UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL CUSCO

FACULTAD DE CIENCIAS QUÍMICAS FÍSICAS Y MATEMÁTICAS

ESCUELA PROFESIONAL DE FÍSICA

TESIS

COMPOSICIÓN DE FUNCIONES DE COVARIANZA EN PROCESOS GAUSSIANOS PARA MEJORAR LA EFICIENCIA EN LA PREDICCIÓN DEL ESTADO DE CARGAS EN BATERÍAS DE ION DE LITIO

PRESENTADO POR:

Br. WILLY CRUZ BORDA

PARA OPTAR AL TÍTULO PROFESIONAL DE FÍSICO

ASESOR:

Dr. EDILBERTO ATAU ENRIQUEZ

CUSCO - PERÚ

2026

Universidad Nacional de San Antonio Abad del Cusco

INFORME DE SIMILITUD

(Aprobado por Resolución Nro.CU-321-2025-UNSAAC)

El que suscribe, el **Asesor** Dr. Edilberto Atau Enriquez ..... quien aplica el software de detección de similitud al trabajo de investigación/tesis titulada: Composición de funciones de Covarianza en procesos gaussianos para mejorar la eficiencia en la predicción del estado de cargas en baterías de ion de litio

Presentado por: Willy Cruz Borda ..... DNI N° 44783656;
presentado por: ..... DNI N°: .....
Para optar el título Profesional/Grado Académico de Físico .....

Informo que el trabajo de investigación ha sido sometido a revisión por 02 veces, mediante el Software de Similitud, conforme al Art. 6° del ***Reglamento para Uso del Sistema Detección de Similitud en la UNSAAC*** y de la evaluación de originalidad se tiene un porcentaje de 08 **%.**

Evaluación y acciones del reporte de coincidencia para trabajos de investigación conducentes a grado académico o título profesional, tesis

Porcentaje	Evaluación y Acciones	Marque con una (X)
Del 1 al 10%	No sobrepasa el porcentaje aceptado de similitud.	X
Del 11 al 30 %	Devolver al usuario para las subsanaciones.	
Mayor a 31%	El responsable de la revisión del documento emite un informe al inmediato jerárquico, conforme al reglamento, quien a su vez eleva el informe al Vicerrectorado de Investigación para que tome las acciones correspondientes; Sin perjuicio de las sanciones administrativas que correspondan de acuerdo a Ley.	

Por tanto, en mi condición de Asesor, firmo el presente informe en señal de conformidad y **adjunto** las primeras páginas del reporte del Sistema de Detección de Similitud.

Cusco, 21 de Julio de 20 26

.....
Firma

Post firma Edilberto Atau Enriquez
Nro. de DNI 23820774
ORCID del Asesor 0000-0002-2762-9525

Se adjunta:

1. Reporte generado por el Sistema Antiplagio.
2. Enlace del Reporte Generado por el Sistema de Detección de Similitud: **oid:** 27259-603758493

Willy Cruz Borda

Tesis Cruz-2_2_4.docx

Universidad Nacional San Antonio Abad del Cusco

Detalles del documento

Identificador de la entrega
trn:oid:::27259:603758493

Fecha de entrega
30 jun 2026, 9:36 a.m. GMT-5

Fecha de descarga
30 jul 2026, 2:42 p.m. GMT-5

Nombre del archivo
Tesis Cruz-2_2_4.docx

Tamaño del archivo
5.4 MB

116 páginas

27.426 palabras

151.582 caracteres

8% Similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para ca...

Exclusiones

- N.º de coincidencias excluidas

Fuentes principales

8% Fuentes de Internet
6% Publicaciones
7% Trabajos entregados (trabajos del estudiante)

Marcas de integridad

N.º de alertas de integridad para revisión

No se han detectado manipulaciones de texto sospechosas.

Los algoritmos de nuestro sistema analizan un documento en profundidad para buscar inconsistencias que permitirían distinguirlo de una entrega normal. Si advertimos algo extraño, lo marcamos como una alerta para que pueda revisarlo.

Una marca de alerta no es necesariamente un indicador de problemas. Sin embargo, recomendamos que preste atención y la revise.

DEDICATORIA

La presente investigación va dedicado especialmente a mi madre Antonia Borda de Cruz, quien es mi fuerza, motivo y razón de ser.

A mi padre Isaías Cruz Soto, quien me enseñó a afrontar los desafíos.

A mis hermanos Alexander, Luis y Fiorella, que son el soporte emocional que me impulsa a seguir adelante

AGREDECIMIENTOS

Agradezco a Dios y Jesús su único hijo, que gracias a la fe que mantengo me protegen y nada me hacen faltar, me guía por senderos favorables que hacen posible alcanzar mis sueños

Agradezco también a mi familia por todo el apoyo brindado durante todos estos años de mi etapa universitaria.

Agradezco a mi asesor, el Dr. Edilberto Atau Enriquez, a quien considero mi mentor. Quien me brindó su apoyo y los conocimientos, que permitieron mi desarrollo profesional.

Agradezco a mi compañera de estudios. Maritza Quispitupa Yupa, por brindarme sus consejos y apoyo incondicional para concluir el presente trabajo de investigación.

Agradezco al jefe de departamento académico Joffré Huamán Nuñez, de la institución en la que actualmente laboro, por darme la confianza y permitirme formar parte de un gran equipo de trabajo.

Agradezco a todos mis docentes de pregrado de mi querida Universidad Nacional de San Antonio Abad del Cusco, por darme una formación digna, que me permitió comprender de forma correcta cómo funciona el mundo.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La presente investigación soluciona la falta de eficiencia en la predicción del estado de carga en baterías de ion de litio mediante algoritmos de procesos gaussianos. Se probó la hipótesis de que estos algoritmos mejoran dicha eficiencia, alcanzando el objetivo de elaborar una composición de funciones de covarianza para incorporarlas en el modelo. La metodología es de tipo aplicada, nivel explicativo-correlacional, enfoque cuantitativo y diseño experimental. El procedimiento consistió en obtener datos de corriente, voltaje, capacidad de carga y temperatura de repositorios de la NASA y Oxford. Los datos se procesaron en Python para analizar la correlación entre ciclo, voltaje y temperatura. Posteriormente, el conjunto de datos se dividió en entrenamiento, prueba y validación; los dos primeros sirvieron para construir los modelos de regresión y el último para validar el modelo más eficiente. La selección se realizó mediante indicadores como el negativo de la probabilidad marginal y la raíz del error cuadrático medio (RMSE).Los resultados muestran valores mínimos de RMSE de 0.027 (NASA) y 0.00001 (Oxford). Se concluye que la composición de funciones de covarianza en procesos gaussianos mejoró la eficiencia en la predicción del estado de carga en un 14.8 % para los datos de la NASA y un 72.8 % para los de Oxford.

Palabras clave: Baterías de ion de litio, Procesos gaussianos, Modelos de predicción, Eficiencia.

ABSTRACT

This research addresses the lack of efficiency in predicting the state of charge in lithium-ion batteries using Gaussian process algorithms. The hypothesis that these algorithms improve prediction efficiency was tested, achieving the objective of developing a composition of covariance functions to incorporate them into the model. The methodology is applied, explanatory-correlational, quantitative, and experimental in design. The procedure consisted of obtaining current, voltage, charge capacity, and temperature data from NASA and Oxford repositories. The data were processed in Python to analyze the correlation between cycle, voltage, and temperature. Subsequently, the dataset was split into training, testing, and validation sets; the first two were used to build the regression models, and the latter to validate the most efficient model. Selection was performed using indicators such as the negative marginal log-likelihood and the root mean square error (RMSE).The results show minimum RMSE values of 0.027 (NASA) and 0.00001 (Oxford). It is concluded that the composition of covariance functions in Gaussian processes improved the efficiency of state of charge prediction by 14.8% for NASA data and 72.8% for Oxford data.

Keywords: Lithium-ion batteries, Gaussian processes, Prediction models, Efficiency.

INTRODUCCIÓN

Existen diversos dispositivos en los que la gestión del sistema de almacenamiento de energía requiere una alta precisión. En la era de la inteligencia artificial, del internet de las cosas y de los dispositivos móviles, el uso de sistemas de almacenamiento de energía de dimensiones pequeñas, como las baterías de ion de litio se ha incrementado exponencialmente (Madani et al., 2025), por lo que la capacidad de almacenamiento de carga y las formas de cómo incrementar el tiempo de vida de las mismas fue y sigue siendo tema de investigación. Hasta el momento las baterías más comerciales llevan en el electrodo positivo los siguientes compuestos: óxido de cobalto y litio ($LiCoO_2$), óxido de litio níquel manganeso y cobalto ($LiNiMnCoO_2$), óxido de litio hierro y fosfato ($LiFePO_4$) u óxido de litio níquel y cobalto ($LiNiCoAlO_2$) y el electrodo negativo está compuesto por una variedad de grafito, mientras que el electrolito usualmente está compuesto por hexafluorofosfato de litio ($LiPF_6$) (Manthiram, 2020). Todas estas posibles composiciones hacen que la capacidad de almacenamiento de carga y estado de carga dependa de factores externos, como el cambio repentino de temperatura o mantener a temperaturas extremas por un periodo prolongado de tiempo (Waldmann et al., 2014) y la humedad y los gases contaminantes también generan un impacto negativo en la capacidad de carga máxima de las baterías (Liu et al., 2012). Por estas razones se enfatiza la importancia del monitoreo constante del comportamiento de las variaciones de los parámetros característicos de las baterías de ion de litio, esto para tomar medidas de precaución y así evitar el desabastecimiento del suministro de energía ya sea en el dispositivo, equipo o sistema en cuestión. Con todo esto muchas investigaciones se dedicaron a proporcionar sistemas para el monitoreo de los parámetros de las baterías, tales parámetros pueden ser corriente, voltaje, temperatura, capacidad de carga y uno de los más importantes es el estado de carga, siendo útil para determinar el tiempo de vida remanente o el estado de salud de la batería (Bundhoo, 2018). Así las investigaciones se dedicaron a buscar modelos capaces de predecir el comportamiento del estado de carga de las baterías, para esto se propusieron inicialmente modelos de regresión lineal, auto regresivos o de media móvil (Khalid et al., 2019). Con el avance y difusión de la inteligencia artificial en todas las áreas tecnológicas, nuevas investigaciones surgieron, con enfoques modernos aplicando la inteligencia artificial como algoritmos de aprendizaje automático como redes neuronales artificiales,

máquina de soporte vectorial, bosques aleatorios, filtro de partículas y KNN (Farmann et al., 2015; Li et al., 2017; Meng et al., 2022; Wei et al., 2018; Zhao et al., 2018). Con valores de la raíz del error cuadrático medio (RMSE) en la predicción del estado de carga del orden de 0.1, este rendimiento es insuficiente para ser considerado en los sistemas de monitoreo y prevención, según lo señalado por Suh et al. (2024. Con el fin de mejorar la eficiencia en la predicción del estado de carga de las baterías de ion de litio, los investigadores desarrollaron una infinidad de modelos, los cuales fueron resultado de la combinación de diferentes algoritmos de aprendizaje automático, como el algoritmo híbrido compuesto por el algoritmo KNN y redes neuronales, cuyos resultados muestran valores del RMSE del orden de 0.01 (Ma et al., 2019), otro modelo que se puede mencionar, es el que usó algoritmos de descomposición modal empírica de conjunto completo con ruido adaptativo con el algoritmo de optimización de ballenas (del inglés, Whale Optimization Algorithm (WOA)) con regresión de soporte vectorial, cuyo valor de RMSE obtenido es del orden de 0.001 (Meng et al., 2022) y el modelo que se obtuvo en base al desarrollo de un algoritmo compuesto por una red neuronal convolucional unidimensional (CNN 1D) y una red neuronal bicapa de memoria a largo plazo (BLSTM) para predecir la vida útil remanente, cuyos valores de RMSE fueron del orden de 0.01 (Mou et al., 2024). Si bien los modelos antes mencionados alcanzan valores de RMSE bajos, lo que no se menciona, es que la complejidad de los algoritmos incrementa el costo computacional de los modelos, siendo esta característica el punto débil de los algoritmos complejos.

Por otro lado, la investigación desarrollada fue de tipo aplicada, con un nivel de investigación correlacional y explicativo, de enfoque cuantitativo y de diseño experimental. Para alcanzar los objetivos establecidos al inicio de la investigación, se acudió a los repositorios de la NASA y de Oxford para obtener los datos de número de ciclo, voltaje y temperatura correspondientes al proceso de envejecimiento de las baterías de ion de litio, a continuación, se llevó cabo la composición de funciones de covarianza con fundamento en las bases teóricas de procesos gaussianos desarrolladas por Rasmussen y Williams (2006). Esta composición dio lugar a 23 nuevas funciones de covarianza, las cuales fueron insertadas en el núcleo de los Procesos Gaussianos. Esta inserción se llevó a cabo a través del desarrollo del código en Python. Una vez realizada la inserción, se hizo uso de los datos de la NASA y de Oxford para el entrenamiento del algoritmo de Procesos Gaussianos. Al finalizar el entrenamiento se obtuvieron 23 modelos de regresión correspondiente a las 23 funciones de covarianza,

luego, a través del uso de los indicadores de eficiencia como son el error cuadrático medio (RMSE) y el logaritmo negativo de la probabilidad marginal (NLML), se llevó a cabo la elección del modelo de regresión más eficiente. Al realizar las predicciones con datos de la NASA se obtuvieron valores del RMSE de 0.29 y 0.27 para las baterías B006 y B018, respectivamente, mientras que para las predicciones con datos de Oxford se obtuvo valores del RMSE de 0.019, 0.00001 y 0.00011, para las baterías etiquetadas como, Celda 4, Celda 7 y Celda 8, respectivamente. Por otro lado, los valores de RMSE obtenidos en los trabajos de Richardson (2019) y Tagade (2020) para la batería B006, fueron de 0.07 y 0.282, respectivamente. Esta metodología permitió alcanzar mejoras del 4.1% y 25.3% en el indicador RMSE frente a los reportes de la literatura. Estos resultados únicamente fueron superados por modelos de gran complejidad técnica y alto costo computacional, según evidencian reportada por Yuan (2023) y Liu (2025). Con respecto a las investigaciones que emplean Procesos Gaussianos para estimar el estado de carga en baterías de ion de litio, la contribución crítica de este estudio consiste en la inserción de funciones de covarianza compuestas en el núcleo del proceso, resolviendo con éxito las limitaciones predictivas tradicionales.

CAPÍTULO I: PLANTEAMINETO DEL PROBLEMA

1.1. Situación Problemática

La evidencia de la existencia del problema fue reportada principalmente en el ámbito internacional, debido a que los fabricantes de equipos electrónicos que usan baterías de ion de litio son principalmente empresas extranjeras de desarrollo industrial. Estas invierten en investigaciones orientadas al desarrollo de modelos que permitan predecir las variaciones de los parámetros característicos de las baterías a través del desarrollo de algoritmos de inteligencia artificial, para incorporarlos en los softwares de monitoreo (Madani et al., 2025).

Publicaciones recientes (Mou et al., 2024; Suh et al., 2024; Wu et al., 2024) resaltan la persistente falta de precisión y eficiencia en la predicción de la vida útil remanente. Esta deficiencia impide un monitoreo efectivo y dificulta la prevención de fenómenos críticos como la combustión espontánea. La literatura reporta incidentes trágicos, incluyendo explosiones e incendios en dispositivos y vehículos eléctricos, siniestros que podrían mitigarse mediante sistemas predictivos robustos que garanticen una supervisión técnica confiable (Xing et al., 2023).

A nivel internacional, uno de los trabajos que exhibe de manera explícita la existencia del problema es el reportado por D. Yang (2018). Se menciona que los algoritmos actuales predicen el tiempo de vida útil y el estado de carga con una precisión insuficiente para ser confiables. Asimismo, Tagade (2020) señala que la causa principal es la falta de adaptabilidad y robustez de los algoritmos basados en máquinas de soporte vectorial y redes neuronales. Como los modelos de regresión no captan de forma precisa la curva de degradación, resulta inviable pronosticar averías súbitas que generan pérdidas considerables y altos costos de mantenimiento.

En el trabajo de Z. Li (2022), se subraya que las limitaciones de eficiencia ocurren porque el mecanismo de carga y descarga bajo diferentes condiciones ambientales genera comportamientos distintos en las curvas de degradación. Dado que la capacidad se degrada de forma gradual, resulta fundamental predecir con precisión su estado de carga; no obstante, los algoritmos desarrollados hasta la fecha presentan una elevada complejidad y un alto costo computacional, derivando en una gestión energética deficiente (Xing et al., 2023).

Dentro de este marco de búsqueda de eficiencia, se identifica que el uso de procesos gaussianos se ha vuelto una alternativa prometedora; sin embargo, se ha evidenciado que los procesos gaussianos sin la composición de covarianza no son tan eficientes para capturar las complejidades no lineales de la degradación, limitando su capacidad predictiva en entornos de monitoreo en tiempo real (Lucu et al., 2020).

1.2. Formulación del Problema

1.2.1. Problema General.

¿Es posible componer funciones de covarianza en procesos gaussianos para mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio?

1.2.2 Problemas Específicos

a. ¿En qué medida influye la composición de funciones de covarianza de procesos gaussianos en el error cuadrático medio?

b. ¿En qué medida influye la composición de funciones de covarianza de procesos gaussianos en el valor del logaritmo negativo de la probabilidad marginal?

1.3. Justificación de la Investigación

Es necesario predecir con exactitud la capacidad de almacenamiento de carga futura y la vida útil remanente de las baterías para garantizar el funcionamiento confiable del sistema y minimizar los costos de mantenimiento.

La naturaleza compleja de la degradación química en las baterías de ion de litio plantea desafíos significativos para el modelado de la pérdida de la capacidad de almacenamiento de carga y el estado de carga. No obstante, el auge de tecnologías como la computación en la nube, el Internet de las Cosas (IoT) y el aprendizaje automático facilita actualmente el procesamiento de grandes volúmenes de datos para desarrollar modelos capaces de capturar el comportamiento no lineal durante los ciclos de carga y descarga. Paralelamente, la proliferación de equipos móviles, laptops y drones, junto con la expansión de los vehículos eléctricos y las inversiones en almacenamiento para energías renovables, fundamentan la relevancia de la presente tesis, la cual está orientada a optimizar la gestión de sistemas de almacenamiento de energía mediante el uso de baterías de ion de litio.

Por otra parte, la presente tesis justifica el empleo de procesos gaussianos debido a su capacidad para modelar comportamientos no lineales mediante la selección flexible de

funciones de covarianza (kernels). Asimismo, destacan por su eficiencia computacional al ser un método de inferencia bayesiana que proporciona predicciones de carácter probabilístico, permitiendo así cuantificar la incertidumbre de las predicciones. Según Fu et al., 2025, los procesos gaussianos poseen el potencial de optimizar la precisión de sus predicciones gracias a su flexibilidad estructural. Esta característica permite incorporar no solo una función de covarianza individual, sino también kernels compuestos mediante la suma o el producto de múltiples funciones, adaptándose así a la complejidad de los datos.

1.4. Objetivos de la Investigación

1.4.1. Objetivo General.

Mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio utilizando composición de funciones de covarianza en procesos gaussianos.

1.4.2. Objetivos Específicos.

a. Aplicar la composición de funciones de covarianza en procesos gaussianos para minimizar el error cuadrático medio.

b. Aplicar la composición de funciones de covarianza en procesos gaussianos para minimizar el valor del logaritmo negativo de la probabilidad marginal.

CAPÍTULO II: MARCO TEÓRICO CONCEPTUAL

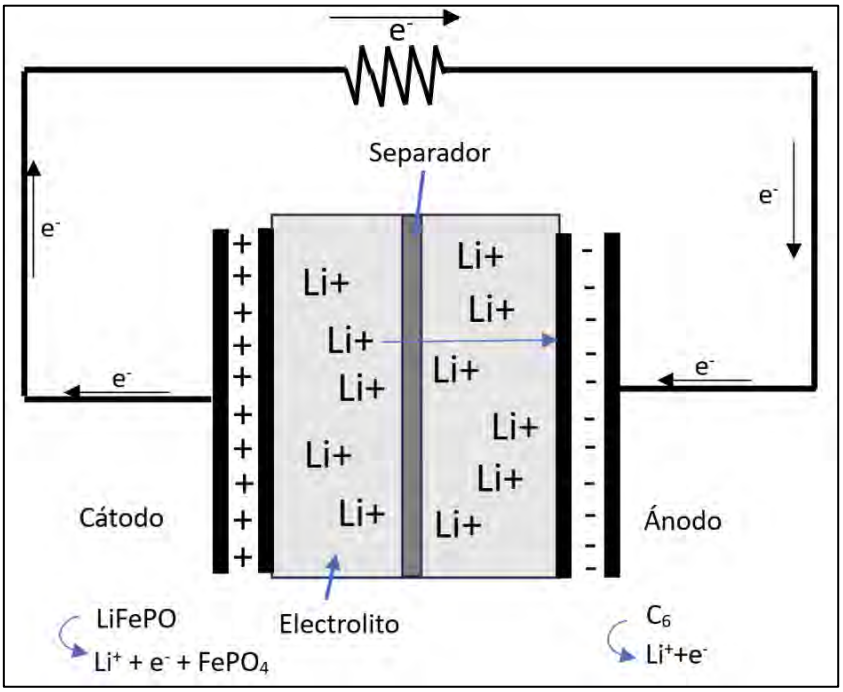
2.1. Bases Teóricas

2.1.1. Baterías

Una batería, es un dispositivo de almacenamiento de energía eléctrica, compuesto por un electrodo positivo (o cátodo), un electrodo negativo (o ánodo) y un electrolito mediador entre los electrodos (Pop, 2008). El electrolito mediador cumple la función de convertir energía química almacenada en corriente eléctrica, permitiendo la movilidad de los iones (o portadores de carga) desde ánodo hacia el cátodo para luego fluir a través de un conductor fuera de la batería, para alimentar un circuito eléctrico externo (Daigle & Kulkarni, 2013).

Por otro lado, las baterías de ion de litio (LIB) tienen características de peso ligero, alta densidad de potencia y mayor tiempo de vida útil en comparación con otras, por ello las LIB son consideradas como dispositivos de almacenamiento de energía óptimos para muchas aplicaciones (J. Zhang & Zhao, 2017; Zhao et al., 2018). Aplicaciones que van desde el transporte hasta el almacenamiento de energía de la red.

Figura 2.1. Diagrama de elementos y funcionamiento de una batería de ion de litio.



Nota. Elaboración propia.

En una batería de ion de litio el cátodo es una placa de aluminio con algún tipo de compuesto de litio encima. En nuestro caso, este compuesto es fosfato de hierro y litio o

LiFePo4. El ánodo, que es el lado negativo de la batería, está compuesto por una placa de cobre recubierto de grafito y en el medio hay un separador de plástico delgado que generalmente es de polietileno o polipropileno y toda esta estructura está llena de electrolito líquido, que tiene muchos iones de litio en su interior.

Cuando se carga la batería de litio, ocurre la transferencia de iones del cátodo hacia el ánodo. El ion de litio viaja a través del electrolito hacia el ánodo (que puede estar compuesto de grafito) en donde se produce la reacción con el electrón que se ha transferido a través del cargador. Si el ánodo es de grafito, el ion de litio más el electrón reaccionan con el grafito para formar un grafito compuesto por litio y cuando este ciclo se completa, la batería está completamente cargada, ya que la batería se descarga. La reacción inversa ocurre en el proceso de descarga.

En una batería de plomo-ácido, el electrolito es ácido sulfúrico. Una cosa a tener en cuenta es que, si tiene una carga alta en el agua, puede causar electrólisis del agua que puede reaccionar para formar hidrógeno y oxígeno. Por supuesto, esa es una mezcla explosiva y es por eso que es importante que las baterías de plomo ácido se ventilen, en el caso de los iones de litio, se tiene un electrolito orgánico.

La diferencia principal entre una batería de plomo ácido y una batería de ion de litio es, que en la batería de plomo ácido usa electrolito de ácido sulfúrico, que en realidad es agua con sulfato y iones de hidrógeno, lo cual constituye una mescla explosiva por lo que requiere ventilación constante, mientras que en las baterías de ion de litio no se requiere ventilación, por lo que las celdas pueden estar selladas. Otra diferencia fundamental es simplemente el peso. En una batería de plomo-ácido, el plomo es un elemento muy pesado, mientras que el litio es extremadamente ligero, esto hace que las baterías de plomo ácido tengan la capacidad de entregar mucha más energía que las baterías de ion de litio. La última diferencia es que, en las baterías de plomo ácido, el desgaste del electrolito da lugar a la generación de cristales de sulfato en los electrodos, haciendo que aumente significativamente la impedancia lo que dificulta el flujo de iones y electrones. Ese tipo de fenómeno denominado sulfatación no existe en una batería de iones de litio.

2.1.2. Proceso Gaussiano

Un Proceso Gaussiano, es una extensión general de la distribución de probabilidad Gaussiana, el cual depende de un conjunto de variables aleatorias y un conjunto de números. Los cuales conforman una Distribución Gaussiana Conjunta.

Los Procesos Gaussianos se definen completamente a través de la función media $m(x)$ y la función de covarianza (kernel de covarianza, o simplemente kernel o núcleo) $k(x, x')$, establecido sobre un evento real descrito por la función $f(x)$.

$$m(x) = E[f(x)] \tag{2.1}$$

$$k(x, x') = E\big[\big(f(x) - m(x)\big)\big(f(x') - m(x')\big)\big] \tag{2.2}$$

En consecuencia, el Proceso Gaussiano (PG) resulta en:

$$f(x) \sim PG\big(m(x), k(x, x')\big) \tag{2.3}$$

2.1.3. Funciones de covarianza o núcleos de covarianza

2.1.3.1. Núcleo Constante

Se define como una magnitud escalar cualquiera que tiene la propiedad de modificar la función media de un proceso Gaussiano.

$$k\big(x_i, x_j\big) = Valor\ constante. \tag{2.4}$$

El valor por defecto del kernel constante en las librerías de Python es 2.

2.1.3.2. Núcleo Producto Punto

Se define como un núcleo no estacionario (autores)

$$k\big(x_i, x_j\big) = \sigma_0^2 + \boldsymbol{x}_i \cdot \boldsymbol{x}_j \tag{2.5}$$

Aquí, $\boldsymbol{x}_i$ y $\boldsymbol{x}_j$ representan vectores del espacio de características. Mas concretamente el núcleo producto punto se puede escribir como.

$$k(X, X') = \sigma_0^2 + X^\top X' \tag{2.6}$$

X y X' son vectores en el espacio de características y $X^\top$ es la transpuesta del vector del espacio de características.

2.1.3.3. Núcleo Matern

Este tipo de kernel se representa de forma general

$$k(x_i, x_j) = \frac{1}{2^{\nu-1}\Gamma(\nu)}\left(\frac{\sqrt{2\nu}}{\ell} d(x_i, x_j)\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}}{\ell} d(x_i, x_j)\right) \tag{2.7}$$

$r = d(x_i, x_j)$, es la distancia euclidiana.

$\nu > 0$, es un parámetro de control de suavidad.

$\ell > 0$, es la longitud de escala,

$$k(r) = \frac{1}{2^{\nu-1}\Gamma(\nu)}\left(\frac{\sqrt{2\nu}}{\ell} r\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}}{\ell} r\right) \tag{2.8}$$

Es la representación general de las funciones de base radial (RBF), donde el parámetro ν cumple con la tarea de suavizar la función resultante. La misma que obedece a la parametrización de la longitud "ℓ", con ℓ siempre mayor que cero (Rasmussen & Williams, 2006).

Donde $d(x_i, x_j)$ es la distancia Euclideana K_{ν} es la función de Bessel modificada y $\Gamma(\nu)$ es la función gamma, esto se puede observar en (Rasmussen & Williams, 2006).

2.1.3.4. Núcleo Matern 3/2.

Cuando el parámetro ν toma el valor de $3/2$ la ecuación del kernel Matern se simplifica y se obtiene el kernel Matern 3/2.

$$k_{3/2}(r) = \frac{1}{2^{\frac{3}{2}-1}\Gamma\left(\frac{3}{2}\right)}\left(\frac{\sqrt{2\frac{3}{2}}}{\ell} r\right)^{\frac{3}{2}} K_{\frac{3}{2}}\left(\frac{\sqrt{2\frac{3}{2}}}{\ell} r\right) \tag{2.9}$$

$$k_{3/2}(r) = \frac{1}{2^{\frac{1}{2}}\Gamma\left(\frac{3}{2}\right)}\left(\frac{\sqrt{3}}{\ell} r\right)^{\frac{3}{2}} K_{\frac{3}{2}}\left(\frac{\sqrt{3}}{\ell} r\right) \tag{2.10}$$

Esto es.

$$\Gamma(z) = \int_0^{\infty} t^{z-1} e^{-t} dt \tag{2.11}$$

Para $\nu = \frac{3}{2}$, se tiene $\Gamma\left(\frac{3}{2}\right) = \Gamma\left(\frac{1}{2} + 1\right) = \frac{1}{2}\Gamma\left(\frac{1}{2}\right) = \frac{1}{2}(\sqrt{\pi})$

$$\Gamma\left(\frac{3}{2}\right) = \frac{\sqrt{\pi}}{2} \tag{2.12}$$

Donde la función de Bessel modificada, para $z = \frac{\sqrt{3}}{\ell} r$

$$K_{3/2}(z) = \sqrt{\frac{\pi}{2z}} e^{-z} \left(1 + \frac{1}{z}\right) \tag{2.13}$$

$$k_{3/2}(r) = \frac{2^{1-\frac{3}{2}}}{\Gamma\left(\frac{3}{2}\right)} \left(\frac{\sqrt{3}r}{\ell}\right)^{\frac{3}{2}} \sqrt{\frac{\pi}{2z}} e^{-z} \left(1 + \frac{1}{z}\right) \tag{2.14}$$

$$k_{3/2}(r) = \frac{2^{1-\frac{3}{2}}}{\Gamma\left(\frac{3}{2}\right)} \left(\frac{\sqrt{3}r}{\ell}\right)^{\frac{3}{2}} \sqrt{\frac{\pi}{2\frac{\sqrt{3}}{\ell} r}} e^{-\frac{\sqrt{3}}{\ell} r} \left(1 + \frac{\ell}{\sqrt{3}r}\right) \tag{2.15}$$

$$k_{3/2}(r) = \frac{2^{1-\frac{3}{2}}}{\frac{\sqrt{\pi}}{2}} \left(\frac{\sqrt{3}r}{\ell}\right)^{\frac{3}{2}} \sqrt{\frac{\pi}{2\frac{\sqrt{3}}{\ell} r}} e^{-\frac{\sqrt{3}}{\ell} r} \left(1 + \frac{\ell}{\sqrt{3}r}\right) \tag{2.16}$$

Simplificando.

$$k_{\nu=3/2}(r) = \left(1 + \frac{\sqrt{3}r}{\ell}\right) exp\left(-\frac{\sqrt{3}r}{\ell}\right) \tag{2.17}$$

2.1.3.5. Núcleo Matern 5/2.

Cuando el parámetro ν toma el valor de $5/2$ la ecuación de la kernel Matern se simplifica y se obtiene el kernel Matern 5/2.

$$k_{5/2}(r) = \frac{1}{2^{\frac{3}{2}}\Gamma\left(\frac{5}{2}\right)} \left(\frac{\sqrt{5}}{\ell} r\right)^{\frac{5}{2}} K_{\frac{5}{2}}\left(\frac{\sqrt{5}}{\ell} r\right) \tag{2.18}$$

Para $\nu = \frac{5}{2}$, se tiene $\Gamma\left(\frac{5}{2}\right) = \Gamma\left(\frac{3}{2} + 1\right) = \frac{3}{2}\Gamma\left(\frac{3}{2}\right) = \frac{3}{2}\left(\frac{\sqrt{\pi}}{2}\right)$

$$\Gamma\left(\frac{5}{2}\right) = \frac{3\sqrt{\pi}}{4} \tag{2.19}$$

Donde la función de Bessel modificada para $z = \frac{\sqrt{5}r}{\ell}$ es.

$$K_{5/2}(z) = \sqrt{\frac{\pi}{2z}} e^{-z} \left(1 + \frac{3}{z} + \frac{3}{z^2}\right) \tag{2.20}$$

$$k(r) = \frac{1}{3}z^2 e^{-z}\left(1 + \frac{1}{z} + \frac{1}{z^2}\right) \tag{2.21}$$

$$k_{5/2}(r) = \frac{1}{2^{\frac{3}{2}}\Gamma\left(\frac{5}{2}\right)}\left(\frac{\sqrt{5}}{\ell}r\right)^{\frac{5}{2}}\sqrt{\frac{\pi}{2z}}e^{-z}\left(1 + \frac{3}{z} + \frac{3}{z^2}\right) \tag{2.22}$$

$$k_{5/2}(r) = \frac{4}{2^{\frac{3}{2}} \cdot 3\sqrt{\pi}}\left(\frac{\sqrt{5}}{\ell}r\right)^{\frac{5}{2}}\sqrt{\frac{\pi}{2\frac{\sqrt{5}r}{\ell}}}e^{-\frac{\sqrt{5}r}{\ell}}\left(1 + \frac{3\ell}{\sqrt{5}r} + \frac{3\ell^2}{5r^2}\right) \tag{2.23}$$

Simplificando.

$$k_{\nu=5/2}(r) = \left(1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right) \tag{2.24}$$

2.1.3.6. Núcleo de Función de Base Radial.

También se le conoce como el núcleo o kernel o función de covarianza exponencial al cuadrado

$$k(x_i, x_j) = exp\left(-\frac{d(x_i, x_j)^2}{2\ell^2}\right) \tag{2.25}$$

Donde l es la escala.

2.1.4. Composición de funciones de covarianza.

La teoría que justifica la combinación lineal en suma y producto de funciones de covarianza (o kernels de covarianza) se justifica con la teoría establecida por (Rasmussen & Williams, 2006), junto con algunas aplicaciones mostradas en (Duvenaud, 2014), Esto es:

$$k = k_1 + k_2 + \cdots + k_n = k_1(x, x') + k_2(x, x') + \cdots + k_n(x, x') \tag{2.26}$$

$$k = k_1 \times k_2 \times \cdots \times k_n = k_1(x, x') \times k_2(x, x') \times \cdots \times k_n(x, x') \tag{2.27}$$

De acuerdo al sustento teórico se puede construir funciones de covarianza

Tabla 2.1. Funciones de covarianza.

Número	Funciones de Covarianza
1	$\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)$
2	$\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
3	$exp\left(-\frac{r^2}{2\ell^2}\right)$
4	$\sigma_0^2+\ x_i\cdot x_j+exp\left(-\frac{r^2}{2l^2}\right)$
5	$\sigma_0^2+\ x_i\cdot x_j+\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)$
6	$\sigma_0^2+\ x_i\cdot x_j+\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
7	$\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)+\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
8	$\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)+exp\left(-\frac{r^2}{2\ell^2}\right)$
9	$\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)+exp\left(-\frac{r^2}{2\ell^2}\right)$
10	$\sigma_0^2+\ x_i\cdot x_j+\sigma_1^2+\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)$
11	$\sigma_0^2+\ x_i\cdot x_j+\sigma_1^2+\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
12	$\sigma_0^2+\ x_i\cdot x_j+\sigma_1^2+exp\left(-\frac{r^2}{2\ell^2}\right)$
13	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)$
14	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
15	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot exp\left(-\frac{r^2}{2\ell^2}\right)$
16	$\sigma_1^2\cdot\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)$
17	$\sigma_1^2\cdot\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
18	$\sigma_1^2\cdot exp\left(-\frac{r^2}{2\ell^2}\right)$
19	$\sigma_1^2\cdot\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)+\sigma_1^2\cdot\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
20	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)+\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
21	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\sigma_1^2+\left(1+\frac{\sqrt{3}r}{\ell}\right)exp\left(-\frac{\sqrt{3}r}{\ell}\right)$
22	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\sigma_1^2+\left(1+\frac{\sqrt{5}r}{\ell}+\frac{5r^2}{3\ell^2}\right)exp\left(-\frac{\sqrt{5}r}{\ell}\right)$
23	$\left(\sigma_0^2+\ x_i\cdot x_j\right)\cdot\sigma_1^2+exp\left(-\frac{r^2}{2\ell^2}\right)$

Nota. Elaboración propia.

