UNIVERSIDAD DE GRANADA

ETS de Ingeniería Informática

MASTER EN CIENCIA DE DATOS E INGENIERÍA DE COMPUTADORES

TRABAJO FIN DE MÁSTER

Predicción de producción de energía fotovoltaica en un horizonte de una hora

UNIVERSIDAD DE GRANADA

Presentado por:

D. Luis Cabezón Manchado

Tutor:

Prof. Dra. María del Carmen Pegalajar Jiménez

Resumen

La energía solar fotovoltaica se encuentra en auge debido a la continua mejora de eficiencia de los paneles fotovoltaicos junto a una tendencia bajista en los costes de producción. Además, el compromiso de la Unión Europea para alcanzar el 30% de cuota de energías renovables está permitiendo a numerosas empresas obtener financiación y poder instalar sus propias placas fotovoltaicas.

Sin embargo, la naturaleza de la energía solar es intermitente e incontrolable, lo que genera una inestabilidad en los sistemas fotovoltaicos que suministran energía. La literatura ha tratado de modelar este problema a través de diferentes modelos desde regresiones multilineales hasta la utilización de redes neuronales profundas. El principal problema es la multitud de factores que pueden influir en la predicción, haciendo que cada modelo sea muy diferente y dificultando la posibilidad de comparación entre modelos entrenados en diferentes lugares.

Este proyecto desarrolla una metodología para el ajuste de un modelo de predicción de producción de energía fotovoltaica en una granja de Escocia. Se realiza un repaso a las técnicas más utilizadas en la literatura, utilizando técnicas estadísticas como medias móviles, pasando por técnicas basadas en árboles, y acabar utilizando modelos de redes neuronales.

El objetivo es obtener un modelo capaz de predecir de una forma precisa la energía que se va a producir en la siguiente hora, utilizando como variables predictoras información del pasado.

Abstract

Photovoltaic solar energy is booming due to the continuous improvement in the efficiency of photovoltaic panels together with a downward trend in production costs. In addition, the European Union's commitment to a 30% share of renewable energy is enabling many companies to obtain financing and install their own PV panels.

However, the nature of solar energy is intermittent and uncontrollable, leading to instability in the PV systems that supply power. The literature has tried to model this problem through different models ranging from multi-linear regressions to the use of deep neural networks. The main problem is the multitude of factors that can influence the prediction, making each model very different and making it difficult to compare models trained in different locations.

This project develops a methodology for fitting a model for predicting photovoltaic energy production on a farm in Scotland. It reviews the most commonly used techniques in the literature, using statistical techniques such as moving averages, through tree-based techniques, and finally using neural network models.

The objective is to obtain a model capable of accurately predicting the energy to be produced in the next hour, using past information as predictor variables.

Tabla de contenido

1 – Introducción

Hoy en día las energía renovables están tomando un mayor peso en la industria energética, en contraposición de las energía fósiles. Esto es debido a que las energías renovables no producen gases de efecto invernadero, causantes del cambio climático, son inagotables y generan residuos de fácil tratamiento. Además, con el avance de la tecnología, los costes de producción y mantenimiento están sufriendo una evolución a la baja[1] como se puede observar en la Fig 1. En esta figura se muestra el precio de producción de diferentes energías renovables a lo largo de los años según IRENA, la Agencia Internacional de las Energías Renovables.

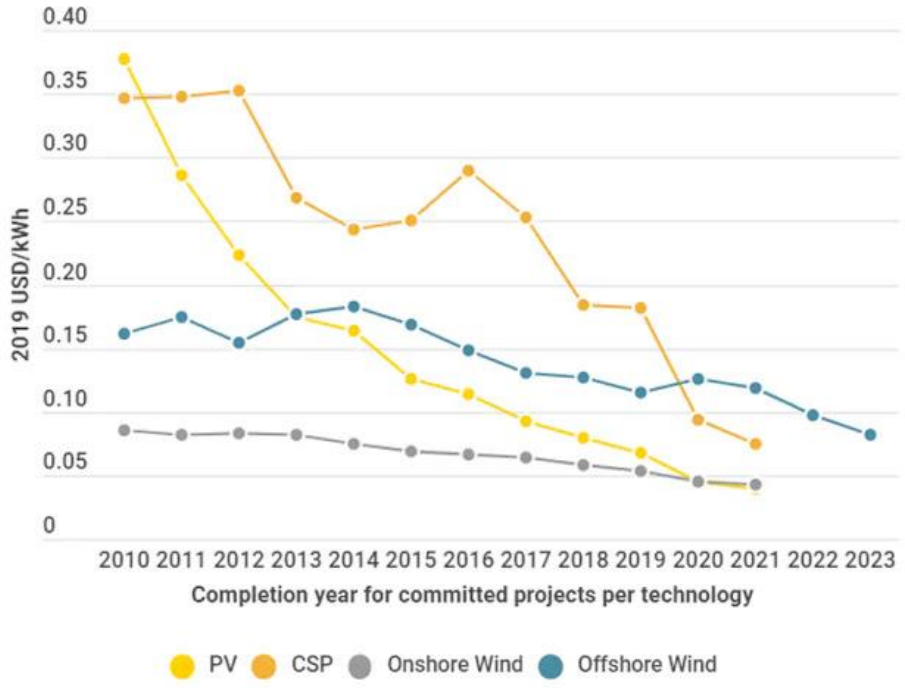


Figura 1 – Predicción del coste de energías renovables por IRENA

El crecimiento de las energías renovables es innegable, viéndose reflejado en las estadísticas de la Agencia Internacional de la Energía (AIE). Según la AIE, la participación en el suministro eléctrico global de las energías renovables alcanzará un 40% en el año 2040 [2], principalmente a través de la energía eólica y fotovoltaica.

Entre las energía renovables, o también llamadas energías limpias, se encuentran:

- **Energía eólica**: energía obtenida a partir del viento.
- **Energía solar:** energía obtenida a partir del sol.
- **Energía hidráulica:** energía obtenida a partir de los ríos y corrientes de agua dulce.
- **Biomasa:** energía obtenida a partir de materia orgánica
- **Energía geotérmica:** energía calorífica contenida en el interior de la Tierra
- **Energía mareomotriz:** energía obtenida a partir de las mareas.
- **...**

La energía solar puede extraerse utilizando principalmente tecnología térmica, aprovechando el calor solar, y tecnología fotovoltaica. Esta última transforma de manera directa la luz solar en electricidad utilizando tecnología basada en el **efecto fotovoltaico.**

El efecto fotovoltaico es una propiedad que tienen ciertos materiales (como el silicio) que permite generar electricidad cuando se ven sometidos a radiación solar. Esto ocurre cuando los fotones "chocan" en el material fotovoltaico y liberan electrones, creando así un flujo de energía eléctrica. La Fig 2 muestra un pequeño resumen visual del funcionamiento de una placa fotovoltaica.

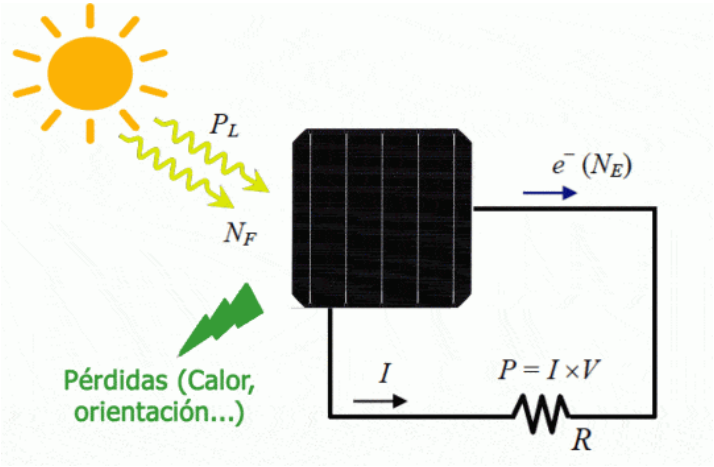


Figura 2 - Funcionamiento de una placa fotovoltaica

El principal inconveniente, y que está suponiendo una fuerte barrera de entrada a numerosas empresas eléctricas, es la volatilidad de la energía producida debido a cambios en el clima.

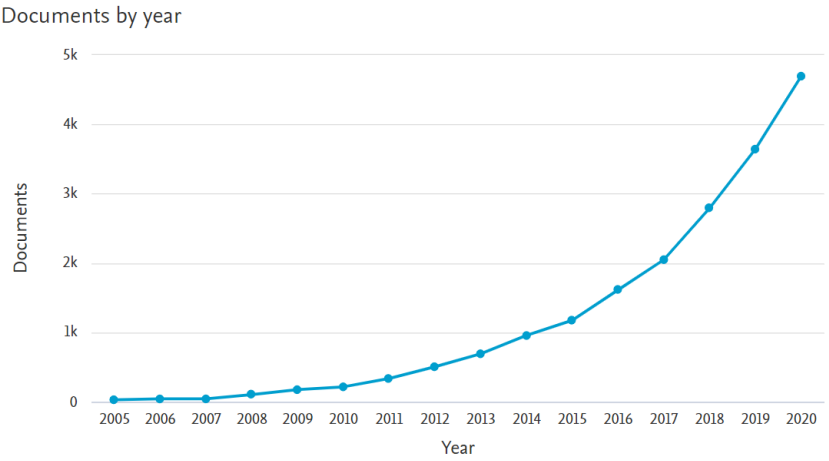
En granjas de energía fotovoltaica de gran escala, un sistema de predicción erróneo puede provocar a la empresa importantes pérdidas económicas. Una predicción a futuro precisa en un pequeño intervalo como puede ser una hora permite realizar una gestión óptima de la misma, permitiendo almacenarla en baterías y/o venderla al mercado.

2 – Estado del arte

Desde que se fabricó la primera placa fotovoltaica con una eficiencia del 1% en 1883, a cargo de Charles Fritz[3], los investigadores se han centrado en desarrollar mejores paneles a través del cambio de materiales con el objetivo de aumentar la eficiencia y, por tanto, la producción de energía.

Hasta la fecha, la mayor eficiencia lograda por una placa solar se obtuvo en 2020, obteniendo una eficiencia del 44.5% [4]. Aunque se esté consiguiendo ir mejorando la eficacia de dichos paneles, la utilización de materiales más complejos está provocando que dichas placas no tengan cabida en el mercado doméstico. En el mercado actual, se están utilizando paneles de silicio que tienen una eficiencia cercana al 20% y alcanzan una vida útil de 20 años[5].

La producción de energía basada en sistemas de producción fotovoltaica ha despertado el interés de investigadores, tanto en el ámbito privado como académico, debido a su potencial. Esto puede observarse en repositorios científicos como Scopus, donde se han publicado más de 23.000 papers relaccionados con *forecast, photovoltaic, power*. La gráfica 1 muestra el número de publicaciones realizadas en Scopus durante los últimos 15 años.



Gráfica 1 - Evolución de publicaciones en Scopus

Hoy en día existan numerosos modelos físicos que proporcionan información sobre la relación entre variables ambientales y la potencia generada por las placas solares. Sin embargo, la propia naturaleza del problema, tan dependiente del tiempo, hace que estos modelos no sean tan precisos. La utilización de modelos de aprendizaje máquina trata de reducir el impacto de dicha problemática.

El modelo base utilizado por numerosos investigadores se conoce como *modelo de persistencia* [6][7][8], el cual asume que la potencia generada en un espacio de tiempo *t* es igual a la potencia generada *d* intervalos de tiempo antes. Por ejemplo, la potencia generada durante una hora concreta del día será igual que la potencia generada durante la misma hora del día anterior. La principal desventaja de este modelo es que asume cierta estabilidad climática a lo largo de un período de tiempo.

Los modelos estadísticos no necesitan información interna para tratar de modelizar el sistema. Es una aproximación *data-driven* capaz de extraer relaciones basadas en el pasado para tratar de predecir el futuro.

La correcta extracción y tratamiento de los datos ha demostrado influir de forma positiva en la obtención de resultados. Muchos autores han estudiado la relación entre la selección de entradas y la precisión del modelo. Almeida et al. (2015) [9] concluyeron que la mejor combinación para predecir la producción energética en una planta fotovoltaica el día siguiente fue utilizar los últimos 30 días.

Otros autores han conseguido resultados prometedores aplicando una clasificación de los datos en función de las condiciones climáticas (soleado, semi-nublado, nublado) y así obtener diferentes modelos[10][11][12][13].

Por último, la aplicación de metaheurísticas para la optimización del conjunto de entrada como los algoritmos genéticos[14], basados en enjambres[15], luciérnagas[16], lógica difusa[17] han demostrado mejorar los resultados previos.

La literatura realiza una separación entre modelos de regresión, modelos basados en técnicas de aprendizaje máquina e híbridos.

2.1 – Modelos

2.1.1 – Modelos regresivos

Dentro de los modelos regresivos, aquellos que tratan de modelar el problema como una relación (lineal o no-lineal) entre las variables de entrada y salida. Las principales técnicas de regresión se pueden dividir en función del tratamiento que reciba la serie temporal (linear o no-lineal y estacionaria o no-estacionaria). Una serie temporal se clasifica como estacionaria cuando la serie fluctúa entorno a una media estática.

- **Linear stationary models**: aquí encontramos modelos autorregresivos (AR)[18], los cuales modelan la salida como una combinación lineal de valores pasados; modelos de medias móviles[19].

- **Linear non-stationary models:** modelos autorregresivos integrados de medias móviles (ARIMA) los cuales consideran la unión entre ambos modelos, y el SARIMA el cual introduce la estacionalidad.

- **Non-Linear stationary models:** modelos como NARMAX

2.1.2 – Modelos de Inteligencia Artificial

Las técnicas de aprendizaje automático más utilizadas dentro del presente ámbito se pueden enumerar como:

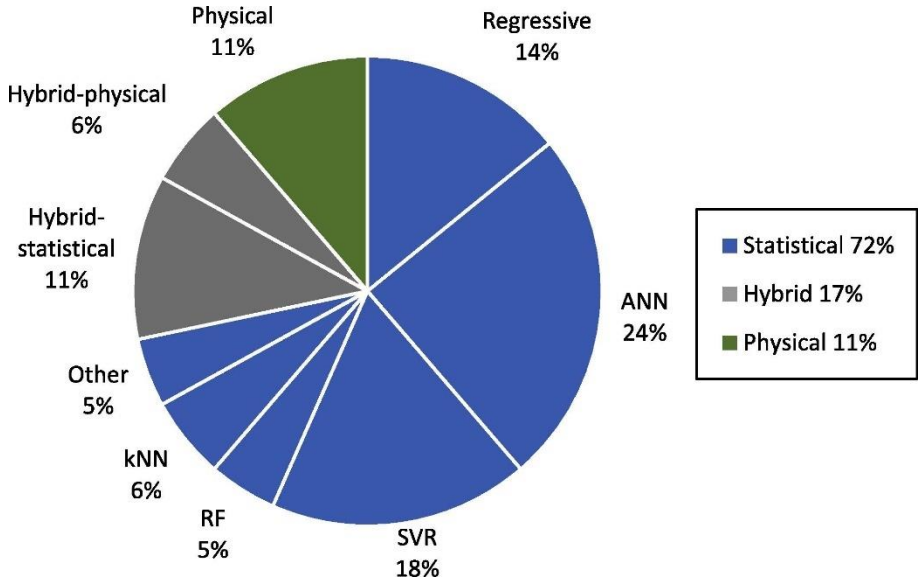
- **Artificial Neural Networks:** basadas en la estructura de una neurona, son una de las técnicas más utilizadas. Existen numerosas topologías, La clasificación más conocida ordena los modelos en función del número de capas ocultas (perceptrón simple o multicapa). Existen otros tipos de redes neuronales como son las redes recurrentes de Elman, LSTM, etc.

