
	HUMEDAD RELATIVA MED	HUMEDAD RELATIVA MIN	HUMEDAD RELATIVA MAX	\
count	12143.000000	11636.000000	11654.000000	
mean	66.118918	35.536525	89.276986	
std	10.186047	12.154398	9.735108	
min	16.800000	3.000000	26.000000	
...				
25%	3.300000	0.000000		
50%	5.400000	0.000000		
75%	7.400000	1.100000		
max	11.900000	59.600000		

Elaboración: propia

3.2.4. Verificación de la Calidad de los Datos

Para la verificación de datos se procede a graficar los datos iniciales, para ver su distribución, así como el conteo de nulos

Imagen 6: Gráficas de las variables en su estado inicial



Elaboración: propia

Como primera observación, se puede evidenciar que solo se grafica 8 variables de las 11 posibles, esto es debido al tipo de dato de estas variables, esto se corregirá en etapas posteriores.

Para el caso del conteo de valores nulos, se determina que ciertas variables tienen una alta cantidad de valores nulos, lo que deriva en un alto porcentaje de faltantes en cada columna, esto puede causar problemas posteriores, esto también va a ser corregido en las siguientes etapas del proceso.

Imagen 7: Valores nulos por variable

	Cantidad de Valores Faltantes	\
FECHA	0	
HUMEDAD RELATIVA MED	301	
HUMEDAD RELATIVA MIN	808	
HUMEDAD RELATIVA MAX	790	
TEMPERATURA MEDIA	278	
TEMPERATURA MAXIMA	1172	
TEMPERATURA MINIMA	915	
HELIOFANIA	838	
PRECIPITACION	813	
VELOCIDAD DEL VIENTO	1333	
TENSION DE VAPOR	4047	
PUNTO DE ROCIO	4048	

	Porcentaje de Valores Faltantes
FECHA	0.000000
HUMEDAD RELATIVA MED	2.418836
HUMEDAD RELATIVA MIN	6.493089
HUMEDAD RELATIVA MAX	6.348441
TEMPERATURA MEDIA	2.234008
TEMPERATURA MAXIMA	9.418194
TEMPERATURA MINIMA	7.352941
HELIOFANIA	6.734169
PRECIPITACION	6.533269
VELOCIDAD DEL VIENTO	10.711990
TENSION DE VAPOR	32.521697
PUNTO DE ROCIO	32.529733

Elaboración: propia

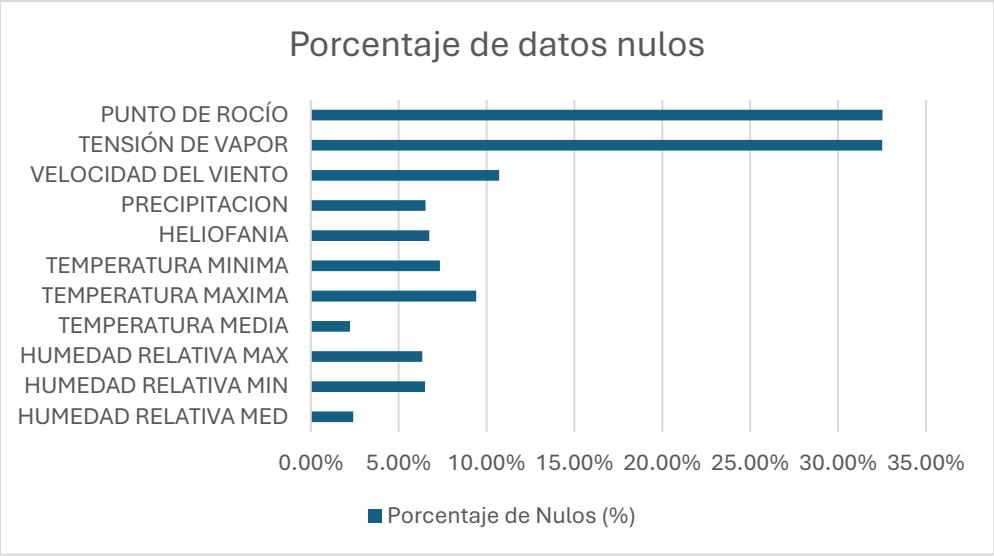
Las variables con mas alto porcentaje de valores nulos son la tensión de vapor y punto de rocío, estas superan el 30%, cantidad que es extremadamente alta para valores faltantes.

3.3. Preparación de Datos

3.3.1. Selección de datos

Considerando la gran cantidad de datos faltantes como se evidencia en la siguiente ilustración, se elige únicamente a los campos que no sobrepasan el 30%, mientras que, los que superan este porcentaje, se procederá con la respectiva eliminación.

Ilustración 2: Porcentaje de datos nulos

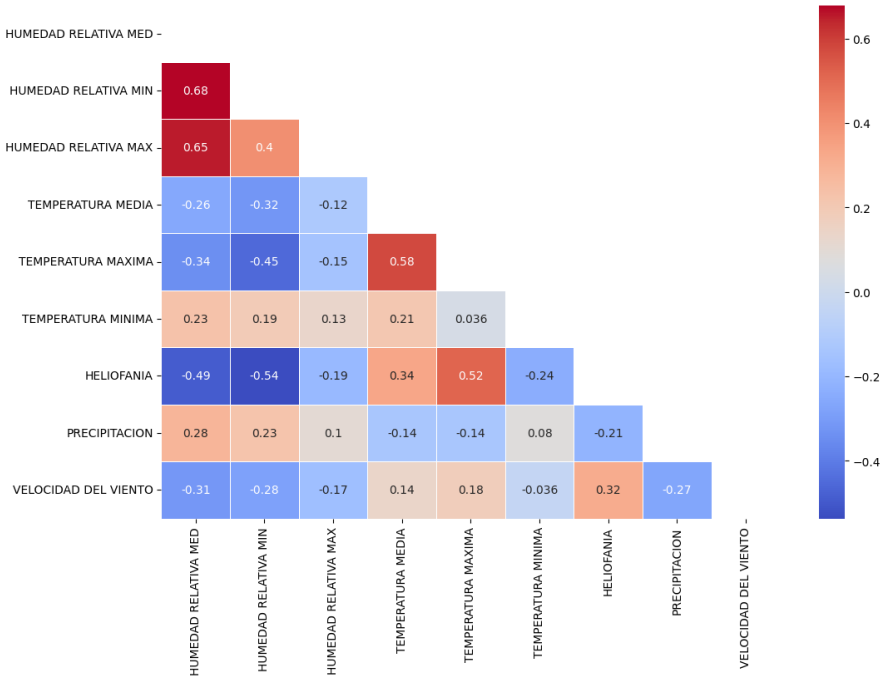


Elaboración: propia

Bajo esta premisa, en la limpieza de datos se procederá con la eliminación de dichas columnas y se quedarán únicamente las que no superen el porcentaje indicado.

Por otro lado, se realiza también la matriz de correlación, para determinar si las variables que quedan tienen correlación directa o indirecta, esto va a permitir decidir si alguna otra variable debe quitarse del dataset.

Imagen 8: Matriz de correlación de variables



Elaboración: propia

De esta gráfica se puede destacar que todas las variables están relacionadas entre sí, ya sea de manera directa o indirecta y a pesar que no tengan una correlación muy elevada, sin embargo, este análisis se retomará en etapas posteriores, donde se podrá identificar cual de las variables es la mas influyente y la que se va a predecir.

3.3.2. Limpieza de datos

En esta sub-fase, se procede con la eliminación de las columnas con porcentajes muy altos de valores nulos, dichas columna son “Punto de Rocío” y “Tensión de Vapor”, además se procede a completar con la media, aquella columna los valores nulos de aquellas columna que tienen hasta el 5% de nulos y mediante K-Nearest Neighbors (K-NN) se procede a imputar los valores nulos de las columnas o variables que tienen valores nulos comprendidos entre el 5% y el 30%; al finalizar este proceso, tenemos los siguientes resultados

Imagen 9: Porcentaje de nulos luego de limpieza de datos

	Porcentaje de Valores Faltantes
FECHA	0.0
HUMEDAD RELATIVA MED	0.0
HUMEDAD RELATIVA MIN	0.0
HUMEDAD RELATIVA MAX	0.0
TEMPERATURA MEDIA	0.0
TEMPERATURA MAXIMA	0.0
TEMPERATURA MINIMA	0.0
HELIOFANIA	0.0
PRECIPITACION	0.0
VELOCIDAD DEL VIENTO	0.0

Elaboración: propia

Como se evidencia en la imagen, los datos están completamente limpios de nulos, se procede con la estandarización de datos, proceso que debe realizarse previo a la verificación de outliers.

La estandarización se aplicó para cada valor de las variables que se mantiene, con la siguiente fórmula

$$z = \frac{x - \mu}{\sigma}$$

Donde:

- M es la media de la columna.
- Σ es la desviación estándar de la columna.
- z es el valor estandarizado.

El rango no es fijo para los datos estandarizados, depende de la distribución original de los datos, sin embargo, la mayoría de los valores están en el rango de -3 a 3 aproximadamente.

Imagen 10: Datos estandarizados

FECHA	HUMEDAD RELATIVA MED	HUMEDAD RELATIVA MIN	HUMEDAD RELATIVA MAX	\
1990-01-01	-1.401769	-1.223763	-2.985087	
1990-01-02	-1.004637	-1.561951	0.396220	
1990-01-03	-1.401769	-1.223763	-1.611431	
1990-01-04	-1.203203	-1.054669	-0.343441	
1990-01-05	-1.004637	-1.392857	0.396220	

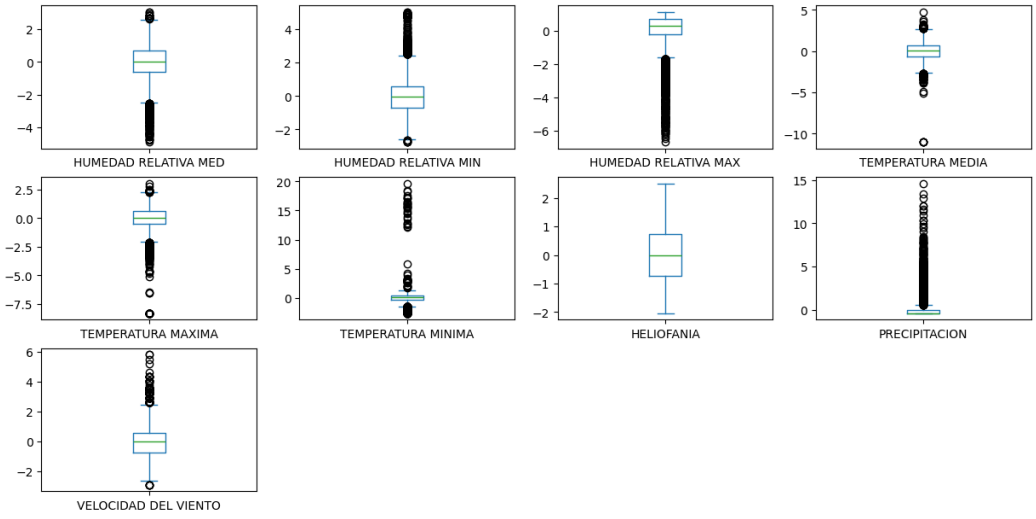
FECHA	TEMPERATURA MEDIA	TEMPERATURA MAXIMA	TEMPERATURA MINIMA	\
1990-01-01	1.493970	0.856720	0.174448	
1990-01-02	0.677574	0.856720	-0.012337	
1990-01-03	-0.057183	0.736268	-1.693400	
1990-01-04	0.677574	0.736268	-0.915130	
1990-01-05	1.412330	1.338529	0.298971	

FECHA	HELIOFANIA	PRECIPITACION	VELOCIDAD DEL VIENTO
1990-01-01	1.247592	-0.40509	1.872684
1990-01-02	1.821731	-0.40509	-0.017094
1990-01-03	0.979660	-0.40509	-0.017094
1990-01-04	1.132764	-0.40509	0.419009
1990-01-05	0.788280	-0.40509	0.419009

Fuente: Propia

Con la estandarización realizada, se procede con la verificación de outlier, para identificar valores atípicos y de esta manera evitar distorsión en los resultados del análisis y la precisión del modelo predictivo final.

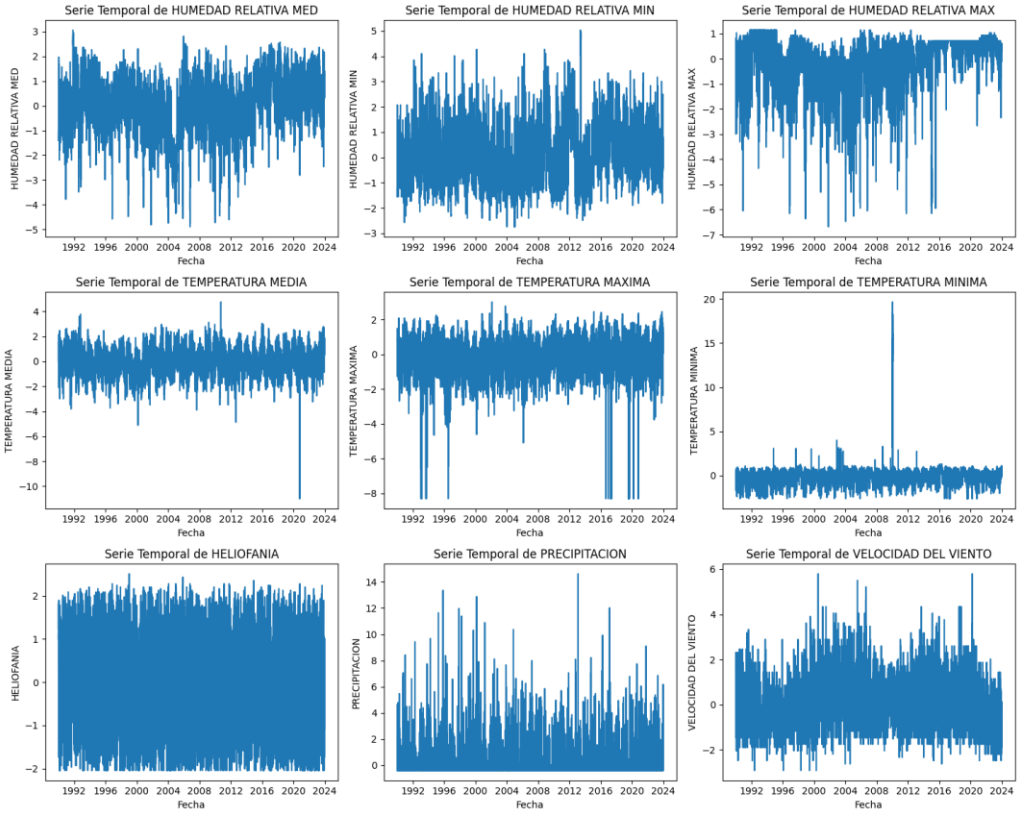
Imagen 11: Verificación de outliers



Fuente: Propia

Otra forma de visualizar los outliers para una mejor visualización se la realizo con graficas de series temporales, lo que ayudo a ver en donde esta la mayor cantidad de outliers y en que variable sobresalen los valores atípicos.