El conjunto de funciones de covarianza o kernels de la Tabla 2.1, son capaces de generar predicciones de valores posteriores de cualquier tipo de evento. Estas predicciones en general corresponden a los procesos gaussianos. En general se desea comparar la eficiencia o capacidad predictiva de los modelos obtenidos con procesos gaussianos y otros modelos obtenidos con algoritmos diferentes a los procesos

gaussianos. Por ello a continuación se presenta otros algoritmos de aprendizaje automático que también cumplen la función de predecir eventos posteriores.

2.1.5. Máquina de Soporte Vectorial.

La máquina de soporte vectorial (SVM, del inglés Support Vector Machine) es un conjunto de algoritmos de aprendizaje automático supervisado que se utilizan para obtener modelos de regresión y clasificación. Para obtener modelos de regresión se utilizan los algoritmos de regresión de soporte vectorial (SVR, Support Vector Regression)

2.1.6. Regresión de Soporte Vectorial (SVR)

Previamente se define el conjunto de datos $\{(X_i, y_i)\}_{i=1}^{n}$, con $X_i \in \mathbb{R}^d$, que representa al conjunto de datos de entrada y $y_i \in \mathbb{R}$, el conjunto de datos de salida.

Se define el objetivo de SVR, el cual es encontrar $f(X)$ (Cristianini & Shawe-Taylor, 2000),

$$f(X) = W^T X + b \tag{2.28}$$

que permite aproximar los valores de y_i haciendo que el error ε sea el máximo permitido, a la vez se minimiza el modelo a través de la formulación del problema de optimización primal, se genera como un problema de minimización (Cristianini & Shawe-Taylor, 2000).

$$\min_{W,b,\xi} \frac{1}{2}\|W\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{2.29}$$

con las restricciones:

$$\begin{cases} y_i - \left(W^\top X_i + b\right) \leq \varepsilon + \xi_i \\ (W^T X_i + b) - y_i \leq \varepsilon + \xi_i \\ \xi_i, \xi_i^* \geq 0, \;\; i = 1, \ldots, n \end{cases} \tag{2.30}$$

Aquí, ε, es el margen de tolerancia del desvío de y_i(o error), ξ_i, ξ_i^*, son variables de holgura que permiten errores mayores a ε, $C > 0$, controla el intercambio entre la complejidad del modelo (margen grande) y el error penalizado fuera del margen.

En general, se busca un modelo capaz de predecir dentro de un intervalo de amplitud de 2ε, alrededor de los datos reales ignorando errores pequeños (menores a ε).

Los errores mayores a ε se penalizan mediante ξ_i, ξ_i^*, y minimizar $\|W\|^2$ es equivalente a hacer la búsqueda de un modelo lo más lineal posible.

A continuación, se tiene una explicación detallada de la formulación de la regresión de soporte vectorial, el cual se obtiene a partir del problema primal (ver apéndice), el cual se obtiene usando multiplicadores de Lagrange, que permitirán usar kernels el cual brinda la ventaja de adaptar y obtener relaciones no lineales. A continuación, se describe la formulación dual (Cristianini & Shawe-Taylor, 2000).

$$\max_{\alpha_i \alpha_i^*} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(X_i, X_j) - \varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) \tag{2.31}$$

Sujeto a la restricción.

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \tag{2.32}$$

Donde, $0 \leq \alpha_i, \alpha_i^* \leq C, \quad \forall\, i = 1, \ldots\ldots, n$

Aclarando que:

$$K(X_i, X_j) = \phi(X_i)^\top \phi(X_j) \tag{2.33}$$

La ecuación (2.33) es la función kernel que permite operaciones en espacios de características en mayor dimensión, cuyo objetivo es determinar ϕ de forma explícita.

Las variables α_i, α_i^* determinan qué puntos de datos son vectores de soporte (aquellos con $\alpha_i \; o \; \alpha_i^*$ no nulos).

El parámetro C controla el intercambio entre el error y la complejidad del modelo.

Cuando se resuelve el problema dual, se obtiene la función de regresión (Cristianini & Shawe-Taylor, 2000).

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(X_i, X) + b \tag{2.34}$$

Donde b, se calcula usando las condiciones de KKT (Karush-Kuhn-Tucker)

La construcción del Lagrangiano, implica la introducción de los multiplicadores de Lagrange para las restricciones (Cristianini & Shawe-Taylor, 2000).

$$\alpha_i \geq 0, \text{ para } y_i - \left(W^\top X_i + b\right) - \varepsilon - \xi_i \leq 0 \tag{2.35}$$

$$\alpha_i^* \geq 0, \text{ para } \left(W^\top X_i + b\right) - y_i - \varepsilon - \xi_i^* \leq 0 \tag{2.36}$$

$$\mu_i \geq 0, \text{ para } \xi_i \geq 0 \tag{2.37}$$

$$\mu_i^* \geq 0, \text{ para } \xi_i^* \geq 0 \tag{2.38}$$

El lagrangiano es:

$$\begin{aligned} L = \frac{1}{2}\|W\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + \sum_{i=1}^{n}\alpha_i\left(y_i - W^\top X_i - b - \varepsilon - \xi_i\right) \\ + \sum_{i=1}^{n}\alpha_i^*\left(W^\top X_i - y_i - b - \varepsilon - \xi_i^*\right) - \sum_{i=1}^{n}\mu_i\xi_i \\ - \sum_{i=1}^{n}\mu_i^*\xi_i^* \end{aligned} \tag{2.39}$$

Condiciones de optimalidad (KKT) (Karush Kuhn Tucker ver en el Anexo 3)

La optimización, consiste en derivar L respecto a W, b, ξ_i, ξ_i^* e iguala a cero:

Derivada de L respecto a W (Cristianini & Shawe-Taylor, 2000).

$$\frac{\partial L}{\partial W} = W - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)X_i = 0 \implies W = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)X_i \tag{2.40}$$

Derivada de L respecto a b

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \implies \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \tag{2.41}$$

Derivada de L respecto a ξ_i (Cristianini & Shawe-Taylor, 2000).

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \mu_i = 0 \implies \alpha_i \leq C \tag{2.42}$$

$$\frac{\partial L}{\partial \xi_i^*} = C - \alpha_i^* - \mu_i^* = 0 \implies \alpha_i^* \leq C \tag{2.43}$$

Con la condición de, $\alpha_i, \alpha_i^* . \mu_i, \mu_i^* \geq 0$.

Ahora se tiene que sustituir en el Lagrangiano.

Se procede a sustituir W y se simplifica L, para obtener la función dual (Cristianini & Shawe-Taylor, 2000).

$$\max_{\alpha_i \alpha_i^*} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^*)\left(\alpha_j - \alpha_j^*\right)X_i^\top X_j - \varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) \tag{2.44}$$

Sujeto a restricciones:

$$\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) = 0,\ 0 \leq \alpha_i, \alpha_i^* \leq C \tag{2.45}$$

Para problemas no lineales, se reemplaza el producto interno por un kernel:

$$X_i^\top X_j \rightarrow K\left(X_i, X_j\right) \tag{2.46}$$

K es una función kernel positiva como RBF, polinomial o sigmoide (Cristianini & Shawe-Taylor, 2000).

Finalmente se obtiene (Cristianini & Shawe-Taylor, 2000)*:*

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(X_i, X) + b \tag{2.47}$$

Tabla 2.2. Resumen visual del proceso de una regresión de soporte vectorial.

	Paso	**Resultado clave**
1.	Primal	2. $\min_{W,b,\xi} \frac{1}{2}\|W\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$
3.	Lagrangiano	4. Introduce $\alpha_i,\ \alpha_i^*, \mu_i, \mu_i^*$
5.	KKT	6. $W = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)X_i$, $\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$
7.	Dual	8. Maximizar función cuadrática en $\alpha_i,\ \alpha_i^*$, con restricciones
9.	Kernel	10. Producto interno reemplazado por $K\left(X_i, X_j\right)$
11.	Función Estimada	12. $f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(X_i, X) + b$

Nota. Obtenido de Cristianini & Shawe-Taylor (2000)

2.1.7. Redes Neuronales Artificiales.

La red neuronal artificial es la simulación computacional del funcionamiento racional y lógico del cerebro humano. Toda la red neuronal artificial está compuesta por unidades denominadas, neuronas artificiales distribuidas por capas y que la información se transfiere entre las neuronas a través de ponderaciones con el uso de funciones no lineales.

Las redes neuronales artificiales (RNA), usualmente son usadas para generar modelos de aprendizaje supervisado y no supervisado. Por ejemplo, para generar modelos de regresión. A continuación, se muestra la formulación matemática básica.

2.1.7.1. Formulación Matemática de las RNA.

Sea una neurona j, en una capa determinada, las entradas son $x_1, x_2, \dots\dots, x_d$, los pesos asociados $w_{j1}, w_{j2}, \dots, w_{jd}$, y un sesgo b_j y la salida es a_j de la neurona, la cual se determina al aplicar la siguiente ecuación:

$$z_j = \sum_{i=i}^{d} w_{ji} x_i + b_j \tag{2.48}$$

donde, $a_j = \varphi(z_j)$, φ, es la función de activación (en redes neuronales se toma la función sigmoide), por consiguiente a_j es el resultado de aplicar la función sigmoide tomando como argumento z_j (Rokach & Maimon, 2015).

Un mejor funcionamiento se consigue cuando se adjunta un conjunto de redes neuronales en múltiples capas, a esto se le conoce como perceptrón multicapa.

2.1.7.2. Perceptrón multicapa

Un perceptrón multicapa (MLP, multilayer perceptron), es una red neuronal de alimentación retroactiva con múltiples capas ocultas entre la capa de entrada y la capa de salida.

Sea L, el número total de capas (que incluyen la entrada y la salida), $l = 1,2, \dots, L$ el índice que recorre las capas, n_l, el número de neuronas en la capa l, $x = a^{(0)} \in \mathbb{R}^{n_0}$ en el vector de entrada en la capa inicial (capa 0), $a^{(l)} \in \mathbb{R}^{n_l}$, el número de neuronas activadas de la capa l, $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$, es la matriz de pesos que conecta la capa $l-1$ con la capa l. $b^{(l)} \in \mathbb{R}^{n_l}$, es el vector de sesgos de la capa l. $\varphi^{(l)}(\cdot)$, es la función de activación aplicada a cada elemento de la capa l (Rokach & Maimon, 2015).

Las ecuaciones de propagación hacia adelante para cada $l = 1,2,\ldots,L$.

$$z^{(l)} = W^{(l)}a^{(l-1)} + b^{(l)} \tag{2.49}$$

$$a^{(l)} = \varphi^{(l)}z^{(l)} \tag{2.50}$$

Aquí, $z^{(l)}$ es el vector de entradas antes de la activación de las neuronas de la capa l y $a^{(l)}$ es el vector de salidas después de la activación de las neuronas.

La salida de la red resulta ser la predicción que tiene la siguiente forma:

$$\hat{y} = a^{(L)} = \varphi^{(L)}\left(W^{(L)}a^{(L-1)} + b^{(L)}\right) \tag{2.51}$$

Aquí $\varphi^{(L)}$ es la función de activación, que puede tomar la forma de la función sigmoide. $\varphi(z) = \frac{1}{1+e^{-z}}$, también puede tomar la función Unidad Lineal Rectificada (ReLU, Rectified Linear Unit) $\varphi(z) = \max(0, z)$, o puede tomar la forma de la función tangente hiperbólico, $\varphi(z) = \tanh(z)$ (Rokach & Maimon, 2015).

El proceso de entrenamiento de una red neuronal inicia sobre un conjunto de datos, denotados de la forma $\left\{\left(X^{(i)}, y^{(i)}\right)\right\}_{i=1}^{m}$, teniendo como objetivo minimizar una función de pérdida $\mathcal{L}\left(\hat{y}^{(i)}, y^{(i)}\right)$ ajustando los pesos $W^{(l)}$ y sesgos $b^{(l)}$ (Rokach & Maimon, 2015).

2.1.7.3. Aplicación de las ecuaciones de retropropagación.

El algoritmo de la retropropagación son útilies para determinar los gradientes de la función de pérdida, respectos a los pesos y los sesgos de una red neuronal, esto, usando la regla de la cadena. El método de optimización , como el gradiente descendiente, se utiliza para la actualización de los parámetros (Goodfellow et al., 2017).

Aquí L, el número total de capas, $a^{(l)}$, representa las activaciones de la capa l, $z^{(l)} = W^{(l)}a^{(l-1)} + b^{(l)}$, es la entrada ponderada antes de la capa l, $\varphi^{(l)}$, es la función de activación de la capa l, $\mathcal{L}(\hat{y}, y)$, representa a la función de pérdida (por ejemplo, puede ser el error cuadrático medio) y $\delta^{(l)}$, es el vector de errores de la capa l, que representa el gradiente local (Goodfellow et al., 2017).

2.1.7.4. Cálculo del error en la capa de salida.

La capa de salida L, tiene como error a $\delta^{(L)}$ y se determina como:

$$\delta^{(L)} = \nabla_{a^{(L)}}\mathcal{L} \odot \varphi^{(L)\prime}\left(z^{(L)}\right) \tag{2.52}$$

$\nabla_{a^{(L)}}\mathcal{L}$, es el gradiente de la pérdida respecto a la activación de la capa de salida.

$\odot$, es el producto elemento a elemento (Hardamard) (Goodfellow et al., 2017).

$\varphi^{(L)\prime}\left(z^{(L)}\right)$ es la derivada de la fusión de activación evaluada en $z^{(L)}$.

Aquí se muestra un ejemplo para el error cuadrático medio y activación lineal en la salida (Goodfellow et al., 2017).

$$\delta^{(L)} = \left(a^{(L)} - y\right) \odot \mathbf{1} = a^{(L)} - y \tag{2.53}$$

2.1.7.5. Propagación del error hacia capas anteriores.

El error se propaga hacia capas previas a la capa final, por lo que todo lo anterior a la última capa pertenece a la capa oculta esto es, $l = L-1, L-2, \ldots . 1$ y los errores se puede escribir como.

$$\delta^{(L)} = \left(W^{(l+1)^{\top}}\delta^{(l+1)}\right) \odot \varphi^{(L)\prime}\left(z^{(L)}\right) \tag{2.54}$$

2.1.7.6. Cálculo de gradientes para actualizar pesos y sesgos.

$$\frac{\partial \mathcal{L}}{\partial W^{(L)}} = \delta^{(l)}\left(a^{(l-1)}\right)^{\top} \tag{2.55}$$

$$\frac{\partial \mathcal{L}}{\partial b^{(L)}} = \delta^{(l)} \tag{2.56}$$

2.1.7.7. Actualización de pesos y sesgos.

Para una tasa de aprendizaje $\eta > 0$ se tiene la actualización.

$$W^{(l)} \leftarrow W^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(L)}} \tag{2.57}$$

$$b^{(l)} \leftarrow b^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial b^{(L)}} \tag{2.58}$$

Tabla 2. 3. Resumen del algoritmo de Redes Neuronales Artificiales.

Pase hacia adelante	**Calcular $z^{(l)}, a^{(l)}$, para todas las capas**
Calcular error en la salida	$\delta^{(l)}$.
Pase hacia atrás	Propagar el error hacia atrás calculando $\delta^{(l)}$ *para,* $l = L-1, L-2, \ldots . 1$.
Calcular gradientes	$\frac{\partial \mathcal{L}}{\partial W^{(L)}}, \frac{\partial \mathcal{L}}{\partial b^{(L)}}$.
Actualizar parámetros	Con gradiente descendiente o alguna variante.

Nota. Obtenido de Goodfellow (2017)

2.1.8. Bosques Aleatorios (Randon Forest)

Es un algoritmo de aprendizaje automático utilizado principalmente para regresión y clasificación. Este algoritmo se genera a través del ensamble de un determinado número de árboles de decisiones (Rokach & Maimon, 2015).

Un árbol de decisiones se define como una función $f: X \rightarrow y$ a través de una estructura cuya morfología es la de un árbol.

$X \subseteq \mathbb{R}^d$:es el espacio de características (variables de entrada)

y: representa a la variable objetivo. Esta variable puede ser del tipo categórico para tareas de clasificación o del tipo real para regresión (Rokach & Maimon, 2015).

Un árbol de decisión se construye particionando de forma recursiva el espacio de características X en conjuntos disjuntos $\{R_1, R_2, \ldots, R_J\}$, de modo que:

$$X = \bigcup_{j=1}^{J} R_j, R_j \cap R_k = \emptyset \text{ para } j \neq k \tag{2.59}$$

donde cada región R_j corresponde a una hoja del árbol.

Las predicciones para un elemento $x \in X$ se denota como: $f(x) = C_j$ si $x \in R_j$, donde C_j es la predicción asignada a la región R_j. Donde para regresiones C_j se determina de la siguiente manera:

$C_j = \frac{1}{|R_j|} \sum_{x_i \in R_j} y_i$. C_j, es el promedio de ponderado de los valores objetivo y_j. Para conseguir esto se debe llevar a cabo la división de nodos (esto es el algoritmo de construcción) (Rokach & Maimon, 2015).

En cada paso, para un nodo con datos $\{(x_i, y_i)\}_{i=1}^{n}$, se busca la mejor división para un atributo j y un umbral θ:

$\left(x^{(j)} \leq \theta\right) \Longrightarrow$ se particiona en dos nodos hijos: $\begin{cases} R_{izquierda} = \{x | x^{(j)} \leq \theta\} \\ R_{derecha} = \{x | x^{(j)} > \theta\} \end{cases}$, la mejor división minimiza la impureza o el error de la partición (Rokach & Maimon, 2015).

Ahora se define la función objetivo para la selección de la mejor división.

En regresión, se suele minimizar la varianza o el error cuadrático medio.

$Var(R) = \frac{1}{|R|}\sum_{x_i \in R}(y_i - \bar{y}_R)^2$, donde $\bar{y}_R$ es el promedio de los y_i en R.

El criterio para obtener la mejor división se lleva a cabo al elegir el atributo j, y umbral θ, que sean capaces de minimizar la suma ponderada de la impureza o ruido de las divisiones (Rokach & Maimon, 2015).

$$\min_{j,\theta} \frac{|R_{izquierda}|}{|R|} Q\left(R_{izquierda}\right) + \frac{|R_{derecha}|}{|R|} Q(R_{derecha}) \tag{2.60}$$

Aquí, $Q(\cdot)$, es la función de ruido o impureza o varianza según el problema. La demostración con datos numéricos se puede observar en el Anexo 3.

2.1.9. Optimizador de Hiperparámetros

2.1.9.1. Hiperparámetro

Un hiper parámetro se define como un parámetro con valores predefinidos previo a la etapa de entrenamiento de cualquier modelo de aprendizaje automático. Los valores de los hiper parámetros, no se obtiene a partir de los datos, sino que se define a través de un ajuste manual o a través del uso de optimizadores automáticos dentro de los cuales los más comunes son la técnica de Grid Search, Random Search, Grey Wolf, Optuna, etc.

La diferencia entre un parámetro y un hiper parámetro es, que los valores de un parámetro se obtienen durante el proceso de entrenamiento (un ejemplo de parámetro son los coeficientes de una red neuronal o los pesos de una red neuronal), mientras que los hiper parámetros se consideran como configuraciones externas del modelo y tienen la capacidad de influir en la forma como se entrena y estructura el modelo (algunos ejemplos de hiper parámetro serían, el ratio de un proceso de aprendizaje, el número de capas de una red neuronal, el valor de "ν" en la función Matern 3/2 o la profundidad máxima de un árbol de decisión).

2.1.9.2. Optimizador Optuna

Se define como un optimizador basado en técnicas bayesianas, que es utilizado para optimizar hiperparámetros de forma automática en algoritmos de aprendizaje automático. Que se encuentra disponible como código abierto y se puede importar en plataformas de compilación de Python, su algoritmo basado en la formulación matemática se simplifica como sigue a continuación:

$$X^* = \arg\min_{x \in X} f(X) \tag{2.61}$$

Siendo X, el vector de hiperparámetros que incluye variables de tipo continuo, categórico y discreto. Con f definida por una función objetivo cuya regla de correspondencia es, $f: X \rightarrow R$ (Akiba et al., 2019).

El proceso para obtener el conjunto de posibles resultados cuyos valores de los hiperparámetros son prometedores, los cuales se obtiene en tres pasos:

Primero:

Se tiene que modelar $p(X|y)$ y $p(y)$ estableciendo un límite superior y^*. De acuerdo al resultado de los ensayos se puede clasificar en buenas observaciones cuando $y < y^*$ y malas observaciones cuando $y \geq y^*$. El modelamiento de las densidades de probabilidad se da de la siguiente manera:

$$l(X) = p(X|y < y^*) \tag{2.62}$$

$$g(X) = p(X|y \geq y^*) \tag{2.63}$$

Donde, $l(X)$ es la probabilidad de obtener el hiperparámetro X dado que $y < y^*$ y $g(X)$ es la probabilidad de obtener el hiperparámetro X dado que $y \geq y^*$ (Akiba et al., 2019).

Segundo:

La selección de los hiperparámetros se lleva a cabo haciendo uso de la siguiente ecuación:

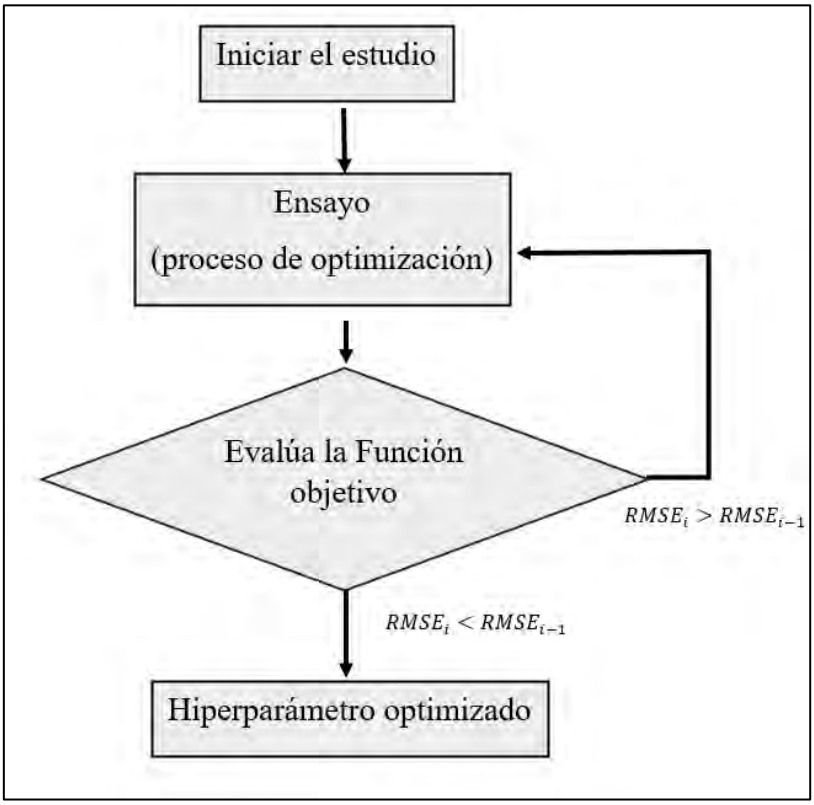
$$X_{t+1} = arg\max_{X} \frac{l(X)}{g(X)} \tag{2.64}$$

Esta ecuación, indica que se elige valores óptimos de X, que se obtienen a partir de la maximización del argumento $l(X)/g(X)$ (Akiba et al., 2019).

Tercero:

Se lleva a cabo la evaluación del valor X_{t+1}, en la fusión predefinida. Esto significa obtener $f(X_{t+1})$, paso siguiente, se actualizan las densidades $l(X)$ y $g(X)$ con las nuevas observaciones. Este proceso (quiere decir los pasos primero, segundo, y tercero), se repite iterativamente (Akiba et al., 2019). Adicionalmente, el optimizador Optuna puede interrumpir ensayos que no son prometedores teniendo en cuenta resultados intermedios.

Figura 2.2.Diagrama de flujo del proceso de optimización con Optuna.



Nota. Elaboración propia.

Los detalles del algoritmo desarrollado del optimizador Optuna se encuntran disponibles en la plataforma GitHub y en el artículo oficial publicado (Akiba et al., 2019). El algoritmo fue desarrollado en código abierto del lenguaje Python y se puede instalar e importar como una librería de Python con la sentencia import optuna.

2.1.9.3. Optimizador Grid Search

Grid Search, se traduce al español como búsqueda de cuadrícula, es un método en la que se busca de forma exhaustiva la mejor combinación de los valores de hiperparámetros. La mejor combinación será la que maximice la eficiencia en la predicción del modelo de aprendizaje automático (Hastie et al., n.d.)..

La Formulación matemática, consiste en lo siguiente.

Dado un modelo de aprendizaje automático, con un conjunto de hiperparámetros, representado por $\theta = (\theta_1, \theta_2, \dots, \theta_d)$, donde cada hiperparámetro θ_i, puede tomar cualquier valor de un conjunto de posibles valores (Hastie et al., n.d.)..

$$\theta_i \in \Theta_i = \{\theta_{i,1}, \theta_{1,2}, \dots., \theta_{i,n_i}\} \tag{2.65}$$

cuya búsqueda se realiza en un espacio del orden del producto cartesiano:

$$\Theta = \Theta_1 \times \Theta_2 \times \dots.\times \Theta_d \tag{2.66}$$

Se formula el problema de optimización al minimizar la función objetivo $J(\theta)$:

$$\theta^* = arg \max_{\theta \in \Theta} J(\theta) \tag{2.67}$$

Por consiguiente, el optimizador Grid Search lleva a cabo para cada $\theta \in \Theta$, el cálculo de $J(\theta)$, para luego obtener el nuevo valor del hiperparámetro $\theta^* = arg \max_{\theta \in \Theta} J(\theta)$(Hastie et al., n.d.).

2.1.9.4. Optimizador Randon Search

Al igual que Grid Search, Randon Search es un algoritmo que busca optimizar hiperparámetros, para el mejor desempeño de modelos de regresión.

La formulación matemática de Randon Search inicia con la suposición de la existencia de un conjunto de hiperparámetros, $\theta = (\theta_1, \theta_2, \dots, \theta_d)$, donde cada elemento del conjunto pertenece a un conjunto discreto o continuo de posibles valores, $\theta_i \in \Theta_i$. Random Search realiza la selección de un subconjunto aleatorio de posibles valores de todo el espacio de posibles valores, este conjunto se representa como. $\{\theta^{(1)}, \theta^{(2)}, \dots., \theta^{(N)}\}$, lo cual se obtiene a través de una regla de muestreo de acuerdo a una distribución $P(\theta)$(distribución puede ser uniforme discreto, uniforme continuo o distribución uniforme logarítmica), definida sobre Θ.

El procedimiento inicia al definir N, tal que, para $j = 1, \dots, N$, se elige una muestra.

$$\theta^{(j)} \sim P(\theta) \tag{2.68}$$

Para luego evaluar la función objetivo J en cada muestra $J\left(\theta^{(j)}\right)$, para luego seleccionar la mejor muestra de acuerdo a la siguiente ecuación (Hastie et al., n.d.).

$$\theta^{*} = arg \max_{j=1,..N} J\left(\theta^{(j)}\right) \tag{2.69}$$

La forma de la función objetivo J, depende del modelo de aprendizaje automático que se usa.

La codificación en Python, tanto de Grid Search como de Random Search están disponibles en las librerías de Python GridSearch y Randomized Search, respectivamente.

2.1.9.5. Capacidad de carga

La capacidad de carga de la batería es la medida de la cantidad de energía almacenada en una batería, cuya unidad está dada en amperio-hora (Ah), definida como la cantidad de horas durante las cuales una batería puede proporcionar una corriente constante, con velocidad de descarga constante y con el valor de voltaje nominal. La unidad Ah se usa generalmente cuando se trabaja con sistemas de baterías, ya que el voltaje de la batería varía a lo largo del ciclo de carga o descarga. Las unidades de la capacidad expresada en Wh, se obtiene al multiplicar la capacidad (expresado en Ah) por el voltaje nominal de la batería. Por otro lado, la capacidad nominal de carga de un componente de almacenamiento (en general) es la cantidad de energía que se puede extraer de él a una corriente constante particular, comenzando desde un estado completamente cargado (Richardson et al., 2017).

El monitoreo de la capacidad de carga de las baterías de un sistema de almacenamiento de energía es de vital importancia. En consecuencia, el parámetro que cuantifica el estado del sistema de almacenamiento en un instante de tiempo dado es el estado de carga (state of charge, SOC)(Richardson et al., 2017).

La fiabilidad de las baterías recargables de iones de litio ha sido reconocida como de gran importancia por una amplia gama de partes interesadas, incluidos fabricantes de baterías, fabricantes de dispositivos alimentados por baterías, agencias reguladoras, investigadores y el público en general. Evaluar los índices actual y futura de las baterías

de iones de litio es esencial para garantizar que las baterías funcionen de manera segura y confiable durante toda su vida útil. El desconocimiento de los índices de las LIB, hace que el sistema opere en un régimen inseguro (Cabrera-Castillo y cols., 2016), lo cual no solo afecta al funcionamiento normal del sistema, sino también existe el riesgo de experimentar eventos catastróficos (Hubbard y cols., 2016), tales como la ignición del dispositivo por el incremento de la temperatura. Para evitar ello, es importante realizar una evaluación exhaustiva y eficiente de los parámetros de control de las LIB. El desconocimiento de los índices, también hace que las baterías del sistema sean a menudo sobredimensionadas y subutilizadas, lo que genera ineficiencias en la valoración económica del sistema. En consecuencia, el pronóstico preciso de los índices, es un procedimiento fundamental de un sistema moderno de administración de baterías.

2.1.9.6. Estado de carga en baterías

El estado de carga de una batería se define en una escala de 0 a 1, donde 0 indica descarga completa y 1 cuando la batería está completamente cargada. Distinguimos aquí entre SOC nominal y SOC aparente (Daigle y Kulkarni, 2013). El SOC nominal se calcula en función de la combinación de los volúmenes de control de la capa superficial y el volumen en el electrodo negativo, mientras que el SOC aparente se calcula solo en función al volumen de la capa superficial. Cuando una batería alcanza el límite de voltaje, el SOC aparente es 0 y el SOC nominal es mayor que 0 (cuanto mayor depende de la diferencia entre la velocidad de difusión y la corriente consumida). Una vez que el gradiente de concentración se estabiliza, la capa superficial se repondrá parcialmente y el SOC aparente aumentará mientras que el SOC nominal permanece igual (Bole y cols., 2014; Hogge y cols., 2015a).

El estado de carga denotado como SOC, se defina matemáticamente como.

$$SOC = \frac{Carga\ eléctrica\ disponible\ enla\ batería}{Capacidad\ total\ de\ la\ batería} \times 100\% \tag{2.70}$$

Cuando se obtiene a partir de la corriente se aplica la siguiente ecuación.

$$SOC = SOC(t_0) + \frac{1}{C_{nom}} \int_{t_0}^{t} I(t)\, dt \tag{2.71}$$

Donde $I(t)$, es la corriente instantánea cuyo valor es positivo en procesos de carga y negativo cuando la batería se descarga. C_{nom}, es la capacidad nominal de carga de la batería en unidades de Ampere hora (Ah). $SOC(t_0)$, es el estado de carga inicial.

Adicionalmente al monitoreo del SOC existen dos parámetros útiles para caracterizar por completo el estado de las baterías, que son el estado de salud (state of health (SOH)) y la vida útil remanente (remaining useful life (RUL)). El SOH representa la relación del estado actual al estado inicial y se define como la razón de la capacidad nominal de carga del i-ésimo ciclo y la capacidad nominal de carga inicial, mientras que la RUL expresa el número de ciclos de carga y descarga útiles restantes y se estima mediante la diferencia entre el ciclo actual y el ciclo final, siendo el ciclo final, el ciclo en el que la batería alcanza su umbral de vida útil, el cual varía entre el 70% a 80% de la capacidad nominal de carga inicial (Ma y cols., 2019):

$$RUL = SOC(actual) - SOC(final) \tag{2.72}$$

Donde el $SOC(actual)$, se refiere al estado de carga máximo que alcanza la batería en el ciclo de descarga, el cual suele disminuir, debido al deterioro de los componentes del dispositivo de almacenamiento. Mientras que el $SOC(final)$, se determina a partir del 70 % de la capacidad de carga máxima del primer ciclo de descarga.

2.1.9.7. Logaritmo negativo de la probabilidad marginal

Se define como una función objetivo que representa la pérdida considerando la incertidumbre de los parámetros. Esto quiere decir que mide cuantitativamente lo deficiente que puede llegar a ser un modelo para explicar los datos. Esto es aplicado para modelos estadísticos de origen bayesiano y en la generación de modelos de aprendizaje automático, específicamente en algoritmos de procesos gaussianos. Matemáticamente se define como sigue:

Para un proceso gaussiano, el logaritmo de la probabilidad marginal se define sobre un conjunto de datos de entrada "**X**" y un conjunto datos de salida "**y**" y con parámetros de control $\boldsymbol{\theta}$ (Murphy P, 2012):

$$\log P(y|X,\theta) = \frac{1}{2} y^T K_\theta^{-1} y + \frac{1}{2} log|K_\theta| + \frac{n}{2} log 2\pi \tag{2.73}$$

Donde, K_θ, es la matriz de covarianza con parámetros $\boldsymbol{\theta}$, y n, el número total de datos.

El valor del parámetro $\boldsymbol{\theta}$ se ajusta a través de un proceso de optimización cuya función principal es la de minimizar para encontrar el ajuste de $\boldsymbol{\theta}$ a los datos. En consecuencia,

obtener un valor menor posible de $\log P(y|X,\theta)$, es favorable para un modelo lo cual indicaría también que el modelo se aleja del sobreajuste (Murphy P, 2012).

2.2. Marco Conceptual

2.2.1. Batería de $LiCoO_2$.

Es del tipo de baterías recargables. Cuyo cátodo está compuesto de óxido de cobalto y litio ($LiCoO_2$). Es muy eficiente a la hora de transmitir iones de litio a través del electrolito de sal de litio (solvente orgánico), lo que le permite suministrar un voltaje constante y mantener un tiempo de vida útil larga en relación a otros tipos de baterías. Esta característica le hace muy útil para su uso en baterías con características de almacenamiento compacto y potente requeridas para sistemas de almacenamiento de energía en vehículos eléctricos y productos electrónicos de consumo como celulares (Waldmann et al., 2014).

2.2.2. Aprendizaje automático

El aprendizaje automático es una rama de la inteligencia artificial que se encarga del desarrollo de algoritmos para dotar a las máquinas la capacidad de aprender a partir de datos a través de un proceso de entrenamiento y posteriormente realizar tareas específicas sin la necesidad de programar(Seeger, 2004).

En general, en el proceso de entrenamiento se proporciona al algoritmo un conjunto de datos, de los cuales el algoritmo extrae patrones para generar modelos de regresión con los cuales la máquina realiza predicciones o toma de decisiones basadas en nuevos datos(Cristianini & Shawe-Taylor, 2000).

El aprendizaje automático se subdivide en aprendizaje supervisado, aprendizaje no supervisado y aprendizaje reforzado. El aprendizaje supervisado constituye un conjunto de algoritmos que generan modelos de regresión y de clasificación. El aprendizaje no supervisado abarca a los algoritmos que generan modelos capaces de identificar patrones y agruparlos según sus características, por último, el aprendizaje reforzado está constituido por algoritmos de toma de decisiones, donde una decisión está sujeto a una recompensa o castigo, con esto el algoritmo mejora su comportamiento(Shalev-Shwartz & Ben-David, 2014).

2.2.3. Indicadores

Los indicadores que se utilizan principalmente, son usados para estimar la precisión y el grado de sobreajuste del modelo obtenido. El sobreajuste se determina con el negativo del logaritmo de la probabilidad marginal (NLML, Negative log Marginal Likelihood), y la precisión se determina con la raíz cuadrada del error cuadrático medio (RMSE, Rational Mean Squared Error). A continuación, se describe la formulación matemática de los indicadores antes mencionados(Rasmussen & Williams, 2006).