- **k-Nearest Neighbors**: una de las técnicas más simples del aprendizaje máquina. Está basado en la similitud entre datos en el espacio de características.

- **SVM:** esta técnica de aprendizaje supervisado, introducida inicialmente por Vapnik y Lerner[20] y desarrollada por Cortes and Vapnik[21] más tarde para problemas de clasificación. Cuando son aplicados para problemas de regresión se conocen como Support Vector Regression Machines (SVR).

- **Random Forests:** son una combinación de árboles predictores tal que cada árbol depende de los valores de un vector aleatorio probado independientemente y con la misma distribución para cada uno de ellos.

Además, se han realizado diferentes estudios que determinan que técnicas aplicar y en qué condiciones son aplicadas para obtener el mejor resultado sobre diferentes situaciones propuestas[22][23][24][26]



Gráfica 2 - Used of techiniques for PV Production Forecasting

2.1.3 – Modelos Híbridos

Cuando modelamos únicamente a partir de un modelo, hay cierta información que se omite debido a la forma en que cada técnica transforma los datos. La combinación de modelos permite, en la mayoría de los casos, aumentar la precisión de los modelos por separado.

La gráfica 2 muestra el porcentaje de uso en los diferentes estudios de las técnicas más comunes relaccionadas con la predicción de producción energética fotovoltaica, siendo las redes neuronales la más usadas.

2.2 – Determinista vs Probabilístico

La predicción de energía se ha aplicado a lo largo del tiempo en diferentes campos (solar, eólica). Cada uno de estos campos presenta sus propias peculiaridades y se pueden encontrar diferencias entre precisiones obtenidas. La predicción de energía solar está en un estado de inmadurez respecto al resto de predicciones energéticas estudiadas por Hong et al [26] debido a la reciente interrupción de la energía solar en el mercado eléctrico como se puede observar en la Fig 3.

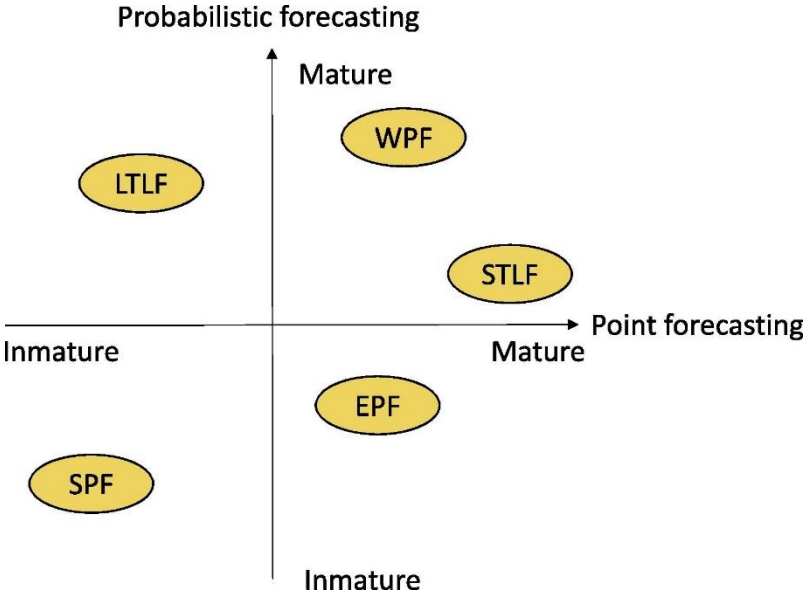


Figura 3 - Estado de madurez de la predicción de energías renovables

La Fig 4 muestra la principal diferencia entre ambos enfoques de predicción. La predicción determinista únicamente indica un único valor de predicción en un horizonte t mientras que la predicción probabilística, a parte, proporciona un intervalo de confianza sobre el que se encuentra la predicción.

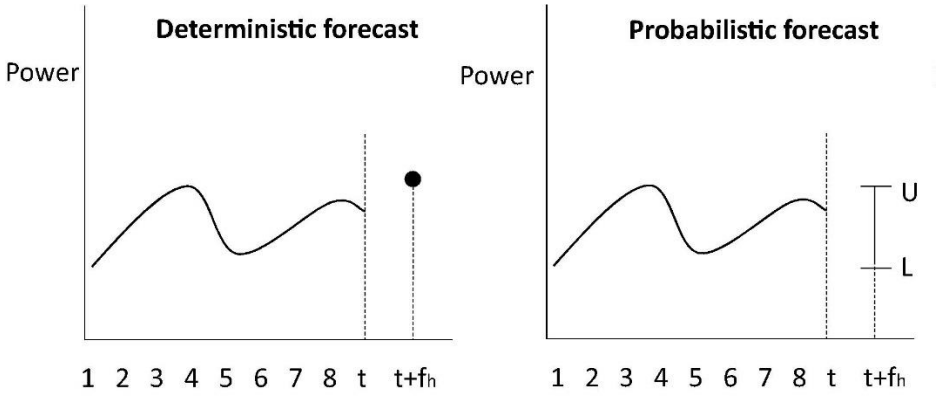


Figura 4 - Tipos de predicción

2.3 – Horizonte de predicción

El horizonte de predicción es una ventana de tiempo entre el último dato medido y aquel que se desea conocer. Existen predicciones a *corto, medio y largo plazo*, y en función del interés de dicha predicción se recomiendan utilizar unos u otros horizontes. Sin embargo, no existe una regla general que permite conocer que modelos entregan mejores resultados en función del horizonte deseado.

La clasificación más utilizada[27] dentro de la literatura se divide:

- **Muy corto plazo:** predicciones con horizontes que van desde unos segundos hasta una media hora.

- **Corto plazo**: pronóstico de la potencia fotovoltaica que se realiza con un horizonte de media o varias horas. La predicción a corto plazo garantiza la programación y envío de energía eléctrica. El pronóstico a corto plazo mejora la seguridad de la operación de la red

- **Medio plazo:** está predicción se realiza en un intervalo de tiempo de seis horas a un día. Este tipo de pronóstico ayuda a mejorar la planificación de mantenimiento.

- **Largo plazo:** los pronósticos que van de un día a una semana o más se agrupan dentro de esta categoría.

La tabla 1 recoge de forma resumida las principales aplicaciones de cada una de las predicciones en función de su horizonte de predicción.

Time horizon	Range	Applications
Very short-term	Few seconds to 30 minutes ahead	- Electricity Market Clearing - Regulation Actions
Short-term	30 minutes to 6 hours ahead	- Economic Load Dispatch Planning - Load Increment/Decrement Decisions
Medium-term	6 hours to 1 day ahead	- Generator Online/Offline Decisions - Operational Security in Day-Ahead Electricity Market
Long-term	1 day to 1 week or more ahead	- Unit Commitment Decisions - Reserve Requirement Decisions - Maintenance Scheduling to Obtain Optimal Operating Cost

Tabla 1 - Tipos de horizonte y sus aplicaciones

3 – Modelos Estadísticos

Los modelos de regresión son aquellos que tratan de modelar el problema como una relación (lineal o no-lineal) entre las variables de entrada y salida

3.1 – Modelos de regresión

En 1970, Box y Jenkins desarrollaron un cuerpo metodológico destinado a identificar y estimar modelos dinámicos de series temporales en los que la variable temporal toma una importancia relevante.

Dado que el objetivo es explicar el valor que toma la variable, dependiente del tiempo, en un instante *t*, se dispone a estudiar el propio pasado de la variable. Una vez estudiado, se busca extraer el modelo que permita explicar su comportamiento y predecir valores futuros en el tiempo. Este procedimiento se hará operativo a través de los modelos ARIMA.

Una serie temporal se puede dividir, siguiendo el análisis clásico, en tres principales componentes:

- Tendencia T(t)
- Estacionalidad S(t)
- Componente Irregular E(t)

A su vez, existen dos formas de combinar dichas componentes, pudiendo obtener dos tipos de descomposiciones de series temporales:

- Descomposición aditiva: bajo este supuesto la magnitud de las fluctuaciones estacionales de la serie no varía al hacerlo la tendencia.

$$y_t = T_t + S_t + E_t$$

- Descomposición multiplicativa: bajo este supuesto la magnitud de las fluctuaciones estacionales de la serie crece y decrece con los crecimientos y decrecimientos de la tendencia.

$$y_t = T_t \times S_t \times E_t$$

El principal requisito para poder modelar de forma paramétrica una serie temporal es que debe ser estacionaria. Una serie estacionaria es aquella cuya media y varianza no varían en el tiempo. Esta propiedad indica que los datos pueden estudiarse bajo un mismo modelo paramétrico independiente del tiempo.

La forma de comprobar si una serie temporal es estacionaria es a través del test de Dickey-Fuller aumentado (ADF). El procedimiento para realizar dicha prueba es semejante al test de Dickey-Fuller, pero aplicado al modelo

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t$$

Donde α es una constante, β el coeficiente sobre una tendencia temporal y p el orden de retraso del proceso autorregresivo.

La prueba se lleva a cabo bajo la hipótesis nula $\gamma = 0$ contra la hipótesis alternativa de $\gamma < 0$. Si el p-valor obtenido es menor que nuestro nivel de significación, rechazamos la hipótesis nula y podemos afirmar que nuestra serie es estacionaria.

3.1.1 – AR: autoregressive

Un modelo se clasifica como autorregresivo si la variable target en un periodo *t* es explicada por las observaciones de ella misma correspondientes a períodos anteriores con cierto error.

Los modelos autorregresivos se abrevian con las siglas AR(p) siendo *p* el orden del modelo. El orden de un modelo expresa el número de observaciones retardadas de la serie temporal utilizadas para la predicción.

La expresión genérica de un modelo AR(p) es el siguiente:

$$Y_t = c + \sum_{i=1}^{p} \varphi_i Y_{t-i} + \varepsilon_t$$

donde $\varphi_1 \dots \varphi_p$ son los parámetros del modelo, *c* una constante y ε_t es ruido blanco. Esto se puede expresar de forma equivalente utilizando el operador de retroceso B como

$$Y_t = c + \sum_{i=1}^{p} \varphi_i B^i Y_t + \varepsilon_t$$

de manera que, moviendo el término sumatorio hacia el lado izquierdo y el uso de la notación polinómica, tenemos

$$\emptyset(B) Y_t = c + \varepsilon_t$$

3.1.2 – MA: moving average

El modelo de medias móviles especifica que la variable de salida depende linealmente del valor actual y varios de los anteriores. Estos modelos se denotan como AM(q) siendo *q* el orden del modelo

La expresión genérica de un modelo MA (q) es el siguiente:

$$Y_t = c - \sum_{i=1}^{q} \theta_i Y_{t-i} + \varepsilon_t$$

Esto se puede expresar de forma equivalente utilizando el operador de retroceso B como

$$Y_t = c - \sum_{i=1}^{q} \theta_i B^i Y_t + \varepsilon_t$$

3.1.3 – ARIMA: Autoregressive integrated mooving average

El modelo autorregresivo integrado de medias móviles, conocido como ARIMA, de orden *p, q, d,* consiste en la unión de un modelo autorregresivo de orden *p*: AR(*p)*, un modelo de medias móviles de orden *q*: AM(*q*) y un modelo integrado de orden *d.*

La formulación matemática del modelo ARIMA(p,d,q) viene dada por:

$$Y_t = -(\Delta^d Y_t - Y_t) + \emptyset_0 + \sum_{i=1}^{p} \emptyset_i \Delta^d Y_{t-1} - \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

en donde *d* corresponde a las d diferencias que son necesarias para convertir la serie en estacionaria, $\emptyset_1 \dots \emptyset_p$ son los parámetros pertenecientes a la parte autorregresiva del modelo, $\theta_1 \dots \theta_q$ los parámetros relativos a la parte de medias móviles y, por último, ε_t es el término de error.

Si dos parámetros del modelo ARIMA toma el valor cero, podemos clasificar dichos modelos según la tabla 2.

Modelo	**Nombre**
ARIMA(0,0,0)	Ruido blanco
ARIMA(0,1,0) sin constante	Paseo aleatorio
ARIMA(0,1,0) con constante	Paseo aleatorio con cambio
ARIMA(p,0,0)	Autorregresivo
ARIMA(0,0,q)	Medias móviles

Tabla 2 - Tipos de ARIMA

3.2.4 – SARIMA: Seasonal autoregressive integrated mooving average

Uno de los principales motivos por los que una serie es no estacionaria es la presencia de la componente de estacionalidad.

Una serie es estacional cuando no es estacionaria en media, pero varía con una pauta cíclica *s* tal que $E[Y_t] = E[Y_{t-s}]$

De esta forma podemos modelar ARIMAs estacionales y estacionarias. Normalmente la estacionalidad se incorpora de forma multiplicativa de forma que el modelo resultante se puede expresar como *SARIMA(p,d,q)x(P,D,Q)*

$$\emptyset(B)\Phi(B)(1-B)^d(1-B^s)^D Y_t = \theta(B)\Theta(B)a_t$$

siendo la parte regular:

$$\emptyset(B) = 1 - \sum_{i=1}^{p} \phi_i B^i$$

$$\theta(B) = 1 + \sum_{i=1}^{q} \theta_i B^i$$

y la parte estacional:

$$\Phi(B) = 1 - \sum_{i=1}^{p} \Phi_i B^{is}$$

$$\Theta(B) = 1 + \sum_{i=1}^{q} \Theta_i B^{is}$$

3.1.4 – Regresión Lineal: Elastic Net

La regresión lineal es un método estadístico que trata de representar la relación entre la variable objetivo y las variables explicativas mediante una ecuación lineal. Un ejemplo de regresión lineal simple lo encontramos en la Fig 5.

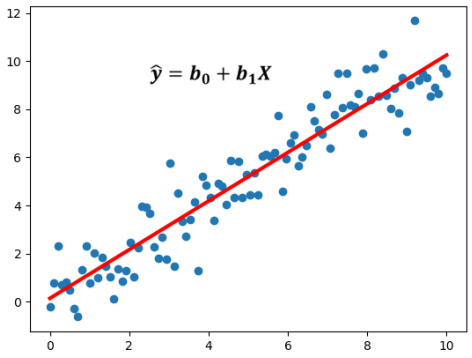


Figura 5 - Ejemplo de regresión lineal simple

El modelo lineal sobre un conjunto de más de una variable explicativa se puede expresar con la función matemática descrita en la imagen Fig X. En ella, se puede ver el valor de *y* para una determinada observación *i*.

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \epsilon_i$$

La interpretación de cada uno de los elementos que conforman el modelo es la siguiente:

- βo: es la ordenada en el origen, se corresponde con el valor que toma la variable objetivo *y* cuando el resto de predictores toman el valor cero.
- βj: es el peso que se le da a dicha variable para el cálculo de la variable objetivo. Nos puede dar una pequeña idea de cuán importante es dicha variable en el cálculo de valores *y*.
- e: residuo o error, la diferencia entre el valor observador y el valor estimado por el modelo.

En la mayoría de los problemas, los valores de βo y βj se desconocen. Es por ello por lo que tendremos que aproximarlos a partir de nuestras muestra de datos. Ajustar un modelo de regresión lineal trata de buscar aquellos valores de βo y βj que mejor ajustan la recta, en caso de regresión simple, o el hiperplano en caso de regresión multivariable.

El método más utilizado para tratar de ajustar el modelo se conoce como ajuste por mínimos cuadrados. Esta técnica consiste en minimizar la suma de las desviaciones (errores) entre cada dato de la muestra y la recta.

3.1.4.2 - Condiciones para la regresión lineal

Para que un modelo de regresión lineal puede aportar información válida y fiable, deben cumplirse una serie de requisitos. Aunque muchos de estos requisitos pueden no cumplirse, se pueden seguir obteniendo diferentes conclusiones, siempre que se asuma que dichas conclusiones no son completamente fiables.