Imagen 12: Otuliers en series temporales



Fuente: Propia

Con los resultados obtenidos, se puede evidenciar que existe valores atípicos que podrían afectar al modelo de predicción, de manera en este punto, podría ser recomendable eliminarlos, por esta razón, se procede a aplicar un modelo de regresión lineal para observar el comportamiento del modelo cuando se eliminan estos valores atípicos, por tal razón, se aplica este modelo dos veces, uno con los datos en el que se incluye outliers y otro sin ellos. Con este ejercicio se pretende tener datos que apoyen la decisión de retirar o no estos valores, para evaluar el resultado del modelo aplicado, se uso la métrica del error cuadrático medio (MSE), obteniendo los siguientes resultados:

MSE con outliers: 0.607191, con el total de datos 12418

MSE sin outliers: 0.322229, filtrando outliers, total de datos 9288

Como se puede evidenciar los resultados son mas precisos al retirar los outlilers, lo que indica que estos datos afectan de manera negativa a la precisión del modelo, en consecuencia, se decide retirar estos valores atípicos.

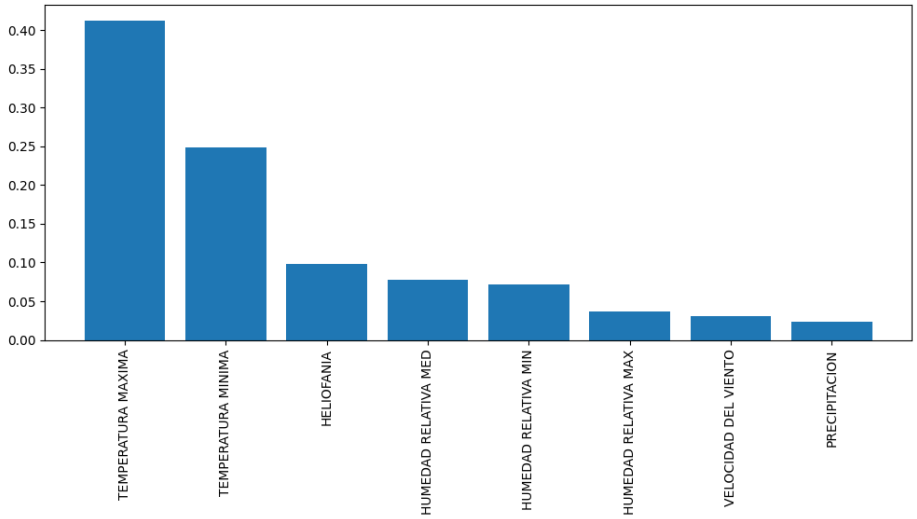
Para eliminar los outliers, se utilizó el modelo de Random Forest, aplicando limites inferior y superior con factor multiplicador de 1.5 y tomando datos de

entrenamiento del 70% y un conjunto de prueba del 30%, se utilizaron los siguiente hiperparámetros que ayudaron a mejorar el rendimiento del modelo

```
n_estimators=500,  # Aumentar el número de árboles
max_depth=20,  # Limitar la profundidad máxima de los árboles
min_samples_split=5,  # Número mínimo de muestras para dividir un nodo
min_samples_leaf=4,  # Número mínimo de muestras que debe tener una hoja
max_features="sqrt",  # Limitar el número de características para dividir un nodo
 bootstrap=True,  # Muestreo con reemplazo
random_state=42,
```

Para evaluar el modelo usando el MSE, que dio como resultado 0.2936638215018761, además este modelo permitió determinar la importancia de las características, que se muestra en la siguiente imagen y en base a estos datos, determinar que la TEMPERATURA MEDIA es la variable objetivo y las variables predictoras TEMPERATURA MAXIMA, TEMPERATURA MINIMA, HELIOFANIA, HUMEDAD RELATIVA MED, HUMEDAD RELATIVA MIN, HUMEDAD RELATIVA MAX, VELOCIDAD DEL VIENTO y PRECIPITACION.

Imagen 13: Importancia de las características



Fuente: Propia

Finalmente, la data queda con un total de 9288 registros listos para proceder con el resto de los procesos de preparación de datos.

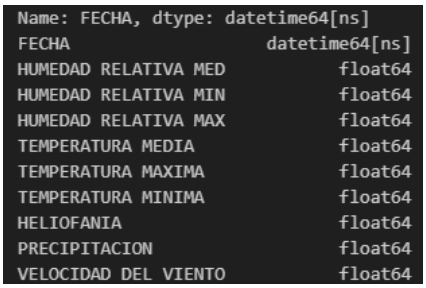
3.3.3. Construcción de datos

Para continuar con la preparación de datos, se procede a crear los campos como día (day), mes (month), año (year) y se los define con índices, y dos adicionales que son el dia de la semana (dayofweek) y semana del año (weekofyear)

3.3.4. Formateo de datos

Transforma tipo de dato fecha a un formato datetime y las variables de tipo object a datos de tipo float64, además se formatea como índice al campo fecha para optimizar los procesos de modelado.

Imagen 14: Formateo de datos



Elaboración: propia

Al finalizar con el proceso de limpieza de datos, se cuenta con un total de 9288 registros que están listos para modelar.

3.4. Modelado

3.4.1. Selección de técnicas de modelado

Por tener datos que están marcados con fechas en orden cronológico, los modelos de series temporales son los más apropiados, y considerando que se desea predecir la temperatura promedio de la zona donde se encuentra la Estación Meteorológica de la ESPOCH, se decide usar dos modelos y luego evaluar sus resultados y determinar cuál de ellos es más eficiente a la hora de predecir valores futuros, los modelos seleccionados para este proyecto son el de VAR (Vector Autoregression), que es usado normalmente en casos donde se necesita analizar y predecir la relación y la dinámica conjunta entre múltiples variables interrelacionadas a lo largo del tiempo y el SLTM (Long Short-Term Memory), que es útil cuando se quiere predecir resultados basados en datos que tienen patrones complejos (no lineales) y relación temporal de largo plazo.

3.4.2. Diseño de prueba

Para el caso del modelo de datos utilizando VAR, se realizó otra prueba usando datos de entrenamiento de 80% y 20% para pruebas.

Por su lado, LSTM se modelo usando datos de entrenamiento del 80% y 20% para pruebas.

3.4.3. Construcción y Evaluación de los modelados

MODELADO CON VAR: se realiza la división de datos, entre pruebas (20%) y entrenamiento (80%), y con estos parámetros, se procede a evaluar el modelo y se obtiene los siguientes resultados respecto a la TEMPERATURA MEDIA:

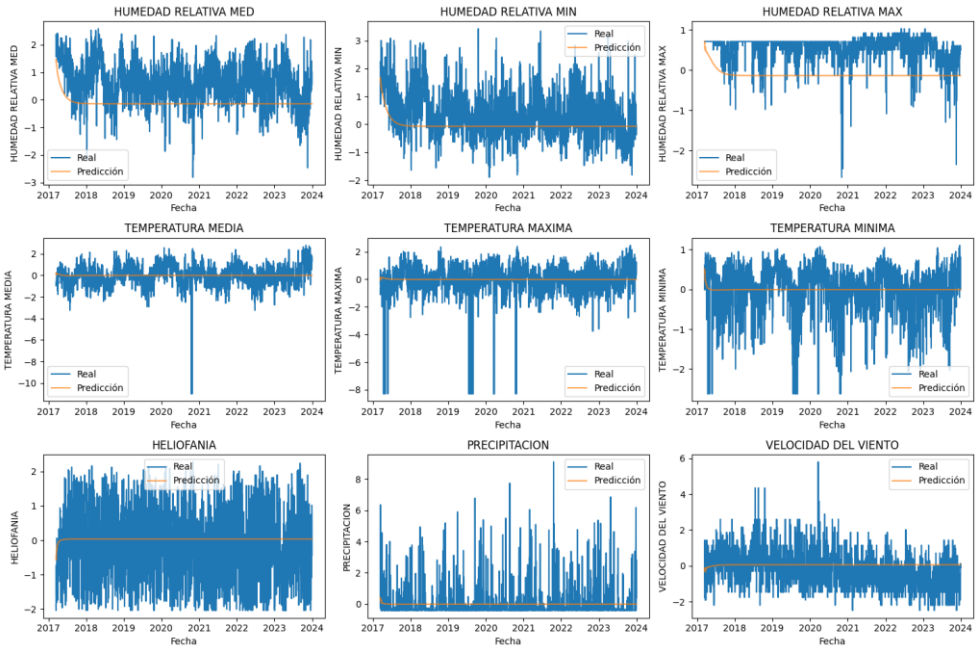
Mean Squared Error (MSE) es de 1.11738875, lo cual significa que el modelo aplicado no se está ajustando adecuadamente y por ende no muy preciso.

Con respecto al Mean Absolute Error (MAE), el valor obtenido es de 0.769516694417, lo que indica un valor elevado de desviación de dato con respecto a la realidad, confirmando el desajuste del modelo VAR.

De igual manera, al analizar R-squared (R^2) que es de -0.0006701875185, se puede notar que es un valor negativo, lo cual sugiere que el modelo no solo no está capturando la relación entre las variables predictoras y la variable objetivo, sino que las predicciones están, en promedio, más lejos de los valores reales.

Para confirmar estas aseveraciones, se procede a realizar la verificación de manera visual, utilizando graficas por cada variable,

Imagen 15: Modelado de datos usando VAR



Elaboración: propia

Con estos resultados se puede interpretar que este modelo no es adecuado para el dataset analizado, pues se evidencia en las gráficas que el modelo no solo falla con la TEMPERTURA MEDIA, sino también con el resto de las variables.

MODELADO CON SLTM

Para el modelo aplicando SLTM, también se usaron datos de entrenamiento del 80% y de pruebas de 20%, con los siguientes resultados de las métricas en promedio de las variables analizadas

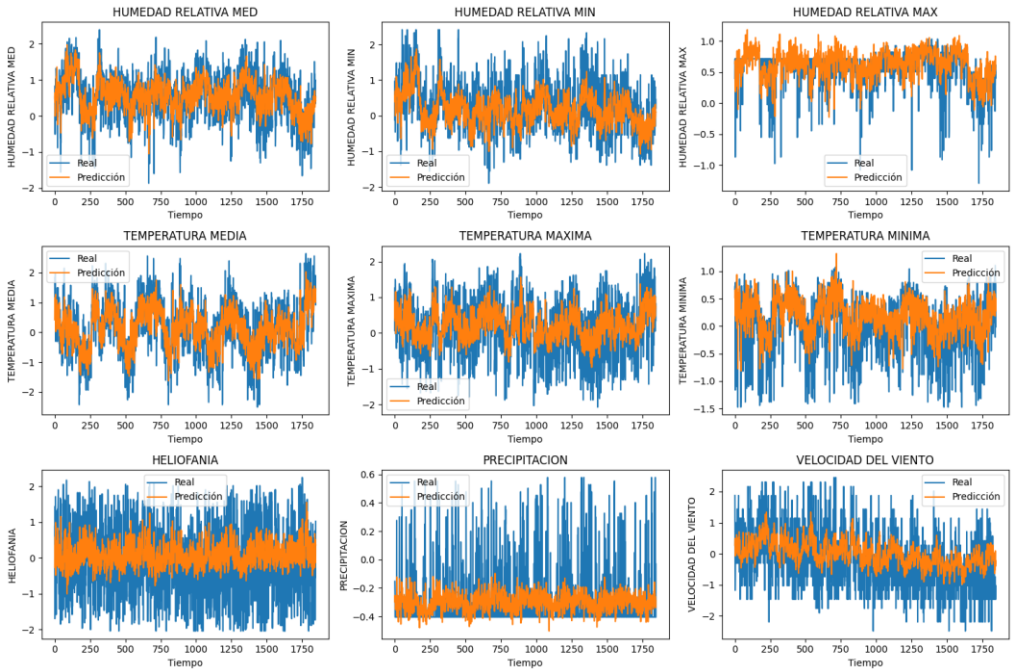
El MSE de 0.40876340783790865, indica que el modelo tiene un error cuadrático medio moderado, hay que recordar que cuanto menor sea el MSE, mejor será el rendimiento del modelo, pero tiene una mejora significativa con respecto al modelado usando VAR

En lo que se refiere a MAE se tiene un valor de 0.4624070149183523, indica que es un error moderado, lo que sugiere que el modelo tiene una precisión razonable

Con respecto a R^2, se obtuvo un valor de 0.19154860319109548, este dato no es muy alentador, pues sugiere que el modelo explica aproximadamente el 19.15% de la variabilidad de los datos, por lo que se sugiere realizar ajustes en sus parámetros.

A pesar de estos resultados, si observamos las gráficas de las diferentes variables, se puede notar una mejora sustancial respecto al modelado con VAR

Imagen 16: Modelado de datos usando LSTM



Elaboración: propia

3.4.4. Ajuste del modelado

Con los resultados obtenidos, se decide continuar con los ajustes del modelo SLTM, ya que el que mejor resultados obtuvo de los dos, de manera que, se procede a realizar los ajustes y mejoras

Antes de realizar el ajuste de los parámetros e hiperparámetros de modelo SLTM, se ajusta Random Forest, para bajar los outliers, que se sugiere son los que están causando mucho ruido y desviación del modelo, el factor multiplicador es Random Foresta se encontraba en 1.5 y se reduce a 1.0, lo que reduce los datos de 9288 a 7498, reduciendo también el valor del MSE de 0.322229 a 0.266567.

Tabla 4: Resultado de Random Forest ajustando outliers

Cantidad de registros	MSE	MAE	R^2
7498	0.266567	0.397061	0602612

Elaboración: propia

Luego se realizan un ajuste del número de unidades LSTM, aumentado a 100 unidades en cada capa LSTM, se ha añadide además, dropout después de cada capa LSTM para evitar el sobreajuste, así como también se ajustó la tasa de aprendizaje del optimizador Adam a 0.001.

Otro cambio fue en el aumentado del hiperparámetro time_step a 20 para capturar patrones más largos y se agregó un callback de early stopping para evitar el sobreentrenamiento y se separa la evaluación de MSE, AME y R^2 por cada variable para notar donde hay menos ajuste del modelo.