2.2.3.1. Raíz del Error Cuadrático Medio.

Se define como una métrica estadística que determina de forma cuantitativa la precisión de un modelo para predecir eventos posteriores y que matemáticamente se formula como la raíz cuadrada media del cuadrado de la diferencia del valor determinado por el modelo y el valor real (u observado) (Murphy P, 2012), donde, el objetivo de cada modelo es que el RMSE se aproxime a cero.

2.2.3.2. Eficiencia

Dado un método de predicción, se evaluará la calidad de las predicciones usando la raíz del error cuadrático medio, el cual se define a partir de los valores que se obtiene a partir de los valores estimados de RMSE, donde y_i, son los valores estimados por el modelo y $\hat{y}_i$, son los valores reales. El término de eficiencia está asociado con el valor del RMSE, que se vaya a obtener. Por ello el valor de RMSE que se obtenga, será la que caracterice la eficiencia del modelo (Richardson et al., 2018).

2.3. Antecedentes de la Investigación

El trabajo de investigación titulado ‘Predicción del estado de la batería en condiciones generalizadas mediante un modelo de transición de proceso gaussiano’, presentado por Richardson (2019), tuvo como objetivo principal, predecir los cambios en la capacidad de almacenamiento de carga y el estado de caga de la batería a lo largo del tiempo utilizando un enfoque no paramétrico bayesiano basado en la regresión del proceso gaussiano, El desarrollo del modelo consistió en formular un enfoque de transición bayesiano no paramétrico que, en lugar de ajustar la capacidad directamente en función del tiempo, predice las diferencias de capacidad entre periodos variables de uso denominados patrones de carga. La innovación clave de este método radica en su etapa de procesamiento y selección de características, usando registros temporales de corriente,

voltaje y temperatura, los cuales se mapean mediante histogramas en vectores de entrada de tamaño fijo, utilizando variables esenciales como el flujo de carga y el tiempo. Para obtener el modelo más eficiente, los autores desarrollaron y evaluaron seis configuraciones de procesos gaussianos en la que se incluyó los siguientes: el proceso gaussiano con el núcleo Matérn 5/2 (adecuado para capturar comportamientos no lineales), procesos gaussianos con núcleos lineales equivalentes a una regresión lineal bayesiana convencional y modelos basados en la técnica de aprendizaje automático Gradient Boosting entrenados mediante regresión cuantílica para aproximar intervalos de confianza. Los resultados demostraron que el modelo de proceso gaussiano con núcleo Matérn 5/2 obtuvo el mejor desempeño absoluto, alcanzando un error cuadrático medio normalizado de 4.3% en la predicción de la capacidad. Por el contrario, los modelos con núcleos lineales arrojaron errores severos debido a su incapacidad para modelar las interacciones no lineales de la degradación, mientras que los modelos de Gradient Boosting resultaron ser deficientes y erróneamente sobre ajustados. Analizando los resultados se observó que el tiempo transcurrido y el flujo de carga son los factores dominantes para el envejecimiento de la batería, demostrando que considerar el comportamiento histórico del estado de carga de la batería optimiza drásticamente la precisión del diagnóstico predictivo a largo plazo. se concluye que el modelo de mejor rendimiento utiliza la regresión del proceso gaussiano con función de núcleo Matérn 5/2, el cual predice con precisión el desvanecimiento de la capacidad de almacenamiento de carga no lineal, con un error cuadrático medio normalizado en el mejor de los casos del 4.3%. Este fue uno de los primeros documentos que cuantifica realmente la precisión predictiva del estado de la batería de manera integral.

El trabajo de investigación de título ‘Regresión del proceso gaussiano profundo para el pronóstico del estado de carga de la batería de iones de litio y el diagnóstico del modo de degradación’, realizado por Tagade (2020), la investigación fue desarrollada en el Instituto de Investigación y Desarrollo de Samsung de India- Bangalore. El objetivo principal fue generar un modelo de predicción de la capacidad de almacenamiento de carga de la batería usando datos de la capacidad, voltaje, temperatura y número de ciclo en el proceso de descarga en baterías de ion de litio obtenidos del repositorio de la NASA, El desarrollo del modelo consistió en diseñar un algoritmo de regresión usando procesos gaussianos profundos multivariados, el cual apila estructuralmente múltiples procesos gaussianos dispersos y aplica una distribución gaussiana de matriz variada para modelar

de forma directa la correlación estadística entre los nodos de una misma capa. La ventaja operativa de esta arquitectura profunda radica en que procesa directamente series temporales parciales de datos (voltaje, corriente, temperatura y tiempo) mediante la extracción automatizada de características, para lo cual multiplica una matriz de datos secuenciales de diez puntos continuos, extraídos aleatoriamente por un vector de proyección optimizado, eliminando así la necesidad de un procesamiento previo manual por parte de expertos. Para validar el sistema, los autores emplearon los conjuntos de datos de ciclo de vida histórico obtenidos del repositorio de la NASA. Los resultados demostraron una elevada precisión predictiva, registrando un coeficiente de correlación superior a 0.90 y un error absoluto medio (MAE) menor a 0.1 (e inferior a 0.03 en condiciones de temperatura extrema). Finalmente, la investigación concluye que las salidas de las capas ocultas del modelo profundo poseen una correlación estadística directa y robusta con parámetros físicos internos como la resistencia a la transferencia de carga y la resistencia de la fase electrolítica, demostrando la viabilidad de implementar diagnósticos integrales a bordo del sistema de gestión de baterías, sin depender de técnicas experimentales invasivas o puramente fuera de línea como la espectroscopía de impedancia electroquímica. Se concluye indicando que la eficiencia obtenida que corresponde a la raíz del error cuadrático medio mínimo obtenido es de 0.3 teniendo posibilidad de mejoras futuras.

El estudio titulado ‘Predicción de la vida útil remanente de una batería de iones de litio mediante la regresión de soporte vectorial y granulación de información difusa’ fue desarrollado por Z. Li (2022). en la Escuela de Ingeniería de la Universidad de Huzhou, República Popular China. El objetivo principal de la investigación fue diseñar un modelo avanzado para predecir con precisión la vida útil remanente de baterías de iones de litio. Para ello, se utilizaron los datos históricos de degradación de capacidad provenientes del repositorio de la NASA (específicamente las celdas B0005, B0006 y B0007). Metodológicamente, la propuesta consistió en un enfoque híbrido guiado por datos que aplica la granulación de información difusa para segmentar las series temporales en ventanas dinámicas de entrenamiento y predicción progresiva. Sobre esta estructura se implementó un algoritmo de regresión de soporte vectorial, optimizado mediante la colonia de abejas artificiales para estimar los límites máximos y mínimos de cada intervalo difuso, reconstruyendo la curva continua mediante interpolación lineal y contrastándola con un umbral de falla crítico fijado entre el 70% y el 75% de la capacidad

nominal. Los resultados experimentales demostraron la robustez del modelo al superar significativamente la precisión de una SVR convencional y de los enfoques de optimización estándar sin granulación. Específicamente, el modelo logró optimizar la estimación de la capacidad de almacenamiento de carga alcanzando un error absoluto medio mínimo de 0.0129 Ah y una raíz del error cuadrático medio de apenas 0.0174 Ah en la batería B0005. Asimismo, en lo que respecta al pronóstico del ciclo de vida útil restante, el error absoluto se redujo drásticamente a un rango de entre 1 y 5 ciclos, garantizando un margen de error porcentual máximo de apenas el 3.1%. Finalmente, la investigación concluye que la integración de la granulación difusa permite al algoritmo mitigar con éxito el impacto de las fluctuaciones físicas provocadas por el fenómeno de regeneración de capacidad de las celdas, consolidándose como una herramienta predictiva de alta fidelidad, viable y escalable para los sistemas de gestión de salud de baterías en vehículos eléctricos.

Los datos del repositorio de la NASA fueron utilizados en la investigación titulada 'Predicción de la vida útil remanente de baterías de iones de litio mediante el modelo CEEMDAN y WOA-SVR', siendo CEEMDAN las siglas en inglés traducidas como descomposición modal empírica de conjunto completo con ruido adaptativo, WOA, las siglas que significa algoritmos de optimización de ballenas y SVR, regresión de soporte vectorial. Esta investigación fue desarrollada por Meng (2022), en la Facultad de Ingeniería Electrónica de la Universidad Xinhua de Anhui, Hefei, China. El objetivo principal fue desarrollar el modelo de predicción de la capacidad de almacenamiento de carga de las baterías de ion de litio usando el algoritmo optimizador de ballenas para optimizar los parámetros de la función del núcleo del algoritmo de regresión de soporte vectorial. Metodológicamente, el proceso inicia aplicando el algoritmo CEEMDAN sobre los datos degradados de capacidad de las baterías de prueba (modelos B0005, B0006, B0007 y B0018) para reducir el impacto de ruidos externos aleatorios descomponiéndolos en funciones de modo intrínseco y seleccionando los componentes sensibles mediante un coeficiente de correlación con un umbral mayor a 0.05. Posteriormente, los datos procesados y normalizados en el rango (-1, 1) se dividieron en subconjuntos de entrenamiento utilizando proporciones del 40% y 50% de los ciclos totales. Con estos datos se construyó un modelo de regresión híbrido donde el algoritmo de optimización de ballenas (WOA) determinó de manera óptima los parámetros críticos de penalización y del núcleo de función de base radial del modelo SVR, minimizando el error cuadrático

medio. En la evaluación del desempeño frente a otras metodologías, los resultados demostraron que el modelo propuesto CEEMDAN-WOA-SVR superó notablemente al enfoque convencional WOA-SVR sin filtrado previo de ruido en todas las pruebas. Específicamente, para la batería B0006 con un inicio de predicción en el ciclo 84, el modelo propuesto logró reducir el error absoluto medio de 0.0270 a 0.0084, registrando una disminución del 69% en esta métrica. Asimismo, las conclusiones muestran que las predicciones de la vida útil remanente alcanzaron una raíz del error cuadrático medio de 0.0084 para la batería B0007, superando también en precisión a algoritmos de la literatura como el método de colonia artificial de abejas combinado con SVR y el modelo de descomposición modal empírica de conjunto combinado con optimización del lobo gris (EEMD-GWO-SVR), consolidando la propuesta como una herramienta altamente predictiva, viable y escalable para los sistemas de gestión de salud de baterías.

En el trabajo de investigación cuyo título es 'Predicción de la vida útil restante de baterías de litio basada en el análisis de componentes principales y regresión del proceso gaussiano mejorado', realizado por Xing (2023), en la Facultad de Ingeniería Mecánica de la Universidad de Ciencia y Tecnología de Shanghái – China, se utilizó algoritmos de procesos gaussianos con un análisis previo de componentes principales para alcanzar el objetivo, que fue el de obtener modelos de regresión para predecir la vida útil remanente. Metodológicamente, el estudio consistió en extraer inicialmente cuatro indicadores de salud a partir de las curvas de ciclo de voltaje durante la carga constante y variación térmica de las baterías (modelos B0005, B0006 y B0007). Tras validar mediante el análisis de correlación de Spearman, los indicadores poseían una alta afinidad con la degradación de la capacidad (valores superiores a 0.90), en estas características se implementó el algoritmo de análisis de componentes principales (PCA) para reducir la dimensionalidad y evitar la redundancia de datos, obteniendo un factor de salud indirecto con una tasa de contribución acumulada superior al 80%. Posteriormente, se desarrolló un modelo de Regresión de Procesos Gaussianos Mejorado (IGPR) que reemplazó la función de media cero tradicional por una función de media lineal de tendencia decreciente para mitigar la incertidumbre en predicciones a largo plazo, acoplada a una función de covarianza compuesta que superpone un término exponencial para la degradación general y otro para las fluctuaciones menores. Durante la fase experimental, el primer 60% de los datos se empleó para el entrenamiento y el 40% restante se usó para pruebas de predicción. El rendimiento del modelo PCA-IGPR se comparó con modelos

convencionales como, la regresión de soporte vectorial (SVR), procesos gaussianos, y con modelos de la literatura como GWO-SVR y LSTM-SVR. Los resultados cuantitativos revelaron que el rendimiento del método tradicional de procesos gaussianos, alcanzó un valor de la raíz del error cuadrático medio (RMSE) de 1.46% (0.0146) para la batería B0005, mientras que el modelo óptimo PCA-IGPR minimizó radicalmente los márgenes de error de los tres conjuntos de datos, registrando un RMSE de 0.59% para la batería B0005, 1.17% para la B0006 y 0.96% para la B0007, con errores porcentuales absolutos medios (MAPE) de solo 0.34%, 0.67% y 0.53% respectivamente. Se concluye formalmente indicando que la fusión de PCA con el modelo IGPR no solo incrementa drásticamente la precisión y la eficiencia computacional en comparación con los esquemas SVR y GPR clásicos, sino que dota al sistema de una robustez superior ante variaciones no lineales y ruidos aleatorios, consolidándose como una estrategia altamente efectiva y con excelentes perspectivas de aplicación práctica para el diagnóstico predictivo de baterías de litio.

El trabajo titulado como 'Método de optimización híbrido de lobo gris en un marco de regresión de soporte vectorial para una predicción altamente precisa de la vida útil restante de baterías de iones de litio' desarrollado por Zhang (2023), en la Escuela de Ingeniería de la Información de la Universidad del Suroeste de Ciencia y Tecnología de Mianyang – China, en el trabajo se utiliza el algoritmo de lobo gris para optimizar hiperparámetros en la etapa de entrenamiento del algoritmo de regresión de soporte vectorial usando datos del proceso de carga y descarga de baterías del repositorio de la NASA. Esto para alcanzar el objetivo que es el de generar un modelo predictivo altamente eficiente. Metodológicamente, la investigación abordó el complejo problema de la selección del parámetro del núcleo de función de base radial y el coeficiente de penalización dentro de un modelo de Regresión de Soporte Vectorial (SVR) utilizando la capacidad como indicador directo de salud. Para solucionar el estancamiento prematuro y la susceptibilidad de caer en óptimos locales del Algoritmo de Optimización de Lobo Gris (GWO) estándar, se introdujo una estrategia de evolución diferencial basada en operadores de varianza, cruzamiento y selección global, formulando así el algoritmo híbrido HGWO. En la fase de experimental, se procesaron los conjuntos de datos de las baterías comerciales B0005, B0006, B0007 y B0018, destinando el primer 50% de los ciclos totales para la etapa de entrenamiento y el restante para pruebas de pronóstico. Durante el análisis comparativo del desempeño frente a los esquemas convencionales

SVR y GWO-SVR independientes, el modelo conjunto propuesto HGWO-SVR redujo drásticamente el error absoluto medio (MAE) y la raíz del error cuadrático medio (RMSE) en todas las celdas. Cuantitativamente, para la batería B0005 el MAE disminuyó en un 83.4% frente a SVR y un 45.3% respecto a GWO-SVR. Asimismo, en la batería B0006 el método HGWO-SVR logró predecir con exactitud absoluta el ciclo de falla real reduciendo el RMSE en un 87.2% en comparación al modelo SVR base. Según los resultados se concluye que se logró mejorar la eficiencia en la predicción de la vida útil remanente de las baterías alcanzando valores de la raíz del error cuadrático medio (RMSE) de 0.0026 (0.26%) para la celda B0006 y de 0.0039 (0.39%) para la celda B0007. El estudio concluye formalmente demostrando que la integración de la estrategia de evolución diferencial con lobo gris estabiliza el RMSE del modelo final dentro de un margen estricto inferior al 1.0% ante diversos puntos de inicio de predicción (40%, 50%, 60% y 70%), cerrando una brecha en la literatura científica al proveer una herramienta predictiva robusta, de alta calidad y rápida convergencia idónea para sistemas avanzados de gestión de salud de baterías.

En la investigación titulada ‘Estimación del estado de salud y predicción de la vida útil restante de una batería de iones de litio con un regresor de apilamiento de dos capas’, presentado por Yuan (2023) y desarrollado en la Facultad de Materiales y Energía de la Universidad de Ciencia y Tecnología Electrónica de Chengdu-China, el objetivo fue obtener un modelo de regresión para predecir el estado de salud (SOH) y la vida útil restante (RUL) de las baterías de ion de litio sobre una base de datos de entrenamiento, evaluación y validación obtenida del repositorio de la NASA. Metodológicamente, se propuso un marco de diagnóstico basado en un regresor de apilamiento (SR) de dos capas donde, tras extraer y limpiar mediante coeficientes de correlación de Pearson quedaron las variables esenciales de voltaje, corriente, número de ciclos y temperatura, los cuales se utilizaron para entrenamiento del algoritmo. Este se construyó en su primera capa mediante la fusión paralela de cinco estimadores bases individuales: bagging, regresión de potenciación de gradiente (GBR), regresión de soporte vectorial (SVR), Hist-GBR y AdaBoost, encargados de procesar simultáneamente los datos de entrada y filtrar la información inválida. Posteriormente, los resultados intermedios generados fluyeron hacia una segunda capa compuesta por un algoritmo de regresión lineal (LR), el cual actuó como meta-aprendiz para realizar el ajuste final de los datos sin requerir un ajuste excesivo de hiperparámetros. En la fase de validación experimental, se utilizaron datos

de 15 baterías sometidas a diferentes políticas de descarga y temperaturas extremas (4 °C, 24 °C y 43 °C), evaluando el rendimiento del modelo frente a redes neuronales recurrentes (RNN), memoria a largo plazo (LSTM), SVR convencional, máquinas de vectores de relevancia (RVM), regresión por procesos gaussianos (GPR) y el modelo avanzado AST-LSTM. Los resultados cuantitativos revelaron que con apenas el 30% de los datos históricos de entrenamiento, el modelo regresor de apilamiento de dos capas superó la falta de generalización de los predictores individuales, logrando para la estimación del estado de salud un RMSE medio de 0.0070 (0.70%), con valores específicos de RMSE de 0.0063 para la batería B0005, 0.0068 para la B0006, 0.0080 para la B0007 y 0.0070 para la B0018. Asimismo, en la predicción del RUL, el modelo SR demostró una alta precisión logrando un error absoluto de cero ciclos al fijar el inicio del pronóstico en el ciclo 70 para la batería B0005, y registrando un error absoluto medio de solo 2 ciclos junto a un RMSE general de 0.0173. Se concluye que el modelo produce predicciones precisas, con valores mínimos de la raíz del error cuadrático medio de 0.0063 al predecir el tiempo de vida útil remanente, demostrando una estabilidad, robustez y capacidad de generalización superiores que lo consolidan como una solución altamente viable para la gestión de salud en diversas tecnologías de baterías de iones.

Otra investigación orientada a estudiar el monitoreo de la degradación de las baterías de ion de litio publicado con título ‘Mejora de la estimación del estado de salud de las baterías de iones de litio mediante datos de carga no etiquetados’, elaborado por Lin (2023) y desarrollado en el Laboratorio Estatal Clave para la Ingeniería de Sistemas de Manufactura de la Universidad Xi’an Jiaotong-China, el objetivo fue generar modelos de regresión para mejorar la eficacia en la predicción del estado de salud (SOH) usando datos de carga abundantes y no etiquetados. Metodológicamente, el estudio propuso un marco de aprendizaje semi-supervisado (SSL) basado en co-entrenamiento, que mitiga la escasez de datos. Para ello, se extrajeron tres indicadores de salud interpretables desde un segmento corto de voltaje de solo 0.1 V en la curva de carga de corriente constante: el tiempo de carga, la desviación estándar del voltaje y la asimetría de la distribución. Estos indicadores alimentaron dos regresores heterogéneos basados en los algoritmos de K-vecinos cercanos (KNN) y regresión de soporte vectorial (SVR), los cuales realizaron un etiquetado mutuo para asignar pseudo-etiquetas de SOH a los datos masivos no etiquetados provenientes de centros de operación real, expandiendo el conjunto de entrenamiento de forma iterativa. Durante la validación experimental con los repositorios

de CALCE, Oxford y XJTU-EVC, el método propuesto fue comparado frente a dos esquemas supervisados tradicionales, uno basado en KNN puro y otro basado en la combinación supervisada KNN+SVR. De acuerdo con los resultados, se consiguió mejorar la eficiencia reduciendo el RMSE promedio en un 47% y 21% frente a ambos, respectivamente. Utilizando datos etiquetados de una única celda de prueba, el enfoque SSL alcanzó un RMSE promedio de apenas 0.55% al evaluar las celdas restantes. Incluso en escenarios de entrenamiento desfavorables empleando una celda con degradación anormal (Celda 2), el modelo semi-supervisado demostró una alta robustez, cuyo resultado del RMSE fue de solo 0.59% para la Celda 5, superando al error de la línea base supervisada de 0.61%. El estudio concluye formalmente indicando que la incorporación sistemática de grandes volúmenes de datos industriales no etiquetados estabiliza la precisión del diagnóstico online y dota al modelo de una generalización superior que aventaja a los métodos supervisados del estado del arte, reduciendo drásticamente los costos y tiempos asociados a los ensayos de envejecimiento en laboratorio.

Con el fin de mejorar aún más la eficiencia en la predicción de la vida útil remanente, el trabajo desarrollado por Wu (2024), cuyo título es 'Predicción de la vida útil restante de baterías de iones de litio basada en una red neuronal y un filtro Kalman adaptativo sin perfume', llevado a cabo en la Escuela de Computación e Información Electrónica de la Universidad de Guangxi de Nanning – China, se edesarrolló un modelo compuesto basado en el algoritmo de K-vecinos cercanos, el algoritmo de memoria de largo y corto plazo y redes neuronales convolucionales (convolutional neural networks (CNN)), el cual usa error relativo porcentual para determinar la precisión en la predicción de la vida útil. Metodológicamente, la investigación propone un enfoque que fusiona de forma sinérgica de métodos basados en datos y basados en modelos físicos, el cual inicia extrayendo tres características indirectas (F1: intervalo de tiempo de caída de voltaje igual entre 3.5 V y 3.9 V, F2: tiempo en que el voltaje de descarga alcanza su punto mínimo, y F3: duración de la carga en modo de corriente constante), validadas con coeficientes de correlación de Pearson superiores a 0.80. Estas variables se normalizaron en el rango (0, 1) e ingresan a una red híbrida profunda denominada CNN-BiLSTM-AM, donde una capa CNN extrae los rasgos espaciales, una red de memoria a largo y corto plazo bidireccional (Bi-LSTM) captura las dependencias secuenciales en ambos sentidos temporales, y un mecanismo de atención Squeeze-and-Excitation (SE-AM) pondera selectivamente las características clave bloqueando el ruido redundante. Los vectores resultantes se emplean

como mediciones predictivas dentro del Filtro Kalman Adaptativo sin Perfume (AUKF), el cual actualiza iterativamente las ecuaciones de estado y covarianza del sistema mediante una transformación unscented, calculando la ganancia de Kalman y ajustando adaptativamente el ruido del proceso y de medición para mitigar la incertidumbre inherente del pronóstico. En la fase experimental se llevó a cabo con los datos de los repositorios de la NASA (baterías B0005, B0007, B0018 y B0046) y CALCE (celdas CS2-36 y CS2-38), el modelo CNN-Bi-LSTM-AM-AUKF fue comparado con algoritmos tradicionales de la literatura como DBO-SVR, INFO-KELM y WOA-ELM. De acuerdo a los resultados se concluye indicando que el modelo de regresión obtenido alcanza una precisión máxima de estimación de la vida útil remanente con un valor mínimo de la raíz del error cuadrático medio (RMSE) de hasta 0.0030 y un error absoluto medio (MAE) mínimo de 0.0024 para la batería CS2-38 en un punto de inicio del 60%, superando drásticamente a la red Bi-LSTM individual y reduciendo el RMSE promedio en un 0.0167 respecto al modelo CNN-Bi-LSTM-AM sin filtro. El estudio concluye formalmente validando que la integración del filtro AUKF no solo dota al sistema de una robustez superior para rastrear los fenómenos de regeneración de capacidad, sino que garantiza el reemplazo oportuno de celdas comerciales degradadas antes de alcanzar el umbral crítico del 70%, previniendo de manera efectiva fallas por fuga térmica y optimizando la confiabilidad global del sistema de gestión de baterías.

La investigación titulada ‘Un nuevo método de estimación inteligente de alta precisión para el estado de salud de la batería’, desarrollado por Y. Liu (2025) en la Facultad de Ingeniería Mecánica y Automatización de la Universidad de Fuzhou - China, con el objetivo de conseguir mejoras en las predicciones del estado de salud y el tiempo de vida útil remanente de las baterías de ion de litio. Para ello, se elaboró el algoritmo de extracción de características de entropía difusa multiescalar con desplazamiento temporal refinado de orden fraccional (FRTSMFE) junto con el algoritmo de estimación de máquina de soporte vectorial de mínimos cuadrados (LSSVM) y el algoritmo de optimización de valle de energía (EVO) para optimizar hiperparámetros. Las características se dividieron en conjuntos de entrenamiento y prueba, implementando el algoritmo metaheurístico EVO para sintonizar automáticamente el factor de penalización y el parámetro del núcleo de función de base radial de la LSSVM, minimizando el error cuadrático medio. Los modelos de regresión se obtuvieron utilizando los datos de la batería B0018 del repositorio de la NASA para la etapa de entrenamiento, mientras que

las celdas B0005, B0006 y B0007 sirvieron para la validación del modelo, extendiendo también las pruebas de robustez hacia los repositorios CALCE, FB, MIT, XJTU y TJU. Durante el análisis comparativo frente a esquemas tradicionales como SVM, GPR y ELM, así como contra optimizadores avanzados de la literatura, la propuesta demostró una precisión significativamente superior. Según los resultados cuantitativos orientados a la predicción del estado de salud, se obtuvo un valor RMSE de 0.5683% (aproximadamente 0.57%) junto con un error porcentual absoluto medio (MAPE) de 0.5009% para la batería B0005, mientras que las celdas B0006 y B0007 registraron valores de RMSE de 0.8026% y 0.6844% respectivamente, superando los márgenes de error de las curvas de capacidad incremental tradicionales. La investigación concluye formalmente determinando que la sinergia entre el extractor FRTSMFE y el optimizador EVO-LSSVM no solo estandariza eficientemente las escalas de muestreo y suprime perturbaciones externas, sino que captura con éxito los fenómenos complejos de regeneración de capacidad y caídas no lineales, consolidándose como un marco predictivo universal, altamente preciso y robusto para sistemas inteligentes de gestión de baterías comerciales permitiendo una eficiencia de 0.57% en la predicción del estado de carga.

2.4. Hipótesis

2.4.1. Hipótesis General

La composición de funciones de covarianza en procesos gaussianos permite mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio.

2.4.2. Hipótesis Específicas

a. La composición de funciones de covarianza en procesos gaussianos permite obtener valores mínimos del error cuadrático medio.

b. La composición de funciones de covarianza en procesos gaussianos permite obtener valores mínimos del logaritmo negativo de la probabilidad marginal.

2.5. Identificación de Variables e Indicadores

Variable Independiente

La composición de funciones de covarianza en procesos gaussianos.

Variable Dependiente

La eficiencia en la predicción del estado de carga en baterías de ion de litio.

2.6. Operacionalización de Variables

Tabla 2.4. Operacionalización de variables.

Variables	Definición Conceptual	Definición Operacional	Dimensiones	Indicadores	Índices
Variable independiente: Composición de funciones de covarianza en procesos gaussianos.	Se define como la combinación algebraica de dos o más funciones de covarianza a través de la suma o producto de las mismas.	Se define como la combinación de funciones que se adaptan a las características no lineales propios del comportamiento de degradación de la capacidad de almacenamiento de carga de las baterías.	Número de ciclo, voltaje, temperatura y capacidad de carga eléctrica de baterías del Repositorio de la NASA (baterías : B005 y B007) y del repositorio de Oxford (baterías: C1, C2,C3 C5, C6)	Raíz del error cuadrático medio	Vida útil remanente (VUR) Capacidad de carga actual (CA) $0.7 \leq VUR \leq CA$
Variable dependiente: Eficiencia en la predicción del estado de carga en baterías de ion de litio	Se define a partir del error cuadrático medio, siendo esta una métrica de precisión.	Se define como la diferencia del valor cuantitativo estimado por el modelo y valor real (esto referente a la capacidad de almacenamiento de carga de la batería)	Número de ciclo, voltaje, temperatura y capacidad de carga eléctrica de baterías del Repositorio de la NASA (baterías : B006 y B018) y del repositorio de Oxford (baterías: C4, C7 y C8)	Logaritmo negativo de la probabilidad marginal	Estado de Salud de la batería (ES) $0 \leq ES \leq 1$

CAPITULO III: METODOLOGÍA

3.1. Ámbito de estudio: localización política y geográfica

El desarrollo de la presente investigación se lleva a cabo en la ciudad de Cusco, pero las características del desarrollo mismo de la investigación no dependen de la ubicación geográfica, ya que el desarrollo y la obtención de los modelos dependen únicamente de los datos y no de las condiciones meteorológicas o geográficas.

3.2. Tipo y nivel de investigación

3.2.1. Tipo de investigación

Dado que se pretende resolver un problema práctico (que es el de predecir de forma eficiente el estado de carga de baterías de ion de litio), utilizando la teoría de procesos gaussianos en el entorno de aprendizaje automático, todo esto para mejorar el proceso de predecir el comportamiento futuro del estado de carga y que esto posteriormente sirva para el monitoreo y una mejor gestión de la energía de almacenamiento, se determina que la presente investigación es del tipo aplicada, es aplicada porque, se usa el conocimiento teórico (de los procesos Gaussianos) para generar un problema práctico (Carrasco, 2019), el cual es mejorar la eficiencia en la predicción del estado de carga de baterías.

3.2.2. Nivel de investigación

En el desarrollo de la presente investigación se manipula la estructura misma de los procesos gaussianos (variable independiente) para observar el efecto que causa en la eficiencia al predecir el estado de carga (variable dependiente) de las baterías de ion de litio. Por consiguiente, la presente investigación es correlacional y explicativo, con un enfoque cuantitativo, y de diseño experimental.

Es correlacional, porque a través de una matriz de correlación se consiguió obtener el grado de correlación entre los parámetros que influyen en el modelo de regresión explicando las causas. Es cuantitativo porque se realizó la cuantificación a través de indicadores la precisión con la cual los modelos de regresión realizan las predicciones del estado de carga eléctrica. Es de diseño experimental, porque, a través de la manipulación de la variable independiente (que es la composición de funciones de covarianza) se observan los efectos en la eficiencia para predecir el estado de carga en las baterías de ion de litio.

3.3. Unidad de análisis

Para este caso la unidad de análisis son las baterías de ion de litio a través de las características de corriente, voltaje, temperatura y capacidad de carga eléctrica. Las baterías usualmente están compuestas por un electrodo positivo de óxido de litio y cobalto $LiCoO_2$ y un electrodo negativo de carbono de litio Li C. con una capacidad nominal de carga de 0,74 Ah (Bole et al., 2014).

Los datos del repositorio de la NASA (Saha & Goebel, 2007), corresponde a baterías compuesta por un electrodo positivo de óxido de litio y cobalto (Li_xCoO_2) y un electrodo negativo de carbono litiado (Li_xC). Estos materiales activos se adhieren a colectores de corriente de lámina metálica en ambos extremos de la celda, aislados mediante un separador de polímero microporoso que permite el flujo de iones de litio. El electrolito facilita la difusión de estos iones (Li+) entre los electrodos, los cuales se intercalan en el material activo durante los ciclos de carga y descarga.

3.4. Población de estudio

De los repositorios de Oxford (Howey & Birkl, 2017) y de la NASA (Saha & Goebel, 2007), se pudo obtener datos de 20 baterías de ion de litio, los cuales representan a la población de estudio, las mismas que contienen datos de características como el número de ciclo, temperatura, voltaje y capacidad de carga, que está compuesto por un total de 280 registros, esto hace un total de 8320 datos.

3.5. Tamaño de muestra

La selección de la muestra se efectuó mediante un muestreo no probabilístico por conveniencia, orientado a asegurar la diversidad y heterogeneidad de las unidades experimentales dentro de la población de estudio. La muestra final quedó constituida por 13 baterías, rotuladas como: C1, C2, C3, C4, C5, C6, C7, C8, B005, B006, B007, B018 y B036, lo cual representa el 65% del universo total. A partir de este conjunto, se consolidó una base de datos con 1352 registros, estructurados a través de variables críticas de operación: número de ciclo, temperatura, voltaje y capacidad de carga.

3.6. Técnicas de selección de muestra

La técnica más apropiada para este tipo de análisis es el muestreo por conveniencia, la elección de los datos de la muestra se llevó a cabo a través del análisis visual de los datos, esto se pudo obtener haciendo uso de los algoritmos que se muestran en la Figura

3.1 y la Figura 3.5 para graficar las curvas de la capacidad de carga eléctrica en fusión del número de ciclo y a través del análisis visual se excluyeron las baterías cuyos datos presentaban demasiado ruido, el resultado final de la elección de los datos se muestran a través de los gráficos de capacidad de carga en fusión del número de ciclo de la Figura 3.2 y la Figura 3.6 lo cual representa al conjunto de datos de la muestra.

3.7. Técnicas de recolección de información

La obtención de los datos se llevó a cabo a través del uso de la técnica de revisión documental (o análisis de contenido) ya que todos los datos se obtuvieron de los repositorios de Oxford (Howey & Birkl, 2017) y de la NASA (Saha & Goebel, 2007).

3.8. Técnicas de análisis e interpretación de la información

En la presente investigación se utilizan datos para desarrollar modelos de regresión mediante técnicas de aprendizaje automático (machine learning). A continuación, se describen las etapas del análisis y las metodologías aplicadas en cada fase para la correcta interpretación de los resultados.

3.8.1. Análisis exploratorio de datos.

En esta etapa se busca entender la estructura, las características y la calidad de los datos, para obtener estos requerimientos se utilizan las siguientes técnicas:

- Estadísticas descriptivas (media, mediana y desviación estándar).
- Diagramas (histogramas, diagramas de caja y gráficos de dispersión).
- Limpieza de datos (valores atípicos) y datos faltantes.
- Análisis de correlación entre variables.

3.8.2. Preprocesamiento y transformación de datos

Para elaborar una estructura de datos organizada que mejore la eficiencia del modelo a obtener, se usan las siguientes técnicas:

- Normalización o estandarización.
- Reducción de dimensionalidad.
- Selección y extracción de características.

3.9. Técnicas para demostrar la hipótesis

La técnica para demostrar la hipótesis se lleva a cabo a través de la evaluación del modelo, para lo cual se mide el rendimiento y la capacidad del modelo para predecir

eventos posteriores, para esto se utiliza la técnica de regresión, del cual se obtizene la métrica de evaluación, el cual es la raíz del error cuadrático medio (RMSE).

3.10. Procesamiento y Análisis.

En aprendizaje automático el procesamiento de los datos y el análisis establecen una interdependencia unidireccional, ya que no se puede realizar un buen análisis sin haber realizado un buen procesamiento previo de los datos presentados en su forma original.

3.10.1. Procesamiento.

Los datos del repositorio de Oxford (Howey & Birkl, 2017) y de la NASA (Saha & Goebel, 2007), fueron descargados y se encuentran en formato de archivos '.mat', los mismos datos fueron cargados al drive de Google, luego fueron importados al editor de código Jupyter Notebook en Google Colab. Para poder abrir los archivos fue necesario utilizar la librería de importación 'scipy.io.loadmat', cuando se importa archivos procedentes de Matlab, ocurre una transformación, en donde todos los conjuntos de datos se ocultan en listas dentro de listas y dentro de diccionarios. Por consiguiente, para poder encontrar y extraer los datos referentes a las características de número de ciclo, temperatura, voltaje, y capacidad de carga de la batería, se tuvo que hacer un análisis de la propia estructura del marco de datos importado.

