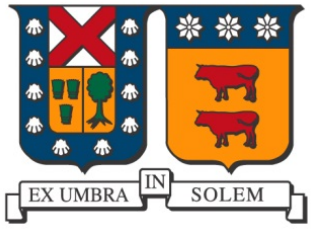
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



“RECOMENDACIONES DE GEOMETRÍAS PARA REVESTIMIENTOS DE MOLINOS EN GRAN MINERÍA BASADAS EN ALGORITMOS DE INTELIGENCIA ARTIFICIAL”

NICOLÁS IGNACIO DUCASEAU GALDAMES

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Oscar Ascencio Alarcón
Profesor Correferente: Luis Hevia Rodríguez

Febrero - 2025

DEDICATORIA

Le dedico este trabajo a mi familia y amigos, en especial a mi madre, quien siempre me apoyó y escuchó cuando tenía problemas. Gracias a todos por estar junto a mí en esta aventura, los quiero.

AGRADECIMIENTOS

En el transcurso de este trabajo pasé por muchas fases, desde la emoción y motivación hasta la frustración y el estrés. Muchas veces me sentí ansioso por recorrer este último tramo de mis estudios universitarios. Fue en estos momentos los cuales las personas que viven junto a mí me ayudaron, ya sea moralmente, dándome su aliento y apoyo, o simplemente con gestos como darme un chocolate o un café.

Por ello, empezaré agradeciendo a América Ducaseau y Rosa Ducaseau, mis dos abuelas, quienes me decían que descansara cuando me veían trabajando o me preparaban algo para comer cuando me veían concentrado. Gracias a ellas pude enfocarme solo en trabajar y seguir avanzando.

Sin embargo, a quien más quiero agradecer es a Rosa Galdames, mi madre, quien me apoyó en los buenos y malos momentos y fue la persona que más me motivó a seguir escribiendo esta memoria. De no ser por ella, no estaría aquí redactando estas palabras y mucho menos terminando este extenso trabajo. Gracias madre, tú fuiste la persona que más escuchó mis problemas y me dio consejos.

También quiero agradecer a todos mis amigos, con quienes conversé sobre mi memoria y que me alentaron a seguir avanzando.

Hacer esta memoria me enriqueció con mucha experiencia en un área que nunca había abordado que es la minería, por ello, también agradezco enormemente a mi profesor guía, don Óscar Ascencio Alarcón. Agradezco la oportunidad que me entregó de indagar en este desafío y pulir mis habilidades en el área de inteligencia artificial.

Finalmente, quiero agradecer a TEGA por la experiencia, y a David Gonzales por ser mi apoyo dentro de la empresa y estar atento a mis preguntas y dudas.

¡Muchas gracias a todos!

RESUMEN

Resumen—Esta memoria tuvo como propósito crear un modelo con técnicas de *machine learning* que permita generar geometrías de revestimientos de molinos SAG. Se realizó una manipulación de datos y características relevantes donde se consideró las proyecciones lineales y no lineales del desgaste esperado. El algoritmo propuesto esta basado en XGBoost y logró una puntuación en la métrica R^2 de 0.828 y un error del 15 % en el RMSE en comparación con los datos reales de desgaste. En consecuencia, la geometría dibujada del perfil se baso en estos resultados.

El modelo permitirá a la empresa contar con una herramienta confiable al momento considerar una duración objetivo de la vida útil de los revestimientos, evitando así campañas de supervisión desfasadas o innecesarias. Ademas, esto llevará a ahorrar en costes operacionales relacionados con la detención de los molinos.

Palabras Clave— Minería, Machine learning, Análisis de datos, Modelado, Molino SAG.

ABSTRACT

Abstract—The purpose of this work was to create a model using machine learning techniques to generate SAG mill liner geometries. A manipulation of relevant data and features was performed where linear and nonlinear projections of expected wear were considered. The proposed algorithm is based on XGBoost and achieved an R^2 metric score of 0.828 and an error of 15 % in the RMSE compared to the actual wear data. Consequently, the drawn profile geometry was based on these results.

The model will allow the company to have a reliable tool when considering a target life of the liners, thus avoiding outdated or unnecessary monitoring campaigns. In addition, this will save on operational costs related to mill shutdowns.

Keywords— Mining, Machine learning, Data analysis, Modeling, SAG mill.

Glosario

CODELCO: Corporación Nacional del Cobre de Chile.

IA: Inteligencia Artificial.

IQR: Interquartile Range (Rango Intercuartílico).

L1: Regularización Lasso (Least Absolute Shrinkage and Selection Operator).

L2: Regularización Ridge (Least Squares).

LSTM: Long Short-Term Memory (tipo de red neuronal recurrente).

ML: Machine Learning (Aprendizaje Automático).

Q (cuartil): Medida estadística que divide un conjunto de datos en cuatro partes iguales.

RNN: Recurrent Neural Network (Red Neuronal Recurrente).

RMSE: Root Mean Squared Error (Raíz del Error Cuadrático Medio).

SAG: Semi-Autogenous Grinding (Molienda Semi-Autógena).

XGBoost: Extreme Gradient Boosting (Algoritmo de aprendizaje automático basado en árboles de decisión).

UTFSM: Universidad técnica Federico Santa María.

ÍNDICE DE CONTENIDOS

RESUMEN iv

ABSTRACT iv

Glosario v

ÍNDICE DE FIGURAS viii

ÍNDICE DE TABLAS ix

INTRODUCCIÓN **1**

CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA **2**
1.1 Contexto . . . 2
1.2 El proceso de molienda . . . 2
1.3 Revestimientos . . . 4
1.4 Complejidad asociada . . . 7
1.5 Objetivo general . . . 8
1.6 Objetivos específicos . . . 8

CAPÍTULO 2: MARCO CONCEPTUAL **9**
2.1 Molinos y Optimización de la etapa de molienda . . . 9
2.2 Parámetros de diseño de revestimientos . . . 13
2.3 Aproximaciones con el uso de técnicas computacionales . . . 14
2.4 Inteligencia artificial . . . 16
2.4.1 Aprendizaje automático . . . 17
2.4.2 Árboles de decisión . . . 17
2.4.3 *Deep learning* . . . 19

CAPÍTULO 3: PROPUESTA DE SOLUCIÓN **20**
3.1 Antecedentes . . . 20
3.2 Composición de los datos . . . 22
3.2.1 Corrección e imputación de datos . . . 23
3.2.2 Determinación de *Outliers* . . . 27
3.3 Análisis por ciclos en campañas . . . 31
3.4 Modificación de datos categóricos . . . 31
3.5 Proyección del desgaste . . . 32
3.5.1 Ajuste por regresión lineal . . . 33
3.5.2 Ajuste por interpolación polinómica . . . 35
3.5.3 Identificación de características relevantes . . . 37
3.5.4 Lasso *L1* . . . 37
3.5.5 Ridge *L2* . . . 38
3.5.6 Tonelaje acumulado . . . 38

3.6 Modelado con técnicas de *machine learning* 40
3.6.1 XGBoost: *extreme Gradient Boosting* 40
3.6.2 Creando el modelo 43
3.6.3 Función objetivo 44
3.7 Geometría del perfil simple 48

CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN **49**
4.1 Fase 1: Modelo XGBoost 50
4.1.1 Modelo básico 50
4.1.2 Modelo con *tunning* simple 50
4.1.3 Modelo utilizando Optuna 51
4.2 Fase 2: Algoritmo de búsqueda de alturas del perfil simple 52
4.3 Fase 3: Dibujo de la geometría propuesta 55

CAPÍTULO 5: CONCLUSIONES **57**
5.1 Alcance y limitaciones 58
5.2 Recomendaciones para trabajos futuros 59

ANEXOS **61**

6 Visualización dataset **61**

7 Técnicas de regularización **62**

8 Manipulación de datos: diferencias entre espesores **65**

9 Entrenamiento XGBoost **67**

10 K-validation **70**

11 Gráficos de desgaste **70**

12 Algoritmo y función objetivo **72**

13 Geometría del perfil **73**

REFERENCIAS BIBLIOGRÁFICAS **75**

ÍNDICE DE FIGURAS

1 Fases de producción 1
2 Molino SAG. 3
3 Especificaciones técnicas de molinos SAG. 4
4 Visualización del funcionamiento interno de un molino SAG 5
5 Revestimiento híbrido Goma-Metal 6
6 Árbol del problema 7
7 Partes de un molino SAG 9
8 Efecto del diseño de revestimientos. 11
9 Efecto de la altura de los levantadores sobre el consumo energético del molino. 12
10 Visualización del funcionamiento interno de un molino SAG 13
11 Comportamiento del impacto de la carga sobre *lifters* y *liners* considerando diferentes parámetros operacionales. 15
12 Tipos de aprendizaje automático. 17
13 Diagrama común de árboles de decisión en *machine learning* 18
14 Jerarquía de la IA 19
15 Visualización de las primeras filas del dataset SAG N°3. 22
16 Resultados test *Shapiro-Wilk* sobre las características 25
17 Histograma de características de los datos. 26
18 Número de datos atípicos por columnas del dataset SAG N°3 utilizando rango intercuartílico (IQR). 27
19 Outliers de características de los datos en SAG N°3. 28
20 Comparación entre las distribuciones antes y después de imputar y escalar las columnas en el molino SAG 3. 30
21 Set de condiciones para generar los cambios de estado del molino. 31
22 Muestra el espesor inicial del revestimiento y el límite de retiro. 32
23 Dispersión de punto de revisión durante la campaña. 33
24 Regresión lineal sobre puntos de revisión durante la campaña. 34
25 Interpolación de puntos de revisión de espesor del revestimiento. 35
26 Proyección polinómica vs lineal. 36
27 Desgaste(mm) vs días en operación. 36
28 Función de costo de un modelo de regresión lineal. 37
29 Gráfico de los pesos de las características utilizando regularización $L1$ y $L2$. . 39
30 Comparación convergencia de características relevantes utilizando *lasso*, *ridge* y selección aleatoria. 39
31 Representación del funcionamiento de *Random Forest*. 41
32 Representación del funcionamiento de *Gradient Boosting* 42
33 Esquema simple de la estructura de entrenamiento y predicción del modelo . 44
34 Diagrama de flujo de la función objetivo 46
35 Funcionamiento de *Cross-validation* 49
36 Modelo simple 50
37 Modelo con *tunning* simple 51
38 Rango de búsqueda de hiperparámetros utilizando Optuna 51

39 Resultados predicción modelo utilizando Optuna para hiperparámetros . . . 52
40 Proyección de los desgastes 53
41 Consola durante una de las iteraciones de búsqueda del algoritmo 54
42 Diseño considerando 100 días de duración y un ángulo de ataque de 45°. . . 55
43 Diseño considerando 100 días de duración y un ángulo de ataque de 53°. . . 55
44 Diseño considerando 100 días de duración y un ángulo de ataque de 60°. . . 56

ÍNDICE DE TABLAS

1 Visualización de los datos vacíos de las mediciones sobre el SAG N°3. 24
2 Resultados RMSE modelo 52
3 Propuestas de alturas del perfil simple de las piezas del revestimiento para diferentes duraciones objetivo 54

INTRODUCCIÓN

La minería de cobre es una actividad esencial tanto en Chile, el mayor exportador mundial de este mineral, como a nivel global, debido a su papel crucial como materia prima para numerosos sectores económicos. Todo indica que su importancia se mantendrá en el futuro debido a su papel en tendencias emergentes como la transición energética hacia fuentes renovables, el uso de automóviles eléctricos, las tecnologías aplicadas a la medicina, la construcción y la industria química, entre muchas otras aplicaciones científicas y socioeconómicas de gran relevancia.

El proceso de producción de cobre se compone de varias grandes fases, desde la exploración y planificación hasta la fundición y el refinamiento. Entre ellas, una de las principales es la conminución, esto es, el proceso de reducción de tamaño de las rocas de mineral hasta convertirlo de un producto que pueda ser procesado de forma eficiente para la recuperación del cobre que contiene. Las fases de conminución en la minería de cobre son fundamentales para preparar el mineral a la siguiente etapa del proceso (típicamente la lixiviación, bio-lixiviación, o bien, la flotación, esto dependiendo del tipo del tipo de mineral) [Codelco, 2018] para separar el mineral de cobre de otros minerales y materiales los cuales no son deseados en el producto final. La eficiencia y la optimización de la conminución son cruciales para el rendimiento general y la rentabilidad de una operación minera de cobre.

Este proceso se compone de viarias fases, en particular el chancado (o trituración) y la molienda primaria y secundaria. Estas últimas se llevan a cabo utilizando molinos SAG y molinos de bolas, respectivamente. La optimización del proceso de molienda en la industria minera es crucial para garantizar la eficiencia y rentabilidad de las operaciones. En este contexto, los revestimientos de los molinos SAG y de bolas juegan un papel fundamental al proteger el equipo y al mismo tiempo influir en su rendimiento. Sin embargo, la generación óptima de la geometría de estos revestimientos sigue siendo una problemática con grandes parámetros y factores que hace que sea un reto importante para la ingeniería.

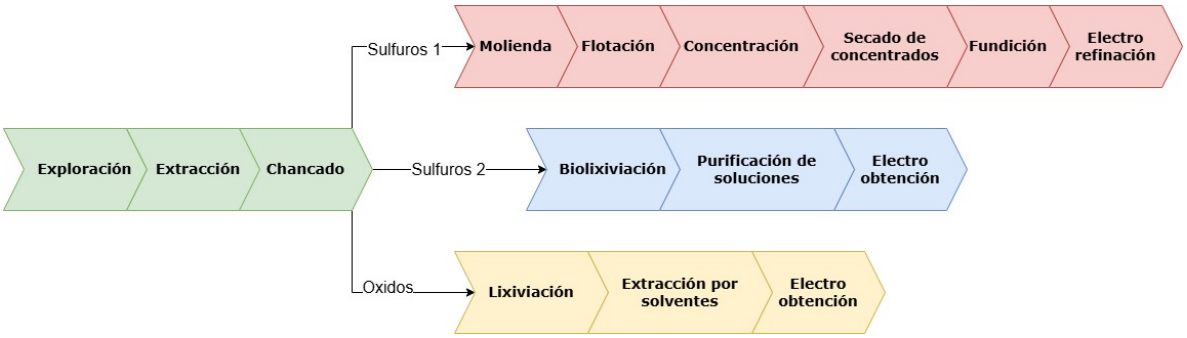


Figura 1: Fases de producción
Fuente: Elaboración propia.

CAPÍTULO 1
DEFINICIÓN DEL PROBLEMA

1.1. Contexto

Esta Memoria aborda el problema de la elección geométrica de revestimientos goma-metal para la empresa Tega Industries mediante el desarrollo de un modelo predictivo basado en técnicas de *machine learning*. La aplicación de estas técnicas tiene como objetivo proporcionar una herramienta robusta y precisa para recomendar una geometría de los revestimientos en función de ciertos parámetros y objetivos. Se busca contribuir a una gestión proactiva y eficiente de los recursos en las operaciones mineras de molienda. Este enfoque busca no solo mejorar la planificación de las campañas de mantenimiento, sino también reducir costos operativos y aumentar la disponibilidad de los equipos, contribuyendo así a la sostenibilidad y competitividad de la empresa minera.

1.2. El proceso de molienda

La molienda es una fase de producción que se enfoca en reducir el tamaño del material mineralizado. Esta ocurre justo luego de la etapa de trituración y el fin principal es obtener, con el uso de molinos, una pulpa compuesta por agua y reactivos, es decir, una pulpa del material rocoso.

Los molinos en la minería son grandes cilindros metálicos que giran para triturar el material extraído. En su interior, bolas de acero o barras impactan y friccionan el mineral, reduciendo su tamaño para su posterior procesamiento. Se utilizan tres tipos de molinos: los molinos de barra, bolas y SAG. Esta memoria se concentrará en los molinos SAG.

Los molinos SAG (semi-autógenos) trituran el material aprovechando las colisiones generadas por su rotación. Para mejorar la eficiencia del proceso, utilizan bolas de acero de 5 pulgadas. Estas bolas de acero se mezclan con el mineral en movimiento e impactan contra las rocas, fragmentándolas en partículas más pequeñas. Hay que notar que la molienda SAG es más afectada por el tipo de mineral que la molienda de bolas, esto por el uso del mismo material como medio de molienda. Esto repercute en la eficiencia de la producción y en la capacidad de tratamiento del material [Zeballos, 2003], siendo este proceso más efectivo en materiales de mayor dureza. En el caso contrario, los molinos de bola solo utilizan bolas de acero, lo cual los hace óptimos para materiales de menor dureza y se caracterizan por generar una granularidad mucho más fina. Por lo general, un porcentaje del material procesado en molinos SAG pasa a ser procesado en molino de bolas de forma que el producto final sea finamente triturado.

[1]Imagen extraída desde https://set.tegaindustrieschile.cl/ramp-up-de-los-molinos-sag/

Figura 2: Molino SAG.
Fuente: Tega Industries [1].

Los molinos son ampliamente utilizados en la industria minera debido a su capacidad de adaptación a distintas operaciones de procesamiento de minerales. Sus tamaños varían desde 4 metros de diámetro y 330 caballos de fuerza hasta grandes unidades de 14.5 metros de diámetro y 3.300 caballos de fuerza, brindando soluciones eficientes para diversas necesidades. En el caso de los molinos SAG, su uso frecuente permite omitir la segunda y tercera etapa de trituración (previas a la molienda). Esto ultimo se debe a que el material producido tiene una granularidad adecuada de 180 micrones [Codelco, 2019], lo que lo hace apto para la etapa de flotación y los procesos posteriores. Un punto importante es que al hacer uso de bolas de metal, estas pueden llegar a ocupar de un 12 % a un 15 % del volumen total del molino. Lo que lleva a que la cantidad de espacio para el material a procesar se vea reducido a poco más de un 85 % del volumen total. A partir de esto se concluye que es de suma importancia la eficiencia del espacio al interior del molino.

La molienda SAG simplifica el procesamiento de minerales permitiendo una mayor razón de reducción del tamaño del material. Este tipo de molienda también se caracteriza por una potencia de funcionamiento elevada, más que la de la molienda de bolas. En el caso de ambas moliendas, la rotación del equipo se logra gracias a la incorporación de motores, en donde se hace uso de particularmente de 2 tipos de motores: accionamiento mediante piñón de corona o accionamiento mediante motor anillo y cicloconversor.

Un punto muy importante es el uso de energía eléctrica de estos grandes equipos ya que, actualmente, alrededor de un 40 % del consumo eléctrico total de la producción proviene del proceso de molienda. El consumo de los molinos se mide en Mega watts (Mw), molinos grandes y de alta potencia pueden alcanzar un consumo de 22 Mw o incluso mayores [Industries, 2024a]. Aunque este consumo depende de las condiciones de operación como el

[2]Imagen extraída desde https://www.911metallurgist.com/metalurgia/molinos-sag-o-ag/

Especificaciones & Tamaños de Molinos SAG

Modelo	Diámetro (m)	Diámetro (ft)	Longitud (m)	Longitud (ft)	Tamaño Alimentac. (cm)	Tamaño Alimentac. (inch)	Motor Principal			
							Modelo	Potencia (kW)	Voltaje (V)	Peso (t)
911MPEZMJ4014	4	13	1.4	4.5	<35	<17	JR148-8	245	10000	75
911MPEZMJ4018	4	13	1.8	6	<35	<17	JR1410-8	320	10000	82
911MPEZMJ5518	5.5	18	1.8	6	<40	<20	TDMK800-36	800	10000	175
911MPEZMJ6522	6.5	21	2.2	7.2	<40	<20	TDM K1600-40	1600	10000	280
911MPEZMJ7525	7.5	24.5	2.5	8.2	<40	<20	TM2500-16	2500	10000	455
911MPEZMJ7528	7.5	24.5	2.8	9.2	<40	<20	TM2500-16	2500	10000	465

Especificaciones & Tamaños de Molinos AG

Figura 3: Especificaciones técnicas de molinos SAG.
Fuente: 911Metallurgist [2].

material que se esta procesando, las dimensiones del equipo, entre otros factores, de todas formas es un consumo relevante que causa grandes gastos económicos. Para hacer una comparación sobre este consumo, 22 Mw equivale al consumo promedio diario de 1000 hogares en Chile (aproximadamente 22 Kw por hogar/día)[Yañez.C., 2018], por lo que es necesario que el sector minero mejore cada año la eficiencia de funcionamiento en sus procesos. En la Figura 3 se aprecia una tabla con algunos modelos de molinos y sus respectivas características técnicas. Se destaca la potencia mostrada según modelo. Notar que no son molinos con dimensiones grandes.

1.3. Revestimientos

Los molinos están compuestos por una variedad de partes que permiten su funcionamiento. Estos equipos tienen una ingeniería muy compleja y nombrar todas sus partes y propósitos no es el tema central de este trabajo, pero es posible realizar una breve descripción de componentes esenciales para entregar más contexto [Lagos, 2018]:

- Chute: También denominado Trunnion de alimentación, es básicamente el conducto de entrada de la carga.
- Muñón: Es la base sobre la que esta el molino.
- Piñón y Corona: Mecanismos que permiten el movimiento del molino.
- Casco o Coraza: Es el tubo cilíndrico que gira. Dentro de este esta el revestimiento que lo protege de las fuerzas de impacto y abrasión producida por el material rocoso.
- Tapas: Utilizadas para la alimentación y descarga de material.

La pieza de interés es el revestimiento interior, el cual protege el *shell* o coraza del molino. Este revestimiento es importante en la protección del molino contra el desgaste debido a la

gran cantidad de colisiones del material pesado que ocurre durante su funcionamiento. Una dinámica típica se puede ver en la Figura 4, aquí se aprecia como el material procesado se mueve junto al molino, subiendo por las paredes interiores del cilindro para luego caer sobre sí mismo en la base. Este impacto genera una acción de trituración sobre el mineral.

Los primeros revestimientos que se inventaron usaban materiales rústicos como la piedra o incluso madera. Actualmente, para la construcción del revestimiento se utiliza comúnmente el metal, esto por su flexibilidad de diseño y también porque permite un mayor volumen de trabajo del molino al ser más delgados. El metal es un material con buenos atributos como resistencia a impactos y resistencia a la abrasión. Ademas, no es necesario que el revestimiento sea muy grueso, dando lugar así a una mayor cantidad de espacio dentro del cilindro. Sin embargo, uno de los problemas de este tipo de revestimiento metálico es el peso que aporta al conjunto, provocando un mayor consumo energético.

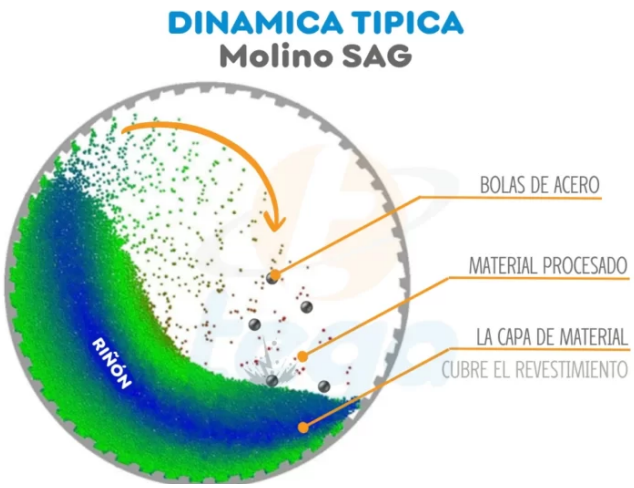


Figura 4: Visualización del funcionamiento interno de un molino SAG
Fuente: Tega Industries [3].

El uso de un material u otro para la construcción de un revestimiento en realidad depende de las condiciones de operación, procesamiento y manipulación del material a usar. El mercado ofrece una amplia gama de diseños para personalizar los revestimientos según su uso y propósito [Walker, 2010]. Algunos ejemplos de material para los *liners* son acero al cromo con bajo o alto contenido de carbono y revestimientos hierro de alto cromo. [Tomaschien, 2019].

Cabe destacar soluciones aparte del metal, en particular, los revestimientos híbridos de caucho-metal que ofrece Tega Industries. Se cita: *"Los revestimientos Dynaprime (goma-metal) han permitido a molinos de grandes dimensiones (36, 40 pies) acercarse rápidamente a la velocidad deseada (74 a 78 % Vc) mejorando la moliendabilidad y ,por consiguiente, el incremento del tonelaje de procesamiento por hora (TPH) en un menor periodo de tiempo. Una contribución directa en la disminución del ramp-up. La ventaja de instalar revestimientos*

[3]Imagen extraída desde https://set.tegaindustrieschile.cl/capacidad-de-ingenieria-tega/
[4]Imagen extraída desde https://set.tegaindustrieschile.cl/dyna-prime/

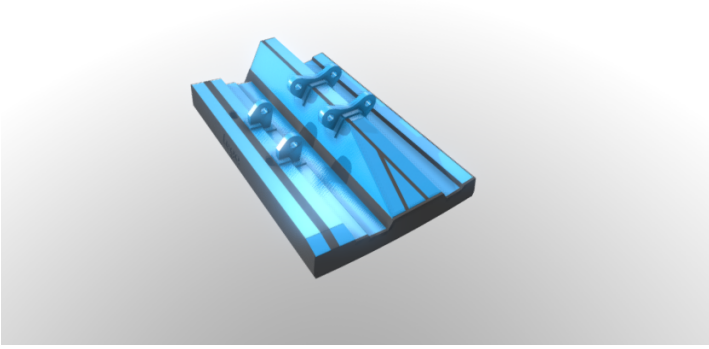


Figura 5: Revestimiento híbrido Goma-Metal
Fuente: Tega Industries [4].

Tega Dynaprime es que en nuestros productos toman más tiempo que el habitual en llegar a esa condición de desgaste en que el movimiento de carga y moliendabilidad disminuyen en forma significativa" [Industries, 2024b]. Notar que el *ramp-up* se refiere al tiempo que demora el molino en llegar a la velocidad deseada. Los molinos no pueden girar rápidamente de un momento a otro, este es un proceso que puede llegar a tomar días.

Actualmente existen una variedad de empresas especialistas en diseño de revestimientos e instalación de las mismos en los molinos. Algunos ejemplos de empresas que operan en Chile en esta área son Tega induestries, flsmidth y fourthane. Esto muestra que es un mercado amplio a nivel nacional e importante también.