3.5. Evaluación

El modelo que mejor comportamiento tuvo es el SLTM, luego de los ajustes se presentaron los siguientes resultados

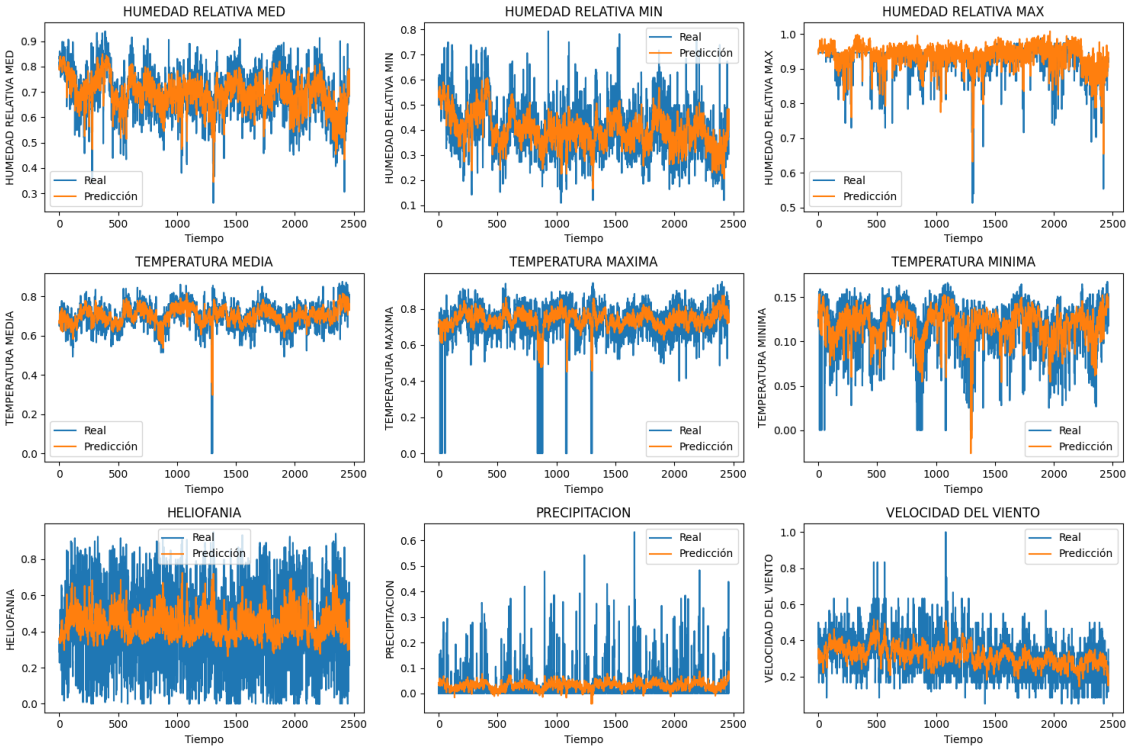
MSE: 0.009504278426456852, lo que sugiere que las predicciones del modelo están, en promedio, muy cercanas a los valores reales, este es un buen indicador de la precisión del modelo en términos de error promedio.

MAE: 0.06077209012516628, significa que, en promedio, las predicciones del modelo se desvían en aproximadamente 0.061 unidades de los valores reales, este valor es bajo, y al igual que en MSE, también indica que las predicciones son bastante precisas.

R^2: 0.2611569201052139, este valor significa que aproximadamente el 26.12% de la variabilidad en los datos está siendo explicada por el modelo y aunque no es un valor alto, sugiere que el modelo está capturando parte de las relaciones en los datos, pero hay una cantidad significativa de variabilidad que no está siendo explicada, esto puede darse por los cambios repentinos propios de la zona, ya que según los expertos de la climatología del lugar, se confirma que la variabilidad suele ser extrema e inesperada.

Gráficamente los resultados se muestran de la siguiente manera

Imagen 17: Resultados del modelo SLTM ajustado



Elaboración: propia

Como se puede evidenciar, las gráficas de la mayoría de las variables ambientales se ajustan al modelo propuesto, que, aunque en algunos casos, están más alejados de los valores reales, en promedio si esta sobre lo deseado, por lo que se modelo puede ser aceptado.

4. CAPITULO 5. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

- La recopilación y limpieza de los datos provenientes de los sensores meteorológicos se realizaron de manera correcta, sin embargo, se pudo notar que los datos almacenados pudieran tener errores de digitación, lo que genera gran cantidad de outliers y ruido, a pesar de ello, se logró manejar los valores faltantes, eliminar outliers y normalizar los datos, lo cual ayudó a mejora la calidad y consistencia de los datos en el modelado.
- Al aplicar el modelo de regresión lineal y Random Forest, se logró determinar tendencias y relaciones iniciales, mismos que ayudaron a la limpieza de datos, para la posterior aplicación de los modelos de series temporales, que permitieron modelar las diferentes variables ambientales, a fin de conseguir el objetivo planteado en el proyecto.
- El modelo LSTM aplicado mostró un rendimiento moderado en términos de error, con un MSE de 0.0095 y MAE de 0.0608, indicando que el modelo fue capaz de hacer predicciones con una precisión aceptable, sin embargo, el R^2 de 0.2612 sugiere que el modelo no capturó completamente las relaciones y patrones entre las variables, lo que limita su capacidad explicativa.

4.2. Recomendaciones

- Implementar un proceso de validación y control de calidad de los datos en tiempo real, que incluya la detección automática de outliers y errores de digitación durante la recolección de datos, esto puede ser realizado mediante el desarrollo de scripts automatizados que detecten y corrijan entradas atípicas antes de almacenar los datos, además, considerar la capacitación del personal en el manejo y entrada de datos para minimizar errores humanos.
- Continuar utilizando modelos como Regresión Lineal y Random Forest para la fase de preprocesamiento y limpieza de datos, estos modelos pueden servir como herramientas eficaces para identificar patrones y relaciones subyacentes, que luego se pueden utilizar para mejorar la precisión de los modelos de series temporales, también se sugiere explorar técnicas de selección de características para identificar cuáles variables aportan más valor en la modelización final.
- Para futuros estudios, se recomienda considerar combinar el LSTM con otros modelos o técnicas como Redes Neuronales Convolucionales (CNN) o Ensemble Learning para mejorar la captura de patrones y relaciones, además evaluar el uso de técnicas de Data Augmentation para enriquecer el conjunto de datos podría ayudar a mejorar la capacidad explicativa y la precisión del modelo.

5. REFERENCIAS

Tukey, J. W. (1977). Exploratory Data Analysis. Addison-Wesley.

Jones, P. D., & Moberg, A. (2003). Hemispheric and large-scale surface air temperature variations: An extensive revision and an update to 2001. Journal of Climate, 16(2), 206-223. https://doi.org/10.1175/1520-0442, (2003)

Peixoto, J. P., & Oort, A. H. (1992). Physics of climate. American Institute of Physics.

WMO. (2010). Guide to Meteorological Instruments and Methods of Observation. World Meteorological Organization. https://www.wmo.int/pages/prog/www/IMOP/CIMO-Guide.html

Wallace, J. M., & Hobbs, P. V. (2006). Atmospheric Science: An Introductory Survey (2nd ed.). Academic Press.

Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.

Domingos, P. (2012). A few useful things to know about machine learning. Communications of the ACM, 55(10), 78-87.

Mitchell, T. M. (1997). Machine learning. McGraw-Hill.

Russell, S., & Norvig, P. (2021). Artificial intelligence: A modern approach (4th ed.). Pearson.

Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis: Forecasting and control. Holden-Day.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21-27.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 226-231.

Greenacre, M. J. (1984). Theory and applications of correspondence analysis. Academic Press.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. Science, 313(5786), 504-507.

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (Vol. 398). John Wiley & Sons.

Johnson, S. C. (1967). Hierarchical clustering schemes. Psychometrika, 32(3), 241-254.

Konda, V. R., & Tsitsiklis, J. N. (2000). Actor-critic algorithms. Advances in Neural Information Processing Systems, 12, 1008-1014.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 281-297.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). Introduction to linear regression analysis. John Wiley & Sons.

IBM. (n.d.-a). ¿Qué es el aprendizaje supervisado? Ibm.com. Retrieved Jun 17, 2024, from https://www.ibm.com/es-es/topics/supervised-learning

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. Philosophical Magazine, 2(11), 559-572.

Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81-106.

IBM. (n.d.-b). ¿Qué es un bosque aleatorio? Ibm.com. Retrieved Jun 17,2024, from https://www.ibm.com/es-es/topics/random-forest

Sutton, R. S., McAllester, D. A., Singh, S. P., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. Advances in Neural Information Processing Systems, 12, 1057-1063.

Tibshirani, R. (1996). Regression shrinkage and selection

IBM. (n.d.-c). What is machine learning? Ibm.com. Retrieved Jun 17,2024, from https://www.ibm.com/topics/machine-learning

Box, G. E. P., Jenkins, G. M., & Reinsel, G. C. (2008). Time series analysis: Forecasting and control (4th ed.). Wiley.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

Wilcox, R. R. (2012). Introduction to robust estimation and hypothesis testing (3rd ed.). Academic Press.

https://www.ibm.com/docs/es/spss-modeler/saas?topic=guide-introduction-crisp-dm

Jolliffe, I. T. (2002). Principal component analysis (2nd ed.). Springer.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). CRISP-DM 1.0: Step-by-step data mining guide. The CRISP-DM consortium.

Shearer, C. (2000). The CRISP-DM model: The new blueprint for data mining. Journal of Data Warehousing, 5(4), 13-22.

Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a standard process model for data mining. In Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining (pp. 29-39).

6. ANEXOS

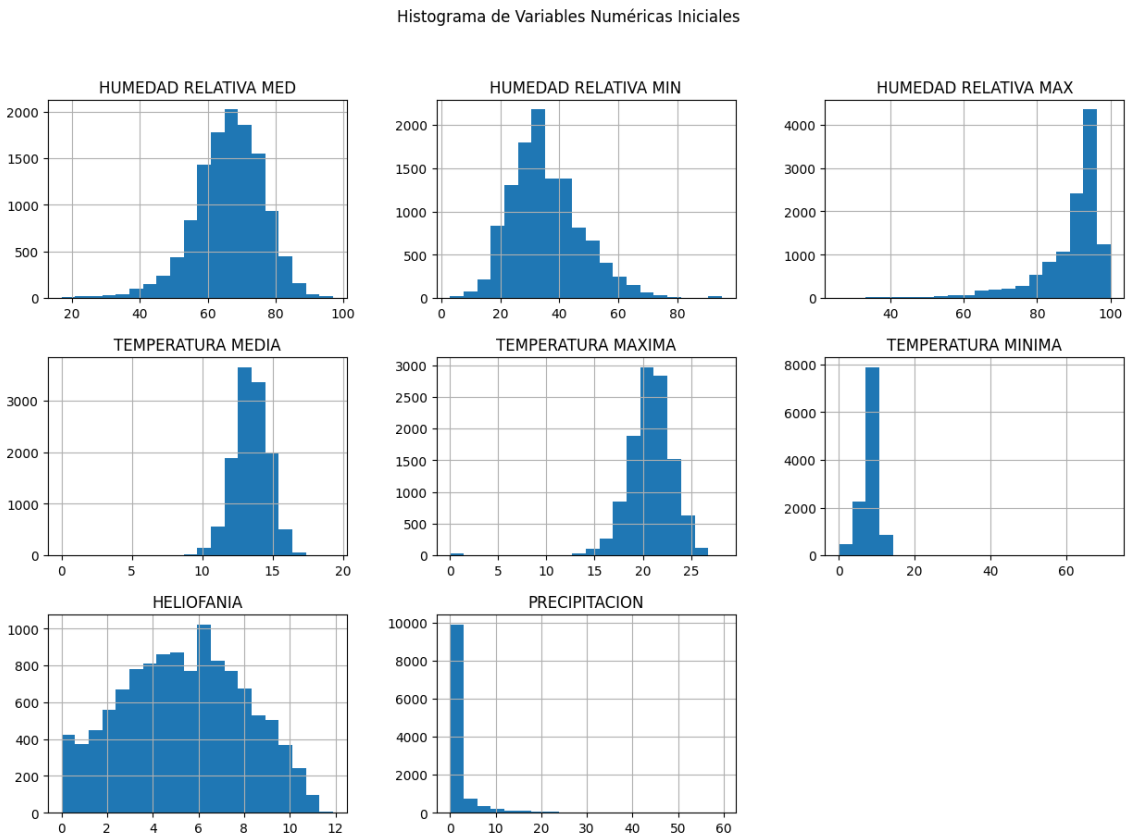
CODIGO FUENTE PHYTON

Análisis Exploratorio de Datos (EDA)
Cargar y Examinar el Dataset

```
import pandas as pd

# Cargar el archivo CSV
file_path = "D:/ BaseMeteorologicaEspoch.csv"
data = pd.read_csv(file_path)

# Mostrar las primeras filas del dataset
print(data.head())

# Información del dataset
print(data.info())

# Descripción estadística del dataset
print(data.describe())
```

Gráfica de datos iniciales

```
import matplotlib.pyplot as plt
import seaborn as sns

data.hist(bins=20, figsize=(15, 10))
plt.suptitle("Histograma de Variables Numéricas Iniciales")
plt.show()
```



Identificar Valores Faltantes

```
# Datos faltantes
# Revisión de valores faltantes
missing_values = data.isnull().sum()
missing_values_percentage = (missing_values / len(data)) * 100

# Crear un DataFrame con la cantidad y el porcentaje de valores faltantes
missing_data = pd.DataFrame(
    {
        "Cantidad de Valores Faltantes": missing_values,
        "Porcentaje de Valores Faltantes": missing_values_percentage,
    }
)

# Mostrar el DataFrame de valores faltantes
print(missing_data)


# export tabla de porcentaje de nulos
missing_data.to_csv("missing_data.csv", index=False)
Llenar valores faltantes de hasta con 5% de nulos
# Imputar valores faltantes con media o moda para columnas con hasta el 5% de valores faltantes
for column in data.columns:
    if missing_values_percentage[column] <= 5:
        if data[column].dtype == "object":  # Si la columna es categórica
            mode_value = data[column].mode()[0]
            data[column] = data[column].fillna(mode_value)
```

```
        else:  # Si la columna es numérica
            mean_value = data[column].mean()
            data[column] = data[column].fillna(mean_value)

# instalamos libreria necesaria
#!pip install scikit-learn

# Rellenar valores faltantes entre 5% y 30%

from sklearn.impute import KNNImputer


# Seleccionar columnas con porcentaje de valores faltantes entre 5% y 30%

columns_to_impute = missing_values_percentage[

    (missing_values_percentage > 5) & (missing_values_percentage <= 30)

].index


# Convertir las columnas seleccionadas a tipo numérico

for column in columns_to_impute:

    data[column] = pd.to_numeric(data[column], errors="coerce")


# Imputar valores faltantes usando KNNImputer para las columnas seleccionadas

knn_imputer = KNNImputer(n_neighbors=5)


# Aplicar la imputación a las columnas seleccionadas

data[columns_to_impute] = knn_imputer.fit_transform(data[columns_to_impute])


# Verificar que no quedan valores faltantes en las columnas imputadas

print(data[columns_to_impute].isnull().sum())

# eliminar variables con mas del 30% de nulos

# Identificar las columnas con más del 30% de valores faltantes
columns_to_drop = missing_values_percentage[missing_values_percentage > 30].index

# Eliminar las columnas identificadas
data = data.drop(columns=columns_to_drop)
```

```
# Mostrar las columnas eliminadas y el nuevo DataFrame
print(f"Columnas eliminadas: {columns_to_drop}")
print(data.head())
```
Columnas eliminadas: Index(['TENSION DE VAPOR', 'PUNTO DE ROCIO'], dtype='object')