3.10.1.1. Procesamiento del archivo extraído del repositorio de Oxford.

El código para la extracción de los datos de número de ciclo, capacidad de carga eléctrica, temperatura y voltaje se muestran en la Figura 3.1, este código es útil únicamente para extraer datos provenientes del repositorio de Oxford (Howey & Birkl, 2017), donde se cuentan con datos de ocho baterías, con etiquetas que van desde la celda 1, hasta la celda 8. La fila 1 del código de Figura 3.1, corresponde a la activación del acceso a los archivos de Google Drive, en la fila 3 se muestra el código para cargar los datos en el objeto 'df1' que representa un marco de datos (Data Frame), esto es posible gracias al uso del método 'scipy.io.loadmat', el cual corresponde a una librería orientada específicamente para acceder a archivos provenientes de Matlab (o archivos en formato '.mat'), en la fila 5, se genera una lista con las etiquetas, que serán asignadas a las características de cada batería. en las filas 7, 8, 9 y 10, se declaran los diccionarios vacíos, correspondiente a número de ciclo, capacidad de carga, temperatura y voltaje, respectivamente.

Figura 3.1. Código para extraer datos del repositorio de Oxford.

colab Extrayendo datos del repositorio de Oxford

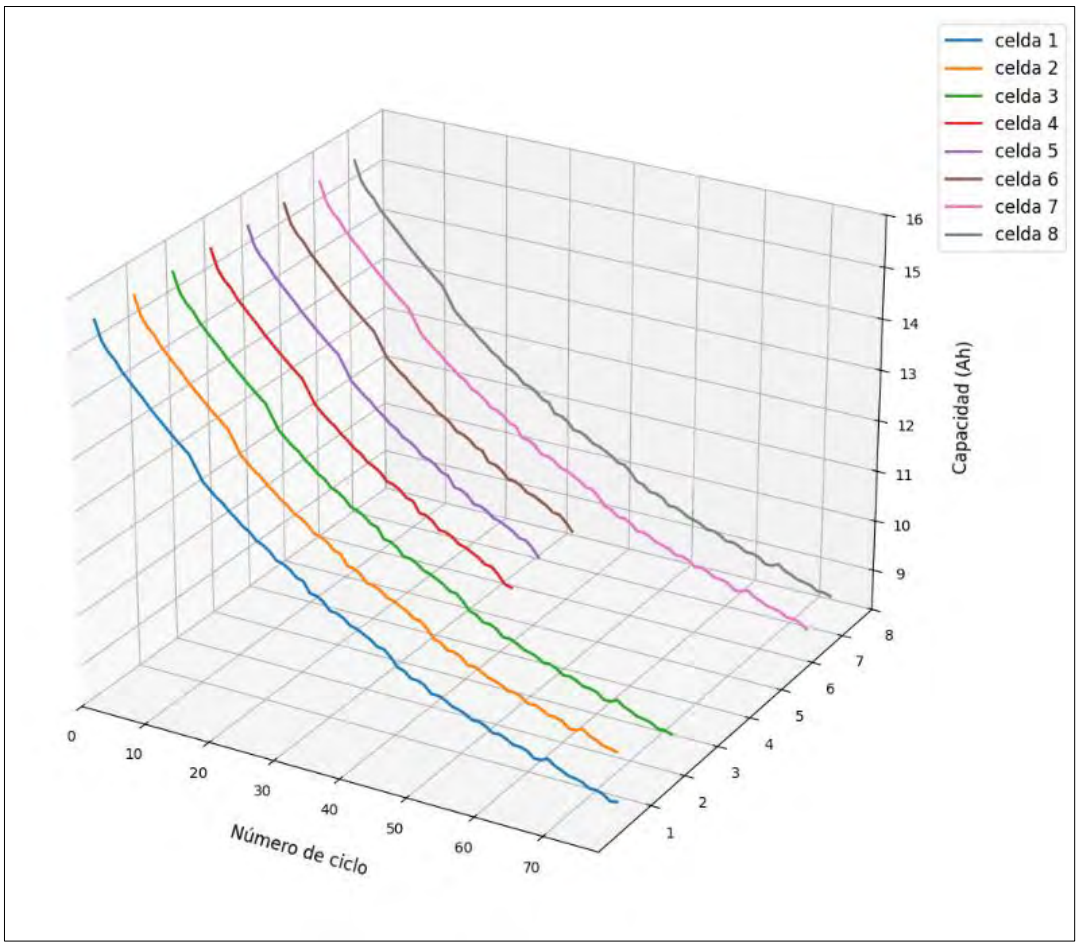
```
1  drive.mount('/content/drive') # accediendo al google drive
2  #Importando datos
3  df1= scipy.io.loadmat('/content/drive/MyDrive/Severson0/Oxford_Battery_Degradation_Dataset_1.mat')
4  # Etiquetas en la silta Cells
5  Cells = ['Cell1', 'Cell2', 'Cell3', 'Cell4', 'Cell5', 'Cell6', 'Cell7', 'Cell8']
6  # Se declara los diccionarios vacíos
7  CycleDic={}     # Diccionario para almacenar el número de ciclo
8  CapacityDic={}  # Diccionario para almacenar la capacidad de la batería
9  TempDic = {}    # Diccionario para almacenar la temeperatura
10 VoltDic = {}    # Diccionario para almacenar el voltaje
11 for k in range(len(Cells)): # bucle para aceder a cada una de las baterías
12   nd0=len(df1[Cells[k]][0][0]) # Longitud de marco de datos df1
13   DicQ={}       # Diccionario para alamcenar la carga
14   Dictime={}    # Diccionario para alamcenar el tiempo
15   DicV={}       # Diccionario para alamcenar el voltaje
16   Dic_T={}      # Diccionario para alamcenar la temperatura
17   for i in range(nd0):
18     lstt = []     # Lista para alamcenar el tiempo
19     lstV = []     # Lista para alamcenar el voltaje
20     lstQ = []     # Lista para alamcenar la carga
21     lst_T = []    # Lista para alamcenar la temperatura
22     for j in range(len(df1['Cell1'][0][0][i][0][0][1][0][0][2])):
23       lstQ.append(df1['Cell1'][0][0][i][0][0][1][0][0][2][j][0])
24       lstt.append(df1['Cell1'][0][0][i][0][0][1][0][0][0][j][0])
25       lstV.append(df1['Cell1'][0][0][i][0][0][1][0][0][1][j][0])
26       lst_T.append(df1['Cell1'][0][0][i][0][0][1][0][0][3][j][0])
27 # Almacenando las carteristica
28     Dictime[i]=lstt
29     DicQ[i]=lstQ
30     DicV[i]=lstV
31     Dic_T[i]=lst_T
32 # Listas temperales
33     lstC=[]
34     Cicle = []
35     Volt = []
36     Temp = []
37 #Almacenando características de número de ciclo, tempreatura, voltaje y capacidad
38     for i in range(len(DicQ)):
39       Cicle.append(i+1)
40       lstC.append((-1)*integrate.trapezoid(DicQ[i],Dictime[i]))
41      # Volt.append(integrate.trapezoid(DicV[i],Dictime[i]))
42       Volt.append(sum(DicV[i]) / len(DicV[i]))
43       Temp.append(sum(Dic_T[i])/len(Dic_T[i]))
44   CycleDic[k] = Cicle
45   CapacityDic[k] = lstC
46   TempDic[k] = Temp
47   VoltDic[k] = Volt
```

Nota. Elaboración propia.

Para poder extraer los datos referentes a las características de las baterías, se desarrolló el algoritmo que inicia en la fila 11 y termina en la fila 43 de la Figura 3.1, previamente se tuvo que realizar un análisis exhaustivo de la estructura interna de datos de ‘df1’. En las filas 22 a 26, se consigue adjuntar los datos de carga, tiempo, voltaje, y temperatura en las listas temporales ‘lstQ’, ‘lstt’, ‘lstV’ y ‘lst_T’, respectivamente, para cada ciclo de descarga. Lo que se desea es obtener, la capacidad de carga eléctrica, voltaje y temperatura como valor único para cada ciclo de descarga, para esto, el bucle ‘for’, de la fila 38, realiza el conteo de los ciclos (fila 39), en la fila 40 se lleva a cabo el cálculo de la capacidad de carga por cada ciclo, este cálculo se realiza a través de la integración trapezoidal de los datos de carga respecto del tiempo, mientras que en las filas 42 y 43 se calculan el voltaje promedio y la temperatura promedio por cada ciclo. Estos resultados

se guardan en los respectivos diccionarios, como se observa en el código que va desde la fila 44 hasta la fila 47 de la Figura 3.1.

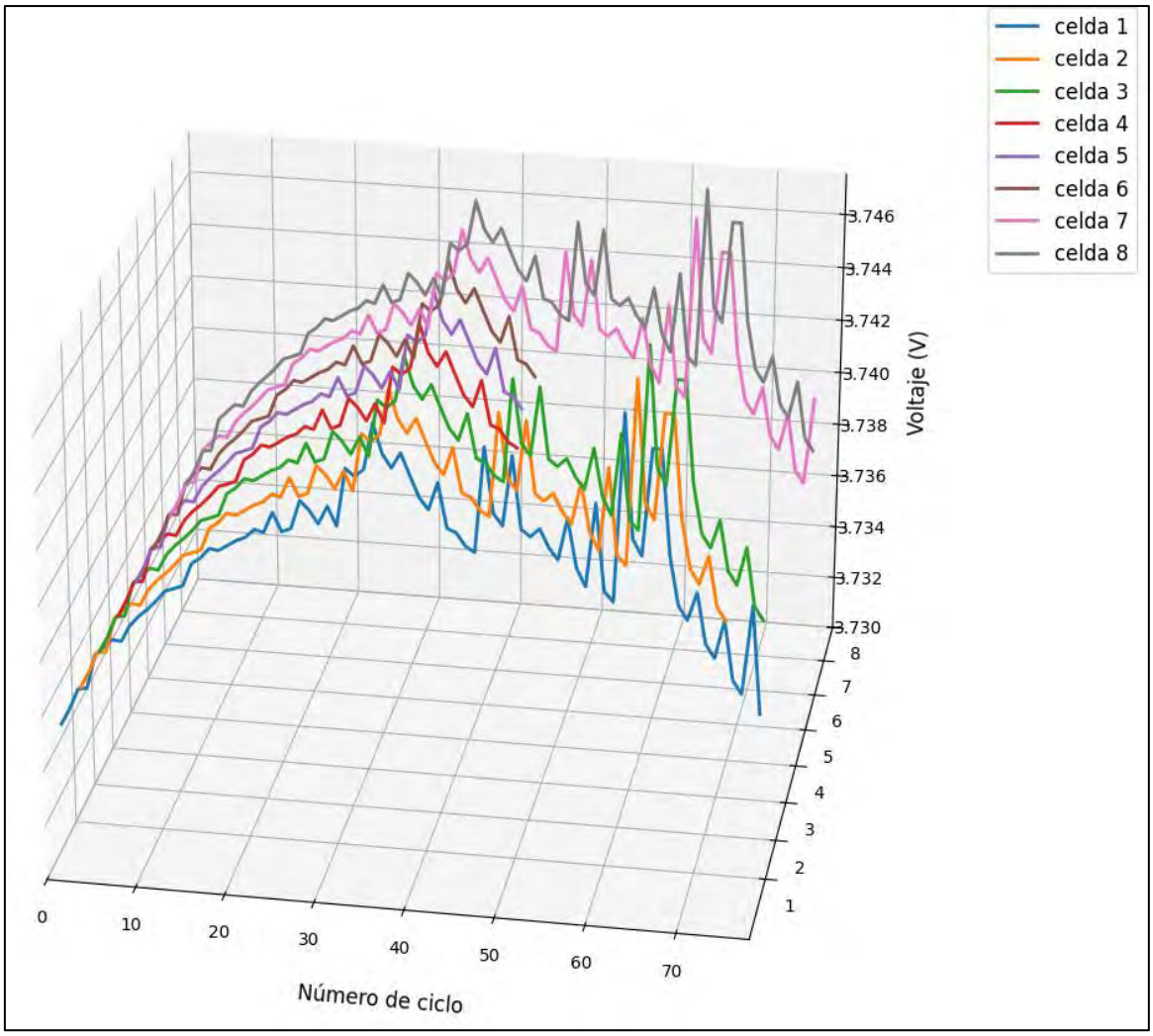
Figura 3.2. Capacidad de carga en función del número de ciclo con datos de Oxford.



Nota. Elaboración propia.

Los datos extraídos de la capacidad de carga de las celdas (o baterías del repositorio de Oxford), se muestran en la Figura 3.2, este es un cuadro comparativo en 3 dimensiones del comportamiento de la degradación de la batería en la capacidad de carga. Se puede observar que la capacidad máxima al inicio es de 15.3 Ah, la cual va disminuyendo de forma inversa al incremento del número de ciclo hasta llegar a un aproximado de 8.84 Ah. este valor de la capacidad de carga se alcanza en el ciclo 78 como máximo.

Figura 3.3. Voltaje en función de número de ciclo para datos de Oxford.

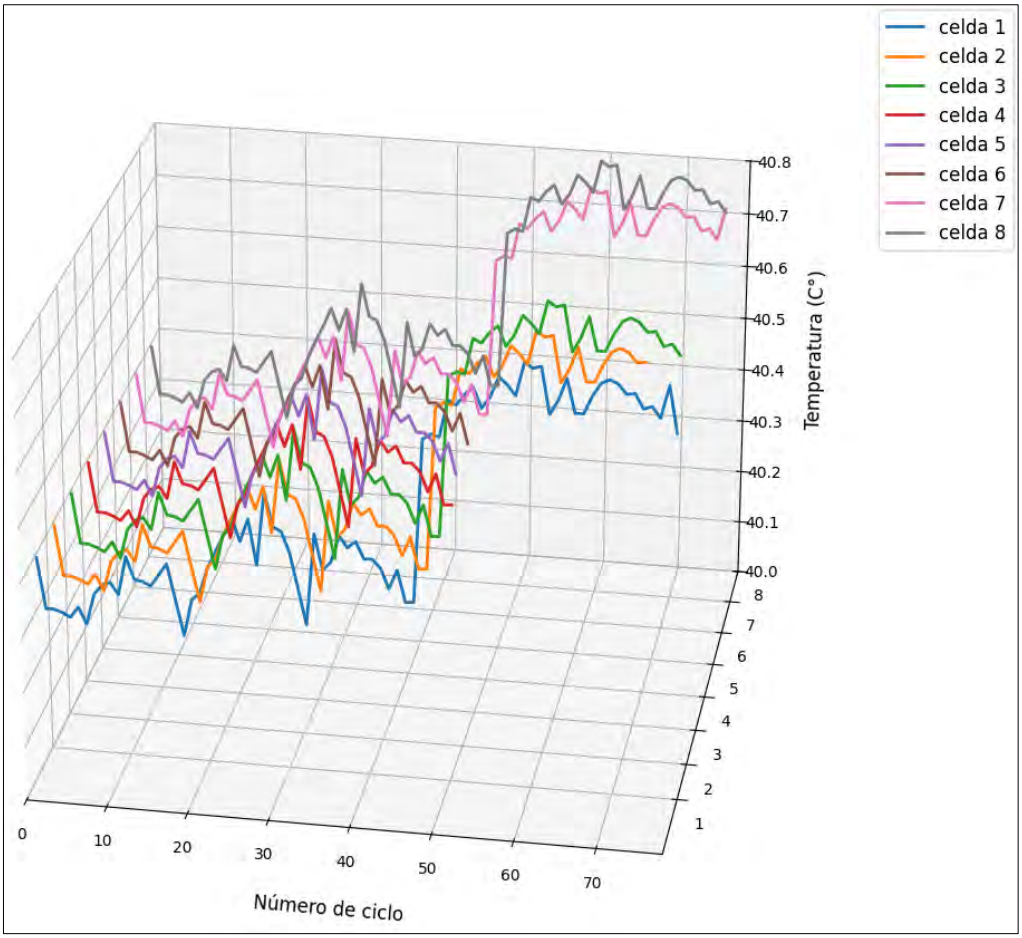


Nota. Elaboración propia.

El comportamiento del voltaje de las baterías de Oxford (Howey & Birkl, 2017), no es lineal a medida que el número de ciclo aumenta, como se puede observar en la Figura 3.3. Este comportamiento es homogéneo, en el sentido de que todas las baterías se asemejan en la variabilidad del voltaje a lo largo de la vida útil. Se puede observar también que el voltaje máximo que es de 3.75 voltios, es suministrada por la batería cuando ésta se encuentra aproximadamente en la mitad de su vida útil.

Luego están los datos de la variación de la temperatura como se puede observar en la Figura 3.4, que también muestra un comportamiento no lineal pero muy diferente a las curvas de capacidad de carga y voltaje. El comportamiento de la variación de la temperatura para todo el conjunto de las baterías, también es homogéneo, la temperatura máxima que alcanza este tipo de baterías es de 40.83 °C, este valor es alcanzado ya casi al final de la vida útil de la batería.

Figura 3.4. Temperatura en función de número de ciclo para datos de Oxford.



Nota. Elaboración propia.

Los valores máximos y mínimos de las características de capacidad de carga, temperatura y voltaje pueden coincidir en las 8 baterías. Un resumen claro de esto se observa en la Tabla 3.1, aquí se puede observar que los valores máximos de la capacidad de carga y voltaje son los mismos para cada celda, al igual que la temperatura mínima y el voltaje mínimo, esto a pesar de que el número de ciclo final es diferente para las 8 baterías.

Tabla 3.1. Resumen de valores máximos de capacidad de carga, voltaje y temperatura (con datos de Oxford).

Batería	Ciclo	Capacidad Máxima (Ah)	Temperatura Máxima (C°)	Voltaje Máximo (V)	Capacidad Mínima (Ah)	Temperatura Mínima(C°)	Voltaje Mínimo (V)
Celda 1	78	15.43	40.83	3.75	8.84	40.30	3.74
Celda 2	73	15.43	40.83	3.75	9.05	40.30	3.74
Celda 3	76	15.43	40.83	3.75	8.93	40.30	3.74
Celda 4	47	15.43	40.57	3.75	10.25	40.30	3.74
Celda 5	46	15.43	40.57	3.75	10.29	40.30	3.74
Celda 6	46	15.43	40.57	3.75	10.29	40.30	3.74
Celda 7	77	15.43	40.83	3.75	8.84	40.30	3.74
Celda 8	76	15.43	40.83	3.75	8.93	40.30	3.74

Por otro lado, los valores máximos de temperatura alcanzados por las baterías muestran cierta variabilidad lo cual parece estar relacionado con el número de ciclo máximo alcanzado por la batería. También existe una variabilidad respecto a la capacidad de carga mínima que experimentan las 8 baterías, lo cual también parece estar relacionado con el número de ciclo máximo que alcanza la batería.

Otra característica que se puede observar de la Tabla 3.1, es que en promedio la diferencia entre la capacidad máxima y la capacidad mínima es de 6 Ah, lo cual es razonable y típico de este tipo de baterías, en cuanto a la diferencia entre la temperatura máxima y mínima es 0.44 °C, esto quiere decir que una ligera variación en la temperatura genera efectos considerables en el estado de carga de las baterías y en referencia a la diferencia entre los valores de voltaje máximo y mínimo, este es de 0.01 voltios. Esto indica que, las variaciones en la capacidad de carga, generan cambios mínimos en el voltaje de la batería, los cual sugiere considerar estas variaciones en el orden de mili voltios.

3.10.1.2. Procesamiento del Archivo extraído del repositorio de la NASA

El segundo conjunto de datos de baterías de ion de litio se obtuvo del repositorio de la NASA (Saha & Goebel, 2007). Se encontró que este conjunto de datos al ser descargado se encontraba en un archivo del tipo '.mat', lo que sugiere que la interface de trabajo y el mismo lenguaje de programación que usaron los autores (Goebel et al., 2008) fue en Matlab. En la Figura 3.5 (a), las filas 2 a 6 , muestran el código correspondiente a la importación de datos a través del método 'scipy.io.loadmat' con el que se accede al archivo '.mat' ubicado en un directorio de Google Drive, en este caso se accede a los

archivos, ‘B0005.mat’, ‘B0006.mat’, ‘B0007.mat’, ‘B0018.mat’ y ‘B0036.mat’. En la fila 8 se agrupa todos los datos extraídos en una sola lista, en la fila 10 se genera las etiquetas para las baterías y en las filas 13 a 16 se declaran los diccionarios vacíos, donde posteriormente se almacena datos de voltaje, temperatura, capacidad de carga y número de ciclo. En la parte (b) de la Figura 3.5, se muestra el código de extracción de características de las baterías, esto inicia con el bucle ‘for’ en la fila 18, el cual recorre los elementos de la lista BS (línea de código 8). Las listas que se muestran en las líneas de código, 20, 21, 22 y 23, corresponde a listas vacías donde se almacenará temporalmente las características de voltaje, temperatura, capacidad de carga y número de ciclo.

Figura 3.5. Código para la extracción de datos del repositorio de la NASA.

colab Extreyendo datos del repositorio de la NASA **(a)**

```
1 #Importando datos del google Drive
2 B05 = scipy.io.loadmat('/content/drive/MyDrive/TesisUNSAAC/B0005.mat')
3 B06 = scipy.io.loadmat('/content/drive/MyDrive/TesisUNSAAC/B0006.mat')
4 B07 = scipy.io.loadmat('/content/drive/MyDrive/TesisUNSAAC/B0007.mat')
5 B18 = scipy.io.loadmat('/content/drive/MyDrive/TesisUNSAAC/B0018.mat')
6 B36 = scipy.io.loadmat('/content/drive/MyDrive/TesisUNSAAC/B0036.mat')
7 #Almacenando todos los datos importandos en la lista BS
8 BS = [B05, B06, B07, B18, B36]
9 # Generando etiquetas para cada conjunto de datos
10 Lst = ["B0005","B0006","B0007", "B0018","B0036"]
11 # Declarando diccionarios para el almacenamiento
12 # de características de voltaje, capacidad, temperatura y voltje
13 dict_V_m = {} # dicionario de voltaje
14 dict_Capacity = {} # diccionario de capacidad
15 dict_Temp_m = {} # diccionario de temperatura
16 dict_Cicle = {} # diccionario de número de ciclo
```

(b)

```
17 # Bucle "for" para ingresar a cada uno de los marcos de datos (Data Frames)
18 for i in range(len(BS)):
19   #Generando listas bacías
20   V_m_List = [] # para voltaje
21   Temp_m_List = [] # para temperatura
22   Capacity_List=[] # para capacidad
23   Cicle_List = []  # para número de ciclo
24   c = 0
25 # En este bucle se ingresa al marco de datos (ejemplo puede ser a B05)
26 # dentro del cual se extrae las caracteristicasa de voltaje, temperatura,
27 # capacidad y número de ciclo
28   for k in range(len(BS[i][Lst[i]]['cycle'][0][0][0]["type"])):
29     if BS[i][Lst[i]]['cycle'][0][0][0]["type"][k][0] == "discharge":
30       c = c+1
31       # Se obtiene el voltaje
32       V_m_List.append(sum(BS[i][Lst[i]]["cycle"][0][0][0][k]["data"]["Voltage_measured"][0][0][0])/
33                       len(BS[i][Lst[i]]["cycle"][0][0][0][k]["data"]["Voltage_measured"][0][0][0]))
34       # integración trapezoidal para obtener la capacidad de la bateria
35       Capacity_List.append(-integrate.trapezoid(BS[i][Lst[i]]["cycle"][0][0][0][k]["data"]["Current_measured"][0][0][0],
36                                                 BS[i][Lst[i]]["cycle"][0][0][0][k]["data"]["Time"][0][0][0]))
37       # Se obtiene la temperatura
38       Temp_m_List.append( sum(BS[i][Lst[i]]["cycle"][0][0][0][k]["data"]["Temperature_measured"][0][0][0])/
39                           len(BS[i][Lst[i]]["cycle"][0][0][0][k]["data"]["Temperature_measured"][0][0][0]) )
40       Cicle_List.append(c)  # Se obtiene el número de ciclo
41   dict_V_m[i] = V_m_List # se guarda el voltaje en el diccionario
42   dict_Capacity[i] = Capacity_List # se almacena la capacidad en el diccionario
43   dict_Temp_m[i] = Temp_m_List # se almacena la temperatura
44   dict_Cicle[i] = Cicle_List    # Se almacena el número de ciclo
```

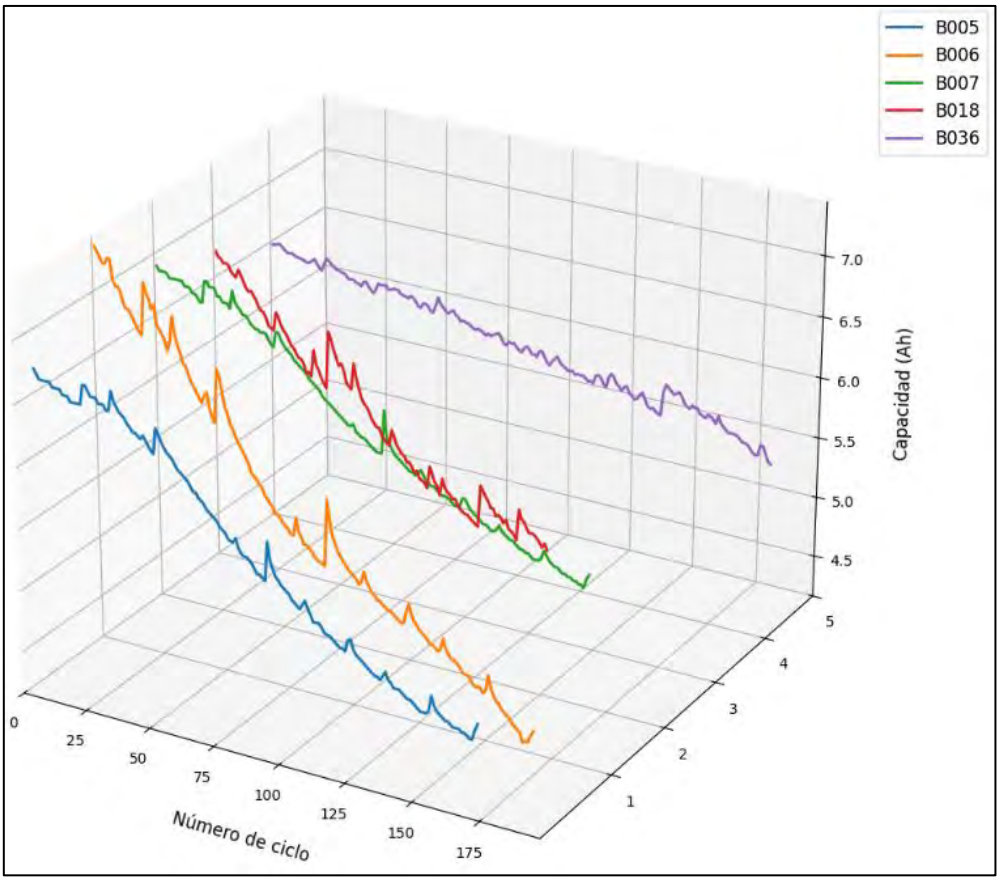
Nota. Elaboración propia.

En la línea de código 18 de la Figura 3.5 (b), se explora e ingresa a la estructura interna del marco de datos, en la línea de código 29 se muestra que solo se extrae

características para el ciclo de descarga. Para esto se obtuvo el voltaje promedio (línea de código 32). La capacidad de carga fue obtenido a través de la integración trapezoidal de los datos de corriente respecto al tiempo (línea de código 35), también se obtuvo la temperatura media y en número de ciclo (líneas de código 38, 39 y 40), luego los datos se guardaron en los diccionarios correspondientes a voltaje, capacidad de carga, temperatura y número de ciclo (esto en las líneas de código 41-44).

Posterior a la extracción de características y almacenamiento de los mismos en sus respectivos diccionarios, es necesario realizar el gráfico de cada una de las características, para tener una idea de su comportamiento a lo largo de su vida útil. Esto se muestra a continuación.

Figura 3.6. Capacidad de carga en función del número de ciclo con datos de la NASA.



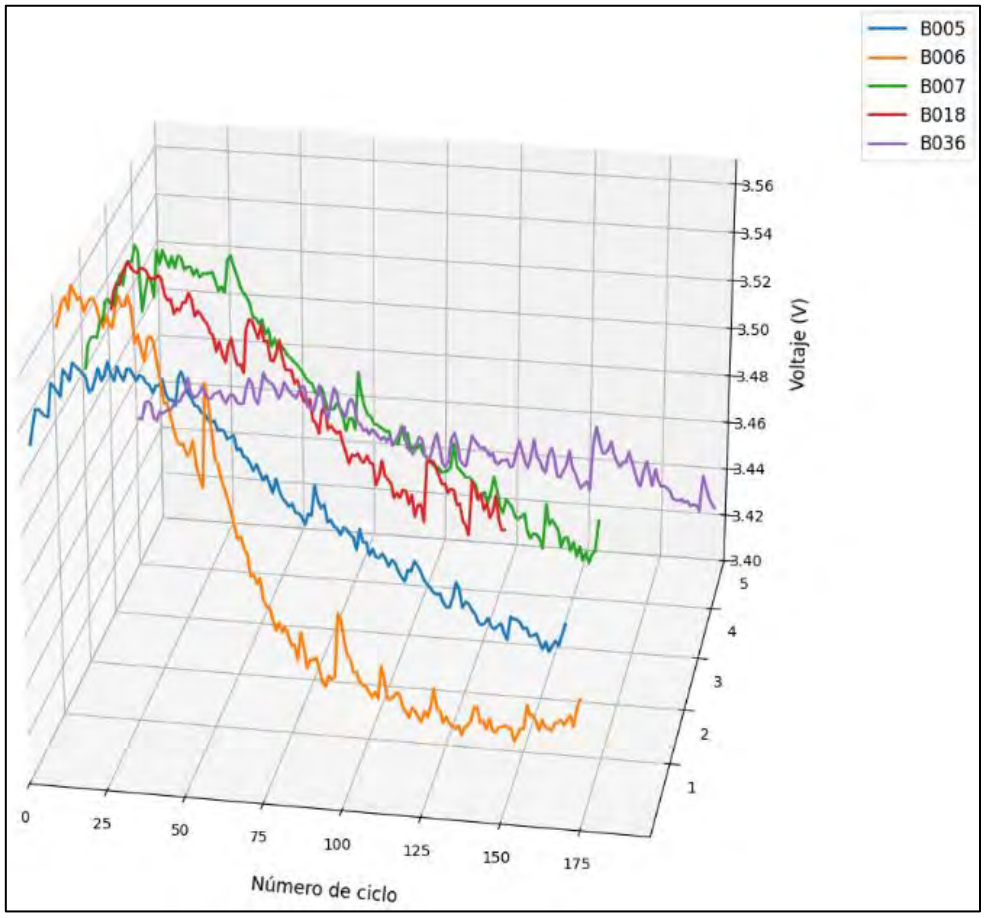
Nota. Elaboración propia.

Como se observó anteriormente, para acceder a los datos de las baterías del repositorio de la NASA (Saha & Goebel, 2007), se generó un código de extracción, el mismo que se muestra en la Figura 3.6, en el código se observa que los datos se encuentran ocultos en diccionarios y listas dentro de listas y que para poder determinar la capacidad

de carga de la batería en cada ciclo como se muestra en la Figura 3.6, se tuvo que hacer una integración trapezoidal, esto se puede observar en las líneas de código 35 y 36 de la Figura 3.5.

La Figura 3.6, muestra un cuadro comparativo del comportamiento de las baterías en capacidad de carga eléctrica a lo largo de los ciclos, claramente la batería B006 alcanza el valor máximo en capacidad que llega hasta 7.37 Ah y un valor mínimo de 4.22 Ah.

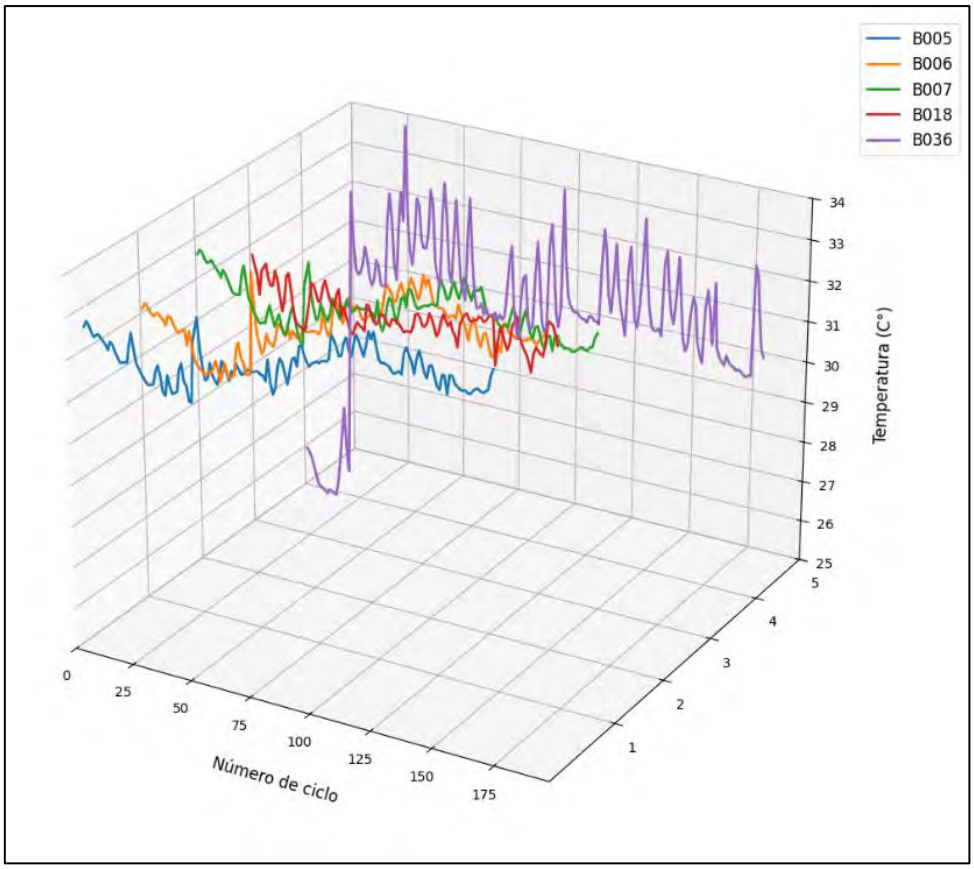
Figura 3.7. Voltaje en función de número de ciclo con datos de la NASA.



Nota. Elaboración propia.

En cuanto a la característica de voltaje, el cuadro comparativo del comportamiento del voltaje a lo largo del número de ciclo de todas las baterías extraídas de la NASA (Saha & Goebel, 2007), se muestra en la Figura 3.7. El comportamiento del voltaje de todo el grupo de baterías no es homogéneo y no existe similitud entre las curvas de voltaje baterías. Los valores máximos y mínimos en voltajes alcanzados fueron de 3.57 V y 3.40 V, respectivamente.

Figura 3.8. Temperatura en función de número de ciclo con datos de la NASA.



Nota. Elaboración propia.

Las variaciones de la temperatura respecto al número de ciclo se muestran en la Figura 3.8, el cuadro comparativo muestra cierta similitud entre las curvas de temperatura de una batería a otra, a excepción de la batería B036, que muestra un comportamiento irregular, con temperatura máxima de 34.7 °C y temperatura mínima de 24.89 °C.

Tabla 3.2. Resumen de Valores máximos y mínimos de características de la batería (con datos de la NASA).

Batería	Ciclo	Capacidad máxima (Ah)	Temperatura máxima (C°)	Voltaje máximo (V)	Capacidad mínima (Ah)	Temperatura mínima (C°)	Voltaje mínimo (V)
B005	168	6.70	34.07	3.56	4.65	31.39	3.46
B006	168	7.37	34.49	3.57	4.22	30.83	3.40
B007	168	6.91	33.51	3.57	5.13	30.99	3.46
B018	132	6.73	31.82	3.55	4.90	30.19	3.44
B036	197	6.50	34.70	3.49	5.61	24.89	3.44

El resumen de los valores máximos y mínimos de la **¡Error! No se encuentra el origen de la referencia.**, muestran la variabilidad de los valores máximos y mínimos de

la capacidad de carga, temperatura y voltaje, cuando se compara entre las baterías. El tiempo de vida de las baterías B005, B006 y B007, alcanza para desarrollar 168 ciclos, estas baterías mostraron cierta semejanza en el comportamiento de las curvas de capacidad de carga, voltaje y temperatura, mientras que las baterías un comportamiento un tanto diferente, pero con la misma tendencia.

