**Reporte**

**Información general**

Nombre del proyecto:

**Desarrollo de benchmark para la Predicción de Generación Energética de Centrales Solares y Eólicas aplicando técnicas de regresión no lineal y algoritmos avanzados**

**Fecha del Reporte:** 05 de Junio de 2024

**Autor:** Ismael Rodríguez Silva

**Objetivos del Proyecto**

Mejorar el benchmark existente para los pronósticos de generación de energía renovable variable (ERV) aplicando técnicas de regresión no lineal y algoritmos avanzados. En la medida que se tengan mejores pronósticos, se reducen los costos totales del sistema y desvíos asociados a ERV.

1. **Recopilación de datos y AED**

La recopilación de datos y el análisis exploratorio se pueden revisar en el Jupyter notebook “Recopilación\_datos.ipynb”. De este archivo se obtienen dos resultados: los datos brutos para la generación solar y para la generación eólica. No me extenderé mucho en esto porque es el mismo del proyecto de clustering.

1. **Preparación de los datos (Preprocesamiento)**

Para ejecutar los modelos, es imperativo que no existan datos faltantes, ya que de lo contrario el modelo generará un error. El código correspondiente a esta etapa se encuentra en el archivo “preparación\_datos.ipynb”.

1. **Eliminación de la primera semana de julio:** Durante el entrenamiento de los modelos, se observó que la ausencia de datos durante una semana completa y la corrección mediante interpolación sesgaban y agravaban el error del pronóstico.
2. **Eliminación de centrales con meses incompletos**: Se revisó la columna “Fecha” y se elaboró un resumen de las centrales con datos faltantes por mes. Se filtraron todas las centrales que presentaban datos incompletos.

Tabla

Descripción generada automáticamente

Resumen Meses Faltantes (Elaboración propia)

1. **Eliminación de centrales por umbral**: Se revisaron las series temporales "Real", "Externo" y "Coordinado". Se definió un umbral de cantidad de datos faltantes a lo largo de estas tres series y se eliminaron las centrales en las que este porcentaje de datos faltantes superaba el 5%. (Las series "Externo" y "Coordinado" son las más importantes, ya que alimentan el modelo de regresión).
2. **Generación Potencial (Transformación de datos)**: Se sumaron las columnas “Real” y “Vertimiento”. Esta será la variable de interés (variable dependiente).
3. **Interpolación para valores faltantes**: Después de disminuir significativamente la cantidad de valores perdidos, se realizó una interpolación por central para rellenar los valores faltantes.
4. **Revisión de distribución:** Luego se revisó que la interpolación no haya modificado significativamente la distribución de los datos.
5. **Eliminación de datos nocturnos en datos solares**: Como no hay generación en estas horas, y para no sobreestimar el error, estos datos fueron eliminados.

Luego de esta serie de pasos, se generaron dos archivos por tecnología (solar y eólica). Uno con los datos interpolados para correr los modelos, llamado “solar\_preprocesado.csv”, y otro con los datos sin interpolar, para calcular correctamente los errores, llamado “solar\_bruto\_filtrado”.

1. **Reconstrucción del modelo Benchmark regresión lineal**

En la actualidad existen 3 fuentes de pronósticos, AWS Truepower, Coordinados y Sistema Experto. El primero corresponde a un servicio externo que entrega pronósticos al Coordinador, el segundo el pronóstico entregado por las empresas Coordinadas, mientras que el Sistema Experto combina los dos primeros pronósticos.

De esta manera, para evaluar la efectividad de los pronósticos de este último sistema, se realiza un modelo de regresión lineal, el cual representa la forma más básica de combinación de las dos fuentes de datos existentes.

El siguiente esquema ilustra los sistemas de pronósticos existentes y el benchmark propuesto.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

La expresión matemática de la regresión lineal implementada corresponde a la siguiente.

Donde,

: Pronóstico Coordinados (no “optimizado”)

Pronósticos AWS Truepower (no optimizado”)

Variable categórica horaria [0,1]

Generación real

Para obtener los pronósticos, se consideró una “ventana rodante” de 60 días para realizar la regresión de la expresión previa, y los coeficientes a, b y c se emplearon para realizar el pronóstico del período t+1. Así, por ejemplo, para realizar el pronóstico del día 30 de septiembre de 2023, se llevó a cabo una regresión, para cada central, utilizando los datos desde el 1 de agosto de 2023 hasta el 29 de septiembre del mismo año; para realizar los pronósticos del día 29 de septiembre de 2023, se realizó una nueva regresión, para cada central, con datos desde el 31 de julio del mismo año hasta el 28 de septiembre, y así sucesivamente, hasta completar una serie de pronósticos desde el 1 de enero de 2023 hasta el 31 de marzo de 2024.