Comprobacion de relleno de valores nulos

```
# Revisión de valores faltantes
missing_values = data.isnull().sum()
missing_values_percentage = (missing_values / len(data)) * 100

# Crear un DataFrame con la cantidad y el porcentaje de valores faltantes
missing_data = pd.DataFrame(
    {
        "Cantidad de Valores Faltantes": missing_values,
        "Porcentaje de Valores Faltantes": missing_values_percentage,
    }
)

# Mostrar el DataFrame de valores faltantes
print(missing_data)

# Exportar el DataFrame limpio a un nuevo archivo CSV llamado 'datos_limpios.csv'
data.to_csv("datos_limpios.csv", index=False)
```

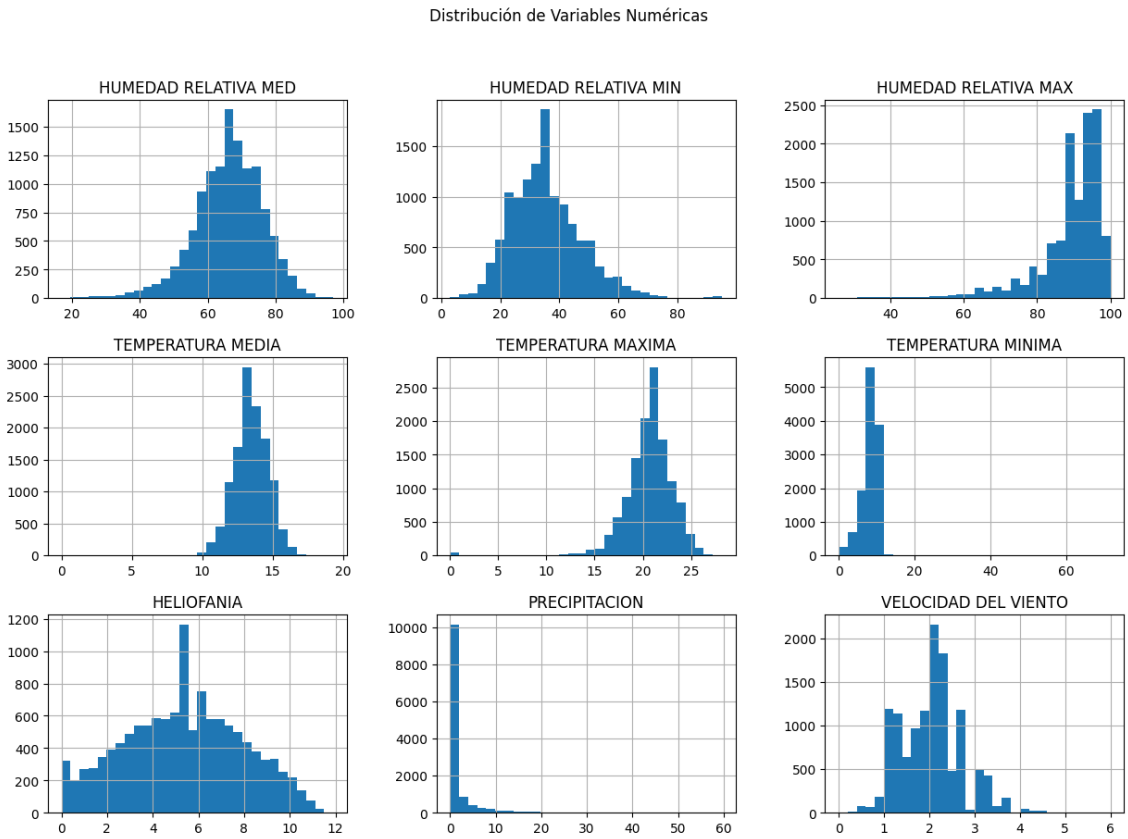
Convertir columna a tipo fecha

```
# Verificar los primeros valores y el tipo de datos en la columna 'FECHA'
print(data["FECHA"].head(10))
print(data["FECHA"].dtype)

# Limpiar los espacios en blanco y caracteres no visibles en la columna 'FECHA'
data["FECHA"] = data["FECHA"].str.strip()

# Convertir la columna 'FECHA' a formato datetime con formato específico si es necesario
# Puedes ajustar el formato según tus datos, por ejemplo, '%Y-%m-%d' o '%d/%m/%Y'
data["FECHA"] = pd.to_datetime(data["FECHA"], errors="coerce", format="%d/%m/%Y")

# Verificar filas con fechas no convertidas
invalid_dates = data[data["FECHA"].isnull()]
print("Filas con fechas no convertidas:")
print(invalid_dates)

# Eliminar filas con fechas no convertidas si es necesario
data.dropna(subset=["FECHA"], inplace=True)

# Establecer 'FECHA' como índice
data.set_index("FECHA", inplace=True)
# Exportar el DataFrame limpio a un nuevo archivo CSV llamado 'datos_limpios.csv'
```

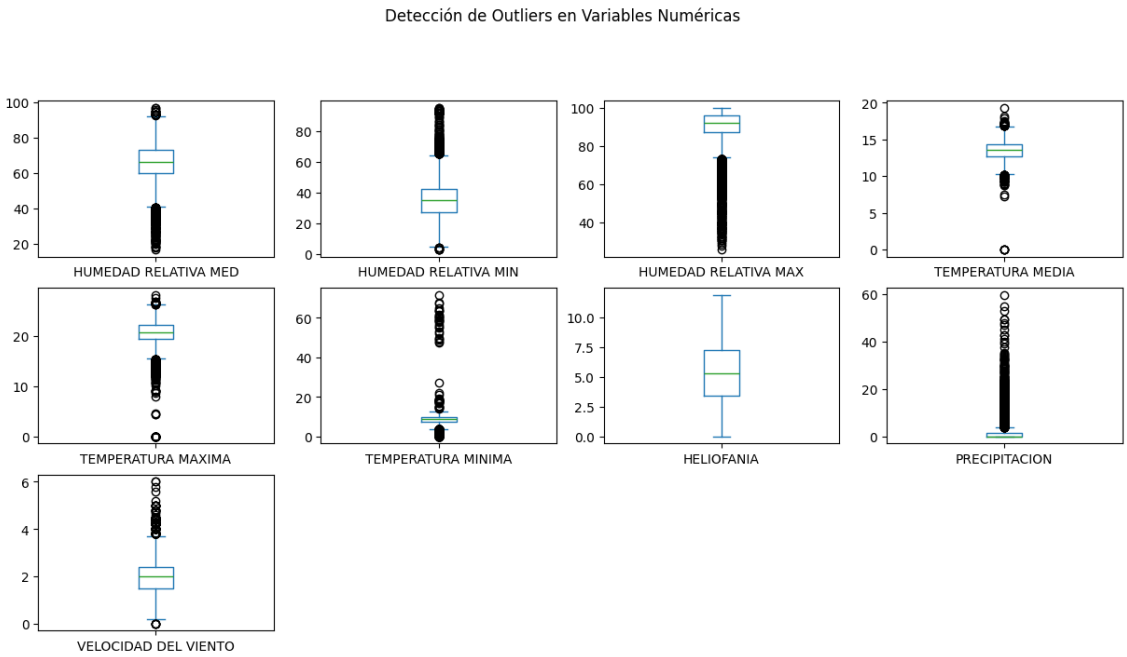
```
# data.to_csv("datos_limpios_error_fecha.csv", index=False)
# Revisión de valores faltantes
missing_values = data.isnull().sum()
missing_values_percentage = (missing_values / len(data)) * 100

# Crear un DataFrame con la cantidad y el porcentaje de valores faltantes
missing_data = pd.DataFrame(
   {
      "Cantidad de Valores Faltantes": missing_values,
      "Porcentaje de Valores Faltantes": missing_values_percentage,
   }
)

# Mostrar el DataFrame de valores faltantes
print(missing_data)
                    Cantidad de Valores Faltantes  \
HUMEDAD RELATIVA MED                             0
HUMEDAD RELATIVA MIN                             0
HUMEDAD RELATIVA MAX                             0
TEMPERATURA MEDIA                                0
TEMPERATURA MAXIMA                               0
TEMPERATURA MINIMA                               0
HELIOFANIA                                       0
PRECIPITACION                                    0
VELOCIDAD DEL VIENTO                             0

                    Porcentaje de Valores Faltantes
HUMEDAD RELATIVA MED                             0.0
HUMEDAD RELATIVA MIN                             0.0
HUMEDAD RELATIVA MAX                             0.0
TEMPERATURA MEDIA                                0.0
TEMPERATURA MAXIMA                               0.0
TEMPERATURA MINIMA                               0.0
HELIOFANIA                                       0.0
PRECIPITACION                                    0.0
VELOCIDAD DEL VIENTO                             0.0
```

Analisis de datos mediante histogramas y bloxplot

```
import matplotlib.pyplot as plt

# Crear histogramas para cada variable numérica
data.select_dtypes(include=["float64", "int64"]).hist(bins=30, figsize=(15, 10))
plt.suptitle("Distribución de Variables Numéricas")
plt.show()
```

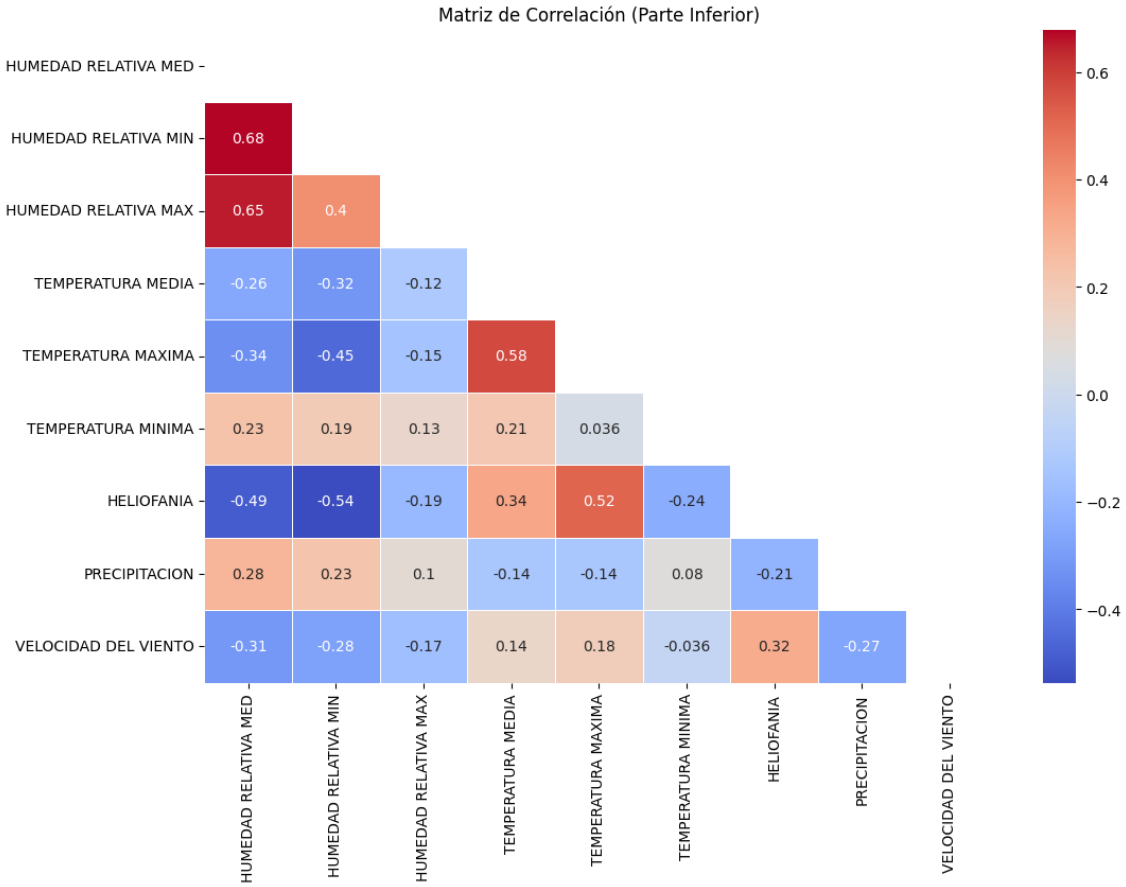



```
# Crear boxplots para cada variable numérica
data.select_dtypes(include=["float64", "int64"]).plot(
    kind="box",
    subplots=True,
    layout=(4, 4),
    figsize=(15, 10),
    title="Detección de Outliers",
)
plt.suptitle("Detección de Outliers en Variables Numéricas")
plt.show()
```



Matriz de correlacion

```
import numpy as np
import seaborn as sns


# Calcular la matriz de correlación

correlation_matrix = data.corr()


# Crear una máscara para la parte superior de la matriz de correlación
mask = np.triu(np.ones_like(correlation_matrix, dtype=bool))

# Crear un heatmap de la matriz de correlación, mostrando solo la parte inferior
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, mask=mask, annot=True, cmap="coolwarm", linewidths=0.5)
plt.title("Matriz de Correlación (Parte Inferior)")
plt.show()
```