3.10.2. Análisis

El análisis consiste en construir la matriz de correlación, para verificar el grado de correlación lineal que guardan las variables, tales como la capacidad de carga eléctrica, voltaje y temperatura, para llevar a cabo el análisis de correlación, se debe realizar previamente la prueba de normalidad, luego se lleva a cabo el entrenamiento de los algoritmos de procesos gaussianos y la obtención de los modelos de procesos gaussianos, adicionalmente para comparar la eficiencia de los modelos de procesos gaussianos, se usa los mismos datos para obtener modelos con otros algoritmos de aprendizaje automático, algoritmos como redes neuronales artificiales, regresión de soporte vectorial y bosques aleatorios (Random Forest).

3.10.2.1. Pruebas Estadísticas

La prueba de normalidad se lleva a cabo mediante la prueba de Shapiro Wilk, cuando se tiene muestras de hasta 50 datos y mediante la prueba de Kolmogorov Smirnov (K-S), cuando se tiene muestras mayores a 50 datos.

Dado que para cada una de las variables consideradas se tienen registros mayores a 50 datos, las pruebas de normalidad se llevan a cabo con la prueba de K-S. Esto, para los datos del repositorio de Oxford (desde la Celda1 hasta la Celda8) y para datos del repositorio de la NASA (B005, B006, B007, B018, B036)

3.10.2.2. Prueba de normalidad para datos de Oxford.

Aquí la evaluación de la normalidad se lleva a cabo considerando a priori el nivel de significancia $\alpha = 0.05$ y se plantea las hipótesis siguientes:

Si P-valor es menor que α, se rechaza la hipótesis nula, por consiguiente, los datos no tienen una distribución normal.

Si P-valor es mayor que α, se acepta la hipótesis nula, por consiguiente, los datos tienen una distribución normal.

Dado que, el conjunto de datos de las baterías de Oxford corresponde a un solo tipo de batería, solo se realiza la prueba de normalidad para la Celda 1 y los resultados se puede generalizar para todo el conjunto de datos. En la Tabla 3.3, se muestra los resultados de la prueba de normalidad K-S que se le aplicó a las características de ciclo, temperatura, voltaje y capacidad de carga. Estos resultados constituyen el representativo para todo el conjunto, que va desde la Celda 1 hasta la Celda 8.

Tabla 3.3. Resultados de la prueba de normalidad para datos de Oxford.

Variable	Estadístico	P-valor	Tipo de distribución
Ciclo	0.062	0.906	Normal
Temperatura	0.191	0.006	No es normal
Voltaje	0.090	0.519	Normal
Capacidad	0.117	0.214	Normal

En cuanto a los resultados que muestra la Tabla 3.3, solo la temperatura no obedece a una distribución normal mientras que los datos referentes a ciclo, voltaje y capacidad de carga si obedecen a una distribución normal.

3.10.2.3. Prueba de normalidad para datos de la NASA

Para los datos de la NASA también se utilizó la prueba de normalidad K-S, debido a que la cantidad de datos supera los 50. Dado que el conjunto de datos corresponde un solo tipo de batería, aquí se realiza la prueba de normalidad solo para la batería B005, para luego generalizar.

Tabla 3.4. Resultados de la prueba de normalidad para datos de la NASA.

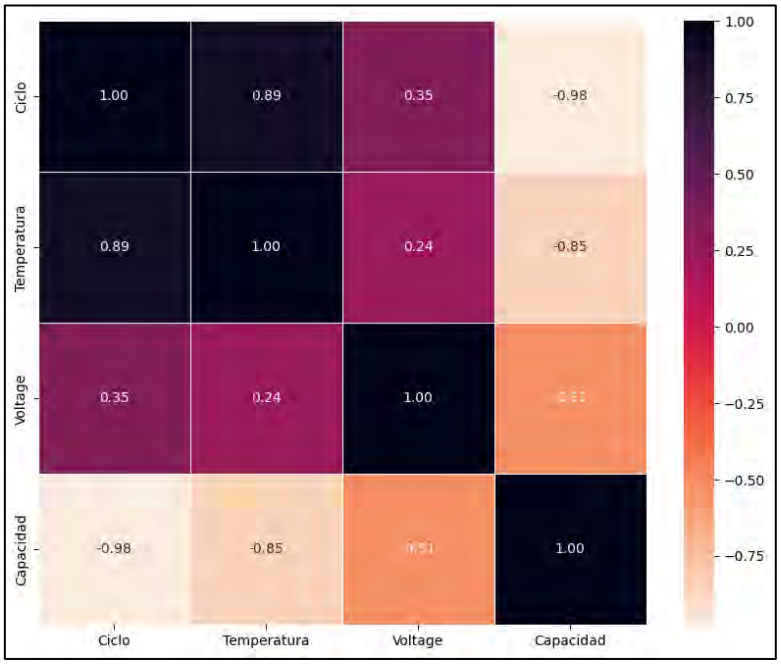
Variable	Estadístico	P-valor	Distribución
Ciclo	0.059	0.571	Normal
Temperatura	0.109	0.032	No es normal
Voltaje	0.124	0.010	No es normal
Capacidad	0.125	0.010	No es normal

Los resultados obtenidos en la Tabla 3.4, muestran que solo los datos del número de ciclo obedecen a una distribución normal y los datos correspondientes a la temperatura, voltaje y capacidad de carga no obedecen a una distribución normal.

3.10.2.4. Análisis de correlación

Para el análisis de correlación es importante tomar en cuenta, tanto los resultados de la Tabla 3.3 y la Tabla 3.4, ya que estos resultados indican la valides de la correlación de Pearson o Spearman. Cuando los datos obedecen a una distribución normal, corresponde llevar a cabo la correlación de Pearson y cuando los datos no obedecen a una distribución normal, corresponde realizar una correlación de Spearman, de acuerdo a esas condiciones se llevó a cabo el análisis de correlación entre las variables predictoras (o parámetros de entrada).

Figura 3.9. Matriz de correlación entre variables para datos de Oxford.

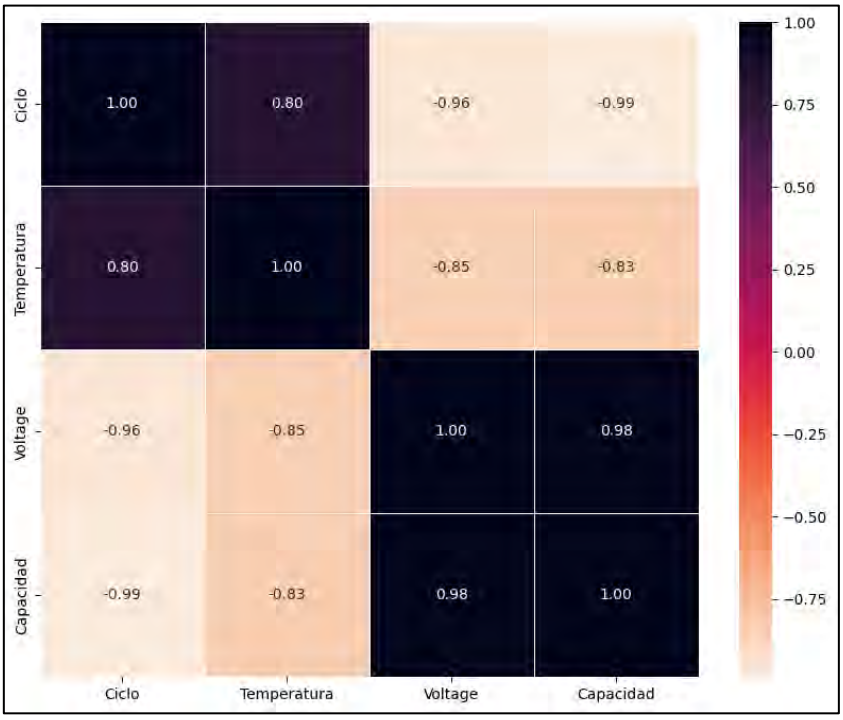


Nota. Elaboración propia.

La Figura 3.9. ilustra de manera concisa el mapa de interacciones para el conjunto de datos de Oxford. El análisis pormenorizado de sus casilleros revela el comportamiento termodinámico y de degradación de las celdas: Capacidad vs. Número de Ciclo con un coeficiente de correlación de -0.98, que denota una correlación negativa casi perfecta. Físicamente, cuantifica el desvanecimiento irreversible de la capacidad conforme la batería acumula ciclos operativos, provocado por fenómenos internos como la pérdida y de gradación litio y la degradación de los electrodos. La capacidad de carga eléctrica con la temperatura obtuvo un coeficiente de correlación de -0.85, lo cual revela una fuerte correlación negativa, lo que significa que los incrementos de temperatura aceleran de forma severa las reacciones químicas secundarias parasitarias, disminuyendo la capacidad

máxima de retención energética. La correlación entre la capacidad de carga y voltaje es de -0.51 la cual tipifica a una correlación negativa moderada. Esto se debe a que las sutiles variaciones en los perfiles de tensión durante el proceso de envejecimiento están vinculadas de manera indirecta con la pérdida de los sitios activos de intercalación en el material activo. Estos resultados tienen implicancias en el modelado, ya que presencia de coeficientes de correlación tan elevados confirma que el fenómeno de degradación química es un sistema altamente interconectado justificando plenamente la integración de una composición de funciones de covarianza (kernels compuestos) en los procesos gaussianos, esta integración permite capturar de forma robusta las interdependencias no lineales y dinámicas.

Figura 3.10. Matriz de correlación entre variables para datos de la NASA.



Nota. Elaboración propia.

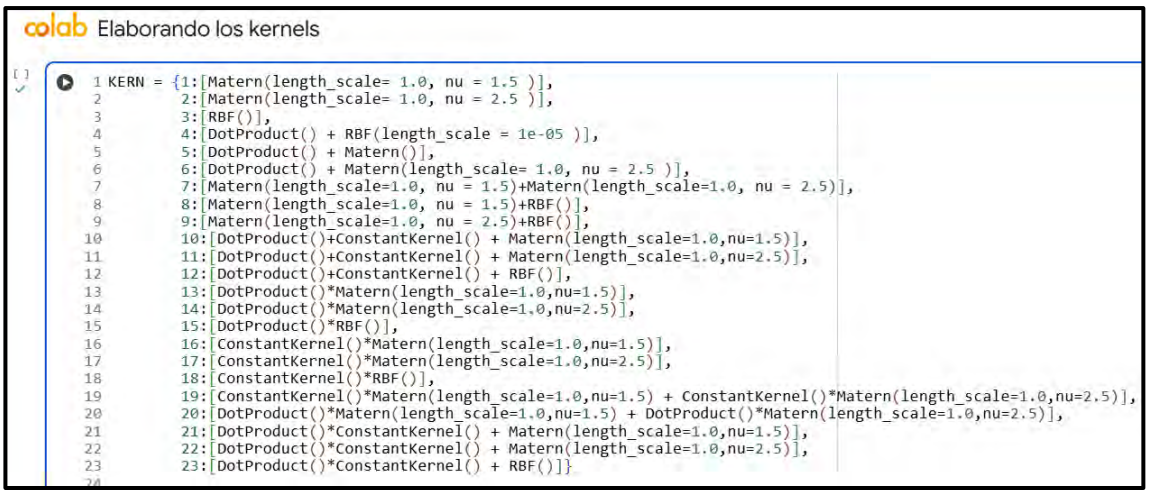
La Figura 3.10 expone la matriz de correlación correspondiente, cuyo desglose detallado permite contrastar estos perfiles con los de Oxford para aportar un valor predictivo fundamental. En primer lugar, la interacción entre capacidad y número de ciclo registra un coeficiente de correlación de -0.99, lo que ratifica una dependencia negativa prácticamente perfecta y valida de forma universal que el avance de los ciclos de descarga constituye el vector principal del envejecimiento de la batería. Por otra parte, al evaluar la capacidad frente al voltaje se obtiene un valor de del coeficiente de correlación de 0.98;

a diferencia de lo observado en Oxford, aquí se manifiesta una correlación positiva extremadamente fuerte que se traduce en que, bajo las condiciones de ensayo en baterías de la NASA, el voltaje operativo decae de forma estrictamente proporcional a la pérdida de capacidad, convirtiéndose en un indicador directo y de alta fidelidad para estimar el estado de carga. Asimismo, la relación de la capacidad frente a la temperatura arroja un valor del coeficiente de correlación de -0.83, lo que confirma la persistencia de una asociación inversa fuerte donde el estrés térmico actúa como un catalizador del envejecimiento celular. Finalmente, al analizar las interacciones mutuas entre las variables predictoras, se observa que la correlación entre ciclo y voltaje es de -0.96 evidencia que el potencial eléctrico máximo disminuye de manera sistemática ciclo tras ciclo, mientras que la correlación de voltaje y temperatura es de -0.85, lo que demuestra la existencia de una caída óhmica interna por la cual, a mayor temperatura, la eficiencia de la tensión en terminales se reduce significativamente; este comportamiento se complementa con la correlación positiva entre ciclo y temperatura la cual es de 0.81, lo que demuestra cómo el uso continuo incrementa progresivamente la resistencia interna de la celda y genera un mayor calentamiento por efecto Joule a lo largo del tiempo.

3.10.2.5. Desarrollo del algoritmo de predicción con procesos gaussianos

Previo al entrenamiento de los algoritmos de procesos gaussianos, se desarrolla el código que define las funciones de covarianza, las mismas que se establecieron en la Tabla 2.1. La composición de funciones de covarianza como una combinación lineal de suma y producto de las funciones de covarianza mencionados en la sección 2.1.2, se muestra en la Figura 3.11. Los valores del parámetro de longitud de escala (length_scale) se inicializaron en un valor de 1.0, para todas las funciones propuestas, valor que posteriormente se actualiza con la optimización de parámetros.

Figura 3.11. Elaboración en código de las funciones (o kernels) de covarianza.



Nota. Elaboración propia.

El desarrollo de todo el algoritmo que permite generar los modelos de procesos gaussianos y guardar los mismos, se llevan a cabo en seis pasos, los cuales se describen a continuación.

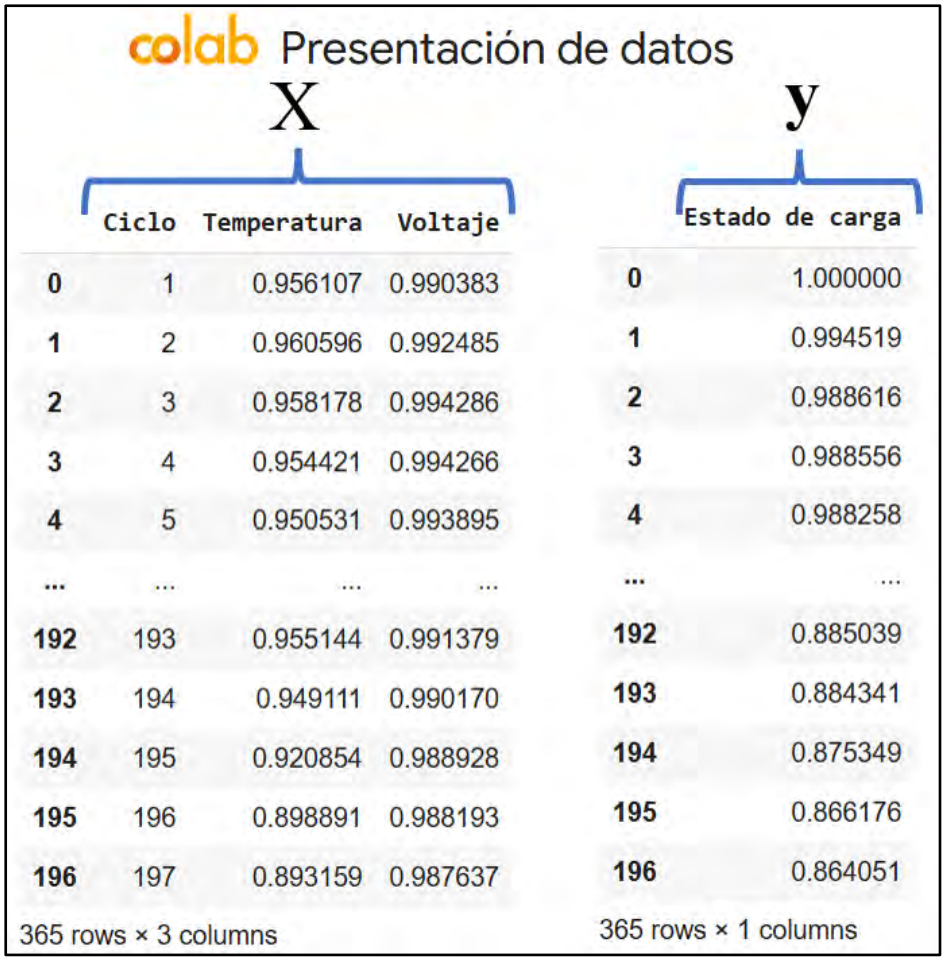
Primero: Todo el conjunto de datos se divide en datos de entrenamiento, datos de prueba y datos de validación. Los datos de entrenamiento para las baterías de Oxford están compuestos por las baterías C1, C2 y C5 y para las baterías de la NASA, son B005 y B036.

Los datos de prueba para las baterías de Oxford están constituidos por C3 y C6, mientras que para las baterías de la NASA se considera únicamente los datos de la batería B007 como datos de prueba (aquí se aclara que las etiquetas que inician con la letra ‘C’, corresponde a las baterías del repositorio de Oxford y las etiquetas que inician con la letra ‘B’, corresponde a las baterías del repositorio de la NASA).

La última partición son los datos de validación, los cuales están constituidos por datos de las baterías C4, C7 y C8 para las baterías de Oxford y los datos de validación para las baterías se la NASA se considera a B006 y B018.

Una muestra del marco de datos elaborado precisamente para que ingresen dentro del algoritmo en el respectivo proceso de entrenamiento se muestra en la Figura 3.12, se muestra la variable de entrada denotado como ‘X’, compuesta por el conjunto de datos conformado por los datos de número de ciclo, temperatura y voltaje y la variable de salida ‘y’, formada únicamente por el estado de carga de la batería, la cual se encuentra normalizada.

Figura 3.12. Estructura de datos.



Nota. Elaboración propia.

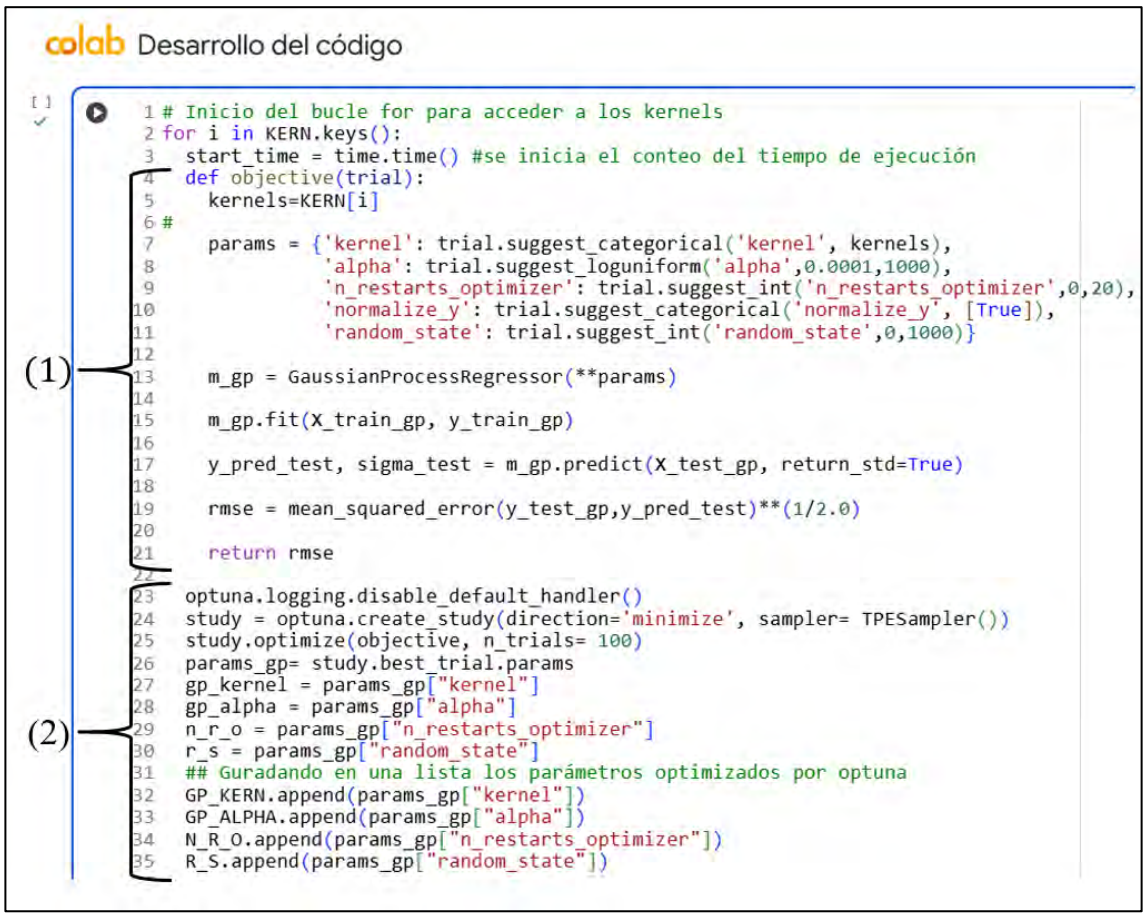
Segundo: En la sección 1 de la Figura 3. **13**, se muestra el código de la función objetivo para la optimización (línea de código 4), dentro del mismo se involucra a las funciones de covarianza desarrolladas en la Tabla 2.1 (línea de código 5) y los hiperparámetros a optimizar (línea de código 7-11). Los hiperparámetros que se utilizan en los Procesos Gaussianos son, 'alpha', el cual es un hiperparámetro que regula la cantidad de ruido que genera el modelo de regresión (en este caso el modelo de regresión de proceso gaussiano), su valor está comprendido en el intervalo entre 0.0001 a 1000 en escala logarítmica. El hiperparámetro 'n_restarts_optimizer', es el número de reinicios que debe realizar el optimizador internamente, esto para evitar que los valores alcancen los mínimos locales, que están entre 0 y 20. El hiperparámetro 'normalize_y', el cual indica la normalización de la variable objetivo, en este caso es valor permanece verdadero ('True'), esto significa que para todos los casos se llevará a cabo la normalización. El hiperparámetro

‘random_state’, indica la asignación aleatoria de valores iniciales en la optimización, esto se realiza en el código interno del optimizador.

Continuando con la descripción del código de la sección 1, de la Figura 3. **13**, se establece la regresión del proceso Gaussiano con los parámetros establecidos previamente (esto en la línea de código 13), en seguida se realiza la regresión y la obtención del modelo de regresión tentativo con los datos de entrenamiento (en la línea de código 15), obtenido del modelo de regresión se procede a evaluar el modelo de regresión, haciendo predicciones y calculando el error cuadrático medio (RMSE) (como se muestra en las líneas de código 17 y 19). El RMSE, se utiliza como indicador de eficiencia, el cual se utiliza en la sección 2 del código de la Figura 3. **13**.

Tercero: El objetivo de la optimización es minimizar el RMSE. Para esto en la línea de código 24 de la sección 2 de la Figura 3. **13**, se declara el objeto ‘study’, el cual cumple con la función de minimizar la muestra asignada, a continuación, se acude a la instancia ‘optimize’ para iniciar el proceso de optimización, asignando como argumento la función ‘objective’ (definida en la línea de código 4 de la sección 1 de la Figura 3. **13**), con una ventana de 100 intentos para encontrar los valores de los hiperparámetros que minimizan el RMSE (línea de código 25). El proceso para encontrar un modelo óptimo, consiste en realizar de forma iterada múltiples regresiones y pruebas hasta encontrar los valores de hiperparámetros que minimicen el RMSE, luego los hiperparámetros optimizados se almacenan en las listas, ‘GP_KERN’, ‘GP_ALPHA’, ‘N_R_O’ y ‘R_S’, este almacenamiento se lleva a cabo recurriendo al método ‘append’, esto quiere decir que cualquier lista seguida de la instancia ‘append’ hace que el valor numérico se agregue a la lista, esto se observa en líneas de código 26-35 de la Figura 3. **13**.

Figura 3. 13. Desarrollo del código de entrenamiento y optimización.

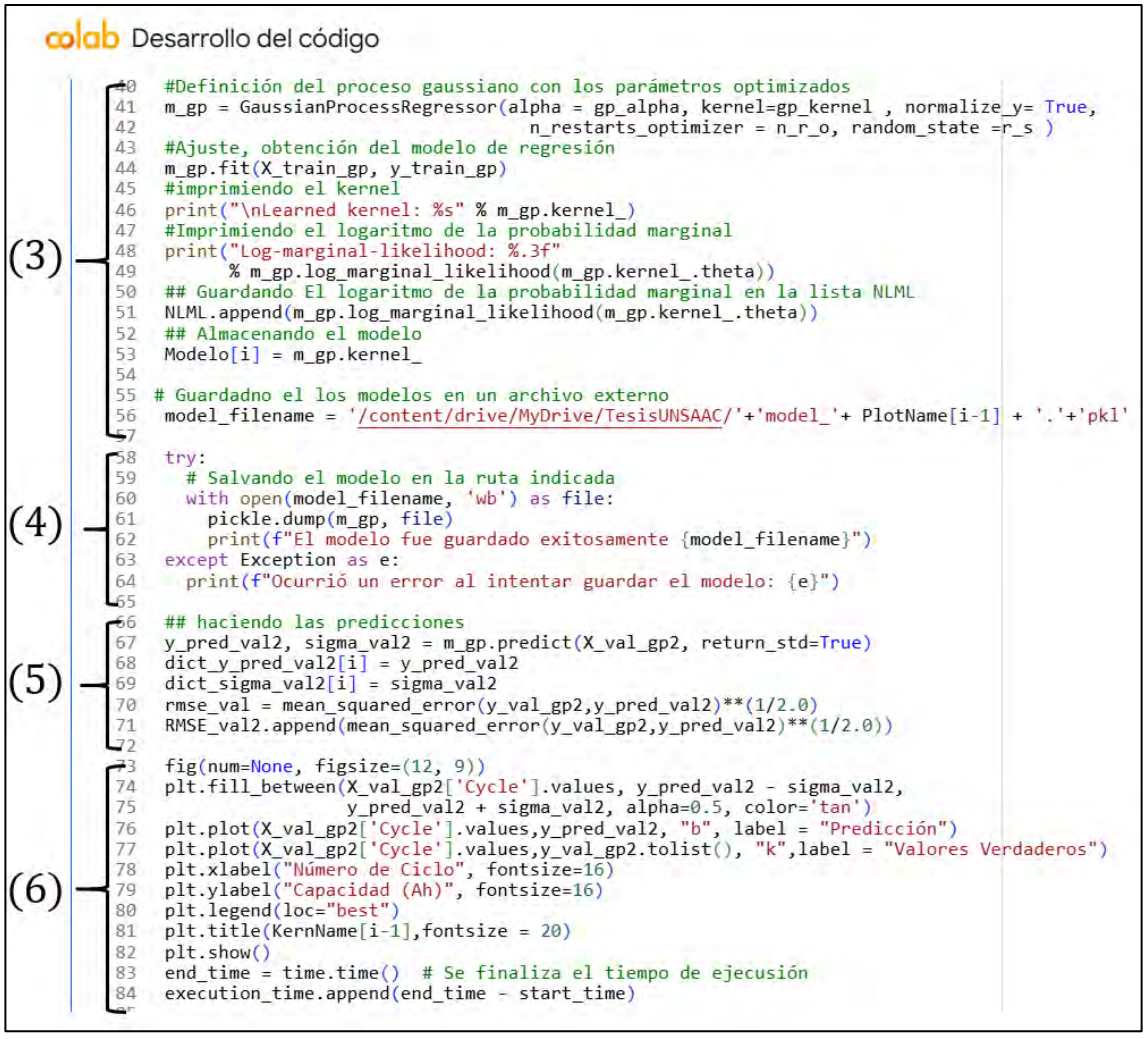



```
# Inicio del bucle for para acceder a los kernels
for i in KERN.keys():
  start_time = time.time() #se inicia el conteo del tiempo de ejecución
  def objective(trial):
    kernels=KERN[i]
#
    params = {'kernel': trial.suggest_categorical('kernel', kernels),
              'alpha': trial.suggest_loguniform('alpha',0.0001,1000),
              'n_restarts_optimizer': trial.suggest_int('n_restarts_optimizer',0,20),
              'normalize_y': trial.suggest_categorical('normalize_y', [True]),
              'random_state': trial.suggest_int('random_state',0,1000)}

    m_gp = GaussianProcessRegressor(**params)

    m_gp.fit(X_train_gp, y_train_gp)

    y_pred_test, sigma_test = m_gp.predict(X_test_gp, return_std=True)

    rmse = mean_squared_error(y_test_gp,y_pred_test)**(1/2.0)

    return rmse

  optuna.logging.disable_default_handler()
  study = optuna.create_study(direction='minimize', sampler= TPESampler())
  study.optimize(objective, n_trials= 100)
  params_gp= study.best_trial.params
  gp_kernel = params_gp["kernel"]
  gp_alpha = params_gp["alpha"]
  n_r_o = params_gp["n_restarts_optimizer"]
  r_s = params_gp["random_state"]
  ## Guradando en una lista los parámetros optimizados por optuna
  GP_KERN.append(params_gp["kernel"])
  GP_ALPHA.append(params_gp["alpha"])
  N_R_O.append(params_gp["n_restarts_optimizer"])
  R_S.append(params_gp["random_state"])
```

Nota. Elaboración propia.

Cuarto: Con los hiperparámetros optimizados, se procede a definir el modelo de proceso gaussiano, esto se muestra en la línea de código 43, de la sección 3 de la Figura 3.14 y se realiza el ajuste del modelo con los datos de entrenamiento (línea de código 44). Con el modelo ya ajustado a los datos se calcula el logaritmo negativo de la probabilidad marginal (NLML) (línea de código 49) y se adjunta el resultado del NLML, la respectiva lista, por otro lado, el modelo obtenido se agrega al diccionario definido previamente con la notación de 'Modelo' (línea de código 53). Luego en la sección 4 de la Figura 3.14, el diccionario 'Modelo' se exporta como un archivo ejecutable en formato '.pkl' en un directorio del Google Drive, para ser utilizado posteriormente.

Figura 3.14. Desarrollo del código de los modelos de regresión y validación de los modelos.



Nota. Elaboración propia.

Quinto: En esta etapa se procede a utilizar los datos de validación para comprobar la verdadera efectividad del modelo, esto corresponde a la sección 5 de la Figura 3.14. Se procede a utilizar los datos de validación para hacer predicciones con el modelo ya optimizado (línea de código 67), los resultados de las predicciones y la desviación estándar se almacenan en sus respectivos diccionarios (línea de código 68 y 69) y el RMSE se guarda en la lista 'RMSE_val2' (línea de código 71). El RMSE es útil para comparar la eficiencia de predicción del modelo de regresión de procesos gaussianos con

otros modelos de regresión pertenecientes a los algoritmos de aprendizaje automático como RNA, SVR, etc.

Sexto: Aquí se realiza un cuadro comparativo entre los valores verdaderos (o reales) (que son los datos de validación) y lo que predice el modelo, esto a través del gráfico de la curva de la evolución del estado de carga de los valores reales y los valores del estado de carga obtenidos en la predicción, el código se muestra en la sección 6 de la Figura 3.14.

Figura 3.15. Código para guardar los resultados en un archivo externo (en el Google drive).

colab Salvar los resultados

```
[ ]
#Aquí se Guarda Lo datos en un archivo pkl.
import os
# Se define la ruta del directorio donde se desea guardar
save_directory = '/content/drive/MyDrive/TesisUNSAAC'
# Combine the data into a dictionary for easier saving
data_to_save = {
    'GP_KERN': GP_KERN,
    'GP_ALPHA': GP_ALPHA,
    'N_R_O': N_R_O,
    'R_S': R_S,
    'RMSE_train': RMSE_train,
    'RMSE_test': RMSE_test,
    'RMSE_val': RMSE_val,
    'RMSE_val2': RMSE_val2,
    'NLML': NLML,
    'execution_time': execution_time,
    'Modelo': Modelo,
    'dict_y_pred_val': dict_y_pred_val,
    'dict_sigma_val': dict_sigma_val,
    'dict_y_pred_val2': dict_y_pred_val2,
    'dict_sigma_val2': dict_sigma_val2
}

# Se define el nombre del archivo pickle
save_filename = 'modelo_resultados.pkl'
save_path = os.path.join(save_directory, save_filename)

# Se guarda los datos en el archivo pickle
with open(save_path, 'wb') as f:
    pickle.dump(data_to_save, f)

print(f"Datos guardados exitosamente en: {save_path}")
```

Nota. Elaboración propia.

Los valores optimizados de los hiperparámetros, las predicciones, las desviaciones estándar y los valores de RMSE, se adjuntan en un solo archivo en formato ‘pkl’ y se guarda en un directorio ubicado en el Google Drive, esto se muestra en la Figura 3.15.

CAPÍTULO IV: RESULTADOS Y DISCUSIÓN

4.1. Resultados.

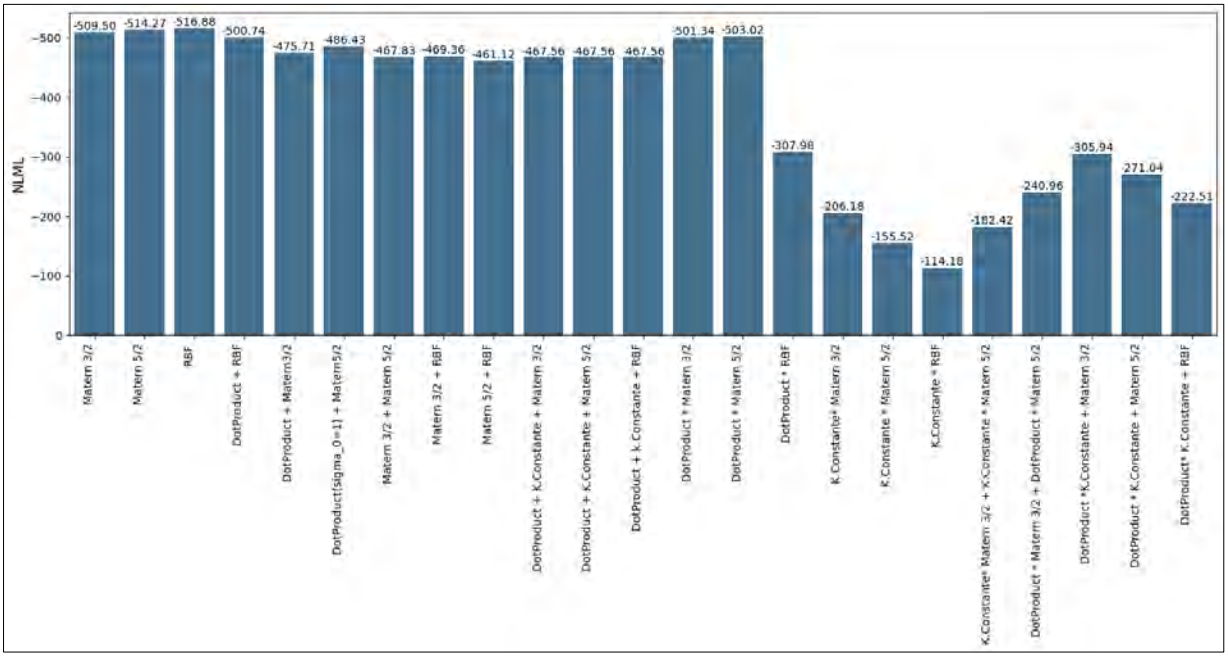
En esta sección se muestra los resultados de las predicciones del estado de carga de las baterías con los modelos obtenidos a partir del algoritmo de procesos gaussianos, usando datos de baterías del repositorio de Oxford. El entrenamiento del algoritmo se llevó a cabo con el algoritmo que contienen las funciones de covarianza desarrollados en la sección 2.1.4 y que se muestran en la Tabla 2.1 y cuyo código en Python se muestra en el algoritmo descrito en la Figura 3. 13.