Entonces el problema que se presenta es la elección de la geometría en el diseño de revestimientos de los molinos SAG. Ante esto, se deben realizar campañas de inspección de los molinos de forma regular para asegurar su funcionamiento y seguridad de uso. Si el molino sufre un desgaste excesivo, es necesario reemplazar las secciones que ya no pueden utilizarse, lo que genera un mayor gasto en operaciones y mantenimiento. Por ello, es fundamental proyectar la geometría óptima de los revestimientos, especialmente los híbridos, con el objetivo de evitar inspecciones anticipadas o fuera de plazo. De esta forma se reducen campañas de revisión innecesarias que aumentan los costos operacionales. Además, esto permitiría extender el ciclo de funcionamiento de los molinos sin interrumpir la producción, incrementando la cantidad de molinos activos y mejorando la eficiencia de la etapa de molienda.

Esta memoria busca ofrecer una recomendación del diseño geométrico de los revestimientos a través del uso de técnicas de inteligencia artificial y creación de modelos. De esta manera se busca optimizar la eficiencia y frecuencia en que se realizan las campañas en los sitios de molienda. Esto tiene un impacto principalmente en tres aristas:

- La primera siendo de intereses económicos para la empresa que utiliza estos molinos, reduciendo tiempos de *stand-by* y programando solo las revisiones necesarias. Aumentando de esta manera la rentabilidad operacional.

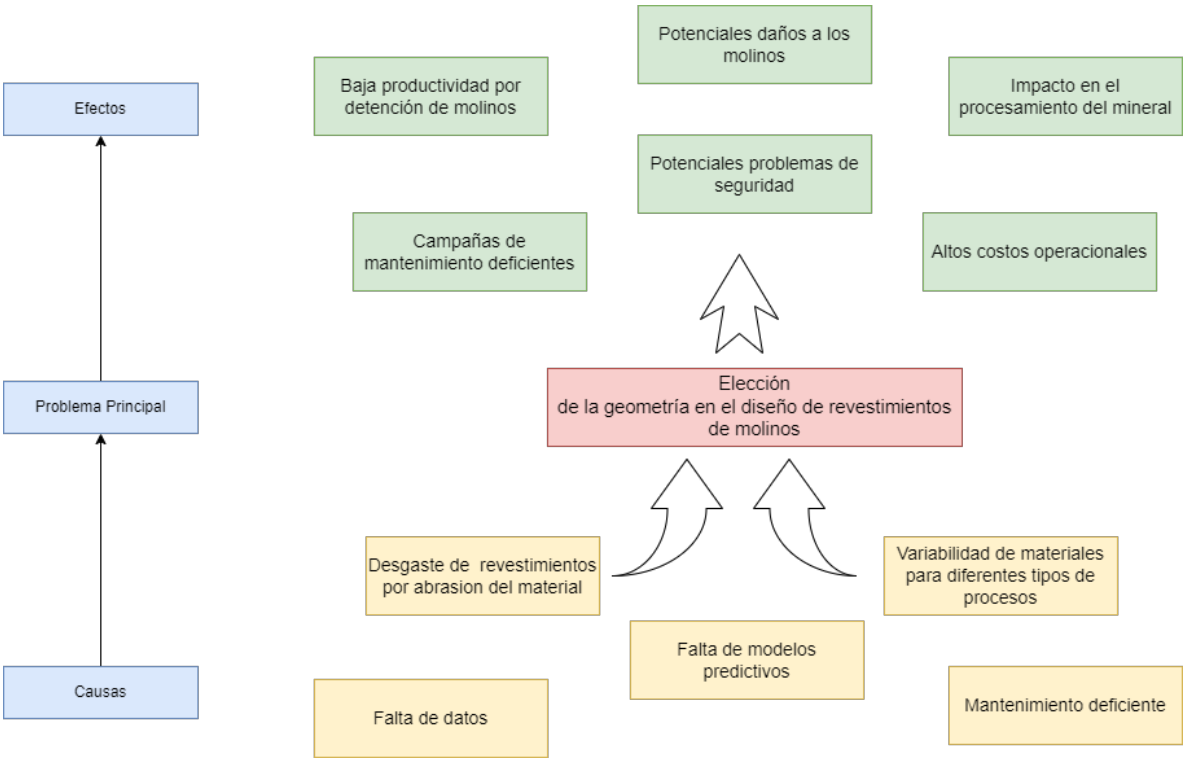


Figura 6: Árbol del problema
Fuente: Elaboración propia.

- La segunda es un impacto sobre los trabajadores que operan y mantienen los molinos, ofreciéndoles una forma de medición más precisa y segura a través del análisis de datos
- Finalmente, la tercera arista está relacionada con el cuidado ambiental. Un proceso de molienda más eficiente reduce los costos energéticos, el consumo de agua y la cantidad de desperdicios o escombros generados por mantenimientos y reemplazos inncesarios de revestimientos.

Pero el impacto más importante de esta memoria es a la empresa encargada de confeccionar e instalar estos revestimientos, ofreciendo una herramienta que permita realizar mejores mediciones, promocionar mejor su producto y ofrecer soluciones más seguras y confiables a sus potenciales clientes en el sector minero.

1.4. Complejidad asociada

Encontrar una solución satisfactoria a este problema ingenieril representa una variedad de desafíos asociados a la búsqueda y análisis de datos y a la correcta implementación de un

modelo basado en técnicas ML. Es propicio conocer en gran medida la nube de datos proporcionados por los sensores de la empresa, además, comprender las diferencias algorítmicas es fundamental para la creación de un modelo que resulte satisfactorio en un entorno de prueba real y con pocos datos históricos de referencia. La suma de todos estos conceptos convierte el tema en un desafío complejo y con varias aristas que se deben explorar, analizar y solucionar con aplicaciones creativas.

1.5. Objetivo general

Diseñar un modelo de aprendizaje automático para recomendar la geometría del diseño de revestimientos goma-metal en molinos SAG. Esto a través del análisis de variables operacionales y planteando una duración objetivo, mejorando así los procesos de toma de decisiones en el uso y mantenimiento industrial en molinos.

1.6. Objetivos específicos

- Resumir conceptos de minería, en particular, las secciones relacionadas con la molienda SAG, con un enfoque en el uso de revestimientos.
- Investigar conceptos y técnicas de machine learning y algoritmos candidatos para la generación del modelo de predicción adecuado.
- Diseñar un modelo que entregue recomendaciones de parámetros geométricos de revestimientos goma-metal de Tega industries.
- Medir efectividad del modelo entrenado con datos sensoriales provistos por Tega Industries, esto a través de la comparación de resultados obtenidos en pruebas sintéticas y el comportamiento del desgaste.
- Crear un software prototipo que permita generar gráficamente la proyección geométrica del perfil simple del revestimiento, esto considerando los resultados del entrenamiento del modelo propuesto.

CAPÍTULO 2
MARCO CONCEPTUAL

2.1. Molinos y Optimización de la etapa de molienda

Los molinos son grandes estructuras del área minera encargadas de triturar el mineral, actualmente están en uso tres tipos principales: molinos semiautogenos (desde ahora SAG), molinos de bola y molinos de barras, este trabajo esta principalmente enfocado en los molinos tipo SAG. Estas maquinarias poseen grandes dimensiones, en especial los molinos SAG, los cuales se caracterizan por el uso de material rocoso y bolas de acero para poder generar la granularidad requerida del material minado.

Los molinos están compuestos por una variedad de partes que permiten su funcionamiento, en este trabajo se estudiaran los revestimientos de los anillos internos del molinos (placas y *lifters*) y no se entrará en mayor detalle de otros elementos que componen el molino, sin embargo, en la figura 7 se observan la mayor parte de los componentes y secciones para una mayor comprensión de la máquina.

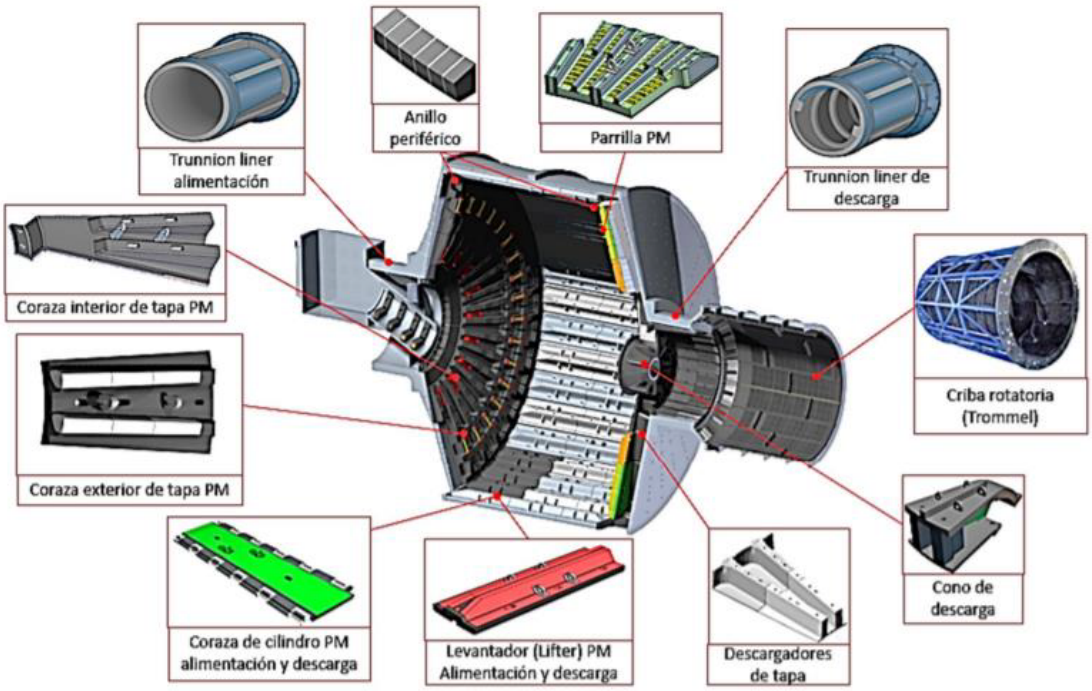


Figura 7: Partes de un molino SAG
Fuente: Estudio de falla y re-diseño de la estructura de un trommel para un molino SAG.

La molienda es un proceso que esta constantemente en evolución, el crecimiento actual de

diversas áreas tecnológicas y económicas promueve una mayor demanda de materiales, lo cual conlleva un incremento en la producción de minerales. Mayor producción significa mayor consumo energético, mayor desgaste de los *liners* y mayores gastos operacionales. La eficiencia de los procesos productivos es fundamental para el crecimiento de una empresa, por ello, el progreso de los revestimientos supone un punto clave para optimizar el consumo energético de los molinos, mejorar las eficacia de las campañas de mantenimiento industrial (las cuales suponen un gasto económico importante) y permitir un ritmo de producción constante sin detenciones innecesarias por reemplazo de *liners* en periodos no óptimos.

Para la optimización de los molinos SAG se deben de tener en cuenta una variedad de factores, entre estos, el tonelaje a utilizar, tipo de mineral a procesar, material de los revestimientos, entre otros. Existen diversos trabajos en este campo que buscan, a través de diferentes técnicas, aumentar la eficiencia de la molienda. Trabajos como el de Aguilar [Aguilar Titi, 2017] se enfocan en dar solución a los fenómenos de reflujo dentro del molino, esto haciendo un rediseño de los levantadores de carga de los revestimientos en la zona de descarga del material. Con esto, busca aumentar la eficacia de la evacuación de pulpa fuera del molino, de forma que el tonelaje a procesar aumente en igual medida, permitiendo un incremento en le eficiencia de producción.

Otro trabajo es el de Arratia [Henríquez, 2006] que propone un modelo matemático basado en mínimos cuadrados para poder hacer una estimación de la vida útil de revestimientos basado en factores como el tonelaje. Se pueden encontrar varias formas de atacar los problemas de eficiencia de la molienda.

De todas las partes que componen un molino los revestimientos son una de las piezas fundamentales de la molienda, sus funciones no solo radican en proteger al casco de los impactos sino que también se encargan de levantar la pulpa rocosa la cual luego cae e impacta sobre si misma. Es gracias a los levantadores o *lifters* que la carga puede alcanzar una altura adecuada que permita generar el impacto propicio para la trituración del material. Sin estos levantadores la carga solo se resbalaría por la superficie interna de la carcasa del molino lo que causaría un mayor gasto energético sin resultados satisfactorios [Rezaeizadeh *et al.*, 2010a].

Los levantadores de carga de los revestimientos poseen un perfil de diseño el cual tiene un impacto en el movimiento del material rocoso dentro del cilindro, diferencias de perfiles altos y bajos producen una transferencia de energía diferente a la carga. En el trabajo de Valderram y Magne [Valderrama y Magne, 1996] sobre el efecto del diseño en los levantadores de carga se confirma la influencia del diseño del perfil sobre el costo energético del molino, particularmente, se hizo un trabajo experimental con distintos ángulos de ataque del revestimientos en relación con la carcasa interna. Además, también se consideró la altura del perfil del *liner* y la velocidad critica alcanzada por el molino en el estudio. De esto último, se destacan resultados donde el ángulo de ataque y la velocidad critica inciden fuertemente en la potencia del molino, por otro lado, los efectos de la altura del perfil se ven fuertemente ligados a los otros dos factores anteriores. Se concluye que para un mismo ángulo de ataque y velocidad, la diferencias operacionales entre una altura del perfil de una a dos veces el diámetro del medio de molienda no posee un impacto significativo.

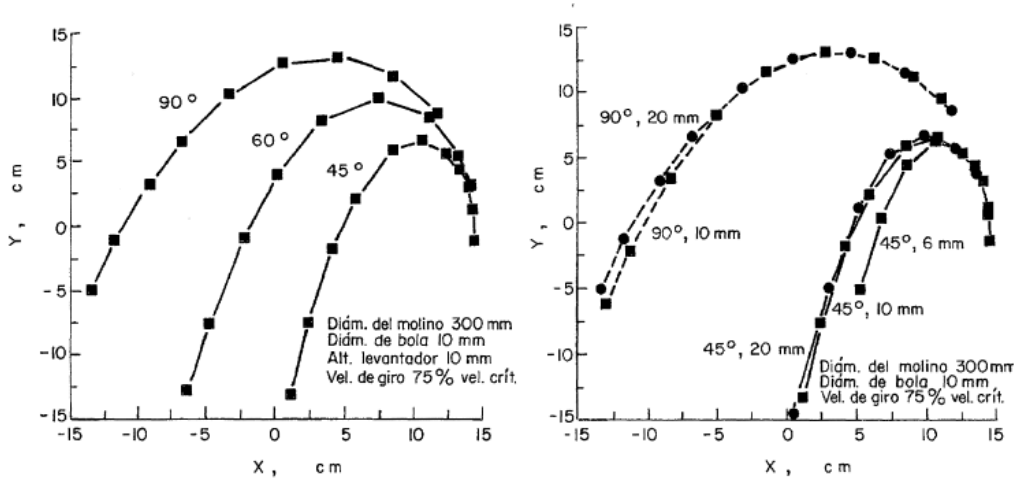


Figura 8: Efecto del diseño de revestimientos. A la izquierda se compara la trayectoria según el ángulo de ataque para una altura constante del perfil, a la derecha se modifica la altura del perfil.
Fuente: Efecto del diseño de revestimientos sobre el consumo de potencia en molienda.

Otro trabajo importante es el de Powell [Rezaeizadeh *et al.*, 2010a] en el cual se hace un estudio sobre el consumo energético en relación al diseño de los levantadores y la carga sobre estos durante el funcionamiento del molino. Haciendo uso de diferentes configuraciones en relación al número de *lifters*, espaciamiento entre *lifters* y velocidad del molino (VC), se obtuvo que la altura de los revestimientos tiene un efecto en el consumo energético. Se determinó que el consumo energético tiende a caer con un mayor número de levantadores de carga y una mayor altura del perfil de estos.

Dejando de lado el consumo energético, se deja en claro algunas conclusiones que son relevantes para el diseño geométrico de los revestimientos según las investigaciones previamente mencionadas:

- La variable más influyente en el funcionamiento de los levantadores de carga es el ángulo de ataque.
- Ángulos por debajo de 30° presentan un efecto claro del resbalamiento de la carga.
- A mayor ángulo de ataque, mayor es la importancia de la altura del perfil.
- A mayor velocidad de operación, mayor la importancia de la altura del perfil del levantador.
- A mayor número de levantadores, mayor la importancia de la altura del perfil del levantador.

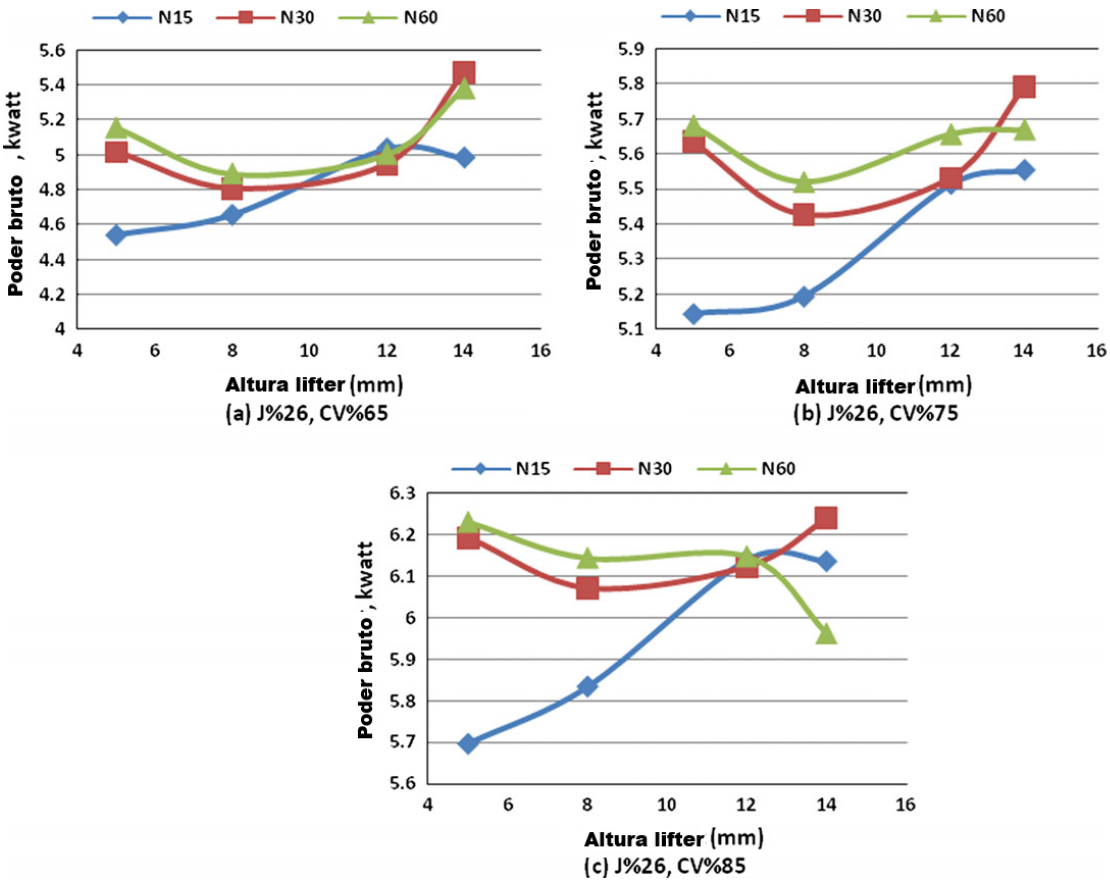


Figura 9: Efecto de la altura de los levantadores sobre el consumo energético del molino. Se evalúan diferentes cantidades de levantadores y rangos de velocidad de operación.
Fuente: *Experimental observations of lifter parameters and mill operation on power draw and liner impact loading*.

Esto da la idea inicial sobre la construcción de un revestimiento que sea óptimo, sin embargo, no se toma en cuenta el desgaste potencial de los revestimientos en función de los factores antes mencionados.

Para entender el porque se produce el desgaste, primero se debe comprender el mecanismo de acción que produce el molino sobre la carga. En este punto se destaca la dinámica de carga dentro del molino como se aprecia en la figura 10 donde se observan dos fenómenos importantes: movimientos de cascada (o riñon) y movimiento de catarata. El fenómeno de cascada ocurre a bajas velocidades en donde los medios de molienda se deslizan junto con el mineral, por otro lado, la catarata ocurre cuando la velocidad es suficiente para que los levantadores de carga eleven las bolas de metal las cuales, por efectos de gravedad, caen generando impactos que fracturan el mineral [Arroyo Murrugarra, 2018] [Rezaeizadeh *et al.*, 2010b].

En una variedad de trabajos desarrollados, se demuestra que la molienda ocurre principalmente en la zona de cascada por los efectos de abrasión y atrición. Por otro lado, los impactos

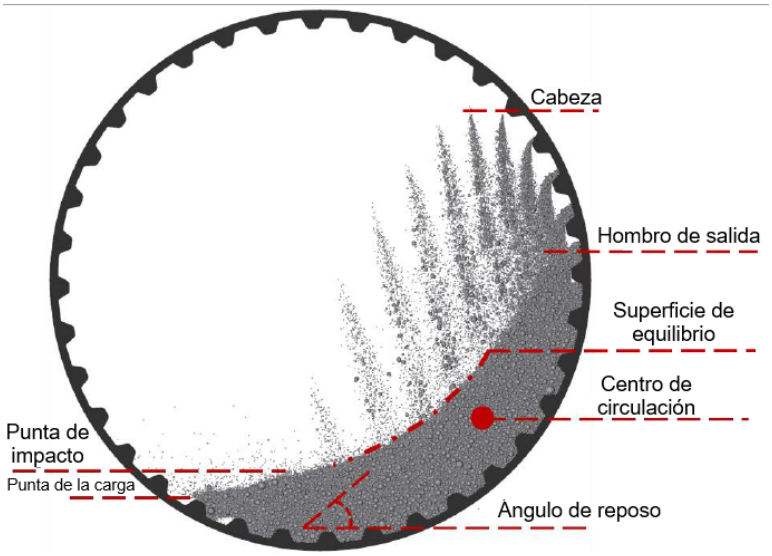


Figura 10: Visualización del funcionamiento interno de un molino SAG
Fuente: Optimización de la eficiencia energética en un molino semi-autógeno mediante el diseño de revestimientos utilizando simulaciones de elementos discretos.

tienen un mayor efecto de desgaste sobre los *liners* que sobre el proceso de molienda en si [Sherman y Rajamani, 1999], lo cual es consecuentemente un factor importante que incide sobre la vida útil de los revestimientos.

2.2. Parámetros de diseño de revestimientos

En el trabajo de powell [Rezaeizadeh *et al.*, 2010a] se estudian las repercusiones sobre los *lifters* y *liners* observando impactos de carga alto (catarata) he impactos de carga bajos (cascada). Para el primero existe la predominancia de un componente normal del impacto que induce el efecto de fractura sobre el material y para la segunda existe un componente tangencial que induce la abrasión sobre el revestimiento. Ambos efectos son contribuyentes al desgaste.

La figura 11 resume gráficamente los efectos de dichas observaciones comparando el impacto en newtons medidos en función de los parámetros estudiados.

A. **Efecto de la altura del *lifter***

- Se encontró que los levantadores con un perfil más alto permiten elevar partículas de mayor tamaño a una altura superior dentro del molino, generando así un impacto más fuerte al caer. Este tipo de movimiento de catarata induce el aplastamiento de los revestimientos, lo cual lleva a la fracturación del mismo. De forma contraria, al reducir la altura del perfil se genera una mayor abrasión.

B. **Efecto del número de *lifters***

- El número de levantadores se asocia a la frecuencia de impacto sobre los revestimientos, a mayor número de levantadores, mayor es la frecuencia de impactos por revolución del molino. Sin embargo, no se encontró relación con la potencia de impacto sobre los revestimientos.

C. **Efecto del porcentaje de carga del molino**

- Se determinó que un menor porcentaje de llenado del molino durante su operación conlleva una mayor carga de impactos sobre los revestimientos y levantadores. Esto ocurre principalmente debido a que los impactos se concentran en la zona del pie de carga, siendo así este un factor importante en el agrietamiento y fracturación de revestimientos que provoca su desgaste acelerado.

D. **Efecto de la velocidad del molino**

- Incrementar la velocidad critica del molino demuestra ser un factor importante a considerar. Mayores velocidades provocan que la carga colisione prontamente sobre los revestimientos debido al efecto de vuelo que proyecta una trayectoria más alta. También se indica un aumento de la frecuencia y potencia de los impactos.

Los revestimientos son el medio de transferencia energético a la carga y el desgaste en estos provoca deficiencias en el proceso de molienda. Por ello, diseñar de forma correcta los *liners* mejora la disponibilidad del molino, el movimiento de la carga y la vida útil durante su funcionamiento. En los trabajos de [Rezaeizadeh *et al.*, 2010b] y [Arroyo Murrugarra, 2018] se mencionan las variables a tener en cuenta en el diseño, aquí el ángulo del *lifter* y su altura cumplen un rol importante en relación al desgaste (puntos que ya se habían mencionado). También, en estos mismos estudios, se hace mención al comportamiento del molino cuando se analiza el espaciamiento entre revestimientos, lo cual influye en el movimiento del riñón de carga y modifica el comportamiento de la dinámica de carga según el porcentaje de llenado del molino.

2.3. Aproximaciones con el uso de técnicas computacionales

Volviendo al trabajo de Arratia [Henríquez, 2006], este propuso un algoritmo que permita conocer el desgaste operacional del revestimiento utilizando mediciones previas basadas en las kilo-toneladas(Ki) procesadas y el desgaste(Di). Este desgaste de los revestimientos se justifica como la forma en que se hace la transferencia energética de los motores a la carga del molino y la tasa de producción que se este utilizando. Su medición es realizada a partir de un escáner de ultrasonido de la integridad estructural de los revestimientos (placas y *lifters*) mapeando puntos muéstrales, considerando espesor y altura del perfil. Estipulando así el recambio cuando se alcance el 75 % a 80 % del desgaste permitido.