1. **Teoría LSTM y LSTM-SSA**

**4.1 Análisis de Espectro SINGULAR (SSA)**

El Análisis de Espectro Singular (SSA) es un método de estimación espectral no paramétrico que combina elementos del análisis de series temporales, sistemas dinámicos y procesamiento de señales. En la práctica, es muy aplicable en aplicaciones de suavizado y pronóstico.

La idea es descomponer las series y utilizarlas como variables exógenas en el modelo.

En la siguiente imagen podemos ver una ilustración del procedimiento.

Diagrama

Descripción generada automáticamente

Imagen:Flowchart (Stratigakos, Bachoumis, Vita, & Zafiropoulos, 2021)

La forma básica del SSA comprende dos etapas secuenciales, a saber, la etapa de descomposición y la etapa de reconstrucción.

* + 1. Fase de descomposición

1. Creación de la matriz de trayectoria:

* Tenemos la una serie tal como:
* Para aplica SSA primero transformamos esta serie en una matriz llamada matriz de trayectoria. Esta matriz se crea tomando “ventanas” de datos consecutivos. La longitud de cada ventana se llama “dimensión de embebido”.
* Si elegimos una ventana de 3 días, la matriz se vería así:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Imagen: Matriz de trayectoria de Nx3

Cada fila es una ventana de datos consecutivos de la serie temporal

1. Descomposición de vectores singulares (SVD)

* Esta matriz se descompone en una suma de matrices más simples mediante un proceso matemático llamado Descomposición de vectores singulares (SVD). Cada una de estas matrices simples captura una parte de la estructura de los datos originales, como las tendencias ciclos repetitivos (oscilaciones) y el ruido (variaciones aleatorias). La descomposición se resultando en una suma de matrices bidimensionales de rango uno hace de la siguiente manera:

Dibujo de una persona

Descripción generada automáticamente con confianza baja

Donde ​, y  denotan el i-ésimo valor propio, vector singular izquierdo y vector singular derecho, respectivamente, en orden descendente de magnitud. La colección ( ​,  ) se denomina el triplete propio de la matriz X.

* + 1. Fase de reconstrucción

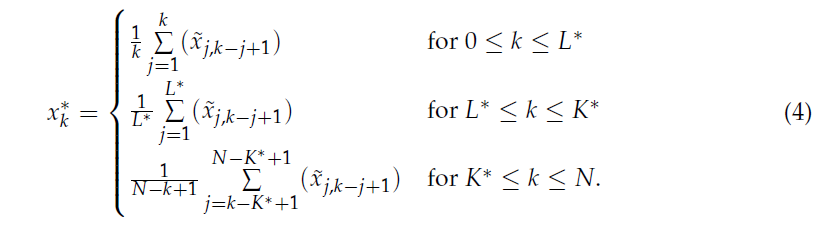
1. Agrupación y suma de componentes: Los componentes descompuestos se agrupan según lo que representan (tendencia, ciclos, ruido) y se suman dentro de cada grupo para crear nuevas matrices. Estas matrices agrupadas se combinan para formar una versión reconstruida de la serie temporal original.

El paso de agrupamiento consiste en particionar los componentes principales (PCs) extraídos producidos por la SVD y sumarlos dentro de cada grupo. El conjunto de índices se particiona en m subconjuntos disjuntos ​, con cada uno consistiendo en un grupo de índices mediante un proceso llamado agrupamiento de tripletes propios. La matriz ​ es una matriz resultante, por lo tanto, la descomposición se puede denotar como sigue:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

1. Promediado diagonal: Para obtener la serie temporal final, las matrices resultantes se transforman nuevamente en una serie temporal mediante un proceso llamado promediado diagonal. Este proceso implica calcular promedios de los elementos a lo largo de las diagonales de la matriz reconstruida.