1 – No colinealidad entre variables predictoras

La colinealidad se entiende como la relación entre varias variables predictoras. Como consecuencia, al interpretar los coeficientes de regresión del modelo no podemos evaluar de forma independiente el peso de cada variable en nuestro modelo.

2 – Distribución normal de la variable objetivo

La variable objetivo debe seguir una distribución normal. Para comprobarlo se pueden utilizar métodos gráficos como histogramas o métodos estadísticos como los test de normalidad.

3 – Varianza constante en la variable objetivo

La homocedasticidad consiste en obtener los errores de predicción constantes a lo largo de las observaciones.

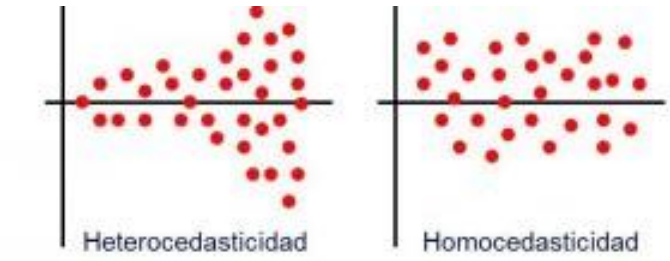


Figura 6 - Heterocedasticidad vs Homocedasticidad

4 – Independencia entre observaciones

Los valores de cada observación no dependen de la observación anterior. Esto tiene una alta relevancia en modelos temporales.

3.1.4.3 – Regularización: Elastic Net

Una forma de suavizar el impacto provocado por el no cumplimiento de alguno de los requisitos explicados anteriormente es través de estrategias de regularización como *ElasticNet*. Esta ténica obliga a los coeficientes de regresión a tender a cero, reduciendo de esta forma la varianza.

3.2 – Modelos de Inteligencia Artificial

En esta sección se presenta de forma teórica el funcionamiento de diferentes modelos orientados al resolución de problemas de regresión.

La variable temporal se *divide*, tratando cada elemento de la fecha (año, mes, día, hora) como variables independientes. De esta forma, tratamos de informar al modelo de la temporalidad a través de dichas variables.

3.2.2 – KNN: K-Nearest Neighbors

El método de k-vecinos cercanos es un modelo no-paramétrico, no presupone ninguna distribución en los datos, que consiste en estimar el valor de un dato desconocido a partir de las características de los datos más próximos a él, calculando esa proximidad a partir de una función de similitud o distancia.

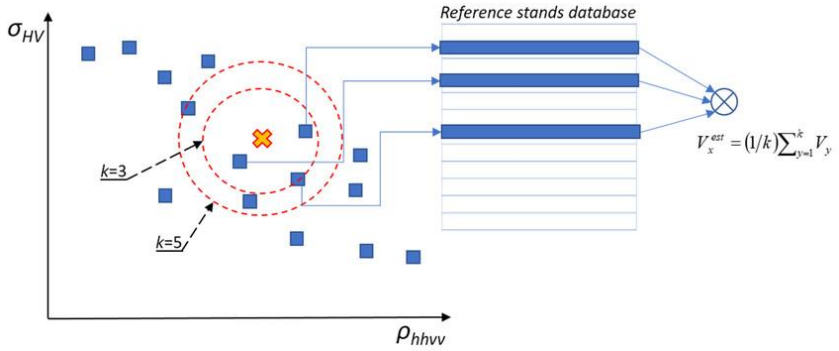


Figura 7 - KNN en regresión

Una vez calculados los k vecinos cercanos, la variable objetivo se calcula como la media del valor de las variables objetivo de sus vecinos cercanos. La Fig 7 muestra un ejemplo de cálculo. La función que expresa el valor de la variable objetivo viene dada como:

$$\hat{y} = \frac{1}{k}\sum_{i=1}^{k} y_i$$

3.1.2.1 – Parametrización del *k-NN*

El principal parámetro *k* indica la cantidad de observaciones se van a tener en cuenta para calcular el valor del elemento nuevo. Una vez ordenados los valores en función de su distancia, se eligen los primeros *k* valores.

La distancia más utilizada dentro de la aplicación de dicho modelo se conoce como distancia euclidiana. Aparte de dicha distancia existen otras como se puede observar en la tabla.

Función	Método
Manhattan	$\sum_{i=1}^{k} \lvert x_i - y_i \rvert$
Minkowski	$\left(\sum_{i=1}^{k} (\lvert x_i - y_i \rvert)^q\right)^{1/q}$

La ponderación designa el método por el cual se calculan los pesos que presentan cada uno de los k-vecinos cercanos. En la tabla 3 se presentan algunos de los métodos más utilizados de asignación de pesos.

Función	Método
Lineal	$w_i = k - (i - 1)$
Medio	$w_i = 1$
Proporcional	$w_i = 1/d_i$

Tabla 3 - Tipos de pesos

3.2.3 – DT: Decision Tree

El árbol de decisión crea diferentes modelos de regresión en forma de estructura de árbol. El modelo divide el espacio de características original en diferentes subconjuntos cada vez más pequeños.

El resultado final es un árbol con nodos de decisión, aquellos que dividen el espacio en dos subespacios de tamaño menor, y nodos hoja. Estos últimos representan la última división del subespacio.

En la Fig 8 se muestra un ejemplo del funcionamiento de un árbol de decisión utilizando en un problema de clasificación aplicado a los supervivientes del Titanic.

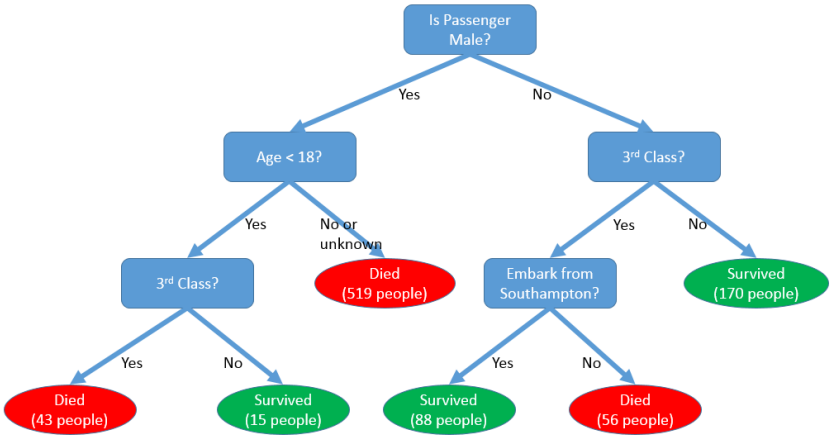


Figura 8 - Ejemplo de árbol de decisión

Los árboles de decisión aplicados a problemas de regresión utilizan el algoritmo CART. El algoritmo CART, *Classification And Regression Trees*, genera árboles de decisión binarios, es decir, cada nodo se divide exactamente en dos ramas.

En la Fig 9 se muestra el funcionamiento de un árbol de decisión que trata de asemejarse a una curva sinusoidal con cierto ruido. Podemos ver que el modelo aumenta la precisión cuanto mayor sea la profundidad de este.

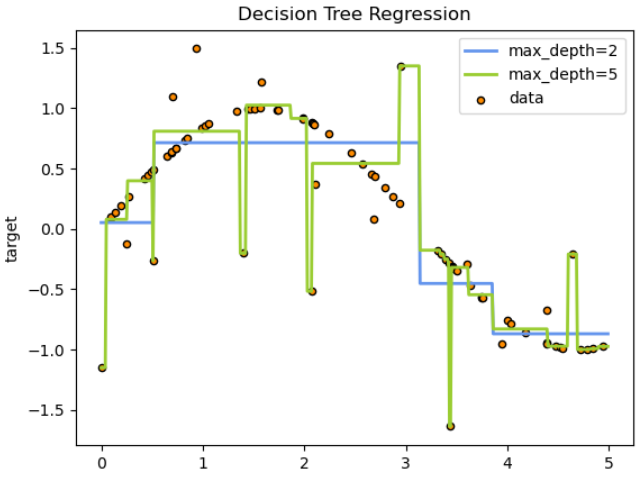


Figura 9 - Curva sinusoidal a través de un DT

Los árboles de decisión para regresión se construyen utilizando un algoritmo *greedy*. Un algoritmo de este tipo es una estrategia de búsqueda consistente en elegir la opción optima en cada paso local con el objetivo de llegar a una solución general óptima.

La función de coste es la siguiente:

$$J(a, l_a) = \frac{m_{izquierdo}}{m} MSE_{izquierdo} + \frac{m_{derecho}}{m} MSE_{derecho}$$

Siendo:

- a – abreviatura del atributo
- l_a – límite del atributo
- m – número de muestras

MSE es el error cuadrático medio, calculándose como se puede observar en la ecuación siguiente:

$$MSE = \frac{1}{M} \sum_{i=1}^{M} (real_i - estimado_i)^2$$

3.2.3.2 – Parametrización del *Decision Tree*

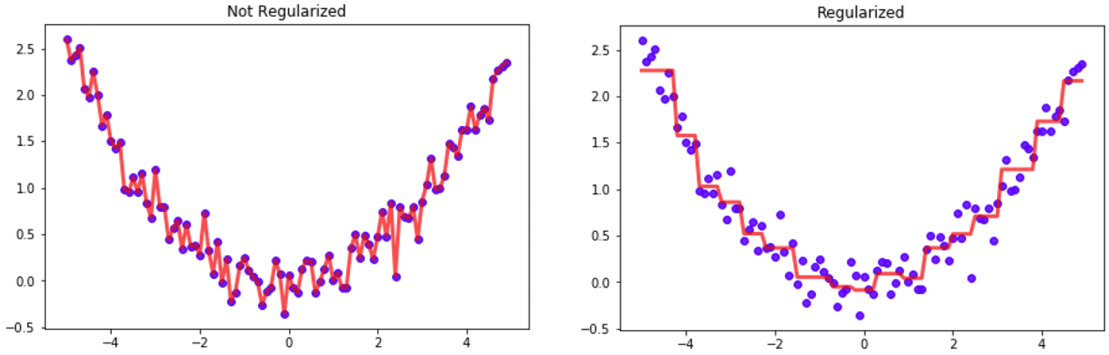
Los árboles de decisión de la librería *scikit-learn* no están regularizados por defecto. Esto significa que utilizarán tantos nodos como necesiten para tratar de reducir al máximo el error. Esto puede provocar un sobreajuste del modelo en la fase de entrenamiento, mientras que en la fase de test el rendimiento sea muy bajo.

La regularización consiste en limitar las capacidades del modelo para obtener modelos robustos y que tengan la capacidad de generalizar ante nuevos datos. Los parámetros más usados para regularizar los árboles son:

- **max_depth:** profundidad máxima del árbol.
- **min_samples_split:** número mínimo de instancias necesarias para poder seguir dividiendo el nodo.
- **min_samples_leaf:** número mínimo de muestras que debe haber en un nodo para ser considerado como nodo hoja.

- **max_leaf_nodes:** número máximo de nodos finales

El gráfico 3 muestra la importancia de regularizar nuestros modelos basados en árboles de decisión, ya que estos tienen a sobreajustarse durante las fases de entrenamiento.



Gráfica 3 - Regularización de un DT

3.2.4 – XGB: Extreme Gradient Boosting

Extreme Gradient Boosting es una librería de código abierto que proporción una implementación eficiente y efectiva del algoritmo de potenciación del gradiente. Esta ténica de aprendizaje automático produce un modelo predictivo aplicando la técnica de Boosting. La idea del *Boosting* es generar múltiples modelos de predicción "débiles" secuencialmente, de forma que cada modelo tome los resultados obtenidos por el modelo anterior, para generar un modelo más "fuerte".

El algoritmo de optimización utilizado para lograr dicho objetivo se conoce como *Gradient Descent*, o *Descenso del Gradiente*. El gradiente evaluado en cualquier punto de una función representa la dirección del ascenso más pronunciado. En el caso de minimizar la función, debemos seguir el negativo del gradiente. Formalmente, si comenzamos en un punto x_o y nos movemos una distancia positiva α en la dirección del gradiente negativo, nuestra nueva posición x_1 será:

$$x_1 = x_o - \alpha \nabla f(x_o)$$

Repitiendo iterativamente dicho proceso n-veces, tendremos que el siguiente punto al punto enésimo será:

$$x_{n+1} = x_n - \alpha \nabla f(x_n)$$

Uno de los principales problemas de este algoritmo de optimización es que no puede determinar si el mínimo encontrado es global o local, pudiendo quedarse en este último. A su vez, el parámetro α permite controlar la convergencia del método.

En la fase de entrenamiento, los parámetros de cada modelo son ajustados iterativamente tratando de minimizar la función objetivo $f(x_n)$, que puede ser el error absoluto medio (MAE) o la raíz del error cuadrático medio (RMSE).

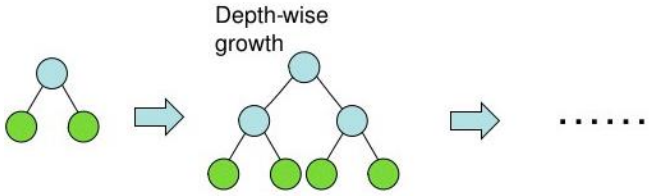


Figura 10 - Funcionamiento de un XGB

Cada modelo es comparado con el anterior. Si este modelo nuevo tiene mejores resultados, entonces se toma como base para realizar nuevas modificaciones. En caso contrario, se utiliza el modelo anterior.

3.2.3.2 – Parametrización del *XGBoost*

La mayoría de los parámetros tratan de regularizar los modelos "débiles", como puede ser la profundizad de cada uno de dichos modelos. En este caso, se ha tratado de ajustar los siguientes hiperparámetros:

- **n_estimators:** indica el número de modelos que se crearán de forma iterativa.
- **max_depth:** profundidad máxima de cada modelo.
- **Eta:** el ratio de aprendizaje para cada modelo.
- **Subsample:** parámetro que controla la proporción de muestreo aleatorio para cada árbol.
- **colsample_by_tree:** Utilizado para controlar la proporción del número de características muestreadas al azar.

3.2.5 – LGBM: Ligth Gradient Boosting

El modelo LGBM, desarrollado por Microsoft, es muy parecido al modelo XGB en cuanto a funcionamiento, basándose ambos en el descenso del gradiente.

La principal diferencia radica en la forma de modificar los modelos 'débiles' en las sucesivas iteraciones. LGBM modifica los árboles utilizando una expansión en profundidad, es decir, modificando los nodos de una rama. En cambio, el modelo XGB expande sus modelo en anchura, fijando un límite de profundidad como se puede observar en la Fig 11.

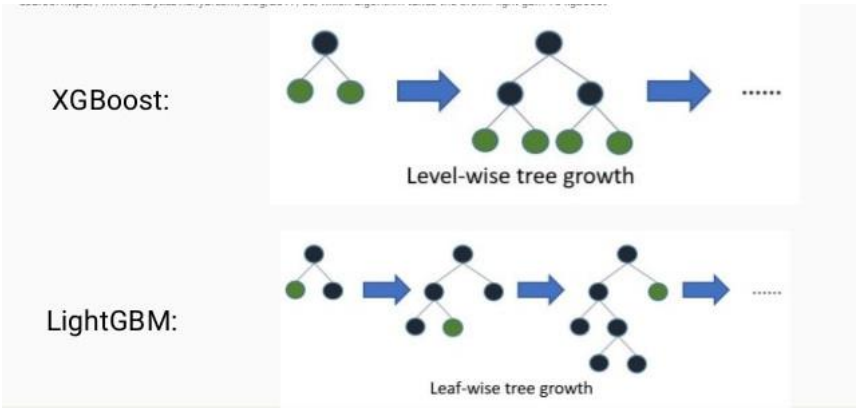


Figura 11 - Modificación de modelos

3.2.6 – ANN: Artificial Neuronal Networks

3.2.1.6 – MLP: Multi-layer Perceptron

El perceptrón multicapa es un algoritmo de aprendizaje supervisado que aprende una función del tipo $f: \mathcal{R}^m \rightarrow \mathcal{R}^o$ sobre el conjunto de entrenamiento, donde m es el número de dimensiones del vector de entrada y o el número de dimensiones del vector salida.