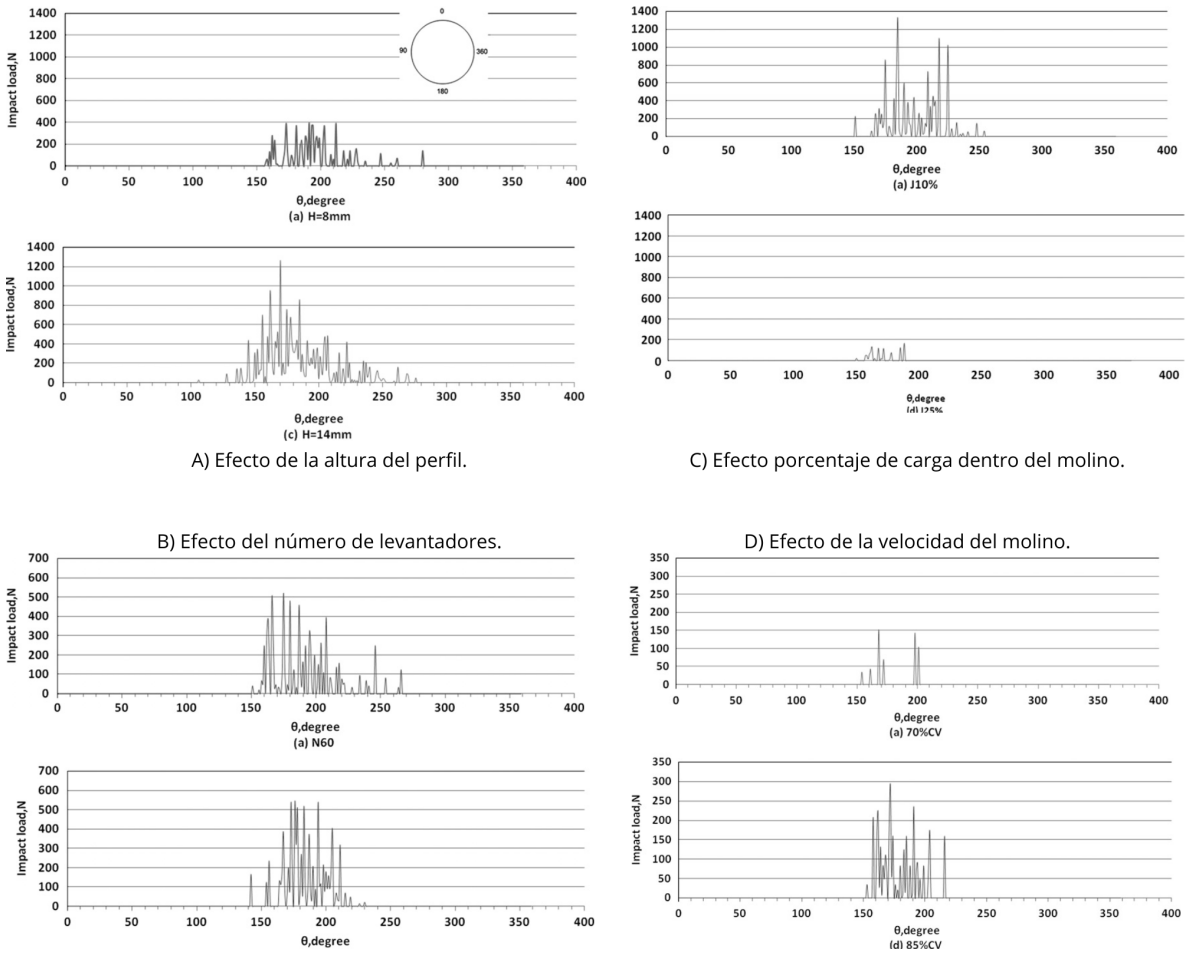


Figura 11: Comportamiento del impacto de la carga sobre *lifters* y *liners* considerando diferentes parámetros operacionales.
Fuente: *Experimental observations of lifter parameters and mill operation on power draw and liner impact loading*.

Arratia recalca la importancia de *la metodología de construcción de modelos matemáticos* en cuatro tópicos importantes que son:

- Definición del problema
- Formulación de modelo
- Hipótesis y teorías matemáticas
- Generación del programa computacional

Apoyado en esos puntos, el modelo creado por Arratia se basa en una regresión lineal que predice el comportamiento del desgaste de los revestimientos y su vida útil, esto realizan-

do un análisis de dispersión de los datos e identificando la relación entre los atributos de kilotonelada y desgaste.

Para terminar, el trabajo de [Rezaeizadeh *et al.*, 2010b] propone que la predicción del desgaste del perfil de los revestimientos se puede conseguir realizando un análisis de los parámetros de ángulo de abrasión, coeficiente de fricción, dureza del revestimiento, presión y relación de velocidad. Aquí también se confirma la importancia del roce sobre el material por parte de la carga y de los impactos generados sobre la misma por la acción de catarata en el pie de carga. Un punto interesante de la descripción del desgaste propuesta en su estudio es que este desgaste esta relacionado con la velocidad relativa entre la carcasa del molino y las partículas que se mueven durante el funcionamiento.

2.4. Inteligencia artificial

La tecnología ha avanzado a grandes pasos durante los últimos 50 años. Los computadores han pasado de ser complejas y grandes máquinas destinadas a usos específicos en la ciencia, a pequeños y portátiles dispositivos que, de una u otra forma, todo el mundo utiliza en su día a día. Sin embargo hay un punto dentro de toda esta área que estos últimos años ha estado tomando relevancia: la **inteligencia artificial**.

La inteligencia artificial podría definirse de varias maneras, Rouhiainen en su libro [Rouhiainen, 2018] lo define como:

> *'...la IA es la capacidad de las máquinas para usar algoritmos, aprender de los datos y utilizar lo aprendido en la toma de decisiones tal y como lo haría un ser humano...'*

Personalmente me gusta entender a la IA como una herramienta que nos permite solucionar problemas que involucren un número elevado de datos los cuales a nosotros, como seres humanos, nos seria imposible analizar. La utilidad práctica para la IA es gigantesca, desde el procesamiento de imágenes y generación de las mismas, hasta inferencia de datos futuros a partir de patrones pasados. El abanico de aplicaciones es ilimitado y cada día surgen más usos.

Ahora, dentro de esta área existen variedades de formas de afrontar un problema y cada problema requiere el uso de una técnica u otra, pero la base en común de la IA es el aprendizaje automático o *machine learning* en ingles.

2.4.1. Aprendizaje automático

La idea de nombrar esta tecnología como 'inteligencia artificial' proviene del objetivo de crear agentes que sean, en efecto, 'inteligentes'. Pero ¿a que nos referimos con inteligencia? esta es una pregunta difícil de responder porque tampoco existe un consenso claro de que es la inteligencia. Sin embargo, para efectos prácticos lo que realmente nos interesa es que la computadora sea capaz de 'emular' ciertas habilidades atribuidas a los humanos, y lo que más nos representa es nuestra capacidad de **aprender**.

Entonces lo que queremos es que la máquina sea capaz de:

- Analizar datos.
- Reconocer patrones.
- Aprender dichos patrones sin (o con poca) intervención humana.
- Inferir resultados en base a lo que aprendió.

Con el uso del aprendizaje automático grandes compañías como *Google* utilizan algoritmos que son capaces de analizar el comportamiento del usuario y sugerir publicidad en base a patrones previamente vistos por el algoritmo, esto sin que nadie en particular este observando al usuario y todo se hace de forma automática. De esta idea de aprendizaje surgen tres ramas importantes en 'como aprende' la máquina, en la figura 12 se observa de forma sencilla los tipos de aprendizaje que existen.

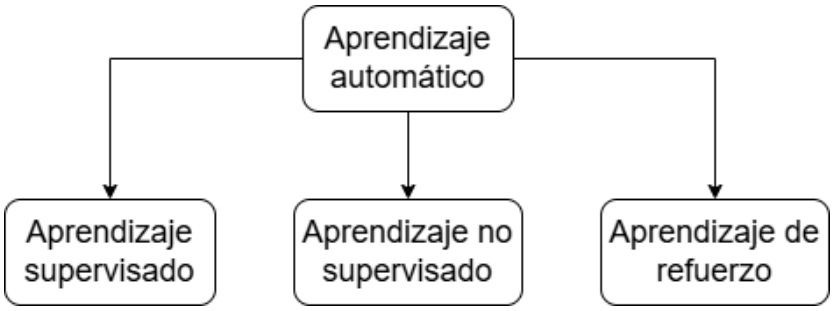


Figura 12: Tipos de aprendizaje automático.
Fuente: Elaboración propia

2.4.2. Árboles de decisión

Una de las formas más comunes de aprendizaje automático supervisado es el uso de árboles de decisión, los cuales son una forma muy inteligente de realizar predicciones. Esta configuración es fácil de entender si se piensa en una diagrama en el cual se tienen varios tipos de

nodos: un nodo raíz, nodos internos y varios nodos ”hoja”. Tal como su nombre lo indica, la forma que suelen tener estos diagramas es la de un árbol que se expande en varias ramas.

Estos árboles funcionan con condiciones sencillas *booleanas* para manejar los datos, de forma que un porcentaje de la información se mueve hacia un lado y lo que resta hacia el otro. Esto último es realmente útil para poder categorizar los datos según los atributos que definan el output.

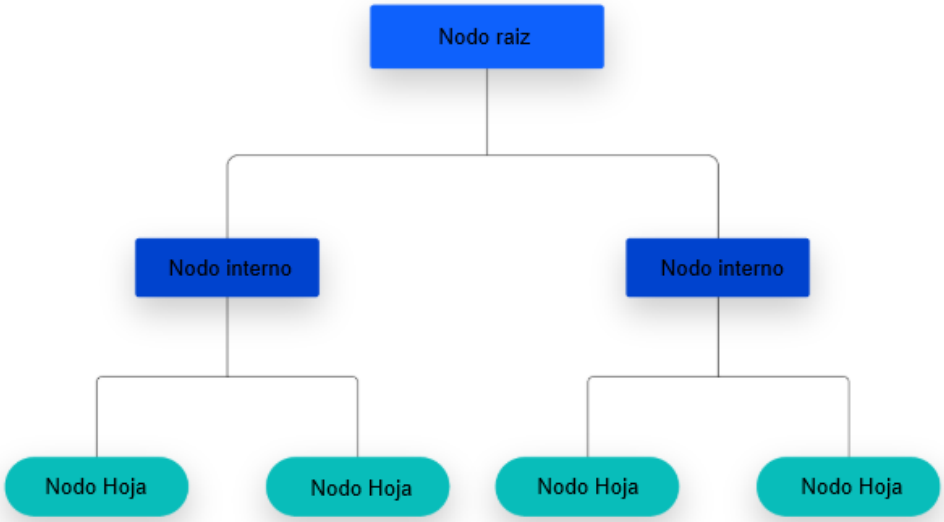


Figura 13: Diagrama común de árboles de decisión en *machine learning*
Fuente: ¿Qué es un árbol de decisión? [IBM, 2024b]

Los árboles de decisión son una forma fácil he intuitiva de realizar predicciones en base a varias características de entrada, sin embargo son muy susceptibles al *overfitting* donde las predicciones se adaptan demasiado bien a los datos de entrenamiento lo cual impide que el modelo sea capaz de generalizar. Para solucionar esto, se realiza la *poda* en la cual se eliminan las ramas donde existan características con baja importancia [IBM, 2024b], esto utilizando métodos de regularización. También es factible reducir este sobreajuste restringiendo la profundidad del árbol, lo cual reduce la complejidad del árbol quitando características que se utilizan en el modelo.

A partir de este modelo básico de entrenamiento supervisado surgen otros métodos más complejos como *random forest* y *XGBoost*.

2.4.3. *Deep learning*

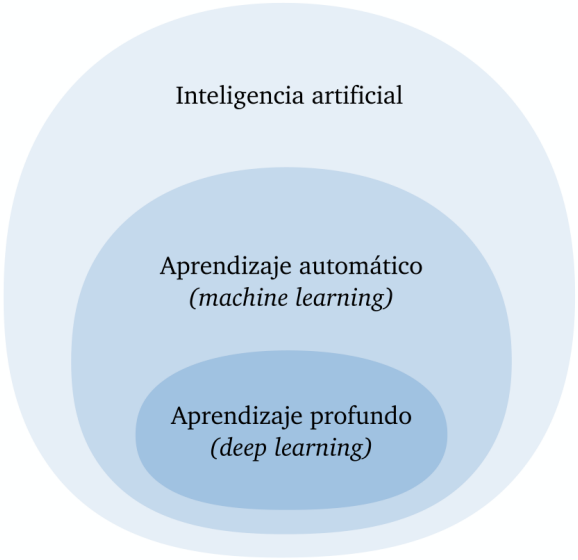


Figura 14: Jerarquía de la IA .
Fuente: Aplicación de técnicas de aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia en sistemas de ingeniería [González-Muñiz, 2023]

Algunos problemas pueden ser muy complejos de resolver, ya sea por su contexto o por la gran cantidad de datos que deben procesarse. En respuesta a ello, surge el *deep learning* o aprendizaje profundo, el cual es una rama del aprendizaje automático. La característica principal de este tipo de aprendizaje es el uso de **redes neuronales** que emulan el comportamiento de las neuronas del cerebro, estas neuronas o nodos están interconectados entre si y son capaces de transferirse información. En la figura 14 se observa la jerarquía del aprendizaje automático.

La construcción de un modelo con esta técnica se basa en capas. Una red neuronal básica se compone de tres capas [Amazon, 2024]:

- Capa de entrada: Entrada de información que luego es analizada y clasificada por los nodos.
- Capa oculta: Transforma y procesa datos.
- Capa de salida: Proporciona el resultado en función del contexto de lo que se desea obtener.

Este trabajo se enfocara en el uso de árboles de decisión para afrontar el problema, en particular, el modelo de regresión llamado **XGBoost**.

CAPÍTULO 3
PROPUESTA DE SOLUCIÓN

La propuesta de solución se centra en la creación de un prototipo simple de software que permita generar recomendaciones geométricas simples de los *liners* en base a un modelo entrenado con datos proporcionados por la empresa. Para ello, se consideraron los puntos fundamentales antes expuestos de trabajos anteriores sobre el comportamiento de la dinámica de carga en relación a los parámetros de entrada y el desgaste generado sobre el revestimiento. El modelo se entrenará para determinar la altura óptima del perfil según una duración objetivo, y el dibujo geométrico considerará este resultado junto con diferentes ángulos de ataque.

Esta memoria esta enfocada en el diseño de revestimientos de goma-metal para la empresa *Tega Industries*. Esta última entregó una variedad de datos que son fundamentales para el desarrollo del modelo basado técnicas de inteligencia artificial.

3.1. Antecedentes

Se debe realizar un procesamiento de datos los cuales son entregados por la empresa, los cuales constan de atributos relacionados con la operación del molino. A continuación se resumen las definiciones de los parámetros (proporcionados por la empresa) que se tienen a disposición para generar el análisis.

- Work Index del mineral (kWh/ton): Índice de trabajo del mineral, que indica la energía necesaria para reducir el tamaño del mineral.
- Tamaño o granulometría de alimentación: Tamaño de las partículas del material de alimentación al molino, medido en pulgadas.
- Tamaño o granulometría de descarga: Tamaño de las partículas del material de descarga del molino, medido en pulgadas.
- Alimentación fresca (tph): Material nuevo que ingresa al molino, en toneladas por hora.
- Generación Pebbles (TPH): Generación de material de tamaño intermedio (pebbles), tamaño imposible de moler en un SAG, requiere de un chancador de pebbles para reducir su tamaño. Medido en toneladas por hora.
- Recirculación (TPH): Cantidad de material recirculado dentro del molino SAG, medida en toneladas por hora.

- Potencia consumida (kW/h): Energía utilizada por el molino, medida en kilovatios por hora.
- Potencia instalada (kW/h): Máxima energía a la que puede operar el molino por diseño del fabricante, en kilovatios por hora.
- Velocidad de giro (rpm): Velocidad de rotación del molino en revoluciones por minuto.
- Sentido de giro: Dirección de rotación del molino.
- Tamaño y consumo de bolas (tph): Tamaño de las bolas de molienda y la tasa de consumo de estas bolas en pulgadas y toneladas por hora, respectivamente.
- CE o CEE: Consumo específico de energía o energía específica, en kWh/t.
- Nivel del molino o Jc (%): Porcentaje del volumen del molino ocupado por el material de molienda.
- Nivel de bolas o Jb (%): Porcentaje del volumen del molino ocupado por las bolas de molienda.
- Faro: Sistema de monitoreo en el molino que permite escanearlo y obtener una nube de puntos.
- Sólidos (%): Porcentaje de sólidos en la suspensión dentro del molino.
- Presión sobre los descansos (psi): Presión ejercida sobre los descansos o cojinetes del molino, medida en libras por pulgada cuadrada (psi), kilopascales (kPa) o bares (bar).
- Diesel: Uso de combustible diésel en el proceso.
- Xantanto: Reactivo utilizado en la flotación de minerales.
- Matcol: Otro reactivo utilizado en la flotación de minerales.
- Nalco: Producto químico usado en el tratamiento de agua y procesos industriales.
- MX: Otro tipo de reactivo o condición del proceso.
- pH: Medida de la acidez o alcalinidad de la suspensión.
- AxB: Parámetro de dureza de la roca que se usa en el modelado y diseño de circuitos de molienda.

Puntos importantes a considerar respecto al manejo de los datos son:

A. Formato de archivos .xlsx, no hubo posibilidad de un acceso remoto a datos.

B. Los datos están en tablas donde cada columna identifica un atributo operativo del molino.

C. Todos los datos son cuantitativos, quitando excepciones:

   - Fecha.
   - Orientación de giro del molino.
   - Datos faltantes, corruptos o Nan.

Con respecto al entorno de ejecución:

- Se utiliza Colab con Python 3.10.12
- También se hizo uso de un entorno local en Ubuntu 20.04 para realizar ciertas pruebas.

3.2. Composición de los datos

El *dataset* presenta datos de dos molinos SAG, todos ubicados en la minera *Pelambres* al norte de nuestro país. Estos datos serán la base del futuro modelo para encontrar las dimensiones del perfil simple el revestimiento. Los datos proporcionados constan de 2058 filas y 27 columnas, donde las dos primeras columnas presentan las fechas de toma de datos. Todas las demás columnas son principalmente datos cuantitativos derivados de mediciones con los sensores incorporados del molino.

	SAG3	Alimentación fresca (tph)	Velocidad de giro (rpm)	Nivel del molino o Jc (%)	pH
0	2020-12-31 19:00:00	2496.7990202761716	9.490755533475081	27.050905970683438	9.000899168333767
1	2021-01-01 19:00:00	2597.504072876253	9.566994319112256	27.19558972767984	9.003933989433913
2	2021-01-02 19:00:00	2033.2997014659197	9.29271529555718	27.22842257500152	8.961196821225325
3	2021-01-03 19:00:00	2152.1957342510714	9.596937119261113	27.20827488070786	8.899953429581132
4	2021-01-04 19:00:00	2165.947108921838	9.045777185730353	27.116402602981164	8.882227487758025
5	2021-01-05 19:00:00	2340.2283326238194	9.685993115451915	27.612380920577287	9.03641146785085
6	2021-01-06 19:00:00	2596.2313419491848	9.69772096240837	27.308207723437306	9.087611806373017
7	2021-01-07 19:00:00	2308.0773200023286	9.788758343110796	27.640748750774417	9.110916380603847
8	2021-01-08 19:00:00	2157.9430411681424	9.139936551368535	26.834474340115467	9.025631112899768
9	2021-01-09 19:00:00	2681.1230235286116	9.679795360877547	26.418701491987534	9.071206057990624

Figura 15: Visualización de las primeras filas del dataset SAG N°3. Solo se incluyeron algunas columnas para la representación.
Fuente: Elaboración propia.

De momento la empresa solo ha registrado data representativa del funcionamiento del molino, pero no ha presentado usos prácticos con estos datos lo cual es el objetivo de realizar este trabajo. Exponer una propuesta que incluya el análisis de datos registrados por los sensores también favorecerá otros posibles usos del *dataset* que la empresa pueda necesitar a futuro.

Conocer la distribución de los datos es importante para seguir con los procedimientos de corrección e imputación. Realizando un análisis simple de las columnas, se eliminan momentáneamente las dos primeras (correspondientes a las fechas donde se realizó la toma de datos) ya que no aportan peso estadístico al análisis pues los datos ya se pueden trabajar de forma ordenada por día (fila a fila).

3.2.1. Corrección e imputación de datos

Existen diversos problemas en los datasets que deben ser resueltos y, por lo general, esta etapa es una de las que más tiempo requiere, ya que implica un análisis detallado. Para comenzar, es importante realizar una primera revisión básica del conjunto de datos. Es fundamental identificar cuántos elementos problemáticos hay por columna, comenzando por los datos vacíos o nulos, los cuales suelen deberse a errores de medición del sensor o errores humanos.

En la tabla 1 se representan los valores nulos solo del SAG 3. Es reconocible que el *dataset* tiene una cantidad no menor de datos que se deben eliminar o imputar. Considerando que solo se disponen de 2058 filas, la imputación es la mejor opción en estos casos ya que no habrá pérdida de datos en la tabla. El procedimiento a que se llevo a cabo es el siguiente:

A. Se modifico el *dataset* para que conserven solo los valores numéricos. Columnas categóricas como el sentido de giro se dejaran para generar columnas representativas más adelante, de momento lo vital es tener un estado visual en gráficos de barra de la distribución las columnas.

B. Columnas como 'CE bolas', 'MX' y 'matcol' tienen una gran cantidad datos vacíos como para que sean de utilidad, por lo que se eliminan directamente. Sin embargo, esta no es la única razón, también se determinó a través de reuniones con Tega que los datos correspondientes a las características de 'Diesel', 'Xantanto', 'Matcol', 'Nalco', 'MX' y 'Faro', no poseen relevancia significativa en el desgaste acumulado.

	Valores Nulos
SAG3	0
Unnamed: 1	0
Work Index del mineral (kWh/ton)	41
Tamaño o granulometría de alimentación (")	11
Tamaño o granulometría de descarga (")	0
Alimentación fresca (tph)	64
Generación Pebbles (TPH)	9
Recirculación (TPH)	114
Potencia consumida e instalada (kW/h)	48
Velocidad de giro (rpm)	48
Sentido de giro	0
Tamaño y consumo de bolas (", tph)	0
CE bolas	1095
Nivel del molino o Jc (%)	0
Nivel de bolas o Jb (%)	1
Faro	0
Sólidos (%)	64
Presión sobre los descansos (psi)	10
Diesel	15
Xantanto	51
Matcol	17
Nalco	27
MX	556
pH	7

Tabla 1: Visualización de los datos vacíos de las mediciones sobre el SAG N°3.
Fuente: Elaboración propia.

Con estos cambios quedan 14 atributos relevantes, de estos es pertinente conocer la distribución que poseen. Un conjunto de datos puede seguir una variedad de distribuciones, como por ejemplo, una distribución normal. Las distribuciones de cada columna pueden afectar al desempeño del modelo.

- Características normalmente distribuidas podrían dominar el aprendizaje del modelo, disminuyendo la relevancia de características con otras distribuciones.
- Escalar las columnas garantiza que todas las características sean comparables en el modelo, pues se le asigna la misma importancia relativa, preservando así la forma de la distribución de cada atributo.
- Esto último facilita el procesamiento de los datos para el modelo ya que los gráficos estarán ajustados a una escala similar.

Los gráficos de la figura 17 muestra la distribuciones de cada característica. Sin embargo, identificar el tipo de distribución de forma visual no es suficiente. Para saber el tipo de distribución que tiene cada atributo se utilizó la prueba de normalidad *Shapiro-Wilk*. En esta

prueba, si el valor resultante es menor a un $\alpha = 0{,}05$ entonces la característica evaluada no sigue un patrón de distribución normal. En la siguiente figura se observan los resultados, curiosamente ninguna característica sigue una distribución normal, pese a que a simple vista si parecía que algunas lo hacían.

```
Columna 'Work Index del mineral (kWh/ton)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Tamaño o granulometría de alimentación (")' → Distribución No Normal → Usando MinMaxScaler
Columna 'Tamaño o granulometría de descarga (")' → Distribución No Normal → Usando MinMaxScaler
Columna 'Alimentación fresca (tph)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Generación Pebbles (TPH)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Recirculación (TPH)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Potencia consumida e instalada (kW/h)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Velocidad de giro (rpm)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Nivel del molino o Jc (%)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Nivel de bolas o Jb (%)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Sólidos (%)' → Distribución No Normal → Usando MinMaxScaler
Columna 'Presión sobre los descansos (psi)' → Distribución No Normal → Usando MinMaxScaler
Columna 'pH' → Distribución No Normal → Usando MinMaxScaler
```

Figura 16: Resultados test *Shapiro-Wilk* sobre las características
Fuente: Elaboración propia.

Considerando estos resultados de la prueba, se decidió utilizar *MinMaxScaler* luego durante el entrenamiento del modelo. Escalar los datos de esta forma tiene como propósito preservar la interpretabilidad de los datos considerando las diferencias en magnitud entre algunas características.

Un dato importante a resaltar en la creación de estos gráficos es que se consideraron las zonas de fin de campaña y las zonas de supervisión de los revestimientos. Dependiendo de si se trata de una u otra zona, el comportamiento de los datos puede mostrar valores vacíos o nulos. Por ello, estos datos fueron imputados mediante la media o por ceros, según lo siguiente:

- El estado de funcionamiento del molino es cíclico, es decir, cada ciclo representa el uso de un revestimiento antes de que este sea cambiado por otro nuevo (*Lifter* y *placas*).
- En cada ciclo de funcionamiento existen de 3 a 9 revisiones de mantenimiento en las cuales el molino SAG debe ser detenido, por lo cual no se registran datos. Esto justifica la existencia de datos vacíos o nulos.
- Lo anterior también sucede al reemplazar los revestimientos al final de cada campaña, proceso que suele durar de 4 a 7 días.
- Valores nulos que se encuentren dentro del rango de estas condiciones fueron imputados por ceros. Otros datos vacíos que no cumplían estas condiciones fueron imputados por la media.