Donde es la matriz reconstruida a partir de los componentes principales. es el elemento de la nueva serie temporal reconstruida. son los límites que definen diferentes rangos de la matriz. Generalmente esta relacionado con la longitud de la ventana de embebido y con el tamaño de la matriz de trayectoria.

Para el rango , la expresión significa que para los primeros términos, cada es el promedio de los elementos de la diagonal correspondiente a la matriz .

Para el rango , la expresión significa que cada se calcula como el promedio de una cantidad constante de términos.

En el último rango se obtiene promediando sobre términos.

Finalmente, la serie temporal original se expresa como una suma de series temporales.

**Beneficios de la descomposición**

Al utilizar estos componentes como regresores exógenos, mejoras la capacidad del modelo para entender y predecir la serie temporal desde una perspectiva más holística. Esto es especialmente útil en series temporales complejas donde múltiples factores influyen en las observaciones futuras. Además, proporciona una manera de integrar conocimientos específicos del dominio, como la periodicidad y tendencias inherentes, en el modelo de predicción.

**4.2 Long-Short Term Memory (LSTM)**

Las redes neuronales recurrentes (RNN) están diseñadas para modelar eficazmente la dinámica de datos secuenciales. Su diseño se basa en el cálculo recursivo de nuevos estados. Sin embargo, estas son propensas a sufrir el problema del gradiente explosivo o del gradiente que se desvanece durante el entrenamiento por retro propagación, por lo que se dice que tienen memoria a corto plazo.

Diagrama

Descripción generada automáticamenteDiagrama

Descripción generada automáticamente

Las redes LSTM nacen como una solución al problema de memoria a corto plazo de las redes recurrentes simples. Estas poseen una memoria de corto y largo plazo.

Calendario

Descripción generada automáticamenteInterfaz de usuario gráfica, Sitio web

Descripción generada automáticamente con confianza media

La celda de estado es la clave del funcionamiento del LSTM; es como una banda transportadora a la que se pueden añadir y remover datos que se desean mantener o descartar de la memoria de la red. Para añadir o remover datos de esta memoria se utilizan varias compuertas:

* **Forget gate:** Permite eliminar elementos de la memoria.
* **Update gate:** Permite añadir nuevos elementos a la memoria.
* **Output gate:** Permite crear el estado oculto actualizado.

Las compuertas funcionan como válvulas; abiertas permiten el paso de información y cerradas lo bloquean. Cada compuerta está conformada por una red neuronal, una función sigmoidal (que da a la red el comportamiento de válvula) y un elemento multiplicador.

**Forget gate** permite eliminar elementos de la memoria, decidiendo qué información se va a descartar y, por tanto, no pasará a la celda de estado. Para ello, toma el estado oculto anterior y la entrada actual , los transforma y los lleva a la función de activación sigmoidal:

Donde se aprenden durante el entrenamiento. La salida de la **Forget gate** genera el vector ; si uno de los valores de este vector es cero o cercano a cero, entonces la LSTM eliminará ese valor, mientras que si alcanza valores cercanos a uno, esta información se mantendrá y llegará a la celda de estado.

Diagrama, Esquemático

Descripción generada automáticamente

Update gate, permite añadir nuevos elementos a la memoria, actualizando la memoria de la celda LSTM. Para ello toma el estado oculto anterior y la entrada actual , los transforma y los lleva de nuevo a una función de activación sigmoidal:

Donde se aprenden durante el entrenamiento. La salida de esta compuerta genera el vector , los valores que se preservan en la memoria de la red son los cercanos a uno.

Para actualizar la celda de estado, se utilizan los vectores y , multiplicando el valor anterior de esta celda por . () para eliminar la información irrelevante de la celda de estado. Luego se genera el vector de valores candidatos a formar parte de la memoria .

Donde y se aprenden durante el entrenamiento. A continuación, se filtran estos valores multiplicando, punto a punto, el vector con el vector y el resultado se suma a los valores anteriores de la celda de estado, generando así la celda actualizada :

Finalmente, se calcula el nuevo estado oculto, utilizando la Output gate. este estado oculto de salida es simplemente una versión filtrada de la celda de estado recién generada. Primero se escala el nuevo estado de celda  para garantizar que este en el rango de -1 a 1, pasando la información por una función tangente hiperbólica. En la Output gate se determinan que porciones de formaran parte del nuevo estado oculto:

Donde y se aprenden durante el entrenamiento. Por último, se filtran los valores de multiplicando por el vector generado por la Output gate.