4.1.1. Resultado de los modelos de regresión usando datos de Oxford.

A continuación, se presenta una descripción completa de cómo se obtiene la predicción con modelos de Procesos Gaussianos con datos de validación.

En primera instancia la elección del modelo de regresión de Proceso Gaussiano más eficiente se realiza a través del indicador NLML.

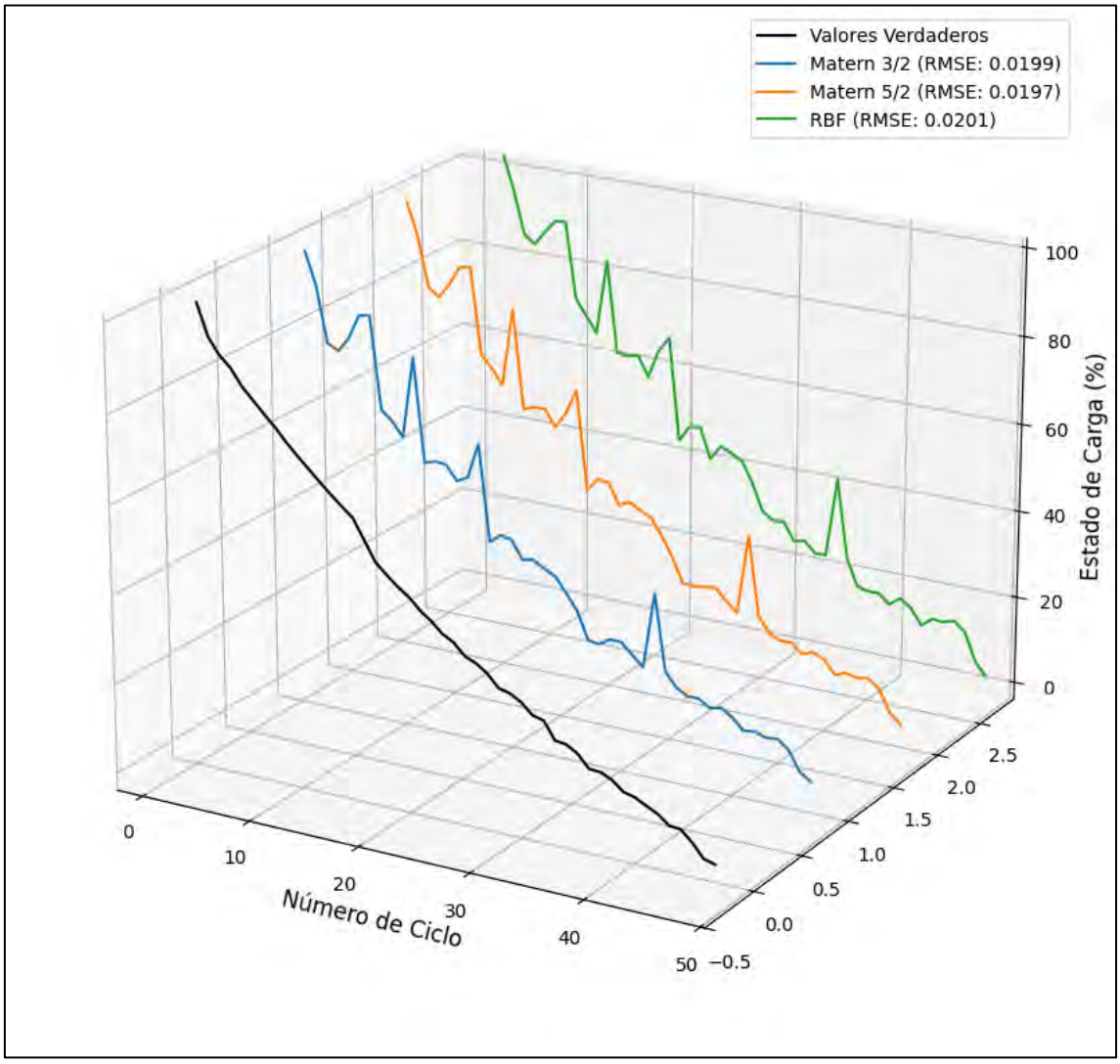
Figura 4.1. Indicador NLML en predicciones con datos de validación provenientes del repositorio del Oxford.



Nota. Elaboración propia.

Para procesos gaussianos el modelo más eficiente será con el que se obtiene el menor valor posible del indicador NLML, de acuerdo al diagrama de barras que se muestra en la Figura 4.1, en este caso el modelo RBF es el más adecuado para predecir el estado de carga en este tipo de baterías, pero la elección del mejor modelo se tiene que llevar cabo tomando en cuenta primero el valor de RMSE.

Figura 4.2. Predicciones con la batería 'Celda 4' (C4).



Nota. Elaboración propia.

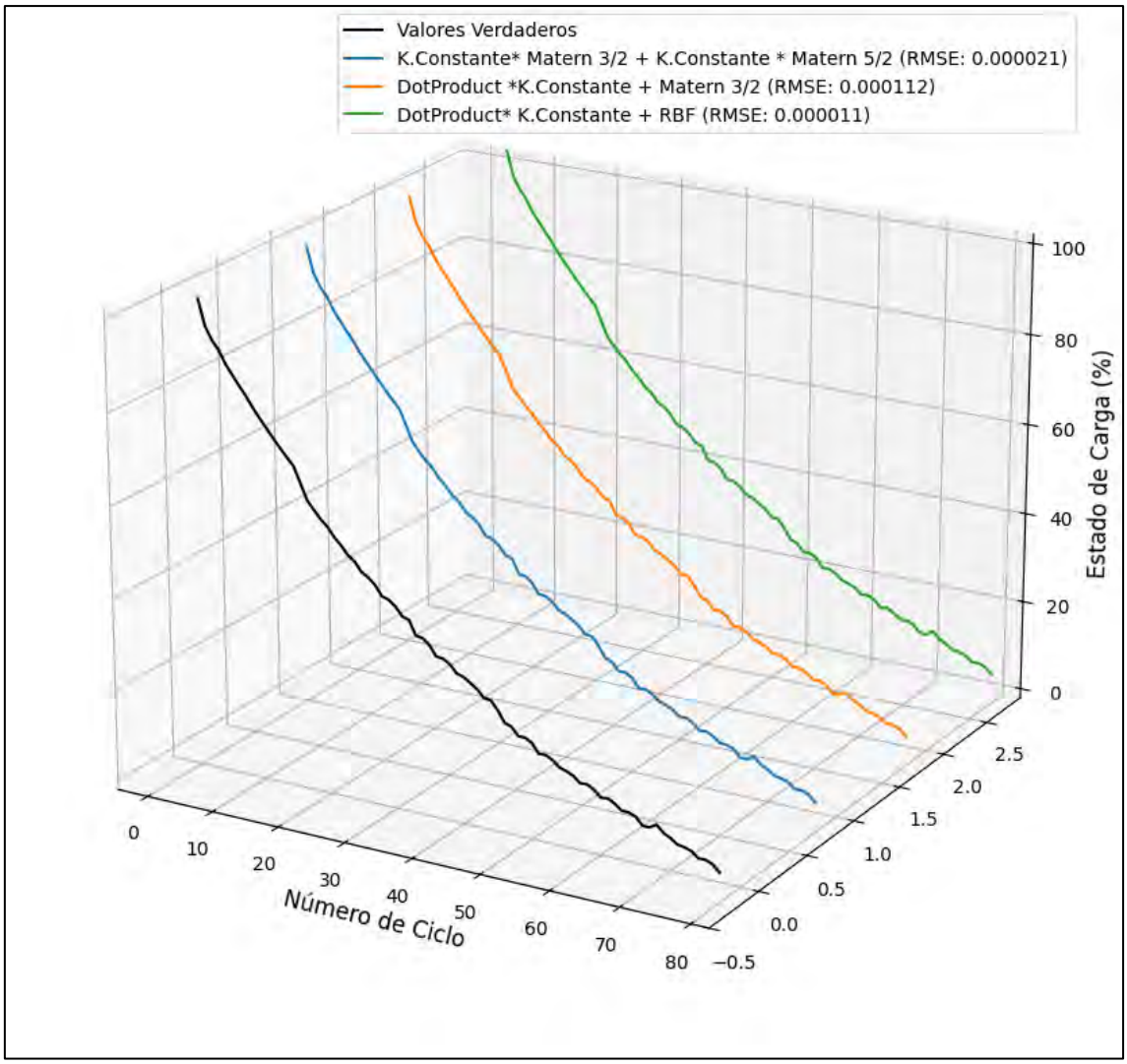
De acuerdo a la Figura 4.2, se tienen tres funciones posibles para el modelo, los cuales son: Matern 3/2, Matern 5/2 y RBF, la elección de estas tres funciones, es debido a que las 23 funciones propuestas, obtuvieron los valores mínimos de RMSE que son 0.0199, 0.0197 y 0.0201, respectivamente, las mismas que obtuvieron valores de NLML de -509.50, -514.27 y -516.88, respectivamente.

$$k = exp\left(-\frac{r^2}{2}\right) \quad (4.\ 1)$$

De las predicciones mostradas en la Figura 4.2, se elige el modelo cuya función de covarianza corresponde al menor valor de NLML, en este caso corresponde a la función de base radial como se muestra en la ecuación (4.1). Con esto se obtiene un modelo con RMSE de 0.020 y un valor de NLML de -516.88.

Ahora se lleva a cabo las predicciones con los datos de las baterías etiquetadas como 'Celda 7' y 'Celda 8'. Los valores de RMSE mínimos se obtuvo para los modelos cuyas funciones de covarianza son las siguientes: K.Constante*Matern 3/2 + K.Constante*Matern 5/2, DotProduct*K.Constante + Matern 3/2 y DotProduct*K.Constante + RBF, para los que se obtuvo valores de RMSE de 0.000021, 0.0000112 y 0.000011 y en cuanto al NLML, según la Figura 4.2, se obtuvieron valores de : -182.42, -305.94 y -222.51, respectivamente. La elección final, da resultado a un modelo con NLML = -305.94 y un RMSE de 0.0000112, esto corresponde a la función DotProduct*K.Constante + Matern 3/2.

Figura 4.3. Predicciones con la batería 'Celda 7' (C7).



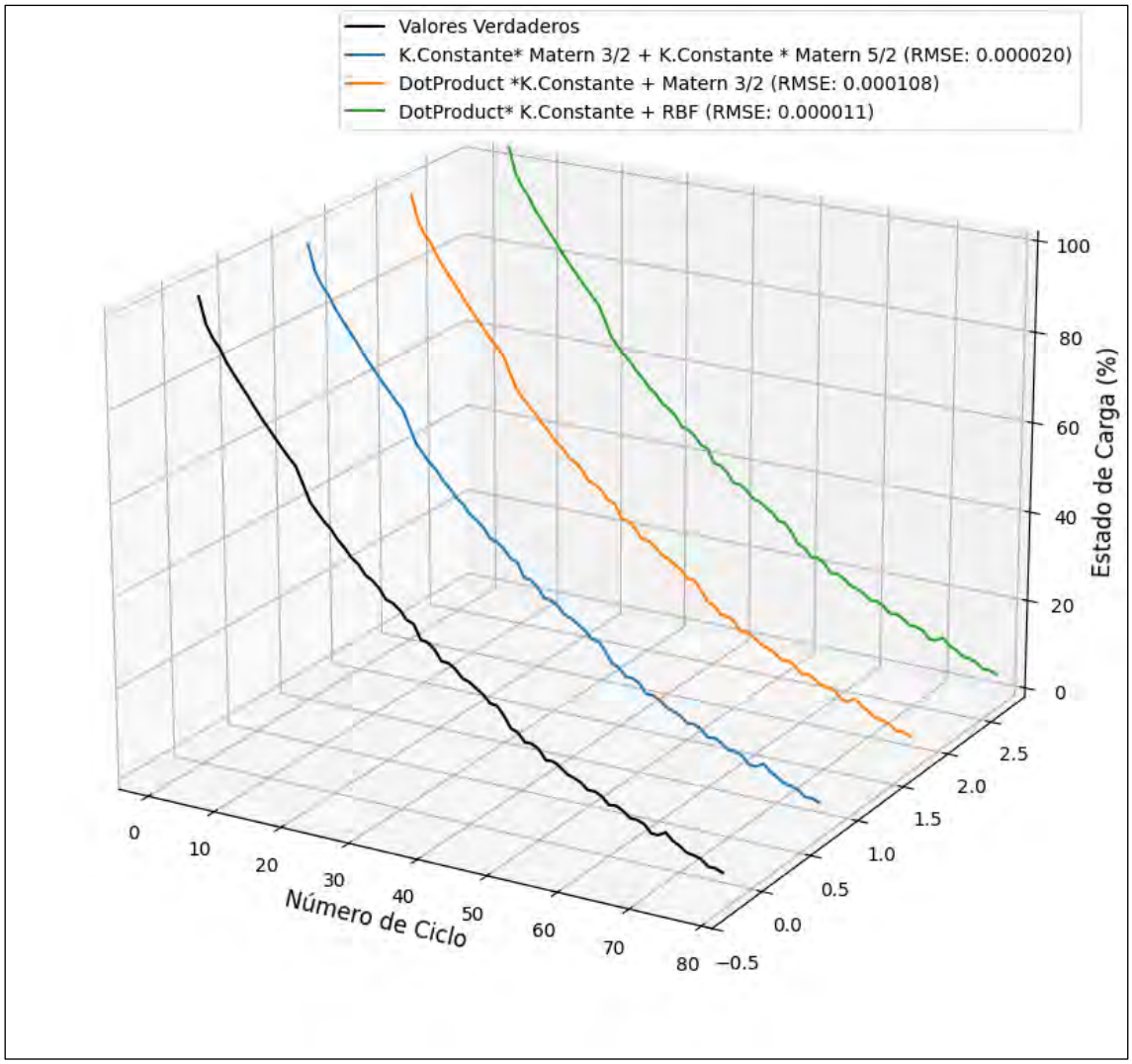
Nota. Elaboración propia.

Por consiguiente, el modelo elegido que representa a las curvas de predicción de la Figura 4.3, se describe en la ecuación 4.2.

$$k = \left(1 + x_i \cdot x_j\right) \cdot \left(1 + \sqrt{3}r\right) exp\left(-\sqrt{3}r\right) \qquad (4.2)$$

Un análisis similar se realizó para la Celda 8. Se observó que el comportamiento del estado de carga es similar al de la Celda 7. Las predicciones del estado de carga para la Celda 8, con los mejores modelos se muestran en la Figura 4.4, de acuerdo a los resultados de RMSE se obtuvo un modelo cuya función de covarianza es de la forma DotProduct*K.Constante + Matern 3/2, con RMSE = 0.000108 y NLML = -305.4, el cual se describe con la ecuación 4.2.

Figura 4.4. Predicciones con la batería 'Celda 8' (C8).



Nota. Elaboración propia.

Las curvas que describen el estado de carga y que se muestran en la Figura 4.2, Figura 4.3 y Figura 4.4 y que se usaron para obtener los modelos de procesos gaussianos en esta sección, tienen un buen comportamiento, en el sentido de que no presenta cambios abruptos y que pueden ser descritas con gran aproximación funciones simples como la exponencial. El desafío se incrementa cuando se dispone obtener modelos que describan el comportamiento de curvas como se muestra en la Figura 3.6, ya que estas, presentan cambios abruptos, cuyo comportamiento no podría se descrito por una sola función de covarianza, por consiguiente, se llevará a cabo la evaluación de la eficiencia en la predicción del estado de carga de los modelos de procesos gaussianos y se realizará una comparación con la eficiencia de modelos generados a partir de algoritmos de redes neuronales, regresión de soporte vectorial y bosques aleatorios, todo esto con datos del repositorio de la NASA.

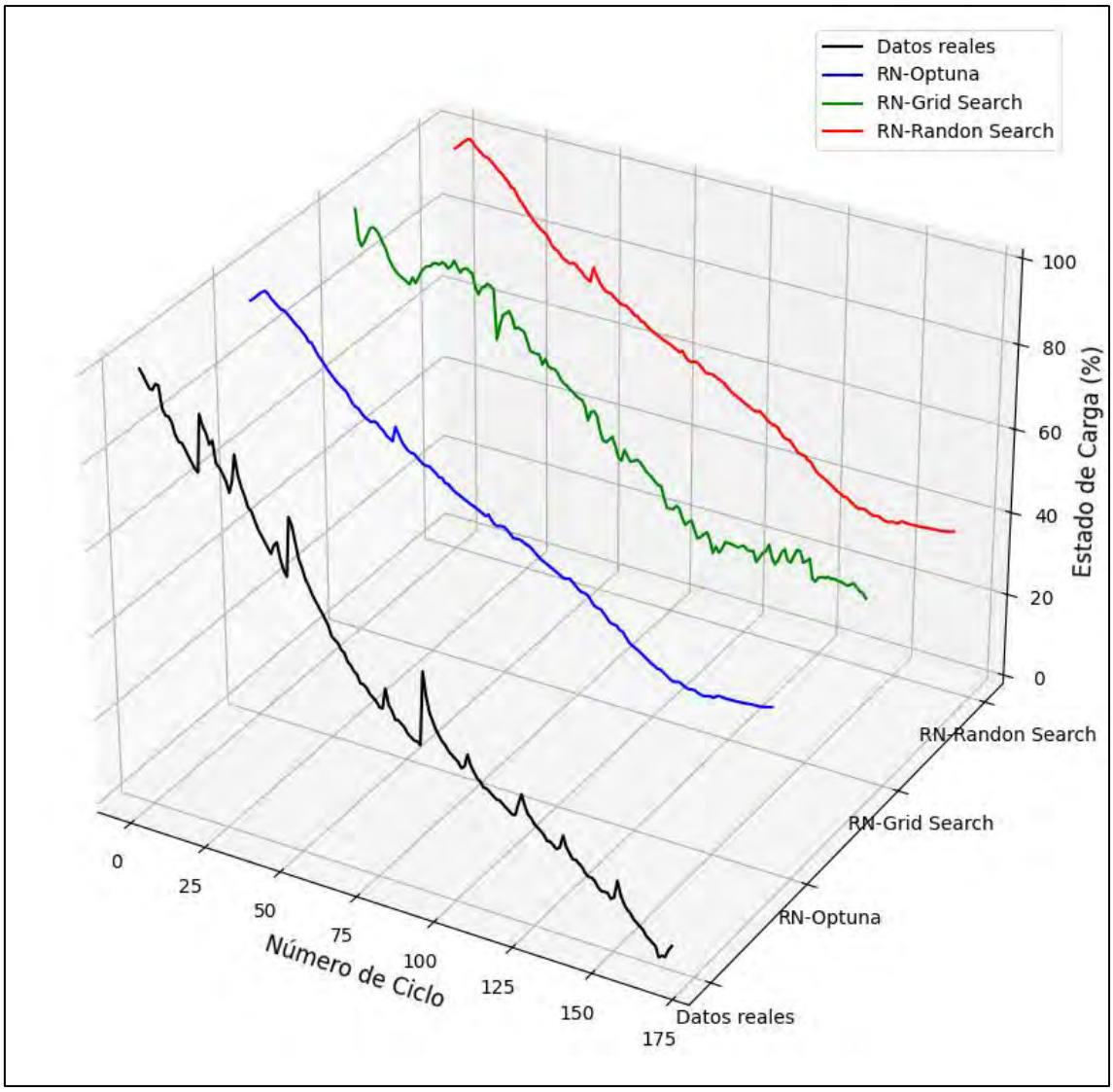
En capítulos anteriores se mencionó (según la literatura) que el optimizador recomendado para la optimización de hiperparámetros en procesos gaussianos es el Optuna, adicionalmente a este optimizador existen muchos otros, siendo los más populares el optimizador Grid Search y el optimizador Randon Search, por consiguiente, queda por determinar el optimizador que permite mayor eficiencia en las predicciones con modelos generados a partir de RNA, SVR y Randon Forest.

En las siguientes tres secciones se lleva a cabo el entrenamiento y la obtención de modelos de regresión con algoritmos de RNA, SVR y Randon Forest, usando los optimizadores, Optuna, Grid Search y Randon Search, posteriormente se evalúa la eficiencia a través del RMSE y se elige el modelo con el optimizador que hace posible la obtención de predicciones más eficientes, toda esta evaluación se realiza ya en la etapa de validación, esto usando datos de validación del repositorio de la NASA, con los datos correspondientes a las baterías B006 y B018.

4.1.2. Predicciones con algoritmos de redes neuronales artificiales

A continuación, en la Figura 4.5, se muestra un cuadro comparativo de las predicciones del estado de carga con algoritmos de RNA, con diferentes optimizadores de hiperparámetros, esto con datos de validación de las baterías B006.

Figura 4.5. Predicciones con RNA con datos de validación de la batería B006.

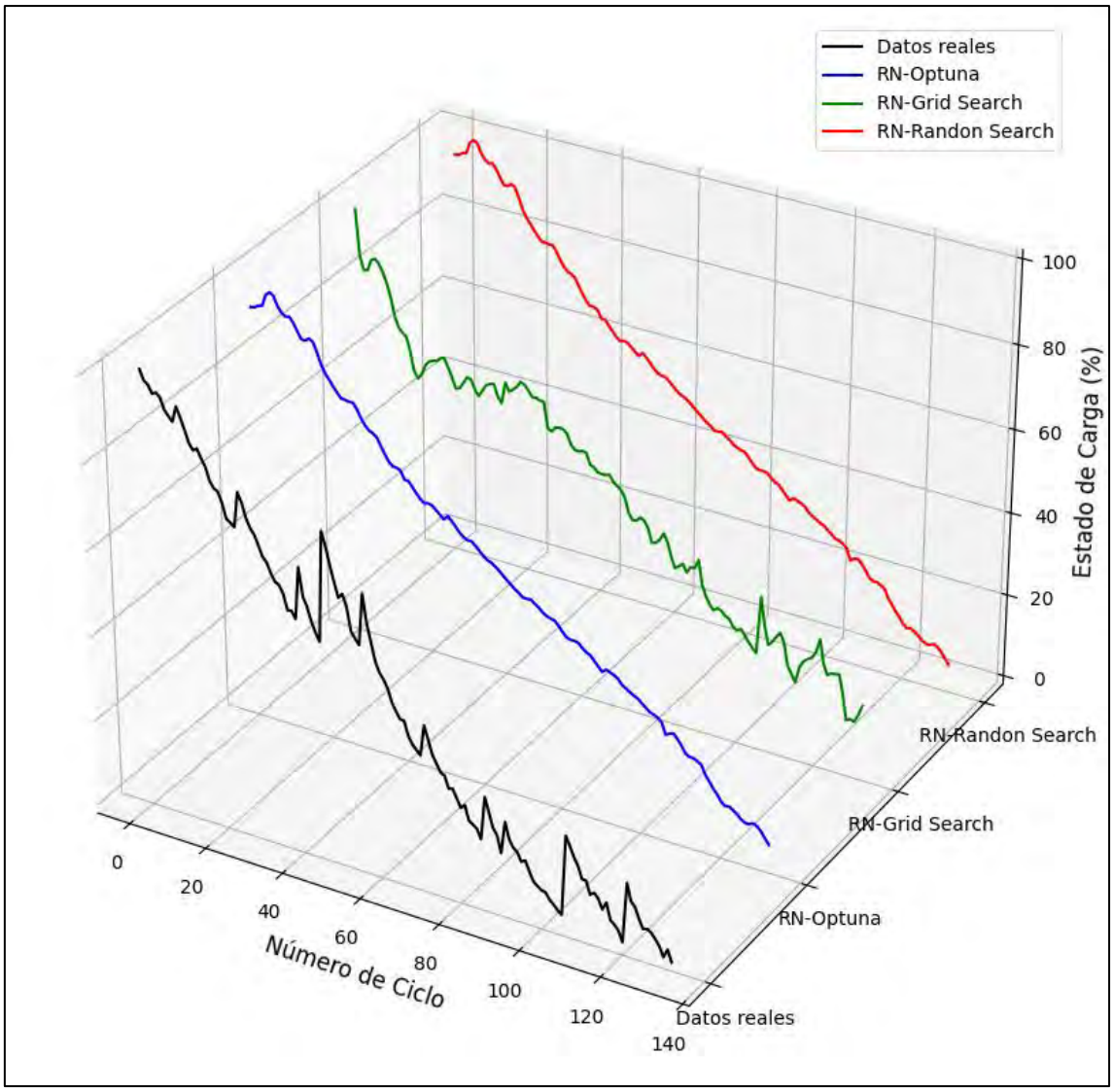


Nota. Elaboración propia.

De acuerdo al cuadro comparativo que se muestra en la Figura 4.5, las predicciones del estado de carga con redes neuronales artificiales usando el optimizador de Grid Search (RN-Grid Search) parece emular el comportamiento no lineal de los datos reales (los datos reales es la curva de color negro), pero esta se aleja de la tendencia media a diferencia de las predicciones del modelo con RNA usando Optuna como optimizador, que de cierta forma se acerca a la tendencia media de los datos reales.

Ahora se muestra la evaluación del modelo de redes neuronales con los optimizadores antes mencionados para hacer predicciones con datos de la batería B018. El cuadro comparativo en 3 dimensiones se muestra en la Figura 4.6, donde también el modelo de predicción de RNA obtenido con el optimizador Optuna, emula la tendencia media de los datos de estado de carga con datos reales.

Figura 4.6. Predicciones con RNA con datos de validación de la batería B018.



Nota. Elaboración propia.

En referencia a la predicción del estado de carga con el modelo de RNA usando Grid Search (RN-Grid Search) de la Figura 4.6, (para B018), esta intenta emular el comportamiento no lineal de la curva de estado de carga original, pero en algunos tramos sobre estima el estado de carga, por esto el modelo de RNA con el optimizador Optuna es el más aceptable y representativo para este algoritmo.

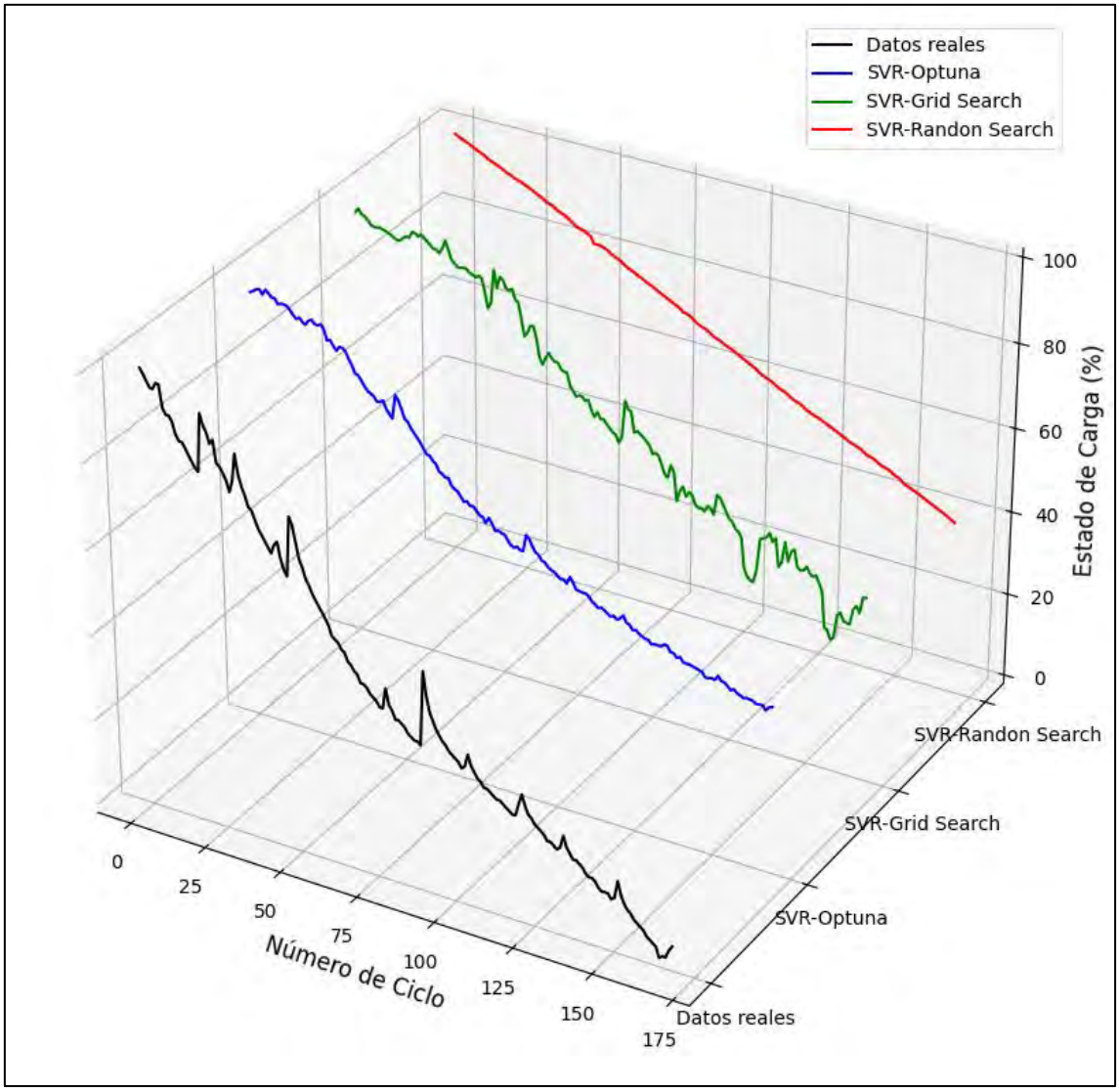
4.1.3. Predicciones con algoritmos de regresión de soporte vectorial

Ahora se hace uso de los algoritmos de SVR para generar modelos de regresión para hacer predicciones del estado de carga con datos de validación correspondiente a las baterías B006, que al igual que se hizo con RNA, aquí se genera modelos de regresión usando los optimizadores Optuna, Grid Search y Randon Search.

La Figura 4.7, muestra un cuadro comparativo en 3d de las predicciones con el modelo de regresión SVR usando los tres diferentes optimizadores de parámetros e hiperparámetros.

A simple vista las predicciones con SVR-Optuna parece predecir la tendencia de forma aceptable. Mientras que SVR- Randon Search se aleja demasiado del intento de emular el comportamiento no lineal de la curva que representa los datos reales del estado de carga, y SVR-Grid Search intenta emular el comportamiento no lineal, pero sobre estima en varios tramos.

Figura 4.7. Predicciones del estado de carga con SVR usando diferentes optimizadores (con B006).

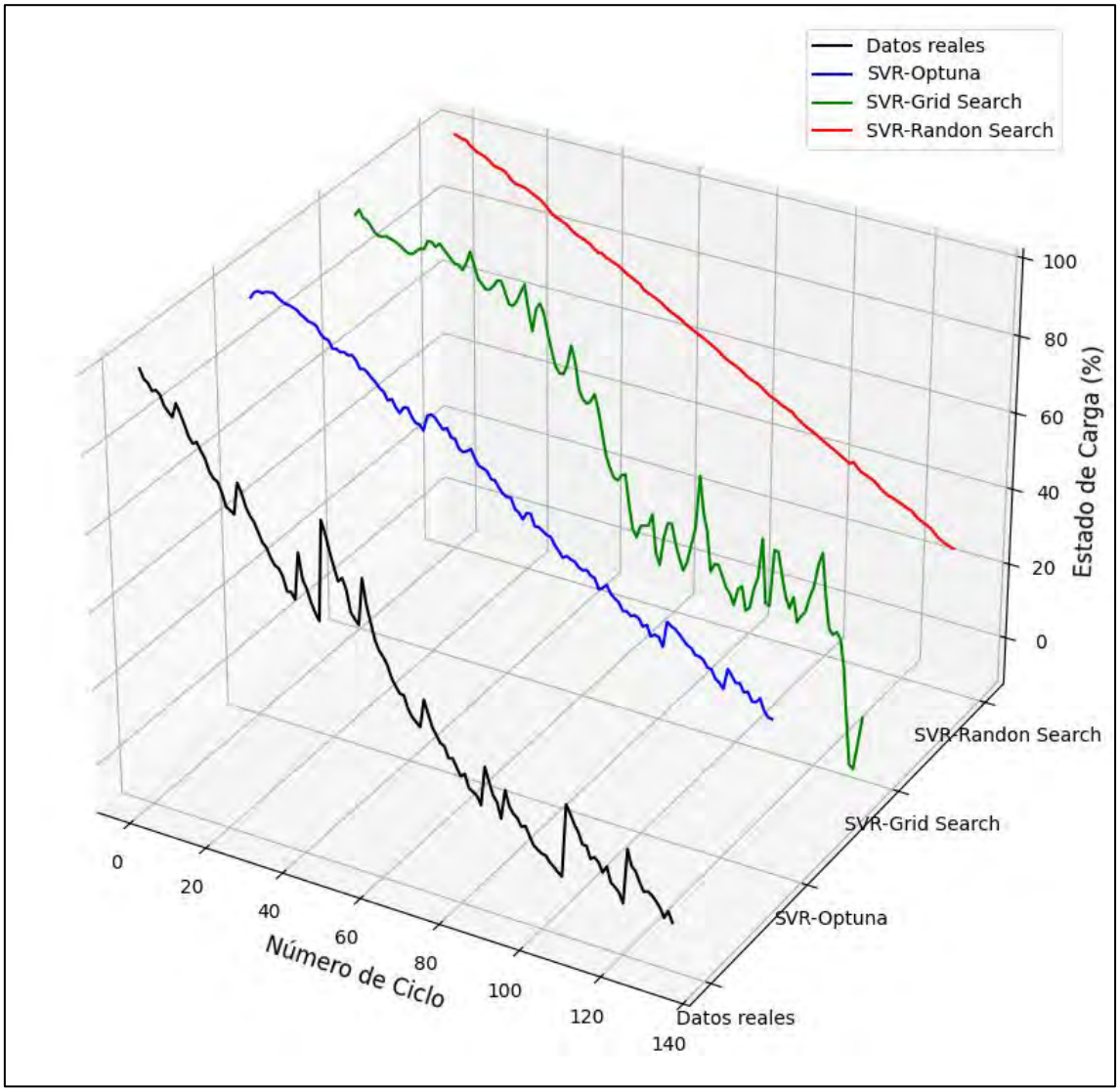


Nota. Elaboración propia.

Ahora se continua con el estudio de la eficiencia en la predicción con modelos SVR con diferentes optimizadores, usando datos de la batería B018, de forma similar a

los resultados de la Figura 4.7. Los resultados de la predicción para la validación del modelo SVR que se observa en la Figura 4.8, como un cuadro comparativo, muestran que la curva obtenida con el modelo SVR-Optuna ofrece una mejor similitud a los datos reales de estado de carga, en comparación con las predicciones que se obtiene con los modelos SVR-Grid Search y SVR-Randon Search.

Figura 4.8. Predicciones del estado de carga con SVR usando diferentes optimizadores (con B018).