Dado un vector de características $X = \{x_1, x_2, \dots, x_m\}$ y una salida y, el MLP es capaz de aprender una función no linear tanto para clasificación como para regresión. Se diferencia de la regresión logística en que, entre el vector de entrada y salida, puede haber una o más capas no lineales conocidas como capas ocultas. La Fig 12 muestra un MLP con una capa oculta.

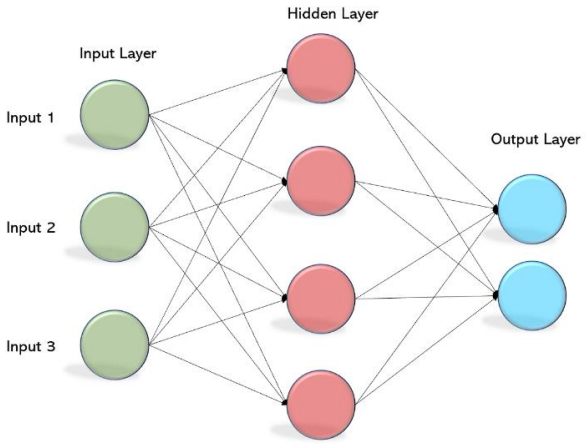


Figura 12 - Esquema de una ANN

La capa *input layer* consiste en una serie de neuronas $\{x_i \mid x_1, x_2, \dots, x_m\}$ que representan cada una de las variables de entrada. Cada neurona en la capa oculta transforma los valores de la anterior capa a través de una combinación lineal ponderada de la forma $w_1 x_1 + \cdots + w_m x_m$ multiplicada for una función de activación. Esta función, de tipo no lineal es de la forma $g: \mathcal{R} \rightarrow \mathcal{R}$.

3.2.1.6.1 – Formulación matemática

Dado un conjunto de entrenamiento $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ donde $x_i \in \mathcal{R}^n$ e $y_i \in \{0, 1\}$, una capa oculta aprende la función $f(x) = W_2(W_1^T x + b_1) + b_2$ donde $W_1 \in \mathcal{R}^m$ y $W_2, b_1, b_2 \in \mathcal{R}$ son parámetros del modelo. W_1, W_2 representan los pesos de las capa de entrada y capa oculta respectivamente, y b_1, b_2 representan la bias añadida a la capa oculta y capa de salida respectivamente.

La función de activación $g: \mathcal{R} \rightarrow \mathcal{R}$ se encarga de transformar la salida de neurona de la capa oculta. Las más conocidas están expresadas en la tabla 4.

Función	**Fórmula**
Tangente Hiperbólica	$f(x) = tanh(x)$
Relu	$f(x) = \max(0, x)$
Identidad	$f(x) = \mathrm{x}$
Logistic	$f(x) = \dfrac{1}{(1 + e^{-x})}$

Tabla 4 - Funciones de activación

MLP utiliza el Error Cuadrático como función de pérdida. Esta función se describe matemáticamente como

$$Loss(\hat{y}, y, W) = \frac{1}{2}||\hat{y} - y||_2^2 + \frac{\alpha}{2}\ ||W||_2^2$$

El modelo inicializa de forma aleatoria el vector de pesos y trata de minimizar la función de pérdida actualizando dicho vector. Después de calcularlo, realiza un propagación hacia atrás realizando una actualización de los pesos.

En el descenso del gradiente, el gradiente $\nabla Loss_W$ se utiliza para actualizar los pesos y se calcula a partir de *W*. Formalmente se expresa como:

$$W^{i+1} = W^i - \epsilon \nabla Loss_W^i$$

Siendo ϵ el ratio de aprendizaje e i la iteración.

El algoritmo finaliza cuando alcanza un número máximo de iteraciones, o cuando la mejora en la función de perdida es menor que cierta tolerancia.

3.2.6.2 – Parametrización del MLP

El modelo de perceptrón multicapa necesita ajustar los parámetros de forma precisa para poder converger a una solución óptima. Los principales parámetros involucrados en dicha tarea son:

- **Estructura de las capas ocultas**: se trata de especificar tanto el número de capas ocultas que va a tener nuestro modelo como el número de neuronas que tendrá cada una de las capas.
- **Solver**: optimizador utilizado para el cálculo de los pesos. Existen varios tipos de optimizadores como:
  - *Stochastic Gradient Descent:* actualiza los parámetros utilizando el gradiente de la función de pérdida.
  - *Adam:* similar a la anterior técnica, pero puede ajustar automáticamente la cantidad para actualizar los parámetros en función de estimaciones adaptativas.
  - *LBGFS:* aproxima la matriz hessiana, la cual representa la derivada parcial de segundo orden de la función.
- **Alpha:** permite regularizar el modelo tratando de evitar el sobreajuste a través de la penalización de pesos de orden alto.
- **Learning rate inicial:** controla el tamaño del paso durante la actualización de los pesos.

3.2.6.2– ENN: Elman Neuronal Network

Las redes *feedforward* como el perceptrón multicapa tienen ciertas limitaciones propias a su diseño que pueden solucionarse realizando cambios en su arquitectura. Las redes Elman son conocidas como redes recurrentes simples y son una mejora en el diseño de las redes *feedforward* gracias a la implementación de retroalimentación entre capas inmediatas contiguas.

Una red neuronal de Elman es una red con tres capas (entrada, oculta y salida) a las que se le añade una capa de 'contexto', siendo esta última la encargada de recordar los pesos anteriores, dotando de esta forma de una cierta memoria en el tiempo.

Cada una de las capas contiene una o varias neuronas que propagar la información entre capas contiguas, teniendo el mismo número de neuronas la capa de contexto y la capa oculta. La Fig 13 muestra una arquitectura de una red neuronal de Elman.

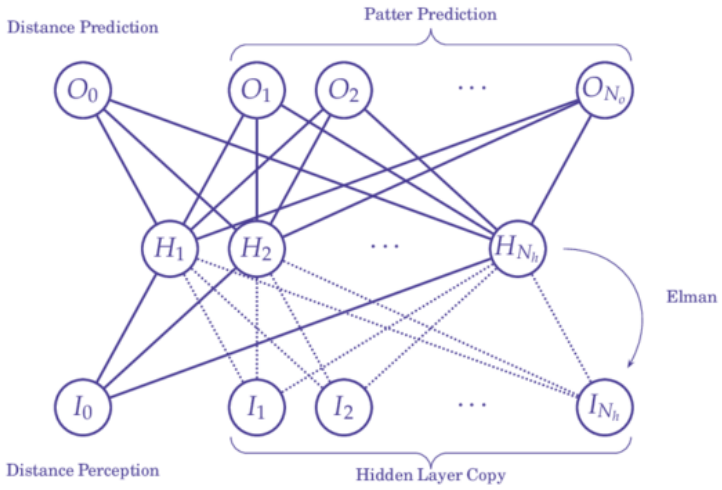


Figura 13 - Esquema de una SRN

El proceso de recuerdo/memoria se produce a través de las neuronas de la capa de contexto que son alimentadas por las neuronas pertenecientes a la capa oculta. Los pesos de las conexiones entre capa oculta y capa de contexto son fijos e iguales uno, permitiendo de esta forma mantener una copia de los pesos de la capa oculta en la iteración anterior.

Las ecuaciones matemáticas que definen el proceso realizado por las redes de Elman son:

$$h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h)$$

$$y_t = \sigma_y(W_y h_t + b_y)$$

Donde x_t es el vector entrada, h_t es el vector de la capa oculta, y_t es el vector de salida, W y U son las matrices de pesos, σ_h y σ_y las funciones de activación de cada una de las capas.

3.2.6.3– LSTM: Long short-term memory

Las redes neuronales recurrentes tienen una arquitectura que permite que parte de las activaciones de salida de una o varias neuronas retroalimenten las entradas de la misma neurona u otras situadas en el mismo o anterior nivel de procesamiento de la red.

Esta característica permite que la próxima vez que entrenemos la red con un ejemplo, estas activaciones calculadas anteriormente sirvan como entradas adicionales (Fig 14). Por ejemplo, si una red está formada por 5 neuronas de entrada y una capa oculta con 20 neuronas, las entradas totales de dicha red serán 20+5=25.

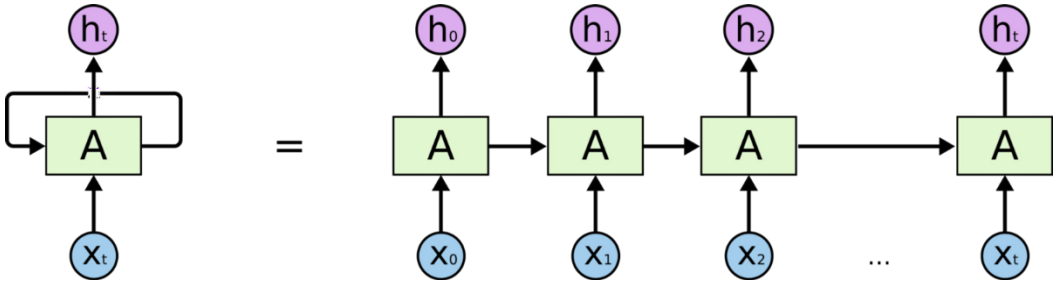


Figura 14 - Esquema de una LSTM

Una red recurrente en la que parte de sus neuronas se retroalimenta con la salida de otras presenta un problema (Fig 15): cada paso que se da, la proporción de información que se guarda de las iteraciones es la misma, haciendo que la relevancia que tiene un paso previo a la salida actual se pierda exponencialmente respecto a la cantidad de pasos.

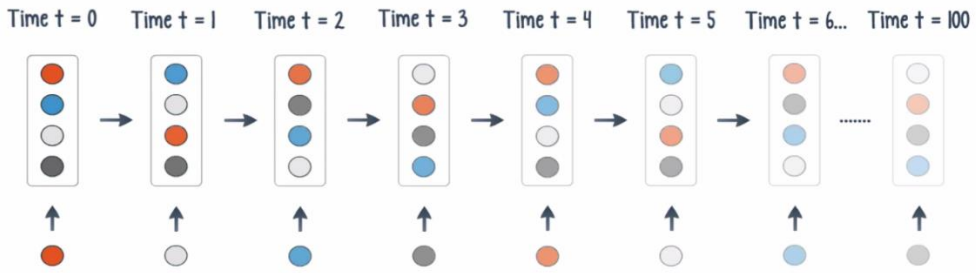


Figura 15 - Problema del desvanecimiento del gradiente

La aparición de la arquitectura LSTM está motivada para paliar dicho problema. Una red LSTM utiliza neuronas que permiten decidir qué información guardar y cuál se olvida en función del estado actual.

Una neurona LSTM está formada por una célula, una puerta de entrada, una puerta de salida y una puerta para olvidar. La célula es la encargada de "recordar" los valores en el tiempo. Cada una de las tres puertas se pueden entender como neuronas " convencionales".

En resumen, una neurona LSTM está formada por:

- Una puerta de salida *f*
- Una puerta de entrada *i*
- Una puerta de salida *o*
- Un vector de estado *h*

- Un vector de memoria *c*

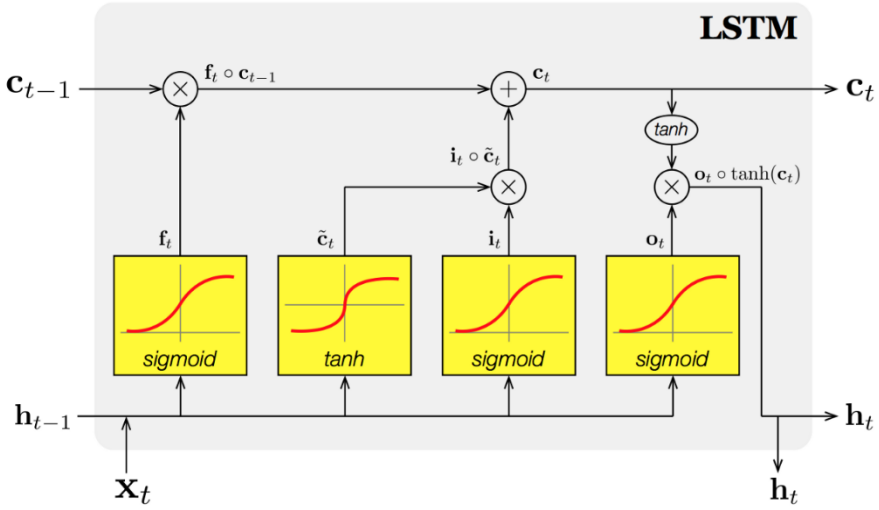


Figura 16 - Esquema de una célula LSTM

La idea principal de la célula es permitir a la red aprender a qué almacenar en la memoria a largo plazo, qué desechar y qué obtener de la misma. La imagen Fig 16 muestra como la memoria a largo plazo c_{t-1} pasa a través de las diferentes puertas. Inicialmente pasa por la puerta de olvido *f*, eliminando algunos de los datos almacenados. La siguiente puerta por la que pasa le permite añadir nueva información a través de las nuevas entradas y, el resultado c_t, se envía fuera de la célula.

Además, ese nuevo vector que contiene información relativa al pasado y presente se copia y pasa por la función de tangente hiperbólica para después pasarla por la puerta de salida. Este paso produce una memoria a corto plazo para dicho espacio temporal.

Las siguientes ecuaciones resumen cómo calcular el estado a largo plazo, corto plazo y su salida en un paso temporal *t*.

$$
\begin{aligned}
i_{(t)} &= \sigma(W_{xi}^T \cdot x_{(t)} + W_{hi}^T \cdot h_{(t-1)} + b_i) \\
f_{(t)} &= \sigma(W_{xf}^T \cdot x_{(t)} + W_{hf}^T \cdot h_{(t-1)} + b_f) \\
o_{(t)} &= \sigma(W_{xo}^T \cdot x_{(t)} + W_{ho}^T \cdot h_{(t-1)} + b_o) \\
g_{(t)} &= tanh(W_{xg}^T \cdot x_{(t)} + W_{hg}^T \cdot h_{(t-1)} + b_g) \\
c_{(t)} &= f_{(t)} \otimes c_{(t-1)} + i_{(t)} \otimes g_{(t)} \\
y_{(t)} &= h_{(t)} = o_{(t)} \otimes tanh(c_{(t)}).
\end{aligned}
$$

En donde:

- $W_{xi}, W_{xf}, W_{xo}, W_{xg}$ son las matrices de pesos para las conexiones entre las cuatro capas y el vector de entrada
- $W_{hi}, W_{hf}, W_{ho}, W_{hg}$ son las matrices de pesos para las conexiones entre las cuatro capas y el estado a corto plazo
- b_i, b_f, b_o, b_g son los términos de bias

4 – Análisis exploratorio de los datos

El análisis exploratorio tiene como objetivo identificar el modelo teórico más adecuado para representar la población de la cual proceden los datos muestreados.

Dicho análisis se basa en gráficos y estadísticos que permiten explorar la distribución de los datos, permitiendo identificar características tales como: valores anormales, concentraciones de valores, forma de la distribución, etc.

4.1 – Origen de los datos

El dataset ha sido extraído de una granja situada en Cononsyth, Escocia. El dataset cuenta originalmente con siete ficheros CSV que contienen información de la energía total producida por las placas fotovoltaicas en un período de 15 minutos.

Ilustración 1 - Granja de Cononsyth

El dataset recoge información desde 2011 hasta 2017, describiendo cada observación a través de características propias de una célula fotovoltaica como puede ser la potencia aparente, voltaje actual y potencia producida.