Se aplica estandarizacion de datos para mejorar el análsis

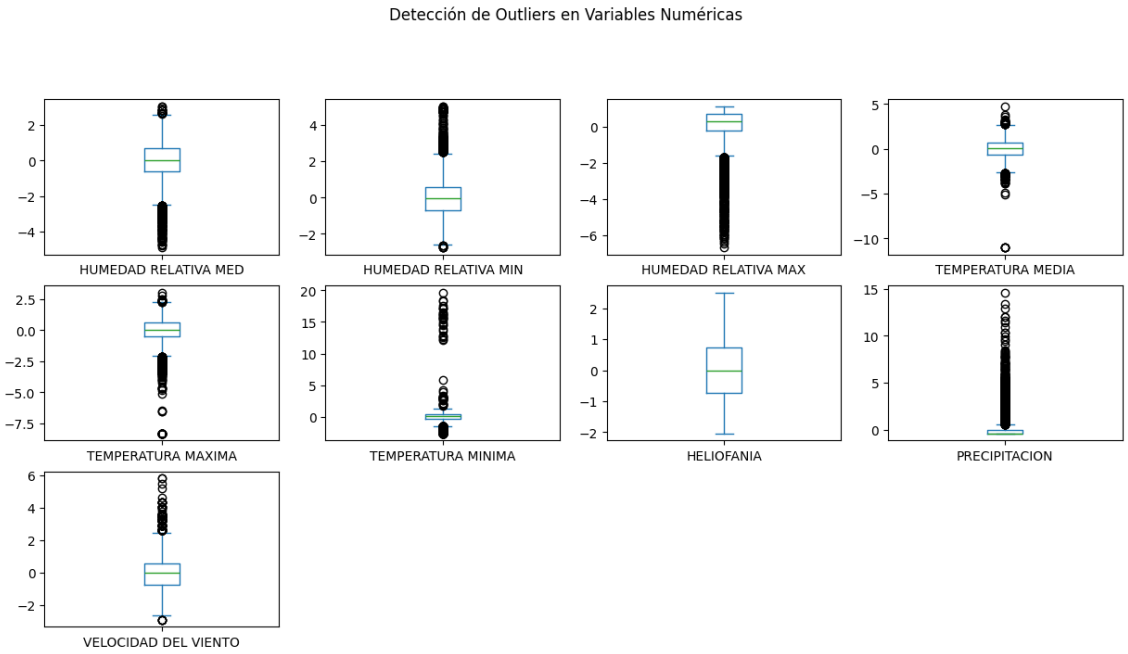
```
from sklearn.preprocessing import StandardScaler

# Seleccionar columnas numéricas para estandarizar
numeric_columns = [
    "HUMEDAD RELATIVA MED",
    "HUMEDAD RELATIVA MIN",
    "HUMEDAD RELATIVA MAX",
    "TEMPERATURA MEDIA",
    "TEMPERATURA MAXIMA",
    "TEMPERATURA MINIMA",
    "HELIOFANIA",
    "PRECIPITACION",
    "VELOCIDAD DEL VIENTO",
]

# Crear un objeto StandardScaler
scaler = StandardScaler()

# Estandarizar todas las columnas numéricas
data[numeric_columns] = scaler.fit_transform(data[numeric_columns])

# Verificar los resultados
print(data[numeric_columns].head())
          HUMEDAD RELATIVA MED    HUMEDAD RELATIVA MIN    HUMEDAD
RELATIVA MAX  \
```

FECHA
1990-01-01 -1.401769 -1.223763 -2.985087
1990-01-02 -1.004637 -1.561951 0.396220
1990-01-03 -1.401769 -1.223763 -1.611431
1990-01-04 -1.203203 -1.054669 -0.343441
1990-01-05 -1.004637 -1.392857 0.396220

TEMPERATURA MEDIA TEMPERATURA MAXIMA TEMPERATURA MINIMA \
FECHA
1990-01-01 1.493970 0.856720 0.174448
1990-01-02 0.677574 0.856720 -0.012337
1990-01-03 -0.057183 0.736268 -1.693400
1990-01-04 0.677574 0.736268 -0.915130
1990-01-05 1.412330 1.338529 0.298971

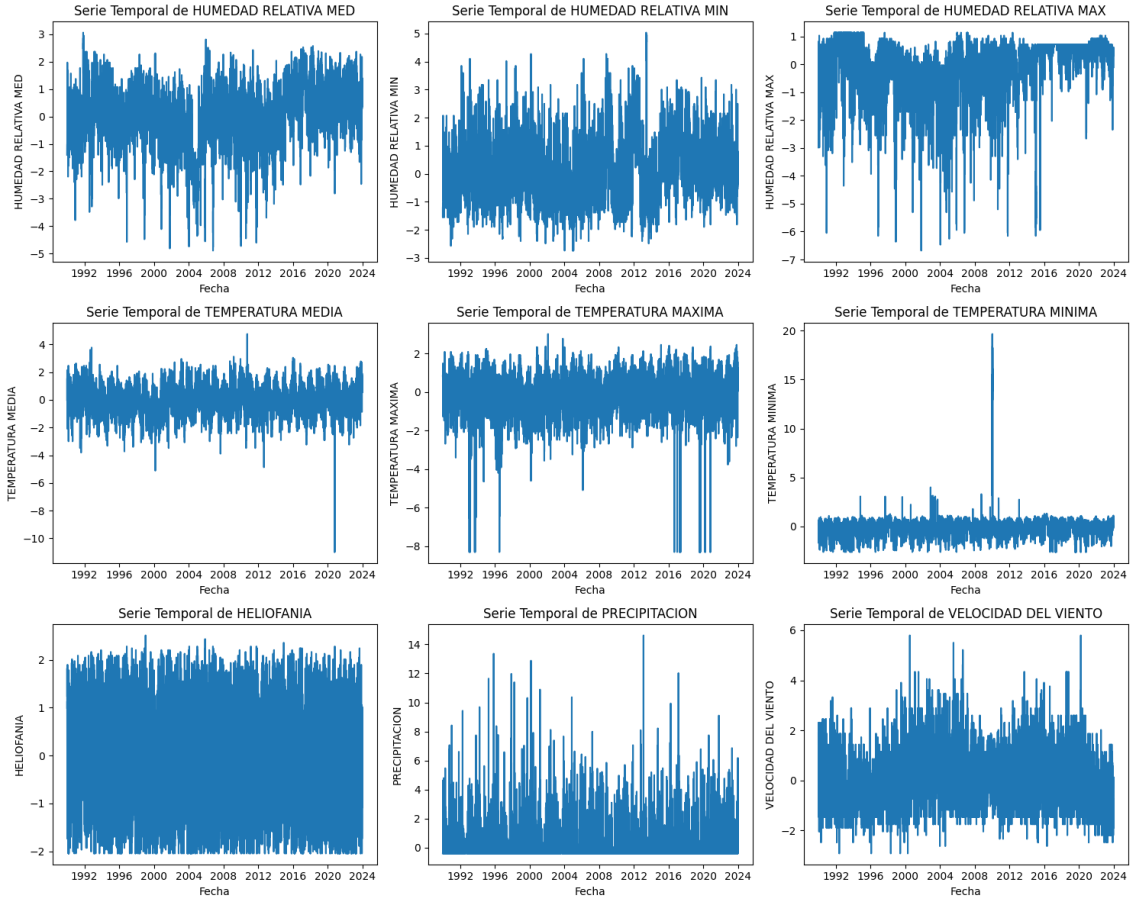
HELIOFANIA PRECIPITACION VELOCIDAD DEL VIENTO
FECHA
1990-01-01 1.247592 -0.40509 1.872684
1990-01-02 1.821731 -0.40509 -0.017094
1990-01-03 0.979660 -0.40509 -0.017094
1990-01-04 1.132764 -0.40509 0.419009
1990-01-05 0.788280 -0.40509 0.419009

```
# Crear boxplots para cada variable numérica despues de la estandarización
data.select_dtypes(include=["float64", "int64"]).plot(
    kind="box",
    subplots=True,
    layout=(4, 4),
    figsize=(15, 10),
    title="Detección de Outliers",
)
plt.suptitle("Detección de Outliers en Variables Numéricas")
plt.show()
```



Identificacion de outliers

```
# en un solo plot
# Número de columnas numéricas
num_columns = len(numeric_columns)

# Determinar el tamaño de la cuadrícula para los subplots
rows = (num_columns + 2) // 3  # Número de filas, con 3 gráficos por fila

# Crear un solo plot con subplots para cada serie temporal
plt.figure(figsize=(15, 4 * rows))

for i, column in enumerate(numeric_columns, 1):
    plt.subplot(rows, 3, i)  # Crear subplot en la posición correspondiente
    sns.lineplot(data=data, x=data.index, y=column)
    plt.title(f"Serie Temporal de {column}")
    plt.xlabel("Fecha")
    plt.ylabel(column)

plt.tight_layout()  # Ajustar el espaciado para evitar solapamientos
plt.show()
```




```
# Calcular estadísticas descriptivas
descriptive_stats = data[numeric_columns].describe()

# Definir límites para detectar outliers
iqr = descriptive_stats.loc["75%"] - descriptive_stats.loc["25%"]
lower_bound = descriptive_stats.loc["25%"] - 1.5 * iqr
```

```
upper_bound = descriptive_stats.loc["75%"] + 1.5 * iqr

# Mostrar los límites
print("Límites Inferiores para detectar outliers:")
print(lower_bound)
print("\nLímites Superiores para detectar outliers:")
print(upper_bound)
print("\nEstadisticas descriptivas:")
print(descriptive_stats)
```

```
Límites Inferiores para detectar outliers:
HUMEDAD RELATIVA MED   -2.518703
HUMEDAD RELATIVA MIN   -2.618786
HUMEDAD RELATIVA MAX   -1.664264
TEMPERATURA MEDIA      -2.588010
TEMPERATURA MAXIMA     -2.094361
TEMPERATURA MINIMA     -1.491050
HELIOFANIA             -2.976161
PRECIPITACION          -1.006329
VELOCIDAD DEL VIENTO   -2.706394
dtype: float64

Límites Superiores para detectar outliers:
HUMEDAD RELATIVA MED    2.604301
HUMEDAD RELATIVA MIN    2.454025
HUMEDAD RELATIVA MAX    2.139706
TEMPERATURA MEDIA       2.636924
TEMPERATURA MAXIMA      2.241922
TEMPERATURA MINIMA      1.622030
HELIOFANIA              2.979579
PRECIPITACION           0.596975
VELOCIDAD DEL VIENTO    2.526838
dtype: float64

Estadisticas descriptivas:
       HUMEDAD RELATIVA MED  HUMEDAD RELATIVA MIN  HUMEDAD RELATIVA
MAX  \
count          1.241800e+04          1.241800e+04          1.241800e+04
mean          -1.098600e-15         -3.021151e-16          1.190150e-15
std            1.000040e+00          1.000040e+00          1.000040e+00
min           -4.896532e+00         -2.745607e+00         -6.683391e+00
25%           -5.975764e-01         -7.164820e-01         -2.377752e-01
50%            1.797838e-02         -4.010712e-02          2.905540e-01
75%            6.831747e-01          5.517209e-01          7.132173e-01
max            3.065968e+00          5.032704e+00          1.135881e+00

       TEMPERATURA MEDIA  TEMPERATURA MAXIMA  TEMPERATURA MINIMA
\
count       1.241800e+04        1.241800e+04        1.241800e+04
mean        5.000921e-16       -2.471851e-16        7.781753e-17
std         1.000040e+00        1.000040e+00        1.000040e+00
min        -1.099689e+01       -8.297655e+00       -2.627324e+00
```

25%	-6.286598e-01	-4.682553e-01	-3.236450e-01
50%	2.445699e-02	5.118276e-02	1.028470e-01
75%	6.775738e-01	6.158154e-01	4.546251e-01
max	4.759554e+00	3.024862e+00	1.966233e+01

	HELIOFANIA	PRECIPITACION	VELOCIDAD DEL VIENTO
count	1.241800e+04	1.241800e+04	1.241800e+04
mean	-1.831001e-17	-2.975376e-17	8.697254e-17
std	1.000040e+00	1.000040e+00	1.000040e+00
min	-2.044141e+00	-4.050900e-01	-2.924445e+00
25%	-7.427582e-01	-4.050900e-01	-7.439318e-01
50%	-3.698417e-03	-4.050900e-01	-1.709400e-02
75%	7.461767e-01	-4.263819e-03	5.643763e-01
max	2.510698e+00	1.461959e+01	5.797609e+00

Modelo con outliers

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Seleccionar una variable objetivo y las características
X = data[numeric_columns].drop("TEMPERATURA MEDIA", axis=1)
y = data["TEMPERATURA MEDIA"]

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)

# Entrenar el modelo
model = LinearRegression()
model.fit(X_train, y_train)

# Evaluar el modelo
y_pred = model.predict(X_test)
print("Error cuadrático medio (MSE) con outliers:", mean_squared_error(y_test, y_pred))
```

Error cuadrático medio (MSE) con outliers: 0.607191295567709

Modelo sin Outliers

```
import numpy as np

# Identificación de outliers usando el rango intercuartílico (IQR)
Q1 = data[numeric_columns].quantile(0.25)
Q3 = data[numeric_columns].quantile(0.75)
IQR = Q3 - Q1

# Definir límites para detectar outliers
lower_bound = Q1 - 1.0 * IQR
upper_bound = Q3 + 1.0 * IQR

# Filtrar outliers
filtered_data = data[
```

```
    ~(
        (data[numeric_columns] < lower_bound) | (data[numeric_columns] > upper_bound)
    ).any(axis=1)
]

# Verificar el número de muestras después del filtrado
print("Número de muestras originales:", data.shape[0])
print("Número de muestras después del filtrado:", filtered_data.shape[0])

# Verificar si hay suficientes muestras para el split
if filtered_data.shape[0] > 0:
    # Redefinir X y y sin outliers
    X_filtered = filtered_data[numeric_columns].drop("TEMPERATURA MEDIA", axis=1)
    y_filtered = filtered_data["TEMPERATURA MEDIA"]

    # Dividir los datos filtrados en conjuntos de entrenamiento y prueba
    X_train_filtered, X_test_filtered, y_train_filtered, y_test_filtered = (
        train_test_split(X_filtered, y_filtered, test_size=0.3, random_state=42)
    )

    # Entrenar el modelo sin outliers
    model_filtered = LinearRegression()
    model_filtered.fit(X_train_filtered, y_train_filtered)