Cuando se analiza una secuencia realmente se tienen replicas de la celda LSTM vista anteriormente, cada una de ellas corresponde a un instante de tiempo diferente dentro de la secuencia.

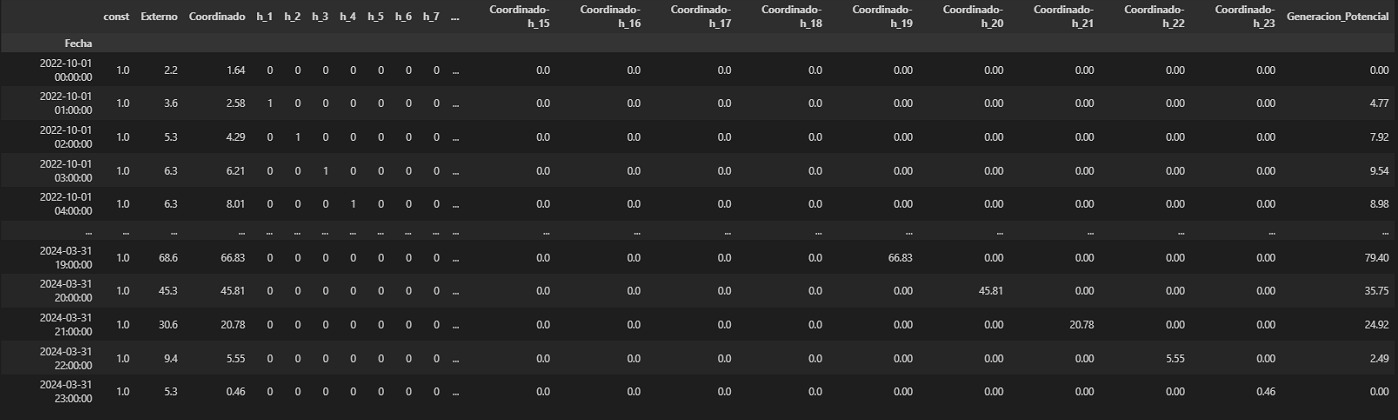
Imagen de la pantalla de un celular con letras

Descripción generada automáticamente con confianza media

1. **Desarrollo Modelo Benchmark Combinación SSA y LSTM**

Para este proyecto desarrolle 2 tipos de modelos LSTM y LSTM-SSA, este último es una combinación de lo mencionado anteriormente. Estos modelos están desarrollados para poder ser comparables con el modelo de regresión lineal inicial.

Como input del modelo usamos la misma matriz usada para la regresión lineal, cosa de que los modelos sean comparables:



**5.1 Metodología de desarrollo de modelos**

La metodología para el desarrollo de cada modelo fue la siguiente:

* + 1. **Separación de los datos en entrenamiento, validación y prueba:**

El primer paso en la metodología consistió en dividir el conjunto de datos en tres subconjuntos: entrenamiento, validación y prueba. Esta división se realizó sin mezclar aleatoriamente los datos debido a la naturaleza secuencial de las series temporales. Se definieron las fechas de inicio y fin para abarcar un periodo desde el 1 de octubre de 2022 hasta el 31 de marzo de 2024, resultando en un total de 548 días. El conjunto de entrenamiento abarcó el 70% de estos días, el de validación el 15%, y el de prueba el 15% restante. Esta partición asegura que el modelo pueda aprender patrones significativos durante el entrenamiento, validar su rendimiento para ajustar hiperparámetros y finalmente evaluar su desempeño en datos no vistos.

* + 1. **Definición de parámetros**

Para configurar el modelo LSTM, se definieron parámetros clave como el tamaño de la ventana deslizante, fijado en 60 días de datos horarios (60\*24). Se especificó también que la salida del modelo sería la predicción de 24 horas futuras, permitiendo evaluar la capacidad del modelo para prever el comportamiento diario de la variable objetivo.