En total, el conjunto de datos está formado por más de 200.000 observaciones, cada una explicada a través de 19 variables. Las variables que explican cada una de las observaciones son las siguientes:

Date	VoltageACL1	EnergyFromBattery
ApparentPower	VoltageACL1L2	EnergyFromGrid
CurrentACL1	VoltageACL2	EnergyToGrid_
CurrentACL2	VoltageACL3	EnergyToBattery
CurrentACL3	VoltageDCMPP1	PV_Production
CurrentDCMPP1	ConsumedDirectly	
Energy	Consumption	

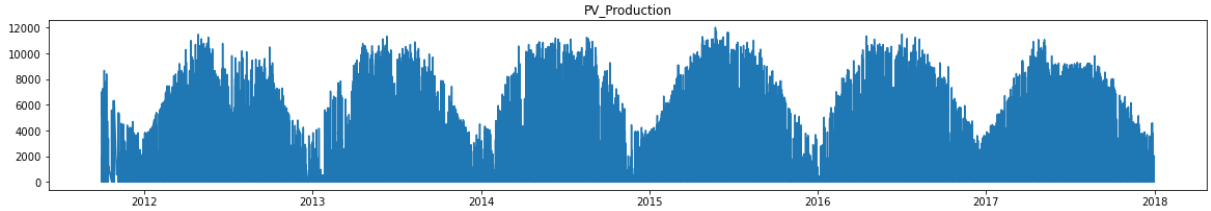
Desde un inicio no se ha utilizado ninguna de las variables salvo la variable *Date* y la variable *PV_Production*, variable objetivo del problema. El principal motivo por el que se han desechado el resto de las variables es el desconocimiento en tiempo real de dichas variables.

4.2 – Limpieza y tratamiento de los datos

El dataset contiene 200.000 observaciones explicadas a través de 16 variables. De estas dieciséis variables, quince corresponden a valores numérico de tipo real, mientras que la restante, la variable *Date*, corresponde al momento de captura de la observación. La variable objetivo del problema es *PV_Production*.

Cada observación se distancia de su antecesora en un intervalo temporal de quince minutos, obteniendo un total de 96 observaciones por día.

Aunque originalmente el dataset contiene 16 variables, se decide únicamente trabajar con aquellas que se disponen en tiempo real para realizar la predicción. En nuestro caso, únicamente trataremos con la variable *Date* y *PV_Production*.

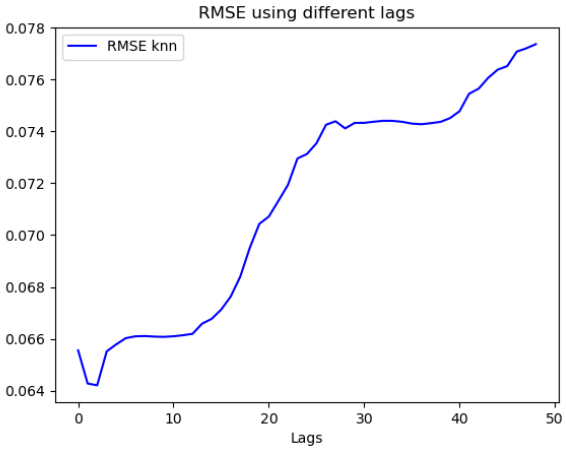


Gráfica 4 - Energía producida a lo largo del tiempo

Se ha decidido agrupar los datos en intervalos de 1 hora (originalmente están divididos en intervalos de 15 minutos) con el fin de reducir el número de observaciones. Esta agrupación contiene la suma de energía producida en ese intervalo, pasando el dataset a contener 54.000 observaciones.

Como la mayoría de los modelos que se van a utilizar no pueden extraer información de la variable temporal se decide dividir dicha variable en cuatro variables explicativas: *año, mes, día, hora*. A su vez, para tratar de informar del pasado a nuestros modelos, se crean tres variables que contendrán información sobre las tres horas anteriores a la hora de predicción llamadas: *t-3, t-2, t-1* haciendo uso de la función *shift* del paquete pandas que permite desplazar una posición todas las filas de una columna.

El número de lags se ha establecido creando varios modelos que implementen diferentes números de lags, seleccionando aquel número de lags que redujera el RMSE, seleccionando el valor de 3 lags como se puede ver en la gráfica 5.



Gráfica 5 - Selección de lags

4.3 – Análisis Univariante

La variable target *PV_Production* nos indica la potencia generada por las placas solares, siendo cada observación la energía producida en el intervalo de una hora.

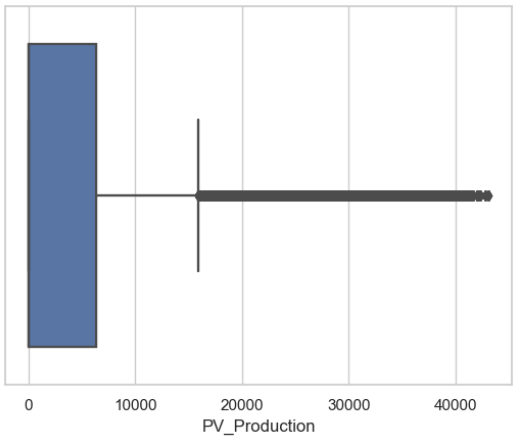
Los valores de esta variable se encuentran en el intervalo [0, 43122], siendo la unidad de medida el vatio-hora (Wh). Esta unidad de medida expresa la cantidad de energía que es capaz de producir durante un tiempo (una hora).

El máximo de potencia generada durante el intervalo de una hora se produce el 7 de mayo de 2017 a las 12 de la mañana, generando un total de 43 kWh. Los siguientes días con mayor producción de energía se muestran en la tabla 5.

Posición	Fecha	PV_Production
2	2015-05-26 12:00:00	42’8 kWh
3	2017-05-07 13:00:00	42’3 kWh
4	2015-06-10 13:00:00	42’2 kWh
5	2013-05-09 12:00:00	42’0 kWh

Tabla 5 - Intervalos con mayor producción PV

Poniendo en contexto dicha producción, se podría decir que en los 7 días que más energía se ha producido equivale al gasto de luz medio mensual de una vivienda en España.[28]

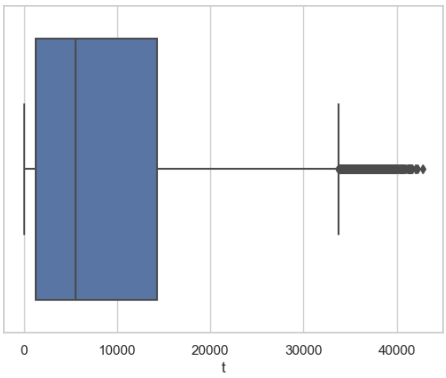


Gráfica 6 - Boxplot de PV

Analizando los datos a través de su gráfico de cajas, como se ve en la gráfica 6, se puede observar que la mayoría de los datos se concentran en el intervalo [0, 10000]. Esto se debe principalmente a que la mayoría de las horas la producción es nula debido a la ausencia del Sol.

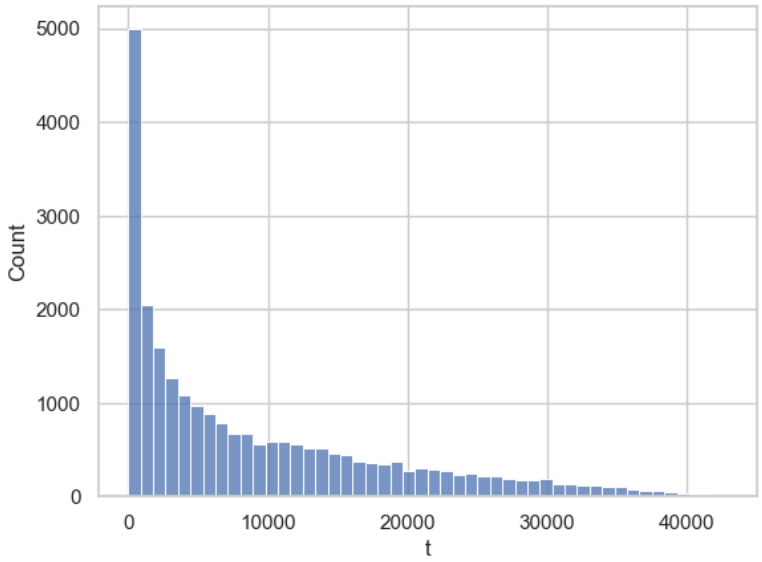
Si se eliminan aquellos valores obtenidos durante la noche para estudiar el resto de los valores, se obtiene el gráfico 7. Se puede observar que el número de outliers se reduce de forma notable respecto al anterior gráfico. La mayor parte de las observaciones, el 75%, tiene un valor menor de 150 kWh.

A la vista del gráfico se puede intuir que la variable no sigue una distribución normal.



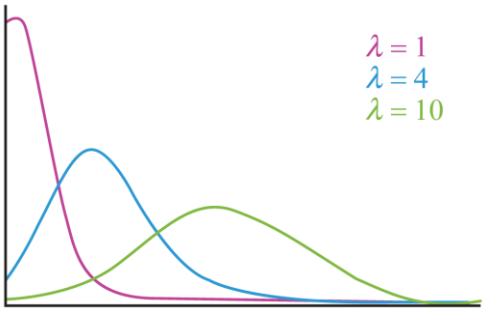
Gráfica 7 - Boxplot de PV tras limpieza

El histograma (gráfica 8) nos muestra que la mayoría de los datos tienen valores cercanos a cero, como se ha podido comprobar con anterioridad a partir de los gráficos de cajas.



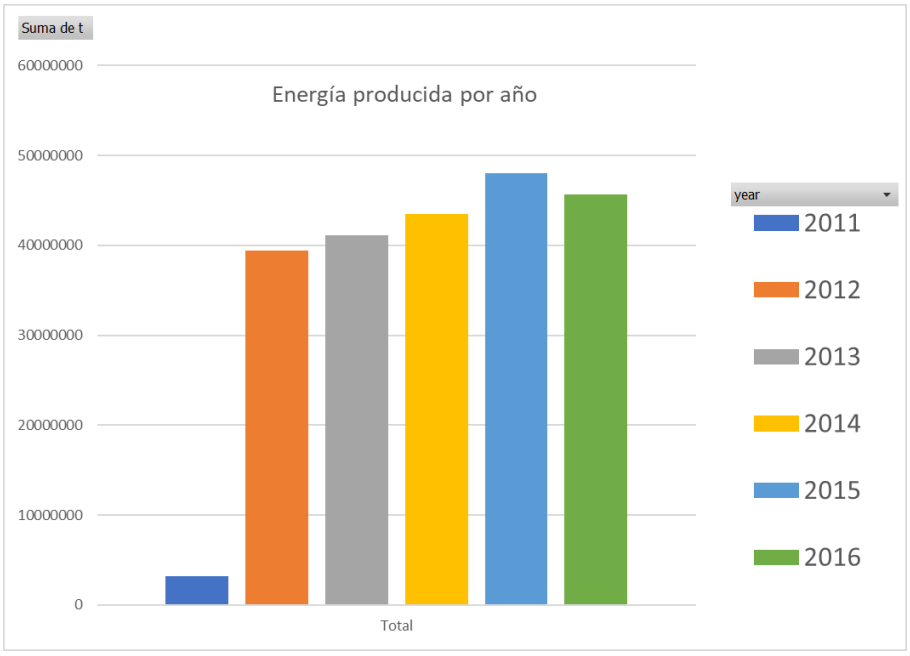
Gráfica 8 - Histograma de PV

La distribución del conjunto de datos se asemeja a una distribución exponencial negativa y/o a una distribución de Poisson.



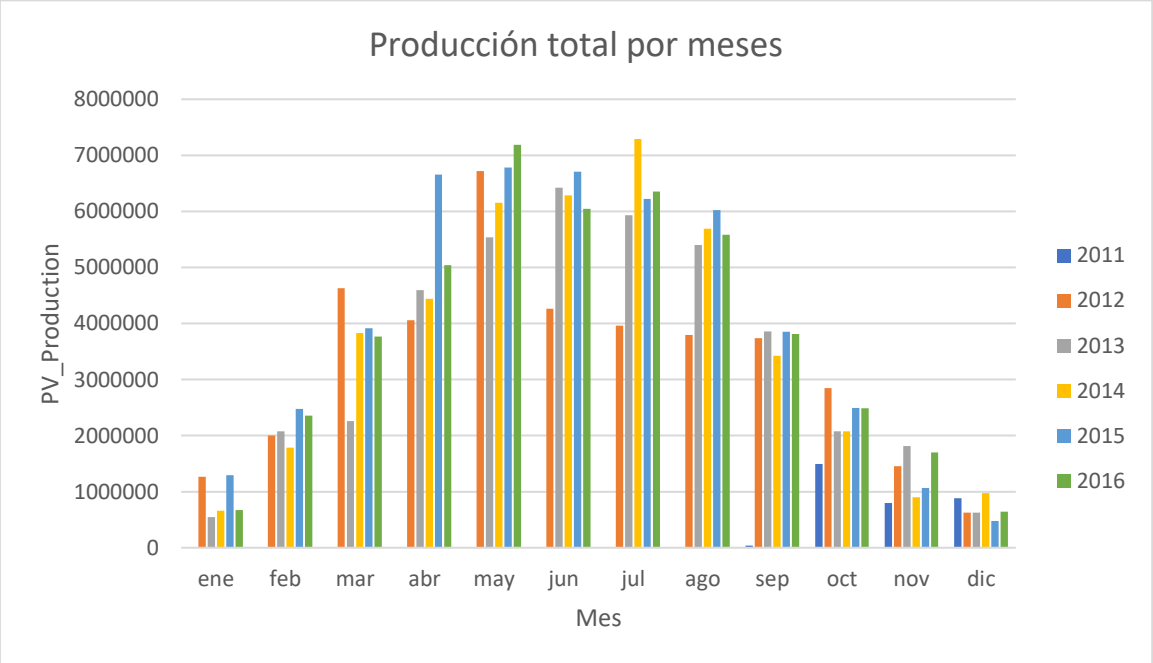
4.3.1 – Graficando la variable

La producción anual no varía de un año a otro, ya que el clima tiene una componente clara estacional. De esta forma, la predicción de energía producida en un horizonte anual resultará muy semejante a la producida años anteriores (grafica 9). El año de 2011 aparece con poca producción total debido a que empezó a medirse a partir octubre.