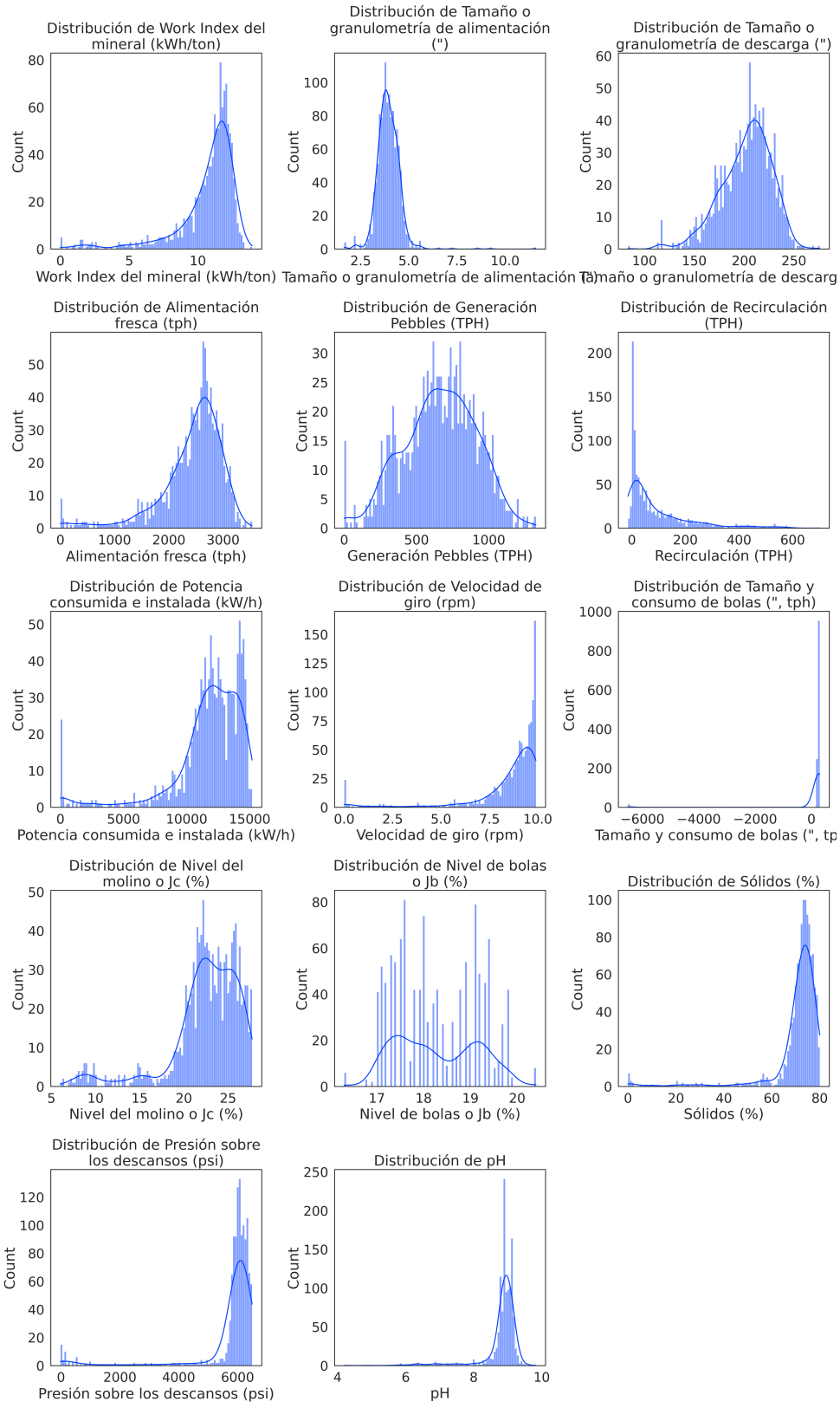


Figura 17: Histograma de características de los datos.
Fuente: Elaboración propia.

3.2.2. Determinación de *Outliers*

Los *outliers* son útiles en el análisis de datos, ya que se utilizan para detectar fallos en sistemas observando mediciones que están fuera del rango correcto de funcionamiento. Considerando esto, la existencia de *outliers* puede interferir en la creación de un modelo predictivo robusto, por lo que es necesario identificar estos datos y ser capaz de realizar las correcciones necesarias. Comúnmente, al momento de manipular esta información hay que tener en cuenta si se cumplen los siguientes criterios:

- Errores de entrada.
- Lecturas erróneas.
- Datos reales pero poco usuales.

La figura 19 sintetiza la distribución de los datos utilizando *Boxplot* o diagrama de caja el cual utiliza cuartiles para comprobar la distribución de los datos. El rectángulo que se extiende de izquierda a derecha indica los datos del 25 % (Q1) al 75 %(Q3) respectivamente del total de datos. Luego, las lineas que se extienden fuera del rectángulo se conocen como 'bigotes de gato' y reflejan el rango en el que caen la mayoría de los datos comunes bajo el 25 % y sobre el 75 %. Por último, los puntos exteriores a los bigotes son los *outliers* o valores atípicos [Siqueira, 2023].

```
En la columna Work Index del mineral (kWh/ton) hay 109 outliers
En la columna Tamaño o granulometría de alimentación (") hay 41 outliers
En la columna Tamaño o granulometría de descarga (") hay 20 outliers
En la columna Alimentación fresca (tph) hay 110 outliers
En la columna Generación Pebbles (TPH) hay 0 outliers
En la columna Recirculación (TPH) hay 96 outliers
En la columna Potencia consumida e instalada (kW/h) hay 112 outliers
En la columna Velocidad de giro (rpm) hay 134 outliers
En la columna Nivel del molino o Jc (%) hay 77 outliers
En la columna Nivel de bolas o Jb (%) hay 0 outliers
En la columna Sólidos (%) hay 147 outliers
En la columna Presión sobre los descansos (psi) hay 124 outliers
En la columna pH hay 104 outliers
```

Figura 18: Número de datos atípicos por columnas del dataset SAG N°3 utilizando rango intercuartílico (IQR).

Fuente: Elaboración propia.

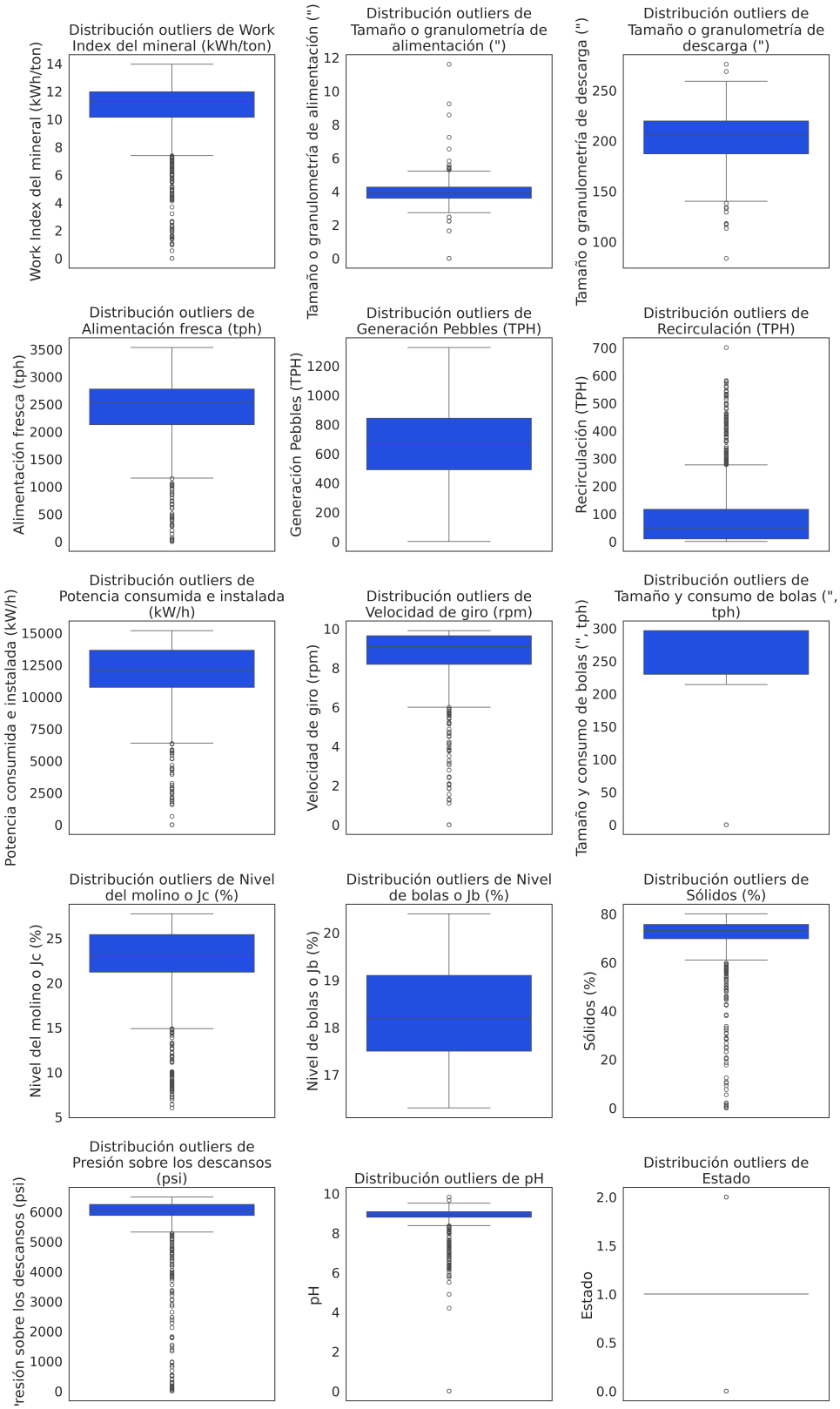


Figura 19: Outliers de características de los datos en SAG N°3.
Fuente: Elaboración propia.

Para entender y corregir estos datos se realizaron reuniones con la sección de innovación y metalurgia de Tega. Aquí se hizo un análisis en conjunto para introducir algunos rangos operativos normales del funcionamiento de los molinos SAG. Se discutieron posibles causas de datos erróneos e interpretación de datos inusuales, tales como datos negativos, los cuales no tenían un sentido lógico dentro del conjunto del *dataset* o el funcionamiento mismo del molino. Un resumen de los puntos mas importantes es el siguiente:

- Datos negativos: indican detención del funcionamiento del molino por lo que su valor real es cero.
- Datos extremadamente cercanos a cero o extremadamente altos: indican detención o en proceso de detención del molino. También pueden ser fallas de medición del sensor, pero se puede comprobar analizando en que sección del ciclo de funcionamiento aparecen los datos corruptos.
- Valores vacíos y ceros: tienen sentido en las secciones de revisión o termino de campaña (Esto ya se menciono previamente).
- Valores textuales o *Strings*: son errores de toma de medición. Estos pueden ser imputados con la media de datos o directamente ser cero si se encuentran en una sección de revisión o fin de campaña.

Utilizando el rango intercuartílico se puede apreciar la dispersión de los datos por columnas lo cual lo hace una medida muy útil para el análisis. La mayoría de las columnas presentan simetría en el conjunto de datos, lo cual sugiere uniformidad de la distribución alrededor de la mediana. Sin embargo existen algunas columnas con un presencia de asimetría fuerte:

- La asimetría en la **Recirculación** indica que la mayoría del tiempo el tonelaje recirculado es bajo.
- La asimetría en la **Velocidad de giro** indica que la mayor parte del tiempo el molino esta operativo sobre velocidades de 8 RPM.

La presencia de asimetría en estos casos tiene un sentido lógico del funcionamiento del equipo, lo cual no presenta ningún problema.

También es relevante tener otra medida para conocer estos valores atípicos y otra forma de hacerlo es utilizando el *z-score*. Como las características operativas no presentan distribuciones normales, es mejor utilizar la mediana como medida ya que esta no es afectada por valores extremos. Esto permite un análisis más robusta cuando existen valores atípicos en el conjunto.

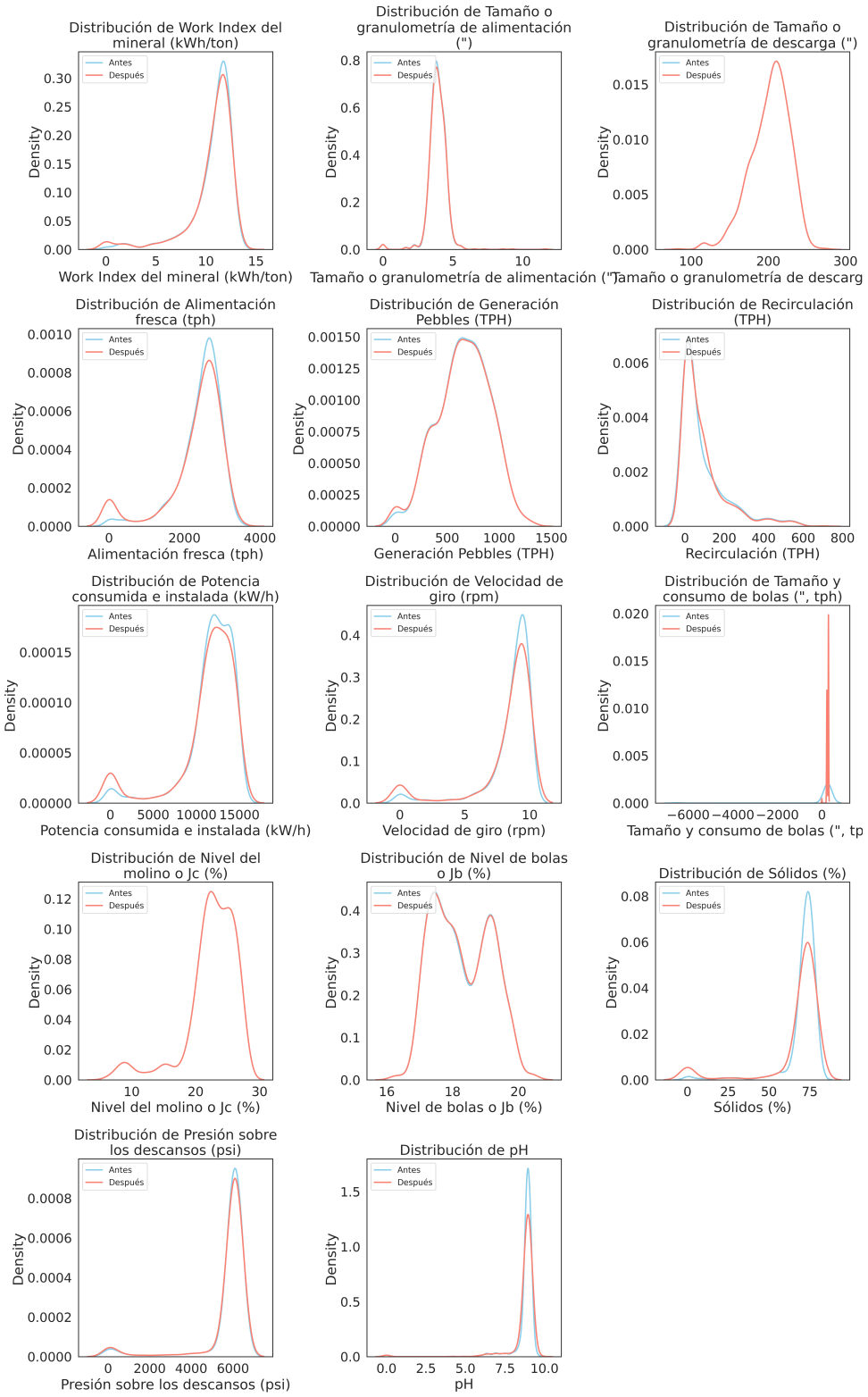


Figura 20: Comparación entre las distribuciones antes y después de imputar y escalar las columnas en el molino SAG 3.

Fuente: Elaboración propia.

A grandes rasgos, solo se corrigieron datos corruptos o errores de toma de datos. Gran parte de los datos que aparecen como outliers se mantuvieron ya que representan el rango de funcionamiento del molino. Por ello, quitar o modificar estos datos hará que las distribuciones se aplanen y no sean representativas de la operación real de los molinos SAG.

3.3. Análisis por ciclos en campañas

Siguiendo los datos de campaña de supervisiones de los molinos, es fácil distinguir la presencia de ciclos de cambio de revestimientos. Conocer estos ciclos aporta información sobre la duración del *lifter* y la placa durante el periodo en que la máquina operó con un set de revestimientos nuevo.

Para lograr esto, se creó un set de reglas sobre el *dataset* con el objetivo de realizar una comparación con los dos parámetros principales que indican un estado del funcionamiento del molino: 'Velocidad de giro (rpm)' y 'Potencia consumida e instalada (kW/h)'. Estas condiciones no se eligieron a la ligera, fueron estipuladas por conversaciones con los profesionales del área en Tega.

```
condiciones = [
    (df['Velocidad de giro (rpm)'] == 0) & (df['Potencia consumida e instalada (kW/h)'] == 0),
    (df['Velocidad de giro (rpm)'] > 0) & (df['Velocidad de giro (rpm)'] <= 7.5),
    (df['Velocidad de giro (rpm)'] > 7.5)
]
```

Figura 21: Set de condiciones para generar los cambios de estado del molino.
Fuente: Elaboración propia.

Con las reglas establecidas de la figura 21, se construyó otra columna que describa el estado de la máquina, eso es, apagado(0), encendido(1) y acelerando(2). Para esto se consideran las variaciones de velocidad y cambios en la potencia de entrada del equipo. Esta característica nueva es una derivada llamada **Estado**.

3.4. Modificación de datos categóricos

Durante el ciclo de funcionamiento hay momentos en los cuales se cambia el sentido de giro del equipo y esto esta expresado como un dato categórico *'horario'* o *'antihorario'*. Los modelos de aprendizaje automático utilizan datos cuantitativos para el entrenamiento, por lo que datos del tipo categórico deben ser modificados para que se expresen de forma numérica.

Como este es un caso sencillo de dos datos categóricos, basta con utilizar el *OneHotEncoder* de *Sklearn* para generar dos columnas que representen la distribución de estas dos clases. Una de las columnas indica la ocurrencia de datos *'horarios'* y la otra *'antihorarios'*, utilizando una representación binaria de unos y ceros.

3.5. Proyección del desgaste

Parte del objetivo principal de este trabajo es crear un modelo robusto que proyecte el desgaste de los *lifter* y placas de forma que se genere una geometría optima del perfil del revestimiento. Para lograr esto, es necesario tener datos previos del comportamiento de los revestimientos en un periodo determinado. Ya se tienen una variedad de datos los cuales serán útiles para generar la relación entre el funcionamiento del molino y el desgaste esperado. Sin embargo, entre estas características proporcionadas, no existen datos continuos del cambio de espesor de las placas y *lifters* tanto de la zona de alimentación como de descarga del molino (notar que se hace referencia a la zona del anillo y no a las tapas).

Durante el periodo de vida de los molinos SAG, estos se ven sometidos a procesos de mantenimiento durante cada campaña que, por lo general, suceden entre 3 a 9 veces por periodo. Durante este proceso, se suelen realizar revisiones de la condición de los revestimientos y una de las técnicas que es utilizada para esto es realizar un mapeo 3D del interior del molino para determinar la presencia de fracturas. También el uso de sensores permite conocer con una mejor precisión los cambios en la altura del perfil, midiendo el desgaste en milímetros.

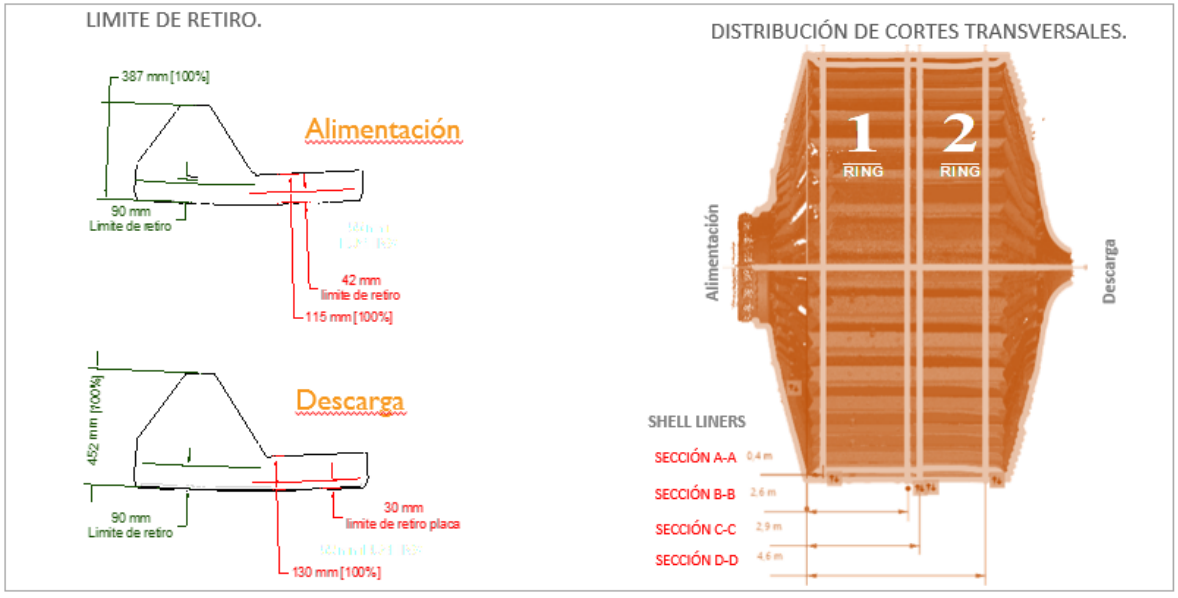


Figura 22: Muestra el espesor inicial del revestimiento y el límite de retiro. También se ve gráficamente la zona de los anillos y tapas.
Fuente: Tega Industries.

Para este trabajo, se dispone de datos de revisión de distintas campañas desde principios de 2021 hasta principios de 2024. Las revisiones son puntuales en cada campaña y, al final de estas, se realizan los recambios por revestimientos nuevos. Estas revisiones contemplan datos numéricos del espesor del *lifter* de alimentación y descarga, así como de las placas de ambos anillos. Sin embargo, estos datos son puntos discontinuos en el dataset, lo que genera el problema de la falta de datos entre las revisiones, los cuales son relevantes para desarrollar un modelo robusto. Para solventar este problema existen técnicas de regresión e interpolación que permiten generar datos que se adapten al comportamiento del desgaste presente en las mediciones. En particular, se explora dos técnicas sencillas pero suficientes: Regresión lineal e interpolación polinomial.

3.5.1. Ajuste por regresión lineal

El método de regresión está dentro de los métodos lineales de predicción de datos y es uno de los más simples en el grupo de aprendizaje supervisado. Sin embargo, que sea el más sencillo no lo hace menos útil; al contrario, puede ofrecer excelentes resultados dependiendo del contexto de los datos y el comportamiento de estos. La regresión lineal es especialmente eficaz cuando existe una relación lineal clara entre las variables, lo que permite realizar predicciones precisas y rápidas.

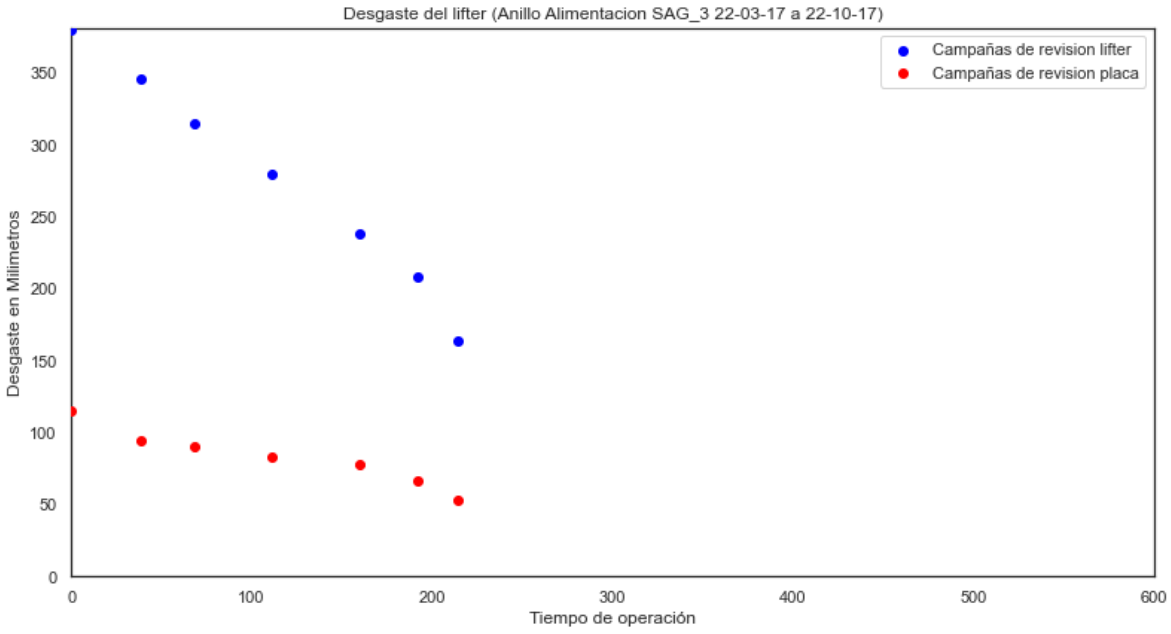


Figura 23: Dispersión de punto de revisión durante la campaña.
Fuente: Elaboración propia.

El muestreo de puntos discontinuos de las mediciones del espesor están compuestos por intervalos de fechas en las cuales se realizó la campaña. En la figura 23 se muestra el ciclo

completo de una campaña con fecha 17 de marzo de 2022, cada punto indica una revisión del estado del revestimiento.

En la figura 23 se observa un patrón claro de descenso del espesor del perfil. El último punto muestreado es aquel cuando se termina la campaña y se reemplaza el revestimiento interno del molino por uno nuevo. Sin embargo, lo que interesa observar es la proyección lineal sobre el conjunto de puntos. La figura 24 muestra dicha proyección, además de contar con más datos para contrastar con la figura 23.

La figura 24, además de agregar la regresión sobre los puntos, indica con lineas paralelas el límite de desgaste máximo permitido antes de recambio tanto para el *lifter* como para las placas. En este caso, Tega realizó los recambios antes de llegar a dicho límite.

La regresión es una opción eficaz y sencilla para rellenar datos faltantes. Sin embargo, se debe tener cuidado con los resultados, ya que utilizar directamente una regresión lineal entre los puntos puede generar un sesgo inherente. Lo ideal sería contar con datos exactos por día del desgaste, pero dadas las limitaciones en el volumen de datos proporcionados, solo se puede recurrir a estas técnicas para generar los puntos faltantes. Por ello, para reducir este sesgo, es importante explorar opciones no lineales que permitan una mejor ajuste de los puntos de desgaste faltantes.