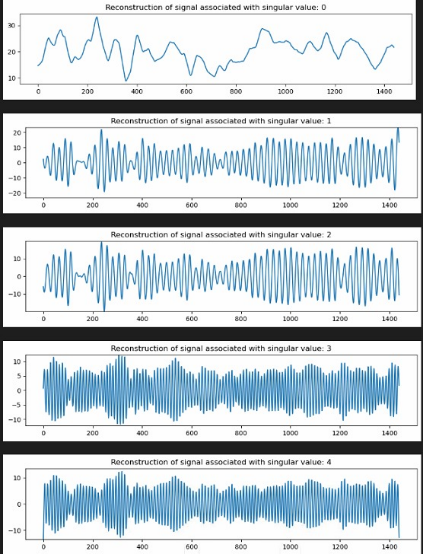
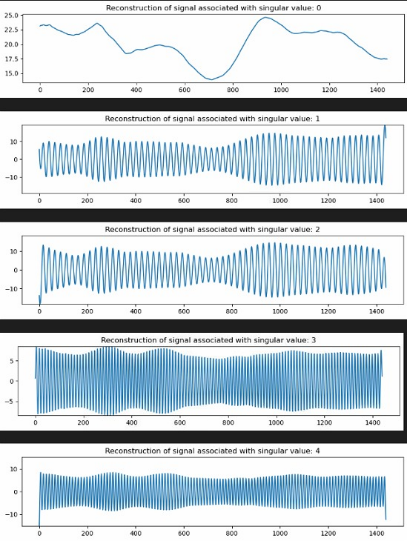
* + 1. **Definición del Dataset supervisado**

En esta etapa se definió la funcionalidad de la ventana deslizante. El modelo se estima, se pronostica el día siguiente y luego la ventana se desplaza un día.

Se distingue entre los modelos LSTM-SSA y LSTM en esta fase. En el modelo LSTM-SSA, antes de estimar el modelo, se ejecuta una función para descomponer la variable dependiente. Esta descomposición se realiza por ventana, añadiendo para cada ventana los componentes principales como variables independientes en la regresión. Debido a la cantidad de ventanas, este proceso no es breve.

Esto genera una desventaja debido a este paso adicional SSA, ya que para mejorar la calidad de la descomposición el costo de tiempo de computo aumenta, por lo que hay que manejar un trade-off entre calidad de descomposición y tiempo de cómputo.

Ante una calidad de descomposición que considero baja, esta etapa tardo 20 minutos aproximadamente, por lo que la iteración y entrenamiento de modelos se vuelve un trabajo bastante extenso en términos de tiempo.

Estas 2 imágenes representan la descomposición de una misma ventana de tiempo. La primera imagen usa una dimensión de embebido de 48 y la segunda de 120. Contando las componentes de arriba hacia abajo, podemos considerar la primera componente como la tendencia, la segunda y tercera como componente estacional y las últimas ruido. Podemos apreciar que a mayor dimensión de embebido la descomposición es más fina y los patrones son más estables.

* + 1. **Escalamiento de los datos**

El escalamiento de los datos es para garantizar que todas las características contribuyan de manera equitativa al entrenamiento del modelo. Se empleó el método MinMaxScaler de scikit-learn para escalar las características de entrada y la variable objetivo, ajustando el escalador a los datos de entrenamiento y aplicándolo posteriormente a los conjuntos de validación y prueba. Este paso aseguró que los datos mantuvieran una escala uniforme, mejorando así la estabilidad y convergencia del modelo durante el entrenamiento.

* + 1. **Algoritmo genético**

Para optimizar la arquitectura del modelo LSTM, se utilizó un algoritmo genético que evaluó diversas configuraciones del número de unidades en las capas LSTM y la inclusión de capas de dropout. Este algoritmo permitió explorar un espacio de hiperparámetros de manera eficiente, seleccionando las configuraciones que maximizaron el desempeño del modelo en términos de error cuadrático medio (MSE). El uso de un enfoque basado en genética proporcionó una metodología robusta para encontrar el diseño óptimo del modelo, evitando el sobreajuste y mejorando la capacidad generalizadora. Este proceso también ocupa bastante tiempo porque se tienen que entrenar muchos modelos para llegar a uno óptimo.