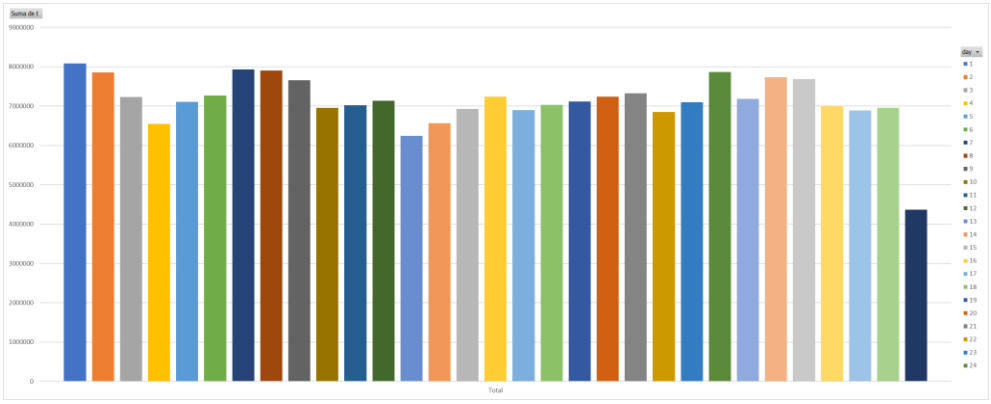
Gráfica 9 - producción PV a lo largo de los años

La producción total de energía se concentra en los meses de mayo a agosto como se puede observar en el gráfico 10. Destaca negativamente el verano de 2012, donde la producción de energía se situó a niveles muy inferiores al resto de años para las mismas fechas (junio, julio, agosto)



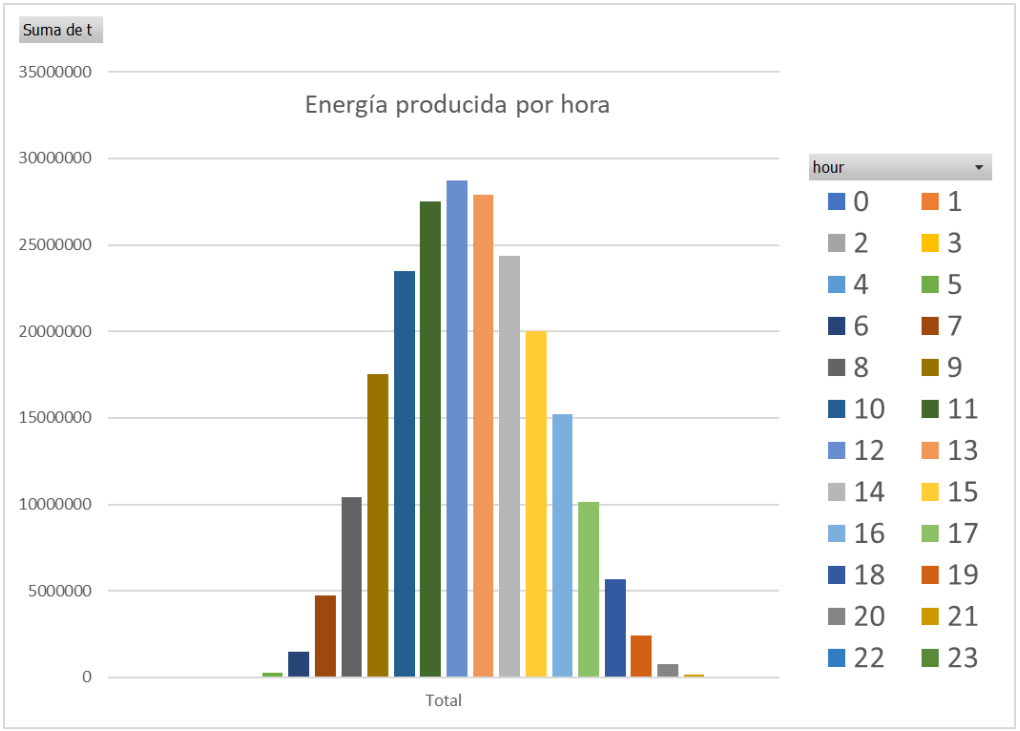
Gráfica 10 - Producción de PV mensual

Desagregando a nivel diario (gráfica 11) no existen diferencias significativas entre lo producido un día de la semana u otro, ya que estamos analizando la producción. En cambio, si mirásemos el consumo de energía sí notaríamos diferencias entre días de la semana como un lunes o fines de semana.



Gráfica 11 - Producción PV diaria

Desagregando a nivel horario (gráfica 12) podemos ver que las horas que más energía corresponde a las horas donde la actividad solar, y por tanto, la irradiación es mayor. Esto se produce en el intervalo 10am a 14pm.



Gráfica 12 - Producción PV horaria

4.4 – Análisis Multivariante

En esta sección se muestran las diferentes relaciones existentes entre las variables del problema a través el gráfico de correlaciones. A su vez, se utilizará en algoritmo de RandomForest para obtener la importancia de cada una de las variables para tratar de predecir la variable objetivo.

4.4.1 – Gráfico de Correlaciones

La matriz de correlación es una matriz cuadrada *NxN* constituida por los coeficientes de correlación de cada pareja de variables; de manera que tendrá unos en la diagonal principal (la correlación es total), y en los elementos no diagonales los correspondientes coeficientes de correlación

La correlación entre dos variables evalúa la tendencia (creciente o decreciente) entre los datos. Dos variables están asociadas cuando una variable nos da información acerca de la otra, aunque no implica causalidad. La correlación entre dos variables viene de expresado de la siguiente forma:

$$r = \frac{\sum[(x_i - \bar{x}) * (y_i - \bar{y})]}{\sqrt{\sum(x_i - \bar{x})^2 * \sum(x_i - \bar{x})^2}}$$

El coeficiente de correlación es un valor comprendido en el intervalo [-1, 1] que se puede interpretar como:

- El signo nos indica la dirección de la relación, un valor positivo nos indica una relación directa, mientras que un valor negativo nos indica una relación negativa.
- La magnitud nos indica la fuerza de la relación, siendo en los extremos del intervalo donde más fuerte será la tendencia de las variables.

La matriz correlación relativa a las variables *[año, mes, día, hora, t-3, t-2, t-1, t]* se muestra en la gráfica 13.

	year	month	day	hour	t-3	t-2	t-1	t
year								
month	-0.08							
day	0.00	-0.03						
hour	0.00	-0.01	0.00					
t-3	0.08	-0.08	0.00	0.55				
t-2	0.08	-0.08	0.00	0.40	0.88			
t-1	0.08	-0.08	0.00	0.20	0.69	0.87		
t	0.08	-0.08	0.00	-0.03	0.46	0.67	0.86	

Gráfica 13 - Matriz de Correlación

El gráfico 13 nos muestra una fuerte relación entre las variables *t* y las variables predecesoras *t-3, t-2, t-1* (información sobre las tres horas anteriores a la hora de predicción), siendo esta última la que mayor correlación tiene.

4.4.2 – Importancia de las variables

Una de las utilidades que nos brindan los modelos basados en bosques de árboles es conocer qué variables han tenido una mayor relevancia a la hora de realizar las bifurcaciones en los árboles del modelo.

La importancia de cada una de las variables se calcula a través de *mean decrease impurity* basado en reducción de la varianza en el caso de regresión. Cuantifica el incremento total de la pureza de los nodos en los que participa la variable predictora.

Aquellas variables que obtienen una mayor reducción de la métrica escogida comprenderán una mayor contribución en la generación del modelo.

Variable	**Importancia**
t-1	0.860
hour	0.065
t-2	0.025
t-3	0.022
month	0.016
day	0.015
year	0.007

Tabla 6 - Tabla de importancia

Se puede observar en tabla 6 que la variable con mayor importancia es *t-1*, la variable que guarda la energía producida durante la hora anterior, seguido de la hora. La hora cobra relevancia ya que la energía producida medida durante 24 horas muestra una clara estacionalidad como se ha visto en el análisis univariante.

5 – Modelos en la práctica

Los modelos explicados anteriormente de forma teórica se han implementado en Python utilizando principalmente las librerías de *TensorFlow* y *scikit-learn*. Ambas librerías contienen los métodos necesarios para llevar a la práctica dichos modelos.

A su vez, en esta sección se pueden encontrar las métricas utilizadas para la comparación de modelos, así como una pequeña introducción a *MLFlow*, tecnología orientada a la gestión de modelos de aprendizaje automático.

5.1 – Métricas usadas

Para evaluar cuán bien responden los modelos ante nuevas instancias y así poder realizar predicciones a futuro de una mejor forma, la literatura ha propuesto numerosas métricas como el RMSE *(Root Mean Squared Error)*, MAE *(Mean Absolute Error),* etc.

En el presenten caso de estudio se han utilizado el R2 (coeficiente de determinación), MAE, RMSE y nRMSE.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2}$$

$$RMSE = \sqrt{\frac{\sum(\hat{y} - y)^2}{n}}$$

$$MAE = \frac{1}{n}\sum|\hat{y} - y|$$

$$nRMSE = \frac{RMSE * 100}{y_{max}}$$

Donde *n* es el número total de observaciones, $\hat{y}$ es la predicción de nuestro modelo e y_{max} es el valor máximo observado.

5.2 – Preprocesamiento de los datos

La estandarización del conjunto de datos es un requisito común para muchos modelos de aprendizaje automático, sobre todo aquellos basados en distancias por *kNN*. A su vez, las redes neuronales son sensibles a las escalas de sus entradas, siendo más sensibles cuando se utilizan funciones de activación como la sigmoidea.

El escalado permite desplazar los valores de una variable para que se concentren en el intervalo [0-1]. La fórmula explicita que utiliza *MinMaxScaler* es:

$$z = \frac{x - x_{min}}{x_{max} - x_{min}}$$

siendo x_{min} el valor mínimo y x_{max} el valor máximo.

5.3 – Feature Engineering

En este sección se describen diferentes modificaciones que se han realizado para tratar de ajustar los datos al problema específico. El conjunto de datos está descrito a partir de las variables [*month, day, hour*], variables que tratan de aportar información de carácter temporal a nuestros modelos.

El principal problema es que es que los valores de dichas variables no expresan el carácter estacional de las mismas (Fig 17). Por ejemplo, la distancia entre dos observaciones cuyo valor en la variable *month* sea 1 (enero) y 12 (diciembre) es igual a 11, cuando únicamente les separa un mes (son de diferentes años).

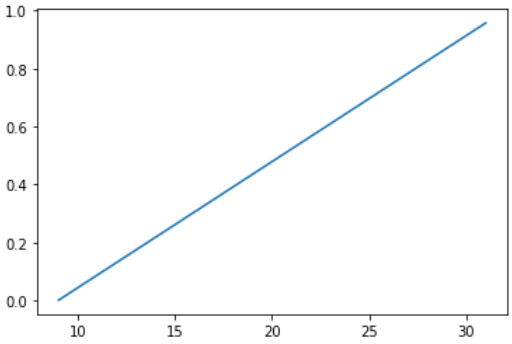


Figura 17 - Representación secuencial del día

La solución a esta problemático consiste en aumentar la dimensión de cada una de las variables haciendo uso de las funciones trigonométricas de seno y coseno. De esta forma, conseguimos crear un par de variables cíclicas como se puede ver en la Fig 18

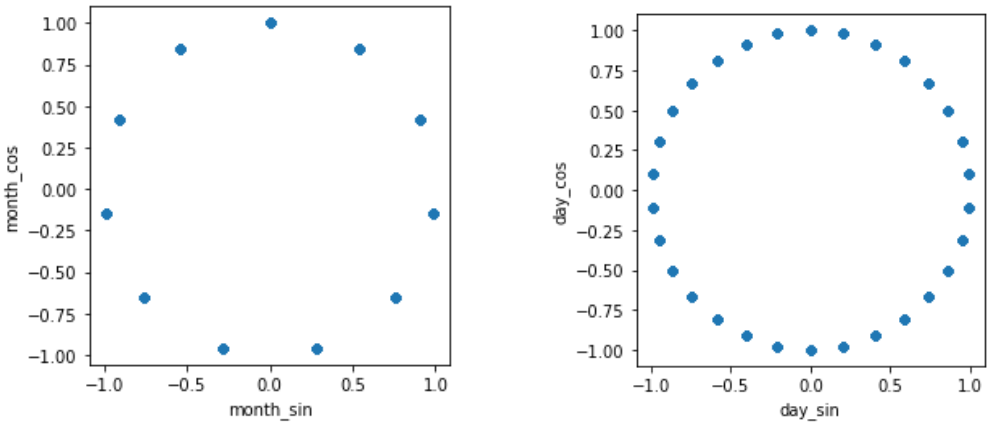


Figura 18 - Representación cíclica del mes y día

De esta forma, el nuevo dataset estará explicado por un total de diez variables, siendo seis variables de caracteres temporal (dos por cada variable temporal anterior [month, day, hour]) junto a tres variables que nos indican los valores que tomó la variable objetivo durante las tres últimas horas.

5.3 – MLFlow: Machine Learning Lifecycle

MLFlow, tal y como se definen ellos mismos, es *una plataforma de código abierto para la gestión del ciclo de vida de modelos de Machine Learning, que se puede usar y aplicar tanto en problemas con pocos datos o más clásicos como ante entrenamientos más complejos.*

El principal objetivo es agrupar en una única herramienta todo el ciclo de vida de desarrollo (Fig 19). Una de las principales ventajas es la gestión de hiperparámetros, *feature Engineering*, simplificándolo a una base de datos.

Figura 19 – MLFlow

La Fig. 19 muestra los diferentes módulos en los que se divide MLFlow. Estos son:

- **MLFlow Tracking**: permite realizar un seguimiento de los hiperparámetros para encontrar aquellos que proporcionan mejores resultados.
- **MLFlow Projects**: permite obtener información del modelo resultado, el cual se empaquete con las dependencias necesarias para ser puesto en producción.
- **MLFlow Models:** permite lanzar nuestro modelo a producción, ejecutándolo en un cluster y siendo accesible desde una API Rest.
- **MLFlow Registry:** permite almacenar o registrar modelos de una forma versionada.

5.4 – Búsqueda de hiperparámetros óptimos

Cuando creamos modelos de aprendizaje máquina, cada modelo presenta una serie de configuraciones que permiten modificar dicho modelo para obtener un mayor ajuste al problema objetivo. La mayoría de las veces se desconoce cuál es el modelo óptimo y el científico de datos se ve obligado a buscar entre las posibilidades.

Antes de buscar cuáles son los mejores hiperparámetros que modelan nuestros datos, hay que realizar un inciso en la importancia de dividir los datos en tres subconjuntos: entrenamiento, validación y test (Fig 20).

En el caso de la producción energética se ha decido dividir el conjunto de datos, aprovechando la estacionalidad, a partir de los años. De esta forma el conjunto de entrenamiento estará formado por las observaciones recogidas entre 2011 y 2015; el conjunto de validación lo forman las observaciones recogidas durante el 2016; y el conjunto de test lo conforman los registros de 2017.

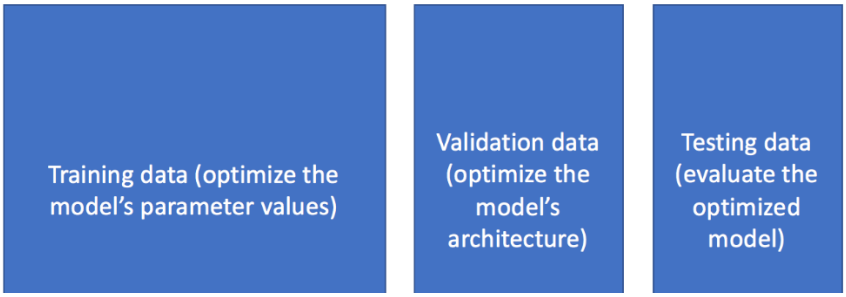


Figura 20 - Separación entre train, val, test

5.4.1 – Búsqueda Aleatoria

La búsqueda aleatoria consiste en utilizar una distribución estadística para cada hiperparámetro desde la cual se eligen los posibles valores de forma aleatoria.

Esta búsqueda se basa en la idea de que, en la mayoría de los casos, los hiperparámetros no son igual de importantes de cara a obtener mejores resultados [Random Search for Hyper-Parameter Optimization].

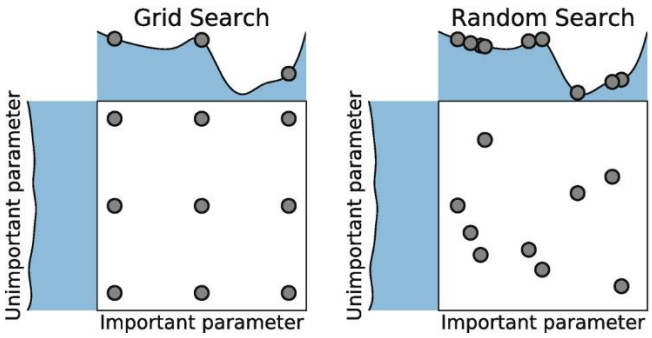


Figura 21 - Tipos de búsqueda de hiperparámetros

Una de las principales ventajas de dicha búsqueda es el ahorro de tiempo que se gana respecto a la búsqueda codiciosa, dando un mayor peso a ciertos hiperparámetros, en contraposición de la búsqueda ambiciosa.