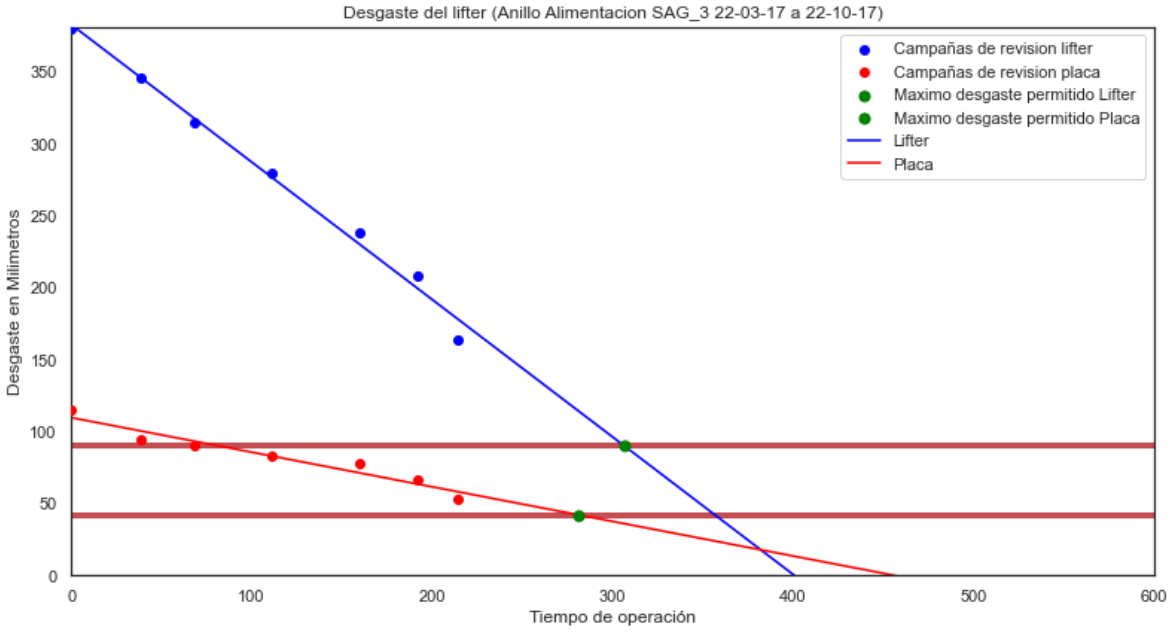


Figura 24: Regresión lineal sobre puntos de revisión durante la campaña.
Fuente: Elaboración propia.

3.5.2. Ajuste por interpolación polinómica

La interpolación polinómica es especialmente útil para realizar aproximaciones de los datos faltantes de un conjunto utilizando pocos puntos. Particularmente, para un conjunto de puntos n, existe un polinomio interpolador de grado máximo $n-1$ [Vadillo, 2023].

Para realizar este paso, la librería de *Pandas* presenta una función útil llamada *interpolate*. Esta función presenta diversas opciones de interpolación, pero en este caso interesa hacer uso de el método polinómico. Para utilizar este método se requiere conocer la cantidad de puntos para indicar al modelo el orden de la función, cosa que no presenta ningún problema ya que es información que se tiene conociendo el número de supervisiones.

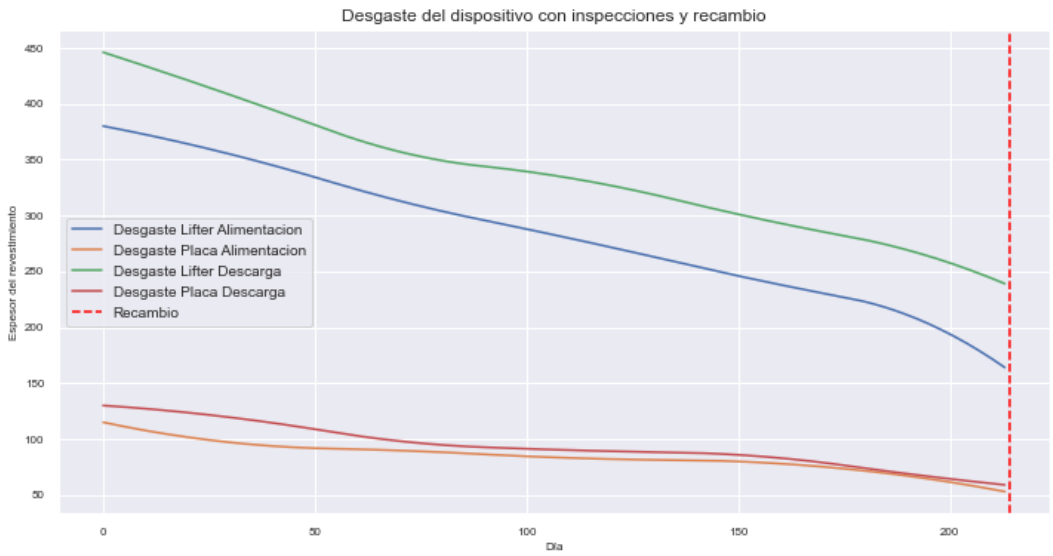


Figura 25: Interpolación de puntos de revisión de espesor del revestimiento.
Fuente: Elaboración propia.

La figura 25 muestra el comportamiento de la interpolación de los puntos de revisión de los revestimientos utilizando una función. Particularmente, este caso es una de las campañas. Aquí se puede notar el decrecimiento del espesor del *lifter* y la placa, se presenta un comportamiento más ajustado sobre los puntos en comparación a la regresión lineal. La figura 25 visualiza mejor las diferencias entre ambas técnicas y la figura 27 muestra el comportamiento cíclico interpolado considerando todas las campañas del molino SAG 2 y SAG 3.

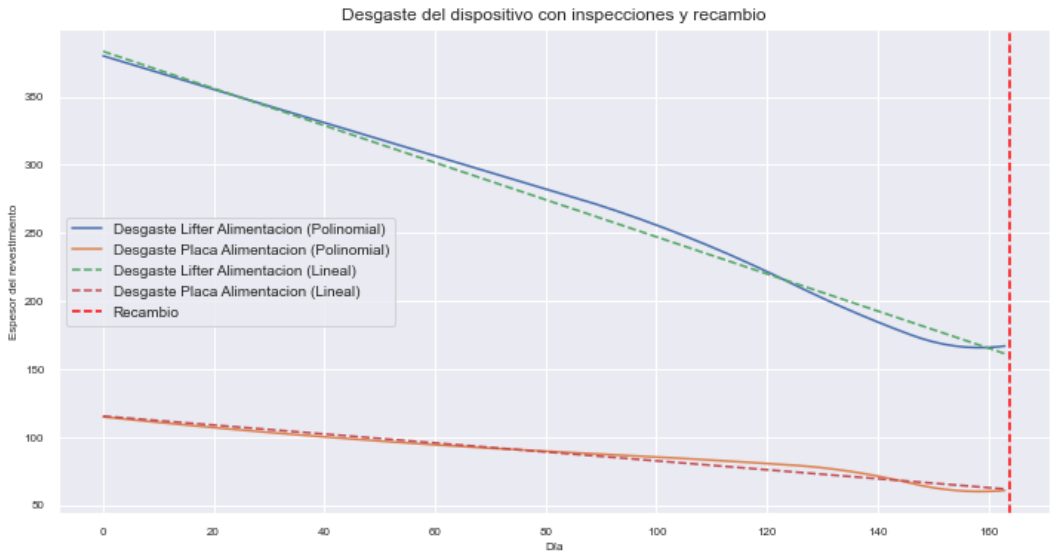


Figura 26: Diferencia entre la proyección de puntos continuos de desgaste de un ciclo de funcionamiento utilizando interpolación polinómica ante la regresión lineal.
Fuente: Elaboración propia.

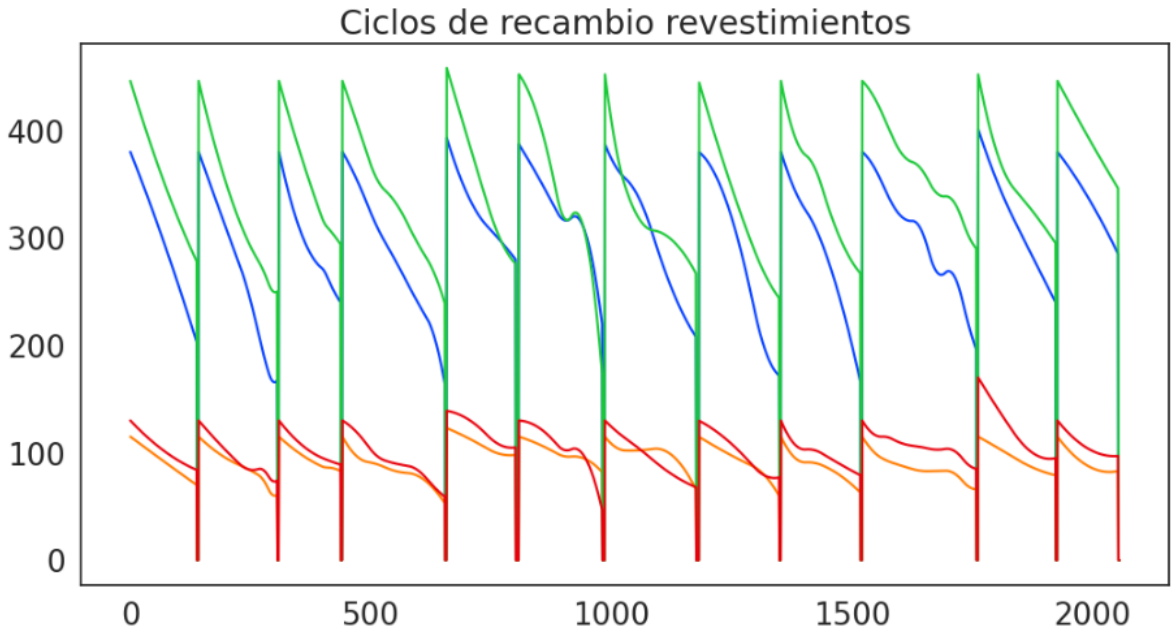


Figura 27: Desgaste(mm) vs días en operación.
Fuente: Elaboración propia.

3.5.3. Identificación de características relevantes

Inicialmente existen 27 atributos que describen la operación del molino. Esta cantidad de características es computacionalmente compleja para realizar los cálculos, en especial cuando se quieren realizar pruebas con hiperparámetros en un modelo. Por ello, desarrollar un algoritmo que permita optimizar la cantidad de columnas a utilizar es importante. Para esto existen formas de conocer la relevancia matemática de estas características en relación al espesor del revestimiento según mediciones previas. Estos métodos se conocen como *Lasso* y *Ridge*, los que consisten en determinar el *peso* de la característica en el modelo, demostrando así su importancia en relación a los otros atributos [Team, 2024].

También conocidas como las normas $L1$ y $L2$, funcionan ajustando la *función de costo* tal que optimicen los pesos de cada una de las características. En términos sencillos, dicha función de costo se encarga de conocer la diferencia entre los valores predichos y los valores reales, calculando así el error de predicción.

$$\sum_{i=1}^{M} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{M} \left(y_i - \sum_{j=0}^{p} w_j \times x_{ij} \right)^2$$

Figura 28: Función de costo de un modelo de regresión lineal donde y_i representa el valor real e $\hat{y}_i$ representa los valores proyectados. En la parte derecha de la ecuación, p es el número de características (columnas), x_{ij} es el valor de la característica y w_j es el peso que se le asigna a dicha característica.
Fuente: Datacamp [Team, 2024].

Conocer la relevancia de cada característica permite la creación de un modelo que evite el sobreajuste y se comporte robustamente ante nuevas entradas. Este proceso de conoce como **regularización**.

3.5.4. Lasso *L1*

Es un método de regularización que introduce un factor de penalización α sobre los coeficientes de la función, actuando como un factor de contracción. Cuando $\alpha = 0$, no hay restricciones sobre la ecuaciones, pero a medida que α aumenta ($\alpha > 0$), la penalización se intensifica. Esto reduce el valor de algunos coeficientes e incluso puede llevarlos a cero, lo que permite la selección automática de variables.

3.5.5. Ridge *L2*

Al igual que *Lasso*, se utilizan penalizaciones sobre los pesos de las características. La diferencia radica en que *Ridge* utiliza el cuadrado de la magnitud de los coeficientes y el factor de reducción se denomina λ. Esta penalización cuadrática evita que los coeficientes se reduzcan a cero, manteniéndolos pequeños. Esto ayuda a manejar la multicolinealidad y puede mejorar la estabilidad del modelo.

La figura 29 muestra los pesos de las características utilizando técnicas de regularización $L1$ y $L2$. En este análisis, se emplearon datos del desgaste en los ciclos de funcionamiento de los molinos, donde se observa que la característica con mayor peso es 'Tonelaje acumulado'.

3.5.6. Tonelaje acumulado

La columna 'Tonelaje acumulado' es una variable artificial (así como lo es la de 'Estado') que representa el tonelaje total procesado por los revestimientos durante cada ciclo de vida. Esta nueva característica se calcula como la suma del tonelaje diario procesado más el tonelaje diario recirculado. Debido a su alta correlación con el espesor del revestimiento, aporta información clave para mejorar el desempeño del modelo.

Se observa en la figura 29 que dentro de las características con mayor peso matemático están 'Tonelaje acumulado' y 'Estado'. Ambas columnas fueron creadas con ingeniería de características y son derivadas de otras columnas operativas.

En la figura 30 se realizó una comparación entre la convergencia de selección de características para cada uno de los métodos. En este estudio de convergencia se fue agregando características una a una en un modelo de regresión simple que realiza predicciones del desgaste. La idea es ver como disminuye el error cuadrático medio (RMSE) de las predicciones a medida que se agregan características a este modelo. Los métodos de *lasso* y *ridge* tienen mejores resultados eligiendo las características más importantes en comparación con elegirlas de forma aleatoria, esto en función del RMSE que se obtuvo el cual representanta entre un 9 % y 15 % de error sobre la altura inicial dependiendo de la pieza del revestimiento.

Finalmente se decidió utilizar las 13 primeras características con mayor peso estadístico en el modelo. Como se muestra en los gráficos de convergencia, no existe mejoría apreciable en el RMSE utilizando más de 13 características. Aunque técnicamente colocar aun más características mejoraría el resultado, también aumentaría el sobreajuste pues se generaría un árbol de decisión con mayor profundidad.

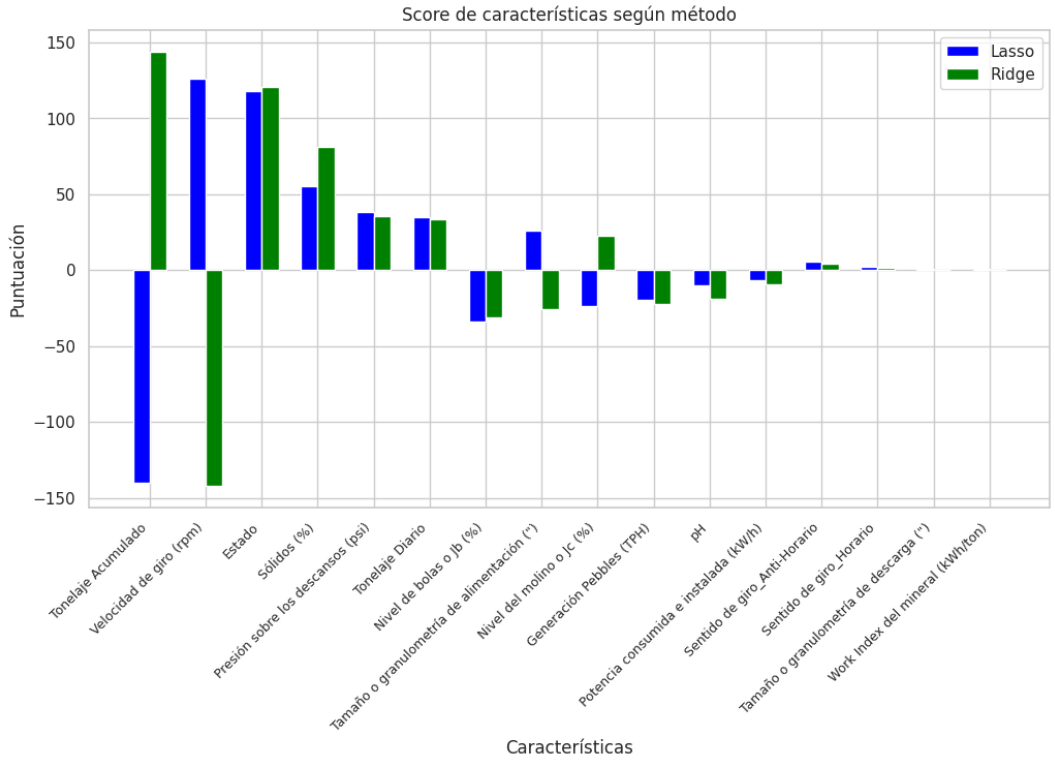


Figura 29: Gráfico de los pesos de las características utilizando regularización $L1$ y $L2$.
Fuente: Elaboración propia.

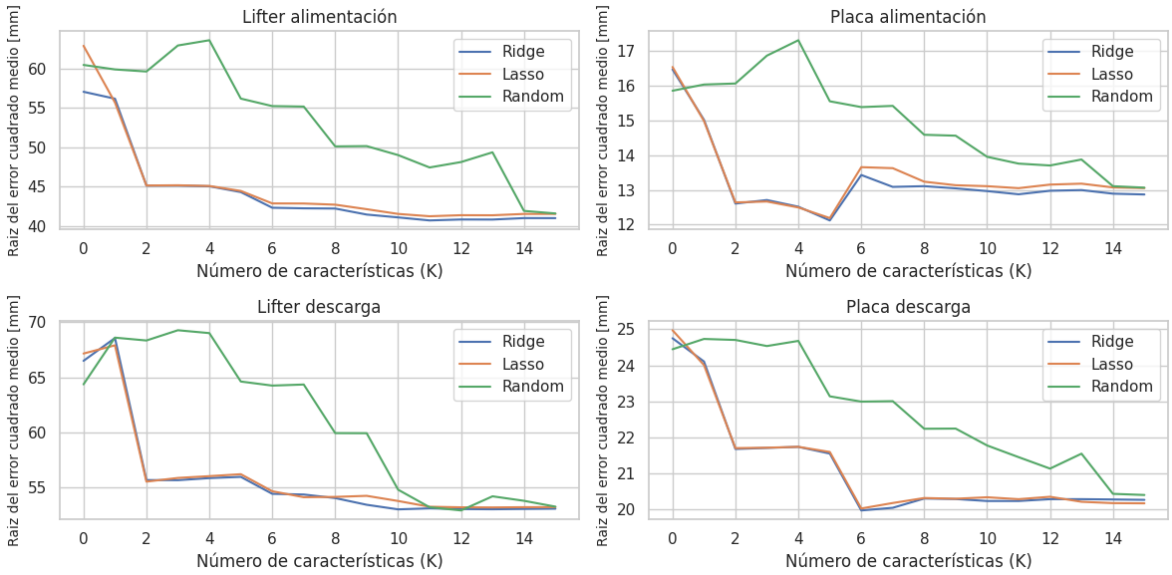


Figura 30: Comparación convergencia de características relevantes utilizando *lasso*, *ridge* y selección aleatoria.

Fuente: Elaboración propia.

3.6. Modelado con técnicas de *machine learning*

Con los datos imputados y las características ya seleccionadas, es factible pasar el siguiente paso de modelado. Se debe tener en cuenta la naturaleza de la información con la que se desea entrenar al modelo. Los datos están en orden temporal considerando un día por fila por un periodo de tres años, entonces, un modelo que tenga en cuenta el desgaste acumulado en el tiempo es necesario para que sea robusto y obtener resultados óptimos.

Existen diversos modelos de entrenamiento supervisado para este tipo de problemas. En redes neuronales hay enfoques diseñados para series temporales, pero suelen estar sesgados a los datos de entrenamiento, lo que dificulta hacer predicciones fuera de ese patrón. Por esta razón, se optó por un modelo que equilibra facilidad de uso y capacidad de capturar patrones complejos en los datos, permitiendo predicciones más precisas. El modelo seleccionado es XGBoost.

3.6.1. XGBoost: *extreme Gradient Boosting*

Este modelo de *machine learning* es una optimización del *gradient boosting* tradicional y ha sido altamente utilizado en estos últimos años por su gran rendimiento predictivo y rapidez de aprendizaje. Para comprender el funcionamiento de este modelo, hay que entender como funciona *gradient boosting*.

Para empezar, hay que explicar como funcionan generalmente los modelos basados en árboles de decisión, en esencia, estos se centran en generar diagramas de decisión en donde los datos escogen un camino u otro del diagrama en base a condicionales simples de *True* o *False*. Cada diagrama tiene un profundidad que esta ligada a la cantidad de variables o características que se buscan modelar. Teóricamente, mientras más profundo sea el árbol, más precisos son los resultados ya que se esta buscando en todo el espectro de posibilidades. Sin embargo, esto último suele llevar a un sobreajuste del modelo a los datos de entrenamiento y le impide generalizar nuevos datos para hacer predicciones confiables.

Es con este mecanismo que se crean modelos como *Random Forest* el cual entrena varios árboles de decisión de forma paralela con un método llamado *bootstrapping*. Esto asegura que los árboles no estén correlacionados, entregando subconjuntos de datos de entrenamiento a cada árbol y llegando a un consenso promedio de los resultados considerando todos los árboles entrenados [IBM, 2024a]. Este proceso de entrenar diferentes árboles y promediar los resultados se conoce como *Bagging*.

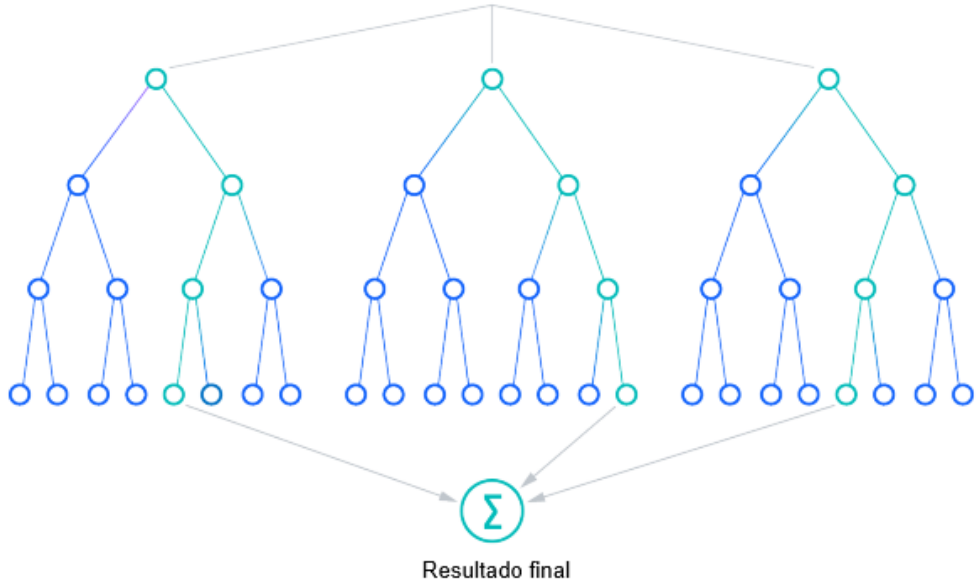


Figura 31: Representación del funcionamiento de *Random Forest*, se calcula el promedio de la decisión de los árboles.
Fuente: IBM [IBM, 2024b]

Volviendo a *gradient boosting*, este también se basa en crear varios árboles de decisión, pero a diferencia de otros algoritmos como *random forest*, *gradient boosting* se centra en mejorar el mismo árbol de forma iterativa. A esto se le conoce como *boosting*. Para resumir, el *boosting* corrige errores de los árboles anteriores utilizando una función de pérdida $L(yi)$ y en cada una de las iteraciones se busca minimizar esta función de pérdida con respecto a la iteración anterior. La explicación matemática de esto es algo extensa para este trabajo [Masui, 2022], así que simplemente se describirán los tres principales pasos:

A. Predicción inicial.
$$F_0(x) = argmin_\gamma(\sum_{i=1}^{n} L(y_i, \gamma))$$

B. Diferencia entre valor actual y valor real.
$$-\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}$$

C. Siguiente valor.
$$F_1 = F_0 + \nu\gamma 1$$

La ecuación final queda tal que así:

$$F_m(x) = F_{m-1}(x) + \left(argmin_{h_m} \left[\sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h_m(x_i)) \right] \right)(x)$$

Existen diferentes funciones de pérdida o *loss function*, pero para modelos de regresión se suele utilizar el *Root Mean Squared Error*. Esta es una métrica fácil de calcular, ya que es representativa de la dispersión de los datos a la media. Gráficamente, el comportamiento de los pasos antes mencionados se vería como en la figura 32.

Entonces, *XGBoost* utiliza exactamente los mismos principios que *gradient boosting*, pero con la diferencia que está optimizado en varios sectores de su funcionamiento, principalmente:

- Utiliza regularización $L1$ y $L2$ para la construcción de los árboles de decisión y así evitar el sobreajuste.
- Posee *Cross-validation* incorporado.
- Optimiza la construcción de los árboles de decisión.