    # Evaluar el modelo sin outliers
    y_pred_filtered = model_filtered.predict(X_test_filtered)
    print(
        "Error cuadrático medio (MSE) sin outliers:",
        mean_squared_error(y_test_filtered, y_pred_filtered),
    )
else:
    print(
        "No hay suficientes muestras después de filtrar los outliers para realizar la división de entrenamiento y prueba."
    )
Número de muestras originales: 12418
Número de muestras después del filtrado: 7498
Error cuadrático medio (MSE) sin outliers: 0.2859160495587039
```

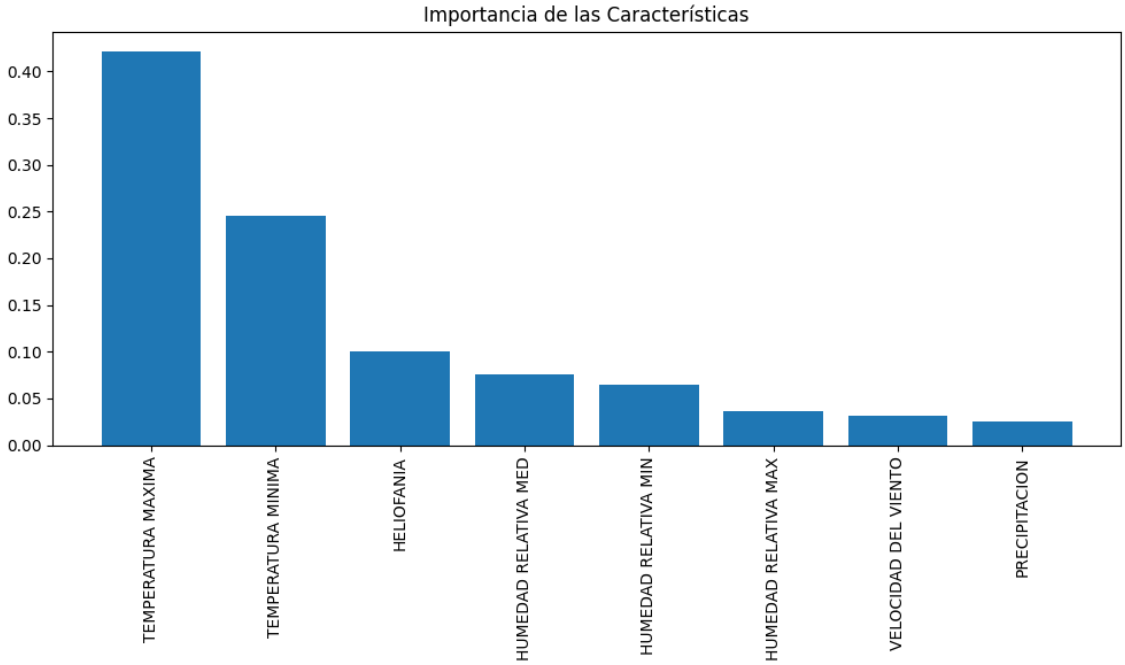
Aplicar random forest

```
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import numpy as np
import matplotlib.pyplot as plt

# Identificación de outliers usando el rango intercuartílico (IQR)
Q1 = data[numeric_columns].quantile(0.25)
Q3 = data[numeric_columns].quantile(0.75)
```

```
IQR = Q3 - Q1

# Definir límites para detectar outliers
lower_bound = Q1 - 1.0 * IQR
upper_bound = Q3 + 1.0 * IQR

# Filtrar outliers
filtered_data = data[
    ~(
        (data[numeric_columns] < lower_bound) | (data[numeric_columns] > upper_bound)
    ).any(axis=1)
]

# Verificar el número de muestras después del filtrado
print("Número de muestras originales:", data.shape[0])
print("Número de muestras después del filtrado:", filtered_data.shape[0])

# Redefinir X y y sin outliers
X_filtered = filtered_data[numeric_columns].drop("TEMPERATURA MEDIA", axis=1)
y_filtered = filtered_data["TEMPERATURA MEDIA"]

# Dividir los datos filtrados en conjuntos de entrenamiento y prueba
X_train_filtered, X_test_filtered, y_train_filtered, y_test_filtered = train_test_split(
    X_filtered, y_filtered, test_size=0.3, random_state=42
)

# Implementar Random Forest Regressor con hiperparámetros ajustados
rf = RandomForestRegressor(
    n_estimators=500,  # Aumentar el número de árboles
    max_depth=20,  # Limitar la profundidad máxima de los árboles
    min_samples_split=5,  # Número mínimo de muestras necesarias para dividir un nodo
    min_samples_leaf=4,  # Número mínimo de muestras que debe tener una hoja
    max_features="sqrt",  # Limitar el número de características consideradas para dividir un nodo
    bootstrap=True,  # Muestreo con reemplazo
    random_state=42,
)

# Entrenar el modelo con los datos de entrenamiento
rf.fit(X_train_filtered, y_train_filtered)

# Evaluar el modelo
y_pred_rf = rf.predict(X_test_filtered)
mse_rf = mean_squared_error(y_test_filtered, y_pred_rf)
print("Error cuadrático medio (MSE) con Random Forest:", mse_rf)

# Visualizar Importancia de las Características
importances = rf.feature_importances_
indices = np.argsort(importances)[::-1]
```

```
plt.figure(figsize=(10, 6))
plt.title("Importancia de las Características")
plt.bar(range(X_train_filtered.shape[1]), importances[indices], align="center")
plt.xticks(
    range(X_train_filtered.shape[1]), X_train_filtered.columns[indices], rotation=90
)
plt.tight_layout()
plt.show()
```

```
Número de muestras originales: 12418
Número de muestras después del filtrado: 7498
Error cuadrático medio (MSE) con Random Forest: 0.26656689719068305
```

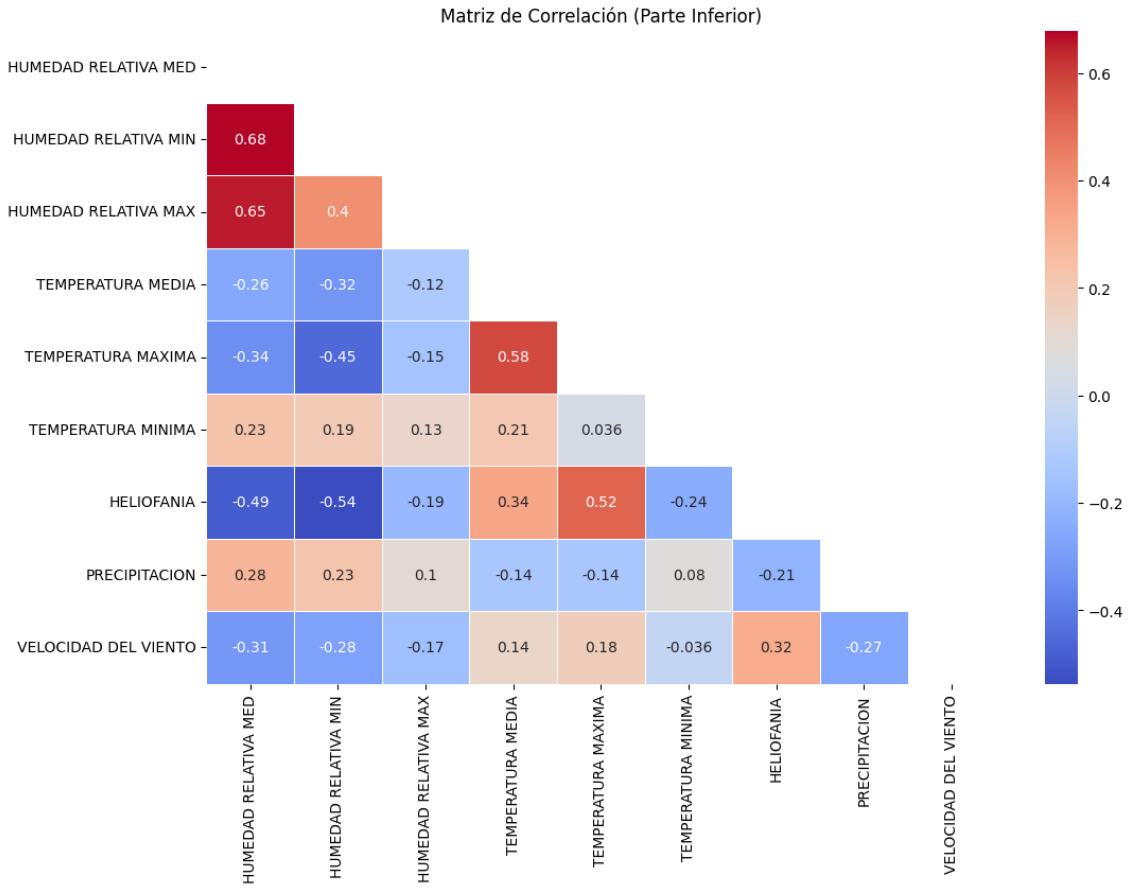



```
# Generar una Tabla con los Estadísticos

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score


# Calcular los estadísticos
mse_rf = mean_squared_error(y_test_filtered, y_pred_rf)
mae_rf = mean_absolute_error(y_test_filtered, y_pred_rf)
r2_rf = r2_score(y_test_filtered, y_pred_rf)

# Crear un diccionario con los estadísticos
resultados = {
    "Modelo": ["Random Forest"],
    "MSE": [mse_rf],
    "MAE": [mae_rf],
    "R²": [r2_rf],
}

# Convertir el diccionario a un DataFrame
resultados_df = pd.DataFrame(resultados)
```

# Mostrar la tabla con los estadísticos
print(resultados_df)
```
         Modelo       MSE       MAE        R²
0  Random Forest  0.266567  0.397061  0.602614
```
# exporto datos sin outiers
data = filtered_data
# Guardar el DataFrame filtrado en un nuevo archivo CSV
filtered_data.to_csv("Datos_sin_outiers_ajustado.csv", index=True)

Matriz de corelacion sin outliers

```
import numpy as np

# Calcular la matriz de correlación

correlation_matrix = data.corr()

# Crear una máscara para la parte superior de la matriz de correlación
mask = np.triu(np.ones_like(correlation_matrix, dtype=bool))

# Crear un heatmap de la matriz de correlación, mostrando solo la parte inferior
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, mask=mask, annot=True, cmap="coolwarm", linewidths=0.5)
plt.title("Matriz de Correlación (Parte Inferior)")
plt.show()
```



Crear variables temporales para analisis de series temporales

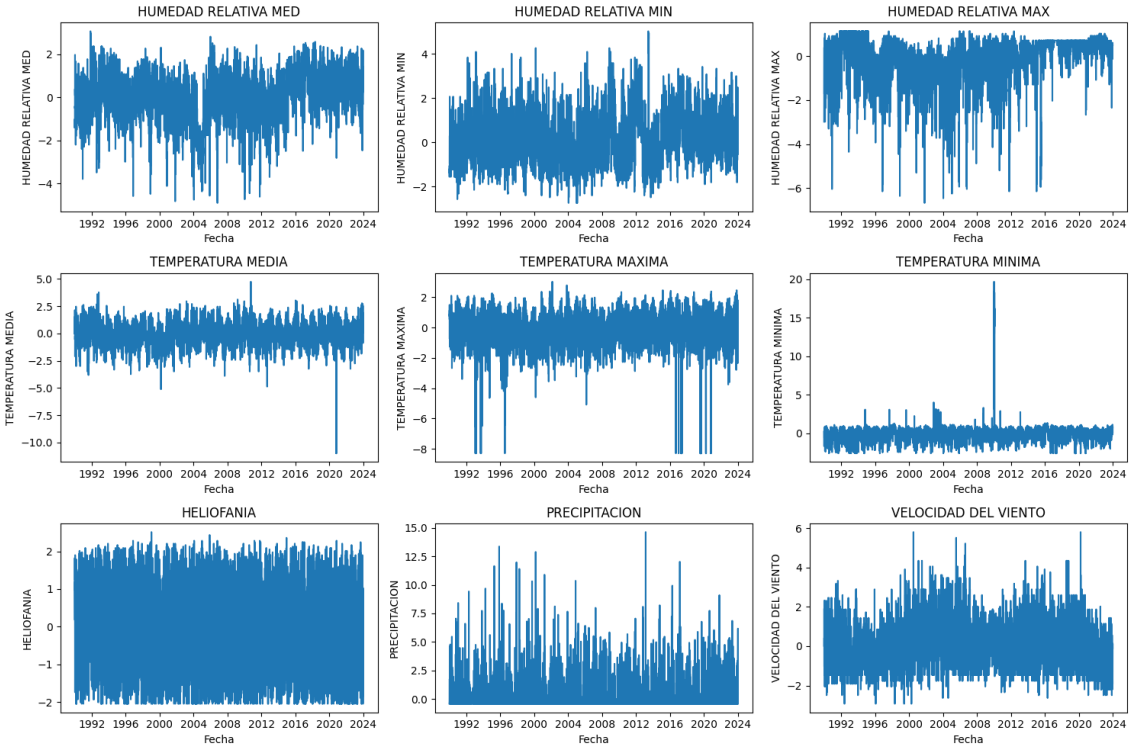
```
# Derivar variables temporales
data["year"] = data.index.year
data["month"] = data.index.month
data["day"] = data.index.day
data["dayofweek"] = data.index.dayofweek
data["weekofyear"] = data.index.isocalendar().week

# Verificar las nuevas columnas
print(data.head())
```

```
            HUMEDAD RELATIVA MED  HUMEDAD RELATIVA MIN  HUMEDAD RELATIVA MAX  \
FECHA
1990-01-01             -1.401769             -1.223763             -2.985087
1990-01-02             -1.004637             -1.561951              0.396220
1990-01-03             -1.401769             -1.223763             -1.611431
1990-01-04             -1.203203             -1.054669             -0.343441
1990-01-05             -1.004637             -1.392857              0.396220

            TEMPERATURA MEDIA  TEMPERATURA MAXIMA  TEMPERATURA MINIMA  \
FECHA
1990-01-01           1.493970            0.856720            0.174448
1990-01-02           0.677574            0.856720           -0.012337
1990-01-03          -0.057183            0.736268           -1.693400
1990-01-04           0.677574            0.736268           -0.915130
1990-01-05           1.412330            1.338529            0.298971

            HELIOFANIA  PRECIPITACION  VELOCIDAD DEL VIENTO  year  month  \
FECHA
1990-01-01    1.247592       -0.40509              1.872684  1990      1
1990-01-02    1.821731       -0.40509             -0.017094  1990      1
1990-01-03    0.979660       -0.40509             -0.017094  1990      1
1990-01-04    1.132764       -0.40509              0.419009  1990      1
1990-01-05    0.788280       -0.40509              0.419009  1990      1

            day  dayofweek  weekofyear
FECHA
1990-01-01    1          0           1
1990-01-02    2          1           1
1990-01-03    3          2           1
1990-01-04    4          3           1
1990-01-05    5          4           1
```

Visualizar graficos de series temporales