5.4.2 – Búsqueda Codiciosa

También llamada búsqueda de exhaustiva o de rejilla entrena un modelo por cada combinación posible de valores. Cada uno de los hiperparámetros se trata con la misma importancia.

Aunque este modo de *hyptertuning* permite obtener buenos resultados, los modelos de machine learning actuales disponen de una cantidad alta de hiperparámetros, penalizando dicha búsqueda debido a que el tiempo de cómputo aumenta de forma considerable.

5.4.2.1 – Regresión Lineal: ElasticNet

Los principales hiperparámetros que se van a tratar de ajustar en la regresión lineal son los propios de aplicar la regularización ElasticNet: *Alpha* y *l1_ratio*

Hiperparámetros:

- Alpha = [0.00, 0.05, 0.10, …, 1.00]
- L1_Ratio = [0.00, 0.05, 0.10, …, 1.00]

La tabla 7 muestra los mejores resultados obtenidos a través del modelo de regresión lineal con regularización ElasticNet

Alpha	l1-Ratio	MAE	RMSE	R2
0.0	1.00	0.046	0.079	0.838
0.0	0.95	0.046	0.079	0.838
0.0	0.90	0.046	0.079	0.838
0.0	0.85	0.046	0.079	0.838

Tabla 7 - Hiperparámetros RL

5.4.2.2 – KNN: K-Nearest Neighbor

El principal hiperparámetro que hay que ajustar es *k*, el número de vecinos cercanos que se utilizan para calcular la nueva observación. Además, se ajusta el parámetro *weights*, utilizado para evitar que un punto vecino lejano influya en el resultado de la misma forma que uno cercano.

Hiperparámetros:

- K = [3, 4, 5, 6, …, 100]
- Weigths = [*distance, uniform]*

La tabla 8 muestra los mejores resultados obtenidos a través del modelo KNN.

K	Weigths	MAE	RMSE	R2
16	*distance*	0.03338	0.07053	0.8768
17	*distance*	0.03394	0.07055	0.8768
22	*distance*	0.03414	0.07056	0.8767
20	*distance*	0.03408	0.07057	0.8767
18	*distance*	0.03401	0.07058	0.8767

Tabla 8 - Hiperparámetros KNN

5.4.2.3 – DT: Decision Tree

Se ha ajustado el modelo tratando de modificar los siguientes hiperparámetros:

- Criterion: [*mse, friedman_mse]*
- Max_depth: [1, 2, 3, …, 10]
- Min_samples_split: [2, 4, 8, 16, 32]
- Min_samples_leaf: [1, 2, 4, 8, 16, 32]

La tabla 9 muestra los mejores resultados obtenidos a través del modelo DT. Siendo el mejor modelo un árbol de profundidad 10.

Criterion	Max	Min_samples_split	Min_samples_leaf	MAE	RMSE	R2
friedman_mse	10	16	32	0.03367	0.07034	0.8775
friedman_mse	10	8	32	0.03367	0.07034	0.8775
friedman_mse	10	4	32	0.03367	0.07034	0.8775
mse	10	32	16	0.03367	0.07034	0.8775
mse	8	8	16	0.03367	0.07034	0.8775

Tabla 9 - Hiperparámetros DT

5.4.2.4 – LGBM: Ligth Gradient Boosting

El modelo LGBM, modelo desarrollado por Microsoft basado en el descenso del gradiente, se ha ajustado tratando de modificar los siguientes hiperparámetros:

- Num_leaves: [10, 20, 30, 40, 50]
- N_estimators: [1, 2, 3, …, 10]
- Max_depth: [100, 200, …, 1000]
- Learning_rate: [0.0001, 0.001, 0.01, 0.1, 0.2]

La tabla 10 muestra los mejores resultados obtenidos a través del modelo LGBM.

Num_Leaves	N_Estimators	Max_Depth	Learning_Rate	MAE	RMSE	R2
20	100	10	0.10	0.03103	0.06432	0.8976
40	600	7	0.01	0.03090	0.06432	0.8976
30	800	7	0.01	0.03089	0.06433	0.8975
20	800	7	0.01	0.03119	0.06433	0.8975
20	800	7	0.01	0.03125	0.06434	0.8975

Tabla 10 - Hiperparámetros LGBM

5.4.2.5 – XGB: Extreme Gradient Boosting

El modelo XGB, modelo basado árboles, se ha ajustado tratando de modificar los siguientes hiperparámetros:

- eta: [0.01, 0.05, 0.1, 0.15, 0.3]
- N_estimators: [100, 200, …, 1000]
- Max_depth: [1, 2, 3, …, 10]
- Subsample: [0.7, 0.8, 1.0
- Colsample_by_tree: [0.8, 1]

La tabla 11 muestra los mejores resultados obtenidos a través del modelo XGB

Eta	Subsample	N_Estimators	Max_Depth	Colsample_bytre	MAE	RMSE	R2
0.01	0.7	900	6	0.8	0.03070	0.06420	0.8980
0.01	0.7	700	6	0.8	0.03089	0.06420	0.8980
0.01	0.7	700	7	0.8	0.03074	0.06421	0.8979
0.01	0.7	1000	6	0.8	0.03070	0.06423	0.8978
0.01	1.0	900	6	0.8	0.03078	0.06427	0.8977

Tabla 11 - Hiperparámetros XGBM

5.4.2.6 – MLP: Multi-Layer Perceptron

El modelo MLP, basado en redes neuronales, se ha ajustado tratando de modificar los siguientes hiperparámetros:

- Hidden_layer_sizes: [(10,), (20,), (30,), …, (100, 100, 100)]
- Alpha: [0.0001, 0.001, 0.01, 0.1]
- Activation: [r*elu, tanh*]
- Learning_rate_init: [0.001, 0.01. 0.1]

La tabla 12 muestra que el mejor modelo está formado por dos capas ocultas, teniendo cada una 30 y 60 neuronas respectivamente.

Hidden_layer_sizes	Alpha	Activation	Learning_rate_init	MAE	RMSE	R2
(30, 60)	0.001	*relu*	0.01	0.03843	0.07085	0.8757
(70, 80, 100)	0.001	*relu*	0.01	0.03620	0.07158	0.8732
(70, 80, 90)	0.001	*relu*	0.01	0.03718	0.07160	0.8731
(50, 60, 90)	0.001	*relu*	0.01	0.03672	0.07166	0.8729
(60, 80)	0.001	*relu*	0.01	0.03600	0.07173	0.8726

Tabla 12 - Hiperparámetros MLP

5.4.2.7 – ENN: Elman Neuronal Network

Este tipo de red neuronal recurrente se ha ajustado utilizando diferentes números de neuronas, siendo k1 el número de neuronas de la capa 1 y k2 el número de neuronas de la capa 2, siguiendo una arquitectura de una capa oculta como se puede observar en la tabla 13.

K1	K2	MAE	RMSE	R2
300	300	0.034	0.07064	0.87
100	100	0.035	0.07148	0.87
500	-	0.038	0.07319	0.86
100	-	0.04055	0.07541	0.85
50	-	0.04057	0.07678	0.85

Tabla 13 - Hiperparámetros SRN

5.4.2.8 – LSTM: Long-short term memory

Este tipo de red neuronal recurrente se ha ajustado utilizando diferentes números de capas ocultas, estando formada cada capa oculta por una red LSTM. La tabla 14 muestra la combinación de mejores resultados, siendo el mejor aquel con dos capas ocultas de 300 neuronas cada una.

K1	K2	MAE	RMSE	R2
300	300	0.04048	0.07853	0.85
100	100	0.04065	0.07875	0.85
500	-	0.04194	0.08044	0.84
100	-	0.04384	0.08056	0.84
50	-	0.04401	0.08079	0.84

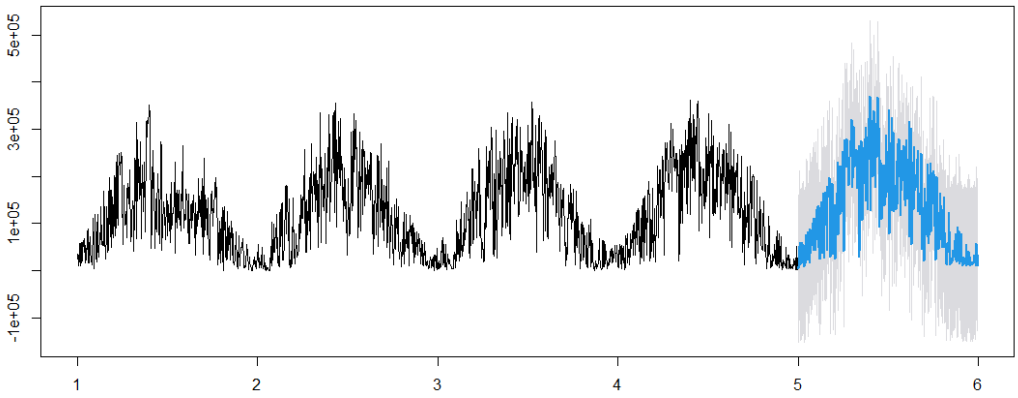
Tabla 14 - Hiperparámetros LSTM

5.4.2.9 – SARIMA: Seasonal autoregressive integrated mooving average

El modelo SARIMA se ha calculado utilizando el lenguaje de programación R, específicamente la función *auto.arima* del paquete *forectast*.

El principal inconveniente de este modelo es el alto coste de computación ante datos con una estacionalidad alta, siendo en este caso de 365*24. Ante esta situación, se ha decidido no utilizar el modelo debido al alto número de horas de entrenamiento que supone.

Si se agrupan las observaciones a nivel diario, el modelo seleccionado por la función es ARIMA(3,0,3)(0,1,0)[365], obteniendo como resultado un R2 de 0.87.



6 – Comparando resultados

En esta sección se presentan los diferentes resultados obtenidos, siendo el horizonte de predicción utilizado de un intervalo de hora. Es decir, predeciremos el total de energía fotovoltaica producida durante la próxima hora utilizando los datos de las tres últimas horas, además de las variables temporales.

El conjunto de test utilizado está formado por las observaciones registradas a lo largo del año 2017.

En términos generales, los resultados muestran una clara semejanza a los valores esperados. En la tabla 15 se muestran los resultados obtenidos. Cabe recordar que los datos han sido previamente escalados en el intervalo [0, 1], por lo que las métricas también lo están. Se puede observar que las redes neuronales obtienen resultados parejos a los modelos tradicionales, siendo XGB el mejor modelo obtenido.

Modelo	MAE	RMSE	R2
Regresión Lineal	0.04586	0.07824	0.8362
KNN	0.03429	0.06967	0.8701
DecisionTree	0.03346	0.06886	0.8731
XGB	**0.03056**	**0.06277**	**0.8945**
LGBM	0.03077	0.06292	0.8941
Multilayer Perceptron	0.03853	0.07044	0.8672
Red de Elman	0.03628	0.07087	0.8656
LSTM	0.03593	0.07173	0.8623

Tabla 15 - Resultados de los modelos

Debido al elevado número de observaciones para el conjunto de test, se ha decidido mostrar únicamente la primera semana de mayo como predicción horaria (Fig 23). El resto de los gráficos muestran las predicciones a nivel hora agrupadas en formato semanal (Fig. 22) y mensual (Fig 24)

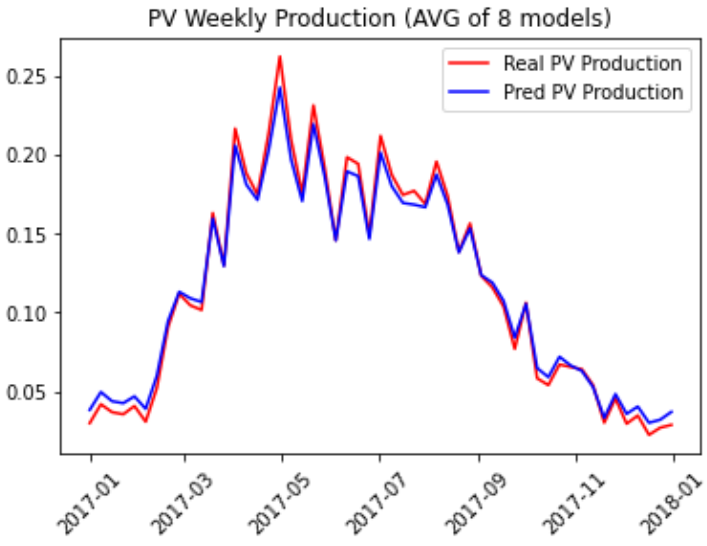


Figura 22 - predicción horaria agrupada a nivel semana

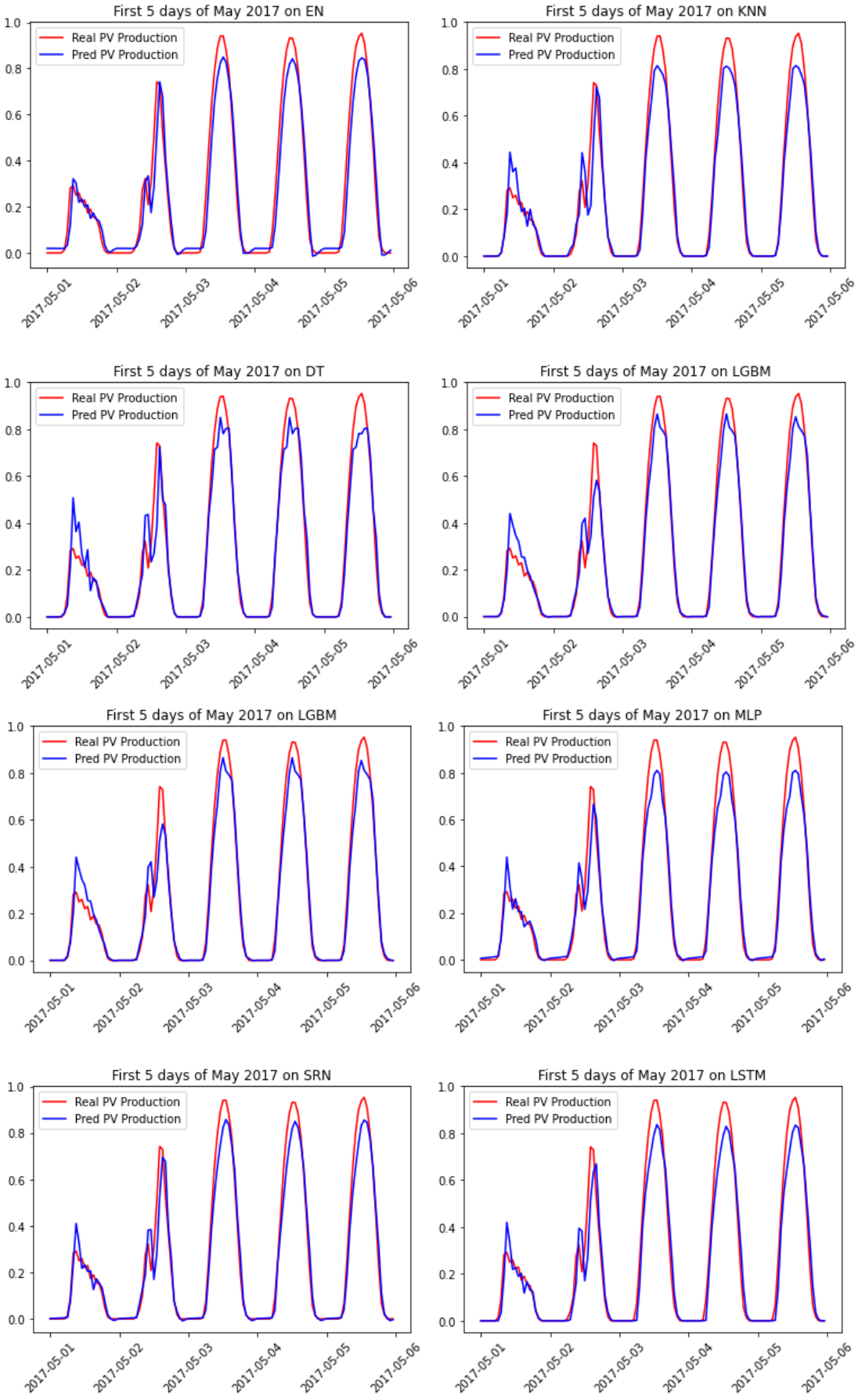


Figura 23 - predicción horaria de los primeros 5 días de mayo 2017

Los modelos se ajustan de una forma correcta al conjunto de test, aunque los picos de producción obtenidos durante las horas de máxima irradiancia solar están siendo subestimados por los modelos. Esto lo podemos comprobar agrupando por mes (Fig 24), donde vemos una subestimación durante los meses con mayor horas de sol.