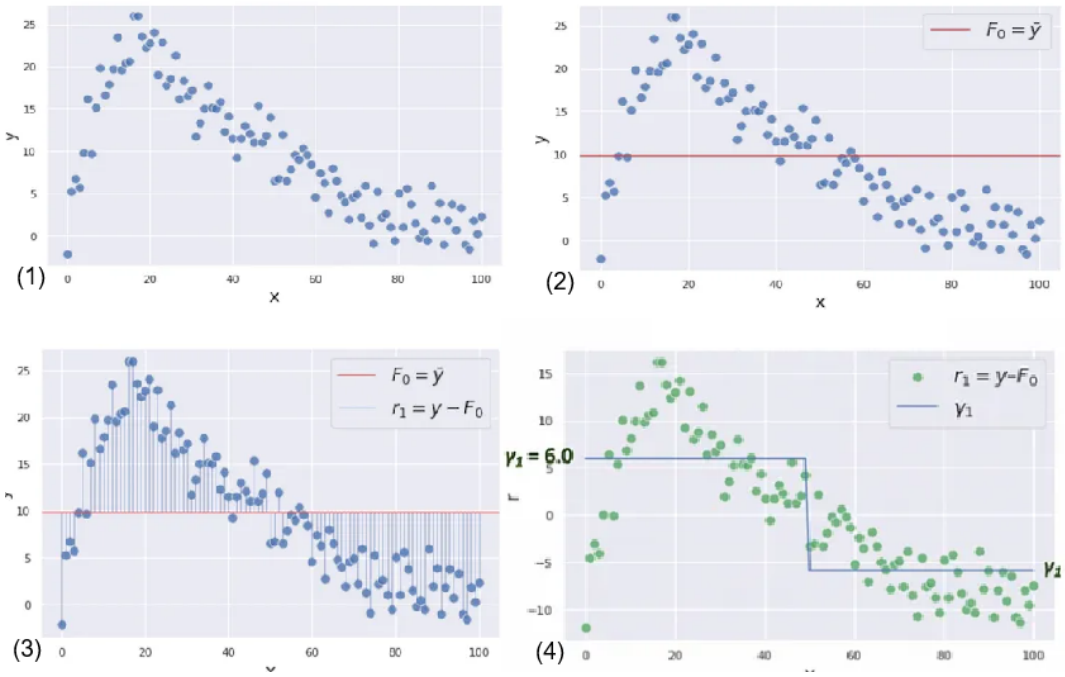


Figura 32: Representación del funcionamiento de *Gradient Boosting*
Fuente: *All You Need to Know about Gradient Boosting Algorithm – Part 1. Regression* [Masui, 2022]

3.6.2. Creando el modelo

Para construir el modelo se utilizaron las características destacadas durante la etapa de regularización, es decir, los atributos con la mayor importancia matemática usando *Lasso*. El modelo *XGBoost* utiliza como base los resultados de la interpolación polinomial antes realizada, esta decisión recae en las siguientes razones:

- Existe un conjunto muy escaso de puntos de revisión, esto provoca que al entrenar el modelo este no sea lo suficientemente robusto.
- En la realidad, el desgaste del revestimiento no es lineal, *XGBoost* toma en cuenta las variables operacionales.
- Podría existir sobreajuste si el modelo se entrena sobre pocos puntos, una aproximación polinómica entrega robustez al modelo y lo acerca a la realidad operativa.

Con respecto al *output* del modelo, en la primera parte de este trabajo se realizó un análisis exhaustivo de los datos, las características y la correlación entre ellas. Ahora, al ser necesario construir el modelo, es importante comprender cómo se hará uso de estas características y los problemas que pueden surgir.

Para empezar, se utilizó el **desgaste diario** en milímetros $[mm]$ como métrica de salida. Aquí hay que recordar que los datos muestran el desgaste de forma cíclica, comenzado con una altura inicial del perfil hasta la altura final que es cuando se cambia el revestimiento. Se modificó esto para dejar de lado los ciclos como tal y solo considerar las diferencias del espesor entre días. Esta decisión se basa en los siguientes puntos:

- Todos los datos proporcionados están totalmente sesgados a una altura inicial fija (no hay alturas iniciales diferentes), lo cual dificulta al modelo para generalizar otros grosores iniciales.
- Para evitar la periodicidad asociada a utilizar siempre la misma altura inicial del perfil, se optó por trabajar con las diferencias de grosores diarios. Esto facilita el modelado, ya que el desgaste en milímetros por día no sigue un patrón cíclico.

Una vez aclarado estos puntos, se pasa directamente a la construcción del modelo utilizando la librería de *XGBoost*. Esta proporciona el modelo y las herramientas necesarias para su uso. Durante el entrenamiento, se dividirá en dos propuestas de complejidad:

- Primero, se observó el comportamiento del entrenamiento utilizando el modelo base. Esto quiere decir que no se realizó ningún tipo de *tunnig* de hiperparámetros ni validación cruzada.

- Segundo, se realizó una búsqueda de parámetros para optimizar el *output* del modelo. Para ello, se hizo uso de *Optuna* la cual es una librería que permite búsqueda exhaustiva de hiperparámetros para una variedad de modelos de aprendizaje automático.
- Tercero, con respecto al método de enseñanza, un 80 % corresponden a entrenamiento y validación; el 20 % restante corresponde a *testing* del modelo (mayor explicación de esto en el siguiente capitulo).

Es importante entender cómo se utiliza el modelo de *XGBoost* en Python, ya que es ligeramente diferente a otros modelos de regresión. Esta librería incluye su propia clase llamada *DMatrix* para guardar y manipular *datasets*, la cual está fuertemente optimizada para mejorar la velocidad y el uso de memoria.

Quitando al pequeña diferencia antes presentada, *XGBoost* se entrena de la misma forma que otros modelos de regresión.

3.6.3. Función objetivo

De momento, se ha realizado un análisis profundo de las características de los datos y también se tiene el modelo. Pero, ¿cómo se utilizan los resultados? El modelo se entrena entregando un conjunto de características X y un conjunto de salida Y (datos reales operativos del molino). Para realizar la predicción, basta con proporcionar un conjunto de características X no utilizadas en el entrenamiento (un histórico del funcionamiento del molino). El output Y consta de 4 salidas que representan el desgaste medido en milímetros de cada una de las piezas: *lifter* de alimentación, placa de alimentación, *lifter* de descarga y placa de descarga. El esquema de la figura 33 lo resume mejor.

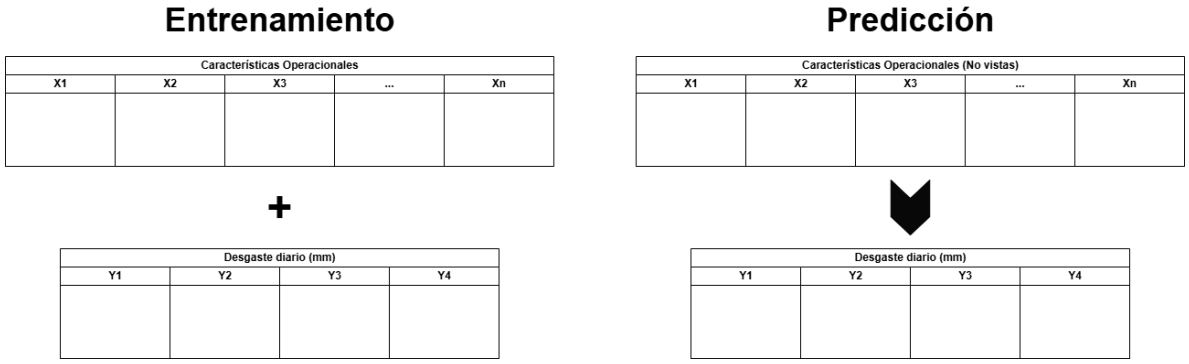


Figura 33: Esquema simple de la estructura de entrenamiento y predicción del modelo
Fuente: Elaboración propia

Para hacer la predicción es necesario conocer un histórico de datos del funcionamiento operacional del molino con el que se esta tratando.

Tener la predicción del desgaste por si sola no es útil, por ello, se creó una **función objetivo** con una penalización con la meta de encontrar un valor inicial de la altura de los revestimientos que minimice dicha penalización. Esta función consta de los siguientes argumentos:

- Alturas: alturas iniciales del perfil simple de los revestimientos.
- Modelo: Modelo de predicción XGBoost ya entrenado.
- Histórico máquina: características operacionales para la simulación.
- Limites de desgaste: lista de límites de desgaste permitidos para cada pieza.
- Días objetivo: días que se desea que dure el revestimiento.
- Scaler: Como se escalan las características.

La altura inicial y los limites de desgaste vienen dados por datos históricos de las campañas de los revestimientos de la empresa, pero son parámetros modificables. Los argumentos importantes de la función son el modelo (claramente) y los datos históricos, ya que en este trabajo se busca entregar un modelo predictivo que se adapte al molino SAG que se esta estudiando. Esto último se debe principalmente a que abarcar un modelo de mayor generalidad necesitaría de un mayor número de datos y estudiar el comportamiento con ángulos de ataque diferentes, lo cual es un trabajo más extenso que no se incluirá en este escrito.

El funcionamiento del algoritmo se resume en los siguientes pasos:

- Se entregan los argumentos a la función, entre estos están el modelo y los datos operativos del molino.
- Se utiliza el modelo previamente entrenado con XGBoost para predecir los desgastes diarios.
- Los desgastes diarios se restan a la altura inicial propuesta.
- Este proceso continúa hasta alcanzar los días objetivo o superar los límites de desgaste.
- La penalización se calcula mediante una función que relaciona:
    A. La diferencia entre la altura inicial y el desgaste proyectado.
    B. Los límites de desgaste previamente establecidos.
- El algoritmo es iterativo, por lo que se irá modificando la altura a medida que pasen las iteraciones.
- El objetivo es minimizar la penalización y encontrar una altura inicial óptima para la duración objetivo.

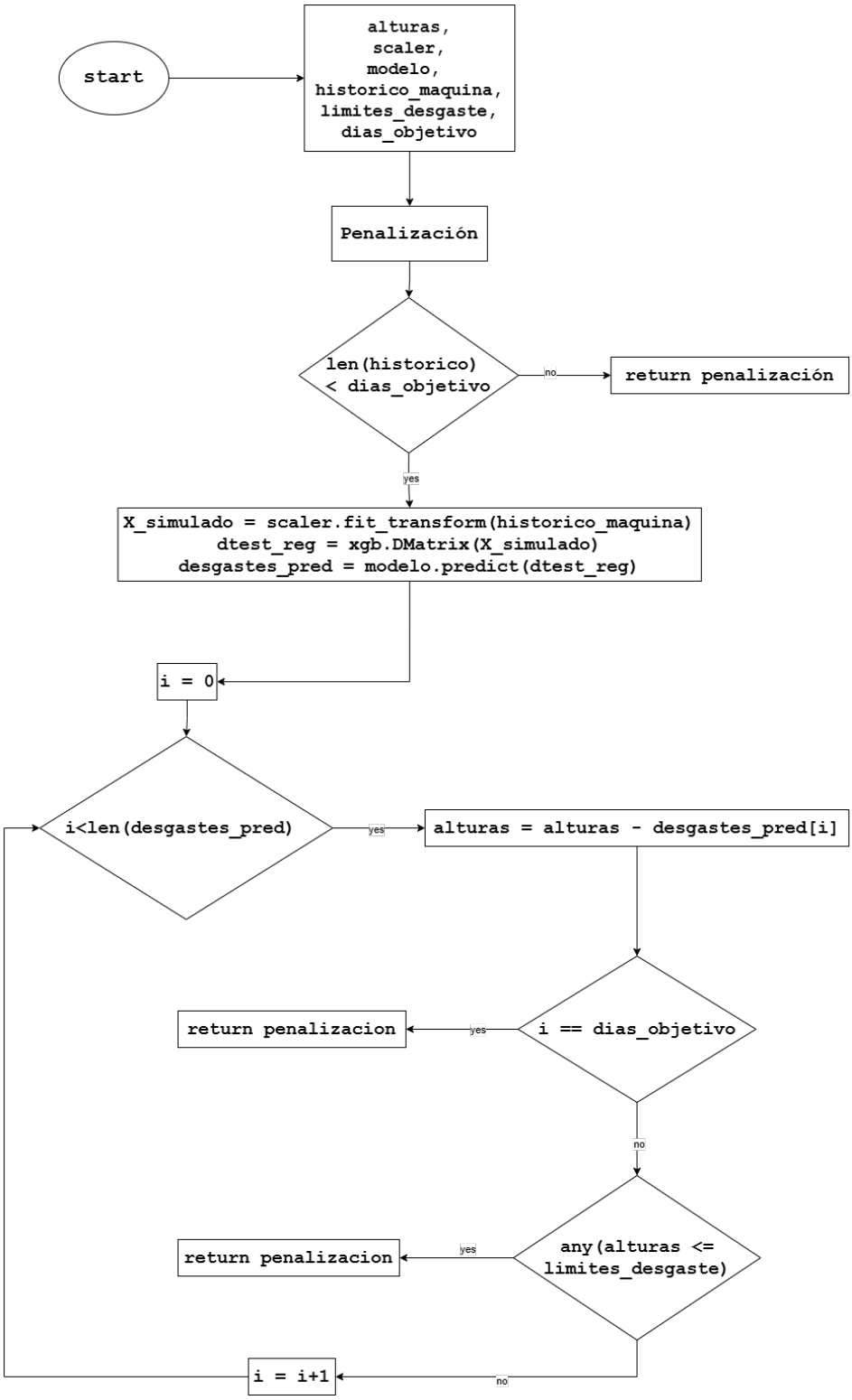


Figura 34: Diagrama de flujo de la función objetivo
Fuente: Elaboración propia

Lo siguiente a definir es la penalización. Esta difiere según el caso, es decir, hay dos penalizaciones diferentes dependiendo si la solución de la iteración factible o no. Tengamos en consideración lo siguiente:

$$x_1 = \text{altura lifter alimentación}$$

$$x_2 = \text{altura placa alimentación}$$

$$x_3 = \text{altura lifter descarga}$$

$$x_4 = \text{altura placa descarga}$$

$$y_1 = \text{límite lifter alimentación}$$

$$y_2 = \text{límite placa alimentación}$$

$$y_3 = \text{límite lifter descarga}$$

$$y_4 = \text{límite placa descarga}$$

$$\text{penalización} = x_1 - y_1 + x_2 - y_2 + x_3 - y_3 + x_4 - y_4$$

Esta primera aproximación muestra que la penalización se define como la suma de las diferencias entre la altura y el límite. Sin embargo, existe un problema: los pesos de la ecuación para cada término son idénticos. Esto sería correcto si el desgaste se comportara de manera proporcional en todas las partes del revestimiento, pero se sabe que esto no es cierto, ya que la placa alcanza su límite antes que el *lifter*.

Si la ecuación se mantiene así, el espesor del *lifter* en la predicción quedará por encima del óptimo buscado. La solución a este problema es ajustar los pesos de la ecuación para penalizar más a los *lifters*. Las magnitudes utilizadas para los pesos son proporcionales a la diferencia entre el grosor del *lifter* y el de la placa (el *lifter* es aproximadamente tres veces más grueso que la placa).

$$\text{penalización} = 3 * (x_1 - y_1) + (x_2 - y_2) + 3 * (x_3 - y_3) + (x_4 - y_4)$$

Esta es la penalización en caso de que la solución encontrada sea factible. Por otro lado, si durante alguna de las iteraciones el desgaste del revestimiento supera el límite máximo permitido, entonces la penalización debe ser más severa ya que estaríamos ante una solución no factible. Por esto último, la ecuación que mejor describe este escenario es una cuadrática.

$$\text{penalización} = 100 * [3 * (x_1 - y_1) + (x_2 - y_2) + 3 * (x_3 - y_3) + (x_4 - y_4)]^2$$

Esta modificación resuelve los siguientes puntos:

- En caso de que la altura se pase del límite de desgaste, entonces la resta entre ambos será negativa, lo cual confundiría al algoritmo. Para solventar esto, se considera el valor cuadrático de la suma (para mantener el resultado positivo).
- La severidad de la penalización debe ser mayor.

Finalmente, se utiliza función objetivo con la función *Minimize* de la librería *scipy*, no se requiere mayor complejidad en este problema, por lo que los parámetros de funcionamiento de *Minimize* son los predeterminados.

3.7. Geometría del perfil simple

Con el algoritmo propuesto ya es posible desarrollar un código que permita obtener una geometría del perfil simple del revestimiento que se adecúe a una duración objetivo. Para realizar esto, se consideró el modelo optimizado con *Optuna* para hacer uso de los mejores hiperparámetros.

Trabajar con la geometría no solo requiere la inferencia del modelo creado respecto a la altura del *lifter* y la placa de los anillos internos del molino SAG, sino también datos adicionales como el ángulo de ataque y el grosor del revestimiento. La generación de estos datos mediante un modelo más complejo podría ser objeto de un estudio más amplio. Sin embargo, en el presente trabajo se tomará como referencia el estudio de [Valderrama y Magne, 1996] y se propondrán diferentes diseños de perfiles con diferentes ángulos de ataque.

Para el diseño, la empresa utiliza el software *AutoDesk AutoCAD*, que permite realizar modelado en 2D y 3D. *AutoCAD* es una herramienta altamente flexible y versátil. Considerando esto y dado que el modelo está basado en Python, para generar las figuras geométricas a partir de los resultados del modelo se utilizó la librería *ezdxf*, que permite automatizar la creación de archivos DXF (formato compatible con *AutoCAD*). Además, los diseños estarán basados ejemplos proporcionados por *Tega*, lo cuales también están en este formato de archivo.

El mayor desafió en esta sección fue calcular los puntos para dibujar el revestimiento dependiendo del ángulo de ataque, para ello se utilizó simple trigonometría. Para mayor detalle, revisar la sección del anexo donde se encuentra el código en python.

CAPÍTULO 4
VALIDACIÓN DE LA SOLUCIÓN

Para esta sección del trabajo se realizaron una serie de pruebas con el modelo propuesto en conjunto con la función objetivo creada. Para comenzar, hay que observar y analizar el comportamiento de los modelos creados con *XGBoost*, comparando el como se construyeron y las diferencias entre el rendimiento promedio de cada uno.

El entorno de entrenamiento y testing se compone tal que:

- Del 100 % de los datos, un 20 % es para pruebas.
- Del 80 % de los datos de prueba, se utilizó *cross-validation* durante el entrenamiento del modelo.

Con respecto a la validación cruzada, se optó por utilizarla debido a la cantidad limitada de datos disponibles, lo que dificulta obtener predicciones ajustadas a la realidad. La validación cruzada se aplica sobre el conjunto de entrenamiento, dividiéndolo en *folds* o particiones. El modelo se entrena de forma iterativa considerando $n - 1$ *folds*, dejando una partición para validación en cada una de las iteraciones. Todo este proceso se llevó a cabo sin modificar el conjunto de prueba, proporcionando así una estimación más realista del *RMSE* y del comportamiento de las predicciones finales.

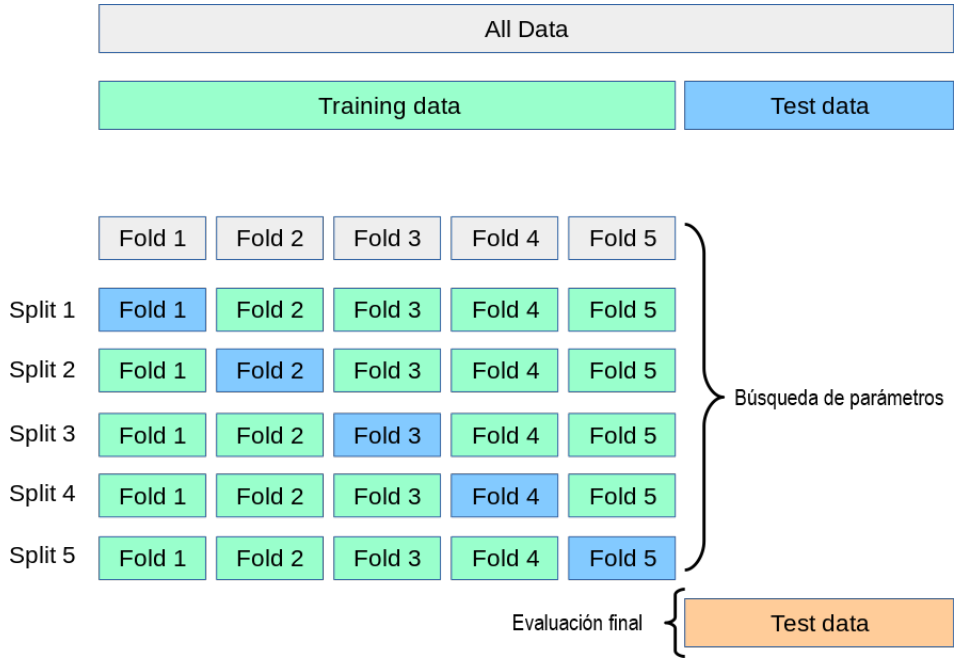


Figura 35: Funcionamiento de *Cross-validation*
Fuente: *Cross-validation: evaluating estimator performance [scikit learn, 2024].*

4.1. Fase 1: Modelo XGBoost

4.1.1. Modelo básico

Utilizando los parámetros predeterminados de *XGBoost* ya es posible obtener resultados decentes, ya que los modelos basados en árboles de decisión son muy flexibles, de rápida ejecución y generalmente de buen rendimiento. El gráfico de la figura 40 representa el comportamiento de las predicciones con este modelo con respecto a los valores de reales (conjunto de prueba).

Se observa que el modelo suele seguir los patrones de desgaste de las distintas zonas de los revestimientos y en donde más difiere la predicción es en la zona de la placa de descarga. Con respecto al rendimiento, se obtuvo un R^2 de 0.776.

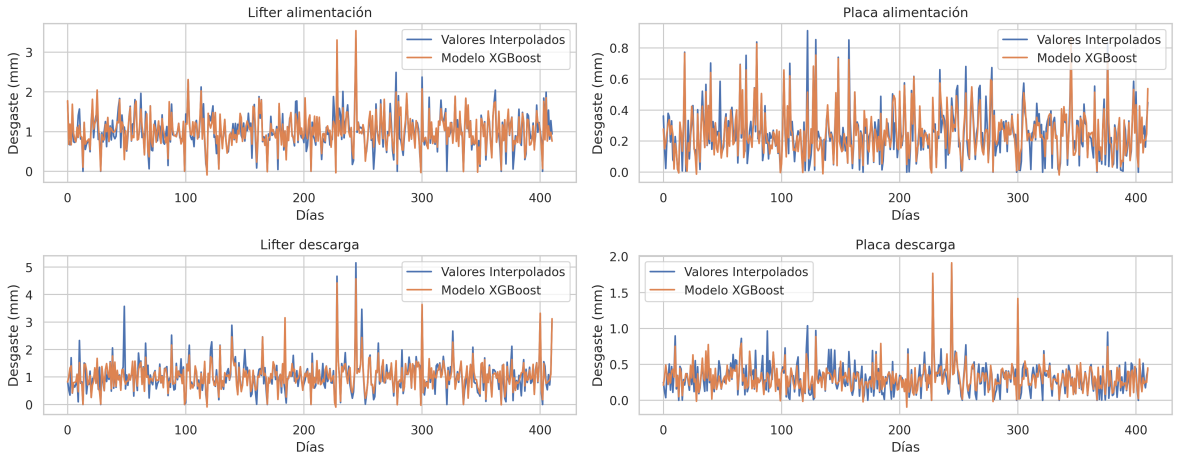


Figura 36: Modelo simple
Fuente: Elaboración propia.

4.1.2. Modelo con *tunning* simple

En esta iteración del modelo, se modificaron algunos de los hiperparámetros antes mostrados. El rendimiento del modelo con unos ajustes simples se ve favorecido, en particular en la zona de la placa de descarga. El rendimiento se esta iteración presenta un R^2 con un valor de 0.809.

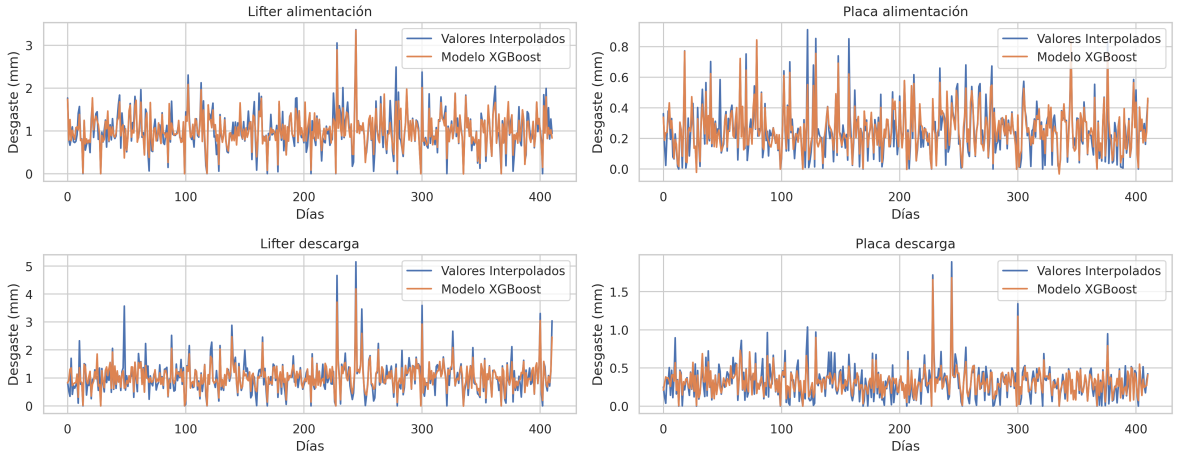


Figura 37: Modelo con *tunning* simple
Fuente: Elaboración propia.

4.1.3. Modelo utilizando Optuna

Finalmente, en este modelo se modificaron todos los hiperparámetros posibles utilizando un *framework* de *auto-tunnig* para facilitar el proceso. Particularmente los rangos en los que se realizó la búsqueda se muestran en la siguiente figura:

```
params = {
    "objective": "reg:squarederror",
    "learning_rate": trial.suggest_float("learning_rate", 0.01, 0.1, log=True),
    "max_depth": trial.suggest_int("max_depth", 6, 12),
    "subsample": trial.suggest_float("subsample", 0.5, 0.9),
    "colsample_bytree": trial.suggest_float("colsample_bytree", 0.5, 0.9),
    "gamma": trial.suggest_float("gamma", 0, 3),
    "reg_alpha": trial.suggest_float("reg_alpha", 1e-8, 1.0, log=True),
    "reg_lambda": trial.suggest_float("reg_lambda", 1e-8, 1.0, log=True),
    "n_estimators": trial.suggest_int("n_estimators", 50, 1000),
    "max_bin": trial.suggest_int("max_bin", 128, 512),
}
```

Figura 38: Rango de búsqueda de hiperparámetros utilizando Optuna
Fuente: Elaboración propia.

Se realizó una búsqueda exhaustiva de 1000 iteraciones sobre la función objetivo implementada en Optuna para encontrar los mejores candidatos de parámetros a usar en el modelo. El rendimiento que se logró extraer es de una puntuación R^2 de 0.828.