```
# Seleccionar variables para visualizar
variables_to_plot = [
    "HUMEDAD RELATIVA MED",
    "HUMEDAD RELATIVA MIN",
    "HUMEDAD RELATIVA MAX",
    "TEMPERATURA MEDIA",
    "TEMPERATURA MAXIMA",
```

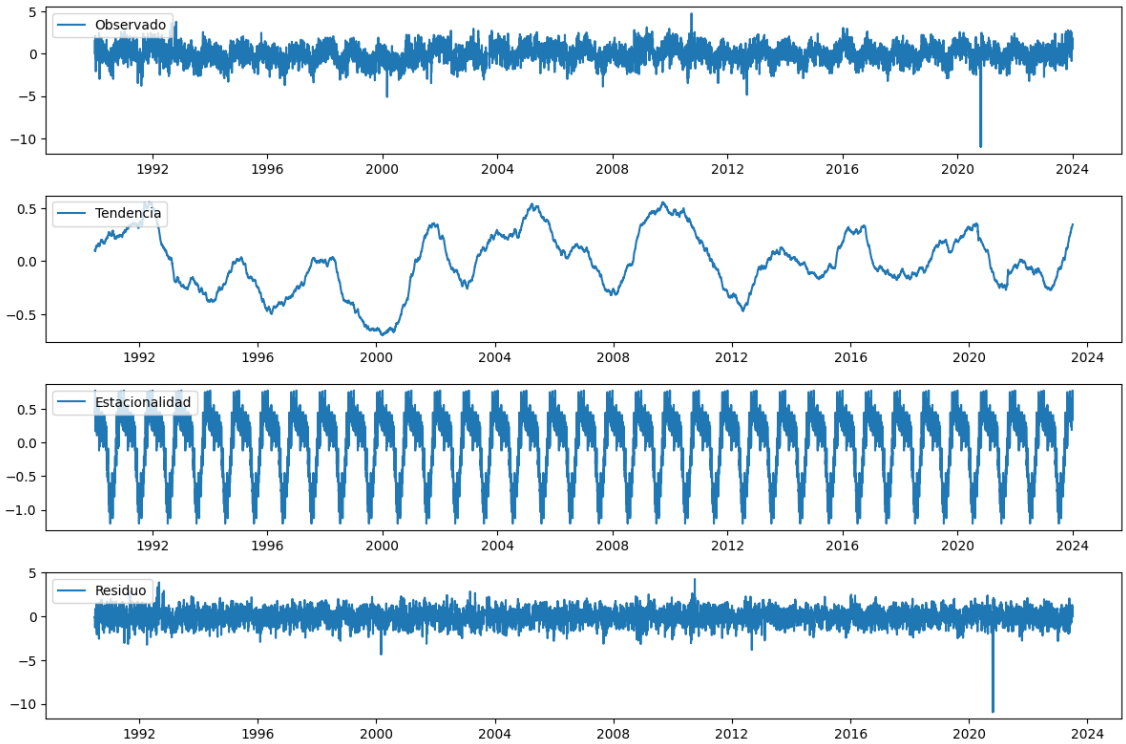
```
    "TEMPERATURA MINIMA",
    "HELIOFANIA",
    "PRECIPITACION",
    "VELOCIDAD DEL VIENTO",
]

# Crear gráficos de series temporales para cada variable
plt.figure(figsize=(15, 10))
for i, var in enumerate(variables_to_plot, 1):
    plt.subplot(3, 3, i)
    plt.plot(data.index, data[var])
    plt.title(var)
    plt.xlabel("Fecha")
    plt.ylabel(var)
    plt.tight_layout()

plt.show()
```



Separacion de componentes

```
from statsmodels.tsa.seasonal import seasonal_decompose

# Seleccionar la variable para la descomposición
variable = "TEMPERATURA MEDIA"

# Realizar la descomposición estacional
result = seasonal_decompose(data[variable], model="additive", period=365)

# Graficar los componentes
plt.figure(figsize=(12, 8))
plt.subplot(411)
plt.plot(result.observed, label="Observado")
```

```
plt.legend(loc="upper left")
plt.subplot(412)
plt.plot(result.trend, label="Tendencia")
plt.legend(loc="upper left")
plt.subplot(413)
plt.plot(result.seasonal, label="Estacionalidad")
plt.legend(loc="upper left")
plt.subplot(414)
plt.plot(result.resid, label="Residuo")
plt.legend(loc="upper left")
plt.tight_layout()
plt.show()
```



Aplicación modelo VAR

```
from statsmodels.tsa.api import VAR

# Seleccionar las variables para el modelo VAR
variables = [
    "HUMEDAD RELATIVA MED",
    "HUMEDAD RELATIVA MIN",
    "HUMEDAD RELATIVA MAX",
    "TEMPERATURA MEDIA",
    "TEMPERATURA MAXIMA",
    "TEMPERATURA MINIMA",
    "HELIOFANIA",
    "PRECIPITACION",
    "VELOCIDAD DEL VIENTO",
]

# Preparar los datos para el modelo VAR
model_data = data[variables].dropna()
```

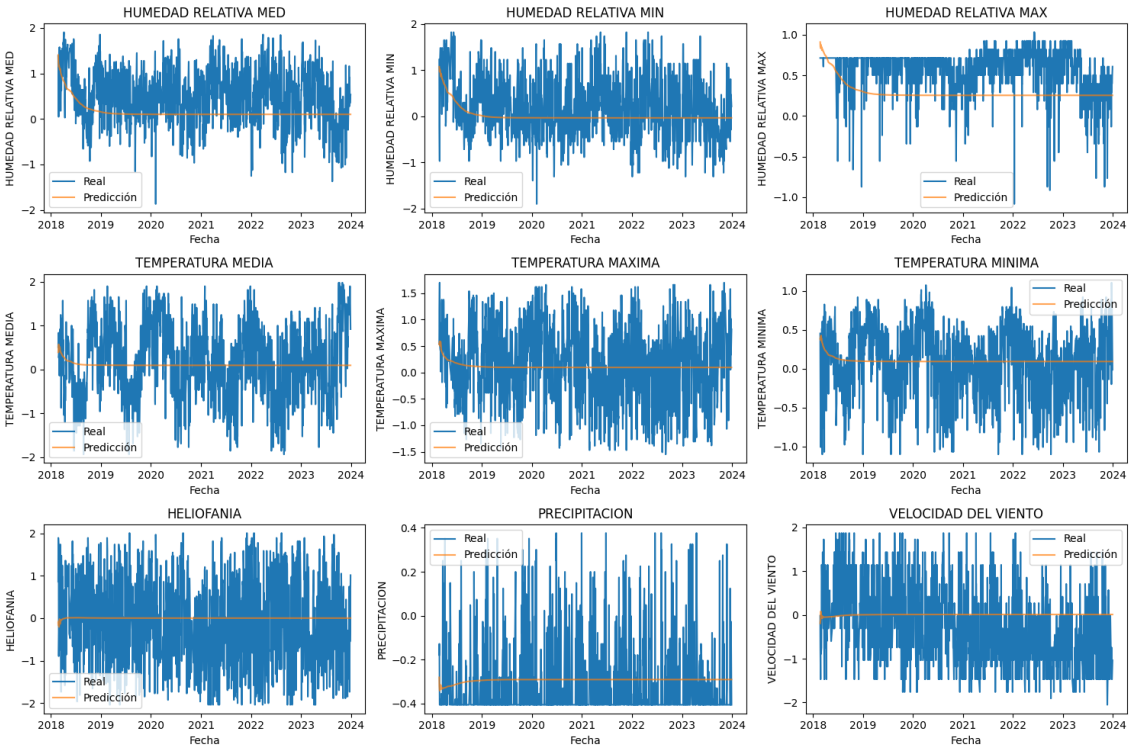
```
# Crear el modelo VAR
model = VAR(model_data)

# Ajustar el modelo
results = model.fit(maxlags=15, ic="aic")

# Verificar la estabilidad del modelo
if not results.is_stable():
    print(
        "El modelo no es estable, considera ajustar los lags o verificar la estacionariedad de los datos."
    )

# Resumen del modelo
print(results.summary())

# Predicciones
forecast_input = model_data.values[-results.k_ar :]
forecast = results.forecast(y=forecast_input, steps=10)
forecast_df = pd.DataFrame(
    forecast,
    index=pd.date_range(start=model_data.index[-1], periods=10, freq="D"),
    columns=variables,
)

print(forecast_df)
```

Evaluar el modelo

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score


# Dividir los datos en entrenamiento y prueba
train_size = int(len(model_data) * 0.8)
train, test = model_data[:train_size], model_data[train_size:]

# Realizar predicciones en el conjunto de prueba
lag_order = results.k_ar
predictions = results.forecast(train.values[-lag_order:], steps=len(test))

# Convertir las predicciones a un DataFrame
predictions_df = pd.DataFrame(predictions, index=test.index, columns=variables)

# Calcular métricas de error para cada variable
mse = {}
mae = {}
r2 = {}

for var in test.columns:
    mse[var] = mean_squared_error(test[var], predictions_df[var])
```

```
    mae[var] = mean_absolute_error(test[var], predictions_df[var])
    r2[var] = r2_score(test[var], predictions_df[var])

# Imprimir las métricas de evaluación
print("Mean Squared Error (MSE):")
print(mse)

print("\nMean Absolute Error (MAE):")
print(mae)

print("\nR-squared (R²):")
print(r2)

# Visualización de las predicciones frente a los valores reales
plt.figure(figsize=(15, 10))

for i, var in enumerate(test.columns, 1):
    plt.subplot(3, 3, i)
    plt.plot(test.index, test[var], label="Real")
    plt.plot(predictions_df.index, predictions_df[var], label="Predicción", alpha=0.7)
    plt.title(var)
    plt.xlabel("Fecha")
    plt.ylabel(var)
    plt.legend()
    plt.tight_layout()

plt.show()
```

```
Mean Squared Error (MSE):
{'HUMEDAD RELATIVA MED': 0.43274447494193713, 'HUMEDAD RELATIVA MIN': 0.3737130531534922, 'HUMEDAD RELATIVA MAX': 0.15236297288331666, 'TEMPERATURA MEDIA': 0.6725304030413678, 'TEMPERATURA MAXIMA': 0.4965616187824628, 'TEMPERATURA MINIMA': 0.1994372811518843, 'HELIOFANIA': 0.9696154663798722, 'PRECIPITACION': 0.0265852893261633, 'VELOCIDAD DEL VIENTO': 0.7630075527374186}

Mean Absolute Error (MAE):
{'HUMEDAD RELATIVA MED': 0.5389216710158258, 'HUMEDAD RELATIVA MIN': 0.4781014426401661, 'HUMEDAD RELATIVA MAX': 0.34628628830767366, 'TEMPERATURA MEDIA': 0.6769667871195522, 'TEMPERATURA MAXIMA': 0.5807199346263112, 'TEMPERATURA MINIMA': 0.35569906412040253, 'HELIOFANIA': 0.8208025690472748, 'PRECIPITACION': 0.12716931557019867, 'VELOCIDAD DEL VIENTO': 0.722067713530053}

R-squared (R²):
{'HUMEDAD RELATIVA MED': -0.33377846938831035, 'HUMEDAD RELATIVA MIN': -0.013613643733494163, 'HUMEDAD RELATIVA MAX': -1.2575303309152406, 'TEMPERATURA MEDIA': -0.0074129499392801, 'TEMPERATURA MAXIMA': -0.0039056375550849953, 'TEMPERATURA MINIMA': -0.008269370930288344, 'HELIOFANIA': -0.019461487719163006, 'PRECIPITACION': -0.03811743964374381, 'VELOCIDAD DEL VIENTO': -0.09366118873573548}
```



VAR CON DATOS DE PRUEBA

```
from statsmodels.tsa.api import VAR
import pandas as pd
from sklearn.metrics import mean_squared_error

# Seleccionar las variables para el modelo VAR
variables = [
    "HUMEDAD RELATIVA MED",
    "HUMEDAD RELATIVA MIN",
    "HUMEDAD RELATIVA MAX",
    "TEMPERATURA MEDIA",
    "TEMPERATURA MAXIMA",
    "TEMPERATURA MINIMA",
    "HELIOFANIA",
    "PRECIPITACION",
    "VELOCIDAD DEL VIENTO",
]

# Preparar los datos para el modelo VAR
model_data = data[variables].dropna()

# Dividir los datos en entrenamiento y prueba (ejemplo: 80% entrenamiento, 20% prueba)
train_size = int(len(model_data) * 0.8)
train_data = model_data.iloc[:train_size]
test_data = model_data.iloc[train_size:]

# Crear el modelo VAR con el conjunto de entrenamiento
model = VAR(train_data)
```

```
# Ajustar el modelo
results = model.fit(maxlags=15, ic="aic")

# Realizar predicciones en el conjunto de prueba
forecast_input = train_data.values[-results.k_ar :]
forecast = results.forecast(y=forecast_input, steps=len(test_data))
forecast_df = pd.DataFrame(forecast, index=test_data.index, columns=variables)

# Evaluar el modelo comparando las predicciones con los datos reales del conjunto
de prueba
mse = mean_squared_error(test_data, forecast_df)
print(f"Error cuadrático medio (MSE) en el conjunto de prueba: {mse}")

# Mostrar las predicciones y los valores reales
print(forecast_df)
print(test_data)
```

modelo LSTM

```
#!pip install tensorflow --timeout 1000
from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score


# Seleccionar las variables para el modelo LSTM

variables = [

    "HUMEDAD RELATIVA MED",

    "HUMEDAD RELATIVA MIN",

    "HUMEDAD RELATIVA MAX",

    "TEMPERATURA MEDIA",

    "TEMPERATURA MAXIMA",

    "TEMPERATURA MINIMA",

    "HELIOFANIA",

    "PRECIPITACION",

    "VELOCIDAD DEL VIENTO",

]
```

```
# Preparar los datos para el modelo LSTM
model_data = data[variables].dropna()

# Normalizar los datos
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(model_data)

# Crear conjuntos de entrenamiento y prueba
train_size = int(len(scaled_data) * 0.8)
train_data = scaled_data[:train_size]
test_data = scaled_data[train_size:]

# Función para crear los conjuntos de datos para LSTM
def create_dataset(dataset, time_step=1):
    X, Y = [], []
    for i in range(len(dataset) - time_step - 1):
        a = dataset[i : (i + time_step), :]
        X.append(a)
        Y.append(dataset[i + time_step, :])
    return np.array(X), np.array(Y)

# Crear los conjuntos de datos para LSTM
time_step = 10
X_train, Y_train = create_dataset(train_data, time_step)
X_test, Y_test = create_dataset(test_data, time_step)

# Remodelar los datos para que sean compatibles con LSTM
```

```
X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], len(variables))

X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], len(variables))
```