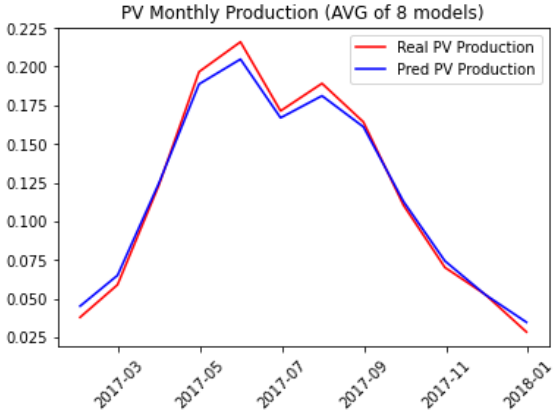
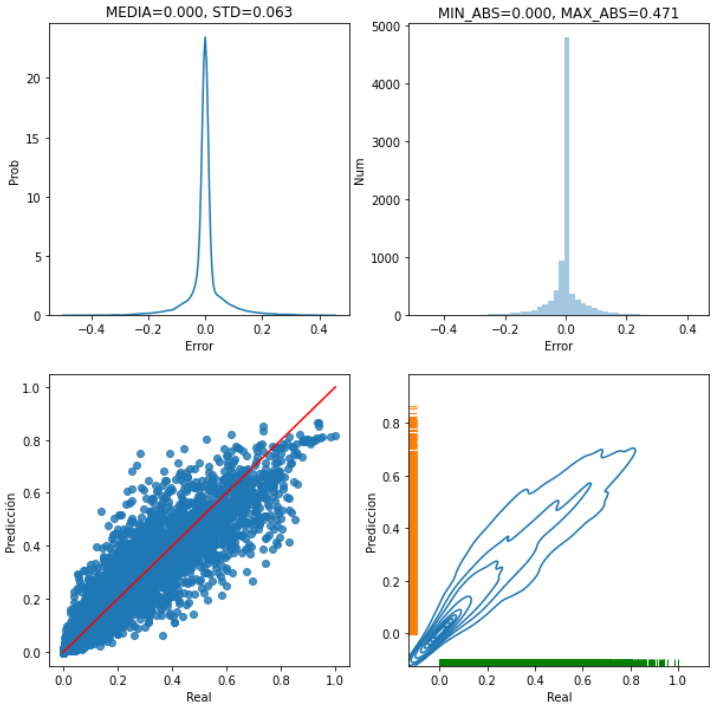


Figura 24 - Predicción horaria agrupada a nivel semana

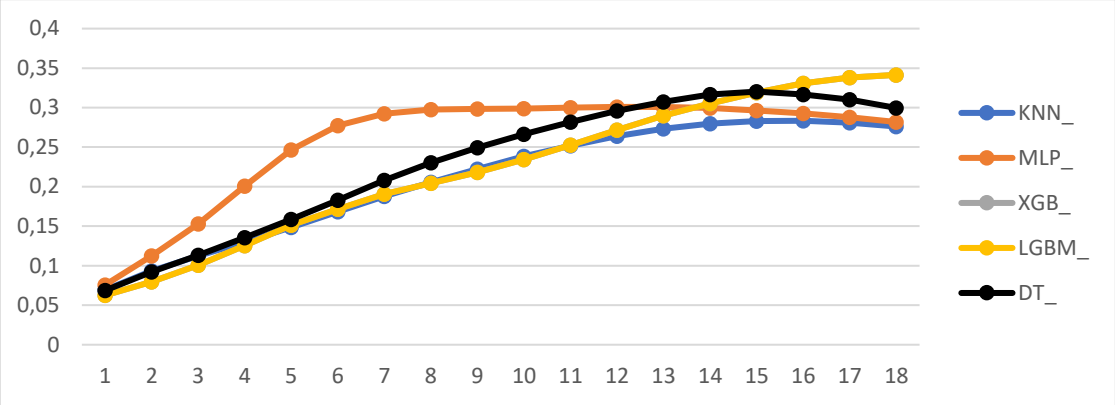
A su vez, se puede observar (Fig 23) que el primer día parece estar nublado debido a que se produce la mitad de energía que el resto de los días. Ante esta situación, los modelos sobreestiman la producción, aunque gracias a trabajar con información de las tres últimas horas logran corregir esa sobreestimación, fallando en las primeras horas únicamente.

Analizando los errores obtenidos por nuestro mejor modelo, el modelo XGB, se puede observar (Fig 25) que el modelo no tiene a sobreestimar o subestimar determinadas instancias, salvo a partir de un valor real mayor de 0.7 donde empieza a subestimar los valores.



Gráfica 14 - Gráficos de errores del modelo XGB

Utilizando los mejores cinco modelos, se ha calculado el RMSE de predicción en horizontes más lejanos, siendo el horizonte máximo de predicción de 18 horas. La gráfica 15 muestra la evolución del error obtenido, aumentando cada vez que el horizonte de predicción aumenta.



Gráfica 15 - RMSE en diferentes horizontes temporales

Se puede observar que, para horizontes de predicción de tres o menos horas, los modelos obtienen un error aceptable ya que sigue teniendo entre sus variables *t-3*, *t-2*, *t-1* al menos un valor real de la serie.

En cambio, a partir de un horizonte igual a 4, las predicciones se basan en anteriores predicciones, perdiendo información de la situación real. En ese momento, queda a decisión del técnico decidir si dicha información resulta de interés asumiendo que puede ser errónea.

t	KNN	MLP	XGB	LGBM	DT
1	0.0693	0.0757	**0.0625**	0.0627	0.0687
2	0.0933	0.1125	**0.0798**	0.0799	0.0920
3	0.1126	0.1530	**0.1005**	**0.1005**	0.1134
4	0.1300	0.2008	**0.1255**	0.1258	0.1354
5	**0.1484**	0.2464	0.1512	0.1518	0.1585
6	**0.1686**	0.2772	0.1716	0.1720	0.1828
7	**0.1878**	0.2922	0.1906	0.1911	0.2079
8	0.2055	0.2975	0.2042	0.2042	**0.2305**
9	0.2224	0.2984	0.2185	**0.2182**	0.2492
10	0.2382	0.2988	**0.2344**	0.2348	0.2663
11	0.2530	0.2999	**0.2524**	0.2525	0.2817
12	**0.2638**	0.3009	0.2718	0.2718	0.2958
13	**0.2732**	0.3009	0.2897	0.2897	0.3071
14	**0.2797**	0.2994	0.3054	0.3054	0.3166
15	**0.2829**	0.2964	0.3192	0.3192	0.3202
16	**0.2834**	0.2926	0.3306	0.3306	0.3168
17	**0.2810**	0.2877	0.3382	0.3382	0.3100
18	**0.2762**	0.2819	0.3414	0.3414	0.2995

Tabla 16 - RMSE en diferentes horizontes

La tabla 16 muestra el error RMSE obtenido por los mejores cinco modelos en diferentes horizontes de predicción. Se puede ver que los modelos basados en árboles obtienen buena puntuación hasta cierto horizonte, a partir del cual el modelo kNN obtiene los mejores resultados.

7 – Conclusiones y trabajos futuros

En esta sección se exponen las conclusiones que se han obtenido durante la realización del trabajo, y se exponen una serie de mejoras al proyecto.

7.1 – Conclusiones

La predicción de potencia juega un papel fundamental en las plantas de producción fotovoltaica. Cada vez aumenta el deseo de tener predicciones más fiables y en un horizonte de predicción mayor.

En este documento se han probado algunas de las técnicas de predicción más utilizadas en la literatura relacionada con este ámbito en un horizonte de predicción a corto plazo (1 hora), siendo el modelo XGB, basado en árboles, aquel que ha conseguido obtener un menor RMSE.

Se ha podido observar que la mayoría de los modelos obtienen buenos resultados a la hora de predecir la potencia generada durante la próxima hora gracias a las variables que contienen información sobre las tres últimas horas previas a la predicción. Sin embargo, en el momento en el que el modelo se basa únicamente en información predicha, en este caso a partir de un horizonte de 4 horas, el modelo pierde la información real de la situación.

El principal problema de este trabajo ha sido la carencia de variables meteorológicas que permitieran aumentar el horizonte de predicción, ya que al aumentarlo sin conocer dichas variables se estaría suponiendo una estabilidad climática que no existe.

7.2 – Trabajos futuros

Dentro de los trabajos futuros, la principal mejora que se propone es tratar de aumentar el tiempo de horizonte a un periodo de tiempo cercano a las 24 horas. Esto se puede lograr aumentando la información del pasado que reciben los modelos, es decir, utilizando un número de lags superior al ya utilizado (3).

A su vez, la inclusión de variables meteorológicas puede servir de gran ayuda a la hora de ajustar los picos de producción que se producen entre las 11 y 13 de la mañana. Dentro de estas variables, se podría introducir la variable *tipo_de_dia*. Esta variable informaría al sistema si es un dia [*soleado, semi-nublado, nublado*] y se obtendría a través de una técnica de agrupación no supervisada como kNN.

Relativo a las técnicas de predicción, se recomienda ahondar en las redes neuronales recurrentes, como LSTM o GRU (Gated Recurrent Unit) probando arquitecturas más complejas que las estudiadas en este proyecto.

8 – Bibliografía

[1] – Renewable energy costs plummet according to IRENA - https://www.evwind.es/2020/06/05/renewable-energy-costs-plummet-according-to-irena/75021

[2] - La energía en el 2040, March 20, 2017, Ariel Yepez - David Lopez Soto https://blogs.iadb.org/energia/es/la-energia-en-el-2040/

[3] – Fritts, C. E. (1883). On a New Form of Selenium Photocell. American J. of Science, 26, 465

[4] – Geisz, J. F., France, R. M., Schulte, K. L., Steiner, M. A., Norman, A. G., Guthrey, H. L., … Moriarty, T. (2020). Six-junction III–V solar cells with 47.1% conversion efficiency under 143 Suns concentration. Nature Energy, 5(4), 326–335

[5] - Vikram, A. (2020). What are the most efficient solar panels on the market? Solar panel cell efficiency explained. Recuperado el 20 de mayo de 2020, de Energysage website: https://news.energysage.com/what-are-the-most-efficient-solar-panels-on-the-market/

[6] – Fernandez-Jimenez, L., Muñoz-Jimenez, A., Falces, A., Mendoza-Villena, M., GarciaGarrido, E., Lara-Santillan, P., Zorzano-Alba, E., Zorzano-Santamaria, P., 2012. Short-term power forecasting system for photovoltaic plants. Renew. Energy 44, 311–317.

[7] – Monteiro, C., Santos, T., Fernandez-Jimenez, L., Ramirez-Rosado, I., Terreros-Olarte, M., 2013b. Short-term power forecasting model for photovoltaic plants based on historical similarity. Energies 6, 2624–2643.

[8] – Lorenz, E., Heinemann, D., Kurz, D., 2011ª Local and regional photovoltaic power prediction for large scale grid integration: assessment of new Algorithm for snow dectection. Prog Photovolt.: Res. Appl. 20 (6), 760.769

[9] – Almeida, M.P., Perpiñán, O., Navarte, L., 2015. PV power forecast using a nonparametric PV model. Sol. Energy 115, 354-368

[10] – Mellit, A., Massi Pavan, A., Lughi, V., 2014. Short-term forecasting of power production in a large-scale photovoltaic plant. Sol. Energy 105, 401–413.

[11] – Shi, J., Lee, W.J., Liu, Y., Yang, Y., Wang, P., 2011. Forecasting power output of photovoltaic system based on weather classification and support vector machine. In: Industry Applications Society Annual Meeting (IAS). IEEE.

[12] – Chen, C., Duan, S., Cai, T., Liu, B., 2011. Online 24-h solar power forecasting based on weather type classification using artificial neural network. Sol. Energy 85, 2856-2870

[13] – Bouzerdoum, M., Mellit, A., Massi Pavan, A., 2013. A hybrid model (SARIMA-SVM) for short-term power forecasting of a small-scale grid-connected photovoltaic plan. Sol. Energy 98, 226-235

[14] – Pedro, H.T.C., Coimbra, C.F.M., 2012. Assessment of forecasting techniques for solar power production with no exogenous inputs. Sol. Energy 86, 2017–2028

[15] – Ogliari, E., Grimaccia, F., Leva, S., Mussetta, M., 2013. Hybrid predictive models for accurate forecasting in PV systems. Energies 6, 1918–1929.

[16] – Haque, A., Nehrir, M., Mandal, P., 2013. Solar PV power generation forecast using a hybrid intelligent approach. In: Power and Energy Society General Meeting (PES). IEEE, pp. 1–5.

[17] – Simonov, M., Mussetta, M., Grimaccia, F., Leva, S., Zich, R., 2012. Artificial intelligence forecast of PV plant production for integration in smart energy systems. Int. Rev. Electr. Eng. 7, 1.

[18] –Bacher, P., Madsen, H., Nielsen, H., 2009. Online short-term solar power forecasting. Sol. Energy 83, 1772–1783.

[20] – Pedro, H.T.C., Coimbra, C.F.M., 2012. Assessment of forecasting techniques for solar power production with no exogenous inputs. Sol. Energy 86, 2017–2028

[21] - Bouzerdoum, M., Mellit, A., Massi Pavan, A., 2013. A hybrid model (SARIMA-SVM) for short-term power forecasting of a small-scale grid-connected photovoltaic plan. Sol. Energy 98, 226-235

[22] – Pedro, H.T.C., Coimbra, C.F.M., 2012. Assessment of forecasting techniques for solar power production with no exogenous inputs. Sol. Energy 86, 2017–2028

[23] – Fernandez-Jimenez, L., Muñoz-Jimenez, A., Falces, A., Mendoza-Villena, M., GarciaGarrido, E., Lara-Santillan, P., Zorzano-Alba, E., Zorzano-Santamaria, P., 2012. Short-term power forecasting system for photovoltaic plants. Renew. Energy 44, 311–317.

[24] – . Alessandrini, S., Delle Monache, L., Sperati, S., Cervone, G., 2015. An analog ensemble for short-term probabilistic solar power forecast. Appl. Energy 157, 95–110.

[25] – Zamo, M., Mestre, O., Arbogast, P., Pannekoucke, O., 2014a. A benchmark of statistical regression methods for short-term forecasting of photovoltaic electricity production. Part I: Deterministic forecast of hourly production. Sol. Energy 105, 792–803.

[26] - Hong, T., Pinson, P., Fan, S., Zareipour, H., Troccoli, A., Hyndman, R.J., in press. Probabilistic energy forecasting: Global Energy Forecasting Competition 2014 and beyond. International Journal of Forecasting.

[27] - J. Antonanzas, N. Osorio, R. Escobar, R. Urraca, F.J. Martinez-de-Pison, F. Antonanzas-Torres, Review of photovoltaic power forecasting, Solar Energy, Volume 136, 2016, Pages 78-111,

[28] - ¿Cuánto cuesta la luz al mes? Consumo medio de luz en España - https://tarifasgasluz.com/faq