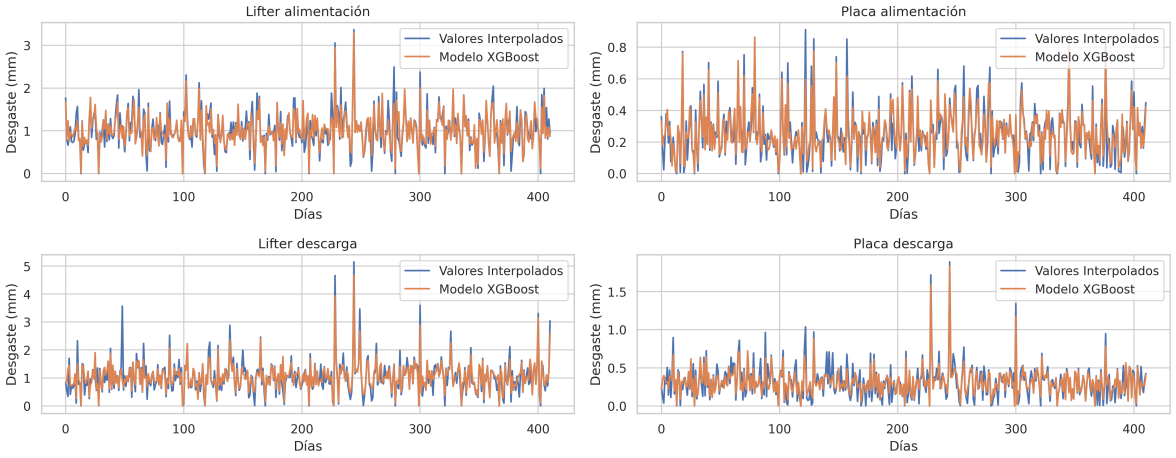


Figura 39: Resultados predicción modelo utilizando Optuna para hiperparámetros
Fuente: Elaboración propia.

Finalmente, se llevó a cabo una prueba de validación cruzada con 5 particiones utilizando el modelo optimizado mediante hiperparámetros ajustados con *Optuna*. Esto permitió evaluar el desempeño del modelo en escenarios más realistas. Debido al mecanismo de validación de *XGBoost*, en esta prueba no se obtuvo un coeficiente de determinación (R^2), pero sí se obtuvo un error cuadrático medio (RMSE) de 0.20 [mm]. Esto se acerca al rendimiento del modelo básico inicial. No obstante, si se empleara el modelo básico en lugar del optimizado con *Optuna*, el RMSE sería superior (lo cual seria un peor resultado). A continuación, se presenta un resumen de los resultados con los RMSE por pieza en la siguiente tabla:

Tabla 2: Resultados RMSE modelo
Fuente: Elaboración Propia.

	Lifter alimentación	Placa alimentación	Lifter descarga	Placa descarga
RMSE [mm]	0.20	0.06	0.29	0.12

Los resultados antes mencionados son los primeros en su tipo para la empresa Tega ya que uno de los motivos para realizar este trabajo es hacer uso de los datos capturados por los sensores. No hay intentos previos para realizar una comparación entre rendimientos. Lo más cercano a lo que se puede comparar este trabajo es un intento de predicción realizado por el área de innovación de Tega el cual obtuvo un R^2 de 0.6, lo cual indica una mejora de 20 puntos.

4.2. Fase 2: Algoritmo de búsqueda de alturas del perfil simple

El funcionamiento del algoritmo ya fue explicado en detalle así que no se profundizará en ello nuevamente, por lo tanto, ahora lo importante son los resultados. Para este experimen-

to se utilizó uno de los ciclos operativos que el algoritmo no vio en el entrenamiento y se compararon los resultados obtenidos por la predicción del desgaste con la interpolación de las campañas de revisión.

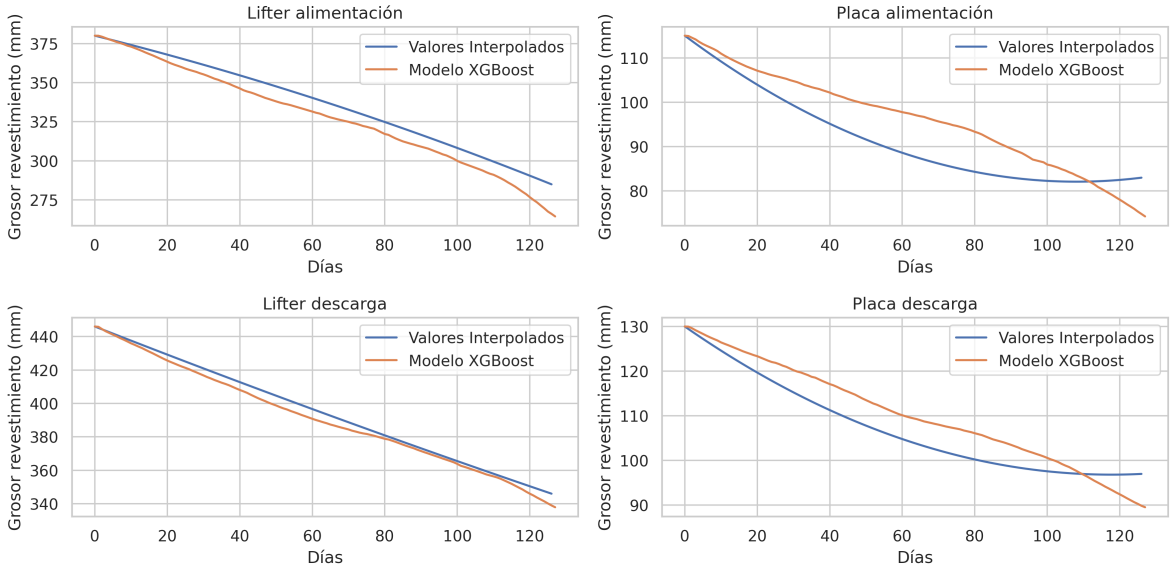


Figura 40: Proyección de los desgastes
Fuente: Elaboración propia.

Se ve que el comportamiento de descenso del grosor del revestimiento es similar. La mayor diferencia se ve en la zona de la placa lo cual puede deberse a diferencias en los materiales de construcción del revestimiento en esa pieza o un diseño diferente como ancho del revestimiento, ángulo de ataque, entre otros factores que no se exploraron en este escrito.

Continuando, la siguiente tabla expone pruebas que se hicieron al algoritmo de búsqueda con duraciones objetivos diferentes. Además, la figura 41 expone en consola como realiza la búsqueda el algoritmo propuesto.

Al analizar en profundidad el comportamiento de las propuestas, se observa que el algoritmo tiende a ser conservador, es decir, deja un margen de días sobre el límite. Esto tiene sentido, ya que en la realidad se aprecia una caída brusca en el grosor del revestimiento durante las fases iniciales de operación de las campañas, la cual tiende a estabilizarse a medida que se acerca al límite. Por lo tanto, es razonable concluir que se debe considerar una altura inicial superior para el perfil, con el fin de evitar sobrepasar los límites de desgaste definidos por la empresa.

Tabla 3: Propuestas de alturas del perfil simple de las piezas del revestimiento para diferentes duraciones objetivo

Fuente: Elaboración Propia.

Duración objetivo [Días]	Lifter alimentación [mm]	Placa alimentación [mm]	Lifter descarga [mm]	Placa descarga [mm]
200	340.03	101.67	412.03	116.67
180	302.44	89.14	374.44	104.14
150	262.48	75.83	334.48	90.82
120	249.30	71.43	321.30	86.43
90	238.24	67.74	310.24	82.74
60	185.99	55.91	241.49	67.47

```
######################
Alturas perfil iniciales:  [339.71989242 101.57348192 411.71989242 116.57349844]
Shape del historico de la maquina:  (214, 13)
Limite de desgaste por pieza:  [90, 42, 90, 42]
Dias objetivo de duracion:  200
######################
Nueva altura: [339.59986449 101.5410858  411.72567378 116.55177241]
Nueva altura: [339.40201599 101.35555244 411.52460614 116.43327531]
Nueva altura: [338.54654808 100.7370503  410.27436998 116.18805316]
Nueva altura: [337.58569641 100.22509217 408.75561407 115.7848244 ]
Nueva altura: [336.93794985  99.73922008 407.31205836 115.50697243]
Nueva altura: [335.98747582  99.11655957 405.93950334 115.15769359]
Nueva altura: [335.18225826  98.5712865  404.52435246 114.92318055]
Nueva altura: [334.35269345  98.04718649 403.14779034 114.57651853]
Nueva altura: [333.59895612  97.56979841 401.78608921 114.26176342]
Nueva altura: [332.87011851  97.0767273  400.46906903 113.9588134 ]
Nueva altura: [332.15735586  96.56190327 399.06912437 113.65978819]
Nueva altura: [331.53766103  95.97195616 397.51975718 113.35058823]
Nueva altura: [330.73757161  95.51008991 395.95294121 113.01823687]
Nueva altura: [330.11620868  95.06149143 394.45976606 112.7296327 ]
Nueva altura: [329.21194995  94.63973466 392.78594902 112.3728036 ]
Nueva altura: [328.46285076  94.21456394 391.0353103  112.00865909]
Nueva altura: [327.70457829  93.77125239 389.44331172 111.69123497]
Nueva altura: [326.90245671  93.3765623  387.81550553 111.34689036]
Nueva altura: [325.95191164  92.85554904 386.25590697 110.98275789]
Nueva altura: [325.25995351  92.32147098 384.87873486 110.66536575]
Nueva altura: [324.49500103  91.77557361 383.54818156 110.34581685]
Nueva altura: [323.9084918   91.25089878 382.20469382 110.03353968]
Nueva altura: [323.30008252  90.76475269 380.89070263 109.72046608]
Nueva altura: [322.55082388  90.25074208 379.54316273 109.40536991]
```

Figura 41: Consola durante una de las iteraciones de búsqueda del algoritmo
Fuente: Elaboración propia.

4.3. Fase 3: Dibujo de la geometría propuesta

Finalmente, con las propuestas de alturas antes vistas, se construyeron figuras geométricas que representarían el revestimiento. Para esta sección se coordinó una reunión con la sección de diseño de la empresa y se facilitaron tres diseños sobre los cuales se basaron las proyecciones. Estos tres diseños se usaron como base y se aplicó la nueva altura propuesta, además de también modificar los ángulos de ataque.

A continuación se presentaran los resultados de diseños en base a parámetros operacionales: ángulo de ataque y duración objetivo. El grosor de los revestimientos se mantuvo constante entre los diseños propuestos ya que en este trabajo no se exploró el impacto de tiene la cantidad de *liners* dentro de los anillos.

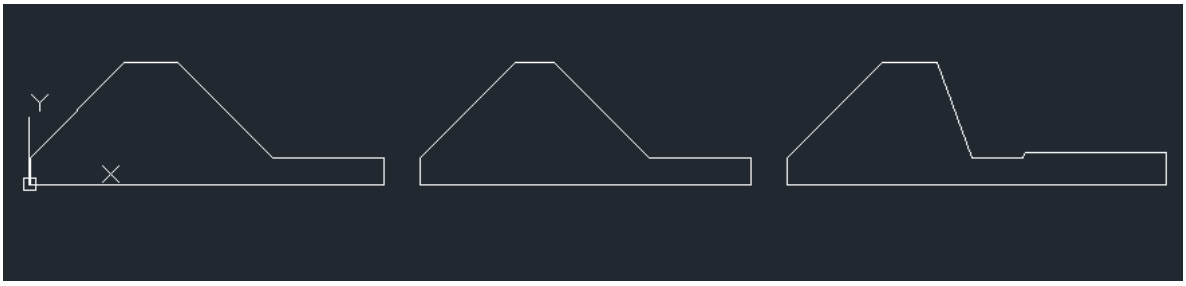


Figura 42: Diseño considerando 100 días de duración y un ángulo de ataque de 45°.
Fuente: *Elaboración propia*.

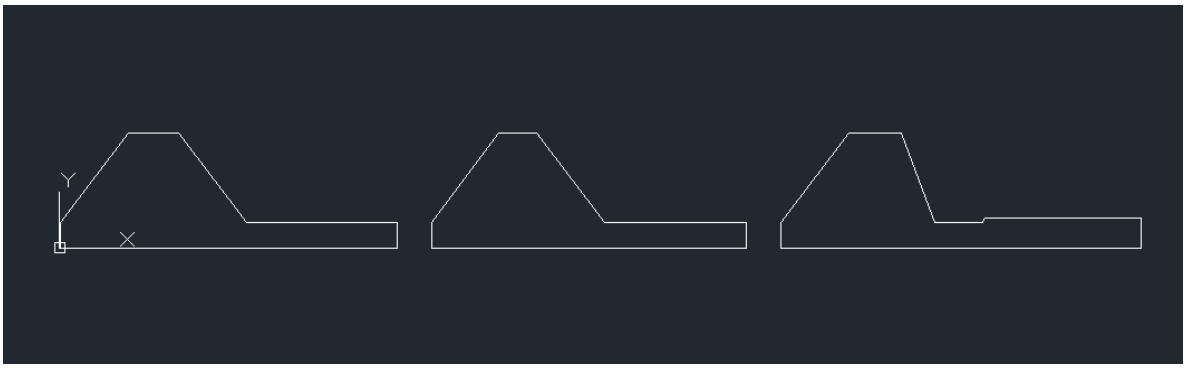


Figura 43: Diseño considerando 100 días de duración y un ángulo de ataque de 53°.
Fuente: *Elaboración propia*.

Figura 44: Diseño considerando 100 días de duración y un ángulo de ataque de 60°.
Fuente: *Elaboración propia*.

CAPÍTULO 5
CONCLUSIONES

Este trabajo consistió en una serie de etapas: comenzando con el análisis e imputación de la nube de datos, seguido de la extracción de características relevantes, continuando con la creación y entrenamiento del modelo de aprendizaje automático basado en *XGBoost*. Finalizando así con el algoritmo de predicción de alturas del perfil simple del revestimiento y la generación del dibujo geométrico en AutoCAD.

El autor de este trabajo comenzó sin nociones previas sobre el sector de la minería ni sus procesos. Sin embargo, a través de una ardua investigación y reuniones con expertos del sector, adquirió los conocimientos necesarios para comprender los mecanismos de funcionamiento de los molinos SAG, las piezas que los componen y los desafíos de ingeniería y logística que requiere la operación de estos grandes molinos en el sector de la gran minería.

Se presentaron una variedad de desafíos durante las etapas de imputación y modelado, además de inconvenientes a medio camino que se tuvieron que solventar con diferentes técnicas. Por supuesto, se requirió de una creatividad que permitiera explorar nuevas opciones y soluciones.

La propuesta de solución presentada abarco varios puntos importantes, entre los cuales destacan:

- Se realizó un análisis exploratorio de la nube de datos que posee la empresa Tega sobre la operación de los molinos SAG. Este análisis contribuye significativamente a la empresa, ya que dichos datos no tenían un uso relevante que pudiera serles de utilidad. Especialmente en el campo de la inteligencia artificial. Por ello, esta memoria representa la primera investigación de este tipo que explora posibles aplicaciones de esta data para la empresa.
- Los datos venían de sensores, los cuales presentaban muchas veces mediciones erróneas o fuera de la lógica operativa del equipo. Esto se solucionó realizando un análisis en profundidad de los datos y corrigiéndolos.
- La ciclabilidad de los datos representaba un problema a la hora de realizar las proyecciones ya que estaba sesgada a alturas iniciales fijas, no existiendo así una variabilidad sobre la cual sostenerse a la hora de inferir alturas nuevas. Esto se solucionó de una forma creativa, pasando así a considerar no el grosor del revestimiento en el tiempo (que es como se construyeron los datos originales), sino la diferencia entre los grosores diarios. Transformando así el problema.
- La empresa utilizaba reglas de tres para poder conocer la alturas de los perfiles del revestimiento tal que dure cierto periodo esperado. Esta forma de decidir el diseño

del revestimiento carece de un sustento operativo real y solo se basa en proyecciones lineales. La propuesta presenta una solución que toma en cuenta las relaciones entre las características operativas del molino y, a partir de ello, modela el desgaste diario en milímetros.

- Finalmente, se creó un algoritmo que permita generar el desgaste esperado sobre diferentes alturas iniciales y mostrar la geometría esperada.

Los resultados del modelo rondan valores sobre un R^2 de 0.80, lo cual para la construcción del primer prototipo es un buen resultado, mostrado así el potencial de mejora a futuro del trabajo. Con respecto a los nube de datos, no se tenía una gran cantidad de estos ya que el sensor recolecta diariamente la data y, durante un periodo de poco más de tres años, solo se utilizaron un total de 2058 datos operativos de los molinos. Esta cantidad de datos supuso un reto al momento de realizar las correcciones ya que no era conveniente eliminar filas considerando que se contaba con tan poca información. Por ello, se tuvo que recurrir a técnicas de imputación como la media o reemplazar los valores erróneos con valores que estén dentro del rango funcionamiento según las diferentes características operacionales del equipo.

Otro punto destacable sobre el tratamiento de los datos fue el escalado. Los modelos tienden a ser sensibles con los extremos y las diferencias entre las magnitudes de una característica a otra perjudicaba el entrenamiento.

Para finalizar, la sección del dibujo del revestimiento requirió el uso de software externo a Python. La razón principal de esta decisión se basó en recomendaciones del área de innovación de la empresa y falta de flexibilidad con las librerías tradicionales de python.

5.1. Alcance y limitaciones

La propuesta cumple con los objetivos planteados: es capaz de inferir nuevas alturas a partir de datos previos del funcionamiento del molino y construir una geometría simple del perfil. Además, la propuesta inicial del modelo tiene un buen comportamiento en los test sintéticos. Pese a esto, hay limitaciones que se deben discutir:

A. El modelo requiere de un historial previo del funcionamiento del molino sobre el cual se quiere proyectar la geometría del nuevo revestimiento. Por esto, el modelo no es capaz generalizar con molinos que posean revestimientos construidos con materiales diferentes al de goma-metal.

B. El modelo tiende a ser conservador, por lo que no exprime al máximo las limitaciones reales del revestimiento. Esto asegura que no se superen las limitaciones del desgaste, pero no es eficiente.

C. La falta de datos con alturas iniciales diferentes sigue siendo un problema a tener en cuenta, aunque se solventa en cierta medida con el modelo de XGBoost.

D. En caso de que exista nueva información operativa para entrenar nuevamente el modelo y robustecerlo, es necesario tener en cuenta que los datos deben ser corregidos e imputados antes de ser utilizados.

E. Debido a que las campañas son muy puntuales en el tiempo, el modelo es dependiente de dos factores:

   - Las aproximaciones polinómicas del grosor del revestimiento.
   - La característica artificial de tonelaje acumulado.

   Sobre esto último, si no se considera el tonelaje acumulado dentro de las características de entrenamiento del modelo, se observa una caída significativa de la métrica R^2.

F. Finalmente, la geometría propuesta es simple, más cercana a un prototipo. Pero hay que recalcar que este trabajo es una primera aproximación a una solución totalmente optima y esta abierta a potenciales mejoras a futuro.

5.2. Recomendaciones para trabajos futuros

Como se mencionó, el modelo esta abierto a potenciales mejoras y, además, se debe considerar amortiguar los problemas antes discutidos. Para esto último se tienen las siguientes propuestas:

- Generar mayor cantidad de información sobre el funcionamiento operativo del molino para suavizar la curva de desgaste. Lo cual permitiría que el modelo aprenda mejor los patrones. Para ello, es interesante considerar la posibilidad de tomar datos a cada hora en vez de una vez al día, incrementando así el volumen de datos.
- Para no depender de un histórico del funcionamiento de los datos, se conversó en una de las reuniones la posibilidad de generar datos sintéticos del funcionamiento promedio de los molinos. Esto busca tener una 'simulación' de los datos con los que el modelo pueda generar las predicciones de desgaste, lo cual aumentará la flexibilidad. Además, permitiría que dichas predicciones no estén fuertemente ligadas a molinos únicos. Por contrario, esta solución también podría provocar una menor precisión en las proyecciones.
- El modelo puede generar una retroalimentación positiva en el futuro mientras más se utilice. A medida que se prueben las recomendaciones de las alturas sugeridas por el modelo, se pueden usar los datos operativos del molino utilizando dicha recomendación para así volver a entrenar el modelo, mejorando la precisión.

- Ser cuidadoso de no mezclar datos operativos que describan el funcionamiento de molinos con revestimientos de diferentes materiales.
- Se utilizaron los puntos discontinuos de supervisión para generar la interpolación polinómica, pero seria interesante probar con otras funciones que quizás puedan representar mejor la curva de desgaste en los ciclos. Se realizaron algunas pruebas con curvas como la función exponencial inversa, decaimiento logístico y decaimiento gaussiano. Todas tienen el comportamiento de que decaen rápido y luego se estabilizan. Este comportamiento se ve remarcado en el mapa de correlaciones donde se observó que el tonelaje acumulado es inversamente proporcional del desgaste diario, especialmente en la zona de descarga.

Además del manejo de los datos y proceso de entrenamiento, está la posibilidad de utilizar otros modelos de ML que no se vieron en esta tesis, pero si se exploraron como posibilidad. Dentro de estos modelos están las redes neuronales recurrentes (RNN) tales como el modelo *Long Short Term Memory (LSTM)* o el modelo *Autoregressive integrated moving average (ARIMA)*, los cuales son basados en series temporales haciéndolos muy eficientes al momento de generar predicciones futuras en base al comportamiento pasado de los datos. Si el lector esta interesado en investigar más sobre estos modelos y sus usos, una excelente fuente de información es la explicación de Christopher Olah: *Understanding LSTM Networks* [Olah, 2015]. El uso de redes neuronales puede mejorar aun más la precisión de la predicción, pero limitaciones en los datos como la falta de variabilidad de alturas iniciales pueden presentar un desafió a resolver.

Otra forma de abordar este problema seria utilizando *Forecasting* con *XGBoost*, o lo que es lo mismo, adaptar este modelo para que sea capaz de analizar series temporales. Esto requeriría más trabajo, pero es una opción viable. Ivan Lee realizó un trabajo sobre la predicción de energía solar con esta técnica de modelado [Lee, 2023].

ANEXOS

Visualización dataset

```
[ ]  #Se cargan los datos desde le dataset
     n_molino = 4
     if n_molino == 3:
       file_name = './SAG_3_SIN_ESCALAR.xlsx'
     elif n_molino == 2:
       file_name = './SAG_2_SIN_ESCALAR.xlsx'
     else:
       file_name = './SAG_IMPUTADO.xlsx'

     df = pd.read_excel(file_name,header=0)
     df.head()

[ ]  df.shape

[ ]
     df_view = df.iloc[:,:-4].drop(columns=['Desgaste Lifter Alimentacion (RL)','Desgaste Placa Alimentacion (RL)',
                                            'Desgaste Lifter Descarga (RL)','Desgaste Placa Descarga (RL)']).copy()

[ ]  '''# Genera el reporte de perfilado
     report = ProfileReport(df_view, sort=None, html={'style':{'full_width':True}})

     # Muestra el reporte en el notebook
     report.to_notebook_iframe()'''

[ ]  report.to_file("reporte.html")
```

Técnicas de regularización

```python
def histogramas_caracteristicas(orden_metodos):
    """
    Genera histogramas comparativos de las puntuaciones de características según diferentes métodos.
    """
    lasso = orden_metodos[1]
    ridge = orden_metodos[0]

    x = np.arange(len(lasso[0]))

    plt.figure(figsize=(12, 6))
    sns.set(style="whitegrid")
    plt.bar(x - 0.15, lasso[1], color='blue', width=0.3, label='Lasso')
    plt.bar(x + 0.15, ridge[1], color='green', width=0.3, label='Ridge')

    plt.xticks(x, lasso[0], rotation=45, ha='right', fontsize=9)
    plt.xlabel('Características')
    plt.ylabel('Puntuación')
    plt.title('Score de características según método')
    plt.legend()
    plt.savefig("analisis_caracteristicas.png", dpi=300, bbox_inches='tight')
    plt.show()


def entrenar(X, y, method, columnas, train_type = 1):
    """
    Entrena un modelo con diferentes métodos de selección de características.
    """
    def k_caracteristicas(method,indices,X_train,X_test,y_train,y_test):
      errors = []
      model = MultiOutputRegressor(LinearRegression())
```

```
  for k in range(1, X.shape[1] + 1):
      selected_indices = indices[:k]
      X_train_selected = X_train.iloc[:, selected_indices]
      X_test_selected = X_test.iloc[:, selected_indices]
      if method == 'lasso':
        model = MultiOutputRegressor(LassoCV())
        model.fit(X_train_selected,y_train)
      elif method == 'ridge':
        model = MultiOutputRegressor(Ridge(alpha=1.0))
        model.fit(X_train_selected,y_train)
      else:
        model = MultiOutputRegressor(LinearRegression())
        model.fit(X_train_selected,y_train)
      y_pred = model.predict(X_test_selected)
      y_pred = pd.DataFrame(y_pred)
      mse_0 = mean_squared_error(y_test.iloc[:,0], y_pred.iloc[:,0])
      mse_1 = mean_squared_error(y_test.iloc[:,1], y_pred.iloc[:,1])
      mse_2 = mean_squared_error(y_test.iloc[:,2], y_pred.iloc[:,2])
      mse_3 = mean_squared_error(y_test.iloc[:,3], y_pred.iloc[:,3])
      mse = np.array([mse_0,mse_1,mse_2,mse_3])
      rmse = np.sqrt(mse)
      errors.append(rmse)
  return errors

if train_type == 1:
  #Entrenamiento considerando ciclos
  split_ratio = 0.8
  split_index = int(len(X) * split_ratio)

  X_train, X_test = X[:split_index], X[split_index:]
  y_train, y_test = y[:split_index], y[split_index:]
```

```
else:
  #Entrenamiento con dispersion de puntos
  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# los metodos soportan múltiples salidas
orden = []
errors = []
if method == "lasso":
    lasso = MultiOutputRegressor(LassoCV())
    lasso.fit(X_train, y_train)
    coeficientes = np.mean([est.coef_ for est in lasso.estimators_], axis=0)
    indices = np.argsort(np.abs(coeficientes))[::-1]
    orden = ([columnas[i] for i in indices], [coeficientes[i] for i in indices])
    print(f"Lasso orden: {orden[1]}")
    print(f"Orden de características Lasso: {orden[0]}")
    errors = k_caracteristicas(method,indices,X_train,X_test,y_train,y_test)
elif method == "ridge":
    ridge = MultiOutputRegressor(Ridge(alpha=1.0))
    ridge.fit(X_train, y_train)
    coeficientes = np.mean([est.coef_ for est in ridge.estimators_], axis=0)
    indices = np.argsort(np.abs(coeficientes))[::-1]
    orden = ([columnas[i] for i in indices], [coeficientes[i] for i in indices])
    print(f"Ridge orden: {orden[1]}")
    print(f"Orden de características Ridge: {orden[0]}")
    errors = k_caracteristicas(method,indices,X_train,X_test,y_train,y_test)
else:
    indices = np.random.permutation(X.shape[1])
    random = MultiOutputRegressor(LinearRegression())
    orden = ([columnas[i] for i in indices], [i for i in indices])
    errors = k_caracteristicas(method,indices,X_train,X_test,y_train,y_test)

return errors, orden
```

```python
#Primero separemos entre datos de entrada y salida
from sklearn.linear_model import Ridge, LassoCV
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import TimeSeriesSplit
from sklearn.multioutput import MultiOutputRegressor

y_columns = [
    'Desgaste Lifter Alimentacion (POLY)',
    'Desgaste Placa Alimentacion (POLY)',
    'Desgaste Lifter Descarga (POLY)',
    'Desgaste Placa Descarga (POLY)'
    ]

y_columns_rl = [
    'Desgaste Lifter Alimentacion (EXP_d)',
    'Desgaste Placa Alimentacion (EXP_d)',
    'Desgaste Lifter Descarga (EXP_d)',
    'Desgaste Placa Descarga (EXP_d)',
    'Desgaste Lifter Alimentacion (POLY)',
    'Desgaste Placa Alimentacion (POLY)',
    'Desgaste Lifter Descarga (POLY)',
    'Desgaste Placa Descarga (POLY)',
    'Desgaste Lifter Alimentacion (RL)',
    'Desgaste Placa Alimentacion (RL)',
    'Desgaste Lifter Descarga (RL)',
    'Desgaste Placa Descarga (RL)'
    ]
```

```python
[ ] #Se escala para hacer la regularizacion
    scaler = MinMaxScaler()
    X_no_escalado = df.drop(columns=y_columns_rl)

    y = df[y_columns]
    X = pd.DataFrame(scaler.fit_transform(X_no_escalado), columns=df.drop(columns=y_columns_rl).columns, index=df.index)
```

```python
def graficos_rmse(df_lasso, df_ridge, df_random):
  fig, ax = plt.subplots(2, 2, figsize=(12, 6))
  for i in range(4):
      row = i // 2
      col = i % 2
      ax[row, col].plot(df_ridge[:, i], label='Ridge')
      ax[row, col].plot(df_lasso[:, i], label='Lasso')
      ax[row, col].plot(df_random[:, i], label='Random')
      if i == 0:
          ax[row, col].set_title('Lifter alimentación')
      elif i == 1:
          ax[row, col].set_title('Placa alimentación')
      elif i == 2:
          ax[row, col].set_title('Lifter descarga')
      else:
          ax[row, col].set_title('Placa descarga')

      ax[row, col].set_xlabel('Número de características (K)')
      ax[row, col].set_ylabel('Raiz del error cuadrado medio [mm]', fontsize=10)
      ax[row, col].legend()

  plt.tight_layout()
```

```
# Selección de características usando múltiples métodos
results = {}
orden_metodos = []

for method in ["lasso", "ridge", "random"]:
    results[method], orden = entrenar(X, y, method, np.array(X.columns))
    orden_metodos.append(orden)

# Generar histogramas
histogramas_caracteristicas(orden_metodos)



graficos_rmse(np.array(results['lasso']),np.array(results['ridge']),np.array(results['random']))
```