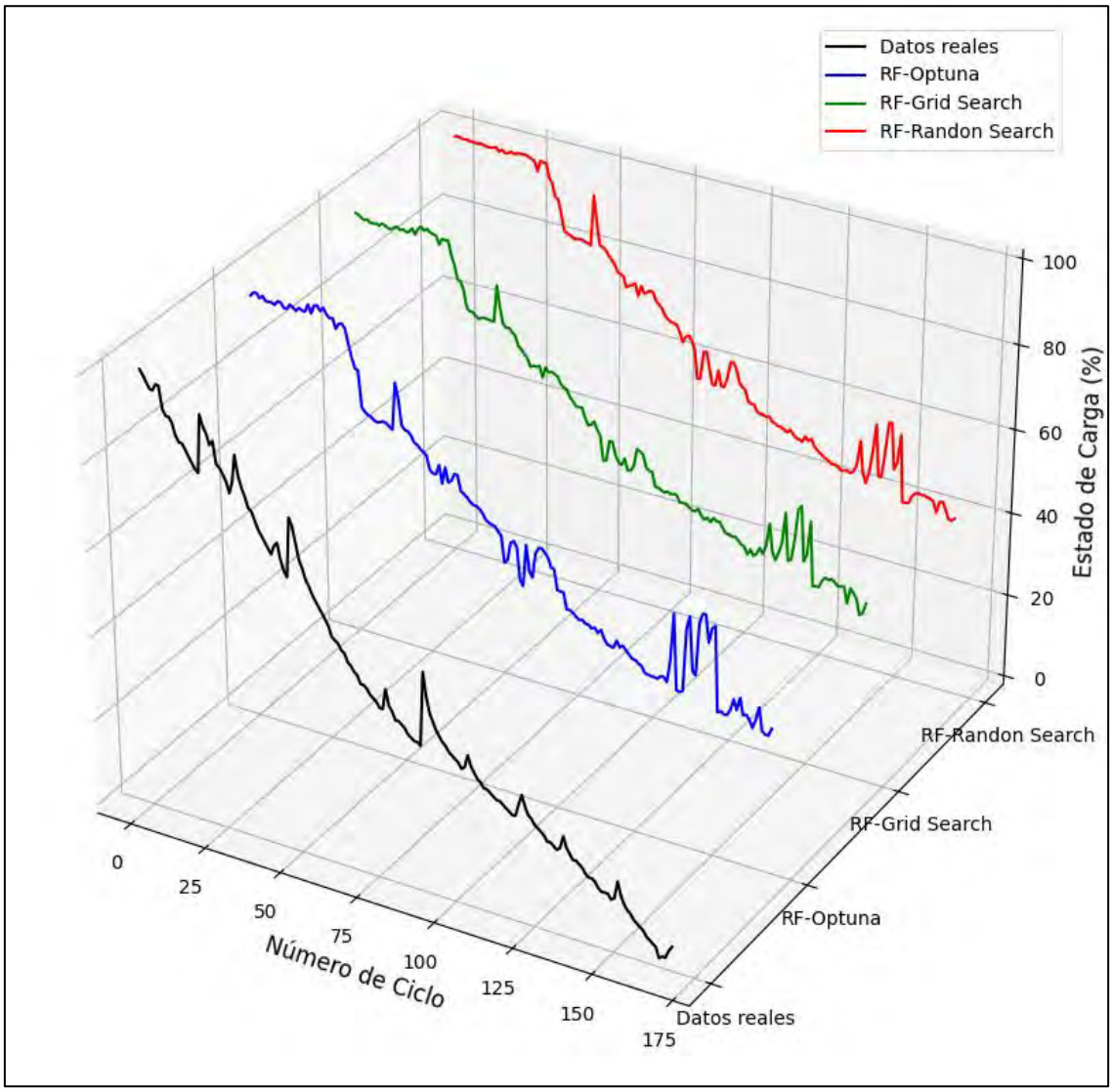
Nota. Elaboración propia.

4.1.4. Predicciones con algoritmos de Bosques aleatorios

El algoritmo de bosque aleatorio fue aplicado con el optimizador Optuna, Grid Search y Randon Search, para optimizar parámetros e hiperparámetros y obtener modelos de regresión. En la Figura 4.9, se observa las predicciones con datos de la batería B006

generadas con los modelos de bosques aleatorios (RF) usando los optimizadores antes mencionados.

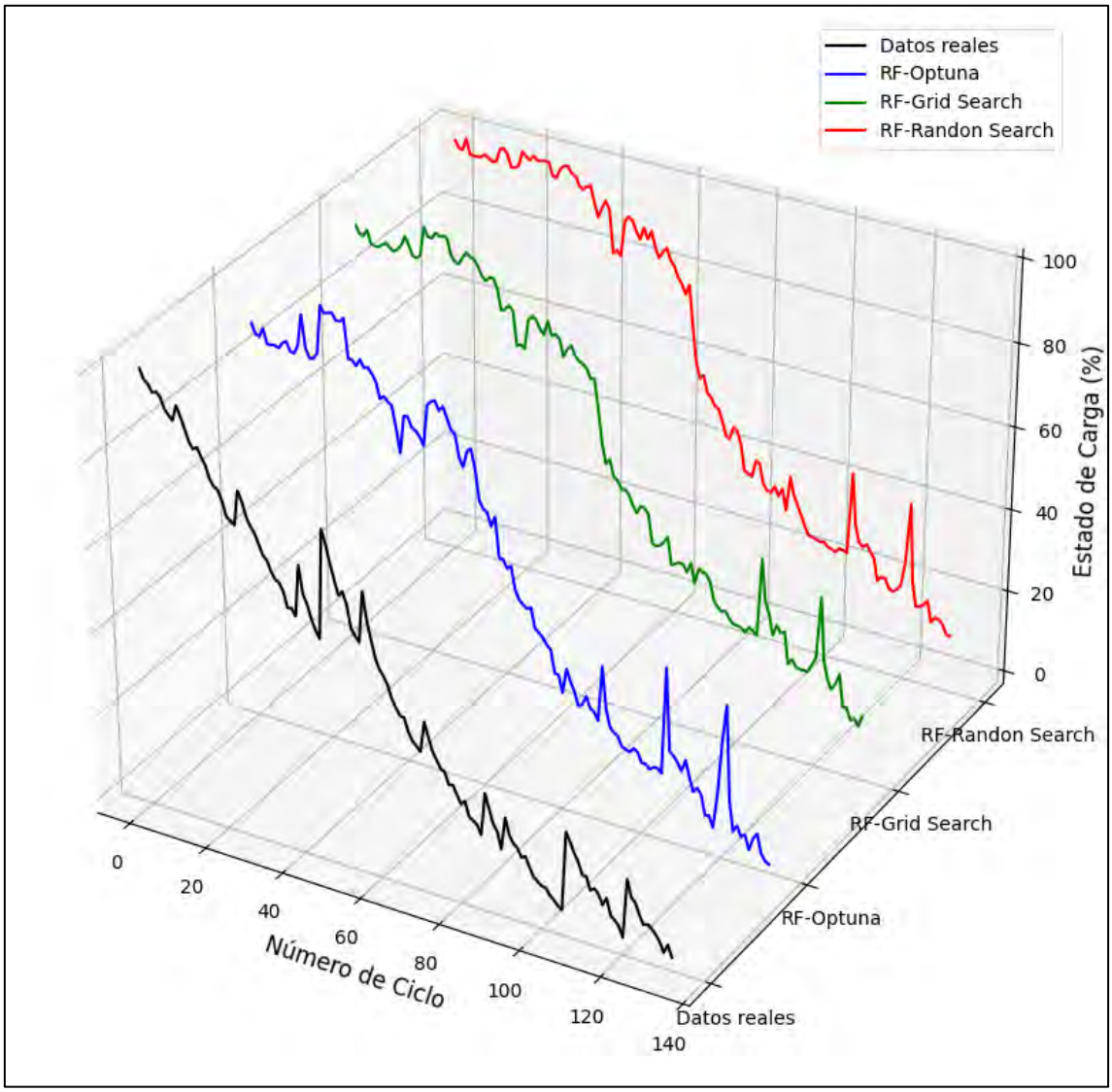
Figura 4.9. Predicciones con modelos de bosques aleatorios obtenidas con diferentes optimizadores (sobre datos de B006).



Nota. Elaboración propia.

Las evaluaciones de la eficiencia en la predicción del estado de carga con los modelos de bosque aleatorio usando los diferentes optimizadores se realizó también con los datos de la batería B018, el cuadro comparativo de las predicciones con los diferentes modelos obtenidos se observa en la Figura 4.10. Las curvas que representan las predicciones con los diferentes modelos tienen cierto parecido y la diferencia en la eficiencia se define al determinar RMSE.

Figura 4.10. Predicciones con modelos de bosques aleatorios obtenidos con diferentes optimizadores (sobre datos de B018).



Nota. Elaboración propia.

Las predicciones de los modelos de regresión con redes neuronales, regresión de soporte vectorial y bosques aleatorios que se presentan en las secciones 4.1.2 , 4.1.3 y 4.1.4, son útiles para mostrar visualmente la variación de las predicciones de los modelos cuando estos fueron desarrollados con diferentes optimizadores de parámetros e hiperparámetros y es necesario cuantificar la eficiencia de las predicciones con cada uno de los modelos obtenidos, esto se consigue a través del cálculo del RMSE, el cual se muestra en la Tabla 4.1 para datos de la batería B006 y la Tabla 4.2 para datos de la batería B018.

Tabla 4.1. Cálculo de la raíz del error cuadrático medio con datos de la batería B006.

Algoritmo	Optimizador		
	Optuna	**Grid Search**	**Randon Search**
Redes Neuronales	0.080636644	0.115384086	0.080636644
SVR	0.09117473	0.107363148	0.123438421
Randon Forest	0.078308435	0.098139225	0.096081134

Tabla 4.2. Cálculo de la raíz del error cuadrático medio con datos de la batería B018.

Algoritmo	Optimizador		
	Optuna	**Grid Search**	**Randon Search**
Redes Neuronales	0.035710592	0.046820519	0.035710592
SVR	0.067054468	0.052706745	0.067724019
Randon Forest	0.03211683	0.047974191	0.045659011

Ahora se tiene que elegir el algoritmo de aprendizaje automático junto con su optimizador, esto a través de la estimación del RMSE haciendo uso de los datos reales y predicciones de valores del estado de carga de las baterías B006 y B018.

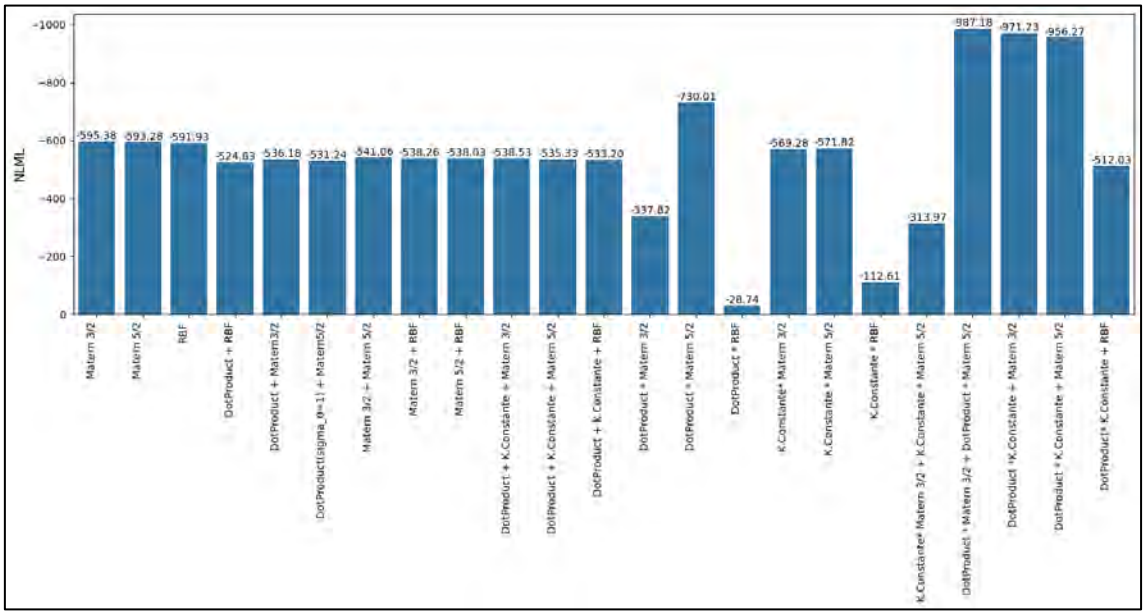
4.1.5. Predicción con algoritmos de Procesos Gaussianos

Para poder llevar a cabo las predicciones con datos de las baterías B006 y B018, se tuvo que realizar previamente la obtención de los modelos de regresión con procesos gaussianos, para esto se utilizó como datos de entrenamiento los datos de las baterías B005 y B036, y como datos de prueba los datos de la batería B007. Todo esto usando los algoritmos que se muestran en la Figura 3. **13** y la Figura 3.14, cuyo funcionamiento se explicó en la sección 3.10.2.5.

Una vez obtenida los modelos de regresión, se realizó las predicciones con los 23 modelos de proceso gaussianos. Como se indicó anteriormente, según la literatura, Optuna es el optimizador más apropiado para optimizar hiperparámetros en procesos gaussianos (Richardson et al., 2017). Para todos los modelos se obtuvo el valor del NLML, estos valores fueron útiles para el modelo que mejor se adapta a los datos de entrenamiento sin incurrir en sobreajuste, recordando también que en modelos de procesos gaussianos la elección del modelo más eficiente se lleva a cabo a través de la

evaluación de los valores del RMSE y los valores del NLML, cuyos resultados se observan en la Figura 4.11.

Figura 4.11. Valores del NLML para todos los modelos para predicciones con las baterías B006 y B018.



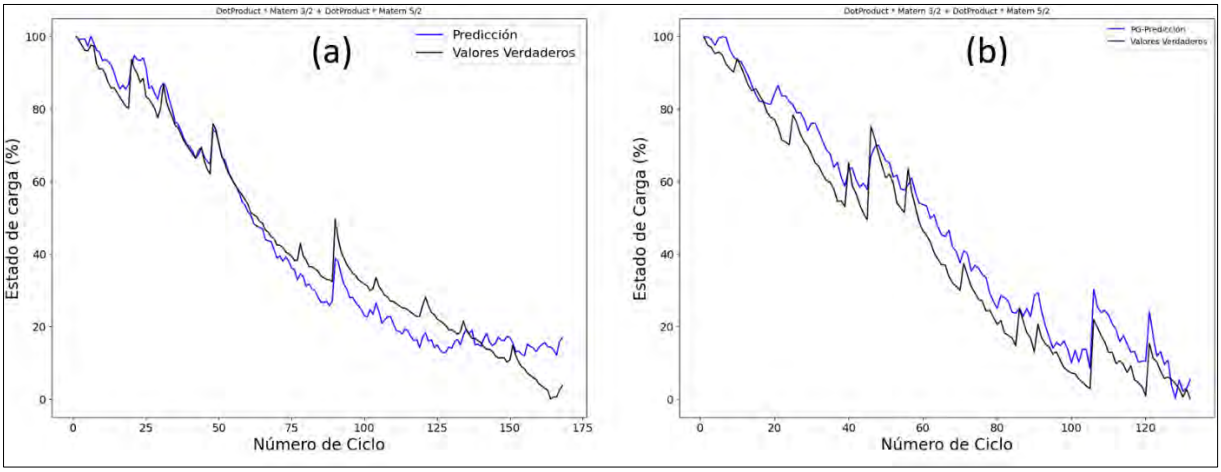
Nota. Elaboración propia.

Según la Figura 4.11, el mejor modelo (o modelo más eficiente) es el modelo de proceso gaussiano con función de covarianza de la forma, DotProduct*Matern 3/2+DotProduct*Matern 5/2 y cuya ecuación matemática es la siguiente:

$$k = (\sigma_0^2 + x_i \cdot x_j) \cdot \left(1 + \frac{\sqrt{3}r}{\ell}\right) exp\left(-\frac{\sqrt{3}r}{\ell}\right) + (\sigma_0^2 + x_i \cdot x_j) \cdot \left(1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2}\right) exp\left(-\frac{\sqrt{5}r}{\ell}\right) \quad (4.3)$$

El valor del indicador NLML, con esta función de covarianza es de -987.18.

Figura 4.12. Predicciones con modelos de procesos gaussianos. (a) Predicciones con datos de la batería B006, (b) Predicción con datos de la batería B018.



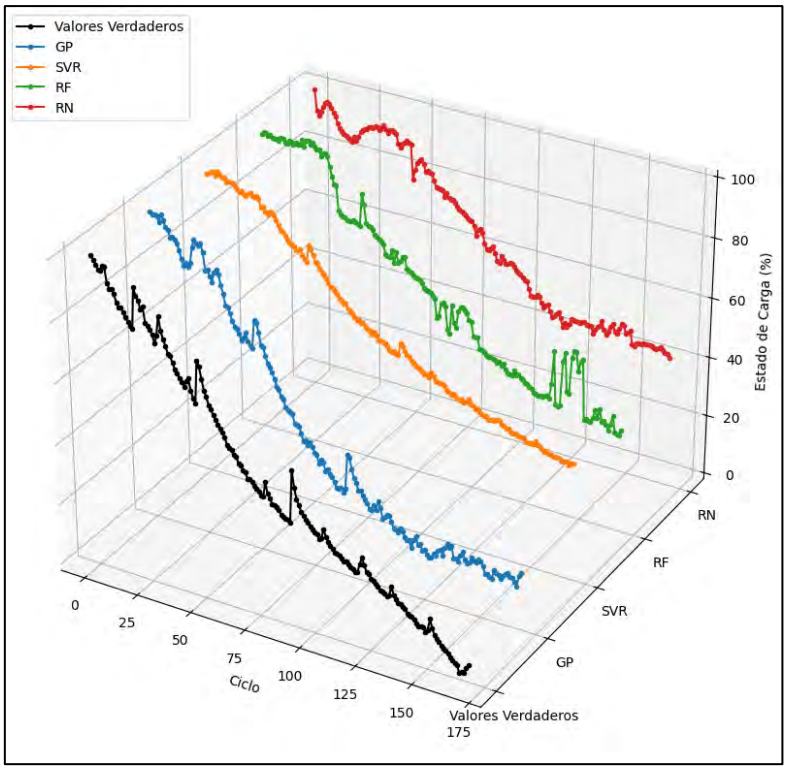
Nota. Elaboración propia.

Las predicciones que se muestran en la Figura 4.12, representan a modelos de regresión más eficientes de los que se obtuvieron valores de RMSE de 0.02857 y 0.0270 para datos de la batería B006 y B018, respectivamente. El modelo que se obtuvo luego de la optimización de parámetros e hiperparámetros es la siguiente ecuación:

$$k = \left((10^5)^2 + x_i \cdot x_j\right) \cdot \left(1 + \frac{\sqrt{3}r}{2.3 * 10^{-4}}\right) \exp\left(-\frac{\sqrt{3}r}{2.3 * 10^{-4}}\right) + \left((0.345)^2 + x_i \cdot x_j\right) \cdot \left(1 + \frac{\sqrt{5}r}{103} + \frac{5r^2}{(103)^2}\right) \exp\left(-\frac{\sqrt{5}r}{103}\right) \quad (4.4)$$

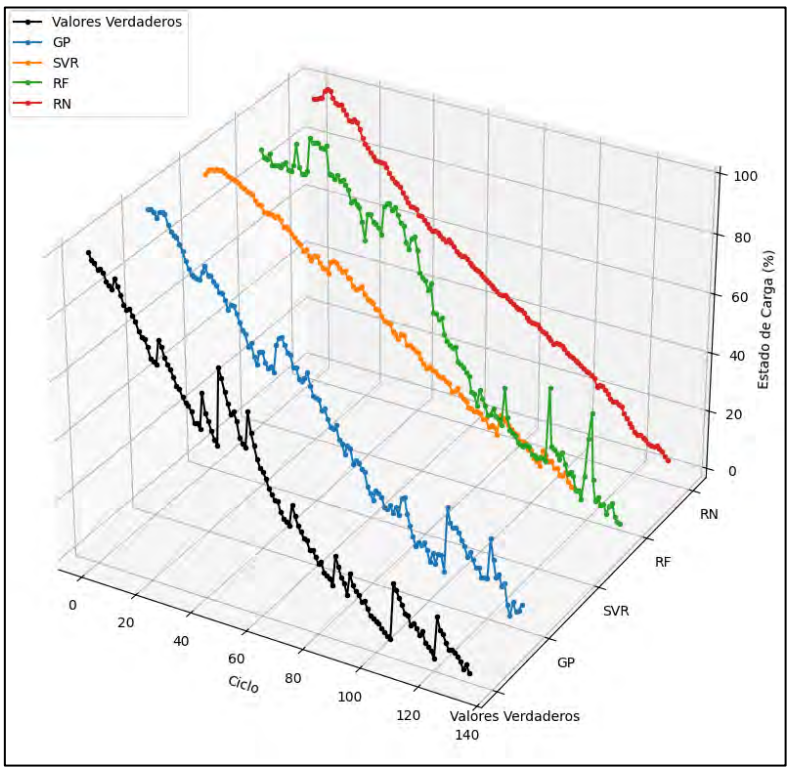
A continuación se muestran los gráficos de las curvas de predicción del estado de carga con modelos de procesos gaussianos y las predicciones realizadas con modelos obtenidos a partir de algoritmos de RNA, SVR y bosques aleatorios usando datos de las baterías B006 y B018, los cuales se obtuvieron en las secciones 4.1.2, 4.1.3 y 4.1.4, respectivamente. Los cuadros comparativos en 3 dimensiones de las curvas de predicción representativo de cada modelo se muestran en la Figura 4.13 y en la Figura 4.14.

Figura 4.13. Cuadro comparativo de las predicciones con modelos de regresión (B006).



Nota. Elaboración propia.

Figura 4.14. Cuadro comparativo de las predicciones con modelos de regresión (B018).



Nota. Elaboración propia.

En las curvas de predicción que se muestran en la Figura 4.13 y la Figura 4.14, se observa que la predicción de los modelos de Proceso Gaussiano, se aproximan más a las

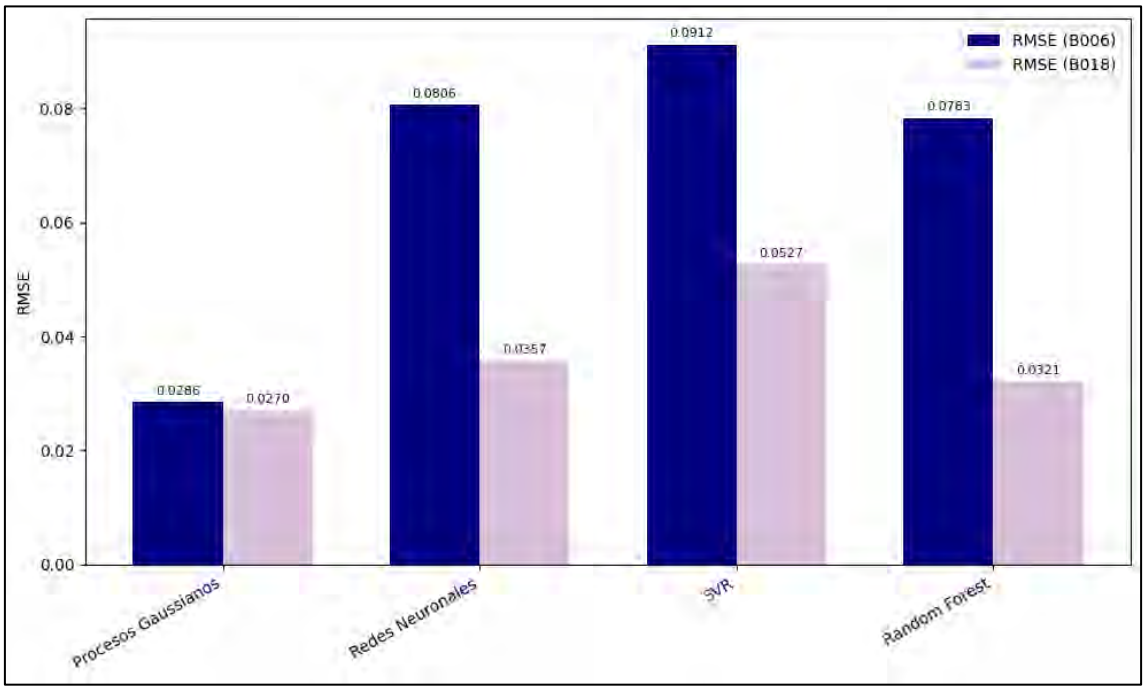
curvas de los datos verdaderos del estado de carga en comparación a las curvas de predicción de SVR, bosques aleatorios (RF) y redes neuronales artificiales (RN).

Tabla 4.3. Resumen de los valores óptimos de RMSE.

Algoritmo	RMSE (B006)	RMSE (B018)
Procesos Gaussianos	0.029	0.027
Redes Neuronales	0.081	0.036
SVR	0.091	0.053
Randon Forest	0.078	0.032

Para una comparación cuantificable y observables se desarrolló la Tabla 4.3, donde se muestra los valores correspondientes al indicador de eficiencia que es el RMSE obtenidos con datos de las baterías B006 y B018 (que son los datos de validación).

Figura 4.15. Cuadro comparativo de los resultados de RMSE.



Nota. Elaboración propia.

Los resultados gráficos de la Tabla 4.3, se muestran en la Figura 4.15, donde se puede observar las diferencias en barras verticales de la eficiencia de los modelos de regresión expresado en valores del RMSE.

4.2. Discusión de Resultados.

Respecto a los resultados de las predicciones del estado de carga que se muestran en la Figura 4.2, Figura 4.3 y Figura 4.4, las predicciones del estado de carga con los modelos PG-CFC en datos de validación, como 'Celda 4', 'Celda 7' y 'Celda 8', muestran una eficiencia de RMSE de 0.020, 0.0000112 y 0.000108, respectivamente y con valores óptimos del NLML de -516.88, -305.41 y -305.41, respectivamente. Además, el NLML, indica que el modelo es el que mejor se ajusta a los datos y a sus posibles variaciones y es el que tiene menor probabilidad de cometer sobre ajustes.

Se entiende por 'sobre ajuste', al hecho de que el modelo no se puede adaptar a ciertos cambios o variaciones de los datos de entrada. Así, si un modelo está sobre ajustado, a pesar que se introduzcan variaciones significantes en los datos de entrada, el modelo no percibe estas variaciones, por consiguiente, la predicción sería siempre la misma.

Respecto al desarrollo de modelos de redes neuronales (en la sección 4.1.2), la elección del modelo de RNA representativo se lleva a cabo de acuerdo a los resultados de RMSE, según los resultados de las predicciones del estado de carga de las baterías B006 y B018, los cuales se muestran en la Tabla 4.1 y a la Tabla 4.2, los algoritmos de RNA son más eficientes cuando usan el optimizador Optuna, para optimizar hiperparámetros. En consecuencia, siempre que se usen algoritmos de RNA, se debe utilizar el Optimizador Optuna.

En cuanto al desarrollo de modelos de regresión usando algoritmos de SVR, se puede observar en la Tabla 4.1 y la Tabla 4.2, que se obtuvo mayor eficiencia usando optimizadores Optuna y Grid Search, para predicciones de datos de las baterías B006 y B018, respectivamente, por consiguiente, se puede afirmar que el desarrollo de modelos basados en SVR se pueden construir usando el optimizador Optuna o Grid Search.

Respecto al desarrollo de modelos de regresión a través de algoritmos de Bosques aleatorios (Random Forest), según los resultados de RMSE que se muestran en la Tabla 4.1 y en la Tabla 4.2, la mayor eficiencia se obtuvo al usar el optimizador Optuna al predecir el estado de carga, tanto de la batería B006 como de la batería B018. Por consiguiente, se puede afirmar que, el desarrollo de modelos de regresión con algoritmos de bosques aleatorios debe ir acompañado del optimizador Optuna para este tipo de baterías.

Respecto a la elección del modelo más eficiente que se obtuvo al entrenar con datos de las baterías B005 y B036, el modelo correspondiente a un NLML de -987.18 corresponde a un modelo cuya función de covarianza corresponde a la función compuesta por la combinación lineal de suma y producto de la función producto punto y las funciones Matern 3/2 y Matern 5/2, tal como se muestra en la sección 4.1.5 y en la fila 20 de la Tabla 2.1.

De acuerdo a la Tabla 4.3 y haciendo uso del indicador de RMSE, las predicciones obtenidas con modelos de Procesos Gaussianos son más eficientes con respecto a los modelos de RNA, SVR y bosques aleatorios. Esto se puede observar para predicciones realizadas tanto para B006 como para B018.

La eficiencia de los modelos obtenidos a partir de algoritmos de procesos gaussianos con núcleos compuestos, también son comparados con los resultados obtenidos en la literatura referente a antecedentes de la investigación de la sección 2.3. Para propósitos comparativos de la eficiencia en la predicción del estado de carga se desarrolló la Tabla 4.4, donde se muestra la referencia, el algoritmo usado, los datos y la raíz del error cuadrático medio (RMSE, que es el indicador de eficiencia). En la Tabla 4.4, en la columna 'Algoritmo', se muestra las abreviaciones que provienen del inglés, como, Gaussian Process (GP), Deep Gaussian Process (DGP), Two-Layer Stacking Regressor (TLSR), Semisupervised Learning (SSL), Energy Valley Optimization-Least Square Support Vector Machine (EVO-LSSVM) y los modelos de la presente investigación, los cuales fueron obtenidos a partir de algoritmos de Procesos Gaussianos con la Composición de Funciones de Covarianza (PG-CFC)

En cuanto a los resultados, se puede observar que el algoritmo más eficiente en la predicción del estado de carga con datos de la batería B006, es el modelo obtenido con el algoritmo EVO-LSSVM con un valor de RMSE = 0.008, seguido por el modelo obtenido con el algoritmo TLSR con un RMSE = 0.010 y seguido por el modelo PG-CFC, con el que se obtuvo un RMSE = 0.029. De todos los algoritmos usados en los antecedentes, los que utilizan procesos gaussianos, son los modelos con algoritmos GP (Richardson et al., 2019) y DGP (Tagade et al., 2020), obteniendo valores de RMSE de 0.070 y 0.282, respectivamente, valores que son superados por el modelo GP-CFC. Por otro lado, los modelos desarrollados con EVO-LSSVM y TLSR, son algoritmos muy complejos. Según la literatura, a mayor complejidad de los algoritmos de aprendizaje automático se obtiene mayor costo computacional (Shalev-Shwartz & Ben-David, 2014). Por consiguiente, los

algoritmos desarrollados en la presente investigación demandan menor costo computacional respecto a algoritmos EVO-LSSVM y TLSR. De todo lo anterior se puede afirmar que las predicciones del estado de carga con modelos PG-FCF, resultan ser competitivos respecto a los modelos desarrollados con EVO-LSSVM y TLSR y resultan ser superiores en eficiencia respecto a los algoritmos desarrollados con procesos gaussianos, como el de GP y DGP.

Un análisis similar se puede hacer para los valores de RMSE en las predicciones del estado de carga con datos de la batería B018, claramente se puede observar en la Tabla 4.4, que el valor de RMSE = 0.027 obtenido con PG-CFC, es superior al valor de RMSE = 0.091, obtenido con TLSR.

En cuanto a la eficiencia en la predicción del estado de carga usando datos de la Celda 4, Celda 7 y la Celda 8, se observa claramente la superioridad en RMSE de los modelos PG-CFC frente a los modelos SSL (Tabla 4.4).

Tabla 4.4. Error cuadrático medio de la predicción del estado de carga.

Referencia	Algoritmo	Datos	RMSE
(Richardson et al., 2019)	GP	B006	0.070
(Tagade et al., 2020)	DGP	B006	0.282
(Yuan et al., 2023)	TLSR	B006	0.010
		B018	0.091
(Lin et al., 2023)	SSL	Celda 4	0.0036
		Celda 7	0.0092
		Celda 8	0.0045
(Luo et al., 2025)	GP	Celda 7	0.000944
		Celda 8	0.000632
(Y. Liu et al., 2025)	EVO-LSSVM	B006	0.008
Desarrollado en la presente investigación	PG-CFC	B006	0.029
		B018	0.027
		Celda 4	0.019
		Celda 7	0.00001
		Celda 8	0.00011

Además de la literatura mencionada antes, existen antecedentes en donde se predice el RUL de las baterías de ion de litio. Por consiguiente, haciendo uso de la ecuación 2.71 se obtuvo las predicciones del RUL para los modelos PG-CFC. La Tabla 4.5, es un cuadro comparativo de la eficiencia en la predicción del RUL con modelos obtenidos con algoritmos, cuyas abreviaciones provienen del inglés, como; Fuzzy Information Granulation-Artificial Bee Colony Optimization-Support Vector Regression (FIG-ABC-

SVR), Complete Ensemble Empirical Mode Decomposition With Adaptive Noise- Whale Optimization Algorithm-Support Vector Regression (CEEDMAN-WOA-SVR), Principal Component Analysis- improved Gaussian process regression (PCA- IGPR), Long Short-Term Memory- Support Vector Regression (LSTM-SVR), Grey Wolf Optimizer-support vector regression (GWO-SVR), Hybrid Gray Wolf Optimization – Support Vector Regression (HGWO- SVR) y Neural Network and Adaptive Unscented Kalman Filter (NN-AUKF)

Los resultados de la medida de la eficiencia en la predicción del RUL usando datos de la batería B006 con procesos gaussianos reportados en (Xing et al., 2023) muestran un RMSE de 2.01, mientras que el RMSE obtenido con el modelo PG-CFC fue de 0.010. aquí se muestra una clara mejora de la eficiencia en la predicción de la vida útil remanente. El modelo HGWO-SVR, resulta ser más eficiente, con un RMSE de 0.0026, pero la demanda de costo computacional es mayor al de los modelos PG-CFC.

El análisis de las predicciones con datos de la batería B018 también se lleva a cabo haciendo uso de la Tabla 4.5. Aquí se observa la superioridad de los modelos GWO-SVR, HGWO-SVR y NN-AUKF, para predecir la RUL, cuyos reportes de eficiencia fueron de 0.005,0.004 y 0.008 frente a la eficiencia alcanzada por GP-CFC, que fue de 0.013. Pero esa superioridad alcanzada, fue a costa de un mayor costo computacional, lo que hace que su aplicabilidad en dispositivos móviles sea cuestionable. Por otro lado, el modelo PG-CFC obtuvo mejoras en la eficiencia de la predicción de la RUL frente a modelos basados en procesos gaussianos. Esto se puede observar en (Xing et al., 2023) donde se reporta una eficiencia de 2.01 con GP frente al resultado en eficiencia obtenido con PG-CFC que fue de 0.013.

Tabla 4.5. Resultados de la raíz del error cuadrático medio en las predicciones de la vida útil remanente.

Referencia	Algoritmo	Datos	RMSE
(Z. Li et al., 2022)	FIG-ABC-SVR	B006	0.031
(Meng et al., 2022)	CEDMAN- WOA-SVR	B006 B018	0.012 0.023
(Xing et al., 2023)	PCA-IGPR, GP, LSTM-SVR	B006	1.17, 2.01, 1.43
(M. Zhang et al., 2023)	GWO-SVR, HGWO-SVR	B006 B018 B018	0.020, 0.0026 0.005, 0.004 0.091
(Wu et al., 2024)	NN-AUKF	B018	0.0082
Desarrollado en la presente investigación	PG-CFC	B006 B018	0.010 0.013

Esta sección de análisis de la Tabla 4.4 y la Tabla 4.5, muestran resultados de RMSE que permiten hacer comparaciones en eficiencia para predecir el estado de carga y la vida útil remanente entre los modelos descritos en antecedentes de la investigación y los modelos desarrollados en la presente investigación. Se puede observar claramente la mejora en eficiencia alcanzada por el modelo PG-CFC respecto a los modelos basados en procesos gaussianos reportados por (Richardson et al., 2019), (Tagade et al., 2020) y (Xing et al., 2023).

CONCLUSIONES

Respecto al primer objetivo específico, orientado a 'minimizar la raíz del error cuadrático medio para mejorar la eficiencia de las predicciones al componer funciones de covarianza en procesos gaussianos', se consiguió minimizar la raíz del error cuadrático medio a través del uso del optimizador Optuna, el cual permitió encontrar los valores óptimos de los hiperparámetros que minimizan el RMSE, obteniéndose un valor mínimo de RMSE de 0.027 y 0.00001 para datos del repositorio de la NASA y Oxford, respectivamente. La evidencia de la incorporación del optimizador en el algoritmo de predicción se muestra en la Figura 3. **13** de la sección 3.10.2.5 y los valores RMSE obtenidos para la composición de funciones de covarianza se muestra en la Tabla 4.3. Con esta evidencia es posible afirmar que se consiguió el primer objetivo específico.

El segundo objetivo específico, tiene que ver con llegar a obtener valores mínimos del logaritmo de la probabilidad marginal negativa. El utilizar el optimizador de hiperparámetros no solo contribuyó a minimizar el indicador RMSE, sino que también minimizó los valores del logaritmo negativo de la probabilidad marginal o NLML, la evidencia del cálculo de este indicador en el algoritmo desarrollado, se muestra en la Figura 3.11 de la sección 3.10.2.5 y los resultados del valor del NLML ya minimizado para todos los modelos de procesos gaussianos con funciones de covarianza compuestas, se muestran en la Figura 4.1 y en la Figura 4.11, con valores de NLML de -516.88 y -987.18, para datos de Oxford y de la NASA, respectivamente, mientras que en la literatura se obtuvo un valor mínimo de NLML igual a -530 con datos de la NASA. Por consiguiente, se confirma haber alcanzado el segundo objetivo específico.