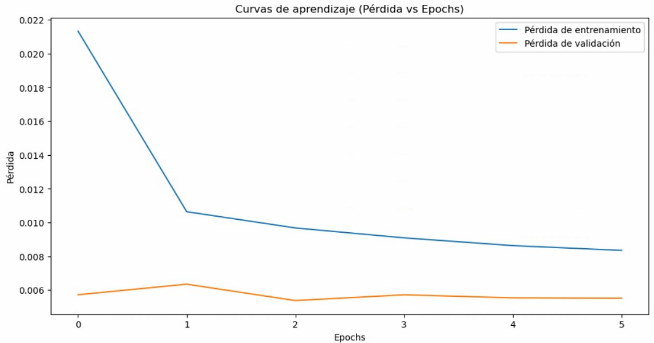
* + 1. **Selección del modelo óptimo**

El modelo óptimo seleccionado a partir del algoritmo genético consistió en una arquitectura con dos capas LSTM y una capa densa final. La primera capa LSTM contenía 287 unidades y la segunda 324 unidades, seguidas por una capa densa con 24 unidades. Se utilizó el optimizador Adam y la función de pérdida MSE. Este modelo se entrenó utilizando early stopping, monitorizando la pérdida de validación para evitar el sobreajuste y asegurando la restauración de los mejores pesos logrados durante el entrenamiento.

* 1. **Resultados**

**5.2.1 Curva de aprendizaje**

Los gráficos presentados ilustran las curvas de aprendizaje de dos modelos: LSTM-SSA y LSTM, evaluando la pérdida de entrenamiento y validación a lo largo de las épocas. Estos gráficos son cruciales para comprender el comportamiento de los modelos en términos de convergencia y su capacidad para evitar el sobreajuste. Ambos modelos muestran una convergencia adecuada y logran evitar el sobreajuste gracias al uso de early stopping, que detiene el entrenamiento cuando la pérdida de validación deja de mejorar.

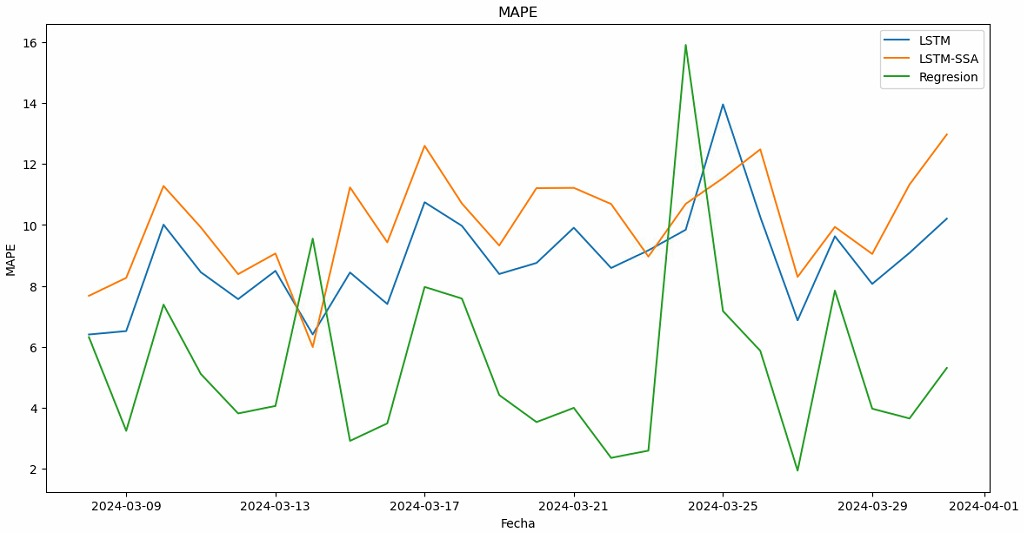
****Gráfico, Gráfico de líneas

Descripción generada automáticamente

Curvas de aprendizaje Modelo LSTM-SSA y Modelo LSTM respetivamente.

**5.2.2 Calculo de los errores**

Finalmente calculamos los errores totales y diarios. En el grafico podemos observar el MAPE diario y en la tabla el MAPE y RMSE total

****

|  |  |  |
| --- | --- | --- |
|  | MAPE | RMSE |
| Regresión | 5.41 | 99.18 |
| LSTM | 8.88 | 116.13 |
| LSTM-SSA | 10.09 | 133.49 |

1. **Conclusiones**

El mejor método continua siendo la regresión lineal. Estos resultados son únicamente para una central eólica por lo que me falta comprobar los modelos con un par de centrales eólicas y solares para validar de forma más robusta los resultados. Juan me recomendó unas últimas afinaciones a mis modelos, las probare y luego procederé con la validación de los resultados con el resto de centrales.