Manipulación de datos: diferencias entre espesores

```
#Calcula las diferencias entre los espesores por dia
def diff_(df_diff):
  columns = df_diff.columns[2:]
  # Aplicar la diferencia con condición adicional
  for col in columns:
      df_diff[col] = np.where(
          (df_diff['Estado'] == 0) | (df_diff['Tonelaje Acumulado'] == 0),  # Condición: Si el estado es 0
          0,  # El resultado es 0
          np.where(
              (df_diff[col].shift(1) == 0) & (df_diff[col] != 0),  # Condición: fila anterior es 0 y fila actual no
              0,  # Si se cumple, el resultado es 0
              df_diff[col].diff().fillna(0).abs()  # Si no, calcular la diferencia como antes
          )
      )
  return df_diff.iloc[:,2:]
```

```
df_xgboost = df.iloc[:,:-12].copy()

df_diff = df[['Tonelaje Acumulado','Estado','Desgaste Lifter Alimentacion (POLY)','Desgaste Placa Alimentacion (POLY
                      'Desgaste Lifter Descarga (POLY)','Desgaste Placa Descarga (POLY)']].copy()

#df_xgboost = df_xgboost.drop(columns = ['Desgaste Lifter Alimentacion (POLY)','Desgaste Placa Alimentacion (POLY)',
#                        'Desgaste Lifter Descarga (POLY)', 'Desgaste Placa Descarga (POLY)'])
```

```
df_diff = diff_(df_diff)
```

```python
#Grafico del desgaste diario del dataset
wear = df_diff.iloc[:,0]
wear2 = df_diff.iloc[:,1]
wear3 = df_diff.iloc[:,2]
wear4 = df_diff.iloc[:,3]
days = range(1, len(wear) + 1)

plt.figure(figsize=(12, 6))
plt.plot(days, wear)
plt.plot(days, wear2)
plt.plot(days, wear3)
plt.plot(days, wear4)
plt.title('Desgaste por Día', fontsize=16)
plt.xlabel('Día')
plt.ylabel('Desgaste', fontsize=10)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

```python
# Error potcentual por columna
def error_porcentual(Y_pred,Y_test):
  Y_pred_df = pd.DataFrame(Y_pred, columns=Y_test.columns)
  Y_test_df = Y_test.reset_index(drop=True)

  # RMSE por cada salida
  rmse_per_output = {}
  for col in Y_test.columns:
      mse = mean_squared_error(Y_test_df[col], Y_pred_df[col])
      rmse = np.sqrt(mse)
      rango = Y_test_df[col].max() - Y_test_df[col].min()  # Rango de la salida
      porcentaje_error = (rmse / rango) * 100
      rmse_per_output[col] = {
          'RMSE': rmse,
          'Porcentaje de error': porcentaje_error
      }

  for salida, valores in rmse_per_output.items():
      print(f"Salida: {salida}")
      print(f"  RMSE: {valores['RMSE']:.4f}")
      print(f"  Porcentaje de error: {valores['Porcentaje de error']:.2f}%\n")
```

```python
#Se crean una columna solo con la medicion del desgate diario
df_xgboost['Desgaste Lifter Alimentacion (EXP_d_desgaste)'] = df_diff.iloc[:,0]
df_xgboost['Desgaste Placa Alimentacion (EXP_d_desgaste)'] = df_diff.iloc[:,1]
df_xgboost['Desgaste Lifter Descarga (EXP_d_desgaste)'] = df_diff.iloc[:,2]
df_xgboost['Desgaste Placa Descarga (EXP_d_desgaste)'] = df_diff.iloc[:,3]
```

```python
df_xgboost.to_excel('SAG_diferencias.xlsx', index=False)
```

```python
#Observando el dataset otra vez

df_view = df_xgboost.copy()

# Genera el reporte de perfilado
report = ProfileReport(df_view, sort=None, html={'style':{'full_width':True}})

# Muestra el reporte en el notebook
report.to_notebook_iframe()
```

Entrenamiento XGBoost

```
df_xgboost_ = df_xgboost[['Nivel del molino o Jc (%)', 'Sólidos (%)', 'Tamaño o granulometría de descarga (")',
                          'Tonelaje Diario', 'Nivel de bolas o Jb (%)', 'Tamaño o granulometría de alimentación (")',
                          'Estado', 'Generación Pebbles (TPH)',
                          'Tonelaje Acumulado', 'Sentido de giro_Horario', 'Sentido de giro_Anti-Horario',
                          'Velocidad de giro (rpm)','pH',
                          'Desgaste Lifter Alimentacion (EXP_d_desgaste)','Desgaste Placa Alimentacion (EXP_d_desgaste)',
                         'Desgaste Lifter Descarga (EXP_d_desgaste)', 'Desgaste Placa Descarga (EXP_d_desgaste)']].copy()

df_xgboost_ = df_xgboost[['Velocidad de giro (rpm)', 'Tonelaje Acumulado', 'Estado', 'Sólidos (%)',
                         'Presión sobre los descansos (psi)', 'Tonelaje Diario', 'Nivel de bolas o Jb (%)',
                         'Nivel del molino o Jc (%)', 'Tamaño o granulometría de alimentación (")',
                         'Potencia consumida e instalada (kW/h)', 'Generación Pebbles (TPH)', 'pH',
                         'Desgaste Lifter Alimentacion (EXP_d_desgaste)','Desgaste Placa Alimentacion (EXP_d_desgaste)',
                         'Desgaste Lifter Descarga (EXP_d_desgaste)', 'Desgaste Placa Descarga (EXP_d_desgaste)']]

df_xgboost_copy = pd.concat([df_xgboost_.iloc[:1763,:], df_xgboost_.iloc[1928:,:]], ignore_index=True)
```

```
#Entrenamiento y validacion
df_e_v = df_xgboost_copy.iloc[:-3,:]
```

```
#Modelo Basico
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error,r2_score


Y = df_e_v.iloc[:,-4:]
X = df_e_v.iloc[:,:-4]

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train)

# Transformar el conjunto de prueba usando el mismo escalador
X_test_scaled = scaler.transform(X_test)

dtrain_reg = xgb.DMatrix(X_train_scaled, Y_train)
dtest_reg = xgb.DMatrix(X_test_scaled, Y_test)

params = {
    "objective": "reg:squarederror",

}
```

```
model_basico= xgb.train(
    params=params,
    dtrain=dtrain_reg,
    num_boost_round=100,
)

Y_pred = model_basico.predict(dtest_reg)
r2_test = r2_score(Y_test, Y_pred)
error = mean_squared_error(Y_test, Y_pred)
print(f"R^2 en conjunto de prueba: {r2_test:.3f}")
print(f"Raiz del error cuadrático medio [mm]: {np.sqrt(error):.3f}")
```

```
#Vemos el error porcentual
error = error_porcentual(Y_pred,Y_test)
error
```

```
#r2_score
r2 = r2_score(Y_test, Y_pred)
print(r2)
```

```
evals = [(dtrain_reg, "train"), (dtest_reg, "validation")]
params = {
    "objective": "reg:squarederror",
    "learning_rate": 0.03,
    "max_depth": 12,
    "subsample": 0.9,
    "colsample_bytree": 0.9,
}
n_b = 1500
model_medio = xgb.train(
   params=params,
   dtrain=dtrain_reg,
   num_boost_round = n_b,
   evals=evals,
   verbose_eval=50, #permite cada N runs como se comporta el modelo
   early_stopping_rounds=50, #Si el modelo no mejora despues de 50 round, el modelo para
)

Y_pred_2 = model_medio.predict(dtest_reg)
error_2 = mean_squared_error(Y_test, Y_pred_2)

#Se observa una mejora
```

```
#Calculando el error promedio
error = error_porcentual(Y_pred_2,Y_test)
error
```

```python
#Modelo de complejidad alta: optimizacion de hiperparametros con optuna
import optuna
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from optuna.pruners import MedianPruner


def objective(trial):
    #  hiperparámetros
    params = {
        "objective": "reg:squarederror",
        "learning_rate": trial.suggest_float("learning_rate", 0.01, 0.05, log=True),
        "max_depth": trial.suggest_int("max_depth", 6, 12),
        "subsample": trial.suggest_float("subsample", 0.5, 0.9),
        "colsample_bytree": trial.suggest_float("colsample_bytree", 0.5, 0.9),
        "gamma": trial.suggest_float("gamma", 0, 3),
        "reg_alpha": trial.suggest_float("reg_alpha", 1e-8, 1.0, log=True),
        "reg_lambda": trial.suggest_float("reg_lambda", 1e-8, 1.0, log=True),
        "n_estimators": trial.suggest_int("n_estimators", 50, 1000),
        "max_bin": trial.suggest_int("max_bin", 128, 512),
    }

    rounds = trial.suggest_int("num_boost_round", 100, 1000)

    model = xgb.train(params=params, dtrain=dtrain_reg, num_boost_round=rounds)
    Y_pred = model.predict(dtest_reg)

    mse = mean_squared_error(Y_test, Y_pred)
    rmse = np.sqrt(mse)
    return rmse
```

```python
best_params = {'learning_rate': 0.014607667101114686, 'max_depth': 6, 'subsample': 0.6763193085524807, 'colsample_
```

```python
#Se entrena el modelo con los mejores parametros
#best_params = study.best_params
final_model = xgb.train(params=best_params, dtrain=dtrain_reg, num_boost_round=best_params["num_boost_round"])
```

```python
#Se evalua
Y_pred_final = final_model.predict(dtest_reg)
error = error_porcentual(Y_pred_final,Y_test)
error
```

```python
#Valor general para comprarlo con el cross-validation
error_final = mean_squared_error(Y_test, Y_pred_final)
error_final = np.sqrt(error_final)
print(f"Raiz del error cuadrático medio [mm]: {error_final}")
```

```python
#r2_score
r2 = r2_score(Y_test, Y_pred_final)
print(r2)
#Se observa mejora
```

K-validation

```
resultados = xgb.cv(
  params=best_params,
  dtrain=dtrain_reg,
  num_boost_round=best_params["num_boost_round"],
  nfold=5, #k-cross-validation
)
```

```
resultados.head()
```

```
best_error = resultados['test-rmse-mean'].min()
best_error

#Aunque el resultado es ligeramente peor, es mas representativo de la realidad
```

```
#Comparando con el resultado del modelo final (porcentual)
print((best_error*100)/error_final)
```

Gráficos de desgaste

```
#Para los graficos con el test de prueba definicivo
def graficos(tipo, df_pred, df_test = []):
  fig, ax = plt.subplots(2, 2, figsize=(15, 6))
  for i in range(4):
      row = i // 2
      col = i % 2
      if tipo == 1:
        ax[row, col].plot(df_test.iloc[:, i], label='Valores Interpolados')
        ax[row, col].plot(df_pred.iloc[:, i], label='Modelo XGBoost')
      else:
        ax[row, col].plot(df_pred.iloc[:, i], label='Modelo XGBoost')

      if i == 0:
          ax[row, col].set_title('Lifter alimentación')
      elif i == 1:
          ax[row, col].set_title('Placa alimentación')
      elif i == 2:
          ax[row, col].set_title('Lifter descarga')
      else:
          ax[row, col].set_title('Placa descarga')

      ax[row, col].set_xlabel('Días')
      ax[row, col].set_ylabel('Desgaste (mm)')
      ax[row, col].legend()

  plt.tight_layout()

  plt.savefig("grafico.png", dpi=300, bbox_inches="tight")
  plt.show()
```

```
'''Y_test = df_t.iloc[:,-4:]
X_test = df_t.drop(columns=['Desgaste Lifter Alimentacion (EXP_d_desgaste)','Desgaste Placa Alimentacion (EXP_d_desgaste
                            'Desgaste Lifter Descarga (EXP_d_desgaste)','Desgaste Placa Descarga (EXP_d_desgaste)'])
'''

dtest_reg_test = xgb.DMatrix(X_test_scaled)
```

```
#Modelo basico
Y_pred = model_basico.predict(dtest_reg_test)
Y_pred_df = pd.DataFrame(Y_pred).reset_index().iloc[:,1:]
Y_test_df = pd.DataFrame(Y_test).reset_index().iloc[:,1:]

graficos(1, Y_pred_df, Y_test_df)
```

```
#Modelo con tunning simple
Y_pred = model_medio.predict(dtest_reg_test)
Y_pred_df = pd.DataFrame(Y_pred).reset_index().iloc[:,1:]
Y_test_df = pd.DataFrame(Y_test).reset_index().iloc[:,1:]

graficos(1, Y_pred_df, Y_test_df)
```

```
#Modelo optimizado con optuna
Y_pred = final_model.predict(dtest_reg_test)
Y_pred_df = pd.DataFrame(Y_pred).reset_index().iloc[:,1:]
Y_test_df = pd.DataFrame(Y_test).reset_index().iloc[:,1:]

graficos(1, Y_pred_df, Y_test_df)
```

```
def graficos_comparacion(tipo, df_pred, df_test = []):
  fig, ax = plt.subplots(2, 2, figsize=(12, 6))
  for i in range(4):
      row = i // 2
      col = i % 2
      if tipo == 1:
        ax[row, col].plot(df_test.iloc[:, i], label='Valores Interpolados')
        ax[row, col].plot(df_pred.iloc[:, i], label='Modelo XGBoost')
      else:
        ax[row, col].plot(df_pred.iloc[:, i], label='Modelo XGBoost')

      if i == 0:
          ax[row, col].set_title('Lifter alimentación')
      elif i == 1:
          ax[row, col].set_title('Placa alimentación')
      elif i == 2:
          ax[row, col].set_title('Lifter descarga')
      else:
          ax[row, col].set_title('Placa descarga')

      ax[row, col].set_xlabel('Días')
      ax[row, col].set_ylabel('Grosor revestimiento (mm)')
      ax[row, col].legend()

  plt.tight_layout()

  plt.savefig("grafico_comparacion.png", dpi=300, bbox_inches="tight")
  plt.show()
```

Algoritmo y función objetivo

```python
# Función objetivo
def funcion_objetivo(alturas, scaler, modelo, historico_maquina, limites_desgaste, dias_objetivo):
    """
    Función objetivo para encontrar las alturas iniciales que minimicen la desviacion
    del límite de desgaste permitido en un número de días objetivo.

    Parámetros:
    - Alturas: Alturas iniciales del perfil simple de los revestimientos
    - modelo: Modelo de predicción.
    - historico_maquina: características operacionales para la simulación.
    - limites_desgaste: Lista de límites de desgaste permitidos para cada pieza.
    - dias_objetivo que se desea que tenga el revestimiento.

    Retorna:
    - La penalizacion de la altura simulada, representada por la diferencia entre esta y el limite de desgaste
    """
    print('######################')
    print("Alturas perfil iniciales: ",alturas)
    print('Shape del historico de la maquina: ',historico_maquina.shape)
    print('Limite de desgaste por pieza: ',limites_desgaste)
    print('Dias objetivo de duracion: ',dias_objetivo)
    print('######################')

    alturas_iniciales = alturas.copy()
    penalizacion = 10e4

    if len(historico_maquina) < dias_objetivo:
      print('No hay suficientes datos historicos de la maquina para realizar la prediccion')
      return penalizacion
```

```python
X_simulado = scaler.fit_transform(historico_maquina)

dtest_reg = xgb.DMatrix(X_simulado)
desgastes_pred = modelo.predict(dtest_reg)

i = 0
for desgaste in desgastes_pred:
  alturas = alturas - desgaste
  print(f'Nueva altura: {alturas}')
  if i == dias_objetivo:
    #penaliza menos por quedarse corto
    print('Se llego a los dias objetivo')
    penalizacion = 3*(alturas_iniciales[0]-limites_desgaste[0])+(alturas_iniciales[1]-limites_desgaste[1])
    +3*(alturas_iniciales[2]-limites_desgaste[2])+(alturas_iniciales[3]-limites_desgaste[3])
    print(f"Penalizacion: {penalizacion}")
    return penalizacion
  elif any(alturas < limites_desgaste):
    #penaliza mas por pasarse del limite
    print("Se paso del limite de desgaste antes de completar los dias objetivo")
    penalizacion = 100*np.power(3*(alturas_iniciales[0]-limites_desgaste[0])+(alturas_iniciales[1]-limites_desgaste[1])
    +3*(alturas_iniciales[2]-limites_desgaste[2])+(alturas_iniciales[3]-limites_desgaste[3]),2)
    print(f"Penalizacion: {penalizacion}")
    return penalizacion
  else:
    i+=1
```

```
#Probemos con un ciclo real
# Lo ideal es tener dats limpios (fuera del entrenamiento)

historico_maquina = df_xgboost_copy.iloc[441:655,:-4]
historico_maquina.head()
```

```
#Parametros
alturas_iniciales = [380, 115, 452, 130] #[lifter alimentacion, placa alimentacion, lifter descarga, placa descarga]
limites_desgaste = [90, 42, 90, 42]
dias_objetivo = 200

try:
    resultado = minimize(
        funcion_objetivo,
        x0=alturas_iniciales,
        args=(scaler, final_model, historico_maquina, limites_desgaste, dias_objetivo),
        bounds=[(91, 380), (43, 115), (91, 452), (43, 130)],
    )
except Exception as e:
    print(f"Error: {e}")
```

Geometría del perfil

```
# Graficar los perfiles de desgaste
import ezdxf

#Parametros
shift = [0,991+100,991+926+200]
angulo = 60
angulo_radianes = np.radians(angulo)

#Espacio de dibujo
doc = ezdxf.new()
msp = doc.modelspace()

l1 = [149,109,156]
l2 = [483,387]
l3 = [991,926,1061]

y1,y2,_,_ = alturas_optimas

for i in range(3):

  if i < 2:
    #figura 1 y 2
    x1 = (y1-y2)/np.tan(angulo_radianes)
    x2 = x1 + l1[i]
    x3 = x2 + x1

    puntos_fig = [(shift[i],0),(shift[i],y2),(x1+shift[i],y1),(x2+shift[i],y1),(x3+shift[i],y2),(l3[i]+shift[i],y2),
     (l3[i]+shift[i],0)]
    msp.add_lwpolyline(puntos_fig, close=True)
```

```
  else:
    #figura 3
    angulo_2 = 70
    angulo_radianes_2 = np.radians(angulo_2)
    l4 = 142
    x1 = (y1-y2)/np.tan(angulo_radianes)
    x2 = x1 + l1[i]
    x3 = x2 + (y1-y2)/np.tan(angulo_radianes_2)
    x4 = x3 + l4
    x5 = x4 + (15/np.tan(angulo_radianes_2))

    puntos_fig = [(shift[i],0),(shift[i],y2),(x1+shift[i],y1),(x2+shift[i],y1),(x3+shift[i],y2),(x4+shift[i],y2),
     (x5+shift[i],y2+15),(l3[i]+shift[i],y2+15),(l3[i]+shift[i],0)]
    msp.add_lwpolyline(puntos_fig, close=True)

# archivo DXF
output_file = "figura.dxf"
doc.saveas(output_file)
print(f"Archivo DXF guardado como {output_file}")
```

REFERENCIAS BIBLIOGRÁFICAS

[Aguilar Titi, 2017] Aguilar Titi, E. E. (2017). Optimizacion de molinos semi-autogenos.

[Amazon, 2024] Amazon (2024). ¿qué es una red neuronal?

[Arroyo Murrugarra, 2018] Arroyo Murrugarra, M. J. (2018). Optimización de la eficiencia energética en un molino semi-autógeno mediante el diseño de revestimientos utilizando simulaciones de elementos discretos.

[Codelco, 2018] Codelco (2018). ¡conoce y aprende del proceso productivo del cobre! *Codelco Educa*.

[Codelco, 2019] Codelco (2019). *Molienda: Todo a la juguera*. Codelco. Breve descripcion del proceso de molienda y maquinaria utilizada por Codelco Chile.

[González-Muñiz, 2023] González-Muñiz, A. (2023). *Aplicación de técnicas de aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia en sistemas de ingeniería*. Tesis doctoral.

[Henríquez, 2006] Henríquez, M. A. A. (2006). Modelo matemático para la estimación de la vida útil de revestimientos en molinos semiautógenos. *Universidad Nacional de Ingeniería, Chile*.

[IBM, 2024a] IBM (2024a). ¿qué es el bosque aleatorio?

[IBM, 2024b] IBM (2024b). ¿qué es un árbol de decisión?

[Industries, 2024a] Industries, T. (2024a). Rompiendo paradigmas: Revestimientos hÍbridos para molinosde bolas, gigantes mundiales. Último acceso el 25 de abril 2024.

[Industries, 2024b] Industries, T. (2024b). Tecnología en revestimiento. Último acceso el 25 de abril 2024.

[Lagos, 2018] Lagos, G. (2018). Sistema de accionamiento de molinos sag. Último acceso el 25 de abril 2024.

[Lee, 2023] Lee, I. (2023). Series de tiempo: Forecasting con xgboost. Último acceso el 25 de enero 2025.

[Masui, 2022] Masui, T. (2022). All you need to know about gradient boosting algorithm.

[Olah, 2015] Olah, C. (2015). Understanding lstm networks.

[Rezaeizadeh *et al.*, 2010a] Rezaeizadeh, M., Fooladi, M., Powell, M., y Mansouri, S. (2010a). Experimental observations of lifter parameters and mill operation on power draw and liner impact loading. *Minerals Engineering*, 23(15):1182–1191.

[Rezaeizadeh *et al.*, 2010b] Rezaeizadeh, M., Fooladi, M., Powell, M., Mansouri, S., y Weerasekara, N. (2010b). A new predictive model of lifter bar wear in mills. *Minerals Engineering*, 23(15):1174–1181.

[Rouhiainen, 2018] Rouhiainen, L. (2018). Inteligencia artificial. *Madrid: Alienta Editorial*, pp. 20–21.

[scikit learn, 2024] scikit learn (2024). Cross-validation: evaluating estimator performance.

[Sherman y Rajamani, 1999] Sherman, M. y Rajamani, R. (1999). The effect of lifter design on alumbrera's sag mill performance: design expectations and optimization. En *31st annual CMP meeting, Ottawa, Ontario*, pp. 255–266.

[Siqueira, 2023] Siqueira, D. (2023). Mejora del análisis con boxplot.

[Team, 2024] Team, D. (2024). Tutorial de lasso y regresión ridge en python.

[Tomaschien, 2019] Tomaschien (2019). Material de revestimiento de molino sag. Último acceso el 25 de abril 2024.

[Vadillo, 2023] Vadillo, F. (2023). Una introducción a la interpolación polinomial.

[Valderrama y Magne, 1996] Valderrama, M. y Magne, L. (1996). Efecto del diseño de revestimientos sobre el consumo de potencia en molienda. *Revista de metalurgia*, 32(4):215–222.

[Walker, 2010] Walker, S. (2010). Revestimientos para los molinos. Último acceso el 22 de abril 2024.

[Yañez.C., 2018] Yañez.C., Fissore.A., L. . C. (2018). Resumen ejecutivo de usos de la energia de los hogares chile. Último acceso el 25 de abril 2024.

[Zeballos, 2003] Zeballos, J. (2003). *Compendio de conminución*. CONCYTEC.