En relación con el objetivo general, se logró optimizar la eficiencia en la predicción del estado de carga (SOC) en las baterías de ion de litio. Esta mejora se constata en el análisis comparativo expuesto en la Tabla 4.3, donde se evalúa el rendimiento predictivo mediante la raíz del error cuadrático medio (RMSE). Los resultados contenidos en dicha tabla corroboran la superioridad de los modelos basados en Procesos Gaussianos configurados con funciones de covarianza compuestas. Específicamente, se demostró un incremento promedio en la eficiencia de predicción del 14.8% para los datos de la NASA y del 72.8% para los de Oxford, consolidando de este modo la validez técnica de la propuesta.

RECOMENDACIONES

Implementación en sistemas embebidos de baja potencia: Dado que el costo computacional de los modelos de regresión propuestos es significativamente bajo como se constata en los tiempos de ejecución de las Tablas A3.1 y A3.2, se sugiere la integración y despliegue de estos algoritmos basados en procesos gaussianos en microcontroladores y sistemas de gestión de baterías comerciales, confirmando que este enfoque es idóneo para aplicaciones en dispositivos pequeños y tecnologías de Edge Computing.

Escalabilidad de la infraestructura de hardware: Para el desarrollo y entrenamiento de los modelos se utilizó la arquitectura en la nube de Google Colab en su versión gratuita, empleando un entorno con 10 GB de memoria RAM, lo cual demostró ser suficiente para regresiones bayesianas que exigen un consumo moderado de recursos. No obstante, se recomienda adquirir licencias avanzadas de procesamiento gráfico (como Google Colab Pro o entornos basados en GPU/TPU dedicadas) si se desea agilizar la sintonización de hiperparámetros en tiempo real o procesar conjuntos de datos masivos a escala industrial, optimizando significativamente los tiempos de convergencia del optimizador Optuna.

Diversificación de químicas y perfiles de degradación: Con el propósito de expandir los alcances y la validez universal de la metodología propuesta, se recomienda para futuras investigaciones evaluar la composición de funciones de covarianza en celdas con químicas distintas a las estudiadas, tales como Ferrofosfato de Litio o Níquel-Manganeso-Cobalto, operadas bajo ciclos de temperatura dinámicos o perfiles de descarga acelerada. Esto permitirá validar la capacidad de transferencia y la robustez del núcleo matemático frente a fenómenos electroquímicos de degradación aún más severos.

REFERENCIAS BIBLIOGRÁFICAS

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). *Optuna: A Next-generation Hyperparameter Optimization Framework*. http://arxiv.org/abs/1907.10902

Bole, B., Kulkarni, C. S., & Daigle, M. (2014). Adaptation of an electrochemistry-based Li-ion battery model to account for deterioration observed under randomized use. *PHM 2014 - Proceedings of the Annual Conference of the Prognostics and Health Management Society 2014*, 502–510.

Bundhoo, Z. M. A. (2018). Microwave-assisted conversion of biomass and waste materials to biofuels. In *Renewable and Sustainable Energy Reviews* (Vol. 82, pp. 1149–1177). Elsevier Ltd. https://doi.org/10.1016/j.rser.2017.09.066

Carrasco, S. (2019). *Metodología de la investigación científica. Pautas metodológicas para diseñar y elaborar el proyecto de investigación* (19th ed., Vol. 1). Editorial San Marcos.

Cristianini, N., & Shawe-Taylor, J. (2000). An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. In *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press. https://doi.org/10.1017/cbo9780511801389

Duvenaud, D. K. (2014). Automatic Model Construction with Gaussian Processes. *Thesis*, *June*, 144. https://www.cs.toronto.edu/~duvenaud/thesis.pdf

Farmann, A., Waag, W., Marongiu, A., & Sauer, D. U. (2015). Critical review of on-board capacity estimation techniques for lithium-ion batteries in electric and hybrid electric vehicles. *Journal of Power Sources*, *281*, 114–130. https://doi.org/10.1016/j.jpowsour.2015.01.129

Fu, L., Jiang, B., Zhu, J., Wei, X., & Dai, H. (2025). Early Remaining Useful Life Prediction for Lithium-Ion Batteries Using a Gaussian Process Regression Model Based on Degradation Pattern Recognition †. *Batteries*, *11*(6). https://doi.org/10.3390/batteries11060221

Goebel, K., Saha, B., Saxena, A., Celaya, J. R., & Christophersen, J. P. (2008). Prognostics in battery health management. *IEEE Instrumentation and Measurement Magazine*, *11*(4), 33–40. https://doi.org/10.1109/MIM.2008.4579269

Goodfellow, Ian., Bengio, Yoshua., & Courville, Aaron. (2017). *Deep Learning (Adaptive Computation and Machine Learning series)* (The MIT Press, Ed.; 1st ed., Vol. 1). The MIT Press.

Hastie, T., Tibshirani, R., & Friedman, J. (n.d.). *Springer Series in Statistics* (Springer, Ed.).

Howey, D., & Birkl, C. (2017). *Oxford Battery Degradation Dataset*. https://doi.org/10.5287/bodleian:KO2kdmYGg

Khalid, A., Sundararajan, A., & Sarwat, A. I. (2019, December 1). An ARIMA-NARX Model to Predict Li-Ion State of Charge for Unknown Charge/Discharge Rates. *2019 IEEE Transportation Electrification Conference, ITEC-India 2019*. https://doi.org/10.1109/ITEC-India48457.2019.ITECIndia2019-1

Li, X., Shu, X., Shen, J., Xiao, R., Yan, W., & Chen, Z. (2017). An on-board remaining useful life estimation algorithm for lithium-ion batteries of electric vehicles. *Energies*, *10*(5). https://doi.org/10.3390/en10050691

Li, Z., Chen, J., Gao, K., Zhang, F., & Xu, J. (2022). Remaining useful life prediction of Lithium-ion battery using improved support vector regression and fuzzy information granulation. *Journal of Physics: Conference Series*, *2301*(1). https://doi.org/10.1088/1742-6596/2301/1/012015

Lin, C., Xu, J., & Mei, X. (2023). Improving state-of-health estimation for lithium-ion batteries via unlabeled charging data. *Energy Storage Materials*, *54*, 85–97. https://doi.org/10.1016/j.ensm.2022.10.030

Liu, D., Bai, Y., Zhao, S., & Zhang, W. (2012). Improved cycling performance of 5 v spinel LiMn 1.5Ni 0.5O 4 by amorphous FePO 4 coating. *Journal of Power Sources*, *219*, 333–338. https://doi.org/10.1016/j.jpowsour.2012.07.058

Liu, D., Luo, Y., Peng, Y., Peng, X., & Pecht, M. (2012). Lithium-ion battery remaining useful life estimation based on nonlinear AR model combined with degradation feature. *Proceedings of the Annual Conference of the Prognostics and Health Management Society 2012, PHM 2012*, 336–342.

Liu, Y., Ding, J., Yao, L., Su, H., Chen, Y., & Wang, Z. (2025). A novel high-accuracy intelligent estimation method for battery state of health. *Measurement: Journal of the International Measurement Confederation*, *245*. https://doi.org/10.1016/j.measurement.2024.116620

Luo, X., Song, Y., Bu, W., Liang, H., & Zheng, M. (2025). A Joint Prediction of the State of Health and Remaining Useful Life of Lithium-Ion Batteries Based on Gaussian Process Regression and Long Short-Term Memory. *Processes*, *13*(1). https://doi.org/10.3390/pr13010239

Ma, G., Zhang, Y., Cheng, C., Zhou, B., Hu, P., & Yuan, Y. (2019). Remaining useful life prediction of lithium-ion batteries based on false nearest neighbors and a hybrid neural network. *Applied Energy*, *253*(July), 113626. https://doi.org/10.1016/j.apenergy.2019.113626

Madani, S. S., Shabeer, Y., Fowler, M., Panchal, S., Chaoui, H., Mekhilef, S., Dou, S. X., & See, K. (2025). Artificial Intelligence and Digital Twin Technologies for Intelligent Lithium-Ion Battery Management Systems: A Comprehensive Review of State Estimation, Lifecycle Optimization, and Cloud-Edge Integration. In *Batteries* (Vol. 11, Issue 8). Multidisciplinary Digital Publishing Institute (MDPI). https://doi.org/10.3390/batteries11080298

Manthiram, A. (2020). A reflection on lithium-ion battery cathode chemistry. In *Nature Communications* (Vol. 11, Issue 1). Nature Research. https://doi.org/10.1038/s41467-020-15355-0

Meng, X., Cai, C., Wang, Y., Wang, Q., & Tan, L. (2022). Remaining useful life prediction of lithium-ion batteries using CEEMDAN and WOA-SVR model. *Frontiers in Energy Research*, *10*. https://doi.org/10.3389/fenrg.2022.984991

Mou, J., Yang, Q., Tang, Y., Liu, Y., Li, J., & Yu, C. (2024). Prediction of the Remaining Useful Life of Lithium-Ion Batteries Based on the 1D CNN-BLSTM Neural Network. *Batteries*, *10*(5). https://doi.org/10.3390/batteries10050152

Murphy P, K. (2012). Machine Learning A Probabilistic Perspective. In MIT-Press (Ed.), *Chance Encounters: Probability in Education* (1st ed.). https://doi.org/10.1007/978-94-011-3532-0_2

Rasmussen, C. E., & Williams, C. K. I. (2006). Gaussian processes for machine learning. 2006. In *The MIT Press, Cambridge, MA, USA* (Vol. 38, Issue 2).

Richardson, R. R., Birkl, C. R., Osborne, M. A., & Howey, D. (2018). Gaussian Process Regression for In-situ Capacity Estimation of Lithium-ion Batteries. *IEEE Transactions on Industrial Informatics*, 1–13. https://doi.org/10.1109/TII.2018.2794997

Richardson, R. R., Osborne, M. A., & Howey, D. A. (2017). Gaussian process regression for forecasting battery state of health. *Journal of Power Sources*, *357*, 209–219. https://doi.org/10.1016/j.jpowsour.2017.05.004

Richardson, R. R., Osborne, M. A., & Howey, D. A. (2019). Battery health prediction under generalized conditions using a Gaussian process transition model. *Journal of Energy Storage*, *23*, 320–328. https://doi.org/10.1016/j.est.2019.03.022

Rokach, Lior., & Maimon, Oded. (2015). *Data mining with decision trees : theory and applications*. World Scientific.

Saha, B., & Goebel, K. (2007). NASA AMES prognostics data repository. In *NASA Ames Research Center*. https://phmdatasets.s3.amazonaws.com/NASA/5.+Battery+Data+Set.zip

Seeger, M. (2004). Gaussian processes for machine learning. In *International journal of neural systems* (Vol. 14, Issue 2). https://doi.org/10.1142/S0129065704001899

Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding machine learning: From theory to algorithms. In Cambridge University Press (Ed.), *Understanding Machine Learning: From Theory to Algorithms* (1st ed., Vol. 1). Cambridge University Press. https://doi.org/10.1017/CBO9781107298019

Suh, S., Mittal, D. A., Bello, H., Zhou, B., Jha, M. S., & Lukowicz, P. (2024). Remaining useful life prediction of Lithium-ion batteries using spatio-temporal multimodal attention networks. *Heliyon*, *10*(16). https://doi.org/10.1016/j.heliyon.2024.e36236

Tagade, P., Hariharan, K. S., Ramachandran, S., Khandelwal, A., Naha, A., Kolake, S. M., & Han, S. H. (2020). Deep Gaussian process regression for lithium-ion battery health prognosis and degradation mode diagnosis. *Journal of Power Sources*, *445*(October 2019), 227281. https://doi.org/10.1016/j.jpowsour.2019.227281

Waldmann, T., Wilka, M., Kasper, M., Fleischhammer, M., & Wohlfahrt-Mehrens, M. (2014). Temperature dependent ageing mechanisms in Lithium-ion batteries - A Post-Mortem study. *Journal of Power Sources*, *262*, 129–135. https://doi.org/10.1016/j.jpowsour.2014.03.112

Wei, J., Dong, G., & Chen, Z. (2018). Remaining Useful Life Prediction and State of Health Diagnosis for Lithium-Ion Batteries Using Particle Filter and Support Vector Regression. *IEEE Transactions on Industrial Electronics*, *65*(7), 5634–5643. https://doi.org/10.1109/TIE.2017.2782224

Wu, L., Guo, W., Tang, Y., Sun, Y., & Qin, T. (2024). Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Neural Network and Adaptive Unscented Kalman Filter. *Electronics (Switzerland)*, *13*(13). https://doi.org/10.3390/electronics13132619

Xing, J., Zhang, H., & Zhang, J. (2023). Remaining useful life prediction of – Lithium batteries based on principal component analysis and improved Gaussian process regression. *International Journal of Electrochemical Science*, *18*(4). https://doi.org/10.1016/j.ijoes.2023.100048

Yang, D., Zhang, X., Pan, R., Wang, Y., & Chen, Z. (2018). A novel Gaussian process regression model for state-of-health estimation of lithium-ion battery using charging curve. *Journal of Power Sources*, *384*(September 2017), 387–395. https://doi.org/10.1016/j.jpowsour.2018.03.015

Yu, J. (2018). State of health prediction of lithium-ion batteries: Multiscale logic regression and Gaussian process regression ensemble. *Reliability Engineering and System Safety*, *174*(June 2017), 82–95. https://doi.org/10.1016/j.ress.2018.02.022

Yuan, J., Qin, Z., Huang, H., Gan, X., Li, S., & Li, B. (2023). State of Health Estimation and Remaining Useful Life Prediction for a Lithium-Ion Battery with a Two-Layer Stacking Regressor. *Energies*, *16*(5). https://doi.org/10.3390/en16052313

Zhang, M., Wang, S., Xie, Y., Yang, X., Hao, X., & Fernandez, C. (2023). *Hybrid Gray Wolf Optimization Method in Support Vector Regression Framework for Highly Precise Prediction of Remaining Useful Life of Lithium-ion Batteries*. https://doi.org/10.21203/rs.3.rs-2834343/v1

Zhao, Q., Qin, X., Zhao, H., & Feng, W. (2018). A novel prediction method based on the support vector regression for the remaining useful life of lithium-ion batteries. *Microelectronics Reliability*, *85*(37), 99–108. https://doi.org/10.1016/j.microrel.2018.04.007

ANEXOS

Anexo 1. Matriz de Consistencia.

Tabla A1.1. Matriz de consistencia.

Título: Composición de Funciones de Covarianza en Procesos Gaussianos Para Mejorar La Eficiencia en la Predicción del Estado de Carga en Baterías de Ion de Litio.

Presentado Por: Willy Cruz Borda

Problema	Objetivo	Justificación	Hipótesis	Variables	Indicadores	Metodología
Problema general ¿Es posible componer funciones de covarianza en procesos gaussianos para mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio? **Problemas Específicos** ¿En qué medida influye la composición de funciones de covarianza de procesos gaussianos en el error cuadrático medio? ¿En qué medida influye la composición de funciones de covarianza de procesos gaussianos en el valor del logaritmo negativo de la probabilidad marginal?	**Objetivo:** Mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio utilizando procesos gaussianos **Objetivos Específicos** Aplicar la composición de funciones de covarianza en procesos gaussianos para minimizar el error cuadrático medio. Aplicar la composición de funciones de covarianza en procesos gaussianos para minimizar el valor del logaritmo negativo de la probabilidad marginal.	Existe la necesidad de garantizar el funcionamiento confiable, minimizar costos de mantenimiento y evitar ineficiencias en la valoración económica, en proyectos de almacenamiento de energía	La composición de funciones de covarianza en procesos gaussianos permite mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio **Hipótesis Específicas** La composición de funciones de covarianza en procesos gaussianos permite obtener valores mínimos del error cuadrático medio. La composición de funciones de covarianza en procesos gaussianos permite obtener valores mínimos del logaritmo negativo de la probabilidad marginal.	**Variable independiente:** Composición de funciones de covarianza en procesos gaussianos. **Variable dependiente:** Mejorar la eficiencia en la predicción del estado de carga en baterías de ion de litio	1. **Raíz del error cuadrático medio**. 2. **Logaritmo de la probabilidad marginal negativa**	a. **Tipo de investigación:** Experimental b. **Diseño de la Investigación:** . Revisión de la literatura. . Estudio de los procesos gaussianos. . Elaboración de núcleos compuestos. . Elaboración del modelo. . Predicción. . Validación del modelo. **c. Instrumentos utilizados:** . Ordenador con 12 GB de RAM instalada. . Lenguaje de programación Python

Anexo 2. Instrumentos de recolección de información.

La recolección de los datos se llevó a cabo a través de la revisión documental de los siguientes links.

Repositorio de la NASA.

https://data.nasa.gov/dataset/randomized-and-recommissioned-battery-dataset

Repositorio de Oxford.

https://ora.ox.ac.uk/objects/uuid:03ba4b01-cfed-46d3-9b1a-7d4a7bdf6fac

Anexo 3. Otros.

Problema primal

En optimización matemática (y en aprendizaje automático), un problema primal es el problema original que se desea resolver, generalmente formulado para minimizar (o maximizar) una función objetivo sujeta a ciertas restricciones.

Para un problema de optimización, que en este caso es el de minimizar una función se plantea como.

$$\min_x f(x) \tag{A3.1}$$

Que está sujeto a restricciones como las funciones g y h:

$$g_i(x) \leq 0, \qquad i = 1, \dots, m \tag{A3.2}$$

$$h_j(x) = 0, \qquad j = 1, \dots, p \tag{A3.3}$$

Las condiciones de Karush Kuhn Tucker (KKT) presupone la existencia de un conjunto de multiplicadores $\lambda_i \geq 0$ (para las restricciones de desigualdad) y μ_j (para las restricciones de igualdad), las cuales tienen que satisfacer ciertas propiedades.

La formulación de KKT de forma simplificada se enuncia de la siguiente forma.

Para una solución x^*, existen $\lambda_i^* \geq 0$ y μ_j^*, tal que cumple las siguientes propiedades

1. La estacionalidad, está dada por.

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^{p} \mu_j^* \nabla h_j(x^*) = 0 \tag{A3.4}$$

2. Factibilidad del primal.

$$g_i(x^*) \leq 0\,, \qquad h_j(x^*) = 0 \tag{A3.5}$$

3. Factibilidad dual.

$$\lambda_i^* \geq 0 \tag{A3.6}$$

4. Complementariedad.

$$\lambda_i^* g_i(x^*) = 0 \quad \forall\, i \tag{A3.7}$$

Esto implica que cada restricción, o bien está activa (igual a cero) y su multiplicador es mayor a cero, o no está activa y su multiplicador es cero.

Las KKT permiten identificar puntos candidatos a ser óptimos considerando tanto la función objetivo como las restricciones. Siendo la base teórica para el desarrollo de algoritmos de optimización con restricciones, como SVM y SVR.

Producto elemento a elemento (Producto Hadamard)

El producto Hadamard es una operación entre dos matrices o vectores del mismo tamaño, y consiste en multiplicar cada elemento correspondiente de ambas matrices o vectores. Se define matemáticamente como lo siguiente.

Dadas dos matrices o vectores $A = [a_{ij}]$ y $B = [b_{ij}]$, de dimensiones $m \times n$, El producto Hadamard se representa como.

$$C = A \odot B \tag{A3.8}$$

tal que.

$$c_{ij} = a_{ij} \times b_{ij}, \text{ para } i = 1, \ldots, m\,, \;\; j = 1, \ldots, n \tag{A3.9}$$

Esto indica que, el elemento c_{ij}, se obtiene al multiplicar directamente a_{ij} por elemeto b_{ij}.

Ejemplo demostrativo de Bosque aleatorios.

Supóngase, el conjunto de datos representado por el vector de características, X, y el vector objetivo continuo Y, cuyos datos se muestran en la siguiente tabla.

Datos de entrenamiento

Índice	X	Y
1	1.0	1.5
2	2.0	1.7
3	3.0	3.2
4	4.0	3.0
5	5.0	4.5

Paso1. Determinar los posibles puntos de división θ. Los posibles umbrales o puntos medios entre los valores de X, son. $\theta \in \{1.5, 2.5, 3.5, 4.5\}$

Paso 2. Calcular la suma de varianzas ponderadas para cada división.

La función de coste para cada regresión es minimizar la suma ponderada de las varianzas, en los nodos hijos. Para cada división de θ:

Por consiguiente, se divide los datos en $R_{izquierda} = \{x_i | x_i \leq \theta\}$ y $R_{derecha} = \{x_i | x_i > \theta\}$, luego se calcula la varianza en cada región y se calcula la suma ponderada de las varianzas.

Para $\boldsymbol{\theta = 1.5}$

$R_{izquierda} = \{(1.0,1.5)\}$

$Varianza = 0$ (solo un dato)

$R_{derecha} = \{(2.0,1.7), (3.0,3.2), (4.0,3.0), (5.0,4.5)\}$

Media: $\bar{y} = \frac{1.7+3.2+3.0+4.5}{4} = \frac{12.4}{4} = 3.1$

$Varianza = \frac{1}{4}[(1.7 - 3.1)^2 + (3.2 - 3.1)^2 + (3.0 - 3.1)^2 + (4.5 - 3.1)^2]$

$Varianza = 0.985$

$Suma\ Ponderada = \frac{1}{5} \times 0 + \frac{4}{5} \times 0.985 = 0.788$

Para $\boldsymbol{\theta = 2.5}$

$R_{izquierda} = \{(1.0,1.5), (2.0,1.7)\}$

Media: $\bar{y} = \frac{1.5+1.7}{2} = 1.6$

$Varianza = \frac{1}{2}[(1.5 - 1.6)^2 + (1.7 - 1.6)^2]$

$Varianza = 0.01$

$R_{derecha} = \{(3.0,3.2), (4.0,3.0), (5.0,4.5)\}$

Media: $\bar{y} = \frac{3.2+3.0+4.5}{3} = \frac{10.7}{3} = 3.567$

$Varianza = \frac{1}{3}[(3.2 - 3.567)^2 + (3.0 - 3.567)^2 + (4.5 - 3.567)^2]$

$Varianza = 0.441$

$Suma\ Ponderada = \frac{2}{5} \times 0.01 + \frac{3}{5} \times 0.441 = 0.269$

Para $\boldsymbol{\theta = 3.5}$

$R_{izquierda} = \{(1.0,1.5), (2.0,1.7), (3.0,3.2)\}$

Media: $\bar{y} = \frac{1.5+1.7+3.2}{3} = \frac{6.4}{3} = 2.133$

$Varianza = \frac{1}{3}[(1.5 - 2.133)^2 + (1.7 - 2.133)^2 + (3.2 - 2.133)^2]$

$Varianza = 0.574$

$R_{derecha} = \{(4.0,3.0), (5.0,4.5)\}$

Media: $\bar{y} = \frac{3.0+4.5}{2} = \frac{7.5}{2} = 3.75$

$Varianza = \frac{1}{2}[(3.0 - 3.75)^2 + (4.5 - 3.75)^2]$

$Varianza = 0.5625$

$Suma\ Ponderada = \frac{3}{5} \times 0.574 + \frac{2}{5} \times 0.5265 = 0.569$

Para $\boldsymbol{\theta = 4.5}$

$R_{izquierda} = \{(1.0,1.5), (2.0,1.7), (3.0,3.2), (4.0,3.0)\}$

Media: $\bar{y} = \frac{1.5+1.7+3.2+3.0}{4} = \frac{9.4}{4} = 2.35$

$Varianza = \frac{1}{4}[(1.5 - 2.35)^2 + (1.7 - 2.35)^2 + (3.2 - 2.35)^2 + (3.0 - 2.35)^2]$

$Varianza = 0.5725$

$R_{derecha} = \{(5.0,4.5)\}$

Media: $\bar{y} = \frac{3.0+4.5}{2} = \frac{7.5}{2} = 3.75$

$Varianza = 0$

$Suma\ Ponderada = \frac{4}{5} \times 0.575 + \frac{1}{5} \times 0 = 0.458$

Paso 3. Pasa por elegir la mejor división, tomando en cuenta que el umbral que minimiza la suma ponderada de varianza es.

$\theta = 2.5$ con coste ≈ 0.269

Paso 4. Predecir en los nodos hoja.

Para $X \leq 2.5$: media de $Y = 1.6$

Para $X > 2.5$: media de $Y \approx 3.567$

Tabla A3.1. Resultados de valores de hiperparámetros e indicadores de modelos de procesos gaussianos obtenidos para los datos del repositorio de Oxford, baterías etiquetadas como Celda 4 y Celda 8 (o C4 y C8).

GP_Kern	GP_Alpha	N_R_O	R_S	RMSE_C4	RMSE_C8	NLML	Tiempo (min)
Matern(length_scale=1, nu=1.5)	0.093336	15	310	0.019934	0.011577	-509.5	5.7748
Matern(length_scale=1, nu=2.5)	0.106098	15	647	0.019716	0.013111	-514.27	5.1447
RBF(length_scale=1)	0.111133	18	425	0.020093	0.01434	-516.88	4.9895
DotProduct(sigma_0=1) + RBF(length_scale=1e-05)	0.183744	13	827	0.021328	0.017868	-500.74	27.86
DotProduct(sigma_0=1) + Matern(length_scale=1, nu=1.5)	0.106669	16	950	0.027663	0.011533	-475.71	35.156

DotProduct(sigma_0=1) + Matern(length_scale=1, nu=2.5)	0.13628	5	675	0.024474	0.014059	-486.43	15.39
Matern(length_scale=1, nu=1.5) + Matern(length_scale=1, nu=2.5)	0.101117	19	722	0.032226	0.010817	-467.83	25.526
Matern(length_scale=1, nu=1.5) + RBF(length_scale=1)	0.105872	13	165	0.029624	0.01139	-469.36	17.115
Matern(length_scale=1, nu=2.5) + RBF(length_scale=1)	0.0001	12	799	0.035108	0.009048	-461.12	10.041
DotProduct(sigma_0=1) + 1**2 + Matern(length_scale=1, nu=1.5)	0.0001	15	997	0.106837	0.008571	-467.56	13.921
DotProduct(sigma_0=1) + 1**2 + Matern(length_scale=1, nu=2.5)	0.0001	17	101	0.106771	0.009407	-467.56	15.999
DotProduct(sigma_0=1) + 1**2 + RBF(length_scale=1)	0.0001	19	209	0.036	0.007767	-467.56	15.269
DotProduct(sigma_0=1) * Matern(length_scale=1, nu=1.5)	0.445218	5	298	0.060937	0.091798	-501.34	18.262
DotProduct(sigma_0=1) * Matern(length_scale=1, nu=2.5)	0.426065	0	830	0.058781	0.091528	-503.02	18.877
DotProduct(sigma_0=1) * RBF(length_scale=1)	0.024245	17	938	0.038008	0.014318	-307.98	51.105
1**2 * Matern(length_scale=1, nu=1.5)	0.145691	0	824	0.083509	0.107004	-206.18	11.104
1**2 * Matern(length_scale=1, nu=2.5)	0.107797	0	110	0.083362	0.106647	-155.52	6.695
1**2 * RBF(length_scale=1)	0.126312	1	160	0.08335	0.106639	-114.18	4.6575
1**2 * Matern(length_scale=1, nu=1.5) + 1**2 * Matern(length_scale=1, nu=2.5)	0.000129	6	483	0.111992	2.02E-05	-182.42	26.037
DotProduct(sigma_0=1) * Matern(length_scale=1, nu=1.5) + DotProduct(sigma_0=1) * Matern(length_scale=1, nu=2.5)	0.953685	9	858	0.061901	0.094408	-240.96	76.389
DotProduct(sigma_0=1) * 1**2 + Matern(length_scale=1, nu=1.5)	0.00121	2	679	0.113787	0.000108	-305.94	14.173

DotProduct(sigma_0=1) * 1**2 + Matern(length_scale=1, nu=2.5)	6.715836	17	139	0.086037	0.117217	-271.04	32.371
DotProduct(sigma_0=1) * 1**2 + RBF(length_scale=1)	0.000123	3	493	0.113802	1.1E-05	-222.51	13.951

Tabla A3.2. Resultados de valores de hiperparámetros e indicadores de modelos de procesos gaussianos obtenidos para los datos del repositorio de la NASA, baterías etiquetadas como B006 y B018.

GP_Kernel	GP_Alpha	N_R_O	R_S	RMSE (B006)	RMSE (B018)	NLML	Tiempo (minutos)
Matern(length_scale=1, nu=1.5)	0.130	10	666	0.123	0.057	-595.383	7.637
Matern(length_scale=1, nu=2.5)	0.135	7	629	0.122	0.056	-593.281	8.233
RBF(length_scale=1)	0.142	10	394	0.121	0.056	-591.928	6.335
DotProduct(sigma_0=1) + RBF(length_scale=1e-05)	0.110	12	873	0.114	0.056	-524.833	36.306
DotProduct(sigma_0=1) + Matern(length_scale=1, nu=1.5)	0.128	19	806	0.118	0.056	-536.183	49.608
DotProduct(sigma_0=1) + Matern(length_scale=1, nu=2.5)	0.118	3	763	0.115	0.056	-531.241	27.763
Matern(length_scale=1, nu=1.5) + Matern(length_scale=1, nu=2.5)	0.147	9	532	0.123	0.064	-541.064	26.289
Matern(length_scale=1, nu=1.5) + RBF(length_scale=1)	0.152	7	101	0.121	0.064	-538.260	17.742
Matern(length_scale=1, nu=2.5) + RBF(length_scale=1)	0.152	9	77	0.121	0.064	-538.029	25.048
DotProduct(sigma_0=1) + 1**2 + Matern(length_scale=1, nu=1.5)	0.136	12	352	0.117	0.059	-538.534	60.604
DotProduct(sigma_0=1) + 1**2 + Matern(length_scale=1, nu=2.5)	0.135	9	477	0.118	0.056	-535.330	48.956

DotProduct(sigma_0=1) + 1**2 + RBF(length_scale=1)	0.138	20	411	0.117	0.056	-533.202	59.143
DotProduct(sigma_0=1) * Matern(length_scale=1, nu=1.5)	0.251	17	931	0.029	0.029	-337.822	67.800
DotProduct(sigma_0=1) * Matern(length_scale=1, nu=2.5)	1.996	12	700	0.030	0.029	-730.010	37.088
DotProduct(sigma_0=1) * RBF(length_scale=1)	0.044	11	761	0.034	0.027	-28.736	26.851
1**2 * Matern(length_scale=1, nu=1.5)	0.020	16	741	0.056	0.039	-569.275	16.378
1**2 * Matern(length_scale=1, nu=2.5)	0.013	19	891	0.034	0.032	-571.821	18.538
1**2 * RBF(length_scale=1)	0.036	8	426	0.036	0.033	-112.614	7.322
1**2 * Matern(length_scale=1, nu=1.5) + 1**2 * Matern(length_scale=1, nu=2.5)	0.023	10	421	0.038	0.035	-313.974	41.223
DotProduct(sigma_0=1) * Matern(length_scale=1, nu=1.5) + DotProduct(sigma_0=1) * Matern(length_scale=1, nu=2.5)	0.031	18	306	0.029	0.029	-987.179	87.886
DotProduct(sigma_0=1) * 1**2 + Matern(length_scale=1, nu=1.5)	5.505	9	520	0.029	0.029	-971.225	22.916
DotProduct(sigma_0=1) * 1**2 + Matern(length_scale=1, nu=2.5)	5.197	7	148	0.029	0.029	-956.267	20.253
DotProduct(sigma_0=1) * 1**2 + RBF(length_scale=1)	0.0001	14	587	0.030	0.040	-512.035	38.011

Tabla A3.3. Resultado de las funciones de covarianza obtenidos en la etapa de entrenamiento de las baterías del repositorio de Oxford.

	Funciones de Covarianza
1	Matern(length_scale=1, nu=1.5)
2	Matern(length_scale=1, nu=2.5)
3	RBF(length_scale=1)
4	DotProduct(sigma_0=1) + RBF(length_scale=1e-05)
5	DotProduct(sigma_0=1) + Matern(length_scale=1, nu=1.5)
6	DotProduct(sigma_0=1) + Matern(length_scale=1, nu=2.5)
7	Matern(length_scale=1, nu=1.5) + Matern(length_scale=1, nu=2.5)
8	Matern(length_scale=1, nu=1.5) + RBF(length_scale=1)
9	Matern(length_scale=1, nu=2.5) + RBF(length_scale=1)
10	DotProduct(sigma_0=1) + 1**2 + Matern(length_scale=1, nu=2.5)
11	DotProduct(sigma_0=1) + 1**2 + Matern(length_scale=1, nu=1.5)
12	DotProduct(sigma_0=1) + 1**2 + RBF(length_scale=1)
13	DotProduct(sigma_0=1) * Matern(length_scale=1, nu=1.5)
14	DotProduct(sigma_0=1) * Matern(length_scale=1, nu=2.5)
15	DotProduct(sigma_0=1) * RBF(length_scale=1)
16	1**2 * Matern(length_scale=1, nu=1.5)
17	1**2 * Matern(length_scale=1, nu=2.5)
18	1**2 * RBF(length_scale=1)
19	1**2 * Matern(length_scale=1, nu=1.5) + 1**2 * Matern(length_scale=1, nu=2.5)
20	DotProduct(sigma_0=1) * Matern(length_scale=1, nu=1.5) + DotProduct(sigma_0=1) * Matern(length_scale=1, nu=2.5)

21	DotProduct(sigma_0=1) * 1**2 + Matern(length_scale=1, nu=1.5)
22	DotProduct(sigma_0=1) * 1**2 + Matern(length_scale=1, nu=2.5)
23	DotProduct(sigma_0=1) * 1**2 + RBF(length_scale=1)

Tabla A3.4. Resultado de las funciones de covarianza obtenidos en la etapa de entrenamiento de las baterías del repositorio de la NASA.

	Funciones de Covarianza
1	Matern(length_scale=0.0413, nu=1.5)
2	Matern(length_scale=0.037, nu=2.5)
3	RBF(length_scale=0.032)
4	DotProduct(sigma_0=0.0311) + RBF(length_scale=0.0304)
5	DotProduct(sigma_0=3.02) + Matern(length_scale=0.0415, nu=1.5)
6	DotProduct(sigma_0=0.00103) + Matern(length_scale=0.036, nu=2.5)
7	Matern(length_scale=119, nu=1.5) + Matern(length_scale=0.0321, nu=2.5)
8	Matern(length_scale=120, nu=1.5) + RBF(length_scale=0.028)
9	Matern(length_scale=96.8, nu=2.5) + RBF(length_scale=0.028)
10	DotProduct(sigma_0=1.71) + 0.124**2 + Matern(length_scale=65.2, nu=1.5)
11	DotProduct(sigma_0=2.96) + 0.153**2 + Matern(length_scale=0.0364, nu=2.5)
12	DotProduct(sigma_0=0.000122) + 2.9**2 + RBF(length_scale=0.0307)
13	DotProduct(sigma_0=1e+05) * Matern(length_scale=1.9e+04, nu=1.5)
14	DotProduct(sigma_0=1e+05) * Matern(length_scale=6.86e+03, nu=2.5)
15	DotProduct(sigma_0=1.01e+04) * RBF(length_scale=307)
16	316**2 * Matern(length_scale=164, nu=1.5)
17	316**2 * Matern(length_scale=65.1, nu=2.5)
18	316**2 * RBF(length_scale=35.8)

19	0.0461**2 * Matern(length_scale=5.37e+03, nu=1.5) + 316**2 * Matern(length_scale=71.8, nu=2.5)
20	DotProduct(sigma_0=1e+05) * Matern(length_scale=2.35e+04, nu=1.5) + DotProduct(sigma_0=0.345) * Matern(length_scale=103, nu=2.5)
21	DotProduct(sigma_0=1.63) * 42.9**2 + Matern(length_scale=3.9e+03, nu=1.5)
22	DotProduct(sigma_0=1.63) * 43.5**2 + Matern(length_scale=8.83e+03, nu=2.5)
23	DotProduct(sigma_0=1.62) * 40.1**2 + RBF(length_scale=0.00814)