Entremiento STML

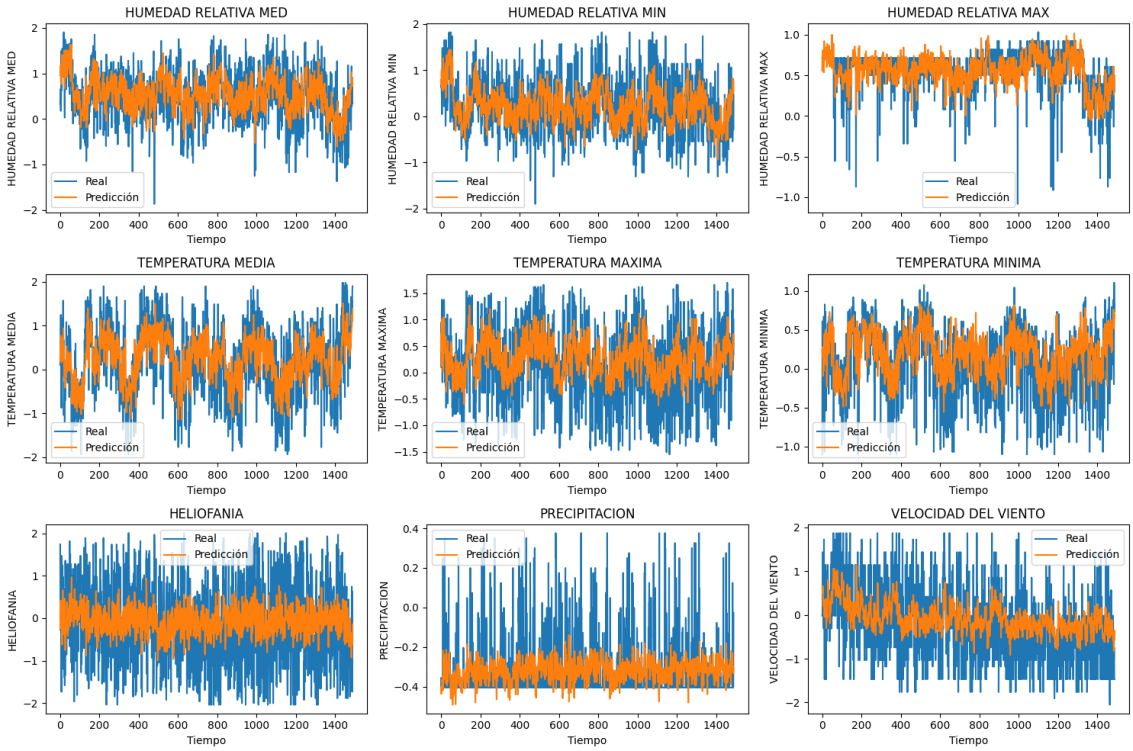
```
# Crear el modelo LSTM
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(time_step, len(variables))))
model.add(LSTM(50, return_sequences=False))
model.add(Dense(len(variables)))

# Compilar el modelo
model.compile(optimizer="adam", loss="mean_squared_error")

# Entrenar el modelo
history = model.fit(
    X_train,
    Y_train,
    epochs=100,
    batch_size=64,
    validation_data=(X_test, Y_test),
    verbose=1,
)

# Realizar predicciones
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)

# Desnormalizar las predicciones y los valores reales
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
Y_train = scaler.inverse_transform(Y_train)
Y_test = scaler.inverse_transform(Y_test)
```

Evaluacion STML

```
# Calcular métricas de evaluación para LSTM
mse = mean_squared_error(Y_test, test_predict)
mae = mean_absolute_error(Y_test, test_predict)
r2 = r2_score(Y_test, test_predict)

print("LSTM Mean Squared Error (MSE):", mse)
print("LSTM Mean Absolute Error (MAE):", mae)
print("LSTM R-squared (R²):", r2)

# Visualizar las predicciones frente a los valores reales
plt.figure(figsize=(15, 10))
for i in range(len(variables)):
    plt.subplot(3, 3, i + 1)
    plt.plot(Y_test[:, i], label="Real")
```

```
    plt.plot(test_predict[:, i], label="Predicción")
    plt.title(variables[i])
    plt.xlabel("Tiempo")
    plt.ylabel(variables[i])
    plt.legend()
    plt.tight_layout()

plt.show()
```

LSTM Mean Squared Error (MSE): 0.36167729996504466
LSTM Mean Absolute Error (MAE): 0.43686036745238516
LSTM R-squared (R^2): 0.13877062060759204



Mejorando el modelo SLTM

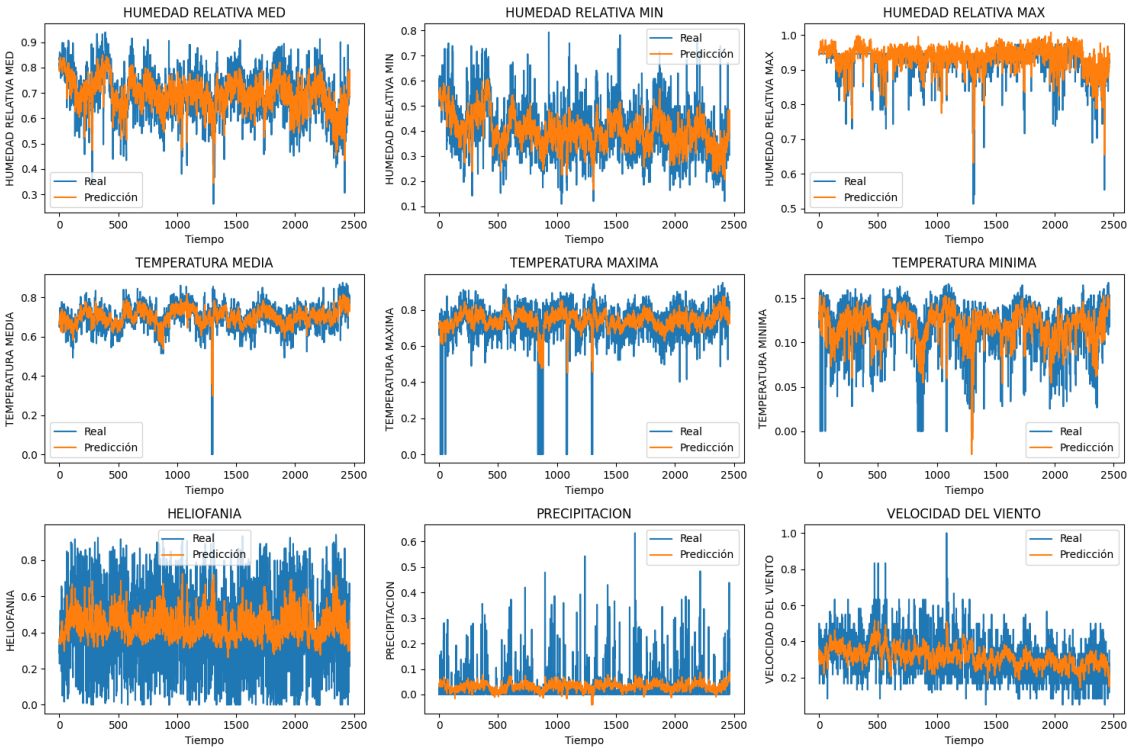
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# Seleccionar las variables para el modelo LSTM
variables = [
    "HUMEDAD RELATIVA MED",
    "HUMEDAD RELATIVA MIN",
    "HUMEDAD RELATIVA MAX",
    "TEMPERATURA MEDIA",
    "TEMPERATURA MAXIMA",
    "TEMPERATURA MINIMA",
```

```
    "HELIOFANIA",
    "PRECIPITACION",
    "VELOCIDAD DEL VIENTO",
]

# Preparar los datos para el modelo LSTM
model_data = data[variables].dropna()

# Normalizar los datos
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(model_data)

# Crear conjuntos de entrenamiento y prueba
train_size = int(len(scaled_data) * 0.8)
train_data = scaled_data[:train_size]
test_data = scaled_data[train_size:]


# Función para crear los conjuntos de datos para LSTM
def create_dataset(dataset, time_step=1):
    X, Y = [], []
    for i in range(len(dataset) - time_step - 1):
        a = dataset[i : (i + time_step), :]
        X.append(a)
        Y.append(dataset[i + time_step, :])
    return np.array(X), np.array(Y)


# Ajuste del time_step
time_step = 20  # Aumentamos el tamaño de la ventana temporal
X_train, Y_train = create_dataset(train_data, time_step)
X_test, Y_test = create_dataset(test_data, time_step)

# Remodelar los datos para que sean compatibles con LSTM
X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], len(variables))
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], len(variables))

# Crear el modelo LSTM con ajustes
model = Sequential()
model.add(LSTM(100,          return_sequences=True,          input_shape=(time_step,
len(variables))))
model.add(Dropout(0.2))  # Añadir Dropout para evitar sobreajuste
model.add(LSTM(100, return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(len(variables)))

# Compilar el modelo con un optimizador Adam ajustado
model.compile(optimizer=Adam(learning_rate=0.001), loss="mean_squared_error")

# Early stopping para evitar sobreentrenamiento
early_stopping = EarlyStopping(
```

```
    monitor="val_loss", patience=10, restore_best_weights=True
)

# Entrenar el modelo
history = model.fit(
    X_train,
    Y_train,
    epochs=100,  # Puedes experimentar con más épocas
    batch_size=64,  # Puedes ajustar este parámetro
    validation_data=(X_test, Y_test),
    verbose=1,
    callbacks=[early_stopping],  # Añadir early stopping
)

# Realizar predicciones
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)
```

Evaluar modelo Mejorado en promedio

```
# Calcular métricas de evaluación para LSTM
mse = mean_squared_error(Y_test, test_predict)
mae = mean_absolute_error(Y_test, test_predict)
r2 = r2_score(Y_test, test_predict)

print("LSTM Mean Squared Error (MSE):", mse)
print("LSTM Mean Absolute Error (MAE):", mae)
print("LSTM R-squared (R²):", r2)

# Visualizar las predicciones frente a los valores reales
plt.figure(figsize=(15, 10))
for i in range(len(variables)):
    plt.subplot(3, 3, i + 1)
    plt.plot(Y_test[:, i], label="Real")
    plt.plot(test_predict[:, i], label="Predicción")
    plt.title(variables[i])
    plt.xlabel("Tiempo")
    plt.ylabel(variables[i])
    plt.legend()
    plt.tight_layout()

plt.show()
LSTM Mean Squared Error (MSE): 0.009504278426456852
LSTM Mean Absolute Error (MAE): 0.06077209012516628
LSTM R-squared (R²): 0.26115692010521396
```



Evalaur Modelo mejorado

```
# Desnormalizar las predicciones y los valores reales
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
Y_train = scaler.inverse_transform(Y_train)
Y_test = scaler.inverse_transform(Y_test)

# Calcular métricas de evaluación para LSTM por cada variable
mse = {}
mae = {}
r2 = {}

for i, var in enumerate(variables):
    mse[var] = mean_squared_error(Y_test[:, i], test_predict[:, i])
    mae[var] = mean_absolute_error(Y_test[:, i], test_predict[:, i])
    r2[var] = r2_score(Y_test[:, i], test_predict[:, i])


mape = {}

for i, var in enumerate(variables):
    mape[var] = (
        np.mean(np.abs((Y_test[:, i] - test_predict[:, i]) / Y_test[:, i])) * 100
    )

# Imprimir MAPE por cada variable
print("LSTM Mean Absolute Percentage Error (MAPE):")
print(mape)
```

```
# Imprimir las métricas de evaluación por variable
print("LSTM Mean Squared Error (MSE):")
print(mse)

print("\nLSTM Mean Absolute Error (MAE):")
print(mae)

print("\nLSTM R-squared (R²):")
print(r2)

# Visualización de las predicciones frente a los valores reales
plt.figure(figsize=(15, 10))
for i in range(len(variables)):
    plt.subplot(3, 3, i + 1)
    plt.plot(Y_test[:, i], label="Real")
    plt.plot(test_predict[:, i], label="Predicción")
    plt.title(variables[i])
    plt.xlabel("Tiempo")
    plt.ylabel(variables[i])
    plt.legend()
    plt.tight_layout()

plt.show()

# Visualizar la pérdida durante el entrenamiento
plt.figure(figsize=(10, 5))
plt.plot(history.history["loss"], label="Train Loss")
plt.plot(history.history["val_loss"], label="Validation Loss")
plt.title("Pérdida durante el entrenamiento")
plt.xlabel("Épocas")
plt.ylabel("Pérdida")
plt.legend()
plt.show()
# Imprimir las métricas de evaluación por variable
print("LSTM Mean Squared Error (MSE):")
print(mse)

print("\nLSTM Mean Absolute Error (MAE):")
print(mae)

print("\nLSTM R-squared (R²):")
print(r2)
```

```
LSTM Mean Squared Error (MSE):
{'HUMEDAD RELATIVA MED': 0.2989990256597573, 'HUMEDAD RELATIVA MIN': 0.4076273656286469, 'HUMEDAD RELATIVA MAX': 0.0662037938026256, 'TEMPERATURA MEDIA': 0.7397947774183841, 'TEMPERATURA MAXIMA': 1.2677528752251053, 'TEMPERATURA MINIMA': 0.27821809210146453, 'HELIOFANIA': 0.9709460161279081, 'PRECIPITACION': 0.886438218678121, 'VELOCIDAD DEL VIENTO': 0.7106120933108938}

LSTM Mean Absolute Error (MAE):
```

{'HUMEDAD RELATIVA MED': 0.4250564446430068, 'HUMEDAD RELATIVA MIN': 0.4888408578759641, 'HUMEDAD RELATIVA MAX': 0.17315272504533213, 'TEMPERATURA MEDIA': 0.5909663062838618, 'TEMPERATURA MAXIMA': 0.6936641371504915, 'TEMPERATURA MINIMA': 0.3842127602175499, 'HELIOFANIA': 0.8212111804654857, 'PRECIPITACION': 0.567117533464422, 'VELOCIDAD DEL VIENTO': 0.6618793049153543}

LSTM R-squared (R^2):
{'HUMEDAD RELATIVA MED': 0.44593614310074525, 'HUMEDAD RELATIVA MIN': 0.384730143380661, 'HUMEDAD RELATIVA MAX': 0.3071908366471019, 'TEMPERATURA MEDIA': 0.3398936750741808, 'TEMPERATURA MAXIMA': 0.16271760913078626, 'TEMPERATURA MINIMA': 0.28543158000631463, 'HELIOFANIA': 0.10596625945046778, 'PRECIPITACION': 0.05996060829712202, 'VELOCIDAD DEL VIENTO': 0.24241007965803874}